



ООП. Полезные дополнения

Основы Python

Что будет на уроке

1. Статические методы и методы класса.
2. Атрибуты и встроенные методы объектов класса.
3. Пример ООП-программы.
4. Создание собственных исключений.
5. Библиотека `psutil`.
6. `Pip` и `virtualenv`. Особенности использования.
7. Библиотека `requests`.

Статические методы и методы класса

@staticmethod

Декоратор для метода класса.

Такой метод вызывается
напрямую через имя класса
(статический метод).

@classmethod

Декоратор для метода класса.

Такой метод получает класс в
качестве первого аргумента.

Атрибуты и встроенные методы объектов классов.

Часть 1

Атрибут	Описание
<code>__name__</code>	Имя класса
<code>__module__</code>	Имя модуля
<code>__dict__</code>	Словарь с атрибутами класса
<code>__bases__</code>	Кортеж с базовыми классами
<code>__doc__</code>	Строка документации класса
<code>__class__</code>	Объект-класс, экземпляром которого является этот инстанс
<code>__init__</code>	Конструктор
<code>__del__</code>	Деструктор
<code>__hash__</code>	Возвращает хеш-значение объекта, равное 32-битному числу

Атрибуты и встроенные методы объектов классов.

Часть 2

Атрибут	Описание
<code>__getattr__</code>	Возвращает атрибут, недоступный обычным способом
<code>__setattr__</code>	Присваивает значение атрибуту
<code>__delattr__</code>	Удаляет атрибут
<code>__call__</code>	Выполняется при вызове экземпляра класса
<code>__str__</code>	Строковое представление объекта
<code>__repr__</code>	Формальное строковое представление объекта
<code>__getitem__</code>	Получение элемента по индексу или ключу
<code>__setitem__</code>	Присваивание элемента с данным ключом или индексом
<code>__delitem__</code>	Удаление элемента с данным ключом или индексом

ООП-программа

1. Сформулировать задачу.
2. Определить объекты предметной области, участвующие в решении задачи.
3. Выделить классы, на основе которых генерируются объекты. При необходимости определить базовые классы и классы-потомки.
4. Установить основные атрибуты и методы объектов.
5. Создать классы, их атрибуты и методы.
6. Создать объекты классов.
7. Выполнить итоговое решение задачи, организовав взаимодействие объектов.

```
180         self.scale_setting = FloatProperty(  
181             name="Scale",  
182             min=0.01, max=1000.0,  
183             default=1.0,  
184         )  
185  
186     def execute(self, context):  
187  
188         # get the folder  
189         folder_path = (os.path.dirname(self.filepath))  
190  
191         # get objects selected in the viewport  
192         viewport_selection = bpy.context.selected_objects  
193  
194         # get export objects  
195         obj_export_list = viewport_selection  
196         if self.use_selection_setting == False:  
197             obj_export_list = [i for i in bpy.context.scene.objects]  
198  
199         # deselect all objects  
200         bpy.ops.object.select_all(action='DESELECT')  
201  
202         for item in obj_export_list:  
203             item.select = True  
204             if item.type == 'MESH':  
205                 file_path = os.path.join(folder_path, "{}.obj".format(item.name))  
206                 bpy.ops.export_scene.obj(filepath=file_path, use_selection=True,  
207                                         axis_forward=self.axis_forward_setting,  
208                                         axis_up=self.axis_up_setting,  
209                                         use_animation=self.use_animation_setting,  
210                                         use_mesh_modifiers=self.use_mesh_modifiers_setting,  
211                                         use_edges=self.use_edges_setting,  
212                                         use_smooth_groups=self.use_smooth_groups_setting,  
213                                         use_smooth_groups_bitflags=self.use_smooth_groups_bitflags_setting,  
214                                         use_normals=self.use_normals_setting,  
215                                         use_uv=self.use_uv_setting,  
216                                         use_materials=self.use_materials_setting,
```

Создание собственных исключений

В Python существует возможность создания собственных классов-исключений — потомков класса Exception.

Exception
ZeroDivisionError
IndexError
KeyError
FileExistsError
FileNotFoundError
IndentationError
TypeError
ValueError



Pip. Особенности использования

Это популярная система управления пакетами. Она участвует в установке и управлении программными пакетами, реализованными с помощью Python.

Команда	Описание
<code>pip help</code>	Получить подсказку о доступных командах
<code>pip install package_name</code>	Установить пакет
<code>pip uninstall package_name</code>	Удалить пакет
<code>pip list</code>	Получить список установленных пакетов
<code>pip search package_name</code>	Найти пакет по имени
<code>pip install -U package_name</code>	Обновить указанный пакет
<code>pip show package_name</code>	Получить информацию об установленном пакете

Virtualenv. Особенности использования

Под виртуальной средой понимают директорию, содержащую необходимые для работы приложения пакеты. С их помощью выполняется изолированный запуск приложения.

Установка virtualenv:

```
pip install virtualenv
```

Создание виртуальной среды:

```
virtualenv my_proj
```

Активация виртуальной среды:

```
my_proj\Scripts\activate - Windows  
source my_proj/venv/bin/activate - Linux, MacOS
```

Деактивация виртуальной среды:

```
my_proj\Scripts\deactivate - Windows  
deactivate - Linux, MacOS
```

Библиотека psutil

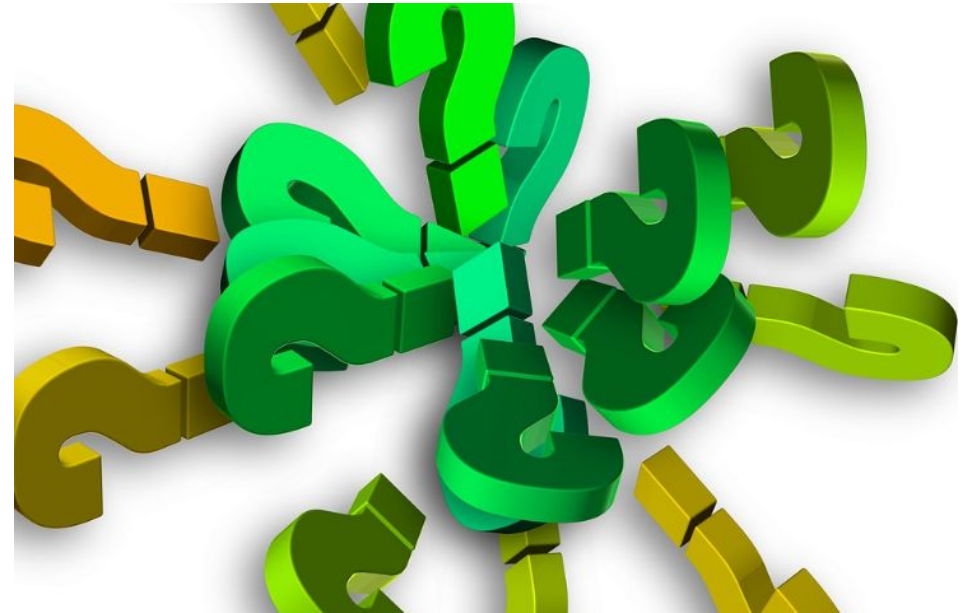
Позволяет получить информацию о параметрах процессора, памяти, дисков.

Это отличная библиотека для управления системой и ресурсами.



Библиотека requests

Сторонний инструмент для выполнения запросов и обработки ответов. Одно из ключевых звеньев для парсинга веб-страниц.



Итоги урока

1. Познакомились с полезными дополнениями механизма ООП в Python: статическим методом и методом класса.
2. Рассмотрели атрибуты и встроенные методы объектов класса.
3. Научились пошагово создавать ООП-программы и собственные исключения.
4. Узнали о некоторых трюках для создания лаконичного и стильного кода.
5. Познакомились с библиотеками: psutil, pip, virtualenv, requests.