



ОНЛАЙН-ОБРАЗОВАНИЕ

Онлайн-образование

Проверить, идет ли запись!





Меня хорошо видно && слышно?

Ставьте  , если все хорошо
Напишите в чат, если есть проблемы

Объектно-ориентированное программирование



Вяземский Семён Валерьевич

QA Automation Lead

Beeline

svvyazemsky@gmail.com

План урока

Концепция ООП



Реализация ООП в Python



Method Resolution Order

Правила вебинара



Активно участвуем



Задаем вопрос в чат или голосом



Off-topic обсуждаем в Slack #канал группы или #general



Вопросы вижу в чате, могу ответить не сразу

Задание на сегодня:

- 1 Создать класс ТестКейс с атрибутами
- 2 Создать подклассы для api из домашнего задания
- 3 Создать conftest и написать тесты для созданных классов

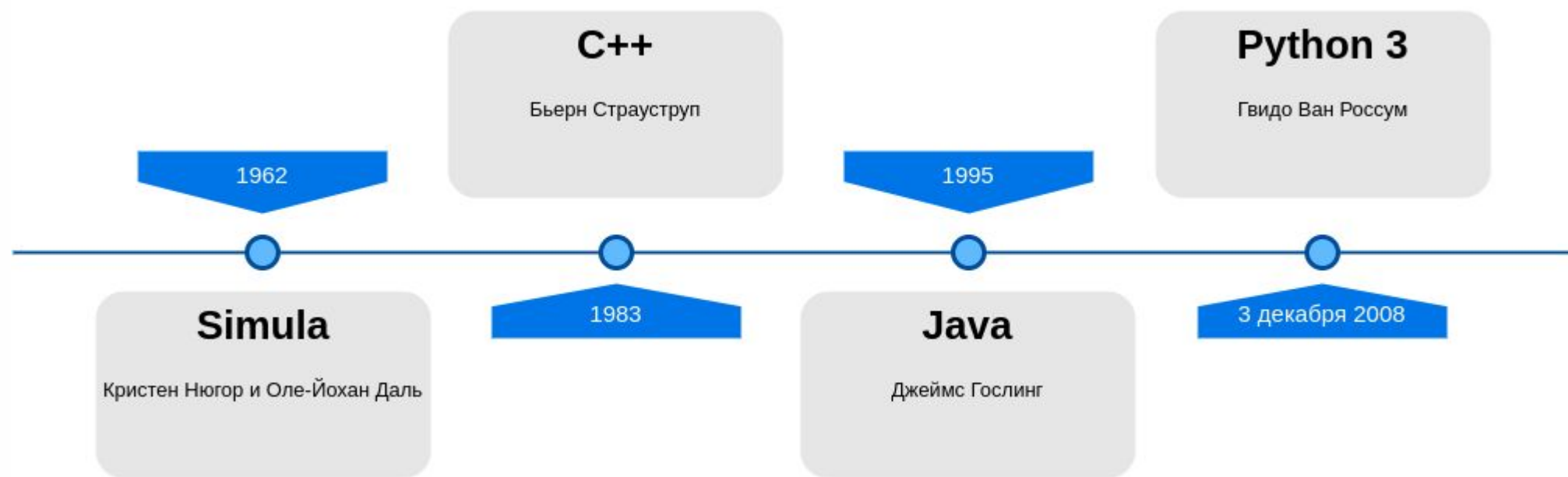


Тайминг: 90 минут

The background of the slide is a high-angle, blue-tinted aerial photograph of a dense urban skyline, likely New York City. Overlaid on this image is a semi-transparent blue band that contains a white network diagram. This diagram consists of numerous small dots connected by thin white lines, creating a complex web-like structure that spans the width of the slide. Centered within this band is the main title in large, white, sans-serif capital letters.

Концепция ООП

Минутка истории



Базовые понятия ООП

Абстракция — способ идентифицировать сущность путем выделения наиболее значимых элементов.

Наследование — концепция ООП согласно которой класс наследник перенимает атрибуты и методы класса родителя.

Инкапсуляция — размещение данных и методов для работы с ними, а также сокрытие внутренней реализации от других компонентов программы, ограничение доступа к атрибутам и методам.

Полиморфизм — возможность частично или полностью переопределять поведение класса в рамках иерархии наследования

Объектно-ориентированное программирование

класс *Транспорт*
подкласс *Автомобиль*
подкласс *СуперКар*
объект



The background of the slide is a blue-tinted aerial photograph of a dense city skyline, likely New York City. Overlaid on this is a semi-transparent network pattern consisting of numerous small dots connected by thin lines, creating a web-like structure. The text "OOP + Python" is centered in the middle of the slide in a large, white, sans-serif font.

OOP + Python

Первый класс



```
class Car:
    name = "c200"
    make = "mercedes"
    model = 2008
    def start(self):
        print ("Заводим двигатель")
    def stop(self):
        print ("Отключаем двигатель")
```


Первый объект



```
my_car = Car()
```

Атрибуты



```
class Car:
    speed = 0

toyota = Car()
lada = Car()
toyota.car_count = 5
print(lada.speed, toyota.speed)

>> 0 5
```


Методы



```
class Car:
    speed = 0
    def start(self):
        self.speed = 5

toyota = Car()
lada = Car()
toyota.start()
print(lada.speed, toyota.speed)

>> 0 5
```

Уровни доступа

```
class Car:
    def __init__(self):
        self.name = "camry"
        self.__make = "toyota"
        self._model = 2015
my_car = Car()
print(my_car.name)
print(my_car._model)
>>camry
>> AttributeError: 'Car' object has no attribute '_model'
```


MRO



```
class Music(object): pass
class Rock(Music): pass
class Gothic(Music): pass
class Metal(Rock): pass
class GothicRock(Rock, Gothic): pass
class GothicMetal(Metal, Gothic): pass
class The69Eyes(GothicRock, GothicMetal): pass
```

Декораторы классов

@staticmethod - делает метод статическим

@classmethod - делает метод, методом класса

@property - превращает методы в свойство

@abstract* - декораторы для создания свойств объекта обязательных для реализации в наследниках

Декораторы для создания абстрактных свойств и методов находятся в пакете abc <https://docs.python.org/3/library/abc.html>

Волшебные методы

В общем виде это некоторый набор методов класса, явная реализация или переопределение которых позволяют управлять поведением класса и его экземпляров.

```
class A:


    def __init__(self, lower_case, value):
        self.lower_case=lower_case
        self.val=value

    def __eq__(self, other):
        return self.val==other.val

    def __lt__(self, other):
        print ('checking less than by definition in __lt__')
        return self.val<other.val

a=A('a',1)
a2=A('a',2)

print(a==a2)
print(a<a2)
```


The background of the image is an aerial photograph of a city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer that features a white network pattern of interconnected dots and lines. The text is centered within this blue layer.

Пройди опрос:
<https://otus.ru/polls/10225/>

Следующий вебинар

Тема: Python.



24.04.2020

Спасибо за внимание!
Приходите на следующие вебинары



Вяземский Семён Валерьевич

QA Automation Lead

Beeline

svvyazemsky@gmail.com