LeanCode

# Programming Mobile Applications in Flutter

Intro lecture

## Who are we?

- mgr inż. Jakub Fijałkowski
- Contact: jakub.fijalkowski@leancode.pl
- Backend Guild Leader at LeanCode
- ~10 yrs of backend development, some experience with mobile
- Rust, .NET, Cloud, DevOps
- Books, Golf, Gigs

## Who are we?

- inż. Mateusz Wojtczak
- Contact: mateusz.wojtczak@leancode.pl
- Mobile Guild Leader at LeanCode
- > 6 years of mobile applications development
- Flutter, React Native, iOS, Xamarin, Android
- producing music after work

LeanCode

## Who are you?

We build digital products.

LeanCode

## What is it all about?

We build digital products.

LeanCode

## Rules

- Points 0-100:
  - **51-60pt - 3**
  - **61-70pt - 3.5**
  - **71-80pt - 4**
  - **81-90pt - 4.5**
  - **91-100pt - 5**

- Project - 100pt
- Activity during lectures - 10pt

## Lectures

1. Introduction and Dart
2. Introduction to Flutter 1
3. Introduction to Flutter 2
4. State Management
5. Async and HTTP
6. State Management with External Libraries
7. Architecture and Dependency Injection
8. Storing Data

## Lectures

- Forms
- Firebase
- Code Generation and Internationalization
- Animations
- Flutter Web and Flutter Desktop
- Communication with Native
- **Waiting for Proposals**

## Labs

1. Getting Started with Flutter
2. Layouts 1
3. Layouts 2
4. Communication with API
5. State Management
6. CodeMagic
7. Authorization

# Project

- Individual multi-layer Flutter application that works at least on one mobile platform (Android/iOS)
- Application should contain at least two screens
- Application should communicate with 3rd party API OR use other data persistency solution
- Application's topic and scope is defined by the student, should be described in initial documentation and approved by the lecturer

## Sample Projects

- TODO List with authorization and synchronization between devices
- Chat with authorization
- Shopping list with categories, search, history
- Feed using 3rd party API
- Online shop with deep links, categories, filters, sort, cart
- PoI browser - map and list, tags, categories, sort by localization

# Documentation

- Initial Documentation
    - Project Description
    - Use cases
- Final Documentation
    - Project description
    - Integrations
    - List of optional requirements
    - Instruction
    - Test account (if applicable)
    - Database/Firestore schema (if applicable)
    - CI/CD description/screenshot (if applicable)

## Example initial docs

- Chat with Authorization
- Description
  - Screens list and short description
    - Login Screen
    - Channel List Screen
    - Message List Screen
- Use cases
  - As a User, I can sign in using Google/Facebook/Instagram/Apple account
  - As a User, I can see a list of channels
  - As a User, I can create/delete/leave channel if I have sufficient permissions
  - As a User, I can send plain text messages
  - As a User, I can send images/videos/files
  - As a User, I can edit messages
  - As a User, I can reply in thread

## Assessment Rules

Implementation of the required project assumptions - **50pt**

- Initial documentation - **5pt**
- Implementation of a multi-layer application - **15pt**
- Code quality - **10pt**
- UI/UX - **10pt**
- Final documentation - **10pt**

**Adherence to the schedule - 10pt**

## Assessment Rules

Optional requirements (**max 50pt)**

- Support for additional platform (Android/iOS/Web/Desktop) - **5pt** each
- Implementing BLoC pattern - **10pt**
- Animations - **10pt**
- Tests - **10pt**
- Signing in process - **10pt**
- Complex form with validation - **10pt**
- CI/CD - **5pt**
- Platform Channels - **10pt**
- Internationalization - **5pt**
- Accessibility - **5pt**
- Custom painting - **10pt**
- Deep links - **10pt**
- Using Camera/Bluetooth/Other platform features - **10pt**
- Offline support - **20pt**

## Timeline

- **24.10.2022** - Initial documentation (Labs)
- **23.01.2023** - Project Submission
    - Source Code and Final Documentation
- **06.02.2023** - Late Project Submission
    - Each day of being late will take a decrease of **5pt** from the total number of gained points

# Any questions?

We build digital products.

LeanCode

[https://github.com/leancodepl/flutter-at-mini](https://github.com/leancodepl/flutter-at-mini)

We build digital products.

LeanCode

## Dart

# Dart

- Statically typed
- Object-oriented
- Garbage-collected
- C-style syntax
- Asynchrony support
- Non-nullable by default
- Open source, developed by Google

```
int fibonacci(int n) {
  if (n == 0 || n == 1) return n;
  return fibonacci(n - 1) + fibonacci(n - 2);
}

var result = fibonacci(20);
```

# Dart

- Designed for client development
  - Optimized for UI & Flutter
  - Productive Development - Make changes iteratively: use hot reload to see the result instantly in your running app
- Compiled to ARM & x64 machine code for mobile, and desktop.
- Compiled to JavaScript on web.
- Dart VM with just-in-time (JIT) compilation and an ahead-of-time (AOT) compiler for producing machine code.

## Why Dart?

## Why Dart?

- Flutter used four primary dimensions for evaluation, and considered the needs of framework authors, developers, and end users:
    - Developer productivity
    - Object-orientation
    - Predictable, high performance
    - Fast allocation
- **Opportunity to work closely with the Dart community, which is actively investing resources in improving Dart for use in Flutter**

## Why Dart? - TLDR

- It looks like JavaScript
- It is fast enough (we have 16.6 ms to render a frame)
- It's garbage-collected
- It can perform tree shaking to reduce code size
- It's tightly connected to Flutter use cases
- It's targeted to run on mobile/desktop/web.

dartpad.dev

## Show me the code!

We build digital products.

LeanCode

# Questions?

We build digital products.

LeanCode