

---

# Kubeflow Fairing

*Release v0.7.0*

Nov 08, 2019



---

## Contents:

---

<b>1</b>	<b>kubeflow.fairing package</b>	<b>3</b>
1.1	Subpackages . . . . .	3
1.2	Submodules . . . . .	31
1.3	kubeflow.fairing.config module . . . . .	31
1.4	kubeflow.fairing.http_utils module . . . . .	31
1.5	kubeflow.fairing.runtime_config module . . . . .	32
1.6	kubeflow.fairing.utils module . . . . .	32
1.7	Module contents . . . . .	32
	<b>Python Module Index</b>	<b>33</b>
	<b>Index</b>	<b>35</b>



Main documentation: <https://www.kubeflow.org/docs/fairing/>

Source code: <https://github.com/kubeflow/fairing/>



## 1.1 Subpackages

### 1.1.1 kubeflow.fairing.backends package

#### Submodules

#### kubeflow.fairing.backends.backends module

**class** `kubeflow.fairing.backends.backends.AWSBackend` (*namespace=None, build\_context\_source=None*)

Bases: `kubeflow.fairing.backends.backends.KubernetesBackend`

Use to create a builder instance and create a deployer to be used with a training job or a serving job for the AWS backend.

**get\_builder** (*preprocessor, base\_image, registry, needs\_deps\_installation=True, pod\_spec\_mutators=None*)

Creates a builder instance with right config for AWS

#### Parameters

- **preprocessor** – Preprocessor to use to modify inputs
- **base\_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs\_deps\_installation** – need depends on installation(Default value = True)
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. e.g. fairing.cloud.gcp.add\_gcp\_credentials\_if\_exists This can used to set things like volumes and security context. (Default value =None)

**get\_serving\_deployer** (*model\_class, service\_type='ClusterIP', pod\_spec\_mutators=None*)

Creates a deployer to be used with a serving job for AWS

**Parameters**

- **model\_class** – the name of the class that holds the predict function.
- **service\_type** – service type (Default value = 'ClusterIP')
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. (Default value = None)

**get\_training\_deployer** (*pod\_spec\_mutators=None*)

Creates a deployer to be used with a training job for AWS

**Parameters** **pod\_spec\_mutators** – list of functions that is used to mutate the podspec.  
(Default value = None)

**Returns** job for handle all the k8s' template building for a training

**class** kubeflow.fairing.backends.backends.**AzureBackend** (*namespace=None, build\_context\_source=None*)  
Bases: *kubeflow.fairing.backends.backends.KubernetesBackend*

Use to create a builder instance and create a deployer to be used with a traing job or a serving job for the Azure backend.

**get\_builder** (*preprocessor, base\_image, registry, needs\_deps\_installation=True, pod\_spec\_mutators=None*)

Creates a builder instance with right config for Azure

**Parameters**

- **preprocessor** – Preprocessor to use to modify inputs
- **base\_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs\_deps\_installation** – need depends on installation(Default value = True)
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. e.g. fairing.cloud.gcp.add\_gcp\_credentials\_if\_exists This can used to set things like volumes and security context. (Default value =None)

**class** kubeflow.fairing.backends.backends.**BackendInterface**

Bases: object

Backend interface. Creating a builder instance or a deployer to be used with a traing job or a serving job for the given backend. And get the appropriate base container or docker registry for the current environment.

**get\_base\_contanier** ()

Returns the appropriate base container for the current environment

**Returns** base image

**get\_builder** (*preprocessor, base\_image, registry*)

Creates a builder instance with right config for the given backend

**Parameters**

- **preprocessor** – Preprocessor to use to modify inputs
- **base\_image** – Base image to use for this builder
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images

**Raises** **NotImplementedError** – not implemented exception



**get\_docker\_registry()**

Returns the appropriate docker registry for the current environment

**Returns** None

**get\_serving\_deployer(model\_class)**

Creates a deployer to be used with a serving job

**Parameters** **model\_class** – the name of the class that holds the predict function.

**Raises** **NotImplementedError** – not implemented exception

**get\_training\_deployer(pod\_spec\_mutators=None)**

Creates a deployer to be used with a training job

**Parameters** **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. e.g. `fairing.cloud.gcp.add_gcp_credentials_if_exists` This can be used to set things like volumes and security context. (Default value = None)

**Raises** **NotImplementedError** – not implemented exception

**class** `kubeflow.fairing.backends.backends.GCPManagedBackend` (*project\_id=None, region=None, training\_scale\_tier=None*)

Bases: `kubeflow.fairing.backends.backends.BackendInterface`

Use to create a builder instance and create a deployer to be used with a training job or a serving job for the GCP.

**get\_builder** (*preprocessor, base\_image, registry, needs\_deps\_installation=True, pod\_spec\_mutators=None*)

Creates a builder instance with right config for GCP

**Parameters**

- **preprocessor** – Preprocessor to use to modify inputs
- **base\_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: `gcr.io/kubeflow-images`
- **needs\_deps\_installation** – need depends on installation (Default value = True)
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. e.g. `fairing.cloud.gcp.add_gcp_credentials_if_exists` This can be used to set things like volumes and security context. (Default value = None)

**get\_docker\_registry()**

Returns the appropriate docker registry for GCP

**Returns** docker registry

**get\_serving\_deployer(model\_class, pod\_spec\_mutators=None)**

Creates a deployer to be used with a serving job for GCP

**Parameters**

- **model\_class** – the name of the class that holds the predict function.
- **service\_type** – service type (Default value = 'ClusterIP')
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. (Default value = None)

**get\_training\_deployer(pod\_spec\_mutators=None)**

Creates a deployer to be used with a training job for GCP

**Parameters** `pod_spec_mutators` – list of functions that is used to mutate the podspec.  
(Default value = None)

**Returns** job for handle all the k8s' template building for a training

```
class kubeflow.fairing.backends.backends.GKEBackend (namespace=None,  
                                                    build_context_source=None)  
Bases: kubeflow.fairing.backends.backends.KubernetesBackend
```

Use to create a builder instance and create a deployer to be used with a traing job or a serving job for the GKE backend. And get the appropriate docker registry for GKE.

```
get_builder (preprocessor,      base_image,      registry,      needs_deps_installation=True,  
              pod_spec_mutators=None)  
Creates a builder instance with right config for GKE
```

**Parameters**

- **preprocessor** – Preprocessor to use to modify inputs
- **base\_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs\_deps\_installation** – need depends on installation(Default value = True)
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. e.g. fairing.cloud.gcp.add\_gcp\_credentials\_if\_exists This can used to set things like volumes and security context. (Default value =None)

```
get_docker_registry ()  
Returns the appropriate docker registry for GKE
```

**Returns** docker registry

```
get_serving_deployer (model_class, service_type='ClusterIP', pod_spec_mutators=None)  
Creates a deployer to be used with a serving job for GKE
```

**Parameters**

- **model\_class** – the name of the class that holds the predict function.
- **service\_type** – service type (Default value = 'ClusterIP')
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. (Default value = None)

```
get_training_deployer (pod_spec_mutators=None)  
Creates a deployer to be used with a training job for GKE
```

**Parameters** `pod_spec_mutators` – list of functions that is used to mutate the podspec.  
(Default value = None)

**Returns** job for handle all the k8s' template building for a training

```
class kubeflow.fairing.backends.backends.KubeflowAWSBackend (namespace=None,  
                                                            build_context_source=None)  
Bases: kubeflow.fairing.backends.backends.AWSBackend
```

Kubeflow for AWS backend refer to AWSBackend

```
class kubeflow.fairing.backends.backends.KubeflowAzureBackend (namespace=None,  
                                                                build_context_source=None)  
Bases: kubeflow.fairing.backends.backends.AzureBackend
```

Kubeflow for Azure backend refer to AzureBackend

```
class kubeflow.fairing.backends.backends.KubeflowBackend (namespace=None,
                                                         build_context_source=None)
    Bases: kubeflow.fairing.backends.backends.KubernetesBackend
    Kubeflow backend refer to KubernetesBackend
```

```
class kubeflow.fairing.backends.backends.KubeflowGKEBackend (namespace=None,
                                                             build_context_source=None)
    Bases: kubeflow.fairing.backends.backends.GKEBackend
    Kubeflow for GKE backend refer to GKEBackend
```

```
class kubeflow.fairing.backends.backends.KubernetesBackend (namespace=None,
                                                             build_context_source=None)
    Bases: kubeflow.fairing.backends.backends.BackendInterface
    Use to create a builder instance and create a deployer to be used with a training job or a serving job for the
    Kubernetes.
```

```
get_builder (preprocessor,          base_image,          registry,          needs_deps_installation=True,
              pod_spec_mutators=None)
    Creates a builder instance with right config for the given Kubernetes
```

**Parameters**

- **preprocessor** – Preprocessor to use to modify inputs
- **base\_image** – Base image to use for this job
- **registry** – Registry to push image to. Example: gcr.io/kubeflow-images
- **needs\_deps\_installation** – need depends on installation(Default value = True)
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. e.g. fairing.cloud.gcp.add\_gcp\_credentials\_if\_exists This can used to set things like volumes and security context. (Default value =None)

```
get_serving_deployer (model_class, service_type='ClusterIP', pod_spec_mutators=None)
    Creates a deployer to be used with a serving job for the Kubernetes
```

**Parameters**

- **model\_class** – the name of the class that holds the predict function.
- **service\_type** – service type (Default value = 'ClusterIP')
- **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. (Default value = None)

```
get_training_deployer (pod_spec_mutators=None)
    Creates a deployer to be used with a training job for the Kubernetes
```

**Parameters** **pod\_spec\_mutators** – list of functions that is used to mutate the podspec. (Default value = None)

**Returns** job for handle all the k8s' template building for a training

## Module contents

### 1.1.2 kubeflow.fairing.builders package

#### Subpackages

## kubeflow.fairing.builders.append package

### Submodules

## kubeflow.fairing.builders.append.append module

```
class kubeflow.fairing.builders.append.append.AppendBuilder (registry=None,  
                                                             image_name='fairing-  
job',  
                                                             base_image='gcr.io/kubeflow-  
images-  
public/fairing:dev',  
                                                             push=True, prepro-  
cessor=None)
```

Bases: *kubeflow.fairing.builders.base\_builder.BaseBuilder*

Builds a docker image by appending a new layer tarball to an existing base image. Does not require docker and runs in userspace.

#### Parameters

- **base\_image** – Base image to use for the build (default: {constants.DEFAULT\_BASE\_IMAGE})
- **image\_name** – image name to use for the new image(default: {constants.DEFAULT\_IMAGE\_NAME})
- **preprocessor** – Preprocessor{BasePreProcessor} to use to modify inputs before sending them to docker build
- **push** – Whether or not to push the image to the registry

**build()**

Will be called when the build needs to start

**timed\_push** (transport, src, img, dst)

Push image to the registry and log the time spent to the log

#### Parameters

- **transport** – the http transport to use for sending requests
- **src** – repo from which to mount blobs
- **img** – the image to be pushed
- **dst** – the fully-qualified name of the tag to push

### Module contents

## kubeflow.fairing.builders.cluster package

### Submodules

**kubeflow.fairing.builders.cluster.azurestorage\_context module**

```
class kubeflow.fairing.builders.cluster.azurestorage_context.StorageContextSource (namespace=
re-
gion=None,
re-
source_grou
stor-
age_account

Bases:
kubeflow.fairing.builders.cluster.context_source.
ContextSourceInterface

Azure storage context source

cleanup ()
    Cleans up the context after the build is complete

generate_pod_spec (image_name, push)
    Generates a pod spec for building the image in the cluster, pointing to the prepared build context

    Parameters pod_spec – pod spec

prepare (context_filename)
    Makes the context somehow available for use in the pod spec

upload_context (context_filename)
```

**kubeflow.fairing.builders.cluster.cluster module**

```
class kubeflow.fairing.builders.cluster.cluster.ClusterBuilder (registry=None,
image_name='fairing-
job',
con-
text_source=None,
preproces-
sor=None,
push=True,
base_image='gcr.io/kubeflow-
images-
public/fairing:dev',
pod_spec_mutators=None,
names-
pace=None,
docker-
file_path=None,
cleanup=False)

Bases: kubeflow.fairing.builders.base_builder.BaseBuilder

Builds a docker image in a Kubernetes cluster.

build ()
    Runs the build
```

**kubeflow.fairing.builders.cluster.context\_source module**

```
class kubeflow.fairing.builders.cluster.context_source.ContextSourceInterface
    Bases: object
```

Interface that provides the build context to the in cluster builder

**cleanup** ()

Cleans up the context after the build is complete

**generate\_pod\_spec** (*pod\_spec*)

Generates a pod spec for building the image in the cluster, pointing to the prepared build context

Parameters **pod\_spec** – pod spec

**prepare** ()

Makes the context somehow available for use in the pod spec

### **kubeflow.fairing.builders.cluster.gcs\_context module**

```
class kubeflow.fairing.builders.cluster.gcs_context.GCSTextSource (gcp_project=None,  
                                                                creden-  
                                                                tials_file=None,  
                                                                names-  
                                                                pace='default')
```

Bases: *kubeflow.fairing.builders.cluster.context\_source.*  
*ContextSourceInterface*

Google cloud storage context for docker builder

**cleanup** ()

Cleans up the context after the build is complete

**generate\_pod\_spec** (*image\_name, push*)

Generates a pod spec for building the image in the cluster, pointing to the prepared build context

Parameters **pod\_spec** – pod spec

**prepare** (*context\_filename*)

Makes the context somehow available for use in the pod spec

**upload\_context** (*context\_filename*)

### **kubeflow.fairing.builders.cluster.s3\_context module**

```
class kubeflow.fairing.builders.cluster.s3_context.S3ContextSource (aws_account=None,  
                                                                re-  
                                                                gion=None,  
                                                                bucket_name=None)
```

Bases: *kubeflow.fairing.builders.cluster.context\_source.*  
*ContextSourceInterface*

aws S3 context for docker builder

**cleanup** ()

Cleans up the context after the build is complete

**generate\_pod\_spec** (*image\_name, push*)

Generates a pod spec for building the image in the cluster, pointing to the prepared build context

Parameters **pod\_spec** – pod spec

**prepare** (*context\_filename*)

Parameters **context\_filename** – context filename

**upload\_context** (*context\_filename*)

Parameters **context\_filename** – context filename

## Module contents

### kubeflow.fairing.builders.docker package

## Submodules

### kubeflow.fairing.builders.docker.docker module

```
class kubeflow.fairing.builders.docker.docker.DockerBuilder (registry=None,
                                                             image_name='fairing-
                                                             job',
                                                             base_image='gcr.io/kubeflow-
                                                             images-
                                                             public/fairing:dev',
                                                             preprocessor=None,
                                                             push=True, docker-
                                                             file_path=None)
```

Bases: *kubeflow.fairing.builders.base\_builder.BaseBuilder*

A builder using the local Docker client

**build**()

Runs the build

**publish**()

push the docker image to the docker registry

## Module contents

## Submodules

### kubeflow.fairing.builders.base\_builder module

```
class kubeflow.fairing.builders.base_builder.BaseBuilder (registry=None,      im-
                                                             age_name=None,
                                                             base_image='gcr.io/kubeflow-
                                                             images-
                                                             public/fairing:dev',
                                                             push=True,      prepro-
                                                             cessor=None,      docker-
                                                             file_path=None)
```

Bases: *kubeflow.fairing.builders.builder.BuilderInterface*

A builder using the local Docker client

**build**()

Runs the build

**full\_image\_name** (*tag*)

Retrun the full image name

**Parameters** `tag` – the new tag for the image

**generate\_pod\_spec()**

This method should return a `V1PodSpec` with the correct image set. This is also where the builder should set the environment variables and volume/volumeMounts that it may need to work

### **kubeflow.fairing.builders.builder module**

**class** `kubeflow.fairing.builders.builder.BuilderInterface`

Bases: `object`

**build()**

Will be called when the build needs to start

**generate\_pod\_spec()**

This method should return a `V1PodSpec` with the correct image set. This is also where the builder should set the environment variables and volume/volumeMounts that it may need to work

### **kubeflow.fairing.builders.dockerfile module**

`kubeflow.fairing.builders.dockerfile.write_dockerfile` (*docker\_command=None*,  
*destination=None*,  
*path\_prefix='/app/'*,  
*base\_image=None*, *install\_reqs\_before\_copy=False*)

Generate dockerfile according to the parameters

#### **Parameters**

- **docker\_command** – string, CMD of the dockerfile (Default value = `None`)
- **destination** – string, destination folder for this dockerfile (Default value = `None`)
- **path\_prefix** – string, WORKDIR (Default value = `constants.DEFAULT_DEST_PREFIX`)
- **base\_image** – string, base image, example: `gcr.io/kubeflow-image`
- **install\_reqs\_before\_copy** – whether to install the prerequisites (Default value = `False`)

## **Module contents**

### **1.1.3 kubeflow.fairing.cloud package**

#### **Submodules**

#### **kubeflow.fairing.cloud.aws module**

**class** `kubeflow.fairing.cloud.aws.S3Uploader` (*region*)

Bases: `object`

For AWS S3 up load

**create\_bucket\_if\_not\_exists** (*bucket\_name*)

Create bucket if this bucket not exists

**Parameters** `bucket_name` – Bucket name



**upload\_to\_bucket** (*blob\_name, bucket\_name, file\_to\_upload*)

Upload a file to an S3 bucket

**Parameters**

- **blob\_name** – S3 object name
- **bucket\_name** – Bucket to upload to
- **file\_to\_upload** – File to upload

`kubeflow.fairing.cloud.aws.add_aws_credentials` (*kube\_manager, pod\_spec, namespace*)  
add AWS credential

**Parameters**

- **kube\_manager** – kube manager for handles communication with Kubernetes' client
- **pod\_spec** – pod spec like volumes and security context
- **namespace** – The custom resource

`kubeflow.fairing.cloud.aws.add_aws_credentials_if_exists` (*kube\_manager, pod\_spec, namespace*)  
add AWS credential

**Parameters**

- **kube\_manager** – kube manager for handles communication with Kubernetes' client
- **pod\_spec** – pod spec like volumes and security context
- **namespace** – The custom resource

`kubeflow.fairing.cloud.aws.add_ecr_config` (*kube\_manager, pod\_spec, namespace*)  
add secret

**Parameters**

- **kube\_manager** – kube manager for handles communication with Kubernetes' client
- **pod\_spec** – pod spec like volumes and security context
- **namespace** – The custom resource

`kubeflow.fairing.cloud.aws.create_ecr_registry` (*registry, repository*)  
create secret registry

**Parameters**

- **registry** – registry
- **repository** – repository name

`kubeflow.fairing.cloud.aws.guess_account_id` ()  
Get account id

`kubeflow.fairing.cloud.aws.is_ecr_registry` (*registry*)  
verify secrte registry

**Parameters** **registry** – registry

## kubeflow.fairing.cloud.azure module

**class** `kubeflow.fairing.cloud.azure.AzureFileUploader` (*namespace, credentials=None, subscription\_id=None*)

Bases: object

```
create_share_if_not_exists (share_service, share_name)  
create_storage_account_if_not_exists (region, resource_group_name, stor-  
                                         age_account_name)  
    Creates the storage account if it does not exist.  
    In either case, returns the StorageAccount class that matches the given arguments.  
delete_uncompressed_files (target_dir)  
get_storage_credentials (resource_group_name, storage_account_name)  
uncompress_tar_gz_file (tar_gz_file, target_dir)  
upload_tar_gz_contents (share_service, share_name, dir_name, tar_gz_file)  
upload_to_share (region, resource_group_name, storage_account_name, share_name, dir_name,  
                  tar_gz_file_to_upload)
```

```
kubeflow.fairing.cloud.azure.add_acr_config (kube_manager, pod_spec, namespace)  
kubeflow.fairing.cloud.azure.add_azure_files (kube_manager, pod_spec, namespace)  
kubeflow.fairing.cloud.azure.create_storage_creds_secret (namespace, con-  
                                                         text_hash, stor-  
                                                         age_account_name,  
                                                         storage_key)  
kubeflow.fairing.cloud.azure.delete_storage_creds_secret (namespace, con-  
                                                         text_hash)  
kubeflow.fairing.cloud.azure.get_azure_credentials (namespace)  
kubeflow.fairing.cloud.azure.get_plain_secret_value (secret_data, key)  
kubeflow.fairing.cloud.azure.is_acr_registry (registry)
```

### kubeflow.fairing.cloud.docker module

```
kubeflow.fairing.cloud.docker.add_docker_credentials (kube_manager, pod_spec,  
                                                         namespace)  
kubeflow.fairing.cloud.docker.add_docker_credentials_if_exists (kube_manager,  
                                                                    pod_spec,  
                                                                    namespace)  
kubeflow.fairing.cloud.docker.create_docker_secret (kube_manager, namespace)  
kubeflow.fairing.cloud.docker.get_docker_secret ()
```

### kubeflow.fairing.cloud.gcp module

```
class kubeflow.fairing.cloud.gcp.GCSUploader (credentials_file=None)  
    Bases: object  
    get_or_create_bucket (bucket_name)  
    upload_to_bucket (blob_name, bucket_name, file_to_upload)  
kubeflow.fairing.cloud.gcp.add_gcp_credentials (kube_manager, pod_spec, namespace)  
    Note: This method will be deprecated soon and will become unavailable by 1.0. All future access will use  
    Workload Identity.
```

```
kubeflow.fairing.cloud.gcp.add_gcp_credentials_if_exists(kube_manager,
                                                         pod_spec, namespace)
kubeflow.fairing.cloud.gcp.get_default_docker_registry()
kubeflow.fairing.cloud.gcp.guess_project_name(credentials_file=None)
```

## kubeflow.fairing.cloud.storage module

```
class kubeflow.fairing.cloud.storage.GCSStorage
    Bases: kubeflow.fairing.cloud.storage.Storage

    copy_cmd (src_url, dst_url, recursive=True)
        gets a command to copy files from/to remote storage from/to local FS

    exists (url)
        checks if the url exists in the given storage

class kubeflow.fairing.cloud.storage.Storage
    Bases: object

    copy_cmd (src_url, dst_url, recursive=True)
        gets a command to copy files from/to remote storage from/to local FS

    exists (url)
        checks if the url exists in the given storage

kubeflow.fairing.cloud.storage.get_storage_class(url)
kubeflow.fairing.cloud.storage.lookup_storage_class(url)
```

## Module contents

### 1.1.4 kubeflow.fairing.constants package

#### Submodules

#### kubeflow.fairing.constants.constants module

#### Module contents

### 1.1.5 kubeflow.fairing.deployers package

#### Subpackages

#### kubeflow.fairing.deployers.gcp package

#### Submodules

#### kubeflow.fairing.deployers.gcp.gcp module

```
class kubeflow.fairing.deployers.gcp.gcp.GCPJob(project_id=None,      region=None,
                                                scale_tier=None, job_config=None)
    Bases: kubeflow.fairing.deployers.deployer.DeployerInterface

    Handle submitting training job to GCP.
```

**create\_request\_dict** (*pod\_template\_spec*)

Return the request to be sent to the ML Engine API.

Parameters **pod\_template\_spec** – pod spec template of the training job

**deploy** (*pod\_template\_spec*)

Deploys the training job

Parameters **pod\_template\_spec** – pod spec template of the training job

**get\_logs** ()

Streams the logs for the training job

## kubeflow.fairing.deployers.gcp.gcpserving module

```
class kubeflow.fairing.deployers.gcp.gcpserving.GCPServingDeployer(model_dir,  
                                                                model_name,  
                                                                ver-  
                                                                sion_name,  
                                                                project_id=None,  
                                                                **de-  
                                                                ploy_kwargs)
```

Bases: *kubeflow.fairing.deployers.deployer.DeployerInterface*

Handle deploying a trained model to GCP.

**deploy** (*pod\_template\_spec*)

Deploys the model to Cloud ML Engine.

Parameters **pod\_template\_spec** – pod spec template of training job

**get\_logs** ()

abstract get log

## Module contents

### kubeflow.fairing.deployers.job package

#### Submodules

### kubeflow.fairing.deployers.job.job module

```
class kubeflow.fairing.deployers.job.job.Job(namespace=None,      runs=1,      out-  
                                           put=None,      cleanup=True,      la-  
                                           bels=None,      job_name='fairing-job',  
                                           stream_log=True,      deployer_type='job',  
                                           pod_spec_mutators=None,      annota-  
                                           tions=None)
```

Bases: *kubeflow.fairing.deployers.deployer.DeployerInterface*

Handle all the k8s' template building for a training

**create\_resource** ()

create job

**deploy** (*pod\_spec*)

deploy the training job using k8s client lib

**Parameters** `pod_spec` – pod spec of deployed training job

**do\_cleanup** ()

clean up the pods after job finished

**generate\_deployment\_spec** (*pod\_template\_spec*)

**Generate a V1Job initialized with correct completion and parallelism** (for HP search) and with the provided V1PodTemplateSpec

**Parameters** `pod_template_spec` – V1PodTemplateSpec

**generate\_pod\_template\_spec** (*pod\_spec*)

**Generate a V1PodTemplateSpec initialized with correct metadata** and with the provided `pod_spec`

**Parameters** `pod_spec` – pod spec

**get\_logs** ()

get logs from the deployed job

**set\_annotatations** (*annotations*)

**set\_labels** (*labels, deployer\_type*)

set labels for the pods of a deployed job

**Parameters**

- **labels** – dictionary of labels {label\_name:label\_value}
- **deployer\_type** – deployer type name

## Module contents

### kubeflow.fairing.deployers.kfserving package

## Submodules

## kubeflow.fairing.deployers.kfserving.kfserving module

```
class kubeflow.fairing.deployers.kfserving.kfserving.KFServing(framework, de-  
fault_storage_uri=None, ca-  
nary_storage_uri=None, ca-  
nary_traffic_percent=0, names-  
pace=None, labels=None, annota-  
tions=None, cus-  
tom_default_container=None, cus-  
tom_canary_container=None, stream_log=False, cleanup=False)
```

Bases: *kubeflow.fairing.deployers.deployer.DeployerInterface*

Serves a prediction endpoint using Kubeflow KFServing.

**deploy** (*isvc*)

deploy kfserving endpoint

**Parameters** **isvc** – InferenceService for deploying.

**generate\_isvc** ()

generate InferenceService

**generate\_predictor\_spec** (*framework, storage\_uri=None, container=None*)

Generate predictor spec according to framework and default\_storage\_uri or custom container.

**get\_logs** ()

get log from prediction pod

**set\_labels** (*labels*)

set label for deployed prediction

**Parameters** **labels** – dictionary of labels {label\_name:label\_value}

## Module contents

### kubeflow.fairing.deployers.serving package

#### Submodules

### kubeflow.fairing.deployers.serving.serving module

```
class kubeflow.fairing.deployers.serving.serving.Serving(serving_class, names-  
pace=None, runs=1, labels=None, ser-  
vice_type='ClusterIP', pod_spec_mutators=None)
```

Bases: *kubeflow.fairing.deployers.job.job.Job*

Serves a prediction endpoint using Kubernetes deployments and services

**delete()**

delete the deployed service

**deploy(*pod\_spec*)**

deploy a seldon-core REST service

Parameters **pod\_spec** – pod spec for the service

**generate\_deployment\_spec(*pod\_template\_spec*)**

generate deployment spec(V1Deployment)

Parameters **pod\_template\_spec** – pod spec template

**generate\_service\_spec()**

generate service spec(V1ServiceSpec)

## Module contents

### kubeflow.fairing.deployers.tfjob package

#### Submodules

#### kubeflow.fairing.deployers.tfjob.tfjob module

```
class kubeflow.fairing.deployers.tfjob.tfjob.TfJob(namespace=None,
                                                    worker_count=1,    ps_count=0,
                                                    chief_count=1,      runs=1,
                                                    job_name='fairing-tfjob-',
                                                    stream_log=True,   labels=None,
                                                    pod_spec_mutators=None,
                                                    cleanup=False,      annota-
                                                    tions=None)
```

Bases: *kubeflow.fairing.deployers.job.job.Job*

Handle all the k8s' template building to create tensorflow training job using Kubeflow TFOperator

**create\_resource()**

create a tfjob training

**generate\_deployment\_spec(*pod\_template\_spec*)**

Returns a TFJob template

Parameters **pod\_template\_spec** – template spec for pod

**get\_logs()**

get logs

**set\_container\_name(*pod\_template\_spec*)**

Sets the name of the main container to *tensorflow*. This is required for TfJobs

Parameters **pod\_template\_spec** – spec for pod template

## Module contents

### Submodules

#### kubeflow.fairing.deployers.deployer module

**class** kubeflow.fairing.deployers.deployer.**DeployerInterface**

Bases: object

Deploys a training job to the cluster

**deploy** (*pod\_template\_spec*)

Deploys the training job

**Parameters** *pod\_template\_spec* – pod template spec

**get\_logs** ()

Streams the logs for the training job

## Module contents

### 1.1.6 kubeflow.fairing.frameworks package

#### Submodules

#### kubeflow.fairing.frameworks.lightgbm module

kubeflow.fairing.frameworks.lightgbm.**execute** (*config*, *docker\_registry*,  
*base\_image*=*'gcr.io/kubeflow-fairing/lightgbm:latest'*, *names-*  
*pace*=*None*, *stream\_log*=*True*,  
*cores\_per\_worker*=*None*,  
*memory\_per\_worker*=*None*,  
*pod\_spec\_mutators*=*None*)

Runs the LightGBM CLI in a single pod in user's Kubeflow cluster. Users can configure it to be a train, predict, and other supported tasks by using the right config. Please refer <https://github.com/microsoft/LightGBM/blob/master/docs/Parameters.rst> for more information on config options.

#### Parameters

- **config** – config entries
- **docker\_registry** – docker registry name
- **base\_image** – base image (Default value = “gcr.io/kubeflow-fairing/lightgbm:latest”)
- **namespace** – k8s namespace (Default value = None)
- **stream\_log** – should that stream log? (Default value = True)
- **cores\_per\_worker** – number of cores per worker (Default value = None)
- **memory\_per\_worker** – memory value per worker (Default value = None)
- **pod\_spec\_mutators** – pod spec mutators (Default value = None)

kubeflow.fairing.frameworks.lightgbm.**generate\_context\_files** (*config*, *con-*  
*fig\_file\_name*,  
*num\_machines*)

generate context files



**Parameters**

- **config** – config entries
- **config\_file\_name** – config file name
- **num\_machines** – number of machines

**kubeflow.fairing.frameworks.lightgbm\_dist\_training\_init module****kubeflow.fairing.frameworks.utils module**

`kubeflow.fairing.frameworks.utils.get_config_value (config, field_names)`

get value for a config entry

**Parameters**

- **config** –
- **field\_names** –

`kubeflow.fairing.frameworks.utils.init_lightgbm_env (config_file, mlist_file)`

initialize env for lightgbm

**Parameters**

- **config\_file** – path to config path
- **mlist\_file** – path to file to write ip list

`kubeflow.fairing.frameworks.utils.load_properties_config_file (config_file)`

load config from a file

**Parameters** **config\_file** – config file path

`kubeflow.fairing.frameworks.utils.nslookup (hostname, retries=600)`

Does nslookup for the hostname and returns the IPs for it.

**Parameters**

- **hostname** – hostname to be looked up
- **retries** – Number of retries before failing. In autoscaled cluster, it might take upto 10mins to create a new node so the default value is set high.(Default value = 600)

`kubeflow.fairing.frameworks.utils.parse_cluster_spec_env ()`

parse cluster spec env variables

`kubeflow.fairing.frameworks.utils.save_properties_config_file (config, file_name=None)`

save config into a file

**Parameters**

- **config** – dictionary of give configs
- **file\_name** – path to config file(Default value = None)

`kubeflow.fairing.frameworks.utils.scrub_fields (config, filed_names)`

scrub fields in config

**Parameters**

- **config** – config spec
- **filed\_names** – name of fields

`kubeflow.fairing.frameworks.utils.update_config_file` (*file\_name*, *field\_name*,  
*new\_value*)  
update config file

**Parameters**

- **file\_name** – file name
- **field\_name** – field name
- **new\_value** – new value to be added/updated

`kubeflow.fairing.frameworks.utils.write_ip_list_file` (*file\_name*, *ips*, *port=None*)  
write list of ips into a file

**Parameters**

- **file\_name** – file name
- **ips** – list of ips
- **port** – default port(Default value = None)

## Module contents

### 1.1.7 kubeflow.fairing.functions package

#### Submodules

#### `kubeflow.fairing.functions.function_shim` module

**class** `kubeflow.fairing.functions.function_shim.ObjectType`  
Bases: `enum.Enum`

An enumeration.

**CLASS** = 2

**FUNCTION** = 1

**NOT\_SUPPORTED** = 3

`kubeflow.fairing.functions.function_shim.call` (*serialized\_fn\_file*)  
Get the content from serialized function.

**Parameters** **serialized\_fn\_file** – the file includes serialized function

**Returns** object: The content of object.

`kubeflow.fairing.functions.function_shim.compare_version` (*local\_python\_version*)  
Compare the Python major and minor version for local and remote python.

**Parameters** **local\_python\_version** – Python version of local environment

**Returns** None.

`kubeflow.fairing.functions.function_shim.get_execution_obj_type` (*obj*)  
Get the execution object type, the object can be a function or class.

**Parameters** **obj** – The name of object such as the function or class

**Returns** int: The corresponding object type.

## Module contents

### 1.1.8 kubeflow.fairing.kubernetes package

#### Submodules

#### kubeflow.fairing.kubernetes.manager module

**class** kubeflow.fairing.kubernetes.manager.**KubeManager**

Bases: object

Handles communication with Kubernetes' client.

**create\_deployment** (*namespace, deployment*)  
Create an V1Deployment in the specified namespace.

#### Parameters

- **namespace** – The custom resource
- **deployment** – Deployment body to create

**Returns** object: Created V1Deployments.

**create\_isvc** (*namespace, isvc*)  
Create the provided InferenceService in the specified namespace.

#### Parameters

- **namespace** – The custom resource
- **InferenceService** – The InferenceService body

**Returns** object: Created InferenceService.

**create\_job** (*namespace, job*)  
Creates a V1Job in the specified namespace.

#### Parameters

- **namespace** – The resource
- **job** – Job definition as kubernetes

**Returns** object: Created Job.

**create\_secret** (*namespace, secret*)  
Create secret in the specified namespace.

#### Parameters

- **namespace** – The custom resource
- **secret** – The secret body

**Returns** object: Created secret.

**create\_tf\_job** (*namespace, job*)  
Create the provided TFJob in the specified namespace. The TFJob version is defined in TF\_JOB\_VERSION in fairing.constants. The version TFJob need to be installed before creating the TFJob.

#### Parameters

- **namespace** – The custom resource

- **job** – The JSON schema of the Resource to create

**Returns** object: Created TFJob.

**delete\_deployment** (*name, namespace*)

Delete an existing model deployment and relinquish all resources associated.

**Parameters**

- **name** – The deployment name
- **namespace** – The custom resource

**Returns** obje deployment.

**delete\_isvc** (*name, namespace*)

Delete the provided InferenceService in the specified namespace.

**Parameters**

- **name** – The custom object
- **namespace** – The custom resource

**Returns** object: The deleted InferenceService.

**delete\_job** (*name, namespace*)

Delete the specified job and related pods.

**Parameters**

- **name** – The job name
- **namespace** – The resource

**Returns** object: the deleted job.

**delete\_tf\_job** (*name, namespace*)

Delete the provided TFJob in the specified namespace.

**Parameters**

- **name** – The custom object
- **namespace** – The custom resource

**Returns** object: The deleted TFJob.

**get\_service\_external\_endpoint** (*name, namespace, selectors=None*)

Get the service external endpoint as [http://ip\\_or\\_hostname:5000/predict](http://ip_or_hostname:5000/predict).

**Parameters**

- **name** – The sevice name
- **namespace** – The custom resource
- **selectors** – A selector to restrict the list of returned objects by their labels.
- **Defaults** – to everything

**Returns** str: the service external endpoint.

**log** (*name, namespace, selectors=None, container=", follow=True*)

Get log of the specified pod.

**Parameters**

- **name** – The pod name

- **namespace** – The custom resource
- **selectors** – A selector to restrict the list of returned objects by their labels.
- **Defaults** – to everything
- **container** – The container for which to stream logs.
- **if** – there is one container in the pod
- **follow** – True or False (Default value = True)

**Returns** str: logs of the specified pod.

**secret\_exists** (*name, namespace*)

Check if the secret exists in the specified namespace.

#### Parameters

- **name** – The secret name
- **namespace** – The custom resource.

**Returns** bool: True if the secret exists, otherwise return False.

## kubeflow.fairing.kubernetes.utils module

`kubeflow.fairing.kubernetes.utils.get_resource_mutator` (*cpu=None, memory=None*)

The mutator for getting the resource setting for pod spec.

The useful example: [https://github.com/kubeflow/fairing/blob/master/examples/train\\_job\\_api/main.ipynb](https://github.com/kubeflow/fairing/blob/master/examples/train_job_api/main.ipynb)

#### Parameters

- **cpu** – Limits and requests for CPU resources (Default value = None)
- **memory** – Limits and requests for memory (Default value = None)

**Returns** object: The mutator function for setting cpu and memory in pod spec.

`kubeflow.fairing.kubernetes.utils.mounting_pvc` (*pvc\_name, pvc\_mount\_path='/mnt'*)

The function for pod\_spec\_mutators to mount persistent volume claim.

#### Parameters

- **pvc\_name** – The name of persistent volume claim
- **pvc\_mount\_path** – Path for the persistent volume claim mounts to.

**Returns** object: function for mount the pvc to pods.

## Module contents

### 1.1.9 kubeflow.fairing.ml\_tasks package

#### Submodules

**kubeflow.fairing.ml\_tasks.tasks module**

```
class kubeflow.fairing.ml_tasks.tasks.BaseTask (entry_point, base_docker_image=None,  
                                              docker_registry=None,          in-  
                                              put_files=None,          backend=None,  
                                              pod_spec_mutators=None)
```

Bases: object

Base class for handling high level ML tasks.

**Parameters**

- **entry\_point** – An object or reference to the source code that has to be deployed.
- **base\_docker\_image** – Name of the base docker image that should be used as a base image when building a new docker image as part of an ML task deployment.
- **docker\_registry** – Docker registry to store output docker images.
- **input\_files** – list of files that needs to be packaged along with the entry point. E.g. local python modules, trained model weights, etc.

```
class kubeflow.fairing.ml_tasks.tasks.PredictionEndpoint (model_class,  
                                                         base_docker_image=None,  
                                                         docker_registry=None,  
                                                         input_files=None,  
                                                         backend=None,          ser-  
                                                         vice_type='ClusterIP',  
                                                         pod_spec_mutators=None)
```

Bases: *kubeflow.fairing.ml\_tasks.tasks.BaseTask*

Create a prediction endpoint.

**create** ()

Create prediction endpoint.

**delete** ()

Delete prediction endpoint.

**predict\_nparray** (data, feature\_names=None)

Return the prediction result.

**Parameters**

- **data** – Data to be predicted.
- **feature\_names** – Feature extracted from data (Default value = None)

```
class kubeflow.fairing.ml_tasks.tasks.TrainJob (entry_point, base_docker_image=None,  
                                              docker_registry=None,          in-  
                                              put_files=None,          backend=None,  
                                              pod_spec_mutators=None)
```

Bases: *kubeflow.fairing.ml\_tasks.tasks.BaseTask*

Create a train job.

**submit** ()

Submit a train job.

## kubeflow.fairing.ml\_tasks.utils module

`kubeflow.fairing.ml_tasks.utils.guess_preprocessor` (*entry\_point*, *input\_files*, *output\_map*)

Preprocessor to use to modify inputs before sending them to docker build

### Parameters

- **entry\_point** – entry\_point which to use
- **input\_files** – input files
- **output\_map** – output

`kubeflow.fairing.ml_tasks.utils.is_docker_daemon_exists` ()

To check if docker daemon exists or not.

## Module contents

### 1.1.10 kubeflow.fairing.notebook package

#### Submodules

#### kubeflow.fairing.notebook.notebook\_util module

`kubeflow.fairing.notebook.notebook_util.get_notebook_name` ()

Return the full path of the jupyter notebook.

`kubeflow.fairing.notebook.notebook_util.is_in_notebook` ()

To check is in notenook or not.

## Module contents

### 1.1.11 kubeflow.fairing.preprocessors package

#### Submodules

#### kubeflow.fairing.preprocessors.base module

**class** `kubeflow.fairing.preprocessors.base.BasePreProcessor` (*input\_files=None*,  
*command=None*,  
*executable=None*,  
*path\_prefix='/app/'*,  
*output\_map=None*)

Bases: object

Prepares a context that gets sent to the builder for the docker build and sets the entrypoint :param input\_files: the source files to be processed :param executable: the file to execute using command (e.g. main.py) :param output\_map: a dict of files to be added without preprocessing :param path\_prefix: the prefix of the path where the files will be added in the container :param command: the command to pass to the builder

**context\_map** ()

Create context mapping from destination to source to avoid duplicates in context archive

**Returns** c\_map: a context map

**context\_tar\_gz** (*output\_file=None*)

Creating docker context file and compute a running cyclic redundancy check checksum.

**Parameters** **output\_file** – output file (Default value = None)

**Returns** output\_file,checksum: docker context file and checksum

**fairing\_runtime\_files** ()

Search the fairing runtime files 'runtime\_config.py' :returns: cmd: the execute with absolute path

**get\_command** ()

Get the execute with absolute path

**Returns** cmd: the execute with absolute path

**is\_requirements\_txt\_file\_present** ()

Verfiy the requirements txt file if it is present.

**Returns** res: get the present required files

**preprocess** ()

Preprocess the 'input\_files'.

**Returns** input\_files: get the input files

**set\_default\_executable** ()

Set the default executable file.

**Returns** executable: get the default executable file if it is not existing, Or None

kubeflow.fairing.preprocessors.base.**reset\_tar\_mtime** (*tarinfo*)

Reset the mtime on the the tarball for reproducibility.

:param tarinfo: the tarball var :returns: tarinfo: the modified tar ball

## kubeflow.fairing.preprocessors.converted\_notebook module

**class** kubeflow.fairing.preprocessors.converted\_notebook.**ConvertNotebookPreprocessor** (*notebook*,  
*note-*  
*book\_pre*  
*'kube-*  
*flow.fairin*  
*ex-*  
*e-*  
*cutable=i*  
*com-*  
*mand=[']*  
*path\_pre*  
*out-*  
*put\_map=*  
*over-*  
*write=Tr*

Bases: *kubeflow.fairing.preprocessors.base.BasePreProcessor*

Convert the notebook preprocessor. :param BasePreProcessor: a context that gets sent to the builder for the docker build and sets the endpoint.

**preprocess** ()

Preprocessor the Notebook :return:[]: the converted notebook list.



---

```
class kubeflow.fairing.preprocessors.converted_notebook.ConvertNotebookPreprocessorWithFire
```

Bases: *kubeflow.fairing.preprocessors.converted\_notebook.ConvertNotebookPreprocessor*

Create an endpoint using pyfire.

**preprocess()**

Preprocessor the Notebook. :return: results: the preprocessed notebook list.

```
class kubeflow.fairing.preprocessors.converted_notebook.FilterIncludeCell (**kw)
```

Bases: *nbconvert.preprocessors.base.Preprocessor*

Notebook preprocessor that only includes cells that have a comment 'fairing:include-cell'. :param NbPreProcessor: the notebook preprocessor.

**filter\_include\_cell** (src)

Filter the cell that have a comment 'fairing:include-cell'.

**Param** src: the source cell.

**Returns** src: if the source cell matched the filter pattern, or Null.

**preprocess\_cell** (cell, resources, index)

Preprocess the notebook cell.

**Parameters**

- **cell** – the notebook cell
- **resources** – the code source of the notebook cell.
- **index** – unused argument.

**Returns** cell,resources: the notebook cell and its filtered with magic pattern commands.

```
class kubeflow.fairing.preprocessors.converted_notebook.FilterMagicCommands (**kw)
```

Bases: *nbconvert.preprocessors.base.Preprocessor*

Notebook preprocessor that have a comment which started with '!' or '%'. :param NbPreProcessor: the notebook preprocessor.

**filter\_magic\_commands** (src)

Filter out the source commands with magic pattern.

**Parameters** **src** – the source commands.

**Returns** filtered: the filtered commands list.

**preprocess\_cell** (*cell, resources, index*)  
preprocessor that includes cells

**Param** cell: the notebook cell.

**Param** resources: the code source of the notebook cell.

**Param** index: unused argumnet.

**Returns** cell,resources: the notebook cell and its filtered with magic pattern commands.

### kubeflow.fairing.preprocessors.full\_notebook module

**class** kubeflow.fairing.preprocessors.full\_notebook.**FullNotebookPreProcessor** (*notebook\_file=None, out-put\_file='fairing\_out', in-put\_files=None, command=None, path\_prefix='/app/', out-put\_map=None*)

Bases: *kubeflow.fairing.preprocessors.base.BasePreProcessor*

The Full notebook preprocess for the context which comes from BasePreProcessor. :param BasePreProcessor: a context that gets sent to the builder for the docker build and sets the entrypoint

**set\_default\_executable** ()

Ingore the default executable setting for the full\_notebook preprocessor.

### kubeflow.fairing.preprocessors.function module

**class** kubeflow.fairing.preprocessors.function.**FunctionPreProcessor** (*function\_obj, path\_prefix='/app/', out-put\_map=None, in-put\_files=None*)

Bases: *kubeflow.fairing.preprocessors.base.BasePreProcessor*

FunctionPreProcessor preprocesses a single function. It sets as the command a function\_shim that calls the function directly. :param BasePreProcessor: a context that gets sent to the builder for the docker build and sets the entrypoint.

**get\_command** ()

Get the execute python command. :returns: command: the command line will be executed

## Module contents

## 1.2 Submodules

### 1.3 kubeflow.fairing.config module

```
class kubeflow.fairing.config.Config
    Bases: object

    deploy (pod_spec)
        deploy the job

        Parameters pod_spec – pod spec of the job

    fn (fn)
        function

        Parameters fn – return func that set the preprocessorr and run

    get_builder (preprocessor)
        get the builder

        Parameters preprocessor – preprocessor function

    get_deployer ()
        get deployer

    get_preprocessor ()
        get the preprocessor

    reset ()
        reset the preprocessor, builder_name and deployer name

    run ()
        run the pipeline for job

    set_builder (name='append', **kwargs)
        set the builder

        Parameters name – builder name (Default value = DEFAULT_BUILDER)

    set_deployer (name='job', **kwargs)
        set the deployer

        Parameters name – deployer name (Default value = DEFAULT_DEPLOYER)

    set_preprocessor (name=None, **kwargs)
        Parameters name – preprocessor name(Default value = None)
```

### 1.4 kubeflow.fairing.http\_utils module

```
kubeflow.fairing.http_utils.configure_http_instance (http=None)
    Configure http instance to modify the request headers to append or modify user-agent.

    Parameters http – Body of googleapiclient (Default value = None)

    Returns object: Configured http contents.
```

## 1.5 kubeflow.fairing.runtime\_config module

```
class kubeflow.fairing.runtime_config.RuntimeConfig
    Bases: object

    A passthrough config shim that runs in the fairing runtime

    fn (func)

    get_builder()

    get_deployer (**kwargs)

    get_preprocessor()

    reset()

    run()

    set_builder (name, **kwargs)

    set_deployer (name, **kwargs)

    set_preprocessor (name, **kwargs)
```

## 1.6 kubeflow.fairing.utils module

```
kubeflow.fairing.utils.crc (file_name)
    Compute a running Cyclic Redundancy Check checksum.

    Parameters file_name – The file name that's for crc checksum.

kubeflow.fairing.utils.get_current_k8s_namespace()
    Get the current namespace of kubernetes.

kubeflow.fairing.utils.get_default_target_namespace()
    Get the default target namespace, if running in the kubernetes cluster, will be current namespace, Otherwiase,
    will be “default”.

kubeflow.fairing.utils.get_image (repository, name)
    Get the full image name by integrating repository and image name.

    Parameters

    • repository – The name of repository

    • name – The short image name

    Returns str: Full image name, format: repo/name.

kubeflow.fairing.utils.is_running_in_k8s()
    Check if running in the kubernetes cluster.

kubeflow.fairing.utils.random_tag()
    Get a random tag.
```

## 1.7 Module contents

- [genindex](#)
- [modindex](#)

### k

kubeflow.fairing, 32  
kubeflow.fairing.backends, 7  
kubeflow.fairing.backends.backends, 3  
kubeflow.fairing.builders, 12  
kubeflow.fairing.builders.append, 8  
kubeflow.fairing.builders.append.append, 8  
kubeflow.fairing.builders.base\_builder, 11  
kubeflow.fairing.builders.builder, 12  
kubeflow.fairing.builders.cluster, 11  
kubeflow.fairing.builders.cluster.azurestorage\_context, 9  
kubeflow.fairing.builders.cluster.cluster, 9  
kubeflow.fairing.builders.cluster.context\_source, 9  
kubeflow.fairing.builders.cluster.gcs\_context, 10  
kubeflow.fairing.builders.cluster.s3\_context, 10  
kubeflow.fairing.builders.docker, 11  
kubeflow.fairing.builders.docker.docker, 11  
kubeflow.fairing.builders.dockerfile, 12  
kubeflow.fairing.cloud, 15  
kubeflow.fairing.cloud.aws, 12  
kubeflow.fairing.cloud.azure, 13  
kubeflow.fairing.cloud.docker, 14  
kubeflow.fairing.cloud.gcp, 14  
kubeflow.fairing.cloud.storage, 15  
kubeflow.fairing.config, 31  
kubeflow.fairing.constants, 15  
kubeflow.fairing.constants.constants, 15  
kubeflow.fairing.deployers, 20  
kubeflow.fairing.deployers.deployer, 20  
kubeflow.fairing.deployers.gcp, 16  
kubeflow.fairing.deployers.gcp.gcp, 15  
kubeflow.fairing.deployers.gcp.gcpserving, 16  
kubeflow.fairing.deployers.job, 17  
kubeflow.fairing.deployers.job.job, 16  
kubeflow.fairing.deployers.kfserving, 18  
kubeflow.fairing.deployers.kfserving.kfserving, 18  
kubeflow.fairing.deployers.serving, 19  
kubeflow.fairing.deployers.serving.serving, 19  
kubeflow.fairing.deployers.tfjob, 20  
kubeflow.fairing.deployers.tfjob.tfjob, 19  
kubeflow.fairing.frameworks, 22  
kubeflow.fairing.frameworks.lightgbm, 20  
kubeflow.fairing.frameworks.lightgbm\_dist\_training, 21  
kubeflow.fairing.frameworks.utils, 21  
kubeflow.fairing.functions, 23  
kubeflow.fairing.functions.function\_shim, 22  
kubeflow.fairing.http\_utils, 31  
kubeflow.fairing.kubernetes, 25  
kubeflow.fairing.kubernetes.manager, 23  
kubeflow.fairing.kubernetes.utils, 25  
kubeflow.fairing.ml\_tasks, 27  
kubeflow.fairing.ml\_tasks.tasks, 26  
kubeflow.fairing.ml\_tasks.utils, 27  
kubeflow.fairing.notebook, 27  
kubeflow.fairing.notebook.notebook\_util, 27  
kubeflow.fairing.preprocessors, 31  
kubeflow.fairing.preprocessors.base, 27  
kubeflow.fairing.preprocessors.converted\_notebook, 28  
kubeflow.fairing.preprocessors.full\_notebook,

[30](#)  
kubeflow.fairing.preprocessors.function,  
[30](#)  
kubeflow.fairing.runtime\_config,[32](#)  
kubeflow.fairing.utils,[32](#)

## A

`add_acr_config()` (in module `kubeflow.fairing.cloud.azure`), 14  
`add_aws_credentials()` (in module `kubeflow.fairing.cloud.aws`), 13  
`add_aws_credentials_if_exists()` (in module `kubeflow.fairing.cloud.aws`), 13  
`add_azure_files()` (in module `kubeflow.fairing.cloud.azure`), 14  
`add_docker_credentials()` (in module `kubeflow.fairing.cloud.docker`), 14  
`add_docker_credentials_if_exists()` (in module `kubeflow.fairing.cloud.docker`), 14  
`add_ecr_config()` (in module `kubeflow.fairing.cloud.aws`), 13  
`add_gcp_credentials()` (in module `kubeflow.fairing.cloud.gcp`), 14  
`add_gcp_credentials_if_exists()` (in module `kubeflow.fairing.cloud.gcp`), 14  
`AppendBuilder` (class in `kubeflow.fairing.builders.append.append`), 8  
`AWSBackend` (class in `kubeflow.fairing.backends.backends`), 3  
`AzureBackend` (class in `kubeflow.fairing.backends.backends`), 4  
`AzureFileUploader` (class in `kubeflow.fairing.cloud.azure`), 13

## B

`BackendInterface` (class in `kubeflow.fairing.backends.backends`), 4  
`BaseBuilder` (class in `kubeflow.fairing.builders.base_builder`), 11  
`BasePreProcessor` (class in `kubeflow.fairing.preprocessors.base`), 27  
`BaseTask` (class in `kubeflow.fairing.ml_tasks.tasks`), 26  
`build()` (`kubeflow.fairing.builders.append.append.AppendBuilder` method), 8  
`build()` (`kubeflow.fairing.builders.base_builder.BaseBuilder`

`method`), 11  
`build()` (`kubeflow.fairing.builders.builder.BuilderInterface` method), 12  
`build()` (`kubeflow.fairing.builders.cluster.cluster.ClusterBuilder` method), 9  
`build()` (`kubeflow.fairing.builders.docker.docker.DockerBuilder` method), 11  
`BuilderInterface` (class in `kubeflow.fairing.builders.builder`), 12

## C

`call()` (in module `kubeflow.fairing.functions.function_shim`), 22  
`CLASS` (`kubeflow.fairing.functions.function_shim.ObjectType` attribute), 22  
`cleanup()` (`kubeflow.fairing.builders.cluster.azurestorage_context.StorageContextSource` method), 9  
`cleanup()` (`kubeflow.fairing.builders.cluster.context_source.ContextSource` method), 10  
`cleanup()` (`kubeflow.fairing.builders.cluster.gcs_context.GCSContextSource` method), 10  
`cleanup()` (`kubeflow.fairing.builders.cluster.s3_context.S3ContextSource` method), 10  
`ClusterBuilder` (class in `kubeflow.fairing.builders.cluster.cluster`), 9  
`compare_version()` (in module `kubeflow.fairing.functions.function_shim`), 22  
`Config` (class in `kubeflow.fairing.config`), 31  
`configure_http_instance()` (in module `kubeflow.fairing.http_utils`), 31  
`context_map()` (`kubeflow.fairing.preprocessors.base.BasePreProcessor` method), 27  
`context_tar_gz()` (`kubeflow.fairing.preprocessors.base.BasePreProcessor` method), 27  
`ContextSourceInterface` (class in `kubeflow.fairing.builders.cluster.context_source`), 9

ConvertNotebookPreprocessor (class in kube-flow.fairing.preprocessors.converted\_notebook), 28  
 ConvertNotebookPreprocessorWithFire (class in kube-flow.fairing.preprocessors.converted\_notebook), 28  
 copy\_cmd() (kube-flow.fairing.cloud.storage.GCSSStorage method), 15  
 copy\_cmd() (kube-flow.fairing.cloud.storage.Storage method), 15  
 crc() (in module kube-flow.fairing.utils), 32  
 create() (kube-flow.fairing.ml\_tasks.tasks.PredictionEndpoint method), 26  
 create\_bucket\_if\_not\_exists() (kube-flow.fairing.cloud.aws.S3Uploader method), 12  
 create\_deployment() (kube-flow.fairing.kubernetes.manager.KubeManager method), 23  
 create\_docker\_secret() (in module kube-flow.fairing.cloud.docker), 14  
 create\_ecr\_registry() (in module kube-flow.fairing.cloud.aws), 13  
 create\_isvc() (kube-flow.fairing.kubernetes.manager.KubeManager method), 23  
 create\_job() (kube-flow.fairing.kubernetes.manager.KubeManager method), 23  
 create\_request\_dict() (kube-flow.fairing.deployers.gcp.gcp.GCPJob method), 15  
 create\_resource() (kube-flow.fairing.deployers.job.job.Job method), 16  
 create\_resource() (kube-flow.fairing.deployers.tfjob.tfjob.TfJob method), 19  
 create\_secret() (kube-flow.fairing.kubernetes.manager.KubeManager method), 23  
 create\_share\_if\_not\_exists() (kube-flow.fairing.cloud.azure.AzureFileUploader method), 13  
 create\_storage\_account\_if\_not\_exists() (kube-flow.fairing.cloud.azure.AzureFileUploader method), 14  
 create\_storage\_creds\_secret() (in module kube-flow.fairing.cloud.azure), 14  
 create\_tf\_job() (kube-flow.fairing.kubernetes.manager.KubeManager method), 23

## D

delete() (kube-flow.fairing.deployers.serving.serving.Serving method), 19  
 delete() (kube-flow.fairing.ml\_tasks.tasks.PredictionEndpoint method), 26  
 delete\_deployment() (kube-flow.fairing.kubernetes.manager.KubeManager method), 24  
 delete\_isvc() (kube-flow.fairing.kubernetes.manager.KubeManager method), 24  
 delete\_job() (kube-flow.fairing.kubernetes.manager.KubeManager method), 24  
 delete\_storage\_creds\_secret() (in module kube-flow.fairing.cloud.azure), 14  
 delete\_tf\_job() (kube-flow.fairing.kubernetes.manager.KubeManager method), 24  
 delete\_uncompressed\_files() (kube-flow.fairing.cloud.azure.AzureFileUploader method), 14  
 deploy() (kube-flow.fairing.config.Config method), 31  
 deploy() (kube-flow.fairing.deployers.deployer.DeployerInterface method), 20  
 deploy() (kube-flow.fairing.deployers.gcp.gcp.GCPJob method), 16  
 deploy() (kube-flow.fairing.deployers.gcp.gcpserving.GCPServingDeploy method), 16  
 deploy() (kube-flow.fairing.deployers.job.job.Job method), 16  
 deploy() (kube-flow.fairing.deployers.kfserving.kfserving.KFServing method), 18  
 deploy() (kube-flow.fairing.deployers.serving.serving.Serving method), 19  
 DeployerInterface (class in kube-flow.fairing.deployers.deployer), 20  
 do\_cleanup() (kube-flow.fairing.deployers.job.job.Job method), 17  
 DockerBuilder (class in kube-flow.fairing.builders.docker.docker), 11

## E

execute() (in module kube-flow.fairing.frameworks.lightgbm), 20  
 exists() (kube-flow.fairing.cloud.storage.GCSSStorage method), 15  
 exists() (kube-flow.fairing.cloud.storage.Storage method), 15

## F

fairing\_runtime\_files() (kube-flow.fairing.preprocessors.base.BasePreProcessor method), 28



<code>filter_include_cell()</code>	( <i>kube-flow.fairing.preprocessors.converted_notebook.FilterIncludeCell</i> method), 29	<code>generate_pod_spec()</code>	( <i>kube-flow.fairing.builders.base_builder.BaseBuilder</i> method), 12
<code>filter_magic_commands()</code>	( <i>kube-flow.fairing.preprocessors.converted_notebook.FilterMagicCommands</i> method), 29	<code>generate_pod_spec()</code>	( <i>kube-flow.fairing.builders.builder.BuilderInterface</i> method), 12
<code>FilterIncludeCell</code>	(class in <i>kube-flow.fairing.preprocessors.converted_notebook</i> ), 29	<code>generate_pod_spec()</code>	( <i>kube-flow.fairing.builders.cluster.azurestorage_context.StorageContext</i> method), 9
<code>FilterMagicCommands</code>	(class in <i>kube-flow.fairing.preprocessors.converted_notebook</i> ), 29	<code>generate_pod_spec()</code>	( <i>kube-flow.fairing.builders.cluster.context_source.ContextSourceInterface</i> method), 10
<code>fn()</code>	( <i>kubeflow.fairing.config.Config</i> method), 31	<code>generate_pod_spec()</code>	( <i>kube-flow.fairing.builders.cluster.gcs_context.GCSContextSource</i> method), 10
<code>fn()</code>	( <i>kubeflow.fairing.runtime_config.RuntimeConfig</i> method), 32	<code>generate_pod_spec()</code>	( <i>kube-flow.fairing.builders.cluster.s3_context.S3ContextSource</i> method), 10
<code>full_image_name()</code>	( <i>kube-flow.fairing.builders.base_builder.BaseBuilder</i> method), 11	<code>generate_pod_template_spec()</code>	( <i>kube-flow.fairing.deployers.job.job.Job</i> method), 17
<code>FullNotebookPreProcessor</code>	(class in <i>kube-flow.fairing.preprocessors.full_notebook</i> ), 30	<code>generate_predictor_spec()</code>	( <i>kube-flow.fairing.deployers.kfserving.kfserving.KFServing</i> method), 18
<code>FUNCTION</code>	( <i>kubeflow.fairing.functions.function_shim.ObjectType</i> attribute), 22	<code>generate_service_spec()</code>	( <i>kube-flow.fairing.deployers.serving.serving.Serving</i> method), 19
<code>FunctionPreProcessor</code>	(class in <i>kube-flow.fairing.preprocessors.function</i> ), 30		

## G

<code>GCPJob</code>	(class in <i>kubeflow.fairing.deployers.gcp.gcp</i> ), 15	<code>get_azure_credentials()</code>	(in module <i>kube-flow.fairing.cloud.azure</i> ), 14
<code>GCPManagedBackend</code>	(class in <i>kube-flow.fairing.backends.backends</i> ), 5	<code>get_base_contanier()</code>	( <i>kube-flow.fairing.backends.backends.BackendInterface</i> method), 4
<code>GCPservingDeployer</code>	(class in <i>kube-flow.fairing.deployers.gcp.gcpserving</i> ), 16	<code>get_builder()</code>	( <i>kube-flow.fairing.backends.backends.AWSBackend</i> method), 3
<code>GCSContextSource</code>	(class in <i>kube-flow.fairing.builders.cluster.gcs_context</i> ), 10	<code>get_builder()</code>	( <i>kube-flow.fairing.backends.backends.AzureBackend</i> method), 4
<code>GCSStorage</code>	(class in <i>kubeflow.fairing.cloud.storage</i> ), 15	<code>get_builder()</code>	( <i>kube-flow.fairing.backends.backends.BackendInterface</i> method), 4
<code>GCSUploader</code>	(class in <i>kubeflow.fairing.cloud.gcp</i> ), 14	<code>get_builder()</code>	( <i>kube-flow.fairing.backends.backends.GCPManagedBackend</i> method), 5
<code>generate_context_files()</code>	(in module <i>kube-flow.fairing.frameworks.lightgbm</i> ), 20	<code>get_builder()</code>	( <i>kube-flow.fairing.backends.backends.GKEBackend</i> method), 6
<code>generate_deployment_spec()</code>	( <i>kube-flow.fairing.deployers.job.job.Job</i> method), 17	<code>get_builder()</code>	( <i>kube-flow.fairing.backends.backends.KubernetesBackend</i> method), 7
<code>generate_deployment_spec()</code>	( <i>kube-flow.fairing.deployers.serving.serving.Serving</i> method), 19	<code>get_builder()</code>	( <i>kubeflow.fairing.config.Config</i> method), 31
<code>generate_deployment_spec()</code>	( <i>kube-flow.fairing.deployers.tfjob.tfjob.TfJob</i> method), 19	<code>get_builder()</code>	( <i>kube-flow.fairing.runtime_config.RuntimeConfig</i> method), 18
<code>generate_isvc()</code>	( <i>kube-flow.fairing.deployers.kfserving.kfserving.KFServing</i> method), 18		

method), 32

get\_command() (kubeflow.fairing.preprocessors.base.BasePreProcessor method), 28

get\_command() (kubeflow.fairing.preprocessors.function.FunctionPreProcessor method), 30

get\_config\_value() (in module kubeflow.fairing.frameworks.utils), 21

get\_current\_k8s\_namespace() (in module kubeflow.fairing.utils), 32

get\_default\_docker\_registry() (in module kubeflow.fairing.cloud.gcp), 15

get\_default\_target\_namespace() (in module kubeflow.fairing.utils), 32

get\_deployer() (kubeflow.fairing.config.Config method), 31

get\_deployer() (kubeflow.fairing.runtime\_config.RuntimeConfig method), 32

get\_docker\_registry() (kubeflow.fairing.backends.backends.BackendInterface method), 4

get\_docker\_registry() (kubeflow.fairing.backends.backends.GCPManagedBackend method), 5

get\_docker\_registry() (kubeflow.fairing.backends.backends.GKEBackend method), 6

get\_docker\_secret() (in module kubeflow.fairing.cloud.docker), 14

get\_execution\_obj\_type() (in module kubeflow.fairing.functions.function\_shim), 22

get\_image() (in module kubeflow.fairing.utils), 32

get\_logs() (kubeflow.fairing.deployers.deployer.DeployerInterface method), 20

get\_logs() (kubeflow.fairing.deployers.gcp.gcp.GCPJob method), 16

get\_logs() (kubeflow.fairing.deployers.gcp.gcpserving.GCPServiceDeployer method), 16

get\_logs() (kubeflow.fairing.deployers.job.job.Job method), 17

get\_logs() (kubeflow.fairing.deployers.kfserving.kfserving.KFServingDeployer method), 18

get\_logs() (kubeflow.fairing.deployers.tfjob.tfjob.TfJob method), 19

get\_notebook\_name() (in module kubeflow.fairing.notebook.notebook\_util), 27

get\_or\_create\_bucket() (kubeflow.fairing.cloud.gcp.GCSUploader method), 14

get\_plain\_secret\_value() (in module kubeflow.fairing.cloud.azure), 14

get\_preprocessor() (kubeflow.fairing.config.Config method), 31

get\_preprocessor() (kubeflow.fairing.runtime\_config.RuntimeConfig method), 32

get\_resource\_mutator() (in module kubeflow.fairing.kubernetes.utils), 25

get\_service\_external\_endpoint() (kubeflow.fairing.kubernetes.manager.KubeManager method), 24

get\_serving\_deployer() (kubeflow.fairing.backends.backends.AWSBackend method), 3

get\_serving\_deployer() (kubeflow.fairing.backends.backends.BackendInterface method), 5

get\_serving\_deployer() (kubeflow.fairing.backends.backends.GCPManagedBackend method), 5

get\_serving\_deployer() (kubeflow.fairing.backends.backends.GKEBackend method), 6

get\_serving\_deployer() (kubeflow.fairing.backends.backends.KubernetesBackend method), 7

get\_storage\_class() (in module kubeflow.fairing.cloud.storage), 15

get\_storage\_credentials() (kubeflow.fairing.cloud.azure.AzureFileUploader method), 14

get\_training\_deployer() (kubeflow.fairing.backends.backends.AWSBackend method), 4

get\_training\_deployer() (kubeflow.fairing.backends.backends.BackendInterface method), 5

get\_training\_deployer() (kubeflow.fairing.backends.backends.GCPManagedBackend method), 5

get\_training\_deployer() (kubeflow.fairing.backends.backends.GKEBackend method), 6

get\_training\_deployer() (kubeflow.fairing.backends.backends.KubernetesBackend method), 7

GKEBackend (class in kubeflow.fairing.backends.backends), 6

guess\_account\_id() (in module kubeflow.fairing.cloud.aws), 13

guess\_preprocessor() (in module kubeflow.fairing.ml\_tasks.utils), 27

guess\_project\_name() (in module kubeflow.fairing.cloud.gcp), 15

## I

`init_lightgbm_env()` (in module `kubeflow.fairing.frameworks.utils`), 21

`is_acr_registry()` (in module `kubeflow.fairing.cloud.azure`), 14

`is_docker_daemon_exists()` (in module `kubeflow.fairing.ml_tasks.utils`), 27

`is_ecr_registry()` (in module `kubeflow.fairing.cloud.aws`), 13

`is_in_notebook()` (in module `kubeflow.fairing.notebook.notebook_util`), 27

`is_requirements_txt_file_present()` (`kubeflow.fairing.preprocessors.base.BasePreProcessor` method), 28

`is_running_in_k8s()` (in module `kubeflow.fairing.utils`), 32

## J

`Job` (class in `kubeflow.fairing.deployers.job.job`), 16

## K

`KFServing` (class in `kubeflow.fairing.deployers.kfserving.kfserving`), 18

`kubeflow.fairing` (module), 32

`kubeflow.fairing.backends` (module), 7

`kubeflow.fairing.backends.backends` (module), 3

`kubeflow.fairing.builders` (module), 12

`kubeflow.fairing.builders.append` (module), 8

`kubeflow.fairing.builders.append.append` (module), 8

`kubeflow.fairing.builders.base_builder` (module), 11

`kubeflow.fairing.builders.builder` (module), 12

`kubeflow.fairing.builders.cluster` (module), 11

`kubeflow.fairing.builders.cluster.azurestorage_context` (module), 9

`kubeflow.fairing.builders.cluster.cluster` (module), 9

`kubeflow.fairing.builders.cluster.context_source` (module), 9

`kubeflow.fairing.builders.cluster.gcs_context` (module), 10

`kubeflow.fairing.builders.cluster.s3_context` (module), 10

`kubeflow.fairing.builders.docker` (module), 11

`kubeflow.fairing.builders.docker.docker` (module), 11

`kubeflow.fairing.builders.dockerfile` (module), 12

`kubeflow.fairing.cloud` (module), 15

`kubeflow.fairing.cloud.aws` (module), 12

`kubeflow.fairing.cloud.azure` (module), 13

`kubeflow.fairing.cloud.docker` (module), 14

`kubeflow.fairing.cloud.gcp` (module), 14

`kubeflow.fairing.cloud.storage` (module), 15

`kubeflow.fairing.config` (module), 31

`kubeflow.fairing.constants` (module), 15

`kubeflow.fairing.constants.constants` (module), 15

`kubeflow.fairing.deployers` (module), 20

`kubeflow.fairing.deployers.deployer` (module), 20

`kubeflow.fairing.deployers.gcp` (module), 16

`kubeflow.fairing.deployers.gcp.gcp` (module), 15

`kubeflow.fairing.deployers.gcp.gcpserving` (module), 16

`kubeflow.fairing.deployers.job` (module), 17

`kubeflow.fairing.deployers.job.job` (module), 16

`kubeflow.fairing.deployers.kfserving` (module), 18

`kubeflow.fairing.deployers.kfserving.kfserving` (module), 18

`kubeflow.fairing.deployers.serving` (module), 19

`kubeflow.fairing.deployers.serving.serving` (module), 18

`kubeflow.fairing.deployers.tfjob` (module), 20

`kubeflow.fairing.deployers.tfjob.tfjob` (module), 19

`kubeflow.fairing.frameworks` (module), 22

`kubeflow.fairing.frameworks.lightgbm` (module), 20

`kubeflow.fairing.frameworks.lightgbm_dist_training` (module), 21

`kubeflow.fairing.frameworks.utils` (module), 21

`kubeflow.fairing.functions` (module), 23

`kubeflow.fairing.functions.function_shim` (module), 22

`kubeflow.fairing.http_utils` (module), 31

`kubeflow.fairing.kubernetes` (module), 25

`kubeflow.fairing.kubernetes.manager` (module), 23

`kubeflow.fairing.kubernetes.utils` (module), 25

[kubeflow.fairing.ml\\_tasks \(module\)](#), 27  
[kubeflow.fairing.ml\\_tasks.tasks \(module\)](#), 26  
[kubeflow.fairing.ml\\_tasks.utils \(module\)](#), 27  
[kubeflow.fairing.notebook \(module\)](#), 27  
[kubeflow.fairing.notebook.notebook\\_util \(module\)](#), 27  
[kubeflow.fairing.preprocessors \(module\)](#), 31  
[kubeflow.fairing.preprocessors.base \(module\)](#), 27  
[kubeflow.fairing.preprocessors.converted\\_notebook \(module\)](#), 28  
[kubeflow.fairing.preprocessors.full\\_notebook \(module\)](#), 30  
[kubeflow.fairing.preprocessors.function \(module\)](#), 30  
[kubeflow.fairing.runtime\\_config \(module\)](#), 32  
[kubeflow.fairing.utils \(module\)](#), 32  
[KubeflowAWSBackend \(class in \[kubeflow.fairing.backends.backends\]\(#\)\)](#), 6  
[KubeflowAzureBackend \(class in \[kubeflow.fairing.backends.backends\]\(#\)\)](#), 6  
[KubeflowBackend \(class in \[kubeflow.fairing.backends.backends\]\(#\)\)](#), 6  
[KubeflowGKEBackend \(class in \[kubeflow.fairing.backends.backends\]\(#\)\)](#), 7  
[KubeManager \(class in \[kubeflow.fairing.kubernetes.manager\]\(#\)\)](#), 23  
[KubernetesBackend \(class in \[kubeflow.fairing.backends.backends\]\(#\)\)](#), 7

## L

[load\\_properties\\_config\\_file\(\) \(in module \[kubeflow.fairing.frameworks.utils\]\(#\)\)](#), 21  
[log\(\) \(kubeflow.fairing.kubernetes.manager.KubeManager method\)](#), 24  
[lookup\\_storage\\_class\(\) \(in module \[kubeflow.fairing.cloud.storage\]\(#\)\)](#), 15

## M

[mounting\\_pvc\(\) \(in module \[kubeflow.fairing.kubernetes.utils\]\(#\)\)](#), 25

## N

[NOT\\_SUPPORTED \(kubeflow.fairing.functions.function\\_shim.ObjectType attribute\)](#), 22  
[nslookup\(\) \(in module \[kubeflow.fairing.frameworks.utils\]\(#\)\)](#), 21

## O

[ObjectType \(class in \[kubeflow.fairing.functions.function\\\_shim\]\(#\)\)](#), 22

## P

[parse\\_cluster\\_spec\\_env\(\) \(in module \[kubeflow.fairing.frameworks.utils\]\(#\)\)](#), 21  
[predict\\_nparray\(\) \(kubeflow.fairing.ml\\_tasks.tasks.PredictionEndpoint method\)](#), 26  
[PredictionEndpoint \(class in \[kubeflow.fairing.ml\\\_tasks.tasks\]\(#\)\)](#), 26  
[prepare\(\) \(kubeflow.fairing.builders.cluster.azurestorage\\_context.StorageBuilder method\)](#), 9  
[prepare\(\) \(kubeflow.fairing.builders.cluster.context\\_source.ContextSource method\)](#), 10  
[prepare\(\) \(kubeflow.fairing.builders.cluster.gcs\\_context.GCSContextSource method\)](#), 10  
[prepare\(\) \(kubeflow.fairing.builders.cluster.s3\\_context.S3ContextSource method\)](#), 10  
[preprocess\(\) \(kubeflow.fairing.preprocessors.base.BasePreProcessor method\)](#), 28  
[preprocess\(\) \(kubeflow.fairing.preprocessors.converted\\_notebook.ConvertNotebook method\)](#), 28  
[preprocess\(\) \(kubeflow.fairing.preprocessors.converted\\_notebook.ConvertNotebook method\)](#), 29  
[preprocess\\_cell\(\) \(kubeflow.fairing.preprocessors.converted\\_notebook.FilterIncludeCell method\)](#), 29  
[preprocess\\_cell\(\) \(kubeflow.fairing.preprocessors.converted\\_notebook.FilterMagicCommand method\)](#), 29  
[publish\(\) \(kubeflow.fairing.builders.docker.docker.DockerBuilder method\)](#), 11

## R

[random\\_tag\(\) \(in module \[kubeflow.fairing.utils\]\(#\)\)](#), 32  
[reset\(\) \(kubeflow.fairing.config.Config method\)](#), 31  
[reset\(\) \(kubeflow.fairing.runtime\\_config.RuntimeConfig method\)](#), 32  
[reset\\_tar\\_mtime\(\) \(in module \[kubeflow.fairing.preprocessors.base\]\(#\)\)](#), 28  
[run\(\) \(kubeflow.fairing.config.Config method\)](#), 31  
[run\(\) \(kubeflow.fairing.runtime\\_config.RuntimeConfig method\)](#), 32  
[RuntimeConfig \(class in \[kubeflow.fairing.runtime\\\_config\]\(#\)\)](#), 32

## S

[S3ContextSource \(class in \[kubeflow.fairing.builders.cluster.s3\\\_context\]\(#\)\)](#),

10

S3Uploader (class in `kubeflow.fairing.cloud.aws`), 12

save\_properties\_config\_file() (in module `kubeflow.fairing.frameworks.utils`), 21

scrub\_fields() (in module `kubeflow.fairing.frameworks.utils`), 21

secret\_exists() (kubeflow.fairing.kubernetes.manager.KubeManager method), 25

Serving (class in `kubeflow.fairing.deployers.serving.serving`), 18

set\_annotatons() (kubeflow.fairing.deployers.job.job.Job method), 17

set\_builder() (kubeflow.fairing.config.Config method), 31

set\_builder() (kubeflow.fairing.runtime\_config.RuntimeConfig method), 32

set\_container\_name() (kubeflow.fairing.deployers.tfjob.tfjob.TfJob method), 19

set\_default\_executable() (kubeflow.fairing.preprocessors.base.BasePreProcessor method), 28

set\_default\_executable() (kubeflow.fairing.preprocessors.full\_notebook.FullNotebookPreProcessor method), 30

set\_deployer() (kubeflow.fairing.config.Config method), 31

set\_deployer() (kubeflow.fairing.runtime\_config.RuntimeConfig method), 32

set\_labels() (kubeflow.fairing.deployers.job.job.Job method), 17

set\_labels() (kubeflow.fairing.deployers.kfserving.kfserving.KFServing method), 18

set\_preprocessor() (kubeflow.fairing.config.Config method), 31

set\_preprocessor() (kubeflow.fairing.runtime\_config.RuntimeConfig method), 32

Storage (class in `kubeflow.fairing.cloud.storage`), 15

StorageContextSource (class in `kubeflow.fairing.builders.cluster.azurestorage_context`), 9

submit() (kubeflow.fairing.ml\_tasks.tasks.TrainJob method), 26

timed\_push() (kubeflow.fairing.builders.append.append.AppendBuilder method), 8

TrainJob (class in `kubeflow.fairing.ml_tasks.tasks`), 26

## U

uncompress\_tar\_gz\_file() (kubeflow.fairing.cloud.azure.AzureFileUploader method), 14

update\_config\_file() (in module `kubeflow.fairing.frameworks.utils`), 21

upload\_context() (kubeflow.fairing.builders.cluster.azurestorage\_context.StorageContext method), 9

upload\_context() (kubeflow.fairing.builders.cluster.gcs\_context.GCSContextSource method), 10

upload\_context() (kubeflow.fairing.builders.cluster.s3\_context.S3ContextSource method), 10

upload\_tar\_gz\_contents() (kubeflow.fairing.cloud.azure.AzureFileUploader method), 14

upload\_to\_bucket() (kubeflow.fairing.cloud.aws.S3Uploader method), 13

upload\_to\_bucket() (kubeflow.fairing.cloud.gcp.GCSUploader method), 14

upload\_to\_share() (kubeflow.fairing.cloud.azure.AzureFileUploader method), 14

## W

write\_dockerfile() (in module `kubeflow.fairing.builders.dockerfile`), 12

write\_ip\_list\_file() (in module `kubeflow.fairing.frameworks.utils`), 22

## T

TfJob (class in `kubeflow.fairing.deployers.tfjob.tfjob`), 19