crumbs Documentation

Release 2.1.1

Alex Brandt

Sep 30, 2018

Contents

Generalized all-in-one parameters. Get configuration through a single interface from configuration files, environment variables, and command line arguments.

CHAPTER 1

Getting Started

CHAPTER 2

crumbs — Parameters

class crumbs.**Parameters**(**args*, ***kwargs*)

Queryable collection of parameters whose values are set by the user.

Using the normal dictionary lookup mechanism (D[]), one can obtain the user set values for particular parameters (set in command line arguments, configuration files, or environment variables) which have been added to `Parameters`.

The three sources (command line arguments, configuration files, and environment variables) can have values for parameters. When two or more sources define values for the same parameter, the first source (highest precedence) in the following list will provide the value for that parameter:

Command line arguments Values parsed from `sys.argv`.

Configuration files Values set in registered `ini` files.

Environment variables Values set in environment variables.

Defaults Default value (if set for the parameter).

Parameters are added via the `add_parameter` method. Configuration files that should be searched can be added with the `add_configuration_file` method. Environment variables are prefixed with an uppercase program name (`sys.argv[0]`) and uppercased with dots '.' and hyphens '-' replaced with underscores (i.e. ARGV0_GROUP_LONG_OPTION where group may be ommitted if it is 'default').

Before querying for parameters' values, the `Parameters` object must have been parsed with the `parse` method. Parsing ensures that the command line, configuration files, and environment variables are read and ready to be queried.

Methods

__getitem__ Return a parameter's value.

__init__ Initialize and return a `Parameters` object.

Add_configuration_file Add a file path to be searched for parameter values.

Add_parameter Add a parameter to `Parameters` object.

Parse Prepare `Parameters` for queries and ensure parameter values can be found.

Read_configuration_files Read all configuration files' values.

Properties

Defaults Dictionary mapping parameter name to default value. Default: {}

Parameters Dictionary mapping parameter name to parameter arguments (arguments passed to `add_parameter`). Default: {}.

Grouped_parameters Dictionary mapping parameter group to parameter dictionary (see parameters property). Default: { 'default': {} }.

Configuration_files Dictionary mapping configuration file path to an active `ConfigParser.ConfigParser`. Default: {}.

Groups Set of all parameter groups. Always includes at least the 'default' group. Default: set(['default']).

Parsed True if `Parameters` has been parsed with the `parse` method; otherwise, False. Default: False.

Example

Basic `Parameters` usage has four stages:

1. Instantiate `Parameters` (cf. `__init__`):

```
p = Parameters()
```

2. Add parameters (cf. `add_parameter`):

```
p.add_parameter(options = [ '--foo' ])
```

3. Parse (cf. `parse`):

```
p.parse()
```

4. Query (cf. `__getitem__`):

```
p['foo']
```

add_configuration_file(*file_name*)
Register a file path from which to read parameter values.

This method can be called multiple times to register multiple files for querying. Files are expected to be `ini` formatted.

No assumptions should be made about the order that the registered files are read and values defined in multiple files may have unpredictable results.

Arguments

File_name Name of the file to add to the parameter search.

add_parameter(***kwargs*)
Add the parameter to `Parameters`.

Arguments

The arguments are lumped into two groups:`Parameters.add_parameter` and `argparse.ArgumentParser.add_argument`. Parameters that are only used by `Parameters.add_parameter` are removed before `kwargs` is passed directly to argparse.ArgumentParser.add_argument''.

Note: Once `parse` has been called `Parameters.parsed` will be True and it is inadvisable to add more parameters to the `Parameters`.

``Parameters.add_parameter`` Arguments

Environment_prefix Prefix to add when searching the environment for this parameter. Default: os.path.basename(sys.argv[0]).

Group Group (namespace or prefix) for parameter (corresponds to section name in configuration files). Default: 'default'.

Options REQUIRED. The list of options to match for this parameter in argv.

Only Iterable containing the components that this parameter applies to (i.e. 'environment', 'configuration', 'argument'). Default: ('environment', 'configuration', 'argument').

``argparse.ArgumentParser.add_argument`` Arguments

Name or flags Positional argument filled in by options keyword argument.

Action The basic type of action to be taken when this argument is encountered at the command line.

Nargs The number of command-line arguments that should be consumed.

Const A constant value required by some action and nargs selections.

Default The value produced if the argument is absent from the command line.

Type The type to which the command-line argument should be converted.

Choices A container of the allowable values for the argument.

Required Whether or not the command-line option may be omitted (optionals only).

Help A brief description of what the argument does.

Metavar A name for the argument in usage messages.

Dest The name of the attribute to be added to the object returned by parse_args().

parse(*only_known=False*)

Ensure all sources are ready to be queried.

Parses `sys.argv` with the contained `argparse.ArgumentParser` and sets `parsed` to True if `only_known` is False. Once `parsed` is set to True, it is inadvisable to add more parameters (cf. `add_parameter`). Also, if `parsed` is not set to True, retrieving items (cf. `__getitem__`) will result in a warning that values are being retrieved from an uparsed Parameters.

Arguments

Only_known If True, do not error or fail when unknown parameters are encountered.

Note: If `only_known` is True, the `--help` and `-h` options on the command line (`sys.argv`) will be ignored during parsing as it is unexpected that these parameters' default behavior would be desired at this stage of execution.

read_configuration_files()

Explicitly read the configuration files.

Reads all configuration files in this Parameters object. Even if inotify is watching or a read has already occurred.

Note: The order that the configuration files are read is not guaranteed.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Index