What is Spring and Spring Boot?

Spring:
- Most popular java framework
- Mature and lightweight
- Inversion of control
- Dependency Injection

Spring Boot:
- Enables quick access to Spring framework
- Opinionated
- Increased developer productivity
Why we use

- Let Spring handle routine tasks
- Manages webdriver instance
- Manage Page objects/fragments
- Maintains multiple test environments
- Localization testing for multiple languages
- Accessing files
- Taking screenshots
- Parallel test execution
- Executing cucumber files with automatic dependency injection
Prerequisites

1. Java – very comfortable
2. Selenium – Basics
3. RestAssured – Basics
4. Junit/TestNg – Basics
5. Cucumber – Not really required
Coverage

- Springboot basics
- Bean scope
- Aspect oriented programming
- Accessing files
- Logging
- Cucumber integration
- Environment management
- Springboot with selenium
Object Dependency

- Page Object (wait for element)
- WebDriverWait
- TimeOut
- WebDriver
- Path / URL / Options
Dependency Injection

- Primitive types
- Properties
- Environment Variables
- URLs
- File/Path
- Default values
- Constructors
- External libraries
Inversion of Control

- Automatic object creation
- Method invocation
- Life cycle management
Webdriver – Bean

Web Driver managed by Spring:

WebDriver managed by you:

```java
@Component
public class WebDriverFactory {
    public WebDriver getDriver(String browser){
        if("chrome".equals(browser)){
            return new ChromeDriver();
        } else {
            return new FirefoxDriver();
        }
    }
}
```

```java
@Autowired
private WebDriverFactory driverFactory;

...;
this.driverFactory.getDriver("chrome");
this.driverFactory.getDriver("firefox");
```
Properties:

- `application-qa.properties`
  - Property: `spring.profiles.active=qa`

- `application-stg.properties`
  - Property: `spring.profiles.active=stg`

- `application.properties`

- `@Value`
Application Context

Singleton
- Configured one time

Prototype
- Configured every time with latest beans available

Spring Container
- Webdriver - Bean
File Properties

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>classpath</td>
<td>classpath: data/udemy.txt</td>
</tr>
<tr>
<td>file:</td>
<td>file:c:/some/path/udemy.txt</td>
</tr>
<tr>
<td>http:</td>
<td><a href="http://sometime.com/udemy.txt">http://sometime.com/udemy.txt</a></td>
</tr>
</tbody>
</table>

- To read any additional property files which are not part of application.properties.
- Instead of injecting values in multiple places, just inject once!
- Properties into a Java object!
Logging

- We can create logger for any class.
- Default – console.output

```java
private static final Logger logger = LoggerFactory.getLogger(GooglePage.class);
```

Levels:
- Error
- WARN
- INFO
- DEBUG
- TRACE
Cucumber Integration

- Integrated with Cucumber
- Dependency Injection in the Step Definitions
- Reporting
- Failed tests screenshots
- Parallel scenarios run
- Filtering tags
- Cucumber Runner & Command line execution
- Closing browsers
Any further questions?

Contact me
@ashengg89@gmail.com