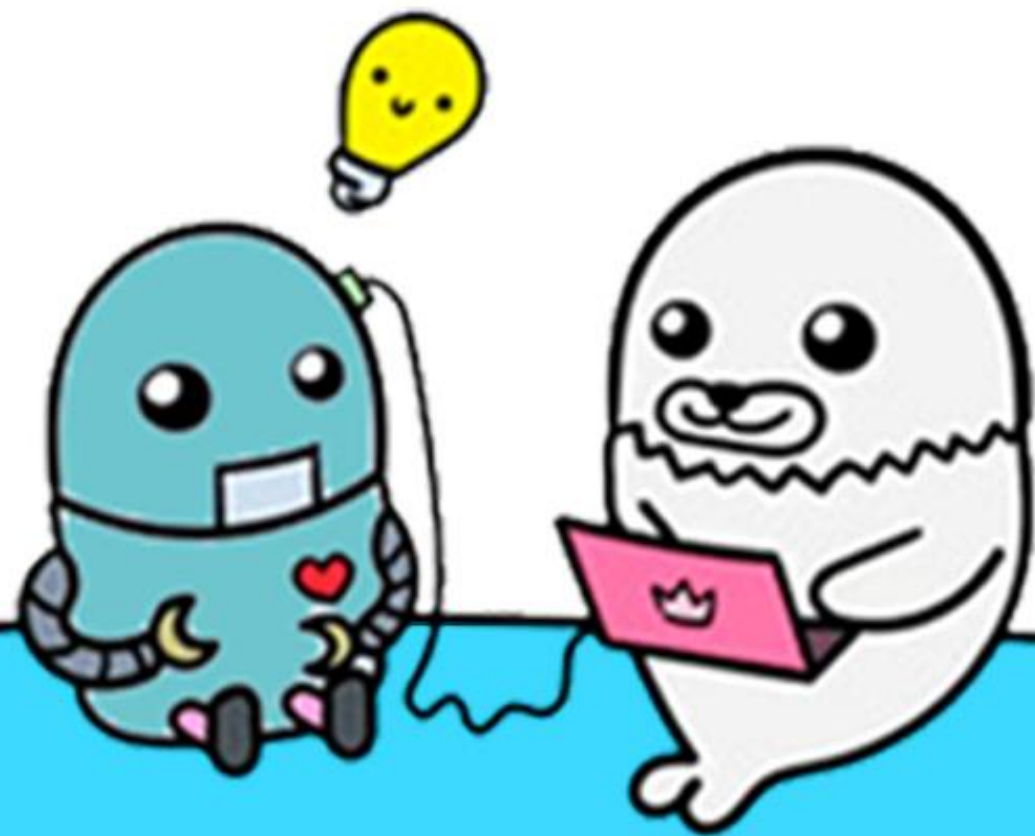


MLOps и production подход к ML исследованиям



28 марта - 28 мая



MLOps и production подход к ML исследованиям

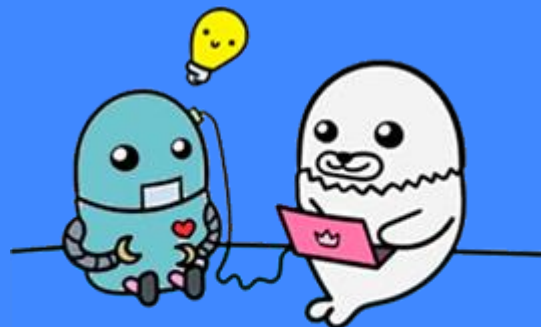
Codestyle, инструменты форматирования, линтеры

Павел Кикин

Газпромнефть ЦР

Руководитель направления NLP

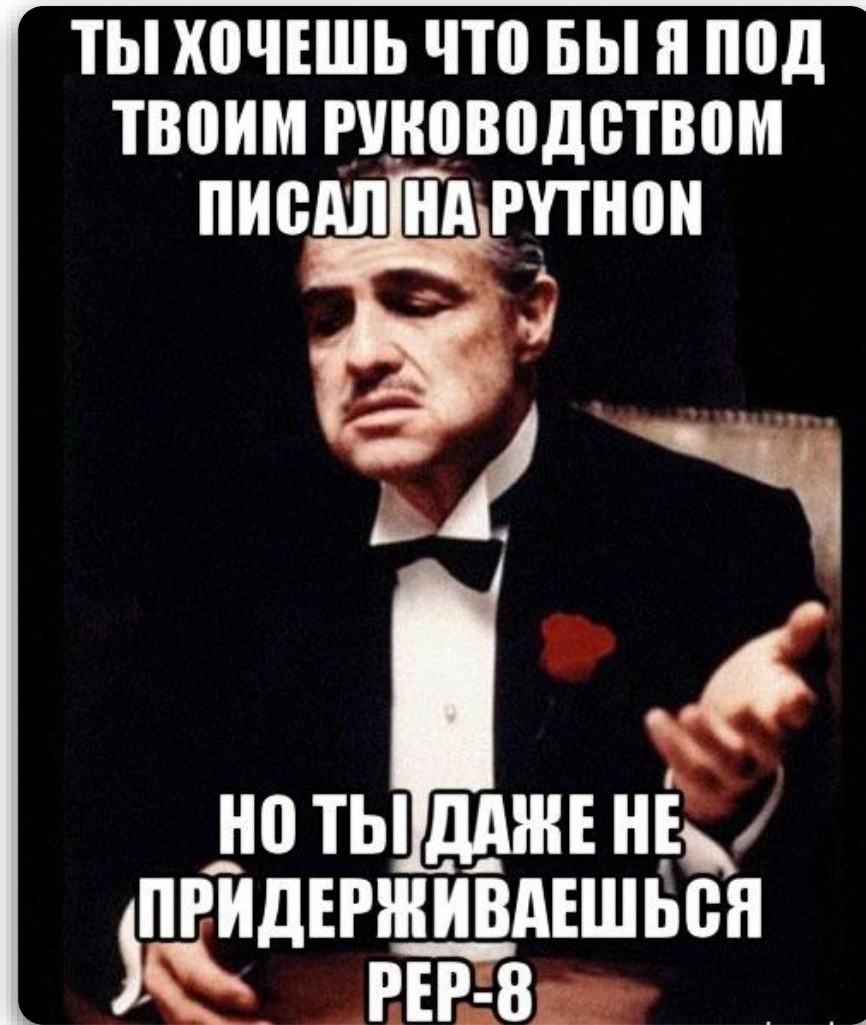
t.me/pavel_kikin





План занятия

- Что это такое и почему важно?
- Стандарты Codestyle (PEP8)
- Dockstring
- Аннотации
- Автоформатеры
- Линтеры
- CI/CD





Почему важно соблюдать codestyle?

- Большая часть времени тратится на чтение кода
- Восприятие кода визуальными паттернами
- Повышение разнородности кода проекта при росте команды
- Снижение эффективности команды
 - Снижение скорости чтения кода
 - Лень
- Репутационные издержки



Почему важно соблюдать codestyle?

, что бы

перед ровными млщиками не позориться



5



2

8:18



Что есть codestyle?

</> Code style

Это форма социального контракта внутри вашей команды





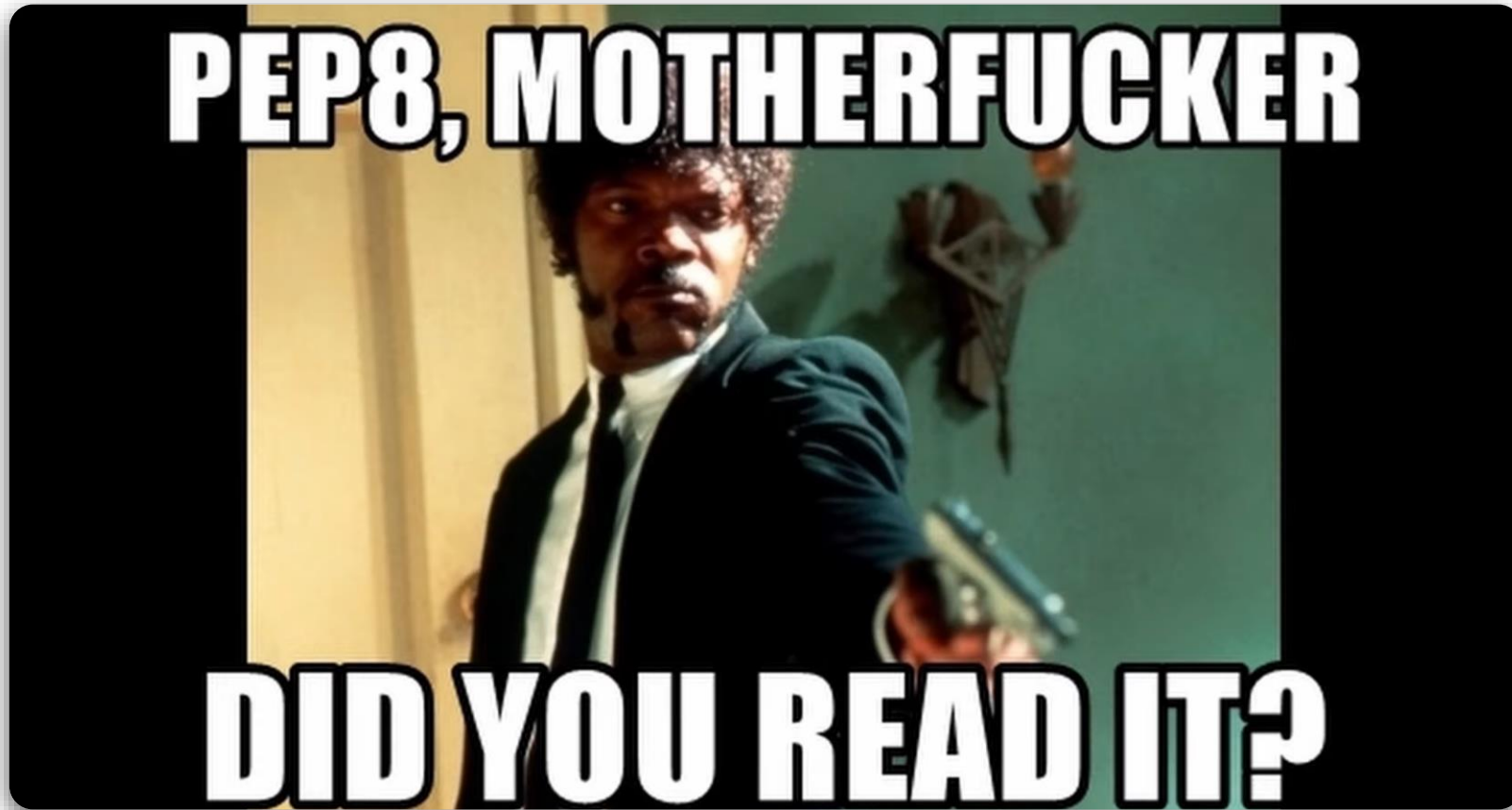
Из чего состоит codestyle?

- **Нейминг:**
 - На уровне смысла
 - На уровне оформления
- **Комментарии**
- **Аннотации**
- **Dockstring**
- **Отступы и переносы**
- **Паттерны проектирования**



PEP8

PEP 8 - Python Enhancement Proposal





PEP8 | Отступы

- 4 пробела на каждый уровень отступа

```
16      def name(self) -> Text:
17          return "action_ues_fetch_personal_account"
18
```




PEP8


Висячие отступы

- Аргументы на первой линии запрещены, если не используется вертикальное выравнивание


```
def run(self,  
    dispatcher: CollectingDispatcher,  
    tracker: Tracker,  
    domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:
```



```
def run(self,  
    dispatcher: CollectingDispatcher,  
    tracker: Tracker,  
    domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:
```




```
def run(  
    self,  
    dispatcher: CollectingDispatcher,  
    tracker: Tracker,  
    domain: Dict[Text, Any]) -> List[Dict[Text, Any]]:  
    login = tracker.sender_id
```





PEP8 | Висячие отступы



```
def run(  
    self,  
    dispatcher: CollectingDispatcher,  
    tracker: Tracker,  
    domain: Dict[Text, Any]  
):
```



```
    login = tracker.sender_id
```

```
def run(  
    self,  
    dispatcher: CollectingDispatcher,  
    tracker: Tracker,  
    domain: Dict[Text, Any]  
) -> List[Dict[Text, Any]]: ?  
  
    login = tracker.sender_id
```



PEP8 | Длина строки

- **79 символов** – PEP8 Default
- **72 символа** – PEP8 Default для документации и комментариев
- **88 символов** – Black Default



PEP8

Отделение элементов пустыми строками


- **1 строка** – методы внутри класса
- **2 строки** – функции верхнего уровня и определения классов




PEP8 | Импорты

- Каждый импорт на отдельной строке
- Возможно подключение атрибутов одного модуля на одной строке
- Импорты всегда в начале файла (после комментариев и документации к модулю)
- Группировка осуществляется через пустую строку:
 - Стандартная библиотека
 - Сторонние библиотеки
 - Модули проекта

```
import os
import requests
```




```
import os, requests
```



PEP 8: E401 multiple imports on one line

```
from typing import Dict, Text, Any, List
```





PEP8

Блоки комментариев

- Объясняет код идущий после него
- Каждая строка начинается с # и одного пробела после
- Абзацы разделяются #



PEP8 | Строчные комментарии

- Объясняют отдельную строку
- Отделяются не менее чем 2мя пробелами от инструкции
- Начинаются с # и одного пробела после
- Считаются не желательными без существенной необходимости



PEP8 | Naming

- **Константы:**

```
ENTITY_ATTRIBUTE_CONFIDENCE
```

- **Методы, функции, модули и переменные:**

```
send_mail
```

- **Классы и исключения:**

```
ActionFetchPersonalAccount
```

- **Пакеты:**

```
filters
```



PEP8 | Naming

- Не используйте **l** (маленькая латинская буква «L») и **O** (заглавная латинская буква «o») как однобуквенные идентификаторы
- Давайте понятные, несущие смысл имена
- Избегайте сокращений и имён состоящих из одного символа



DockString

- Начинаются и заканчиваются `"""`
- Используются для публичных функций, классов и методов

```
19  def clean_data(df: pd.DataFrame) -> pd.DataFrame:
20      """Function removes excess columns and enforces
21          correct data types.
22          :param df: Original DataFrame
23          :return: Updated DataFrame
24      """
```



Аннотации

- Лямбды
- Километровые строчки Pandas
- Лапша из if-else

```
def indent(string: str) -> int:
    """Count the indentation in whitespace characters.

    Args:
        string (str): text with indents

    Returns:
        int: Number of whitespace indentations

    """
    return sum(4 if char == "\t" else 1 for char in string[: -len(string.lstrip())])
```



Общие рекомендации

- Сравнения с **None** реализуются через **is** или **is not**
- Используйте **def** вместо присваивания **лямбда**-выражений имени
- Не сравнивайте булевы переменные с **True** и **False** с помощью **==**
- Не пишите слишком длинные выражения (например с Pandas)
- Избегайте «лапши» из if-else

```
for(int j=0; j<n; j++)
{
    if(n!=2)
    {
        if(i<r)
        {
            if(j<t)
            {
                if(j>=k)
                {
                    if(j==k)
                    {
                        if(l==1)
                        {
                            {
                                a[i,j] = k + 1;
                            }
                        }
                    }
                    else{ a[i,j] = k; }
                    else{ a[i,j] = a[i,k]; }
                }
            }
            else{ a[i,j] = a[i-1, j]; }
        }
    }
    else
    {
        if(a[i, j-1] - 1 > 0)
```



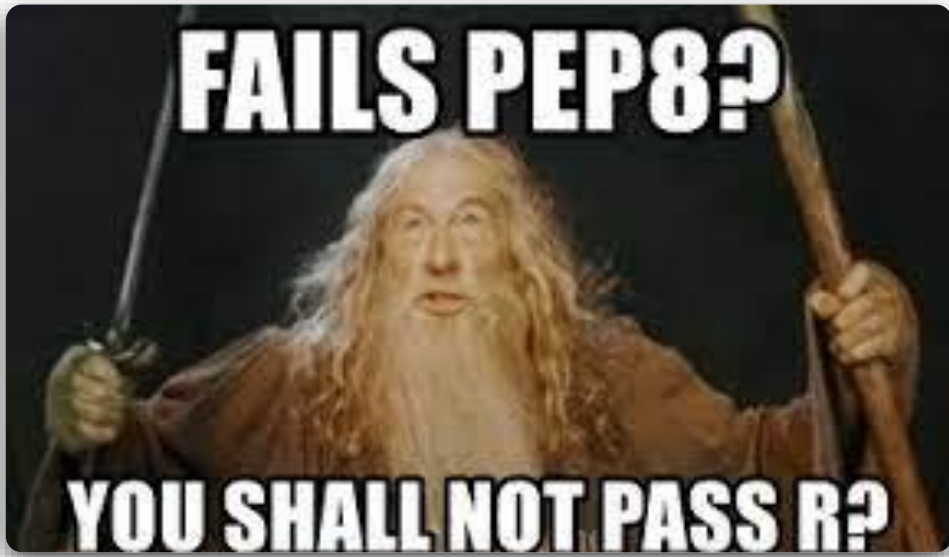
Автоформатеры





Линтеры

: my[py]





Линтеры

Lintер	Category	Description
Pylint	Logical & Stylistic	Checks for errors, tries to enforce a coding standard, looks for code smells
PyFlakes	Logical	Analyzes programs and detects various errors
pycodestyle	Stylistic	Checks against some of the style conventions in PEP 8
pydocstyle	Stylistic	Checks compliance with Python docstring conventions
Bandit	Logical	Analyzes code to find common security issues
MyPy	Logical	Checks for optionally-enforced static types



MLOps и production подход к ML исследованиям

Хранение и версионирование кода с git

Павел Кикин

Газпромнефть ЦР

Руководитель направления NLP

t.me/pavel_kikin

