

Practical engineering II

Webapps

CS-214 - 11 Nov 2024
Clément Pit-Claudel



Quick announcements

find-lazy callback

cancelled to give you time
(good luck with CompArch!)

Debrief for week 8

is up, with laziness tips

Signal-processing lecture

is up as a literate program

Final exam arrangements

are still being confirmed

Unguided lab teams

are due this Friday

Unguided lab instructions

are up!

Planning ahead for the webapp lab

Topic: Multi-user webapps

Teams: 3-4 people (**not 2!**)

Steps:

- Register a team (Week 9)
- Write a proposal (Week 10)
- Implement (Week 10-13)
- Submit code (Week 13)
- Get a checkoff (Week 14)

Training: webapp-rps

This week:

Webapps

Learning objectives:

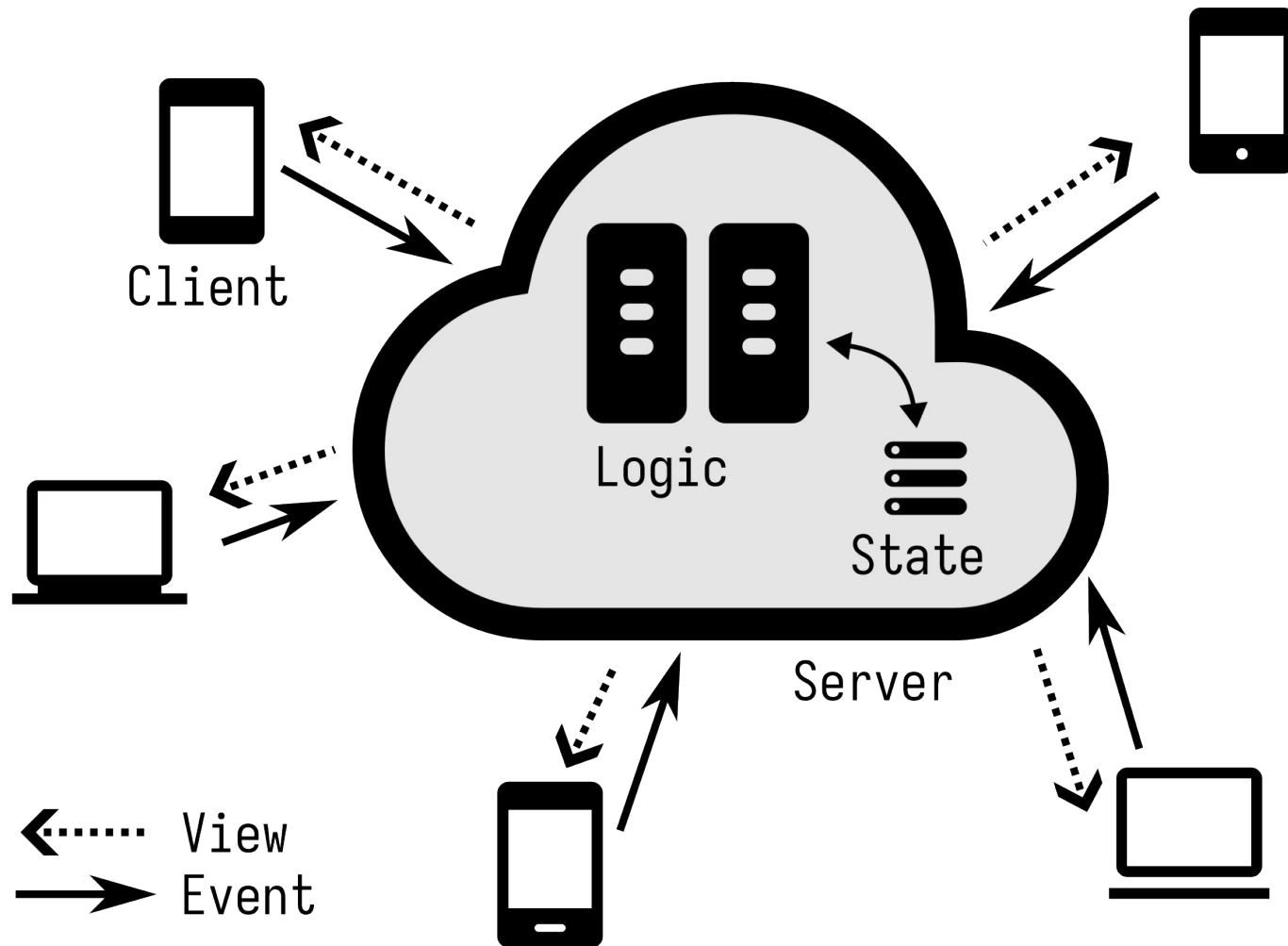
Decompose multi-user applications into states, views, and events.

Implement interactive UIs.

Test complex sequences of events.

- Structure of a webapp
- Library intro and demo
- User stories
- Requirements
- Logic
- UI
- Tests

Structure of a webapp



Demo

- TicTacToe
- Memory / Concentration

Part I: User stories

Part II: Requirements

- **States, actions, views**
- **Client and Server interfaces**

Part III: Logic

- **Transitions**

Part IV: UI

- Text
- HTML

Part V: Tests

How do you test that?

Most other labs:

Unit test

- One function at a time
- Single input, single output

```
test("play once"):
  assertEquals(
    update(State(...), 0, 1),
    State(board = ...,
           playerId = UID1)
  )
```

This lab:

(Complex) integration tests

- Multiple functions
- Indirect tests (views+events)

```
test("play once"):
  val st = transition(
    init(UIDS), Move(0, 1))
  val vw = project(st)(UID0)
  assertEquals(vw.playerId, UID1)
  (0 to 3).forall: r =>
    (0 to 3).forall: c =>
      assert(...vw.board(r, c)...)
  )
```

How do you test that?

Most other labs: Unit test

- One function at a time
- Single input, single output

```
test("encode"):  
  assertEquals(  
    encode(...),  
    Obj("abc" → ..., "def" → ...)  
  )
```

This lab: (Complex) integration tests

- Multiple functions
- Indirect tests (views+events)

```
test("decode-encode"):  
  assertEquals(  
    xyz,  
    decode(encode(xyz))  
  )
```

- Will this work if you use arrays or classes instead of Vectors and case classes?
- What kind of bugs will this fail to catch?

Part VI (time permitting)

Looking around the library

Tips to be successful

- For webapp-rps
 - Do the exercises
 - Look around (`Wires.scala`)
 - Write your own tests
- For the unguided lab
 - Don't skip webapp-rps
 - Review this lecture
 - Plan and communicate (plan first, code second)
 - Need proposal tips? **Ask in person**

Practice: Design an online card game

- **States**
- **Transition function**
- **Events**
- **Views**
- **Projection function**

Practice: Design an online card game

- **States**

Playing, Finished

+ Game-specific (Jass: Bidding, Tarot: Écart...)

- **Transition function**

Playing → Playing, Playing → Finished

+ Game-specific ones (Bidding → Playing...)

- **Events**

Bid, Pass, PlayCard

- **Views**

Game phase, individual hand

- **Projection function**

Hide other players' cards