SOFTWARE GUILD

Spring Core Unit – Software Development Lifecycle

Lesson 1 - Spring from 10,000 Feet

Objectives

- Understand the origins and philosophy of Spring
- Understand the 4 key Spring strategies
- Understand that Spring is built on the foundation of Dependency Injection and Aspect Oriented Programming
- Get a general sense of the depth and breadth of the overall Spring project

Why Spring?

- Spring's sole purpose is to simplify Java programming
- We must start with the base because all other aspects of Spring rely on the core
- A little history…
    - o EJBs
    - o Heavy frameworks

How does Spring help?

- Four main strategies:
  - Lightweight development with (P)lain (O)ld (J)ava (O)bjects (POJOs)
  - Loose coupling via dependency injection (DI) and interface orientation
  - Declarative programming via Aspect Oriented Programming (AOP) and conventions
  - Boilerplate reduction via templates

Development with POJOs

- Many frameworks require extension of their classes - locking you in
- Spring allows you to use POJOs
  - More testable
  - Not locked in
- Spring strives to be minimally invasive

Dependency Injection

- AKA - Inversion of Control
- Sounds scarier than it is…
- Classes that obtain their own references to collaborating objects lead to tightly coupled code

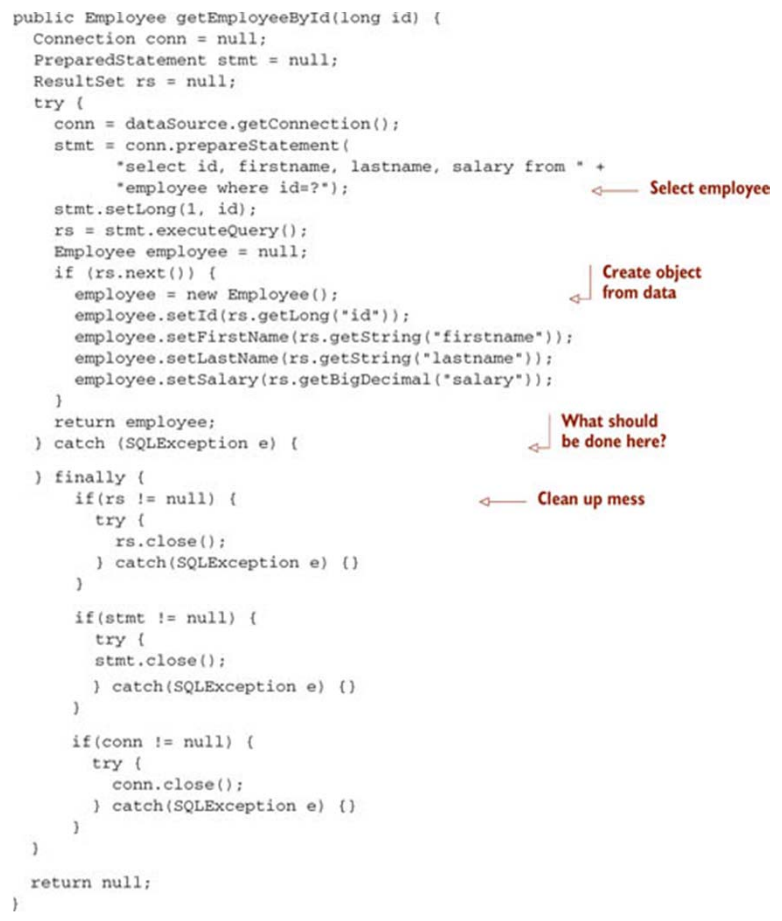
Aspect Oriented Programming (AOP)

- AOP allows system wide code to be placed in reusable containers
- Promotes good separation of concerns
- Without AOP code for cross cutting concerns (i.e. logging, security, etc) is spread across the code base - violates D.R.Y.
- Without AOP, components are littered with code that is not core to their functionality

Templates

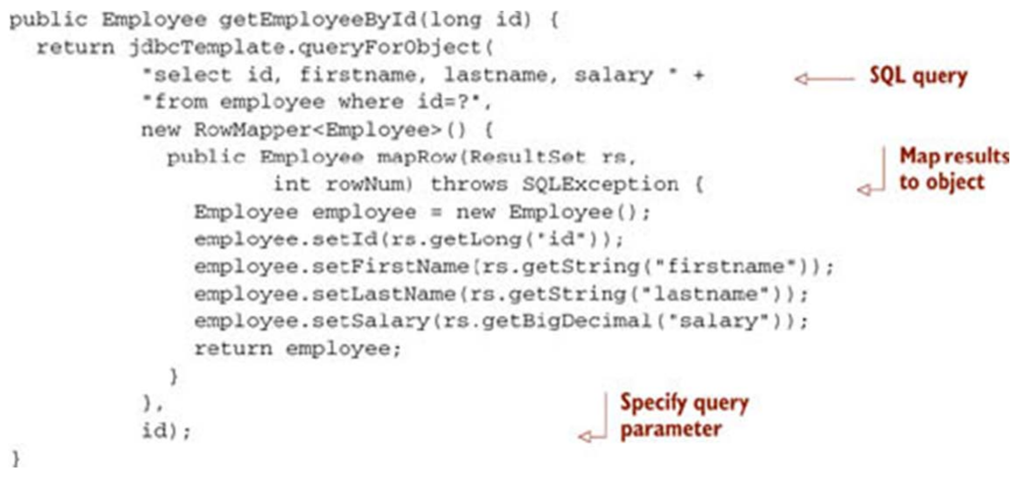
- Templates reduce ‘boilerplate’ code
- See the following example
    - Even if you don’t understand the code you can see that the second example is simpler and cleaner than the first

JDBC Boilerplate Code



```java
public Employee getEmployeeById(long id) {
  Connection conn = null;
  PreparedStatement stmt = null;
  ResultSet rs = null;
  try {
    conn = dataSource.getConnection();
    stmt = conn.prepareStatement(
          "select id, firstname, lastname, salary from " +
          "employee where id=?");
    stmt.setLong(1, id);
    rs = stmt.executeQuery();
    Employee employee = null;
    if (rs.next()) {
      employee = new Employee();
      employee.setId(rs.getLong("id"));
      employee.setFirstName(rs.getString("firstname"));
      employee.setLastName(rs.getString("lastname"));
      employee.setSalary(rs.getBigDecimal("salary"));
    }
    return employee;
  } catch (SQLException e) {

  } finally {
      if(rs != null) {
        try {
          rs.close();
        } catch(SQLException e) {}
      }

      if(stmt != null) {
        try {
        stmt.close();
        } catch(SQLException e) {}
      }

      if(conn != null) {
        try {
          conn.close();
        } catch(SQLException e) {}
      }
  }

  return null;
}
```

Template Code

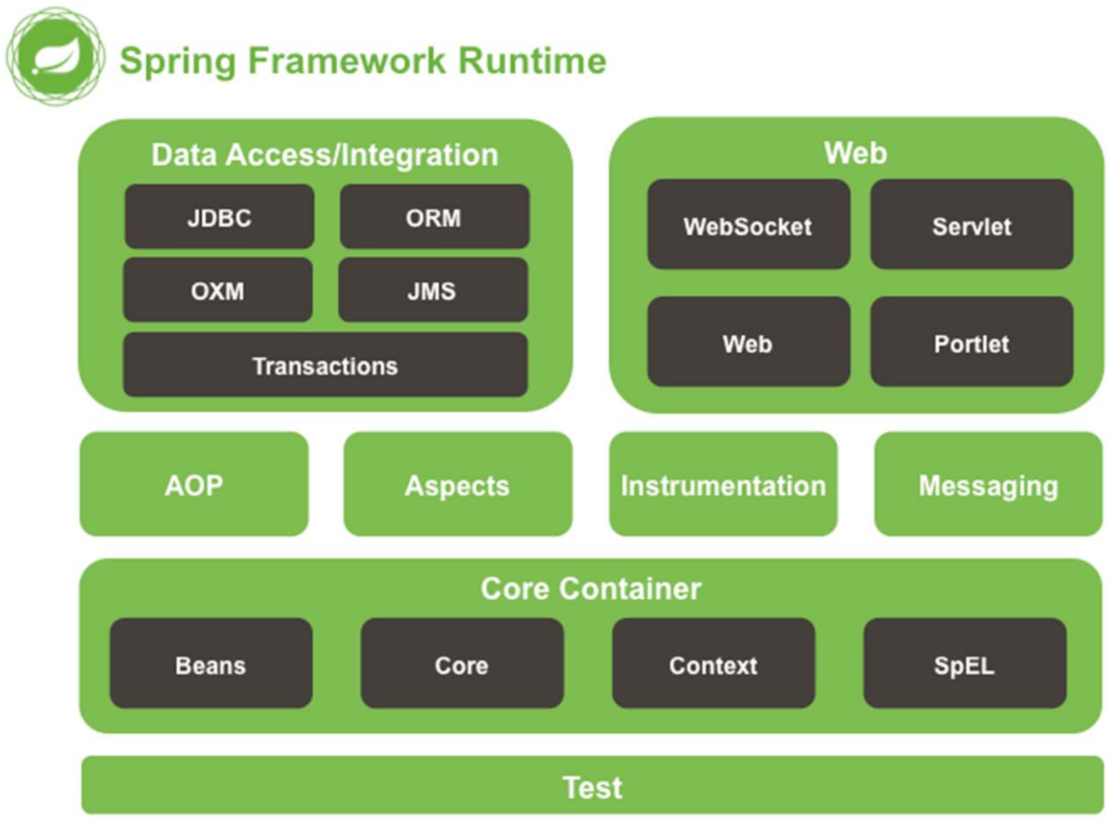


```java
public Employee getEmployeeById(long id) {
  return jdbcTemplate.queryForObject(
           "select id, firstname, lastname, salary " +
           "from employee where id=?",
           new RowMapper<Employee>() {
             public Employee mapRow(ResultSet rs,
                     int rowNum) throws SQLException {
               Employee employee = new Employee();
               employee.setId(rs.getLong("id"));
               employee.setFirstName(rs.getString("firstname"));
               employee.setLastName(rs.getString("lastname"));
               employee.setSalary(rs.getBigDecimal("salary"));
               return employee;
             }
           },
           id);
}
```

Spring Container

- Manages objects - beans are wired, created, and destroyed by the container
- The environment is called the **Application Context** - it is defined in an XML file
- Spring manages the lifecycle of components we place under its control
- We'll code an example when we talk about DI

Core Spring Framework



Spring Ecosystem

- Spring Core is just the beginning
- Many related projects
- Let’s take a look at spring.io