
troposphere Documentation

Release 2.4.1

cloudtools

Feb 14, 2019

Table of Contents

1	About	3
2	Installation	5
3	Examples	7
3.1	Examples of the error checking (full tracebacks removed for clarity):	8
4	Currently supported AWS resource types	9
5	Currently supported OpenStack resource types	13
6	Duplicating a single instance sample would look like this	15
7	Community	17
8	Licensing	19
8.1	Quick Start	19
8.2	examples	22
8.3	troposphere	25
8.4	tests	132
8.5	Changelog	146
8.6	Contributing	175
8.7	Releasing	176
8.8	Code of Conduct	177
8.9	License	178
9	Indices and tables	179
	Python Module Index	181

CHAPTER 1

About

troposphere - library to create [AWS CloudFormation](#) descriptions

The troposphere library allows for easier creation of the [AWS CloudFormation JSON](#) by writing Python code to describe the AWS resources. troposphere also includes some basic support for [OpenStack resources](#) via Heat.

To facilitate catching CloudFormation or JSON errors early the library has property and type checking built into the classes.

CHAPTER 2

Installation

troposphere can be installed using the pip distribution system for Python by issuing:

```
$ pip install troposphere
```

To install troposphere with [awacs](#) (recommended soft dependency):

```
$ pip install troposphere[policy]
```

Alternatively, you can run use setup.py to install by cloning this repository and issuing:

```
$ python setup.py install # you may need sudo depending on your python installation
```


A simple example to create an instance would look like this:

```
>>> from troposphere import Ref, Template
>>> import troposphere.ec2 as ec2
>>> t = Template()
>>> instance = ec2.Instance("myinstance")
>>> instance.ImageId = "ami-951945d0"
>>> instance.InstanceType = "t1.micro"
>>> t.add_resource(instance)
<troposphere.ec2.Instance object at 0x101bf3390>
>>> print(t.to_json())
{
  "Resources": {
    "myinstance": {
      "Properties": {
        "ImageId": "ami-951945d0",
        "InstanceType": "t1.micro"
      },
      "Type": "AWS::EC2::Instance"
    }
  }
}
```

A simple example to create an instance (YAML) would look like this:

```
>>> from troposphere import Ref, Template
>>> import troposphere.ec2 as ec2
>>> t = Template()
>>> instance = ec2.Instance("myinstance")
>>> instance.ImageId = "ami-951945d0"
>>> instance.InstanceType = "t1.micro"
>>> t.add_resource(instance)
<troposphere.ec2.Instance object at 0x101bf3390>
>>> print(t.to_yaml())
```

(continues on next page)

(continued from previous page)

```
Resources:
  myinstance:
    Properties:
      ImageId: ami-951945d0
      InstanceType: t1.micro
    Type: AWS::EC2::Instance
```

Alternatively, parameters can be used instead of properties:

```
>>> instance = ec2.Instance("myinstance", ImageId="ami-951945d0", InstanceType="t1.
↳micro")
>>> t.add_resource(instance)
<troposphere.ec2.Instance object at 0x101bf3550>
```

And `add_resource()` returns the object to make it easy to use with `Ref()`:

```
>>> instance = t.add_resource(ec2.Instance("myinstance", ImageId="ami-951945d0",
↳InstanceType="t1.micro"))
>>> Ref(instance)
<troposphere.Ref object at 0x101bf3490>
```

3.1 Examples of the error checking (full tracebacks removed for clarity):

Incorrect property being set on AWS resource:

```
>>> import troposphere.ec2 as ec2
>>> ec2.Instance("ec2instance", image="i-XXXX")
Traceback (most recent call last):
...
AttributeError: AWS::EC2::Instance object does not support attribute image
```

Incorrect type for AWS resource property:

```
>>> ec2.Instance("ec2instance", ImageId=1)
Traceback (most recent call last):
...
TypeError: ImageId is <type 'int'>, expected <type 'basestring'>
```

Missing required property for the AWS resource:

```
>>> from troposphere import Template
>>> import troposphere.ec2 as ec2
>>> t = Template()
>>> t.add_resource(ec2.Instance("ec2instance", InstanceType="m3.medium"))
<troposphere.ec2.Instance object at 0x109ee2e50>
>>> print(t.to_json())
Traceback (most recent call last):
...
ValueError: Resource ImageId required in type AWS::EC2::Instance
```

Currently supported AWS resource types

- AWS::AmazonMQ
- AWS::ApiGateway
- AWS::ApiGatewayV2
- AWS::AppStream
- AWS::AppSync
- AWS::ApplicationAutoScaling
- AWS::Athena
- AWS::AutoScaling
- AWS::AutoScalingPlans
- AWS::Batch
- AWS::Budgets
- AWS::CertificateManager
- AWS::Cloud9
- AWS::CloudFormation
- AWS::CloudFront
- AWS::CloudTrail
- AWS::CloudWatch
- AWS::CodeBuild
- AWS::CodeCommit
- AWS::CodeDeploy
- AWS::CodePipeline
- AWS::Cognito

- AWS::Config
- AWS::DAX
- AWS::DLM
- AWS::DMS
- AWS::DataPipeline
- AWS::DirectoryService
- AWS::DocDB
- AWS::DynamoDB
- AWS::EC2
- AWS::ECR
- AWS::ECS
- AWS::EFS
- AWS::EKS
- AWS::EMR
- AWS::ElastiCache
- AWS::ElasticBeanstalk
- AWS::ElasticLoadBalancing
- AWS::ElasticLoadBalancingV2
- AWS::Elasticsearch
- AWS::Events
- AWS::Glue
- AWS::GuardDuty
- AWS::IAM
- AWS::Inspector
- AWS::IoT
- AWS::IoT1Click
- AWS::IoTAnalytics
- AWS::KMS
- AWS::Kinesis
- AWS::KinesisAnalytics
- AWS::KinesisFirehose
- AWS::Lambda
- AWS::Logs
- AWS::Neptune
- AWS::OpsWorks
- AWS::OpsWorksCM

- AWS::RDS
- AWS::Redshift
- AWS::Route53
- AWS::Route53Resolver
- AWS::S3
- AWS::SDB
- AWS::SES
- AWS::SNS
- AWS::SQS
- AWS::SSM
- AWS::SageMaker
- AWS::SecretsManager
- AWS::Serverless
- AWS::ServiceCatalog
- AWS::ServiceDiscovery
- AWS::StepFunctions
- AWS::WAF
- AWS::WAFRegional
- AWS::WorkSpaces

Currently supported OpenStack resource types

- OS::Neutron::Firewall
- OS::Neutron::FirewallPolicy
- OS::Neutron::FirewallRule
- OS::Neutron::FloatingIP
- OS::Neutron::FloatingIPAssociation
- OS::Neutron::HealthMonitor
- OS::Neutron::Pool
- OS::Neutron::LoadBalancer
- OS::Neutron::Net
- OS::Neutron::PoolMember
- OS::Neutron::Port
- OS::Neutron::SecurityGroup
- OS::Nova::FloatingIP
- OS::Nova::FloatingIPAssociation
- OS::Nova::KeyPair
- OS::Nova::Server

Todo:

- Add additional validity checks

Duplicating a single instance sample would look like this

```
# Converted from EC2InstanceSample.template located at:
# http://aws.amazon.com/cloudformation/aws-cloudformation-templates/

from troposphere import Base64, FindInMap, GetAtt
from troposphere import Parameter, Output, Ref, Template
import troposphere.ec2 as ec2

template = Template()

keyname_param = template.add_parameter(Parameter(
    "KeyName",
    Description="Name of an existing EC2 KeyPair to enable SSH "
               "access to the instance",
    Type="String",
))

template.add_mapping('RegionMap', {
    "us-east-1": {"AMI": "ami-7f418316"},
    "us-west-1": {"AMI": "ami-951945d0"},
    "us-west-2": {"AMI": "ami-16fd7026"},
    "eu-west-1": {"AMI": "ami-24506250"},
    "sa-east-1": {"AMI": "ami-3e3be423"},
    "ap-southeast-1": {"AMI": "ami-74dda626"},
    "ap-northeast-1": {"AMI": "ami-dcfa4edd"}
})

ec2_instance = template.add_resource(ec2.Instance(
    "Ec2Instance",
    ImageId=FindInMap("RegionMap", Ref("AWS::Region"), "AMI"),
    InstanceType="t1.micro",
    KeyName=Ref(keyname_param),
    SecurityGroups=["default"],
    UserData=Base64("80")
))
```

(continues on next page)

```
)  
  
template.add_output([  
    Output(  
        "InstanceId",  
        Description="InstanceId of the newly created EC2 instance",  
        Value=Ref(ec2_instance),  
    ),  
    Output(  
        "AZ",  
        Description="Availability Zone of the newly created EC2 instance",  
        Value=GetAtt(ec2_instance, "AvailabilityZone"),  
    ),  
    Output(  
        "PublicIP",  
        Description="Public IP address of the newly created EC2 instance",  
        Value=GetAtt(ec2_instance, "PublicIp"),  
    ),  
    Output(  
        "PrivateIP",  
        Description="Private IP address of the newly created EC2 instance",  
        Value=GetAtt(ec2_instance, "PrivateIp"),  
    ),  
    Output(  
        "PublicDNS",  
        Description="Public DNSName of the newly created EC2 instance",  
        Value=GetAtt(ec2_instance, "PublicDnsName"),  
    ),  
    Output(  
        "PrivateDNS",  
        Description="Private DNSName of the newly created EC2 instance",  
        Value=GetAtt(ec2_instance, "PrivateDnsName"),  
    ),  
])  
  
print(template.to_json())
```

CHAPTER 7

Community

We have a Google Group, [cloudtools-dev](#), where you can ask questions and engage with the troposphere community. Issues and pull requests are always welcome!

troposphere is licensed under the [BSD 2-Clause license](#). See [LICENSE](#) for the troposphere full license text.

8.1 Quick Start

Troposphere closely follows CloudFormation, so there isn't much documentation specific to Troposphere. In this documentation there are various examples but for the most part the CloudFormation docs should be used.

8.1.1 CloudFormation Basics

- [Template Anatomy](#) - structure of a CloudFormation template.
- [Resources](#) are the basic blocks and required in any template.
- [Outputs](#) are optional but can be used to create cross-stack references. Having everything in one stack will make it very hard to manage the infrastructure. Instead, values from one stack (for example, network setup) can be exported in this section and [imported](#) by another stack (for example, EC2 setup). This way a stack used to set up a certain application can be managed or deleted without affecting other applications that might be present on the same network.
- [Intrinsic Functions](#) should be used to manipulate values that are only available at runtime. For example, assume a template that creates a subnet and attaches a routing table and network ACL to that subnet. The subnet doesn't exist when the template is created, so its ID can't be known. Instead, the route and network ACL resources are going to get the ID at runtime, by using the [Ref](#) function against the subnet.

8.1.2 Basic Usage

The following two pieces of code are intended to demonstrate basic usage of Troposphere and CloudFormation templates. First template will create two subnets and export their IDs. The second one will create an EC2 instance in one of the subnets. The comments explain how it works and where to find documentation regarding the use of CloudFormation and Troposphere.

```

#!/usr/bin/env python3
#
# learncf_subnet.py
#
# Generate a CloudFormation template that will create two subnets. This
# template exports the subnet IDs to be used by a second template which
# will create an EC2 instance in one of those subnets.
#
from __future__ import print_function

from troposphere import ec2
from troposphere import Tags, GetAtt, Ref, Sub, Export
from troposphere import Template, Output

# Create the object that will generate our template
t = Template()

# Define resources that CloudFormation will generate for us
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/resources-section-
↳structure.html

# Define the first subnet. We know that 'Subnet()' is in the ec2 module
# because in CloudFormation the Subnet resource is defined under EC2:
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-
↳subnet.html
net_learncf_1a = ec2.Subnet("netLearnCfla")

# Information about the possible properties of Subnet() can be found
# in CloudFormation docs:
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-
↳subnet.html#aws-resource-ec2-subnet-properties
net_learncf_1a.AvailabilityZone = "eu-west-1a"
net_learncf_1a.CidrBlock = "172.30.126.80/28" # ADJUST THIS VALUE
net_learncf_1a.VpcId = "vpc-abcdefgh" # ADJUST THIS VALUE
# Tags can be declared in two ways. One way is
# (1) in AWS/boto format, as a list of dictionaries where each item in the
# list has (at least) two elements. The "Key" key will be the tag key and
# the "Value" key will be the tag's Value. Confusing, but it allows for
# additional settings to be specified for each tag. For example, if a tag
# attached to an autoscaling group should be inherited by the EC2 instances
# the group launches or not.
net_learncf_1a.Tags = [
    {"Key": "Name", "Value": "learncf-1a"},
    {"Key": "Comment", "Value": "CloudFormation+Troposphere test"}]

# The subnet resource defined above must be added to the template
t.add_resource(net_learncf_1a)

# The same thing can be achieved by setting parameters to Subnet() function
# instead of properties of the object created by Subnet(). Shown below.
#
# For the second subnet we use the other method of defining tags,
# (2) by using the Tags helper function, which is defined in Troposphere
# and doesn't have an equivalent in CloudFormation.
#
# Also, we use GetAtt to read the value of an attribute from a previously
# created resource, i.e. VPC ID from the first subnet. For demo purposes.

```

(continues on next page)

(continued from previous page)

```

#
# The attributes returned by each resource can be found in the CloudFormation
# documentation, in the Returns section for that resource:
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-
# ↪subnet.html#aws-resource-ec2-subnet-getatt
#
# GetAtt documentation:
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/intrinsic-function-
# ↪reference-getatt.html
net_learncf_1b = ec2.Subnet(
    "netLearnCf1b",
    AvailabilityZone="eu-west-1b",
    CidrBlock="172.30.126.96/28", # ADJUST THIS VALUE
    VpcId=GetAtt(net_learncf_1a, "VpcId"),
    Tags=Tags(
        Name="learncf-1b",
        Comment="CloudFormation+Troposphere test"))

t.add_resource(net_learncf_1b)

# Outputs section will export the subnet IDs to be used by other stacks
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/outputs-section-
# ↪structure.html
out_net_learncf_1a = Output("outNetLearnCf1a")

# Ref is another CloudFormation intrinsic function:
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/intrinsic-function-
# ↪reference-ref.html
# If pointed to a subnet, Ref will return the subnet ID:
# https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-
# ↪subnet.html#aws-resource-ec2-subnet-ref
out_net_learncf_1a.Value = Ref(net_learncf_1a)
# Append the subnet title (Logical ID) to the stack name and set that as the
# exported property. Importing it in another stack will return the Value
# we set above to that stack.
#
# Sub stands for 'substitute', another CloudFormation intrinsic function.
out_net_learncf_1a.Export = Export(Sub(
    "${AWS::StackName}-" + net_learncf_1a.title))

# Similar output for the second subnet
out_net_learncf_1b = Output("outNetLearnCf1b")
out_net_learncf_1b.Value = Ref(net_learncf_1b)
out_net_learncf_1b.Export = Export(Sub(
    "${AWS::StackName}-" + net_learncf_1b.title))

# Add outputs to template
t.add_output(out_net_learncf_1a)
t.add_output(out_net_learncf_1b)

# Finally, write the template to a file
with open('learncf-subnet.yaml', 'w') as f:
    f.write(t.to_yaml())

```

And the EC2 instance template:

```
#!/usr/bin/env python3
#
# learncf_ec2.py
#
# Generate a CloudFormation template that creates an EC2 instance in a
# subnet which was created previously by another template (learncf-subnet)
#
from __future__ import print_function

from troposphere import ec2
from troposphere import Tags, ImportValue
from troposphere import Template

# create the object that will generate our template
t = Template()

ec2_learncf_1a = ec2.Instance("ec2LearnCf1a")
ec2_learncf_1a.ImageId = "ami-e487179d" # ADJUST IF NEEDED
ec2_learncf_1a.InstanceType = "t2.micro"
# We set the subnet to start this instance in by importing the subnet ID
# from the other CloudFormation stack, which previously created it.
# An example of cross-stack reference used to split stacks into
# manageable pieces. Each export must have a unique name in its account
# and region, so the template name was prepended to the resource name.
ec2_learncf_1a.SubnetId = ImportValue("learncf-subnet-netLearnCf1a")
ec2_learncf_1a.Tags = Tags(
    Name="learncf",
    Comment="Learning CloudFormation and Troposphere")

t.add_resource(ec2_learncf_1a)

# Finally, write the template to a file
with open('learncf-ec2.yaml', 'w') as f:
    f.write(t.to_yaml())
```

After the .yaml files are generated using the code above stacks can be created from the command line like this:

```
aws cloudformation create-stack --stack-name learncf-subnet --template-body file://
↳learncf-subnet.yaml
aws cloudformation create-stack --stack-name learncf-ec2 --template-body file://
↳learncf-ec2.yaml
```

8.2 examples

8.2.1 examples package

Submodules

examples.ApiGateway module

examples.ApplicationAutoScalingSample module

examples.ApplicationELB module

`examples.Autoscaling` module

`examples.AutoscalingHTTPRequests` module

`examples.Batch` module

`examples.CertificateManagerSample` module

`examples.ClassExtensions` module

`examples.CloudFormation_Init_ConfigSet` module

`examples.CloudFront_Distribution_S3` module

`examples.CloudFront_StreamingDistribution_S3` module

`examples.CloudTrail` module

`examples.CloudWatchEventsSample` module

`examples.CodeBuild` module

`examples.CodeDeploy` module

`examples.CodePipeline` module

`examples.CustomResource` module

`examples.Dlm` module

`examples.DynamoDB_Table` module

`examples.DynamoDB_Table_With_GSI_And_NonKeyAttributes_Projection` module

`examples.DynamoDB_Table_With_GlobalSecondaryIndex` module

`examples.DynamoDB_Tables_OnDemand` module

`examples.EC2Conditions` module

`examples.EC2InstanceSample` module

`examples.EC2_Remove_Ephemeral_Drive` module

`examples.ECRSample` module

`examples.ECSCluster` module

`examples.ECSFargate` module

`examples.EFS` module

examples.ELBSample module
examples.EMR_Cluster module
examples.ElastiCacheRedis module
examples.ElasticBeanstalk_Nodejs_Sample module
examples.ElasticsearchDomain module
examples.Firehose_with_Redshift module
examples.IAM_Policies_SNS_Publish_To_SQS module
examples.IAM_Roles_and_InstanceProfiles module
examples.IAM_Users_Groups_and_Policies module
examples.IAM_Users_snippet module
examples.IoTSample module
examples.Kinesis_Stream module
examples.Lambda module
examples.Metadata module
examples.NatGateway module
examples.NetworkLB module
examples.OpenStack_AutoScaling module
examples.OpenStack_Server module
examples.OpsWorksSnippet module
examples.RDS_Snapshot_On_Delete module
examples.RDS_VPC module
examples.RDS_with_DBParameterGroup module
examples.Redshift module
examples.RedshiftClusterInVpc module
examples.Route53_A module
examples.Route53_CNAME module

examples.Route53_RoundRobin module

examples.S3_Bucket module

examples.S3_Bucket_With_AccelerateConfiguration module

examples.S3_Bucket_With_Versioning_And_Lifecycle_Rules module

examples.S3_Website_Bucket_With_Retain_On_Delete module

examples.SQSDLQ module

examples.SQSEncrypt module

examples.SQS_With_CloudWatch_Alarms module

examples.SSMExample module

examples.Secretsmanager module

examples.Secretsmanager_Rds module

examples.Serverless_Api_Backend module

examples.Serverless_Deployment_Preference module

examples.Serverless_S3_Processor module

examples.VPC_EC2_Instance_With_Multiple_Dynamic_IPAddresses module

examples.VPC_With_VPN_Connection module

examples.VPC_single_instance_in_subnet module

examples.WAF_Common_Attacks_Sample module

examples.WAF_Regional_Common_Attacks_Sample module

examples.WaitObject module

Module contents

8.3 troposphere

8.3.1 troposphere package

Subpackages

troposphere.helpers package

Submodules

troposphere.helpers.userdata module

`troposphere.helpers.userdata.from_file` (*filepath*, *delimiter="*, *blanklines=False*)
Imports userdata from a file.

:param filepath The absolute path to the file.

Param *delimiter* Delimiter to use with the `troposphere.Join()`.

:param blanklines If blank lines should be ignored

rtype: `troposphere.Base64` :return The base64 representation of the file.

Module contents

troposphere.openstack package

Submodules

troposphere.openstack.heat module

Openstack Heat

Due to the strange nature of the OpenStack compatibility layer, some values that should be integers fail to validate and need to be represented as strings. For this reason, we duplicate the `AWS::AutoScaling::AutoScalingGroup` and change these types.

```
class troposphere.openstack.heat.AWSAutoScalingGroup (title, template=None, validation=True, **kwargs)
```

Bases: `troposphere.AWSObject`

Fix issues with OpenStack compatibility layer.

Due to the strange nature of the OpenStack compatibility layer, some values that should be integers fail to validate and need to be represented as strings. For this reason, we duplicate the `AWS::AutoScaling::AutoScalingGroup` and change these types.

```
props = {'AvailabilityZones': (<type 'list'>, True), 'Cooldown': (<function integer to string at 0x7f30f87f5938>, True), 'ResourceType': 'AWS::AutoScaling::AutoScalingGroup'}
```

troposphere.openstack.neutron module

```
class troposphere.openstack.neutron.AddressPair (title=None, **kwargs)
```

Bases: `troposphere.AWSPROPERTY`

```
props = {'ip_address': (<type 'basestring'>, True), 'mac_address': (<type 'basestring'>, True)}
```

```
class troposphere.openstack.neutron.Firewall (title, template=None, validation=True, **kwargs)
```

Bases: `troposphere.AWSObject`

```
props = {'admin_state_up': (<function boolean to string at 0x7f30f87f5938>, False), 'description': (<type 'basestring'>, True)}
```

```

    resource_type = 'OS::Neutron::Firewall'
class troposphere.openstack.neutron.FirewallPolicy(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'audited': (<function boolean at 0x7f30f87f5938>, False), 'description': (<
    resource_type = 'OS::Neutron::FirewallPolicy'
class troposphere.openstack.neutron.FirewallRule(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'action': (<type 'basestring'>, False), 'description': (<type 'basestring'>
    resource_type = 'OS::Neutron::FirewallRule'
    validate()
class troposphere.openstack.neutron.FixedIP(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ip_address': (<type 'basestring'>, False), 'subnet_id': (<type 'basestring'>
class troposphere.openstack.neutron.FloatingIP(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'fixed_ip_address': (<type 'basestring'>, False), 'floating_network_id': (<
    resource_type = 'OS::Neutron::FloatingIP'
class troposphere.openstack.neutron.FloatingIPAssociation(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'fixed_ip_address': (<type 'basestring'>, False), 'floatingip_id': (<type '
    resource_type = 'OS::Neutron::FloatingIPAssociation'
class troposphere.openstack.neutron.HealthMonitor(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'admin_state_up': (<function boolean at 0x7f30f87f5938>, False), 'delay': (<
    resource_type = 'OS::Neutron::HealthMonitor'
    validate()
class troposphere.openstack.neutron.LoadBalancer(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'members': (<type 'list'>, False), 'pool_id': (<class 'troposphere.openstack
    resource_type = 'OS::Neutron::LoadBalancer'
class troposphere.openstack.neutron.Net(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'admin_state_up': (<function boolean at 0x7f30f87f5938>, False), 'name': (<
    resource_type = 'OS::Neutron::Net'

```

```
class troposphere.openstack.neutron.Pool (title, template=None, validation=True,
                                         **kwargs)
    Bases: troposphere.AWSObject
    props = {'admin_state_up': (<function boolean at 0x7f30f87f5938>, False), 'description':
    resource_type = 'OS::Neutron::Pool'
    validate()

class troposphere.openstack.neutron.PoolMember (title, template=None, validation=True,
                                                  **kwargs)
    Bases: troposphere.AWSObject
    props = {'address': (<type 'basestring'>, True), 'admin_state_up': (<function boolean
    resource_type = 'OS::Neutron::PoolMember'

class troposphere.openstack.neutron.Port (title, template=None, validation=True,
                                           **kwargs)
    Bases: troposphere.AWSObject
    props = {'admin_state_up': (<function boolean at 0x7f30f87f5938>, False), 'allowed_ad
    resource_type = 'OS::Neutron::Port'

class troposphere.openstack.neutron.SecurityGroup (title, template=None, valida-
                                                    tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'description': (<type 'basestring'>, True), 'name': (<type 'basestring'>, F
    resource_type = 'OS::Neutron::SecurityGroup'

class troposphere.openstack.neutron.SecurityGroupRule (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'direction': (<type 'basestring'>, False), 'ethertype': (<type 'basestring'
    validate()

class troposphere.openstack.neutron.SessionPersistence (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'cookie_name': (<type 'basestring'>, False), 'type': (<type 'basestring'>,
    validate()

class troposphere.openstack.neutron.VIP (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'address': (<type 'basestring'>, False), 'admin_state_up': (<function boolean
```

troposphere.openstack.nova module

```
class troposphere.openstack.nova.BlockDeviceMapping (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'delete_on_termination': (<function boolean at 0x7f30f87f5938>, False), 'dev

class troposphere.openstack.nova.BlockDeviceMappingV2 (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'boot_index': (<function integer at 0x7f30f87f59b0>, False), 'delete_on_term
    validate()
```



```
class troposphere.openstack.nova.FloatingIP(title, template=None, validation=True,
                                           **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'pool': (<type 'basestring'>, False)}
```

```
resource_type = 'OS::Nova::FloatingIP'
```

```
class troposphere.openstack.nova.FloatingIPAssociation(title, template=None, validation=True,
                                                       **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'floating_ip': (<type 'basestring'>, True), 'server_ip': (<type 'basestring'>, True)}
```

```
resource_type = 'OS::Nova::FloatingIPAssociation'
```

```
class troposphere.openstack.nova.KeyPair(title, template=None, validation=True,
                                          **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'name': (<type 'basestring'>, True), 'public_key': (<type 'basestring'>, False)}
```

```
resource_type = 'OS::Nova::KeyPair'
```

```
class troposphere.openstack.nova.Network(title=None, **kwargs)
```

Bases: *troposphere.AWSProperty*

```
props = {'fixed_ip': (<type 'basestring'>, False), 'network': (<type 'basestring'>, True)}
```

```
class troposphere.openstack.nova.Server(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'admin_pass': (<type 'basestring'>, False), 'admin_user': (<type 'basestring'>, True)}
```

```
resource_type = 'OS::Nova::Server'
```

```
validate()
```

Module contents

OpenStack

The package to support OpenStack templates using troposphere.

Submodules

troposphere.amazonmq module

```
class troposphere.amazonmq.Broker(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AutoMinorVersionUpgrade': (<function boolean at 0x7f30f87f5938>, True), 'BrokerName': (<type 'basestring'>, True)}
```

```
resource_type = 'AWS::AmazonMQ::Broker'
```

```
class troposphere.amazonmq.Configuration(title, template=None, validation=True,
                                          **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'Data': (<type 'basestring'>, True), 'Description': (<type 'basestring'>, False)}
```

```
resource_type = 'AWS::AmazonMQ::Configuration'
```

```
class troposphere.amazonmq.ConfigurationAssociation(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Broker': (<type 'basestring'>, True), 'Configuration': (<class 'troposphere.amazonmq.ConfigurationAssociation'>, True)}
    resource_type = 'AWS::AmazonMQ::ConfigurationAssociation'

class troposphere.amazonmq.ConfigurationId(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Id': (<type 'basestring'>, True), 'Revision': (<function integer at 0x7f30f87f5938>, True)}

class troposphere.amazonmq.LogsConfiguration(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Audit': (<function boolean at 0x7f30f87f5938>, False), 'General': (<function boolean at 0x7f30f87f5938>, False)}

class troposphere.amazonmq.MaintenanceWindow(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DayOfWeek': (<type 'basestring'>, True), 'TimeOfDay': (<type 'basestring'>, True)}

class troposphere.amazonmq.User(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ConsoleAccess': (<function boolean at 0x7f30f87f5938>, False), 'Groups': (<function boolean at 0x7f30f87f5938>, False)}
```

troposphere.analytics module

```
class troposphere.analytics.Application(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApplicationCode': (<type 'basestring'>, False), 'ApplicationDescription': (<type 'basestring'>, True)}
    resource_type = 'AWS::KinesisAnalytics::Application'

class troposphere.analytics.ApplicationOutput(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApplicationName': (<type 'basestring'>, True), 'Output': (<class 'troposphere.analytics.ApplicationOutput'>, True)}
    resource_type = 'AWS::KinesisAnalytics::ApplicationOutput'

class troposphere.analytics.ApplicationReferenceDataSource(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApplicationName': (<type 'basestring'>, True), 'ReferenceDataSource': (<class 'troposphere.analytics.ApplicationReferenceDataSource'>, True)}
    resource_type = 'AWS::KinesisAnalytics::ApplicationReferenceDataSource'

class troposphere.analytics.CSVMappingParameters(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'RecordColumnDelimiter': (<type 'basestring'>, True), 'RecordRowDelimiter': (<type 'basestring'>, True)}

class troposphere.analytics.DestinationSchema(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'RecordFormatType': (<type 'basestring'>, False)}
```

```

class troposphere.analytics.Input (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'InputParallelism': (<class 'troposphere.analytics.InputParallelism'>, False)}

class troposphere.analytics.InputLambdaProcessor (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceARN': (<type 'basestring'>, True), 'RoleARN': (<type 'basestring'>, True)}

class troposphere.analytics.InputParallelism (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Count': (<function integer at 0x7f30f87f59b0>, True)}

class troposphere.analytics.InputProcessingConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'InputLambdaProcessor': (<class 'troposphere.analytics.InputLambdaProcessor'>, True)}

class troposphere.analytics.InputSchema (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'RecordColumns': ([<class 'troposphere.analytics.RecordColumn'>], True), 'RecordSchema': (<type 'basestring'>, True)}

class troposphere.analytics.JSONMappingParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'RecordRowPath': (<type 'basestring'>, True)}

class troposphere.analytics.KinesisFirehoseInput (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceARN': (<type 'basestring'>, True), 'RoleARN': (<type 'basestring'>, True)}

class troposphere.analytics.KinesisFirehoseOutput (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceARN': (<type 'basestring'>, True), 'RoleARN': (<type 'basestring'>, True)}

class troposphere.analytics.KinesisStreamsInput (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceARN': (<type 'basestring'>, True), 'RoleARN': (<type 'basestring'>, True)}

class troposphere.analytics.KinesisStreamsOutput (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceARN': (<type 'basestring'>, True), 'RoleARN': (<type 'basestring'>, True)}

class troposphere.analytics.LambdaOutput (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceARN': (<type 'basestring'>, True), 'RoleARN': (<type 'basestring'>, True)}

class troposphere.analytics.MappingParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CSVMappingParameters': (<class 'troposphere.analytics.CSVMappingParameters'>, True)}

class troposphere.analytics.Output (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DestinationSchema': (<class 'troposphere.analytics.DestinationSchema'>, True)}

class troposphere.analytics.RecordColumn (title=None, **kwargs)
    Bases: troposphere.AWSProperty

```

```
    props = {'Mapping': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, True)
class troposphere.analytics.RecordFormat (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MappingParameters': (<class 'troposphere.analytics.MappingParameters'>, False)
class troposphere.analytics.ReferenceDataSource (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ReferenceSchema': (<class 'troposphere.analytics.ReferenceSchema'>, True),
class troposphere.analytics.ReferenceSchema (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'RecordColumns': ([<class 'troposphere.analytics.RecordColumn'>], True), 'Re
class troposphere.analytics.S3ReferenceDataSource (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BucketARN': (<type 'basestring'>, False), 'FileKey': (<type 'basestring'>, True)
```

troposphere.apigateway module

```
class troposphere.apigateway.AccessLogSetting (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DestinationArn': (<type 'basestring'>, False), 'Format': (<type 'basestring'>, True)
class troposphere.apigateway.Account (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CloudWatchRoleArn': (<type 'basestring'>, False)}
    resource_type = 'AWS::ApiGateway::Account'
class troposphere.apigateway.ApiKey (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CustomerId': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, True)
    resource_type = 'AWS::ApiGateway::ApiKey'
class troposphere.apigateway.ApiStage (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ApiId': (<type 'basestring'>, False), 'Stage': (<type 'basestring'>, False)
class troposphere.apigateway.Authorizer (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AuthType': (<type 'basestring'>, False), 'AuthorizerCredentials': (<type 'basestring'>, True)
    resource_type = 'AWS::ApiGateway::Authorizer'
class troposphere.apigateway.BasePathMapping (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'BasePath': (<type 'basestring'>, False), 'DomainName': (<type 'basestring'>, True)
    resource_type = 'AWS::ApiGateway::BasePathMapping'
class troposphere.apigateway.CanarySetting (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```

    props = {'DeploymentId': (<type 'basestring'>, False), 'PercentTraffic': ([<function
class troposphere.apigateway.ClientCertificate (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False)}
    resource_type = 'AWS::ApiGateway::ClientCertificate'
class troposphere.apigateway.Deployment (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DeploymentCanarySettings': (<class 'troposphere.apigateway.DeploymentCanary
    resource_type = 'AWS::ApiGateway::Deployment'
troposphere.apigateway.DeploymentCanarySetting
    alias of troposphere.apigateway.DeploymentCanarySettings
class troposphere.apigateway.DeploymentCanarySettings (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PercentTraffic': ([<function double at 0x7f30f87f5b90>], False), 'StageVari
class troposphere.apigateway.DocumentationPart (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'Location': (<class 'troposphere.apigateway.Location'>, True), 'Properties':
    resource_type = 'AWS::ApiGateway::DocumentationPart'
class troposphere.apigateway.DocumentationVersion (title, template=None, valida-
                                                tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'DocumentationVersion': (<type
    resource_type = 'AWS::ApiGateway::DocumentationVersion'
class troposphere.apigateway.DomainName (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CertificateArn': (<type 'basestring'>, False), 'DomainName': (<type 'bases
    resource_type = 'AWS::ApiGateway::DomainName'
class troposphere.apigateway.EndpointConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Types': ([<type 'basestring'>], False)}
class troposphere.apigateway.GatewayResponse (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'ResponseParameters': (<type 'dict'>, False), 'ResponseTemplates': (<type '
    resource_type = 'AWS::ApiGateway::GatewayResponse'
class troposphere.apigateway.Integration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CacheKeyParameters': ([<type 'basestring'>], False), 'CacheNamespace': (<t
class troposphere.apigateway.IntegrationResponse (title=None, **kwargs)
    Bases: troposphere.AWSProperty

```

```
    props = {'ContentHandling': (<type 'basestring'>, False), 'ResponseParameters': (<ty
class troposphere.apigateway.Location (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Method': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, False)
class troposphere.apigateway.Method (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiKeyRequired': (<type 'bool'>, False), 'AuthorizationScopes': ([<type 'b
    resource_type = 'AWS::ApiGateway::Method'
class troposphere.apigateway.MethodResponse (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ResponseModels': (<type 'dict'>, False), 'ResponseParameters': (<type 'dic
class troposphere.apigateway.MethodSetting (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CacheDataEncrypted': (<type 'bool'>, False), 'CacheTtlInSeconds': (<functi
class troposphere.apigateway.Model (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ContentType': (<type 'basestring'>, False), 'Description': (<type 'basestr
    resource_type = 'AWS::ApiGateway::Model'
    validate()
class troposphere.apigateway.QuotaSettings (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Limit': (<function positive_integer at 0x7f30f87f5a28>, False), 'Offset':
class troposphere.apigateway.RequestValidator (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, True), 'RestApiId': (<type 'basestring'>, True
    resource_type = 'AWS::ApiGateway::RequestValidator'
class troposphere.apigateway.Resource (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ParentId': (<type 'basestring'>, True), 'PathPart': (<type 'basestring'>,
    resource_type = 'AWS::ApiGateway::Resource'
class troposphere.apigateway.RestApi (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiKeySourceType': (<type 'basestring'>, False), 'BinaryMediaTypes': ([<ty
    resource_type = 'AWS::ApiGateway::RestApi'
class troposphere.apigateway.S3Location (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Bucket': (<type 'basestring'>, False), 'ETag': (<type 'basestring'>, False)
class troposphere.apigateway.Stage (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
```

```

    props = {'AccessLogSetting': (<class 'troposphere.apigateway.AccessLogSetting'>, False),
             'resource_type': 'AWS::ApiGateway::Stage'}
troposphere.apigateway.StageCanarySetting
    alias of troposphere.apigateway.CanarySetting
class troposphere.apigateway.StageDescription(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AccessLogSetting': (<class 'troposphere.apigateway.AccessLogSetting'>, False),
             'validate': ()}
class troposphere.apigateway.StageKey(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'RestApiId': (<type 'basestring'>, False), 'StageName': (<type 'basestring'>, False)}
class troposphere.apigateway.ThrottleSettings(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BurstLimit': (<function positive_integer at 0x7f30f87f5a28>, False), 'RateLimit': (<function positive_integer at 0x7f30f87f5a28>, False)}
class troposphere.apigateway.UsagePlan(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiStages': ([<class 'troposphere.apigateway.ApiStage'>], False), 'Description': (<type 'basestring'>, False),
             'resource_type': 'AWS::ApiGateway::UsagePlan'}
class troposphere.apigateway.UsagePlanKey(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'KeyId': (<type 'basestring'>, True), 'KeyType': (<type 'basestring'>, True),
             'resource_type': 'AWS::ApiGateway::UsagePlanKey'}
class troposphere.apigateway.VpcLink(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, False),
             'resource_type': 'AWS::ApiGateway::VpcLink'}
troposphere.apigateway.validate_authorizer_ttl(ttl_value)
    Validate authorizer ttl timeout :param ttl_value: The TTL timeout in seconds :return: The provided TTL value if valid
troposphere.apigateway.validate_gateway_response_type(response_type)
    Validate response type :param response_type: The GatewayResponse response type :return: The provided value if valid

```

troposphere.apigatewayv2 module

```

class troposphere.apigatewayv2.AccessLogSettings(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DestinationArn': (<type 'basestring'>, False), 'Format': (<type 'basestring'>, False)}
class troposphere.apigatewayv2.Api(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiKeySelectionExpression': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False),

```

```
    resource_type = 'AWS::ApiGatewayV2::Api'
class troposphere.apigatewayv2.Authorizer(title, template=None, validation=True,
                                          **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'AuthorizerCredentialsArn': (<type 'b
    resource_type = 'AWS::ApiGatewayV2::Authorizer'
class troposphere.apigatewayv2.Deployment(title, template=None, validation=True,
                                          **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'Description': (<type 'basestring'>,
    resource_type = 'AWS::ApiGatewayV2::Deployment'
class troposphere.apigatewayv2.Integration(title, template=None, validation=True,
                                          **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'ConnectionType': (<type 'basestring'
    resource_type = 'AWS::ApiGatewayV2::Integration'
class troposphere.apigatewayv2.IntegrationResponse(title, template=None, valida-
                                                    tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'ContentHandlingStrategy': (<function
    resource_type = 'AWS::ApiGatewayV2::IntegrationResponse'
class troposphere.apigatewayv2.Model(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'ContentType': (<type 'basestring'>,
    resource_type = 'AWS::ApiGatewayV2::Model'
    validate()
class troposphere.apigatewayv2.Route(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'ApiKeyRequired': (<function boolean
    resource_type = 'AWS::ApiGatewayV2::Route'
class troposphere.apigatewayv2.RouteResponse(title, template=None, validation=True,
                                          **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'ModelSelectionExpression': (<type 'b
    resource_type = 'AWS::ApiGatewayV2::RouteResponse'
class troposphere.apigatewayv2.RouteSettings(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DataTraceEnabled': (<type 'basestring'>, False), 'DetailedMetricsEnabled':
class troposphere.apigatewayv2.Stage(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AccessLogSettings': (<class 'troposphere.apigatewayv2.AccessLogSettings'>,
```



```
resource_type = 'AWS::ApiGatewayV2::Stage'
```

```
troposphere.apigatewayv2.validate_authorizer_ttl(ttl_value)
```

Validate authorizer ttl timeout :param ttl_value: The TTL timeout in seconds :return: The provided TTL value if valid

```
troposphere.apigatewayv2.validate_authorizer_type(authorizer_type)
```

```
troposphere.apigatewayv2.validate_content_handling_strategy(content_handling_strategy)
```

```
troposphere.apigatewayv2.validate_integration_type(integration_type)
```

```
troposphere.apigatewayv2.validate_logging_level(logging_level)
```

```
troposphere.apigatewayv2.validate_passthrough_behavior(passthrough_behavior)
```

troposphere.applicationautoscaling module

```
class troposphere.applicationautoscaling.CustomizedMetricSpecification(title=None,
                                                                    **kwargs)
```

Bases: *troposphere.AWSProperty*

```
props = {'Dimensions': ([<class 'troposphere.applicationautoscaling.MetricDimension'>
```

```
class troposphere.applicationautoscaling.MetricDimension(title=None, **kwargs)
```

Bases: *troposphere.AWSProperty*

```
props = {'Name': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}
```

```
class troposphere.applicationautoscaling.PredefinedMetricSpecification(title=None,
                                                                    **kwargs)
```

Bases: *troposphere.AWSProperty*

```
props = {'PredefinedMetricType': (<type 'basestring'>, True), 'ResourceLabel': (<type
```

```
class troposphere.applicationautoscaling.ScalableTarget(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'MaxCapacity': (<function integer at 0x7f30f87f59b0>, True), 'MinCapacity':
resource_type = 'AWS::ApplicationAutoScaling::ScalableTarget'
```

```
class troposphere.applicationautoscaling.ScalableTargetAction(title=None,
                                                            **kwargs)
```

Bases: *troposphere.AWSProperty*

```
props = {'MaxCapacity': (<function integer at 0x7f30f87f59b0>, False), 'MinCapacity':
```

```
class troposphere.applicationautoscaling.ScalingPolicy(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'PolicyName': (<type 'basestring'>, True), 'PolicyType': (<type 'basestring'
resource_type = 'AWS::ApplicationAutoScaling::ScalingPolicy'
```

```
class troposphere.applicationautoscaling.ScheduledAction(title=None, **kwargs)
```

Bases: *troposphere.AWSProperty*

```
props = {'EndTime': (<type 'basestring'>, False), 'ScalableTargetAction': (<class 't
```

```
class troposphere.applicationautoscaling.StepAdjustment(title=None, **kwargs)
```

Bases: *troposphere.AWSProperty*

```
props = {'MetricIntervalLowerBound': (<function integer at 0x7f30f87f59b0>, False), 'I
```

```
class troposphere.applicationautoscaling.StepScalingPolicyConfiguration (title=None,
                                                                    **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AdjustmentType': (<type 'basestring'>, False), 'Cooldown': (<function integer at 0x7f30f87f5938>, True)}

class troposphere.applicationautoscaling.TargetTrackingScalingPolicyConfiguration (title=None,
                                                                    **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CustomizedMetricSpecification': (<class 'troposphere.applicationautoscaling.TargetTrackingScalingPolicyConfiguration.CustomizedMetricSpecification'>, True)}
```

troposphere.appstream module

```
class troposphere.appstream.ApplicationSettings (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Enabled': (<function boolean at 0x7f30f87f5938>, True), 'SettingsGroup': (<type 'basestring'>, False)}

class troposphere.appstream.ComputeCapacity (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DesiredInstances': (<function integer at 0x7f30f87f59b0>, True)}

class troposphere.appstream.DirectoryConfig (title, template=None, validation=True,
                                                                    **kwargs)
    Bases: troposphere.AWSObject
    props = {'DirectoryName': (<type 'basestring'>, True), 'OrganizationalUnitDistinguisher': (<type 'basestring'>, False)}
    resource_type = 'AWS::AppStream::DirectoryConfig'

class troposphere.appstream.DomainJoinInfo (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DirectoryName': (<type 'basestring'>, False), 'OrganizationalUnitDistinguisher': (<type 'basestring'>, False)}

class troposphere.appstream.Fleet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ComputeCapacity': (<class 'troposphere.appstream.ComputeCapacity'>, True), 'Description': (<type 'basestring'>, False)}
    resource_type = 'AWS::AppStream::Fleet'

class troposphere.appstream.ImageBuilder (title, template=None, validation=True,
                                                                    **kwargs)
    Bases: troposphere.AWSObject
    props = {'AppstreamAgentVersion': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False)}
    resource_type = 'AWS::AppStream::ImageBuilder'

class troposphere.appstream.ServiceAccountCredentials (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AccountName': (<type 'basestring'>, True), 'AccountPassword': (<type 'basestring'>, False)}

class troposphere.appstream.Stack (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApplicationSettings': (<class 'troposphere.appstream.ApplicationSettings'>, True), 'Description': (<type 'basestring'>, False)}
    resource_type = 'AWS::AppStream::Stack'
```

```

class troposphere.appstream.StackFleetAssociation(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'FleetName': (<type 'basestring'>, True), 'StackName': (<type 'basestring'>, True)}
    resource_type = 'AWS::AppStream::StackFleetAssociation'

class troposphere.appstream.StackUserAssociation(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AuthenticationType': (<type 'basestring'>, True), 'SendEmailNotification': (<type 'boolean'>, True)}
    resource_type = 'AWS::AppStream::StackUserAssociation'

class troposphere.appstream.StorageConnector(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'ConnectorType': (<type 'basestring'>, True), 'Domains': ([<type 'basestring'>], True)}

class troposphere.appstream.User(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AuthenticationType': (<type 'basestring'>, True), 'FirstName': (<type 'basestring'>, True)}
    resource_type = 'AWS::AppStream::User'

class troposphere.appstream.UserSetting(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Action': (<type 'basestring'>, True), 'Permission': (<type 'basestring'>, True)}

class troposphere.appstream.VpcConfig(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'SecurityGroupIds': ([<type 'basestring'>], False), 'SubnetIds': ([<type 'basestring'>], False)}

```

troposphere.appsync module

```

class troposphere.appsync.ApiKey(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiKey': (<type 'basestring'>, True), 'Description': (<type 'basestring'>, True)}
    resource_type = 'AWS::AppSync::ApiKey'

class troposphere.appsync.AuthorizationConfig(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'AuthorizationType': (<type 'basestring'>, True), 'AwsIamConfig': (<class 'troposphere.appsync.AwsIamConfig'>, True)}

class troposphere.appsync.AwsIamConfig(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'SigningRegion': (<type 'basestring'>, False), 'SigningServiceName': (<type 'basestring'>, True)}

class troposphere.appsync.DataSource(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiKey': (<type 'basestring'>, True), 'Description': (<type 'basestring'>, True)}
    resource_type = 'AWS::AppSync::DataSource'

class troposphere.appsync.DynamoDBConfig(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

```

```
    props = {'AwsRegion': (<type 'basestring'>, True), 'TableName': (<type 'basestring'>
class troposphere.appsync.ElasticsearchConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AwsRegion': (<type 'basestring'>, True), 'Endpoint': (<type 'basestring'>,
class troposphere.appsync.FunctionConfiguration (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'DataSourceName': (<type 'basestring'>
    resource_type = 'AWS::AppSync::FunctionConfiguration'
class troposphere.appsync.GraphQLApi (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AuthenticationType': (<type 'basestring'>, True), 'LogConfig': (<class 'tr
    resource_type = 'AWS::AppSync::GraphQLApi'
class troposphere.appsync.GraphQLSchema (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'Definition': (<type 'basestring'>, F
    resource_type = 'AWS::AppSync::GraphQLSchema'
class troposphere.appsync.HttpConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AuthorizationConfig': (<class 'troposphere.appsync.AuthorizationConfig'>, F
class troposphere.appsync.LambdaConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'LambdaFunctionArn': (<type 'basestring'>, True)}
class troposphere.appsync.LogConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CloudWatchLogsRoleArn': (<type 'basestring'>, False), 'FieldLogLevel': (<t
class troposphere.appsync.OpenIDConnectConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AuthTTL': (<type 'float'>, False), 'ClientId': (<type 'basestring'>, False)
class troposphere.appsync.PipelineConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Functions': ([<type 'basestring'>], False)}
class troposphere.appsync.RdsHttpEndpointConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AwsRegion': (<type 'basestring'>, False), 'AwsSecretStoreArn': (<type 'bas
class troposphere.appsync.RelationalDatabaseConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'RdsHttpEndpointConfig': (<class 'troposphere.appsync.RdsHttpEndpointConfig'
class troposphere.appsync.Resolver (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApiId': (<type 'basestring'>, True), 'DataSourceName': (<type 'basestring'
```

```

    resource_type = 'AWS::AppSync::Resolver'
class troposphere.appsync.UserPoolConfig (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'AppIdClientRegex': (<type 'basestring'>, False), 'AwsRegion': (<type 'base
troposphere.appsync.resolver_kind_validator(x)

```

troposphere.ask module

```

class troposphere.ask.AuthenticationConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DefaultAttributes': (<function json_checker at 0x7f30f87fb500>, False), 'De
class troposphere.ask.Skill (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AuthenticationConfiguration': (<class 'troposphere.ask.AuthenticationConfig
    resource_type = 'Alexa::ASK::Skill'
class troposphere.ask.SkillPackage (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'ClientId': (<type 'basestring'>, True), 'ClientSecret': (<type 'basestring

```

troposphere.athena module

```

class troposphere.athena.NamedQuery (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Database': (<type 'basestring'>, True), 'Description': (<type 'basestring'>
    resource_type = 'AWS::Athena::NamedQuery'

```

troposphere.autoscaling module

```

class troposphere.autoscaling.AutoScalingGroup (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoScalingGroupName': (<type 'basestring'>, False), 'AvailabilityZones':
    resource_type = 'AWS::AutoScaling::AutoScalingGroup'
    validate()
class troposphere.autoscaling.BlockDeviceMapping (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DeviceName': (<type 'basestring'>, True), 'Ebs': (<class 'troposphere.auto
class troposphere.autoscaling.CustomizedMetricSpecification (title=None,
                                                             **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Dimensions': ([<class 'troposphere.autoscaling.MetricDimension'>], False),
class troposphere.autoscaling.EBSBlockDevice (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

```

```
    props = {'DeleteOnTermination': (<function boolean at 0x7f30f87f5938>, False), 'EncryptionInTransit': (False, False)}
class troposphere.autoscaling.InstancesDistribution(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'OnDemandAllocationStrategy': (<type 'basestring'>, False), 'OnDemandBaseCapacity': (False, False)}
class troposphere.autoscaling.LaunchConfiguration(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AssociatePublicIpAddress': (<function boolean at 0x7f30f87f5938>, False), 'EBSOptimized': (False, False), 'InstanceProfileName': (<type 'basestring'>, True), 'IamInstanceProfile': (<type 'basestring'>, True), 'InstanceType': (<type 'basestring'>, True), 'KeyName': (<type 'basestring'>, True), 'LinuxMarketplaceOptions': (<type 'dict'>, True), 'MachineImage': (<type 'basestring'>, True), 'MachineImageOwner': (<type 'basestring'>, True), 'MachineImageVersion': (<type 'basestring'>, True), 'MetadataOptions': (<type 'dict'>, True), 'NetworkInterfaces': (<type 'list'>, True), 'Placement': (<type 'dict'>, True), 'SubnetId': (<type 'basestring'>, True), 'TagSpecifications': (<type 'list'>, True), 'UserData': (<type 'basestring'>, True), 'WeightedInstancesCount': (<type 'integer'>, True)}
    resource_type = 'AWS::AutoScaling::LaunchConfiguration'
class troposphere.autoscaling.LaunchTemplate(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'LaunchTemplateSpecification': (<class 'troposphere.autoscaling.LaunchTemplateSpecification'>, True)}
class troposphere.autoscaling.LaunchTemplateOverrides(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'InstanceType': (<type 'basestring'>, False)}
class troposphere.autoscaling.LaunchTemplateSpecification(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'LaunchTemplateId': (<type 'basestring'>, False), 'LaunchTemplateName': (<type 'basestring'>, True), 'Version': (<type 'basestring'>, True)}
    validate()
class troposphere.autoscaling.LifecycleHook(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoScalingGroupName': (<type 'basestring'>, True), 'DefaultResult': (<type 'basestring'>, True), 'LifecycleTransition': (<type 'basestring'>, True)}
    resource_type = 'AWS::AutoScaling::LifecycleHook'
class troposphere.autoscaling.LifecycleHookSpecification(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DefaultResult': (<type 'basestring'>, False), 'HeartbeatTimeout': (<type 'integer'>, True)}
class troposphere.autoscaling.Metadata(init, authentication=None)
    Bases: troposphere.AWSHelperFn
    validate(init, authentication)
class troposphere.autoscaling.MetricDimension(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Name': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}
class troposphere.autoscaling.MetricsCollection(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Granularity': (<type 'basestring'>, True), 'Metrics': (<type 'list'>, False)}
class troposphere.autoscaling.MixedInstancesPolicy(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'InstancesDistribution': (<class 'troposphere.autoscaling.InstancesDistribution'>, True)}
class troposphere.autoscaling.NotificationConfigurations(title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```

    props = {'NotificationTypes': (<type 'list'>, True), 'TopicARN': (<type 'basestring'>
class troposphere.autoscaling.PredefinedMetricSpecification (title=None,
                                                                **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PredefinedMetricType': (<type 'basestring'>, True), 'ResourceLabel': (<type
class troposphere.autoscaling.ScalingPolicy (title, template=None, validation=True,
                                              **kwargs)
    Bases: troposphere.AWSObject
    props = {'AdjustmentType': (<type 'basestring'>, False), 'AutoScalingGroupName': (<type
    resource_type = 'AWS::AutoScaling::ScalingPolicy'
class troposphere.autoscaling.ScheduledAction (title, template=None, validation=True,
                                               **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoScalingGroupName': (<type 'basestring'>, True), 'DesiredCapacity': (<type
    resource_type = 'AWS::AutoScaling::ScheduledAction'
class troposphere.autoscaling.StepAdjustments (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MetricIntervalLowerBound': (<function integer at 0x7f30f87f59b0>, False), 'I
class troposphere.autoscaling.Tag (key, value, propagate)
    Bases: troposphere.AWSHelperFn
class troposphere.autoscaling.Tags (**kwargs)
    Bases: troposphere.AWSHelperFn
    defaultPropagateAtLaunch = True
    manyType = [<type 'list'>, <type 'tuple'>]
    to_dict ()
class troposphere.autoscaling.TargetTrackingConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CustomizedMetricSpecification': (<class 'troposphere.autoscaling.Customized
class troposphere.autoscaling.Trigger (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoScalingGroupName': (<type 'basestring'>, True), 'BreachDuration': (<type
    resource_type = 'AWS::AutoScaling::Trigger'

```

troposphere.autoscalingplans module

```

class troposphere.autoscalingplans.ApplicationSource (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CloudFormationStackARN': (<type 'basestring'>, False), 'TagFilters': ([<type

```

```
class troposphere.autoscalingplans.CustomizedLoadMetricSpecification (title,  
                                                                    tem-  
                                                                    plate=None,  
                                                                    valida-  
                                                                    tion=True,  
                                                                    **kwargs)  
  
    Bases: troposphere.AWSObject  
  
    props = {'Dimensions': ([<class 'troposphere.autoscalingplans.MetricDimension'>], False)}  
class troposphere.autoscalingplans.CustomizedScalingMetricSpecification (title=None,  
                                                                    **kwargs)  
  
    Bases: troposphere.AWSPROPERTY  
  
    props = {'Dimensions': ([<class 'troposphere.autoscalingplans.MetricDimension'>], False)}  
class troposphere.autoscalingplans.MetricDimension (title=None, **kwargs)  
    Bases: troposphere.AWSPROPERTY  
  
    props = {'Name': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}  
class troposphere.autoscalingplans.PredefinedLoadMetricSpecification (title=None,  
                                                                    **kwargs)  
  
    Bases: troposphere.AWSPROPERTY  
  
    props = {'PredefinedLoadMetricType': (<type 'basestring'>, True), 'ResourceLabel': (<type 'basestring'>, True)}  
class troposphere.autoscalingplans.PredefinedScalingMetricSpecification (title=None,  
                                                                    **kwargs)  
  
    Bases: troposphere.AWSPROPERTY  
  
    props = {'PredefinedScalingMetricType': (<type 'basestring'>, True), 'ResourceLabel': (<type 'basestring'>, True)}  
class troposphere.autoscalingplans.ScalingInstruction (title=None, **kwargs)  
    Bases: troposphere.AWSPROPERTY  
  
    props = {'CustomizedLoadMetricSpecification': (<class 'troposphere.autoscalingplans.CustomizedLoadMetricSpecification'>),  
             'CustomizedScalingMetricSpecification': (<class 'troposphere.autoscalingplans.CustomizedScalingMetricSpecification'>),  
             'PredefinedLoadMetricSpecification': (<class 'troposphere.autoscalingplans.PredefinedLoadMetricSpecification'>),  
             'PredefinedScalingMetricSpecification': (<class 'troposphere.autoscalingplans.PredefinedScalingMetricSpecification'>)}  
class troposphere.autoscalingplans.ScalingPlan (title, template=None, validation=True,  
                                                                    **kwargs)  
  
    Bases: troposphere.AWSObject  
  
    props = {'ApplicationSource': (<class 'troposphere.autoscalingplans.ApplicationSource'>),  
             'ResourceLabel': (<type 'basestring'>, True),  
             'resource_type': 'AWS::AutoScalingPlans::ScalingPlan'}  
class troposphere.autoscalingplans.TagFilter (title=None, **kwargs)  
    Bases: troposphere.AWSPROPERTY  
  
    props = {'Key': (<type 'basestring'>, True), 'Values': ([<type 'basestring'>], False)}  
class troposphere.autoscalingplans.TargetTrackingConfiguration (title=None,  
                                                                    **kwargs)  
  
    Bases: troposphere.AWSPROPERTY  
  
    props = {'CustomizedScalingMetricSpecification': (<class 'troposphere.autoscalingplans.CustomizedScalingMetricSpecification'>),  
             'PredefinedScalingMetricSpecification': (<class 'troposphere.autoscalingplans.PredefinedScalingMetricSpecification'>)}  
  
troposphere.autoscalingplans.validate_predictivescalingmaxcapacitybehavior (predictivescalingmaxcapacitybehavior)  
    Validate PredictiveScalingMaxCapacityBehavior for ScalingInstruction  
  
troposphere.autoscalingplans.validate_predictivescalingmode (predictivescalingmode)  
    Validate PredictiveScalingMode for ScalingInstruction  
  
troposphere.autoscalingplans.validate_scalingpolicyupdatebehavior (scalingpolicyupdatebehavior)  
    Validate ScalingPolicyUpdateBehavior for ScalingInstruction
```


troposphere.awslambda module

```

class troposphere.awslambda.Alias (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'FunctionName': (<type 'basest
    resource_type = 'AWS::Lambda::Alias'

class troposphere.awslambda.AliasRoutingConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AdditionalVersionWeights': ([<class 'troposphere.awslambda.VersionWeight'>]

class troposphere.awslambda.Code (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    static check_zip_file (zip_file)
    props = {'S3Bucket': (<type 'basestring'>, False), 'S3Key': (<type 'basestring'>, Fa
    validate ()

class troposphere.awslambda.Content (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'S3Bucket': (<type 'basestring'>, True), 'S3Key': (<type 'basestring'>, Tru

class troposphere.awslambda.DeadLetterConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'TargetArn': (<type 'basestring'>, False)}

class troposphere.awslambda.Environment (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Variables': (<function validate_variables_name at 0x7f30f87c9488>, True)}

class troposphere.awslambda.EventSourceMapping (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'BatchSize': (<function positive_integer at 0x7f30f87f5a28>, False), 'Enabled
    resource_type = 'AWS::Lambda::EventSourceMapping'

class troposphere.awslambda.Function (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Code': (<class 'troposphere.awslambda.Code'>, True), 'DeadLetterConfig': (
    resource_type = 'AWS::Lambda::Function'

class troposphere.awslambda.LayerVersion (title, template=None, validation=True,
                                           **kwargs)
    Bases: troposphere.AWSObject
    props = {'CompatibleRuntimes': ([<type 'basestring'>], False), 'Content': (<class 't
    resource_type = 'AWS::Lambda::LayerVersion'

class troposphere.awslambda.LayerVersionPermission (title, template=None, valida
                                                    tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Action': (<type 'basestring'>, True), 'LayerVersionArn': (<type 'basestrin

```

```
    resource_type = 'AWS::Lambda::LayerVersionPermission'
class troposphere.awslambda.Permission(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Action': (<type 'basestring'>, True), 'EventSourceToken': (<type 'basestring'>, True)}
    resource_type = 'AWS::Lambda::Permission'
class troposphere.awslambda.TracingConfig(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Mode': (<type 'basestring'>, False)}
class troposphere.awslambda.VPCConfig(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'SecurityGroupIds': (<type 'list'>, True), 'SubnetIds': (<type 'list'>, True)}
class troposphere.awslambda.Version(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CodeSha256': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, True)}
    resource_type = 'AWS::Lambda::Version'
class troposphere.awslambda.VersionWeight(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'FunctionVersion': (<type 'basestring'>, True), 'FunctionWeight': (<type 'float'>, True)}
troposphere.awslambda.validate_memory_size(memory_value)
    Validate memory size for Lambda Function :param memory_value: The memory size specified in the Function
    :return: The provided memory size if it is valid
troposphere.awslambda.validate_variables_name(variables)
```

troposphere.batch module

```
class troposphere.batch.ComputeEnvironment(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ComputeEnvironmentName': (<type 'basestring'>, False), 'ComputeResources': troposphere.batch.ComputeResources}
    resource_type = 'AWS::Batch::ComputeEnvironment'
class troposphere.batch.ComputeEnvironmentOrder(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ComputeEnvironment': (<type 'basestring'>, True), 'Order': (<function positive_integer at 0x7f30f87f5a28>, True)}
class troposphere.batch.ComputeResources(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BidPercentage': (<function positive_integer at 0x7f30f87f5a28>, False), 'DefaultComputeEnvironment': (<type 'basestring'>, True)}
class troposphere.batch.ContainerProperties(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Command': ([<type 'basestring'>], False), 'Environment': ([<class 'troposphere.batch.Environment'>], True)}
class troposphere.batch.Environment(title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```

    props = {'Name': (<type 'basestring'>, False), 'Value': (<type 'basestring'>, False)}
class troposphere.batch.JobDefinition (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ContainerProperties': (<class 'troposphere.batch.ContainerProperties'>, True),
            'resource_type': 'AWS::Batch::JobDefinition'}
class troposphere.batch.JobQueue (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ComputeEnvironmentOrder': ([<class 'troposphere.batch.ComputeEnvironmentOrder'>], True),
            'resource_type': 'AWS::Batch::JobQueue'}
class troposphere.batch.LaunchTemplateSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'LaunchTemplateId': (<type 'basestring'>, False), 'LaunchTemplateName': (<type 'basestring'>, False)}
    validate()
class troposphere.batch.MountPoints (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ContainerPath': (<type 'basestring'>, False), 'ReadOnly': (<type 'bool'>, False)}
class troposphere.batch.RetryStrategy (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Attempts': (<function positive_integer at 0x7f30f87f5a28>, False)}
class troposphere.batch.Timeout (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AttemptDurationSeconds': (<function integer at 0x7f30f87f59b0>, False)}
class troposphere.batch.Ulimit (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'HardLimit': (<function positive_integer at 0x7f30f87f5a28>, True), 'Name': (<type 'basestring'>, False)}
class troposphere.batch.Volumes (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Host': (<class 'troposphere.batch.VolumesHost'>, False), 'Name': (<type 'basestring'>, False)}
class troposphere.batch.VolumesHost (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'SourcePath': (<type 'basestring'>, False)}
troposphere.batch.validate_environment_state (environment_state)
    Validate response type :param environment_state: State of the environment :return: The provided value if valid
troposphere.batch.validate_queue_state (queue_state)
    Validate response type :param queue_state: State of the queue :return: The provided value if valid

```

troposphere.budgets module

```

class troposphere.budgets.Budget (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Budget': (<class 'troposphere.budgets.BudgetData'>, True), 'NotificationsWithSubscribers': (<type 'list'>, False)}

```

```
    resource_type = 'AWS::Budgets::Budget'
class troposphere.budgets.BudgetData (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'BudgetLimit': (<class 'troposphere.budgets.Spend'>, False), 'BudgetName':
class troposphere.budgets.CostTypes (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'IncludeCredit': (<function boolean at 0x7f30f87f5938>, False), 'IncludeDisc
class troposphere.budgets.Notification (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'ComparisonOperator': (<type 'basestring'>, True), 'NotificationType': (<ty
class troposphere.budgets.NotificationWithSubscribers (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Notification': (<class 'troposphere.budgets.Notification'>, True), 'Subscri
class troposphere.budgets.Spend (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Amount': (<type 'float'>, True), 'Unit': (<type 'basestring'>, True)}
class troposphere.budgets.Subscriber (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Address': (<type 'basestring'>, True), 'SubscriptionType': (<type 'basestr
class troposphere.budgets.TimePeriod (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'End': (<type 'basestring'>, False), 'Start': (<type 'basestring'>, False)}
```

troposphere.certificatemanager module

```
class troposphere.certificatemanager.Certificate (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DomainName': (<type 'basestring'>, True), 'DomainValidationOptions': ([<cl
    resource_type = 'AWS::CertificateManager::Certificate'
class troposphere.certificatemanager.DomainValidationOption (title=None,
    **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DomainName': (<type 'basestring'>, True), 'ValidationDomain': (<type 'base
```

troposphere.cloud9 module

```
class troposphere.cloud9.EnvironmentEC2 (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutomaticStopTimeMinutes': (<function integer at 0x7f30f87f59b0>, False), '
    resource_type = 'AWS::Cloud9::EnvironmentEC2'
```

```

class troposphere.cloud9.Repository (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'PathComponent': (<type 'basestring'>, True), 'RepositoryUrl': (<type 'base

```

troposphere.cloudformation module

```

class troposphere.cloudformation.AWSCustomObject (title, template=None, valida-
                                                tion=True, **kwargs)
    Bases: troposphere.BaseAWSObject

    dictname = 'Properties'

class troposphere.cloudformation.Authentication (data)
    Bases: troposphere.AWSHelperFn

    validate (data)

class troposphere.cloudformation.AuthenticationBlock (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'accessKeyId': (<type 'basestring'>, False), 'buckets': ([<type 'basestring'

class troposphere.cloudformation.CustomResource (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.cloudformation.AWSCustomObject

    props = {'ServiceToken': (<type 'basestring'>, True)}

    resource_type = 'AWS::CloudFormation::CustomResource'

class troposphere.cloudformation.Init (data, **kwargs)
    Bases: troposphere.AWSHelperFn

    validate (data, config_sets)

class troposphere.cloudformation.InitConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'commands': (<type 'dict'>, False), 'files': (<type 'dict'>, False), 'group

class troposphere.cloudformation.InitConfigSets (**kwargs)
    Bases: troposphere.AWSHelperFn

    validate (config_sets)

class troposphere.cloudformation.InitFile (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'authentication': (<type 'basestring'>, False), 'content': (<type 'basetri

class troposphere.cloudformation.InitFileContext (data)
    Bases: troposphere.AWSHelperFn

class troposphere.cloudformation.InitFiles (data)
    Bases: troposphere.AWSHelperFn

    validate (data)

class troposphere.cloudformation.InitService (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'commands': (<type 'list'>, False), 'enabled': (<function boolean at 0x7f30

```

```
class troposphere.cloudformation.InitServices(data)
    Bases: troposphere.AWSHelperFn

    validate(data)

class troposphere.cloudformation.Macro(title, template=None, validation=True, **kwargs)
    Bases: troposphere.cloudformation.AWSCustomObject

    props = {'Description': (<type 'basestring'>, False), 'FunctionName': (<type 'basest...
    resource_type = 'AWS::CloudFormation::Macro'

class troposphere.cloudformation.Metadata(*args)
    Bases: troposphere.AWSHelperFn

    to_dict()

class troposphere.cloudformation.Stack(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'NotificationARNs': ([<type 'basestring'>], False), 'Parameters': (<type 'd...
    resource_type = 'AWS::CloudFormation::Stack'

class troposphere.cloudformation.WaitCondition(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Count': (<function integer at 0x7f30f87f59b0>, False), 'Handle': (<type 'b...
    resource_type = 'AWS::CloudFormation::WaitCondition'

    validate()

class troposphere.cloudformation.WaitConditionHandle(title, template=None, valida-
tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {}

    resource_type = 'AWS::CloudFormation::WaitConditionHandle'

troposphere.cloudformation.validate_authentication_type(auth_type)
```

troposphere.cloudfront module

```
class troposphere.cloudfront.CacheBehavior(title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AllowedMethods': ([<type 'basestring'>], False), 'CachedMethods': ([<type...

class troposphere.cloudfront.CloudFrontOriginAccessIdentity(title, template=None, val-
idation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'CloudFrontOriginAccessIdentityConfig': (<class 'troposphere.cloudfront.Clou...
    resource_type = 'AWS::CloudFront::CloudFrontOriginAccessIdentity'

class troposphere.cloudfront.CloudFrontOriginAccessIdentityConfig(title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Comment': (<type 'basestring'>, True)}
```

```

class troposphere.cloudfront.Cookies (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Forward': (<function cloudfront_forward_type at 0x7f30f87fbc80>, True), 'Wh

class troposphere.cloudfront.CustomErrorResponse (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'ErrorCachingMinTTL': (<function positive_integer at 0x7f30f87f5a28>, False),

class troposphere.cloudfront.CustomOriginConfig (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'HTTPPort': (<function network_port at 0x7f30f87f5cf8>, False), 'HTTPSPort':

class troposphere.cloudfront.DefaultCacheBehavior (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'AllowedMethods': ([<type 'basestring'>], False), 'CachedMethods': ([<type

class troposphere.cloudfront.Distribution (title, template=None, validation=True,
                                           **kwargs)
    Bases: troposphere.AWSObject

    props = {'DistributionConfig': (<class 'troposphere.cloudfront.DistributionConfig'>,
    resource_type = 'AWS::CloudFront::Distribution'

class troposphere.cloudfront.DistributionConfig (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Aliases': (<type 'list'>, False), 'CacheBehaviors': ([<class 'troposphere.

class troposphere.cloudfront.ForwardedValues (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Cookies': (<class 'troposphere.cloudfront.Cookies'>, False), 'Headers': ([

class troposphere.cloudfront.GeoRestriction (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Locations': ([<type 'basestring'>], False), 'RestrictionType': (<function

class troposphere.cloudfront.LambdaFunctionAssociation (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'EventType': (<function cloudfront_event_type at 0x7f30f87fbb18>, False), 'L

class troposphere.cloudfront.Logging (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Bucket': (<type 'basestring'>, True), 'IncludeCookies': (<function boolean

class troposphere.cloudfront.Origin (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'CustomOriginConfig': (<class 'troposphere.cloudfront.CustomOriginConfig'>,

class troposphere.cloudfront.OriginCustomHeader (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'HeaderName': (<type 'basestring'>, True), 'HeaderValue': (<type 'basestrin

class troposphere.cloudfront.Restrictions (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'GeoRestriction': (<class 'troposphere.cloudfront.GeoRestriction'>, True)}

```

```
class troposphere.cloudfront.S3Origin (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DomainName': (<type 'basestring'>, True), 'OriginAccessIdentity': (<type 'basestring'>, True)}

class troposphere.cloudfront.S3OriginConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'OriginAccessIdentity': (<type 'basestring'>, False)}

class troposphere.cloudfront.StreamingDistribution (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'StreamingDistributionConfig': (<class 'troposphere.cloudfront.StreamingDistributionConfig'>, True)}
    resource_type = 'AWS::CloudFront::StreamingDistribution'

class troposphere.cloudfront.StreamingDistributionConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Aliases': ([<type 'basestring'>], False), 'Comment': (<type 'basestring'>, True), 'Enabled': (True, True)}

class troposphere.cloudfront.TrustedSigners (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AwsAccountNumbers': ([<type 'basestring'>], False), 'Enabled': (True, True)}

class troposphere.cloudfront.ViewerCertificate (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AcmCertificateArn': (<type 'basestring'>, False), 'CloudFrontDefaultCertificate': (True, True)}
```

troposphere.cloudtrail module

```
class troposphere.cloudtrail.DataResource (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Type': (<type 'basestring'>, True), 'Values': ([<type 'basestring'>], False)}

class troposphere.cloudtrail.EventSelector (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DataResources': ([<class 'troposphere.cloudtrail.DataResource'>], False), 'Enabled': (True, True)}

class troposphere.cloudtrail.Trail (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'CloudWatchLogsLogGroupArn': (<type 'basestring'>, False), 'CloudWatchLogsRoleArn': (<type 'basestring'>, False)}
    resource_type = 'AWS::CloudTrail::Trail'
```

troposphere.cloudwatch module

```
class troposphere.cloudwatch.Alarm (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ActionsEnabled': (<function boolean at 0x7f30f87f5938>, False), 'AlarmAction': (<type 'basestring'>, True)}
    resource_type = 'AWS::CloudWatch::Alarm'

    validate()
```



```

class troposphere.cloudwatch.Dashboard (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DashboardBody': ((<type 'basestring'>, <type 'dict'>), True), 'DashboardName': (
    resource_type = 'AWS::CloudWatch::Dashboard'

    validate()

class troposphere.cloudwatch.Metric (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Dimensions': ([<class 'troposphere.cloudwatch.MetricDimension'>], False), 'Id': (

class troposphere.cloudwatch.MetricDataQuery (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Expression': (<type 'basestring'>, False), 'Id': (<type 'basestring'>, True)

class troposphere.cloudwatch.MetricDimension (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}

class troposphere.cloudwatch.MetricStat (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Metric': (<class 'troposphere.cloudwatch.Metric'>, True), 'Period': (<func

troposphere.cloudwatch.validate_unit (unit)
    Validate Units

```

troposphere.codebuild module

```

class troposphere.codebuild.Artifacts (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ArtifactIdentifier': (<type 'basestring'>, False), 'EncryptionDisabled': (

    validate()

class troposphere.codebuild.CloudWatchLogs (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'GroupName': (<type 'basestring'>, False), 'Status': (<function validate_st

class troposphere.codebuild.Environment (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Certificate': (<type 'basestring'>, False), 'ComputeType': (<type 'basestr

    validate()

class troposphere.codebuild.EnvironmentVariable (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, True), 'Type': (<type 'basestring'>, False),

    validate()

class troposphere.codebuild.LogsConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CloudWatchLogs': (<class 'troposphere.codebuild.CloudWatchLogs'>, False), '

```

```
class troposphere.codebuild.Project (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Artifacts': (<class 'troposphere.codebuild.Artifacts'>, True), 'BadgeEnabled': (
    resource_type = 'AWS::CodeBuild::Project'

class troposphere.codebuild.ProjectCache (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Location': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, True),
    validate ()

class troposphere.codebuild.ProjectTriggers (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Webhook': (<function boolean at 0x7f30f87f5938>, False)}

class troposphere.codebuild.RegistryCredential (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Credential': (<type 'basestring'>, True), 'CredentialProvider': (<function
    validate ()

class troposphere.codebuild.S3Logs (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Location': (<type 'basestring'>, False), 'Status': (<function validate_status
    validate ()

class troposphere.codebuild.Source (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Auth': (<class 'troposphere.codebuild.SourceAuth'>, False), 'BuildSpec': (<
    validate ()

class troposphere.codebuild.SourceAuth (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Resource': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, True),
    validate ()

class troposphere.codebuild.VpcConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'SecurityGroupIds': ([<type 'basestring'>], True), 'Subnets': ([<type 'basestring'>], True)}

troposphere.codebuild.validate_credentials_provider (credential_provider)
    Validate CredentialProvider for Project's RegistryCredential

troposphere.codebuild.validate_image_pull_credentials (image_pull_credentials)
    Validate ImagePullCredentialsType for Project

troposphere.codebuild.validate_status (status)
    Validate status :param status: The Status of CloudWatchLogs or S3Logs :return: The provided value if valid
```

troposphere.codecommit module

```
class troposphere.codecommit.Repository (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'RepositoryDescription': (<type 'basestring'>, False), 'RepositoryName': (<
    resource_type = 'AWS::CodeCommit::Repository'
```

```

class troposphere.codecommit.Trigger (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Branches': ([<type 'basestring'>], False), 'CustomData': (<type 'basestring'>, False)}
    validate()

```

troposphere.codedeploy module

```

class troposphere.codedeploy.Alarm (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Name': (<type 'basestring'>, False)}

class troposphere.codedeploy.AlarmConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Alarms': ([<class 'troposphere.codedeploy.Alarm'>], False), 'Enabled': (<type 'bool'>, False)}

class troposphere.codedeploy.Application (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApplicationName': (<type 'basestring'>, False), 'ComputePlatform': (<type 'basestring'>, False)}
    resource_type = 'AWS::CodeDeploy::Application'

class troposphere.codedeploy.AutoRollbackConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Enabled': (<type 'bool'>, False), 'Events': ([<type 'basestring'>], False)}

class troposphere.codedeploy.Deployment (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Description': (<type 'basestring'>, False), 'IgnoreApplicationStopFailures': (<type 'bool'>, False)}

class troposphere.codedeploy.DeploymentConfig (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DeploymentConfigName': (<type 'basestring'>, False), 'MinimumHealthyHosts': (<type 'int'>, 0)}
    resource_type = 'AWS::CodeDeploy::DeploymentConfig'

class troposphere.codedeploy.DeploymentGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AlarmConfiguration': (<class 'troposphere.codedeploy.AlarmConfiguration'>, False), 'ApplicationName': (<type 'basestring'>, False)}
    resource_type = 'AWS::CodeDeploy::DeploymentGroup'
    validate()

class troposphere.codedeploy.DeploymentStyle (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DeploymentOption': (<function deployment_option_validator at 0x7f30f7c80320'>, False)}

class troposphere.codedeploy.Ec2TagFilters (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Key': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, False)}

```

```
class troposphere.codedeploy.Ec2TagSet (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Ec2TagSetList': ([<class 'troposphere.codedeploy.Ec2TagSetListObject'>], Fa

class troposphere.codedeploy.Ec2TagSetListObject (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Ec2TagGroup': ([<class 'troposphere.codedeploy.Ec2TagFilters'>], False)}

class troposphere.codedeploy.ElbInfoList (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, False)}

class troposphere.codedeploy.GitHubLocation (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CommitId': (<type 'basestring'>, True), 'Repository': (<type 'basestring'>

class troposphere.codedeploy.LoadBalancerInfo (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ElbInfoList': ([<class 'troposphere.codedeploy.ElbInfoList'>], False), 'Tar

    validate()

class troposphere.codedeploy.MinimumHealthyHosts (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Type': (<type 'basestring'>, False), 'Value': (<function positive_integer

class troposphere.codedeploy.OnPremisesInstanceTagFilters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Key': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, False),

class troposphere.codedeploy.OnPremisesTagSet (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'OnPremisesTagSetList': (<class 'troposphere.codedeploy.OnPremisesTagSetList

class troposphere.codedeploy.OnPremisesTagSetList (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'OnPremisesTagSetList': ([<class 'troposphere.codedeploy.OnPremisesTagSetObj

class troposphere.codedeploy.OnPremisesTagSetObject (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'OnPremisesTagGroup': ([<class 'troposphere.codedeploy.TagFilters'>], False)

class troposphere.codedeploy.Revision (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'GitHubLocation': (<class 'troposphere.codedeploy.GitHubLocation'>, False),

class troposphere.codedeploy.S3Location (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Bucket': (<type 'basestring'>, True), 'BundleType': (<type 'basestring'>,

class troposphere.codedeploy.TagFilters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Key': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, False),
```

```
class troposphere.codedeploy.TargetGroupInfoList (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'Name': (<type 'basestring'>, False)}
```

```
class troposphere.codedeploy.TriggerConfig (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'TriggerEvents': ([<type 'basestring'>], False), 'TriggerName': (<type 'bas
```

```
troposphere.codedeploy.deployment_option_validator(x)
```

```
troposphere.codedeploy.deployment_type_validator(x)
```

troposphere.codepipeline module

```
class troposphere.codepipeline.ActionTypeId (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'Category': (<type 'basestring'>, True), 'Owner': (<type 'basestring'>, True)
```

```
class troposphere.codepipeline.Actions (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'ActionTypeId': (<class 'troposphere.codepipeline.ActionTypeId'>, True), 'Co
```

```
class troposphere.codepipeline.ArtifactDetails (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'MaximumCount': (<function integer at 0x7f30f87f59b0>, True), 'MinimumCount'
```

```
class troposphere.codepipeline.ArtifactStore (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'EncryptionKey': (<class 'troposphere.codepipeline.EncryptionKey'>, False),
```

```
class troposphere.codepipeline.ArtifactStoreMap (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'ArtifactStore': (<class 'troposphere.codepipeline.ArtifactStore'>, True), 'I
```

```
class troposphere.codepipeline.Blockers (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'Name': (<type 'basestring'>, True), 'Type': (<type 'basestring'>, True)}
```

```
class troposphere.codepipeline.ConfigurationProperties (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'Description': (<type 'basestring'>, False), 'Key': (<function boolean at 0
```

```
class troposphere.codepipeline.CustomActionType (title, template=None, validation=True,
                                                  **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Category': (<type 'basestring'>, True), 'ConfigurationProperties': ([<clas
```

```
    resource_type = 'AWS::CodePipeline::CustomActionType'
```

```
class troposphere.codepipeline.DisableInboundStageTransitions (title=None,
                                                                **kwargs)
```

```
    Bases: troposphere.AWSPROPERTY
```

```
    props = {'Reason': (<type 'basestring'>, True), 'StageName': (<type 'basestring'>, T
```

```
class troposphere.codepipeline.EncryptionKey (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Id': (<type 'basestring'>, True), 'Type': (<type 'basestring'>, True)}

class troposphere.codepipeline.InputArtifacts (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, True)}

class troposphere.codepipeline.OutputArtifacts (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, True)}

class troposphere.codepipeline.Pipeline (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ArtifactStore': (<class 'troposphere.codepipeline.ArtifactStore'>, False),
            'resource_type' = 'AWS::CodePipeline::Pipeline'

class troposphere.codepipeline.Settings (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'EntityUrlTemplate': (<type 'basestring'>, False), 'ExecutionUrlTemplate':

class troposphere.codepipeline.Stages (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Actions': ([<class 'troposphere.codepipeline.Actions'>], True), 'Blockers':

class troposphere.codepipeline.Webhook (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Authentication': (<type 'basestring'>, True), 'AuthenticationConfiguration'
            'resource_type' = 'AWS::CodePipeline::Webhook'

class troposphere.codepipeline.WebhookAuthConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AllowedIPRange': (<type 'basestring'>, False), 'SecretToken': (<type 'base

class troposphere.codepipeline.WebhookFilterRule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'JsonPath': (<type 'basestring'>, True), 'MatchEquals': (<type 'basestring'
```

troposphere.cognito module

```
class troposphere.cognito.AdminCreateUserConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AllowAdminCreateUserOnly': (<function boolean at 0x7f30f87f5938>, False),

class troposphere.cognito.AttributeType (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, False)}

class troposphere.cognito.CognitoIdentityProvider (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ClientId': (<type 'basestring'>, False), 'ProviderName': (<type 'basestring'
```

```

class troposphere.cognito.CognitoStreams (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'RoleArn': (<type 'basestring'>, False), 'StreamName': (<type 'basestring'>

class troposphere.cognito.DeviceConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'ChallengeRequiredOnNewDevice': (<function boolean at 0x7f30f87f5938>, False)

class troposphere.cognito.EmailConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'ReplyToEmailAddress': (<type 'basestring'>, False), 'SourceArn': (<type 'b

class troposphere.cognito.IdentityPool (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AllowUnauthenticatedIdentities': (<type 'bool'>, True), 'CognitoEvents': (
    resource_type = 'AWS::Cognito::IdentityPool'

class troposphere.cognito.IdentityPoolRoleAttachment (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'IdentityPoolId': (<type 'basestring'>, True), 'RoleMappings': (<type 'dict
    resource_type = 'AWS::Cognito::IdentityPoolRoleAttachment'

class troposphere.cognito.InviteMessageTemplate (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'EmailMessage': (<type 'basestring'>, False), 'EmailSubject': (<type 'bases

class troposphere.cognito.LambdaConfig (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'CreateAuthChallenge': (<type 'basestring'>, False), 'CustomMessage': (<type

class troposphere.cognito.MappingRule (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Claim': (<type 'basestring'>, True), 'MatchType': (<type 'basestring'>, Tr

class troposphere.cognito.NumberAttributeConstraints (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'MaxValue': (<type 'basestring'>, False), 'MinValue': (<type 'basestring'>,

class troposphere.cognito.PasswordPolicy (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'MinimumLength': (<function positive_integer at 0x7f30f87f5a28>, False), 'Re

class troposphere.cognito.Policies (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'PasswordPolicy': (<class 'troposphere.cognito.PasswordPolicy'>, False)}

class troposphere.cognito.PushSync (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'ApplicationArns': ([<type 'basestring'>], False), 'RoleArn': (<type 'bases

class troposphere.cognito.RoleMapping (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

```

```

    props = {'AmbiguousRoleResolution': (<type 'basestring'>, False), 'RulesConfiguration': (<type 'basestring'>, False)}
class troposphere.cognito.RulesConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Rules': ([[<class 'troposphere.cognito.MappingRule'>]], True)}
class troposphere.cognito.SchemaAttribute (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AttributeDataType': (<type 'basestring'>, False), 'DeveloperOnlyAttribute': (<type 'basestring'>, False)}
class troposphere.cognito.SmsConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ExternalId': (<type 'basestring'>, False), 'SnsCallerArn': (<type 'basestring'>, False)}
class troposphere.cognito.StringAttributeConstraints (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MaxLength': (<type 'basestring'>, False), 'MinLength': (<type 'basestring'>, False)}
class troposphere.cognito.UserPool (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AdminCreateUserConfig': (<class 'troposphere.cognito.AdminCreateUserConfig'>, False)}
    resource_type = 'AWS::Cognito::UserPool'
class troposphere.cognito.UserPoolClient (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ClientName': (<type 'basestring'>, False), 'ExplicitAuthFlows': ([[<type 'basestring'>]], False)}
    resource_type = 'AWS::Cognito::UserPoolClient'
class troposphere.cognito.UserPoolGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'GroupName': (<type 'basestring'>, False)}
    resource_type = 'AWS::Cognito::UserPoolGroup'
class troposphere.cognito.UserPoolUser (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DesiredDeliveryMediums': ([[<type 'basestring'>]], False), 'ForceAliasCreation': (<type 'boolean'>, False)}
    resource_type = 'AWS::Cognito::UserPoolUser'
class troposphere.cognito.UserPoolUserToGroupAttachment (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'GroupName': (<type 'basestring'>, True), 'UserPoolId': (<type 'basestring'>, True)}
    resource_type = 'AWS::Cognito::UserPoolUserToGroupAttachment'

```

troposphere.config module

```

class troposphere.config.AccountAggregationSources (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AccountIds': ([[<type 'basestring'>]], True), 'AllAwsRegions': (<function bo

```



```

class troposphere.config.AggregationAuthorization(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AuthorizedAccountId': (<type 'basestring'>, True), 'AuthorizedAwsRegion':
    resource_type = 'AWS::Config::AggregationAuthorization'

class troposphere.config.ConfigRule(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ConfigRuleName': (<type 'basestring'>, False), 'Description': (<type 'base
    resource_type = 'AWS::Config::ConfigRule'

class troposphere.config.ConfigSnapshotDeliveryProperties(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DeliveryFrequency': (<type 'basestring'>, False)}

class troposphere.config.ConfigurationAggregator(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AccountAggregationSources': ([<class 'troposphere.config.AccountAggregation
    resource_type = 'AWS::Config::ConfigurationAggregator'

class troposphere.config.ConfigurationRecorder(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, False), 'RecordingGroup': (<class 'tropospher
    resource_type = 'AWS::Config::ConfigurationRecorder'

class troposphere.config.DeliveryChannel(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ConfigSnapshotDeliveryProperties': (<class 'troposphere.config.ConfigSnapsh
    resource_type = 'AWS::Config::DeliveryChannel'

class troposphere.config.OrganizationAggregationSource(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AllAwsRegions': (<function boolean at 0x7f30f87f5938>, False), 'AwsRegions'

class troposphere.config.RecordingGroup(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AllSupported': (<function boolean at 0x7f30f87f5938>, False), 'IncludeGloba

class troposphere.config.Scope(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ComplianceResourceId': (<type 'basestring'>, False), 'ComplianceResourceTyp

class troposphere.config.Source(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Owner': (<type 'basestring'>, True), 'SourceDetails': ([<class 'tropospher

class troposphere.config.SourceDetails(title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'EventSource': (<type 'basestring'>, True), 'MaximumExecutionFrequency': (<

```

```
validate()
```

troposphere.constants module

troposphere.datapipeline module

```
class troposphere.datapipeline.ObjectField(title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Key': (<type 'basestring'>, True), 'RefValue': (<type 'basestring'>, False)}
```

```
class troposphere.datapipeline.ParameterObject(title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Attributes': ([<class 'troposphere.datapipeline.ParameterObjectAttribute'>]}
```

```
class troposphere.datapipeline.ParameterObjectAttribute(title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Key': (<type 'basestring'>, True), 'StringValue': (<type 'basestring'>, False)}
```

```
class troposphere.datapipeline.ParameterValue(title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Id': (<type 'basestring'>, True), 'StringValue': (<type 'basestring'>, True)}
```

```
class troposphere.datapipeline.Pipeline(title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Activate': (<function boolean at 0x7f30f87f5938>, False), 'Description': (None)}
```

```
    resource_type = 'AWS::DataPipeline::Pipeline'
```

```
class troposphere.datapipeline.PipelineObject(title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Fields': ([<class 'troposphere.datapipeline.ObjectField'>], True), 'Id': (None)}
```

```
class troposphere.datapipeline.PipelineTag(title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Key': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}
```

troposphere.dax module

```
class troposphere.dax.Cluster(title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'AvailabilityZones': (<type 'basestring'>, False), 'ClusterName': (<type 'basestring'>, True)}
```

```
    resource_type = 'AWS::DAX::Cluster'
```

```
class troposphere.dax.ParameterGroup(title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Description': (<type 'basestring'>, False), 'ParameterGroupName': (<type 'basestring'>, True)}
```

```
    resource_type = 'AWS::DAX::ParameterGroup'
```

```
class troposphere.dax.SSESpecification(title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'SSEEnabled': (<function boolean at 0x7f30f87f5938>, False)}
```

```

class troposphere.dax.SubnetGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'SubnetGroupName': (<type 'bas
    resource_type = 'AWS::DAX::SubnetGroup'

```

troposphere.directoryservice module

```

class troposphere.directoryservice.MicrosoftAD (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject

    props = {'CreateAlias': (<function boolean at 0x7f30f87f5938>, False), 'Edition': (<
    resource_type = 'AWS::DirectoryService::MicrosoftAD'

```

```

class troposphere.directoryservice.SimpleAD (title, template=None, validation=True,
                                              **kwargs)
    Bases: troposphere.AWSObject

    props = {'CreateAlias': (<function boolean at 0x7f30f87f5938>, False), 'Description':
    resource_type = 'AWS::DirectoryService::SimpleAD'

```

```

class troposphere.directoryservice.VpcSettings (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'SubnetIds': ([<type 'basestring'>], True), 'VpcId': (<type 'basestring'>,

```

troposphere.dlm module

```

class troposphere.dlm.CreateRule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Interval': (<function validate_interval at 0x7f30f7acc8c0>, True), 'Interva

```

```

class troposphere.dlm.LifecyclePolicy (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'ExecutionRoleArn': (<type 'ba
    resource_type = 'AWS::DLM::LifecyclePolicy'

```

```

class troposphere.dlm.PolicyDetails (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceTypes': ([<type 'basestring'>], False), 'Schedules': ([<class 'trop

```

```

class troposphere.dlm.RetainRule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Count': (<function integer at 0x7f30f87f59b0>, True)}

```

```

class troposphere.dlm.Schedule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CopyTags': (<function boolean at 0x7f30f87f5938>, False), 'CreateRule': (<

```

```

troposphere.dlm.validate_interval (interval)
    Interval validation rule.

```

`troposphere.dlm.validate_interval_unit` (*interval_unit*)
Interval unit validation rule.

`troposphere.dlm.validate_state` (*state*)
State validation rule.

troposphere.dms module

class `troposphere.dms.Certificate` (*title*, *template=None*, *validation=True*, ***kwargs*)
Bases: `troposphere.AWSObject`

props = {'CertificateIdentifier': (<type 'basestring'>, False), 'CertificatePem': (<type 'basestring'>, False), 'CertificateType': (<type 'basestring'>, False), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::Certificate')}

class `troposphere.dms.DynamoDBSettings` (*title=None*, ***kwargs*)
Bases: `troposphere.AWSProperty`

props = {'ServiceAccessRoleArn': (<type 'basestring'>, True)}

class `troposphere.dms.Endpoint` (*title*, *template=None*, *validation=True*, ***kwargs*)
Bases: `troposphere.AWSObject`

props = {'CertificateArn': (<type 'basestring'>, False), 'DatabaseName': (<type 'basestring'>, True), 'EndpointIdentifier': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::Endpoint')}

class `troposphere.dms.EventSubscription` (*title*, *template=None*, *validation=True*, ***kwargs*)
Bases: `troposphere.AWSObject`

props = {'Enabled': (<function boolean at 0x7f30f87f5938>, False), 'EventCategories': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::EventSubscription')}

class `troposphere.dms.MongoDbSettings` (*title=None*, ***kwargs*)
Bases: `troposphere.AWSProperty`

props = {'AuthMechanism': (<type 'basestring'>, False), 'AuthSource': (<type 'basestring'>, True), 'DatabaseName': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::MongoDBSettings')}

class `troposphere.dms.ReplicationInstance` (*title*, *template=None*, *validation=True*, ***kwargs*)
Bases: `troposphere.AWSObject`

props = {'AllocatedStorage': (<function integer at 0x7f30f87f59b0>, False), 'AutoMinorVersionUpgrade': (<type 'boolean'>, True), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::ReplicationInstance')}

class `troposphere.dms.ReplicationSubnetGroup` (*title*, *template=None*, *validation=True*, ***kwargs*)
Bases: `troposphere.AWSObject`

props = {'ReplicationSubnetGroupDescription': (<type 'basestring'>, True), 'ReplicationSubnetGroupIdentifier': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::ReplicationSubnetGroup')}

class `troposphere.dms.ReplicationTask` (*title*, *template=None*, *validation=True*, ***kwargs*)
Bases: `troposphere.AWSObject`

props = {'CdcStartTime': (<function positive_integer at 0x7f30f87f5a28>, False), 'MigrationType': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::ReplicationTask')}

class `troposphere.dms.S3Settings` (*title=None*, ***kwargs*)
Bases: `troposphere.AWSProperty`

props = {'BucketFolder': (<type 'basestring'>, False), 'BucketName': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, 'AWS::DMS::S3Settings')}

troposphere.docdb module

```

class troposphere.docdb.DBCluster (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AvailabilityZones': ([<type 'basestring'>], False), 'BackupRetentionPeriod':
    resource_type = 'AWS::DocDB::DBCluster'

class troposphere.docdb.DBClusterParameterGroup (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, True), 'Family': (<type 'basestring'>,
    resource_type = 'AWS::DocDB::DBClusterParameterGroup'

class troposphere.docdb.DBInstance (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AutoMinorVersionUpgrade': (<function boolean at 0x7f30f87f5938>, False), 'A
    resource_type = 'AWS::DocDB::DBInstance'

class troposphere.docdb.DBSubnetGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DBSubnetGroupDescription': (<type 'basestring'>, True), 'DBSubnetGroupName'
    resource_type = 'AWS::DocDB::DBSubnetGroup'

```

troposphere.dynamodb module

```

class troposphere.dynamodb.AttributeDefinition (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AttributeName': (<type 'basestring'>, True), 'AttributeType': (<function a

class troposphere.dynamodb.GlobalSecondaryIndex (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'IndexName': (<type 'basestring'>, True), 'KeySchema': ([<class 'tropospher

class troposphere.dynamodb.Key (title=None, **kwargs)
    Bases: troposphere.dynamodb.KeySchema

    For backwards compatibility.

class troposphere.dynamodb.KeySchema (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AttributeName': (<type 'basestring'>, True), 'KeyType': (<function key_typ

class troposphere.dynamodb.LocalSecondaryIndex (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'IndexName': (<type 'basestring'>, True), 'KeySchema': ([<class 'tropospher

class troposphere.dynamodb.PointInTimeRecoverySpecification (title=None,
    **kwargs)
    Bases: troposphere.AWSProperty

    props = {'PointInTimeRecoveryEnabled': (<function boolean at 0x7f30f87f5938>, False)}

```

```
class troposphere.dynamodb.Projection (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'NonKeyAttributes': ([<type 'basestring'>], False), 'ProjectionType': (<fun

class troposphere.dynamodb.ProvisionedThroughput (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ReadCapacityUnits': (<type 'int'>, True), 'WriteCapacityUnits': (<type 'in

class troposphere.dynamodb.SSESpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'SSEEnabled': (<function boolean at 0x7f30f87f5938>, True)}

class troposphere.dynamodb.StreamSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'StreamViewType': (<type 'basestring'>, True)}

class troposphere.dynamodb.Table (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AttributeDefinitions': ([<class 'troposphere.dynamodb.AttributeDefinition'>
    resource_type = 'AWS::DynamoDB::Table'

    validate()

class troposphere.dynamodb.TimeToLiveSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AttributeName': (<type 'basestring'>, True), 'Enabled': (<function boolean

troposphere.dynamodb.attribute_type_validator(x)
troposphere.dynamodb.billing_mode_validator(x)
troposphere.dynamodb.key_type_validator(x)
troposphere.dynamodb.projection_type_validator(x)
```

troposphere.dynamodb2 module

troposphere.ec2 module

```
class troposphere.ec2.AssociationParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Key': (<type 'basestring'>, True), 'Value': ([<type 'basestring'>], True)}

class troposphere.ec2.BlockDeviceMapping (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DeviceName': (<type 'basestring'>, True), 'Ebs': (<class 'troposphere.ec2.

class troposphere.ec2.ClassicLoadBalancer (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, True)}

class troposphere.ec2.ClassicLoadBalancersConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ClassicLoadBalancers': ([<class 'troposphere.ec2.ClassicLoadBalancer'>], Tr
```

```

class troposphere.ec2.CreditSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CPUCredits': (<type 'basestring'>, False)}

class troposphere.ec2.CustomerGateway (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'BgpAsn': (<function integer at 0x7f30f87f59b0>, True), 'IpAddress': (<type
    resource_type = 'AWS::EC2::CustomerGateway'

class troposphere.ec2.DHCPOptions (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DomainName': (<type 'basestring'>, False), 'DomainNameServers': (<type 'li
    resource_type = 'AWS::EC2::DHCPOptions'

class troposphere.ec2.EBSBlockDevice (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DeleteOnTermination': (<function boolean at 0x7f30f87f5938>, False), 'Encryp

class troposphere.ec2.EC2Fleet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ExcessCapacityTerminationPolicy': (<type 'basestring'>, False), 'LaunchTemp
    resource_type = 'AWS::EC2::EC2Fleet'

class troposphere.ec2.EIP (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Domain': (<type 'basestring'>, False), 'InstanceId': (<type 'basestring'>,
    resource_type = 'AWS::EC2::EIP'

class troposphere.ec2.EIPAssociation (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AllocationId': (<type 'basestring'>, False), 'EIP': (<type 'basestring'>, F
    resource_type = 'AWS::EC2::EIPAssociation'

class troposphere.ec2.EgressOnlyInternetGateway (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject

    props = {'VpcId': (<type 'basestring'>, True)}
    resource_type = 'AWS::EC2::EgressOnlyInternetGateway'

class troposphere.ec2.ElasticGpuSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Type': (<type 'basestring'>, True)}

class troposphere.ec2.ElasticInferenceAccelerator (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Type': (<function validate_elasticinferenceaccelerator_type at 0x7f30f876c4

class troposphere.ec2.FleetLaunchTemplateConfigRequest (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'LaunchTemplateSpecification': (<class 'troposphere.ec2.FleetLaunchTemplateS

```

```
class troposphere.ec2.FleetLaunchTemplateOverridesRequest (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AvailabilityZone': (<type 'basestring'>, False), 'InstanceType': (<type 'basestring'>, False)}

class troposphere.ec2.FleetLaunchTemplateSpecificationRequest (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'LaunchTemplateId': (<type 'basestring'>, False), 'LaunchTemplateName': (<type 'basestring'>, False)}

class troposphere.ec2.FlowLog (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DeliverLogsPermissionArn': (<type 'basestring'>, False), 'LogDestination': (<type 'basestring'>, False), 'ResourceId': (<type 'basestring'>, False)}
    resource_type = 'AWS::EC2::FlowLog'

class troposphere.ec2.Host (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoPlacement': (<type 'basestring'>, False), 'AvailabilityZone': (<type 'basestring'>, False)}
    resource_type = 'AWS::EC2::Host'

class troposphere.ec2.ICMP (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Code': (<function integer at 0x7f30f87f59b0>, False), 'Type': (<function integer at 0x7f30f87f59b0>, False)}

class troposphere.ec2.IamInstanceProfile (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Arn': (<type 'basestring'>, False)}

class troposphere.ec2.Instance (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Affinity': (<type 'basestring'>, False), 'AvailabilityZone': (<type 'basestring'>, False), 'SubnetId': (<type 'basestring'>, False)}
    resource_type = 'AWS::EC2::Instance'

class troposphere.ec2.InstanceMarketOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MarketType': (<type 'basestring'>, False), 'SpotOptions': (<class 'troposphere.ec2.InstanceMarketOptions.SpotOptions'>, False)}

class troposphere.ec2.InternetGateway (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Tags': ((<class 'troposphere.Tags'>, <type 'list'>), False)}
    resource_type = 'AWS::EC2::InternetGateway'

class troposphere.ec2.Ipv6Addresses (address)
    Bases: troposphere.AWSHelperFn

class troposphere.ec2.LaunchSpecifications (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BlockDeviceMappings': ([<class 'troposphere.ec2.BlockDeviceMapping'>], False)}

class troposphere.ec2.LaunchTemplate (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'LaunchTemplateData': (<class 'troposphere.ec2.LaunchTemplateData'>, False), 'Tags': ((<class 'troposphere.Tags'>, <type 'list'>), False)}
    resource_type = 'AWS::EC2::LaunchTemplate'
```



```

class troposphere.ec2.LaunchTemplateConfigs (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'LaunchTemplateSpecification': (<class 'troposphere.ec2.LaunchTemplateSpecif

class troposphere.ec2.LaunchTemplateCreditSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CpuCredits': (<type 'basestring'>, False)}

class troposphere.ec2.LaunchTemplateData (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'BlockDeviceMappings': ([<class 'troposphere.ec2.BlockDeviceMapping'>], Fals

class troposphere.ec2.LaunchTemplateOverrides (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AvailabilityZone': (<type 'basestring'>, False), 'InstanceType': (<type 'b

class troposphere.ec2.LaunchTemplateSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'LaunchTemplateId': (<type 'basestring'>, False), 'LaunchTemplateName': (<t

class troposphere.ec2.LicenseSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'LicenseConfigurationArn': (<type 'basestring'>, True)}

class troposphere.ec2.LoadBalancersConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ClassicLoadBalancersConfig': ([<class 'troposphere.ec2.ClassicLoadBalancers

class troposphere.ec2.Monitoring (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Enabled': (<function boolean at 0x7f30f87f5938>, False)}

class troposphere.ec2.MountPoint (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Device': (<type 'basestring'>, True), 'VolumeId': (<type 'basestring'>, Tr

class troposphere.ec2.NatGateway (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AllocationId': (<type 'basestring'>, True), 'SubnetId': (<type 'basestring

    resource_type = 'AWS::EC2::NatGateway'

class troposphere.ec2.NetworkAcl (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Tags': ((<class 'troposphere.Tags'>, <type 'list'>), False), 'VpcId': (<ty

    resource_type = 'AWS::EC2::NetworkAcl'

class troposphere.ec2.NetworkAclEntry (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'CidrBlock': (<type 'basestring'>, False), 'Egress': (<function boolean at

    resource_type = 'AWS::EC2::NetworkAclEntry'

    validate()

```

```
class troposphere.ec2.NetworkInterface (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'GroupSet': (<type 'list'>, Fa
    resource_type = 'AWS::EC2::NetworkInterface'

class troposphere.ec2.NetworkInterfaceAttachment (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DeleteOnTermination': (<function boolean at 0x7f30f87f5938>, False), 'Device
    resource_type = 'AWS::EC2::NetworkInterfaceAttachment'

class troposphere.ec2.NetworkInterfacePermission (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AwsAccountId': (<type 'basestring'>, True), 'NetworkInterfaceId': (<type '
    resource_type = 'AWS::EC2::NetworkInterfacePermission'

class troposphere.ec2.NetworkInterfaceProperty (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AssociatePublicIpAddress': (<function boolean at 0x7f30f87f5938>, False), '

class troposphere.ec2.NetworkInterfaces (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AssociatePublicIpAddress': (<function boolean at 0x7f30f87f5938>, False), '

class troposphere.ec2.OnDemandOptionsRequest (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AllocationStrategy': (<type 'basestring'>, False)}

class troposphere.ec2.Placement (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AvailabilityZone': (<type 'basestring'>, False), 'GroupName': (<type 'base

class troposphere.ec2.PlacementGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Strategy': (<type 'basestring'>, True)}

    resource_type = 'AWS::EC2::PlacementGroup'

class troposphere.ec2.PortRange (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'From': (<function network_port at 0x7f30f87f5cf8>, False), 'To': (<functio

class troposphere.ec2.PrivateIpAddressSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Primary': (<function boolean at 0x7f30f87f5938>, True), 'PrivateIpAddress':

class troposphere.ec2.Route (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DestinationCidrBlock': (<type 'basestring'>, False), 'DestinationIpv6CidrBl
    resource_type = 'AWS::EC2::Route'

    validate()
```

```

class troposphere.ec2.RouteTable (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Tags': ((<class 'troposphere.Tags'>, <type 'list'>), False), 'VpcId': (<type 'basestring'>, True)}
    resource_type = 'AWS::EC2::RouteTable'

class troposphere.ec2.SecurityGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'GroupDescription': (<type 'basestring'>, True), 'GroupName': (<type 'basestring'>, True)}
    resource_type = 'AWS::EC2::SecurityGroup'

class troposphere.ec2.SecurityGroupEgress (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CidrIp': (<type 'basestring'>, False), 'CidrIpv6': (<type 'basestring'>, False)}
    resource_type = 'AWS::EC2::SecurityGroupEgress'
    validate()

class troposphere.ec2.SecurityGroupIngress (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CidrIp': (<type 'basestring'>, False), 'CidrIpv6': (<type 'basestring'>, False)}
    resource_type = 'AWS::EC2::SecurityGroupIngress'
    validate()

class troposphere.ec2.SecurityGroupRule (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CidrIp': (<type 'basestring'>, False), 'CidrIpv6': (<type 'basestring'>, False)}

class troposphere.ec2.SecurityGroups (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'GroupId': (<type 'basestring'>, False)}

class troposphere.ec2.SpotFleet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'SpotFleetRequestConfigData': (<class 'troposphere.ec2.SpotFleetRequestConfigData'>, True)}
    resource_type = 'AWS::EC2::SpotFleet'

class troposphere.ec2.SpotFleetRequestConfigData (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AllocationStrategy': (<type 'basestring'>, False), 'ExcessCapacityTerminationProtection': (False, True)}
    validate()

class troposphere.ec2.SpotFleetTagSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ResourceType': (<type 'basestring'>, True), 'Tags': ((<class 'troposphere.Tags'>, <type 'list'>), False)}

class troposphere.ec2.SpotOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'InstanceInterruptionBehavior': (<type 'basestring'>, False), 'MaxPrice': (<type 'basestring'>, True)}

```

```
class troposphere.ec2.SpotOptionsRequest (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AllocationStrategy': (<type 'basestring'>, False), 'InstanceInterruptionBeha

class troposphere.ec2.SsmAssociations (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AssociationParameters': ([<class 'troposphere.ec2.AssociationParameters'>],

class troposphere.ec2.Subnet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AssignIpv6AddressOnCreation': (<function boolean at 0x7f30f87f5938>, False)
    resource_type = 'AWS::EC2::Subnet'

    validate()

class troposphere.ec2.SubnetCidrBlock (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Ipv6CidrBlock': (<type 'basestring'>, True), 'SubnetId': (<type 'basestrin
    resource_type = 'AWS::EC2::SubnetCidrBlock'

class troposphere.ec2.SubnetNetworkAclAssociation (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'NetworkAclId': (<type 'basestring'>, True), 'SubnetId': (<type 'basestring
    resource_type = 'AWS::EC2::SubnetNetworkAclAssociation'

class troposphere.ec2.SubnetRouteTableAssociation (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'RouteTableId': (<type 'basestring'>, True), 'SubnetId': (<type 'basestring
    resource_type = 'AWS::EC2::SubnetRouteTableAssociation'

class troposphere.ec2.Tag (key=None, value=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Key': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}

class troposphere.ec2.TagSpecifications (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ResourceType': (<type 'basestring'>, False), 'Tags': ((<class 'troposphere

class troposphere.ec2.TargetCapacitySpecificationRequest (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DefaultTargetCapacityType': (<type 'basestring'>, False), 'OnDemandTargetCap

class troposphere.ec2.TargetGroup (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Arn': (<type 'basestring'>, True)}

class troposphere.ec2.TransitGateway (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AmazonSideAsn': (<function integer at 0x7f30f87f59b0>, False), 'AutoAcceptS
    resource_type = 'AWS::EC2::TransitGateway'
```

```

class troposphere.ec2.TransitGatewayAttachment (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'SubnetIds': ([<type 'basestring'>], True), 'Tags': ((<class 'troposphere.Tags'>),
                                                                <type 'list'>), False), 'ResourceType':
    resource_type = 'AWS::EC2::TransitGatewayAttachment'

class troposphere.ec2.TransitGatewayRoute (title, template=None, validation=True,
                                           **kwargs)
    Bases: troposphere.AWSObject
    props = {'Blackhole': (<function boolean at 0x7f30f87f5938>, False), 'DestinationCidrBlock':
    resource_type = 'AWS::EC2::TransitGatewayRoute'

class troposphere.ec2.TransitGatewayRouteTable (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'Tags': ((<class 'troposphere.Tags'>, <type 'list'>), False), 'TransitGatewayAttachmentId':
    resource_type = 'AWS::EC2::TransitGatewayRouteTable'

class troposphere.ec2.TransitGatewayRouteTableAssociation (title, template=None,
                                                          validation=True,
                                                          **kwargs)
    Bases: troposphere.AWSObject
    props = {'TransitGatewayAttachmentId': (<type 'basestring'>, True), 'TransitGatewayRouteTableId':
    resource_type = 'AWS::EC2::TransitGatewayRouteTableAssociation'

class troposphere.ec2.TransitGatewayRouteTablePropagation (title, template=None,
                                                          validation=True,
                                                          **kwargs)
    Bases: troposphere.AWSObject
    props = {'TransitGatewayAttachmentId': (<type 'basestring'>, True), 'TransitGatewayRouteTableId':
    resource_type = 'AWS::EC2::TransitGatewayRouteTablePropagation'

class troposphere.ec2.VPC (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CidrBlock': (<type 'basestring'>, True), 'EnableDnsHostnames': (<function boolean at
    resource_type = 'AWS::EC2::VPC'

class troposphere.ec2.VPCCidrBlock (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AmazonProvidedIpv6CidrBlock': (<function boolean at 0x7f30f87f5938>, False), 'CidrBlock':
    resource_type = 'AWS::EC2::VPCCidrBlock'

class troposphere.ec2.VPCDHCPOptionsAssociation (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'DhcpOptionsId': (<type 'basestring'>, True), 'VpcId': (<type 'basestring'>,
    resource_type = 'AWS::EC2::VPCDHCPOptionsAssociation'

class troposphere.ec2.VPCEndpoint (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'PolicyDocument': ((<type 'dict'>, <class 'awacs.aws.Policy'>), False), 'PolicyName':

```

```
    resource_type = 'AWS::EC2::VPCEndpoint'
class troposphere.ec2.VPCEndpointConnectionNotification (title, template=None, vali-
    dation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ConnectionEvents': ([<type 'basestring'>], True), 'ConnectionNotificationArn':
    resource_type = 'AWS::EC2::VPCEndpointConnectionNotification'
class troposphere.ec2.VPCEndpointService (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'AcceptanceRequired': (<function boolean at 0x7f30f87f5938>, False), 'NetworkInterfaceId':
    resource_type = 'AWS::EC2::VPCEndpointService'
class troposphere.ec2.VPCEndpointServicePermissions (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AllowedPrincipals': ([<type 'basestring'>], False), 'ServiceId': (<type 'basestring'>,
    resource_type = 'AWS::EC2::VPCEndpointServicePermissions'
class troposphere.ec2.VPCGatewayAttachment (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'InternetGatewayId': (<type 'basestring'>, False), 'VpcId': (<type 'basestring'>,
    resource_type = 'AWS::EC2::VPCGatewayAttachment'
class troposphere.ec2.VPCPeeringConnection (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'PeerOwnerId': (<type 'basestring'>, False), 'PeerRegion': (<type 'basestring'>,
    resource_type = 'AWS::EC2::VPCPeeringConnection'
class troposphere.ec2.VPNConnection (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CustomerGatewayId': (<type 'basestring'>, True), 'StaticRoutesOnly': (<function boolean at
    resource_type = 'AWS::EC2::VPNConnection'
class troposphere.ec2.VPNConnectionRoute (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'DestinationCidrBlock': (<type 'basestring'>, True), 'VpnConnectionId': (<type 'basestring'>,
    resource_type = 'AWS::EC2::VPNConnectionRoute'
class troposphere.ec2.VPNGateway (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AmazonSideAsn': (<function positive_integer at 0x7f30f87f5a28>, False), 'TagSet':
    resource_type = 'AWS::EC2::VPNGateway'
class troposphere.ec2.VPNGatewayRoutePropagation (title, template=None, valida-
    tion=True, **kwargs)
    Bases: troposphere.AWSObject
```

```

    props = {'RouteTableIds': ([<type 'basestring'>], True), 'VpnGatewayId': (<type 'bas
    resource_type = 'AWS::EC2::VPNGatewayRoutePropagation'
class troposphere.ec2.Volume (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoEnableIO': (<function boolean at 0x7f30f87f5938>, False), 'AvailabilityZ
    resource_type = 'AWS::EC2::Volume'
class troposphere.ec2.VolumeAttachment (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Device': (<type 'basestring'>, True), 'InstanceId': (<type 'basestring'>,
    resource_type = 'AWS::EC2::VolumeAttachment'
class troposphere.ec2.VpnTunnelOptionsSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PreSharedKey': (<function vpn_pre_shared_key at 0x7f30f87fb7d0>, False), 'T
troposphere.ec2.check_ports (props)
troposphere.ec2.instance_tenancy (value)
troposphere.ec2.validate_elasticinferenceaccelerator_type (elasticinferenceaccelerator_type)
    Validate ElasticInferenceAccelerator for Instance

```

troposphere.ecr module

```

class troposphere.ecr.LifecyclePolicy (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'LifecyclePolicyText': (<type 'basestring'>, False), 'RegistryId': (<type '
class troposphere.ecr.Repository (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'LifecyclePolicy': (<class 'troposphere.ecr.LifecyclePolicy'>, False), 'Repo
    resource_type = 'AWS::ECR::Repository'

```

troposphere.ecs module

```

class troposphere.ecs.AwsvpcConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AssignPublicIp': (<type 'basestring'>, False), 'SecurityGroups': (<type '
class troposphere.ecs.Cluster (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ClusterName': (<type 'basestring'>, False)}
    resource_type = 'AWS::ECS::Cluster'
class troposphere.ecs.ContainerDefinition (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Command': ([<type 'basestring'>], False), 'Cpu': (<function positive_integ

```

```
class troposphere.ecs.DeploymentConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'MaximumPercent': (<function positive_integer at 0x7f30f87f5a28>, False), 'M

class troposphere.ecs.Device (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ContainerPath': (<type 'basestring'>, False), 'HostPath': (<type 'basestri

class troposphere.ecs.DockerVolumeConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Autoprovision': (<function boolean at 0x7f30f87f5938>, False), 'Driver': (

class troposphere.ecs.Environment (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}

class troposphere.ecs.HealthCheck (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Command': ([<type 'basestring'>], True), 'Interval': (<function integer at

class troposphere.ecs.Host (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'SourcePath': (<type 'basestring'>, False)}

class troposphere.ecs.HostEntry (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Hostname': (<type 'basestring'>, True), 'IpAddress': (<type 'basestring'>,

class troposphere.ecs.KernelCapabilities (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Add': ([<type 'basestring'>], False), 'Drop': ([<type 'basestring'>], Fals

class troposphere.ecs.LinuxParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Capabilities': (<class 'troposphere.ecs.KernelCapabilities'>, False), 'Devi

class troposphere.ecs.LoadBalancer (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ContainerName': (<type 'basestring'>, False), 'ContainerPort': (<function

class troposphere.ecs.LogConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'LogDriver': (<type 'basestring'>, True), 'Options': (<type 'dict'>, False)

class troposphere.ecs.MountPoint (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ContainerPath': (<type 'basestring'>, True), 'ReadOnly': (<function boolea

class troposphere.ecs.NetworkConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AwsvpcConfiguration': (<class 'troposphere.ecs.AwsvpcConfiguration'>, False

class troposphere.ecs.PlacementConstraint (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```



```

    props = {'Expression': (<type 'basestring'>, False), 'Type': (<function placement_co
class troposphere.ecs.PlacementStrategy (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Field': (<type 'basestring'>, False), 'Type': (<function placement_strateg
class troposphere.ecs.PortMapping (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ContainerPort': (<function network_port at 0x7f30f87f5cf8>, True), 'HostPor
class troposphere.ecs.RepositoryCredentials (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CredentialsParameter': (<type 'basestring'>, False)}
class troposphere.ecs.Service (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Cluster': (<type 'basestring'>, False), 'DeploymentConfiguration': (<class
        resource_type = 'AWS::ECS::Service'
class troposphere.ecs.ServiceRegistry (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ContainerName': (<type 'basestring'>, False), 'ContainerPort': (<function
class troposphere.ecs.TaskDefinition (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ContainerDefinitions': ([<class 'troposphere.ecs.ContainerDefinition'>], Tr
        resource_type = 'AWS::ECS::TaskDefinition'
class troposphere.ecs.Ulimit (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'HardLimit': (<function integer at 0x7f30f87f59b0>, True), 'Name': (<type '
class troposphere.ecs.Volume (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DockerVolumeConfiguration': (<class 'troposphere.ecs.DockerVolumeConfigurat
class troposphere.ecs.VolumesFrom (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ReadOnly': (<function boolean at 0x7f30f87f5938>, False), 'SourceContainer'
troposphere.ecs.launch_type_validator (x)
troposphere.ecs.placement_constraint_validator (x)
troposphere.ecs.placement_strategy_validator (x)
troposphere.ecs.scope_validator (x)

```

troposphere.efs module

```

class troposphere.efs.FileSystem (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Encrypted': (<function boolean at 0x7f30f87f5938>, False), 'FileSystemTags'

```

```
    resource_type = 'AWS::EFS::FileSystem'
class troposphere.efs.MountTarget (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'FileSystemId': (<type 'basestring'>, True), 'IpAddress': (<type 'basestring'>, True), 'MountTargetId': (<type 'basestring'>, True), 'SubnetId': (<type 'basestring'>, True), 'VolumeId': (<type 'basestring'>, True)}
    resource_type = 'AWS::EFS::MountTarget'
troposphere.efs.provisioned_throughput_validator (throughput)
troposphere.efs.throughput_mode_validator (mode)
```

troposphere.eks module

```
class troposphere.eks.Cluster (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, False), 'ResourcesVpcConfig': (<class 'troposphere.eks.ResourcesVpcConfig'>, True), 'SubnetIds': (<type 'basestring'>, True), 'Tags': (<type 'basestring'>, True), 'Taints': (<type 'basestring'>, True), 'Version': (<type 'basestring'>, True)}
    resource_type = 'AWS::EKS::Cluster'
class troposphere.eks.ResourcesVpcConfig (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'SecurityGroupIds': ([<type 'basestring'>], False), 'SubnetIds': ([<type 'basestring'>], False), 'VpcId': (<type 'basestring'>, True)}
```

troposphere.elasticache module

```
class troposphere.elasticache.CacheCluster (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AZMode': (<type 'basestring'>, False), 'AutoMinorVersionUpgrade': (<function validate_auto_minor_version_upgrade at 0x7f30f7582c08>, True), 'CacheParameterGroupFamily': (<type 'basestring'>, True), 'CacheSubnetGroupName': (<type 'basestring'>, True), 'Engine': (<type 'basestring'>, True), 'EngineVersion': (<type 'basestring'>, True), 'InstanceProfileName': (<type 'basestring'>, True), 'NetworkType': (<type 'basestring'>, True), 'NumCacheNodes': (<type 'basestring'>, True), 'NumNodesPerAZ': (<type 'basestring'>, True), 'ParameterGroup': (<type 'basestring'>, True), 'ReplicationGroup': (<type 'basestring'>, True), 'SnapshotName': (<type 'basestring'>, True), 'SnapshotRetentionPeriod': (<type 'basestring'>, True), 'Tags': (<type 'basestring'>, True), 'VpcSubnet': (<type 'basestring'>, True)}
    resource_type = 'AWS::Elasticache::CacheCluster'
    validate ()
class troposphere.elasticache.NodeGroupConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'NodeGroupId': (<function validate_node_group_id at 0x7f30f7582c08>, False), 'NodeGroupType': (<type 'basestring'>, True), 'NodeGroupVersion': (<type 'basestring'>, True)}
class troposphere.elasticache.ParameterGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CacheParameterGroupFamily': (<type 'basestring'>, True), 'Description': (<type 'basestring'>, True), 'ParameterNames': (<type 'basestring'>, True), 'Tags': (<type 'basestring'>, True)}
    resource_type = 'AWS::Elasticache::ParameterGroup'
class troposphere.elasticache.ReplicationGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AtRestEncryptionEnabled': (<function boolean at 0x7f30f87f5938>, False), 'CacheParameterGroupFamily': (<type 'basestring'>, True), 'CacheSubnetGroupName': (<type 'basestring'>, True), 'Engine': (<type 'basestring'>, True), 'EngineVersion': (<type 'basestring'>, True), 'InstanceProfileName': (<type 'basestring'>, True), 'NetworkType': (<type 'basestring'>, True), 'NumCacheNodes': (<type 'basestring'>, True), 'NumNodesPerAZ': (<type 'basestring'>, True), 'ParameterGroup': (<type 'basestring'>, True), 'ReplicationGroup': (<type 'basestring'>, True), 'SnapshotName': (<type 'basestring'>, True), 'SnapshotRetentionPeriod': (<type 'basestring'>, True), 'Tags': (<type 'basestring'>, True), 'VpcSubnet': (<type 'basestring'>, True)}
    resource_type = 'AWS::Elasticache::ReplicationGroup'
    validate ()
```

```
class troposphere.elasticache.SecurityGroup(title, template=None, validation=True,
                                           **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'Description': (<type 'basestring'>, False)}
```

```
resource_type = 'AWS::ElasticCache::SecurityGroup'
```

```
class troposphere.elasticache.SecurityGroupIngress(title, template=None, validation=True,
                                                    **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'CacheSecurityGroupName': (<type 'basestring'>, True), 'EC2SecurityGroupName'
```

```
resource_type = 'AWS::ElasticCache::SecurityGroupIngress'
```

```
class troposphere.elasticache.SubnetGroup(title, template=None, validation=True,
                                           **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'CacheSubnetGroupName': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False)}
```

```
resource_type = 'AWS::ElasticCache::SubnetGroup'
```

```
troposphere.elasticache.validate_node_group_id(node_group_id)
```

troposphere.elasticbeanstalk module

```
class troposphere.elasticbeanstalk.Application(title, template=None, validation=True,
                                                **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'ApplicationName': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False)}
```

```
resource_type = 'AWS::ElasticBeanstalk::Application'
```

```
class troposphere.elasticbeanstalk.ApplicationResourceLifecycleConfig(title=None,
                                                                        **kwargs)
```

Bases: *troposphere.AWSPROPERTY*

```
props = {'ServiceRole': (<type 'basestring'>, False), 'VersionLifecycleConfig': (<type 'basestring'>, False)}
```

```
class troposphere.elasticbeanstalk.ApplicationVersion(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'ApplicationName': (<type 'basestring'>, True), 'Description': (<type 'basestring'>, False)}
```

```
resource_type = 'AWS::ElasticBeanstalk::ApplicationVersion'
```

```
class troposphere.elasticbeanstalk.ApplicationVersionLifecycleConfig(title=None,
                                                                        **kwargs)
```

Bases: *troposphere.AWSPROPERTY*

```
props = {'MaxAgeRule': (<class 'troposphere.elasticbeanstalk.MaxAgeRule'>, False), 'MinAgeRule': (<class 'troposphere.elasticbeanstalk.MinAgeRule'>, False)}
```

```
class troposphere.elasticbeanstalk.ConfigurationTemplate(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'ApplicationName': (<type 'basestring'>, True), 'Description': (<type 'basestring'>, False)}
```

```
resource_type = 'AWS::ElasticBeanstalk::ConfigurationTemplate'
```

```
class troposphere.elasticbeanstalk.Environment (title, template=None, validation=True,  
                                             **kwargs)  
    Bases: troposphere.AWSObject  
    props = {'ApplicationName': (<type 'basestring'>, True), 'CNAMEPrefix': (<type 'basestring'>, True), 'EnvironmentName': (<type 'basestring'>, True), 'Profile': (<type 'basestring'>, True), 'ServiceRole': (<type 'basestring'>, True), 'Subnets': (<type 'list'>, True), 'Tags': (<type 'dict'>, True), 'VPCSubnet': (<type 'basestring'>, True), 'VPCZoneIdentifier': (<type 'basestring'>, True), 'VirtualizationConfiguration': (<type 'dict'>, True), 'WeightedRoundRobin': (<type 'dict'>, True)}  
    resource_type = 'AWS::ElasticBeanstalk::Environment'  
  
class troposphere.elasticbeanstalk.MaxAgeRule (title=None, **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'DeleteSourceFromS3': (<function boolean at 0x7f30f87f5938>, False), 'Enabled': (<type 'boolean'>, True), 'MaxAge': (<type 'integer'>, True)}  
  
class troposphere.elasticbeanstalk.MaxCountRule (title=None, **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'DeleteSourceFromS3': (<function boolean at 0x7f30f87f5938>, False), 'Enabled': (<type 'boolean'>, True), 'MaxCount': (<type 'integer'>, True)}  
  
class troposphere.elasticbeanstalk.OptionSettings (title=None, **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'Namespace': (<type 'basestring'>, True), 'OptionName': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}  
  
class troposphere.elasticbeanstalk.SourceBundle (title=None, **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'S3Bucket': (<type 'basestring'>, True), 'S3Key': (<type 'basestring'>, True), 'Version': (<type 'basestring'>, True)}  
  
class troposphere.elasticbeanstalk.SourceConfiguration (title=None, **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'ApplicationName': (<type 'basestring'>, True), 'TemplateName': (<type 'basestring'>, True), 'Version': (<type 'basestring'>, True)}  
  
class troposphere.elasticbeanstalk.Tier (title=None, **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'Name': (<function validate_tier_name at 0x7f30f73982a8>, False), 'Type': (<type 'basestring'>, True)}  
  
troposphere.elasticbeanstalk.validate_tier_name (name)  
troposphere.elasticbeanstalk.validate_tier_type (tier_type)
```

troposphere.elasticloadbalancing module

```
class troposphere.elasticloadbalancing.AccessLoggingPolicy (title=None,  
                                                         **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'EmitInterval': (<function integer at 0x7f30f87f59b0>, False), 'Enabled': (<type 'boolean'>, True), 'LogGroupName': (<type 'basestring'>, True)}  
  
class troposphere.elasticloadbalancing.AppCookieStickinessPolicy (title=None,  
                                                                    **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'CookieName': (<type 'basestring'>, True), 'PolicyName': (<type 'basestring'>, True), 'PolicyType': (<type 'basestring'>, True)}  
  
class troposphere.elasticloadbalancing.ConnectionDrainingPolicy (title=None,  
                                                                    **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'Enabled': (<type 'bool'>, True), 'Timeout': (<function integer at 0x7f30f87f59b0>, True)}  
  
class troposphere.elasticloadbalancing.ConnectionSettings (title=None, **kwargs)  
    Bases: troposphere.AWSProperty  
    props = {'IdleTimeout': (<function integer at 0x7f30f87f59b0>, True)}
```

```

class troposphere.elasticloadbalancing.HealthCheck (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'HealthyThreshold': (<function integer_range_checker at 0x7f30f875f230>, True)

class troposphere.elasticloadbalancing.LBCookieStickinessPolicy (title=None,
                                                                **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CookieExpirationPeriod': (<type 'basestring'>, False), 'PolicyName': (<type

class troposphere.elasticloadbalancing.Listener (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'InstancePort': (<function network_port at 0x7f30f87f5cf8>, True), 'Instance

class troposphere.elasticloadbalancing.LoadBalancer (title, template=None, valida-
                                                    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AccessLoggingPolicy': (<class 'troposphere.elasticloadbalancing.AccessLoggi
    resource_type = 'AWS::ElasticLoadBalancing::LoadBalancer'

class troposphere.elasticloadbalancing.Policy (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Attributes': ([<type 'dict'>], False), 'InstancePorts': (<type 'list'>, Fa

```

troposphere.elasticloadbalancingv2 module

```

class troposphere.elasticloadbalancingv2.Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'FixedResponseConfig': (<class 'troposphere.elasticloadbalancingv2.FixedResp
    validate()

class troposphere.elasticloadbalancingv2.Certificate (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CertificateArn': (<type 'basestring'>, False)}

class troposphere.elasticloadbalancingv2.Condition (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Field': (<type 'basestring'>, True), 'Values': ([<type 'basestring'>], True)

class troposphere.elasticloadbalancingv2.FixedResponseConfig (title=None,
                                                                **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ContentType': (<type 'basestring'>, False), 'MessageBody': (<type 'basestr
    validate()

class troposphere.elasticloadbalancingv2.Listener (title, template=None, valida-
                                                    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Certificates': ([<class 'troposphere.elasticloadbalancingv2.Certificate'>],
    resource_type = 'AWS::ElasticLoadBalancingV2::Listener'

```

```
class troposphere.elasticloadbalancingv2.ListenerCertificate (title, template=None, validation=True, **kwargs)

Bases: troposphere.AWSObject

props = {'Certificates': ([<class 'troposphere.elasticloadbalancingv2.Certificate'>],
resource_type = 'AWS::ElasticLoadBalancingV2::ListenerCertificate'

class troposphere.elasticloadbalancingv2.ListenerRule (title, template=None, validation=True, **kwargs)

Bases: troposphere.AWSObject

props = {'Actions': ([<class 'troposphere.elasticloadbalancingv2.Action'>], True), 'C
resource_type = 'AWS::ElasticLoadBalancingV2::ListenerRule'

class troposphere.elasticloadbalancingv2.LoadBalancer (title, template=None, validation=True, **kwargs)

Bases: troposphere.AWSObject

props = {'IpAddressType': (<type 'basestring'>, False), 'LoadBalancerAttributes': ([
resource_type = 'AWS::ElasticLoadBalancingV2::LoadBalancer'

validate ()

class troposphere.elasticloadbalancingv2.LoadBalancerAttributes (title=None, **kwargs)

Bases: troposphere.AWSProperty

props = {'Key': (<type 'basestring'>, False), 'Value': (<type 'basestring'>, False)}

class troposphere.elasticloadbalancingv2.Matcher (title=None, **kwargs)

Bases: troposphere.AWSProperty

props = {'HttpCode': (<type 'basestring'>, False)}

class troposphere.elasticloadbalancingv2.RedirectConfig (title=None, **kwargs)

Bases: troposphere.AWSProperty

props = {'Host': (<type 'basestring'>, False), 'Path': (<type 'basestring'>, False),
validate ()

class troposphere.elasticloadbalancingv2.SubnetMapping (title=None, **kwargs)

Bases: troposphere.AWSProperty

props = {'AllocationId': (<type 'basestring'>, True), 'SubnetId': (<type 'basestring'

class troposphere.elasticloadbalancingv2.TargetDescription (title=None, **kwargs)

Bases: troposphere.AWSProperty

props = {'AvailabilityZone': (<type 'basestring'>, False), 'Id': (<type 'basestring'

class troposphere.elasticloadbalancingv2.TargetGroup (title, template=None, validation=True, **kwargs)

Bases: troposphere.AWSObject

props = {'HealthCheckIntervalSeconds': (<function integer at 0x7f30f87f59b0>, False),
resource_type = 'AWS::ElasticLoadBalancingV2::TargetGroup'
```

```
class troposphere.elasticloadbalancingv2.TargetGroupAttribute (title=None,
                                                             **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Key': (<type 'basestring'>, False), 'Value': (<type 'basestring'>, False)}
```

troposphere.elasticsearch module

```
class troposphere.elasticsearch.Domain (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AccessPolicies': ((<type 'dict'>, <class 'awacs.aws.Policy'>), False), 'AdvancedOptions': (None, False), 'ResourceType': ('AWS::Elasticsearch::Domain', False)}
```

```
class troposphere.elasticsearch.EBSOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'EBSEnabled': (<function boolean at 0x7f30f87f5938>, False), 'Iops': (<function validate at 0x7f30f87f5938>, False)}
```

```
class troposphere.elasticsearch.ElasticsearchClusterConfig (title=None,
                                                            **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DedicatedMasterCount': (<function integer at 0x7f30f87f59b0>, False), 'DedicatedMasterEnabled': (None, False)}
```

```
troposphere.elasticsearch.ElasticsearchDomain
    alias of troposphere.elasticsearch.Domain
```

```
class troposphere.elasticsearch.EncryptionAtRestOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Enabled': (<function boolean at 0x7f30f87f5938>, False), 'KmsKeyId': (<type 'basestring'>, False)}
```

```
class troposphere.elasticsearch.SnapshotOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AutomatedSnapshotStartHour': (<function integer_range_checker at 0x7f30f739f739>, False)}
```

```
class troposphere.elasticsearch.VPCOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'SecurityGroupIds': ([<type 'basestring'>], False), 'SubnetIds': ([<type 'basestring'>], False)}
```

```
troposphere.elasticsearch.validate_volume_type (volume_type)
    Validate VolumeType for ElasticsearchDomain
```

troposphere.emr module

```
class troposphere.emr.Application (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AdditionalInfo': (<function additional_info_validator at 0x7f30f864d9b0>, False)}
```

```
class troposphere.emr.AutoScalingPolicy (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Constraints': (<class 'troposphere.emr.ScalingConstraints'>, True), 'Rules': (None, False)}
```

```
class troposphere.emr.BootstrapActionConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'Name': (<type 'basestring'>, True), 'ScriptBootstrapAction': (<class 'troposphere.emr.ScriptBootstrapAction'>, True)}
class troposphere.emr.CloudWatchAlarmDefinition (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ComparisonOperator': (<type 'basestring'>, True), 'Dimensions': ([<class 'troposphere.emr.CloudWatchAlarmDefinition.Dimension'>], False)}
class troposphere.emr.Cluster (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AdditionalInfo': (<type 'dict'>, False), 'Applications': ([<class 'troposphere.emr.Cluster.Application'>], False), 'ResourceType': ('AWS::EMR::Cluster', True)}
    resource_type = 'AWS::EMR::Cluster'
class troposphere.emr.Configuration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Classification': (<type 'basestring'>, False), 'ConfigurationProperties': ([<class 'troposphere.emr.Configuration.ConfigurationProperty'>], False)}
class troposphere.emr.EbsBlockDeviceConfigs (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'VolumeSpecification': (<class 'troposphere.emr.EbsBlockDeviceConfigs.VolumeSpecification'>, True)}
class troposphere.emr.EbsConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'EbsBlockDeviceConfigs': ([<class 'troposphere.emr.EbsBlockDeviceConfigs'>], False)}
class troposphere.emr.HadoopJarStepConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Args': ([<type 'basestring'>], False), 'Jar': (<type 'basestring'>, True), 'Name': (<type 'basestring'>, True)}
class troposphere.emr.InstanceFleetConfig (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ClusterId': (<type 'basestring'>, True), 'InstanceFleetType': (<type 'basestring'>, True), 'ResourceType': ('AWS::EMR::InstanceFleetConfig', True)}
    resource_type = 'AWS::EMR::InstanceFleetConfig'
class troposphere.emr.InstanceFleetConfigProperty (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'InstanceTypeConfigs': ([<class 'troposphere.emr.InstanceTypeConfig'>], False)}
class troposphere.emr.InstanceFleetProvisioningSpecifications (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'SpotSpecification': (<class 'troposphere.emr.InstanceFleetProvisioningSpecifications.SpotSpecification'>, True)}
class troposphere.emr.InstanceGroupConfig (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoScalingPolicy': (<class 'troposphere.emr.InstanceGroupConfig.AutoScalingPolicy'>, False), 'ResourceType': ('AWS::EMR::InstanceGroupConfig', True)}
    resource_type = 'AWS::EMR::InstanceGroupConfig'
class troposphere.emr.InstanceGroupConfigProperty (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AutoScalingPolicy': (<class 'troposphere.emr.InstanceGroupConfig.AutoScalingPolicy'>, False), 'Name': (<type 'basestring'>, True)}
```



```

class troposphere.emr.InstanceTypeConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'BidPrice': (<type 'basestring'>, False), 'BidPriceAsPercentageOfOnDemandPri

class troposphere.emr.JobFlowInstancesConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AdditionalMasterSecurityGroups': ([<type 'basestring'>], False), 'Additional

class troposphere.emr.KerberosAttributes (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ADDomainJoinPassword': (<type 'basestring'>, False), 'ADDomainJoinUser': (

class troposphere.emr.KeyValue (key=None, value=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Key': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}

troposphere.emr.MetricDimension
    alias of troposphere.emr.KeyValue

class troposphere.emr.PlacementType (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AvailabilityZone': (<type 'basestring'>, True)}

class troposphere.emr.ScalingAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Market': (<function market_validator at 0x7f30f864db18>, False), 'SimpleSca

class troposphere.emr.ScalingConstraints (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'MaxCapacity': (<function integer at 0x7f30f87f59b0>, True), 'MinCapacity':

class troposphere.emr.ScalingRule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Action': (<class 'troposphere.emr.ScalingAction'>, True), 'Description': (

class troposphere.emr.ScalingTrigger (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CloudWatchAlarmDefinition': (<class 'troposphere.emr.CloudWatchAlarmDefinit

class troposphere.emr.ScriptBootstrapActionConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Args': ([<type 'basestring'>], False), 'Path': (<type 'basestring'>, True)}

class troposphere.emr.SecurityConfiguration (title, template=None, validation=True,
                                             **kwargs)
    Bases: troposphere.AWSObject

    props = {'Name': (<type 'basestring'>, False), 'SecurityConfiguration': (<type 'dict

    resource_type = 'AWS::EMR::SecurityConfiguration'

class troposphere.emr.SimpleScalingPolicyConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AdjustmentType': (<type 'basestring'>, False), 'CoolDown': (<function posi

    validate()

```

```
class troposphere.emr.SpotProvisioningSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'BlockDurationMinutes': (<function positive_integer at 0x7f30f87f5a28>, False)

class troposphere.emr.Step (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ActionOnFailure': (<function action_on_failure_validator at 0x7f30f864dc08>, False)
    resource_type = 'AWS::EMR::Step'

class troposphere.emr.StepConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ActionOnFailure': (<function validate_action_on_failure at 0x7f30f864d938>, False)

class troposphere.emr.VolumeSpecification (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Iops': (<function integer at 0x7f30f87f59b0>, False), 'SizeInGB': (<function integer at 0x7f30f87f59b0>, False)

troposphere.emr.action_on_failure_validator(x)
troposphere.emr.additional_info_validator(xs)
troposphere.emr.market_validator(x)
troposphere.emr.properties_validator(xs)
troposphere.emr.validate_action_on_failure(action_on_failure)
    Validate action on failure for EMR StepConfig
troposphere.emr.volume_type_validator(x)
```

troposphere.events module

```
class troposphere.events.EcsParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'TaskCount': (<type 'int'>, False), 'TaskDefinitionArn': (<type 'basestring'>, False)

class troposphere.events.InputTransformer (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'InputPathsMap': (<type 'dict'>, False), 'InputTemplate': (<type 'basestring'>, False)

class troposphere.events.KinesisParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'PartitionKeyPath': (<type 'basestring'>, True)}

class troposphere.events.Rule (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'EventPattern': (<type 'dict'>, False)
    resource_type = 'AWS::Events::Rule'

class troposphere.events.RunCommandParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'RunCommandTargets': ([<class 'troposphere.events.RunCommandTarget'>], True)

class troposphere.events.RunCommandTarget (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```

    props = {'Key': (<type 'basestring'>, True), 'Values': ([<type 'basestring'>], True)}
class troposphere.events.SqsParameters (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'MessageGroupId': (<type 'basestring'>, True)}
class troposphere.events.Target (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Arn': (<type 'basestring'>, True), 'EcsParameters': (<class 'troposphere.e

```

troposphere.firehose module

```

class troposphere.firehose.BufferingHints (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'IntervalInSeconds': (<function positive_integer at 0x7f30f87f5a28>, True),
class troposphere.firehose.CloudWatchLoggingOptions (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Enabled': (<function boolean at 0x7f30f87f5938>, False), 'LogGroupName': (
class troposphere.firehose.CopyCommand (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'CopyOptions': (<type 'basestring'>, False), 'DataTableColumns': (<type 'ba
class troposphere.firehose.DeliveryStream (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'DeliveryStreamName': (<type 'basestring'>, False), 'DeliveryStreamType': (
        resource_type = 'AWS::KinesisFirehose::DeliveryStream'
class troposphere.firehose.ElasticsearchDestinationConfiguration (title=None,
    **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'BufferingHints': (<class 'troposphere.firehose.BufferingHints'>, True), 'Cl
class troposphere.firehose.EncryptionConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'KMSEncryptionConfig': (<class 'troposphere.firehose.KMSEncryptionConfig'>,
class troposphere.firehose.ExtendedS3DestinationConfiguration (title=None,
    **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'BucketARN': (<type 'basestring'>, True), 'BufferingHints': (<class 'troposph
class troposphere.firehose.KMSEncryptionConfig (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'AWSKMSKeyARN': (<type 'basestring'>, True)}
class troposphere.firehose.KinesisStreamSourceConfiguration (title=None,
    **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'KinesisStreamARN': (<type 'basestring'>, True), 'RoleARN': (<type 'basestrin

```

```
class troposphere.firehose.ProcessingConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Enabled': (<function boolean at 0x7f30f87f5938>, True), 'Processors': ([<c

class troposphere.firehose.Processor (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Parameters': ([<class 'troposphere.firehose.ProcessorParameter'>], True), '

class troposphere.firehose.ProcessorParameter (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ParameterName': (<type 'basestring'>, True), 'ParameterValue': (<type 'bas

class troposphere.firehose.RedshiftDestinationConfiguration (title=None,
                                                                **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CloudWatchLoggingOptions': (<class 'troposphere.firehose.CloudWatchLoggingO

class troposphere.firehose.RetryOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DurationInSeconds': (<function positive_integer at 0x7f30f87f5a28>, True)}

class troposphere.firehose.S3Configuration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BucketARN': (<type 'basestring'>, True), 'BufferingHints': (<class 'troposph

class troposphere.firehose.S3DestinationConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BucketARN': (<type 'basestring'>, True), 'BufferingHints': (<class 'troposph

class troposphere.firehose.SplunkDestinationConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CloudWatchLoggingOptions': (<class 'troposphere.firehose.CloudWatchLoggingO

class troposphere.firehose.SplunkRetryOptions (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DurationInSeconds': (<function positive_integer at 0x7f30f87f5a28>, True)}

troposphere.firehose.delivery_stream_type_validator(x)
troposphere.firehose.index_rotation_period_validator(x)
troposphere.firehose.processor_type_validator(x)
troposphere.firehose.s3_backup_mode_elastic_search_validator(x)
troposphere.firehose.s3_backup_mode_extended_s3_validator(x)
```

troposphere.glue module

```
class troposphere.glue.Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Arguments': (<type 'dict'>, False), 'JobName': (<type 'basestring'>, False)

class troposphere.glue.Classifier (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
```

```

    props = {'GrokClassifier': (<class 'troposphere.glue.GrokClassifier'>, False), 'JsonC
    resource_type = 'AWS::Glue::Classifier'
class troposphere.glue.Column (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Comment': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, True)
class troposphere.glue.Condition (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'JobName': (<type 'basestring'>, False), 'LogicalOperator': (<type 'basestr
class troposphere.glue.Connection (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CatalogId': (<type 'basestring'>, True), 'ConnectionInput': (<class 'tropo
    resource_type = 'AWS::Glue::Connection'
class troposphere.glue.ConnectionInput (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ConnectionProperties': (<type 'dict'>, True), 'ConnectionType': (<function
class troposphere.glue.ConnectionsList (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Connections': ([<type 'basestring'>], False)}
class troposphere.glue.Crawler (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Classifiers': ([<type 'basestring'>], False), 'Configuration': (<type 'bas
    resource_type = 'AWS::Glue::Crawler'
class troposphere.glue.Database (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CatalogId': (<type 'basestring'>, True), 'DatabaseInput': (<class 'troposp
    resource_type = 'AWS::Glue::Database'
class troposphere.glue.DatabaseInput (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Description': (<type 'basestring'>, False), 'LocationUri': (<type 'basestr
class troposphere.glue.DevEndpoint (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'EndpointName': (<type 'basestring'>, False), 'ExtraJarsS3Path': (<type 'ba
    resource_type = 'AWS::Glue::DevEndpoint'
class troposphere.glue.ExecutionProperty (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MaxConcurrentRuns': (<function positive_integer at 0x7f30f87f5a28>, False)}
class troposphere.glue.GrokClassifier (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Classification': (<type 'basestring'>, True), 'CustomPatterns': (<type 'ba

```

```
class troposphere.glue.JdbcTarget (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ConnectionName': (<type 'basestring'>, False), 'Exclusions': ([<type 'basestring'>], False)}

class troposphere.glue.Job (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AllocatedCapacity': (<function double at 0x7f30f87f5b90>, False), 'Command': (<type 'basestring'>, False), 'ResourceArn': (<type 'basestring'>, False), 'ResourceType': ('AWS::Glue::Job', False)}
    resource_type = 'AWS::Glue::Job'

class troposphere.glue.JobCommand (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, False), 'ScriptLocation': (<type 'basestring'>, False)}

class troposphere.glue.JsonClassifier (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'JsonPath': (<type 'basestring'>, True), 'Name': (<type 'basestring'>, False)}

class troposphere.glue.Order (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Column': (<type 'basestring'>, True), 'SortOrder': (<function integer_range at 0x7f30f87f5b90>, False)}

class troposphere.glue.Partition (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'CatalogId': (<type 'basestring'>, True), 'DatabaseName': (<type 'basestring'>, False), 'Table': (<type 'basestring'>, False)}
    resource_type = 'AWS::Glue::Partition'

class troposphere.glue.PartitionInput (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Parameters': (<type 'dict'>, False), 'StorageDescriptor': (<class 'troposphere.glue.StorageDescriptor'>, False)}

class troposphere.glue.PhysicalConnectionRequirements (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AvailabilityZone': (<type 'basestring'>, True), 'SecurityGroupIdList': ([<type 'basestring'>], False)}

class troposphere.glue.Predicate (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Conditions': ([<class 'troposphere.glue.Condition'>], False), 'Logical': (<type 'basestring'>, False)}

class troposphere.glue.S3Target (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Exclusions': ([<type 'basestring'>], False), 'Path': (<type 'basestring'>, False)}

class troposphere.glue.Schedule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ScheduleExpression': (<type 'basestring'>, False)}

class troposphere.glue.SchemaChangePolicy (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DeleteBehavior': (<function delete_behavior_validator at 0x7f30f70982a8>, False)}

class troposphere.glue.SerdeInfo (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```

    props = {'Name': (<type 'basestring'>, False), 'Parameters': (<type 'dict'>, False),
class troposphere.glue.SkewedInfo (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'SkewedColumnNames': ([[<type 'basestring'>]], False), 'SkewedColumnValueLocat
class troposphere.glue.StorageDescriptor (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BucketColumns': ([[<type 'basestring'>]], False), 'Columns': ([[class 'tropo
class troposphere.glue.Table (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CatalogId': (<type 'basestring'>, True), 'DatabaseName': (<type 'basestrin
    resource_type = 'AWS::Glue::Table'
class troposphere.glue.TableInput (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Description': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, '
class troposphere.glue.Targets (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'JdbcTargets': ([[class 'troposphere.glue.JdbcTarget'>]], False), 'S3Targets'
class troposphere.glue.Trigger (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Actions': ([[class 'troposphere.glue.Action'>]], True), 'Description': (<ty
    resource_type = 'AWS::Glue::Trigger'
class troposphere.glue.XMLClassifier (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Classification': (<type 'basestring'>, True), 'Name': (<type 'basestring'>
troposphere.glue.connection_type_validator (type)
troposphere.glue.delete_behavior_validator (value)
troposphere.glue.table_type_validator (type)
troposphere.glue.trigger_type_validator (type)
troposphere.glue.update_behavior_validator (value)

```

troposphere.guardduty module

```

class troposphere.guardduty.Condition (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Eq': ([[<type 'basestring'>]], False), 'Gte': (<function integer at 0x7f30f8
class troposphere.guardduty.Detector (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Enable': (<function boolean at 0x7f30f87f5938>, True)}
    resource_type = 'AWS::GuardDuty::Detector'

```

```
class troposphere.guardduty.Filter (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Action': (<type 'basestring'>, True), 'Description': (<type 'basestring'>,
    resource_type = 'AWS::GuardDuty::Filter'

class troposphere.guardduty.FindingCriteria (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Criterion': (<type 'dict'>, False), 'ItemType': (<class 'troposphere.guardduty.FindingCriteria.ItemType'>, True)

class troposphere.guardduty.IPSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Activate': (<function boolean at 0x7f30f87f5938>, True), 'DetectorId': (<type 'basestring'>, True),
    resource_type = 'AWS::GuardDuty::IPSet'

class troposphere.guardduty.Master (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DetectorId': (<type 'basestring'>, True), 'InvitationId': (<type 'basestring'>, True),
    resource_type = 'AWS::GuardDuty::Master'

class troposphere.guardduty.Member (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DetectorId': (<type 'basestring'>, True), 'DisableEmailNotification': (<type 'boolean'>, True),
    resource_type = 'AWS::GuardDuty::Member'

class troposphere.guardduty.ThreatIntelSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Activate': (<function boolean at 0x7f30f87f5938>, True), 'DetectorId': (<type 'basestring'>, True),
    resource_type = 'AWS::GuardDuty::ThreatIntelSet'
```

troposphere.iam module

```
class troposphere.iam.AccessKey (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Serial': (<function integer at 0x7f30f87f59b0>, False), 'Status': (<function boolean at 0x7f30f87f59b0>, True),
    resource_type = 'AWS::IAM::AccessKey'

class troposphere.iam.Group (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'GroupName': (<function iam_group_name at 0x7f30f87fb2a8>, False), 'ManagedPolicyArns': (<type 'list'>, True),
    resource_type = 'AWS::IAM::Group'

class troposphere.iam.InstanceProfile (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'InstanceProfileName': (<type 'basestring'>, False), 'Path': (<function iam_instance_profile_path at 0x7f30f87fb2a8>, True),
    resource_type = 'AWS::IAM::InstanceProfile'

class troposphere.iam.LoginProfile (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```



```

    props = {'Password': (<type 'basestring'>, True), 'PasswordResetRequired': (<function
class troposphere.iam.ManagedPolicy (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'Groups': ([<type 'basestring'>],
    resource_type = 'AWS::IAM::ManagedPolicy'
class troposphere.iam.Policy (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PolicyDocument': ((<type 'dict'>, <class 'awacs.aws.Policy'>), True), 'Poli
troposphere.iam.PolicyProperty
    alias of troposphere.iam.Policy
class troposphere.iam.PolicyType (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Groups': ([<type 'basestring'>], False), 'PolicyDocument': ((<type 'dict'>
    resource_type = 'AWS::IAM::Policy'
class troposphere.iam.Role (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AssumeRolePolicyDocument': ((<type 'dict'>, <class 'awacs.aws.Policy'>), Tr
    resource_type = 'AWS::IAM::Role'
class troposphere.iam.ServiceLinkedRole (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AWSServiceName': (<type 'basestring'>, True), 'CustomSuffix': (<type 'base
    resource_type = 'AWS::IAM::ServiceLinkedRole'
class troposphere.iam.User (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Groups': ([<type 'basestring'>], False), 'LoginProfile': (<class 'troposph
    resource_type = 'AWS::IAM::User'
class troposphere.iam.UserToGroupAddition (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'GroupName': (<type 'basestring'>, True), 'Users': (<type 'list'>, True)}
    resource_type = 'AWS::IAM::UserToGroupAddition'

```

troposphere.inspector module

```

class troposphere.inspector.AssessmentTarget (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'AssessmentTargetName': (<type 'basestring'>, False), 'ResourceGroupArn': (
    resource_type = 'AWS::Inspector::AssessmentTarget'
class troposphere.inspector.AssessmentTemplate (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject

```

```
    props = {'AssessmentTargetArn': (<type 'basestring'>, True), 'AssessmentTemplateName':
    resource_type = 'AWS::Inspector::AssessmentTemplate'
class troposphere.inspector.ResourceGroup (title, template=None, validation=True,
                                           **kwargs)
    Bases: troposphere.AWSObject
    props = {'ResourceGroupTags': (<class 'troposphere.Tags'>, True)}
    resource_type = 'AWS::Inspector::ResourceGroup'
```

troposphere.iot module

```
class troposphere.iot.Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CloudwatchAlarm': (<class 'troposphere.iot.CloudwatchAlarmAction'>, False),
class troposphere.iot.Certificate (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'CertificateSigningRequest': (<type 'basestring'>, True), 'Status': (<type
    resource_type = 'AWS::IoT::Certificate'
class troposphere.iot.CloudwatchAlarmAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AlarmName': (<type 'basestring'>, True), 'RoleArn': (<type 'basestring'>,
class troposphere.iot.CloudwatchMetricAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MetricName': (<type 'basestring'>, True), 'MetricNamespace': (<type 'bases
class troposphere.iot.DynamoDBAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'HashKeyField': (<type 'basestring'>, True), 'HashKeyType': (<type 'basestr
class troposphere.iot.DynamoDBv2Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PutItem': (<class 'troposphere.iot.PutItemInput'>, False), 'RoleArn': (<ty
class troposphere.iot.ElasticsearchAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Endpoint': (<type 'basestring'>, True), 'Id': (<type 'basestring'>, True),
class troposphere.iot.FirehoseAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DeliveryStreamName': (<type 'basestring'>, True), 'RoleArn': (<type 'bases
class troposphere.iot.KinesisAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PartitionKey': (<type 'basestring'>, False), 'RoleArn': (<type 'basestring
class troposphere.iot.LambdaAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'FunctionArn': (<type 'basestring'>, True)}
```

```

class troposphere.iot.Policy (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'PolicyDocument': ((<type 'dict'>, <class 'awacs.aws.Policy'>), True), 'PolicyName': (<type 'basestring'>, True), 'Resource': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, True)}
    resource_type = 'AWS::IoT::Policy'

class troposphere.iot.PolicyPrincipalAttachment (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'PolicyName': (<type 'basestring'>, True), 'Principal': (<type 'basestring'>, True), 'Resource': (<type 'basestring'>, True), 'ResourceType': (<type 'basestring'>, True)}
    resource_type = 'AWS::IoT::PolicyPrincipalAttachment'

class troposphere.iot.PutItemInput (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'TableName': (<type 'basestring'>, True)}

class troposphere.iot.RepublishAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'RoleArn': (<type 'basestring'>, True), 'Topic': (<type 'basestring'>, True), 'TopicArn': (<type 'basestring'>, True)}

class troposphere.iot.S3Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BucketName': (<type 'basestring'>, True), 'Key': (<type 'basestring'>, True), 'RoleArn': (<type 'basestring'>, True)}

class troposphere.iot.SnsAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MessageFormat': (<type 'basestring'>, False), 'RoleArn': (<type 'basestring'>, True), 'Topic': (<type 'basestring'>, True), 'TopicArn': (<type 'basestring'>, True)}

class troposphere.iot.SqsAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'QueueUrl': (<type 'basestring'>, True), 'RoleArn': (<type 'basestring'>, True), 'Topic': (<type 'basestring'>, True), 'TopicArn': (<type 'basestring'>, True)}

class troposphere.iot.Thing (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AttributePayload': (<type 'dict'>, False), 'ThingName': (<type 'basestring'>, True), 'ThingType': (<type 'basestring'>, True)}
    resource_type = 'AWS::IoT::Thing'

class troposphere.iot.ThingPrincipalAttachment (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Principal': (<type 'basestring'>, True), 'ThingName': (<type 'basestring'>, True), 'ThingType': (<type 'basestring'>, True)}
    resource_type = 'AWS::IoT::ThingPrincipalAttachment'

class troposphere.iot.TopicRule (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'RuleName': (<type 'basestring'>, False), 'TopicRulePayload': (<class 'troposphere.iot.TopicRulePayload'>, True)}
    resource_type = 'AWS::IoT::TopicRule'

class troposphere.iot.TopicRulePayload (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Actions': ([<class 'troposphere.iot.Action'>], True), 'AwsIotSqlVersion': (<type 'basestring'>, True)}

```

troposphere.iot1click module

```
class troposphere.iot1click.Device (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DeviceId': (<type 'basestring'>, True), 'Enabled': (<function boolean at 0x7f30f87fb500>, True), 'ResourceType': ('AWS::IoT1Click::Device',)}
    resource_type = ('AWS::IoT1Click::Device',)

class troposphere.iot1click.Placement (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AssociatedDevices': (<function json_checker at 0x7f30f87fb500>, False), 'PlacementTemplate': (<class 'troposphere.iot1click.PlacementTemplate'>, True), 'ResourceType': ('AWS::IoT1Click::Placement',)}
    resource_type = 'AWS::IoT1Click::Placement'

class troposphere.iot1click.PlacementTemplate (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DefaultAttributes': (<function json_checker at 0x7f30f87fb500>, False), 'PlacementTemplate': (<class 'troposphere.iot1click.PlacementTemplate'>, True), 'ResourceType': ('AWS::IoT1Click::PlacementTemplate',)}
    resource_type = 'AWS::IoT1Click::PlacementTemplate'

class troposphere.iot1click.Project (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'PlacementTemplate': (<class 'troposphere.iot1click.PlacementTemplate'>, True), 'ResourceType': ('AWS::IoT1Click::Project',)}
    resource_type = 'AWS::IoT1Click::Project'
```

troposphere.iotanalytics module

```
class troposphere.iotanalytics.Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ActionName': (<type 'basestring'>, True), 'ContainerAction': (<class 'troposphere.iotanalytics.ContainerAction'>, True), 'ResourceType': ('AWS::IoTAnalytics::Action',)}
    resource_type = 'AWS::IoTAnalytics::Action'

class troposphere.iotanalytics.Activity (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AddAttributes': (<class 'troposphere.iotanalytics.AddAttributes'>, True), 'Channel': (<class 'troposphere.iotanalytics.Channel'>, True), 'ResourceType': ('AWS::IoTAnalytics::Activity',)}
    resource_type = 'AWS::IoTAnalytics::Activity'

class troposphere.iotanalytics.ActivityChannel (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ChannelName': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, True), 'ResourceType': ('AWS::IoTAnalytics::ActivityChannel',)}
    resource_type = 'AWS::IoTAnalytics::ActivityChannel'

class troposphere.iotanalytics.AddAttributes (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Attributes': (<function json_checker at 0x7f30f87fb500>, False), 'Name': (<type 'basestring'>, True), 'ResourceType': ('AWS::IoTAnalytics::AddAttributes',)}
    resource_type = 'AWS::IoTAnalytics::AddAttributes'

class troposphere.iotanalytics.Channel (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ChannelName': (<type 'basestring'>, False), 'RetentionPeriod': (<class 'troposphere.iotanalytics.RetentionPeriod'>, True), 'ResourceType': ('AWS::IoTAnalytics::Channel',)}
    resource_type = 'AWS::IoTAnalytics::Channel'

class troposphere.iotanalytics.ContainerAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ExecutionRoleArn': (<type 'basestring'>, True), 'Image': (<type 'basestring'>, True), 'ResourceType': ('AWS::IoTAnalytics::ContainerAction',)}
    resource_type = 'AWS::IoTAnalytics::ContainerAction'

class troposphere.iotanalytics.Dataset (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
```

```

    props = {'Actions': ([<class 'troposphere.iotanalytics.Action'>], True), 'DatasetName':
    resource_type = 'AWS::IoTAnalytics::Dataset'

class troposphere.iotanalytics.DatasetContentVersionValue (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'DatasetName': (<type 'basestring'>, False)}

class troposphere.iotanalytics.Datastore (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject

    props = {'DatastoreName': (<type 'basestring'>, False), 'RetentionPeriod': (<class '
    resource_type = 'AWS::IoTAnalytics::Datastore'

class troposphere.iotanalytics.DeltaTime (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'OffsetSeconds': (<function integer at 0x7f30f87f59b0>, True), 'TimeExpressi

class troposphere.iotanalytics.DeviceRegistryEnrich (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Attribute': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, Fa

class troposphere.iotanalytics.DeviceShadowEnrich (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Attribute': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, Fa

class troposphere.iotanalytics.Filter (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Filter': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, False

class troposphere.iotanalytics.Lambda (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'BatchSize': (<function integer at 0x7f30f87f59b0>, False), 'LambdaName': (

class troposphere.iotanalytics.Math (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Attribute': (<type 'basestring'>, False), 'Math': (<type 'basestring'>, Fa

class troposphere.iotanalytics.OutputFileUriValue (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'FileName': (<type 'basestring'>, False)}

class troposphere.iotanalytics.Pipeline (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'PipelineActivities': ([<class 'troposphere.iotanalytics.Activity'>], True),
    resource_type = 'AWS::IoTAnalytics::Pipeline'

class troposphere.iotanalytics.QueryAction (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Filters': ([<class 'troposphere.iotanalytics.QueryActionFilter'>], False),

class troposphere.iotanalytics.QueryActionFilter (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'DeltaTime': (<class 'troposphere.iotanalytics.DeltaTime'>, False)}

```

```
class troposphere.iotanalytics.RemoveAttributes (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Attributes': ([<type 'basestring'>], False), 'Name': (<type 'basestring'>,
class troposphere.iotanalytics.ResourceConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ComputeType': (<type 'basestring'>, True), 'VolumeSizeInGB': (<function int
class troposphere.iotanalytics.RetentionPeriod (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'NumberOfDays': (<function integer at 0x7f30f87f59b0>, False), 'Unlimited':
class troposphere.iotanalytics.Schedule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ScheduleExpression': (<type 'basestring'>, True)}
class troposphere.iotanalytics.SelectAttributes (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Attributes': ([<type 'basestring'>], False), 'Name': (<type 'basestring'>,
class troposphere.iotanalytics.Trigger (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Schedule': (<class 'troposphere.iotanalytics.Schedule'>, False), 'Triggering
class troposphere.iotanalytics.TriggeringDataset (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DatasetName': (<type 'basestring'>, True)}
class troposphere.iotanalytics.Variable (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DatasetContentVersionValue': (<class 'troposphere.iotanalytics.DatasetConte
```

troposphere.kinesis module

```
class troposphere.kinesis.Stream (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Name': (<type 'basestring'>, False), 'RetentionPeriodHours': (<function in
    resource_type = 'AWS::Kinesis::Stream'
class troposphere.kinesis.StreamConsumer (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ConsumerName': (<type 'basestring'>, True), 'StreamARN': (<type 'basestring'
    resource_type = 'AWS::Kinesis::StreamConsumer'
class troposphere.kinesis.StreamEncryption (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'EncryptionType': (<type 'basestring'>, True), 'KeyId': (<type 'basestring'
```

troposphere.kms module

```
class troposphere.kms.Alias (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'AliasName': (<type 'basestring'>, True), 'TargetKeyId': (<type 'basestring'
```

```
    resource_type = 'AWS::KMS::Alias'
```

```
class troposphere.kms.Key (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Description': (<type 'basestring'>, False), 'EnableKeyRotation': (<function
```

```
    resource_type = 'AWS::KMS::Key'
```

troposphere.logs module

```
class troposphere.logs.Destination (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'DestinationName': (<type 'basestring'>, True), 'DestinationPolicy': (<type
```

```
    resource_type = 'AWS::Logs::Destination'
```

```
class troposphere.logs.LogGroup (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'LogGroupName': (<type 'basestring'>, False), 'RetentionInDays': (<function
```

```
    resource_type = 'AWS::Logs::LogGroup'
```

```
class troposphere.logs.LogStream (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'LogGroupName': (<type 'basestring'>, True), 'LogStreamName': (<type 'bases
```

```
    resource_type = 'AWS::Logs::LogStream'
```

```
class troposphere.logs.MetricFilter (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'FilterPattern': (<type 'basestring'>, True), 'LogGroupName': (<type 'bases
```

```
    resource_type = 'AWS::Logs::MetricFilter'
```

```
class troposphere.logs.MetricTransformation (title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'DefaultValue': (<type 'float'>, False), 'MetricName': (<type 'basestring'>
```

```
class troposphere.logs.SubscriptionFilter (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'DestinationArn': (<type 'basestring'>, True), 'FilterPattern': (<type 'bas
```

```
    resource_type = 'AWS::Logs::SubscriptionFilter'
```

troposphere.neptune module

```
class troposphere.neptune.DBCluster (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'AvailabilityZones': ([<type 'basestring'>], False), 'BackupRetentionPeriod':
    resource_type = 'AWS::Neptune::DBCluster'
class troposphere.neptune.DBClusterParameterGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, True), 'Family': (<type 'basestring'>,
    resource_type = 'AWS::Neptune::DBClusterParameterGroup'
class troposphere.neptune.DBInstance (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AllowMajorVersionUpgrade': (<function boolean at 0x7f30f87f5938>, False),
    resource_type = 'AWS::Neptune::DBInstance'
class troposphere.neptune.DBParameterGroup (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, True), 'Family': (<type 'basestring'>,
    resource_type = 'AWS::Neptune::DBParameterGroup'
class troposphere.neptune.DBSubnetGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DBSubnetGroupDescription': (<type 'basestring'>, True), 'DBSubnetGroupName':
    resource_type = 'AWS::Neptune::DBSubnetGroup'
```

troposphere.opsworks module

```
class troposphere.opsworks.App (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AppSource': (<class 'troposphere.opsworks.Source'>, False), 'Attributes':
    resource_type = 'AWS::OpsWorks::App'
class troposphere.opsworks.AutoScalingThresholds (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CpuThreshold': (<type 'float'>, False), 'IgnoreMetricsTime': (<function in
class troposphere.opsworks.BlockDeviceMapping (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DeviceName': (<type 'basestring'>, False), 'Ebs': (<class 'troposphere.ops
    validate()
class troposphere.opsworks.ChefConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'BerkshelfVersion': (<type 'basestring'>, False), 'ManageBerkshelf': (<func
class troposphere.opsworks.DataSource (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Arn': (<type 'basestring'>, False), 'DatabaseName': (<type 'basestring'>,
```



```

class troposphere.opsworks.EbsBlockDevice (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DeleteOnTermination': (<function boolean at 0x7f30f87f5938>, False), 'Iops'

class troposphere.opsworks.ElasticIp (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Ip': (<type 'basestring'>, True), 'Name': (<type 'basestring'>, False)}

class troposphere.opsworks.ElasticLoadBalancerAttachment (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ElasticLoadBalancerName': (<type 'basestring'>, True), 'LayerId': (<type 'basestring'>, True)}
    resource_type = 'AWS::OpsWorks::ElasticLoadBalancerAttachment'

class troposphere.opsworks.EngineAttribute (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, False), 'Value': (<type 'basestring'>, False)}

class troposphere.opsworks.Environment (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Key': (<type 'basestring'>, True), 'Secure': (<type 'bool'>, False), 'Value': (<type 'basestring'>, True)}

class troposphere.opsworks.Instance (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AgentVersion': (<type 'basestring'>, False), 'AmiId': (<type 'basestring'>, True)}
    resource_type = 'AWS::OpsWorks::Instance'

class troposphere.opsworks.Layer (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Attributes': (<type 'dict'>, False), 'AutoAssignElasticIps': (<function boolean at 0x7f30f87f5938>, True)}
    resource_type = 'AWS::OpsWorks::Layer'

class troposphere.opsworks.LifecycleConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ShutdownEventConfiguration': (<class 'troposphere.opsworks.ShutdownEventConfiguration'>, True)}

class troposphere.opsworks.LoadBasedAutoScaling (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DownScaling': (<class 'troposphere.opsworks.AutoScalingThresholds'>, False)}

class troposphere.opsworks.RdsDbInstance (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DbPassword': (<type 'basestring'>, True), 'DbUser': (<type 'basestring'>, True)}

class troposphere.opsworks.Recipes (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Configure': ([<type 'basestring'>], False), 'Deploy': ([<type 'basestring'>], True)}

class troposphere.opsworks.Server (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'BackupId': (<type 'basestring'>, False), 'BackupRetentionCount': (<function boolean at 0x7f30f87f5938>, True)}
    resource_type = 'AWS::OpsWorksCM::Server'

```

```
class troposphere.opsworks.ShutdownEventConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DelayUntilElbConnectionsDrained': (<function boolean at 0x7f30f87f5938>, False)}

class troposphere.opsworks.Source (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Password': (<type 'basestring'>, False), 'Revision': (<type 'basestring'>, False)}

class troposphere.opsworks.SslConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Certificate': (<type 'basestring'>, True), 'Chain': (<type 'basestring'>, False)}

class troposphere.opsworks.Stack (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AgentVersion': (<type 'basestring'>, False), 'Attributes': (<type 'dict'>, {}),
            'resource_type': 'AWS::OpsWorks::Stack'}

    validate()

class troposphere.opsworks.StackConfigurationManager (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, False), 'Version': (<type 'basestring'>, False)}

class troposphere.opsworks.TimeBasedAutoScaling (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Friday': (<type 'dict'>, False), 'Monday': (<type 'dict'>, False), 'Saturday': (<type 'dict'>, False),
            'Sunday': (<type 'dict'>, False), 'Thursday': (<type 'dict'>, False), 'Wednesday': (<type 'dict'>, False),
            'Tuesday': (<type 'dict'>, False)}

class troposphere.opsworks.UserProfile (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AllowSelfManagement': (<function boolean at 0x7f30f87f5938>, False), 'IamUser': (<type 'basestring'>, False)}

    resource_type = 'AWS::OpsWorks::UserProfile'

class troposphere.opsworks.Volume (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Ec2VolumeId': (<type 'basestring'>, True), 'MountPoint': (<type 'basestring'>, False)}

    resource_type = 'AWS::OpsWorks::Volume'

class troposphere.opsworks.VolumeConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Encrypted': (<function boolean at 0x7f30f87f5938>, False), 'Iops': (<function integer at 0x7f30f87f5938>, False)}

    validate()

troposphere.opsworks.validate_data_source_type (data_source_type)

troposphere.opsworks.validate_volume_type (volume_type)
```

troposphere.policies module

```
class troposphere.policies.AutoScalingCreationPolicy (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'MinSuccessfulInstancesPercent': (<function integer at 0x7f30f87f59b0>, False)}
```

```

class troposphere.policies.AutoScalingReplacingUpdate (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'WillReplace': (<function boolean at 0x7f30f87f5938>, False)}

class troposphere.policies.AutoScalingRollingUpdate (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'MaxBatchSize': (<function positive_integer at 0x7f30f87f5a28>, False), 'Min

class troposphere.policies.AutoScalingScheduledAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'IgnoreUnmodifiedGroupSizeProperties': (<function boolean at 0x7f30f87f5938>

class troposphere.policies.CodeDeployLambdaAliasUpdate (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AfterAllowTrafficHook': (<type 'basestring'>, False), 'ApplicationName': (

class troposphere.policies.CreationPolicy (title=None, **kwargs)
    Bases: troposphere.AWSAttribute

    props = {'AutoScalingCreationPolicy': (<class 'troposphere.policies.AutoScalingCreati

class troposphere.policies.ResourceSignal (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Count': (<function positive_integer at 0x7f30f87f5a28>, False), 'Timeout':

class troposphere.policies.UpdatePolicy (title=None, **kwargs)
    Bases: troposphere.AWSAttribute

    props = {'AutoScalingReplacingUpdate': (<class 'troposphere.policies.AutoScalingRepla

```

troposphere.rds module

```

class troposphere.rds.DBCluster (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AvailabilityZones': ([<type 'basestring'>], False), 'BacktrackWindow': (<f
    resource_type = 'AWS::RDS::DBCluster'

class troposphere.rds.DBClusterParameterGroup (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, True), 'Family': (<type 'basestring'>,
    resource_type = 'AWS::RDS::DBClusterParameterGroup'

class troposphere.rds.DBInstance (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AllocatedStorage': (<function positive_integer at 0x7f30f87f5a28>, False),
    resource_type = 'AWS::RDS::DBInstance'

    validate ()

class troposphere.rds.DBParameterGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'Family': (<type 'basestring'>

```

```
    resource_type = 'AWS::RDS::DBParameterGroup'
class troposphere.rds.DBSecurityGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DBSecurityGroupIngress': (<type 'list'>, True), 'EC2VpcId': (<type 'basestr
    resource_type = 'AWS::RDS::DBSecurityGroup'
class troposphere.rds.DBSecurityGroupIngress (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'CIDRIP': (<type 'basestring'>, False), 'DBSecurityGroupName': (<type 'bases
    resource_type = 'AWS::RDS::DBSecurityGroupIngress'
class troposphere.rds.DBSubnetGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DBSubnetGroupDescription': (<type 'basestring'>, True), 'DBSubnetGroupName'
    resource_type = 'AWS::RDS::DBSubnetGroup'
class troposphere.rds.EventSubscription (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Enabled': (<function boolean at 0x7f30f87f5938>, False), 'EventCategories':
    resource_type = 'AWS::RDS::EventSubscription'
class troposphere.rds.OptionConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DBSecurityGroupMemberships': ([<type 'basestring'>], False), 'OptionName':
class troposphere.rds.OptionGroup (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'EngineName': (<type 'basestring'>, True), 'MajorEngineVersion': (<type 'ba
    resource_type = 'AWS::RDS::OptionGroup'
class troposphere.rds.OptionSetting (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Name': (<type 'basestring'>, False), 'Value': (<type 'basestring'>, False)
class troposphere.rds.ProcessorFeature (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Name': (<type 'basestring'>, False), 'Value': (<type 'basestring'>, False)
class troposphere.rds.RDSecurityGroup (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'CIDRIP': (<type 'basestring'>, False), 'EC2SecurityGroupId': (<type 'basestr
class troposphere.rds.ScalingConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AutoPause': (<function boolean at 0x7f30f87f5938>, False), 'MaxCapacity':
troposphere.rds.validate_backup_retention_period (days)
    Validate BackupRetentionPeriod for DBInstance
troposphere.rds.validate_backup_window (window)
    Validate PreferredBackupWindow for DBInstance
```

```

troposphere.rds.validate_capacity (capacity)
    Validate ScalingConfiguration capacity for serverless DBCluster

troposphere.rds.validate_engine (engine)
    Validate database Engine for DBInstance

troposphere.rds.validate_engine_mode (engine_mode)
    Validate database EngineMode for DBCluster

troposphere.rds.validate_iops (iops)
    DBInstance Iops validation rules.

troposphere.rds.validate_license_model (license_model)
    Validate LicenseModel for DBInstance

troposphere.rds.validate_maintenance_window (window)
    Validate PreferredMaintenanceWindow for DBInstance

troposphere.rds.validate_storage_type (storage_type)
    Validate StorageType for DBInstance

```

troposphere.redshift module

```

class troposphere.redshift.AmazonRedshiftParameter (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ParameterName': (<type 'basestring'>, True), 'ParameterValue': (<type 'bas
class troposphere.redshift.Cluster (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'AllowVersionUpgrade': (<function boolean at 0x7f30f87f5938>, False), 'Autom
    resource_type = 'AWS::Redshift::Cluster'

class troposphere.redshift.ClusterParameterGroup (title, template=None, valida-
                                                tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, True), 'ParameterGroupFamily': (<type
    resource_type = 'AWS::Redshift::ClusterParameterGroup'

class troposphere.redshift.ClusterSecurityGroup (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, True), 'Tags': (<class 'troposphere.Ta
    resource_type = 'AWS::Redshift::ClusterSecurityGroup'

class troposphere.redshift.ClusterSecurityGroupIngress (title, template=None, vali-
                                                dation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'CIDRIP': (<type 'basestring'>, False), 'ClusterSecurityGroupName': (<type '
    resource_type = 'AWS::Redshift::ClusterSecurityGroupIngress'

class troposphere.redshift.ClusterSubnetGroup (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, True), 'SubnetIds': (<type 'list'>, Tr

```

```
resource_type = 'AWS::Redshift::ClusterSubnetGroup'
```

```
class troposphere.redshift.LoggingProperties (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'BucketName': (<type 'basestring'>, True), 'S3KeyPrefix': (<type 'basestrin
```

troposphere.route53 module

```
class troposphere.route53.AlarmIdentifier (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'Name': (<type 'basestring'>, True), 'Region': (<type 'basestring'>, True)}
```

```
class troposphere.route53.AliasTarget (hostedzoneid=None, dnsname=None, evaluatetar-
                                         gethealth=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'DNSName': (<type 'basestring'>, True), 'EvaluateTargetHealth': (<function I
```

```
class troposphere.route53.BaseRecordSet
    Bases: object
```

```
    props = {'AliasTarget': (<class 'troposphere.route53.AliasTarget'>, False), 'Comment'
```

```
class troposphere.route53.GeoLocation (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'ContinentCode': (<type 'basestring'>, False), 'CountryCode': (<type 'bases
```

```
class troposphere.route53.HealthCheck (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
```

```
    props = {'HealthCheckConfig': (<class 'troposphere.route53.HealthCheckConfiguration'>
    resource_type = 'AWS::Route53::HealthCheck'
```

```
class troposphere.route53.HealthCheckConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'AlarmIdentifier': (<class 'troposphere.route53.AlarmIdentifier'>, False), 'C
```

```
class troposphere.route53.HostedZone (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
```

```
    props = {'HostedZoneConfig': (<class 'troposphere.route53.HostedZoneConfiguration'>,
    resource_type = 'AWS::Route53::HostedZone'
```

```
class troposphere.route53.HostedZoneConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'Comment': (<type 'basestring'>, False)}
```

```
class troposphere.route53.HostedZoneVPCs (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'VPCId': (<type 'basestring'>, True), 'VPCRegion': (<type 'basestring'>, Tr
```

```
class troposphere.route53.IpAddressRequest (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'Ip': (<type 'basestring'>, False), 'SubnetId': (<type 'basestring'>, True)
```

```

class troposphere.route53.QueryLoggingConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'CloudWatchLogsLogGroupArn': (<type 'basestring'>, True)}

class troposphere.route53.RecordSet (title=None, **kwargs)
    Bases: troposphere.AWSProperty, troposphere.route53.BaseRecordSet

class troposphere.route53.RecordSetGroup (title, template=None, validation=True,
                                           **kwargs)
    Bases: troposphere.AWSObject

    props = {'Comment': (<type 'basestring'>, False), 'HostedZoneId': (<type 'basestring'>, True),
             'ResourceRecordTypes': (<type 'basestring'>, True), 'ResourceRecords': (<type 'list'>, True),
             'ResourceType': (<type 'basestring'>, True)}
    resource_type = 'AWS::Route53::RecordSetGroup'

class troposphere.route53.RecordSetType (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject, troposphere.route53.BaseRecordSet

    resource_type = 'AWS::Route53::RecordSet'

class troposphere.route53.ResolverEndpoint (title, template=None, validation=True,
                                           **kwargs)
    Bases: troposphere.AWSObject

    props = {'Direction': (<type 'basestring'>, True), 'IpAddresses': ([<class 'troposphere.route53.IpAddress'>, True),
             'ResourceRecordTypes': (<type 'basestring'>, True), 'ResourceRecords': (<type 'list'>, True),
             'ResourceType': (<type 'basestring'>, True)}
    resource_type = 'AWS::Route53Resolver::ResolverEndpoint'

class troposphere.route53.ResolverRule (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DomainName': (<type 'basestring'>, True), 'Name': (<type 'basestring'>, False), 'ResourceRecordTypes': (<type 'basestring'>, True),
             'ResourceRecords': (<type 'list'>, True), 'ResourceType': (<type 'basestring'>, True)}
    resource_type = 'AWS::Route53Resolver::ResolverRule'

class troposphere.route53.ResolverRuleAssociation (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Name': (<type 'basestring'>, False), 'ResolverRuleId': (<type 'basestring'>, True), 'TargetId': (<type 'basestring'>, True)}
    resource_type = 'AWS::Route53Resolver::ResolverRuleAssociation'

class troposphere.route53.TargetAddress (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Ip': (<type 'basestring'>, True), 'Port': (<type 'basestring'>, True)}

troposphere.route53.validate_ruletype (ruletype)
    Validate RuleType for ResolverRule.

```

troposphere.s3 module

```

class troposphere.s3.AbortIncompleteMultipartUpload (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'DaysAfterInitiation': (<function positive_integer at 0x7f30f87f5a28>, True)}

class troposphere.s3.AccelerateConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AccelerationStatus': (<function s3_transfer_acceleration_status at 0x7f30f87f5a28>, True)}

```

```
class troposphere.s3.AccessControlTranslation (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Owner': (<type 'basestring'>, True)}

class troposphere.s3.AnalyticsConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Id': (<type 'basestring'>, True), 'Prefix': (<type 'basestring'>, False),

class troposphere.s3.Bucket (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    access_control_types = ['Private', 'PublicRead', 'PublicReadWrite', 'AuthenticatedRead']
    props = {'AccelerateConfiguration': (<class 'troposphere.s3.AccelerateConfiguration'>,
    resource_type = 'AWS::S3::Bucket'

    validate()

class troposphere.s3.BucketEncryption (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'ServerSideEncryptionConfiguration': ([<class 'troposphere.s3.ServerSideEncryptionConfiguration'>], True)}

class troposphere.s3.BucketPolicy (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Bucket': (<type 'basestring'>, True), 'PolicyDocument': ((<type 'dict'>, <type 'dict'>), True)}
    resource_type = 'AWS::S3::BucketPolicy'

class troposphere.s3.CorsConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'CorsRules': ([<class 'troposphere.s3.CorsRules'>], True)}

class troposphere.s3.CorsRules (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'AllowedHeaders': ([<type 'basestring'>], False), 'AllowedMethods': ([<type 'basestring'>], False)}

class troposphere.s3.DataExport (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Destination': (<class 'troposphere.s3.Destination'>, True), 'OutputSchemaVersion': (<type 'basestring'>, True)}

class troposphere.s3.Destination (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'BucketAccountId': (<type 'basestring'>, False), 'BucketArn': (<type 'basestring'>, True)}

class troposphere.s3.EncryptionConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'ReplicaKmsKeyID': (<type 'basestring'>, True)}

class troposphere.s3.Filter (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'S3Key': (<class 'troposphere.s3.S3Key'>, True)}

class troposphere.s3.InventoryConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Destination': (<class 'troposphere.s3.Destination'>, True), 'Enabled': (<type 'boolean'>, True)}
```



```

class troposphere.s3.LambdaConfigurations (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Event': (<type 'basestring'>, True), 'Filter': (<class 'troposphere.s3.Fil

class troposphere.s3.LifecycleConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Rules': ([<class 'troposphere.s3.LifecycleRule'>], True)}

class troposphere.s3.LifecycleRule (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'AbortIncompleteMultipartUpload': (<class 'troposphere.s3.AbortIncompleteMul
    validate()

class troposphere.s3.LifecycleRuleTransition (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'StorageClass': (<type 'basestring'>, True), 'TransitionDate': (<type 'base

class troposphere.s3.LoggingConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DestinationBucketName': (<function s3_bucket_name at 0x7f30f87f5de8>, False

class troposphere.s3.MetricsConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Id': (<type 'basestring'>, True), 'Prefix': (<type 'basestring'>, False),

class troposphere.s3.NoncurrentVersionTransition (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'StorageClass': (<type 'basestring'>, True), 'TransitionInDays': (<function

class troposphere.s3.NotificationConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'LambdaConfigurations': ([<class 'troposphere.s3.LambdaConfigurations'>], Fa

class troposphere.s3.PublicAccessBlockConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'BlockPublicAcls': (<function boolean at 0x7f30f87f5938>, False), 'BlockPubl

class troposphere.s3.QueueConfigurations (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Event': (<type 'basestring'>, True), 'Filter': (<class 'troposphere.s3.Fil

class troposphere.s3.RedirectAllRequestsTo (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'HostName': (<type 'basestring'>, True), 'Protocol': (<type 'basestring'>,

class troposphere.s3.RedirectRule (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'HostName': (<type 'basestring'>, False), 'HttpRedirectCode': (<type 'bases

class troposphere.s3.ReplicationConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Role': (<type 'basestring'>, True), 'Rules': ([<class 'troposphere.s3.Repl

```

```
class troposphere.s3.ReplicationConfigurationRules (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Destination': (<class 'troposphere.s3.ReplicationConfigurationRulesDestination'>, True)}

class troposphere.s3.ReplicationConfigurationRulesDestination (title=None,
                                                                **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'AccessControlTranslation': (<class 'troposphere.s3.AccessControlTranslation'>, True)}

class troposphere.s3.RoutingRule (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'RedirectRule': (<class 'troposphere.s3.RedirectRule'>, True), 'RoutingRuleCondition': (<class 'troposphere.s3.RoutingRuleCondition'>, True)}

class troposphere.s3.RoutingRuleCondition (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'HttpErrorCodeReturnedEquals': (<type 'basestring'>, False), 'KeyPrefixEquals': (<type 'basestring'>, True)}

class troposphere.s3.Rules (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Name': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}

class troposphere.s3.S3Key (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Rules': ([<class 'troposphere.s3.Rules'>], True)}

class troposphere.s3.ServerSideEncryptionByDefault (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'KMSMasterKeyID': (<type 'basestring'>, False), 'SSEAlgorithm': (<type 'basestring'>, True)}

class troposphere.s3.ServerSideEncryptionRule (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'ServerSideEncryptionByDefault': (<class 'troposphere.s3.ServerSideEncryptionByDefault'>, True)}

class troposphere.s3.SourceSelectionCriteria (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'SseKmsEncryptedObjects': (<class 'troposphere.s3.SseKmsEncryptedObjects'>, True)}

class troposphere.s3.SseKmsEncryptedObjects (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Status': (<type 'basestring'>, True)}

class troposphere.s3.StorageClassAnalysis (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DataExport': (<class 'troposphere.s3.DataExport'>, False)}

class troposphere.s3.TagFilter (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Key': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}

class troposphere.s3.TopicConfigurations (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Event': (<type 'basestring'>, True), 'Filter': (<class 'troposphere.s3.Filter'>, True)}
```

```
class troposphere.s3.VersioningConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'Status': (<type 'basestring'>, False)}
```

```
class troposphere.s3.WebsiteConfiguration (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'ErrorDocument': (<type 'basestring'>, False), 'IndexDocument': (<type 'bas
```

troposphere.sagemaker module

```
class troposphere.sagemaker.ContainerDefinition (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'ContainerHostname': (<type 'basestring'>, False), 'Environment': (<type 'd
```

```
class troposphere.sagemaker.Endpoint (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
```

```
    props = {'EndpointConfigName': (<type 'basestring'>, True), 'EndpointName': (<type '1
    resource_type = 'AWS::SageMaker::Endpoint'
```

```
class troposphere.sagemaker.EndpointConfig (title, template=None, validation=True,
                                             **kwargs)
    Bases: troposphere.AWSObject
```

```
    props = {'EndpointConfigName': (<type 'basestring'>, False), 'KmsKeyId': (<type 'bas
    resource_type = 'AWS::SageMaker::EndpointConfig'
```

```
class troposphere.sagemaker.Model (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
```

```
    props = {'Containers': ([<class 'troposphere.sagemaker.ContainerDefinition'>], False)
    resource_type = 'AWS::SageMaker::Model'
```

```
class troposphere.sagemaker.NotebookInstance (title, template=None, validation=True,
                                             **kwargs)
    Bases: troposphere.AWSObject
```

```
    props = {'DirectInternetAccess': (<type 'basestring'>, False), 'InstanceType': (<typ
    resource_type = 'AWS::SageMaker::NotebookInstance'
```

```
class troposphere.sagemaker.NotebookInstanceLifecycleConfig (title,
                                                             tem-
                                                             plate=None, val-
                                                             idation=True,
                                                             **kwargs)
    Bases: troposphere.AWSObject
```

```
    props = {'NotebookInstanceLifecycleConfigName': (<type 'basestring'>, False), 'OnCreat
    resource_type = 'AWS::SageMaker::NotebookInstanceLifecycleConfig'
```

```
class troposphere.sagemaker.NotebookInstanceLifecycleHook (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'Content': (<type 'basestring'>, False)}
```

```
class troposphere.sagemaker.ProductionVariant (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
    props = {'InitialInstanceCount': (<function integer at 0x7f30f87f59b0>, True), 'Initi
class troposphere.sagemaker.VpcConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'SecurityGroupIds': ([<type 'basestring'>], True), 'Subnets': ([<type 'base
```

troposphere.sdb module

```
class troposphere.sdb.Domain (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {}
    resource_type = 'AWS::SDB::Domain'
```

troposphere.secretsmanager module

```
class troposphere.secretsmanager.GenerateSecretString (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ExcludeCharacters': (<type 'basestring'>, False), 'ExcludeLowercase': (<fun
class troposphere.secretsmanager.ResourcePolicy (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'ResourcePolicy': ((<type 'dict'>, <class 'awacs.aws.Policy'>), True), 'Secr
    resource_type = 'AWS::SecretsManager::ResourcePolicy'
class troposphere.secretsmanager.RotationRules (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'AutomaticallyAfterDays': (<function integer at 0x7f30f87f59b0>, False)}
class troposphere.secretsmanager.RotationSchedule (title, template=None, valida
    tion=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'RotationLambdaARN': (<type 'basestring'>, True), 'RotationRules': (<class '
    resource_type = 'AWS::SecretsManager::RotationSchedule'
class troposphere.secretsmanager.Secret (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'GenerateSecretString': (<clas
    resource_type = 'AWS::SecretsManager::Secret'
class troposphere.secretsmanager.SecretTargetAttachment (title, template=None, vali
    dation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'SecretId': (<type 'basestring'>, True), 'TargetId': (<type 'basestring'>, '
    resource_type = 'AWS::SecretsManager::SecretTargetAttachment'
troposphere.secretsmanager.validate_target_types (target_type)
    Target types validation rule.
```

troposphere.serverless module

```
class troposphere.serverless.AlexaSkillEvent (title, template=None, validation=True,
                                             **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {}
```

```
    resource_type = 'AlexaSkill'
```

```
class troposphere.serverless.Api (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'AccessLogSetting': (<class 'troposphere.apigateway.AccessLogSetting'>, False),
```

```
            resource_type = 'AWS::Serverless::Api'
```

```
    validate ()
```

```
class troposphere.serverless.ApiEvent (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Method': (<type 'basestring'>, True), 'Path': (<type 'basestring'>, True),
```

```
            resource_type = 'Api'
```

```
class troposphere.serverless.Auth (title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Authorizers': (<class 'troposphere.serverless.Authorizers'>, False), 'Default':
```

```
class troposphere.serverless.Authorizers (title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'CognitoAuth': (<class 'troposphere.serverless.CognitoAuth'>, False), 'Default':
```

```
class troposphere.serverless.CloudWatchEvent (title, template=None, validation=True,
                                             **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Input': (<type 'basestring'>, False), 'InputPath': (<type 'basestring'>, False),
```

```
            resource_type = 'CloudWatchEvent'
```

```
class troposphere.serverless.CognitoAuth (title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Identity': (<class 'troposphere.serverless.CognitoAuthIdentity'>, False), 'Type':
```

```
class troposphere.serverless.CognitoAuthIdentity (title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'Header': (<type 'basestring'>, False), 'ValidationExpression': (<type 'basestring'>,
```

```
class troposphere.serverless.Cors (title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'AllowCredentials': (<type 'basestring'>, False), 'AllowHeaders': (<type 'basestring'>,
```

```
class troposphere.serverless.DeadLetterQueue (title=None, **kwargs)
```

```
    Bases: troposphere.AWSProperty
```

```
    props = {'TargetArn': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, False),
```

```
    validate ()
```

```
class troposphere.serverless.DeploymentPreference (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Alarms': (<type 'list'>, False), 'Enabled': (<type 'bool'>, False), 'Hooks'

class troposphere.serverless.DynamoDBEvent (title, template=None, validation=True,
                                             **kwargs)
    Bases: troposphere.AWSObject
    props = {'BatchSize': (<function positive_integer at 0x7f30f87f5a28>, False), 'Startin
    resource_type = 'DynamoDB'

class troposphere.serverless.Function (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AutoPublishAlias': (<type 'basestring'>, False), 'CodeUri': ((<class 'trop
    resource_type = 'AWS::Serverless::Function'
    validate()

class troposphere.serverless.FunctionForPackaging (title, template=None, valida
                                             tion=True, **kwargs)
    Bases: troposphere.serverless.Function
    Render Function without requiring 'CodeUri'.
    This exception to the Function spec is for use with the cloudformation/sam package commands which add
    CodeUri automatically.
    props = {'AutoPublishAlias': (<type 'basestring'>, False), 'CodeUri': ((<class 'trop
    resource_type = 'AWS::Serverless::Function'
    validate()

class troposphere.serverless.Hooks (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PostTraffic': (<type 'basestring'>, False), 'PreTraffic': (<type 'basetri

class troposphere.serverless.IoTRuleEvent (title, template=None, validation=True,
                                             **kwargs)
    Bases: troposphere.AWSObject
    props = {'AwsIotSqlVersion': (<type 'basestring'>, False), 'Sql': (<type 'basestring
    resource_type = 'IoTRule'

class troposphere.serverless.KinesisEvent (title, template=None, validation=True,
                                             **kwargs)
    Bases: troposphere.AWSObject
    props = {'BatchSize': (<function positive_integer at 0x7f30f87f5a28>, False), 'Startin
    resource_type = 'Kinesis'

class troposphere.serverless.LambdaRequestAuth (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'FunctionArn': (<type 'basestring'>, False), 'FunctionInvokeRole': (<type '

class troposphere.serverless.LambdaRequestAuthIdentity (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Context': ([<type 'basestring'>], False), 'Headers': ([<type 'basestring'>
```

```

class troposphere.serverless.LambdaTokenAuth (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'FunctionArn': (<type 'basestring'>, False), 'FunctionInvokeRole': (<type 'basestring'>, False)}

class troposphere.serverless.LambdaTokenAuthIdentity (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Header': (<type 'basestring'>, False), 'ReauthorizeEvery': (<type 'basestring'>, False)}

class troposphere.serverless.LayerVersion (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'CompatibleRuntimes': ([<type 'basestring'>], False), 'ContentUri': (<type 'basestring'>, False)}
    resource_type = 'AWS::Serverless::LayerVersion'

class troposphere.serverless.PrimaryKey (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Name': (<type 'basestring'>, False), 'Type': (<function primary_key_type_validator at 0x7f30f87f5a28>, True)}

class troposphere.serverless.S3Event (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Bucket': (<type 'basestring'>, True), 'Events': (<type 'list'>, True), 'FilterPrefix': (<type 'basestring'>, False)}
    resource_type = 'S3'

class troposphere.serverless.S3Location (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Bucket': (<type 'basestring'>, True), 'Key': (<type 'basestring'>, True), 'Prefix': (<type 'basestring'>, False)}

class troposphere.serverless.SNSEvent (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Topic': (<type 'basestring'>, True)}
    resource_type = 'SNS'

class troposphere.serverless.SQSEvent (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'BatchSize': (<function positive_integer at 0x7f30f87f5a28>, True), 'Queue': (<type 'basestring'>, True)}
    resource_type = 'SQS'
    validate()

class troposphere.serverless.ScheduleEvent (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Input': (<type 'basestring'>, False), 'Schedule': (<type 'basestring'>, True)}
    resource_type = 'Schedule'

class troposphere.serverless.SimpleTable (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'PrimaryKey': (<class 'troposphere.serverless.PrimaryKey'>, False), 'ProvisionedThroughput': (<type 'integer'>, True)}
    resource_type = 'AWS::Serverless::SimpleTable'

troposphere.serverless.primary_key_type_validator(x)

```

```
troposphere.serverless.starting_position_validator(x)
```

troposphere.servicecatalog module

```
class troposphere.servicecatalog.AcceptedPortfolioShare(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AcceptLanguage': (<type 'basestring'>, False), 'PortfolioId': (<type 'basestring'>, False),  
resource_type = 'AWS::ServiceCatalog::AcceptedPortfolioShare'
```

```
class troposphere.servicecatalog.CloudFormationProduct(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AcceptLanguage': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False),  
resource_type = 'AWS::ServiceCatalog::CloudFormationProduct'
```

```
class troposphere.servicecatalog.CloudFormationProvisionedProduct(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AcceptLanguage': (<type 'basestring'>, False), 'NotificationArns': ([<type 'basestring'>, False]),  
resource_type = 'AWS::ServiceCatalog::CloudFormationProvisionedProduct'
```

```
class troposphere.servicecatalog.LaunchNotificationConstraint(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AcceptLanguage': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False),  
resource_type = 'AWS::ServiceCatalog::LaunchNotificationConstraint'
```

```
class troposphere.servicecatalog.LaunchRoleConstraint(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AcceptLanguage': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False),  
resource_type = 'AWS::ServiceCatalog::LaunchRoleConstraint'
```

```
class troposphere.servicecatalog.LaunchTemplateConstraint(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AcceptLanguage': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False),  
resource_type = 'AWS::ServiceCatalog::LaunchTemplateConstraint'
```

```
class troposphere.servicecatalog.Portfolio(title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.AWSObject*

```
props = {'AcceptLanguage': (<type 'basestring'>, False), 'Description': (<type 'basestring'>, False),  
resource_type = 'AWS::ServiceCatalog::Portfolio'
```



```

class troposphere.servicecatalog.PortfolioPrincipalAssociation(title,
                                                             template=None,
                                                             validation=True,
                                                             **kwargs)

    Bases: troposphere.AWSObject

    props = {'AcceptLanguage': (<type 'basestring'>, False), 'PortfolioId': (<type 'basestring'>, True),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceCatalog::PortfolioPrincipalAssociation'

class troposphere.servicecatalog.PortfolioProductAssociation(title,
                                                             template=None,
                                                             validation=True,
                                                             **kwargs)

    Bases: troposphere.AWSObject

    props = {'AcceptLanguage': (<type 'basestring'>, False), 'PortfolioId': (<type 'basestring'>, True),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceCatalog::PortfolioProductAssociation'

class troposphere.servicecatalog.PortfolioShare(title, template=None, validation=True,
                                                **kwargs)

    Bases: troposphere.AWSObject

    props = {'AcceptLanguage': (<type 'basestring'>, False), 'AccountId': (<type 'basestring'>, True),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceCatalog::PortfolioShare'

class troposphere.servicecatalog.ProvisioningArtifactProperties(title=None,
                                                                **kwargs)

    Bases: troposphere.AWSPROPERTY

    props = {'Description': (<type 'basestring'>, False), 'Info': (<type 'dict'>, True),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceCatalog::ProvisioningArtifactProperties'

class troposphere.servicecatalog.ProvisioningParameter(title=None, **kwargs)

    Bases: troposphere.AWSPROPERTY

    props = {'Key': (<type 'basestring'>, False), 'Value': (<type 'basestring'>, False),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceCatalog::ProvisioningParameter'

class troposphere.servicecatalog.TagOption(title, template=None, validation=True,
                                           **kwargs)

    Bases: troposphere.AWSObject

    props = {'Active': (<function boolean at 0x7f30f87f5938>, False), 'Key': (<type 'basestring'>, True),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceCatalog::TagOption'

class troposphere.servicecatalog.TagOptionAssociation(title, template=None, validation=True,
                                                      **kwargs)

    Bases: troposphere.AWSObject

    props = {'ResourceId': (<type 'basestring'>, True), 'TagOptionId': (<type 'basestring'>, True),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceCatalog::TagOptionAssociation'

```

troposphere.servicediscovery module

```

class troposphere.servicediscovery.DnsConfig(title=None, **kwargs)

    Bases: troposphere.AWSPROPERTY

    props = {'DnsRecords': ([<class 'troposphere.servicediscovery.DnsRecord'>], True), 'Name': (<type 'basestring'>, True),
             'ResourceType': (<type 'basestring'>, True),
             'resource_type = 'AWS::ServiceDiscovery::DnsConfig'

```

```
class troposphere.servicediscovery.DnsRecord (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'TTL': (<type 'basestring'>, True), 'Type': (<type 'basestring'>, True)}

class troposphere.servicediscovery.HealthCheckConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'FailureThreshold': (<type 'float'>, False), 'ResourcePath': (<type 'basest...

class troposphere.servicediscovery.HealthCheckCustomConfig (title=None,
                                                                **kwargs)
    Bases: troposphere.AWSProperty

    props = {'FailureThreshold': (<type 'float'>, True)}

class troposphere.servicediscovery.HttpNamespace (title, template=None, valida-
                                                    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, '
    resource_type = 'AWS::ServiceDiscovery::HttpNamespace'

class troposphere.servicediscovery.Instance (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject

    props = {'InstanceAttributes': (<type 'dict'>, True), 'InstanceId': (<type 'basestri...
    resource_type = 'AWS::ServiceDiscovery::Instance'

class troposphere.servicediscovery.PrivateDnsNamespace (title, template=None, vali-
                                                          dation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, '
    resource_type = 'AWS::ServiceDiscovery::PrivateDnsNamespace'

class troposphere.servicediscovery.PublicDnsNamespace (title, template=None, valida-
                                                         tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, '
    resource_type = 'AWS::ServiceDiscovery::PublicDnsNamespace'

class troposphere.servicediscovery.Service (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject

    props = {'Description': (<type 'basestring'>, False), 'DnsConfig': (<class 'troposph...
    resource_type = 'AWS::ServiceDiscovery::Service'
```

troposphere.ses module

```
class troposphere.ses.Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'AddHeaderAction': (<class 'troposphere.ses.AddHeaderAction'>, False), 'Bound...

class troposphere.ses.AddHeaderAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```

    props = {'HeaderName': (<type 'basestring'>, True), 'HeaderValue': (<type 'basestring'>, True)}
class troposphere.ses.BounceAction(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Message': (<type 'basestring'>, True), 'Sender': (<type 'basestring'>, True)}
class troposphere.ses.CloudWatchDestination(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DimensionConfigurations': ([<class 'troposphere.ses.DimensionConfiguration'>],)}
class troposphere.ses.ConfigurationSet(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, False)}
    resource_type = 'AWS::SES::ConfigurationSet'
class troposphere.ses.ConfigurationSetEventDestination(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ConfigurationSetName': (<type 'basestring'>, True), 'EventDestination': (<class 'troposphere.ses.EventDestination'>,)}
    resource_type = 'AWS::SES::ConfigurationSetEventDestination'
class troposphere.ses.DimensionConfiguration(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DefaultDimensionValue': (<type 'basestring'>, True), 'DimensionName': (<type 'basestring'>, True)}
class troposphere.ses.EmailTemplate(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'HtmlPart': (<type 'basestring'>, False), 'SubjectPart': (<type 'basestring'>, True)}
class troposphere.ses.EventDestination(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'CloudWatchDestination': (<class 'troposphere.ses.CloudWatchDestination'>, False), 'EventSubscriptionArn': (<type 'basestring'>, True)}
class troposphere.ses.Filter(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'IpFilter': (<class 'troposphere.ses.IpFilter'>, True), 'Name': (<type 'basestring'>, True)}
class troposphere.ses.IpFilter(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'Cidr': (<type 'basestring'>, True), 'Policy': (<type 'basestring'>, True)}
class troposphere.ses.KinesisFirehoseDestination(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'DeliveryStreamARN': (<type 'basestring'>, True), 'IAMRoleARN': (<type 'basestring'>, True)}
class troposphere.ses.LambdaAction(title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY
    props = {'FunctionArn': (<type 'basestring'>, True), 'InvocationType': (<type 'basestring'>, True)}
class troposphere.ses.ReceiptFilter(title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Filter': (<class 'troposphere.ses.Filter'>, True)}
    resource_type = 'AWS::SES::ReceiptFilter'

```

```
class troposphere.ses.ReceiptRule (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'After': (<type 'basestring'>, False), 'Rule': (<class 'troposphere.ses.Rule'>, False)}
    resource_type = 'AWS::SES::ReceiptRule'

class troposphere.ses.ReceiptRuleSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'RuleSetName': (<type 'basestring'>, False)}
    resource_type = 'AWS::SES::ReceiptRuleSet'

class troposphere.ses.Rule (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Actions': ([<class 'troposphere.ses.Action'>], False), 'Enabled': (<function '...'>, False)}

class troposphere.ses.S3Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'BucketName': (<type 'basestring'>, True), 'KmsKeyArn': (<type 'basestring'>, False)}

class troposphere.ses.SNSAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Encoding': (<type 'basestring'>, False), 'TopicArn': (<type 'basestring'>, False)}

class troposphere.ses.StopAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Scope': (<type 'basestring'>, True), 'TopicArn': (<type 'basestring'>, False)}

class troposphere.ses.Template (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Template': (<class 'troposphere.ses.EmailTemplate'>, False)}
    resource_type = 'AWS::SES::Template'

class troposphere.ses.WorkmailAction (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'OrganizationArn': (<type 'basestring'>, True), 'TopicArn': (<type 'basestring'>, False)}
```

troposphere.sns module

```
class troposphere.sns.Subscription (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'Endpoint': (<type 'basestring'>, True), 'Protocol': (<type 'basestring'>, False)}

class troposphere.sns.SubscriptionResource (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DeliveryPolicy': (<type 'dict'>, False), 'Endpoint': (<type 'basestring'>, True)}
    resource_type = 'AWS::SNS::Subscription'

class troposphere.sns.Topic (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DisplayName': (<type 'basestring'>, False), 'KmsMasterKeyId': (<type 'basestring'>, False)}
```

```
resource_type = 'AWS::SNS::Topic'
```

```
class troposphere.sns.TopicPolicy(title, template=None, validation=True, **kwargs)
```

```
Bases: troposphere.AWSObject
```

```
props = {'PolicyDocument': ((<type 'dict'>, <class 'awacs.aws.Policy'>), True), 'TopicName': (<type 'basestring'>, True)}
```

```
resource_type = 'AWS::SNS::TopicPolicy'
```

troposphere.sqs module

```
class troposphere.sqs.Queue(title, template=None, validation=True, **kwargs)
```

```
Bases: troposphere.AWSObject
```

```
props = {'ContentBasedDeduplication': (<type 'bool'>, False), 'DelaySeconds': (<type 'int'>, True), 'QueueName': (<type 'basestring'>, True)}
```

```
resource_type = 'AWS::SQS::Queue'
```

```
validate()
```

```
class troposphere.sqs.QueuePolicy(title, template=None, validation=True, **kwargs)
```

```
Bases: troposphere.AWSObject
```

```
props = {'PolicyDocument': ((<type 'dict'>, <class 'awacs.aws.Policy'>), False), 'QueueName': (<type 'basestring'>, True)}
```

```
resource_type = 'AWS::SQS::QueuePolicy'
```

```
class troposphere.sqs.RedrivePolicy(title=None, **kwargs)
```

```
Bases: troposphere.AWSProperty
```

```
props = {'deadLetterTargetArn': (<type 'basestring'>, False), 'maxReceiveCount': (<type 'int'>, True)}
```

troposphere.ssm module

```
class troposphere.ssm.Association(title, template=None, validation=True, **kwargs)
```

```
Bases: troposphere.AWSObject
```

```
props = {'AssociationName': (<type 'basestring'>, False), 'DocumentVersion': (<type 'int'>, True), 'Name': (<type 'basestring'>, True)}
```

```
resource_type = 'AWS::SSM::Association'
```

```
class troposphere.ssm.Document(title, template=None, validation=True, **kwargs)
```

```
Bases: troposphere.AWSObject
```

```
props = {'Content': (<type 'dict'>, True), 'DocumentType': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, True)}
```

```
resource_type = 'AWS::SSM::Document'
```

```
class troposphere.ssm.InstanceAssociationOutputLocation(title=None, **kwargs)
```

```
Bases: troposphere.AWSProperty
```

```
props = {'S3Location': (<class 'troposphere.ssm.S3OutputLocation'>, False)}
```

```
class troposphere.ssm.LoggingInfo(title=None, **kwargs)
```

```
Bases: troposphere.AWSProperty
```

```
props = {'Region': (<type 'basestring'>, True), 'S3Bucket': (<function s3_bucket_name at 0x7f30f87f5938>, True)}
```

```
class troposphere.ssm.MaintenanceWindow(title, template=None, validation=True, **kwargs)
```

```
Bases: troposphere.AWSObject
```

```
props = {'AllowUnassociatedTargets': (<function boolean at 0x7f30f87f5938>, True), 'CronExpression': (<type 'basestring'>, True)}
```

```
resource_type = 'AWS::SSM::MaintenanceWindow'
```

```
class troposphere.ssm.MaintenanceWindowAutomationParameters (title=None,
                                                             **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DocumentVersion': (<type 'basestring'>, False), 'Parameters': (<type 'dict
```

```
class troposphere.ssm.MaintenanceWindowLambdaParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ClientContext': (<type 'basestring'>, False), 'Payload': (<function json_c
```

```
class troposphere.ssm.MaintenanceWindowRunCommandParameters (title=None,
                                                             **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Comment': (<type 'basestring'>, False), 'DocumentHash': (<type 'basestring
```

```
class troposphere.ssm.MaintenanceWindowStepFunctionsParameters (title=None,
                                                                **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Input': (<type 'basestring'>, False), 'Name': (<type 'basestring'>, False)
```

```
class troposphere.ssm.MaintenanceWindowTarget (title, template=None, validation=True,
                                                **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'Name': (<type 'basestring'>,
    resource_type = 'AWS::SSM::MaintenanceWindowTarget'
```

```
class troposphere.ssm.MaintenanceWindowTask (title, template=None, validation=True,
                                              **kwargs)
    Bases: troposphere.AWSObject
    props = {'Description': (<type 'basestring'>, False), 'LoggingInfo': (<class 'troposp
    resource_type = 'AWS::SSM::MaintenanceWindowTask'
```

```
class troposphere.ssm.NotificationConfig (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'NotificationArn': (<type 'basestring'>, False), 'NotificationEvents': (<fun
```

```
class troposphere.ssm.Parameter (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'AllowedPattern': (<type 'basestring'>, False), 'Description': (<type 'base
    resource_type = 'AWS::SSM::Parameter'
```

```
class troposphere.ssm.PatchBaseline (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ApprovalRules': (<class 'troposphere.ssm.RuleGroup'>, False), 'ApprovedPatch
    resource_type = 'AWS::SSM::PatchBaseline'
```

```
class troposphere.ssm.PatchFilter (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Key': (<type 'basestring'>, True), 'Values': ([<type 'basestring'>], True)
```

```
class troposphere.ssm.PatchFilterGroup (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PatchFilters': ([<class 'troposphere.ssm.PatchFilter'>], False)}
```

```

class troposphere.ssm.ResourceDataSync (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'BucketName': (<type 'basestring'>, True), 'BucketPrefix': (<type 'basestring'>, True), 'ResourceDataSync': (<type 'basestring'>, True)}
    resource_type = 'AWS::SSM::ResourceDataSync'

class troposphere.ssm.Rule (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ApproveAfterDays': (<function integer at 0x7f30f87f59b0>, False), 'ComplianceParameters': (<type 'basestring'>, True), 'Description': (<type 'basestring'>, True), 'Name': (<type 'basestring'>, True), 'Severity': (<type 'basestring'>, True), 'Source': (<type 'basestring'>, True), 'TargetId': (<type 'basestring'>, True), 'TargetName': (<type 'basestring'>, True), 'TargetType': (<type 'basestring'>, True), 'Version': (<type 'basestring'>, True)}

class troposphere.ssm.RuleGroup (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'PatchRules': ([<class 'troposphere.ssm.Rule'>], False)}

class troposphere.ssm.S3OutputLocation (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'OutputS3BucketName': (<type 'basestring'>, False), 'OutputS3KeyPrefix': (<type 'basestring'>, True)}

class troposphere.ssm.Targets (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Key': (<type 'basestring'>, True), 'Values': ([<type 'basestring'>], True)}

class troposphere.ssm.TaskInvocationParameters (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'MaintenanceWindowAutomationParameters': (<class 'troposphere.ssm.MaintenanceWindowAutomationParameters'>, True)}

```

troposphere.stepfunctions module

```

class troposphere.stepfunctions.Activity (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, True)}
    resource_type = 'AWS::StepFunctions::Activity'

class troposphere.stepfunctions.StateMachine (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DefinitionString': (<type 'basestring'>, True), 'RoleArn': (<type 'basestring'>, True)}
    resource_type = 'AWS::StepFunctions::StateMachine'

```

troposphere.template_generator module

This module makes it possible to instantiate a new Troposphere Template object from an existing CloudFormation Template.

```

Usage: from troposphere.template_generator import TemplateGenerator
import json

with open("myCloudFormationTemplate.json") as f:
    json_template = json.load(f)
    template = TemplateGenerator(json_template).to_json()

```

```

exception troposphere.template_generator.ResourceTypeNotDefined (resource)
    Bases: exceptions.Exception

```

exception troposphere.template_generator.**ResourceTypeNotFound** (*resource*, *resource_type*)

Bases: exceptions.Exception

class troposphere.template_generator.**TemplateGenerator** (*cf_template*, ****kwargs**)

Bases: *troposphere.Template*

DEPRECATED_MODULES = ['troposphere.dynamodb2']

inspect_functions

Returns a map of *FunctionName: FunctionClass*

inspect_members

Returns the list of all troposphere members we are able to construct

inspect_resources

Returns a map of *ResourceType: ResourceClass*

troposphere.utils module

troposphere.utils.**get_events** (*conn*, *stackname*)

Get the events in batches and return in chronological order

troposphere.utils.**tail** (*conn*, *stack_name*, *log_func*=<function *_tail_print*>, *sleep_time*=5, *include_initial*=True)

Show and then tail the event log

troposphere.validators module

troposphere.validators.**boolean** (*x*)

troposphere.validators.**check_required** (*class_name*, *properties*, *conditionals*)

troposphere.validators.**cloudfront_event_type** (*event_type*)

troposphere.validators.**cloudfront_forward_type** (*forward*)

troposphere.validators.**cloudfront_restriction_type** (*restriction_type*)

troposphere.validators.**cloudfront_viewer_protocol_policy** (*viewer_protocol_policy*)

troposphere.validators.**compliance_level** (*level*)

troposphere.validators.**defer** (*x*)

Method to indicate deferring property validation

troposphere.validators.**double** (*x*)

troposphere.validators.**elb_name** (*b*)

troposphere.validators.**encoding** (*encoding*)

troposphere.validators.**exactly_one** (*class_name*, *properties*, *conditionals*)

troposphere.validators.**iam_group_name** (*group_name*)

troposphere.validators.**iam_names** (*b*)

troposphere.validators.**iam_path** (*path*)

troposphere.validators.**iam_role_name** (*role_name*)

troposphere.validators.**iam_user_name** (*user_name*)


```

troposphere.validators.ignore (x)
    Method to indicate bypassing property validation
troposphere.validators.integer (x)
troposphere.validators.integer_list_item (allowed_values)
troposphere.validators.integer_range (minimum_val, maximum_val)
troposphere.validators.json_checker (prop)
troposphere.validators.key_usage_type (key)
troposphere.validators.mutually_exclusive (class_name, properties, conditionals)
troposphere.validators.network_port (x)
troposphere.validators.notification_event (events)
troposphere.validators.notification_type (notification)
troposphere.validators.one_of (class_name, properties, property, conditionals)
troposphere.validators.operating_system (os)
troposphere.validators.positive_integer (x)
troposphere.validators.priceclass_type (price_class)
troposphere.validators.s3_bucket_name (b)
troposphere.validators.s3_transfer_acceleration_status (value)
troposphere.validators.scalable_dimension_type (scalable_dimension)
troposphere.validators.service_namespace_type (service_namespace)
troposphere.validators.statistic_type (statistic)
troposphere.validators.status (status)
troposphere.validators.task_type (task)
troposphere.validators.tg_healthcheck_port (x)
troposphere.validators.vpc_endpoint_type (endpoint_type)
troposphere.validators.vpn_pre_shared_key (key)
troposphere.validators.vpn_tunnel_inside_cidr (cidr)

```

troposphere.waf module

```

class troposphere.waf.Action (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

    props = {'Type': (<type 'basestring'>, True)}

class troposphere.waf.ByteMatchSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'ByteMatchTuples': ([[<class 'troposphere.waf.ByteMatchTuples'>]], False), 'Name': <class 'troposphere.waf.ByteMatchSet'>}
    resource_type = 'AWS::WAF::ByteMatchSet'

class troposphere.waf.ByteMatchTuples (title=None, **kwargs)
    Bases: troposphere.AWSPROPERTY

```

```
    props = {'FieldToMatch': (<class 'troposphere.waf.FieldToMatch'>, True), 'PositionalC
class troposphere.waf.FieldToMatch (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Data': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, True)}
class troposphere.waf.IPSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'IPSetDescriptors': ([<class 'troposphere.waf.IPSetDescriptors'>], False), '
    resource_type = 'AWS::WAF::IPSet'
class troposphere.waf.IPSetDescriptors (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Type': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}
class troposphere.waf.Predicates (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DataId': (<type 'basestring'>, True), 'Negated': (<function boolean at 0x7
class troposphere.waf.Rule (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'MetricName': (<type 'basestring'>, True), 'Name': (<type 'basestring'>, Tr
    resource_type = 'AWS::WAF::Rule'
class troposphere.waf.Rules (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Action': (<class 'troposphere.waf.Action'>, True), 'Priority': (<function
class troposphere.waf.SizeConstraint (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'ComparisonOperator': (<type 'basestring'>, True), 'FieldToMatch': (<class
class troposphere.waf.SizeConstraintSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, True), 'SizeConstraints': ([<class 'troposphere
    resource_type = 'AWS::WAF::SizeConstraintSet'
class troposphere.waf.SqlInjectionMatchSet (title, template=None, validation=True,
    **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, True), 'SqlInjectionMatchTuples': ([<class 't
    resource_type = 'AWS::WAF::SqlInjectionMatchSet'
class troposphere.waf.SqlInjectionMatchTuples (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'FieldToMatch': (<class 'troposphere.waf.FieldToMatch'>, True), 'TextTransfo
class troposphere.waf.WebACL (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'DefaultAction': (<class 'troposphere.waf.Action'>, True), 'MetricName': (<
    resource_type = 'AWS::WAF::WebACL'
```

```

class troposphere.waf.XssMatchSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'Name': (<type 'basestring'>, True), 'XssMatchTuples': ([<class 'troposphere.waf.XssMatchTuple'>], False)}
    resource_type = 'AWS::WAF::XssMatchSet'

class troposphere.waf.XssMatchTuple (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'FieldToMatch': (<class 'troposphere.waf.FieldToMatch'>, True), 'TextTransformation': (<type 'basestring'>, True)}

```

troposphere.wafregional module

```

class troposphere.wafregional.Action (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Type': (<type 'basestring'>, True)}

class troposphere.wafregional.ByteMatchSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'ByteMatchTuples': ([<class 'troposphere.wafregional.ByteMatchTuples'>], False)}
    resource_type = 'AWS::WAFRegional::ByteMatchSet'

class troposphere.wafregional.ByteMatchTuples (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'FieldToMatch': (<class 'troposphere.wafregional.FieldToMatch'>, True), 'PositionalArguments': (<type 'basestring'>, True)}

class troposphere.wafregional.FieldToMatch (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Data': (<type 'basestring'>, False), 'Type': (<type 'basestring'>, True)}

class troposphere.wafregional.IPSet (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'IPSetDescriptors': ([<class 'troposphere.wafregional.IPSetDescriptors'>], False)}
    resource_type = 'AWS::WAFRegional::IPSet'

class troposphere.wafregional.IPSetDescriptors (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Type': (<type 'basestring'>, True), 'Value': (<type 'basestring'>, True)}

class troposphere.wafregional.Predicates (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'DataId': (<type 'basestring'>, True), 'Negated': (<function boolean at 0x7f8c1b1b1b1b>, True)}

class troposphere.wafregional.Rule (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject
    props = {'MetricName': (<type 'basestring'>, True), 'Name': (<type 'basestring'>, True), 'Priority': (<type 'basestring'>, True)}
    resource_type = 'AWS::WAFRegional::Rule'

class troposphere.wafregional.Rules (title=None, **kwargs)
    Bases: troposphere.AWSProperty
    props = {'Action': (<class 'troposphere.wafregional.Action'>, True), 'Priority': (<type 'basestring'>, True)}

```

```
class troposphere.wafregional.SizeConstraint (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'ComparisonOperator': (<type 'basestring'>, True), 'FieldToMatch': (<class
class troposphere.wafregional.SizeConstraintSet (title, template=None, validation=True,
                                                    **kwargs)
    Bases: troposphere.AWSObject

    props = {'Name': (<type 'basestring'>, True), 'SizeConstraints': ([<class 'troposphere
    resource_type = 'AWS::WAFRegional::SizeConstraintSet'

class troposphere.wafregional.SqlInjectionMatchSet (title, template=None, valida-
                                                    tion=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'Name': (<type 'basestring'>, True), 'SqlInjectionMatchTuples': ([<class 't
    resource_type = 'AWS::WAFRegional::SqlInjectionMatchSet'

class troposphere.wafregional.SqlInjectionMatchTuples (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'FieldToMatch': (<class 'troposphere.wafregional.FieldToMatch'>, True), 'Tex
class troposphere.wafregional.WebACL (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'DefaultAction': (<class 'troposphere.wafregional.Action'>, True), 'MetricNa
    resource_type = 'AWS::WAFRegional::WebACL'

class troposphere.wafregional.WebACLAssociation (title, template=None, validation=True,
                                                    **kwargs)
    Bases: troposphere.AWSObject

    props = {'ResourceArn': (<type 'basestring'>, True), 'WebACLId': (<type 'basestring'
    resource_type = 'AWS::WAFRegional::WebACLAssociation'

class troposphere.wafregional.XssMatchSet (title, template=None, validation=True,
                                                    **kwargs)
    Bases: troposphere.AWSObject

    props = {'Name': (<type 'basestring'>, True), 'XssMatchTuples': ([<class 'troposphere
    resource_type = 'AWS::WAFRegional::XssMatchSet'

class troposphere.wafregional.XssMatchTuple (title=None, **kwargs)
    Bases: troposphere.AWSProperty

    props = {'FieldToMatch': (<class 'troposphere.wafregional.FieldToMatch'>, True), 'Tex
```

troposphere.workspaces module

```
class troposphere.workspaces.Workspace (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'BundleId': (<type 'basestring'>, True), 'DirectoryId': (<type 'basestring'
    resource_type = 'AWS::WorkSpaces::Workspace'

class troposphere.workspaces.WorkspaceProperties (title=None, **kwargs)
    Bases: troposphere.AWSProperty
```

```
props = {'ComputeTypeName': (<type 'basestring'>, False), 'RootVolumeSizeGib': (<fun
```

Module contents

```
class troposphere.AWSAttribute (title=None, **kwargs)
```

Bases: *troposphere.BaseAWSObject*

dictname = None

Used for CloudFormation Resource Attribute objects [http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/ aws-product-attribute-reference.html](http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-product-attribute-reference.html)

```
class troposphere.AWSDeclaration (title, **kwargs)
```

Bases: *troposphere.BaseAWSObject*

Used for CloudFormation Resource Property objects <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/ aws-product-property-reference.html>

Ref ()

ref ()

```
class troposphere.AWSHelperFn
```

Bases: *object*

getdata (*data*)

to_dict ()

```
class troposphere.AWSObject (title, template=None, validation=True, **kwargs)
```

Bases: *troposphere.BaseAWSObject*

GetAtt (*value*)

Ref ()

dictname = 'Properties'

get_att (*value*)

ref ()

```
class troposphere.AWSProperty (title=None, **kwargs)
```

Bases: *troposphere.BaseAWSObject*

Used for CloudFormation Resource Property objects <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/ aws-product-property-reference.html>

dictname = None

```
class troposphere.And (cond_one, cond_two, *conds)
```

Bases: *troposphere.AWSHelperFn*

```
class troposphere.Base64 (data)
```

Bases: *troposphere.AWSHelperFn*

```
class troposphere.BaseAWSObject (title, template=None, validation=True, **kwargs)
```

Bases: *object*

add_to_template ()

classmethod from_dict (*title, d*)

no_validation ()

to_dict ()

```
    validate()
    validate_title()
class troposphere.Cidr(ipblock, count, sizemask=None)
    Bases: troposphere.AWSHelperFn
class troposphere.Condition(data)
    Bases: troposphere.AWSHelperFn
class troposphere.Equals(value_one, value_two)
    Bases: troposphere.AWSHelperFn
class troposphere.Export(name)
    Bases: troposphere.AWSHelperFn
class troposphere.FindInMap(mapname, key, value)
    Bases: troposphere.AWSHelperFn
class troposphere.GenericHelperFn(data)
    Bases: troposphere.AWSHelperFn
    Used as a fallback for the template generator
    to_dict()
class troposphere.GetAZs(region="")
    Bases: troposphere.AWSHelperFn
class troposphere.GetAtt(logicalName, attrName)
    Bases: troposphere.AWSHelperFn
class troposphere.If(cond, true, false)
    Bases: troposphere.AWSHelperFn
class troposphere.ImportValue(data)
    Bases: troposphere.AWSHelperFn
class troposphere.Join(delimiter, values)
    Bases: troposphere.AWSHelperFn
class troposphere.Name(data)
    Bases: troposphere.AWSHelperFn
class troposphere.Not(cond)
    Bases: troposphere.AWSHelperFn
class troposphere.Or(cond_one, cond_two, *conds)
    Bases: troposphere.AWSHelperFn
class troposphere.Output(title, **kwargs)
    Bases: troposphere.AWSDeclaration
    add_to_template()
    props = {'Description': (<type 'basestring'>, False), 'Export': (<class 'troposphere
class troposphere.Parameter(title, **kwargs)
    Bases: troposphere.AWSDeclaration
    NUMBER_PROPERTIES = ['MaxValue', 'MinValue']
    STRING_PROPERTIES = ['AllowedPattern', 'MaxLength', 'MinLength']
    add_to_template()
```

```

    props = {'AllowedPattern': (<type 'basestring'>, False), 'AllowedValues': (<type 'li
    validate()
    validate_title()

class troposphere.Ref(data)
    Bases: troposphere.AWSHelperFn

class troposphere.Select(indx, objects)
    Bases: troposphere.AWSHelperFn

class troposphere.Split(delimiter, values)
    Bases: troposphere.AWSHelperFn

class troposphere.Sub(input_str, dict_values=None, **values)
    Bases: troposphere.AWSHelperFn

class troposphere.Tags(*args, **kwargs)
    Bases: troposphere.AWSHelperFn

    classmethod from_dict(title=None, **kwargs)

    to_dict()

class troposphere.Template(Description=None, Metadata=None)
    Bases: object

    add_condition(name, condition)
    add_description(description)
    add_mapping(name, mapping)
    add_metadata(metadata)
    add_output(output)
    add_parameter(parameter)
    add_parameter_to_group(parameter, group_name)
        Add a parameter under a group (created if needed). :type parameter: str or Parameter :type group_name:
        str
    add_resource(resource)
    add_transform(transform)
    add_version(version=None)
    get_or_add_parameter(parameter)
    handle_duplicate_key(key)
    props = {'AWSTemplateFormatVersion': (<type 'basestring'>, False), 'Description': (<
    set_description(description)
    set_metadata(metadata)
    set_parameter_label(parameter, label)
        Sets the Label used in the User Interface for the given parameter. :type parameter: str or Parameter :type
        label: str
    set_transform(transform)
    set_version(version=None)

```

```
to_dict ()
to_json (indent=4, sort_keys=True, separators=(', ', ': '))
to_yaml (clean_up=False, long_form=False)
class troposphere.UpdatePolicy (title, **kwargs)
    Bases: troposphere.BaseAWSObject
troposphere.depends_on_helper (obj)
    Handles using .title if the given object is a troposphere resource.

    If the given object is a troposphere resource, use the .title attribute of that resource. If it's a string, just use the string. This should allow more pythonic use of DependsOn.
troposphere.encode_to_dict (obj)
troposphere.is_aws_object_subclass (cls)
troposphere.validate_delimiter (delimiter)
troposphere.validate_pausetime (pausetime)
```

8.4 tests

8.4.1 tests package

Submodules

tests.test_apigateway module

```
class tests.test_apigateway.TestGatewayResponse (methodName='runTest')
    Bases: unittest.case.TestCase
    test_response_type ()
class tests.test_apigateway.TestModel (methodName='runTest')
    Bases: unittest.case.TestCase
    test_schema ()
```

tests.test_apigatewayv2 module

```
class tests.test_apigatewayv2.TestAuthorizer (methodName='runTest')
    Bases: unittest.case.TestCase
    test_response_type ()
class tests.test_apigatewayv2.TestIntegrationResponse (methodName='runTest')
    Bases: unittest.case.TestCase
    test_response_type ()
class tests.test_apigatewayv2.TestModel (methodName='runTest')
    Bases: unittest.case.TestCase
    test_schema ()
```


tests.test_appsync module

```

class tests.test_appsync.TestAppsyncResolver (methodName='runTest')
    Bases: unittest.case.TestCase

    test_resolver()

    test_resolver_kind_bad_value()

```

tests.test_asg module

```

class tests.test_asg.TestAutoScalingGroup (methodName='runTest')
    Bases: unittest.case.TestCase

    test_AutoScalingRollingUpdate_all_defaults()

    test_AutoScalingRollingUpdate_validation()

    test_exclusive()

    test_helperfn_as_AutoScalingRollingUpdate()

    test_helperfn_as_updatepolicy()

    test_instanceid()

    test_launchconfigurationname()

    test_none()

    test_size_if()

```

tests.test_awslambda module

```

class tests.test_awslambda.TestAWSLambda (methodName='runTest')
    Bases: unittest.case.TestCase

    test_check_zip_file()

    test_environment_variable_invalid_name()

    test_environment_variable_not_reserved()

    test_environment_variable_reserved()

    test_exclusive()

    test_zip_file()

```

tests.test_basic module

```

class tests.test_basic.FakeAWSObject (title, template=None, validation=True, **kwargs)
    Bases: troposphere.AWSObject

    props = {'callcorrect': (<function call_correct at 0x7f30f87c9de8>, False), 'callinco

    type = 'Fake::AWS::Object'

    validate()

```

```

class tests.test_basic.FakeAWSProperty (title=None, **kwargs)
    Bases: troposphere.AWSProperty

```

```
    props = {}  
  
class tests.test_basic.TestAttributes (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    test_BogusAttribute ()  
  
    test_UpdateReplacePolicy ()  
  
class tests.test_basic.TestBasic (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    test_badproperty ()  
  
    test_badrequired ()  
  
    test_badtype ()  
  
    test_depends_on_helper_with_resource ()  
  
    test_depends_on_helper_with_string ()  
  
    test_extraattribute ()  
  
    test_goodrequired ()  
  
    test_pickle_ok ()  
  
    test_resource_depends_on ()  
  
    test_resource_depends_on_attr ()  
  
    test_resource_depends_on_list ()  
  
class tests.test_basic.TestCidr (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    test_getcidr ()  
  
    test_getcidr_withsizemask ()  
  
class tests.test_basic.TestDuplicate (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    test_output ()  
  
    test_parameter ()  
  
    test_resource ()  
  
class tests.test_basic.TestHealthCheck (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    test_healthy_interval_ok ()  
  
    test_healthy_interval_too_low ()  
  
class tests.test_basic.TestJoin (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    test_join ()  
  
class tests.test_basic.TestName (methodName='runTest')  
    Bases: unittest.case.TestCase  
  
    test_ref ()  
  
class tests.test_basic.TestOutput (methodName='runTest')  
    Bases: unittest.case.TestCase
```

```
test_empty_awsproperty_outputs_empty_object ()
test_noproperty ()
class tests.test_basic.TestParameter (methodName='runTest')
    Bases: unittest.case.TestCase
    test_add_or_get_returns_with_out_adding_duplicate ()
    test_get_or_add_adds ()
    test_invalid_parameter_property_in_template ()
    test_noproperty ()
    test_property_default ()
    test_property_validator ()
class tests.test_basic.TestProperty (methodName='runTest')
    Bases: unittest.case.TestCase
    test_awsproperty_invalid_property ()
    test_noproperty ()
class tests.test_basic.TestRef (methodName='runTest')
    Bases: unittest.case.TestCase
    test_ref ()
    test_ref_eq ()
    test_ref_hash ()
class tests.test_basic.TestSplit (methodName='runTest')
    Bases: unittest.case.TestCase
    test_split ()
class tests.test_basic.TestSub (methodName='runTest')
    Bases: unittest.case.TestCase
    test_sub_with_vars_mix ()
    test_sub_with_vars_not_unpakaged ()
    test_sub_with_vars_unpakaged ()
    test_sub_without_vars ()
class tests.test_basic.TestValidation (methodName='runTest')
    Bases: unittest.case.TestCase
    test_no_validation_method ()
    test_novalidation ()
    test_validation ()
class tests.test_basic.TestValidators (methodName='runTest')
    Bases: unittest.case.TestCase
    test_badlist ()
    test_badmutilist ()
    test_callcorrect ()
```

```
test_callincorrect ()
test_exception ()
test_helperfun ()
test_list ()
test_multilist ()
test_mutualexclusion ()
test_tuples ()
```

```
tests.test_basic.call_correct (x)
tests.test_basic.call_incorrect (x)
```

tests.test_cloudformation module

```
class tests.test_cloudformation.TestWaitCondition (methodName='runTest')
    Bases: unittest.case.TestCase
    test_CreationPolicy ()
    test_CreationPolicyWithProps ()
    test_RequiredProps ()
```

tests.test_cloudwatch module

```
class tests.test_cloudwatch.TestModel (methodName='runTest')
    Bases: unittest.case.TestCase
    test_dashboard ()
```

tests.test_codebuild module

```
class tests.test_codebuild.TestCodeBuild (methodName='runTest')
    Bases: unittest.case.TestCase
    test_linux_environment ()
    test_source_codepipeline ()
    test_windows_environment ()
```

tests.test_codecommit module

```
class tests.test_codecommit.TestCodeCommit (methodName='runTest')
    Bases: unittest.case.TestCase
    test_trigger ()
```

tests.test_config module

```

class tests.test_config.TestConfig (methodName='runTest')
    Bases: unittest.case.TestCase

    test_SourceDetails ()

    test_invalid_SourceDetails_MaximumExecutionFrequency ()

```

tests.test_dict module

```

class tests.test_dict.TestDict (methodName='runTest')
    Bases: unittest.case.TestCase

    setUp ()
        Hook method for setting up the test fixture before exercising it.

    test_invalid_sub_property ()

    test_invalid_subproperty_definition ()

    test_invalid_toplevel_property ()

    test_sub_property_helper_fn ()

    test_tags_from_dict ()

    test_toplevel_helper_fn ()

    test_valid_data ()

```

tests.test_ec2 module

```

class tests.test_ec2.TestEC2 (methodName='runTest')
    Bases: unittest.case.TestCase

    test_securitygroupegress ()

```

tests.test_ecs module

```

class tests.test_ecs.TestECS (methodName='runTest')
    Bases: unittest.case.TestCase

    test_allow_container_healthcheck ()

    test_allow_placement_strategy_constraint ()

    test_allow_port_mapping_protocol ()

    test_allow_ref_cluster ()

    test_allow_ref_task_role_arn ()

    test_allow_scheduling_strategy ()

    test_allow_string_cluster ()

    test_allow_string_task_role_arn ()

    test_docker_volume_configuration ()

    test_fargate_launch_type ()

```

```
test_port_mapping_does_not_require_protocol()
```

```
test_task_role_arn_is_optional()
```

```
class tests.test_ecs.TestECSValidators (methodName='runTest')
```

```
Bases: unittest.case.TestCase
```

```
test_scope_validator()
```

tests.test_efs module

```
class tests.test_efs.TestEfs (methodName='runTest')
```

```
Bases: unittest.case.TestCase
```

```
test_validData()
```

```
test_validateProvisionedThroughputInMibps()
```

```
test_validateThroughputMode()
```

```
class tests.test_efs.TestEfsTemplate (methodName='runTest')
```

```
Bases: unittest.case.TestCase
```

```
test_bucket_template()
```

tests.test_elasticloadbalancerv2 module

```
class tests.test_elasticloadbalancerv2.TestListenerActions (methodName='runTest')
```

```
Bases: unittest.case.TestCase
```

```
test_fixed_response_action()
```

```
test_fixed_response_config_one_of()
```

```
test_fixed_response_config_only_with_fixed_response()
```

```
test_fixed_response_requires_fixed_response_config()
```

```
test_forward_action()
```

```
test_forward_action_requires_target_arn()
```

```
test_redirect_action()
```

```
test_redirect_action_config_one_of()
```

```
test_redirect_action_requires_redirect_config()
```

```
test_redirect_config_only_with_redirect()
```

```
test_target_arn_only_forward()
```

tests.test_emr module

```
class tests.test_emr.TestEMR (methodName='runTest')
```

```
Bases: unittest.case.TestCase
```

```
generate_rules (rules_name)
```

```
simple_helper (adjustment, scaling)
```

```
test_SimpleScalingPolicyConfiguration()
```

```
test_allow_string_cluster()
```

tests.test_examples module

```
class tests.test_examples.TestExamples (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    expected_output = None
```

```
    filename = None
```

```
    maxDiff = None
```

```
    test_example()
```

```
tests.test_examples.create_test_class (testname, **kwargs)
```

```
tests.test_examples.load_tests (loader, tests, pattern)
```

tests.test_examples_template_generator module

```
class tests.test_examples_template_generator.TestTemplateGenerator (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    expected_output = None
```

```
    filename = None
```

```
    maxDiff = None
```

```
    test_template_generator()
```

```
        Ensures that all example outputs can be loaded into the template generator and back to JSON with no difference.
```

```
tests.test_examples_template_generator.create_test_class (testname, **kwargs)
```

```
tests.test_examples_template_generator.load_tests (loader, tests, pattern)
```

tests.test_guardduty module

```
class tests.test_guardduty.TestGuardDuty (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    test_guardduty_detector()
```

```
    test_guardduty_ipset()
```

```
    test_guardduty_threatintelset()
```

tests.test_int_type module

```
class tests.test_int_type.TestIntTypeShouldNotBeUsed (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    assertNoIntType (obj)
```

```
    get_aws_objects()
```

`test_there_should_not_be_any_awsobject_with_int_in_props()`

Anything that requires an integer should use `validators.integer` rather than `int`, because `int` does not work with types that behave like an integer (i.e. `long` in Python 2).

tests.test_iot1click module

`class tests.test_iot1click.TestPlacementTemplate (methodName='runTest')`

Bases: `unittest.case.TestCase`

`test_placement_template()`

tests.test_logs module

`class tests.test_logs.TestLogs (methodName='runTest')`

Bases: `unittest.case.TestCase`

`test_log_destination()`

`test_loggroup_deletionpolicy_is_preserved()`

`test_loggroup_retention()`

tests.test_opsworks module

`class tests.test_opsworks.TestOpsWorksStack (methodName='runTest')`

Bases: `unittest.case.TestCase`

`test_custom_json()`

`test_no_required()`

`test_nosubnet()`

`test_required()`

`test_stack()`

tests.test_parameters module

`class tests.test_parameters.TestInitArguments (methodName='runTest')`

Bases: `unittest.case.TestCase`

`test_ref_can_be_requested()`

`test_title_max_length()`

tests.test_policies module

`class tests.test_policies.TestCreationPolicy (methodName='runTest')`

Bases: `unittest.case.TestCase`

`test_auto_scaling_creation_policy()`

`test_auto_scaling_creation_policy_json()`

`test_invalid_pausetime()`

`test_json()`


```

    test_works ()
class tests.test_policies.TestUpdatePolicy (methodName='runTest')
    Bases: unittest.case.TestCase
    test_invalid_pausetime ()
    test_json ()
    test_mininstances ()
    test_mininstances_maxsize_is_ref ()
    test_mininstances_mininstancesinservice_is_ref ()
    test_works ()

```

tests.test_rds module

```

class tests.test_rds.TestRDS (methodName='runTest')
    Bases: unittest.case.TestCase
    test_az_and_multiaz_funcs ()
    test_fail_az_and_multiaz ()
    test_iol_storage_type_and_iops ()
    test_it_allows_an_rds_instance_created_from_a_snapshot ()
    test_it_allows_an_rds_instance_with_iops ()
    test_it_allows_an_rds_instance_with_master_username_and_password ()
    test_it_allows_an_rds_replica ()
    test_it_rds_instances_require_either_a_snapshot_or_credentials ()
    test_it_rds_instances_require_encryption_if_kms_key_provided ()
    test_no_snapshot_or_engine ()
    test_optiongroup ()
    test_replica_settings_are_inherited ()
    test_snapshot ()
    test_snapshot_and_engine ()
    test_storage_to_iops_ratio ()
class tests.test_rds.TestRDSValidators (methodName='runTest')
    Bases: unittest.case.TestCase
    test_validate_backup_retention_period ()
    test_validate_backup_window ()
    test_validate_capacity ()
    test_validate_engine ()
    test_validate_engine_mode ()
    test_validate_iops ()
    test_validate_license_model ()

```

```
test_validate_maintenance_window()  
test_validate_storage_type()
```

tests.test_s3 module

```
class tests.test_s3.TestBucket (methodName='runTest')  
    Bases: unittest.case.TestCase  
    test_bucket_accesscontrol()  
    test_bucket_accesscontrol_bad_string()  
    test_bucket_accesscontrol_bad_type()  
    test_bucket_accesscontrol_ref()  
class tests.test_s3.TestBucketTemplate (methodName='runTest')  
    Bases: unittest.case.TestCase  
    test_bucket_template()  
class tests.test_s3.TestS3AccelerateConfiguration (methodName='runTest')  
    Bases: unittest.case.TestCase  
    test_accelerate_configuration_enabled()  
    test_accelerate_configuration_invalid_value()  
    test_accelerate_configuration_suspended()  
    test_s3_bucket_accelerate_configuration()
```

tests.test_serverless module

```
class tests.test_serverless.TestServerless (methodName='runTest')  
    Bases: unittest.case.TestCase  
    swagger = {'info': {'title': 'swagger test'}, 'paths': {'/test': {'get': {}}}, 's  
    test_DLQ()  
    test_exactly_one_code()  
    test_layer_version()  
    test_optional_auto_publish_alias()  
    test_optional_deployment_preference()  
    test_packaging()  
    test_policy_document()  
    test_required_api_both()  
    test_required_api_definitionbody()  
    test_required_api_definitionuri()  
    test_required_function()  
    test_s3_filter()  
    test_s3_location()
```

```
test_simple_table()
```

```
test_tags()
```

tests.test_sqs module

```
class tests.test_sqs.TestQueue (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    test_QueueName()
```

tests.test_stepfunctions module

```
class tests.test_stepfunctions.TestStepFunctions (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    test_activity()
```

```
    test_statemachine()
```

```
    test_statemachine_missing_parameter()
```

tests.test_tags module

```
class tests.test_tags.TestTags (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    test_ASTagAddition()
```

```
    test_Formats()
```

```
    test_TagAddition()
```

```
    test_Unsortable()
```

tests.test_template module

```
class tests.test_template.TestAwsInterface (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    test_parameter_group()
```

```
    test_parameter_label()
```

```
    test_parameter_label_replace()
```

```
class tests.test_template.TestEquality (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    test_eq()
```

```
    test_hash()
```

```
    test_ne()
```

```
class tests.test_template.TestInitArguments (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
    test_description()
```

```
    test_description_default()
```

```
test_metadata()
```

```
test_metadata_default()
```

```
test_transform()
```

```
class tests.test_template.TestValidate (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
test_max_mappings()
```

```
test_max_outputs()
```

```
test_max_parameters()
```

```
test_max_resources()
```

tests.test_template_generator module

```
class tests.test_template_generator.MyCustomResource (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Foo': (<type 'basestring'>, True), 'ServiceToken': (<type 'basestring'>, True)}
```

```
    resource_type = 'Custom::Resource'
```

```
class tests.test_template_generator.MyMacroResource (title, template=None, validation=True, **kwargs)
```

```
    Bases: troposphere.AWSObject
```

```
    props = {'Foo': (<type 'basestring'>, True)}
```

```
    resource_type = 'Some::Special::Resource'
```

```
class tests.test_template_generator.TestTemplateGenerator (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
test_custom_resource_override()
```

```
    Ensures that a custom member can be defined.
```

```
test_custom_resource_type()
```

```
    Ensures that a custom resource type is implicitly defined.
```

```
test_macro_resource()
```

```
    Ensures that a custom member can be defined.
```

```
test_no_nested_name()
```

```
    Prevent regression for ensuring no nested Name (Issue #977)
```

```
test_resource_type_not_defined()
```

```
test_unknown_resource_type()
```

tests.test_userdata module

```
class tests.test_userdata.TestUserdata (methodName='runTest')
```

```
    Bases: unittest.case.TestCase
```

```
create_answer (command_list, delimiter="")
```

```
create_result (file, delimiter="")
```

```
setUp()  
    Hook method for setting up the test fixture before exercising it.  
test_char_escaping()  
test_empty_file()  
test_nonexistant_file()  
test_one_line_file()  
test_simple()
```

tests.test_validators module

```
class tests.test_validators.TestValidators (methodName='runTest')  
    Bases: unittest.case.TestCase  
    test_boolean()  
    test_compliance_level()  
    test_elb_name()  
    test_encoding()  
    test_iam_group_name()  
    test_iam_names()  
    test_iam_path()  
    test_iam_role_name()  
    test_iam_user_name()  
    test_integer()  
    test_integer_range()  
    test_mutually_exclusive()  
    test_mutually_exclusive_novalue()  
    test_network_port()  
    test_network_port_ref()  
    test_notification_event()  
    test_notification_type()  
    test_one_of()  
    test_operating_system()  
    test_positive_integer()  
    test_s3_bucket_name()  
    test_status()  
    test_task_type()  
    test_tg_healthcheck_port()  
    test_tg_healthcheck_port_ref()
```

tests.test_yaml module

```
class tests.test_yaml.TestYAML (methodName='runTest')
    Bases: unittest.case.TestCase

    test_s3_bucket ()

    test_yaml_long_form ()
```

Module contents

8.5 Changelog

8.5.1 2.4.3 (2019*02*13)

- Fix Glue StorageDescriptor NumberOfBuckets spelling (#1310)
- ServiceDiscovery::Service DNSConfig is no longer required
- Sphinx docs (#1311)
- Add autogeneration of troposphere index files
- Fix ApiGateway AccessLogSetting prop spelling (#1316)
- Docs update (#1314)
- Add AWS::ApiGatewayV2 Resources (#1312)
- Updates for new resources being added

8.5.2 2.4.2 (2019*02*02)

- Add AWS::DocDB
- Add UpdateReplacePolicy attribute
- Use a dict instead of the Tags object for the Tags property on the dax resource (#1045) (#1046)
- Add better method names for Troposphere objects. (#1169)
- Update integer_list_item to always cast value to an int for comparison (#1192)
- Remove name parameter from json_checker (#1260)
- Remove duplicate VpcConfig/DomainJoinInfo classes from AppStream (#1285)
- Add 'Kind' property to AWS::AppSync::Resolver (#1287)
- Add missing region information. (#1288)
- Fix tag sorting on py3 (#1289)
- Updated autoscalingplans to match cloudformation doco (#1291)
- ResourceGroupArn is no longer mandatory for AWS::Inspector::AssessmentTarget (#1292)
- Fix creating RotationSchedule for SecretsManager (#1293)
- Add missing serverless properties (Fixes #1294)
- Make DataSourceName non*mandatory in appsync resolvers (#1296)

- Add new properties to AWS::CodeBuild::Project, per 2019 Jan 24 update (#1297)
- Add new AWS::OpsWorksCM::Server resource, per 2019 Jan 24 update (#1298)
- Add AWS::Serverless::LayerVersion (#1305)
- Fix for AWS Lambda reserved environment variables (#1306)
- Add SqsParameters support to Rule Target (#1307)
- Add DestinationPrefixListId to EC2 SecurityGroupRule (#1309)
- Fix for pyflakes 2.1.0

8.5.3 2.4.1 (2019*01*09)

- Add a S3OriginConfig object to distinguish between Distribution and StreamingDistribution properties (#1273)
- Add SSM Example for patch baselines and filter groups (#1274)
- Add better validation for AWS::CloudWatch::Alarm properties (#1276)
- Allow empty To/From port ranges for SG's for certain IP protocols (#1277)
- Add additional properties to AWS::Serverless::Api (#1278)
- Fixes DynamoDB validator error (#1280)

8.5.4 2.4.0 (2019*01*06)

- Setup tox (#1187)
- Set line length for Python files in EditorConfig (#1188)
- Fix EC2 SpotFleet properties #1195 (#1198)
- Add MultiValueAnswer property for AWS::Route53::RecordSet (#1199)
- adding RDS properties, per Nov 9 2018 update <https://docs.aws.amazon...> (#1201)
- Add Secrets Managers resources, per Nov 9 2018 update (#1202)
- Add DLM support, per Nov 12 2018 update (#1203)
- Adds support for Permissions Boundaries on AWS::IAM::Role and AWS::IAM::User (#1205)
- Add support for multi*region action in CodePipeline (#1207)
- Added support for Aurora BacktrackWindow. (#1210)
- Add AWS::AppStream resources
- Add Tags and WorkspaceProperties to WorkSpaces::Workspace
- Add support for AWS::AutoScalingPlans::ScalingPlan (#1197)
- adding KmsMasterKeyId to Topics, per Nov 19 2018 update
- adding PublicAccessBlockConfiguration to s3 buckets, per Nov 19 2018 update
- Validate Lambda environment variable names (#1186)
- Fix DockerVolumeConfiguration Labels and DriverOpts definition (#1194)
- Setup to_dict for Tags AWSHelper (#1189)
- Delete CodeDeploy EC2TagSetList class as it is just a property of EC2TagSet (#1212)

- Fix bugs and add missing properties in sagemaker (#1214)
- adding DeletionProtection property to RDS, per Nov 19 2018 update (#1215)
- adding PublicAccessBlockConfiguration to s3 buckets, per Nov 19 2018 update (#1216)
- Merge pull request #1217 from axelpavageau/feature/sns*20181119*update
- Add volume encryption, per Nov 19 2018 update (#1218)
- Add PublicIpv4Pool property to EIPs, per Nov 19 2018 update (#1219)
- Add new Lambda resources and props, per Nov 29 2018 update (#1242)
- Add MixedInstancesPolicy property to autoscaling groups, per Nov 19 2018 update. (#1220)
- Add tags to API Gateway resources, per Nov 19 2018 update (#1221)
- Add various EMR properties, per Nov 19 2018 update (#1222)
- Add new kinesis resource, per Nov 20 2018 update (#1224)
- Make Lambda::LayerVersion CompatibleRuntimes a list of strings
- Add new route53 resources, per Nov 20 2018 update (#1223)
- Add new EC2Fleet resource, per Nov 20 2018 update (#1225)
- Add new appsync FunctionConfiguration resource & properties, per Nov 20 2018 update (#1226)
- Update AWS::CloudWatch::Alarm, per Nov 20 2018 update (#1227)
- CloudWatch MetricDataQuery Id is required
- Add DatapointsToAlarm to AWS::CloudWatch::Alarm (#1244)
- Alphabetize DatapointsToAlarm in CloudWatch
- Update Autoscalingplans properties, per Nov 20 2018 update (#1228)
- Add Iot1click resources (#1229)
- Add new Transit Gateway resources, per Nov 26 2018 release (#1232)
- Fix online merge issue
- Fixes EC2 SpotFleet LoadBalancersConfig structure (#1233)
- Sets InstanceType in EC2 LaunchTemplateData to not required. (#1234)
- Add new HttpNamespace resource & various servicediscovery props, per Nov 28 2018 update (#1237)
- Add new ec2 properties, per Nov 28 2018 update (#1238)
- EC2 Instance LicenseConfigurationArn is required
- Add on*demand billing for DynamoDB tables (#1243)
- Correct RoleArn case for OrganizationAggregationSource (#1247)
- Add various codebuild properties, per Dec 6 2018 update (#1249)
- Add support for DeploymentPreference to AWS::Serverless::Function (#1251)
- Update typo on EnableCloudwatchLogsExports (#1253)
- Add new AmazonMQ resource, per Dec 13 2018 update (#1254)
- Add Alexa Skill resource, per Nov 20 2018 update (#1230)
- Add new IoTAnalytics resources, per Dec 13 2018 update (#1255)

- Extend Action to support Redirect and FixedResponse for AWS::ElasticLoadBalancingV2::ListenerRule (#1140)
- Add support for extensible resource definitions in template generator (#1154)
- Updates CloudFront with missing parameters and validators (#1235)
- Added support for AWS Batch PlacementGroup & LaunchTemplate (#1262)
- Add DeleteAutomatedBackups to RDS DBInstance (#1263)
- Add missing KMS key properties (#1265)
- Fix pep errors due to online merge
- Fix EC2Fleet class definition to match functional correctness of CloudFormation (#1266)
- Add Tags property to AWS::AmazonMQ::Broker, per 2019 Jan 3 update (#1267)
- Add Containers property to AWS::SageMaker::Model per 2019 Jan 3 update (#1268)
- Add AWS::Route53Resolver::ResolverRuleAssociation resource, per 2019 Jan 3 update (#1269)
- Fix nested 'Name' sections in Output import (#1270)
- README.rst: Use SVG build status badge (#1271)
- Add test for nested Name in TemplateGenerator fixed via #1270

8.5.5 2.3.4 (2018*11*04)

- Add CloudFormation::Macro
- Instance ImageId is no longer required, specifically if using Launch Templates; updated tests (#1137)
- Fix amazonmq missing properties (#1143)
- Update AmazonMQ::Broker properties to use [basestring] instead of list
- Update the OnPremisesInstanceTagFilters parameter for AWS::CodeDeploy::DeploymentGroup (#1145)
- Update constants.py (#1147)
- Fix AutoScalingRollingUpdate validation failure (#1148)
- Adding UseOnlineResharding policy per 09/20/2018 update (#1149)
- Add SchedulingStrategy as a prop to ecs.Service (#1150)
- Added ConnectionId and ConnectionType to API GW method integration (#1153)
- Use dict as aws expects for ApiGateway::RestApi Parameters (#1156)
- Add support for AWS*interface metadata (#1171)
- Add new properties to ServiceRegistry (#1172)
- [#1167] Add support for DockerVolumeConfiguration in AWS::ECS::TaskDefinition (#1168)
- Add missing Codebuild source types (#1160)
- [#1155] Aurora serverless support (#1166)
- Missing RepositoryCredentials attribute for ContainerDefinition object (#1165)
- Update for new S3 destination option in flow logs (#1158)
- updates rds vpc example and closes #985 (#1157)

- Update apigateway as of 09/20/18 (#1173)
- Add missing APIGateway properties
- Update codebuild as of 09/20/18 (#1175)
- Update ec2 as of 09/20/18 (#1177)
- Additional codebuild source types (#1178)
- Use basestring to allow percentage definition in MaintenanceWindowTask (#1151)
- Fix issues with CanarySettings properties (#1181)
- 9/20/2018 update * NodeGroupId for Elasticache (#1182)
- Update codedeploy as of 09/20/18 (#1176)
- Add LambdaPermission in Example CloudWatchEventsSample.py (#1141)
- improve double validation and fix some property datatypes (#1179)
- Fix #1174 TemplateGenerator fail to parse template Fn::Sub with variable (#1180)

8.5.6 2.3.3 (2018*09*05)

- Revert schedule expression validation (#1114)

8.5.7 2.3.2 (2018*09*04)

- Auto add Parameter and Output to template when specified (#1018)
- Changed policy to AmazonDynamoDBFullAccess for delete and put (#1106)
- Fix CPUCredits casing and implement LaunchTemplateCreditSpecification class (#1100)
- Add UsernameAttributes to Cognito (#1104)
- Add SQS Event to serverless.py (#1103)
- Add support for Windows containers in CodeBuild (#1097)
- Generate class stubs necessary for autocompletion (#1079)
- Add AWS::IAM::ServiceLinkedRole (#1110)
- Made S3 Prefix in Firehose optional (#1102)
- Prefix is still required in ExtendedS3DestinationConfiguration
- SimpleTable has more attributes (#1108)
- Alphabetize properties in servlerless::SimpleTable
- AccountAggregationSources must be a list (#1111)
- Schedule expression validation (#1114)
- Add EndpointIdnetifier property to AWS::DMS::Endpoint object (#1117)
- Add get_or_add parameter method (#1118)
- Added HealthCheckCustomConfig to ServiceDiscovery Service (#1120)
- Tags support for SQS queues (#1121)
- VPCPeeringConnection PeerRegion (#1123)

- Add FilterPolicy as a property of SubscriptionResource (#1125)
- Add missing properties to SNS::Subscription
- Add ThroughputMode and ProvisionedThroughputInMibps to EFS (#1124) (#1126)
- Add AWS::EC2::VPCEndpointServicePermissions (#1130)
- AMAZON_LINUX_2 is now supported by SSM (#1133)
- [codebuild] Source * use value comparison instead of identity (#1134)
- InvitationId in GuardDuty::Master is now optional
- Fix missing boolean import in sns
- Add CodePipeline::Webhook resource
- Add ReportBuildStatus to CodeBuild Source property
- Add HttpConfig to AppSync::DataSource
- Add FieldLevelEncryptionId to CacheBehavior properties
- Add Timeout to Batch::JobDefinition
- Add EncryptionDisabled and OverrideArtifactName to CodeBuild Artifacts
- Add SSESpecification to DAX::Cluster
- Add KerberosAttributes to EMR::Cluster
- Add ValidationMethod to CertificateManager::Certificate
- Add Classifiers and Configuration to Glue resources
- Add SecondaryArtifacts and SecondarySources to CodeBuild::Project
- Add Logs to AmazonMQ::Broker

8.5.8 2.3.1 (2018*07*01)

- Add support for AWS::Neptune
- Add support for AWS::EKS
- Add support for AWS::AmazonMQ
- Add support for AWS::SageMaker
- Fix use of to_yaml long_form parameter (#1055)
- Adding CENTOS to validators.operating_system (#1058)
- Update constants with additional EC2 instances (#1059)
- Fix casing of CreditSpecification CpuCredits (#1068)
- Add 'Name' property for AWS::Serverless::Api (#1070)
- Add equality methods to Template (#1072)
- AWS PrivateLink support (#1084)
- Add return value to template.add_condition() (#1087)
- Add tests for to_yaml parameters
- Use endpoint_type for vpc_endpoint_type param instead of type

- Add resource EC2::VPCEndpointConnectionNotification
- Add resource SSM::ResourceDataSync

8.5.9 2.3.0 (2018*05*26)

- Allow Refs to be hashable using their data (#1053)
- Add AWS::Budgets
- Add new AWS::ServiceCatalog resources
- Add Policy to ApiGateway::RestApi
- Add ServiceLinkedRoleARN to AutoScaling::AutoScalingGroup
- Add LaunchConfigurationName to AutoScaling::LaunchConfiguration
- Add Edition to DirectoryService::MicrosoftAD
- Add PointInTimeRecoverySpecification to DynamoDB::Table
- Add ServiceRegistries to ECS::Service
- Add HealthCheck to ECS::TaskDefinition ContainerDefinition
- Add EncryptionAtRestOptions to Elasticsearch::Domain
- Add MaxSessionDuration to IAM::Role
- Add SplunkDestinationConfiguration to KinesisFirehose::DeliveryStream
- StartingPosition is no longer required in Lambda::EventSourceMapping
- Add DefaultValue to Logs::MetricFilter MetricTransformation
- Add OutputLocation to SSM::Association
- Add AutoScaling and EC2 LaunchTemplate support (#1038)
- Add LaunchTemplate to EC2::Instance
- Adding ECS Container Healthchecks tests (#1024)
- Rename ActionTypeID to ActionTypeId in CodePipeline

8.5.10 2.2.2 (2018*05*23)

- Allow up to 50:1 ratio for iops and allocated storage
- Correct Spot Fleet TagSpecifications (#1010)
- Change GetCidr to Cidr (Fixes #1013)
- Add missing OpsWorks::Instance properties (Fixes #1014)
- Adding SUSE to list of operating systems for SSM (#1015)
- Updates for latest pycodestyle warnings
- Add AWS::AppSync
- Add AWS::ServiceCatalog
- Special case Tags support in gen.py
- Add constants for EC2 C5 instance types (#1025)

- Update guardduty.py (#1037)
- Add OpenIdConnectConfig to AppSync::GraphQLApi
- Update AWS Config features (updates #1022)
- Updated appsync apikey expires to be an int. (#1040)
- Fix AutoScalingRole in EMR: Fixes #984 (#1036)
- Rename SES Template to EmailTemplate (#1047)
- Add GuardDuty::Filter
- Remove python 3.3 support since it's EOL (#1049)
- Corrected the description of NatGateway (#1005)
- Update deprecated modules (#1007)
- Updared CodeBuild Source Options (#1017)
- Allow Ref's to test equality against their data (#1048)
- Update to cfn*flip 1.0.2 (#1003)
- Eliminate infinite loop when pickle loads BaseAWSObject and objects derived from it. (#1016)
- Allow multiple NoValue properties in mutually_exclusive (#1050)

8.5.11 2.2.1 (2018*03*10)

- type is not required for EnvironmentVariable (#975)
- Properly handle list objects used with DependsOn (Fixes #982)
- Explicitly convert allocated_storage to integer before using it in comparisons (#983)
- Allow CreationPolicy override of props on WaitCondition (#988)
- "JobDefinitionName" property in JobDefinition class is not required (#995)
- ApiGateway::DomainName CertificateArn fix (#996)
- Tags support for SSM documents #999 (#1000)
- Add SSESpecification to DynamoDB::Table (#981)
- Add GitCloneDepth and InsecureSsl to CodeBuild Source
- Add Trippers property to CodeBuild::Project
- Add aurora*mysql to list of valid RDS engines
- Batch ContainerProperties is required
- Add Regions to Route53 HealthCheckConfiguration
- Add ClusterIdentifier to Redshift::Cluster
- Add DBClusterIdentifier to RDS::DBCluster
- Add TagSpecification to EC2::SpotFleet LaunchSpecifcations
- Add DisableScaleIn to ApplicationAutoScaling
- Add ApiKeySourceType and MinimumCompressionSize to ApiGateway::RestApi
- Add AutoScalingGroupName to AutoScaling::AutoScalingGroup

- Add `AWS::ApiGateway::VpcLink`
- Add `AWS::GuardDuty::Master` and `AWS::GuardDuty::Member`
- Add `AWS::SES`
- Add `GetCidr` function for `Fn::GetCidr`

8.5.12 2.2.0 (2018*01*29)

- Add `AWS::Inspector`
- Add `AWS::ServiceDiscovery`
- Add `InputProcessingConfiguration` to `KinesisAnalytics::Application`
- `EndpointConfiguration` in `ApiGateway::DomainName` is not required
- Allow setting `Subnets` and `SubnetMappings` properties on `ELBv2 LoadBalancers` (#934)
- increase `lambda` memory limit to support up to 3008 MB (#936)
- Stop validation if `CodeBuild Source Type` is a `Ref` (#940)
- Added support for `AutoPublishAlias` to `AWS::Serverless::Function` as specified https://github.com/awslabs/serverless*application*model/blob/master/versions/2016*10*31.md (#941)
- Add `resource_type` value and unit tests for `guardduty AWSObject`'s (#945)
- Added `elasticsearch` instance types for `m4`, `c4` and `r4` generations (#948)
- Correct type in `API Gateway GatewayResponse` type (#950)
- Fixes the `lifecyclepolicy` problem reported at `Issue #953` (#954)
- Add constants for `EC2 M5` instance types (#955)
- Adding support for `Block Device Mapping V2` (#960)
- Add support for `Policy Document` in `SAM` template. (#961)
- Stab at documenting `Troposphere basics` (#963)
- Adding `HealthCheckGracePeriodSeconds` into `ECS Service` (#966)
- Add `AllowedPattern` to `Parameter` (#968)
- Add long form parameter to `to_yaml` (#972)
- Use `S3.Filter` for the `serverless S3Event Filter` property
- Remove erroneous print in `tests/test_serverless.py`
- Add `FunctionForPackaging` class to `serverless`
- Add `AssociationName` to `AWS::SSM::Association`
- Update `S3::Bucket` with 20180123 property changes
- Add `DBSubnetGroupName` to `AWS::RDS::DBSubnetGroup`
- Add `ReservedConcurrentExecutions` to `AWS::Lambda::Function`
- Add `StreamEncryption` to `AWS::Kinesis::Stream`
- Add `LambdaOutput` to `KinesisAnalytics ApplicationOutput` property
- Update required fields in `IoT TopicRule DynamoDBAction`

- Add validator for InstanceTenancy in EC2::VPC
- Add CreditSpecification and ElasticGpuSpecifications to EC2::Instance

8.5.13 2.1.2 (2017*12*03)

- In SpotFleet::SpotFleetRequestConfigData SpotPrice is optional
- Add RoutingConfig to AWS::Lambda::Alias
- Update AWS::CodeDeploy
- Add CodeDeployLambdaAliasUpdate to UpdatePolicy
- Add AWS::GuardDuty
- Add AWS::Cloud9
- Add initial python resource spec generator
- Update AWS::CodeBuild::Project to 20171201 changes
- Change AWS::Batch::ComputeResources.Tags type to dict (#867)
- Update README for YAML template (#925)
- Typo fix in examples/ElastiCacheRedis.py (#926)
- Adds Fargate support to ECS types (#929)
- Fix SSM NotificationConfig validator type (#930)
- Fix SQS::Queue validation in the case of no QueueName specified (#931)

8.5.14 2.1.1 (2017*11*26)

- Add support for VPCOptions in ElasticSearch (#862)
- Add Description property for security group ingress and egress (#910)
- Add QueryLoggingConfig to Route53::HostedZone
- Add SourceRegion to RDS::DBInstance
- Add RootVolumeSize and caleDownBehavior to EMR::Cluster
- Add new properties to ElastiCache::ReplicationGroup
- Add LinuxParameters to ECS::TaskDefinition ContainerDefinitions
- Add LifecyclePolicy to ECR::Repository
- Add ScheduledActions to ApplicationAutoScaling::ScalableTarget
- Add new properties into ApiGateway

8.5.15 2.1.0 (2017*11*19)

- Output yaml (to_yaml) using cfn_flip (Fixes #567)
- Allow AWSHelperFn for CodeCommit Trigger Event(s) (#869)
- Adding the AWS::Glue resources (#872)

- Use a list for Serverless::Function Tags (#873)
- Support ProcessingConfiguration for Elasticsearch and Redshift (#876)
- Fixes incorrect class definition. (#877)
- Add TargetGroupInfo to DeploymentGroup #884 (#895)
- Reverting #810 as AWS has changed the casing again (#896)
- Add EMR Cluster MasterInstanceFleet and CoreInstanceFleet properties (#897)
- Add EMR Cluster CustomAmiId (#888) (#898)
- Add SecurityGroupRule Description property (#885) (#899)
- Add support for tags in AWS::KMS::Key. (#900)
- Adding OriginReadTimeout aka OriginResponseTimeout to cloudfront origin settings (#901)
- Added property for OriginKeepaliveTimeout
- Add CloudFrontOriginAccessIdentity type (#903)
- Added support for VpnTunnelOptionsSpecifications (#904)
- Allow ref on Parameter (#905)
- Adds Tags to Cloudfront Distribution (#906)
- CloudFront: add IPV6Enabled property for DistributionConfig (#908)
- Add OptionVersion to RDS:OptionConfigurations
- Add Tags to OpsWorks Layer and Stack
- Add LifecycleHookSpecification in AutoScalingGroup
- Add AmazonSideAsn to EC2::VPNGateway
- Add StateMachineName to StepFunctions::StateMachine
- Change KMS::Key to accept a standard Tags
- Add LambdaFunctionAssociations to CloudFront CacheBehaviors
- Add ResourceName to elasticbeanstalk OptionSettings
- Add AnalyticsConfigurations and InventoryConfigurations to S3::Bucket
- Add RequestValidatorId and OperationName to ApiGateway::Method
- Add deprecation warning for StageName in ApiGateway StageDescription
- Add AWS::CloudFront::StreamingDistribution

8.5.16 2.0.2 (2017*10*23)

- Set EC2 BlockDeviceMapping NoDevice property to type dict (#866)

8.5.17 2.0.1 (2017*10*21)

- Allow s3.Bucket AccessControl to be an AWSHelperFn
- Add AWS::ElasticLoadBalancingV2::ListenerCertificate
- Add serverless FunctionName and change how Tags are implemented

- Make AdjustmentType an optional property of ScalingPolicy as it is not used/supported for target (#849)
- Add maintenance window for SSM (#851)
- Add Tags, Tracing, KmsKeyArn, DLQ to serverless(SAM) (#853)
- Add new AWS::SSM resources (#854)
- EC2 NoDevice should be type boolean not dict (#858)
- Fixes RecordColumns cardinality for InputSchema and ReferenceSchema (#859)
- Make AWS::Batch::JobQueue::JobQueueName optional (#860)
- Fixes ApplicationOutput/Output cardinality (#863)

8.5.18 2.0.0 (2017*10*07)

- Note: the s3.Bucket change (#844) *may* cause a breaking change for non*named arguments.
- Add DefinitionBody to serverless API (#822)
- Adding kinesis stream source to firehose (#823)
- Add Event::Rule::Target::EcsParameters (#824)
- Add S3 Transfer Acceleration to AWS::S3::Bucket (#833)
- Add AvailabilityZone property to TargetDescription (#834)
- Add Tags to NATGateway (#835)
- Add ResourceLifecycleConfig to ElasticBeanstalk (#836)
- Add AWS::Athena::NamedQuery (#837)
- Added platformArn to Environment and ConfigurationTemplate (#839)
- Events target (fixes #830) (#840)
- Refactor s3.Bucket to remove custom __init__() and add tests (#844)
- Be more explicit on the use of the Tags object for Tags (#845)

8.5.19 1.9.6 (2017*09*24)

- Added missing EU_WEST_2 constants. (#776)
- Override object validation (#780)
- Update PyPI Information (#785)
- Adding IPv6 changes to AWS::EC2::Subnet (#786)
- NetworkACL Protocol Constants (#787)
- Add support for EFS encryption (#789)
- Add AWS::ApiGateway::GatewayResponse (#790)
- Add support for aurora*postgresql as a valid DB engine (#791)
- adding sqs server side encryption (#793)
- Support new code deploy options (#794)
- Add AWS Batch Support (#796)

- VPC expansion support (#797)
- Add NLB Functionality (#806)
- Fix typos in examples/DynamoDB_Table.py (#807)
- Revert “Accept Join type as parameter default value as it returns a string (#752)” (#808)
- Change Cognito UserPool SchemaAttribute required value to boolean (#809)
- Updating case of ‘AssignIPv6AddressOnCreation’ (#810)
- Fix spelling error to in RedshiftVPC example (#811)
- EFS example: SecurityGroupRule can’t be referred to as a Ref (#813)
- Update README.rst with current supported resources (#814)
- Add CloudTrail EventSelectors (#815)
- Add DAX support (#818)
- Add KinesisAnalytics support (#819)
- Add new ApiGateway resources (#820)
- Add autoscaling example for http requests that closes #630 (#821)
- Add new S3 Lifecycle Rule properties
- Add IoT DynamoDBv2Action and update DynamoDBAction properties
- Add EventSourceToken to Lambda::Permission
- Add new pseudo parameters
- Add DocumentationVersion to AWS::ApiGateway::Stage
- Add S3 Bucket MetricsConfiguration and fix TagFilter spelling
- Add TargetType to ELBv2::TargetGroup
- Add TargetTrackingConfiguration to AutoScaling::ScalingPolicy
- Add ReplaceUnhealthyInstances and Type to SpotFleetRequestConfigData
- Add ExtendedS3DestinationConfiguration to firehose DeliveryStream
- Add AWS::EC2::NetworkInterfacePermission

8.5.20 1.9.5 (2017*07*26)

- Add support for latest Cloudwatch alarms properties (#694)
- Raise ValueError for Outputs and Mappings * Fix Issue #732 (#733)
- Add AWS::EMR::SecurityConfiguration support (#738)
- Create CODE_OF_CONDUCT.md (#740)
- Added UsagePlans to API Gateway example (#741)
- EMR AutoScaling Complex Validation and Introduction of an ignore validator type (#743)
- Add PrivilegedMode option to CodeBuild Environments (#744)
- EFS DependsOn Ref to object fix (#746)
- README * add syntax highlighting (#747)

- Make handling of DependsOn more pythonic (#748)
- Accept Join type as parameter default value as it returns a string (#752)
- AWS SAM support (#754)
- Fixed UsagePlan example to proper Ref (#755)
- Fix cognito StringAttributeConstraints property names (Fixes #756)
- Add 'SourceAuth' property to CodeBuild Source (#758)
- Make it easier to get at hidden attributes (Fixes #760)
- Size/IOPS should be positive_integers (#761)
- Check that FIFO Queues end with .fifo (#757)
- Add AWS::CloudWatch::Dashboard (Fixes #763)
- Ulimit's HardLimit and SoftLimit validator change (#764)
- Adding EgressOnlyInternetGateway to EC2::Route (#765)
- Allow passing in a dict into DashboardBody (#767)
- Handle SQS QueueName using an AWSHelperFn (Fixes #773)
- LifecycleHook NotificationTargetARN and RoleARN are now optional
- Remove RoleArn from Events::Rule and add to Target property
- Add TracingConfig property to AWS::Lambda::Function
- Add Tags to some RedShift resources
- Add AWS::ApiGateway::DomainName
- Add AWS::EC2::EgressOnlyInternetGateway
- Add AWS::EMR::InstanceFleetConfig
- Add BinaryMediaTypes to ApiGateway::RestApi
- Add TargetTrackingScalingPolicyConfiguration
- Add TrailName to CloudTrail::Trail
- Add AlarmConfiguration and TriggerConfigurations
- Add Tags and TimeToLiveSpecification to DynamoDB::Table
- Add RetentionPeriodHours to Kinesis::Stream
- Add ReplicationSourceIdentifier to RDS::DBCluster
- Add LoggingProperties to Redshift::Cluster
- Add AWS Database Migration Service (DMS) support
- Add test target to Makefile
- Make it easier to download the latest CF resource spec
- **Added and reverted out of this release:**
 - Fix pycodestyle issue in tests/test_yaml.py
 - Output yaml (to_yaml) using cfn_flip (Fixes #567)
 - Special case If during parameter checking (Fixes #772)

- Raise TypeError when a scaler AWSHelperFn is used in a list context (#751)

8.5.21 1.9.4 (2017*06*04)

- Fix typo in S3_Bucket.py example (#696)
- Added .Ref & .GetAtt helper methods (#697)
- Add Pseudo Parameter Ref objects (#698)
- Fix NamespaceType typo in codebuild::Artifacts() (fixes #701)
- Add IpAddressType property to elbv2. (#703)
- Add new AWS::Lambda::Function Tags support (#705)
- Added ECS PlacementConstraints, PlacementStrategy, and ServiceName (#706)
- Add missing CidrIpv6 property to securityrule. (#710)
- Add missing properties to various objects in ec2.py (#711)
- logs.LogGroup#RetentionInDays is strictly defined list (#712)
- Add ManagedPolicyName to AWS::IAM::ManagedPolicy (Fixes #714)
- Add better validations for Parameter Default types (Fixes #717)
- Add AWS::Cognito (fixes #720)
- Add required attribute, JobFlowId, to EMR::InstanceGroupConfig (#722)
- Add WAFRegional support (#723)
- fix for ElastiCacheRedis.py example to use awacs (#725)
- Add EMR autoscaling (#729)
- Add SshUsername to AWS::OpsWorks::UserProfile
- Add PlacementConstraints to AWS::ECS::TaskDefinition
- Add MaximumExecutionFrequency to Config SourceDetails

8.5.22 1.9.3 (2017*04*13)

- Fix pycodestyle by using an explicit exception type
- Add more details to pycodestyle errors for travis runs
- Fix validator function exception test
- Remove limit check on conditions * Fixes #657
- Allow valid value for TargetGroup HealthCheckPort (#659)
- Added step functions and basic tests (#661)
- Adding example for CloudTrail (from http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws*resource*cloudtrail*trail.html) (#667)
- Fix ApiGateway.py sample (#666)
- Update comment on type checking
- Added missing props to ec2.NetworkInterfaces (#669) (#670)

- Add WAF Common Attacks Sample (#675)
- Updated constants with new instance types (#674)
- SSM Targets * fix spelling mistake (Value => Values) (#673)
- Do json validation on ApiGateway::Model Schema (Fix #679) (#681)
- SQS: Add FifoQueue and ContentBasedDeduplication (#687)
- VPCPeeringConnection: add PeerOwnerId & PeerRoleArn (#688)
- IAM: Add InstanceProfileName to InstanceProfile (#689)
- Add ApiGateway UsagePlanKey resource
- Add DeadLetterConfig property to Lambda::Function
- Add new subproperties to route53 and redshift (#690)
- Route53: ChildHealthChecks is a list of strings (#690)
- Fix typo in S3_Bucket_With_Versioning_And_Lifecycle_Rules.py (#693)
- Allow a dict to be passed in to initialize Tags (#692)
- Add SSM::Parameter
- Update autoscaling example to remove rebinding (#695)

8.5.23 1.9.2 (2017*01*29)

- Extra template validation (#635)
- Update ECS to Jan 17, 2017 release (#642)
- Add Timezone property to DBInstance (#643)
- Test Python 3.6 (#644)
- Adding RDS engine support for oracle*se2 (#646)
- Correct required in ecs.Service (#645)
- Add Separator property to IoT Firehose Action
- Add Fn::Split function (#647)
- Added to_dict() method to troposphere.Template (#651)
- Allow use of AWSHelperFn for IOPS (#652)
- Allow HelperFN w/ autoscaling policies (#654)

8.5.24 1.9.1 (2017*01*03)

- Improve readability of AssumeRolePolicyDocument attribute (#591)
- Add Environment to Lambda Function (#616)
- Adding DataSources to OpsWorks App and RdsDbInstances to OpsWorks Stack (#621)
- Added SNS::Subscription resource (SubscriptionResource) (#622)
- Added CodeBuild Project resource and a CodeBuild example (#624)
- Add back support for Python 2.6 (#626)

- Fixed missing `add_resource` in example Cloudwatch rule (#629)
- Create new property Environment for aws lambda Function (#631)
- Add `KmsKeyArn` to Lambda Function
- Add `CopyTagsToSnapshot` to `RDS::DBInstance`
- Fix `pycodestyle` issues with `examples/Lambda.py`
- Add `AWS::SSM::Association`
- Add `AWS::EC2::SubnetCidrBlock` and `AWS::EC2::VPCCidrBlock`
- Add `mutually_exclusive` validator
- Add `DocumentType` to `AWS::SSM::Document`
- Add OpsWorks Resources: `UserProfile` and `Volume`
- Update opsworks per 2016*11*22 changes
- Allow both dict and string for opswork `CustomJson`
- Add IPv6 support from 2016*12*01 update

8.5.25 1.9.0 (2016*11*15)

- Note: the dynamodb change below may cause backwards compatibility issues. There have been deprecation warnings for a while.
- Replace dynamodb module with dynamodb2 (#564)
- Add `CodeCommit` as a supported AWS resource type
- Add update of github Releases page to RELEASE doc
- Update elasticache for 2016*10*12 changes (#592)
- Support for S3 Lifecycle Rule property `NoncurrentVersionTransitions` (#596)
- Include resource title in required attr exception (#597)
- Added `Placement` class for the `Placement` property in `LaunchSpecifications`. (#598)
- Add EFS example (#601)
- Add support to old mysql db engine (#602)
- Fix typo in Example Allowed Values (#603)
- Remove `title` validation. Fixes #428 (#605)
- Add support for conditions in `cfn2py` script (#606)
- Added MongoDB default port to constants (#608)
- Add `HttpVersion` prop to `DistributionConfig` (CloudFront HTTP/2 Support) (#609)
- Added missing `QueryStringCacheKeys` property to `CloudFront ForwardedValues` (#612)
- Add a validator for ELB names (#615)

8.5.26 1.8.2 (2016*10*08)

- Add SpotPrice to SpotFleet LaunchSpecifications
- Add new properties to ECS (Clustername to Cluster and Family to TaskDefinition)
- Add Alias object to KMS (fixes #568)
- Added cross*stack references (#569)
- Handle lambda => awslambda mapping in cfn2py (#573)
- Add support for Tags to Certificate Manager Certificates (#574)
- Adding enhanced monitoring to rds.DBInstance (#575)
- Add support for LogGroupName in Logs::LogGroup (#576)
- Update Export param (#579)
- Add support for *Fn::Sub* (#582)
- RDS DBInstance Engine required even when DBSnapshotIdentifier is set (#583)
- Resource updates for 2016*10*06 changes (Fixes #584)
- Add AWS::ApiGateway::UsagePlan (fixes #585)
- Add AWS::CodeCommit::Repository (fixes #586)
- Provide better type checking for values in from_dict (#587)
- Allow HelperFn in UpdatePolicy for ASG (#588)
- Fixed from_dict case where you have a list of non BaseAWSObjects (#589)

8.5.27 1.8.1 (2016*09*12)

- Add TargetGroupArn and fix ContainerPort (#549)
- Update ApiGateway resources (#550)
- Add support for AutoScalingCreationPolicy (#552)
- Change param type for resource: RestAPI (#553)
- Add support for IAM Roles in ECS Task Definitions (#556)
- Allow Tags on AWS::CloudFormation::Stack (#557)
- Added support for protocol in container definition PortMapping property. (#558)
- Add Tags prop to Kinesis::Stream (#565)
- Add a sample ECS Cluster template (#559)
- Add support for ElasticsearchVersion in Elasticsearch Domain (#560)
- WAF SizeConstraint needs to be an AWSProperty (Fixes #561)
- Add Tags prop to Kinesis::Stream (#565)

8.5.28 1.8.0 (2016*08*15)

- Support “UserName” property for AWS::IAM::User #529
- Remove double S from S3ObjectVersion (fixes #530) (#531)
- Fix TemplateGenerator import logic. (#533)
- Add Name attributes for IAM groups and roles (#535)
- Automatically check if zip_file exceeds 4096 chars #537
- Add AWS Certificate Manager (#538)
- Add Application Auto Scaling (#539)
- CloudFront updates (Aug 9, 2016) (#540)
- Add PerformanceMode to FileSystem resource (#541)
- Add AWS Internet of Things (#542)
- Extend Template constructor. (#543)
- Add application loadbalancer objects and properties (#544)
- Improve check_zip_file to calculate a minimum length (#548)

8.5.29 1.7.0 (2016*07*07)

- Convert fake AWSHelperFns into AWSProperties (#478)
- cfn script: allow update (#484)
- Validate the template against AWS before stack creation (#485)
- Fix capitalization in README (#487)
- Remove duplicate waf FieldToMatch class (fixes #489)
- Tune validation logic and test cases for S3 bucket names (#486)
- waf XssMatchTuple should be an AWSProperty (Fixes #498)
- Allow setting a different region for S3 upload (#491)
- fix attribute for ApiKey (Enable -> Enabled) (#492)
- Invoke join correctly (#493)
- EMR: fix EBS configuration (#497)
- EMR: Action on Failure Fix (CONTINUE_AND_WAIT*>CANCEL_AND_WAIT) (#501)
- Rewritten the helper to be more flexible (#502)
- Added support for Kinesis Firehose (#505)
- Add support for VPC Flow Logs (#507)
- Syntax highlighting for readme python sample (#508)
- Added Name property to Kinesis streams (#510)
- Availability zones and EC2 instance type (#512)
- Add *AutoScalingReplacingUpdate* to *UpdatePolicy* (#513)

- Removed validation for DBSubnetGroupName when creating a read replica with SourceDBInstanceIdentifier (#515)
- EMR configurations values: also allow AWS helper functions (#516)
- Fix AssociationParameters Value type to list of strings (#518)
- Add DependsOn to Deployment and remove Enabled from StageKey (#519)
- Update fields in apigateway StageDescription (#521)
- Fix rename pep8*>pycodestyle and bump to fixed pyflakes (#522)
- Allows MultiAZ=false with AvailabilityZone in rds (#524)
- Do not require Status as a param in iam.AccessKey (#525)
- Fix badges in README

8.5.30 1.6.0 (2016*05*04)

- Remove unnecessary AWSHelperFn from props
- ReplicationConfigurationRules Destination is now an object (#380)
- Add WAF SizeConstraintSet and XssMatchSet
- Logs SubscriptionFilter (#413)
- Elasticsearch support (#415)
- Fixed ConfigSnapshotDeliveryProperties type (#420)
- Adding support for EMR resources (#421)
- Fix *ecs.TaskDefinition.Volumes* that was incorrectly flagged as required (#422)
- AWS::ECR test example (#423)
- Add cloudfront hostedzoneid for route53 (#427)
- Typo in variable name (431)
- ScalingAdjustment is an integer (#432)
- Add Compress to CloudFront (#433)
- Added missing S3OriginConfig parameter(#437)
- Allow both GetAtt and a basestring (#440)
- Add VpcConfig to AWS::Lambda::Function (#442)
- Add Version Resource to awslambda (#443)
- Add Alias Resource to awslambda (#444)
- Ignore If expression during validation of ASG (#446)
- Add test and tweak fix for ASG MaxSize If fix (#446)
- Provide Valid Lambda Function Memory Values for use in Parameters (#449)
- Add FunctionName to Lambda::Function (#452)
- Add support for EBS volume configuration in EMR resources (#453)
- Add elasticsearch instance type constants (#454)

- DomainName isn't a required parameter (#457)
- Create Documentation To Help Contributors (#458)
- Move Groups to property, add policy template version (#460)
- Fix Elasticsarch Domain object naming and add backward compatibility (#461)
- EC2 update FromPort, ToPort and Egress as optional (#463)
- ApiGateway Resources (#466)
- Added CloudWatch Events support (#467)
- Import JSON Templates (#468)
- Fix config Source object to take a list of SourceDetails (#469)
- Update Contribute Document to Use Requirements.txt (#470)
- Update to Apr 25, 2016 release (#471)
- Implement LifecycleRule Transitions property (#472)
- Better AWSHelperFn support in template generator (#473)
- Fix Bucket AccessControl to allow Ref (#475)
- Fix baseclass for AWS::Logs::Destination (#481)
- Add test for AWS::Logs::Destination (#482)

8.5.31 1.5.0 (2016*03*01)

- Add MariaDB to list of RDS engines [GH*368]
- Add ap*northeast [GH*373]
- Add T2 Nano [GH*374]
- capability support for cfn [GH*375]
- Update to resource list in documentation [GH*383]
- More info from validator function errors [GH*385]
- Add testing for python 3.5 [GH*388]
- Extended title validation [GH*389]
- EC2 NAT Gateway [GH*394]
- Add AWS::ECR::Repository [GH*395]
- Add KmsKeyId and StorageEncrypted to DBCluster [GH*396]
- Add awacs soft dependency [GH*397]
- New dynamodb2 module to replace dynamodb for consistent interface [GH*398]
- Add IsMultiRegionTrail support [GH*399]
- Add IncludeGlobalResourceTypes to RecordingGroup [GH*400]
- Capitalize examples [GH*404]
- use location constants for bucket creation in cfn [GH*409]

8.5.32 1.4.0 (2016*01*01)

- Add RDS Aurora support [GH*335]
- Change DeploymentGroup Ec2TagFilters to list [GH*337]
- Correct EC2 SpotFleet LaunchSpecifications [GH*338]
- RDS::DBCluster change AvailabilityZone to AvailabilityZones [GH*341]
- ECS LoadBalancerName property is a string [GH*342]
- CodeDeploy S3Location Version property is not a default requirement [GH*345]
- Add AutoEnableIO to AWS::EC2::Volume
- Only discard Properties in JSONrepr [GH*354]
- CodeDeploy added ApplicationName [GH*357]
- CodeDeploy DeploymentGroupName property missing [GH*358]
- Add in cloudfront properties for max, default [GH*360]
- Allow RDS iops to be 0 [GH*361]
- Add CodePipeline support [GH*362]
- Implemented CloudFormation changes from Dec 3 and Dec 28 [GH*366]
- Add AWS::Config, AWS::KMS, AWS::SSM

8.5.33 1.3.0 (2015*10*21)

- **Add new resources from 2015*10*01 CloudFormation release:**
 - AWS::CodeDeploy
 - AWS::DirectoryService::SimpleAD
 - AWS::EC2::PlacementGroup and AWS::EC2::SpotFleet
 - AWS::Lambda::EventSourceMapping and AWS::Lambda::Permission
 - AWS::Logs::SubscriptionFilter
 - AWS::RDS::DBCluster and AWS::RDS::DBClusterParameter
 - AWS::WorkSpaces::Workspace
- **Add updates to these resources from 2015*10*01 CloudFormation release:**
 - AWS::ElastiCache::ReplicationGroup
 - AWS::OpsWorks::Stack
 - AWS::OpsWorks::App
 - AWS::S3::Bucket
- Add ElastiCache (Redis) Example [GH*329]
- RDS: Added postgresql*license [GH*324]
- tail: only add unseen events [GH*327]
- Make Ref() work with datapipeline.ObjectField.RefValue [GH*328]

- Fix DeploymentGroup resource_type (AWS::CodeDeploy::DeploymentGroup) [GH*333]
- Add concatenation operator function `__add__` for Tags [GH*334]

8.5.34 1.2.2 (2015*09*15)

- Give more info about type errors [GH*312]
- Move `tail` within the troposphere library. This lets external libraries leverage this function [GH*315]
- Improve opsworks validation [GH*319]
- Fix RDS validation with conditional parameters [GH*320]

8.5.35 1.2.1 (2015*09*07)

- Bugfix for RDS Ref/GetAtt issue [GH*310]

8.5.36 1.2.0 (2015*09*04)

- Add support for EFS
- Elasticache: only validate az choices if azs is a list [GH*292]
- Add `from_dict` function to BaseAWSObject [GH*294]
- IAM: Path is optional for Role and InstanceProfile [GH*295]
- Validate parameter options based on Type [GH*296]
- RDS: Add more specific validators to DBInstance [GH*297]
- Add constants for the parameter types [GH*300]
- Add lambda ZipFile property [GH*301]
- Adds VPCEndpoint resource type [GH*304]
- Supports tags in ElasticBeanstalk environments [GH*308]
- Move cloudformation attribute setting to `__setattr__` [GH*309]

8.5.37 1.1.2 (2015*07*23)

- Clarify the license is a [BSD 2*Clause license](http://opensource.org/licenses/BSD*2*Clause)
- Add FindInMap type check for AutoScalingGroup validation of group sizes [GH*285]
- Implement the template Metadata section [GH*286]

8.5.38 1.1.1 (2015*07*12)

- Rename lambda*>awslambda [GH*268]
- Add t2 large instance type [GH*269]
- IAM: status required and managedpolicyarns [GH*272]
- Fix wrong prop name in rds.OptionGroup OptionGroupConfigurations*>OptionConfigurations [GH*274]

- Add CloudFormation CustomResource [GH*278]
- Add rds snapshot on delete example [GH*280]
- Unable to pass Cluster name as String [GH*281]
- Fix unable to set StringValue on ObjectField in DataPipeline [GH*283]

8.5.39 1.1.0 (2015*06*15)

- added AWS::CloudFormation::Stack NotificationARNs property [GH*243]
- Add additional import for PrivateIpAddressSpecification [GH*247]
- Add true s3 bucket name validator [GH*249]
- Replace strict *int* comparison by flexible *troposphere.validators.integer* [GH*251]
- Add validation for AZMode property on CacheCluster objects [GH*252]
- Fixing Opsworks Naming (ThresholdWaitTime -> ThresholdsWaitTime) [GH*253]
- Adding AutoScalingType to OpsWorks Instance [GH*255]
- Allow extending classes + tests [GH*257]
- Release June 11, 2015 [GH*259]
- Add M4 instances and Memcached port [GH*260]
- Add property for Subnet: MapPublicIpOnLaunch [GH*261]
- Minor improvements and fixes [GH*262]
- Update LoginProfile. Note: this is a breaking change and requires adding a `Password=` keyword parameter into LoginProfile. [GH*264]
- Add 2 additional properties (elasticache:CacheCluster:SnapshotName and opsworks:Layer:LifecycleEventConfiguration) [GH*265]

8.5.40 1.0.0 (2015*05*11)

- Fix two elasticache properties [GH*196]
- Add interim MinimumProtocolVersion to CloudFront ViewerCertificate [GH*218]
- Missing OriginPath in cloudfront.py [GH*220]
- Fix DBInstance constraints in order to allow the creation of RDS read*only replicas [GH*221]
- Add properties CharacterSetName, KmsKeyId, and StorageEncrypted to AWS::RDS::DBInstance [GH*224]
- Add Route53 HostedZoneVPCs, HostedZoneTags, HealthCheckTags
- Add new properties from 2015*04*16 CloudFormation release [GH*225, GH*240]
- Allow default region for GetAZs() [GH*232]
- Make AvailabilityZones parameter optional in AutoScalingGroup
- EventSubscription resource + EC2 types [GH*227]
- Python 3.4 support [GH*228]
- examples fix: users is list [GH*237]

- SNS Topic fields are not required [GH*230]
- Make AvailabilityZones parameter optional in AutoScalingGroup [GH*236]

8.5.41 0.7.2 (2015*03*23)

- Support AWS helper functions in lists during validation [GH*179]
- Update README [GH*183]
- Fixing RedshiftClusterInVpc example; incorrect SG setup [GH*186]
- Add optional NonKeyAttributes to DynamoDB Projection class [GH*188]
- Change AutoScaling ScheduledAction StartTime, EndTime, and Recurrence to optional [GH*189]
- CloudFront forwarded values required on cache behavior [GH*191]
- DynamoDB attribute definitions required [GH*192]
- Add some ec2 required fields [GH*193]
- Fix ElasticBeanstalk resources [GH*213]
- Fix iam Policy Resource/Property bug [GH*214]

8.5.42 0.7.1 (2015*01*11)

- Fix UpdatePolicy validation [GH*173]
- Add AWS::CloudFormation::Init ConfigSets support [GH*176]
- Change CloudWatch Alarm's Threshold prop to be basestring [GH*178]

8.5.43 0.7.0 (2015*01*02)

- Added new Google Group for discussion: https://groups.google.com/forum/#!forum/cloudtools*dev
- Fixing ValueError message to refer the correct package [GH*135]
- Change cfn to add R with no argument lists all the Stacks [GH*138]
- Add eu*central*1 region (Frankfurt) [GH*139]
- ConfigurationTemplate is an Object [GH*140]
- Release: AWS CloudFormation on 2014*11*06 [GH*141]
- Remove duplicate security_group from port [GH*143]
- UpdatePolicy and CreationPolicy [GH*144]
- Fixes duplicate key error reporting [GH*145]
- Fix warning in CloudFront example description [GH*148]
- Cfn script create bucket in the specified region [GH*149]
- Remove Unnecessary EOL whitespace [GH*150] Note: this changes the default JSON separators.
- More metadata options [GH*153]
- Metadata auth [GH*155]

- Fixed CreationPolicy [GH*157] [GH*160]
- Added AWS template VPC_Single_Instance_In_Subnet example [GH*162]
- Add 2014*12*24 CloudFormation release changes [GH*167] [GH*168] [GH*169]
- Add GSI & LSI Functionality [GH*161] [GH*172]
- Fixed landscape.io issues [GH*170]

8.5.44 0.6.2 (2014*10*09)

- Update to 2014*09*29 AWS release [GH*132]
- Add ElastiCache & Port # Constants [GH*132]
- Add ELB Tag Support [GH*133]
- Fix DBSecurityGroupIngress required properties [GH*134]

8.5.45 0.6.1 (2014*09*28)

- Update InitConfig per AWS docs [GH*120]
- S3 improvement + essential constants [GH*125]
- Allow FindInMap() for ec2.NetworkInterfaceProperty.GroupSet [GH*128]

8.5.46 0.6.0 (2014*08*26)

- Use subnet group for param, not vpc securitygroup [GH*65]
- Add support for Equals function and Condition [GH*66]
- Added ELB access logs and CrossZone test [GH*67]
- Added support for more condition functions [GH*69]
- Tweaked a few integer validation messages [GH*71]
- Fix resource.name backward compatibility regression
- Fix pep8 errors due to new pep8 1.5.x changes [GH*72]
- Allow Ref() in VPNGatewayRoutePropagation RouteTableIds list [GH*73]
- Add OpsWorks Support [GH*74]
- Add AutoScalingGroup TerminationPolicies [GH*77, GH*87]
- Add new property MetricsCollection [GH*79]
- Patching Users class to use basestring or Ref type for Groups [GH*80]
- Added support for Kinesis [GH*81]
- Allow autoscaling group to support 'min instances in service' and 'max size' values that are Refs [GH*82]
- Added support for Redshift [GH*84]
- Add DestinationSecurityGroupId in ec2.SecurityGroupRule [GH*85]
- Add CloudFront CacheBehavior [GH*86]

- Tweak UpdatePolicy properties [GH*88]
- Tweaks to rds.DNInstance [GH*89]
- Tweaks to EC2 DeviceIndex property values [GH*90]
- Fix AutoScalingGroup MinSize MaxSize [GH*92]
- Add Encrypted option to AWS::EC2::Volume [GH*96]
- Add missing config to s3.Bucket [GH*97]
- Add CloudFront DistributionConfig, CacheBehavior and DefaultCacheBehavior [GH*98]
- Add EC2 Instance InstanceInitiatedShutdownBehavior [GH*99]
- Updating the block device options for AutoScalingGroups [GH*100]
- Added support for AWS::CloudFormation::Init in AutoScalingGroup [GH*101]
- Added VPCPeering class [GH*102]
- Opworks CustomJson property expects a JSON object not a string [GH*103]
- Add support for VersioningConfiguration on S3 buckets [GH*104]
- Added Logs resource type [GH*105]
- Add PlacementGroup param to AutoScalingGroup [GH*111]
- Add VpcPeeringConnectionId parameter to EC2 Route [GH*113]
- Make RDS DBInstance MasterUsername and MasterPassword optional [GH*116]
- Add CloudTrail, tweaks to CloudWatch Alarm, and support route53 AliasTarget EvaluateTargetHealth [GH*117]
- Add LogDeliveryWrite canned ACL for S3 bucket [GH*118]

8.5.47 0.5.0 (2014*03*21)

- Add OpenStack native types [GH*61]
- Make *integer()* validator work with any integer*like object [GH*57]
- Add support to ELB ConnectionDrainingPolicy [GH*62]
- Add more OpenStack resource types and validation [GH*63]

8.5.48 0.4.0 (2014*02*19)

- Allow to extend resource classes by adding custom attributes [GH*16]
- Add AWS::ElastiCache::SubnetGroup [GH*27]
- Fix examples/VPC_EC2_Instance_With_Multiple_Dynamic_IPAddresses.py [GH*29]
- CacheSecurityGroupNames not required if using VpcSecurityGroupIds [GH*31]
- Add VPNConnectionRoute object and attribute to VPNConnection [GH*33]
- add new CrossZone option to ELB [GH*34]
- Add VPC_With_VPN_Connection example
- Fixup some of the network related validators and pep8 changes

- Add support for Tags and PortRange
- Add more resource name properties per CloudFormation release 2013*12*19
- Add Tier Environment property per CloudFormation release 2013*12*19
- Add VPNGatewayRoutePropagation per CloudFormation release 2013*11*22
- Add Tags properties per CloudFormation release 2013*09*24
- Add network changes from CloudFormation release 2013*09*17
- Canonicalize integer and bool values for more consistent output [GH*35]
- Add travis*ci for automated testing [GH*38]
- Check output of examples in test_examples [GH*40]
- Add support for conditions and for Fn::If() [GH*41]
- Tweak ELB ranges to match ec2 console [GH*43]
- Handle bool values better in cfn2py [GH*45]
- Allow strings (as well as Refs) for Subnet VpcId [GH*47]
- Add InstanceID to AutoScalingGroup and LaunchConfiguration
- ec2.DHCPOptions NTPservers -> NtpServers [GH*54]
- Add SQS dead letter queue from CloudFormation release 2014*01*29
- Add AutoScaling ScheduledAction from release 2014*01*27
- Add Tags for SecurityGroups [GH*55]
- RecordSets in Route53 not formatted correctly [GH*51]
- Allow Ref() in NetworkInterfaceProperty GroupSet list [GH*56]

8.5.49 0.3.4 (2013*12*05)

- Adding separators options to print to json function [GH*19]
- Add cfn2py script to convert json templates to Python [GH*22]
- Add EnableDnsSupport and EnableDnsHostnames properties for VPC [GH*23]
- Add VPC support to elasticache [GH*24]
- Fix missing Import Ref [GH*26]
- Add missing AWS::SQS::Queue properties
- Add resource naming (Name Type)
- Allow Ref's in the list objects

8.5.50 0.3.3 (2013*10*04)

- Fix Ref() to output the name only [GH*17]
- Add Ref test.
- Fix some IAM issues

8.5.51 0.3.2 (2013*09*25)

- Convert VPCDHCOptionsAssociation to not have `__init__`
- Fix Output, Parameter and UpdatePolicy to not output a Properties dict
- Raise a ValueError if adding a duplicate object to the template
- Set the correct dictname for UpdatePolicy

8.5.52 0.3.1 (2013*09024)

- Make the code more DRY [GH*15]
- Add a optional *name* argument to AWSProperty constructor
- Add ability to push large stack templates to S3
- InstanceType is not required (defaults to m1.small)
- Add AssociatePublicIpAddress property for AutoScaling LaunchConfiguration
- Make Tags an AWSHelperFn to make it easier to assign
- Resource property types should not be in a Properties dictionary
- Clean up “required” error checking and handle property types better

8.5.53 0.3.0 (2013*08*07)

- Do not validate AWSHelperFun’s [GH*8] [GH*9]
- Add missing return in integer_range [GH*10]
- integer_range validator for ELB HealthyCheckInt
- Convert RDS::DBInstance::VPCSecurityGroups to new list type checking
- VPCSecurityGroups for RDS should be a list, not a basestring

8.5.54 0.2.9 (2013*05*07)

- Fixing ELB LoadBalancerPorts not required (error in the AWS docs)

8.5.55 0.2.8 (2013*04*24)

- EC2 SecurityGroup Egress & Ingress rules should be objects
- Fix Attributes validator for ELB Policies
- Allow PolicyDocuments to use (if present) awacs Policy objects
- Add test for matching against a tuple of types

8.5.56 0.2.6 (2013*03*26)

- Add cfn script to create, tail and show stack resources

8.5.57 0.2.5 (2013*03*25)

- UpdatePolicy validation enhancements [GH*5]
- Add VPCSecurityGroups property to AWS::RDS::DBInstance
- DefaultCacheBehavior is a required property for DistributionConfig
- Fix CustomGateway -> CustomerGateway
- Domain does not have any properties
- Fix VPC security group rule bugs
- Added EBSBlockDevice and BlockDeviceMapping classes.
- Add a post*validator to allow individual object to validate themselves
- Add ability to use validate functions on property values
- Add validation of list element types
- Add unit tests

8.5.58 0.2.0 (2013*02*26)

- Add support for autoscaling notification configurations [GH*2]
- Add support for AWS::ElastiCache and AWS::RDS
- Move to AWSProperty entirely.
- Add shortcuts for the NotificationTypes strings
- Add Python 3.x compatibility support
- Pass pep8 and pyflakes

8.5.59 0.1.2 (2013*02*21)

- First PyPI release
- Add S3 bucket support [GH*1]

8.6 Contributing

8.6.1 tl;dr

1. Fork <https://github.com/cloudtools/troposphere>
2. Make the code better
3. Make the code pass tests
4. Create a Pull Request

8.6.2 How to Get Help

We have a Google Group, [cloudtools-dev](#), where you can ask questions and engage with the troposphere community. Issues and pull requests are always welcome!

8.6.3 How to Test Your Code

The latest test scripts can be found at [Travis-CI](#). If you look at the details of a job, you can see what automated tests will be run against any commits to the project.

1. Create a virtualenv (e.g. `virtualenv ~/virtualenv/troposphere`)
2. Activate it: `source ~/virtualenv/troposphere/bin/activate`
3. `pip install --upgrade pip setuptools wheel`
4. `pip install -r requirements-dev.txt`
5. **Run tests**
 - (a) `pycodestyle .`
 - (b) `pyflakes .`
 - (c) `python setup.py test`
6. Alternatively, `make` can be used to run the tests, i.e. `make test`.

Tests are run against Python 2.7, 3.4, 3.5 and 3.6.

8.6.4 Contributing Example Code

New example code should go into `troposphere/examples`. The expected CloudFormation Template should be stored in `troposphere/tests/examples_output/`. When tests are run the output of the code in the examples directory will be compared with the expected results in the `example_output` directory.

8.7 Releasing

8.7.1 Steps to release a new version

- Change version in `troposphere/__init__.py`
- Update `CHANGELOG.md` with changes made since last release
- Create a signed tag: `git tag --sign -m "Release 1.1.1" 1.1.1`
- Create PyPI release: `python setup.py sdist upload --sign`
- Push commits: `git push`
- Push tag: `git push --tags`
- Update github release page: <https://github.com/cloudtools/troposphere/releases>

8.7.2 Helper to create CHANGELOG entries

```
git log --reverse --pretty=format:"%s" | tail -100 | sed 's/^/* /'
```

8.7.3 Helper to list supported resources

```
grep -h 'resource_type = "AWS::"' troposphere/* | sed 's/[ ]*resource_type =  
"'"// | cut -f1-3 -d: | sort | uniq | sed 's/^/- /'
```

8.8 Code of Conduct

8.8.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and expression, level of experience, nationality, personal appearance, race, religion, or sexual identity and orientation.

8.8.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

8.8.3 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

8.8.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

8.8.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at cloudtools-maintainers@groups.google.com. The project team will review and investigate all complaints, and will respond in a way that it deems appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

8.8.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant [homepage](#), version 1.4.

8.9 License

Copyright (c) 2012-2017, Mark Peek <mark@peek.org> All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

CHAPTER 9

Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

t

tests, 146
tests.test_apigateway, 132
tests.test_apigatewayv2, 132
tests.test_appsync, 133
tests.test_asg, 133
tests.test_awslambda, 133
tests.test_basic, 133
tests.test_cloudformation, 136
tests.test_cloudwatch, 136
tests.test_codebuild, 136
tests.test_codecommit, 136
tests.test_config, 137
tests.test_dict, 137
tests.test_ec2, 137
tests.test_ecs, 137
tests.test_efs, 138
tests.test_elasticloadbalancerv2, 138
tests.test_emr, 138
tests.test_examples, 139
tests.test_examples_template_generator, 139
tests.test_guardduty, 139
tests.test_int_type, 139
tests.test_iot1click, 140
tests.test_logs, 140
tests.test_opsworks, 140
tests.test_parameters, 140
tests.test_policies, 140
tests.test_rds, 141
tests.test_s3, 142
tests.test_serverless, 142
tests.test_sqs, 143
tests.test_stepfunctions, 143
tests.test_tags, 143
tests.test_template, 143
tests.test_template_generator, 144
tests.test_userdata, 144
tests.test_validators, 145
tests.test_yaml, 146
troposphere, 129
troposphere.amazonmq, 29
troposphere.analytics, 30
troposphere.apigateway, 32
troposphere.apigatewayv2, 35
troposphere.applicationautoscaling, 37
troposphere.appstream, 38
troposphere.appsync, 39
troposphere.ask, 41
troposphere.athena, 41
troposphere.autoscaling, 41
troposphere.autoscalingplans, 43
troposphere.awslambda, 45
troposphere.batch, 46
troposphere.budgets, 47
troposphere.certificatemanager, 48
troposphere.cloud9, 48
troposphere.cloudformation, 49
troposphere.cloudfront, 50
troposphere.cloudtrail, 52
troposphere.cloudwatch, 52
troposphere.codebuild, 53
troposphere.codecommit, 54
troposphere.codedeploy, 55
troposphere.codepipeline, 57
troposphere.cognito, 58
troposphere.config, 60
troposphere.constants, 62
troposphere.datapipeline, 62
troposphere.dax, 62
troposphere.directoryservice, 63
troposphere.dlm, 63
troposphere.dms, 64
troposphere.docdb, 65
troposphere.dynamodb, 65
troposphere.dynamodb2, 66
troposphere.ec2, 66
troposphere.ecr, 75
troposphere.ecs, 75

- troposphere.efs, 77
- troposphere.eks, 78
- troposphere.elasticache, 78
- troposphere.elasticbeanstalk, 79
- troposphere.elasticloadbalancing, 80
- troposphere.elasticloadbalancingv2, 81
- troposphere.elasticsearch, 83
- troposphere.emr, 83
- troposphere.events, 86
- troposphere.firehose, 87
- troposphere.glue, 88
- troposphere.guardduty, 91
- troposphere.helpers, 26
- troposphere.helpers.userdata, 26
- troposphere.iam, 92
- troposphere.inspector, 93
- troposphere.iot, 94
- troposphere.iot1click, 96
- troposphere.iotanalytics, 96
- troposphere.kinesis, 98
- troposphere.kms, 99
- troposphere.logs, 99
- troposphere.neptune, 99
- troposphere.openstack, 29
- troposphere.openstack.heat, 26
- troposphere.openstack.neutron, 26
- troposphere.openstack.nova, 28
- troposphere.opsworks, 100
- troposphere.policies, 102
- troposphere.rds, 103
- troposphere.redshift, 105
- troposphere.route53, 106
- troposphere.s3, 107
- troposphere.sagemaker, 111
- troposphere.sdb, 112
- troposphere.secretsmanager, 112
- troposphere.serverless, 113
- troposphere.servicecatalog, 116
- troposphere.servicediscovery, 117
- troposphere.ses, 118
- troposphere.sns, 120
- troposphere.sqs, 121
- troposphere.ssm, 121
- troposphere.stepfunctions, 123
- troposphere.template_generator, 123
- troposphere.utils, 124
- troposphere.validators, 124
- troposphere.waf, 125
- troposphere.wafregional, 127
- troposphere.workspaces, 128

A

- AbortIncompleteMultipartUpload (class in troposphere.s3), 107
- AccelerateConfiguration (class in troposphere.s3), 107
- AcceptedPortfolioShare (class in troposphere.servicecatalog), 116
- access_control_types (troposphere.s3.Bucket attribute), 108
- AccessControlTranslation (class in troposphere.s3), 107
- AccessKey (class in troposphere.iam), 92
- AccessLoggingPolicy (class in troposphere.elasticloadbalancing), 80
- AccessLogSetting (class in troposphere.apigateway), 32
- AccessLogSettings (class in troposphere.apigatewayv2), 35
- Account (class in troposphere.apigateway), 32
- AccountAggregationSources (class in troposphere.config), 60
- Action (class in troposphere.elasticloadbalancingv2), 81
- Action (class in troposphere.glue), 88
- Action (class in troposphere.iot), 94
- Action (class in troposphere.iotanalytics), 96
- Action (class in troposphere.ses), 118
- Action (class in troposphere.waf), 125
- Action (class in troposphere.wafregional), 127
- action_on_failure_validator() (in module troposphere.emr), 86
- Actions (class in troposphere.codepipeline), 57
- ActionTypeId (class in troposphere.codepipeline), 57
- Activity (class in troposphere.iotanalytics), 96
- Activity (class in troposphere.stepfunctions), 123
- ActivityChannel (class in troposphere.iotanalytics), 96
- add_condition() (troposphere.Template method), 131
- add_description() (troposphere.Template method), 131
- add_mapping() (troposphere.Template method), 131
- add_metadata() (troposphere.Template method), 131
- add_output() (troposphere.Template method), 131
- add_parameter() (troposphere.Template method), 131
- add_parameter_to_group() (troposphere.Template method), 131
- add_resource() (troposphere.Template method), 131
- add_to_template() (troposphere.BaseAWSObject method), 129
- add_to_template() (troposphere.Output method), 130
- add_to_template() (troposphere.Parameter method), 130
- add_transform() (troposphere.Template method), 131
- add_version() (troposphere.Template method), 131
- AddAttributes (class in troposphere.iotanalytics), 96
- AddHeaderAction (class in troposphere.ses), 118
- additional_info_validator() (in module troposphere.emr), 86
- AddressPair (class in troposphere.openstack.neutron), 26
- AdminCreateUserConfig (class in troposphere.cognito), 58
- AggregationAuthorization (class in troposphere.config), 60
- Alarm (class in troposphere.cloudwatch), 52
- Alarm (class in troposphere.codedeploy), 55
- AlarmConfiguration (class in troposphere.codedeploy), 55
- AlarmIdentifier (class in troposphere.route53), 106
- AlexaSkillEvent (class in troposphere.serverless), 113
- Alias (class in troposphere.awslambda), 45
- Alias (class in troposphere.kms), 99
- AliasRoutingConfiguration (class in troposphere.awslambda), 45
- AliasTarget (class in troposphere.route53), 106
- AmazonRedshiftParameter (class in troposphere.redshift), 105
- AnalyticsConfiguration (class in troposphere.s3), 108
- And (class in troposphere), 129
- Api (class in troposphere.apigatewayv2), 35
- Api (class in troposphere.serverless), 113
- ApiEvent (class in troposphere.serverless), 113
- ApiKey (class in troposphere.apigateway), 32
- ApiKey (class in troposphere.appsync), 39
- ApiStage (class in troposphere.apigateway), 32
- App (class in troposphere.opsworks), 100

- AppCookieStickinessPolicy (class in troposphere.elasticloadbalancing), 80
 - Application (class in troposphere.analytics), 30
 - Application (class in troposphere.codedeploy), 55
 - Application (class in troposphere.elasticbeanstalk), 79
 - Application (class in troposphere.emr), 83
 - ApplicationOutput (class in troposphere.analytics), 30
 - ApplicationReferenceDataSource (class in troposphere.analytics), 30
 - ApplicationResourceLifecycleConfig (class in troposphere.elasticbeanstalk), 79
 - ApplicationSettings (class in troposphere.appstream), 38
 - ApplicationSource (class in troposphere.autoscalingplans), 43
 - ApplicationVersion (class in troposphere.elasticbeanstalk), 79
 - ApplicationVersionLifecycleConfig (class in troposphere.elasticbeanstalk), 79
 - ArtifactDetails (class in troposphere.codepipeline), 57
 - Artifacts (class in troposphere.codebuild), 53
 - ArtifactStore (class in troposphere.codepipeline), 57
 - ArtifactStoreMap (class in troposphere.codepipeline), 57
 - assertNoIntType() (tests.test_int_type.TestIntTypeShouldNotBeUsed method), 139
 - AssessmentTarget (class in troposphere.inspector), 93
 - AssessmentTemplate (class in troposphere.inspector), 93
 - Association (class in troposphere.ssm), 121
 - AssociationParameters (class in troposphere.ec2), 66
 - attribute_type_validator() (in module troposphere.dynamodb), 66
 - AttributeDefinition (class in troposphere.dynamodb), 65
 - AttributeType (class in troposphere.cognito), 58
 - Auth (class in troposphere.serverless), 113
 - Authentication (class in troposphere.cloudformation), 49
 - AuthenticationBlock (class in troposphere.cloudformation), 49
 - AuthenticationConfiguration (class in troposphere.ask), 41
 - AuthorizationConfig (class in troposphere.appsync), 39
 - Authorizer (class in troposphere.apigateway), 32
 - Authorizer (class in troposphere.apigatewayv2), 36
 - Authorizers (class in troposphere.serverless), 113
 - AutoRollbackConfiguration (class in troposphere.codedeploy), 55
 - AutoScalingCreationPolicy (class in troposphere.policies), 102
 - AutoScalingGroup (class in troposphere.autoscaling), 41
 - AutoScalingPolicy (class in troposphere.emr), 83
 - AutoScalingReplacingUpdate (class in troposphere.policies), 102
 - AutoScalingRollingUpdate (class in troposphere.policies), 103
 - AutoScalingScheduledAction (class in troposphere.policies), 103
 - AutoScalingThresholds (class in troposphere.opsworks), 100
 - AWSAttribute (class in troposphere), 129
 - AWSAutoScalingGroup (class in troposphere.openstack.heat), 26
 - AWSCustomObject (class in troposphere.cloudformation), 49
 - AWSDeclaration (class in troposphere), 129
 - AWSHelperFn (class in troposphere), 129
 - AwsIamConfig (class in troposphere.appsync), 39
 - AWSObject (class in troposphere), 129
 - AWSProperty (class in troposphere), 129
 - AwsvpcConfiguration (class in troposphere.ecs), 75
- ## B
- Base64 (class in troposphere), 129
 - BaseAWSObject (class in troposphere), 129
 - BasePathMapping (class in troposphere.apigateway), 32
 - BaseRecordSet (class in troposphere.route53), 106
 - billing_mode_validator() (in module troposphere.dynamodb), 66
 - BlockDeviceMapping (class in troposphere.autoscaling), 41
 - BlockDeviceMapping (class in troposphere.ec2), 66
 - BlockDeviceMapping (class in troposphere.openstack.nova), 28
 - BlockDeviceMapping (class in troposphere.opsworks), 100
 - BlockDeviceMappingV2 (class in troposphere.openstack.nova), 28
 - Blockers (class in troposphere.codepipeline), 57
 - boolean() (in module troposphere.validators), 124
 - BootstrapActionConfig (class in troposphere.emr), 83
 - BounceAction (class in troposphere.ses), 119
 - Broker (class in troposphere.amazonmq), 29
 - Bucket (class in troposphere.s3), 108
 - BucketEncryption (class in troposphere.s3), 108
 - BucketPolicy (class in troposphere.s3), 108
 - Budget (class in troposphere.budgets), 47
 - BudgetData (class in troposphere.budgets), 48
 - BufferingHints (class in troposphere.firehose), 87
 - ByteMatchSet (class in troposphere.waf), 125
 - ByteMatchSet (class in troposphere.wafregional), 127
 - ByteMatchTuples (class in troposphere.waf), 125
 - ByteMatchTuples (class in troposphere.wafregional), 127
- ## C
- CacheBehavior (class in troposphere.cloudfront), 50
 - CacheCluster (class in troposphere.elasticache), 78
 - call_correct() (in module tests.test_basic), 136
 - call_incorrect() (in module tests.test_basic), 136
 - CanarySetting (class in troposphere.apigateway), 32
 - Certificate (class in troposphere.certificatemanager), 48
 - Certificate (class in troposphere.dms), 64

- Certificate (class in troposphere.elasticloadbalancingv2), 81
- Certificate (class in troposphere.iot), 94
- Channel (class in troposphere.iotanalytics), 96
- check_ports() (in module troposphere.ec2), 75
- check_required() (in module troposphere.validators), 124
- check_zip_file() (troposphere.awslambda.Code static method), 45
- ChefConfiguration (class in troposphere.opsworks), 100
- Cidr (class in troposphere), 130
- ClassicLoadBalancer (class in troposphere.ec2), 66
- ClassicLoadBalancersConfig (class in troposphere.ec2), 66
- Classifier (class in troposphere.glue), 88
- ClientCertificate (class in troposphere.apigateway), 33
- CloudFormationProduct (class in troposphere.servicecatalog), 116
- CloudFormationProvisionedProduct (class in troposphere.servicecatalog), 116
- cloudfront_event_type() (in module troposphere.validators), 124
- cloudfront_forward_type() (in module troposphere.validators), 124
- cloudfront_restriction_type() (in module troposphere.validators), 124
- cloudfront_viewer_protocol_policy() (in module troposphere.validators), 124
- CloudFrontOriginAccessIdentity (class in troposphere.cloudfront), 50
- CloudFrontOriginAccessIdentityConfig (class in troposphere.cloudfront), 50
- CloudwatchAlarmAction (class in troposphere.iot), 94
- CloudWatchAlarmDefinition (class in troposphere.emr), 84
- CloudWatchDestination (class in troposphere.ses), 119
- CloudWatchEvent (class in troposphere.serverless), 113
- CloudWatchLoggingOptions (class in troposphere.firehose), 87
- CloudWatchLogs (class in troposphere.codebuild), 53
- CloudwatchMetricAction (class in troposphere.iot), 94
- Cluster (class in troposphere.dax), 62
- Cluster (class in troposphere.ecs), 75
- Cluster (class in troposphere.eks), 78
- Cluster (class in troposphere.emr), 84
- Cluster (class in troposphere.redshift), 105
- ClusterParameterGroup (class in troposphere.redshift), 105
- ClusterSecurityGroup (class in troposphere.redshift), 105
- ClusterSecurityGroupIngress (class in troposphere.redshift), 105
- ClusterSubnetGroup (class in troposphere.redshift), 105
- Code (class in troposphere.awslambda), 45
- CodeDeployLambdaAliasUpdate (class in troposphere.policies), 103
- CognitoAuth (class in troposphere.serverless), 113
- CognitoAuthIdentity (class in troposphere.serverless), 113
- CognitoIdentityProvider (class in troposphere.cognito), 58
- CognitoStreams (class in troposphere.cognito), 58
- Column (class in troposphere.glue), 89
- compliance_level() (in module troposphere.validators), 124
- ComputeCapacity (class in troposphere.appstream), 38
- ComputeEnvironment (class in troposphere.batch), 46
- ComputeEnvironmentOrder (class in troposphere.batch), 46
- ComputeResources (class in troposphere.batch), 46
- Condition (class in troposphere), 130
- Condition (class in troposphere.elasticloadbalancingv2), 81
- Condition (class in troposphere.glue), 89
- Condition (class in troposphere.guardduty), 91
- ConfigRule (class in troposphere.config), 61
- ConfigSnapshotDeliveryProperties (class in troposphere.config), 61
- Configuration (class in troposphere.amazonmq), 29
- Configuration (class in troposphere.emr), 84
- ConfigurationAggregator (class in troposphere.config), 61
- ConfigurationAssociation (class in troposphere.amazonmq), 29
- ConfigurationId (class in troposphere.amazonmq), 30
- ConfigurationProperties (class in troposphere.codepipeline), 57
- ConfigurationRecorder (class in troposphere.config), 61
- ConfigurationSet (class in troposphere.ses), 119
- ConfigurationSetEventDestination (class in troposphere.ses), 119
- ConfigurationTemplate (class in troposphere.elasticbeanstalk), 79
- Connection (class in troposphere.glue), 89
- connection_type_validator() (in module troposphere.glue), 91
- ConnectionDrainingPolicy (class in troposphere.elasticloadbalancing), 80
- ConnectionInput (class in troposphere.glue), 89
- ConnectionSettings (class in troposphere.elasticloadbalancing), 80
- ConnectionsList (class in troposphere.glue), 89
- ContainerAction (class in troposphere.iotanalytics), 96
- ContainerDefinition (class in troposphere.ecs), 75
- ContainerDefinition (class in troposphere.sagemaker), 111
- ContainerProperties (class in troposphere.batch), 46
- Content (class in troposphere.awslambda), 45
- Cookies (class in troposphere.cloudfront), 51
- CopyCommand (class in troposphere.firehose), 87
- Cors (class in troposphere.serverless), 113

- CorsConfiguration (class in troposphere.s3), 108
 - CorsRules (class in troposphere.s3), 108
 - CostTypes (class in troposphere.budgets), 48
 - Crawler (class in troposphere.glue), 89
 - create_answer() (tests.test_userdata.TestUserdata method), 144
 - create_result() (tests.test_userdata.TestUserdata method), 144
 - create_test_class() (in module tests.test_examples), 139
 - create_test_class() (in module tests.test_examples_template_generator), 139
 - CreateRule (class in troposphere.dlm), 63
 - CreationPolicy (class in troposphere.policies), 103
 - CreditSpecification (class in troposphere.ec2), 66
 - CSVMappingParameters (class in troposphere.analytics), 30
 - CustomActionType (class in troposphere.codepipeline), 57
 - CustomerGateway (class in troposphere.ec2), 67
 - CustomErrorResponse (class in troposphere.cloudfront), 51
 - CustomizedLoadMetricSpecification (class in troposphere.autoscalingplans), 43
 - CustomizedMetricSpecification (class in troposphere.applicationautoscaling), 37
 - CustomizedMetricSpecification (class in troposphere.autoscaling), 41
 - CustomizedScalingMetricSpecification (class in troposphere.autoscalingplans), 44
 - CustomOriginConfig (class in troposphere.cloudfront), 51
 - CustomResource (class in troposphere.cloudformation), 49
- D**
- Dashboard (class in troposphere.cloudwatch), 52
 - Database (class in troposphere.glue), 89
 - DatabaseInput (class in troposphere.glue), 89
 - DataExport (class in troposphere.s3), 108
 - DataResource (class in troposphere.cloudtrail), 52
 - Dataset (class in troposphere.iotanalytics), 96
 - DatasetContentVersionValue (class in troposphere.iotanalytics), 97
 - DataSource (class in troposphere.appsync), 39
 - DataSource (class in troposphere.opsworks), 100
 - Datastore (class in troposphere.iotanalytics), 97
 - DBCluster (class in troposphere.docdb), 65
 - DBCluster (class in troposphere.neptune), 99
 - DBCluster (class in troposphere.rds), 103
 - DBClusterParameterGroup (class in troposphere.docdb), 65
 - DBClusterParameterGroup (class in troposphere.neptune), 100
 - DBClusterParameterGroup (class in troposphere.rds), 103
 - DBInstance (class in troposphere.docdb), 65
 - DBInstance (class in troposphere.neptune), 100
 - DBInstance (class in troposphere.rds), 103
 - DBParameterGroup (class in troposphere.neptune), 100
 - DBParameterGroup (class in troposphere.rds), 103
 - DBSecurityGroup (class in troposphere.rds), 104
 - DBSecurityGroupIngress (class in troposphere.rds), 104
 - DBSubnetGroup (class in troposphere.docdb), 65
 - DBSubnetGroup (class in troposphere.neptune), 100
 - DBSubnetGroup (class in troposphere.rds), 104
 - DeadLetterConfig (class in troposphere.awslambda), 45
 - DeadLetterQueue (class in troposphere.serverless), 113
 - DefaultCacheBehavior (class in troposphere.cloudfront), 51
 - defaultPropagateAtLaunch (troposphere.autoscaling.Tags attribute), 43
 - defer() (in module troposphere.validators), 124
 - delete_behavior_validator() (in module troposphere.glue), 91
 - delivery_stream_type_validator() (in module troposphere.firehose), 88
 - DeliveryChannel (class in troposphere.config), 61
 - DeliveryStream (class in troposphere.firehose), 87
 - DeltaTime (class in troposphere.iotanalytics), 97
 - depends_on_helper() (in module troposphere), 132
 - Deployment (class in troposphere.apigateway), 33
 - Deployment (class in troposphere.apigatewayv2), 36
 - Deployment (class in troposphere.codedeploy), 55
 - deployment_option_validator() (in module troposphere.codedeploy), 57
 - deployment_type_validator() (in module troposphere.codedeploy), 57
 - DeploymentCanarySetting (in module troposphere.apigateway), 33
 - DeploymentCanarySettings (class in troposphere.apigateway), 33
 - DeploymentConfig (class in troposphere.codedeploy), 55
 - DeploymentConfiguration (class in troposphere.ecs), 75
 - DeploymentGroup (class in troposphere.codedeploy), 55
 - DeploymentPreference (class in troposphere.serverless), 113
 - DeploymentStyle (class in troposphere.codedeploy), 55
 - DEPRECATED_MODULES (troposphere.template_generator.TemplateGenerator attribute), 124
 - Destination (class in troposphere.logs), 99
 - Destination (class in troposphere.s3), 108
 - DestinationSchema (class in troposphere.analytics), 30
 - Detector (class in troposphere.guardduty), 91
 - DevEndpoint (class in troposphere.glue), 89
 - Device (class in troposphere.ecs), 76
 - Device (class in troposphere.iot1click), 96
 - DeviceConfiguration (class in troposphere.cognito), 59

- DeviceRegistryEnrich (class in troposphere.iotanalytics), 97
- DeviceShadowEnrich (class in troposphere.iotanalytics), 97
- DHCPOptions (class in troposphere.ec2), 67
- dictname (troposphere.AWSAttribute attribute), 129
- dictname (troposphere.AWSObject attribute), 129
- dictname (troposphere.AWSProperty attribute), 129
- dictname (troposphere.cloudformation.AWSCustomObject attribute), 49
- DimensionConfiguration (class in troposphere.ses), 119
- DirectoryConfig (class in troposphere.appstream), 38
- DisableInboundStageTransitions (class in troposphere.codepipeline), 57
- Distribution (class in troposphere.cloudfront), 51
- DistributionConfig (class in troposphere.cloudfront), 51
- DnsConfig (class in troposphere.servicediscovery), 117
- DnsRecord (class in troposphere.servicediscovery), 117
- DockerVolumeConfiguration (class in troposphere.ecs), 76
- Document (class in troposphere.ssm), 121
- DocumentationPart (class in troposphere.apigateway), 33
- DocumentationVersion (class in troposphere.apigateway), 33
- Domain (class in troposphere.elasticsearch), 83
- Domain (class in troposphere.sdb), 112
- DomainJoinInfo (class in troposphere.appstream), 38
- DomainName (class in troposphere.apigateway), 33
- DomainValidationOption (class in troposphere.certificatemanager), 48
- double() (in module troposphere.validators), 124
- DynamoDBAction (class in troposphere.iot), 94
- DynamoDBConfig (class in troposphere.appsync), 39
- DynamoDBEvent (class in troposphere.serverless), 114
- DynamoDBSettings (class in troposphere.dms), 64
- DynamoDBv2Action (class in troposphere.iot), 94
- ## E
- EBSBlockDevice (class in troposphere.autoscaling), 41
- EBSBlockDevice (class in troposphere.ec2), 67
- EbsBlockDevice (class in troposphere.opsworks), 100
- EbsBlockDeviceConfigs (class in troposphere.emr), 84
- EbsConfiguration (class in troposphere.emr), 84
- EBSOptions (class in troposphere.elasticsearch), 83
- EC2Fleet (class in troposphere.ec2), 67
- Ec2TagFilters (class in troposphere.codedeploy), 55
- Ec2TagSet (class in troposphere.codedeploy), 55
- Ec2TagSetListObject (class in troposphere.codedeploy), 56
- EcsParameters (class in troposphere.events), 86
- EgressOnlyInternetGateway (class in troposphere.ec2), 67
- EIP (class in troposphere.ec2), 67
- EIPAssociation (class in troposphere.ec2), 67
- ElasticGpuSpecification (class in troposphere.ec2), 67
- ElasticInferenceAccelerator (class in troposphere.ec2), 67
- ElasticIp (class in troposphere.opsworks), 101
- ElasticLoadBalancerAttachment (class in troposphere.opsworks), 101
- ElasticsearchAction (class in troposphere.iot), 94
- ElasticsearchClusterConfig (class in troposphere.elasticsearch), 83
- ElasticsearchConfig (class in troposphere.appsync), 40
- ElasticsearchDestinationConfiguration (class in troposphere.firehose), 87
- ElasticsearchDomain (in module troposphere.elasticsearch), 83
- elb_name() (in module troposphere.validators), 124
- ElbInfoList (class in troposphere.codedeploy), 56
- EmailConfiguration (class in troposphere.cognito), 59
- EmailTemplate (class in troposphere.ses), 119
- encode_to_dict() (in module troposphere), 132
- encoding() (in module troposphere.validators), 124
- EncryptionAtRestOptions (class in troposphere.elasticsearch), 83
- EncryptionConfiguration (class in troposphere.firehose), 87
- EncryptionConfiguration (class in troposphere.s3), 108
- EncryptionKey (class in troposphere.codepipeline), 57
- Endpoint (class in troposphere.dms), 64
- Endpoint (class in troposphere.sagemaker), 111
- EndpointConfig (class in troposphere.sagemaker), 111
- EndpointConfiguration (class in troposphere.apigateway), 33
- EngineAttribute (class in troposphere.opsworks), 101
- Environment (class in troposphere.awslambda), 45
- Environment (class in troposphere.batch), 46
- Environment (class in troposphere.codebuild), 53
- Environment (class in troposphere.ecs), 76
- Environment (class in troposphere.elasticbeanstalk), 79
- Environment (class in troposphere.opsworks), 101
- EnvironmentEC2 (class in troposphere.cloud9), 48
- EnvironmentVariable (class in troposphere.codebuild), 53
- Equals (class in troposphere), 130
- EventDestination (class in troposphere.ses), 119
- EventSelector (class in troposphere.cloudtrail), 52
- EventSourceMapping (class in troposphere.awslambda), 45
- EventSubscription (class in troposphere.dms), 64
- EventSubscription (class in troposphere.rds), 104
- exactly_one() (in module troposphere.validators), 124
- ExecutionProperty (class in troposphere.glue), 89
- expected_output (tests.test_examples.TestExamples attribute), 139
- expected_output (tests.test_examples_template_generator.TestTemplateGenerator attribute), 139
- Export (class in troposphere), 130

ExtendedS3DestinationConfiguration (class in troposphere.firehose), 87

F

FakeAWSObject (class in tests.test_basic), 133

FakeAWSProperty (class in tests.test_basic), 133

FieldToMatch (class in troposphere.waf), 126

FieldToMatch (class in troposphere.wafregional), 127

filename (tests.test_examples.TestExamples attribute), 139

filename (tests.test_examples_template_generator.TestTemplateGenerator attribute), 139

FileSystem (class in troposphere.efs), 77

Filter (class in troposphere.guardduty), 91

Filter (class in troposphere.iotanalytics), 97

Filter (class in troposphere.s3), 108

Filter (class in troposphere.ses), 119

FindingCriteria (class in troposphere.guardduty), 92

FindInMap (class in troposphere), 130

FirehoseAction (class in troposphere.iot), 94

Firewall (class in troposphere.openstack.neutron), 26

FirewallPolicy (class in troposphere.openstack.neutron), 27

FirewallRule (class in troposphere.openstack.neutron), 27

FixedIP (class in troposphere.openstack.neutron), 27

FixedResponseConfig (class in troposphere.elasticloadbalancingv2), 81

Fleet (class in troposphere.appstream), 38

FleetLaunchTemplateConfigRequest (class in troposphere.ec2), 67

FleetLaunchTemplateOverridesRequest (class in troposphere.ec2), 67

FleetLaunchTemplateSpecificationRequest (class in troposphere.ec2), 68

FloatingIP (class in troposphere.openstack.neutron), 27

FloatingIP (class in troposphere.openstack.nova), 28

FloatingIPAssociation (class in troposphere.openstack.neutron), 27

FloatingIPAssociation (class in troposphere.openstack.nova), 29

FlowLog (class in troposphere.ec2), 68

ForwardedValues (class in troposphere.cloudfront), 51

from_dict() (troposphere.BaseAWSObject class method), 129

from_dict() (troposphere.Tags class method), 131

from_file() (in module troposphere.helpers.userdata), 26

Function (class in troposphere.awslambda), 45

Function (class in troposphere.serverless), 114

FunctionConfiguration (class in troposphere.appsync), 40

FunctionForPackaging (class in troposphere.serverless), 114

G

GatewayResponse (class in troposphere.apigateway), 33

generate_rules() (tests.test_emr.TestEMR method), 138

GenerateSecretString (class in troposphere.secretsmanager), 112

GenericHelperFn (class in troposphere), 130

GeoLocation (class in troposphere.route53), 106

GeoRestriction (class in troposphere.cloudfront), 51

get_att() (troposphere.AWSObject method), 129

get_aws_objects() (tests.test_int_type.TestIntTypeShouldNotBeUsed method), 139

get_events() (in module troposphere.utils), 124

get_or_add_parameter() (troposphere.Template method), 131

GetAtt (class in troposphere), 130

GetAtt() (troposphere.AWSObject method), 129

GetAZs (class in troposphere), 130

getdata() (troposphere.AWSHelperFn method), 129

GitHubLocation (class in troposphere.codedeploy), 56

GlobalSecondaryIndex (class in troposphere.dynamodb), 65

GraphQLApi (class in troposphere.appsync), 40

GraphQLSchema (class in troposphere.appsync), 40

GrokClassifier (class in troposphere.glue), 89

Group (class in troposphere.iam), 92

H

HadoopJarStepConfig (class in troposphere.emr), 84

handle_duplicate_key() (troposphere.Template method), 131

HealthCheck (class in troposphere.ecs), 76

HealthCheck (class in troposphere.elasticloadbalancing), 81

HealthCheck (class in troposphere.route53), 106

HealthCheckConfig (class in troposphere.servicediscovery), 118

HealthCheckConfiguration (class in troposphere.route53), 106

HealthCheckCustomConfig (class in troposphere.servicediscovery), 118

HealthMonitor (class in troposphere.openstack.neutron), 27

Hooks (class in troposphere.serverless), 114

Host (class in troposphere.ec2), 68

Host (class in troposphere.ecs), 76

HostedZone (class in troposphere.route53), 106

HostedZoneConfiguration (class in troposphere.route53), 106

HostedZoneVPCs (class in troposphere.route53), 106

HostEntry (class in troposphere.ecs), 76

HttpConfig (class in troposphere.appsync), 40

HttpNamespace (class in troposphere.servicediscovery), 118

I

iam_group_name() (in module troposphere.validators),

- 124
- iam_names() (in module troposphere.validators), 124
 - iam_path() (in module troposphere.validators), 124
 - iam_role_name() (in module troposphere.validators), 124
 - iam_user_name() (in module troposphere.validators), 124
 - IamInstanceProfile (class in troposphere.ec2), 68
 - ICMP (class in troposphere.ec2), 68
 - IdentityPool (class in troposphere.cognito), 59
 - IdentityPoolRoleAttachment (class in troposphere.cognito), 59
 - If (class in troposphere), 130
 - ignore() (in module troposphere.validators), 124
 - ImageBuilder (class in troposphere.appstream), 38
 - ImportValue (class in troposphere), 130
 - index_rotation_period_validator() (in module troposphere.firehose), 88
 - Init (class in troposphere.cloudformation), 49
 - InitConfig (class in troposphere.cloudformation), 49
 - InitConfigSets (class in troposphere.cloudformation), 49
 - InitFile (class in troposphere.cloudformation), 49
 - InitFileContext (class in troposphere.cloudformation), 49
 - InitFiles (class in troposphere.cloudformation), 49
 - InitService (class in troposphere.cloudformation), 49
 - InitServices (class in troposphere.cloudformation), 49
 - Input (class in troposphere.analytics), 30
 - InputArtifacts (class in troposphere.codepipeline), 58
 - InputLambdaProcessor (class in troposphere.analytics), 31
 - InputParallelism (class in troposphere.analytics), 31
 - InputProcessingConfiguration (class in troposphere.analytics), 31
 - InputSchema (class in troposphere.analytics), 31
 - InputTransformer (class in troposphere.events), 86
 - inspect_functions (troposphere.template_generator.TemplateGenerator attribute), 124
 - inspect_members (troposphere.template_generator.TemplateGenerator attribute), 124
 - inspect_resources (troposphere.template_generator.TemplateGenerator attribute), 124
 - Instance (class in troposphere.ec2), 68
 - Instance (class in troposphere.opsworks), 101
 - Instance (class in troposphere.servicediscovery), 118
 - instance_tenancy() (in module troposphere.ec2), 75
 - InstanceAssociationOutputLocation (class in troposphere.ssm), 121
 - InstanceFleetConfig (class in troposphere.emr), 84
 - InstanceFleetConfigProperty (class in troposphere.emr), 84
 - InstanceFleetProvisioningSpecifications (class in troposphere.emr), 84
 - InstanceGroupConfig (class in troposphere.emr), 84
 - InstanceGroupConfigProperty (class in troposphere.emr), 84
 - InstanceMarketOptions (class in troposphere.ec2), 68
 - InstanceProfile (class in troposphere.iam), 92
 - InstancesDistribution (class in troposphere.autoscaling), 42
 - InstanceTypeConfig (class in troposphere.emr), 84
 - integer() (in module troposphere.validators), 125
 - integer_list_item() (in module troposphere.validators), 125
 - integer_range() (in module troposphere.validators), 125
 - Integration (class in troposphere.apigateway), 33
 - Integration (class in troposphere.apigatewayv2), 36
 - IntegrationResponse (class in troposphere.apigateway), 33
 - IntegrationResponse (class in troposphere.apigatewayv2), 36
 - InternetGateway (class in troposphere.ec2), 68
 - InventoryConfiguration (class in troposphere.s3), 108
 - InviteMessageTemplate (class in troposphere.cognito), 59
 - IoTRuleEvent (class in troposphere.serverless), 114
 - IpAddressRequest (class in troposphere.route53), 106
 - IpFilter (class in troposphere.ses), 119
 - IPSet (class in troposphere.guardduty), 92
 - IPSet (class in troposphere.waf), 126
 - IPSet (class in troposphere.wafregional), 127
 - IPSetDescriptors (class in troposphere.waf), 126
 - IPSetDescriptors (class in troposphere.wafregional), 127
 - Ipv6Addresses (class in troposphere.ec2), 68
 - is_aws_object_subclass() (in module troposphere), 132
- ## J
- JdbcTarget (class in troposphere.glue), 89
 - Job (class in troposphere.glue), 90
 - JobCommand (class in troposphere.glue), 90
 - JobDefinition (class in troposphere.batch), 47
 - JobFlowInstancesConfig (class in troposphere.emr), 85
 - JobQueue (class in troposphere.batch), 47
 - Join (class in troposphere), 130
 - json_checker() (in module troposphere.validators), 125
 - JsonClassifier (class in troposphere.glue), 90
 - JSONMappingParameters (class in troposphere.analytics), 31
- ## K
- KerberosAttributes (class in troposphere.emr), 85
 - KernelCapabilities (class in troposphere.ecs), 76
 - Key (class in troposphere.dynamodb), 65
 - Key (class in troposphere.kms), 99
 - key_type_validator() (in module troposphere.dynamodb), 66
 - key_usage_type() (in module troposphere.validators), 125
 - KeyPair (class in troposphere.openstack.nova), 29
 - KeySchema (class in troposphere.dynamodb), 65

- KeyValue (class in troposphere.emr), 85
 - KinesisAction (class in troposphere.iot), 94
 - KinesisEvent (class in troposphere.serverless), 114
 - KinesisFirehoseDestination (class in troposphere.ses), 119
 - KinesisFirehoseInput (class in troposphere.analytics), 31
 - KinesisFirehoseOutput (class in troposphere.analytics), 31
 - KinesisParameters (class in troposphere.events), 86
 - KinesisStreamsInput (class in troposphere.analytics), 31
 - KinesisStreamSourceConfiguration (class in troposphere.firehose), 87
 - KinesisStreamsOutput (class in troposphere.analytics), 31
 - KMSEncryptionConfig (class in troposphere.firehose), 87
- L**
- Lambda (class in troposphere.iotanalytics), 97
 - LambdaAction (class in troposphere.iot), 94
 - LambdaAction (class in troposphere.ses), 119
 - LambdaConfig (class in troposphere.appsync), 40
 - LambdaConfig (class in troposphere.cognito), 59
 - LambdaConfigurations (class in troposphere.s3), 108
 - LambdaFunctionAssociation (class in troposphere.cloudfront), 51
 - LambdaOutput (class in troposphere.analytics), 31
 - LambdaRequestAuth (class in troposphere.serverless), 114
 - LambdaRequestAuthIdentity (class in troposphere.serverless), 114
 - LambdaTokenAuth (class in troposphere.serverless), 114
 - LambdaTokenAuthIdentity (class in troposphere.serverless), 115
 - launch_type_validator() (in module troposphere.ecs), 77
 - LaunchConfiguration (class in troposphere.autoscaling), 42
 - LaunchNotificationConstraint (class in troposphere.servicecatalog), 116
 - LaunchRoleConstraint (class in troposphere.servicecatalog), 116
 - LaunchSpecifications (class in troposphere.ec2), 68
 - LaunchTemplate (class in troposphere.autoscaling), 42
 - LaunchTemplate (class in troposphere.ec2), 68
 - LaunchTemplateConfigs (class in troposphere.ec2), 69
 - LaunchTemplateConstraint (class in troposphere.servicecatalog), 116
 - LaunchTemplateCreditSpecification (class in troposphere.ec2), 69
 - LaunchTemplateData (class in troposphere.ec2), 69
 - LaunchTemplateOverrides (class in troposphere.autoscaling), 42
 - LaunchTemplateOverrides (class in troposphere.ec2), 69
 - LaunchTemplateSpecification (class in troposphere.autoscaling), 42
 - LaunchTemplateSpecification (class in troposphere.batch), 47
 - LaunchTemplateSpecification (class in troposphere.ec2), 69
 - Layer (class in troposphere.opsworks), 101
 - LayerVersion (class in troposphere.awslambda), 45
 - LayerVersion (class in troposphere.serverless), 115
 - LayerVersionPermission (class in troposphere.awslambda), 45
 - LBCookieStickinessPolicy (class in troposphere.elasticloadbalancing), 81
 - LicenseSpecification (class in troposphere.ec2), 69
 - LifeCycleConfiguration (class in troposphere.opsworks), 101
 - LifecycleConfiguration (class in troposphere.s3), 109
 - LifecycleHook (class in troposphere.autoscaling), 42
 - LifecycleHookSpecification (class in troposphere.autoscaling), 42
 - LifecyclePolicy (class in troposphere.dlm), 63
 - LifecyclePolicy (class in troposphere.ecr), 75
 - LifecycleRule (class in troposphere.s3), 109
 - LifecycleRuleTransition (class in troposphere.s3), 109
 - LinuxParameters (class in troposphere.ecs), 76
 - Listener (class in troposphere.elasticloadbalancing), 81
 - Listener (class in troposphere.elasticloadbalancingv2), 81
 - ListenerCertificate (class in troposphere.elasticloadbalancingv2), 81
 - ListenerRule (class in troposphere.elasticloadbalancingv2), 82
 - load_tests() (in module tests.test_examples), 139
 - load_tests() (in module tests.test_examples_template_generator), 139
 - LoadBalancer (class in troposphere.ecs), 76
 - LoadBalancer (class in troposphere.elasticloadbalancing), 81
 - LoadBalancer (class in troposphere.elasticloadbalancingv2), 82
 - LoadBalancer (class in troposphere.openstack.neutron), 27
 - LoadBalancerAttributes (class in troposphere.elasticloadbalancingv2), 82
 - LoadBalancerInfo (class in troposphere.codedeploy), 56
 - LoadBalancersConfig (class in troposphere.ec2), 69
 - LoadBasedAutoScaling (class in troposphere.opsworks), 101
 - LocalSecondaryIndex (class in troposphere.dynamodb), 65
 - Location (class in troposphere.apigateway), 34
 - LogConfig (class in troposphere.appsync), 40
 - LogConfiguration (class in troposphere.ecs), 76
 - Logging (class in troposphere.cloudfront), 51
 - LoggingConfiguration (class in troposphere.s3), 109
 - LoggingInfo (class in troposphere.ssm), 121

- LoggingProperties (class in troposphere.redshift), 106
 LogGroup (class in troposphere.logs), 99
 LoginProfile (class in troposphere.iam), 92
 LogsConfig (class in troposphere.codebuild), 53
 LogsConfiguration (class in troposphere.amazonmq), 30
 LogStream (class in troposphere.logs), 99
- ## M
- Macro (class in troposphere.cloudformation), 50
 MaintenanceWindow (class in troposphere.amazonmq), 30
 MaintenanceWindow (class in troposphere.ssm), 121
 MaintenanceWindowAutomationParameters (class in troposphere.ssm), 121
 MaintenanceWindowLambdaParameters (class in troposphere.ssm), 122
 MaintenanceWindowRunCommandParameters (class in troposphere.ssm), 122
 MaintenanceWindowStepFunctionsParameters (class in troposphere.ssm), 122
 MaintenanceWindowTarget (class in troposphere.ssm), 122
 MaintenanceWindowTask (class in troposphere.ssm), 122
 ManagedPolicy (class in troposphere.iam), 93
 manyType (troposphere.autoscaling.Tags attribute), 43
 MappingParameters (class in troposphere.analytics), 31
 MappingRule (class in troposphere.cognito), 59
 market_validator() (in module troposphere.emr), 86
 Master (class in troposphere.guardduty), 92
 Matcher (class in troposphere.elasticloadbalancingv2), 82
 Math (class in troposphere.iotanalytics), 97
 MaxAgeRule (class in troposphere.elasticbeanstalk), 80
 MaxCountRule (class in troposphere.elasticbeanstalk), 80
 maxDiff (tests.test_examples.TestExamples attribute), 139
 maxDiff (tests.test_examples_template_generator.TestTemplateGenerator attribute), 139
 Member (class in troposphere.guardduty), 92
 Metadata (class in troposphere.autoscaling), 42
 Metadata (class in troposphere.cloudformation), 50
 Method (class in troposphere.apigateway), 34
 MethodResponse (class in troposphere.apigateway), 34
 MethodSetting (class in troposphere.apigateway), 34
 Metric (class in troposphere.cloudwatch), 53
 MetricDataQuery (class in troposphere.cloudwatch), 53
 MetricDimension (class in troposphere.applicationautoscaling), 37
 MetricDimension (class in troposphere.autoscaling), 42
 MetricDimension (class in troposphere.autoscalingplans), 44
 MetricDimension (class in troposphere.cloudwatch), 53
 MetricDimension (in module troposphere.emr), 85
 MetricFilter (class in troposphere.logs), 99
 MetricsCollection (class in troposphere.autoscaling), 42
 MetricsConfiguration (class in troposphere.s3), 109
 MetricStat (class in troposphere.cloudwatch), 53
 MetricTransformation (class in troposphere.logs), 99
 MicrosoftAD (class in troposphere.directoryservice), 63
 MinimumHealthyHosts (class in troposphere.codedeploy), 56
 MixedInstancesPolicy (class in troposphere.autoscaling), 42
 Model (class in troposphere.apigateway), 34
 Model (class in troposphere.apigatewayv2), 36
 Model (class in troposphere.sagemaker), 111
 MongoDBSettings (class in troposphere.dms), 64
 Monitoring (class in troposphere.ec2), 69
 MountPoint (class in troposphere.ec2), 69
 MountPoint (class in troposphere.ecs), 76
 MountPoints (class in troposphere.batch), 47
 MountTarget (class in troposphere.efs), 78
 mutually_exclusive() (in module troposphere.validators), 125
 MyCustomResource (class in tests.test_template_generator), 144
 MyMacroResource (class in tests.test_template_generator), 144
- ## N
- Name (class in troposphere), 130
 NamedQuery (class in troposphere.athena), 41
 NatGateway (class in troposphere.ec2), 69
 Net (class in troposphere.openstack.neutron), 27
 Network (class in troposphere.openstack.nova), 29
 network_port() (in module troposphere.validators), 125
 NetworkAcl (class in troposphere.ec2), 69
 NetworkAclEntry (class in troposphere.ec2), 69
 NetworkConfiguration (class in troposphere.ecs), 76
 NetworkInterface (class in troposphere.ec2), 69
 NetworkInterfaceAttachment (class in troposphere.ec2), 70
 NetworkInterfacePermission (class in troposphere.ec2), 70
 NetworkInterfaceProperty (class in troposphere.ec2), 70
 NetworkInterfaces (class in troposphere.ec2), 70
 no_validation() (troposphere.BaseAWSObject method), 129
 NodeGroupConfiguration (class in troposphere.elasticache), 78
 NoncurrentVersionTransition (class in troposphere.s3), 109
 Not (class in troposphere), 130
 NotebookInstance (class in troposphere.sagemaker), 111
 NotebookInstanceLifecycleConfig (class in troposphere.sagemaker), 111
 NotebookInstanceLifecycleHook (class in troposphere.sagemaker), 111
 Notification (class in troposphere.budgets), 48

- notification_event() (in module troposphere.validators), 125
 - notification_type() (in module troposphere.validators), 125
 - NotificationConfig (class in troposphere.ssm), 122
 - NotificationConfiguration (class in troposphere.s3), 109
 - NotificationConfigurations (class in troposphere.autoscaling), 42
 - NotificationWithSubscribers (class in troposphere.budgets), 48
 - NUMBER_PROPERTIES (troposphere.Parameter attribute), 130
 - NumberAttributeConstraints (class in troposphere.cognito), 59
- O**
- ObjectField (class in troposphere.datapipeline), 62
 - OnDemandOptionsRequest (class in troposphere.ec2), 70
 - one_of() (in module troposphere.validators), 125
 - OnPremisesInstanceTagFilters (class in troposphere.codedeploy), 56
 - OnPremisesTagSet (class in troposphere.codedeploy), 56
 - OnPremisesTagSetList (class in troposphere.codedeploy), 56
 - OnPremisesTagSetObject (class in troposphere.codedeploy), 56
 - OpenIDConnectConfig (class in troposphere.appsync), 40
 - operating_system() (in module troposphere.validators), 125
 - OptionConfiguration (class in troposphere.rds), 104
 - OptionGroup (class in troposphere.rds), 104
 - OptionSetting (class in troposphere.rds), 104
 - OptionSettings (class in troposphere.elasticbeanstalk), 80
 - Or (class in troposphere), 130
 - Order (class in troposphere.glue), 90
 - OrganizationAggregationSource (class in troposphere.config), 61
 - Origin (class in troposphere.cloudfront), 51
 - OriginCustomHeader (class in troposphere.cloudfront), 51
 - Output (class in troposphere), 130
 - Output (class in troposphere.analytics), 31
 - OutputArtifacts (class in troposphere.codepipeline), 58
 - OutputFileUriValue (class in troposphere.iotanalytics), 97
- P**
- Parameter (class in troposphere), 130
 - Parameter (class in troposphere.ssm), 122
 - ParameterGroup (class in troposphere.dax), 62
 - ParameterGroup (class in troposphere.elasticache), 78
 - ParameterObject (class in troposphere.datapipeline), 62
 - ParameterObjectAttribute (class in troposphere.datapipeline), 62
 - ParameterValue (class in troposphere.datapipeline), 62
 - Partition (class in troposphere.glue), 90
 - PartitionInput (class in troposphere.glue), 90
 - PasswordPolicy (class in troposphere.cognito), 59
 - PatchBaseline (class in troposphere.ssm), 122
 - PatchFilter (class in troposphere.ssm), 122
 - PatchFilterGroup (class in troposphere.ssm), 122
 - Permission (class in troposphere.awslambda), 46
 - PhysicalConnectionRequirements (class in troposphere.glue), 90
 - Pipeline (class in troposphere.codepipeline), 58
 - Pipeline (class in troposphere.datapipeline), 62
 - Pipeline (class in troposphere.iotanalytics), 97
 - PipelineConfig (class in troposphere.appsync), 40
 - PipelineObject (class in troposphere.datapipeline), 62
 - PipelineTag (class in troposphere.datapipeline), 62
 - Placement (class in troposphere.ec2), 70
 - Placement (class in troposphere.iot1click), 96
 - placement_constraint_validator() (in module troposphere.ecs), 77
 - placement_strategy_validator() (in module troposphere.ecs), 77
 - PlacementConstraint (class in troposphere.ecs), 76
 - PlacementGroup (class in troposphere.ec2), 70
 - PlacementStrategy (class in troposphere.ecs), 77
 - PlacementTemplate (class in troposphere.iot1click), 96
 - PlacementType (class in troposphere.emr), 85
 - PointInTimeRecoverySpecification (class in troposphere.dynamodb), 65
 - Policies (class in troposphere.cognito), 59
 - Policy (class in troposphere.elasticloadbalancing), 81
 - Policy (class in troposphere.iam), 93
 - Policy (class in troposphere.iot), 94
 - PolicyDetails (class in troposphere.dlm), 63
 - PolicyPrincipalAttachment (class in troposphere.iot), 95
 - PolicyProperty (in module troposphere.iam), 93
 - PolicyType (class in troposphere.iam), 93
 - Pool (class in troposphere.openstack.neutron), 27
 - PoolMember (class in troposphere.openstack.neutron), 28
 - Port (class in troposphere.openstack.neutron), 28
 - Portfolio (class in troposphere.servicecatalog), 116
 - PortfolioPrincipalAssociation (class in troposphere.servicecatalog), 117
 - PortfolioProductAssociation (class in troposphere.servicecatalog), 117
 - PortfolioShare (class in troposphere.servicecatalog), 117
 - PortMapping (class in troposphere.ecs), 77
 - PortRange (class in troposphere.ec2), 70
 - positive_integer() (in module troposphere.validators), 125
 - PredefinedLoadMetricSpecification (class in troposphere.autoscalingplans), 44
 - PredefinedMetricSpecification (class in troposphere.applicationautoscaling), 37
 - PredefinedMetricSpecification (class in troposphere.autoscaling), 43

- PredefinedScalingMetricSpecification (class in troposphere.autoscalingplans), 44
- Predicate (class in troposphere.glue), 90
- Predicates (class in troposphere.waf), 126
- Predicates (class in troposphere.wafregional), 127
- priceclass_type() (in module troposphere.validators), 125
- primary_key_type_validator() (in module troposphere.serverless), 115
- PrimaryKey (class in troposphere.serverless), 115
- PrivateDnsNamespace (class in troposphere.servicediscovery), 118
- PrivateIpAddressSpecification (class in troposphere.ec2), 70
- ProcessingConfiguration (class in troposphere.firehose), 87
- Processor (class in troposphere.firehose), 88
- processor_type_validator() (in module troposphere.firehose), 88
- ProcessorFeature (class in troposphere.rds), 104
- ProcessorParameter (class in troposphere.firehose), 88
- ProductionVariant (class in troposphere.sagemaker), 111
- Project (class in troposphere.codebuild), 53
- Project (class in troposphere.iot1click), 96
- ProjectCache (class in troposphere.codebuild), 54
- Projection (class in troposphere.dynamodb), 65
- projection_type_validator() (in module troposphere.dynamodb), 66
- ProjectTriggers (class in troposphere.codebuild), 54
- properties_validator() (in module troposphere.emr), 86
- props (tests.test_basic.FakeAWSObject attribute), 133
- props (tests.test_basic.FakeAWSProperty attribute), 133
- props (tests.test_template_generator.MyCustomResource attribute), 144
- props (tests.test_template_generator.MyMacroResource attribute), 144
- props (troposphere.amazonmq.Broker attribute), 29
- props (troposphere.amazonmq.Configuration attribute), 29
- props (troposphere.amazonmq.ConfigurationAssociation attribute), 30
- props (troposphere.amazonmq.ConfigurationId attribute), 30
- props (troposphere.amazonmq.LogsConfiguration attribute), 30
- props (troposphere.amazonmq.MaintenanceWindow attribute), 30
- props (troposphere.amazonmq.User attribute), 30
- props (troposphere.analytics.Application attribute), 30
- props (troposphere.analytics.ApplicationOutput attribute), 30
- props (troposphere.analytics.ApplicationReferenceDataSource attribute), 30
- props (troposphere.analytics.CSVMappingParameters attribute), 30
- props (troposphere.analytics.DestinationSchema attribute), 30
- props (troposphere.analytics.Input attribute), 31
- props (troposphere.analytics.InputLambdaProcessor attribute), 31
- props (troposphere.analytics.InputParallelism attribute), 31
- props (troposphere.analytics.InputProcessingConfiguration attribute), 31
- props (troposphere.analytics.InputSchema attribute), 31
- props (troposphere.analytics.JSONMappingParameters attribute), 31
- props (troposphere.analytics.KinesisFirehoseInput attribute), 31
- props (troposphere.analytics.KinesisFirehoseOutput attribute), 31
- props (troposphere.analytics.KinesisStreamsInput attribute), 31
- props (troposphere.analytics.KinesisStreamsOutput attribute), 31
- props (troposphere.analytics.LambdaOutput attribute), 31
- props (troposphere.analytics.MappingParameters attribute), 31
- props (troposphere.analytics.Output attribute), 31
- props (troposphere.analytics.RecordColumn attribute), 31
- props (troposphere.analytics.RecordFormat attribute), 32
- props (troposphere.analytics.ReferenceDataSource attribute), 32
- props (troposphere.analytics.ReferenceSchema attribute), 32
- props (troposphere.analytics.S3ReferenceDataSource attribute), 32
- props (troposphere.apigateway.AccessLogSetting attribute), 32
- props (troposphere.apigateway.Account attribute), 32
- props (troposphere.apigateway.ApiKey attribute), 32
- props (troposphere.apigateway.ApiStage attribute), 32
- props (troposphere.apigateway.Authorizer attribute), 32
- props (troposphere.apigateway.BasePathMapping attribute), 32
- props (troposphere.apigateway.CanarySetting attribute), 32
- props (troposphere.apigateway.ClientCertificate attribute), 33
- props (troposphere.apigateway.Deployment attribute), 33
- props (troposphere.apigateway.DeploymentCanarySettings attribute), 33
- props (troposphere.apigateway.DocumentationPart attribute), 33
- props (troposphere.apigateway.DocumentationVersion attribute), 33
- props (troposphere.apigateway.DomainName attribute), 33
- props (troposphere.apigateway.EndpointConfiguration attribute), 33

- tribute), 33
- props (troposphere.apigateway.GatewayResponse attribute), 33
- props (troposphere.apigateway.Integration attribute), 33
- props (troposphere.apigateway.IntegrationResponse attribute), 33
- props (troposphere.apigateway.Location attribute), 34
- props (troposphere.apigateway.Method attribute), 34
- props (troposphere.apigateway.MethodResponse attribute), 34
- props (troposphere.apigateway.MethodSetting attribute), 34
- props (troposphere.apigateway.Model attribute), 34
- props (troposphere.apigateway.QuotaSettings attribute), 34
- props (troposphere.apigateway.RequestValidator attribute), 34
- props (troposphere.apigateway.Resource attribute), 34
- props (troposphere.apigateway.RestApi attribute), 34
- props (troposphere.apigateway.S3Location attribute), 34
- props (troposphere.apigateway.Stage attribute), 34
- props (troposphere.apigateway.StageDescription attribute), 35
- props (troposphere.apigateway.StageKey attribute), 35
- props (troposphere.apigateway.ThrottleSettings attribute), 35
- props (troposphere.apigateway.UsagePlan attribute), 35
- props (troposphere.apigateway.UsagePlanKey attribute), 35
- props (troposphere.apigateway.VpcLink attribute), 35
- props (troposphere.apigatewayv2.AccessLogSettings attribute), 35
- props (troposphere.apigatewayv2.Api attribute), 35
- props (troposphere.apigatewayv2.Authorizer attribute), 36
- props (troposphere.apigatewayv2.Deployment attribute), 36
- props (troposphere.apigatewayv2.Integration attribute), 36
- props (troposphere.apigatewayv2.IntegrationResponse attribute), 36
- props (troposphere.apigatewayv2.Model attribute), 36
- props (troposphere.apigatewayv2.Route attribute), 36
- props (troposphere.apigatewayv2.RouteResponse attribute), 36
- props (troposphere.apigatewayv2.RouteSettings attribute), 36
- props (troposphere.apigatewayv2.Stage attribute), 36
- props (troposphere.applicationautoscaling.CustomizedMetricSpecification attribute), 37
- props (troposphere.applicationautoscaling.MetricDimension attribute), 37
- props (troposphere.applicationautoscaling.PredefinedMetricSpecification attribute), 37
- props (troposphere.applicationautoscaling.ScalableTarget attribute), 37
- props (troposphere.applicationautoscaling.ScalableTargetAction attribute), 37
- props (troposphere.applicationautoscaling.ScalingPolicy attribute), 37
- props (troposphere.applicationautoscaling.ScheduledAction attribute), 37
- props (troposphere.applicationautoscaling.StepAdjustment attribute), 37
- props (troposphere.applicationautoscaling.StepScalingPolicyConfiguration attribute), 38
- props (troposphere.applicationautoscaling.TargetTrackingScalingPolicyConfiguration attribute), 38
- props (troposphere.appstream.ApplicationSettings attribute), 38
- props (troposphere.appstream.ComputeCapacity attribute), 38
- props (troposphere.appstream.DirectoryConfig attribute), 38
- props (troposphere.appstream.DomainJoinInfo attribute), 38
- props (troposphere.appstream.Fleet attribute), 38
- props (troposphere.appstream.ImageBuilder attribute), 38
- props (troposphere.appstream.ServiceAccountCredentials attribute), 38
- props (troposphere.appstream.Stack attribute), 38
- props (troposphere.appstream.StackFleetAssociation attribute), 39
- props (troposphere.appstream.StackUserAssociation attribute), 39
- props (troposphere.appstream.StorageConnector attribute), 39
- props (troposphere.appstream.User attribute), 39
- props (troposphere.appstream.UserSetting attribute), 39
- props (troposphere.appstream.VpcConfig attribute), 39
- props (troposphere.appsync.ApiKey attribute), 39
- props (troposphere.appsync.AuthorizationConfig attribute), 39
- props (troposphere.appsync.AwsIamConfig attribute), 39
- props (troposphere.appsync.DataSource attribute), 39
- props (troposphere.appsync.DynamoDBConfig attribute), 39
- props (troposphere.appsync.ElasticsearchConfig attribute), 40
- props (troposphere.appsync.FunctionConfiguration attribute), 40
- props (troposphere.appsync.GraphQLApi attribute), 40
- props (troposphere.appsync.GraphQLSchema attribute), 40
- props (troposphere.appsync.HttpConfig attribute), 40
- props (troposphere.appsync.LambdaConfig attribute), 40
- props (troposphere.appsync.LogConfig attribute), 40
- props (troposphere.appsync.OpenIDConnectConfig attribute), 40

- tribute), 40
- props (troposphere.appsync.PipelineConfig attribute), 40
- props (troposphere.appsync.RdsHttpEndpointConfig attribute), 40
- props (troposphere.appsync.RelationalDatabaseConfig attribute), 40
- props (troposphere.appsync.Resolver attribute), 40
- props (troposphere.appsync.UserPoolConfig attribute), 41
- props (troposphere.ask.AuthenticationConfiguration attribute), 41
- props (troposphere.ask.Skill attribute), 41
- props (troposphere.ask.SkillPackage attribute), 41
- props (troposphere.athena.NamedQuery attribute), 41
- props (troposphere.autoscaling.AutoScalingGroup attribute), 41
- props (troposphere.autoscaling.BlockDeviceMapping attribute), 41
- props (troposphere.autoscaling.CustomizedMetricSpecification attribute), 41
- props (troposphere.autoscaling.EBSBlockDevice attribute), 41
- props (troposphere.autoscaling.InstancesDistribution attribute), 42
- props (troposphere.autoscaling.LaunchConfiguration attribute), 42
- props (troposphere.autoscaling.LaunchTemplate attribute), 42
- props (troposphere.autoscaling.LaunchTemplateOverrides attribute), 42
- props (troposphere.autoscaling.LaunchTemplateSpecification attribute), 42
- props (troposphere.autoscaling.LifecycleHook attribute), 42
- props (troposphere.autoscaling.LifecycleHookSpecification attribute), 42
- props (troposphere.autoscaling.MetricDimension attribute), 42
- props (troposphere.autoscaling.MetricsCollection attribute), 42
- props (troposphere.autoscaling.MixedInstancesPolicy attribute), 42
- props (troposphere.autoscaling.NotificationConfigurations attribute), 42
- props (troposphere.autoscaling.PredefinedMetricSpecification attribute), 43
- props (troposphere.autoscaling.ScalingPolicy attribute), 43
- props (troposphere.autoscaling.ScheduledAction attribute), 43
- props (troposphere.autoscaling.StepAdjustments attribute), 43
- props (troposphere.autoscaling.TargetTrackingConfiguration attribute), 43
- props (troposphere.autoscaling.Trigger attribute), 43
- props (troposphere.autoscalingplans.ApplicationSource attribute), 43
- props (troposphere.autoscalingplans.CustomizedLoadMetricSpecification attribute), 44
- props (troposphere.autoscalingplans.CustomizedScalingMetricSpecification attribute), 44
- props (troposphere.autoscalingplans.MetricDimension attribute), 44
- props (troposphere.autoscalingplans.PredefinedLoadMetricSpecification attribute), 44
- props (troposphere.autoscalingplans.PredefinedScalingMetricSpecification attribute), 44
- props (troposphere.autoscalingplans.ScalingInstruction attribute), 44
- props (troposphere.autoscalingplans.ScalingPlan attribute), 44
- props (troposphere.autoscalingplans.TagFilter attribute), 44
- props (troposphere.autoscalingplans.TargetTrackingConfiguration attribute), 44
- props (troposphere.awslambda.Alias attribute), 45
- props (troposphere.awslambda.AliasRoutingConfiguration attribute), 45
- props (troposphere.awslambda.Code attribute), 45
- props (troposphere.awslambda.Content attribute), 45
- props (troposphere.awslambda.DeadLetterConfig attribute), 45
- props (troposphere.awslambda.Environment attribute), 45
- props (troposphere.awslambda.EventSourceMapping attribute), 45
- props (troposphere.awslambda.Function attribute), 45
- props (troposphere.awslambda.LayerVersion attribute), 45
- props (troposphere.awslambda.LayerVersionPermission attribute), 45
- props (troposphere.awslambda.Permission attribute), 46
- props (troposphere.awslambda.TracingConfig attribute), 46
- props (troposphere.awslambda.Version attribute), 46
- props (troposphere.awslambda.VersionWeight attribute), 46
- props (troposphere.awslambda.VPCCConfig attribute), 46
- props (troposphere.batch.ComputeEnvironment attribute), 46
- props (troposphere.batch.ComputeEnvironmentOrder attribute), 46
- props (troposphere.batch.ComputeResources attribute), 46
- props (troposphere.batch.ContainerProperties attribute), 46
- props (troposphere.batch.Environment attribute), 46
- props (troposphere.batch.JobDefinition attribute), 47
- props (troposphere.batch.JobQueue attribute), 47
- props (troposphere.batch.LaunchTemplateSpecification

- attribute), 47
- props (troposphere.batch.MountPoints attribute), 47
- props (troposphere.batch.RetryStrategy attribute), 47
- props (troposphere.batch.Timeout attribute), 47
- props (troposphere.batch.Ulimit attribute), 47
- props (troposphere.batch.Volumes attribute), 47
- props (troposphere.batch.VolumesHost attribute), 47
- props (troposphere.budgets.Budget attribute), 47
- props (troposphere.budgets.BudgetData attribute), 48
- props (troposphere.budgets.CostTypes attribute), 48
- props (troposphere.budgets.Notification attribute), 48
- props (troposphere.budgets.NotificationWithSubscribers attribute), 48
- props (troposphere.budgets.Spend attribute), 48
- props (troposphere.budgets.Subscriber attribute), 48
- props (troposphere.budgets.TimePeriod attribute), 48
- props (troposphere.certificatemanager.Certificate attribute), 48
- props (troposphere.certificatemanager.DomainValidationOptions attribute), 48
- props (troposphere.cloud9.EnvironmentEC2 attribute), 48
- props (troposphere.cloud9.Repository attribute), 49
- props (troposphere.cloudformation.AuthenticationBlock attribute), 49
- props (troposphere.cloudformation.CustomResource attribute), 49
- props (troposphere.cloudformation.InitConfig attribute), 49
- props (troposphere.cloudformation.InitFile attribute), 49
- props (troposphere.cloudformation.InitService attribute), 49
- props (troposphere.cloudformation.Macro attribute), 50
- props (troposphere.cloudformation.Stack attribute), 50
- props (troposphere.cloudformation.WaitCondition attribute), 50
- props (troposphere.cloudformation.WaitConditionHandle attribute), 50
- props (troposphere.cloudfront.CacheBehavior attribute), 50
- props (troposphere.cloudfront.CloudFrontOriginAccessIdentity attribute), 50
- props (troposphere.cloudfront.CloudFrontOriginAccessIdentityConfig attribute), 50
- props (troposphere.cloudfront.Cookies attribute), 51
- props (troposphere.cloudfront.CustomErrorResponse attribute), 51
- props (troposphere.cloudfront.CustomOriginConfig attribute), 51
- props (troposphere.cloudfront.DefaultCacheBehavior attribute), 51
- props (troposphere.cloudfront.Distribution attribute), 51
- props (troposphere.cloudfront.DistributionConfig attribute), 51
- props (troposphere.cloudfront.ForwardedValues attribute), 51
- props (troposphere.cloudfront.GeoRestriction attribute), 51
- props (troposphere.cloudfront.LambdaFunctionAssociation attribute), 51
- props (troposphere.cloudfront.Logging attribute), 51
- props (troposphere.cloudfront.Origin attribute), 51
- props (troposphere.cloudfront.OriginCustomHeader attribute), 51
- props (troposphere.cloudfront.Restrictions attribute), 51
- props (troposphere.cloudfront.S3Origin attribute), 52
- props (troposphere.cloudfront.S3OriginConfig attribute), 52
- props (troposphere.cloudfront.StreamingDistribution attribute), 52
- props (troposphere.cloudfront.StreamingDistributionConfig attribute), 52
- props (troposphere.cloudfront.TrustedSigners attribute), 52
- props (troposphere.cloudfront.ViewerCertificate attribute), 52
- props (troposphere.cloudtrail.DataResource attribute), 52
- props (troposphere.cloudtrail.EventSelector attribute), 52
- props (troposphere.cloudtrail.Trail attribute), 52
- props (troposphere.cloudwatch.Alarm attribute), 52
- props (troposphere.cloudwatch.Dashboard attribute), 53
- props (troposphere.cloudwatch.Metric attribute), 53
- props (troposphere.cloudwatch.MetricDataQuery attribute), 53
- props (troposphere.cloudwatch.MetricDimension attribute), 53
- props (troposphere.cloudwatch.MetricStat attribute), 53
- props (troposphere.codebuild.Artifacts attribute), 53
- props (troposphere.codebuild.CloudWatchLogs attribute), 53
- props (troposphere.codebuild.Environment attribute), 53
- props (troposphere.codebuild.EnvironmentVariable attribute), 53
- props (troposphere.codebuild.LogsConfig attribute), 53
- props (troposphere.codebuild.Project attribute), 54
- props (troposphere.codebuild.ProjectCache attribute), 54
- props (troposphere.codebuild.ProjectTriggers attribute), 54
- props (troposphere.codebuild.RegistryCredential attribute), 54
- props (troposphere.codebuild.S3Logs attribute), 54
- props (troposphere.codebuild.Source attribute), 54
- props (troposphere.codebuild.SourceAuth attribute), 54
- props (troposphere.codebuild.VpcConfig attribute), 54
- props (troposphere.codecommit.Repository attribute), 54
- props (troposphere.codecommit.Trigger attribute), 55
- props (troposphere.codedeploy.Alarm attribute), 55

- props (troposphere.codedeploy.AlarmConfiguration attribute), 55
- props (troposphere.codedeploy.Application attribute), 55
- props (troposphere.codedeploy.AutoRollbackConfiguration attribute), 55
- props (troposphere.codedeploy.Deployment attribute), 55
- props (troposphere.codedeploy.DeploymentConfig attribute), 55
- props (troposphere.codedeploy.DeploymentGroup attribute), 55
- props (troposphere.codedeploy.DeploymentStyle attribute), 55
- props (troposphere.codedeploy.Ec2TagFilters attribute), 55
- props (troposphere.codedeploy.Ec2TagSet attribute), 56
- props (troposphere.codedeploy.Ec2TagSetListObject attribute), 56
- props (troposphere.codedeploy.ElbInfoList attribute), 56
- props (troposphere.codedeploy.GitHubLocation attribute), 56
- props (troposphere.codedeploy.LoadBalancerInfo attribute), 56
- props (troposphere.codedeploy.MinimumHealthyHosts attribute), 56
- props (troposphere.codedeploy.OnPremisesInstanceTagFilters attribute), 56
- props (troposphere.codedeploy.OnPremisesTagSet attribute), 56
- props (troposphere.codedeploy.OnPremisesTagSetList attribute), 56
- props (troposphere.codedeploy.OnPremisesTagSetObject attribute), 56
- props (troposphere.codedeploy.Revision attribute), 56
- props (troposphere.codedeploy.S3Location attribute), 56
- props (troposphere.codedeploy.TagFilters attribute), 56
- props (troposphere.codedeploy.TargetGroupInfoList attribute), 57
- props (troposphere.codedeploy.TriggerConfig attribute), 57
- props (troposphere.codepipeline.Actions attribute), 57
- props (troposphere.codepipeline.ActionTypeId attribute), 57
- props (troposphere.codepipeline.ArtifactDetails attribute), 57
- props (troposphere.codepipeline.ArtifactStore attribute), 57
- props (troposphere.codepipeline.ArtifactStoreMap attribute), 57
- props (troposphere.codepipeline.Blockers attribute), 57
- props (troposphere.codepipeline.ConfigurationProperties attribute), 57
- props (troposphere.codepipeline.CustomActionType attribute), 57
- props (troposphere.codepipeline.DisableInboundStageTransitions attribute), 57
- props (troposphere.codepipeline.EncryptionKey attribute), 58
- props (troposphere.codepipeline.InputArtifacts attribute), 58
- props (troposphere.codepipeline.OutputArtifacts attribute), 58
- props (troposphere.codepipeline.Pipeline attribute), 58
- props (troposphere.codepipeline.Settings attribute), 58
- props (troposphere.codepipeline.Stages attribute), 58
- props (troposphere.codepipeline.Webhook attribute), 58
- props (troposphere.codepipeline.WebhookAuthConfiguration attribute), 58
- props (troposphere.codepipeline.WebhookFilterRule attribute), 58
- props (troposphere.cognito.AdminCreateUserConfig attribute), 58
- props (troposphere.cognito.AttributeType attribute), 58
- props (troposphere.cognito.CognitoIdentityProvider attribute), 58
- props (troposphere.cognito.CognitoStreams attribute), 59
- props (troposphere.cognito.DeviceConfiguration attribute), 59
- props (troposphere.cognito.EmailConfiguration attribute), 59
- props (troposphere.cognito.IdentityPool attribute), 59
- props (troposphere.cognito.IdentityPoolRoleAttachment attribute), 59
- props (troposphere.cognito.InviteMessageTemplate attribute), 59
- props (troposphere.cognito.LambdaConfig attribute), 59
- props (troposphere.cognito.MappingRule attribute), 59
- props (troposphere.cognito.NumberAttributeConstraints attribute), 59
- props (troposphere.cognito.PasswordPolicy attribute), 59
- props (troposphere.cognito.Policies attribute), 59
- props (troposphere.cognito.PushSync attribute), 59
- props (troposphere.cognito.RoleMapping attribute), 59
- props (troposphere.cognito.RulesConfiguration attribute), 60
- props (troposphere.cognito.SchemaAttribute attribute), 60
- props (troposphere.cognito.SmsConfiguration attribute), 60
- props (troposphere.cognito.StringAttributeConstraints attribute), 60
- props (troposphere.cognito.UserPool attribute), 60
- props (troposphere.cognito.UserPoolClient attribute), 60
- props (troposphere.cognito.UserPoolGroup attribute), 60
- props (troposphere.cognito.UserPoolUser attribute), 60
- props (troposphere.cognito.UserPoolUserToGroupAttachment attribute), 60
- props (troposphere.config.AccountAggregationSources attribute), 60

- props (troposphere.config.AggregationAuthorization attribute), 61
- props (troposphere.config.ConfigRule attribute), 61
- props (troposphere.config.ConfigSnapshotDeliveryProperties attribute), 61
- props (troposphere.config.ConfigurationAggregator attribute), 61
- props (troposphere.config.ConfigurationRecorder attribute), 61
- props (troposphere.config.DeliveryChannel attribute), 61
- props (troposphere.config.OrganizationAggregationSource attribute), 61
- props (troposphere.config.RecordingGroup attribute), 61
- props (troposphere.config.Scope attribute), 61
- props (troposphere.config.Source attribute), 61
- props (troposphere.config.SourceDetails attribute), 61
- props (troposphere.datapipeline.ObjectField attribute), 62
- props (troposphere.datapipeline.ParameterObject attribute), 62
- props (troposphere.datapipeline.ParameterObjectAttribute attribute), 62
- props (troposphere.datapipeline.ParameterValue attribute), 62
- props (troposphere.datapipeline.Pipeline attribute), 62
- props (troposphere.datapipeline.PipelineObject attribute), 62
- props (troposphere.datapipeline.PipelineTag attribute), 62
- props (troposphere.dax.Cluster attribute), 62
- props (troposphere.dax.ParameterGroup attribute), 62
- props (troposphere.dax.SSESpecification attribute), 62
- props (troposphere.dax.SubnetGroup attribute), 63
- props (troposphere.directoryservice.MicrosoftAD attribute), 63
- props (troposphere.directoryservice.SimpleAD attribute), 63
- props (troposphere.directoryservice.VpcSettings attribute), 63
- props (troposphere.dlm.CreateRule attribute), 63
- props (troposphere.dlm.LifecyclePolicy attribute), 63
- props (troposphere.dlm.PolicyDetails attribute), 63
- props (troposphere.dlm.RetainRule attribute), 63
- props (troposphere.dlm.Schedule attribute), 63
- props (troposphere.dms.Certificate attribute), 64
- props (troposphere.dms.DynamoDBSettings attribute), 64
- props (troposphere.dms.Endpoint attribute), 64
- props (troposphere.dms.EventSubscription attribute), 64
- props (troposphere.dms.MongoDbSettings attribute), 64
- props (troposphere.dms.ReplicationInstance attribute), 64
- props (troposphere.dms.ReplicationSubnetGroup attribute), 64
- props (troposphere.dms.ReplicationTask attribute), 64
- props (troposphere.dms.S3Settings attribute), 64
- props (troposphere.docdb.DBCluster attribute), 65
- props (troposphere.docdb.DBClusterParameterGroup attribute), 65
- props (troposphere.docdb.DBInstance attribute), 65
- props (troposphere.docdb.DBSubnetGroup attribute), 65
- props (troposphere.dynamodb.AttributeDefinition attribute), 65
- props (troposphere.dynamodb.GlobalSecondaryIndex attribute), 65
- props (troposphere.dynamodb.KeySchema attribute), 65
- props (troposphere.dynamodb.LocalSecondaryIndex attribute), 65
- props (troposphere.dynamodb.PointInTimeRecoverySpecification attribute), 65
- props (troposphere.dynamodb.Projection attribute), 66
- props (troposphere.dynamodb.ProvisionedThroughput attribute), 66
- props (troposphere.dynamodb.SSESpecification attribute), 66
- props (troposphere.dynamodb.StreamSpecification attribute), 66
- props (troposphere.dynamodb.Table attribute), 66
- props (troposphere.dynamodb.TimeToLiveSpecification attribute), 66
- props (troposphere.ec2.AssociationParameters attribute), 66
- props (troposphere.ec2.BlockDeviceMapping attribute), 66
- props (troposphere.ec2.ClassicLoadBalancer attribute), 66
- props (troposphere.ec2.ClassicLoadBalancersConfig attribute), 66
- props (troposphere.ec2.CreditSpecification attribute), 67
- props (troposphere.ec2.CustomerGateway attribute), 67
- props (troposphere.ec2.DHCPOptions attribute), 67
- props (troposphere.ec2.EBSBlockDevice attribute), 67
- props (troposphere.ec2.EC2Fleet attribute), 67
- props (troposphere.ec2.EgressOnlyInternetGateway attribute), 67
- props (troposphere.ec2.EIP attribute), 67
- props (troposphere.ec2.EIPAssociation attribute), 67
- props (troposphere.ec2.ElasticGpuSpecification attribute), 67
- props (troposphere.ec2.ElasticInferenceAccelerator attribute), 67
- props (troposphere.ec2.FleetLaunchTemplateConfigRequest attribute), 67
- props (troposphere.ec2.FleetLaunchTemplateOverridesRequest attribute), 68
- props (troposphere.ec2.FleetLaunchTemplateSpecificationRequest attribute), 68
- props (troposphere.ec2.FlowLog attribute), 68
- props (troposphere.ec2.Host attribute), 68
- props (troposphere.ec2.IamInstanceProfile attribute), 68
- props (troposphere.ec2.ICMP attribute), 68

- props (troposphere.ec2.Instance attribute), 68
- props (troposphere.ec2.InstanceMarketOptions attribute), 68
- props (troposphere.ec2.InternetGateway attribute), 68
- props (troposphere.ec2.LaunchSpecifications attribute), 68
- props (troposphere.ec2.LaunchTemplate attribute), 68
- props (troposphere.ec2.LaunchTemplateConfigs attribute), 69
- props (troposphere.ec2.LaunchTemplateCreditSpecification attribute), 69
- props (troposphere.ec2.LaunchTemplateData attribute), 69
- props (troposphere.ec2.LaunchTemplateOverrides attribute), 69
- props (troposphere.ec2.LaunchTemplateSpecification attribute), 69
- props (troposphere.ec2.LicenseSpecification attribute), 69
- props (troposphere.ec2.LoadBalancersConfig attribute), 69
- props (troposphere.ec2.Monitoring attribute), 69
- props (troposphere.ec2.MountPoint attribute), 69
- props (troposphere.ec2.NatGateway attribute), 69
- props (troposphere.ec2.NetworkAcl attribute), 69
- props (troposphere.ec2.NetworkAclEntry attribute), 69
- props (troposphere.ec2.NetworkInterface attribute), 70
- props (troposphere.ec2.NetworkInterfaceAttachment attribute), 70
- props (troposphere.ec2.NetworkInterfacePermission attribute), 70
- props (troposphere.ec2.NetworkInterfaceProperty attribute), 70
- props (troposphere.ec2.NetworkInterfaces attribute), 70
- props (troposphere.ec2.OnDemandOptionsRequest attribute), 70
- props (troposphere.ec2.Placement attribute), 70
- props (troposphere.ec2.PlacementGroup attribute), 70
- props (troposphere.ec2.PortRange attribute), 70
- props (troposphere.ec2.PrivateIpAddressSpecification attribute), 70
- props (troposphere.ec2.Route attribute), 70
- props (troposphere.ec2.RouteTable attribute), 71
- props (troposphere.ec2.SecurityGroup attribute), 71
- props (troposphere.ec2.SecurityGroupEgress attribute), 71
- props (troposphere.ec2.SecurityGroupIngress attribute), 71
- props (troposphere.ec2.SecurityGroupRule attribute), 71
- props (troposphere.ec2.SecurityGroups attribute), 71
- props (troposphere.ec2.SpotFleet attribute), 71
- props (troposphere.ec2.SpotFleetRequestConfigData attribute), 71
- props (troposphere.ec2.SpotFleetTagSpecification attribute), 71
- props (troposphere.ec2.SpotOptions attribute), 71
- props (troposphere.ec2.SpotOptionsRequest attribute), 72
- props (troposphere.ec2.SsmAssociations attribute), 72
- props (troposphere.ec2.Subnet attribute), 72
- props (troposphere.ec2.SubnetCidrBlock attribute), 72
- props (troposphere.ec2.SubnetNetworkAclAssociation attribute), 72
- props (troposphere.ec2.SubnetRouteTableAssociation attribute), 72
- props (troposphere.ec2.Tag attribute), 72
- props (troposphere.ec2.TagSpecifications attribute), 72
- props (troposphere.ec2.TargetCapacitySpecificationRequest attribute), 72
- props (troposphere.ec2.TargetGroup attribute), 72
- props (troposphere.ec2.TransitGateway attribute), 72
- props (troposphere.ec2.TransitGatewayAttachment attribute), 73
- props (troposphere.ec2.TransitGatewayRoute attribute), 73
- props (troposphere.ec2.TransitGatewayRouteTable attribute), 73
- props (troposphere.ec2.TransitGatewayRouteTableAssociation attribute), 73
- props (troposphere.ec2.TransitGatewayRouteTablePropagation attribute), 73
- props (troposphere.ec2.Volume attribute), 75
- props (troposphere.ec2.VolumeAttachment attribute), 75
- props (troposphere.ec2.VPC attribute), 73
- props (troposphere.ec2.VPCCidrBlock attribute), 73
- props (troposphere.ec2.VPCDHCPOptionsAssociation attribute), 73
- props (troposphere.ec2.VPCEndpoint attribute), 73
- props (troposphere.ec2.VPCEndpointConnectionNotification attribute), 74
- props (troposphere.ec2.VPCEndpointService attribute), 74
- props (troposphere.ec2.VPCEndpointServicePermissions attribute), 74
- props (troposphere.ec2.VPCGatewayAttachment attribute), 74
- props (troposphere.ec2.VPCPeeringConnection attribute), 74
- props (troposphere.ec2.VPNConnection attribute), 74
- props (troposphere.ec2.VPNConnectionRoute attribute), 74
- props (troposphere.ec2.VPNGateway attribute), 74
- props (troposphere.ec2.VPNGatewayRoutePropagation attribute), 74
- props (troposphere.ec2.VpnTunnelOptionsSpecification attribute), 75
- props (troposphere.ecr.LifecyclePolicy attribute), 75
- props (troposphere.ecr.Repository attribute), 75
- props (troposphere.ecs.AwsVpcConfiguration attribute), 75

- props (troposphere.ecs.Cluster attribute), 75
- props (troposphere.ecs.ContainerDefinition attribute), 75
- props (troposphere.ecs.DeploymentConfiguration attribute), 76
- props (troposphere.ecs.Device attribute), 76
- props (troposphere.ecs.DockerVolumeConfiguration attribute), 76
- props (troposphere.ecs.Environment attribute), 76
- props (troposphere.ecs.HealthCheck attribute), 76
- props (troposphere.ecs.Host attribute), 76
- props (troposphere.ecs.HostEntry attribute), 76
- props (troposphere.ecs.KernelCapabilities attribute), 76
- props (troposphere.ecs.LinuxParameters attribute), 76
- props (troposphere.ecs.LoadBalancer attribute), 76
- props (troposphere.ecs.LogConfiguration attribute), 76
- props (troposphere.ecs.MountPoint attribute), 76
- props (troposphere.ecs.NetworkConfiguration attribute), 76
- props (troposphere.ecs.PlacementConstraint attribute), 76
- props (troposphere.ecs.PlacementStrategy attribute), 77
- props (troposphere.ecs.PortMapping attribute), 77
- props (troposphere.ecs.RepositoryCredentials attribute), 77
- props (troposphere.ecs.Service attribute), 77
- props (troposphere.ecs.ServiceRegistry attribute), 77
- props (troposphere.ecs.TaskDefinition attribute), 77
- props (troposphere.ecs.Ulimit attribute), 77
- props (troposphere.ecs.Volume attribute), 77
- props (troposphere.ecs.VolumesFrom attribute), 77
- props (troposphere.efs.FileSystem attribute), 77
- props (troposphere.efs.MountTarget attribute), 78
- props (troposphere.eks.Cluster attribute), 78
- props (troposphere.eks.ResourcesVpcConfig attribute), 78
- props (troposphere.elasticache.CacheCluster attribute), 78
- props (troposphere.elasticache.NodeGroupConfiguration attribute), 78
- props (troposphere.elasticache.ParameterGroup attribute), 78
- props (troposphere.elasticache.ReplicationGroup attribute), 78
- props (troposphere.elasticache.SecurityGroup attribute), 79
- props (troposphere.elasticache.SecurityGroupIngress attribute), 79
- props (troposphere.elasticache.SubnetGroup attribute), 79
- props (troposphere.elasticbeanstalk.Application attribute), 79
- props (troposphere.elasticbeanstalk.ApplicationResourceLifecycleConfig attribute), 79
- props (troposphere.elasticbeanstalk.ApplicationVersion attribute), 79
- props (troposphere.elasticbeanstalk.ApplicationVersionLifecycleConfig attribute), 79
- props (troposphere.elasticbeanstalk.ConfigurationTemplate attribute), 79
- props (troposphere.elasticbeanstalk.Environment attribute), 80
- props (troposphere.elasticbeanstalk.MaxAgeRule attribute), 80
- props (troposphere.elasticbeanstalk.MaxCountRule attribute), 80
- props (troposphere.elasticbeanstalk.OptionSettings attribute), 80
- props (troposphere.elasticbeanstalk.SourceBundle attribute), 80
- props (troposphere.elasticbeanstalk.SourceConfiguration attribute), 80
- props (troposphere.elasticbeanstalk.Tier attribute), 80
- props (troposphere.elasticloadbalancing.AccessLoggingPolicy attribute), 80
- props (troposphere.elasticloadbalancing.AppCookieStickinessPolicy attribute), 80
- props (troposphere.elasticloadbalancing.ConnectionDrainingPolicy attribute), 80
- props (troposphere.elasticloadbalancing.ConnectionSettings attribute), 80
- props (troposphere.elasticloadbalancing.HealthCheck attribute), 81
- props (troposphere.elasticloadbalancing.LBCookieStickinessPolicy attribute), 81
- props (troposphere.elasticloadbalancing.Listener attribute), 81
- props (troposphere.elasticloadbalancing.LoadBalancer attribute), 81
- props (troposphere.elasticloadbalancing.Policy attribute), 81
- props (troposphere.elasticloadbalancingv2.Action attribute), 81
- props (troposphere.elasticloadbalancingv2.Certificate attribute), 81
- props (troposphere.elasticloadbalancingv2.Condition attribute), 81
- props (troposphere.elasticloadbalancingv2.FixedResponseConfig attribute), 81
- props (troposphere.elasticloadbalancingv2.Listener attribute), 81
- props (troposphere.elasticloadbalancingv2.ListenerCertificate attribute), 82
- props (troposphere.elasticloadbalancingv2.ListenerRule attribute), 82
- props (troposphere.elasticloadbalancingv2.LoadBalancer attribute), 82
- props (troposphere.elasticloadbalancingv2.LoadBalancerAttributes attribute), 82
- props (troposphere.elasticloadbalancingv2.Matcher attribute), 82

- tribute), 82
- props (troposphere.elasticloadbalancingv2.RedirectConfig attribute), 82
- props (troposphere.elasticloadbalancingv2.SubnetMapping attribute), 82
- props (troposphere.elasticloadbalancingv2.TargetDescription attribute), 82
- props (troposphere.elasticloadbalancingv2.TargetGroup attribute), 82
- props (troposphere.elasticloadbalancingv2.TargetGroupAttributes attribute), 83
- props (troposphere.elasticsearch.Domain attribute), 83
- props (troposphere.elasticsearch.EBSOptions attribute), 83
- props (troposphere.elasticsearch.ElasticsearchClusterConfig attribute), 83
- props (troposphere.elasticsearch.EncryptionAtRestOptions attribute), 83
- props (troposphere.elasticsearch.SnapshotOptions attribute), 83
- props (troposphere.elasticsearch.VPCOptions attribute), 83
- props (troposphere.emr.Application attribute), 83
- props (troposphere.emr.AutoScalingPolicy attribute), 83
- props (troposphere.emr.BootstrapActionConfig attribute), 83
- props (troposphere.emr.CloudWatchAlarmDefinition attribute), 84
- props (troposphere.emr.Cluster attribute), 84
- props (troposphere.emr.Configuration attribute), 84
- props (troposphere.emr.EbsBlockDeviceConfigs attribute), 84
- props (troposphere.emr.EbsConfiguration attribute), 84
- props (troposphere.emr.HadoopJarStepConfig attribute), 84
- props (troposphere.emr.InstanceFleetConfig attribute), 84
- props (troposphere.emr.InstanceFleetConfigProperty attribute), 84
- props (troposphere.emr.InstanceFleetProvisioningSpecification attribute), 84
- props (troposphere.emr.InstanceGroupConfig attribute), 84
- props (troposphere.emr.InstanceGroupConfigProperty attribute), 84
- props (troposphere.emr.InstanceTypeConfig attribute), 85
- props (troposphere.emr.JobFlowInstancesConfig attribute), 85
- props (troposphere.emr.KerberosAttributes attribute), 85
- props (troposphere.emr.KeyValue attribute), 85
- props (troposphere.emr.PlacementType attribute), 85
- props (troposphere.emr.ScalingAction attribute), 85
- props (troposphere.emr.ScalingConstraints attribute), 85
- props (troposphere.emr.ScalingRule attribute), 85
- props (troposphere.emr.ScalingTrigger attribute), 85
- props (troposphere.emr.ScriptBootstrapActionConfig attribute), 85
- props (troposphere.emr.SecurityConfiguration attribute), 85
- props (troposphere.emr.SimpleScalingPolicyConfiguration attribute), 85
- props (troposphere.emr.SpotProvisioningSpecification attribute), 86
- props (troposphere.emr.Step attribute), 86
- props (troposphere.emr.StepConfig attribute), 86
- props (troposphere.emr.VolumeSpecification attribute), 86
- props (troposphere.events.EcsParameters attribute), 86
- props (troposphere.events.InputTransformer attribute), 86
- props (troposphere.events.KinesisParameters attribute), 86
- props (troposphere.events.Rule attribute), 86
- props (troposphere.events.RunCommandParameters attribute), 86
- props (troposphere.events.RunCommandTarget attribute), 86
- props (troposphere.events.SqsParameters attribute), 87
- props (troposphere.events.Target attribute), 87
- props (troposphere.firehose.BufferingHints attribute), 87
- props (troposphere.firehose.CloudWatchLoggingOptions attribute), 87
- props (troposphere.firehose.CopyCommand attribute), 87
- props (troposphere.firehose.DeliveryStream attribute), 87
- props (troposphere.firehose.ElasticsearchDestinationConfiguration attribute), 87
- props (troposphere.firehose.EncryptionConfiguration attribute), 87
- props (troposphere.firehose.ExtendedS3DestinationConfiguration attribute), 87
- props (troposphere.firehose.KinesisStreamSourceConfiguration attribute), 87
- props (troposphere.firehose.KMSEncryptionConfig attribute), 87
- props (troposphere.firehose.ProcessingConfiguration attribute), 88
- props (troposphere.firehose.Processor attribute), 88
- props (troposphere.firehose.ProcessorParameter attribute), 88
- props (troposphere.firehose.RedshiftDestinationConfiguration attribute), 88
- props (troposphere.firehose.RetryOptions attribute), 88
- props (troposphere.firehose.S3Configuration attribute), 88
- props (troposphere.firehose.S3DestinationConfiguration attribute), 88
- props (troposphere.firehose.SplunkDestinationConfiguration attribute), 88
- props (troposphere.firehose.SplunkRetryOptions attribute), 88

- props (troposphere.glue.Action attribute), 88
- props (troposphere.glue.Classifier attribute), 88
- props (troposphere.glue.Column attribute), 89
- props (troposphere.glue.Condition attribute), 89
- props (troposphere.glue.Connection attribute), 89
- props (troposphere.glue.ConnectionInput attribute), 89
- props (troposphere.glue.ConnectionsList attribute), 89
- props (troposphere.glue.Crawler attribute), 89
- props (troposphere.glue.Database attribute), 89
- props (troposphere.glue.DatabaseInput attribute), 89
- props (troposphere.glue.DevEndpoint attribute), 89
- props (troposphere.glue.ExecutionProperty attribute), 89
- props (troposphere.glue.GrokClassifier attribute), 89
- props (troposphere.glue.JdbcTarget attribute), 90
- props (troposphere.glue.Job attribute), 90
- props (troposphere.glue.JobCommand attribute), 90
- props (troposphere.glue.JsonClassifier attribute), 90
- props (troposphere.glue.Order attribute), 90
- props (troposphere.glue.Partition attribute), 90
- props (troposphere.glue.PartitionInput attribute), 90
- props (troposphere.glue.PhysicalConnectionRequirements attribute), 90
- props (troposphere.glue.Predicate attribute), 90
- props (troposphere.glue.S3Target attribute), 90
- props (troposphere.glue.Schedule attribute), 90
- props (troposphere.glue.SchemaChangePolicy attribute), 90
- props (troposphere.glue.SerdeInfo attribute), 90
- props (troposphere.glue.SkewedInfo attribute), 91
- props (troposphere.glue.StorageDescriptor attribute), 91
- props (troposphere.glue.Table attribute), 91
- props (troposphere.glue.TableInput attribute), 91
- props (troposphere.glue.Targets attribute), 91
- props (troposphere.glue.Trigger attribute), 91
- props (troposphere.glue.XMLClassifier attribute), 91
- props (troposphere.guardduty.Condition attribute), 91
- props (troposphere.guardduty.Detector attribute), 91
- props (troposphere.guardduty.Filter attribute), 92
- props (troposphere.guardduty.FindingCriteria attribute), 92
- props (troposphere.guardduty.IPSet attribute), 92
- props (troposphere.guardduty.Master attribute), 92
- props (troposphere.guardduty.Member attribute), 92
- props (troposphere.guardduty.ThreatIntelSet attribute), 92
- props (troposphere.iam.AccessKey attribute), 92
- props (troposphere.iam.Group attribute), 92
- props (troposphere.iam.InstanceProfile attribute), 92
- props (troposphere.iam.LoginProfile attribute), 92
- props (troposphere.iam.ManagedPolicy attribute), 93
- props (troposphere.iam.Policy attribute), 93
- props (troposphere.iam.PolicyType attribute), 93
- props (troposphere.iam.Role attribute), 93
- props (troposphere.iam.ServiceLinkedRole attribute), 93
- props (troposphere.iam.User attribute), 93
- props (troposphere.iam.UserToGroupAddition attribute), 93
- props (troposphere.inspector.AssessmentTarget attribute), 93
- props (troposphere.inspector.AssessmentTemplate attribute), 93
- props (troposphere.inspector.ResourceGroup attribute), 94
- props (troposphere.iot.Action attribute), 94
- props (troposphere.iot.Certificate attribute), 94
- props (troposphere.iot.CloudwatchAlarmAction attribute), 94
- props (troposphere.iot.CloudwatchMetricAction attribute), 94
- props (troposphere.iot.DynamoDBAction attribute), 94
- props (troposphere.iot.DynamoDBv2Action attribute), 94
- props (troposphere.iot.ElasticsearchAction attribute), 94
- props (troposphere.iot.FirehoseAction attribute), 94
- props (troposphere.iot.KinesisAction attribute), 94
- props (troposphere.iot.LambdaAction attribute), 94
- props (troposphere.iot.Policy attribute), 95
- props (troposphere.iot.PolicyPrincipalAttachment attribute), 95
- props (troposphere.iot.PutItemInput attribute), 95
- props (troposphere.iot.RepublishAction attribute), 95
- props (troposphere.iot.S3Action attribute), 95
- props (troposphere.iot.SnsAction attribute), 95
- props (troposphere.iot.SqsAction attribute), 95
- props (troposphere.iot.Thing attribute), 95
- props (troposphere.iot.ThingPrincipalAttachment attribute), 95
- props (troposphere.iot.TopicRule attribute), 95
- props (troposphere.iot.TopicRulePayload attribute), 95
- props (troposphere.iotIclick.Device attribute), 96
- props (troposphere.iotIclick.Placement attribute), 96
- props (troposphere.iotIclick.PlacementTemplate attribute), 96
- props (troposphere.iotIclick.Project attribute), 96
- props (troposphere.iotanalytics.Action attribute), 96
- props (troposphere.iotanalytics.Activity attribute), 96
- props (troposphere.iotanalytics.ActivityChannel attribute), 96
- props (troposphere.iotanalytics.AddAttributes attribute), 96
- props (troposphere.iotanalytics.Channel attribute), 96
- props (troposphere.iotanalytics.ContainerAction attribute), 96
- props (troposphere.iotanalytics.Dataset attribute), 96
- props (troposphere.iotanalytics.DatasetContentVersionValue attribute), 97
- props (troposphere.iotanalytics.Datastore attribute), 97
- props (troposphere.iotanalytics.DeltaTime attribute), 97
- props (troposphere.iotanalytics.DeviceRegistryEnrich attribute), 97

- props (troposphere.iotanalytics.DeviceShadowEnrich attribute), 97
- props (troposphere.iotanalytics.Filter attribute), 97
- props (troposphere.iotanalytics.Lambda attribute), 97
- props (troposphere.iotanalytics.Math attribute), 97
- props (troposphere.iotanalytics.OutputFileUriValue attribute), 97
- props (troposphere.iotanalytics.Pipeline attribute), 97
- props (troposphere.iotanalytics.QueryAction attribute), 97
- props (troposphere.iotanalytics.QueryActionFilter attribute), 97
- props (troposphere.iotanalytics.RemoveAttributes attribute), 98
- props (troposphere.iotanalytics.ResourceConfiguration attribute), 98
- props (troposphere.iotanalytics.RetentionPeriod attribute), 98
- props (troposphere.iotanalytics.Schedule attribute), 98
- props (troposphere.iotanalytics.SelectAttributes attribute), 98
- props (troposphere.iotanalytics.Trigger attribute), 98
- props (troposphere.iotanalytics.TriggeringDataset attribute), 98
- props (troposphere.iotanalytics.Variable attribute), 98
- props (troposphere.kinesis.Stream attribute), 98
- props (troposphere.kinesis.StreamConsumer attribute), 98
- props (troposphere.kinesis.StreamEncryption attribute), 98
- props (troposphere.kms.Alias attribute), 99
- props (troposphere.kms.Key attribute), 99
- props (troposphere.logs.Destination attribute), 99
- props (troposphere.logs.LogGroup attribute), 99
- props (troposphere.logs.LogStream attribute), 99
- props (troposphere.logs.MetricFilter attribute), 99
- props (troposphere.logs.MetricTransformation attribute), 99
- props (troposphere.logs.SubscriptionFilter attribute), 99
- props (troposphere.neptune.DBCluster attribute), 99
- props (troposphere.neptune.DBClusterParameterGroup attribute), 100
- props (troposphere.neptune.DBInstance attribute), 100
- props (troposphere.neptune.DBParameterGroup attribute), 100
- props (troposphere.neptune.DBSubnetGroup attribute), 100
- props (troposphere.openstack.heat.AWSAutoScalingGroup attribute), 26
- props (troposphere.openstack.neutron.AddressPair attribute), 26
- props (troposphere.openstack.neutron.Firewall attribute), 26
- props (troposphere.openstack.neutron.FirewallPolicy attribute), 27
- props (troposphere.openstack.neutron.FirewallRule attribute), 27
- props (troposphere.openstack.neutron.FixedIP attribute), 27
- props (troposphere.openstack.neutron.FloatingIP attribute), 27
- props (troposphere.openstack.neutron.FloatingIPAssociation attribute), 27
- props (troposphere.openstack.neutron.HealthMonitor attribute), 27
- props (troposphere.openstack.neutron.LoadBalancer attribute), 27
- props (troposphere.openstack.neutron.Net attribute), 27
- props (troposphere.openstack.neutron.Pool attribute), 28
- props (troposphere.openstack.neutron.PoolMember attribute), 28
- props (troposphere.openstack.neutron.Port attribute), 28
- props (troposphere.openstack.neutron.SecurityGroup attribute), 28
- props (troposphere.openstack.neutron.SecurityGroupRule attribute), 28
- props (troposphere.openstack.neutron.SessionPersistence attribute), 28
- props (troposphere.openstack.neutron.VIP attribute), 28
- props (troposphere.openstack.nova.BlockDeviceMapping attribute), 28
- props (troposphere.openstack.nova.BlockDeviceMappingV2 attribute), 28
- props (troposphere.openstack.nova.FloatingIP attribute), 29
- props (troposphere.openstack.nova.FloatingIPAssociation attribute), 29
- props (troposphere.openstack.nova.KeyPair attribute), 29
- props (troposphere.openstack.nova.Network attribute), 29
- props (troposphere.openstack.nova.Server attribute), 29
- props (troposphere.opsworks.App attribute), 100
- props (troposphere.opsworks.AutoScalingThresholds attribute), 100
- props (troposphere.opsworks.BlockDeviceMapping attribute), 100
- props (troposphere.opsworks.ChefConfiguration attribute), 100
- props (troposphere.opsworks.DataSource attribute), 100
- props (troposphere.opsworks.EbsBlockDevice attribute), 101
- props (troposphere.opsworks.ElasticIp attribute), 101
- props (troposphere.opsworks.ElasticLoadBalancerAttachment attribute), 101
- props (troposphere.opsworks.EngineAttribute attribute), 101
- props (troposphere.opsworks.Environment attribute), 101
- props (troposphere.opsworks.Instance attribute), 101
- props (troposphere.opsworks.Layer attribute), 101
- props (troposphere.opsworks.LifeCycleConfiguration attribute), 101

- tribute), 101
- props (troposphere.opsworks.LoadBasedAutoScaling attribute), 101
- props (troposphere.opsworks.RdsDbInstance attribute), 101
- props (troposphere.opsworks.Recipes attribute), 101
- props (troposphere.opsworks.Server attribute), 101
- props (troposphere.opsworks.ShutdownEventConfiguration attribute), 102
- props (troposphere.opsworks.Source attribute), 102
- props (troposphere.opsworks.SslConfiguration attribute), 102
- props (troposphere.opsworks.Stack attribute), 102
- props (troposphere.opsworks.StackConfigurationManager attribute), 102
- props (troposphere.opsworks.TimeBasedAutoScaling attribute), 102
- props (troposphere.opsworks.UserProfile attribute), 102
- props (troposphere.opsworks.Volume attribute), 102
- props (troposphere.opsworks.VolumeConfiguration attribute), 102
- props (troposphere.Output attribute), 130
- props (troposphere.Parameter attribute), 130
- props (troposphere.policies.AutoScalingCreationPolicy attribute), 102
- props (troposphere.policies.AutoScalingReplacingUpdate attribute), 103
- props (troposphere.policies.AutoScalingRollingUpdate attribute), 103
- props (troposphere.policies.AutoScalingScheduledAction attribute), 103
- props (troposphere.policies.CodeDeployLambdaAliasUpdate attribute), 103
- props (troposphere.policies.CreationPolicy attribute), 103
- props (troposphere.policies.ResourceSignal attribute), 103
- props (troposphere.policies.UpdatePolicy attribute), 103
- props (troposphere.rds.DBCluster attribute), 103
- props (troposphere.rds.DBClusterParameterGroup attribute), 103
- props (troposphere.rds.DBInstance attribute), 103
- props (troposphere.rds.DBParameterGroup attribute), 103
- props (troposphere.rds.DBSecurityGroup attribute), 104
- props (troposphere.rds.DBSecurityGroupIngress attribute), 104
- props (troposphere.rds.DBSubnetGroup attribute), 104
- props (troposphere.rds.EventSubscription attribute), 104
- props (troposphere.rds.OptionConfiguration attribute), 104
- props (troposphere.rds.OptionGroup attribute), 104
- props (troposphere.rds.OptionSetting attribute), 104
- props (troposphere.rds.ProcessorFeature attribute), 104
- props (troposphere.rds.RDSSecurityGroup attribute), 104
- props (troposphere.rds.ScalingConfiguration attribute), 104
- props (troposphere.redshift.AmazonRedshiftParameter attribute), 105
- props (troposphere.redshift.Cluster attribute), 105
- props (troposphere.redshift.ClusterParameterGroup attribute), 105
- props (troposphere.redshift.ClusterSecurityGroup attribute), 105
- props (troposphere.redshift.ClusterSecurityGroupIngress attribute), 105
- props (troposphere.redshift.ClusterSubnetGroup attribute), 105
- props (troposphere.redshift.LoggingProperties attribute), 106
- props (troposphere.route53.AlarmIdentifier attribute), 106
- props (troposphere.route53.AliasTarget attribute), 106
- props (troposphere.route53.BaseRecordSet attribute), 106
- props (troposphere.route53.GeoLocation attribute), 106
- props (troposphere.route53.HealthCheck attribute), 106
- props (troposphere.route53.HealthCheckConfiguration attribute), 106
- props (troposphere.route53.HostedZone attribute), 106
- props (troposphere.route53.HostedZoneConfiguration attribute), 106
- props (troposphere.route53.HostedZoneVPCs attribute), 106
- props (troposphere.route53.IpAddressRequest attribute), 106
- props (troposphere.route53.QueryLoggingConfig attribute), 107
- props (troposphere.route53.RecordSetGroup attribute), 107
- props (troposphere.route53.ResolverEndpoint attribute), 107
- props (troposphere.route53.ResolverRule attribute), 107
- props (troposphere.route53.ResolverRuleAssociation attribute), 107
- props (troposphere.route53.TargetAddress attribute), 107
- props (troposphere.s3.AbortIncompleteMultipartUpload attribute), 107
- props (troposphere.s3.AccelerateConfiguration attribute), 107
- props (troposphere.s3.AccessControlTranslation attribute), 108
- props (troposphere.s3.AnalyticsConfiguration attribute), 108
- props (troposphere.s3.Bucket attribute), 108
- props (troposphere.s3.BucketEncryption attribute), 108
- props (troposphere.s3.BucketPolicy attribute), 108
- props (troposphere.s3.CorsConfiguration attribute), 108
- props (troposphere.s3.CorsRules attribute), 108
- props (troposphere.s3.DataExport attribute), 108
- props (troposphere.s3.Destination attribute), 108

- props (troposphere.s3.EncryptionConfiguration attribute), 108
- props (troposphere.s3.Filter attribute), 108
- props (troposphere.s3.InventoryConfiguration attribute), 108
- props (troposphere.s3.LambdaConfigurations attribute), 109
- props (troposphere.s3.LifecycleConfiguration attribute), 109
- props (troposphere.s3.LifecycleRule attribute), 109
- props (troposphere.s3.LifecycleRuleTransition attribute), 109
- props (troposphere.s3.LoggingConfiguration attribute), 109
- props (troposphere.s3.MetricsConfiguration attribute), 109
- props (troposphere.s3.NoncurrentVersionTransition attribute), 109
- props (troposphere.s3.NotificationConfiguration attribute), 109
- props (troposphere.s3.PublicAccessBlockConfiguration attribute), 109
- props (troposphere.s3.QueueConfigurations attribute), 109
- props (troposphere.s3.RedirectAllRequestsTo attribute), 109
- props (troposphere.s3.RedirectRule attribute), 109
- props (troposphere.s3.ReplicationConfiguration attribute), 109
- props (troposphere.s3.ReplicationConfigurationRules attribute), 110
- props (troposphere.s3.ReplicationConfigurationRulesDestinations attribute), 110
- props (troposphere.s3.RoutingRule attribute), 110
- props (troposphere.s3.RoutingRuleCondition attribute), 110
- props (troposphere.s3.Rules attribute), 110
- props (troposphere.s3.S3Key attribute), 110
- props (troposphere.s3.ServerSideEncryptionByDefault attribute), 110
- props (troposphere.s3.ServerSideEncryptionRule attribute), 110
- props (troposphere.s3.SourceSelectionCriteria attribute), 110
- props (troposphere.s3.SseKmsEncryptedObjects attribute), 110
- props (troposphere.s3.StorageClassAnalysis attribute), 110
- props (troposphere.s3.TagFilter attribute), 110
- props (troposphere.s3.TopicConfigurations attribute), 110
- props (troposphere.s3.VersioningConfiguration attribute), 111
- props (troposphere.s3.WebsiteConfiguration attribute), 111
- props (troposphere.sagemaker.ContainerDefinition attribute), 111
- props (troposphere.sagemaker.Endpoint attribute), 111
- props (troposphere.sagemaker.EndpointConfig attribute), 111
- props (troposphere.sagemaker.Model attribute), 111
- props (troposphere.sagemaker.NotebookInstance attribute), 111
- props (troposphere.sagemaker.NotebookInstanceLifecycleConfig attribute), 111
- props (troposphere.sagemaker.NotebookInstanceLifecycleHook attribute), 111
- props (troposphere.sagemaker.ProductionVariant attribute), 111
- props (troposphere.sagemaker.VpcConfig attribute), 112
- props (troposphere.sdb.Domain attribute), 112
- props (troposphere.secretsmanager.GenerateSecretString attribute), 112
- props (troposphere.secretsmanager.ResourcePolicy attribute), 112
- props (troposphere.secretsmanager.RotationRules attribute), 112
- props (troposphere.secretsmanager.RotationSchedule attribute), 112
- props (troposphere.secretsmanager.Secret attribute), 112
- props (troposphere.secretsmanager.SecretTargetAttachment attribute), 112
- props (troposphere.serverless.AlexaSkillEvent attribute), 113
- props (troposphere.serverless.Api attribute), 113
- props (troposphere.serverless.ApiEvent attribute), 113
- props (troposphere.serverless.Auth attribute), 113
- props (troposphere.serverless.Authorizers attribute), 113
- props (troposphere.serverless.CloudWatchEvent attribute), 113
- props (troposphere.serverless.CognitoAuth attribute), 113
- props (troposphere.serverless.CognitoAuthIdentity attribute), 113
- props (troposphere.serverless.Cors attribute), 113
- props (troposphere.serverless.DeadLetterQueue attribute), 113
- props (troposphere.serverless.DeploymentPreference attribute), 114
- props (troposphere.serverless.DynamoDBEvent attribute), 114
- props (troposphere.serverless.Function attribute), 114
- props (troposphere.serverless.FunctionForPackaging attribute), 114
- props (troposphere.serverless.Hooks attribute), 114
- props (troposphere.serverless.IoTRuleEvent attribute), 114
- props (troposphere.serverless.KinesisEvent attribute), 114
- props (troposphere.serverless.LambdaRequestAuth at-

- tribute), 114
- props (troposphere.serverless.LambdaRequestAuthIdentity attribute), 114
- props (troposphere.serverless.LambdaTokenAuth attribute), 115
- props (troposphere.serverless.LambdaTokenAuthIdentity attribute), 115
- props (troposphere.serverless.LayerVersion attribute), 115
- props (troposphere.serverless.PrimaryKey attribute), 115
- props (troposphere.serverless.S3Event attribute), 115
- props (troposphere.serverless.S3Location attribute), 115
- props (troposphere.serverless.ScheduleEvent attribute), 115
- props (troposphere.serverless.SimpleTable attribute), 115
- props (troposphere.serverless.SNSEvent attribute), 115
- props (troposphere.serverless.SQSEvent attribute), 115
- props (troposphere.servicecatalog.AcceptedPortfolioShare attribute), 116
- props (troposphere.servicecatalog.CloudFormationProduct attribute), 116
- props (troposphere.servicecatalog.CloudFormationProvisioningPlan attribute), 116
- props (troposphere.servicecatalog.LaunchNotificationConstraint attribute), 116
- props (troposphere.servicecatalog.LaunchRoleConstraint attribute), 116
- props (troposphere.servicecatalog.LaunchTemplateConstraint attribute), 116
- props (troposphere.servicecatalog.Portfolio attribute), 116
- props (troposphere.servicecatalog.PortfolioPrincipalAssociation attribute), 117
- props (troposphere.servicecatalog.PortfolioProductAssociation attribute), 117
- props (troposphere.servicecatalog.PortfolioShare attribute), 117
- props (troposphere.servicecatalog.ProvisioningArtifactProperties attribute), 117
- props (troposphere.servicecatalog.ProvisioningParameter attribute), 117
- props (troposphere.servicecatalog.TagOption attribute), 117
- props (troposphere.servicecatalog.TagOptionAssociation attribute), 117
- props (troposphere.servicediscovery.DnsConfig attribute), 117
- props (troposphere.servicediscovery.DnsRecord attribute), 118
- props (troposphere.servicediscovery.HealthCheckConfig attribute), 118
- props (troposphere.servicediscovery.HealthCheckCustomConfig attribute), 118
- props (troposphere.servicediscovery.HttpNamespace attribute), 118
- props (troposphere.servicediscovery.Instance attribute), 118
- props (troposphere.servicediscovery.PrivateDnsNamespace attribute), 118
- props (troposphere.servicediscovery.PublicDnsNamespace attribute), 118
- props (troposphere.servicediscovery.Service attribute), 118
- props (troposphere.ses.Action attribute), 118
- props (troposphere.ses.AddHeaderAction attribute), 118
- props (troposphere.ses.BounceAction attribute), 119
- props (troposphere.ses.CloudWatchDestination attribute), 119
- props (troposphere.ses.ConfigurationSet attribute), 119
- props (troposphere.ses.ConfigurationSetEventDestination attribute), 119
- props (troposphere.ses.DimensionConfiguration attribute), 119
- props (troposphere.ses.EmailTemplate attribute), 119
- props (troposphere.ses.EventDestination attribute), 119
- props (troposphere.ses.Filter attribute), 119
- props (troposphere.ses.IpFilter attribute), 119
- props (troposphere.ses.KinesisFirehoseDestination attribute), 119
- props (troposphere.ses.LambdaAction attribute), 119
- props (troposphere.ses.ReceiptFilter attribute), 119
- props (troposphere.ses.ReceiptRule attribute), 120
- props (troposphere.ses.ReceiptRuleSet attribute), 120
- props (troposphere.ses.Rule attribute), 120
- props (troposphere.ses.S3Action attribute), 120
- props (troposphere.ses.SNSAction attribute), 120
- props (troposphere.ses.StopAction attribute), 120
- props (troposphere.ses.Template attribute), 120
- props (troposphere.ses.WorkmailAction attribute), 120
- props (troposphere.sns.Subscription attribute), 120
- props (troposphere.sns.SubscriptionResource attribute), 120
- props (troposphere.sns.Topic attribute), 120
- props (troposphere.sns.TopicPolicy attribute), 121
- props (troposphere.sqs.Queue attribute), 121
- props (troposphere.sqs.QueuePolicy attribute), 121
- props (troposphere.sqs.RedrivePolicy attribute), 121
- props (troposphere.ssm.Association attribute), 121
- props (troposphere.ssm.Document attribute), 121
- props (troposphere.ssm.InstanceAssociationOutputLocation attribute), 121
- props (troposphere.ssm.LoggingInfo attribute), 121
- props (troposphere.ssm.MaintenanceWindow attribute), 121
- props (troposphere.ssm.MaintenanceWindowAutomationParameters attribute), 122
- props (troposphere.ssm.MaintenanceWindowLambdaParameters attribute), 122

- props (troposphere.ssm.MaintenanceWindowRunCommandParameters attribute), 122
 - props (troposphere.ssm.MaintenanceWindowStepFunctionsParameters attribute), 122
 - props (troposphere.ssm.MaintenanceWindowTarget attribute), 122
 - props (troposphere.ssm.MaintenanceWindowTask attribute), 122
 - props (troposphere.ssm.NotificationConfig attribute), 122
 - props (troposphere.ssm.Parameter attribute), 122
 - props (troposphere.ssm.PatchBaseline attribute), 122
 - props (troposphere.ssm.PatchFilter attribute), 122
 - props (troposphere.ssm.PatchFilterGroup attribute), 122
 - props (troposphere.ssm.ResourceDataSync attribute), 123
 - props (troposphere.ssm.Rule attribute), 123
 - props (troposphere.ssm.RuleGroup attribute), 123
 - props (troposphere.ssm.S3OutputLocation attribute), 123
 - props (troposphere.ssm.Targets attribute), 123
 - props (troposphere.ssm.TaskInvocationParameters attribute), 123
 - props (troposphere.stepfunctions.Activity attribute), 123
 - props (troposphere.stepfunctions.StateMachine attribute), 123
 - props (troposphere.Template attribute), 131
 - props (troposphere.waf.Action attribute), 125
 - props (troposphere.waf.ByteMatchSet attribute), 125
 - props (troposphere.waf.ByteMatchTuples attribute), 125
 - props (troposphere.waf.FieldToMatch attribute), 126
 - props (troposphere.waf.IPSet attribute), 126
 - props (troposphere.waf.IPSetDescriptors attribute), 126
 - props (troposphere.waf.Predicates attribute), 126
 - props (troposphere.waf.Rule attribute), 126
 - props (troposphere.waf.Rules attribute), 126
 - props (troposphere.waf.SizeConstraint attribute), 126
 - props (troposphere.waf.SizeConstraintSet attribute), 126
 - props (troposphere.waf.SqlInjectionMatchSet attribute), 126
 - props (troposphere.waf.SqlInjectionMatchTuples attribute), 126
 - props (troposphere.waf.WebACL attribute), 126
 - props (troposphere.waf.XssMatchSet attribute), 127
 - props (troposphere.waf.XssMatchTuple attribute), 127
 - props (troposphere.wafregional.Action attribute), 127
 - props (troposphere.wafregional.ByteMatchSet attribute), 127
 - props (troposphere.wafregional.ByteMatchTuples attribute), 127
 - props (troposphere.wafregional.FieldToMatch attribute), 127
 - props (troposphere.wafregional.IPSet attribute), 127
 - props (troposphere.wafregional.IPSetDescriptors attribute), 127
 - props (troposphere.wafregional.Predicates attribute), 127
 - props (troposphere.wafregional.Rule attribute), 127
 - props (troposphere.wafregional.Rules attribute), 127
 - props (troposphere.wafregional.SizeConstraint attribute), 128
 - props (troposphere.wafregional.SizeConstraintSet attribute), 128
 - props (troposphere.wafregional.SqlInjectionMatchSet attribute), 128
 - props (troposphere.wafregional.SqlInjectionMatchTuples attribute), 128
 - props (troposphere.wafregional.WebACL attribute), 128
 - props (troposphere.wafregional.WebACLAssociation attribute), 128
 - props (troposphere.wafregional.XssMatchSet attribute), 128
 - props (troposphere.wafregional.XssMatchTuple attribute), 128
 - props (troposphere.workspaces.Workspace attribute), 128
 - props (troposphere.workspaces.WorkspaceProperties attribute), 128
 - provisioned_throughput_validator() (in module troposphere.efs), 78
 - ProvisionedThroughput (class in troposphere.dynamodb), 66
 - ProvisioningArtifactProperties (class in troposphere.servicecatalog), 117
 - ProvisioningParameter (class in troposphere.servicecatalog), 117
 - PublicAccessBlockConfiguration (class in troposphere.s3), 109
 - PublicDnsNamespace (class in troposphere.servicediscovery), 118
 - PushSync (class in troposphere.cognito), 59
 - PutItemInput (class in troposphere.iot), 95
- ## Q
- QueryAction (class in troposphere.iotanalytics), 97
 - QueryActionFilter (class in troposphere.iotanalytics), 97
 - QueryLoggingConfig (class in troposphere.route53), 106
 - Queue (class in troposphere.sqs), 121
 - QueueConfigurations (class in troposphere.s3), 109
 - QueuePolicy (class in troposphere.sqs), 121
 - QuotaSettings (class in troposphere.apigateway), 34
- ## R
- RdsDbInstance (class in troposphere.opsworks), 101
 - RdsHttpEndpointConfig (class in troposphere.appsync), 40
 - RDSSecurityGroup (class in troposphere.rds), 104
 - ReceiptFilter (class in troposphere.ses), 119
 - ReceiptRule (class in troposphere.ses), 120
 - ReceiptRuleSet (class in troposphere.ses), 120
 - Recipes (class in troposphere.opsworks), 101
 - RecordColumn (class in troposphere.analytics), 31
 - RecordFormat (class in troposphere.analytics), 32

- RecordingGroup (class in troposphere.config), 61
- RecordSet (class in troposphere.route53), 107
- RecordSetGroup (class in troposphere.route53), 107
- RecordSetType (class in troposphere.route53), 107
- RedirectAllRequestsTo (class in troposphere.s3), 109
- RedirectConfig (class in troposphere.elasticloadbalancingv2), 82
- RedirectRule (class in troposphere.s3), 109
- RedrivePolicy (class in troposphere.sqs), 121
- RedshiftDestinationConfiguration (class in troposphere.firehose), 88
- Ref (class in troposphere), 131
- Ref() (troposphere.AWSDeclaration method), 129
- ref() (troposphere.AWSDeclaration method), 129
- Ref() (troposphere.AWSObject method), 129
- ref() (troposphere.AWSObject method), 129
- ReferenceDataSource (class in troposphere.analytics), 32
- ReferenceSchema (class in troposphere.analytics), 32
- RegistryCredential (class in troposphere.codebuild), 54
- RelationalDatabaseConfig (class in troposphere.appsync), 40
- RemoveAttributes (class in troposphere.iotanalytics), 98
- ReplicationConfiguration (class in troposphere.s3), 109
- ReplicationConfigurationRules (class in troposphere.s3), 109
- ReplicationConfigurationRulesDestination (class in troposphere.s3), 110
- ReplicationGroup (class in troposphere.elasticache), 78
- ReplicationInstance (class in troposphere.dms), 64
- ReplicationSubnetGroup (class in troposphere.dms), 64
- ReplicationTask (class in troposphere.dms), 64
- Repository (class in troposphere.cloud9), 48
- Repository (class in troposphere.codecommit), 54
- Repository (class in troposphere.ecr), 75
- RepositoryCredentials (class in troposphere.ecs), 77
- RepublishAction (class in troposphere.iot), 95
- RequestValidator (class in troposphere.apigateway), 34
- Resolver (class in troposphere.appsync), 40
- resolver_kind_validator() (in module troposphere.appsync), 41
- ResolverEndpoint (class in troposphere.route53), 107
- ResolverRule (class in troposphere.route53), 107
- ResolverRuleAssociation (class in troposphere.route53), 107
- Resource (class in troposphere.apigateway), 34
- resource_type (tests.test_template_generator.MyCustomResource attribute), 144
- resource_type (tests.test_template_generator.MyMacroResource attribute), 144
- resource_type (troposphere.amazonmq.Broker attribute), 29
- resource_type (troposphere.amazonmq.Configuration attribute), 29
- resource_type (troposphere.amazonmq.ConfigurationAssociation attribute), 30
- resource_type (troposphere.analytics.Application attribute), 30
- resource_type (troposphere.analytics.ApplicationOutput attribute), 30
- resource_type (troposphere.analytics.ApplicationReferenceDataSource attribute), 30
- resource_type (troposphere.apigateway.Account attribute), 32
- resource_type (troposphere.apigateway.ApiKey attribute), 32
- resource_type (troposphere.apigateway.Authorizer attribute), 32
- resource_type (troposphere.apigateway.BasePathMapping attribute), 32
- resource_type (troposphere.apigateway.ClientCertificate attribute), 33
- resource_type (troposphere.apigateway.Deployment attribute), 33
- resource_type (troposphere.apigateway.DocumentationPart attribute), 33
- resource_type (troposphere.apigateway.DocumentationVersion attribute), 33
- resource_type (troposphere.apigateway.DomainName attribute), 33
- resource_type (troposphere.apigateway.GatewayResponse attribute), 33
- resource_type (troposphere.apigateway.Method attribute), 34
- resource_type (troposphere.apigateway.Model attribute), 34
- resource_type (troposphere.apigateway.RequestValidator attribute), 34
- resource_type (troposphere.apigateway.Resource attribute), 34
- resource_type (troposphere.apigateway.RestApi attribute), 34
- resource_type (troposphere.apigateway.Stage attribute), 35
- resource_type (troposphere.apigateway.UsagePlan attribute), 35
- resource_type (troposphere.apigateway.UsagePlanKey attribute), 35
- resource_type (troposphere.apigateway.VpcLink attribute), 35
- resource_type (troposphere.apigatewayv2.Api attribute), 35
- resource_type (troposphere.apigatewayv2.Authorizer attribute), 36
- resource_type (troposphere.apigatewayv2.Deployment attribute), 36
- resource_type (troposphere.apigatewayv2.Integration attribute), 36

- resource_type (troposphere.apigatewayv2.IntegrationResponse attribute), 36
- resource_type (troposphere.apigatewayv2.Model attribute), 36
- resource_type (troposphere.apigatewayv2.Route attribute), 36
- resource_type (troposphere.apigatewayv2.RouteResponse attribute), 36
- resource_type (troposphere.apigatewayv2.Stage attribute), 36
- resource_type (troposphere.applicationautoscaling.ScalableTarget attribute), 37
- resource_type (troposphere.applicationautoscaling.ScalingPolicy attribute), 37
- resource_type (troposphere.appstream.DirectoryConfiguration attribute), 38
- resource_type (troposphere.appstream.Fleet attribute), 38
- resource_type (troposphere.appstream.ImageBuilder attribute), 38
- resource_type (troposphere.appstream.Stack attribute), 38
- resource_type (troposphere.appstream.StackFleetAssociation attribute), 39
- resource_type (troposphere.appstream.StackUserAssociation attribute), 39
- resource_type (troposphere.appstream.User attribute), 39
- resource_type (troposphere.appsync.ApiKey attribute), 39
- resource_type (troposphere.appsync.DataSource attribute), 39
- resource_type (troposphere.appsync.FunctionConfiguration attribute), 40
- resource_type (troposphere.appsync.GraphQLApi attribute), 40
- resource_type (troposphere.appsync.GraphQLSchema attribute), 40
- resource_type (troposphere.appsync.Resolver attribute), 41
- resource_type (troposphere.ask.Skill attribute), 41
- resource_type (troposphere.athena.NamedQuery attribute), 41
- resource_type (troposphere.autoscaling.AutoScalingGroup attribute), 41
- resource_type (troposphere.autoscaling.LaunchConfiguration attribute), 42
- resource_type (troposphere.autoscaling.LifecycleHook attribute), 42
- resource_type (troposphere.autoscaling.ScalingPolicy attribute), 43
- resource_type (troposphere.autoscaling.ScheduledAction attribute), 43
- resource_type (troposphere.autoscaling.Trigger attribute), 43
- resource_type (troposphere.autoscalingplans.ScalingPlan attribute), 44
- resource_type (troposphere.awslambda.Alias attribute), 45
- resource_type (troposphere.awslambda.EventSourceMapping attribute), 45
- resource_type (troposphere.awslambda.Function attribute), 45
- resource_type (troposphere.awslambda.LayerVersion attribute), 45
- resource_type (troposphere.awslambda.LayerVersionPermission attribute), 45
- resource_type (troposphere.awslambda.Permission attribute), 46
- resource_type (troposphere.awslambda.Version attribute), 46
- resource_type (troposphere.batch.ComputeEnvironment attribute), 46
- resource_type (troposphere.batch.JobDefinition attribute), 47
- resource_type (troposphere.batch.JobQueue attribute), 47
- resource_type (troposphere.budgets.Budget attribute), 47
- resource_type (troposphere.certificatemanager.Certificate attribute), 48
- resource_type (troposphere.cloud9.EnvironmentEC2 attribute), 48
- resource_type (troposphere.cloudformation.CustomResource attribute), 49
- resource_type (troposphere.cloudformation.Macro attribute), 50
- resource_type (troposphere.cloudformation.Stack attribute), 50
- resource_type (troposphere.cloudformation.WaitCondition attribute), 50
- resource_type (troposphere.cloudformation.WaitConditionHandle attribute), 50
- resource_type (troposphere.cloudfront.CloudFrontOriginAccessIdentity attribute), 50
- resource_type (troposphere.cloudfront.Distribution attribute), 51
- resource_type (troposphere.cloudfront.StreamingDistribution attribute), 52
- resource_type (troposphere.cloudtrail.Trail attribute), 52
- resource_type (troposphere.cloudwatch.Alarm attribute), 52
- resource_type (troposphere.cloudwatch.Dashboard attribute), 53
- resource_type (troposphere.codebuild.Project attribute), 54
- resource_type (troposphere.codecommit.Repository attribute), 54
- resource_type (troposphere.codedeploy.Application attribute), 55
- resource_type (troposphere.codedeploy.DeploymentConfiguration attribute), 55
- resource_type (troposphere.codedeploy.DeploymentGroup attribute), 55

attribute), 55

resource_type (troposphere.codepipeline.CustomActionType attribute), 57

resource_type (troposphere.codepipeline.Pipeline attribute), 58

resource_type (troposphere.codepipeline.Webhook attribute), 58

resource_type (troposphere.cognito.IdentityPool attribute), 59

resource_type (troposphere.cognito.IdentityPoolRoleAttachment attribute), 59

resource_type (troposphere.cognito.UserPool attribute), 60

resource_type (troposphere.cognito.UserPoolClient attribute), 60

resource_type (troposphere.cognito.UserPoolGroup attribute), 60

resource_type (troposphere.cognito.UserPoolUser attribute), 60

resource_type (troposphere.cognito.UserPoolUserToGroupAttachment attribute), 60

resource_type (troposphere.config.AggregationAuthorization attribute), 61

resource_type (troposphere.config.ConfigRule attribute), 61

resource_type (troposphere.config.ConfigurationAggregator attribute), 61

resource_type (troposphere.config.ConfigurationRecorder attribute), 61

resource_type (troposphere.config.DeliveryChannel attribute), 61

resource_type (troposphere.datapipeline.Pipeline attribute), 62

resource_type (troposphere.dax.Cluster attribute), 62

resource_type (troposphere.dax.ParameterGroup attribute), 62

resource_type (troposphere.dax.SubnetGroup attribute), 63

resource_type (troposphere.directoryservice.MicrosoftAD attribute), 63

resource_type (troposphere.directoryservice.SimpleAD attribute), 63

resource_type (troposphere.dlm.LifecyclePolicy attribute), 63

resource_type (troposphere.dms.Certificate attribute), 64

resource_type (troposphere.dms.Endpoint attribute), 64

resource_type (troposphere.dms.EventSubscription attribute), 64

resource_type (troposphere.dms.ReplicationInstance attribute), 64

resource_type (troposphere.dms.ReplicationSubnetGroup attribute), 64

resource_type (troposphere.dms.ReplicationTask attribute), 64

resource_type (troposphere.docdb.DBCluster attribute), 65

resource_type (troposphere.docdb.DBClusterParameterGroup attribute), 65

resource_type (troposphere.docdb.DBInstance attribute), 65

resource_type (troposphere.docdb.DBSubnetGroup attribute), 65

resource_type (troposphere.dynamodb.Table attribute), 66

resource_type (troposphere.ec2.CustomerGateway attribute), 67

resource_type (troposphere.ec2.DHCPOptions attribute), 67

resource_type (troposphere.ec2.EC2Fleet attribute), 67

resource_type (troposphere.ec2.EgressOnlyInternetGateway attribute), 67

resource_type (troposphere.ec2.EIP attribute), 67

resource_type (troposphere.ec2.EIPAssociation attribute), 67

resource_type (troposphere.ec2.FlowLog attribute), 68

resource_type (troposphere.ec2.Host attribute), 68

resource_type (troposphere.ec2.Instance attribute), 68

resource_type (troposphere.ec2.InternetGateway attribute), 68

resource_type (troposphere.ec2.LaunchTemplate attribute), 68

resource_type (troposphere.ec2.NatGateway attribute), 69

resource_type (troposphere.ec2.NetworkAcl attribute), 69

resource_type (troposphere.ec2.NetworkAclEntry attribute), 69

resource_type (troposphere.ec2.NetworkInterface attribute), 70

resource_type (troposphere.ec2.NetworkInterfaceAttachment attribute), 70

resource_type (troposphere.ec2.NetworkInterfacePermission attribute), 70

resource_type (troposphere.ec2.PlacementGroup attribute), 70

resource_type (troposphere.ec2.Route attribute), 70

resource_type (troposphere.ec2.RouteTable attribute), 71

resource_type (troposphere.ec2.SecurityGroup attribute), 71

resource_type (troposphere.ec2.SecurityGroupEgress attribute), 71

resource_type (troposphere.ec2.SecurityGroupIngress attribute), 71

resource_type (troposphere.ec2.SpotFleet attribute), 71

resource_type (troposphere.ec2.Subnet attribute), 72

resource_type (troposphere.ec2.SubnetCidrBlock attribute), 72

resource_type (troposphere.ec2.SubnetNetworkAclAssociation attribute), 72

- resource_type (troposphere.ec2.SubnetRouteTableAssociation attribute), 72
- resource_type (troposphere.ec2.TransitGateway attribute), 72
- resource_type (troposphere.ec2.TransitGatewayAttachment attribute), 73
- resource_type (troposphere.ec2.TransitGatewayRoute attribute), 73
- resource_type (troposphere.ec2.TransitGatewayRouteTable attribute), 73
- resource_type (troposphere.ec2.TransitGatewayRouteTableAssociation attribute), 73
- resource_type (troposphere.ec2.TransitGatewayRouteTablePropagation attribute), 73
- resource_type (troposphere.ec2.Volume attribute), 75
- resource_type (troposphere.ec2.VolumeAttachment attribute), 75
- resource_type (troposphere.ec2.VPC attribute), 73
- resource_type (troposphere.ec2.VPCCidrBlock attribute), 73
- resource_type (troposphere.ec2.VPCDHCPOptionsAssociation attribute), 73
- resource_type (troposphere.ec2.VPCEndpoint attribute), 74
- resource_type (troposphere.ec2.VPCEndpointConnectionNotification attribute), 74
- resource_type (troposphere.ec2.VPCEndpointService attribute), 74
- resource_type (troposphere.ec2.VPCEndpointServicePermissions attribute), 74
- resource_type (troposphere.ec2.VPCGatewayAttachment attribute), 74
- resource_type (troposphere.ec2.VPCPeeringConnection attribute), 74
- resource_type (troposphere.ec2.VPNConnection attribute), 74
- resource_type (troposphere.ec2.VPNConnectionRoute attribute), 74
- resource_type (troposphere.ec2.VPNGateway attribute), 74
- resource_type (troposphere.ec2.VPNGatewayRoutePropagation attribute), 75
- resource_type (troposphere.ecr.Repository attribute), 75
- resource_type (troposphere.ecs.Cluster attribute), 75
- resource_type (troposphere.ecs.Service attribute), 77
- resource_type (troposphere.ecs.TaskDefinition attribute), 77
- resource_type (troposphere.efs.FileSystem attribute), 77
- resource_type (troposphere.efs.MountTarget attribute), 78
- resource_type (troposphere.eks.Cluster attribute), 78
- resource_type (troposphere.elasticache.CacheCluster attribute), 78
- resource_type (troposphere.elasticache.ParameterGroup attribute), 78
- resource_type (troposphere.elasticache.ReplicationGroup attribute), 78
- resource_type (troposphere.elasticache.SecurityGroup attribute), 79
- resource_type (troposphere.elasticache.SecurityGroupIngress attribute), 79
- resource_type (troposphere.elasticache.SubnetGroup attribute), 79
- resource_type (troposphere.elasticbeanstalk.Application attribute), 79
- resource_type (troposphere.elasticbeanstalk.ApplicationVersion attribute), 79
- resource_type (troposphere.elasticbeanstalk.ConfigurationTemplate attribute), 79
- resource_type (troposphere.elasticbeanstalk.Environment attribute), 80
- resource_type (troposphere.elasticloadbalancing.LoadBalancer attribute), 81
- resource_type (troposphere.elasticloadbalancingv2.Listener attribute), 81
- resource_type (troposphere.elasticloadbalancingv2.ListenerCertificate attribute), 82
- resource_type (troposphere.elasticloadbalancingv2.ListenerRule attribute), 82
- resource_type (troposphere.elasticloadbalancingv2.LoadBalancer attribute), 82
- resource_type (troposphere.elasticloadbalancingv2.TargetGroup attribute), 82
- resource_type (troposphere.elasticsearch.Domain attribute), 83
- resource_type (troposphere.emr.Cluster attribute), 84
- resource_type (troposphere.emr.InstanceFleetConfig attribute), 84
- resource_type (troposphere.emr.InstanceGroupConfig attribute), 84
- resource_type (troposphere.emr.SecurityConfiguration attribute), 85
- resource_type (troposphere.emr.Step attribute), 86
- resource_type (troposphere.events.Rule attribute), 86
- resource_type (troposphere.firehose.DeliveryStream attribute), 87
- resource_type (troposphere.glue.Classifier attribute), 89
- resource_type (troposphere.glue.Connection attribute), 89
- resource_type (troposphere.glue.Crawler attribute), 89
- resource_type (troposphere.glue.Database attribute), 89
- resource_type (troposphere.glue.DevEndpoint attribute), 89
- resource_type (troposphere.glue.Job attribute), 90
- resource_type (troposphere.glue.Partition attribute), 90
- resource_type (troposphere.glue.Table attribute), 91
- resource_type (troposphere.glue.Trigger attribute), 91
- resource_type (troposphere.guardduty.Detector attribute), 91
- resource_type (troposphere.guardduty.Filter attribute), 92

- resource_type (troposphere.guarddduty.IPSet attribute), 92
- resource_type (troposphere.guarddduty.Master attribute), 92
- resource_type (troposphere.guarddduty.Member attribute), 92
- resource_type (troposphere.guarddduty.ThreatIntelSet attribute), 92
- resource_type (troposphere.iam.AccessKey attribute), 92
- resource_type (troposphere.iam.Group attribute), 92
- resource_type (troposphere.iam.InstanceProfile attribute), 92
- resource_type (troposphere.iam.ManagedPolicy attribute), 93
- resource_type (troposphere.iam.PolicyType attribute), 93
- resource_type (troposphere.iam.Role attribute), 93
- resource_type (troposphere.iam.ServiceLinkedRole attribute), 93
- resource_type (troposphere.iam.User attribute), 93
- resource_type (troposphere.iam.UserToGroupAddition attribute), 93
- resource_type (troposphere.inspector.AssessmentTarget attribute), 93
- resource_type (troposphere.inspector.AssessmentTemplate attribute), 94
- resource_type (troposphere.inspector.ResourceGroup attribute), 94
- resource_type (troposphere.iot.Certificate attribute), 94
- resource_type (troposphere.iot.Policy attribute), 95
- resource_type (troposphere.iot.PolicyPrincipalAttachment attribute), 95
- resource_type (troposphere.iot.Thing attribute), 95
- resource_type (troposphere.iot.ThingPrincipalAttachment attribute), 95
- resource_type (troposphere.iot.TopicRule attribute), 95
- resource_type (troposphere.iot1click.Device attribute), 96
- resource_type (troposphere.iot1click.Placement attribute), 96
- resource_type (troposphere.iot1click.Project attribute), 96
- resource_type (troposphere.iotanalytics.Channel attribute), 96
- resource_type (troposphere.iotanalytics.Dataset attribute), 97
- resource_type (troposphere.iotanalytics.Datastore attribute), 97
- resource_type (troposphere.iotanalytics.Pipeline attribute), 97
- resource_type (troposphere.kinesis.Stream attribute), 98
- resource_type (troposphere.kinesis.StreamConsumer attribute), 98
- resource_type (troposphere.kms.Alias attribute), 99
- resource_type (troposphere.kms.Key attribute), 99
- resource_type (troposphere.logs.Destination attribute), 99
- resource_type (troposphere.logs.LogGroup attribute), 99
- resource_type (troposphere.logs.LogStream attribute), 99
- resource_type (troposphere.logs.MetricFilter attribute), 99
- resource_type (troposphere.logs.SubscriptionFilter attribute), 99
- resource_type (troposphere.neptune.DBCluster attribute), 100
- resource_type (troposphere.neptune.DBClusterParameterGroup attribute), 100
- resource_type (troposphere.neptune.DBInstance attribute), 100
- resource_type (troposphere.neptune.DBParameterGroup attribute), 100
- resource_type (troposphere.neptune.DBSubnetGroup attribute), 100
- resource_type (troposphere.openstack.heat.AWSAutoScalingGroup attribute), 26
- resource_type (troposphere.openstack.neutron.Firewall attribute), 26
- resource_type (troposphere.openstack.neutron.FirewallPolicy attribute), 27
- resource_type (troposphere.openstack.neutron.FirewallRule attribute), 27
- resource_type (troposphere.openstack.neutron.FloatingIP attribute), 27
- resource_type (troposphere.openstack.neutron.FloatingIPAssociation attribute), 27
- resource_type (troposphere.openstack.neutron.HealthMonitor attribute), 27
- resource_type (troposphere.openstack.neutron.LoadBalancer attribute), 27
- resource_type (troposphere.openstack.neutron.Net attribute), 27
- resource_type (troposphere.openstack.neutron.Pool attribute), 28
- resource_type (troposphere.openstack.neutron.PoolMember attribute), 28
- resource_type (troposphere.openstack.neutron.Port attribute), 28
- resource_type (troposphere.openstack.neutron.SecurityGroup attribute), 28
- resource_type (troposphere.openstack.nova.FloatingIP attribute), 29
- resource_type (troposphere.openstack.nova.FloatingIPAssociation attribute), 29
- resource_type (troposphere.openstack.nova.KeyPair attribute), 29
- resource_type (troposphere.openstack.nova.Server attribute), 29
- resource_type (troposphere.opsworks.App attribute), 100
- resource_type (troposphere.opsworks.ElasticLoadBalancerAttachment attribute), 101
- resource_type (troposphere.opsworks.Instance attribute), 101
- resource_type (troposphere.opsworks.Layer attribute),

- 101
- resource_type (troposphere.opsworks.Server attribute), 101
- resource_type (troposphere.opsworks.Stack attribute), 102
- resource_type (troposphere.opsworks.UserProfile attribute), 102
- resource_type (troposphere.opsworks.Volume attribute), 102
- resource_type (troposphere.rds.DBCluster attribute), 103
- resource_type (troposphere.rds.DBClusterParameterGroup attribute), 103
- resource_type (troposphere.rds.DBInstance attribute), 103
- resource_type (troposphere.rds.DBParameterGroup attribute), 103
- resource_type (troposphere.rds.DBSecurityGroup attribute), 104
- resource_type (troposphere.rds.DBSecurityGroupIngress attribute), 104
- resource_type (troposphere.rds.DBSubnetGroup attribute), 104
- resource_type (troposphere.rds.EventSubscription attribute), 104
- resource_type (troposphere.rds.OptionGroup attribute), 104
- resource_type (troposphere.redshift.Cluster attribute), 105
- resource_type (troposphere.redshift.ClusterParameterGroup attribute), 105
- resource_type (troposphere.redshift.ClusterSecurityGroup attribute), 105
- resource_type (troposphere.redshift.ClusterSecurityGroupIngress attribute), 105
- resource_type (troposphere.redshift.ClusterSubnetGroup attribute), 105
- resource_type (troposphere.route53.HealthCheck attribute), 106
- resource_type (troposphere.route53.HostedZone attribute), 106
- resource_type (troposphere.route53.RecordSetGroup attribute), 107
- resource_type (troposphere.route53.RecordSetType attribute), 107
- resource_type (troposphere.route53.ResolverEndpoint attribute), 107
- resource_type (troposphere.route53.ResolverRule attribute), 107
- resource_type (troposphere.route53.ResolverRuleAssociation attribute), 107
- resource_type (troposphere.s3.Bucket attribute), 108
- resource_type (troposphere.s3.BucketPolicy attribute), 108
- resource_type (troposphere.sagemaker.Endpoint attribute), 111
- resource_type (troposphere.sagemaker.EndpointConfig attribute), 111
- resource_type (troposphere.sagemaker.Model attribute), 111
- resource_type (troposphere.sagemaker.NotebookInstance attribute), 111
- resource_type (troposphere.sagemaker.NotebookInstanceLifecycleConfig attribute), 111
- resource_type (troposphere.sdb.Domain attribute), 112
- resource_type (troposphere.secretsmanager.ResourcePolicy attribute), 112
- resource_type (troposphere.secretsmanager.RotationSchedule attribute), 112
- resource_type (troposphere.secretsmanager.Secret attribute), 112
- resource_type (troposphere.secretsmanager.SecretTargetAttachment attribute), 112
- resource_type (troposphere.serverless.AlexaSkillEvent attribute), 113
- resource_type (troposphere.serverless.Api attribute), 113
- resource_type (troposphere.serverless.ApiEvent attribute), 113
- resource_type (troposphere.serverless.CloudWatchEvent attribute), 113
- resource_type (troposphere.serverless.DynamoDBEvent attribute), 114
- resource_type (troposphere.serverless.Function attribute), 114
- resource_type (troposphere.serverless.FunctionForPackaging attribute), 114
- resource_type (troposphere.serverless.IoTRuleEvent attribute), 114
- resource_type (troposphere.serverless.KinesisEvent attribute), 114
- resource_type (troposphere.serverless.LayerVersion attribute), 115
- resource_type (troposphere.serverless.S3Event attribute), 115
- resource_type (troposphere.serverless.ScheduleEvent attribute), 115
- resource_type (troposphere.serverless.SimpleTable attribute), 115
- resource_type (troposphere.serverless.SNSEvent attribute), 115
- resource_type (troposphere.serverless.SQSEvent attribute), 115
- resource_type (troposphere.servicecatalog.AcceptedPortfolioShare attribute), 116
- resource_type (troposphere.servicecatalog.CloudFormationProduct attribute), 116
- resource_type (troposphere.servicecatalog.CloudFormationProvisionedProduct attribute), 116
- resource_type (troposphere.servicecatalog.LaunchNotificationConstraint attribute), 116

- attribute), 116
- resource_type (troposphere.servicecatalog.LaunchRoleConstraint attribute), 116
- resource_type (troposphere.servicecatalog.LaunchTemplateConstraint attribute), 116
- resource_type (troposphere.servicecatalog.Portfolio attribute), 116
- resource_type (troposphere.servicecatalog.PortfolioPrincipalAssociation attribute), 117
- resource_type (troposphere.servicecatalog.PortfolioProductAssociation attribute), 117
- resource_type (troposphere.servicecatalog.PortfolioShare attribute), 117
- resource_type (troposphere.servicecatalog.TagOption attribute), 117
- resource_type (troposphere.servicecatalog.TagOptionAssociation attribute), 117
- resource_type (troposphere.servicediscovery.HttpNamespace attribute), 118
- resource_type (troposphere.servicediscovery.Instance attribute), 118
- resource_type (troposphere.servicediscovery.PrivateDnsNameSpace attribute), 118
- resource_type (troposphere.servicediscovery.PublicDnsNameSpace attribute), 118
- resource_type (troposphere.servicediscovery.Service attribute), 118
- resource_type (troposphere.ses.ConfigurationSet attribute), 119
- resource_type (troposphere.ses.ConfigurationSetEventDestination attribute), 119
- resource_type (troposphere.ses.ReceiptFilter attribute), 119
- resource_type (troposphere.ses.ReceiptRule attribute), 120
- resource_type (troposphere.ses.ReceiptRuleSet attribute), 120
- resource_type (troposphere.ses.Template attribute), 120
- resource_type (troposphere.sns.SubscriptionResource attribute), 120
- resource_type (troposphere.sns.Topic attribute), 120
- resource_type (troposphere.sns.TopicPolicy attribute), 121
- resource_type (troposphere.sqs.Queue attribute), 121
- resource_type (troposphere.sqs.QueuePolicy attribute), 121
- resource_type (troposphere.ssm.Association attribute), 121
- resource_type (troposphere.ssm.Document attribute), 121
- resource_type (troposphere.ssm.MaintenanceWindow attribute), 121
- resource_type (troposphere.ssm.MaintenanceWindowTarget attribute), 122
- resource_type (troposphere.ssm.MaintenanceWindowTask attribute), 122
- resource_type (troposphere.ssm.Parameter attribute), 122
- resource_type (troposphere.ssm.PatchBaseline attribute), 122
- resource_type (troposphere.ssm.ResourceDataSync attribute), 123
- resource_type (troposphere.stepfunctions.Activity attribute), 123
- resource_type (troposphere.stepfunctions.StateMachine attribute), 123
- resource_type (troposphere.waf.ByteMatchSet attribute), 125
- resource_type (troposphere.waf.IPSet attribute), 126
- resource_type (troposphere.waf.Rule attribute), 126
- resource_type (troposphere.waf.SizeConstraintSet attribute), 126
- resource_type (troposphere.waf.SqlInjectionMatchSet attribute), 126
- resource_type (troposphere.waf.WebACL attribute), 126
- resource_type (troposphere.waf.XssMatchSet attribute), 127
- resource_type (troposphere.wafregional.ByteMatchSet attribute), 127
- resource_type (troposphere.wafregional.IPSet attribute), 127
- resource_type (troposphere.wafregional.Rule attribute), 127
- resource_type (troposphere.wafregional.SizeConstraintSet attribute), 128
- resource_type (troposphere.wafregional.SqlInjectionMatchSet attribute), 128
- resource_type (troposphere.wafregional.WebACL attribute), 128
- resource_type (troposphere.wafregional.WebACLAssociation attribute), 128
- resource_type (troposphere.wafregional.XssMatchSet attribute), 128
- resource_type (troposphere.workspaces.Workspace attribute), 128
- ResourceConfiguration (class in troposphere.iotanalytics), 98
- ResourceDataSync (class in troposphere.ssm), 122
- ResourceGroup (class in troposphere.inspector), 94
- ResourcePolicy (class in troposphere.secretsmanager), 112
- ResourceSignal (class in troposphere.policies), 103
- ResourcesVpcConfig (class in troposphere.eks), 78
- ResourceTypeNotDefined, 123
- ResourceTypeNotFound, 123
- RestApi (class in troposphere.apigateway), 34
- Restrictions (class in troposphere.cloudfront), 51
- RetainRule (class in troposphere.dlm), 63
- RetentionPeriod (class in troposphere.iotanalytics), 98
- RetryOptions (class in troposphere.firehose), 88

- RetryStrategy (class in troposphere.batch), 47
 - Revision (class in troposphere.codedeploy), 56
 - Role (class in troposphere.iam), 93
 - RoleMapping (class in troposphere.cognito), 59
 - RotationRules (class in troposphere.secretsmanager), 112
 - RotationSchedule (class in troposphere.secretsmanager), 112
 - Route (class in troposphere.apigatewayv2), 36
 - Route (class in troposphere.ec2), 70
 - RouteResponse (class in troposphere.apigatewayv2), 36
 - RouteSettings (class in troposphere.apigatewayv2), 36
 - RouteTable (class in troposphere.ec2), 70
 - RoutingRule (class in troposphere.s3), 110
 - RoutingRuleCondition (class in troposphere.s3), 110
 - Rule (class in troposphere.events), 86
 - Rule (class in troposphere.ses), 120
 - Rule (class in troposphere.ssm), 123
 - Rule (class in troposphere.waf), 126
 - Rule (class in troposphere.wafregional), 127
 - RuleGroup (class in troposphere.ssm), 123
 - Rules (class in troposphere.s3), 110
 - Rules (class in troposphere.waf), 126
 - Rules (class in troposphere.wafregional), 127
 - RulesConfiguration (class in troposphere.cognito), 60
 - RunCommandParameters (class in troposphere.events), 86
 - RunCommandTarget (class in troposphere.events), 86
- S**
- s3_backup_mode_elastic_search_validator() (in module troposphere.firehose), 88
 - s3_backup_mode_extended_s3_validator() (in module troposphere.firehose), 88
 - s3_bucket_name() (in module troposphere.validators), 125
 - s3_transfer_acceleration_status() (in module troposphere.validators), 125
 - S3Action (class in troposphere.iot), 95
 - S3Action (class in troposphere.ses), 120
 - S3Configuration (class in troposphere.firehose), 88
 - S3DestinationConfiguration (class in troposphere.firehose), 88
 - S3Event (class in troposphere.serverless), 115
 - S3Key (class in troposphere.s3), 110
 - S3Location (class in troposphere.apigateway), 34
 - S3Location (class in troposphere.codedeploy), 56
 - S3Location (class in troposphere.serverless), 115
 - S3Logs (class in troposphere.codebuild), 54
 - S3Origin (class in troposphere.cloudfront), 51
 - S3OriginConfig (class in troposphere.cloudfront), 52
 - S3OutputLocation (class in troposphere.ssm), 123
 - S3ReferenceDataSource (class in troposphere.analytics), 32
 - S3Settings (class in troposphere.dms), 64
 - S3Target (class in troposphere.glue), 90
 - scalable_dimension_type() (in module troposphere.validators), 125
 - ScalableTarget (class in troposphere.applicationautoscaling), 37
 - ScalableTargetAction (class in troposphere.applicationautoscaling), 37
 - ScalingAction (class in troposphere.emr), 85
 - ScalingConfiguration (class in troposphere.rds), 104
 - ScalingConstraints (class in troposphere.emr), 85
 - ScalingInstruction (class in troposphere.autoscalingplans), 44
 - ScalingPlan (class in troposphere.autoscalingplans), 44
 - ScalingPolicy (class in troposphere.applicationautoscaling), 37
 - ScalingPolicy (class in troposphere.autoscaling), 43
 - ScalingRule (class in troposphere.emr), 85
 - ScalingTrigger (class in troposphere.emr), 85
 - Schedule (class in troposphere.dlm), 63
 - Schedule (class in troposphere.glue), 90
 - Schedule (class in troposphere.iotanalytics), 98
 - ScheduledAction (class in troposphere.applicationautoscaling), 37
 - ScheduledAction (class in troposphere.autoscaling), 43
 - ScheduleEvent (class in troposphere.serverless), 115
 - SchemaAttribute (class in troposphere.cognito), 60
 - SchemaChangePolicy (class in troposphere.glue), 90
 - Scope (class in troposphere.config), 61
 - scope_validator() (in module troposphere.ecs), 77
 - ScriptBootstrapActionConfig (class in troposphere.emr), 85
 - Secret (class in troposphere.secretsmanager), 112
 - SecretTargetAttachment (class in troposphere.secretsmanager), 112
 - SecurityConfiguration (class in troposphere.emr), 85
 - SecurityGroup (class in troposphere.ec2), 71
 - SecurityGroup (class in troposphere.elasticache), 78
 - SecurityGroup (class in troposphere.openstack.neutron), 28
 - SecurityGroupEgress (class in troposphere.ec2), 71
 - SecurityGroupIngress (class in troposphere.ec2), 71
 - SecurityGroupIngress (class in troposphere.elasticache), 79
 - SecurityGroupRule (class in troposphere.ec2), 71
 - SecurityGroupRule (class in troposphere.openstack.neutron), 28
 - SecurityGroups (class in troposphere.ec2), 71
 - Select (class in troposphere), 131
 - SelectAttributes (class in troposphere.iotanalytics), 98
 - SerdeInfo (class in troposphere.glue), 90
 - Server (class in troposphere.openstack.nova), 29
 - Server (class in troposphere.opsworks), 101
 - ServerSideEncryptionByDefault (class in troposphere.s3), 110

- ServerSideEncryptionRule (class in troposphere.s3), 110
- Service (class in troposphere.ecs), 77
- Service (class in troposphere.servicediscovery), 118
- service_namespace_type() (in module troposphere.validators), 125
- ServiceAccountCredentials (class in troposphere.appstream), 38
- ServiceLinkedRole (class in troposphere.iam), 93
- ServiceRegistry (class in troposphere.ecs), 77
- SessionPersistence (class in troposphere.openstack.neutron), 28
- set_description() (troposphere.Template method), 131
- set_metadata() (troposphere.Template method), 131
- set_parameter_label() (troposphere.Template method), 131
- set_transform() (troposphere.Template method), 131
- set_version() (troposphere.Template method), 131
- Settings (class in troposphere.codepipeline), 58
- setUp() (tests.test_dict.TestDict method), 137
- setUp() (tests.test_userdata.TestUserdata method), 144
- ShutdownEventConfiguration (class in troposphere.opsworks), 102
- simple_helper() (tests.test_emr.TestEMR method), 138
- SimpleAD (class in troposphere.directoryservice), 63
- SimpleScalingPolicyConfiguration (class in troposphere.emr), 85
- SimpleTable (class in troposphere.serverless), 115
- SizeConstraint (class in troposphere.waf), 126
- SizeConstraint (class in troposphere.wafregional), 127
- SizeConstraintSet (class in troposphere.waf), 126
- SizeConstraintSet (class in troposphere.wafregional), 128
- SkewedInfo (class in troposphere.glue), 91
- Skill (class in troposphere.ask), 41
- SkillPackage (class in troposphere.ask), 41
- SmsConfiguration (class in troposphere.cognito), 60
- SnapshotOptions (class in troposphere.elasticsearch), 83
- SnsAction (class in troposphere.iot), 95
- SNSAction (class in troposphere.ses), 120
- SNSEvent (class in troposphere.serverless), 115
- Source (class in troposphere.codebuild), 54
- Source (class in troposphere.config), 61
- Source (class in troposphere.opsworks), 102
- SourceAuth (class in troposphere.codebuild), 54
- SourceBundle (class in troposphere.elasticbeanstalk), 80
- SourceConfiguration (class in troposphere.elasticbeanstalk), 80
- SourceDetails (class in troposphere.config), 61
- SourceSelectionCriteria (class in troposphere.s3), 110
- Spend (class in troposphere.budgets), 48
- Split (class in troposphere), 131
- SplunkDestinationConfiguration (class in troposphere.firehose), 88
- SplunkRetryOptions (class in troposphere.firehose), 88
- SpotFleet (class in troposphere.ec2), 71
- SpotFleetRequestConfigData (class in troposphere.ec2), 71
- SpotFleetTagSpecification (class in troposphere.ec2), 71
- SpotOptions (class in troposphere.ec2), 71
- SpotOptionsRequest (class in troposphere.ec2), 71
- SpotProvisioningSpecification (class in troposphere.emr), 85
- SqlInjectionMatchSet (class in troposphere.waf), 126
- SqlInjectionMatchSet (class in troposphere.wafregional), 128
- SqlInjectionMatchTuples (class in troposphere.waf), 126
- SqlInjectionMatchTuples (class in troposphere.wafregional), 128
- SqsAction (class in troposphere.iot), 95
- SQSEvent (class in troposphere.serverless), 115
- SqsParameters (class in troposphere.events), 87
- SseKmsEncryptedObjects (class in troposphere.s3), 110
- SSESpecification (class in troposphere.dax), 62
- SSESpecification (class in troposphere.dynamodb), 66
- SslConfiguration (class in troposphere.opsworks), 102
- SsmAssociations (class in troposphere.ec2), 72
- Stack (class in troposphere.appstream), 38
- Stack (class in troposphere.cloudformation), 50
- Stack (class in troposphere.opsworks), 102
- StackConfigurationManager (class in troposphere.opsworks), 102
- StackFleetAssociation (class in troposphere.appstream), 38
- StackUserAssociation (class in troposphere.appstream), 39
- Stage (class in troposphere.apigateway), 34
- Stage (class in troposphere.apigatewayv2), 36
- StageCanarySetting (in module troposphere.apigateway), 35
- StageDescription (class in troposphere.apigateway), 35
- StageKey (class in troposphere.apigateway), 35
- Stages (class in troposphere.codepipeline), 58
- starting_position_validator() (in module troposphere.serverless), 115
- StateMachine (class in troposphere.stepfunctions), 123
- statistic_type() (in module troposphere.validators), 125
- status() (in module troposphere.validators), 125
- Step (class in troposphere.emr), 86
- StepAdjustment (class in troposphere.applicationautoscaling), 37
- StepAdjustments (class in troposphere.autoscaling), 43
- StepConfig (class in troposphere.emr), 86
- StepScalingPolicyConfiguration (class in troposphere.applicationautoscaling), 37
- StopAction (class in troposphere.ses), 120
- StorageClassAnalysis (class in troposphere.s3), 110
- StorageConnector (class in troposphere.appstream), 39
- StorageDescriptor (class in troposphere.glue), 91
- Stream (class in troposphere.kinesis), 98

- StreamConsumer (class in troposphere.kinesis), 98
- StreamEncryption (class in troposphere.kinesis), 98
- StreamingDistribution (class in troposphere.cloudfront), 52
- StreamingDistributionConfig (class in troposphere.cloudfront), 52
- StreamSpecification (class in troposphere.dynamodb), 66
- STRING_PROPERTIES (troposphere.Parameter attribute), 130
- StringAttributeConstraints (class in troposphere.cognito), 60
- Sub (class in troposphere), 131
- Subnet (class in troposphere.ec2), 72
- SubnetCidrBlock (class in troposphere.ec2), 72
- SubnetGroup (class in troposphere.dax), 62
- SubnetGroup (class in troposphere.elasticache), 79
- SubnetMapping (class in troposphere.elasticloadbalancingv2), 82
- SubnetNetworkAclAssociation (class in troposphere.ec2), 72
- SubnetRouteTableAssociation (class in troposphere.ec2), 72
- Subscriber (class in troposphere.budgets), 48
- Subscription (class in troposphere.sns), 120
- SubscriptionFilter (class in troposphere.logs), 99
- SubscriptionResource (class in troposphere.sns), 120
- swagger (tests.test_serverless.TestServerless attribute), 142
- ## T
- Table (class in troposphere.dynamodb), 66
- Table (class in troposphere.glue), 91
- table_type_validator() (in module troposphere.glue), 91
- TableInput (class in troposphere.glue), 91
- Tag (class in troposphere.autoscaling), 43
- Tag (class in troposphere.ec2), 72
- TagFilter (class in troposphere.autoscalingplans), 44
- TagFilter (class in troposphere.s3), 110
- TagFilters (class in troposphere.codedeploy), 56
- TagOption (class in troposphere.servicecatalog), 117
- TagOptionAssociation (class in troposphere.servicecatalog), 117
- Tags (class in troposphere), 131
- Tags (class in troposphere.autoscaling), 43
- TagSpecifications (class in troposphere.ec2), 72
- tail() (in module troposphere.utils), 124
- Target (class in troposphere.events), 87
- TargetAddress (class in troposphere.route53), 107
- TargetCapacitySpecificationRequest (class in troposphere.ec2), 72
- TargetDescription (class in troposphere.elasticloadbalancingv2), 82
- TargetGroup (class in troposphere.ec2), 72
- TargetGroup (class in troposphere.elasticloadbalancingv2), 82
- TargetGroupAttribute (class in troposphere.elasticloadbalancingv2), 82
- TargetGroupInfoList (class in troposphere.codedeploy), 56
- Targets (class in troposphere.glue), 91
- Targets (class in troposphere.ssm), 123
- TargetTrackingConfiguration (class in troposphere.autoscaling), 43
- TargetTrackingConfiguration (class in troposphere.autoscalingplans), 44
- TargetTrackingScalingPolicyConfiguration (class in troposphere.applicationautoscaling), 38
- task_type() (in module troposphere.validators), 125
- TaskDefinition (class in troposphere.ecs), 77
- TaskInvocationParameters (class in troposphere.ssm), 123
- Template (class in troposphere), 131
- Template (class in troposphere.ses), 120
- TemplateGenerator (class in troposphere.template_generator), 124
- test_accelerate_configuration_enabled() (tests.test_s3.TestS3AccelerateConfiguration method), 142
- test_accelerate_configuration_invalid_value() (tests.test_s3.TestS3AccelerateConfiguration method), 142
- test_accelerate_configuration_suspended() (tests.test_s3.TestS3AccelerateConfiguration method), 142
- test_activity() (tests.test_stepfunctions.TestStepFunctions method), 143
- test_add_or_get_returns_with_out_adding_duplicate() (tests.test_basic.TestParameter method), 135
- test_allow_container_healthcheck() (tests.test_ecs.TestECS method), 137
- test_allow_placement_strategy_constraint() (tests.test_ecs.TestECS method), 137
- test_allow_port_mapping_protocol() (tests.test_ecs.TestECS method), 137
- test_allow_ref_cluster() (tests.test_ecs.TestECS method), 137
- test_allow_ref_task_role_arn() (tests.test_ecs.TestECS method), 137
- test_allow_scheduling_strategy() (tests.test_ecs.TestECS method), 137
- test_allow_string_cluster() (tests.test_ecs.TestECS method), 137
- test_allow_string_cluster() (tests.test_emr.TestEMR method), 138
- test_allow_string_task_role_arn() (tests.test_ecs.TestECS method), 137
- test_ASTagAddition() (tests.test_tags.TestTags method), 143

test_auto_scaling_creation_policy() (tests.test_policies.TestCreationPolicy method), 140

test_auto_scaling_creation_policy_json() (tests.test_policies.TestCreationPolicy method), 140

test_AutoScalingRollingUpdate_all_defaults() (tests.test_asg.TestAutoScalingGroup method), 133

test_AutoScalingRollingUpdate_validation() (tests.test_asg.TestAutoScalingGroup method), 133

test_awsproperty_invalid_property() (tests.test_basic.TestProperty method), 135

test_az_and_multiaz_funcs() (tests.test_rds.TestRDS method), 141

test_badlist() (tests.test_basic.TestValidators method), 135

test_badmultilist() (tests.test_basic.TestValidators method), 135

test_badproperty() (tests.test_basic.TestBasic method), 134

test_badrequired() (tests.test_basic.TestBasic method), 134

test_badtype() (tests.test_basic.TestBasic method), 134

test_BogusAttribute() (tests.test_basic.TestAttributes method), 134

test_boolean() (tests.test_validators.TestValidators method), 145

test_bucket_accesscontrol() (tests.test_s3.TestBucket method), 142

test_bucket_accesscontrol_bad_string() (tests.test_s3.TestBucket method), 142

test_bucket_accesscontrol_bad_type() (tests.test_s3.TestBucket method), 142

test_bucket_accesscontrol_ref() (tests.test_s3.TestBucket method), 142

test_bucket_template() (tests.test_efs.TestEfsTemplate method), 138

test_bucket_template() (tests.test_s3.TestBucketTemplate method), 142

test_callcorrect() (tests.test_basic.TestValidators method), 135

test_callincorrect() (tests.test_basic.TestValidators method), 135

test_char_escaping() (tests.test_userdata.TestUserdata method), 145

test_check_zip_file() (tests.test_awslambda.TestAWSLambda method), 133

test_compliance_level() (tests.test_validators.TestValidators method), 145

test_CreationPolicy() (tests.test_cloudformation.TestWaitCondition method), 136

test_CreationPolicyWithProps() (tests.test_cloudformation.TestWaitCondition method), 136

test_custom_json() (tests.test_opsworks.TestOpsWorksStack method), 140

test_custom_resource_override() (tests.test_template_generator.TestTemplateGenerator method), 144

test_custom_resource_type() (tests.test_template_generator.TestTemplateGenerator method), 144

test_dashboard() (tests.test_cloudwatch.TestModel method), 136

test_depends_on_helper_with_resource() (tests.test_basic.TestBasic method), 134

test_depends_on_helper_with_string() (tests.test_basic.TestBasic method), 134

test_description() (tests.test_template.TestInitArguments method), 143

test_description_default() (tests.test_template.TestInitArguments method), 143

test_DLQ() (tests.test_serverless.TestServerless method), 142

test_docker_volume_configuration() (tests.test_ecs.TestECS method), 137

test_elb_name() (tests.test_validators.TestValidators method), 145

test_empty_awsproperty_outputs_empty_object() (tests.test_basic.TestOutput method), 134

test_empty_file() (tests.test_userdata.TestUserdata method), 145

test_encoding() (tests.test_validators.TestValidators method), 145

test_environment_variable_invalid_name() (tests.test_awslambda.TestAWSLambda method), 133

test_environment_variable_not_reserved() (tests.test_awslambda.TestAWSLambda method), 133

test_environment_variable_reserved() (tests.test_awslambda.TestAWSLambda method), 133

test_eq() (tests.test_template.TestEquality method), 143

test_exactly_one_code() (tests.test_serverless.TestServerless method), 142

test_example() (tests.test_examples.TestExamples method), 139

test_exception() (tests.test_basic.TestValidators method), 136

test_exclusive() (tests.test_asg.TestAutoScalingGroup method), 133

test_exclusive() (tests.test_awslambda.TestAWSLambda method), 133

test_extraattribute() (tests.test_basic.TestBasic method),

- 134
- test_fail_az_and_multiaz() (tests.test_rds.TestRDS method), 141
- test_fargate_launch_type() (tests.test_ecs.TestECS method), 137
- test_fixed_response_action() (tests.test_elasticloadbalancerv2.TestListenerActions method), 138
- test_fixed_response_config_one_of() (tests.test_elasticloadbalancerv2.TestListenerActions method), 138
- test_fixed_response_config_only_with_fixed_response() (tests.test_elasticloadbalancerv2.TestListenerActions method), 138
- test_fixed_response_requires_fixed_response_config() (tests.test_elasticloadbalancerv2.TestListenerActions method), 138
- test_Formats() (tests.test_tags.TestTags method), 143
- test_forward_action() (tests.test_elasticloadbalancerv2.TestListenerActions method), 138
- test_forward_action_requires_target_arn() (tests.test_elasticloadbalancerv2.TestListenerActions method), 138
- test_get_or_add_adds() (tests.test_basic.TestParameter method), 135
- test_getcidr() (tests.test_basic.TestCidr method), 134
- test_getcidr_withsizemask() (tests.test_basic.TestCidr method), 134
- test_goodrequired() (tests.test_basic.TestBasic method), 134
- test_guarddduty_detector() (tests.test_guarddduty.TestGuardDuty method), 139
- test_guarddduty_ipset() (tests.test_guarddduty.TestGuardDuty method), 139
- test_guarddduty_threatintelset() (tests.test_guarddduty.TestGuardDuty method), 139
- test_hash() (tests.test_template.TestEquality method), 143
- test_healthy_interval_ok() (tests.test_basic.TestHealthCheck method), 134
- test_healthy_interval_too_low() (tests.test_basic.TestHealthCheck method), 134
- test_helperfn_as_AutoScalingRollingUpdate() (tests.test_asg.TestAutoScalingGroup method), 133
- test_helperfn_as_updatepolicy() (tests.test_asg.TestAutoScalingGroup method), 133
- test_helperfun() (tests.test_basic.TestValidators method), 136
- test_iam_group_name() (tests.test_validators.TestValidators method), 145
- test_iam_names() (tests.test_validators.TestValidators method), 145
- test_iam_path() (tests.test_validators.TestValidators method), 145
- test_iam_role_name() (tests.test_validators.TestValidators method), 145
- test_iam_user_name() (tests.test_validators.TestValidators method), 145
- test_instanceid() (tests.test_asg.TestAutoScalingGroup method), 133
- test_integer() (tests.test_validators.TestValidators method), 145
- test_integer_range() (tests.test_validators.TestValidators method), 145
- test_invalid_parameter_property_in_template() (tests.test_basic.TestParameter method), 135
- test_invalid_pausetime() (tests.test_policies.TestCreationPolicy method), 140
- test_invalid_pausetime() (tests.test_policies.TestUpdatePolicy method), 141
- test_invalid_SourceDetails_MaximumExecutionFrequency() (tests.test_config.TestConfig method), 137
- test_invalid_sub_property() (tests.test_dict.TestDict method), 137
- test_invalid_subproperty_definition() (tests.test_dict.TestDict method), 137
- test_invalid_toplevel_property() (tests.test_dict.TestDict method), 137
- test_io1_storage_type_and_iops() (tests.test_rds.TestRDS method), 141
- test_it_allows_an_rds_instance_created_from_a_snapshot() (tests.test_rds.TestRDS method), 141
- test_it_allows_an_rds_instance_with_iops() (tests.test_rds.TestRDS method), 141
- test_it_allows_an_rds_instance_with_master_username_and_password() (tests.test_rds.TestRDS method), 141
- test_it_allows_an_rds_replica() (tests.test_rds.TestRDS method), 141
- test_it_rds_instances_require_either_a_snapshot_or_credentials() (tests.test_rds.TestRDS method), 141
- test_it_rds_instances_require_encryption_if_kms_key_provided() (tests.test_rds.TestRDS method), 141
- test_join() (tests.test_basic.TestJoin method), 134
- test_json() (tests.test_policies.TestCreationPolicy method), 140
- test_json() (tests.test_policies.TestUpdatePolicy method), 141
- test_launchconfigurationname() (tests.test_asg.TestAutoScalingGroup method), 133
- test_layer_version() (tests.test_serverless.TestServerless

- method), 142
- test_linux_environment() (tests.test_codebuild.TestCodeBuild method), 136
- test_list() (tests.test_basic.TestValidators method), 136
- test_log_destination() (tests.test_logs.TestLogs method), 140
- test_loggroup_deletionpolicy_is_preserved() (tests.test_logs.TestLogs method), 140
- test_loggroup_retention() (tests.test_logs.TestLogs method), 140
- test_macro_resource() (tests.test_template_generator.TestTemplateGenerator method), 144
- test_max_mappings() (tests.test_template.TestValidate method), 144
- test_max_outputs() (tests.test_template.TestValidate method), 144
- test_max_parameters() (tests.test_template.TestValidate method), 144
- test_max_resources() (tests.test_template.TestValidate method), 144
- test_metadata() (tests.test_template.TestInitArguments method), 144
- test_metadata_default() (tests.test_template.TestInitArguments method), 144
- test_mininstances() (tests.test_policies.TestUpdatePolicy method), 141
- test_mininstances_maxsize_is_ref() (tests.test_policies.TestUpdatePolicy method), 141
- test_mininstances_mininstancesinservice_is_ref() (tests.test_policies.TestUpdatePolicy method), 141
- test_multilist() (tests.test_basic.TestValidators method), 136
- test_mutualexclusion() (tests.test_basic.TestValidators method), 136
- test_mutually_exclusive() (tests.test_validators.TestValidators method), 145
- test_mutually_exclusive_novalue() (tests.test_validators.TestValidators method), 145
- test_ne() (tests.test_template.TestEquality method), 143
- test_network_port() (tests.test_validators.TestValidators method), 145
- test_network_port_ref() (tests.test_validators.TestValidators method), 145
- test_no_nested_name() (tests.test_template_generator.TestTemplateGenerator method), 144
- test_no_required() (tests.test_opsworks.TestOpsWorksStack method), 140
- test_no_snapshot_or_engine() (tests.test_rds.TestRDS method), 141
- test_no_validation_method() (tests.test_basic.TestValidation method), 135
- test_none() (tests.test_asg.TestAutoScalingGroup method), 133
- test_nonexistent_file() (tests.test_userdata.TestUserData method), 145
- test_noproperty() (tests.test_basic.TestOutput method), 135
- test_noproperty() (tests.test_basic.TestParameter method), 135
- test_noproperty() (tests.test_basic.TestProperty method), 135
- test_nosubnet() (tests.test_opsworks.TestOpsWorksStack method), 140
- test_notification_event() (tests.test_validators.TestValidators method), 145
- test_notification_type() (tests.test_validators.TestValidators method), 145
- test_novalidation() (tests.test_basic.TestValidation method), 135
- test_one_line_file() (tests.test_userdata.TestUserData method), 145
- test_one_of() (tests.test_validators.TestValidators method), 145
- test_operating_system() (tests.test_validators.TestValidators method), 145
- test_optional_auto_publish_alias() (tests.test_serverless.TestServerless method), 142
- test_optional_deployment_preference() (tests.test_serverless.TestServerless method), 142
- test_optiongroup() (tests.test_rds.TestRDS method), 141
- test_output() (tests.test_basic.TestDuplicate method), 134
- test_packaging() (tests.test_serverless.TestServerless method), 142
- test_parameter() (tests.test_basic.TestDuplicate method), 134
- test_parameter_group() (tests.test_template.TestAwsInterface method), 143
- test_parameter_label() (tests.test_template.TestAwsInterface method), 143
- test_parameter_label_replace() (tests.test_template.TestAwsInterface method), 143
- test_pickle_ok() (tests.test_basic.TestBasic method), 134
- test_placement_template() (tests.test_iamclick.TestPlacementTemplate method), 140
- test_policy_document() (tests.test_serverless.TestServerless method), 142
- test_port_mapping_does_not_require_protocol() (tests.test_ecs.TestECS method), 137

- test_positive_integer() (tests.test_validators.TestValidators method), 145
- test_property_default() (tests.test_basic.TestParameter method), 135
- test_property_validator() (tests.test_basic.TestParameter method), 135
- test_QueueName() (tests.test_sqs.TestQueue method), 143
- test_redirect_action() (tests.test_elasticloadbalancingv2.TestListenerAction method), 138
- test_redirect_action_config_one_of() (tests.test_elasticloadbalancingv2.TestListenerActions method), 138
- test_redirect_action_requires_redirect_config() (tests.test_elasticloadbalancingv2.TestListenerActions method), 138
- test_redirect_config_only_with_redirect() (tests.test_elasticloadbalancingv2.TestListenerActions method), 138
- test_ref() (tests.test_basic.TestName method), 134
- test_ref() (tests.test_basic.TestRef method), 135
- test_ref_can_be_requested() (tests.test_parameters.TestInitArguments method), 140
- test_ref_eq() (tests.test_basic.TestRef method), 135
- test_ref_hash() (tests.test_basic.TestRef method), 135
- test_replica_settings_are_inherited() (tests.test_rds.TestRDS method), 141
- test_required() (tests.test_opsworks.TestOpsWorksStack method), 140
- test_required_api_both() (tests.test_serverless.TestServerless method), 142
- test_required_api_definitionbody() (tests.test_serverless.TestServerless method), 142
- test_required_api_definitionuri() (tests.test_serverless.TestServerless method), 142
- test_required_function() (tests.test_serverless.TestServerless method), 142
- test_RequiredProps() (tests.test_cloudformation.TestWaitCondition method), 136
- test_resolver() (tests.test_appsync.TestAppsyncResolver method), 133
- test_resolver_kind_bad_value() (tests.test_appsync.TestAppsyncResolver method), 133
- test_resource() (tests.test_basic.TestDuplicate method), 134
- test_resource_depends_on() (tests.test_basic.TestBasic method), 134
- test_resource_depends_on_attr() (tests.test_basic.TestBasic method), 134
- test_resource_depends_on_list() (tests.test_basic.TestBasic method), 134
- test_resource_type_not_defined() (tests.test_template_generator.TestTemplateGenerator method), 144
- test_response_type() (tests.test_apigateway.TestGatewayResponse method), 132
- test_response_type() (tests.test_apigatewayv2.TestAuthorizer method), 132
- test_response_type() (tests.test_apigatewayv2.TestIntegrationResponse method), 132
- test_s3_bucket() (tests.test_yaml.TestYAML method), 146
- test_s3_bucket_accelerate_configuration() (tests.test_s3.TestS3AccelerateConfiguration method), 142
- test_s3_bucket_name() (tests.test_validators.TestValidators method), 145
- test_s3_filter() (tests.test_serverless.TestServerless method), 142
- test_s3_location() (tests.test_serverless.TestServerless method), 142
- test_schema() (tests.test_apigateway.TestModel method), 132
- test_schema() (tests.test_apigatewayv2.TestModel method), 132
- test_scope_validator() (tests.test_ecs.TestECSValidators method), 138
- test_securitygroup_egress() (tests.test_ec2.TestEC2 method), 137
- test_simple() (tests.test_userdata.TestUserdata method), 145
- test_simple_table() (tests.test_serverless.TestServerless method), 142
- test_SimpleScalingPolicyConfiguration() (tests.test_emr.TestEMR method), 138
- test_size_if() (tests.test_asg.TestAutoScalingGroup method), 133
- test_snapshot() (tests.test_rds.TestRDS method), 141
- test_snapshot_and_engine() (tests.test_rds.TestRDS method), 141
- test_source_codepipeline() (tests.test_codebuild.TestCodeBuild method), 136
- test_SourceDetails() (tests.test_config.TestConfig method), 137
- test_split() (tests.test_basic.TestSplit method), 135
- test_stack() (tests.test_opsworks.TestOpsWorksStack method), 140
- test_statemachine() (tests.test_stepfunctions.TestStepFunctions method), 143
- test_statemachine_missing_parameter() (tests.test_stepfunctions.TestStepFunctions method), 143
- test_status() (tests.test_validators.TestValidators method), 134

- 145
- test_storage_to_iops_ratio() (tests.test_rds.TestRDS method), 141
- test_sub_property_helper_fn() (tests.test_dict.TestDict method), 137
- test_sub_with_vars_mix() (tests.test_basic.TestSub method), 135
- test_sub_with_vars_not_unpackaged() (tests.test_basic.TestSub method), 135
- test_sub_with_vars_unpackaged() (tests.test_basic.TestSub method), 135
- test_sub_without_vars() (tests.test_basic.TestSub method), 135
- test_TagAddition() (tests.test_tags.TestTags method), 143
- test_tags() (tests.test_serverless.TestServerless method), 143
- test_tags_from_dict() (tests.test_dict.TestDict method), 137
- test_target_arn_only_forward() (tests.test_elasticloadbalancerv2.TestListenerActions method), 138
- test_task_role_arn_is_optional() (tests.test_ecs.TestECS method), 138
- test_task_type() (tests.test_validators.TestValidators method), 145
- test_template_generator() (tests.test_examples_template_generator.TestTemplateGenerator method), 139
- test_tg_healthcheck_port() (tests.test_validators.TestValidators method), 145
- test_tg_healthcheck_port_ref() (tests.test_validators.TestValidators method), 145
- test_there_should_not_be_any_awsobject_with_int_in_properties() (tests.test_int_type.TestIntTypeShouldNotBeUsed method), 139
- test_title_max_length() (tests.test_parameters.TestInitArguments method), 140
- test_toplevel_helper_fn() (tests.test_dict.TestDict method), 137
- test_transform() (tests.test_template.TestInitArguments method), 144
- test_trigger() (tests.test_codecommit.TestCodeCommit method), 136
- test_tuples() (tests.test_basic.TestValidators method), 136
- test_unknown_resource_type() (tests.test_template_generator.TestTemplateGenerator method), 144
- test_Unsortable() (tests.test_tags.TestTags method), 143
- test_UpdateReplacePolicy() (tests.test_basic.TestAttributes method), 134
- test_valid_data() (tests.test_dict.TestDict method), 137
- test_validate_backup_retention_period() (tests.test_rds.TestRDSValidators method), 141
- test_validate_backup_window() (tests.test_rds.TestRDSValidators method), 141
- test_validate_capacity() (tests.test_rds.TestRDSValidators method), 141
- test_validate_engine() (tests.test_rds.TestRDSValidators method), 141
- test_validate_engine_mode() (tests.test_rds.TestRDSValidators method), 141
- test_validate_iops() (tests.test_rds.TestRDSValidators method), 141
- test_validate_license_model() (tests.test_rds.TestRDSValidators method), 141
- test_validate_maintenance_window() (tests.test_rds.TestRDSValidators method), 141
- test_validate_storage_type() (tests.test_rds.TestRDSValidators method), 142
- test_validateProvisionedThroughputInMibps() (tests.test_efs.TestEfs method), 138
- test_validateProvisionedThroughputMode() (tests.test_efs.TestEfs method), 138
- test_validation() (tests.test_basic.TestValidation method), 135
- test_validData() (tests.test_efs.TestEfs method), 138
- test_windows_environment() (tests.test_codebuild.TestCodeBuild method), 136
- test_works() (tests.test_policies.TestCreationPolicy method), 140
- test_works() (tests.test_policies.TestUpdatePolicy method), 141
- test_yaml_long_form() (tests.test_yaml.TestYAML method), 146
- test_zip_file() (tests.test_awslambda.TestAWSLambda method), 133
- TestAppsyncResolver (class in tests.test_appsync), 133
- TestAttributes (class in tests.test_basic), 134
- TestAuthorizer (class in tests.test_apigatewayv2), 132
- TestAutoScalingGroup (class in tests.test_asg), 133
- TestAwsInterface (class in tests.test_template), 143
- TestAWSLambda (class in tests.test_awslambda), 133
- TestBasic (class in tests.test_basic), 134
- TestBucket (class in tests.test_s3), 142
- TestBucketTemplate (class in tests.test_s3), 142
- TestCidr (class in tests.test_basic), 134
- TestCodeBuild (class in tests.test_codebuild), 136
- TestCodeCommit (class in tests.test_codecommit), 136

- TestConfig (class in tests.test_config), 137
- TestCreationPolicy (class in tests.test_policies), 140
- TestDict (class in tests.test_dict), 137
- TestDuplicate (class in tests.test_basic), 134
- TestEC2 (class in tests.test_ec2), 137
- TestECS (class in tests.test_ecs), 137
- TestECSValidators (class in tests.test_ecs), 138
- TestEfs (class in tests.test_efs), 138
- TestEfsTemplate (class in tests.test_efs), 138
- TestEMR (class in tests.test_emr), 138
- TestEquality (class in tests.test_template), 143
- TestExamples (class in tests.test_examples), 139
- TestGatewayResponse (class in tests.test_apigateway), 132
- TestGuardDuty (class in tests.test_guardduty), 139
- TestHealthCheck (class in tests.test_basic), 134
- TestInitArguments (class in tests.test_parameters), 140
- TestInitArguments (class in tests.test_template), 143
- TestIntegrationResponse (class in tests.test_apigatewayv2), 132
- TestIntTypeShouldNotBeUsed (class in tests.test_int_type), 139
- TestJoin (class in tests.test_basic), 134
- TestListenerActions (class in tests.test_elasticloadbalancerv2), 138
- TestLogs (class in tests.test_logs), 140
- TestModel (class in tests.test_apigateway), 132
- TestModel (class in tests.test_apigatewayv2), 132
- TestModel (class in tests.test_cloudwatch), 136
- TestName (class in tests.test_basic), 134
- TestOpsWorksStack (class in tests.test_opsworks), 140
- TestOutput (class in tests.test_basic), 134
- TestParameter (class in tests.test_basic), 135
- TestPlacementTemplate (class in tests.test_iot1click), 140
- TestProperty (class in tests.test_basic), 135
- TestQueue (class in tests.test_sqs), 143
- TestRDS (class in tests.test_rds), 141
- TestRDSValidators (class in tests.test_rds), 141
- TestRef (class in tests.test_basic), 135
- tests (module), 146
- tests.test_apigateway (module), 132
- tests.test_apigatewayv2 (module), 132
- tests.test_appsync (module), 133
- tests.test_asg (module), 133
- tests.test_awslambda (module), 133
- tests.test_basic (module), 133
- tests.test_cloudformation (module), 136
- tests.test_cloudwatch (module), 136
- tests.test_codebuild (module), 136
- tests.test_codecommit (module), 136
- tests.test_config (module), 137
- tests.test_dict (module), 137
- tests.test_ec2 (module), 137
- tests.test_ecs (module), 137
- tests.test_efs (module), 138
- tests.test_elasticloadbalancerv2 (module), 138
- tests.test_emr (module), 138
- tests.test_examples (module), 139
- tests.test_examples_template_generator (module), 139
- tests.test_guardduty (module), 139
- tests.test_int_type (module), 139
- tests.test_iot1click (module), 140
- tests.test_logs (module), 140
- tests.test_opsworks (module), 140
- tests.test_parameters (module), 140
- tests.test_policies (module), 140
- tests.test_rds (module), 141
- tests.test_s3 (module), 142
- tests.test_serverless (module), 142
- tests.test_sqs (module), 143
- tests.test_stepfunctions (module), 143
- tests.test_tags (module), 143
- tests.test_template (module), 143
- tests.test_template_generator (module), 144
- tests.test_userdata (module), 144
- tests.test_validators (module), 145
- tests.test_yaml (module), 146
- TestS3AccelerateConfiguration (class in tests.test_s3), 142
- TestServerless (class in tests.test_serverless), 142
- TestSplit (class in tests.test_basic), 135
- TestStepFunctions (class in tests.test_stepfunctions), 143
- TestSub (class in tests.test_basic), 135
- TestTags (class in tests.test_tags), 143
- TestTemplateGenerator (class in tests.test_examples_template_generator), 139
- TestTemplateGenerator (class in tests.test_template_generator), 144
- TestUpdatePolicy (class in tests.test_policies), 141
- TestUserData (class in tests.test_userdata), 144
- TestValidate (class in tests.test_template), 144
- TestValidation (class in tests.test_basic), 135
- TestValidators (class in tests.test_basic), 135
- TestValidators (class in tests.test_validators), 145
- TestWaitCondition (class in tests.test_cloudformation), 136
- TestYAML (class in tests.test_yaml), 146
- tg_healthcheck_port() (in module troposphere.validators), 125
- Thing (class in troposphere.iot), 95
- ThingPrincipalAttachment (class in troposphere.iot), 95
- ThreatIntelSet (class in troposphere.guardduty), 92
- ThrottleSettings (class in troposphere.apigateway), 35
- throughput_mode_validator() (in module troposphere.efs), 78
- Tier (class in troposphere.elasticbeanstalk), 80

TimeBasedAutoScaling (class in troposphere.opsworks), 102

Timeout (class in troposphere.batch), 47

TimePeriod (class in troposphere.budgets), 48

TimeToLiveSpecification (class in troposphere.dynamodb), 66

to_dict() (troposphere.autoscaling.Tags method), 43

to_dict() (troposphere.AWSHelperFn method), 129

to_dict() (troposphere.BaseAWSObject method), 129

to_dict() (troposphere.cloudformation.Metadata method), 50

to_dict() (troposphere.GenericHelperFn method), 130

to_dict() (troposphere.Tags method), 131

to_dict() (troposphere.Template method), 131

to_json() (troposphere.Template method), 132

to_yaml() (troposphere.Template method), 132

Topic (class in troposphere.sns), 120

TopicConfigurations (class in troposphere.s3), 110

TopicPolicy (class in troposphere.sns), 121

TopicRule (class in troposphere.iot), 95

TopicRulePayload (class in troposphere.iot), 95

TracingConfig (class in troposphere.awslambda), 46

Trail (class in troposphere.cloudtrail), 52

TransitGateway (class in troposphere.ec2), 72

TransitGatewayAttachment (class in troposphere.ec2), 72

TransitGatewayRoute (class in troposphere.ec2), 73

TransitGatewayRouteTable (class in troposphere.ec2), 73

TransitGatewayRouteTableAssociation (class in troposphere.ec2), 73

TransitGatewayRouteTablePropagation (class in troposphere.ec2), 73

Trigger (class in troposphere.autoscaling), 43

Trigger (class in troposphere.codecommit), 54

Trigger (class in troposphere.glue), 91

Trigger (class in troposphere.iotanalytics), 98

trigger_type_validator() (in module troposphere.glue), 91

TriggerConfig (class in troposphere.codedeploy), 57

TriggeringDataset (class in troposphere.iotanalytics), 98

troposphere (module), 129

troposphere.amazonmq (module), 29

troposphere.analytics (module), 30

troposphere.apigateway (module), 32

troposphere.apigatewayv2 (module), 35

troposphere.applicationautoscaling (module), 37

troposphere.appstream (module), 38

troposphere.appsync (module), 39

troposphere.ask (module), 41

troposphere.athena (module), 41

troposphere.autoscaling (module), 41

troposphere.autoscalingplans (module), 43

troposphere.awslambda (module), 45

troposphere.batch (module), 46

troposphere.budgets (module), 47

troposphere.certificatemanager (module), 48

troposphere.cloud9 (module), 48

troposphere.cloudformation (module), 49

troposphere.cloudfront (module), 50

troposphere.cloudtrail (module), 52

troposphere.cloudwatch (module), 52

troposphere.codebuild (module), 53

troposphere.codecommit (module), 54

troposphere.codedeploy (module), 55

troposphere.codepipeline (module), 57

troposphere.cognito (module), 58

troposphere.config (module), 60

troposphere.constants (module), 62

troposphere.datapipeline (module), 62

troposphere.dax (module), 62

troposphere.directoryservice (module), 63

troposphere.dlm (module), 63

troposphere.dms (module), 64

troposphere.docdb (module), 65

troposphere.dynamodb (module), 65

troposphere.dynamodb2 (module), 66

troposphere.ec2 (module), 66

troposphere.ecr (module), 75

troposphere.ecs (module), 75

troposphere.efs (module), 77

troposphere.eks (module), 78

troposphere.elasticache (module), 78

troposphere.elasticbeanstalk (module), 79

troposphere.elasticloadbalancing (module), 80

troposphere.elasticloadbalancingv2 (module), 81

troposphere.elasticsearch (module), 83

troposphere.emr (module), 83

troposphere.events (module), 86

troposphere.firehose (module), 87

troposphere.glue (module), 88

troposphere.guardduty (module), 91

troposphere.helpers (module), 26

troposphere.helpers.userdata (module), 26

troposphere.iam (module), 92

troposphere.inspector (module), 93

troposphere.iot (module), 94

troposphere.iot1click (module), 96

troposphere.iotanalytics (module), 96

troposphere.kinesis (module), 98

troposphere.kms (module), 99

troposphere.logs (module), 99

troposphere.neptune (module), 99

troposphere.openstack (module), 29

troposphere.openstack.heat (module), 26

troposphere.openstack.neutron (module), 26

troposphere.openstack.nova (module), 28

troposphere.opsworks (module), 100

troposphere.policies (module), 102

troposphere.rds (module), 103

troposphere.redshift (module), 105

troposphere.route53 (module), 106
 troposphere.s3 (module), 107
 troposphere.sagemaker (module), 111
 troposphere.sdb (module), 112
 troposphere.secretsmanager (module), 112
 troposphere.serverless (module), 113
 troposphere.servicecatalog (module), 116
 troposphere.servicediscovery (module), 117
 troposphere.ses (module), 118
 troposphere.sns (module), 120
 troposphere.sqs (module), 121
 troposphere.ssm (module), 121
 troposphere.stepfunctions (module), 123
 troposphere.template_generator (module), 123
 troposphere.utils (module), 124
 troposphere.validators (module), 124
 troposphere.waf (module), 125
 troposphere.wafregional (module), 127
 troposphere.workspaces (module), 128
 TrustedSigners (class in troposphere.cloudfront), 52
 type (tests.test_basic.FakeAWSObject attribute), 133

U

Ulimit (class in troposphere.batch), 47
 Ulimit (class in troposphere.ecs), 77
 update_behavior_validator() (in module troposphere.glue), 91
 UpdatePolicy (class in troposphere), 132
 UpdatePolicy (class in troposphere.policies), 103
 UsagePlan (class in troposphere.apigateway), 35
 UsagePlanKey (class in troposphere.apigateway), 35
 User (class in troposphere.amazonmq), 30
 User (class in troposphere.appstream), 39
 User (class in troposphere.iam), 93
 UserPool (class in troposphere.cognito), 60
 UserPoolClient (class in troposphere.cognito), 60
 UserPoolConfig (class in troposphere.appsync), 41
 UserPoolGroup (class in troposphere.cognito), 60
 UserPoolUser (class in troposphere.cognito), 60
 UserPoolUserToGroupAttachment (class in troposphere.cognito), 60
 UserProfile (class in troposphere.opsworks), 102
 UserSetting (class in troposphere.appstream), 39
 UserToGroupAddition (class in troposphere.iam), 93

V

validate() (tests.test_basic.FakeAWSObject method), 133
 validate() (troposphere.apigateway.Model method), 34
 validate() (troposphere.apigateway.StageDescription method), 35
 validate() (troposphere.apigatewayv2.Model method), 36
 validate() (troposphere.autoscaling.AutoScalingGroup method), 41

validate() (troposphere.autoscaling.LaunchTemplateSpecification method), 42
 validate() (troposphere.autoscaling.Metadata method), 42
 validate() (troposphere.awslambda.Code method), 45
 validate() (troposphere.BaseAWSObject method), 129
 validate() (troposphere.batch.LaunchTemplateSpecification method), 47
 validate() (troposphere.cloudformation.Authentication method), 49
 validate() (troposphere.cloudformation.Init method), 49
 validate() (troposphere.cloudformation.InitConfigSets method), 49
 validate() (troposphere.cloudformation.InitFiles method), 49
 validate() (troposphere.cloudformation.InitServices method), 50
 validate() (troposphere.cloudformation.WaitCondition method), 50
 validate() (troposphere.cloudwatch.Alarm method), 52
 validate() (troposphere.cloudwatch.Dashboard method), 53
 validate() (troposphere.codebuild.Artifacts method), 53
 validate() (troposphere.codebuild.Environment method), 53
 validate() (troposphere.codebuild.EnvironmentVariable method), 53
 validate() (troposphere.codebuild.ProjectCache method), 54
 validate() (troposphere.codebuild.Source method), 54
 validate() (troposphere.codebuild.SourceAuth method), 54
 validate() (troposphere.codecommit.Trigger method), 55
 validate() (troposphere.codedeploy.DeploymentGroup method), 55
 validate() (troposphere.codedeploy.LoadBalancerInfo method), 56
 validate() (troposphere.config.SourceDetails method), 61
 validate() (troposphere.dynamodb.Table method), 66
 validate() (troposphere.ec2.NetworkAclEntry method), 69
 validate() (troposphere.ec2.Route method), 70
 validate() (troposphere.ec2.SecurityGroupEgress method), 71
 validate() (troposphere.ec2.SecurityGroupIngress method), 71
 validate() (troposphere.ec2.SpotFleetRequestConfigData method), 71
 validate() (troposphere.ec2.Subnet method), 72
 validate() (troposphere.elasticache.CacheCluster method), 78
 validate() (troposphere.elasticache.ReplicationGroup method), 78
 validate() (troposphere.elasticloadbalancingv2.Action method), 81

- validate() (troposphere.elasticloadbalancingv2.FixedResponseAction method), 81
- validate() (troposphere.elasticloadbalancingv2.LoadBalancer method), 82
- validate() (troposphere.elasticloadbalancingv2.RedirectConfig method), 82
- validate() (troposphere.elasticsearch.EBSOptions method), 83
- validate() (troposphere.emr.SimpleScalingPolicyConfiguration method), 85
- validate() (troposphere.openstack.neutron.FirewallRule method), 27
- validate() (troposphere.openstack.neutron.HealthMonitor method), 27
- validate() (troposphere.openstack.neutron.Pool method), 28
- validate() (troposphere.openstack.neutron.SecurityGroupRule method), 28
- validate() (troposphere.openstack.neutron.SessionPersistence method), 28
- validate() (troposphere.openstack.nova.BlockDeviceMapping method), 28
- validate() (troposphere.openstack.nova.Server method), 29
- validate() (troposphere.opsworks.BlockDeviceMapping method), 100
- validate() (troposphere.opsworks.Stack method), 102
- validate() (troposphere.opsworks.VolumeConfiguration method), 102
- validate() (troposphere.Parameter method), 131
- validate() (troposphere.rds.DBInstance method), 103
- validate() (troposphere.s3.Bucket method), 108
- validate() (troposphere.s3.LifecycleRule method), 109
- validate() (troposphere.serverless.Api method), 113
- validate() (troposphere.serverless.DeadLetterQueue method), 113
- validate() (troposphere.serverless.Function method), 114
- validate() (troposphere.serverless.FunctionForPackaging method), 114
- validate() (troposphere.serverless.SQSEvent method), 115
- validate() (troposphere.sqs.Queue method), 121
- validate_action_on_failure() (in module troposphere.emr), 86
- validate_authentication_type() (in module troposphere.cloudformation), 50
- validate_authorizer_ttl() (in module troposphere.apigateway), 35
- validate_authorizer_ttl() (in module troposphere.apigatewayv2), 37
- validate_authorizer_type() (in module troposphere.apigatewayv2), 37
- validate_backup_retention_period() (in module troposphere.rds), 104
- validate_backup_window() (in module troposphere.rds), 104
- validate_capacity() (in module troposphere.rds), 105
- validate_content_handling_strategy() (in module troposphere.apigatewayv2), 37
- validate_credentials_provider() (in module troposphere.codebuild), 54
- validate_data_source_type() (in module troposphere.opsworks), 102
- validate_delimiter() (in module troposphere), 132
- validate_elasticinferenceaccelerator_type() (in module troposphere.ec2), 75
- validate_engine() (in module troposphere.rds), 105
- validate_engine_mode() (in module troposphere.rds), 105
- validate_environment_state() (in module troposphere.batch), 47
- validate_gateway_response_type() (in module troposphere.apigateway), 35
- validate_image_pull_credentials() (in module troposphere.codebuild), 54
- validate_integration_type() (in module troposphere.apigatewayv2), 37
- validate_interval() (in module troposphere.dlm), 63
- validate_interval_unit() (in module troposphere.dlm), 63
- validate_iops() (in module troposphere.rds), 105
- validate_license_model() (in module troposphere.rds), 105
- validate_logging_level() (in module troposphere.apigatewayv2), 37
- validate_maintenance_window() (in module troposphere.rds), 105
- validate_memory_size() (in module troposphere.awslambda), 46
- validate_node_group_id() (in module troposphere.elasticache), 79
- validate_passthrough_behavior() (in module troposphere.apigatewayv2), 37
- validate_pausetime() (in module troposphere), 132
- validate_predictivescalingmaxcapacitybehavior() (in module troposphere.autoscalingplans), 44
- validate_predictivescalingmode() (in module troposphere.autoscalingplans), 44
- validate_queue_state() (in module troposphere.batch), 47
- validate_ruletype() (in module troposphere.route53), 107
- validate_scalingpolicyupdatebehavior() (in module troposphere.autoscalingplans), 44
- validate_state() (in module troposphere.dlm), 64
- validate_status() (in module troposphere.codebuild), 54
- validate_storage_type() (in module troposphere.rds), 105
- validate_target_types() (in module troposphere.secretsmanager), 112
- validate_tier_name() (in module troposphere.elasticbeanstalk), 80
- validate_tier_type() (in module troposphere.elasticbeanstalk), 80

- sphere.elasticbeanstalk), 80
 - validate_title() (troposphere.BaseAWSObject method), 130
 - validate_title() (troposphere.Parameter method), 131
 - validate_unit() (in module troposphere.cloudwatch), 53
 - validate_variables_name() (in module troposphere.awslambda), 46
 - validate_volume_type() (in module troposphere.elasticsearch), 83
 - validate_volume_type() (in module troposphere.opsworks), 102
 - Variable (class in troposphere.iotanalytics), 98
 - Version (class in troposphere.awslambda), 46
 - VersioningConfiguration (class in troposphere.s3), 110
 - VersionWeight (class in troposphere.awslambda), 46
 - ViewerCertificate (class in troposphere.cloudfront), 52
 - VIP (class in troposphere.openstack.neutron), 28
 - Volume (class in troposphere.ec2), 75
 - Volume (class in troposphere.ecs), 77
 - Volume (class in troposphere.opsworks), 102
 - volume_type_validator() (in module troposphere.emr), 86
 - VolumeAttachment (class in troposphere.ec2), 75
 - VolumeConfiguration (class in troposphere.opsworks), 102
 - Volumes (class in troposphere.batch), 47
 - VolumesFrom (class in troposphere.ecs), 77
 - VolumesHost (class in troposphere.batch), 47
 - VolumeSpecification (class in troposphere.emr), 86
 - VPC (class in troposphere.ec2), 73
 - vpc_endpoint_type() (in module troposphere.validators), 125
 - VPCCidrBlock (class in troposphere.ec2), 73
 - VpcConfig (class in troposphere.appstream), 39
 - VPConfig (class in troposphere.awslambda), 46
 - VpcConfig (class in troposphere.codebuild), 54
 - VpcConfig (class in troposphere.sagemaker), 112
 - VPCDHCPOptionsAssociation (class in troposphere.ec2), 73
 - VPCEndpoint (class in troposphere.ec2), 73
 - VPCEndpointConnectionNotification (class in troposphere.ec2), 74
 - VPCEndpointService (class in troposphere.ec2), 74
 - VPCEndpointServicePermissions (class in troposphere.ec2), 74
 - VPCGatewayAttachment (class in troposphere.ec2), 74
 - VpcLink (class in troposphere.apigateway), 35
 - VPCOptions (class in troposphere.elasticsearch), 83
 - VPCPeeringConnection (class in troposphere.ec2), 74
 - VpcSettings (class in troposphere.directoryservice), 63
 - vpn_pre_shared_key() (in module troposphere.validators), 125
 - vpn_tunnel_inside_cidr() (in module troposphere.validators), 125
 - VPNConnection (class in troposphere.ec2), 74
 - VPNConnectionRoute (class in troposphere.ec2), 74
 - VPNGateway (class in troposphere.ec2), 74
 - VPNGatewayRoutePropagation (class in troposphere.ec2), 74
 - VpnTunnelOptionsSpecification (class in troposphere.ec2), 75
- ## W
- WaitCondition (class in troposphere.cloudformation), 50
 - WaitConditionHandle (class in troposphere.cloudformation), 50
 - WebACL (class in troposphere.waf), 126
 - WebACL (class in troposphere.wafregional), 128
 - WebACLAssociation (class in troposphere.wafregional), 128
 - Webhook (class in troposphere.codepipeline), 58
 - WebhookAuthConfiguration (class in troposphere.codepipeline), 58
 - WebhookFilterRule (class in troposphere.codepipeline), 58
 - WebsiteConfiguration (class in troposphere.s3), 111
 - WorkmailAction (class in troposphere.ses), 120
 - Workspace (class in troposphere.workspaces), 128
 - WorkspaceProperties (class in troposphere.workspaces), 128
- ## X
- XMLClassifier (class in troposphere.glue), 91
 - XssMatchSet (class in troposphere.waf), 126
 - XssMatchSet (class in troposphere.wafregional), 128
 - XssMatchTuple (class in troposphere.waf), 127
 - XssMatchTuple (class in troposphere.wafregional), 128