commandIO Documentation

Release latest

Jan 21, 2023

Contents:

This library provides a simple way to expose any C/C++ function in a Read-eval-print loop (REPL) interactive environment.

Features:

- Easy interface definition.
- Interface support for: - Positional parameters. - Optional parameters with default values. - Flags.
- Class methods.
- Automatic parameter- and return type inference.
- Full help system.
- Method discovery.

CHAPTER 1

Examples

1.1 Demo

We show how to use simple functions in the demo program. In this program we export three functions: *greet*, *inc* and *mul*.

The built in *help* function shows a list of available commands.

```
> help
Available commands:
  greet          Say hi to someone.
  inc            Increment a value.
  mul            Multiply a floating point number.
  help           Help on a specific command.
  exit           Exit.
```

For more information about a specific command, pass the name of a command to the *help* function.

```
> help greet
greet: Say hi to someone.

positional arguments:
  name           someone's name (type string)

optional arguments:
  -t             greet multiple times (type int, default: 1)
  -s             shout (type flag)

returns:
  string
```

This particular command has one positional (mandatory) parameter and two optional parameters, of which one is a flag.

From the description, we see that we can call the *greet* function by providing only one argument as follows.

```
> greet world
Hi world.
```

Stings consisting of multiple words should be quoted.

```
> greet "Dan the man"
Hi Dan the man.
```

We can override the default value of the optional parameter by adding the *-t* option.

```
> greet -t 3 world
Hi world.
Hi world.
Hi world.
```

Flags do not take an additional argument.

```
> greet -t 3 -s world
HI world!
HI world!
HI world!
```

Optional arguments can be provided in any order.

```
> greet -s world -t 3
HI world!
HI world!
HI world!
```

1.2 Calculator

In the calculator program we show how to use class methods. In this program we export some simple arithmetic functions.

```
> help
name (return type: parameter types) ; documentation

add (void: int) ; Add something.
sub (void: int) ; Subtract something.
show (int:) ; Show result.
help (string:) ; This help message.
exit (void:) ; Exit.
```

These functions operate on an object.

```
$ ./calculator
> show
0
> add 10
> sub 2
> show
8
> exit
```

1.2.1 Contributors

- Jeroen F.J. Laros <jlaros@fixedpoint.nl> (Original author, maintainer)

Find out who contributed:

```
git shortlog -s -e
```