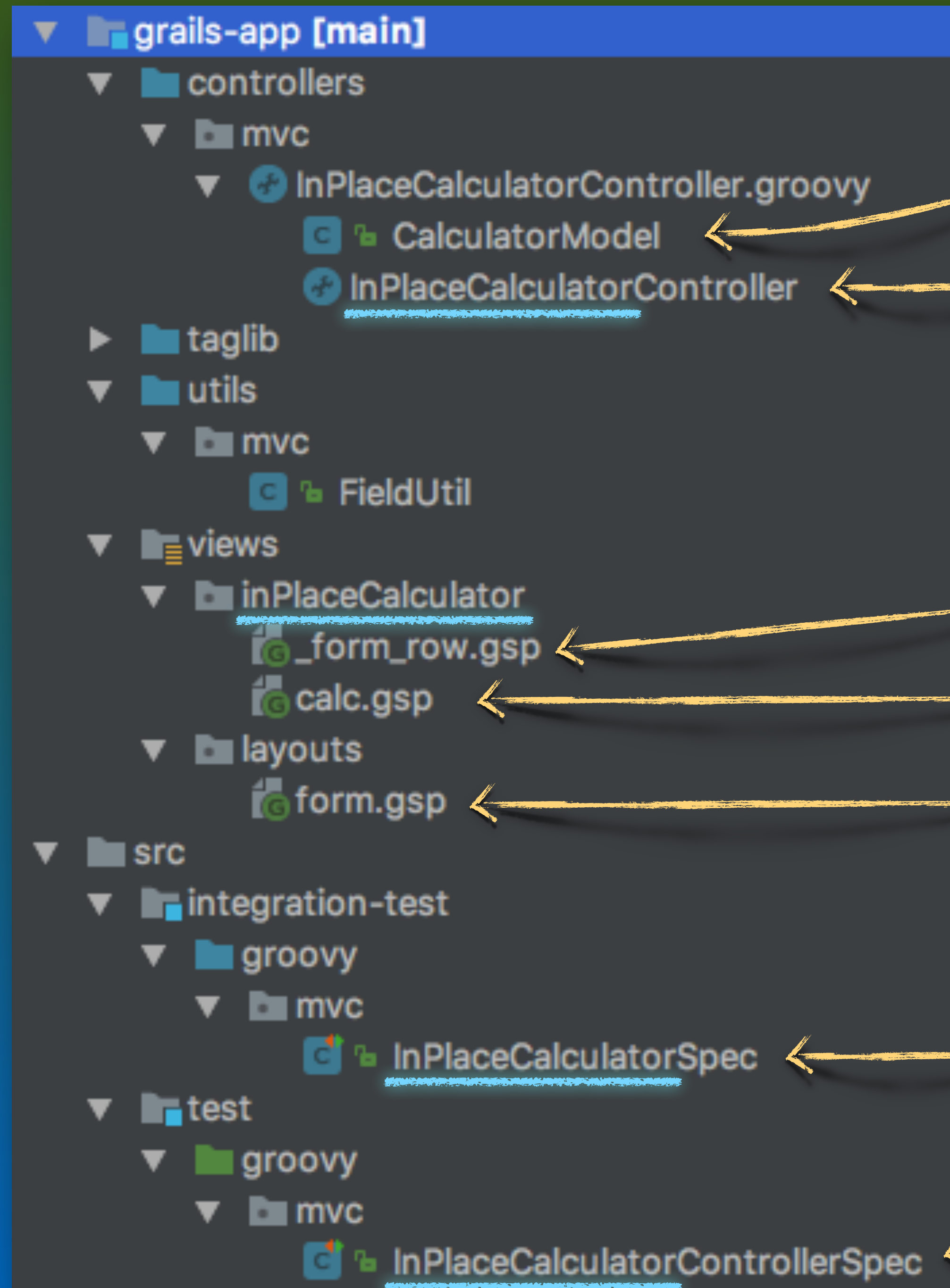


# Persistent State

Web Engineering

Prof. D. König

# Overall Structure

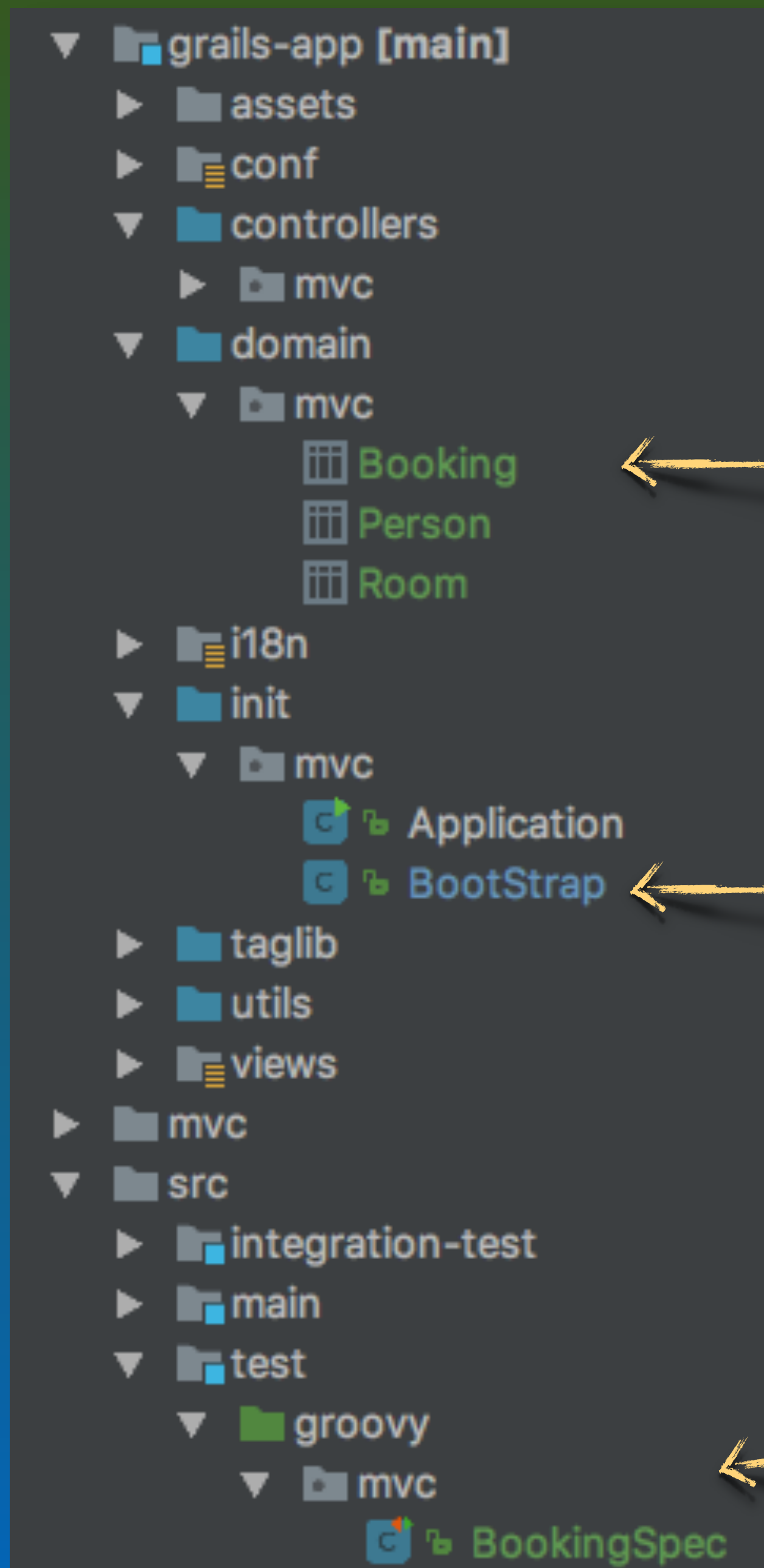


model  
controller

template  
server page  
layout

UI test  
unit test

# Overall Structure



*domain model*

*bootstrap*

*unit test*

# Domain Classes

```
package mvc

class Room {

    String description
    int capacity

    String toString() { description + "(" + capacity + ")" }

    static constraints = {
    }
}
```

# CRUD Methods

```
new Room(description: "5.3A17", capacity: 40).save()  
Room.list()  
Room.findAllByCapacityGreaterThan(20)  
def firstRoom = Room.get(1)  
firstRoom.delete()
```



# Grails ORM

<http://docs.grails.org/latest/guide/GORM.html>

Domain classes serve as DAO/DTO/Model

DB is set up automatically

Dynamic finder methods simplify usage

Keep it simple

# Scaffolding

static scaffold = *domainClassName*

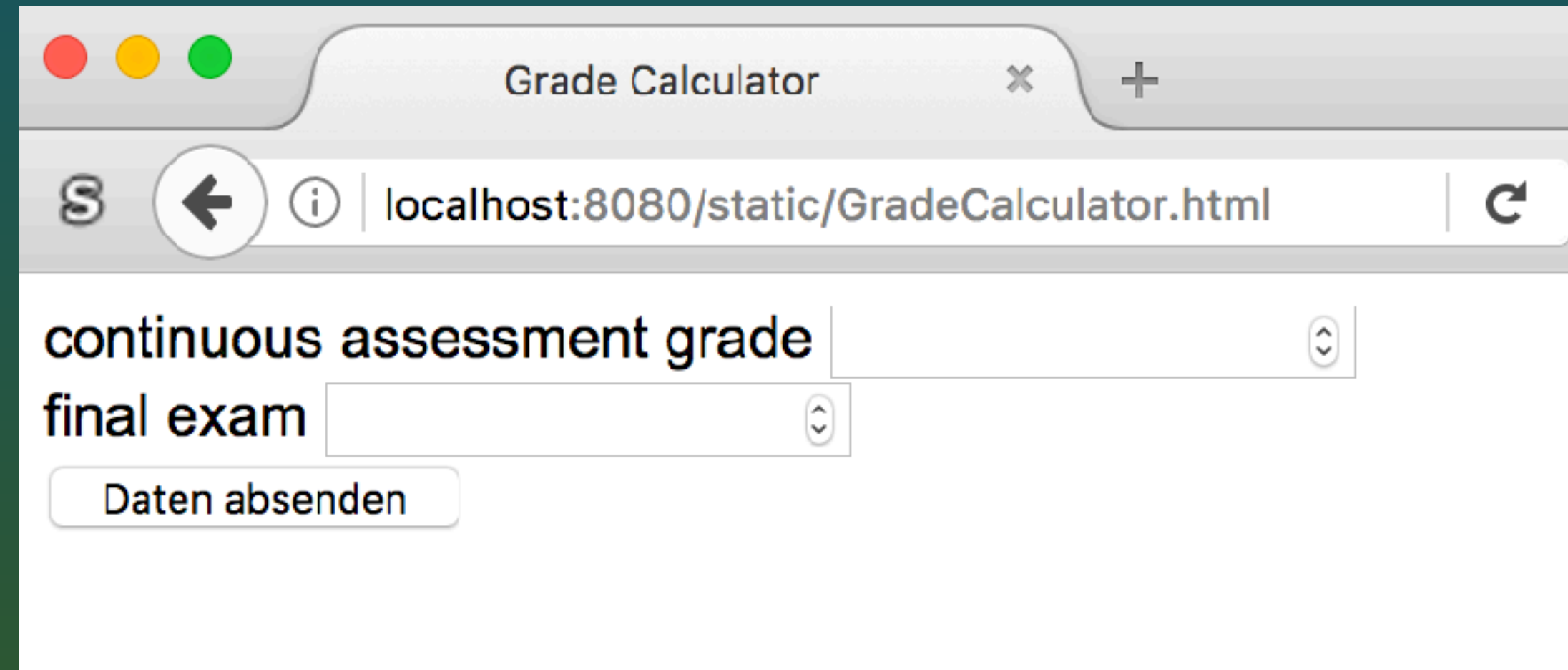
Controller actions and views are created transparently behind the scenes

We can selectively override

# HTML, CSS, JavaScript

*static*

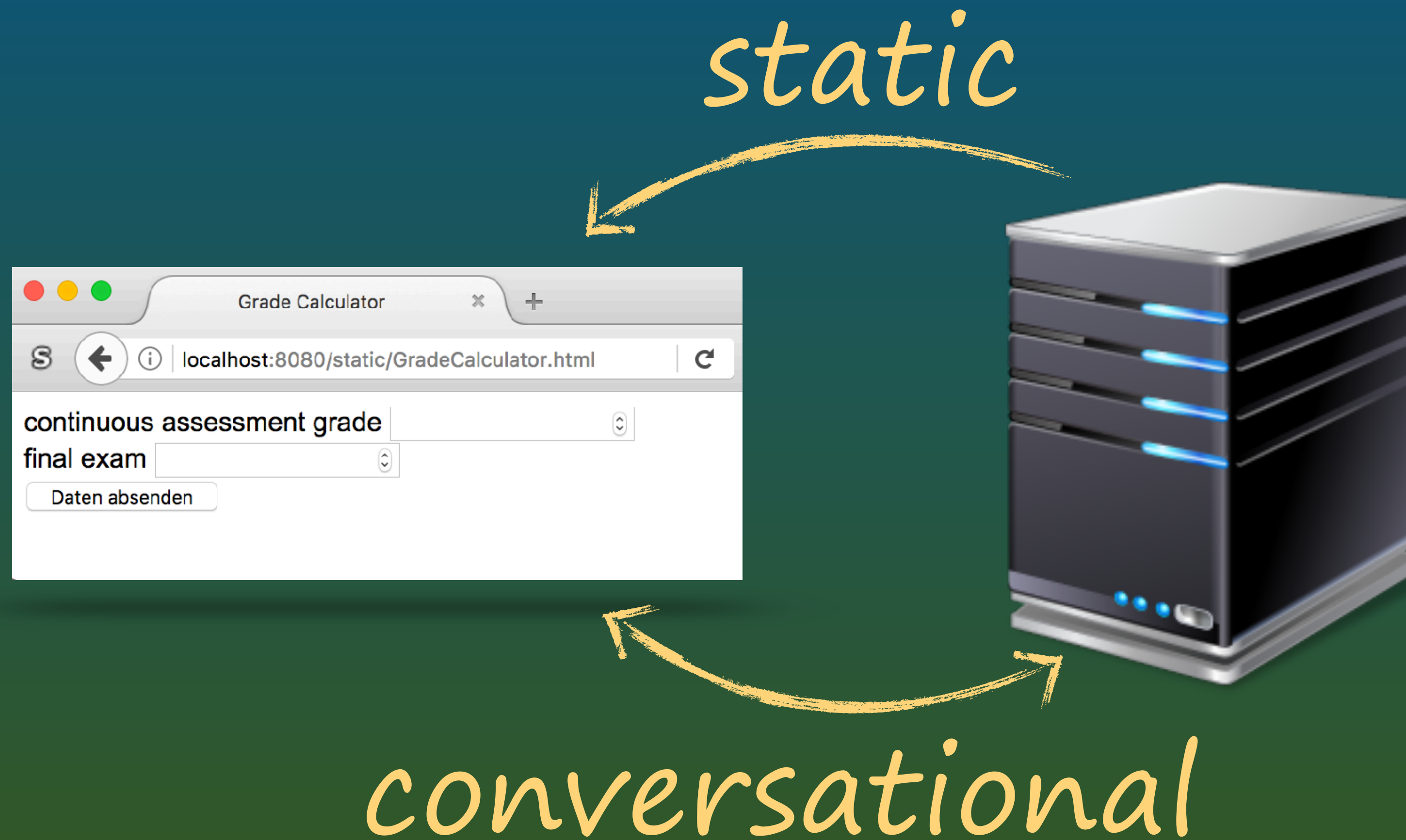
*dynamic*



A screenshot of a web browser window titled "Grade Calculator". The address bar shows the URL "localhost:8080/static/GradeCalculator.html". The form contains two input fields: "continuous assessment grade" and "final exam", both with dropdown arrows on the right. Below the fields is a button labeled "Daten absenden".

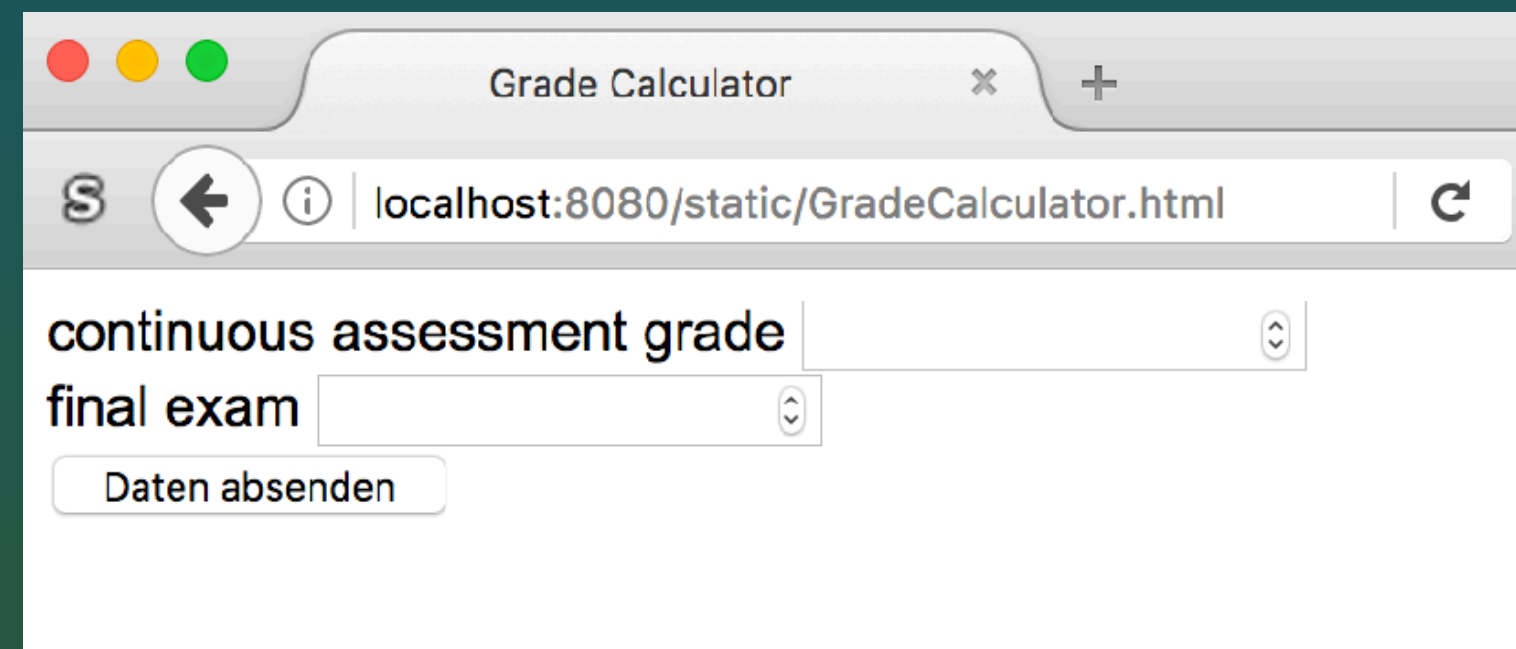


# Web MVC, Server Pages



# Three Tier Web Applications

*Client*



*Server*



*DB*



*persist*

# Where State Lives

HTML static	<i>DOM, forms</i>
HTML dynamic	<i>JS, localStorage</i>
Server static	<i>Deployment</i>
Conversation	<i>Session</i>
Persistent	<i>Database</i>