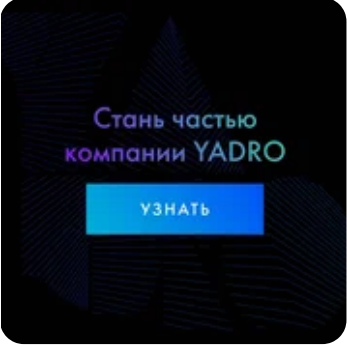


# Модуль cmd в Python, создание интерактивных оболочек



oneweekoffer.yadro.com

РЕКЛАМА

## Ищем ведущих программистов в команду YADRO.

Ждем амбициозных и талантливых, которые горят инновационными идеями  
Заполни анкету

Узнать больше

[Стандартная библиотека Python3.](#) / Модуль cmd в Python, создание интерактивных оболочек

## Программирование строчно-ориентированных интерпретаторов команд

[Модуль cmd](#) обеспечивает простую структуру для написания строчно-ориентированных интерпретаторов команд. Они часто полезны для тестирования, инструментов администрирования и прототипов программ, которые позже будут обернуты в более сложный интерфейс.

Проще говоря этот модуль позволяет с минимальными затратами написать подобие интерфейса командной строки с необходимым набором команд.

## Пример использования модуля cmd:

Модуль cmd в основном полезен для создания оболочек, которые позволяют пользователю работать с программой в интерактивном режиме.

Здесь представлен простой пример того, как построить оболочку вокруг нескольких команд модуля [turtle](#).

Базовые команды модуля turtle, такие как turtle.forward(), добавляются в подкласс [cmd.Cmd\(\)](#) с методом do\_forward() (смотри код ниже). Аргумент преобразуется в число и отправляется модулю turtle. Строка документации используется в служебной программе, предоставляемой оболочкой.

Пример также включает базовое средство записи и воспроизведения, реализованное с помощью [метода Cmd.precmd\(\)](#), который отвечает за преобразование ввода в нижний регистр и запись команд в файл. Метод do\_playback() (смотри код ниже) считывает файл и добавляет записанные команды в cmdqueue для немедленного воспроизведения:

```
import cmd, sys
from turtle import *

class TurtleShell(cmd.Cmd):
    intro = 'Welcome to the turtle shell.  Type help or ? to list commands.\n'
    prompt = '(turtle) '
    file = None

    # ----- basic turtle commands -----
    def do_forward(self, arg):
        'Move the turtle forward by the specified distance:  FORWARD 10'
        forward(*parse(arg))
    def do_right(self, arg):
        'Turn turtle right by given number of degrees:  RIGHT 20'
        right(*parse(arg))
    def do_left(self, arg):
        'Turn turtle left by given number of degrees:  LEFT 90'
        left(*parse(arg))
    def do_goto(self, arg):
        'Move turtle to an absolute position with changing orientation.  GOTO 100 200'
        goto(*parse(arg))
    def do_home(self, arg):
        'Return turtle to the home position:  HOME'
        home()
    def do_circle(self, arg):
        'Draw circle with given radius an options extent and steps:  CIRCLE 50'
        circle(*parse(arg))
```

```

def do_position(self, arg):
    'Print the current turtle position:  POSITION'
    print('Current position is %d %d\n' % position())
def do_heading(self, arg):
    'Print the current turtle heading in degrees:  HEADING'
    print('Current heading is %d\n' % (heading(),))
def do_color(self, arg):
    'Set the color:  COLOR BLUE'
    color(arg.lower())
def do_undo(self, arg):
    'Undo (repeatedly) the last turtle action(s):  UNDO'
def do_reset(self, arg):
    'Clear the screen and return turtle to center:  RESET'
    reset()
def do_bye(self, arg):
    'Stop recording, close the turtle window, and exit:  BYE'
    print('Thank you for using Turtle')
    self.close()
    bye()
    return True

# ----- record and playback -----
def do_record(self, arg):
    'Save future commands to filename:  RECORD rose.cmd'
    self.file = open(arg, 'w')
def do_playback(self, arg):
    'Playback commands from a file:  PLAYBACK rose.cmd'
    self.close()
    with open(arg) as f:
        self.cmdqueue.extend(f.read().splitlines())
def precmd(self, line):
    line = line.lower()
    if self.file and 'playback' not in line:
        print(line, file=self.file)
    return line
def close(self):
    if self.file:
        self.file.close()
        self.file = None

def parse(arg):
    'Convert a series of zero or more numbers to an argument tuple'
    return tuple(map(int, arg.split()))

if __name__ == '__main__':
    TurtleShell().cmdloop()

```

Вот пример сеанса turtle shell, показывающий функции справки, использование пустых строк для повторения команд и простое средство записи и воспроизведения:

```
Welcome to the turtle shell.  Type help or ? to list commands.
```

```
(turtle) ?
```

```
Documented commands (type help <topic>):
```

```
=====
```

```
bye      color      goto      home      playback  record  right
circle   forward    heading   left      position  reset   undo
```

```
(turtle) help forward
```

```
Move the turtle forward by the specified distance:  FORWARD 10
```

```
(turtle) record spiral.cmd
```

```
(turtle) position
```

```
Current position is 0 0
```

```
(turtle) heading
```

```
Current heading is 0
```


```
(turtle) reset
(turtle) circle 20
(turtle) right 30
(turtle) circle 40
(turtle) right 30
(turtle) circle 60
(turtle) right 30
(turtle) circle 80
(turtle) right 30
(turtle) circle 100
(turtle) right 30
(turtle) circle 120
(turtle) right 30
(turtle) circle 120
(turtle) heading
Current heading is 180


(turtle) forward 100
(turtle)
(turtle) right 90
(turtle) forward 100
(turtle)
(turtle) right 90
(turtle) forward 400
(turtle) right 90
(turtle) forward 500
(turtle) right 90
(turtle) forward 400
(turtle) right 90
(turtle) forward 300
(turtle) playback spiral.cmd
Current position is 0 0


Current heading is 0

Current heading is 180


(turtle) bye
Thank you for using Turtle
```



РЕКЛАМА 



ГАЗПРОМБАНК



Подробнее

gazprombank.ru

Подать заявку


Реклама


Содержание раздела:

- [КРАТКИЙ ОБЗОР МАТЕРИАЛА.](#)
- [Класс Cmd\(\) модуля cmd](#)
- [Атрибуты и методы объекта Cmd модуля cmd](#)

ХОЧУ ПОМОЧЬ  
ПРОЕКТУ

РЕКЛАМА · 18+ ⓘ



 practicum.yandex.ru

**Бесплатное  
занятие  
английским  
в Яндекс  
Практикуме**

Полноценное занятие  
с преподавателем, а не  
презентация курсов

Устный тест на уровень  
языка

>

Практика английского

>

Узнать больше