

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

ДЗ 6

Выполнил:

Пономарев Константин

Группа К33402

Проверил:

Добряков Д. И.

**Санкт-Петербург
2024 г.**

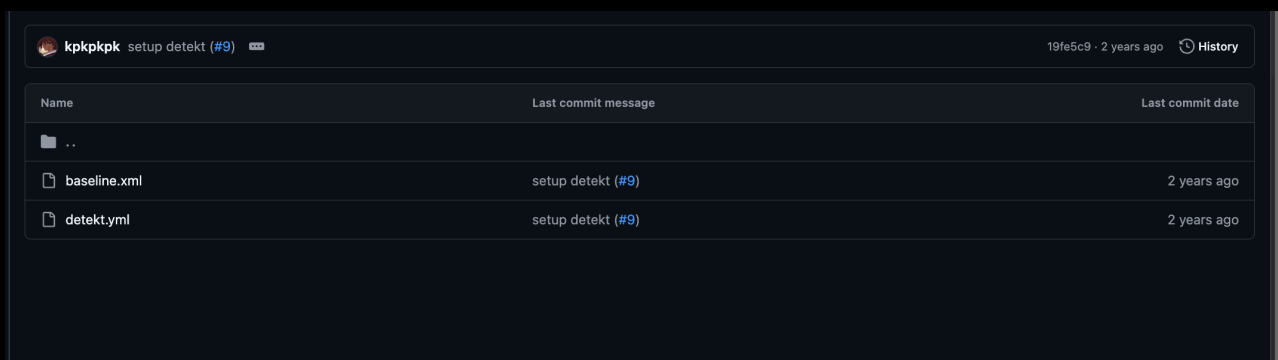
Задача:

Нужно сделать пайп на Github actions или Gitlab

Ход работы:

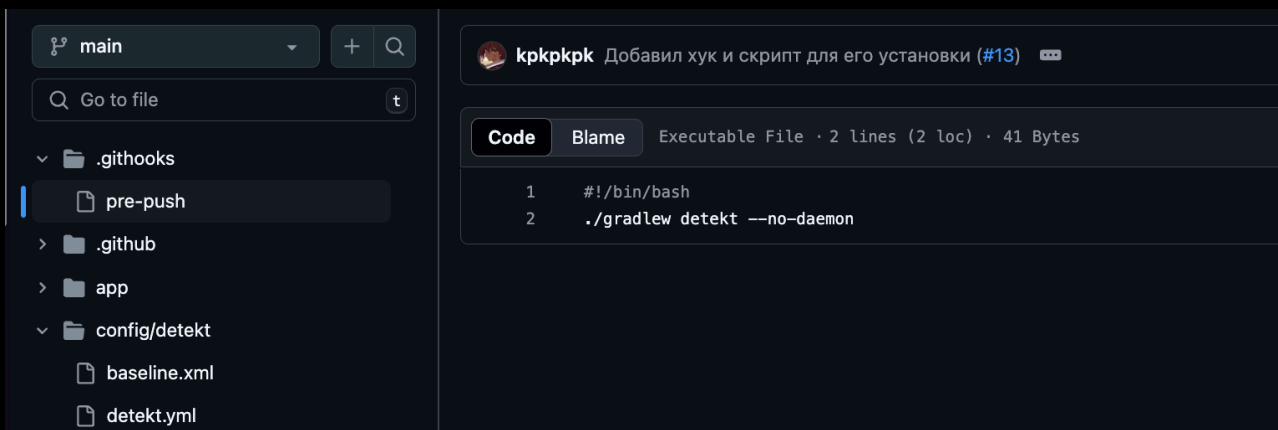
Решил показать вам пример самого базового пайплайна среднестатистического андроид проекта. Когда я был молодым и полным энтузиазма, мечтая перейти в команду платформы, решил наконец ознакомиться с тем, как вообще работает CI.

<https://github.com/kpkrpkpk/OneOfTheProject> вот собственно такой проект, где я решил отточить навыки. На самом деле Github прям очень прост в настройке, мне очень понравилось.



Name	Last commit message	Last commit date
..		
baseline.xml	setup detekt (#9)	2 years ago
detekt.yml	setup detekt (#9)	2 years ago

Для начала засетапил либу detekt, которая будет отслеживать за качеством кода на проекте. Далее решил убить зайца: прогонять детект не только на CI, но и через хуки (неважно, что их можно обойти через `--no-verify` флаг)))



main

Go to file

..

pre-push

github

app

config/detekt

baseline.xml

detekt.yml

kpkrpkpk

Добавил хук и скрипт для его установки (#13)

CodeBlameExecutable File · 2 lines (2 loc) · 41 Bytes

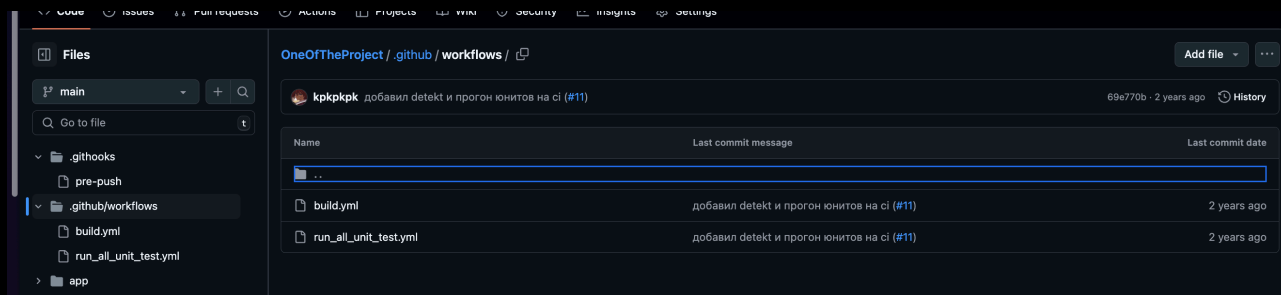
1

#!/bin/bash

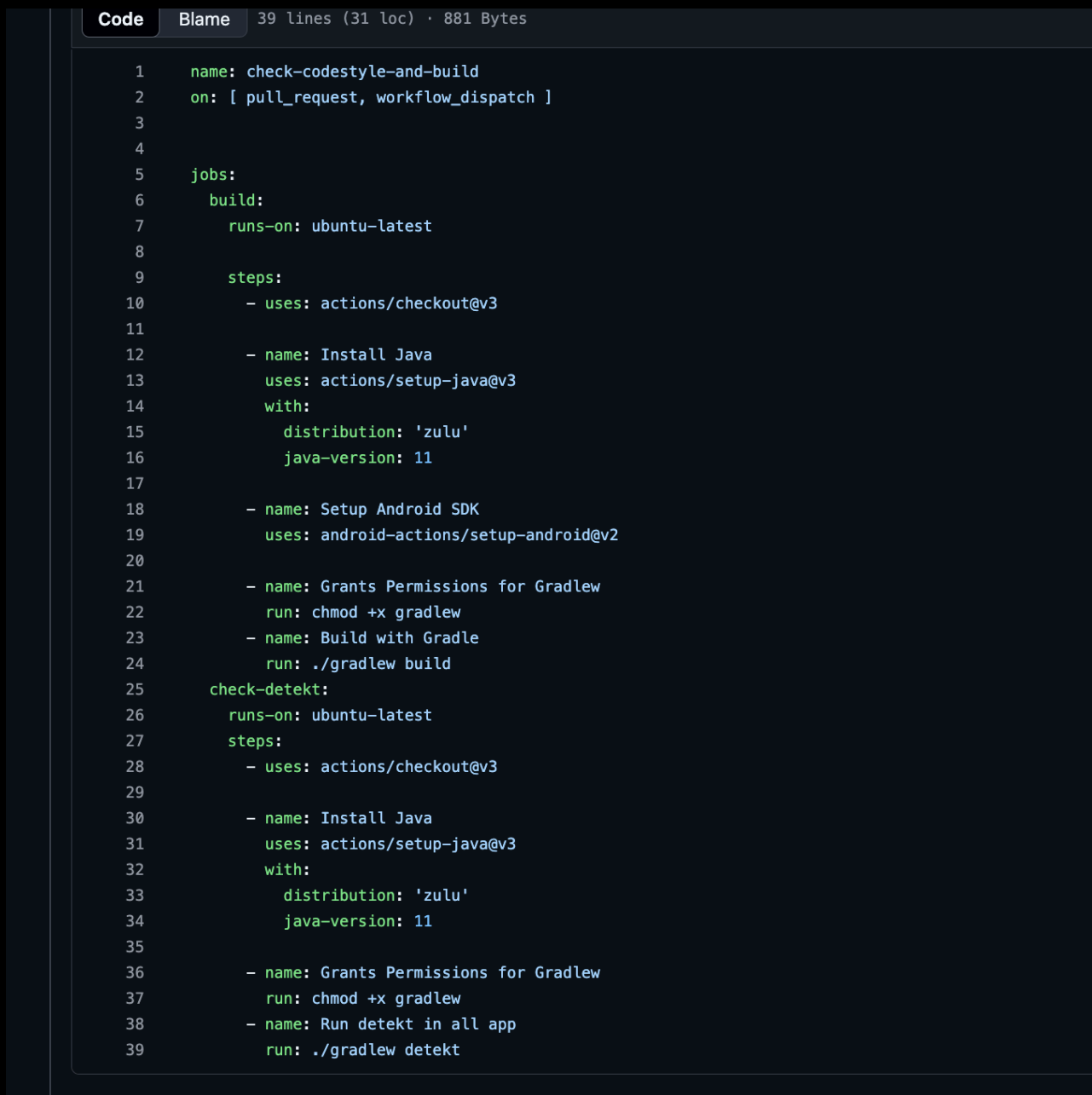
2

./gradlew detekt --no-daemon

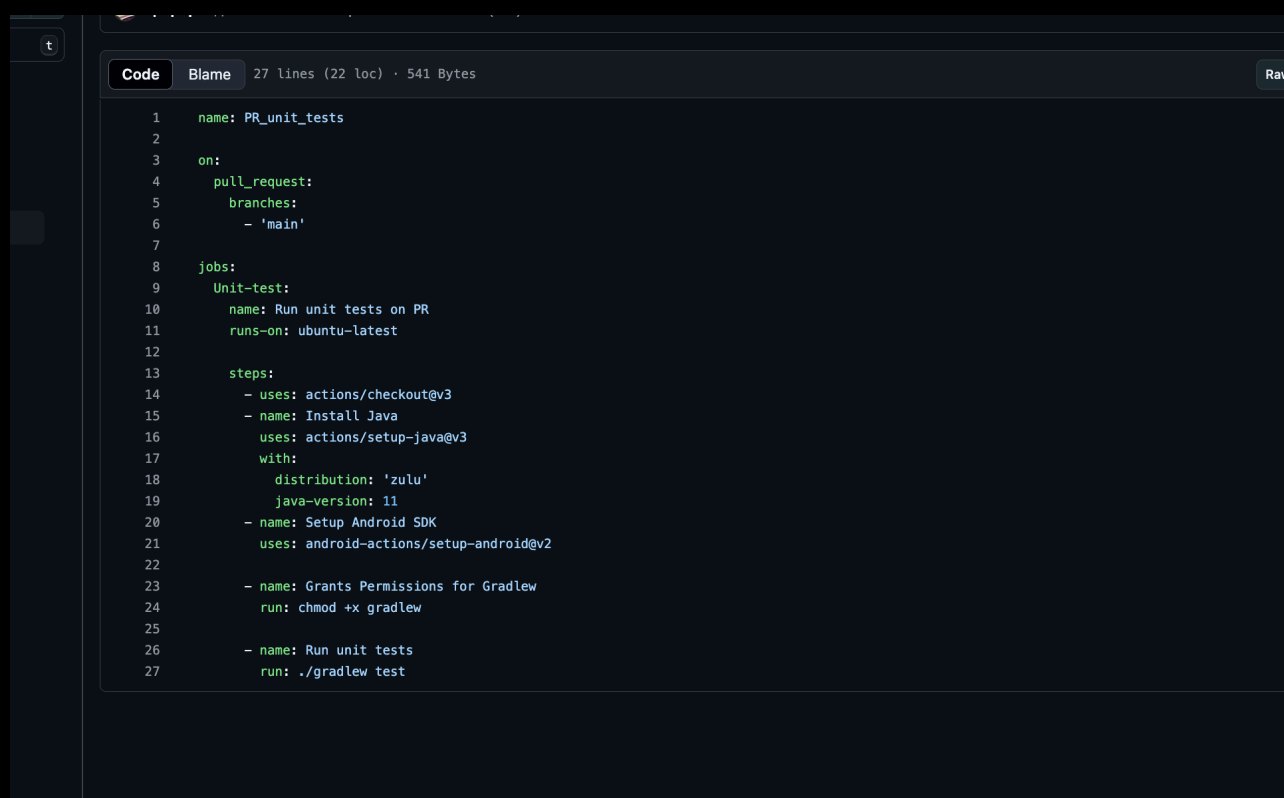
Засетапил дефолт хук на препуш. Ну а теперь настало самое интересное — джобы. Каждый пайп создается в таком ямнике, как и на gitlab



Внутри крутятся джобы:



Даешь имя пайплайну, затем настраиваешь, что когда человек пушит в мр, то пайп триггерится заново. Джобы идут по порядку, зависимы друг от друга, поскольку хранятся все на одной тачке. Ну и получается здесь, как с ts: скачали зависимости java и sdk, запустили сборку через gradlew build, проверили, что не падает. Потом надо проверить качество кода (хотя на самом деле я бы сейчас сначала бы запускал детект, а уже потом билд, потому что билд обычно долгий на больших проектах).



```
1  name: PR_unit_tests
2
3  on:
4    pull_request:
5      branches:
6        - 'main'
7
8  jobs:
9    Unit-test:
10     name: Run unit tests on PR
11     runs-on: ubuntu-latest
12
13     steps:
14       - uses: actions/checkout@v3
15       - name: Install Java
16         uses: actions/setup-java@v3
17         with:
18           distribution: 'zulu'
19           java-version: 11
20       - name: Setup Android SDK
21         uses: android-actions/setup-android@v2
22
23       - name: Grants Permissions for Gradlew
24         run: chmod +x gradlew
25
26       - name: Run unit tests
27         run: ./gradlew test
```

Ну и допом докинул юнит тесты туда, чтобы точно петпроект не сломался!!

Вывод:

Хоть я и рассказал об этом так, будто бы это просто, но на самом деле - это только самые простейшие пайплайны. На примере гитлаба, там можно вообще обращаться к ботам, которые будут за тебя двигать тасочки в jira, проверять код

стайл, тесты, билд, бампать версию после мержа и так далее. Возможности ограничены лишь способностями делающего