nisum®
technologies

Behavior Driven Development using Cucumber JVM

Building Success Together®

table of contents

Introduction

Gherkin basics

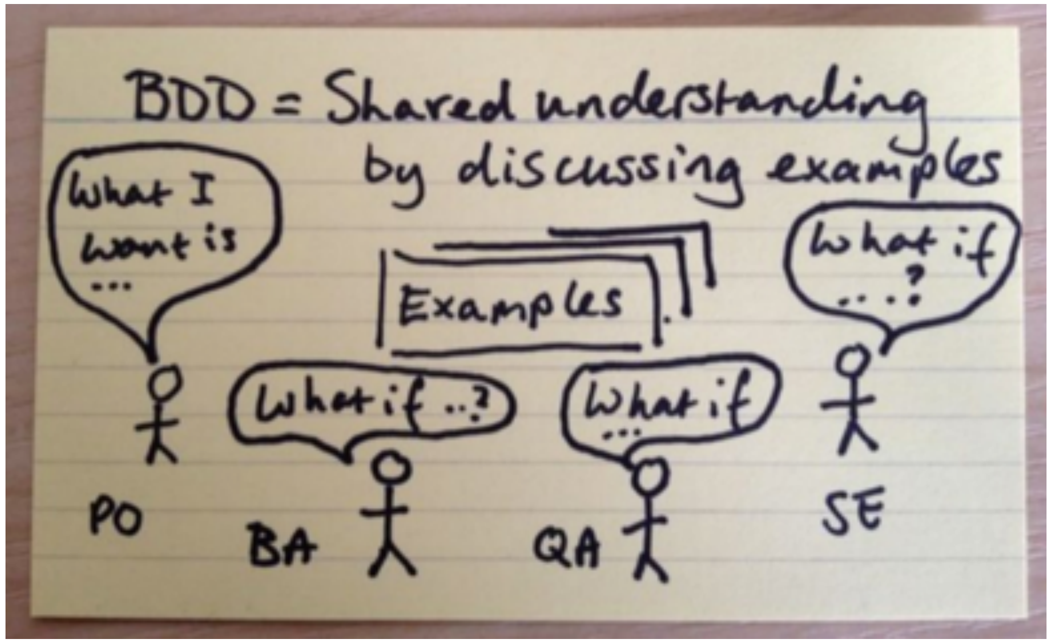
Sample project with Cucumber-JVM setup and covering various Gherkin keywords

References

Introduction

Building Success Together®

- What is BDD?
  - ➢BDD is a second-generation, outside-in, pull-based, multiple-stakeholder, multiple-scale, high-automation, agile methodology. It describes a cycle of interactions with well-defined outputs, resulting in the delivery of working, tested software that matters.
  - ➢Inspires collaboration. TDD evolved
  - ➢Behavior vs Implementation
  - ➢Living Documentation
  - ➢Ubiquitous language



- What is Cucumber?
  - Software starts as an idea and needs to be communicated. Collaboration among team members requires high-quality communication and it is critical for success.
  - These tests are different from unit tests, which are aimed at developers and help them to drive out and check their software designs. It's sometimes said that unit tests ensure **you build the thing right**, while acceptance tests ensure **you build the right thing**.
  - Cucumber helps facilitate the discovery and use of a ubiquitous language within the team, by giving the two sides of the linguistic divide a place where they can meet true value of acceptance tests: as a communication and collaboration tool.

Gherkin Basics

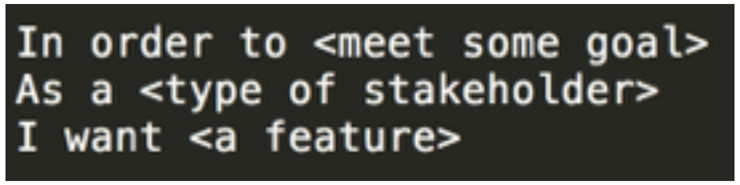
Building Success Together®

- What is Gherkin?
  - ➢Cucumber tests are expressed using a syntax called Gherkin. Gherkin files are plain text and have a .feature extension.
  - ➢Keywords
    - **Feature**
    - **Background**
    - **Scenario**
    - **Given**
    - **When**
    - **Then**
    - **And**
    - **But**
    - *
    - **Scenario Outline**
    - **Examples**
    - **Scenarios**

- Feature
  - ➢This is the feature title
  - ➢This is the description of the feature, which can span multiple lines. You can even include empty lines, like this one:

    In fact, everything until the next Gherkin keyword is included in the description.

    Feature Injection template:

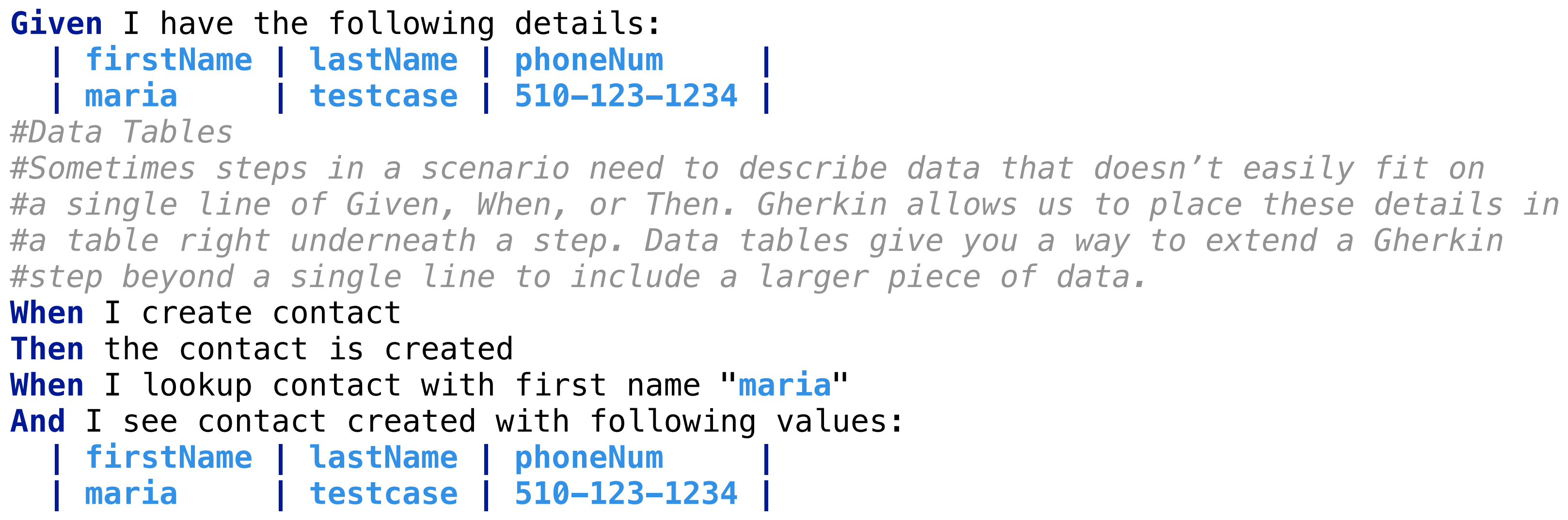


```
In order to <meet some goal>
As a <type of stakeholder>
I want <a feature>
```

```
Feature: Contact Management service
  In order to manage my contacts
  As a user
  I want to add/edit/view my contacts
```

- Background
  - ➢ A background section in a feature file allows you to specify a set of steps that are common to every scenario in the file. Instead of having to repeat those steps over and over for each scenario, you move them up into a Background element.
  - ➢ There are a couple of advantages to doing this:
    - If you ever need to change those steps, you have to change them in only one place.
    - The importance of those steps fades into the background so that when you're reading each individual scenario, you can focus on what is unique and important about that scenario

- Scenario
  - ➢To actually express the behavior we want, each feature contains several scenarios.
  - ➢Each scenario is a single concrete example of how the system should behave in a particular situation. If you add together the behavior defined by all of the scenarios, that's the expected behavior of the feature itself.
  - ➢Each scenario must make sense and be able to be executed independently of any other scenario.



```
Given I have the following details:
  | firstName | lastName | phoneNum     |
  | maria     | testcase | 510-123-1234 |
#Data Tables
#Sometimes steps in a scenario need to describe data that doesn’t easily fit on
#a single line of Given, When, or Then. Gherkin allows us to place these details in
#a table right underneath a step. Data tables give you a way to extend a Gherkin
#step beyond a single line to include a larger piece of data.
When I create contact
Then the contact is created
When I lookup contact with first name "maria"
And I see contact created with following values:
  | firstName | lastName | phoneNum     |
  | maria     | testcase | 510-123-1234 |
```

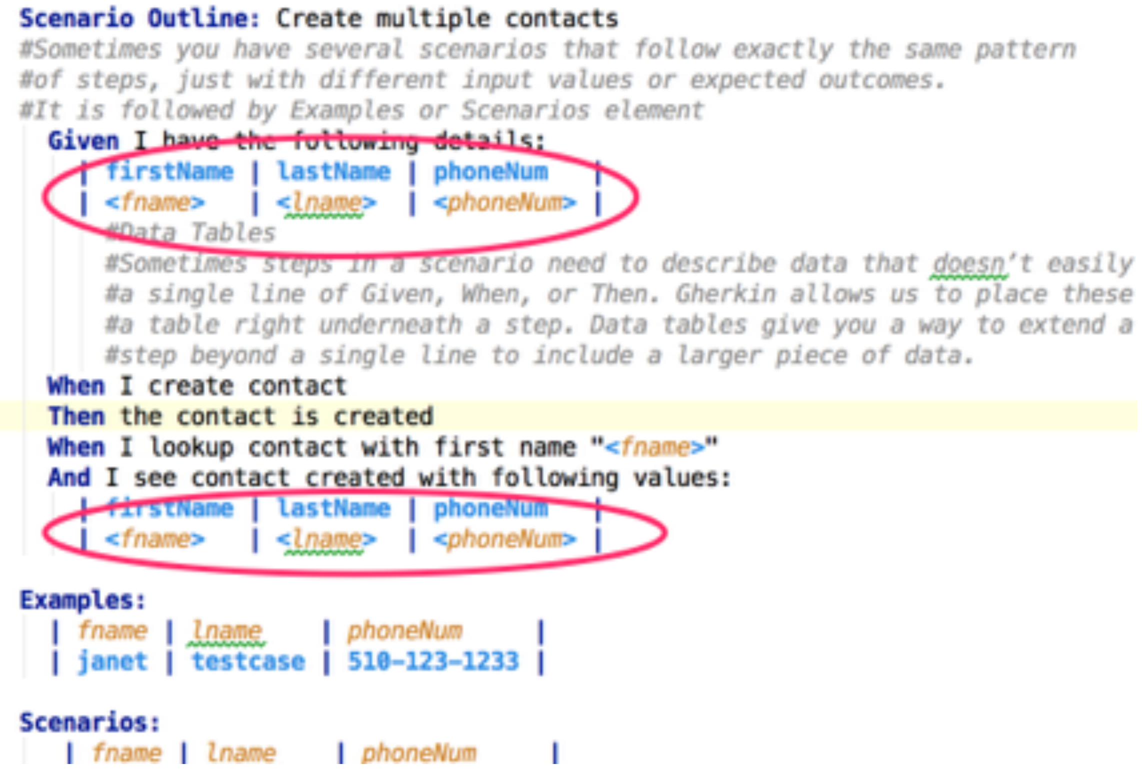
Scenario Outline

- Sometimes you have several scenarios that follow exactly the same pattern of steps, just with different input values or expected outcomes.
- It is followed by Examples or Scenarios element

```
Scenario Outline: Create multiple contacts
 Given I have the following details:
    | firstName | lastName | phoneNum   |
    | <fname>   | <lname>  | <phoneNum> |
      #Data Tables
      #Sometimes steps in a scenario need to describe data that doesn’t easily fit on
      #a single line of Given, When, or Then. Gherkin allows us to place these details in
      #a table right underneath a step. Data tables give you a way to extend a Gherkin
      #step beyond a single line to include a larger piece of data.
  When I create contact
  Then the contact is created
  When I lookup contact with first name "<fname>"
  And I see contact created with following values:
    | firstName | lastName | phoneNum   |
    | <fname>   | <lname>  | <phoneNum> |

Examples:
  | fname | lname    | phoneNum     |
  | janet | testcase | 510-123-1233 |

Scenarios:
   | fname | lname    | phoneNum     |
   | james | testcase | 510-123-1233 |
```

Data Tables

- ➢Sometimes steps in a scenario need to describe data that doesn't easily fit on a single line of Given, When, or Then.
- ➢Gherkin allows us to place these details in a table right underneath a step. Data tables give you a way to extend a Gherkin step beyond a single line to include a larger piece of data.



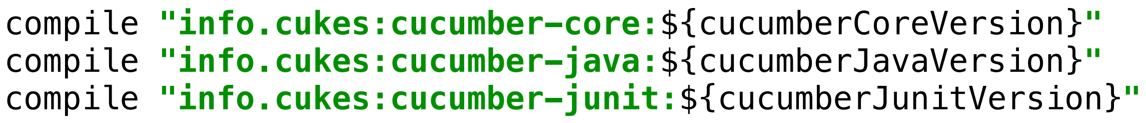
```
Scenario Outline: Create multiple contacts
#Sometimes you have several scenarios that follow exactly the same pattern
#of steps, just with different input values or expected outcomes.
#It is followed by Examples or Scenarios element
  Given I have the following details:
    | firstName | lastName | phoneNum   |
    | <fname>   | <lname>  | <phoneNum> |
    #Data Tables
    #Sometimes steps in a scenario need to describe data that doesn't easily
    #a single line of Given, When, or Then. Gherkin allows us to place these
    #a table right underneath a step. Data tables give you a way to extend a
    #step beyond a single line to include a larger piece of data.
  When I create contact
  Then the contact is created
  When I lookup contact with first name "<fname>"
  And I see contact created with following values:
    | firstName | lastName | phoneNum   |
    | <fname>   | <lname>  | <phoneNum> |

Examples:
  | fname | lname    | phoneNum     |
  | janet | testcase | 510-123-1233 |

Scenarios:
   | fname | lname    | phoneNum     |
```

Sample project with Cucumber JVM setup covering various Gherkin keywords

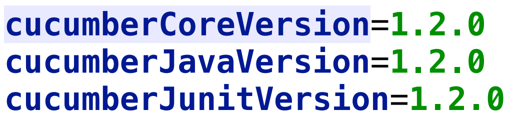
Building Success Together®

- Sample project with Cucumber JVM setup
  - ➢Sample project can be accessed at https://github.com/mmuduganti/contact-app
  - ➢It is gradle project and can be imported into either Eclipse or Intellij
  - ➢Please follow the READEME instruction to execute the project locally.

- Cucumber JVM dependencies

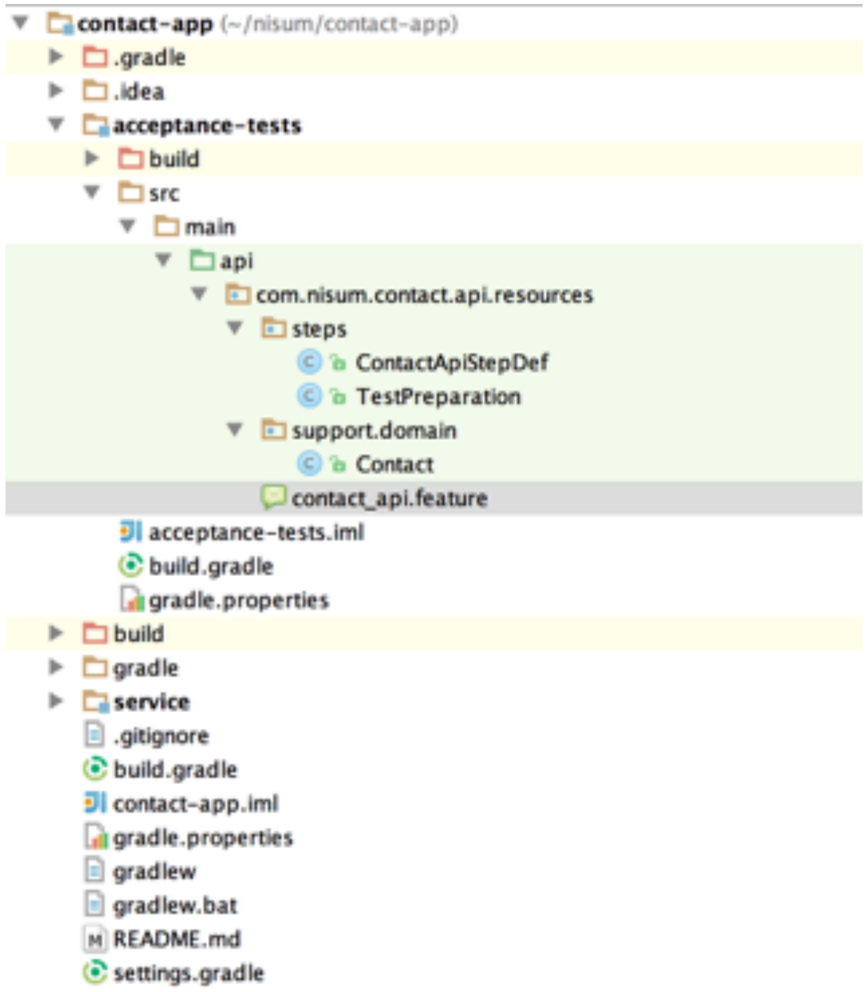


```
compile "info.cukes:cucumber-core:${cucumberCoreVersion}"
compile "info.cukes:cucumber-java:${cucumberJavaVersion}"
compile "info.cukes:cucumber-junit:${cucumberJunitVersion}"
```



```
cucumberCoreVersion=1.2.0
cucumberJavaVersion=1.2.0
cucumberJunitVersion=1.2.0
```

- Project Structure



```
▼ contact-app (~/nisum/contact-app)
  ▶ .gradle
  ▶ .idea
  ▼ acceptance-tests
    ▶ build
    ▼ src
      ▼ main
        ▼ api
          ▼ com.nisum.contact.api.resources
            ▼ steps
                ContactApiStepDef
                TestPreparation
            ▼ support.domain
                Contact
            contact_api.feature
    acceptance-tests.iml
    build.gradle
    gradle.properties
  ▶ build
  ▶ gradle
  ▶ service
    .gitignore
    build.gradle
    contact-app.iml
    gradle.properties
    gradlew
    gradlew.bat
    README.md
    settings.gradle
```

Reference

Building Success Together®

Thanks! Please reach out if you have questions.

www.nisum.com
500 S. Kraemer Blvd, Suite 301
Brea, CA 92821

Building Success Together®