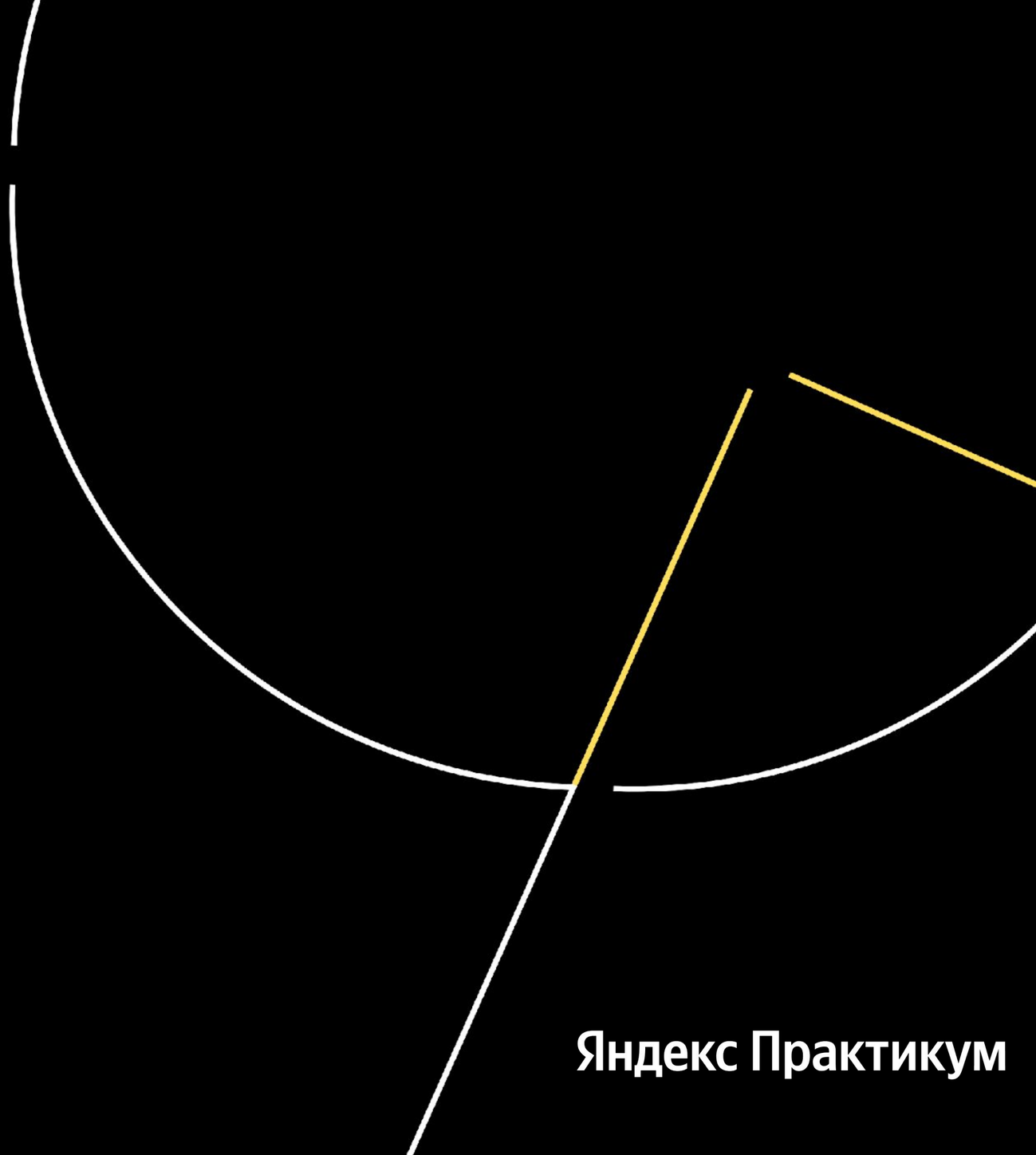
Яндекс.Практикум

Makefile, mocks, linters



Яндекс Практикум

Что нас сегодня ждет

- Научимся создавать и использовать моки
- Познакомимся с make
- Поанализируем код с помощью линтеров

Go mocks

Mock-объект – фиктивная реализация интерфейса для тестирования зависимых от него компонентов.

Сравнение основных библиотек для мокирования можно найти здесь:

https://gist.github.com/maratori/8772fe158ff705ca543a0620863977c2

Мы будем рассматривать мокирование на примере gomock:

https://github.com/uber-go/mock

Паттерн tools + mockery

Существует удобный паттерн управления зависимостями для разработки и тестирования (т.н. “devDependencies”).

Расмотрим его на примере пакета *mockery*.

Makefile

Makefile - файл конфигурации утилиты *make*. Правила описываются на специальном (тьюринг-полном!) языке.

Как правило используется для автоматизации сборки програм. Особенно полезно, если этапы сборки зависят друг от друга.

Makefile помещается в корень проекта. Правило по умолчанию обычно позволяет полностью собрать приложение.

Линтеры

Go поставляет несколько стандартных линтеров:

- *gofmt*
- *govet*

VSCode использует несколько дополнительных линтеров.

Стандартные линтеры ловят не все проблемы :(

golangci-lint

Для Go написано много дополнительных линтеров. golangci-lint позволяет конфигурировать и запускать их для проверки кода проекта.

Преимущество: единая конфигурация, можно использовать как локально, так и в CI/CD

slog

Проблема пакета ***log*** – остутствие возможности (полу)-структурированного логирования

slog добавляет в стандартную библиотеку универсальный инструмент структурированного логирования

Блог-пост для базового понимания: https://go.dev/blog/slog

Спасибо за внимание!

Вопросы?