# Обзор Intellij IDEA

Круглов Иван, Кузнецов Илья

22 сентября 2022 г.

# Мы будем исследовать такие функции как

# Создание проекта

При открытие IDE мы попадаем на стартовое окно создания/загрузки проекта

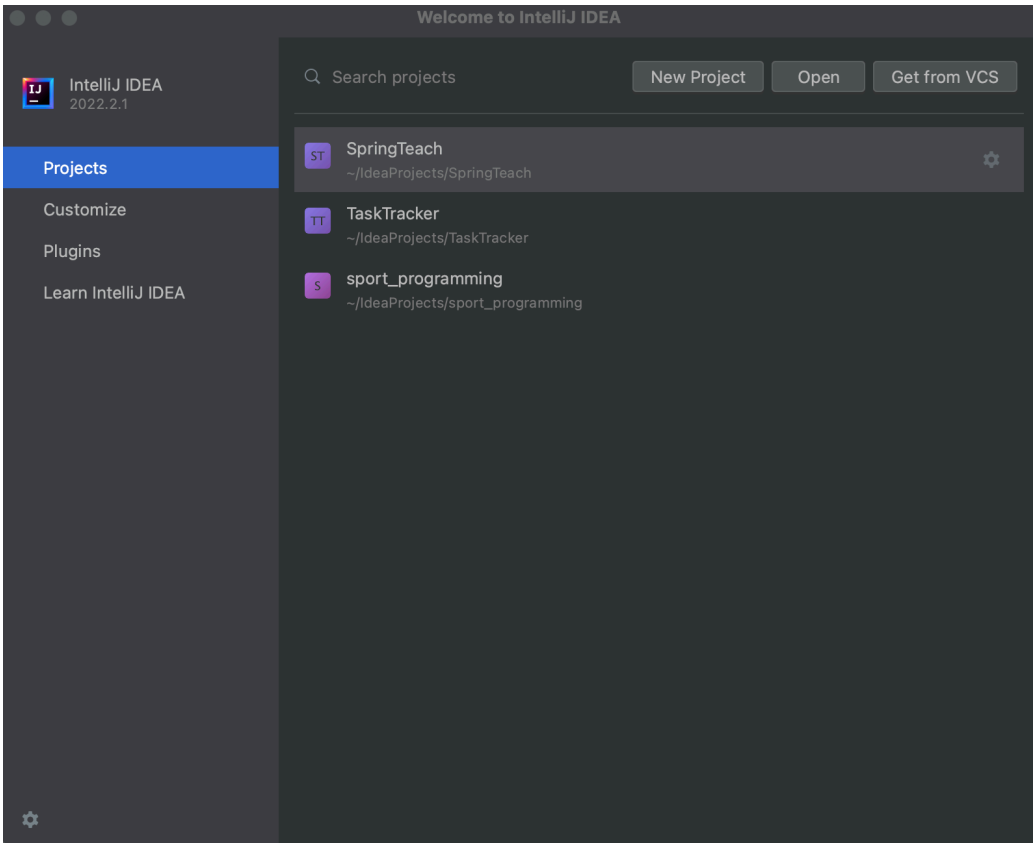


Рис. 1: Выбор дальнейшего действия

Далее мы можем сделать выбор, что мы хотим сделать с проектом:

- Создать новый
- Открыть существующий
- Или же скопировать проект из системы контроля версий (git, github, gitlab, etc.)

## Создаем новый проект

Производим первоначальную настройку нашего проекта:

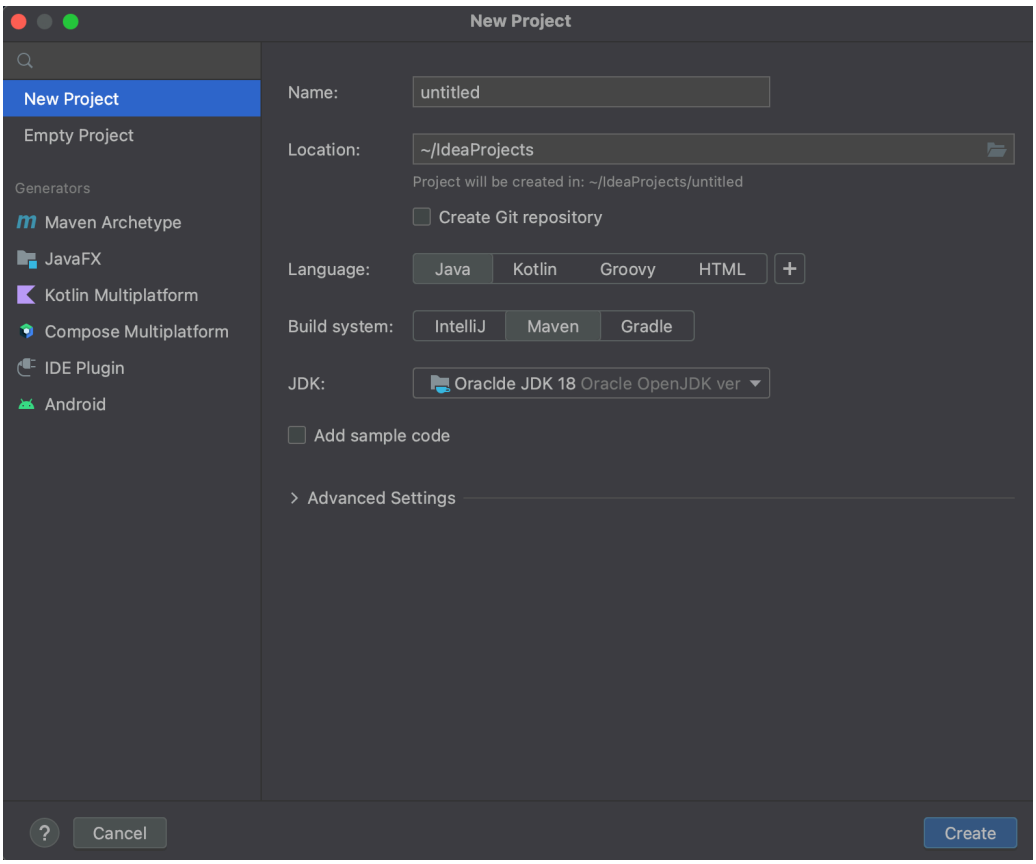


Рис. 2: Создание проекта

Обычно для настройки нам достаточно

- Ввести название проекта
- Выбрать ЯП (для нас это будет java)
- Выбрать систему сборки (Рассмотрим в случае с Maven)
- Создать или не создавать локальный репозиторий для проекта
- Выбрать доступную JDK
  Если таковых не существует, в том же окне нам предложат авоматически установить новую

## Копируем проект из VCS

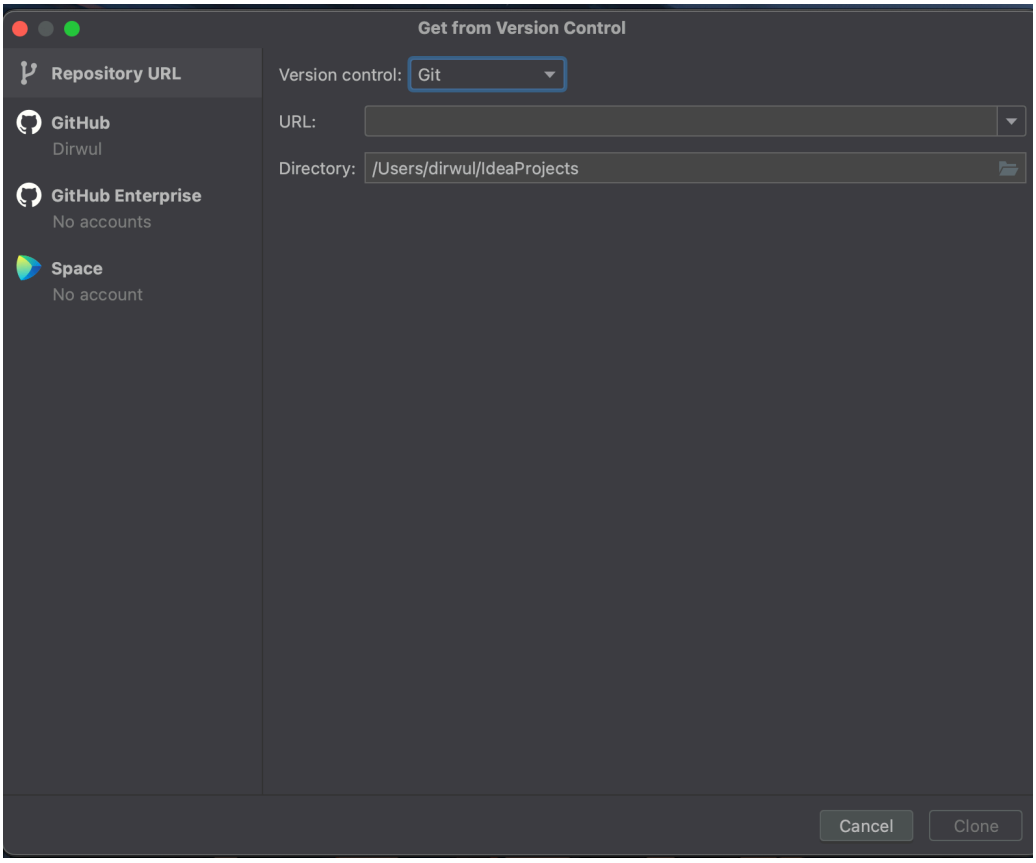


Рис. 3: Выбор VCS

В данном окне нам предоставляется выбор:

- Скопировать репозиторий по URL
- Залогиниться в GitHub и выбрать репозиторий оттуда
- Выбрать репозиторий из GitHub Enterprise
- Выбрать репозиторий из JetBrains Space

# Создание первой программы на Java

## Создание java class'а

Для создания первой программы на Java нам потребуется спуститься по папкам: src -> main -> java и создать там наш первый Java-class.

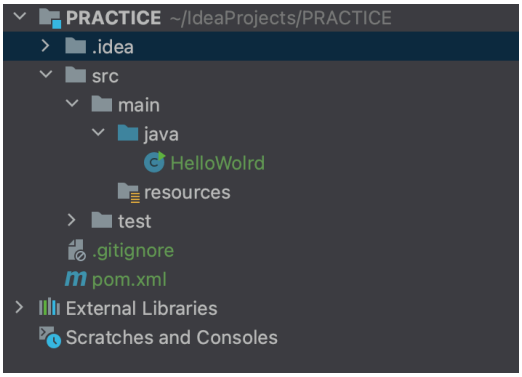


Рис. 4: Правильная директория для класса в Maven

Так же при создании класса IDE предложит добавить наш файл в git(отслеживать его)

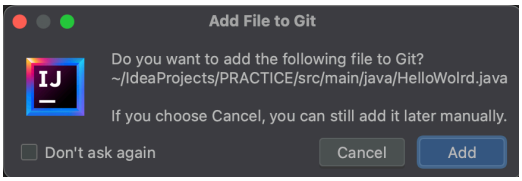


Рис. 5: Добавление в git

Исходный код нашего Hello world выглядит так:

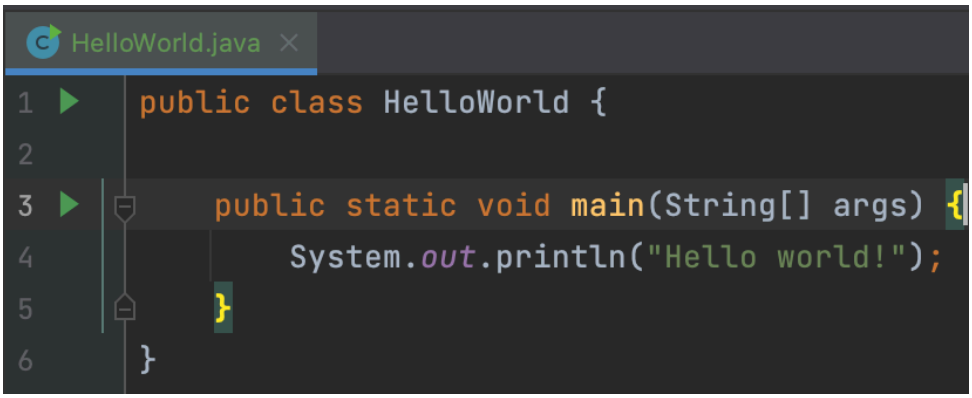


Рис. 6: Hello wolrd

## Создание конфигурации для запуска

Чтобы запустить наш java class нам потребуется создать конфигурацию и настроить ее:

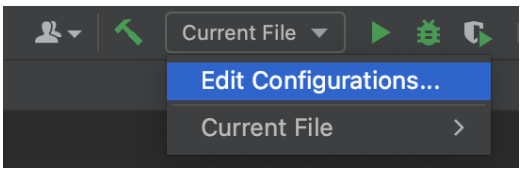


Рис. 7: Заходим в настройки конфигурации

В настройках конфигурации нас интересует два параметра:

- Название конфигурации, которое ни на что не влияет, но в реальном проекте у нас может быть несколько конфигураций запуска
- Main класс для запуска (HelloWorld)

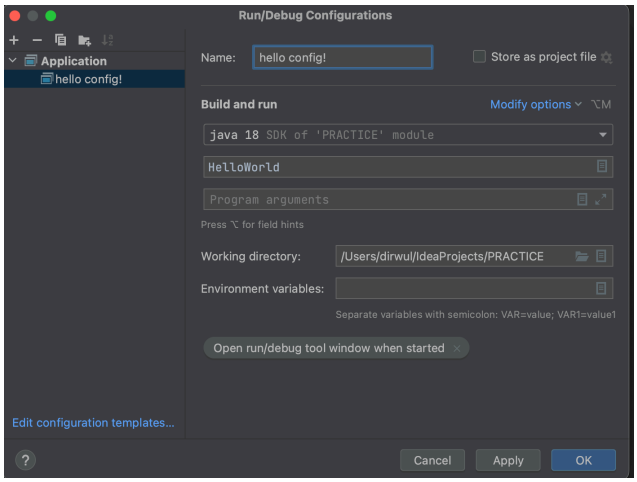


Рис. 8: Создаем конфигурацию

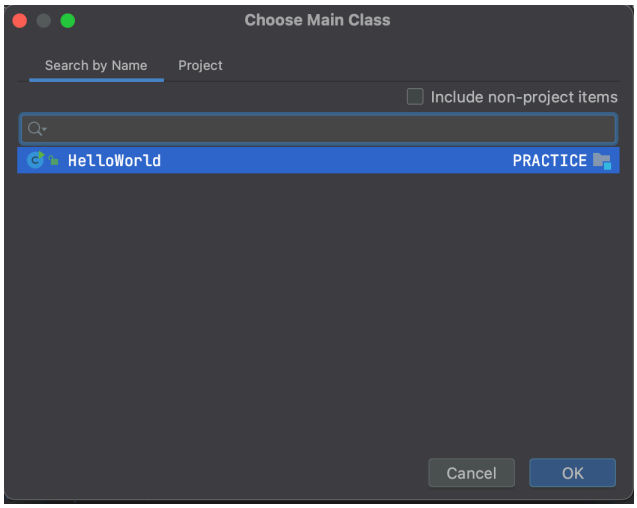


Рис. 9: Окно выбора main-class'а

# Запуск и отладка

## Стандартный запуск

*Здесь и далее, все использованные хоткеи будут озвучиваться для MacOS, однако на windows в бОльшинстве слуаев достаточно заменить системные клавиши на эквивалетные*

Для запуска программы нам достаточно нажать *ctrl+R* и созданная нами ранее конфигурация запустит наш Java class. После чего снизу откроется окно консоли с выводом программы на экран:

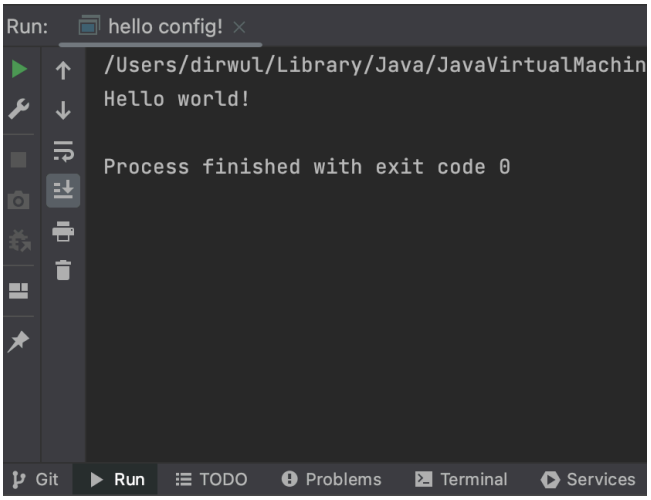


Рис. 10: Output window

## Запуск в режиме отладки

Теперь немного усложним программу, добавим в нее две переменные и вывод на экран "Hello break point". Вывод этой строки на экран отметим как breakpoint.

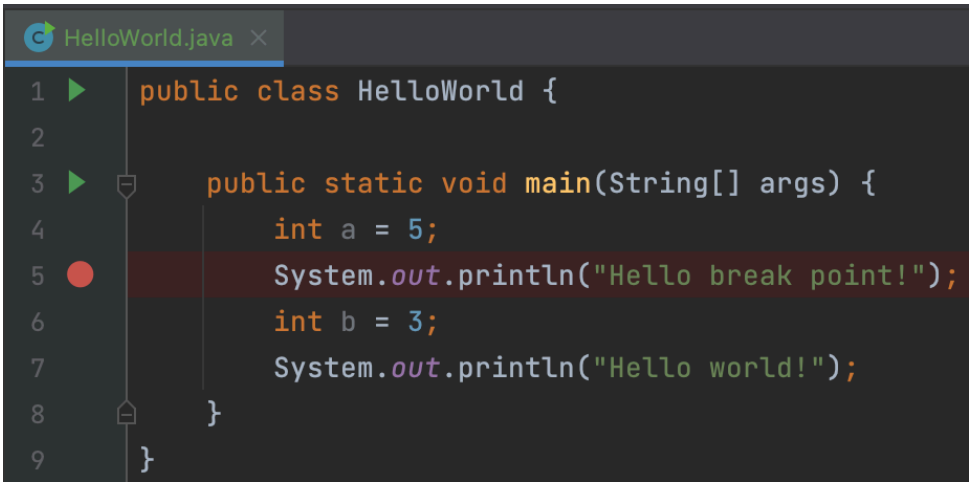


Рис. 11: Создание breakpoint'a

Теперь запустим нашу программу в режиме отладки *(ctrl+D)*

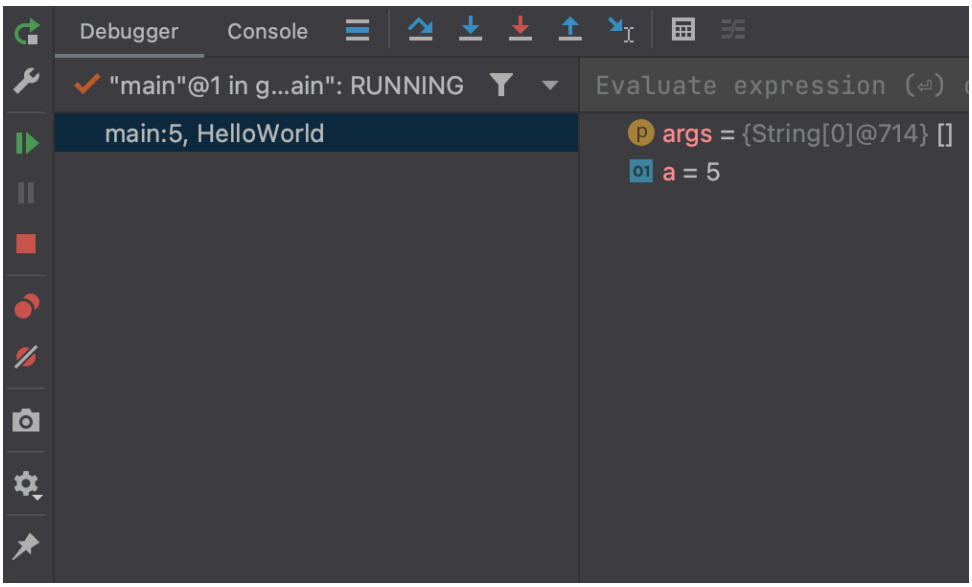


Рис. 12: Состояние переменных на момент остановки у breakpoint'а

То есть в режиме отладки IDE позволяем нам перемещаться от одного breakpoint'а к другому, зная состояние каждой переменной на каждом шагу, что, несомненно, очень удобно при отладке приложений.

# Настройки и форматирование

Общие настройки проекта, форматирования и т.д. доступны при нажатии на *command* + , Рассматривать их не имеет смысла в данном случае, т.к. их слишком много и каждый подбирает настройки под себя Поэтому пройдемся по основным пунктам:

- Общие настройки IDE
- Хоткеи
- Редактор кода
- VCS и все с ними связанное
- Настройки билда, компилятора и дебаггера
- Находящиеся под управлением IDE языки и фреймворки
- Прочие инструменты
- Расширенные настройки

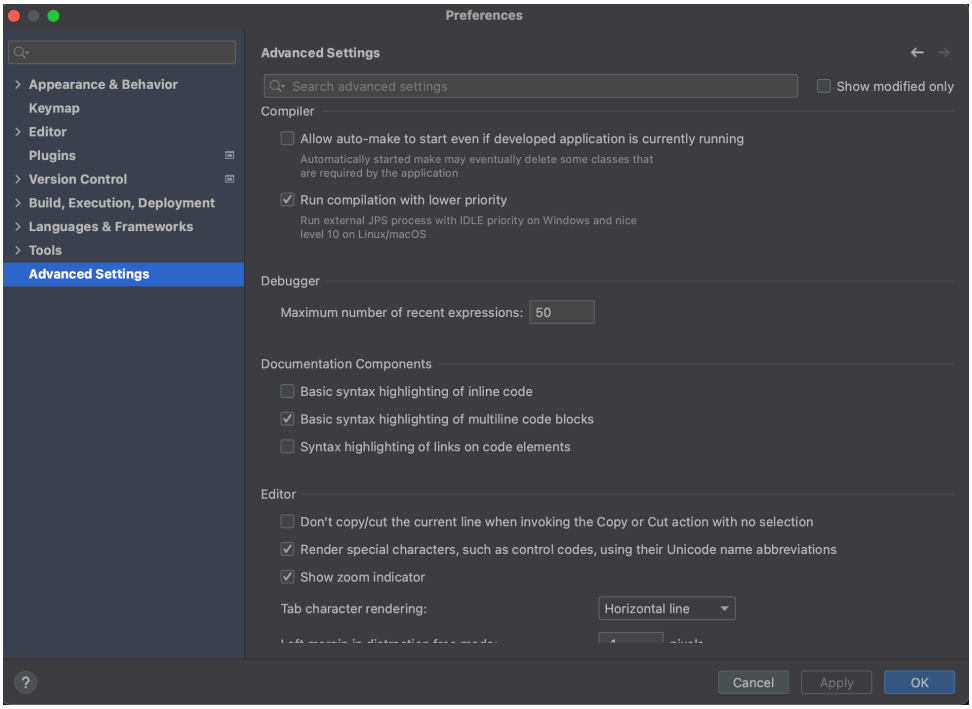


Рис. 13: Настройки

Форматирование кода для всего файла можно запустить через *option* + *command* + *L*


Рис. 14: Форматирование

# Настройка репозитория

Чтобы создать удаленный репозиторий, нам достатоточно запушить что-либо в него. Предварительно так же требуется залогиниться в github/gitlab/other vcs. Рассмотрим пример с гитхабом, где нам доступно две опции:

- Редирект ссылка на github
- Создание специального токена

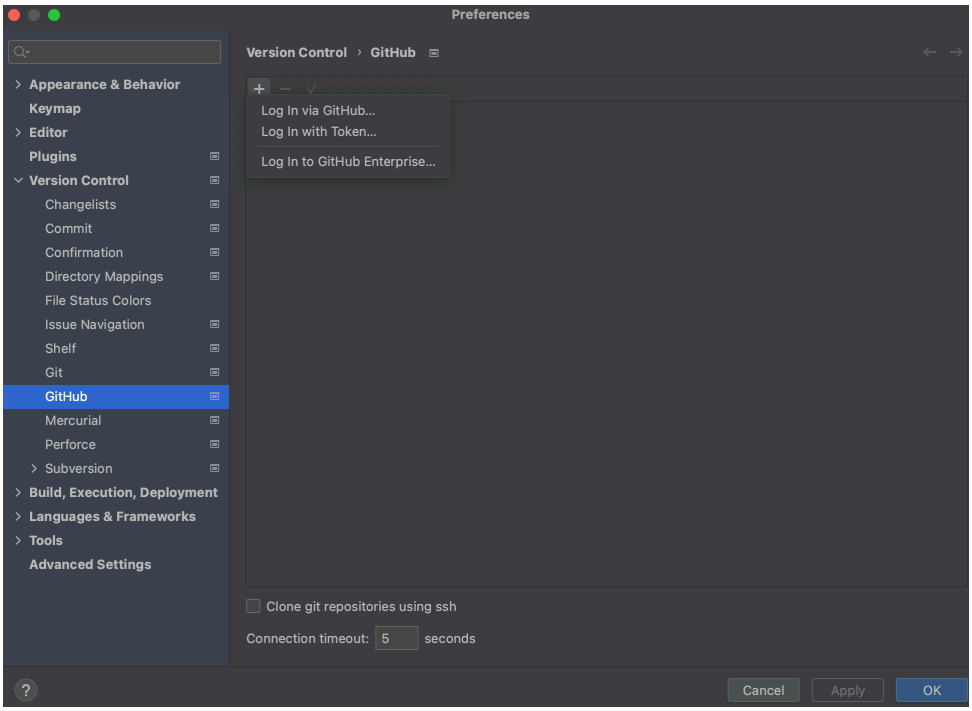


Рис. 15: Логин в гитхаб

Также нам требуется обновить .gitignore Сделаем папку *.idea* и файл *pom.xml* нетрекаемыми *(я не знаю как правильно это называется)* для git

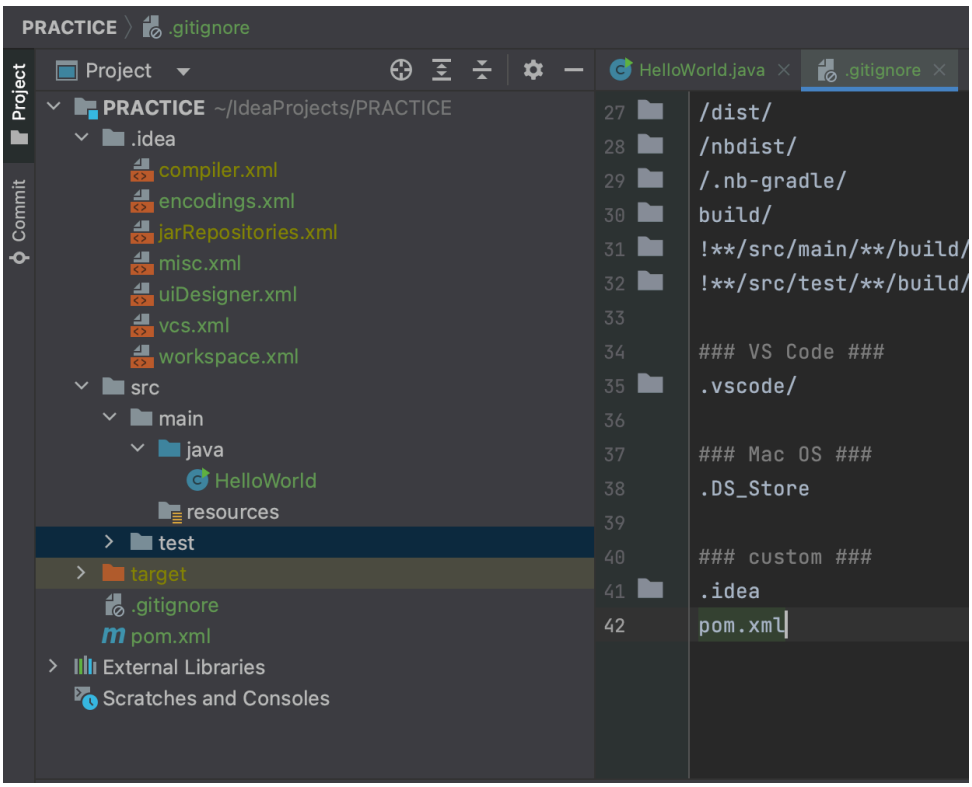


Рис. 16: Обновленный .gitignore

Однако, как вы могли заметить, файл gitignore обновлен, а некоторые файлы в папке *.idea* и файл *pom.xml* все еще отображаются как обновленные

*Ремарка: когда в Intellij IDEA или VSCode меняется файл gitignore, все изменения (обновлен ли файл или может быть он untracked) видны в тот же момент*

Это небольшая проблема, которую приходится чинить каждый раз, когда создается новый репозиторий в IDEA Достаточно прописать в терминале среды git rm –cached -r -f .idea и git rm –cached -r -f pom.xml

```
dirwul@MacBook-Air PRACTICE % git rm --cached -r -f .idea
rm '.idea/encodings.xml'
rm '.idea/misc.xml'
rm '.idea/uiDesigner.xml'
rm '.idea/vcs.xml'
rm '.idea/workspace.xml'
```

Рис. 17: Изменение кэша локального гита

Таким образом, мы убрали папку и файл из кэша гита и теперь у нас все отображается (и работает) верно.

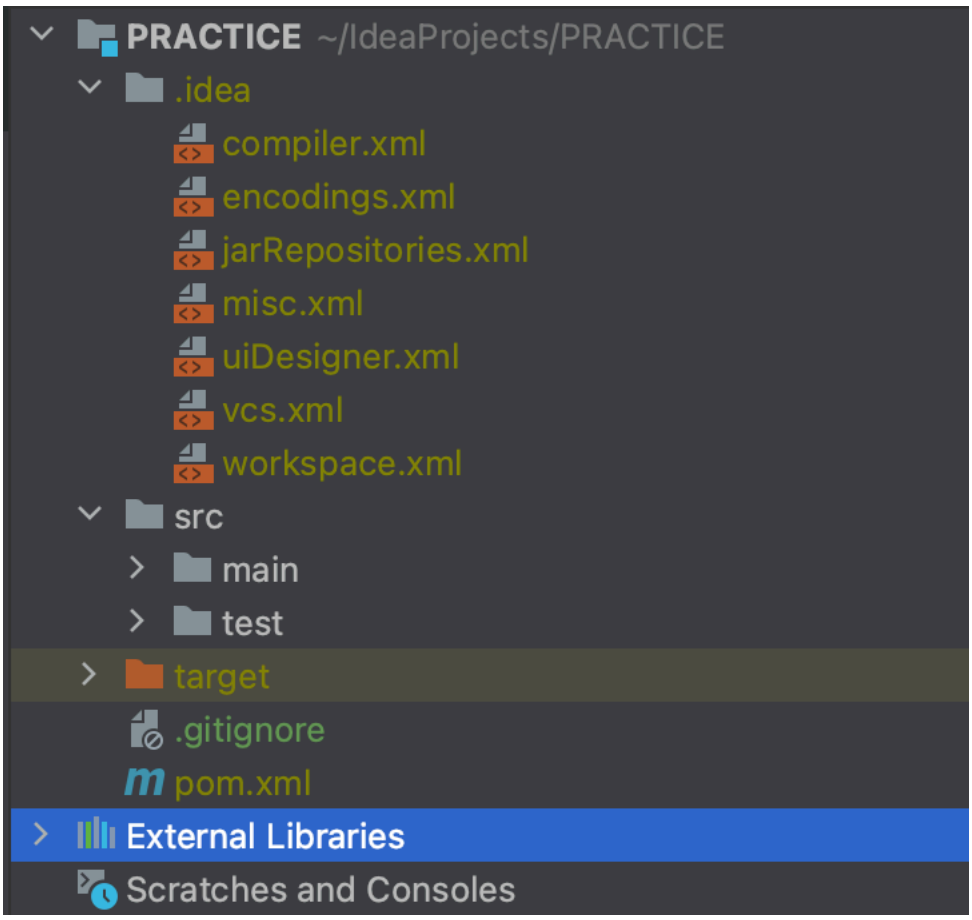


Рис. 18: Правильное состояние файлов

# Базовые операции с удаленным репозиторием

## New branch

Создаем новую ветку, которая наследуется от master и делаем в нее checkout (автоматически)

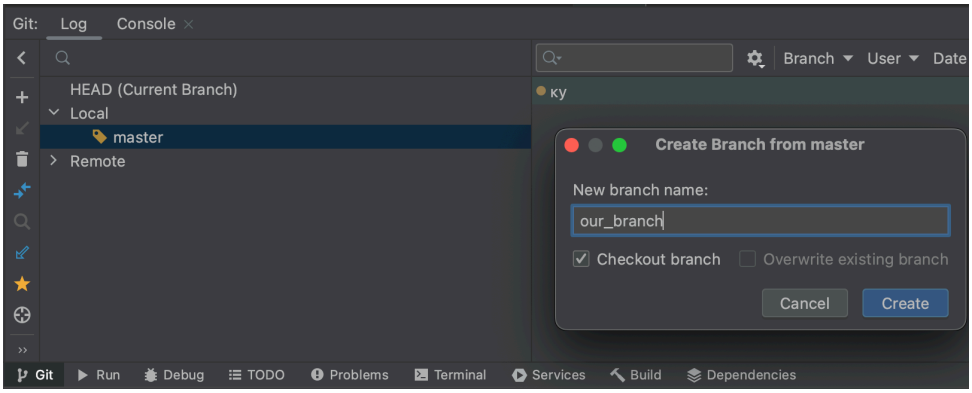


Рис. 19: Новая ветка

## Commit

Хоткей: *ctrl + K* Создаем коммит, указываем сообщение, прикрепленное к комиту и выбираем файлы, которые мы хотим внести в данный коммит

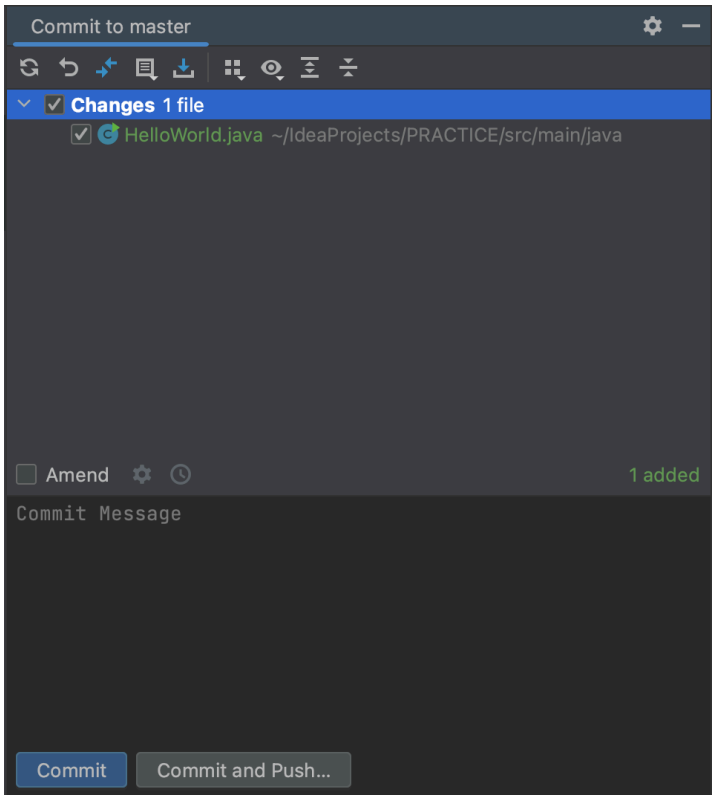


Рис. 20: Коммит

## Push

Хоткей: *shift + ctrl + K* Пушим уже имеющийся в локальном репозитории коммит

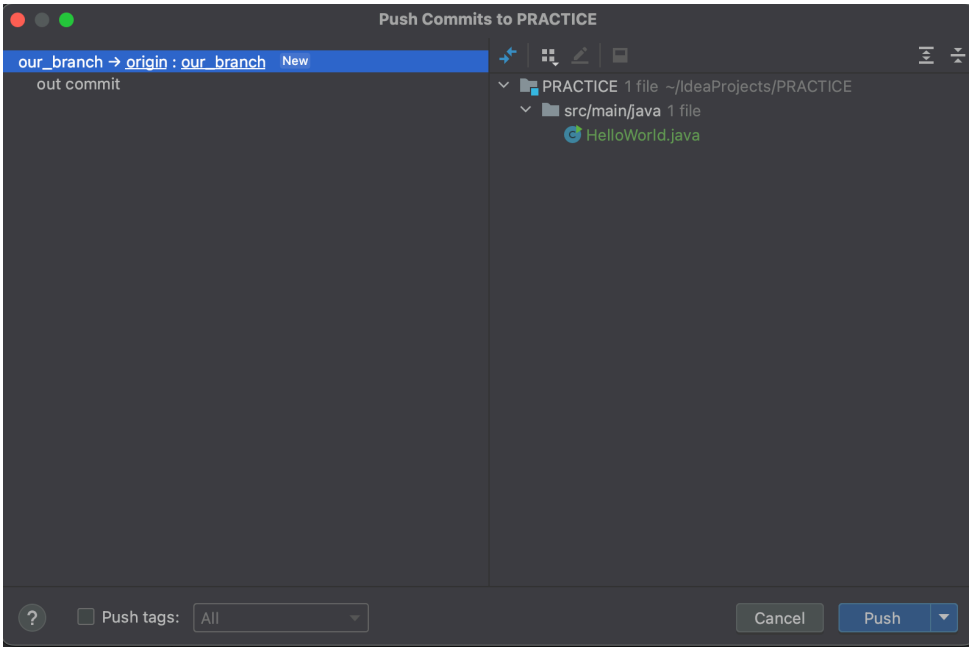


Рис. 21: Пуш

## Merge

Заходим в PR (PullRequest) панель слева и нажимаем *command + N* Создаем наш PR и сразу же появится предложение сделать Merge.

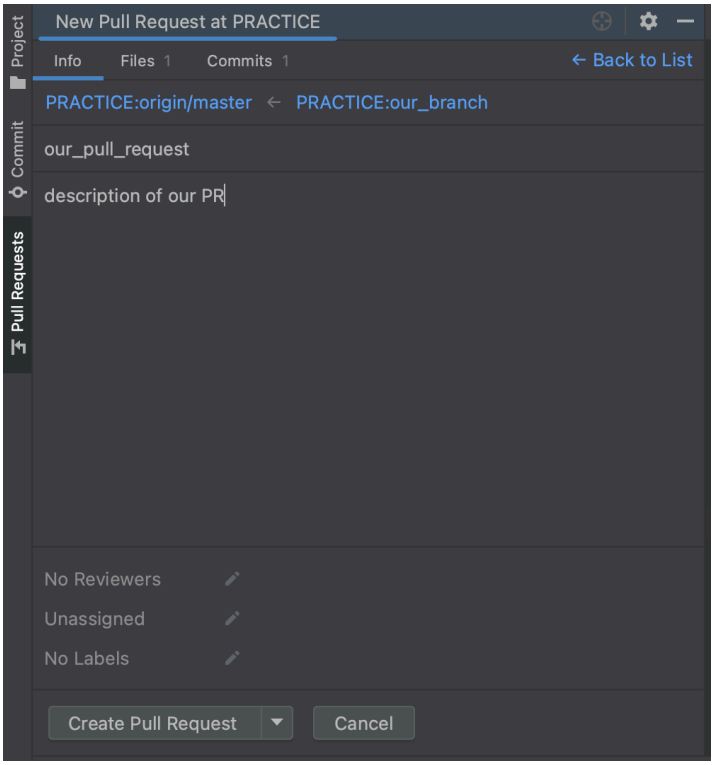


Рис. 22: Pull Request

После чего делаем merge нашей ветки в master, если не требуется разрешить конфликт Если требуется - разрешаем конфликт и делаем merge

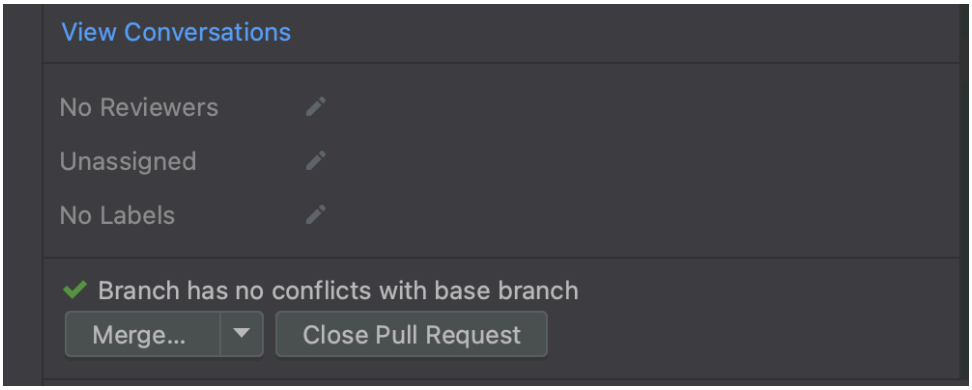


Рис. 23: Merge

Далее мы переключаемся (checkout) на master ветку и нажимаем *command* + *T* IDEA предлагает нам обновить нашу локальную master ветку

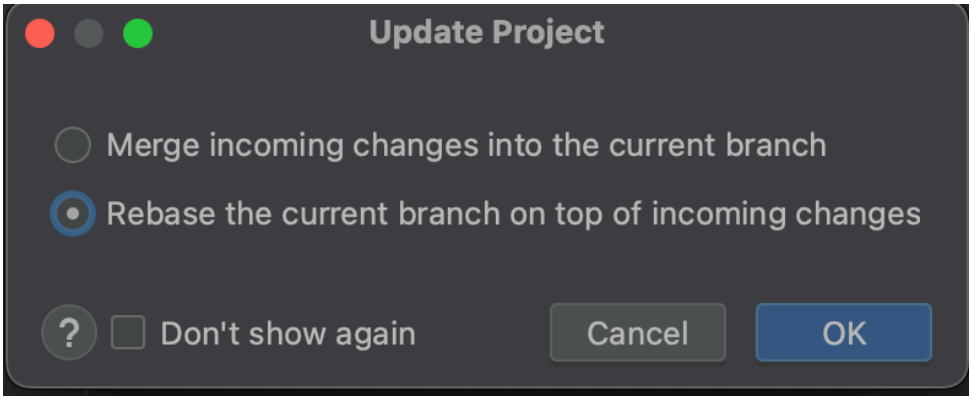


Рис. 24: Rebase

Делаем rebase и радуемся жизни! Все изменения так же видны в виде графа снизу:

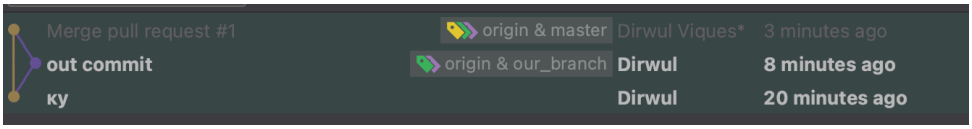


Рис. 25: git actions