

Git – КОНТРОЛЬ версий





Git — абсолютный лидер по популярности среди современных систем управления версиями.

Это развитый проект с активной поддержкой и открытым исходным кодом. Система Git была изначально разработана в 2005 году Линусом Торвальдсом — создателем ядра операционной системы Linux.



GitHub — это система управления проектами и версиями кода, а также платформа социальных сетей, созданная для разработчиков.

Sourcetree — это визуальный клиент, который предоставляет доступ к репозиториям Git и Mercurial на базе Windows и Mac и визуализирует процесс разработки без доступа к командной строке.

Bitbucket Cloud — это органичный Git-инструмент внутри решения Atlassian Open DevOps, обладающий лучшей в своем классе интеграцией с Jira и встроенными возможностями CI/CD.

Git — система управления версиями с распределенной архитектурой. В отличие от некогда популярных систем вроде CVS и Subversion (SVN), где полная история версий проекта доступна лишь в одном месте, в Git каждая рабочая копия кода сама по себе является репозиторием. Это позволяет всем разработчикам хранить историю изменений в полном объеме.

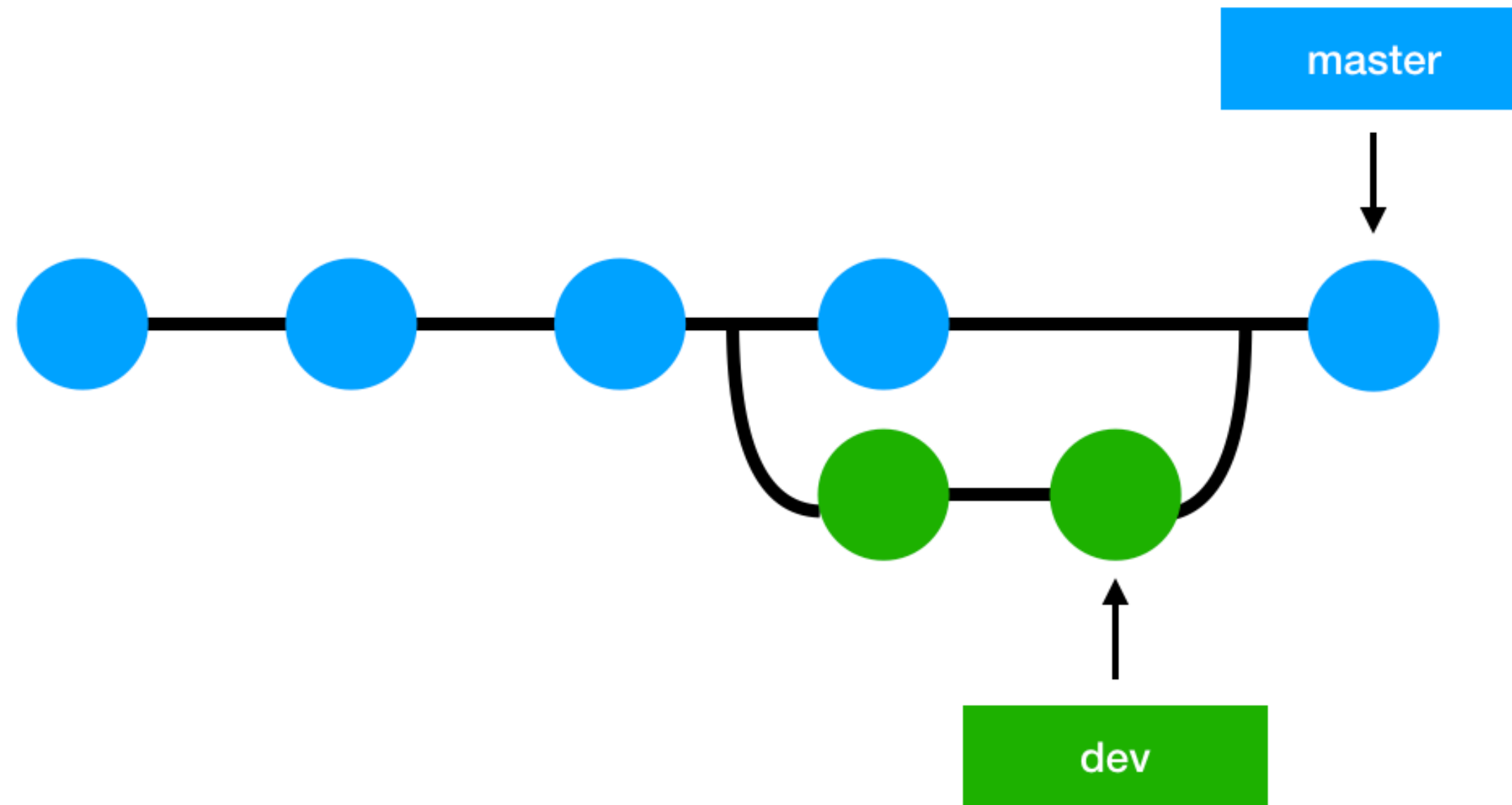
Разработка в Git ориентирована на обеспечение высокой производительности, безопасности и гибкости распределенной системы.



Зачем нам нужен GIT?

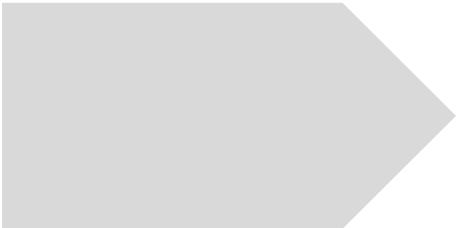
Git — это наиболее распространенная система контроля версий. Git отслеживает изменения, которые вы вносите в файлы, чтобы у вас была запись о том, что было сделано, и вы могли вернуться к определенным версиям, если вам это понадобится. Git также облегчает совместную работу, позволяя объединять изменения, внесенные несколькими людьми, в один источник.

Поэтому независимо от того, пишете ли вы код, который будете видеть только вы, или работаете в команде, Git будет вам полезен.

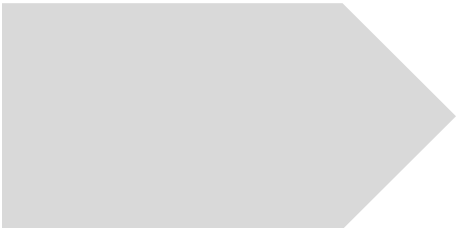


Git открывает возможность работать многим людям над одним проектом. Вы можете создавать неограниченное количество ветвлений, которые будут отличны от master ветки. Что позволит вам не бояться испортить код главной ветки и тестировать функционал

Следит за целостностью данных



Перед сохранением любого файла Git вычисляет контрольную сумму, и она становится индексом этого файла. Поэтому невозможно изменить содержимое файла или каталога так, чтобы Git не узнал об этом. Если информация потеряется при передаче или повредится на диске, Git всегда это выявит.



Механизм, используемый Git'ом для вычисления контрольных сумм, называется SHA-1 хешем. Это строка из 40 шестнадцатеричных символов (0-9 и a-f), вычисляемая в Git'e на основе содержимого файла или структуры каталога. SHA-1 хеш выглядит примерно так: 24b9da6552252987aa493b52f8696cd6d3b00373

Зафиксированное состояние файла



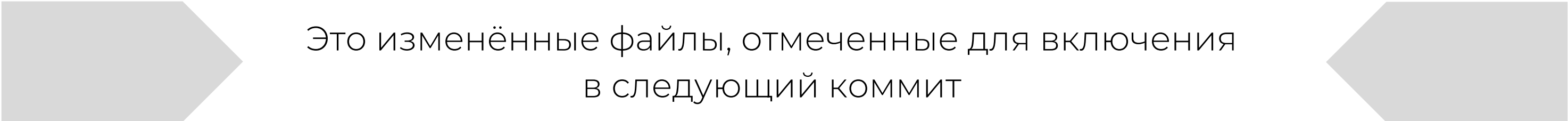
Значит, что файл уже сохранён в вашей локальной базе

Измененное состояние файла



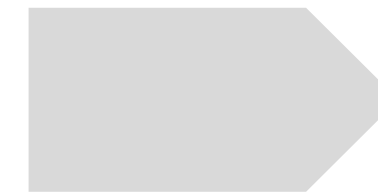
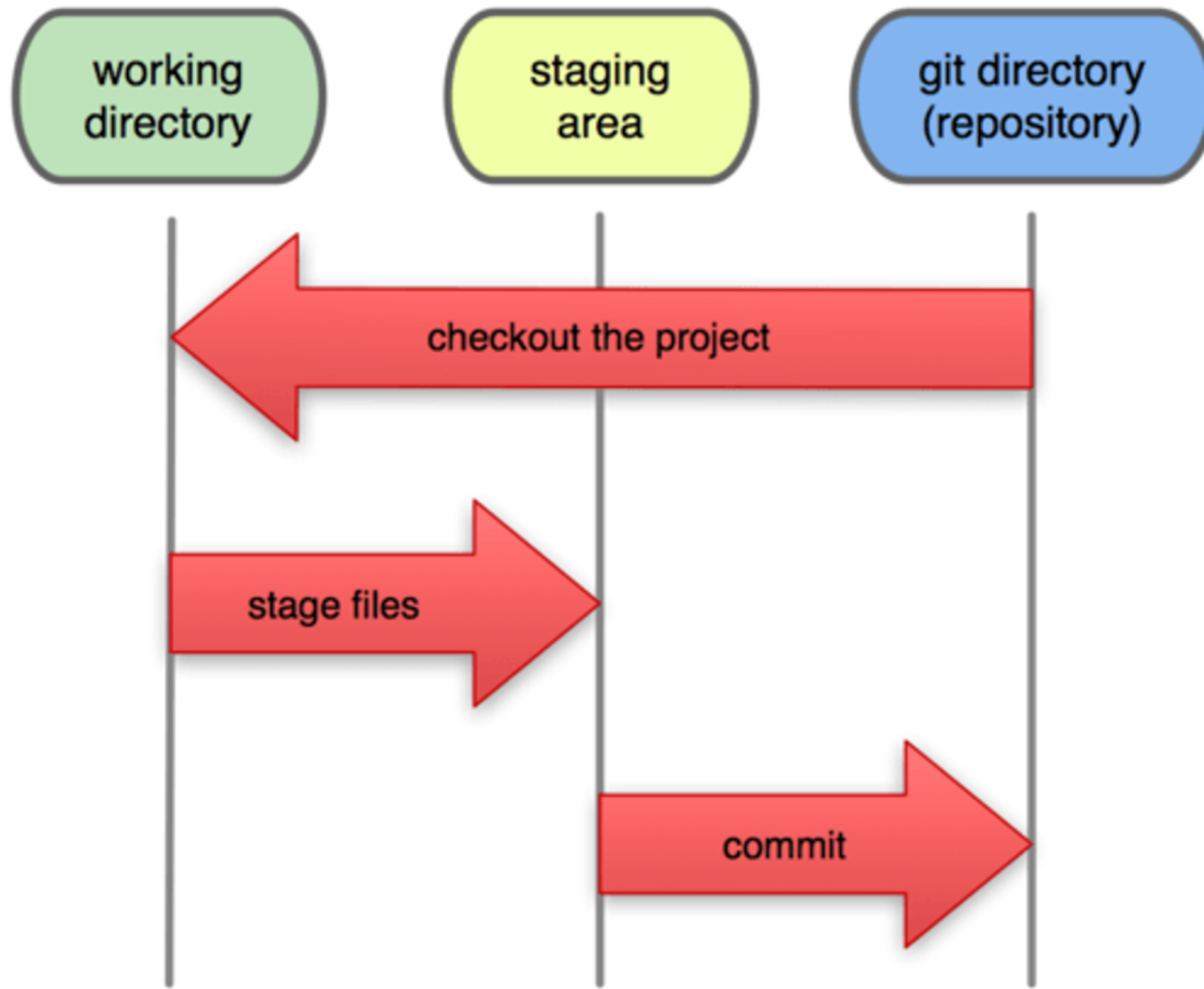
К изменённым относятся файлы, которые поменялись, но ещё не были зафиксированы

Подготовленное состояние файла



Это изменённые файлы, отмеченные для включения в следующий коммит

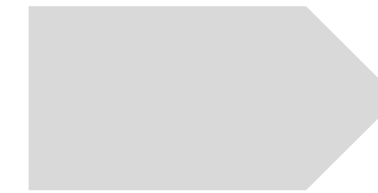
Local Operations



Каталог Git



Рабочий каталог



Область подготовленных
файлов

Начало работы с Git

1 ШАГ

```
$ git config --global user.name "user name"  
$ git config --global user.email user.email@example.com
```

2 ШАГ

```
$ git config --list
```

Help
step

```
$ git help config
```

Создание Git-репозитория

1. Первый подход — импорт в Git уже существующего проекта или каталога;
2. Второй — клонирование уже существующего репозитория с сервера.

Flake8 — зачем нужен?

Все пишут по-разному

```
arr = [1, 2, 3]
```

```
arr = [  
    1, 2, 3  
]
```

```
arr = [  
    1,  
    2,  
    3,  
]
```

```
def sim_pearson(prefs,p1,p2):
    si = {}
    for item in prefs[p1]:
        if item in prefs[p2]:
            si[item]=1
            n = len(si)
    if len(si) == 0:
        return 0

    sum1=sum([prefs[p1][it] for it in si])
    sum2=sum([prefs[p2][it] for it in si])
    sum1Sq=sum([pow(prefs[p1][it],2) for it in si])
    sum2Sq=sum([pow(prefs[p2][it],2) for it in si])
    pSum=sum([prefs[p1][it]*prefs[p2][it] for it in si])
    num = pSum - (sum1 * sum2 / n)

    den = sqrt((sum1Sq - pow(sum1, 2) / n) * (sum2Sq - pow(sum2, 2) / n))
    if den == 0:
        return 0
    r = num / den
    return r
```

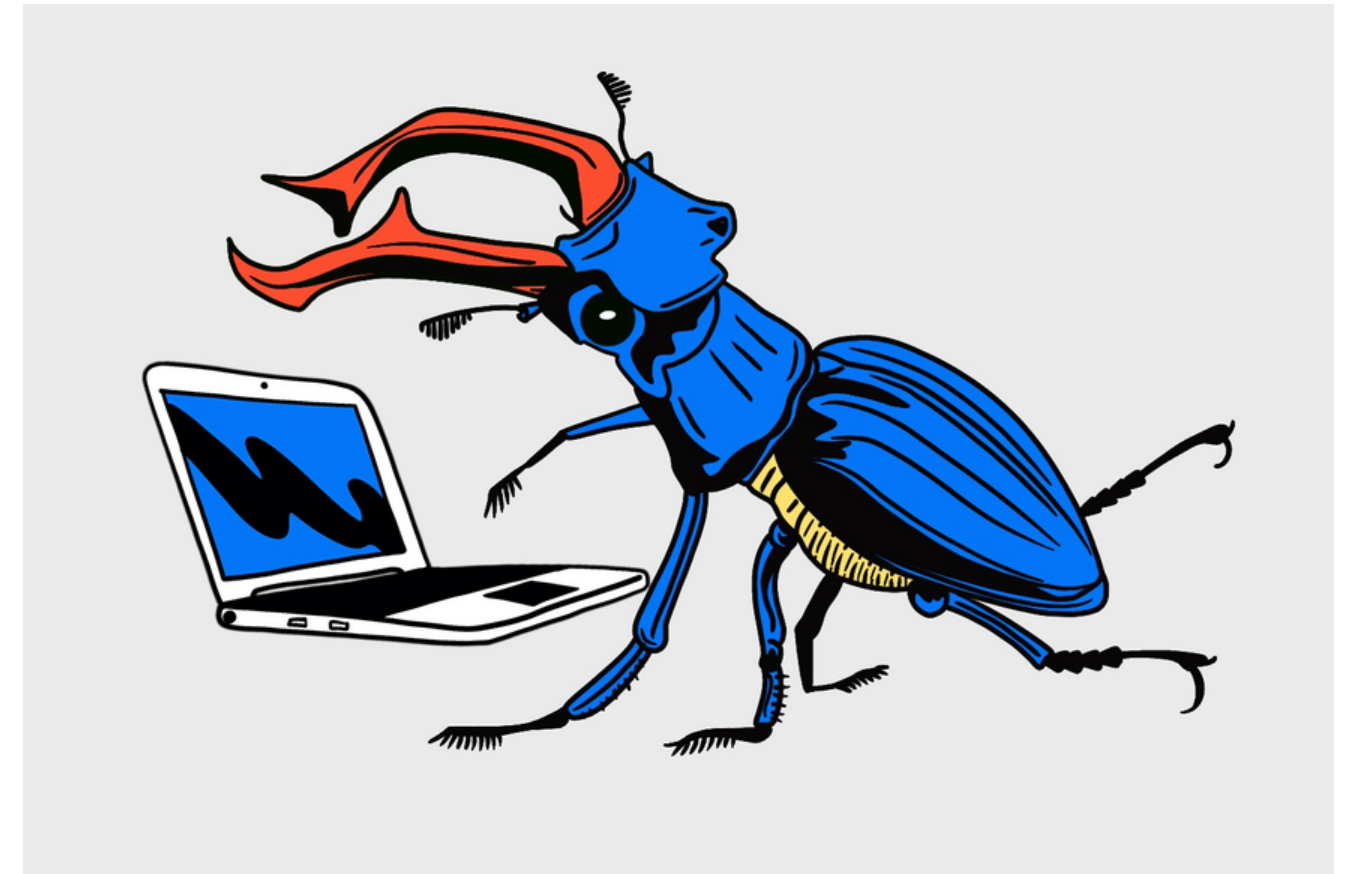
```
:231 missing whitespace after ','
:231 missing whitespace after ','
:225 missing whitespace around operator
:271 multiple spaces after keyword
:225 missing whitespace around operator
:225 missing whitespace around operator
E225 missing whitespace around operator
E231 missing whitespace after ','
E225 missing whitespace around operator
E231 missing whitespace after ','
:225 missing whitespace around operator
F821 undefined name 'sqrt'
W292 _no newline at end of file
```

Использование Flake8 поможет

Уменьшить количество багов

Получить сложность кода

Универсальный код

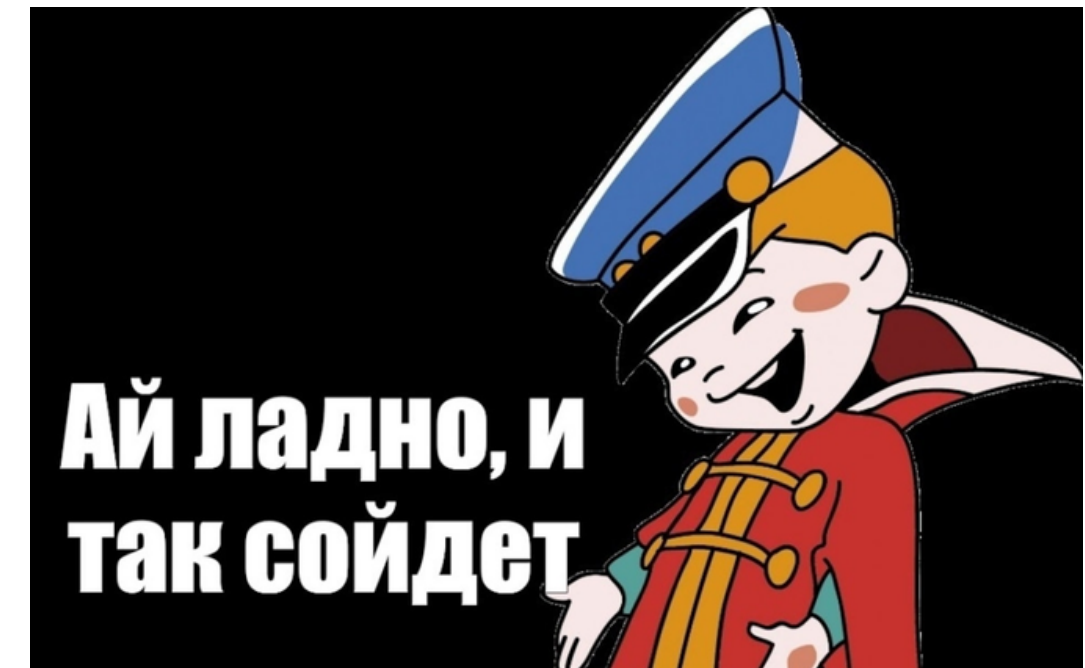


Чтобы внедрить Flake8 нужно



Общаться с командой

Анализировать смысл
использования



Работать над ошибками

Quickstart

```
python -m pip install flake8
```

Install flake8

```
flake8 name_of_file
```

Using flake8 for file

Freeze

```
pip install freeze
```

```
python -m pip freeze [options]
```

```
pip freeze
```

```
freeze > requirements.txt
```

Спасибо за внимание!



**KEEP
LEARNING
AND
HAPPY
CODING**