Entreprenöriell programmering

TDP028

LINKÖPINGS UNIVERSITET

Ändring sedan förra året

I år kör vi Flutter!

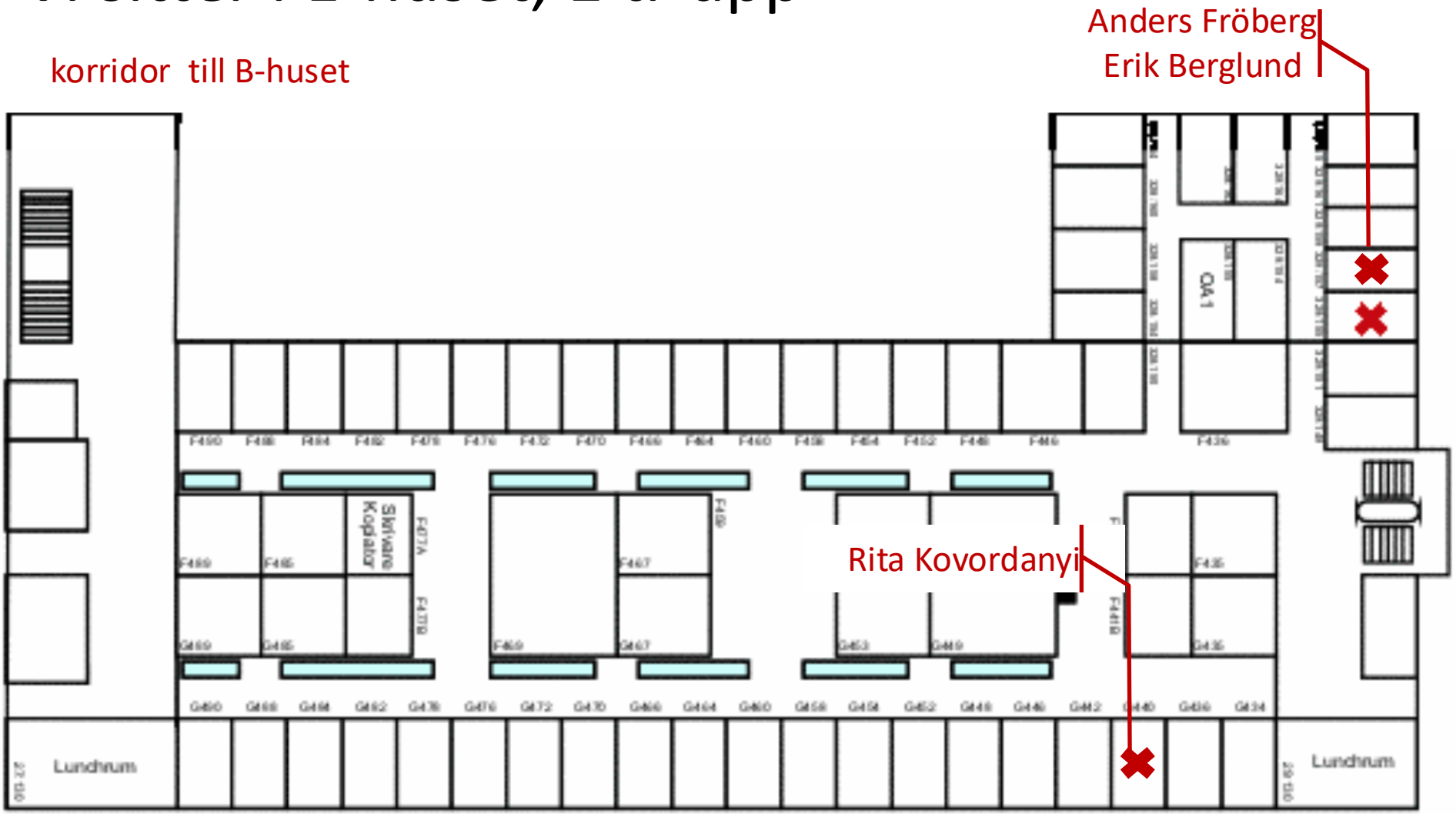
Översikt

- Kursupplägg
  - Projekt
  - Examination
- Köra emulator
  - Eller öppna upp sin telefon
- Flutter introduktion

Kurspersonal

- **Rita Kovordanyi** – kursansvarig, handledare
- **Erik Berglund** – lärare, entreprenöriella aspekter
- **Anders Fröberg** – lärare, Flutter, tekniska frågor
- **Liam Andersson** – handledare
- **Hussnain Haidar** – handledare
- **Lucas Granberg** – handledare
- **Hanan Mohsen** – kursadministratör

Vi sitter i E-huset, 1 tr upp



Handledning

- LIS1 – Grupp 1 – Rita Kovordanyi
- LIS2 – Grupp 2 – Liam Andersson
- LIS3 – Grupp 3 – Hussnain Haidar
- LIS4 – Grupp 4 – Lucas Granberg

Bygga en häftig mobil app

Vi kör Flutter

Quick Comparison Between	Kotlin	Flutter
Language	Kotlin	Dart
Performance	Fast	Fast
Popularity	35.3K stars on GitHub	114K stars on GitHub
Learning Curve	Limited	Comprehensive documentation
UI design	Great Experience & customization	Full Customization

Kursmål

- Eget större projekt
  - Definiera och planera egen produkt
  - Använda ramverk och tekniker relevanta för entreprenöriella projekt
- Bli medveten om hur positionering och marknadsföring påverkar produktutveckling
- Utnyttja vetenskapliga rön i produktutveckling
- Öva på muntlig presentation - screencasts

Projektet

- Genomförs individuellt
  - Stöd i föreläsningar
    - Flutter
    - Konkurrensanalys
    - Litteratursökning
  - Seminarier
  - Handledning

Kurslitteratur

On-line tutorials, best practices, dokumentation, etc.

https://docs.flutter.dev

Bra startpunkt:

https://docs.flutter.dev/get-started/test-drive

Ändringar i kursen för i år

Bytt programmeringsspråk

- **Flutter**
  - Ramverk för att utveckla för web, iOS och Android
    - Högnivåkod kompileras till kod för olika plattformar
  - Lätt att bygga användargränssnitt (UI)
- Vi använder **Firestore** som backend
  - Lagring av data

Anpassade tekniska krav

- Tekniska kraven på projektet har anpassats till Flutter
  - Best practices
  - Bibliotek
  - Strukturering av koden
    - Dela upp kod i skärmar, filer, mappar
    - Dela upp enligt något design-mönster

Slopat kravet på commits

- Vi fokuserar istället på att appen byggs upp enligt en lean metodik
  - Stegvis, med gradvis ökad app-komplexitet och fler funktioner

Obligatoriska milstolpar

1. Appbeskrivning + konkurrensanalys
2. Skärm med text och knapp man kan trycka på
3. Gå mellan två skärmar
4. Hämta data över nätet (från Firestore)
5. Tekniskt PM + veckoplanering framåt
6. Halvtids-screencast (2 min demo)

Appen ska byggas upp i steg

1. Appbeskr. + konk.analys
2. Enklare UI-komponenter
3. Navigera mellan skärmar
4. Visa innehåll från databas
5. Tekniskt PM + tidsplan
   - Vilken funktionalitet ska implementeras vilken vecka
6. Halvtids-screencast

1. Applan.seminarium
2. Fö: Intro Flutter
3. Fö: Navigering
4. Fö: Backend
5. Fö: Litteratursökning

En milstolpe per vecka

- Milstolparna måste lämnas in i tid
  - Kan missa max 2 milstolpe-deadlines
  - Missad milstolpe måste dock lämnas in före slutdemo
- Halvtids-screencast måste lämnas in i tid
  - Ska visa förenklad prototyp av appen (ca 2 min inspelning av hur appen används)

Betyg – samla projektpoäng

Samla projektpoäng

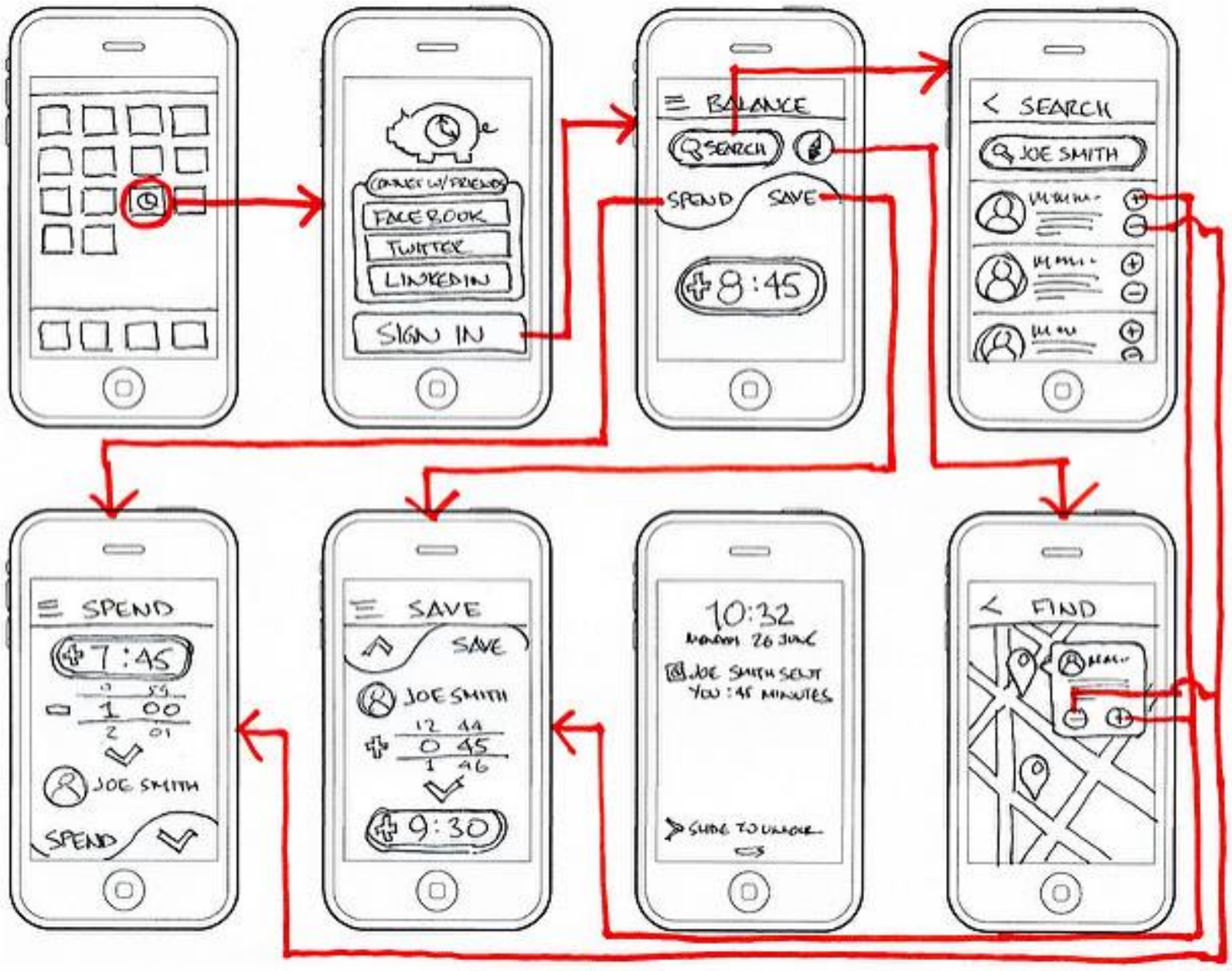
- Ni bestämmer själva vilken betygsnivå ni vill jobba mot
  - Samla projektpoäng genom att implementera viss funktionalitet i appen, tillämpa snygga lösningar, använda API:er
- Fler projektpoäng ger högre betyg
  - Se vidare kurshemsidan (Projektkrav och examination)

Deluppgifter i projektet

Saker som måste vara avklarade för betyg G

App-beskrivning

- Kort beskrivning av appen
  - Användningsområde
  - Tänkt användargrupp
  - Syfte: Vad appen är tänkt att göra
- Wireframe som visar skärmarna + övergångar
  - Olika sätt att ta sig runt i appen
  - Kort beskrivning / etiketter på actions och skärmövergångar vid tap, click, knapptryck, etc.


FACEBOOK
TWITTER
LINKEDIN
SIGN IN
BALANCE
SEARCH
SPEND
SAVE
8:45
SEARCH
JOE SMITH
SPEND
7:45
JOE SMITH
SPEND
SAVE
SAVE
JOE SMITH
9:30
10:32
JOE SMITH SELECT
YOU : 45 MINUTES
FIND

Konkurrensanalys-övning

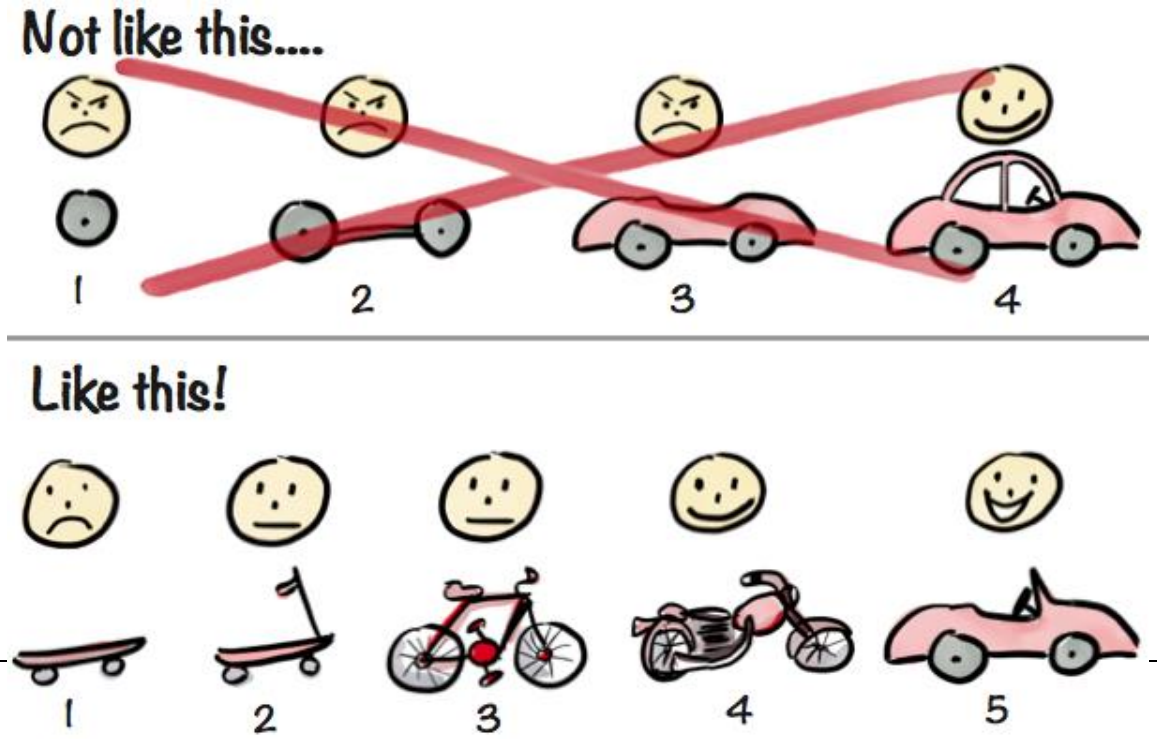
- Välj existerande app
- Analysera
  - Vad **måste** en *ny* app ha för features, för att kunna konkurrera?
  - Vad **kan** en *ny* app ha för att ”sticka ut” / profilera sig mot den existerande appen?
- En bra ny app skulle då ha minsta + det där extra
  - Underlag för detaljerad planering av egna appen

Lean utveckling

- Build-measure-learn feedback loop
  - Snabbt få ut en minimal app för att se kundernas reaktion → förbättra
- Minimum viable product (MVP)
  - T.ex. Facebook startade som studentkatalog
- Läs vidare:
  - https://myva360.com/blog/examples-of-minimum-viable-products
  - http://theleanstartup.com/principles

’Lean’ utveckling

- Stegvis utveckling där varje steg ger en användbar produkt



Vecka	Milstolpe	Obligatoriska milstolpar
v36	Milstolpe-1	Appbeskrivning (tänkt funktionalitet och wireframe-navigationsdiagram), samt konkurrensanalys inlämnade i repo. Commit:en taggad med Milstolpe-1. Lägg en kortversion av app-beskrivningen i readme.md i repot. (Readme är en slags introduktion till repot, det som besökaren ser allra först.) **Slides som visar design-stegen.**
v37	Milstolpe-2	En enkel början på din egen app, t.ex. med någon text, och en fungerande knapp (något ska hända på skärmen när man trycker på knappen), inlämnad i repo. Commit:en taggad med Milstolpe-2. **Instruktioner för hur man sätter upp sin miljö** **Ett sätt att komma igång med skärmar är att göra denna Codelab.**
v38	Milstolpe-3	Två skärmar i din app, med navigering emellan, inlämnad i repo. Commit:en taggad med Milstolpe-3. Öva på klick-hantering mellan två skärmar, och hur man skickar med information när man navigerar till nästa skärm. Gärna även hur man skickar information bakåt, när man är klar med en skärm. **Läs på om olika sätt att navigera** **Enkelt sätt att navigera mellan två skärmar**
v39	Milstolpe-4	Visa faktiskt innehåll på skärmarna, dvs. inkoppling av backend (t.ex. Firestore). Kod inlämnad i repo. Commit:en taggad med Milstolpe-4.
v40	Milstolpe-5	Tekniskt PM, samt veckoplanering av resterande projektarbetet, med kravlista (funktionella krav) på vad som kommer att implementeras, och vad som ska uppnås varje vecka, är inlämnade i repo, t.ex. i en docs-mapp, eller liknande mapp, så att de lätt kan hittas av rättande lärare. Commit:en taggad med Milstolpe-5
v42	Milstolpe-6	**Halvtids-screencast** på ca. 2 minuter, där du demar en tidig version av din app (ungefär motsvarande en Minimal Viable Product), inlämnat i repo. Commit:en taggad med Milstolpe-6

- Krav:
  - Alla milstolpar avklarade
    - 3 milstolpar inlämnade i tid
  - Halvtids-screencast inlämnad i tid

Klara av milstolpar i tid

- **Fredag denna vecka**
  - Appbeskrivning och konkurrensanalys inlämnade i repo. Tagga med **Milstolpe-1**

    > git add .

    > git commit –m ”….” → *ger hash a029ac*

    > git push origin main

    > git tag –a Milstolpe-1 *a029ac* –m “Klar med …”

    > git push origin main --tag

Blir datumstämplat

Freemium-seminarium

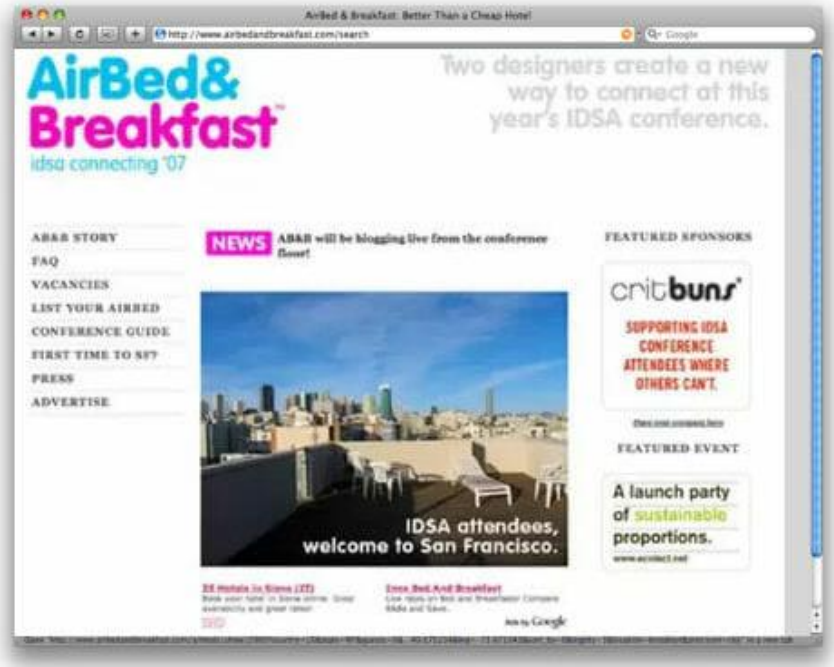
- Läs och diskutera artiklar:
  - Monetizing an infinite runner
  - Why do players buy in-game content? An empirical study on concrete purchase motivations
  - A Study of Crucial Factors for In-App Purchase of Game Software

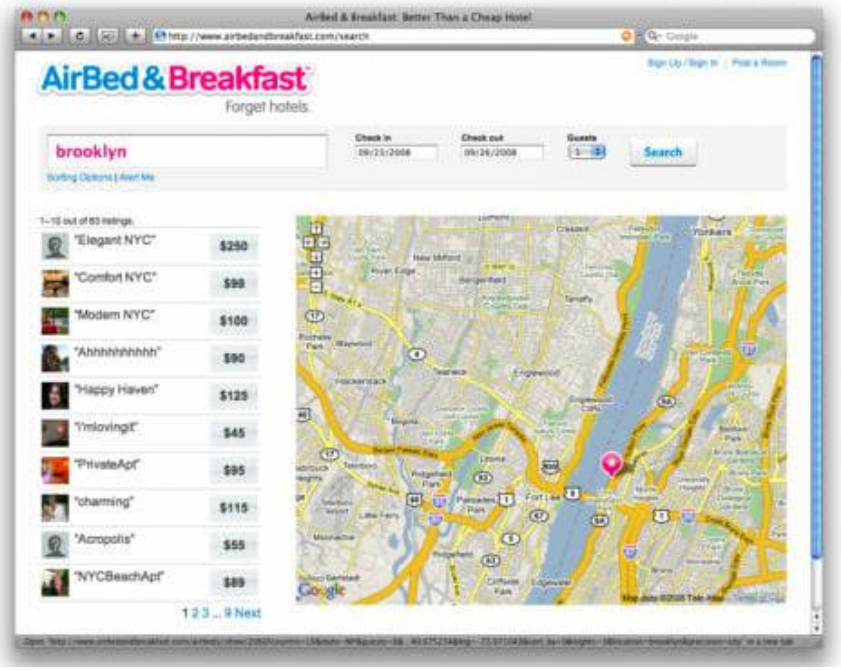
Tekniskt PM

- Litteratursökning och referenshantering
  - T.ex. hur man kan använda analytics i lean utveckling
  - T.ex. hur modularisera kod i Android
    - Model View ViewModel (MVVM)?
  - Andra aspekter av kodkvalitet
    - Underhållbarhet?

Halvtids-screencast (2 min)

- Dema en minimal viable product (MVP) av appen
  - MVP: Mest grundläggande funktionerna





Appen i halvtids-screencast

- Ska uppfylla de krav som ställs i milstolparna:
  - Ska följa app-beskrivningen
  - Ha flera skärmar och navigering
  - Visa innehåll från Firestore
  - Ha funktionalitet som ligger i linje med app-beskrivningen

Slutdemo

- Veckan före jul
  - Räkna med ca. 15-20 min + ”sätta upp appen”
- Visa upp hur appen används
- Vilka funktioner
  - Fokus på de funktioner ni vill ha projektpoäng för
- Vara beredd att visa koden och svara på frågor om koden

Slutscreencast

- Ca 5 min
- Dema den app-funktionalitet ni vill ha projektpoäng för
  - För exempel, se kurshemsidan

Inlämnade koden

- Lyft fram med kodkommentarer de tekniska lösningar du vill få projektpoäng för
  - T.ex. de betygsgrundande features, och API:er du har implementerat

Grundkrav

För betyg G

- Halvtidssreencast i tid
  - Minst tre andra milstolpar i tid
- Grundläggande app-komplexitet
  - Minst 4 skärmar (förutom inlogg) med navigering
- Antal egenvalda tekniska och entreprenöriella krav implementerade i appen
- Allt lämnas in via Gitlab-repo
  - Dokument, ritade skisser och annat kan läggas i en docs-mapp i repot

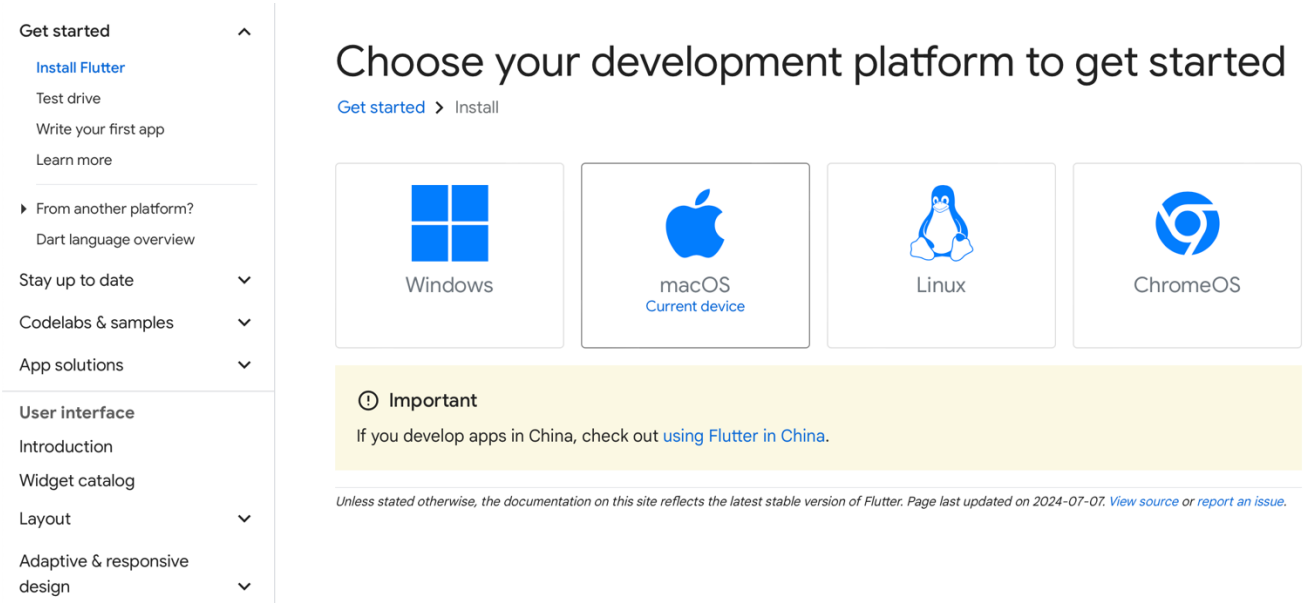
I README i repo

- Ska visas som ”front page” i Gitlab
- En kortversion av app-beskrivningen
- Ange betygsnivån du har siktat på
- Lista sedan:
  - Poänggivande API:er i appen, med kort beskrivning / motivering av varje
  - Poänggivande tekniska features, med kort beskrivning /motivering

Praktiska detaljer

Installera miljön: Visual Studio Code

- Guide hur man installerar Flutter i VS Code: https://docs.flutter.dev/get-started/install



Utveckla helst i ditt OS

- Installera gärna allt som krävs för att kunna köra native i ditt OS (på datorn)
  - Ger snabbare utvecklingsloop
  - Kan köra ’hot reload’ (ingen omkompilering)
    - Snabbar upp varven av kodändring och körning
- Kan i senare skede övergå i att köra på fysisk mobil
  - För att testa sensorer, som GPS, kamera, etc.

Efter installation

> flutter doctor

Listar potentiella problem med utvecklingsmiljö, samt råd hur man kan fixa dem

Sista koll efter installation: kör flutter doctor

```
07_14.16:42:58 mac00408:namer$ flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.22.2, on macOS 14.5 23F79 darwi
[!] Android toolchain - develop for Android devices (Android SD
    ✗ cmdline-tools component is missing
      Run `path/to/sdkmanager --install "cmdline-tools;latest"`
      See https://developer.android.com/studio/command-line fo
    ✗ Android license status unknown.
      Run `flutter doctor --android-licenses` to accept the SDK
      See https://flutter.dev/docs/get-started/install/macos#an
[!] Xcode - develop for iOS and macOS (Xcode 15.4)
    ✗ Unable to get list of installed Simulator runtimes.
[✓] Chrome - develop for the web
[✓] Android Studio (version 2024.1)
[✓] VS Code (version 1.91.1)
[✓] Connected device (2 available)
[✓] Network resources
```

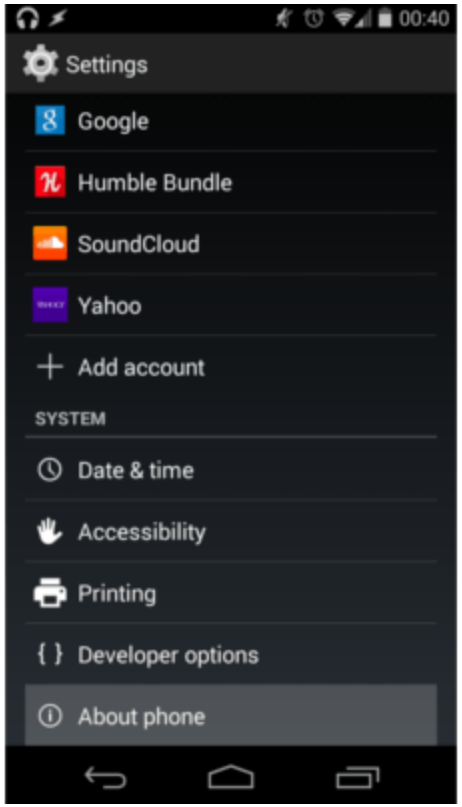
Googla hur man fixar specifika problem

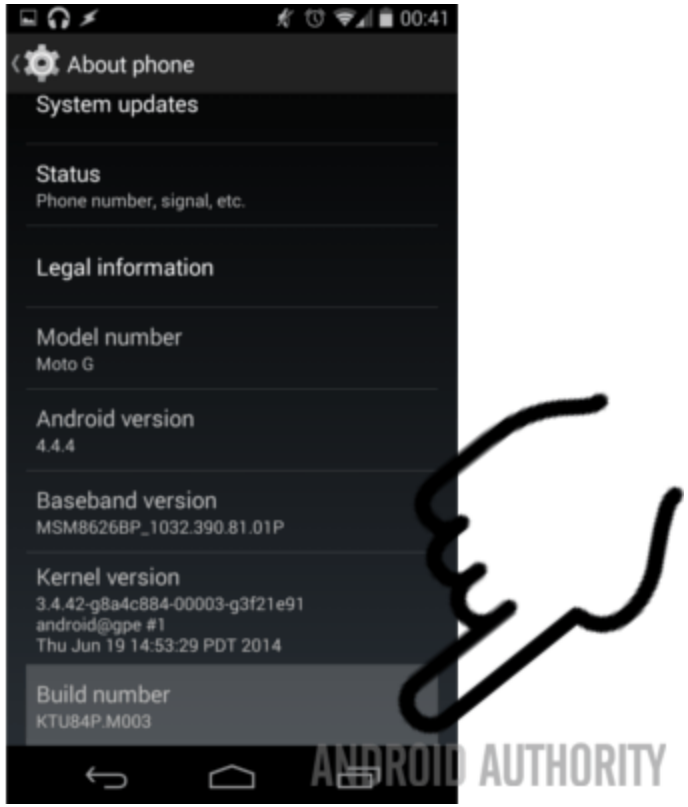
```
08_10.16:07:27 mac00408:routing_ex$ flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.24.0, on macOS 14.5 23F79 da
[✓] Android toolchain - develop for Android devices (Androi
[✓] Xcode - develop for iOS and macOS (Xcode 15.4)
[✓] Chrome - develop for the web
[✓] Android Studio (version 2024.1)
[✓] VS Code (version 1.91.1)
[✓] Connected device (3 available)
[✓] Network resources

• No issues found!
```

Hur köra appen?

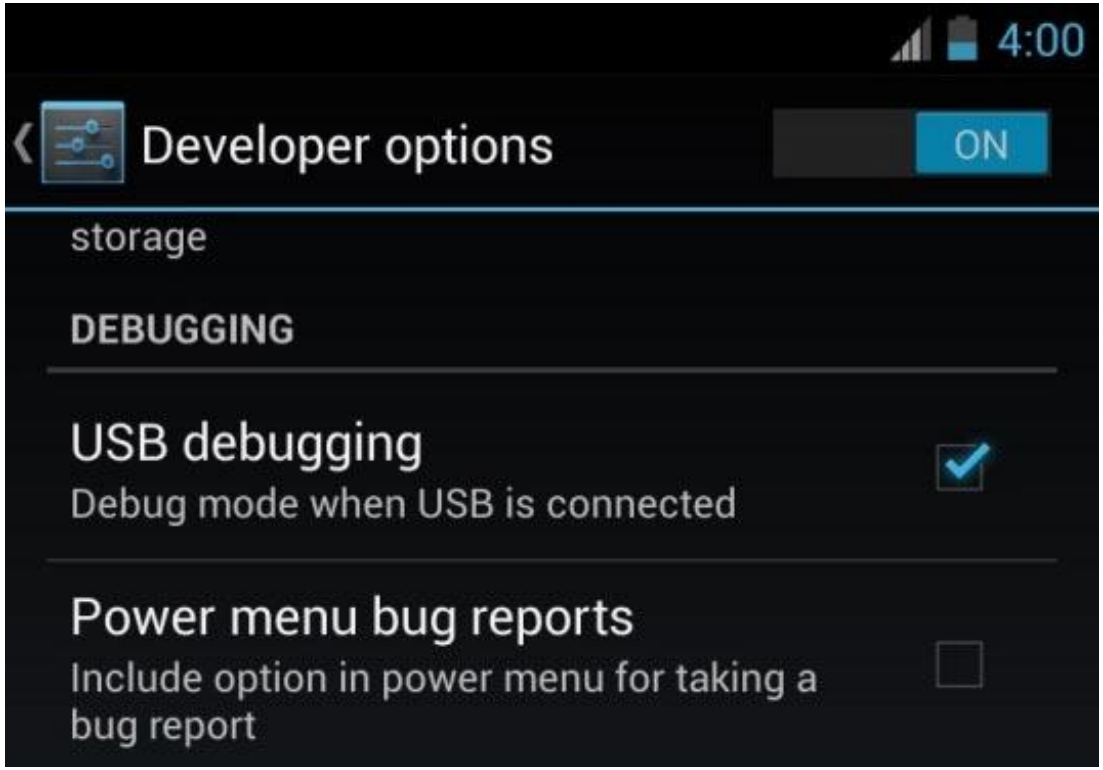
- Under utveckling
  - Kör gärna på datorn (OS:et som datorn kör)
    - Får ‘hot reload’ av appen
  - Kör web-baserat
- För demo
  - Kör på egen telefon (Android, iOS)
  - Emulator
    - Viss funktionalitet blir begränsad (t.ex. GPS)

Egen Android mobil eller platta

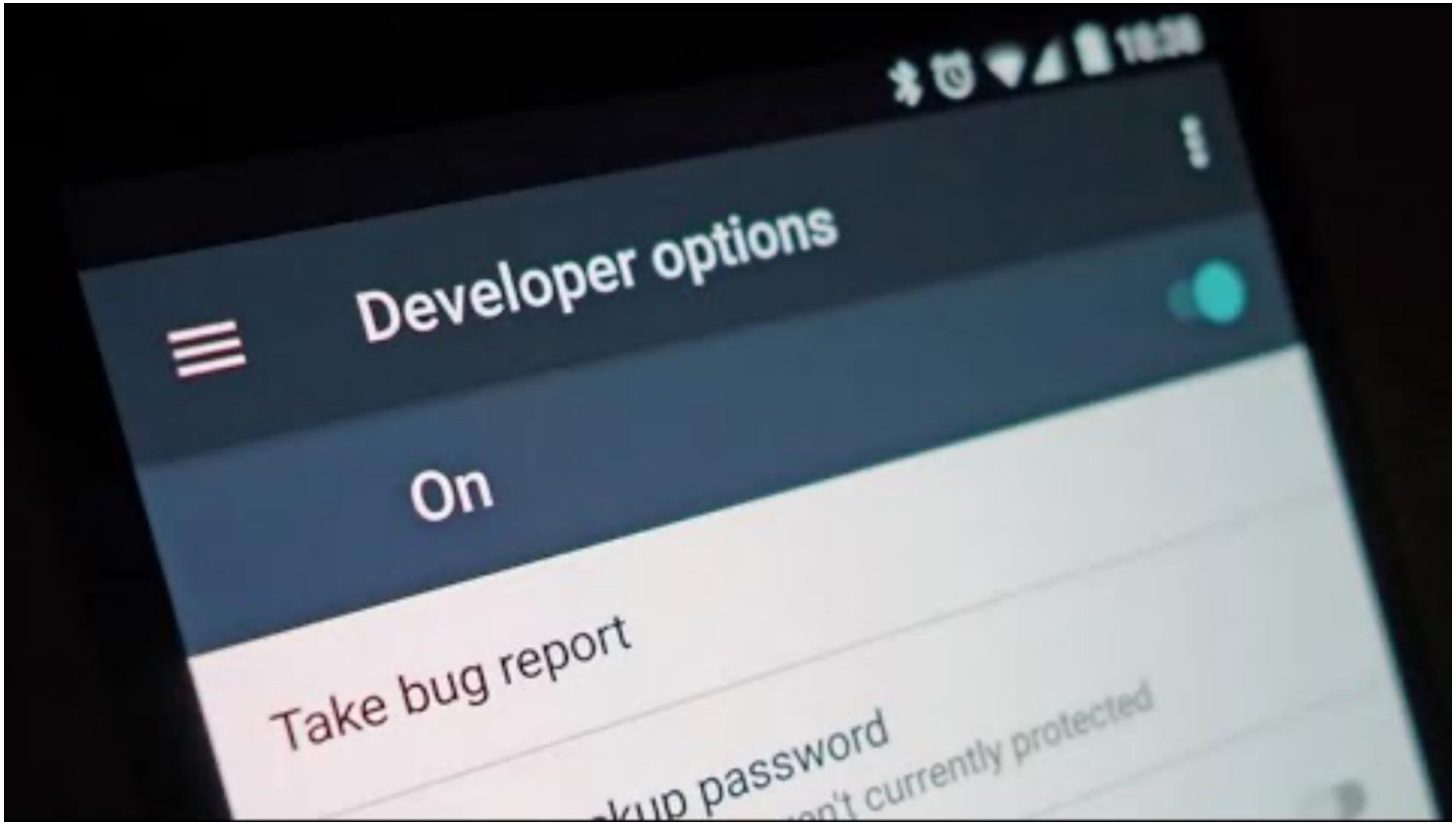




Felsökning via USB-koppling



Developer mode på Android



https://www.youtube.com/watch?v=7K6AWPCAkiU

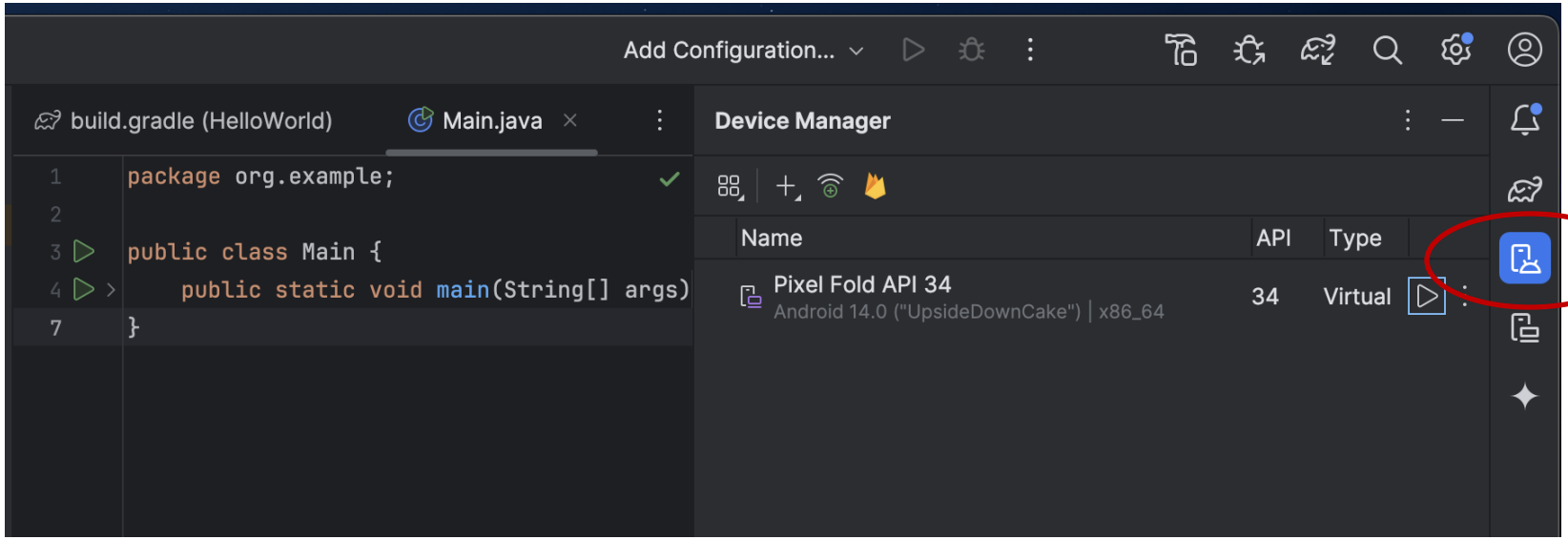
Developer mode på iPhone

https://youtu.be/89Dv65twGaY?si=VE_8RUFj3Oef6sq0

Emulatorn

kontroller

Emulatorer hanteras i Android Studio ’ (även om man utvecklar i VSCode)

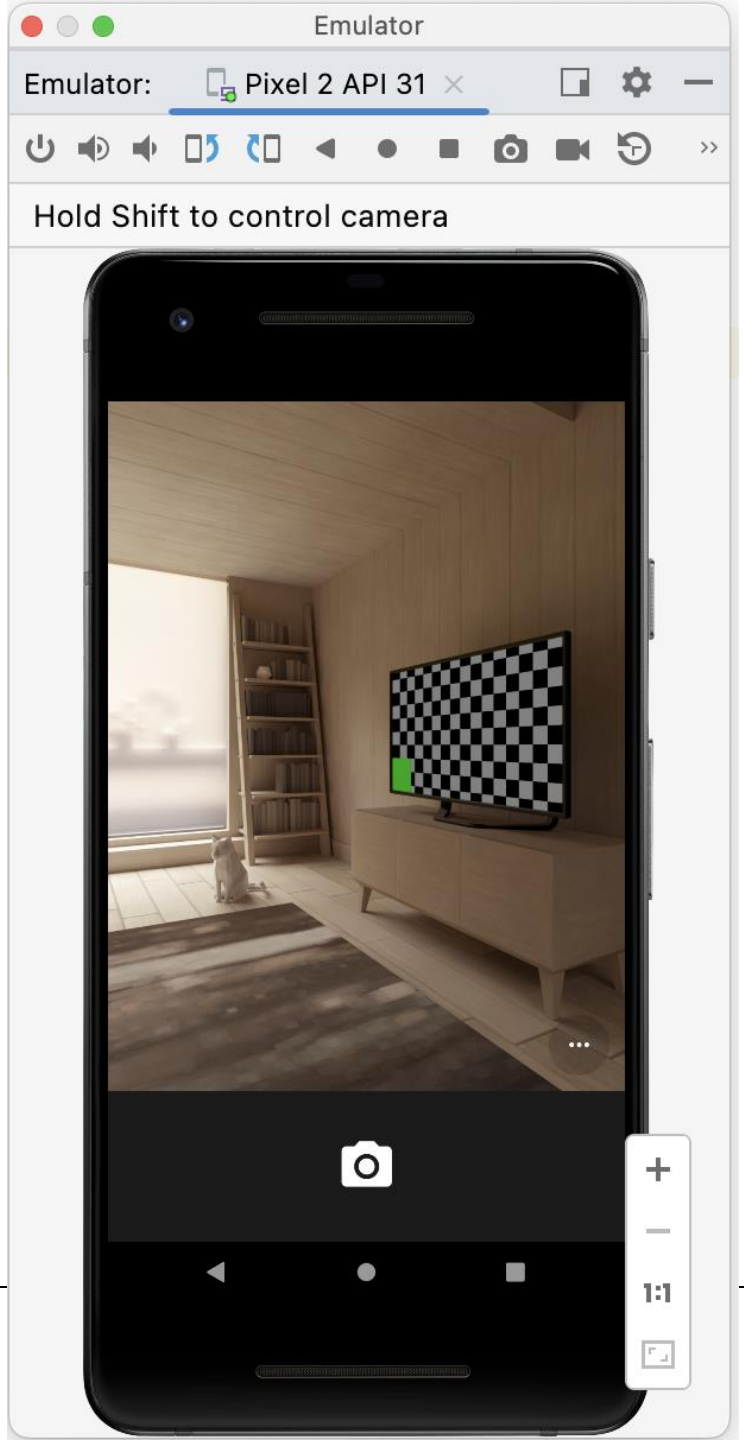


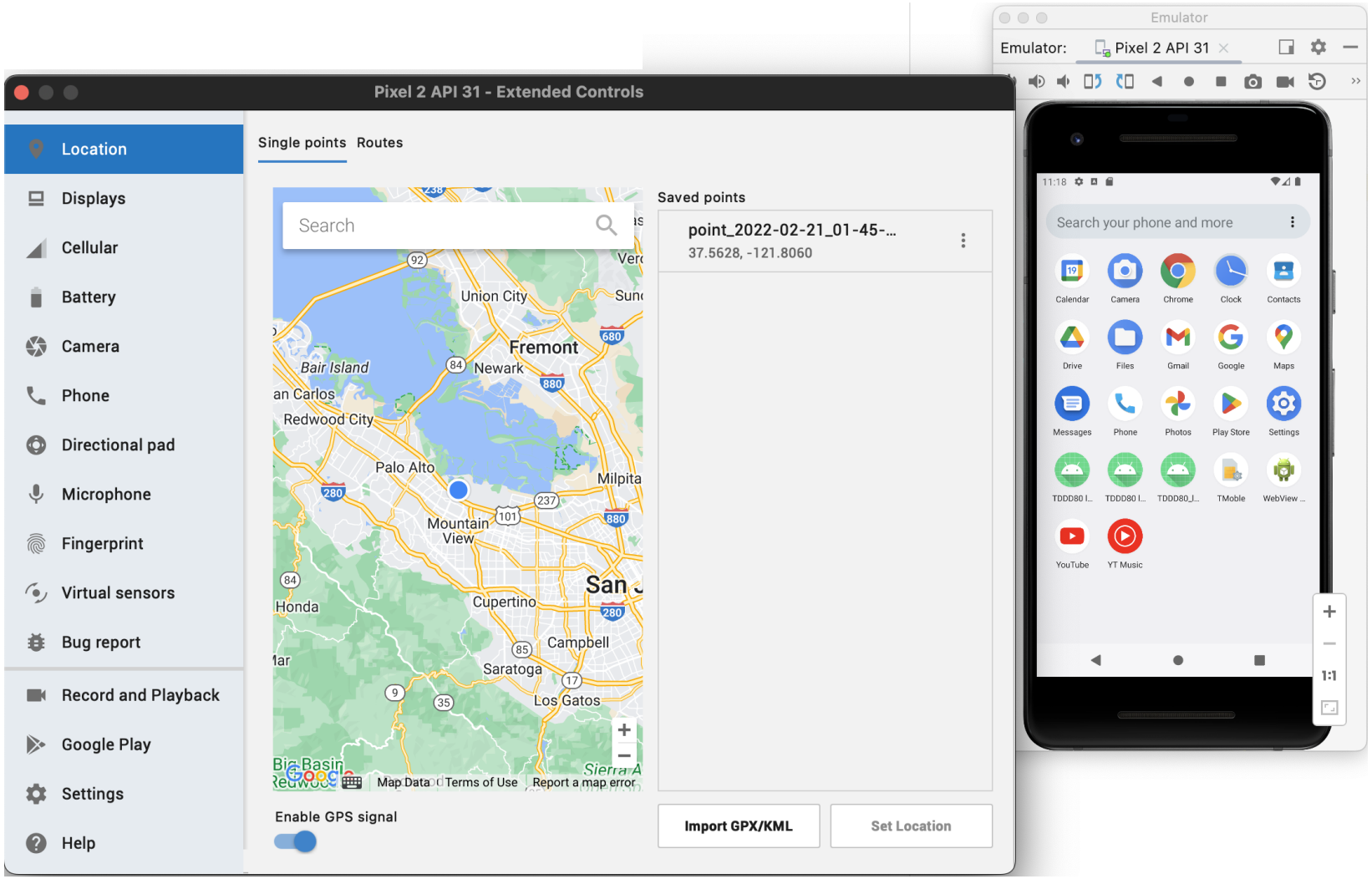
Internet

- Emulatorn delar datorns internetuppkoppling
- Kolla genom att t.ex. öppna emulatorns webläsare, och se om kontakt med internet
  - Annars: Inställningar på datorn för att dela internet

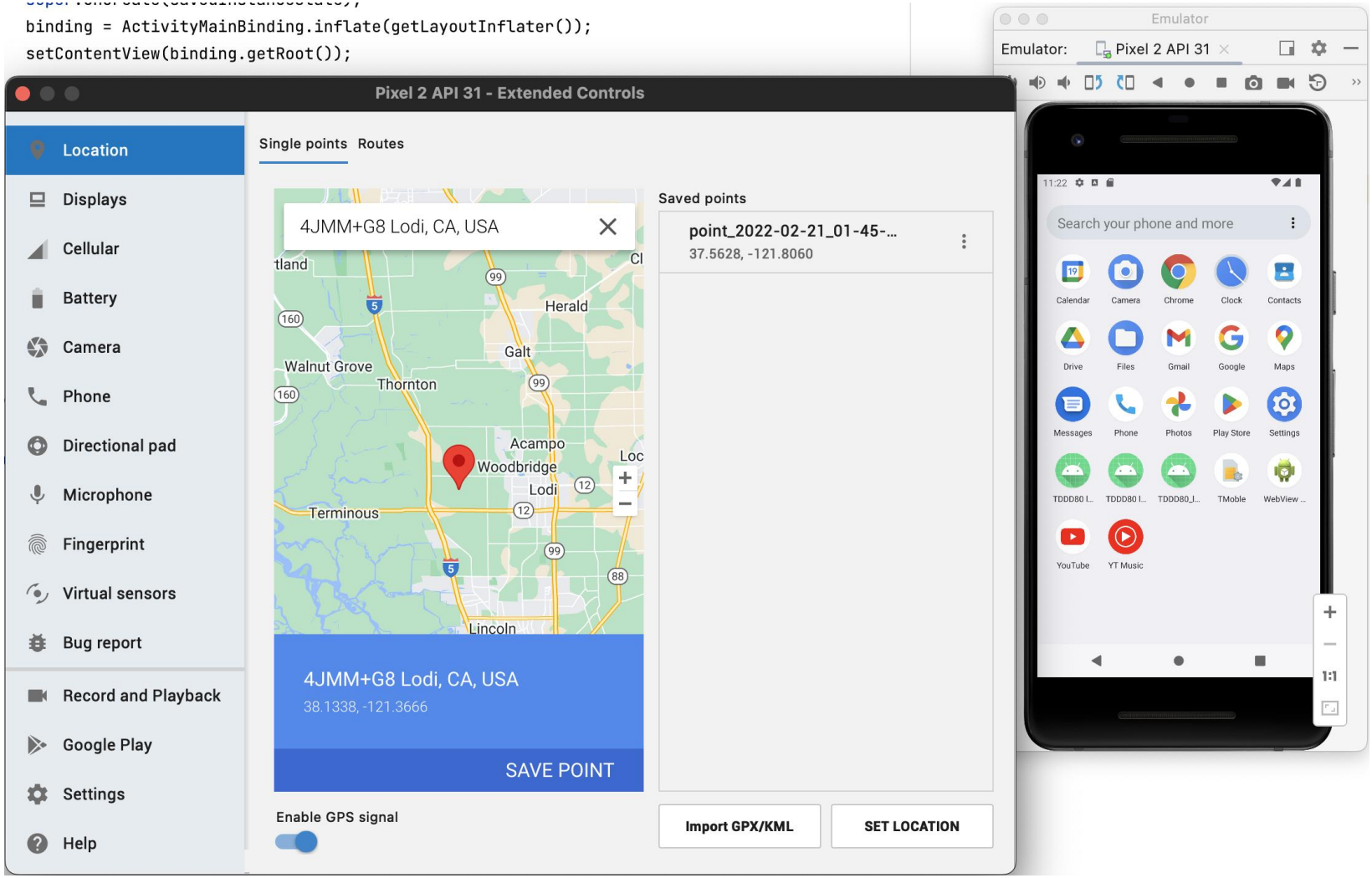
Virtuell kamera

Nås genom att klicka på appen Kamera




Pixel 2 API 31 - Extended Controls
Location
Displays
Cellular
Battery
Camera
Phone
Directional pad
Microphone
Fingerprint
Virtual sensors
Bug report
Record and Playback
Google Play
Settings
Help
Single points
Routes
Search
Saved points
point_2022-02-21_01-45-...
37.5628, -121.8060
Map Data
Terms of Use
Report a map error
Enable GPS signal
Import GPX/KML
Set Location
Emulator
Emulator:
Pixel 2 API 31
Search your phone and more

```
binding = ActivityMainBinding.inflate(getLayoutInflater());
setContentView(binding.getRoot());
```



Flutter

Bygger på språket Dart

Bra ställen att botanisera

- Flutter
  - https://docs.flutter.dev/codelabs
- Dart
  - Dokumentation: https://dart.dev/language
  - Codelab:
    - https://dart.dev/codelabs/dart-cheatsheet
  - Testa kodsnuttar i DartPad: https://dartpad.dev

Flutter är GUI-orienterat

Python

```
means = []
stds = []
for i in range(len(scores)):
  means.append(scores[i][1]
  stds.append(scores[i][0])
```

Imperativ: hur det ska göras

Flutter

```
Widget build(BuildContext context) {
  return Scaffold(
    appBar: CustomAppBar(),
    body: ...
  );
}
```

Deklarativ: hur det ska vara

Bygg GUI:t

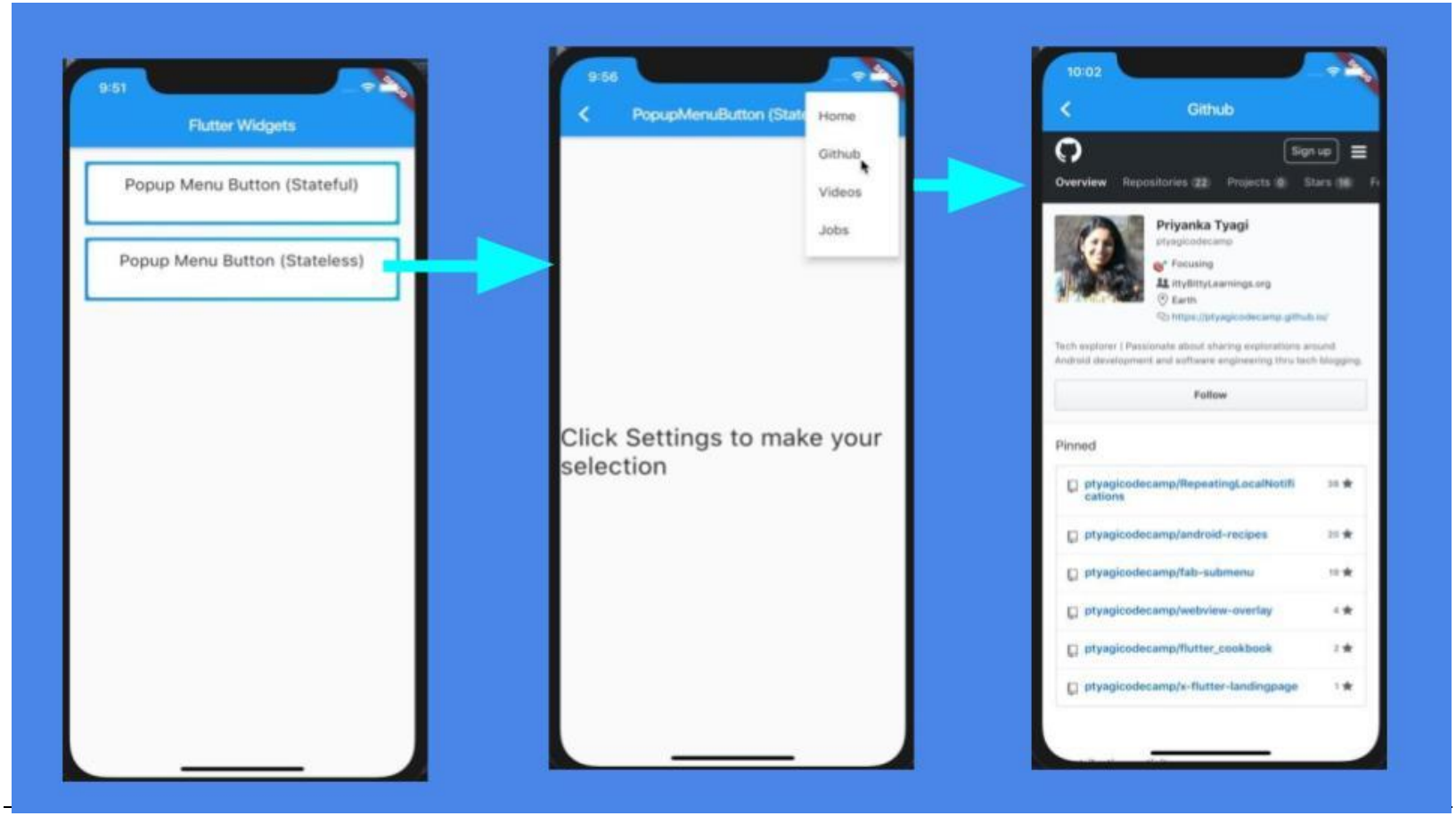
Skapa instans

Instansen ska skapas med de här argumenten

Flutter jobbar med Widgets

- Måste finnas build-metod för varje Widget
- Argument till build beskriver hur UI:t ska se ut
  - Vilka komponent-widgets
  - Var komponenterna ska vara placerade
  - Färger, teman
  - Etc.

Widgets (UI komponenter)



Widgets (UI komponenter)

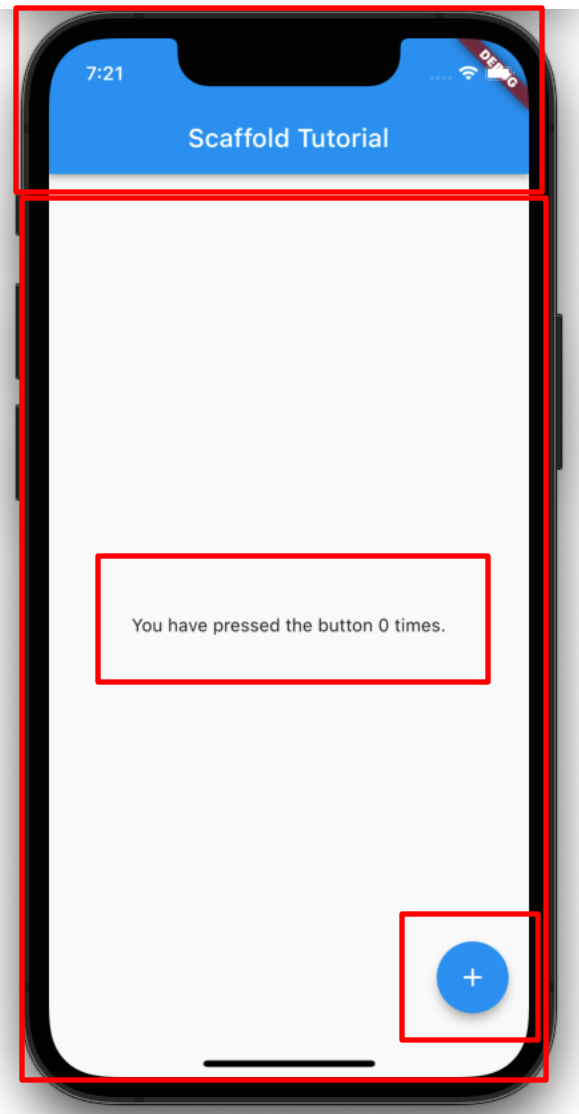
```
class HomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: CustomAppBar(),
      body: ... // Your content here
    );
  }
}
```

AppBar


```
AppBar(
 leading: Icon(Icons.menu),
 title: Text('Page title'),
 actions: [
  Icon(Icons.favorite),
  Padding(
   padding: EdgeInsets.symmetric(horizontal: 16),
   child: Icon(Icons.search),
  ),
  Icon(Icons.more_vert),
 ],
 backgroundColor: Colors.purple,
```

Scaffold

- Top-nivå widget
- Scaffold
  - appBar
  - *body*
    - Center
      - Kolumn
        - » Text
  - floatingActionButton



Widgets kan nästlas

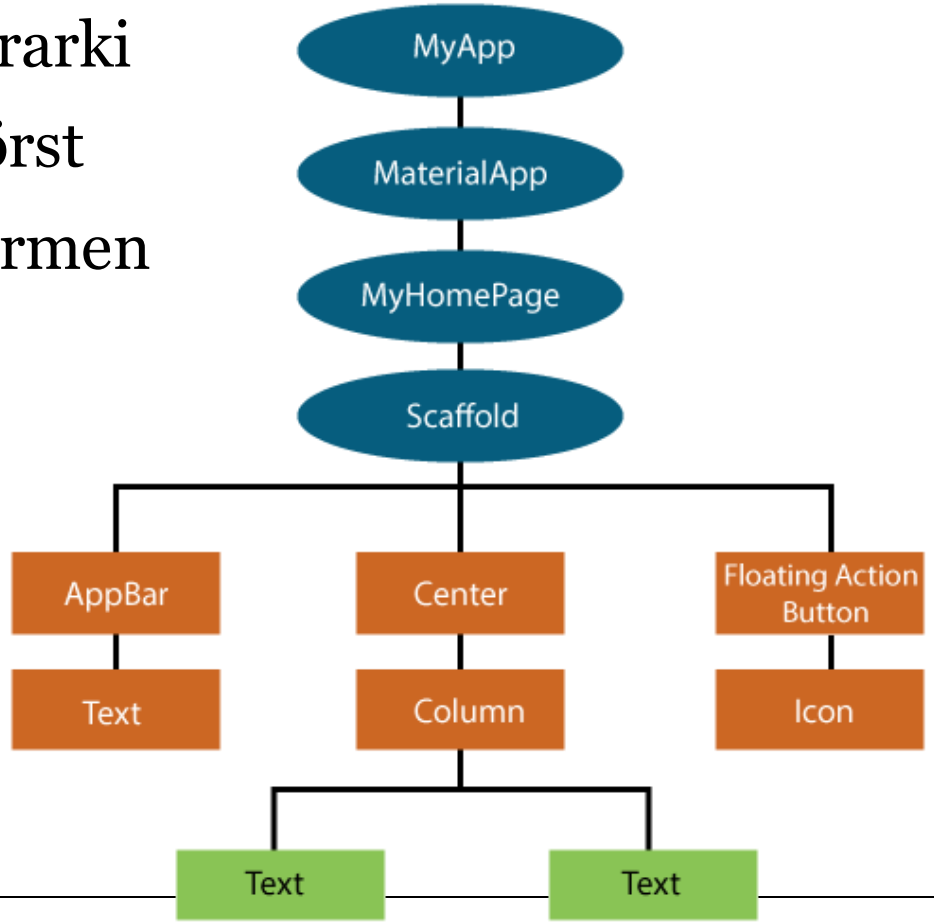
```
Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    backgroundColor: Theme.of(context).colorScheme.inversePrimary,
    title: const Text('Second page'),
   ),
   body: const Center(
    child: Column(
     mainAxisAlignment: MainAxisAlignment.center,
     children: <Widget>[
      Text(
       'This is your next screen',
```

Widget build

- build kan returnera vilken Widget som helst:
  - Scaffold
  - Column
  - Text
  - Egendefnierad klass
  - Etc.
- Man kan/bör alltså dela upp en mastig build-metod i flera del-metoder och/eller hjälp-klasser

Widget hierarki

- Bygger widget hierarki
- Ritas från roten först
  - Underst på skärmen
- Löven ritas sist
  - Ligger överst på skärmen



```
void main() {
  runApp(MyApp());
}

class MyApp ... {

  Widget build(BuildContext context) {
      return MaterialApp(
        title: 'My Owesome App',
        theme: ...,
        home: MainPage(),
  }

class MyAppState ... {
  var user = ...;
}

class MainPage extends StatelessWidget {

  Widget build(BuildContext context) {
     var appState = ...;

     return Scaffold(
       body: Column(
         children: [
           const Text('Välkommen $appState.user'),
```

Kör igång appen

Sätter upp appen

Globalt app-tillstånd

Definierar första skärmen

Flutter bygger på språket Dart

$

```
return Scaffold(
    body: Column(
        children: [
            const Text('Välkommen $appState.user'),
        ],
    ),
);
```

String interpolation

- ${3 +2} -> ‘5’
- ${‘word’.toUpperCase()} -> ‘WORD’
- $myObject -> 'The value of myObject.toString()

Const

```
return Scaffold(
    body: Column(
        children: [
            const Text(’Välkommen $appState.user’),
```

Static (statiska variabler)

- Tillhör klassen
  - Delas mellan alla instanser av klassen
- Ex.

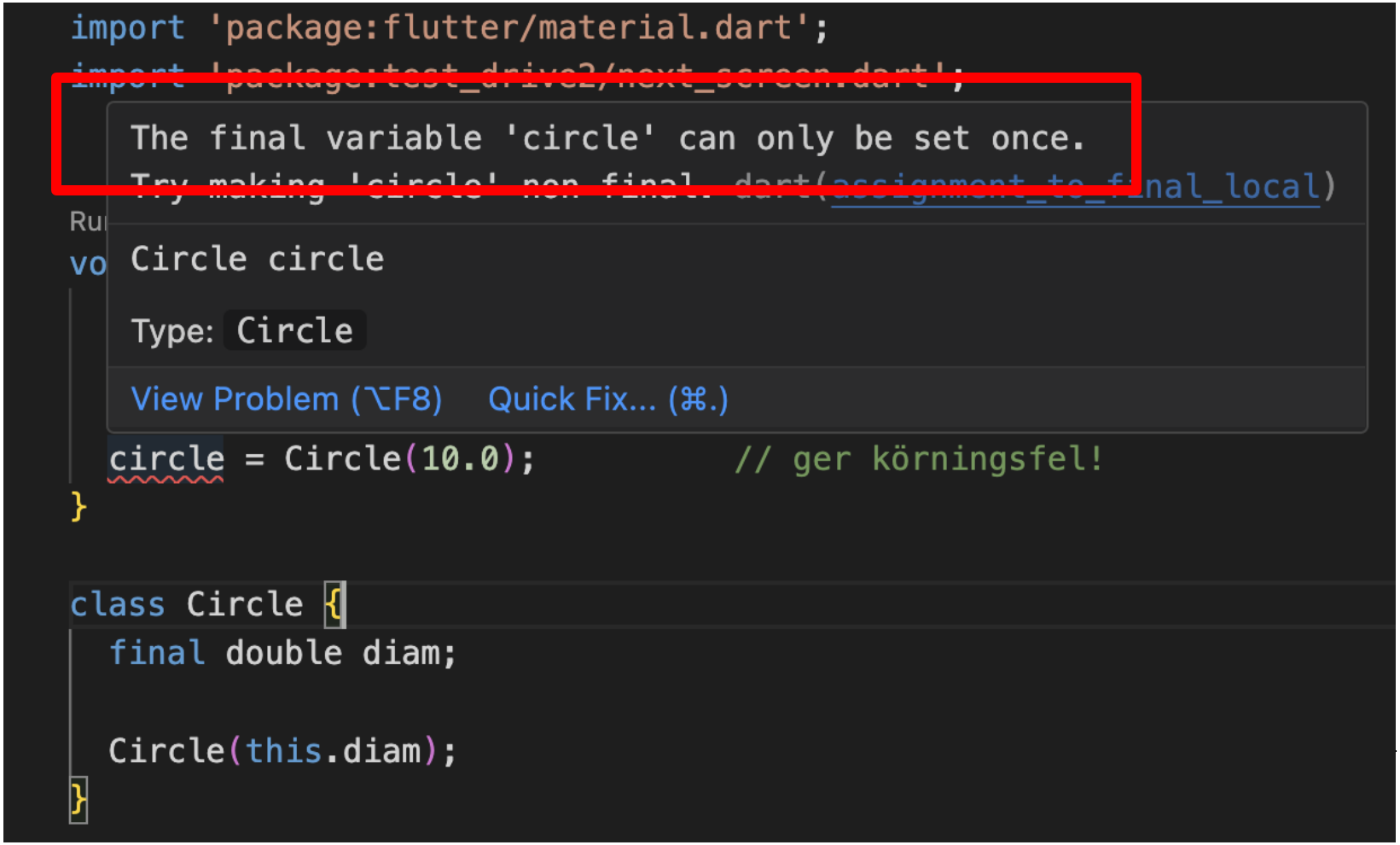
```
class SomeClass {
    static int staticField = 4;
}
```

Final

```
void main() {
  final Circle circle = Circle(5.0);  // tilldelas vid körning
  print(circle.calculateArea());
  circle = Circle(10.0);              // ger körningsfel
```

Upptäcks även av kompilatorn…

Final


import 'package:flutter/material.dart';
The final variable 'circle' can only be set once.
Circle circle
Type: Circle
View Problem (⌥F8) Quick Fix... (⌘.)
circle = Circle(10.0); // ger körningsfel!
}
class Circle {
final double diam;
Circle(this.diam);
}

Const

```
void main() {
  const double pi = 3.14;   // sätts här och nu, i koden

  print(pi);

  pi = 3.14159;             // kompilatorfel!
}
```

Variabler i Dart

- Typinferens
  - Behöver inte ange typ…
    - … men korrekt typning kan förtydliga koden

```
var name = 'Voyager I’;
var antennaDiameter = 3.7;
var flybyObjects = ['Jupiter', 'Saturn', 'Uranus'];
var image = { 'tags': ['saturn’],
              'url': '//path/to/saturn.jpg' };
```

Variabler (forts)

- Vill inte begränsa typen

```
Object name = 'Bob';
dynamic name = 'Bob';
```

Typ kan förändras under körning

- Vill ange typen

```
String name = 'Bob';
```

Får hjälp av kompilatorn att upptäcka fel innan körning

rita.kovordanyi@liu.se

www.liu.se

LINKÖPINGS UNIVERSITET