1. **1. argparse介绍**

是python的一个命令行解析包，非常编写可读性非常好的程序

1. **2. 基本用法**

prog.py是我在linux下测试argparse的文件，放在/tmp目录下，其内容如下：

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser()

parser.parse\_args()

测试：

yarving@yarving-VirtualBox /tmp $ python prog.py

yarving@yarving-VirtualBox /tmp $ python prog.py --help

usage: prog.py [-h]

optional arguments:

-h, --help show this help message and exit

yarving@yarving-VirtualBox /tmp $ python prog.py -v

usage: prog.py [-h]

prog.py: error: unrecognized arguments: -v

yarving@yarving-VirtualBox /tmp $ python prog.py foo

usage: prog.py [-h]

prog.py: error: unrecognized arguments: foo

* 第一个没有任何输出和出错
* 第二个测试为打印帮助信息，argparse会自动生成帮助文档
* 第三个测试为未定义的-v参数，会出错
* 第四个测试为未定义的参数foo，出错

1. **3. positional arguments**

positional arguments为英文定义，中文名叫有翻译为定位参数的，用法是不用带-就可用

修改prog.py的内容如下：

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser()

parser.add\_argument("echo")

args = parser.parse\_args()

print args.echo

执行测试如下

yarving@yarving-VirtualBox /tmp $ python prog.py

usage: prog.py [-h] echo

prog.py: error: too few arguments

yarving@yarving-VirtualBox /tmp $ python prog.py -h

usage: prog.py [-h] echo

positional arguments:

echo

optional arguments:

-h, --help show this help message and exit

yarving@yarving-VirtualBox /tmp $ python prog.py hahahaha

hahahaha

定义了一个叫echo的参数，默认必选

* 第一个测试为不带参数，由于echo参数为空，所以报错，并给出用法（usage）和错误信息
* 第二个测试为打印帮助信息
* 第三个测试为正常用法，回显了输入字符串hahahaha

1. **4. optional arguments**

中文名叫可选参数，有两种方式：

1. 一种是通过一个-来指定的短参数，如-h；
2. 一种是通过--来指定的长参数，如--help

这两种方式可以同存，也可以只存在一个，修改prog.py内容如下：

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser()

parser.add\_argument("-v", "--verbosity", help="increase output verbosity")

args = parser.parse\_args()

if args.verbosity:

print "verbosity turned on"

注意这一行：parser.add\_argument("-v", "--verbosity", help="increase output verbosity")

定义了可选参数-v或--verbosity，通过解析后，其值保存在args.verbosity变量中

用法如下：

yarving@yarving-VirtualBox /tmp $ python prog.py -v 1

verbosity turned on

yarving@yarving-VirtualBox /tmp $ python prog.py --verbosity 1

verbosity turned on

yarving@yarving-VirtualBox /tmp $ python prog.py -h

usage: prog.py [-h] [-v VERBOSITY]

optional arguments:

-h, --help show this help message and exit

-v VERBOSITY, --verbosity VERBOSITY

increase output verbosity

yarving@yarving-VirtualBox /tmp $ python prog.py -v

usage: prog.py [-h] [-v VERBOSITY]

prog.py: error: argument -v/--verbosity: expected one argument

* 测试1中，通过-v来指定参数值
* 测试2中，通过--verbosity来指定参数值
* 测试3中，通过-h来打印帮助信息
* 测试4中，没有给-v指定参数值，所以会报错

1. **5. action='store\_true'**

上一个用法中-v必须指定参数值，否则就会报错，有没有像-h那样，不需要指定参数值的呢，答案是有，通过定义参数时指定action="store\_true"即可，用法如下

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser()

parser.add\_argument("-v", "--verbose", help="increase output verbosity",

action="store\_true")

args = parser.parse\_args()

if args.verbose:

print "verbosity turned on"

测试：

yarving@yarving-VirtualBox /tmp $ python prog.py -v

verbosity turned on

yarving@yarving-VirtualBox /tmp $ python prog.py -h

usage: prog.py [-h] [-v]

optional arguments:

-h, --help show this help message and exit

-v, --verbose increase output verbosity

第一个例子中，-v没有指定任何参数也可，其实存的是True和False，如果出现，则其值为True，否则为False

1. **6. 类型 type**

默认的参数类型为str，如果要进行数学计算，需要对参数进行解析后进行类型转换，如果不能转换则需要报错，这样比较麻烦

argparse提供了对参数类型的解析，如果类型不符合，则直接报错。如下是对参数进行平方计算的程序：

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser()

parser.add\_argument('x', type=int, help="the base")

args = parser.parse\_args()

answer = args.x \*\* 2

print answer

测试

yarving@yarving-VirtualBox /tmp $ python prog.py 2

4

yarving@yarving-VirtualBox /tmp $ python prog.py two

usage: prog.py [-h] x

prog.py: error: argument x: invalid int value: 'two'

yarving@yarving-VirtualBox /tmp $ python prog.py -h

usage: prog.py [-h] x

positional arguments:

x the base

optional arguments:

-h, --help show this help message and exit

* 第一个测试为计算2的平方数，类型为int，正常
* 第二个测试为一个非int数，报错
* 第三个为打印帮助信息

1. **7. 可选值choices=[]**

5中的action的例子中定义了默认值为True和False的方式，如果要限定某个值的取值范围，比如6中的整形，限定其取值范围为0， 1， 2，该如何进行呢？

修改prog.py文件如下：

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser()

parser.add\_argument("square", type=int,

help="display a square of a given number")

parser.add\_argument("-v", "--verbosity", type=int, choices=[0, 1, 2],

help="increase output verbosity")

args = parser.parse\_args()

answer = args.square\*\*2

if args.verbosity == 2:

print "the square of {} equals {}".format(args.square, answer)

elif args.verbosity == 1:

print "{}^2 == {}".format(args.square, answer)

else:

print answer

测试如下：

yarving@yarving-VirtualBox /tmp $ python prog.py 4 -v 0

16

yarving@yarving-VirtualBox /tmp $ python prog.py 4 -v 1

4^2 == 16

yarving@yarving-VirtualBox /tmp $ python prog.py 4 -v 2

the square of 4 equals 16

yarving@yarving-VirtualBox /tmp $ python prog.py 4 -v 3

usage: prog.py [-h] [-v {0,1,2}] square

prog.py: error: argument -v/--verbosity: invalid choice: 3 (choose from 0, 1, 2)

yarving@yarving-VirtualBox /tmp $ python prog.py -h

usage: prog.py [-h] [-v {0,1,2}] square

positional arguments:

square display a square of a given number

optional arguments:

-h, --help show this help message and exit

-v {0,1,2}, --verbosity {0,1,2}

increase output verbosity

* 测试1， 2， 3 为可选值范围，通过其值，打印不同的格式输出；
* 测试4的verbosity值不在可选值范围内，打印错误
* 测试5打印帮助信息

1. **8. 自定义帮助信息help**

上面很多例子中都为help赋值，如

parser.add\_argument("square", type=int, help="display a square of a given number")

在打印输出时，会有如下内容

positional arguments:

square display a square of a given number

也就是help为什么，打印输出时，就会显示什么

1. **9. 程序用法帮助**

8中介绍了为每个参数定义帮助文档，那么给整个程序定义帮助文档该怎么进行呢？

通过argparse.ArgumentParser(description="calculate X to the power of Y")即可

修改prog.py内容如下：

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser(description="calculate X to the power of Y")

group = parser.add\_mutually\_exclusive\_group()

group.add\_argument("-v", "--verbose", action="store\_true")

group.add\_argument("-q", "--quiet", action="store\_true")

parser.add\_argument("x", type=int, help="the base")

parser.add\_argument("y", type=int, help="the exponent")

args = parser.parse\_args()

answer = args.x\*\*args.y

if args.quiet:

print answer

elif args.verbose:

print "{} to the power {} equals {}".format(args.x, args.y, answer)

else:

print "{}^{} == {}".format(args.x, args.y, answer)

打印帮助信息时即显示*calculate X to the power of Y*

yarving@yarving-VirtualBox /tmp $ python prog.py -h

usage: prog.py [-h] [-v | -q] x y

calculate X to the power of Y

positional arguments:

x the base

y the exponent

optional arguments:

-h, --help show this help message and exit

-v, --verbose

-q, --quiet

1. **10. 互斥参数**

在上个例子中介绍了互斥的参数

group = parser.add\_mutually\_exclusive\_group()

group.add\_argument("-v", "--verbose", action="store\_true")

group.add\_argument("-q", "--quiet", action="store\_true")

第一行定义了一个互斥组，第二、三行在互斥组中添加了-v和-q两个参数，用上个例子中的程序进行如下测试：

yarving@yarving-VirtualBox /tmp $ python prog.py 4 2

4^2 == 16

yarving@yarving-VirtualBox /tmp $ python prog.py 4 2 -v

4 to the power 2 equals 16

yarving@yarving-VirtualBox /tmp $ python prog.py 4 2 -q

16

yarving@yarving-VirtualBox /tmp $ python prog.py 4 2 -q -v

可以看出，-q和-v不出现，或仅出现一个都可以，同时出现就会报错。

可定义多个互斥组

1. **11.参数默认值**

介绍了这么多，有没有参数默认值该如何定义呢？

修改prog.py内容如下：

#!/usr/bin/env python

# encoding: utf-8

import argparse

parser = argparse.ArgumentParser(description="calculate X to the power of Y")

parser.add\_argument("square", type=int,

help="display a square of a given number")

parser.add\_argument("-v", "--verbosity", type=int, choices=[0, 1, 2], default=1,

help="increase output verbosity")

args = parser.parse\_args()

answer = args.square\*\*2

if args.verbosity == 2:

print "the square of {} equals {}".format(args.square, answer)

elif args.verbosity == 1:

print "{}^2 == {}".format(args.square, answer)

else:

print answer

测试结果如下

yarving@yarving-VirtualBox /tmp $ python prog.py 8

8^2 == 64

yarving@yarving-VirtualBox /tmp $ python prog.py 8 -v 0

64

yarving@yarving-VirtualBox /tmp $ python prog.py 8 -v 1

8^2 == 64

yarving@yarving-VirtualBox /tmp $ python prog.py 8 -v 2

the square of 8 equals 64

可以看到如果不指定-v的值，args.verbosity的值默认为1，为了更清楚的看到默认值，也可以直接打印进行测试。

作者：yarving  
链接：https://www.jianshu.com/p/fef2d215b91d  
來源：简书  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。