

Objectives

- Review: databases, SQL, JDBC
- Software Engineering Tools
 - Maven
- Project Organization
 - MVC

May 7, 2021

Sprenkle - CSCI335

1

1

Review: Quality Attributes and Databases

- Quality attributes
 - What are quality attributes of web applications?
- Databases
 - Why are databases useful?
 - What are tables made up of?
 - What language do we use to query and update relational databases?
 - What is the syntax for the **SELECT** statement?
 - What is a *primary key* vs a *foreign key*?
 - How are SQL and JDBC related?
 - Compare and contrast Statements vs PreparedStatement in JDBC
 - What tips and tricks did you learn about writing/using SQL?

May 7, 2021

Sprenkle - CSCI335

2

2

Understanding the Code Base

PROJECTS

May 7, 2021

Sprenkle - CSCI335

3

3



May 7, 2021

Sprenkle - CSCI335

4

4

Apache Maven™

- Maven: Yiddish word meaning *accumulator of knowledge*
- For building and managing any Java-based project
 - Uses a **project object model (POM)**
- Goal: download and build a project quickly
- Can be used as standalone tool or within Eclipse (what we'll use)

<http://maven.apache.org/>

May 7, 2021

Sprenkle - CSCI335

5

5

Maven Philosophy: Convention Over Configuration

- Maven's location assumptions:
 - source code: `${basedir}/src/main/java`
 - Resources: `${basedir}/src/main/resources`
 - Tests: `${basedir}/src/test`
- Other assumptions:
 - Produce a JAR file in `${basedir}/target`
 - Compile byte code to `${basedir}/target/classes`

How does this convention philosophy help us?

May 7, 2021

Sprenkle - CSCI335

6

6

Maven Philosophy: Convention Over Configuration

- Benefit: reduces effort
 - Put source in the correct directory
 - Maven handles the rest
- Beyond location conventions...
- **Core plugins** apply a common set of conventions for compiling source code, packaging distributions, generating web sites, and many other processes

May 7, 2021

Sprenkle - CSCI335

7

7

Consequences of Convention Over Configuration

- Users may feel forced to use a particular methodology or approach
- Most defaults can be customized
- Can create custom plugins for your requirements

May 7, 2021

Sprenkle - CSCI335

8

8

Maven Build Lifecycle

- Defined by a list of *build phases*
- Example build phases
 - `compile` - compile the source code of the project
 - `test` - test the compiled source code using a suitable unit testing framework
 - `package` - take the compiled code and package it in its distributable format, such as a JAR
- When execute a phase, executes life cycle's previous phases first, in order
 - E.g., calling `package` would execute `compile` and then `test`

May 7, 2021

Sprenkle - CSCI335

9

9

Maven Build Lifecycle

- 3 built-in build lifecycles
 - `default` lifecycle handles project deployment
 - `clean` lifecycle handles project cleaning
 - `site` lifecycle handles the creation of project's site documentation

May 7, 2021

Sprenkle - CSCI335

10

10

Maven Repository

<https://mvnrepository.com/>

- Holds build artifacts and dependencies
- Typically: looking for a stable release
 - rc = release candidate (*not* what you want)

May 7, 2021

Sprenkle - CSCI335

11

11

Adding a Dependency

- Several different ways dependency can be added to `pom.xml` file

Maven | Gradle | SBT | Ivy | Grape | Leiningen | Buildr

```
<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-api</artifactId>
  <version>5.7.1</version>
  <scope>test</scope>
</dependency>
```

- Can copy the XML code provided on the Maven repository site

May 7, 2021

Sprenkle - CSCI335

12

12

Adding a Dependency

```
Maven  Gradle  SBT  Ivy  Grape  Leiningen  Buildr
<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-api</artifactId>
  <version>5.7.1</version>
  <scope>test</scope>
</dependency>
```

- After adding a repository, give Eclipse some time to work it out
 - Eclipse will download new dependencies
- View the Maven Dependencies in Eclipse

May 7, 2021

Sprenkle - CSCI335

13

13

Updating a Dependency

```
Maven  Gradle  SBT  Ivy  Grape  Leiningen  Buildr
<!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
<dependency>
  <groupId>org.junit.jupiter</groupId>
  <artifactId>junit-jupiter-api</artifactId>
  <version>5.7.1</version>
  <scope>test</scope>
</dependency>
```

- Changing versions is easy
 - Will see errors in POM if causes conflicts between dependencies
 - Will see compilation errors if upgraded version causes issues in your current code

May 7, 2021

Sprenkle - CSCI335

14

14

SPRING

May 7, 2021

Sprenkle - CSCI335

15

15

Spring Framework

- Open-source, powerful, and flexible framework focused on building Java applications
- Spring handles a lot of the common, “boilerplate” code so that you can focus on the logic for your application
 - Example: connecting to a database is fairly routine; just need to know which database, the user name, password, ...
- Dependence on conventions over configuration
- Multiple components, e.g., WebMVC, Boot, ...

May 7, 2021

Sprenkle - CSCI335

16

16

Git Repositories Organization

- Our projects [will] have a private and public repository
- Configuration of projects contain private/sensitive information
 - Server location
 - User names and passwords
- Our solution
 - Keep our work private until reach a checkpoint to make public
 - Put placeholders into the config files

May 7, 2021

Sprenkle - CSCI335

17

17

Software Development Review

- What is MVC?
 - In general, how does it apply to web applications?
 - Specifically, take the Login example (from yesterday) and identify the MVC

May 7, 2021

Sprenkle - CSCI335

18

18

Understanding a Code Base

- Sooo much code!
- Find your foothold
 - Where can you find a starting point and start exploring?
- Switch between low-level and high-level
 - Low-level: looking at the files and how they belong to the big picture
 - High-level: Look in the code for something you know the application does—follow the flow

May 7, 2021

Sprenkle - CSCI335

19

19

Documentation

Sunday night

- Document the code in the README.md and other files in your git repository
- Draw a picture of the architecture
 - Include in the documentation
- Identify the MVC
- Label the purposes of directories
 - Where are the configuration files? What can you configure?
- Follow the complete flow of one operation

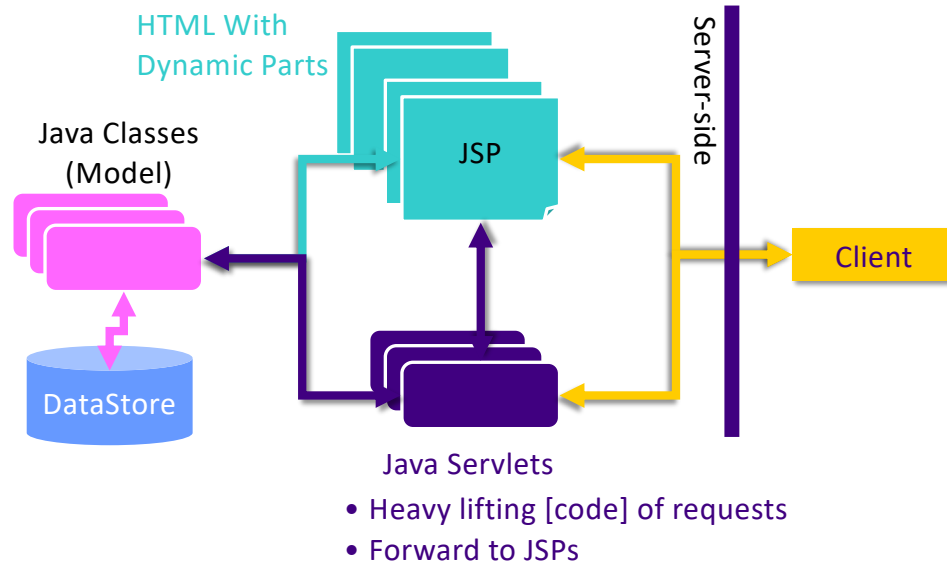
May 7, 2021

Sprenkle - CSCI335

20

20

Web Application Architecture



May 4, 2021

Sprenkle - CS335

21

21

Project Teams

- AGP
 - Ana, Grace, Will, Scott
- ChemTutor
 - Abdul, Jacob, Laurie, Alyssa, Jeremy
- Logic
 - Luke, Sam, Leslie, Yoseph

May 7, 2021

Sprenkle - CSCI335

22

22

Looking Ahead

- Project exploration and document – Sunday at 11:59 p.m. in your team's git repo
- Exam on Wednesday