

Multiplatform App Development Flutter

WF-ENG, IMA17

Lab-Session Weiland

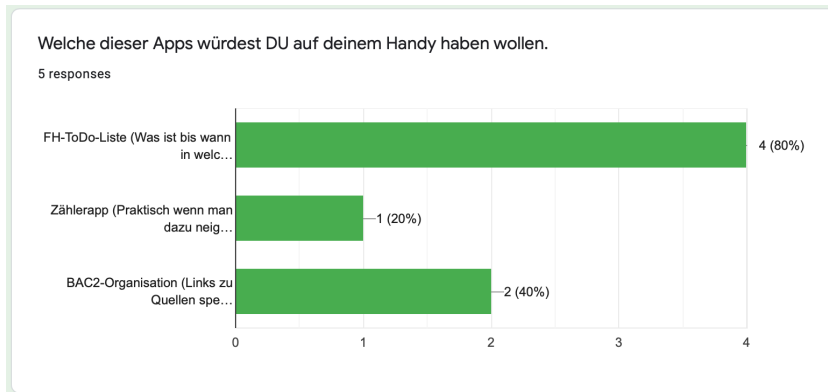
Agenda

- Feedback Online Course
 - Successful Setup?
- Introduction
- Cloning & Setup
- Expanding the App
- Questions?

INTRODUCTION

Background

- Majority vote decided on FH Manager

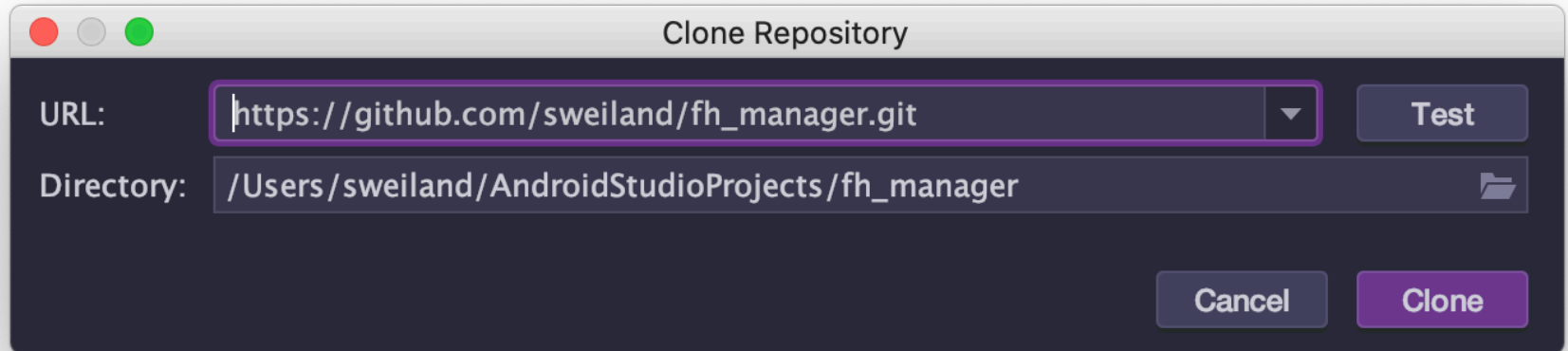


- Based on [Fluttery Todo](#) (MIT License, great preconditions)

CLONING & SETUP

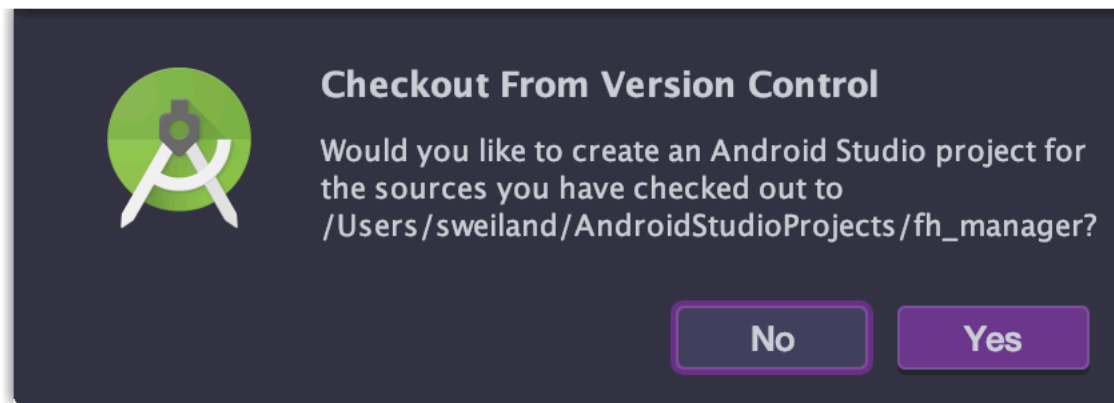
Cloning

- Clone Repository from Link in Moodle using Android Studio



Cloning contd.

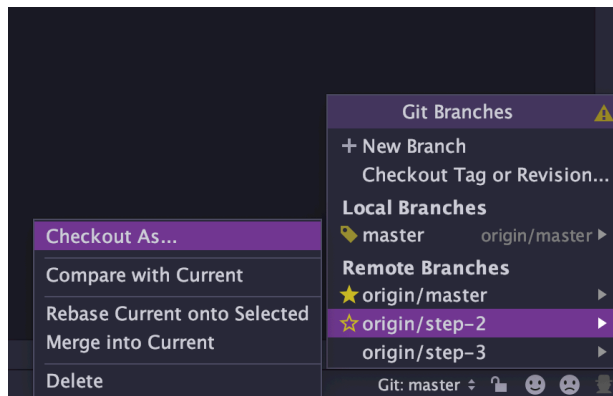
- Allow creation of AS-Project from source files



- “Create Project from existing sources”
- Next until finished

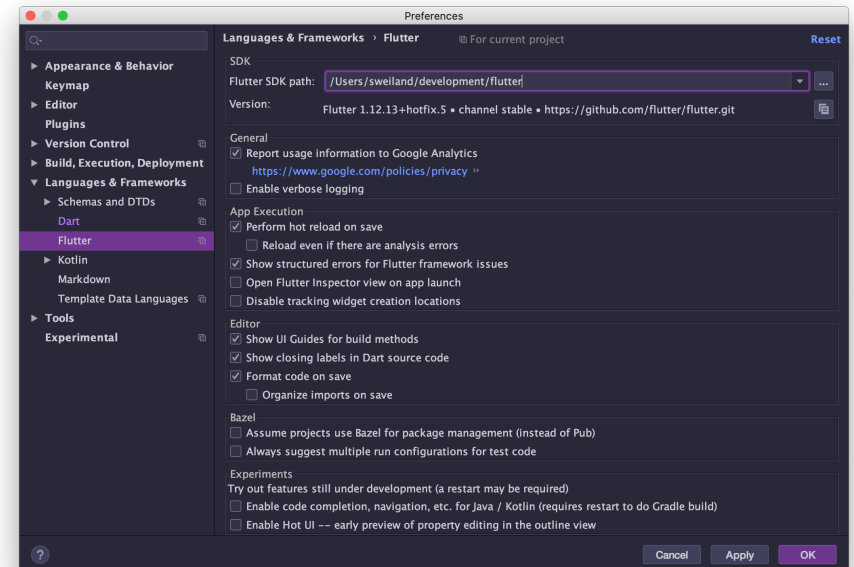
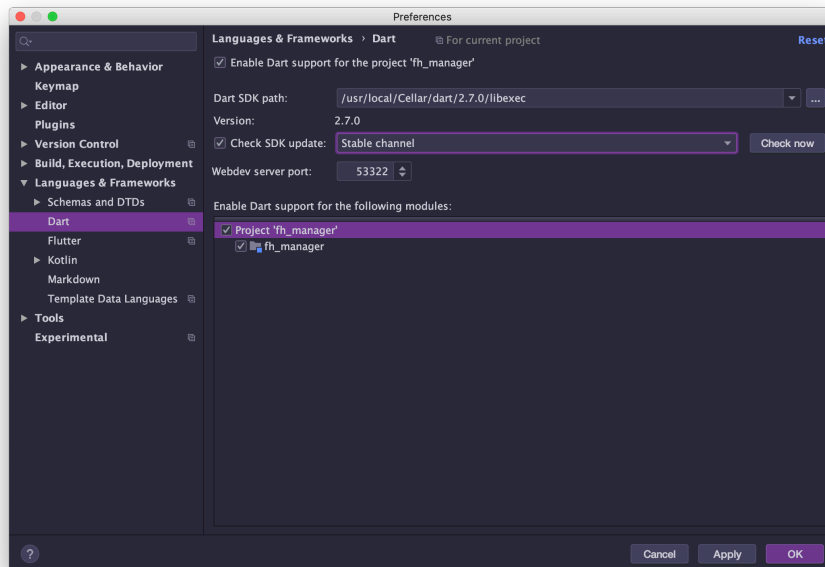
Setup

- Switch Branch to 'step-2'



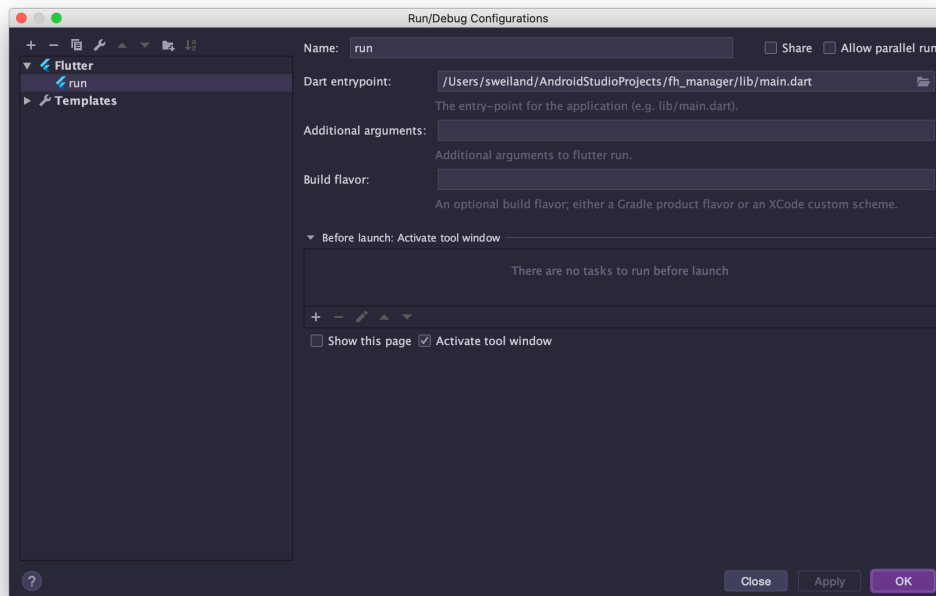
- Open Settings and add your installations of Flutter and Dart under "Languages & Frameworks"

Setup contd.



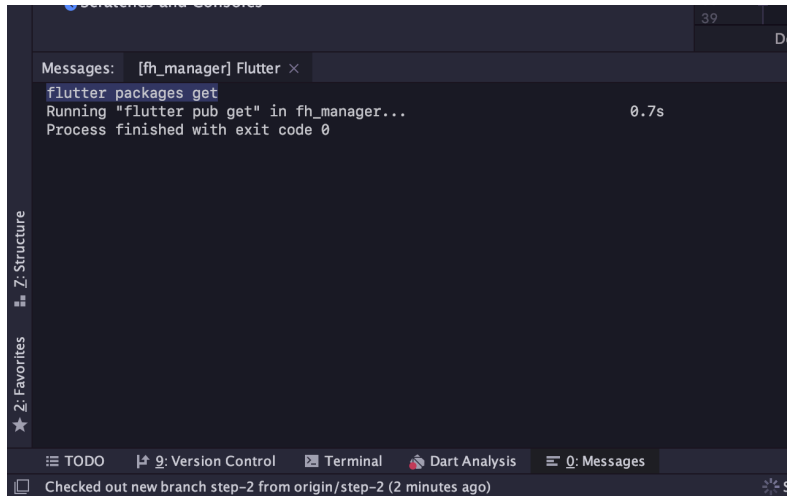
Setup contd.

- Add a Run Configuration pointing to 'main.dart'



Setup contd.

- Open 'pubspec.yaml' and click on "Packages get"



The screenshot shows an IDE interface with a terminal window. The terminal output indicates that the command 'flutter packages get' was executed successfully in the 'fh_manager' directory. The process took 0.7 seconds and finished with an exit code of 0. The IDE's sidebar on the left shows a file explorer with 'Z: Structure' and 'Z: Favorites' sections. The bottom status bar indicates that a new branch 'step-2' was checked out from 'origin/step-2' 2 minutes ago.

```
Messages: [fh_manager] Flutter x
flutter packages get
Running "flutter pub get" in fh_manager... 0.7s
Process finished with exit code 0
```

- Done!

Look Around

- Model -> Data Model
- Db -> Database access
- Page -> Displaying information
- Main.dart -> Entry point

Everything's a Widget

- In Flutter, even components like Padding are modelled as Widgets
 - Reusable components which can be instantiated indefinitely

EXPANDING THE APP!

Expanding the Task-Model

- Would be nice to have more information on Tasks!
 - Description
 - Date
- Add them to TaskModel (both as final String)
 - DateTime cannot be easily persisted, so we resort to using String
- Also add them to Task constructor & 'copy'

Expanding the Task-Model contd.

```
@JsonSerializable()
class Task {
    final String id, parent;
    final String name;
    final String description;
    final String dueDate;
    @JsonKey(name: 'completed')
    final int isCompleted;

    Task(this.name,
        {@required this.parent,
         this.isCompleted = 0,
         String id,
         this.dueDate,
         this.description})
        : this.id = id ?? Uuid().generateV4();

    Task copy(
        {String name,
         int isCompleted,
         int id,
         int parent,
         String description,
         DateTime dueDate}) {
        return Task(
            name ?? this.name,
            isCompleted: isCompleted ?? this.isCompleted,
            id: id ?? this.id,
            parent: parent ?? this.parent,
        );
    }
}
```


Generate Code & Expand DB

- `part 'task_model.g.dart';`
- Open Terminal in AS
 - `> flutter pub run build_runner build`
 - This will automatically generate the .g.dart-file which handles JSON serialization
- Open `db_provider.dart`
 - Modify `CREATE TABLE Task` to include new fields
 - `"description TEXT,"`
`"dueDate TEXT"`

Modify Add Task Page

- Add local variables description and dueDate to class
- Initialize them appropriately in initState()
 - `description = '';`
`dueDate = DateTime.now().toIso8601String();`
- Copy-Paste TextField from name twice (more elegant: Widget)
 - Change onChanged to set correct variable
 - Change hintText
- Include them in call to model.addToDo

Test your app

- With your device connected, run your app
 - iOS: change your developer certificate now
 - In case of errors: flutter clean
- Add a Subject, and a Task
 - Where is Description and dueDate?
- We need to add it to the page!

Displaying description and dueDate

- Open detail_screen.dart
 - Add subtitle to ListTile (around line 214)
 - Displays todo.description
 - Add onTap to ListTile
 - Displays a Snackbar with dueDate

```
subtitle: Text(  
  todo.description ?? "",  
)  
,  
onTap: () {  
  if (todo.dueDate != null) {  
    final snackBar = SnackBar(  
      content: Text(todo.dueDate.toString()),  
      backgroundColor: _color,  
    );  
    Scaffold.of(context).showSnackBar(snackBar);  
  }  
}
```

Test your app (again)

- Add a Subject and a Task with description and Date
- In TaskList you should see description right away
- Tapping on the Task should display the Date

Optional:

- Expand db_provider.dart
 - Default Tasks don't yet include Dates and Descriptions
 - Add that information to them like this:

```
Task(  
  'Course Becker',  
  dueDate: '2019-12-09T13:45',  
  description: 'Vuforia',  
  parent: '1',  
  isCompleted: 1,  
)
```
- Uninstall and reinstall your app to reflect these changes!

Finally: Remove Debug Marker

- In main.dart set debugShowCheckedModeBanner to false
- Now we are production ready!

Thanks for listening!

And Happy Holidays!