9th UCAAT User Conference on Advanced Automated Testing

# A Spring in Tester's life Using Spring Boot

AshokKumar Rajendiran

PagoNxt | Getnet

15/09/2022

# 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

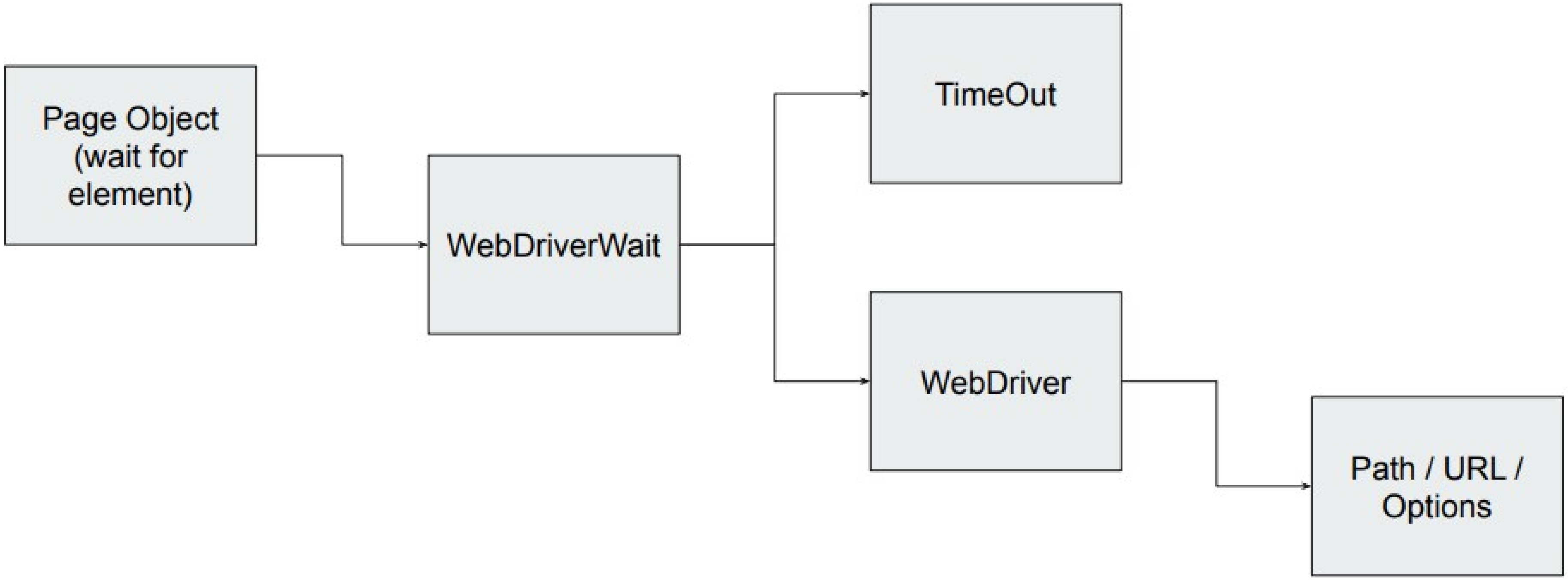
# Prerequisites

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

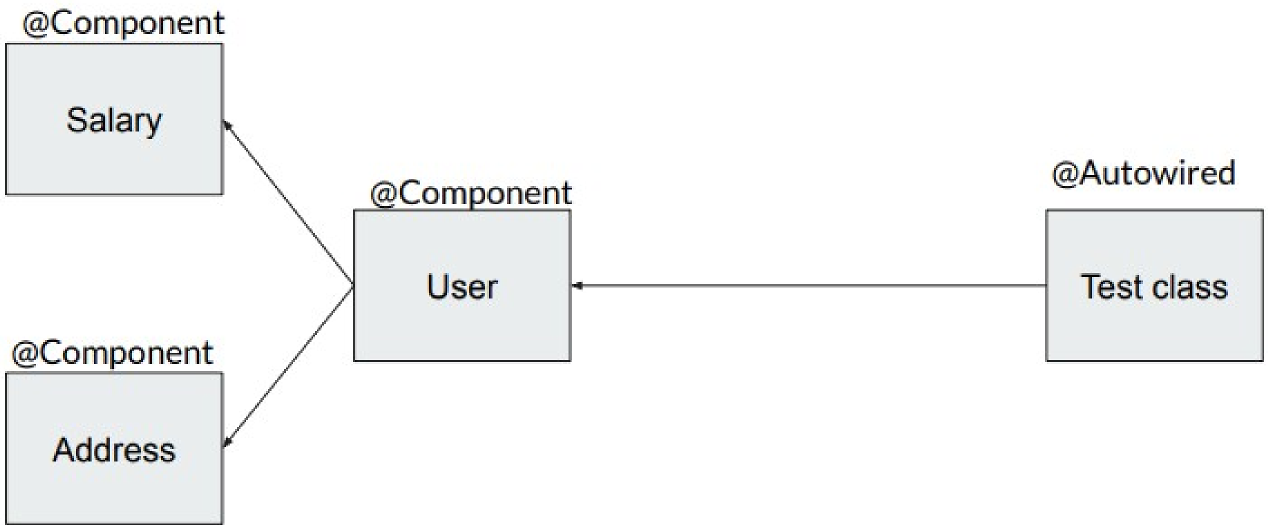
# Coverage

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

# Object Dependency



# Dependency Injection



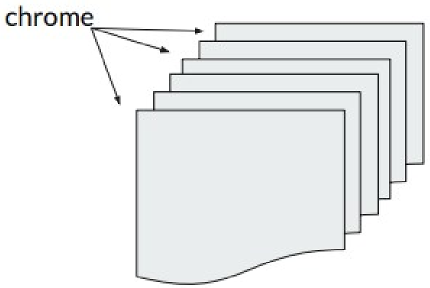
- Primitive types
- Properties
- Environment Variables
- URLs
- File/Path
- Default values
- Constructors
- External libraries

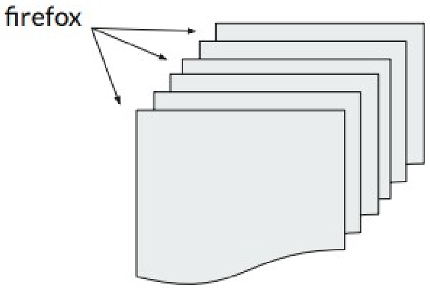
# Inversion of Control

- Automatic object creation
- Method invocation
- Life cycle management

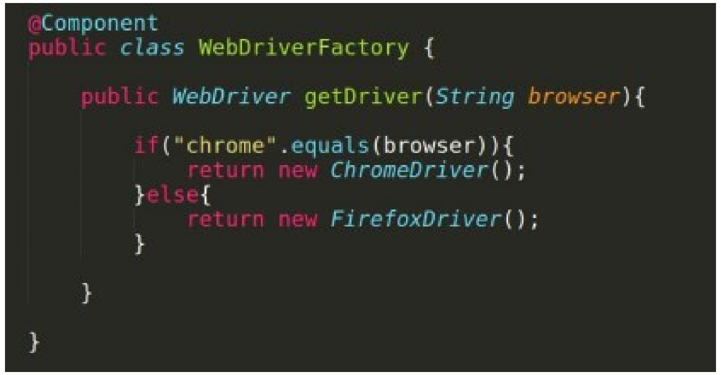
# Webdriver – Bean

Web Driver managed by Spring:

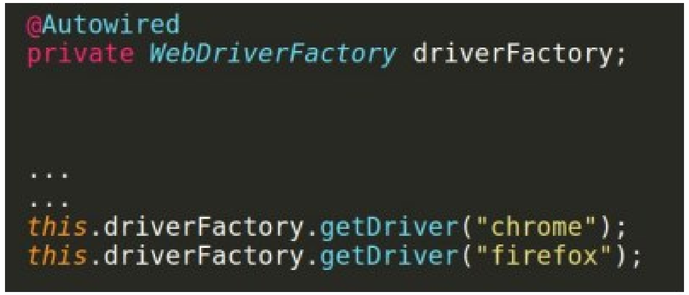




WebDriver managed by you:

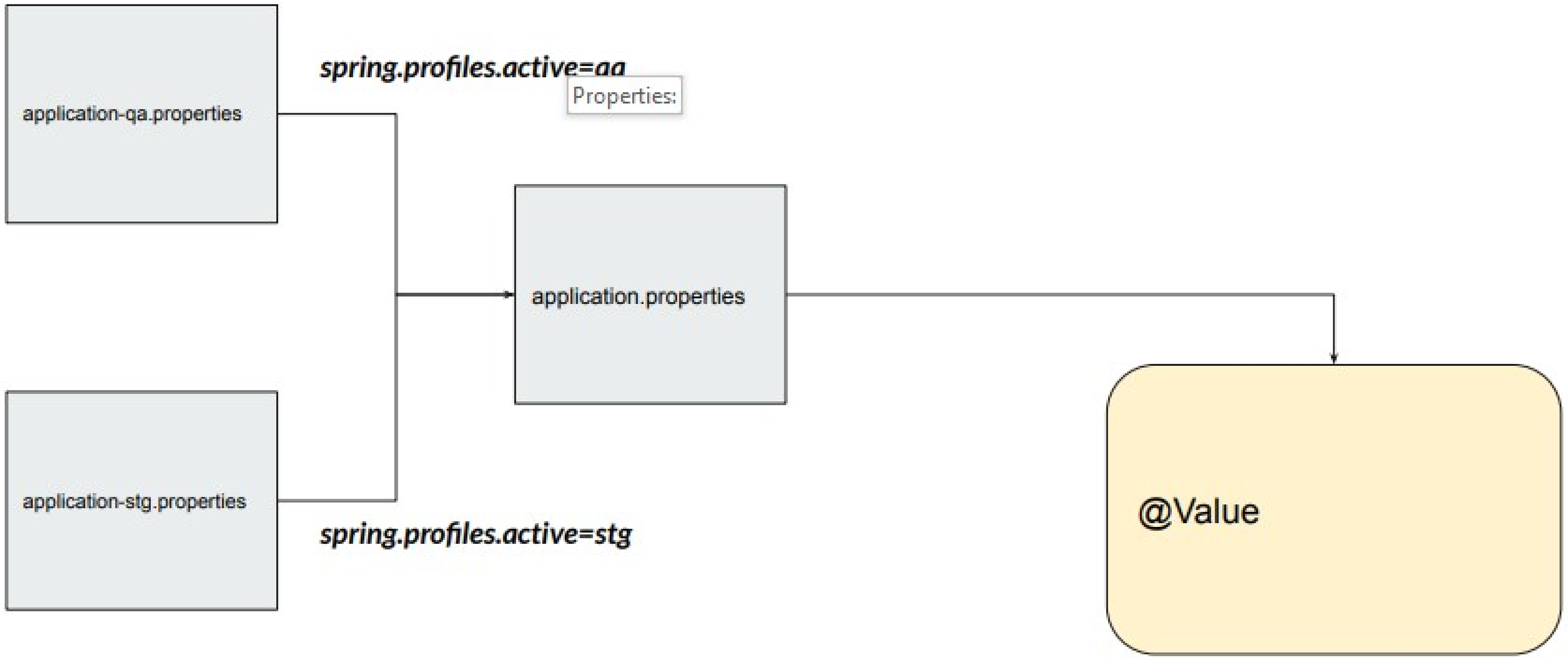


```
@Component
public class WebDriverFactory {

    public WebDriver getDriver(String browser){

        if("chrome".equals(browser)){
            return new ChromeDriver();
        }else{
            return new FirefoxDriver();
        }

    }

}
```



```
@Autowired
private WebDriverFactory driverFactory;



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

# Properties:



application-qa.properties

spring.profiles.active=qa

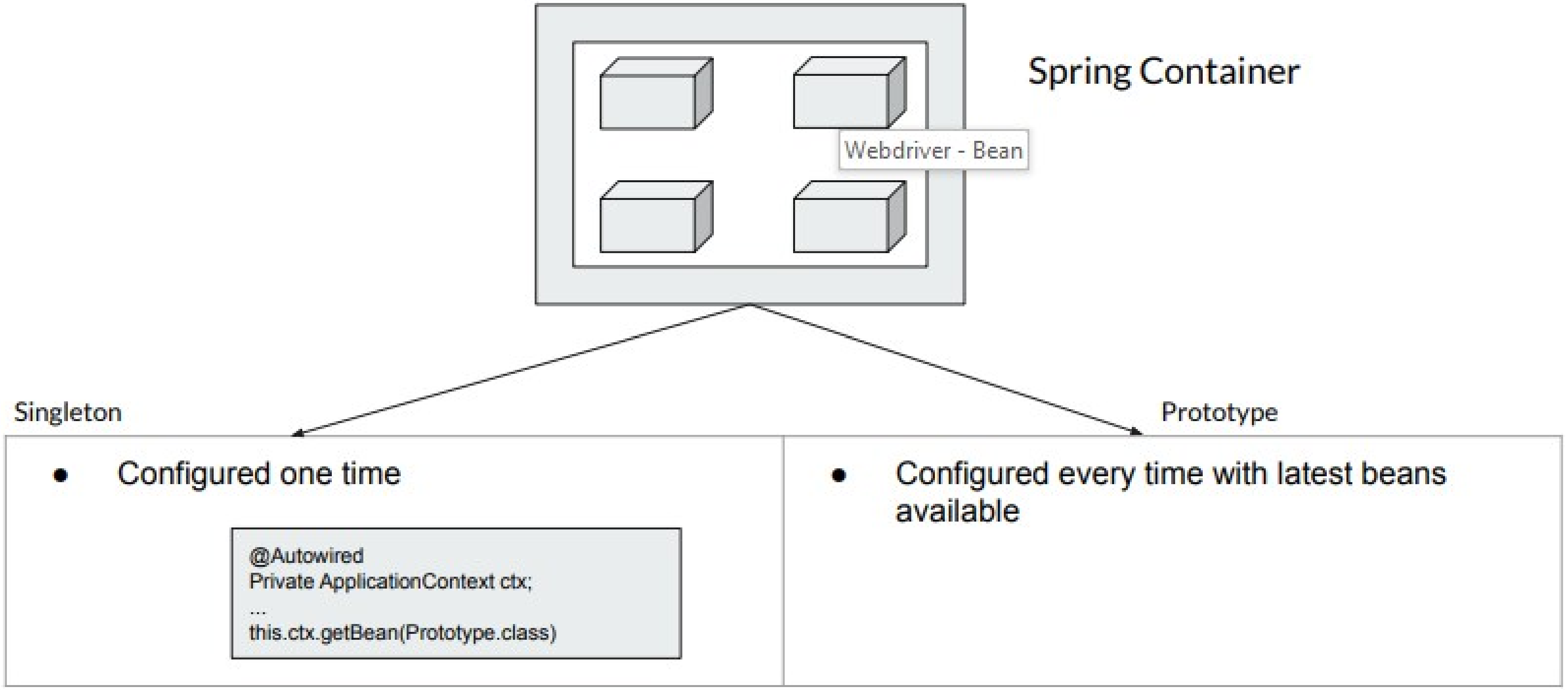
application-stg.properties

spring.profiles.active=stg

application.properties

@Value

# Application Context



Singleton

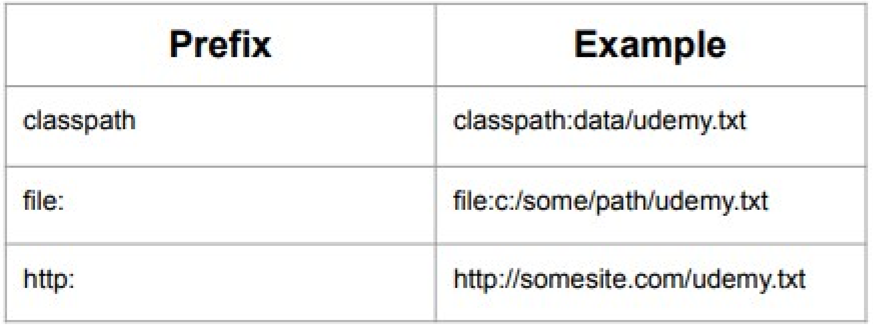
- Configured one time

```
@Autowired
Private ApplicationContext ctx;
...
this.ctx.getBean(Prototype.class)
```

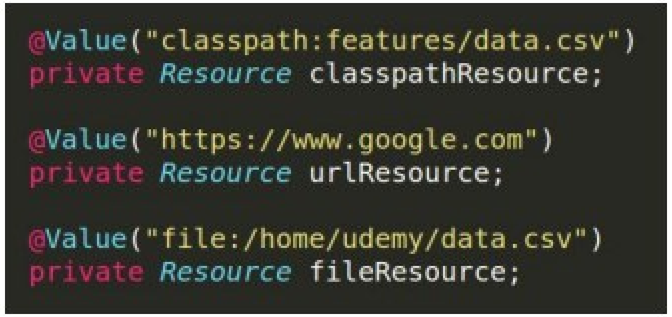
Prototype

- Configured every time with latest beans available

# File Properties



| Prefix | Example |
|---|---|
| classpath | classpath:data/udemy.txt |
| file: | file:c:/some/path/udemy.txt |
| http: | http://somesite.com/udemy.txt |

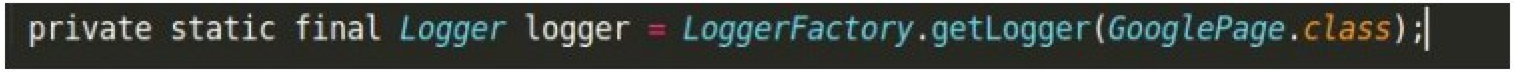


```
@Value("classpath:features/data.csv")
private Resource classpathResource;

@Value("https://www.google.com")
private Resource urlResource;

@Value("file:/home/udemy/data.csv")
private Resource fileResource;
```

- 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



```
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