Week 6 Day 2

Spring Boot and Web

Spring Boot Intro

REVATURE

Open-source Spring framework

- Rapidly creates Java projects
- Simplifies project creation
  - Lets you pick and choose spring frameworks
  - Configures said frameworks for you
- Built in tomcat server
- Autoconfiguration with @SpringBootApplication
  - @SpringBootConfiguration
  - @EnabledAutoConfiguration
  - @ComponentScan

spring boot

Lombok

Java plugin installed in the IDE

- Generates boilerplate code at runtime
- Including
  - Getters and Setters
  - Constructors
  - toString() methods
  - equals() and hashCode()

MVC Review

Model View Controller pattern used to design user interfaces and structure applications

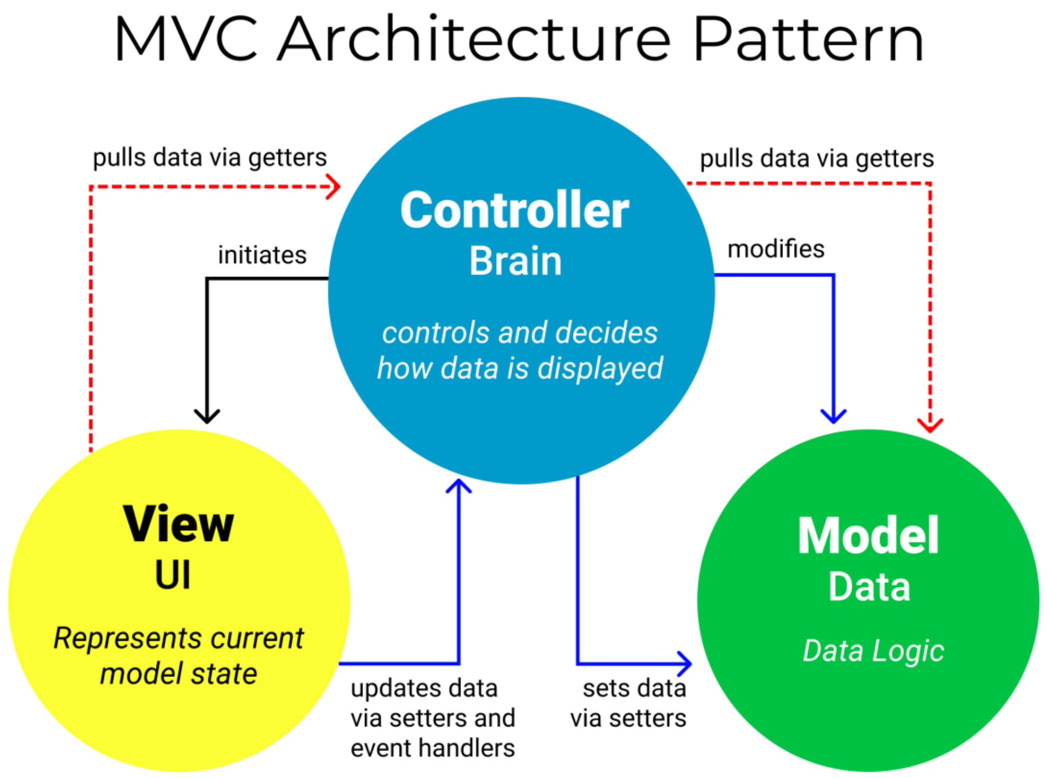


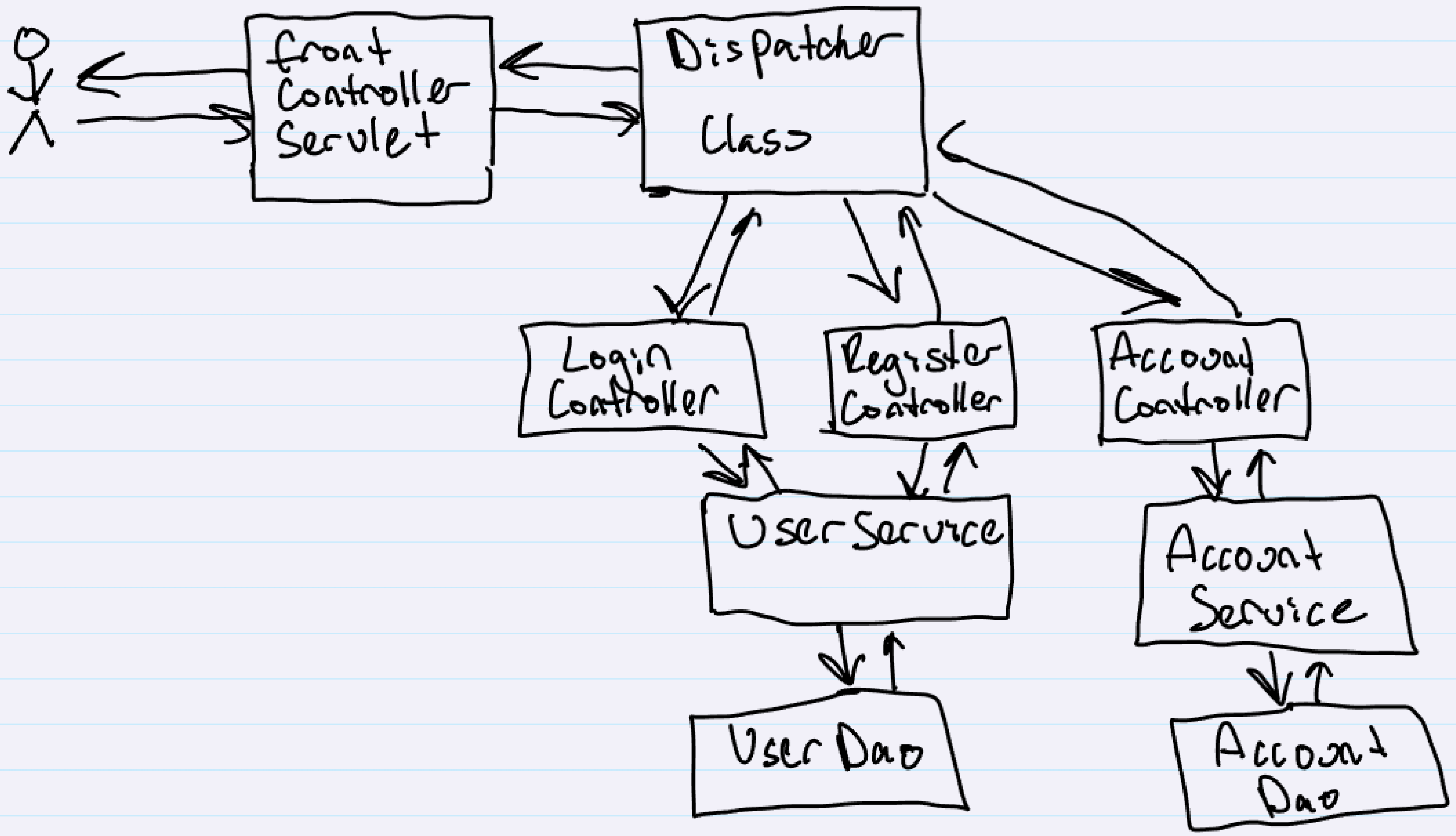
Image source: https://www.freecodecamp.org/news/the-model-view-controller-pattern-mvc-architecture-and-frameworks-explained/

Spring MVC

Request driven structure centered around a central servlet using Front Controller design pattern

- Uses servlets under the hood
- Front Controller Design
  - Provides a single handler for all incoming requests (front controller)
  - Dispatches the requests through a helper api (dispatcher)
  - Helper classes hold logic for specific features (controllers)
- InternalViewResourceResolver
  - Used to get views for our controllers

Front Controller Design Pattern



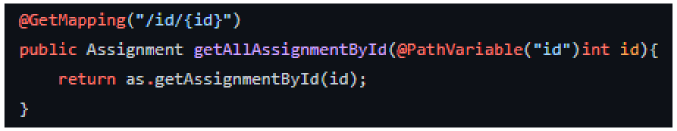
Spring MVC: Annotations

REVATURE

- @Controller
  - Marks class as a controller
- @RequestMapping
  - Used to map a uri to a class or method
- @RequestBody
  - Gain access to the request body
- @ResponseBody
  - Automatically convert the methods returned object to a json object to be sent in the request
- @RestController
  - Combines @RequestMapping and @ResponseBody

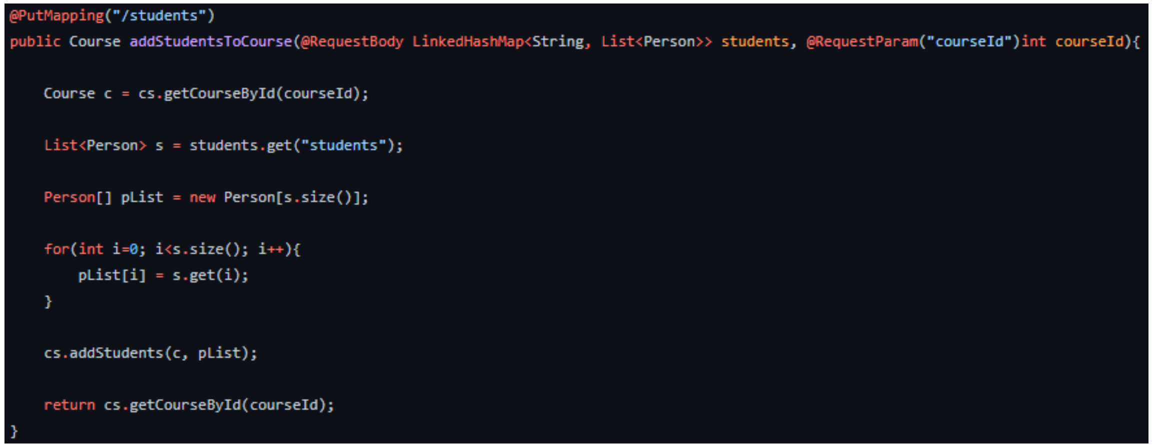
Spring MVC: Path and Query Parameters

- Path Variables/Parameters allow you to add values after a slash in your uri
  - http://url/get/{variable}



```
@GetMapping("/id/{id}")
public Assignment getAllAssignmentById(@PathVariable("id")int id){
    return as.getAssignmentById(id);
}
```

- Query Parameters allow you to use ? In the uri then define your key and value
  - http://url/get?var=name



```
@PutMapping("/students")
public Course addStudentsToCourse(@RequestBody LinkedHashMap<String, List<Person>> students, @RequestParam("courseId")int courseId){

    Course c = cs.getCourseById(courseId);

    List<Person> s = students.get("students");

    Person[] pList = new Person[s.size()];

    for(int i=0; i<s.size(); i++){
        pList[i] = s.get(i);
    }

    cs.addStudents(c, pList);

    return cs.getCourseById(courseId);
}
```

REVATURE

Spring Boot Webservice DEMO