Week 6 Day 4

Additional Spring Boot Tools

Spring Boot DevTools

REVATURE

- Cache Disabling
  - Caching can hinder development,
- Automatic Restarts
  - Any time your code changes the server restarts

Spring Boot Actuator

REVATURE

Library that exposes tools for monitoring and gathering metrics about your app

- Enable or disable features in the application.properties
  - /health
  - /beans
  - /env
  - /info
  - /logfile
  - /loggers
  - /mapping
  - /metrics
  - /shutdown

Spring Validator

REVATURE

Spring interface which is used to validate/verify objects

- Checks the object vs different criteria
- On failed validation throws a Custom Spring Error object
- The Error Object stores special information about the failed validation

RestTemplates

REVATURE

Used to make Http requests with Java/Spring apps

- Acts as a webclient
- getForEntity()
- getForObject()
- exchange()

```
RestTemplate rest = new RestTemplate();

String url = "https://jsonplaceholder.typicode.com/posts";

ResponseEntity<String> res = rest.getForEntity(url, String.class);

return res;
```

Aspect Oriented Programming (AOP)

REVATURE

- Concern ourselves with Aspects
  - Modularize particular concerns which present themselves across multiple classes
- Examples
  - Database access
  - Data entities
  - Error handling
  - Logging
- Help reduce code redundancy

@AspectJ

REVATURE

Refers to a style of declaring aspects through the use of Java classes

- Spring interprets the same annotations as AspectJ
- You must enable support for Spring AOP based AspectJ aspects in your beans.xml
- Spring also uses autoproxying of beans which needs to be enabled

Spring AOP

REVATURE

- Spring module to help you manage Cross Cutting Concerns
- Cross Cutting Concerns
  - Parts of a program that rely on, or must affect other parts of the system
- Manages these with Advice, JoinPoints, and PointCuts

AOP: Advice

REVATURE

- Action taken at a Joinpoint
  - Before: executed before a joinpoint
  - After Returning: executed after returning from a joinpoint successfully
  - After Throwing: executed after an exception occured
  - After: executed after a joinpoint regardless of the outcome
  - Around: Allows you to run logic before and after the joinpoint

AOP: Pointcut and Joinpoints

REVATURE

- Pointcuts determine joinpoints
  - Comprised of a name, parameters
  - Determine EXACTLY which methods we are interested in
- Joinpoints are specific moments during execution to take advice

```
//Execute this advice before any method from PersonService is called
@Before("execution(* com.example.services.PersonService.*(..))")
public void beforeAnyPersonService(JoinPoint jp){
    System.out.println("Person service method: " + jp.getSignature() + " was called");
}
```

REVATURE

DEMO: AOP