命令行工具开发

2019.7.19

JCBioinformatics-2019-Python

HZAU

教程主页: https://nanguage.github.io/JCBioinformatics-Python/

Contents

- 简介
- 常见概念
- argparse
- Click
- Fire

Command Line Interface

- 提供一个给 shell 用户的调用接口。
- 向脚本传递比较复杂的参数。
- 向用户提供详细的脚本使用说明。

Conceptions

- Arguments
  - positional
  - options　(--arg / -a)
    - flag
  - default value
  - type
  - number of args
  - help
- Sub-command
  - command group
- Documents
- Complete (tab 补全)

```
cutadapt -o output.fasta.gz input.fastq.gz
```

```
-> ~ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone      Clone a repository into a new directory
   init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add        Add file contents to the index
   mv         Move or rename a file, a directory, or a symlink
   reset      Reset current HEAD to the specified state
   rm         Remove files from the working tree and from the index
```

argparse

- 属于Python标准模块:
  - 文档
  - 意味着更容易移植

```
import argparse

parser = argparse.ArgumentParser(description='Process some integers.')
parser.add_argument('integers', metavar='N', type=int, nargs='+',
                    help='an integer for the accumulator')
parser.add_argument('--sum', dest='accumulate', action='store_const',
                    const=sum, default=max,
                    help='sum the integers (default: find the max)')

args = parser.parse_args()
print(args.accumulate(args.integers))
```

创建一个 parser，添加描述

添加 positional argument，接受多个 args

添加 一个 option

解析输入的 arguments，存入 args

帮助信息

```
$ python prog.py -h
usage: prog.py [-h] [--sum] N [N ...]

Process some integers.

positional arguments:
 N           an integer for the accumulator

optional arguments:
 -h, --help  show this help message and exit
 --sum       sum the integers (default: find the max)
```

使用示例

```
$ python prog.py 1 2 3 4
4

$ python prog.py 1 2 3 4 --sum
10
```

Click

- https://github.com/pallets/click/
- 装饰器语法，直接将函数包装为CLI

```python
import click

@click.command()
@click.option('--count', default=1, help='Number of greetings.')
@click.option('--name', prompt='Your name',
              help='The person to greet.')
def hello(count, name):
    """Simple program that greets NAME for a total of COUNT times."""
    for x in range(count):
        click.echo('Hello %s!' % name)

if __name__ == '__main__':
    hello()
```

```
$ python hello.py --count=3
Your name: John
Hello John!
Hello John!
Hello John!
```

```
$ python hello.py --help
Usage: hello.py [OPTIONS]

  Simple program that greets NAME for a total of COUNT times.

Options:
  --count INTEGER  Number of greetings.
  --name TEXT      The person to greet.
  --help           Show this message and exit.
```

Fire

- https://github.com/google/python-fire
- 根据对象生成CLI，使用非常简单
- 可以作为 debug 工具

```python
import fire

def add(x, y):
  return x + y

def multiply(x, y):
  return x * y

if __name__ == '__main__':
  fire.Fire()
```

```
$ python example.py add 10 20
30
$ python example.py multiply 10 20
200
```

```python
import fire

class BrokenCalculator(object):

  def __init__(self, offset=1):
      self._offset = offset

  def add(self, x, y):
    return x + y + self._offset

  def multiply(self, x, y):
    return x * y + self._offset

if __name__ == '__main__':
  fire.Fire(BrokenCalculator)
```

```
$ python example.py add 10 20 --offset=0
30
$ python example.py multiply 10 20 --offset=0
200
```

```python
class IngestionStage(object):

  def run(self):
    return 'Ingesting! Nom nom nom...'

class DigestionStage(object):

  def run(self, volume=1):
    return ' '.join(['Burp!'] * volume)

  def status(self):
    return 'Satiated.'

class Pipeline(object):

  def __init__(self):
    self.ingestion = IngestionStage()
    self.digestion = DigestionStage()

  def run(self):
    self.ingestion.run()
    self.digestion.run()

if __name__ == '__main__':
  fire.Fire(Pipeline)
```

```
$ python example.py run
Ingesting! Nom nom nom...
Burp!
$ python example.py ingestion run
Ingesting! Nom nom nom...
$ python example.py digestion run
Burp!
$ python example.py digestion status
Satiated.
```

docopt

- https://github.com/docopt/docopt
- 根据 Help(Docstring) 生成 CLI

```
"""Naval Fate.

Usage:
  naval_fate.py ship new <name>...
  naval_fate.py ship <name> move <x> <y> [--speed=<kn>]
  naval_fate.py ship shoot <x> <y>
  naval_fate.py mine (set|remove) <x> <y> [--moored | --drifting]
  naval_fate.py (-h | --help)
  naval_fate.py --version

Options:
  -h --help     Show this screen.
  --version     Show version.
  --speed=<kn>  Speed in knots [default: 10].
  --moored      Moored (anchored) mine.
  --drifting    Drifting mine.

"""
from docopt import docopt


if __name__ == '__main__':
    arguments = docopt(__doc__, version='Naval Fate 2.0')
    print(arguments)
```