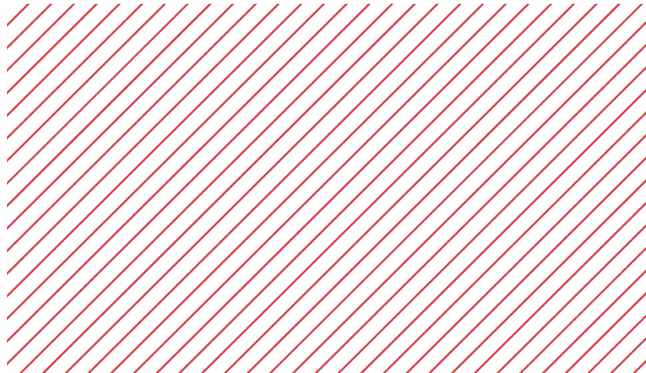
академия больших данных

mail.ru group

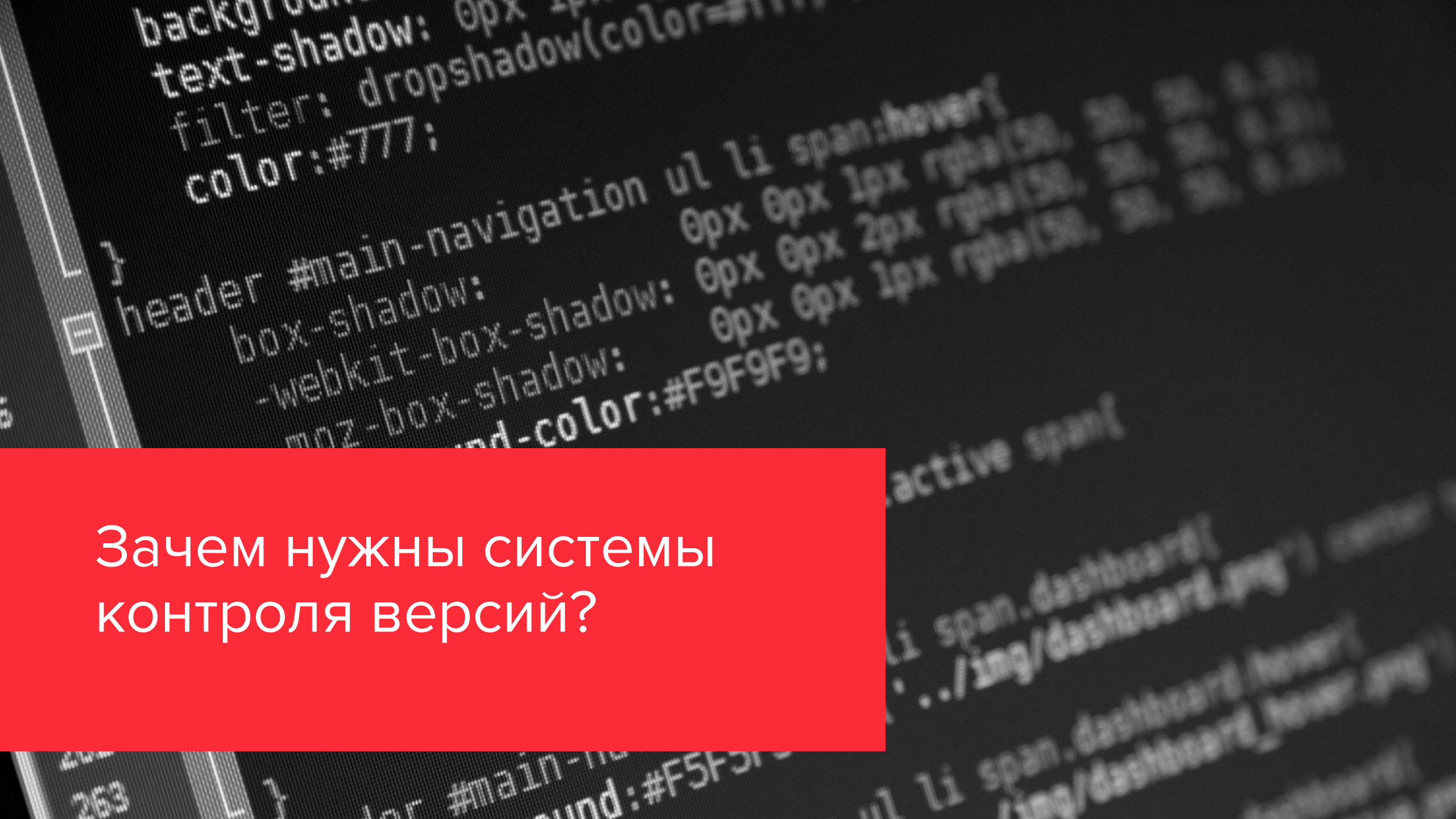
Основы GIT

Михаил Марюфич, MLE



План

1) Зачем нужны системы контроля версия
2) GIT - основные возможности

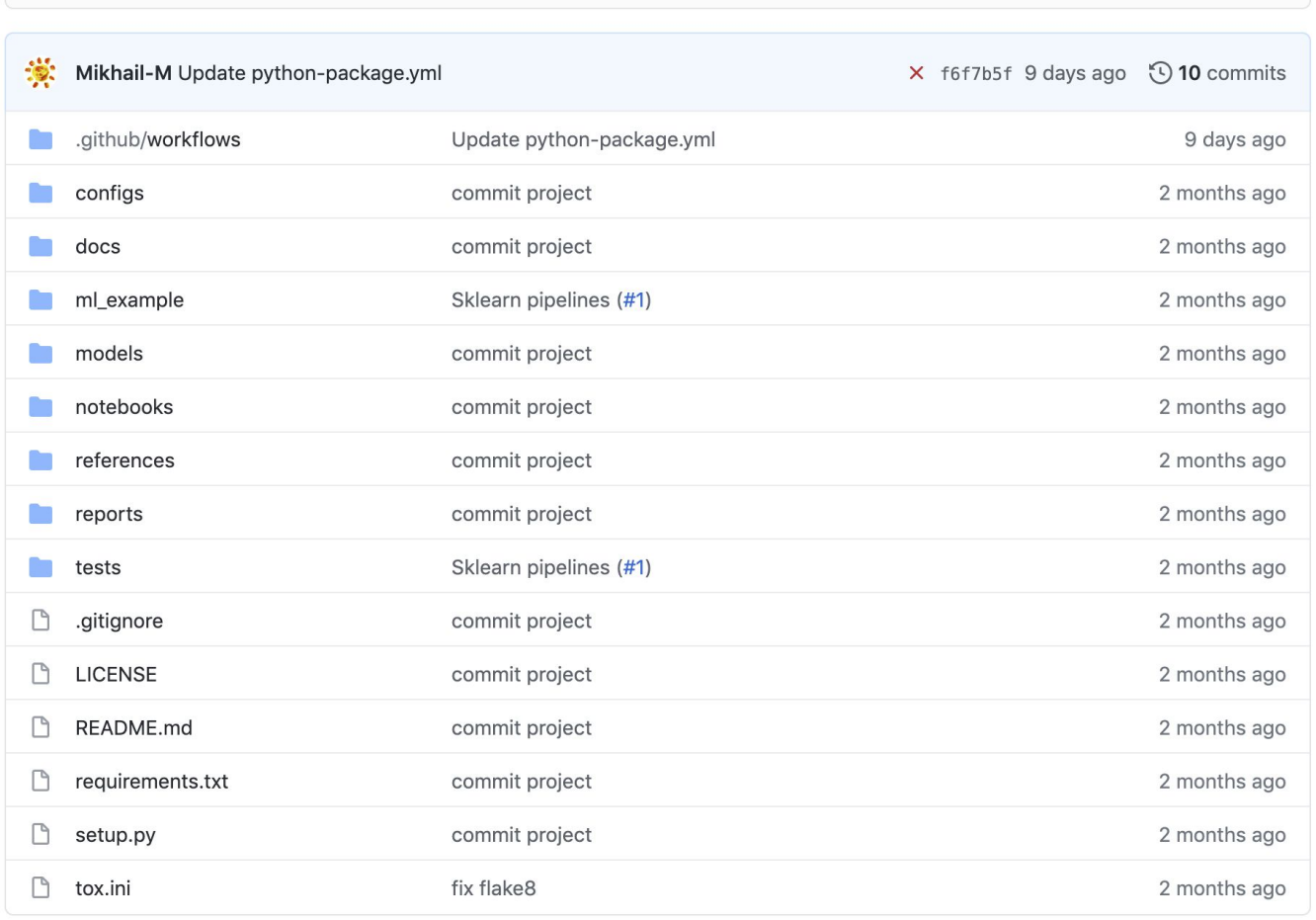


Зачем нужны системы контроля версий?

Системы контроля версий

1) Возврат к старым версиям (safety net)
2) Отслеживание изменений
3) Хранение истории
4) Совместная работа

Системы контроля версий



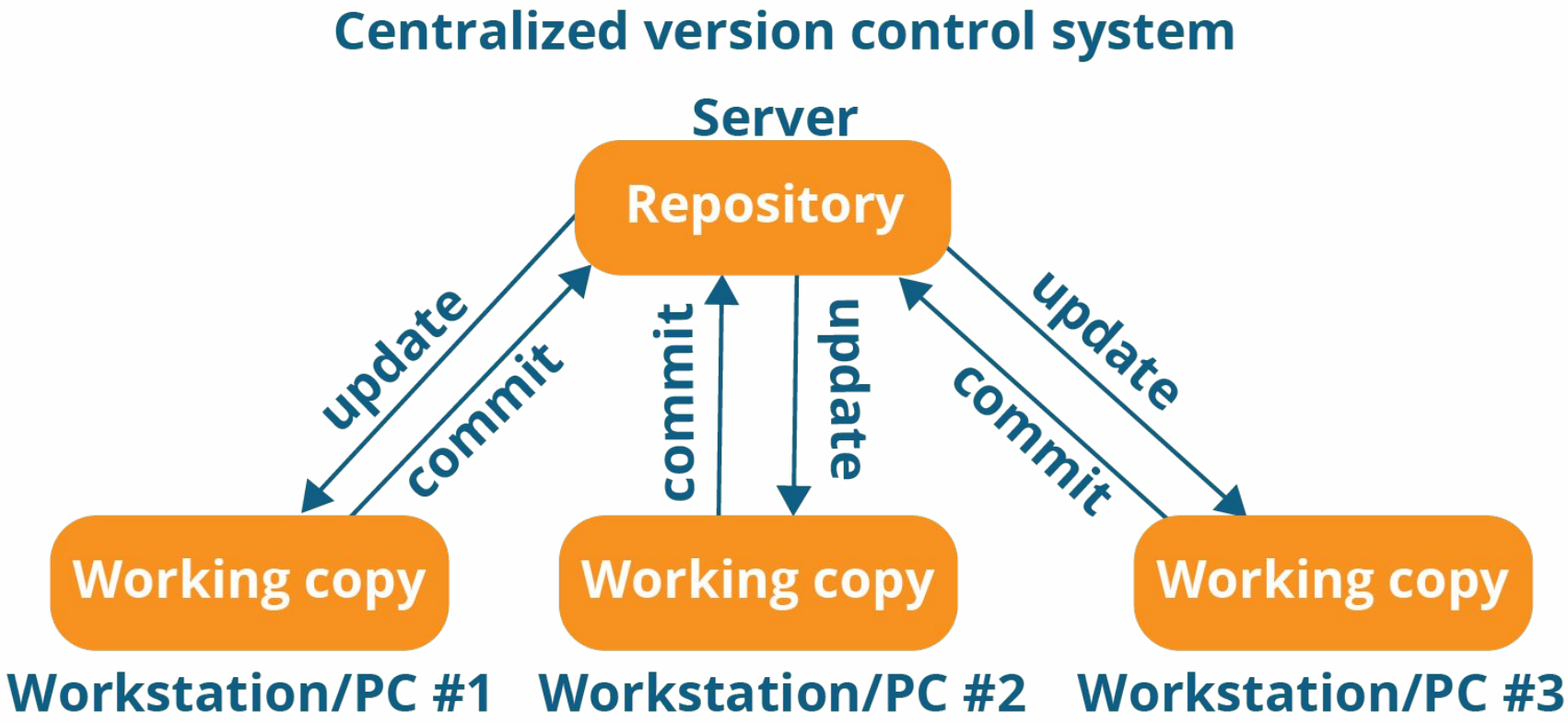
Mikhail-M Update python-package.yml	`f6f7b5f` 9 days ago	**10** commits
.github/workflows	Update python-package.yml	9 days ago
configs	commit project	2 months ago
docs	commit project	2 months ago
ml_example	Sklearn pipelines (#1)	2 months ago
models	commit project	2 months ago
notebooks	commit project	2 months ago
references	commit project	2 months ago
reports	commit project	2 months ago
tests	Sklearn pipelines (#1)	2 months ago
.gitignore	commit project	2 months ago
LICENSE	commit project	2 months ago
README.md	commit project	2 months ago
requirements.txt	commit project	2 months ago
setup.py	commit project	2 months ago
tox.ini	fix flake8	2 months ago

Как можно сделать?

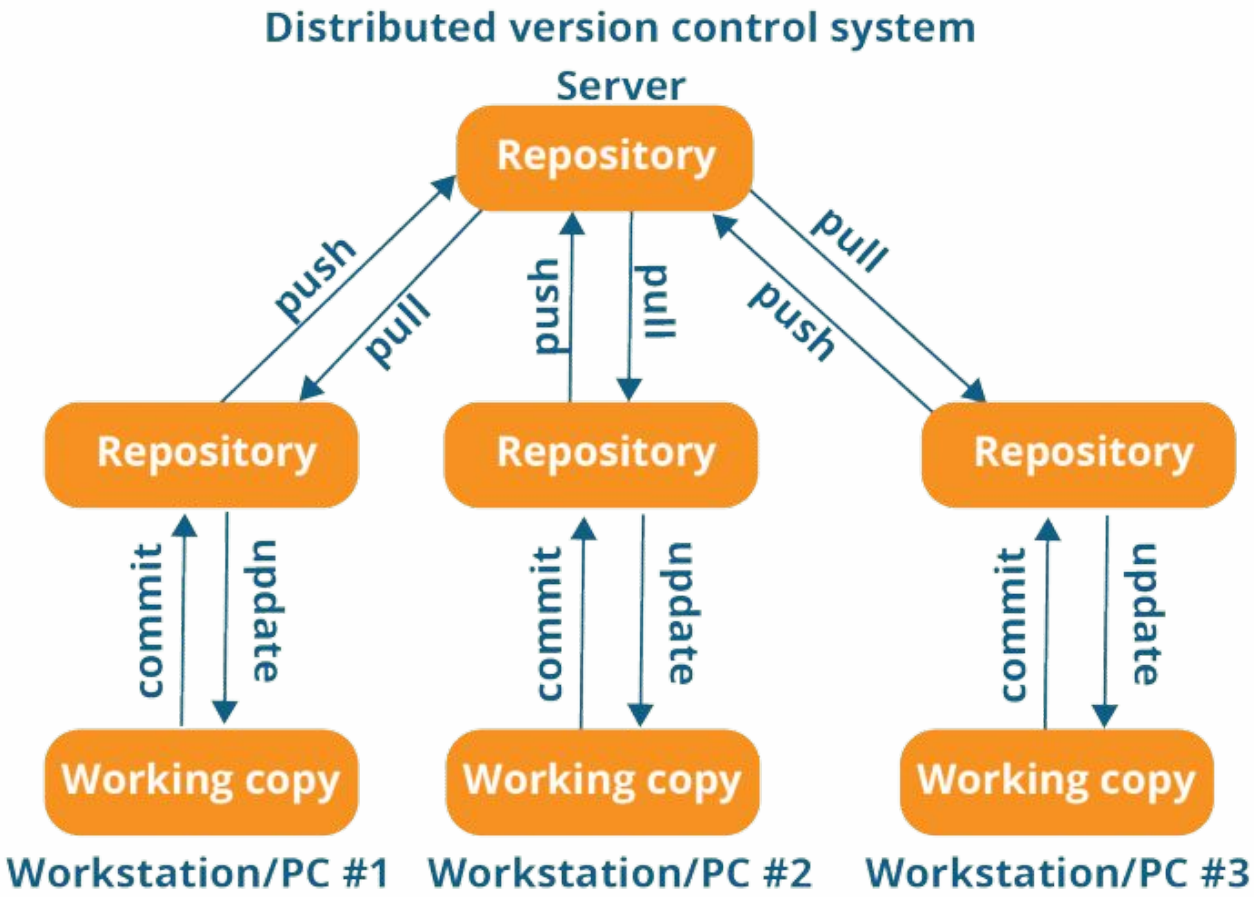
Версионировать файлы

```
 9  13
10      - def impute_features(df: pd.DataFrame, strategy: str) -> pd.DataFrame:
        14 + def get_imputer(strategy: str) -> _BaseImputer:
11  15          imputer = SimpleImputer(missing_values=np.nan, strategy=strategy)
12      -       features_transformed = imputer.fit_transform(df)
13      -       features_pandas = pd.DataFrame(
14      -           features_transformed, columns=df.columns, index=df.index,
15      -       )
16      -       return features_pandas
17      -
        16 +       return imputer
18  17
19      - def impute_categorical_features(df: pd.DataFrame):
20      -       return impute_features(df, strategy="most_frequent")
21  18
        19 + def get_categorical_imputer() -> _BaseImputer:
        20 +       return get_imputer(strategy="most_frequent")
22  21
23      - def impute_numerical_features(df: pd.DataFrame):
24      -       return impute_features(df, strategy="mean")
```

Центральная система контроля версия



Распределенная система контроля версия



Какие бывают?

Git

- Стандарт де-факто
- Распределенный
- github.com, gitlab.com, https://bitbucket

Github

- Стандарт де-факто
- Распределенный
- github.com, gitlab.com, https://bitbucket

git add

Добавляем файлик под контроль версий

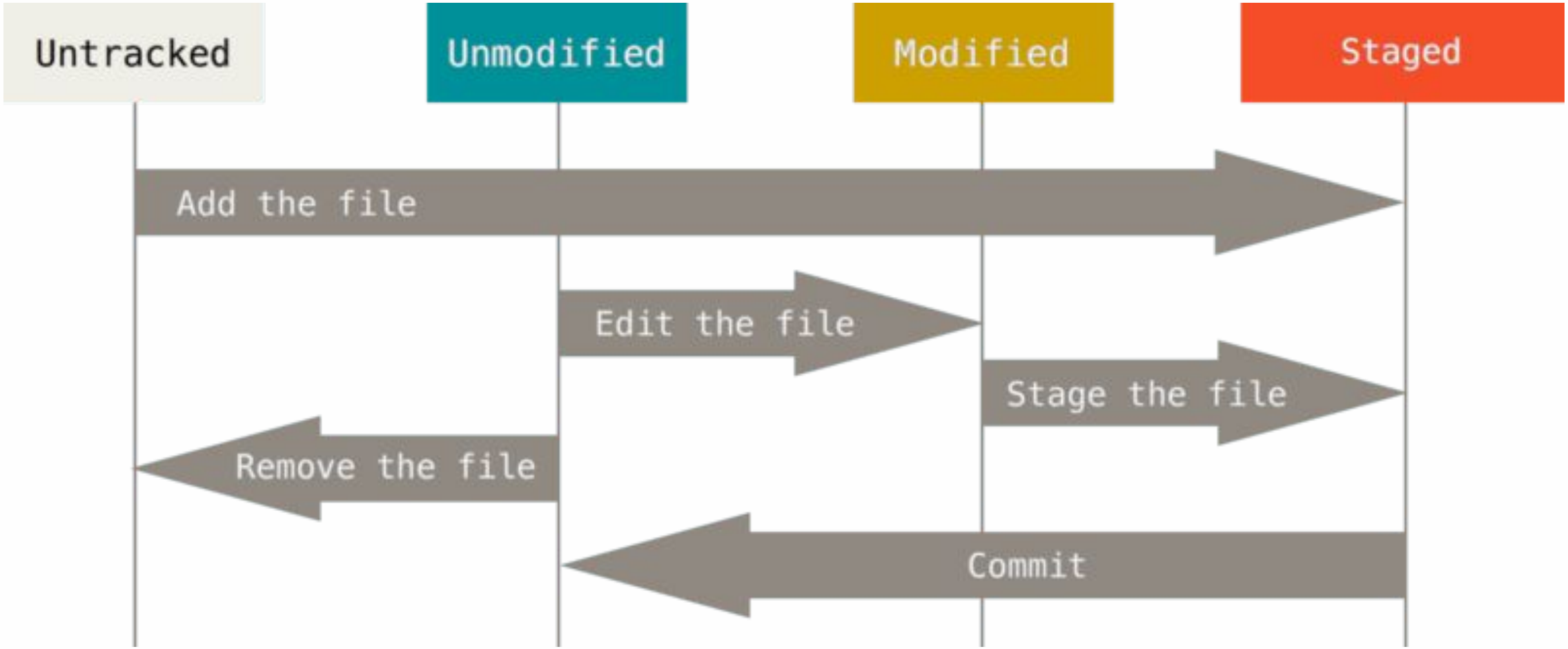
git add file.txt

git commit

Фиксируем изменение

git commit -m “my commit”

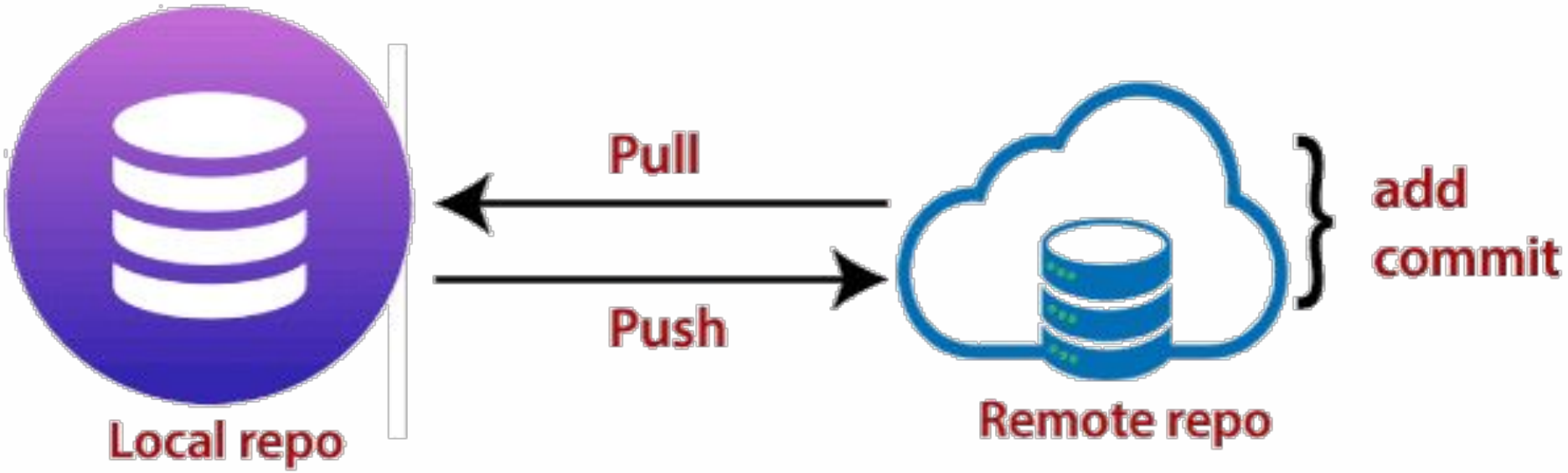
Состояния файлов



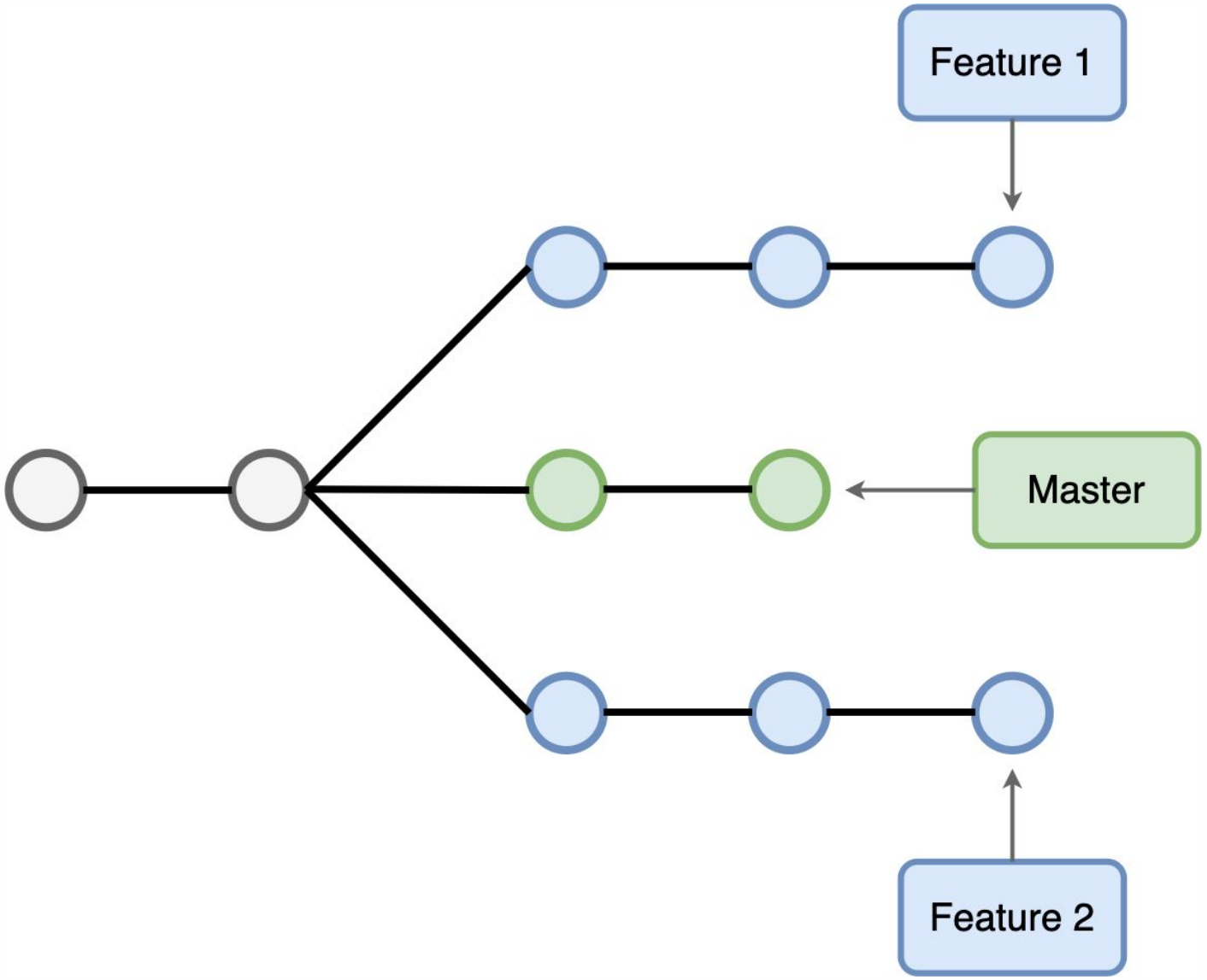
git status

```
@:~/week-4-game <master>$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   assets/css/style.css
        modified:   index.html

Unmerged paths:
  (use "git reset HEAD <file>..." to unstage)
  (use "git add <file>..." to mark resolution)

        both modified:   assets/javascript/game.js

@:~/week-4-game <master>$
```

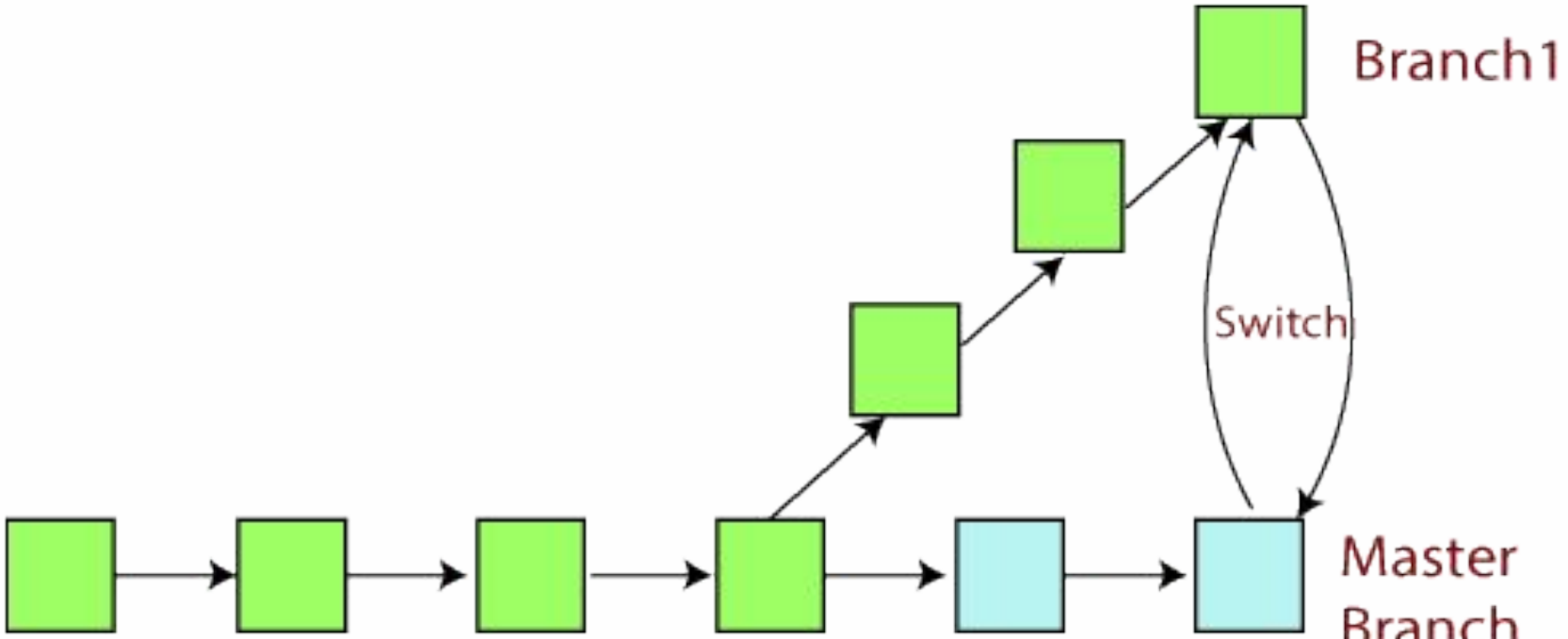
git pull/push



git branch



git checkout



Git Checkout

Играемся с гитом

https://git-school.github.io/visualizing-git/

Практика

1) Создаем репозиторий
2) Создаем ветку
3) Создаем файлы, делаем git add/commit/push
4) Создаем пулл реквест в мастер
5) Добиваемся конфликтной ситуации и разрешаем ее
6) git reset/revert

Структура хранения в GIT

1) https://habr.com/ru/company/badoo/blog/163853/
2) https://git-scm.com/book/ru/v2/Git-%D0%B8%D0%B7%D0%BD%D1%83%D1%82%D1%80%D0%B8-%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D1%8B-Git

Итог

Гита на таком уровне должно быть достаточно=)

Первая бонусная активность

1) сделать свой репозиторий
2) закоммитить в ветку readme
3) создать PR и добавить меня в аппруверы

изи, 5 баллов

https://github.com/organizations/made-ml-in-prod-2021