Glue Code: Automating Scenarios

John Ferguson Smart
AUTHOR OF “BDD IN ACTION”

@wakaleo www.johnfergusonsmart.com

Overview

Test automation layering

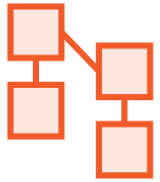
Test runner classes

Cucumber expressions

Layers of Test Automation Code

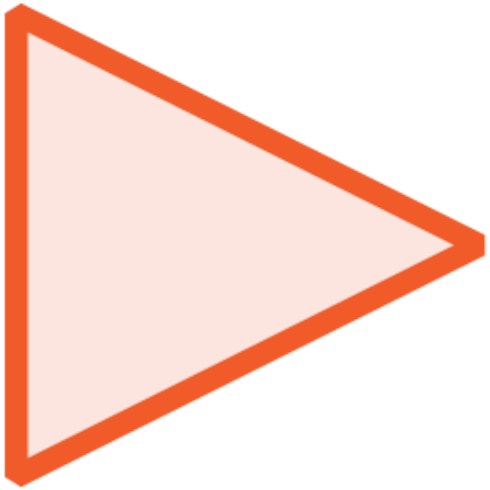
Executable specifications (Gherkin)

Glue code (Cucumber Step Definitions)



Test domain layer

Running a Gherkin Scenario

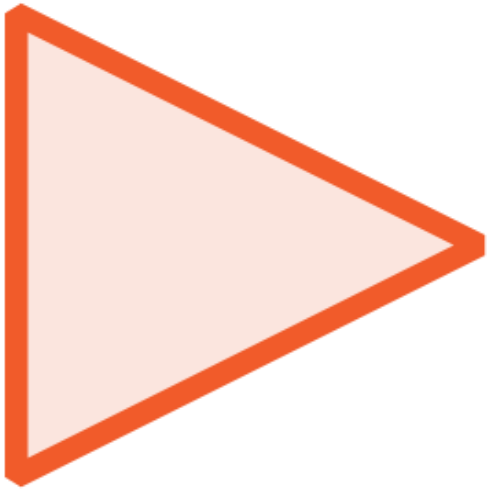


Test runner class

Feature files

Glue code

Running a Gherkin Scenario



Test runner class

Feature files

Glue code

```java
import cucumber.api.CucumberOptions;
import cucumber.api.junit.Cucumber;
import org.junit.runner.RunWith

@RunWith(Cucumber.class)
@CucumberOptions(plugin = {"pretty"},
                 glue = "com.pluralsight.bdd.steps"
                 features = "classpath:features")
public class AcceptanceTestSuite {}
```

Test Runner Class

- ✓ **Executes one or more feature files**
- ✓ **Configures Cucumber execution options**

```java
import cucumber.api.CucumberOptions;
import cucumber.api.junit.Cucumber;
import org.junit.runner.RunWith

@RunWith(Cucumber.class)
@CucumberOptions(plugin = {"pretty"},
                 glue = "com.pluralsight.bdd.steps"
                 features = "classpath:features")
public class AcceptanceTestSuite {}
```

Test Runner Options

- ✓ **Define the package for your step definitions**
- ✓ **Define the resources folder for your feature files**

Glue Code: Cucumber Expressions

```
Scenario: Ordering many smoothies
  Given I have ordered 9 smoothies
```

```
import cucumber.api.java.en.Given;

public class OrderingSmoothiesStepDefinitions {

    @Given("I have ordered 9 smoothies")
    public void iHaveOrdered9Smoothies() {
        //...
    }
}
```

Integer parameter

```
Scenario: Ordering many smoothies
  Given I have ordered 9 smoothies
```

```
import cucumber.api.java.en.Given;

public class OrderingSmoothiesStepDefinitions {

    @Given("I have ordered {int} smoothies")
    public void iHaveOrdered(int smoothieCount) {
        //...
    }
}
```

Scenario: Ordering many smoothies
Given I have ordered a "Green Goodness Smoothie"

String

```
import cucumber.api.java.en.Given;

public class OrderingSmoothiesStepDefinitions {

    @Given("I have ordered {string}")
    public void iHaveOrdered(String order) {
        //...
    }
}
```

Scenario: Ordering many smoothies
Given I have ordered 9 “Green Goodness Smoothie”

```
import cucumber.api.java.en.Given;

public class OrderingSmoothiesStepDefinitions {

    @Given("I have ordered {int} {string}")
    public void iHaveOrdered(int count, String order) {
        //...
    }
}
```

```
Scenario: Ordering smoothies as a Gold member
  Given I am a Gold Morning Freshness member
```

Word

```
import cucumber.api.java.en.Given;

public class OrderingSmoothiesStepDefinitions {

    @Given("I am a {word} Morning Freshness member")
    public void iAmAMorningFreshnessMember(String memberLevel) {
        //...
    }
}
```

Scenario: Ordering a smoothie
Given a "Strawberry Smoothie" costs $5.95

Float

```
import cucumber.api.java.en.Given;

public class OrderingSmoothiesStepDefinitions {

    @Given("a {string} costs ${float}")
    public void costOfASmoothie(String smoothie,
                                float cost) {

        //...
    }
}
```

```
Scenario: Ordering a smoothie
  Given Michael is a Morning Smoothie member
  When he orders a Green Goodness Smoothie
```

Anonymous

```
import cucumber.api.java.en.Given;

public class OrderingSmoothiesStepDefinitions {

    @Given("he orders a {} Smoothie")
    public void ordersASmoothie(String smoothie) {
        //...
    }
}
```

Glue Code: Regular Expressions

Regular Expressions in Cucumber

Identify which parts of your scenarios should be passed as parameters

More powerful than Cucumber Expressions

Harder to read than Cucumber Expressions

Simple Regular Expressions

Regular Expression	Meaning
^	The start of the string
$	The end of the string
(.*)	Matches anything (including nothing)
(.+)	Matches at least one of anything

```
Scenario: Ordering smoothies
  Given I have ordered 9 smoothies
```

```
@Given("^I have ordered (.*) smoothies$")
public void iHaveOrdered(int smoothieCount) {…}
```

More Regular Expressions

Regular Expression	Meaning
`(\\d+)`	A sequence of digits
`(\\w+)`	A sequence of letters or digits

```
Scenario: Ordering smoothies
  Given I have ordered 9 Green Goodness smoothies
```

```
@Given("^I have ordered (\\d+) (.*) smoothies$")
public void iHaveOrdered(int smoothieCount
                         String smoothieType) {…}
```

Equivalent Expressions

"Given I order a smoothie"

"Given I have ordered a smoothie"

"Given I have ordered an apple smoothie"

Matching Optional Text

Regular Expression	Meaning
`s?he`	Optional character: matches “she” or “he”
`an?`	Matches “a” or “an”
`(?:order|have ordered)`	Matches “orders” or “has ordered”, but does not capture it as a parameter

```
Scenario: Michael orders a smoothie
  Given Michael is a Morning Freshness member
  When he orders 2 Green Goodness smoothies
  Then the total cost should be $10.0
```

```
@Given("^s?he (?:orders|has ordered) (\\d+) (.*) smoothies?$")
public void hasOrdered(String memberName,
                       Integer count,
                       String itemName) {…}
```

Glue Code: Working with Tables

```
Scenario: Proposing appropriate special deals
  Given Michael is a Morning Freshness member
  And his favorite flavors are:
    | Banana |
    | Apple  |
```

```
@Given("his favorite flavors are:")
public void his_favorite_flavors_are(List<String> favorites) {…}
```

```
When the daily specials are:
| Title                          | Flavors                  |
| 20% on all Banana Smoothies    | Banana                   |
| 10% on all Berry Smoothies     | Strawberry, Blueberry    |
```

```
@When("the daily specials are:")
public void the_daily_specials_are(List<Map<String,String>> specials) {…}
```

```
Then Michael should see the following specials:
| Title                          | Flavors |
| 20% on all Banana Smoothies | Banana  |
```

```
@Then("{} should see the following specials:")
public void the_daily_specials_are(String memberName,
                                   List<Map<String,String>> specials) {…}
```

```
And he should see the following calorie counts:
  | Basic Banana Smoothie          | 180 |
  | Deluxe Banana Smoothie         | 240 |
  | Banana and Ice Cream Smoothie  | 350 |
```

```
@Then("he should see the following calorie counts:")
public void calorie_counts_are(Map<String,Integer> calorieCounts) {…}
```

```
Scenario: Michael orders a smoothie
  Given Michael is a Morning Freshness member
  When he orders 2 Green Goodness smoothies
  Then the total cost should be $10.0
```

```
@Given("^s?he (?:orders|has ordered) (\\d+) (.*) smoothies?$")
public void hasOrdered(String memberName,
                       Integer count,
                       String itemName) {…}
```

```
Scenario Outline: Michael orders a smoothie
  Given Michael is a Morning Freshness member
  When he orders <Quantity> Green Goodness smoothies
  Then the total cost should be $<Total>

Examples:
  | Quantity | Total |
  | 1        | 5.00  |
  | 2        | 10.00 |
  | 3        | 15.00 |
```

```
@Given("^s?he (?:orders|has ordered) (\\d+) (.*) smoothies?$")
public void hasOrdered(String memberName,
                       Integer count,
                       String itemName) {…}
```

Demo

Writing Glue Code

Demo

Cucumber Step Definitions

Demo

Implementing Step Definition Methods

Demo

Implementing Step Definition Methods

Part 2

Summary

What have you learned?

- Test automation framework layers
- Cucumber Expressions
- Regular Expressions
- Working with lists and maps
- Working with scenario outlines