BDD in Action

Building Software Right
Building the Right Software

Miyamoto Musashi
Japan
1584 – 1645

http://www.wakaleo.com

john.smart@wakaleo.com

wakaleo

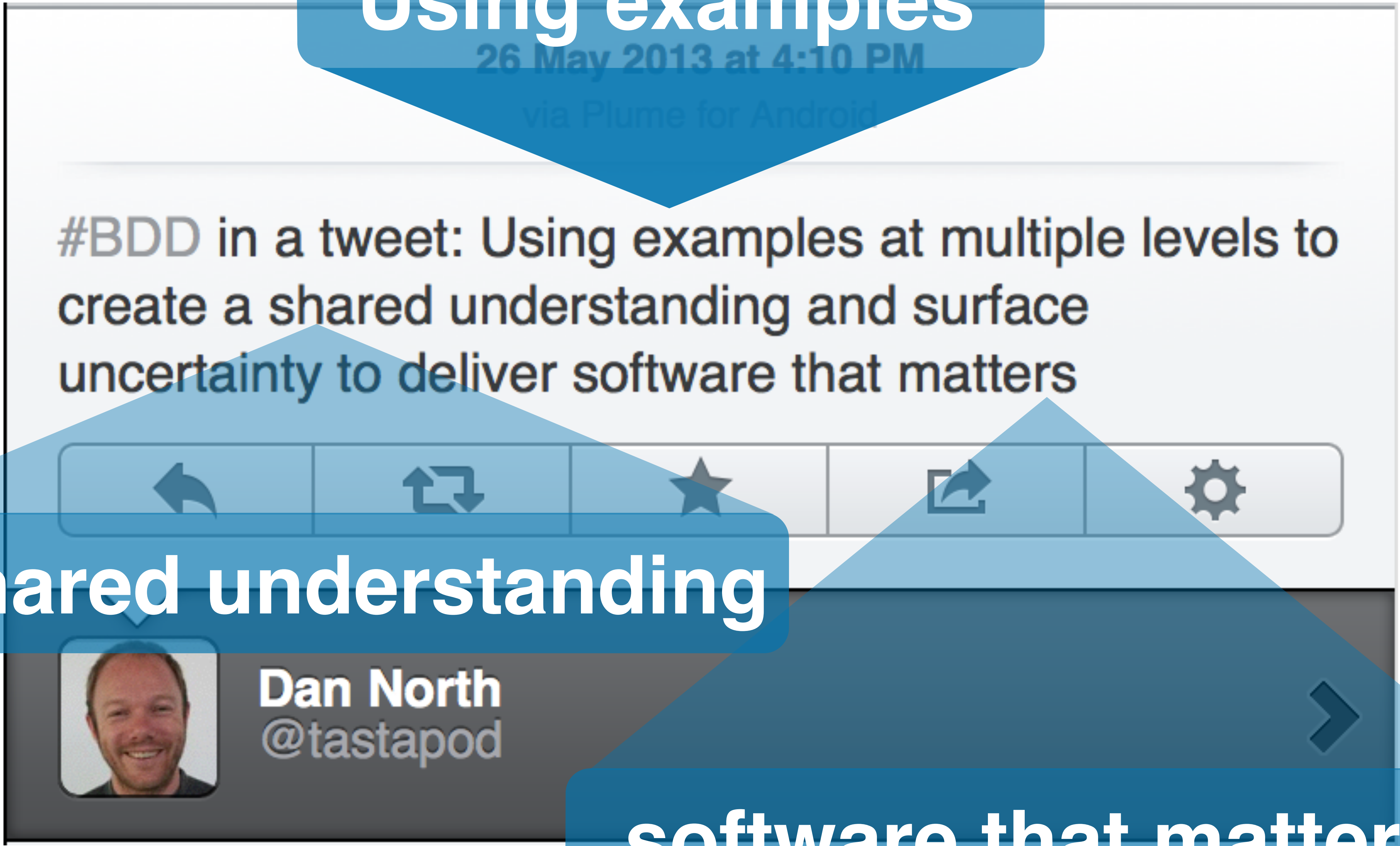
- ☑ Consultant
- ☑ Trainer
- ☑ Mentor
- ☑ Author
- ☑ Speaker
- ☑ Coder

John Ferguson Smart

WAKALEO CONSULTING
OPTIMIZING YOUR SOFTWARE DEVELOPMENT PROCESS

- ✓ What is BDD
- ✓ A typical BDD workflow
- ✓ What tools should I use?
- ✓ Effective BDD automation
- ✓ BDD Gotchas

So what is this BDD thing?



26 May 2013 at 4:10 PM
via Plume for Android

#BDD in a tweet: Using examples at multiple levels to create a shared understanding and surface uncertainty to deliver software that matters

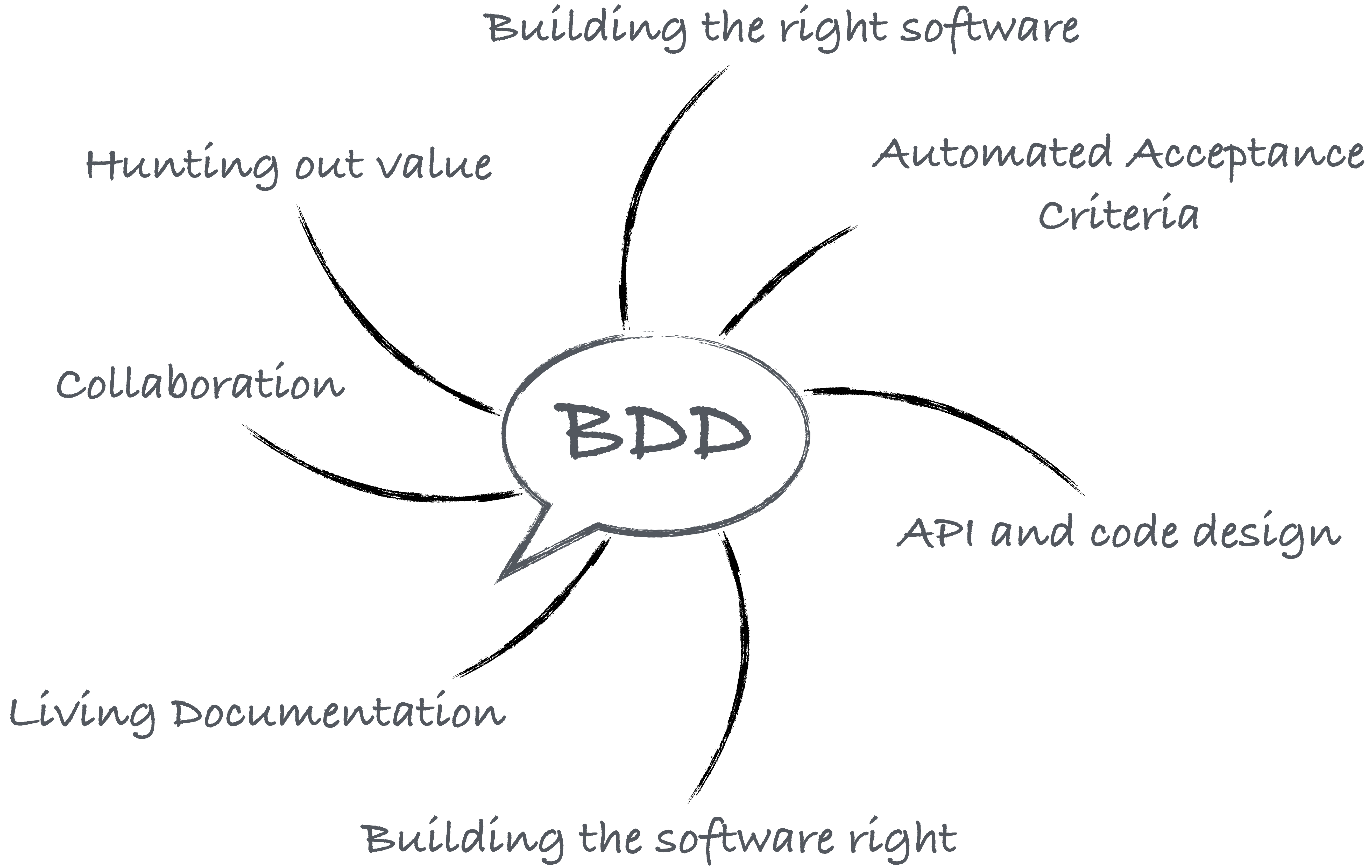
Dan North
@tastapod

Using examples

a shared understanding

software that matters

So what is this BDD thing?



Building the right software
Hunting out value
Automated Acceptance Criteria
Collaboration
BDD
API and code design
Living Documentation
Building the software right

BDD in a nutshell

1 The business owner tells the business analyst what he wants

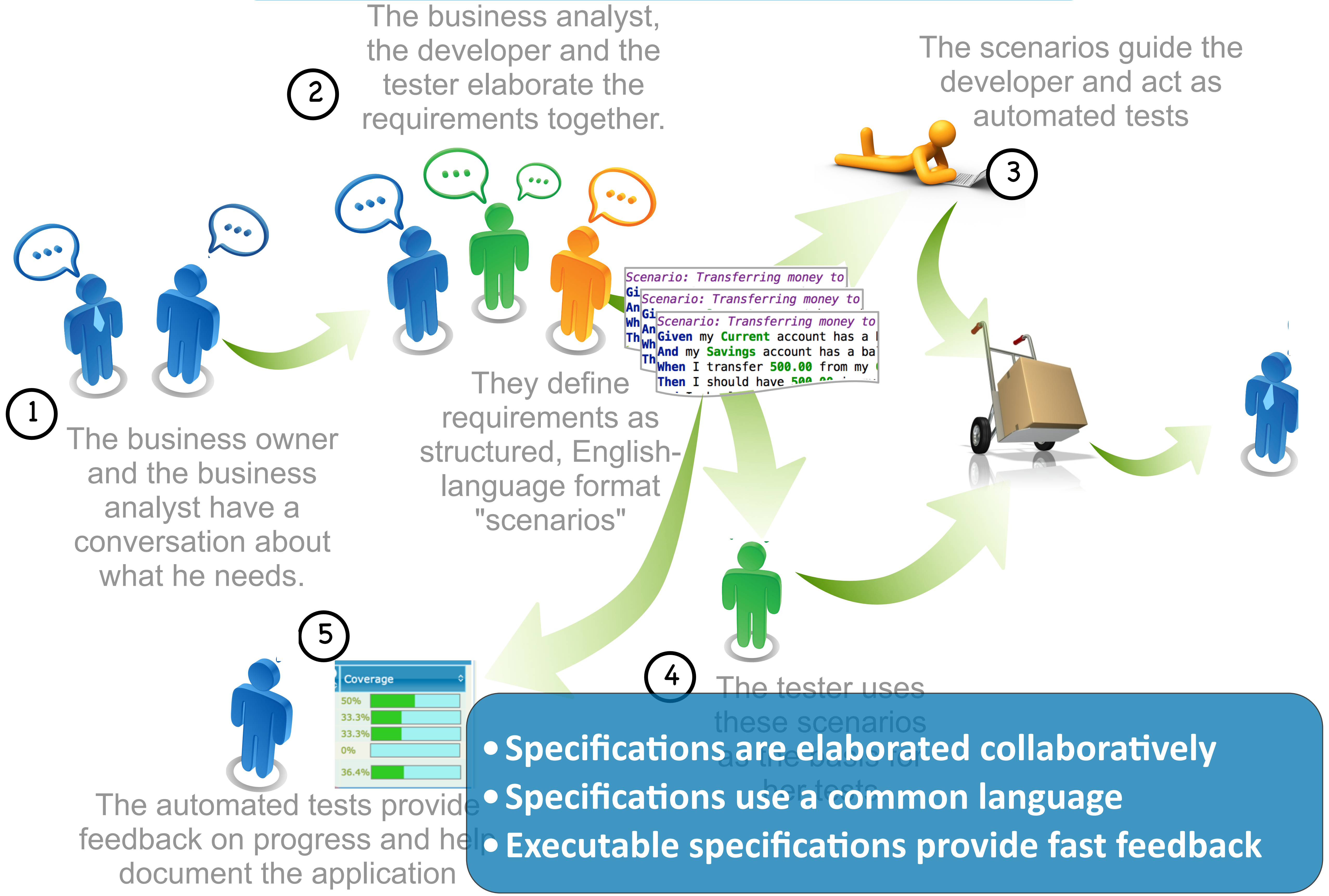
A traditional development process

BDD in a nutshell

① The business owner and the business analyst have a conversation about what he needs.

A BDD development process

BDD in a nutshell


1
The business owner and the business analyst have a conversation about what he needs.
2
The business analyst, the developer and the tester elaborate the requirements together.
They define requirements as structured, English-language format "scenarios"
Scenario: Transferring money to
Given my Current account has a
And my Savings account has a ba
When I transfer 500.00 from my
Then I should have 500.00
3
The scenarios guide the developer and act as automated tests
4
The tester uses these scenarios as the basis for her tests
5
Coverage
50%
33.3%
33.3%
0%
36.4%
The automated tests provide feedback on progress and help document the application

- Specifications are elaborated collaboratively
- Specifications use a common language
- Executable specifications provide fast feedback

A typical BDD workflow

Frequent Flyer Application
Goal: Encourage travellers to fly with Flying High airlines more often by allowing them to cumulate Frequent Flyer points that they can spend on cheaper flights.

Goals

Earning points from flights

Earning points from spending with partners

Viewing points earned

Spending points on bookings

Capabilities

Viewing current points balance

View points needed to achieve the next status level

Calculating points needed for a given destination

Features

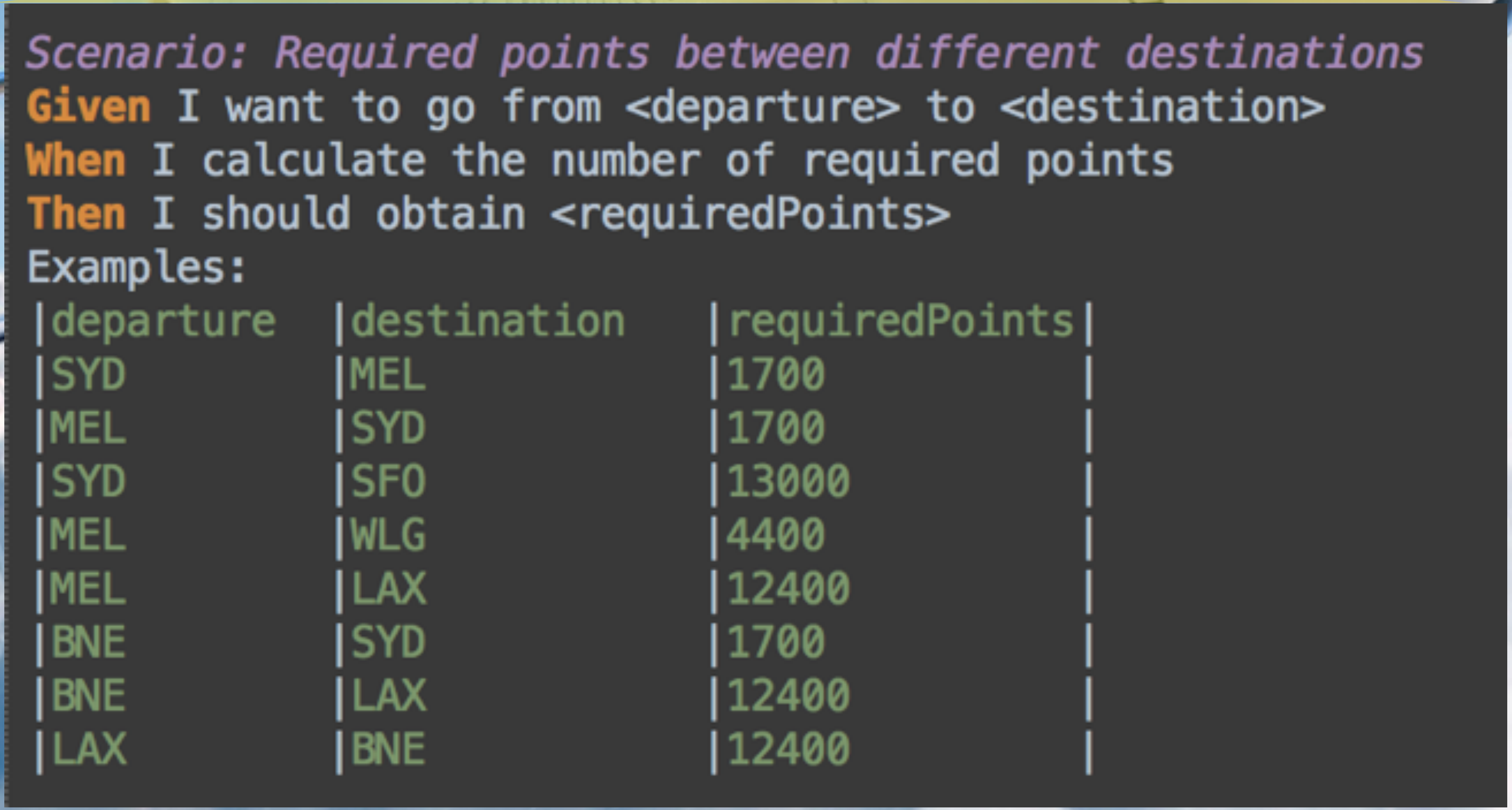
A typical BDD workflow

Calculating points needed for a given destination
As a traveller
I want to know how many points I need to go to a given destination
So that I can plan my next trip with Flying High Airlines

Feature

Acceptance Criteria
- Need 2 points per km
- Members can calculate points needed on their account home page

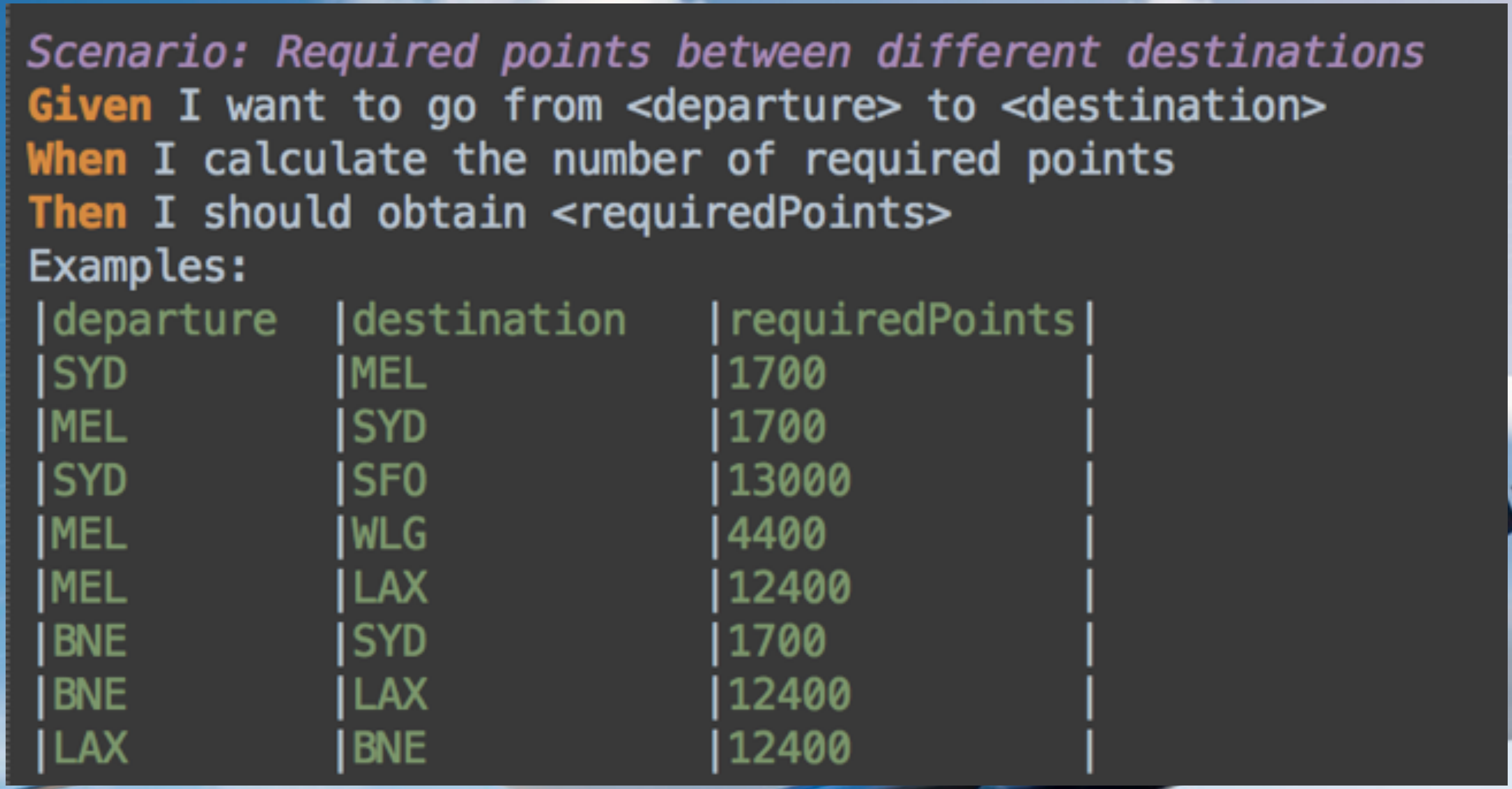
Acceptance Criteria



```
Scenario: Required points between different destinations
Given I want to go from <departure> to <destination>
When I calculate the number of required points
Then I should obtain <requiredPoints>
Examples:
|departure   |destination     |requiredPoints|
|SYD         |MEL             |1700          |
|MEL         |SYD             |1700          |
|SYD         |SFO             |13000         |
|MEL         |WLG             |4400          |
|MEL         |LAX             |12400         |
|BNE         |SYD             |1700          |
|BNE         |LAX             |12400         |
|LAX         |BNE             |12400         |
```

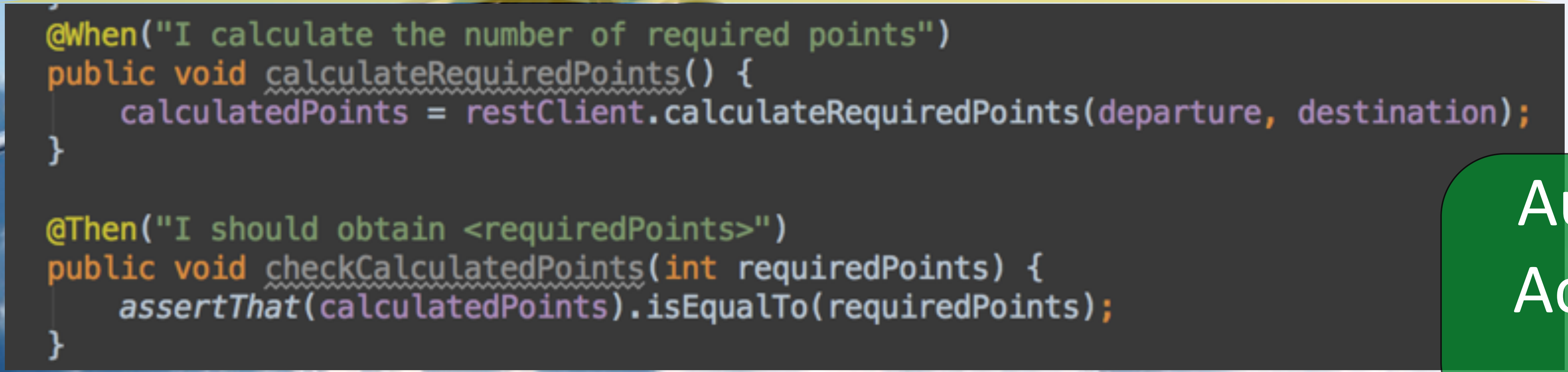
Automated Acceptance Criteria

A typical BDD workflow



```
Scenario: Required points between different destinations
Given I want to go from <departure> to <destination>
When I calculate the number of required points
Then I should obtain <requiredPoints>
Examples:
|departure   |destination    |requiredPoints|
|SYD         |MEL            |1700          |
|MEL         |SYD            |1700          |
|SYD         |SFO            |13000         |
|MEL         |WLG            |4400          |
|MEL         |LAX            |12400         |
|BNE         |SYD            |1700          |
|BNE         |LAX            |12400         |
|LAX         |BNE            |12400         |
```

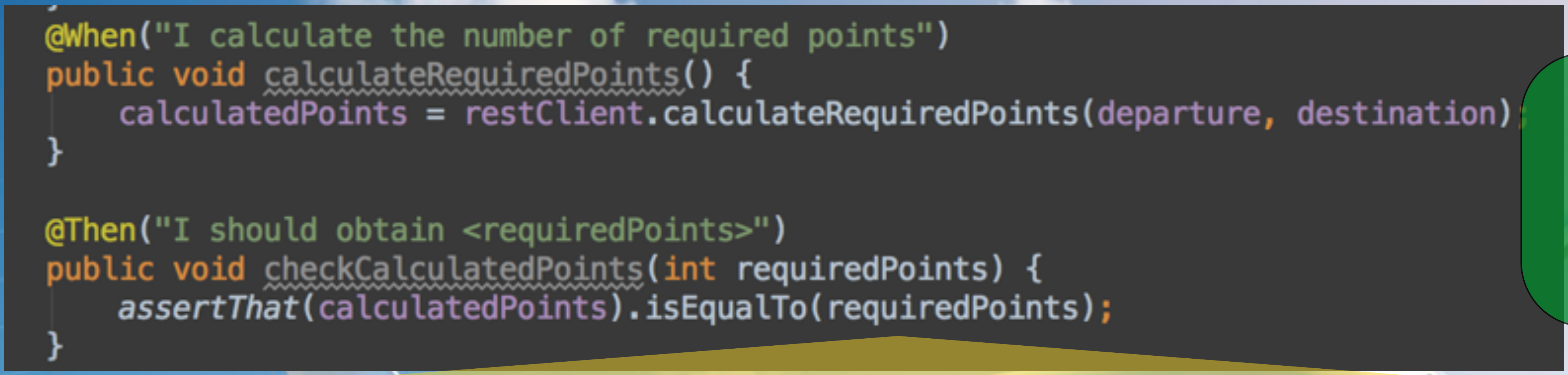
Automated Acceptance Criteria



```
@When("I calculate the number of required points")
public void calculateRequiredPoints() {
    calculatedPoints = restClient.calculateRequiredPoints(departure, destination);
}

@Then("I should obtain <requiredPoints>")
public void checkCalculatedPoints(int requiredPoints) {
    assertThat(calculatedPoints).isEqualTo(requiredPoints);
}
```

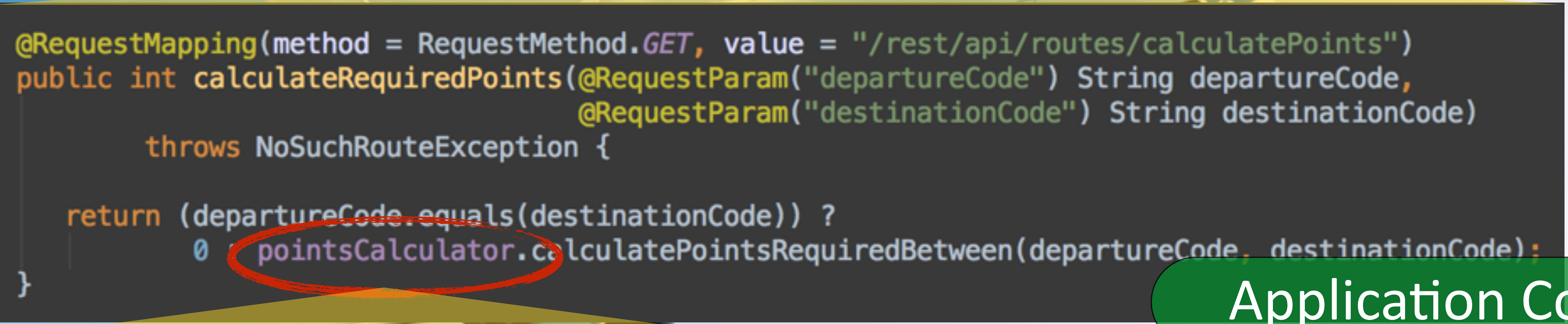
Automated Acceptance Tests

A typical BDD workflow



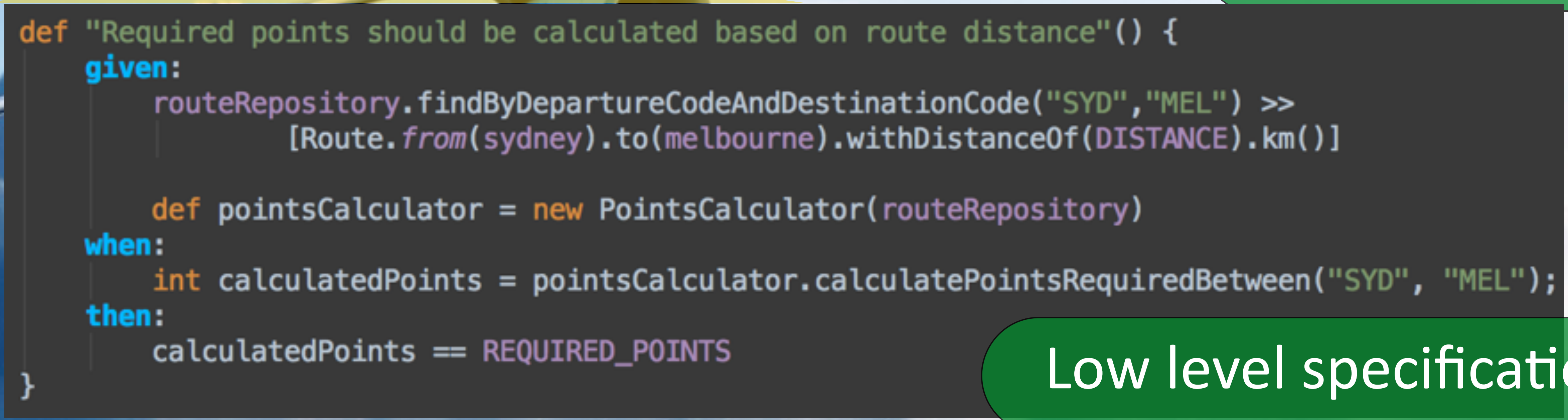
```
@When("I calculate the number of required points")
public void calculateRequiredPoints() {
    calculatedPoints = restClient.calculateRequiredPoints(departure, destination);
}

@Then("I should obtain <requiredPoints>")
public void checkCalculatedPoints(int requiredPoints) {
    assertThat(calculatedPoints).isEqualTo(requiredPoints);
}
```

Automated Acceptance Tests



```
@RequestMapping(method = RequestMethod.GET, value = "/rest/api/routes/calculatePoints")
public int calculateRequiredPoints(@RequestParam("departureCode") String departureCode,
                                  @RequestParam("destinationCode") String destinationCode)
        throws NoSuchRouteException {

    return (departureCode.equals(destinationCode)) ?
            0 : pointsCalculator.calculatePointsRequiredBetween(departureCode, destinationCode);
}
```

Application Code

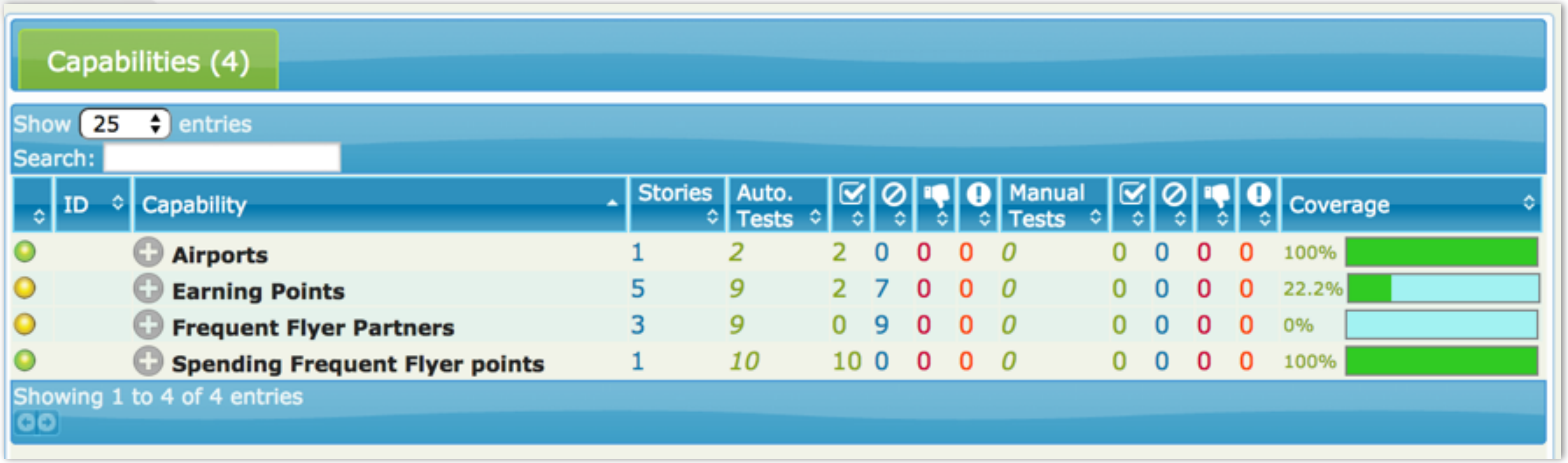


```
def "Required points should be calculated based on route distance"() {
    given:
        routeRepository.findByDepartureCodeAndDestinationCode("SYD","MEL") >>
                [Route.from(sydney).to(melbourne).withDistanceOf(DISTANCE).km()]

        def pointsCalculator = new PointsCalculator(routeRepository)
    when:
        int calculatedPoints = pointsCalculator.calculatePointsRequiredBetween("SYD", "MEL");
    then:
        calculatedPoints == REQUIRED_POINTS
}
```

Low level specifications

A typical BDD workflow

"But what are the deliverables?



Capabilities (4)

Show 25 entries

Search:

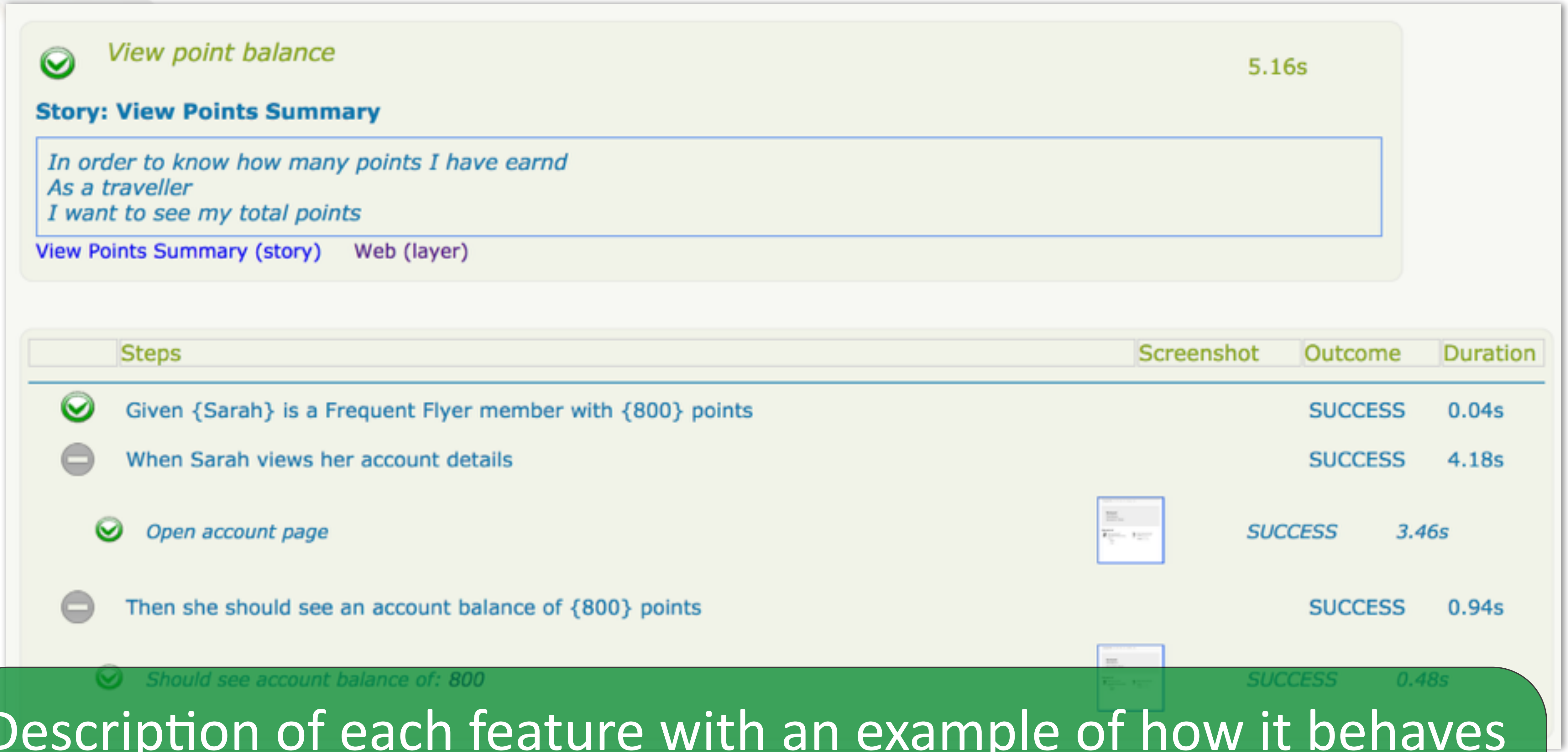
	ID	Capability	Stories	Auto. Tests	✓	⊘	👎	!	Manual Tests	✓	⊘	👎	!	Coverage
🟢		⊕ **Airports**	1	*2*	2	0	0	0	*0*	0	0	0	0	100%
🟡		⊕ **Earning Points**	5	*9*	2	7	0	0	*0*	0	0	0	0	22.2%
🟡		⊕ **Frequent Flyer Partners**	3	*9*	0	9	0	0	*0*	0	0	0	0	0%
🟢		⊕ **Spending Frequent Flyer points**	1	*10*	10	0	0	0	*0*	0	0	0	0	100%

Showing 1 to 4 of 4 entries

Meaningful feedback on what requirements have (and have not) been delivered

An incremental approach

"But what are the deliverables?



View point balance 5.16s

Story: View Points Summary

In order to know how many points I have earnd
As a traveller
I want to see my total points

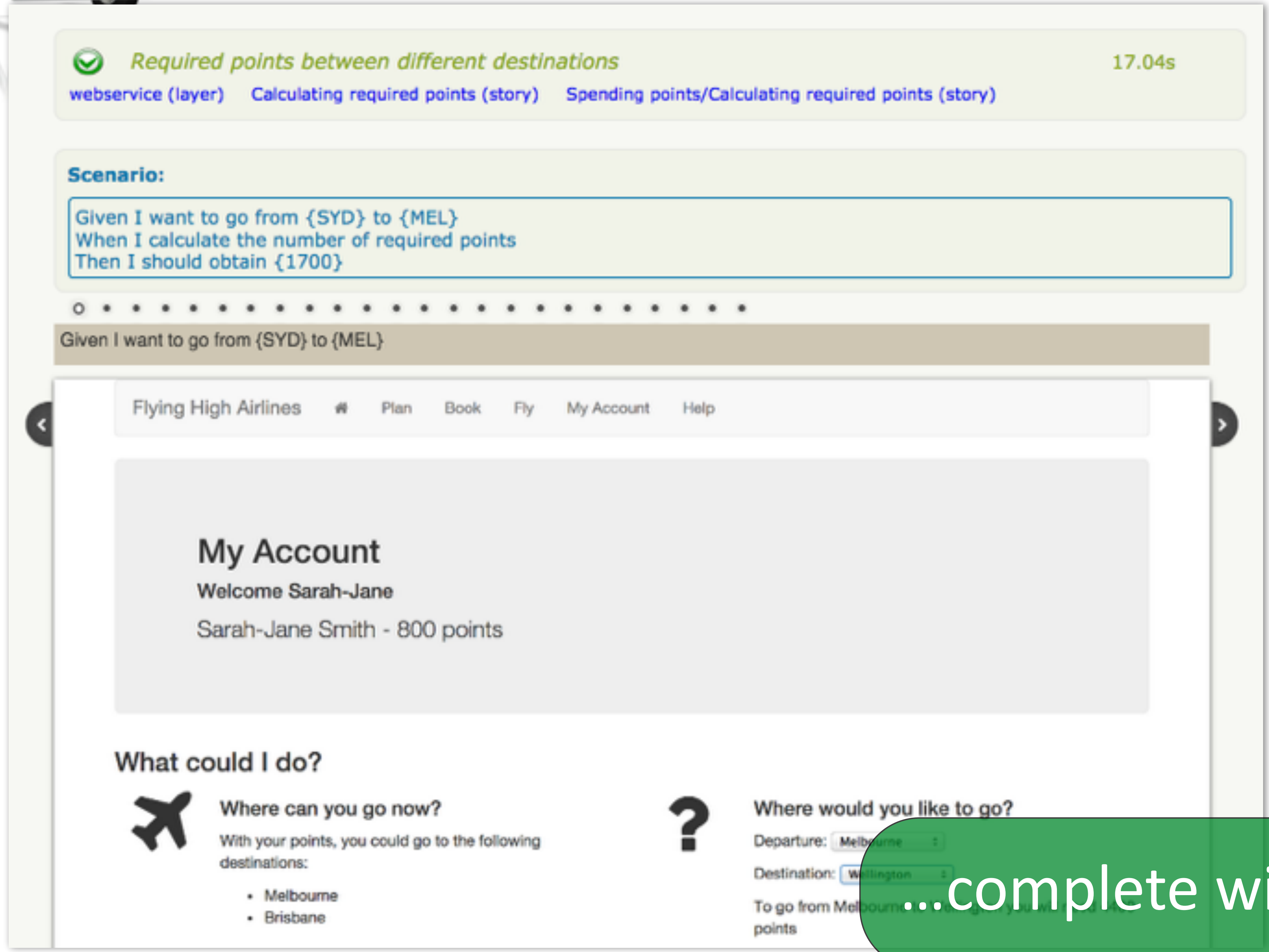
View Points Summary (story) Web (layer)

	Steps	Screenshot	Outcome	Duration
✓	Given {Sarah} is a Frequent Flyer member with {800} points		SUCCESS	0.04s
⊖	When Sarah views her account details		SUCCESS	4.18s
✓	*Open account page*	▭	*SUCCESS*	*3.46s*
⊖	Then she should see an account balance of {800} points		SUCCESS	0.94s
✓	*Should see account balance of: 800*	▭	*SUCCESS*	*0.48s*

Description of each feature with an example of how it behaves

An incremental approach

"But what are the deliverables?



✓ *Required points between different destinations* 17.04s

webservice (layer) Calculating required points (story) Spending points/Calculating required points (story)

Scenario:

```
Given I want to go from {SYD} to {MEL}
When I calculate the number of required points
Then I should obtain {1700}
```

Given I want to go from {SYD} to {MEL}

My Account

Welcome Sarah-Jane

Sarah-Jane Smith - 800 points

What could I do?

Where can you go now?

With your points, you could go to the following destinations:

- Melbourne
- Brisbane

Where would you like to go?

Departure: Melbourne

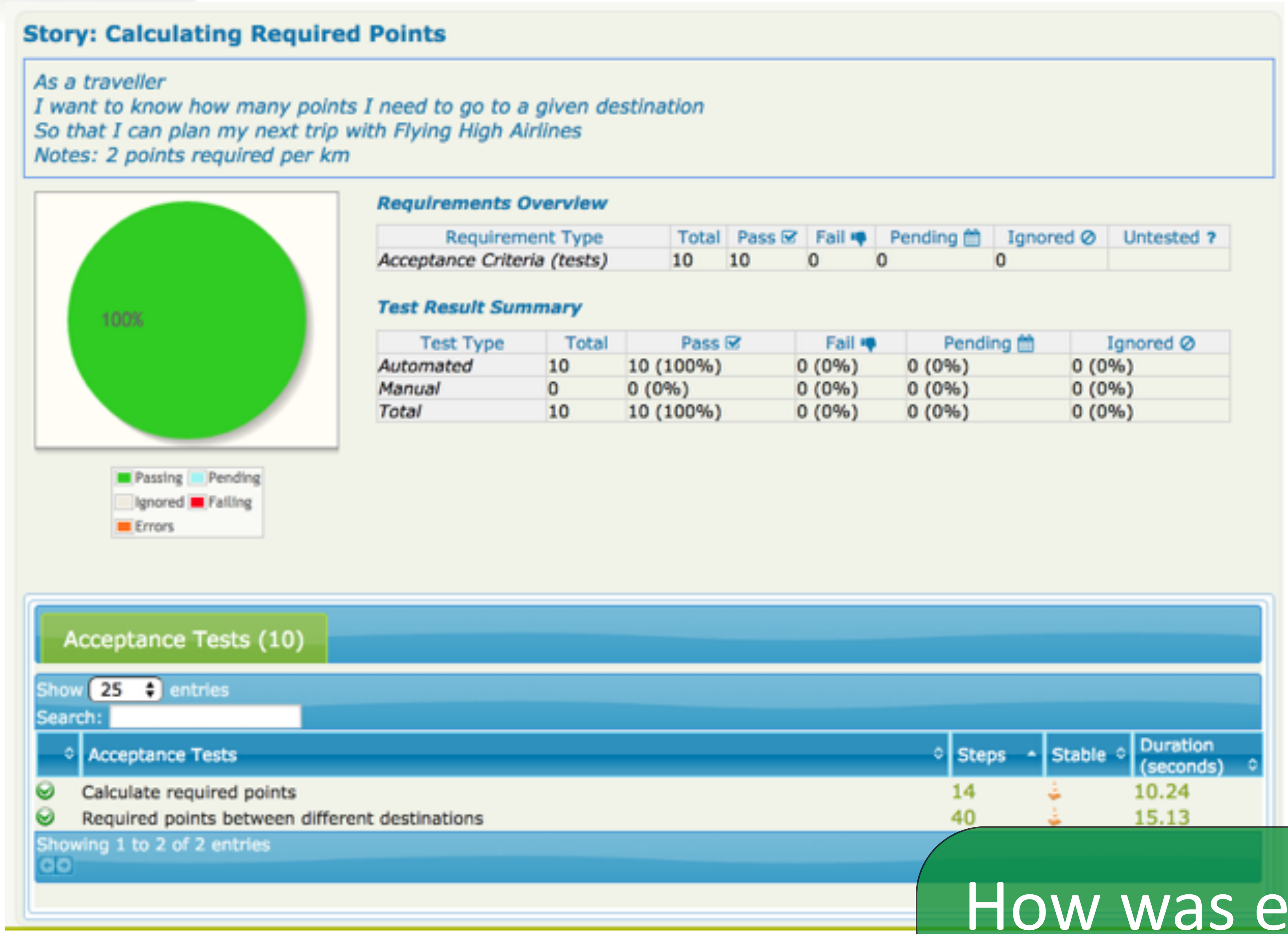
Destination: Wellington

To go from Melbourne ... points

...complete with illustrations

An incremental approach

"But what are the deliverables?



Story: Calculating Required Points

As a traveller
I want to know how many points I need to go to a given destination
So that I can plan my next trip with Flying High Airlines
Notes: 2 points required per km

Requirements Overview

Requirement Type	Total	Pass	Fail	Pending	Ignored	Untested ?
Acceptance Criteria (tests)	10	10	0	0	0	

Test Result Summary

Test Type	Total	Pass	Fail	Pending	Ignored
Automated	10	10 (100%)	0 (0%)	0 (0%)	0 (0%)
Manual	0	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Total	10	10 (100%)	0 (0%)	0 (0%)	0 (0%)

Acceptance Tests (10)

Show 25 entries
Search:

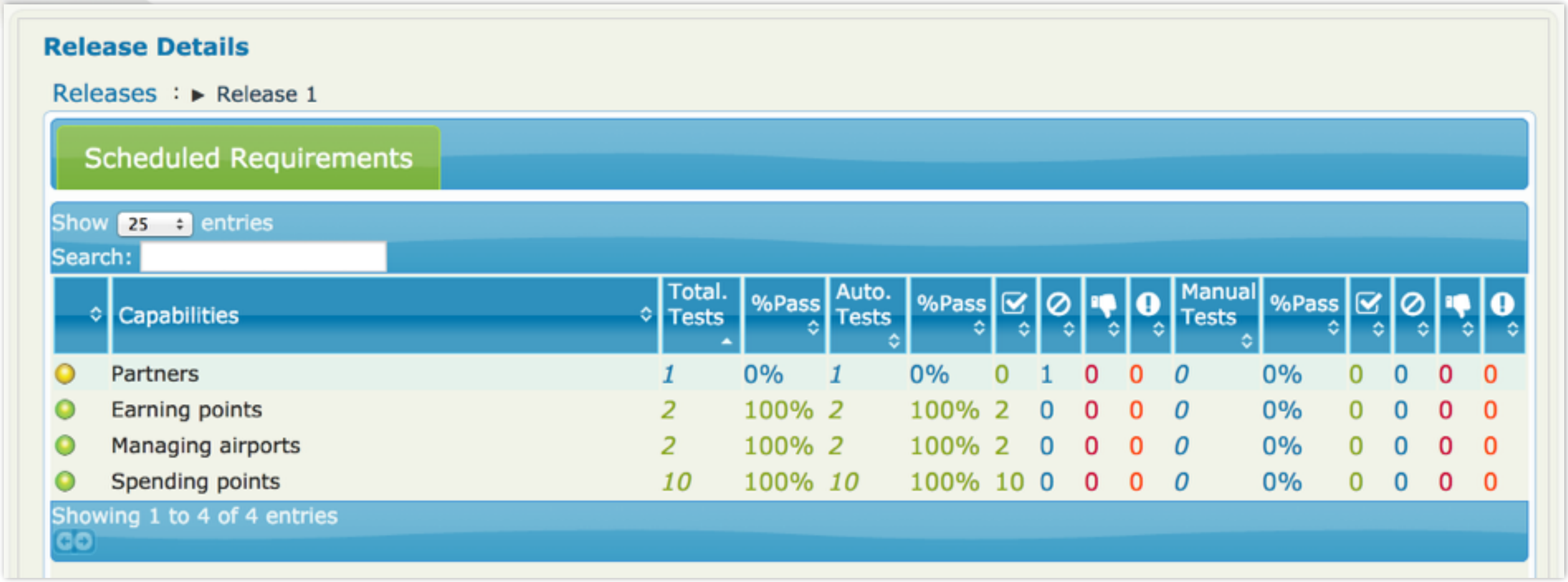
	Acceptance Tests	Steps	Stable	Duration (seconds)
✓	Calculate required points	14		10.24
✓	Required points between different destinations	40		15.13

Showing 1 to 2 of 2 entries

How was each feature tested?

An incremental approach

"But what are the deliverables?



Release Details

Releases : ▶ Release 1

Scheduled Requirements

Show 25 entries

Search:

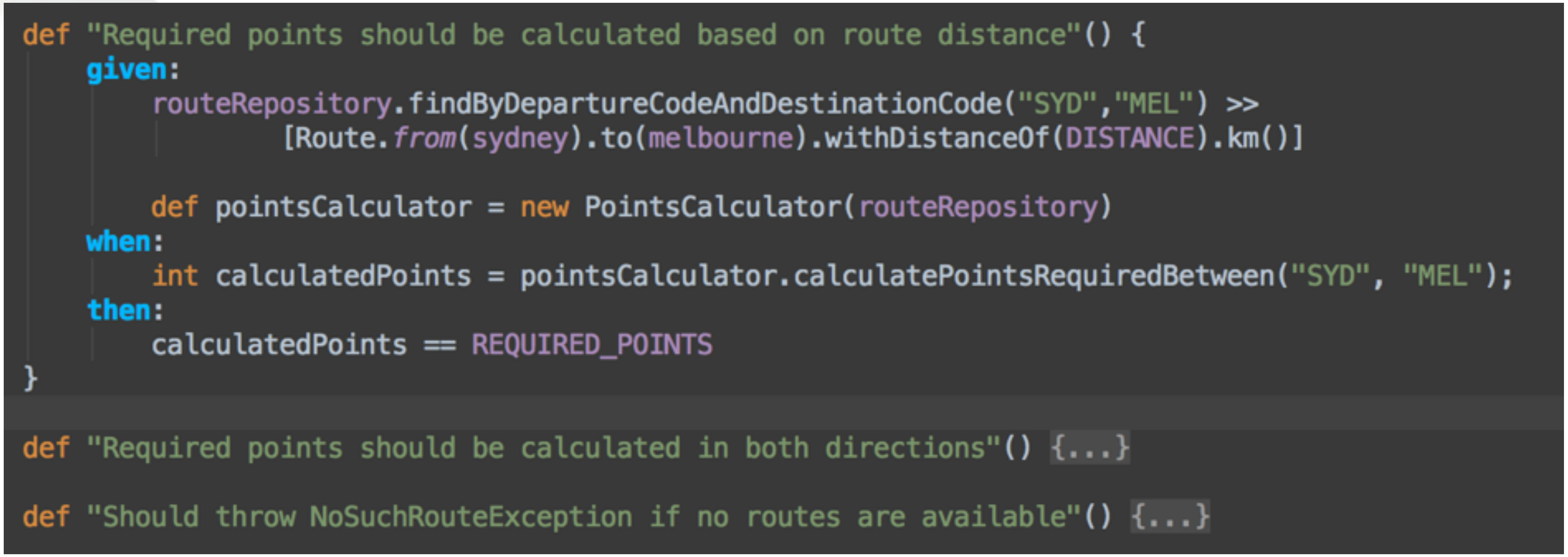
	Capabilities	Total. Tests	%Pass	Auto. Tests	%Pass	☑	⊘	👎	❗	Manual Tests	%Pass	☑	⊘	👎	❗
●	Partners	*1*	0%	*1*	0%	0	1	0	0	*0*	0%	0	0	0	0
●	Earning points	*2*	100%	*2*	100%	2	0	0	0	*0*	0%	0	0	0	0
●	Managing airports	*2*	100%	*2*	100%	2	0	0	0	*0*	0%	0	0	0	0
●	Spending points	*10*	100%	*10*	100%	10	0	0	0	*0*	0%	0	0	0	0

Showing 1 to 4 of 4 entries

Documentation about what you plan to deliver in each release

An incremental approach

"But what are the deliverables?

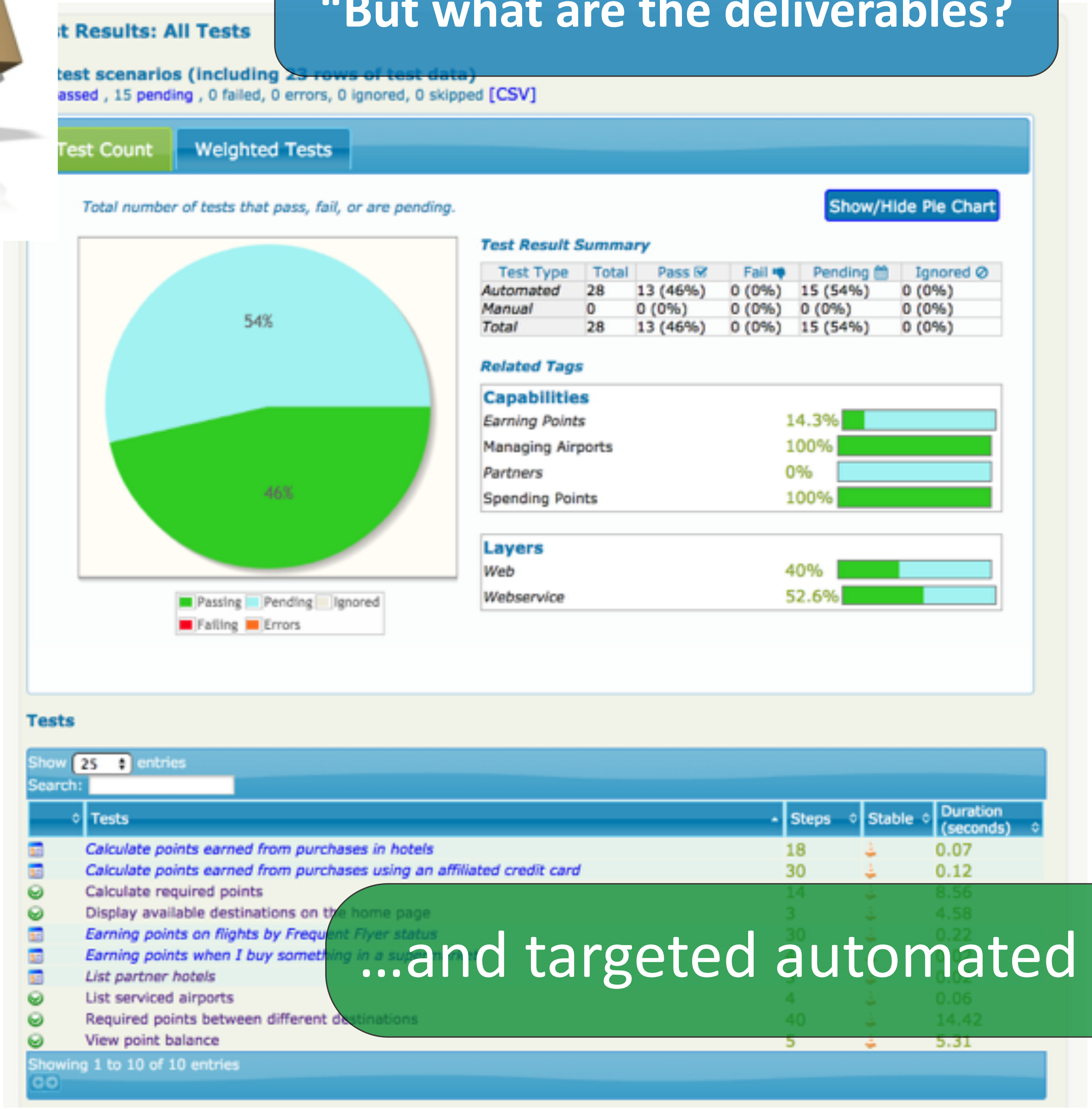


```
def "Required points should be calculated based on route distance"() {
    given:
        routeRepository.findByDepartureCodeAndDestinationCode("SYD","MEL") >>
            [Route.from(sydney).to(melbourne).withDistanceOf(DISTANCE).km()]

        def pointsCalculator = new PointsCalculator(routeRepository)
    when:
        int calculatedPoints = pointsCalculator.calculatePointsRequiredBetween("SYD", "MEL");
    then:
        calculatedPoints == REQUIRED_POINTS
}

def "Required points should be calculated in both directions"() {...}

def "Should throw NoSuchRouteException if no routes are available"() {...}
```

Useful low-level technical documentation with little overhead

An incremental approach

"But what are the deliverables?



t Results: All Tests

test scenarios (including 23 rows of test data)
assed , 15 pending , 0 failed, 0 errors, 0 ignored, 0 skipped [CSV]

Total number of tests that pass, fail, or are pending.

Test Result Summary

Test Type	Total	Pass	Fail	Pending	Ignored
Automated	28	13 (46%)	0 (0%)	15 (54%)	0 (0%)
Manual	0	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Total	28	13 (46%)	0 (0%)	15 (54%)	0 (0%)

Related Tags

Capabilities

Earning Points	14.3%
Managing Airports	100%
Partners	0%
Spending Points	100%

Layers

Web	40%
Webservice	52.6%

Tests

Show 25 entries
Search:

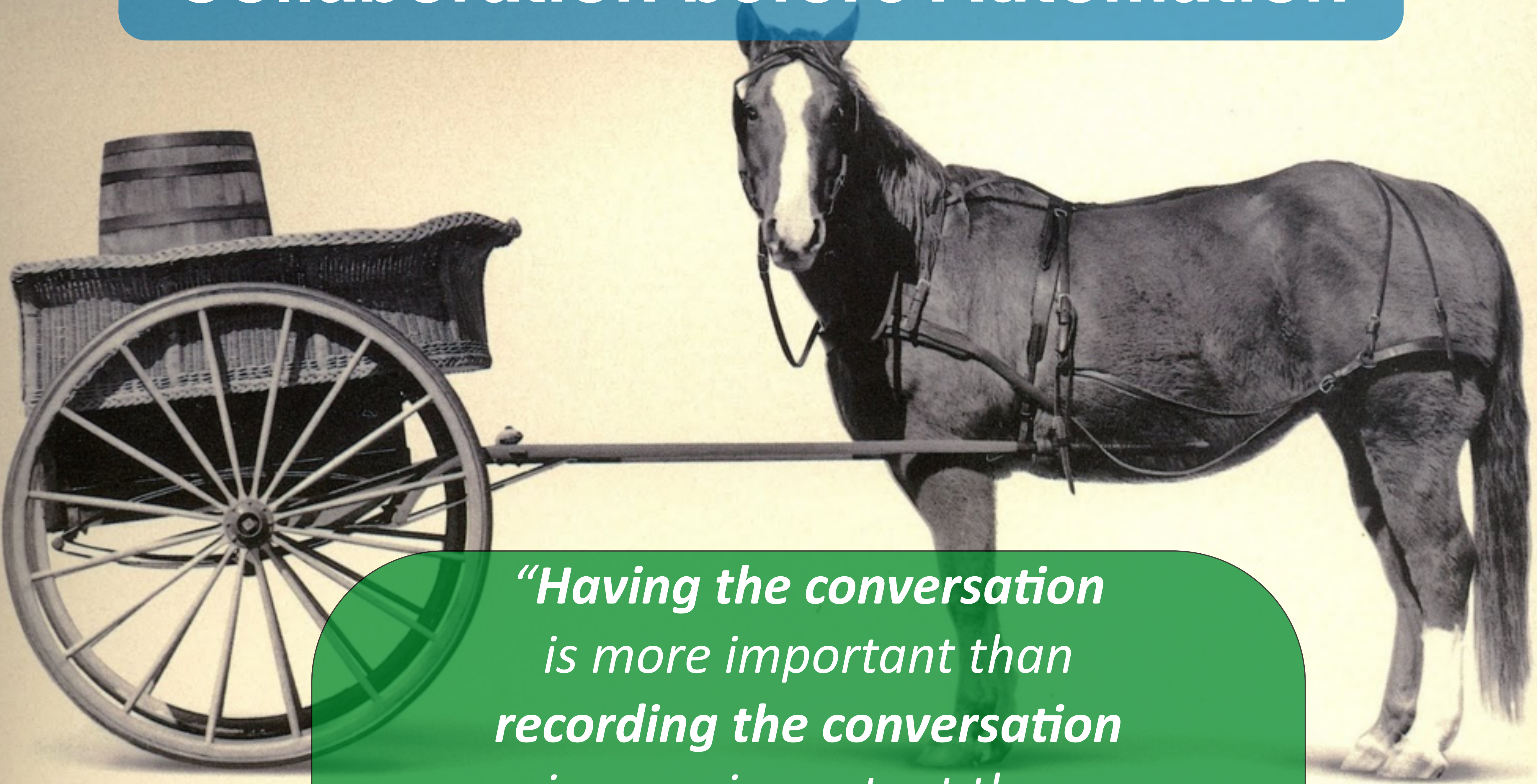
	Tests	Steps	Stable	Duration (seconds)
	Calculate points earned from purchases in hotels	18		0.07
	Calculate points earned from purchases using an affiliated credit card	30		0.12
✓	Calculate required points	14		8.56
✓	Display available destinations on the home page	3		4.58
	Earning points on flights by Frequent Flyer status	30		0.22
	Earning points when I buy something in a supermarket	[illegible]		[illegible]
	List partner hotels	[illegible]		[illegible]
✓	List serviced airports	4		0.06
✓	Required points between different destinations	40		14.42
✓	View point balance	5		5.31

Showing 1 to 10 of 10 entries

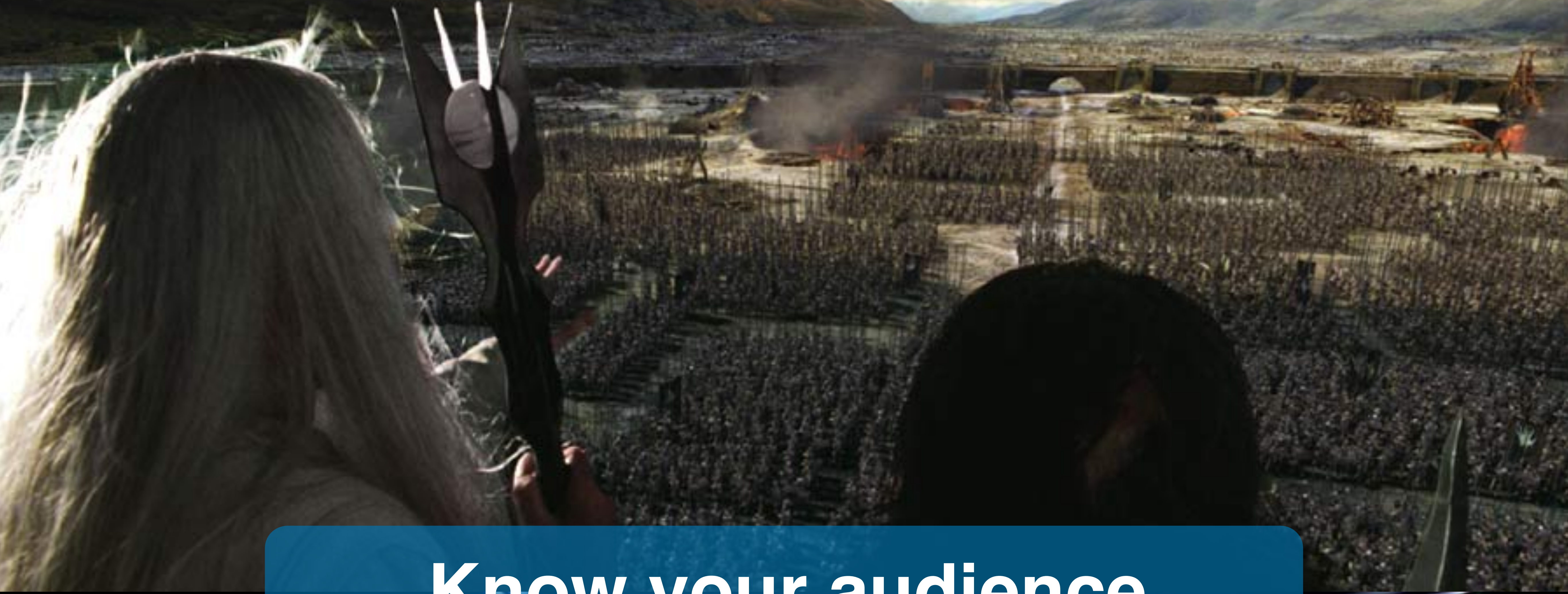
…and targeted automated regression tests

What tool should I use?

Collaboration before Automation



"Having the conversation
is more important than
recording the conversation
is more important than
automating the conversation"
- Liz Keogh



Know your audience

High-level BDD

- ✓ Reporting for non-developers as well as developers
- ✓ Communicate about features that business owners will understand and find meaningful
- ✓ Higher automation and maintenance costs

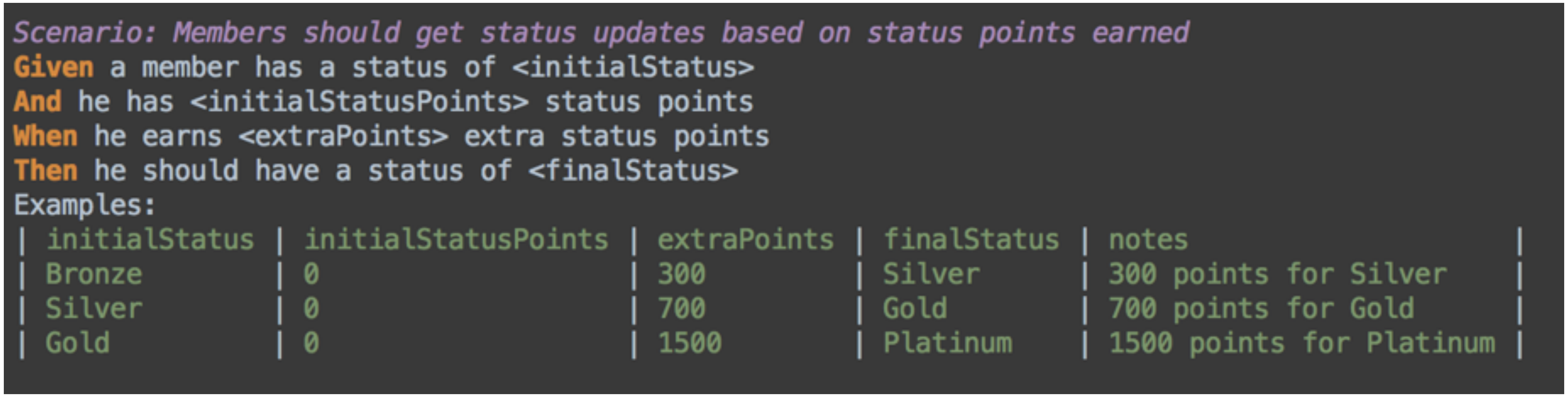
jbehave

Cucumber

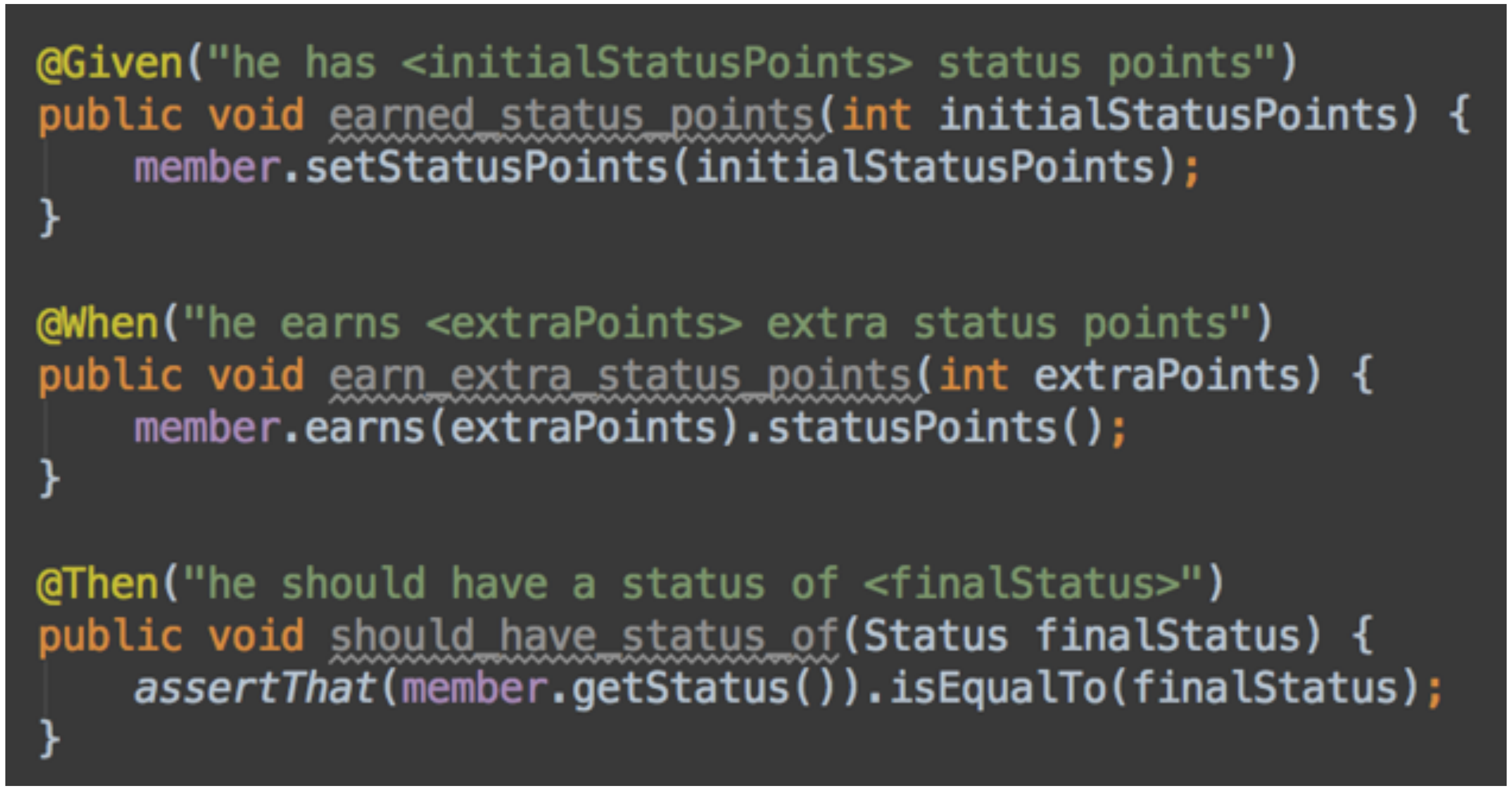
specflow
Cucumber for .NET

thucydides

...

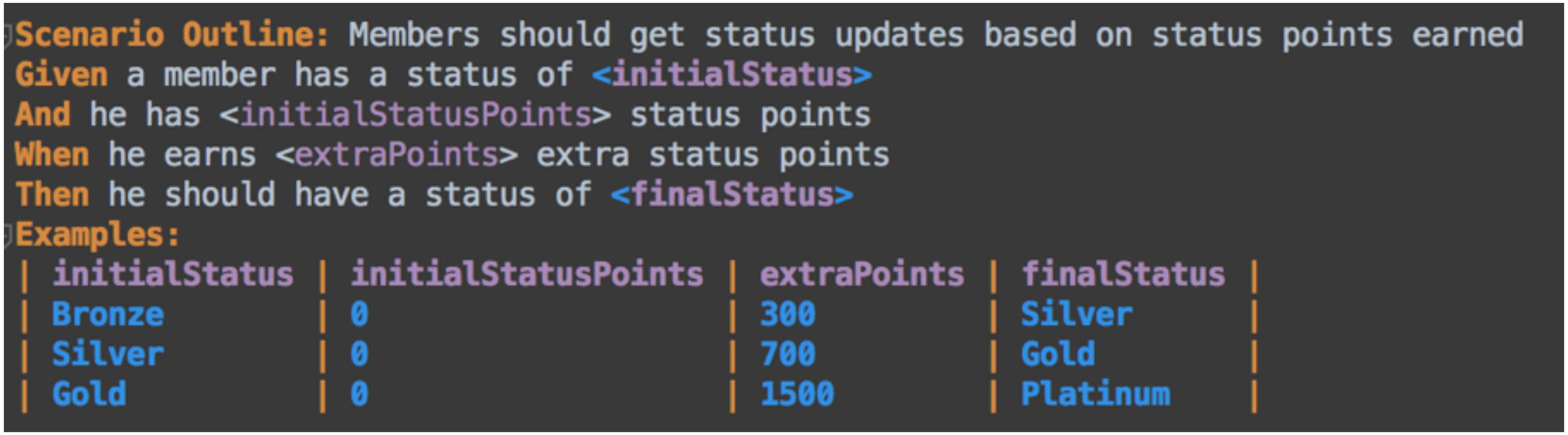


```
Scenario: Members should get status updates based on status points earned
Given a member has a status of <initialStatus>
And he has <initialStatusPoints> status points
When he earns <extraPoints> extra status points
Then he should have a status of <finalStatus>
Examples:
| initialStatus | initialStatusPoints | extraPoints | finalStatus | notes                    |
| Bronze        | 0                   | 300         | Silver      | 300 points for Silver    |
| Silver        | 0                   | 700         | Gold        | 700 points for Gold      |
| Gold          | 0                   | 1500        | Platinum    | 1500 points for Platinum |
```

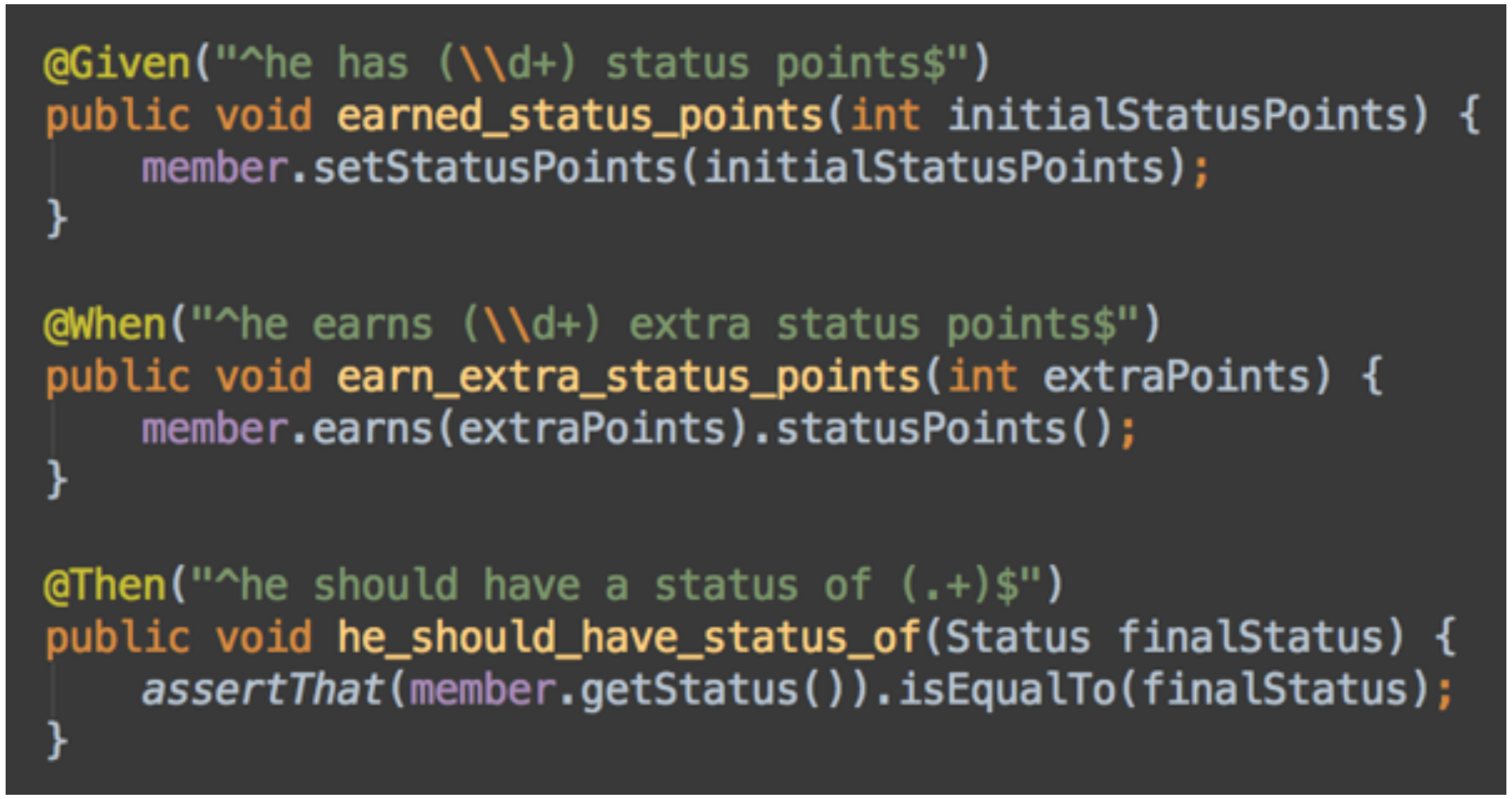


```java
@Given("he has <initialStatusPoints> status points")
public void earned_status_points(int initialStatusPoints) {
    member.setStatusPoints(initialStatusPoints);
}

@When("he earns <extraPoints> extra status points")
public void earn_extra_status_points(int extraPoints) {
    member.earns(extraPoints).statusPoints();
}

@Then("he should have a status of <finalStatus>")
public void should_have_status_of(Status finalStatus) {
    assertThat(member.getStatus()).isEqualTo(finalStatus);
}
```

Cucumber

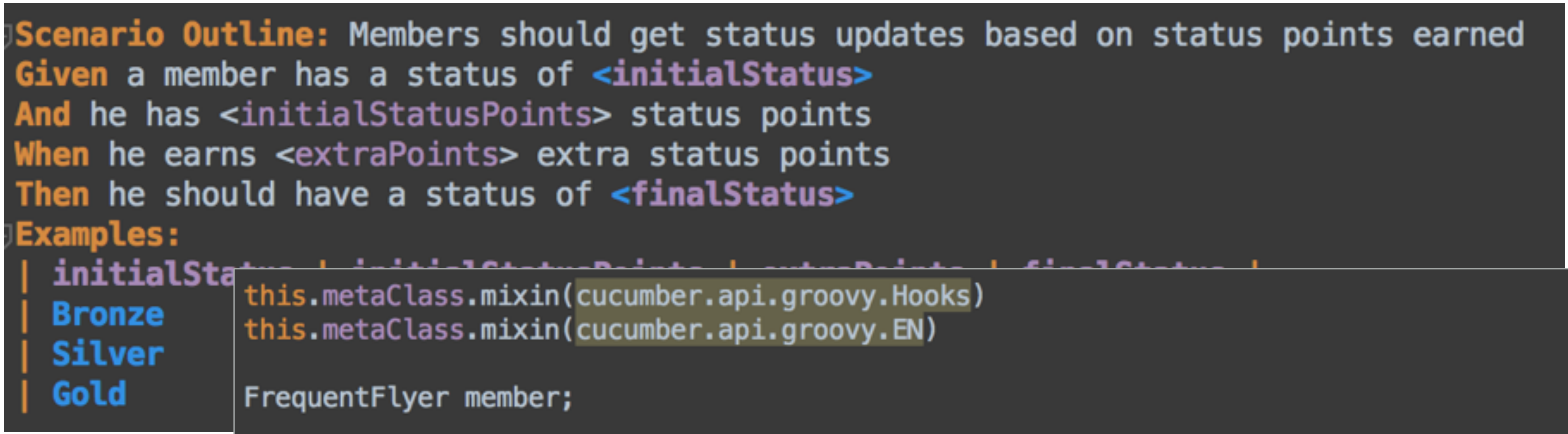


```
Scenario Outline: Members should get status updates based on status points earned
Given a member has a status of <initialStatus>
And he has <initialStatusPoints> status points
When he earns <extraPoints> extra status points
Then he should have a status of <finalStatus>
Examples:
| initialStatus | initialStatusPoints | extraPoints | finalStatus |
| Bronze        | 0                   | 300         | Silver      |
| Silver        | 0                   | 700         | Gold        |
| Gold          | 0                   | 1500        | Platinum    |
```

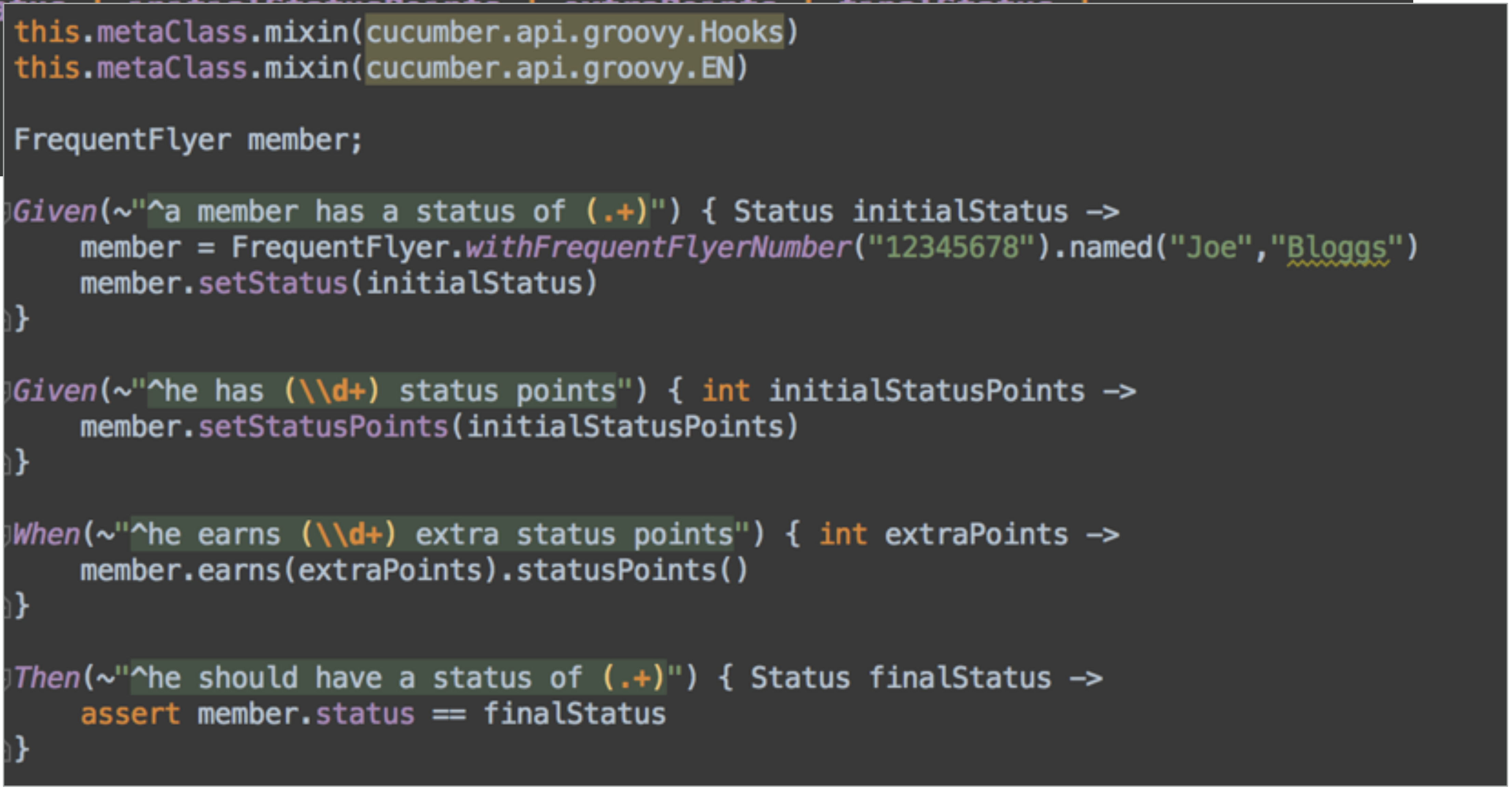


```
@Given("^he has (\\d+) status points$")
public void earned_status_points(int initialStatusPoints) {
    member.setStatusPoints(initialStatusPoints);
}

@When("^he earns (\\d+) extra status points$")
public void earn_extra_status_points(int extraPoints) {
    member.earns(extraPoints).statusPoints();
}

@Then("^he should have a status of (.+)$")
public void he_should_have_status_of(Status finalStatus) {
    assertThat(member.getStatus()).isEqualTo(finalStatus);
}
```

Cucumber + Groovy



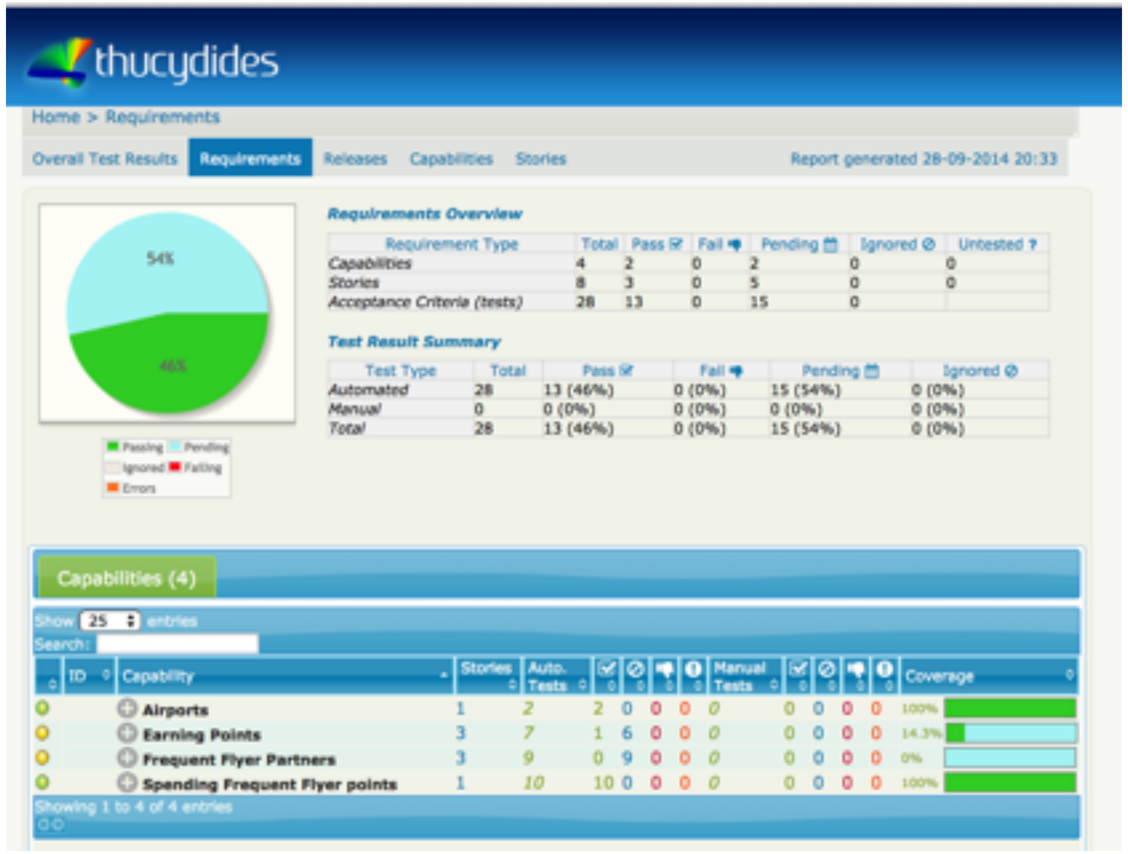
```
Scenario Outline: Members should get status updates based on status points earned
Given a member has a status of <initialStatus>
And he has <initialStatusPoints> status points
When he earns <extraPoints> extra status points
Then he should have a status of <finalStatus>
Examples:
| initialSta
| Bronze
| Silver
| Gold
```



```
this.metaClass.mixin(cucumber.api.groovy.Hooks)
this.metaClass.mixin(cucumber.api.groovy.EN)

FrequentFlyer member;

Given(~"^a member has a status of (.+)") { Status initialStatus ->
    member = FrequentFlyer.withFrequentFlyerNumber("12345678").named("Joe","Bloggs")
    member.setStatus(initialStatus)
}

Given(~"^he has (\\d+) status points") { int initialStatusPoints ->
    member.setStatusPoints(initialStatusPoints)
}

When(~"^he earns (\\d+) extra status points") { int extraPoints ->
    member.earns(extraPoints).statusPoints()
}

Then(~"^he should have a status of (.+)") { Status finalStatus ->
    assert member.status == finalStatus
}
```

thucydides

- ☑ Living documentation
- ☑ Requirements reporting
- ☑ Encourages good test architecture
- ☑ Good WebDriver integration
- ☑ Works with other BDD tools



thucydides

Home > Requirements

Overall Test Results | Requirements | Releases | Capabilities | Stories

Report generated 28-09-2014 20:33

Requirements Overview

Requirement Type	Total	Pass ✓	Fail 👎	Pending 📅	Ignored ⊘	Untested ?
Capabilities	4	2	0	2	0	0
Stories	8	3	0	5	0	0
Acceptance Criteria (tests)	28	13	0	15	0	

Test Result Summary

Test Type	Total	Pass ✓	Fail 👎	Pending 📅	Ignored ⊘
Automated	28	13 (46%)	0 (0%)	15 (54%)	0 (0%)
Manual	0	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Total	28	13 (46%)	0 (0%)	15 (54%)	0 (0%)

Capabilities (4)

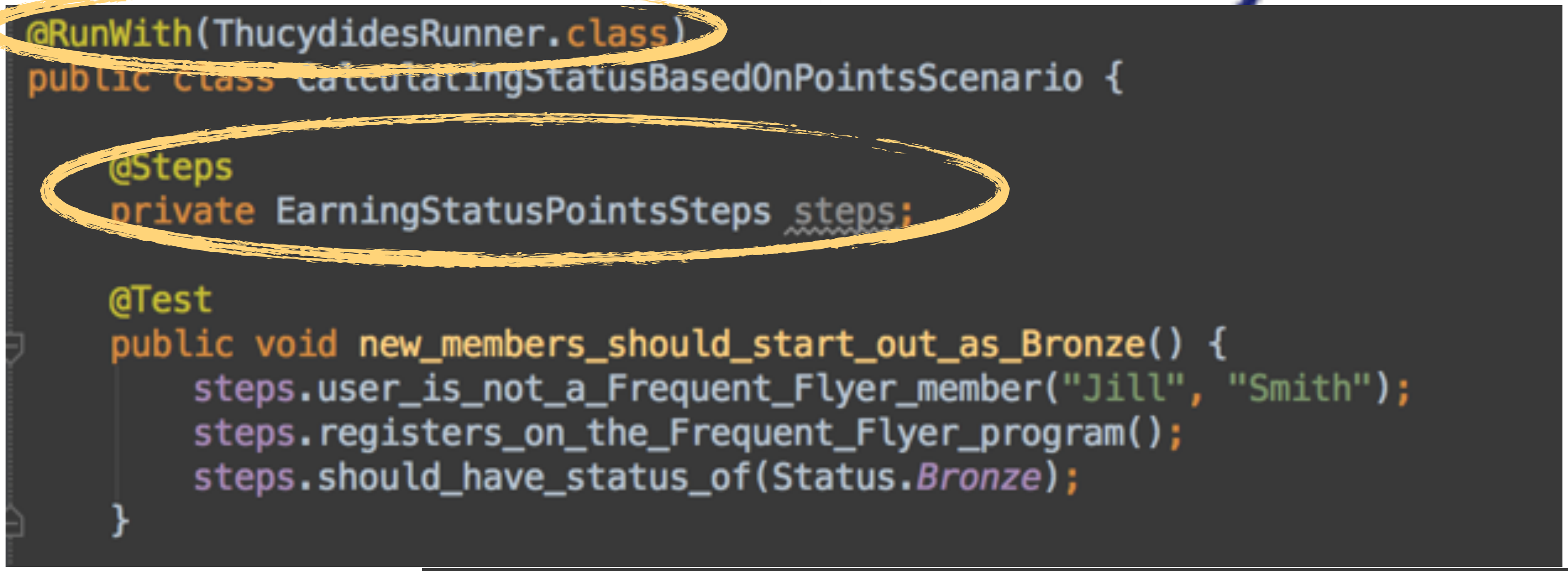
Show 25 entries

Search:

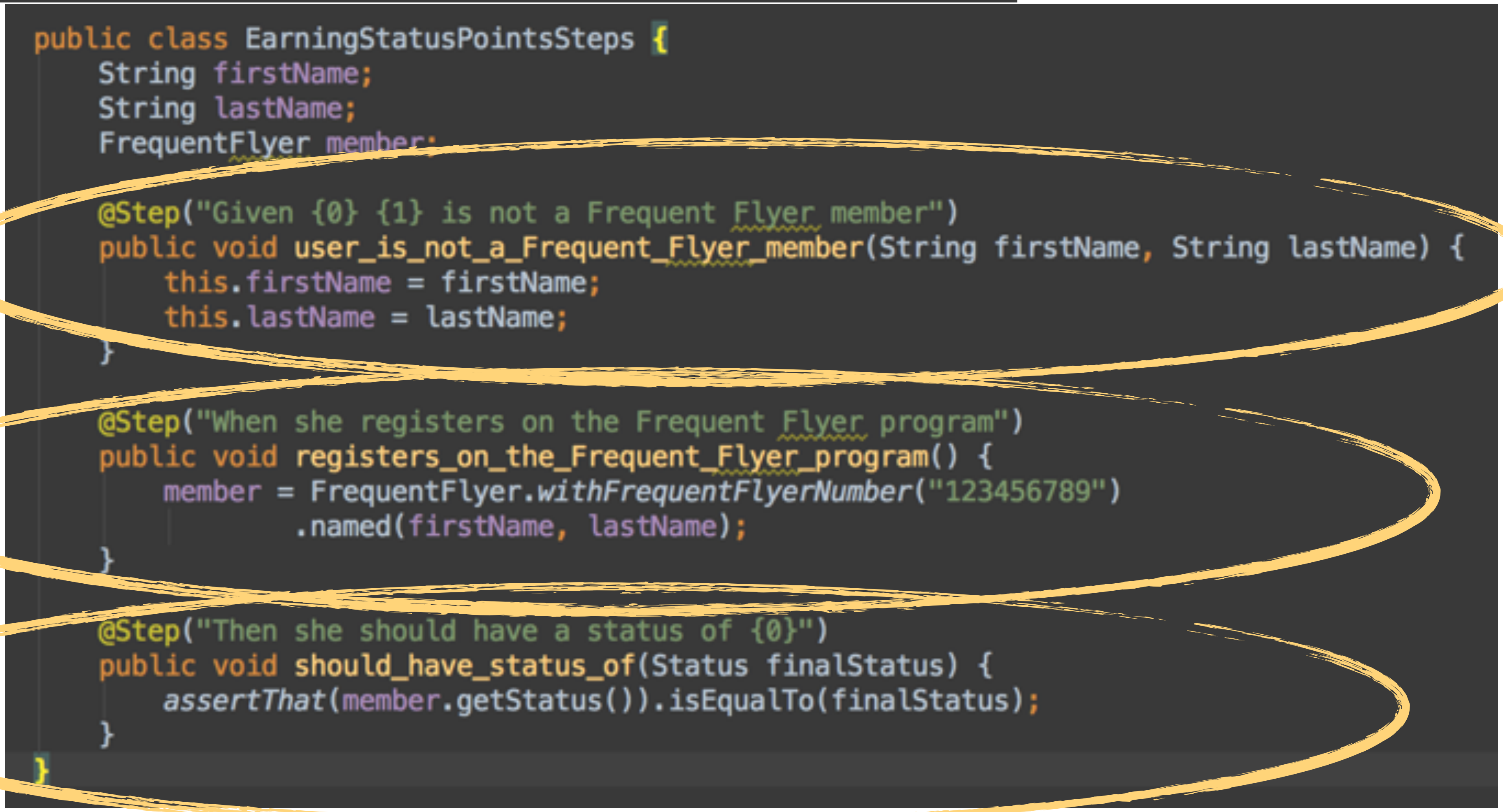
	ID	Capability	Stories	Auto. Tests	✓ 0	⊘ 0	👎 0	! 0	Manual Tests	✓ 0	⊘ 0	👎 0	! 0	Coverage
●		**Airports**	1	2	2	0	0	0	0	0	0	0	0	100%
●		**Earning Points**	3	7	1	6	0	0	0	0	0	0	0	14.3%
●		**Frequent Flyer Partners**	3	9	0	9	0	0	0	0	0	0	0	0%
●		**Spending Frequent Flyer points**	1	10	10	0	0	0	0	0	0	0	0	100%

Showing 1 to 4 of 4 entries

JUnit + thucydides

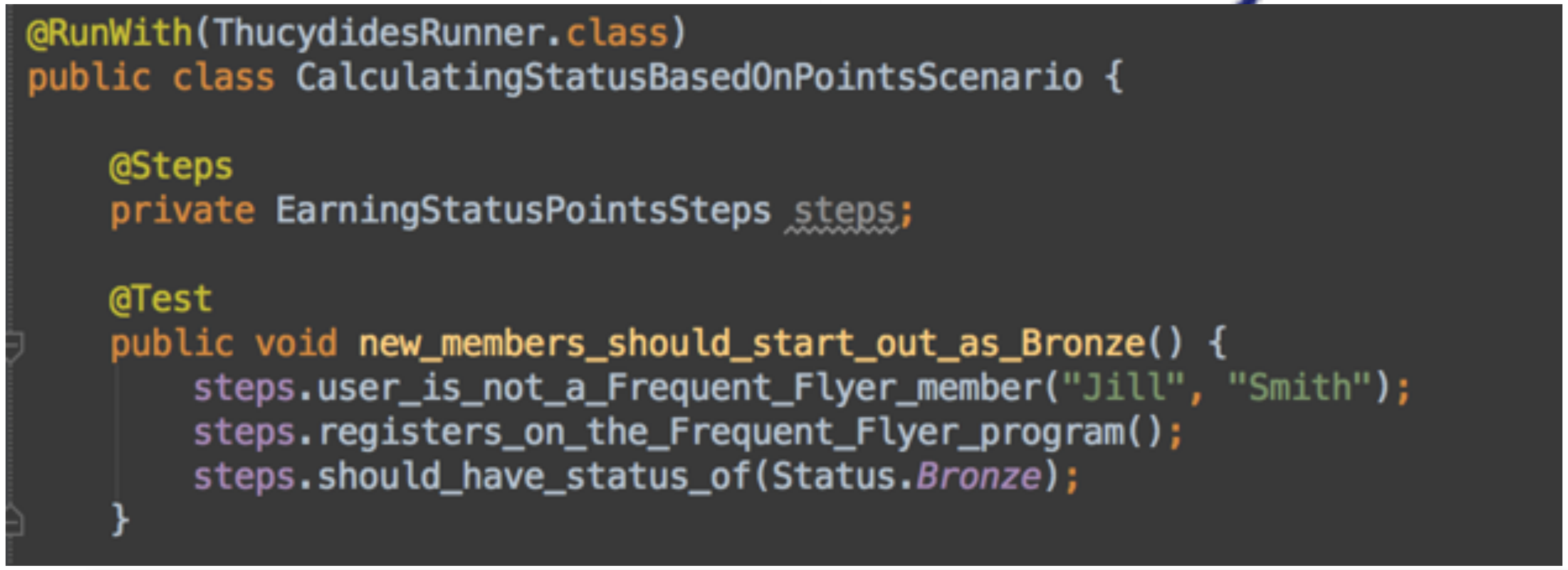


```java
@RunWith(ThucydidesRunner.class)
public class CalculatingStatusBasedOnPointsScenario {

    @Steps
    private EarningStatusPointsSteps steps;

    @Test
    public void new_members_should_start_out_as_Bronze() {
        steps.user_is_not_a_Frequent_Flyer_member("Jill", "Smith");
        steps.registers_on_the_Frequent_Flyer_program();
        steps.should_have_status_of(Status.Bronze);
    }
```

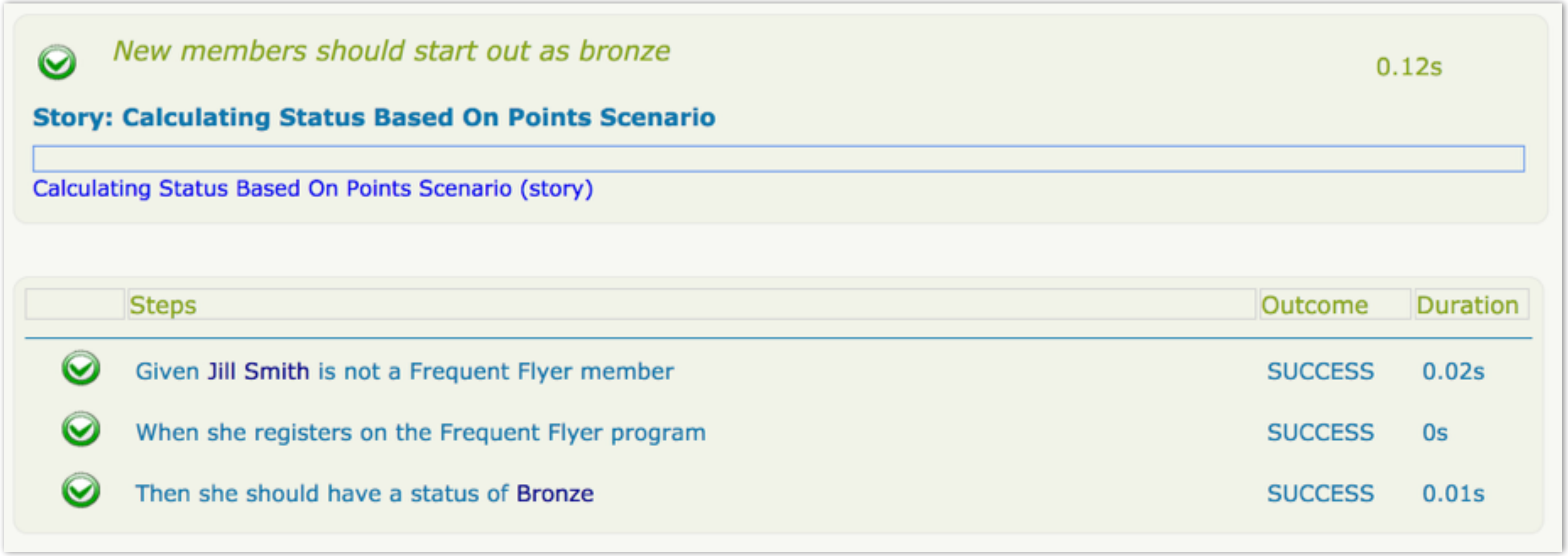


```java
public class EarningStatusPointsSteps {
    String firstName;
    String lastName;
    FrequentFlyer member;

    @Step("Given {0} {1} is not a Frequent Flyer member")
    public void user_is_not_a_Frequent_Flyer_member(String firstName, String lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
    }

    @Step("When she registers on the Frequent Flyer program")
    public void registers_on_the_Frequent_Flyer_program() {
        member = FrequentFlyer.withFrequentFlyerNumber("123456789")
                .named(firstName, lastName);
    }

    @Step("Then she should have a status of {0}")
    public void should_have_status_of(Status finalStatus) {
        assertThat(member.getStatus()).isEqualTo(finalStatus);
    }
}
```

JUnit + thucydides



```java
@RunWith(ThucydidesRunner.class)
public class CalculatingStatusBasedOnPointsScenario {

    @Steps
    private EarningStatusPointsSteps steps;

    @Test
    public void new_members_should_start_out_as_Bronze() {
        steps.user_is_not_a_Frequent_Flyer_member("Jill", "Smith");
        steps.registers_on_the_Frequent_Flyer_program();
        steps.should_have_status_of(Status.Bronze);
    }
```



✓ *New members should start out as bronze* 0.12s

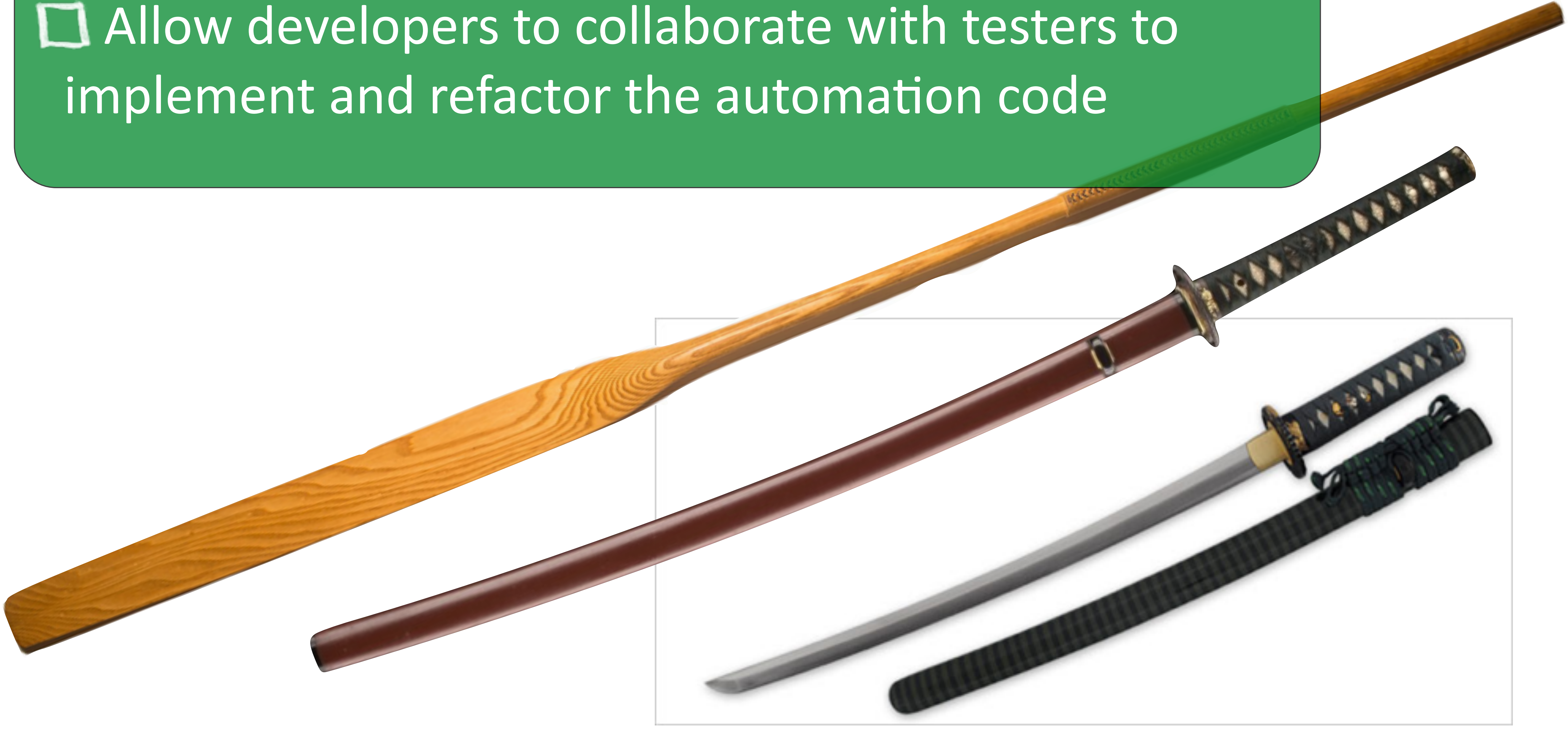
Story: Calculating Status Based On Points Scenario

Calculating Status Based On Points Scenario (story)

	Steps	Outcome	Duration
✓	Given Jill Smith is not a Frequent Flyer member	SUCCESS	0.02s
✓	When she registers on the Frequent Flyer program	SUCCESS	0s
✓	Then she should have a status of Bronze	SUCCESS	0.01s

A high-level BDD tool should

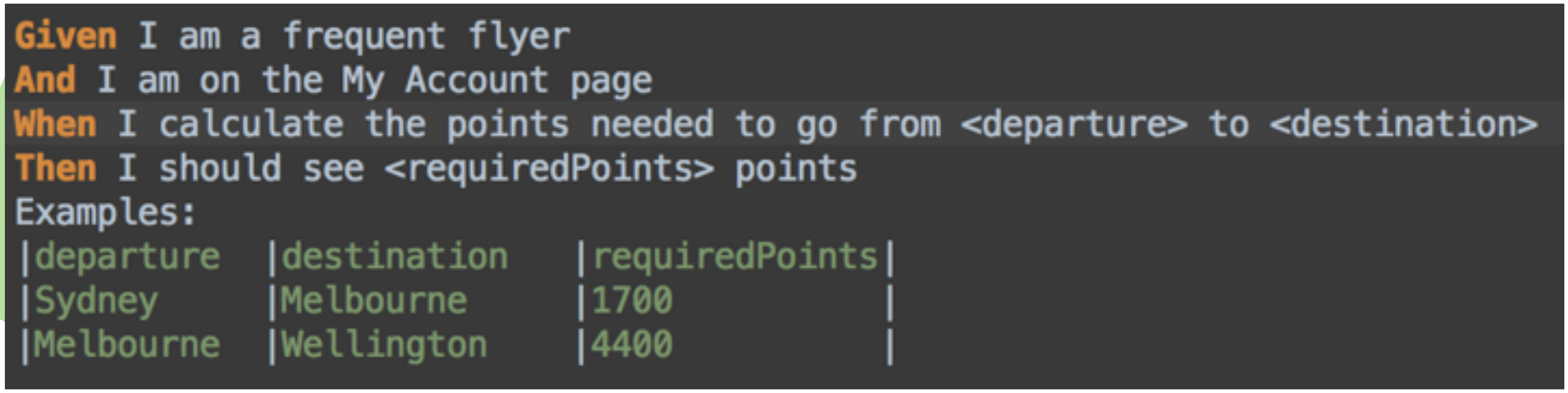
- ☐ Produce business-readable results
- ☐ Fit smoothly into your build pipeline
- ☐ Integrate with your development infrastructure
- ☐ Allow developers to collaborate with testers to implement and refactor the automation code



Tips for more effective BDD Test Automation

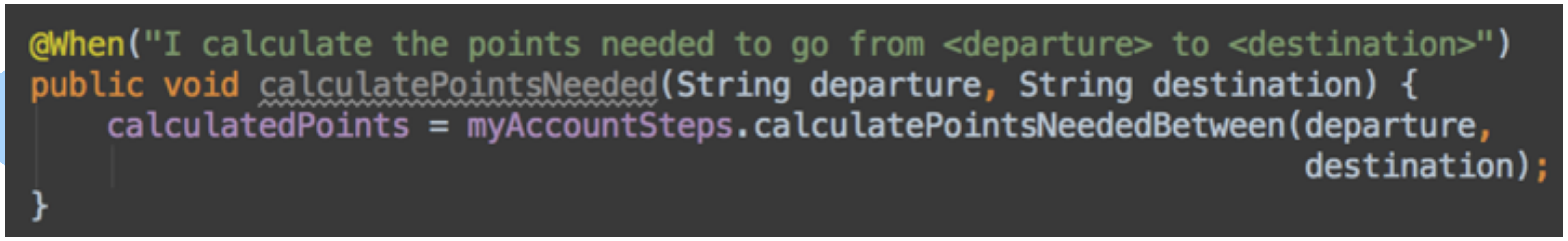
Tip #1 - Use Layers

Business Rules



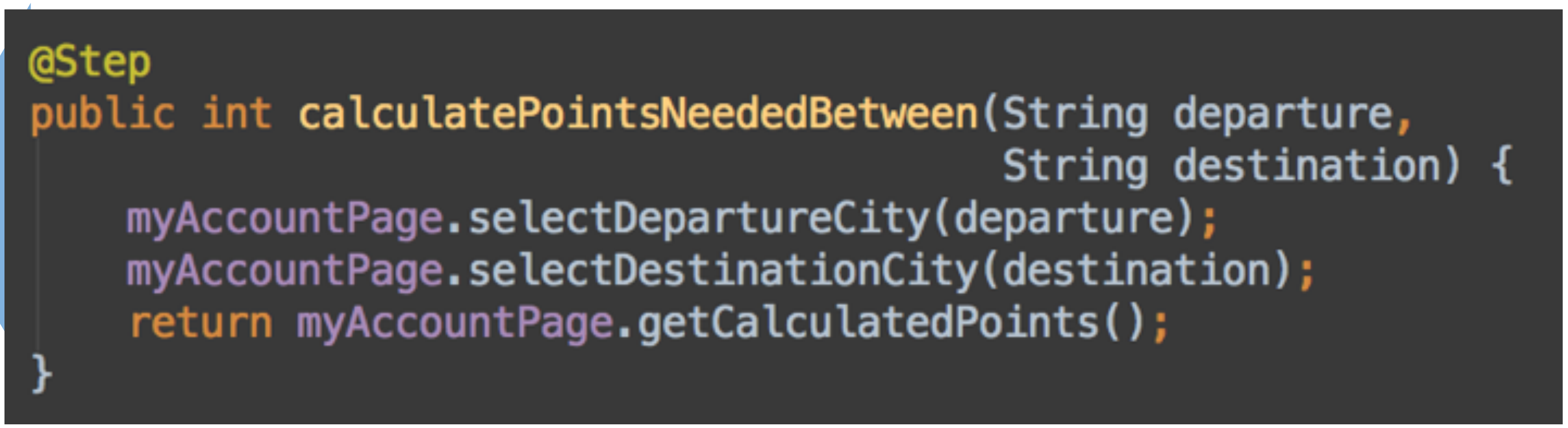
```
Given I am a frequent flyer
And I am on the My Account page
When I calculate the points needed to go from <departure> to <destination>
Then I should see <requiredPoints> points
Examples:
|departure   |destination    |requiredPoints|
|Sydney      |Melbourne      |1700          |
|Melbourne   |Wellington     |4400          |
```

Business Flow



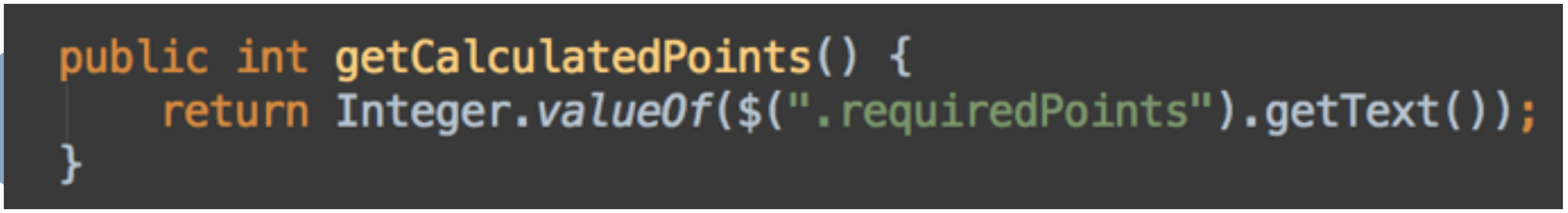
```
@When("I calculate the points needed to go from <departure> to <destination>")
public void calculatePointsNeeded(String departure, String destination) {
    calculatedPoints = myAccountSteps.calculatePointsNeededBetween(departure,
                                                                  destination);
}
```

Page/Component interactions



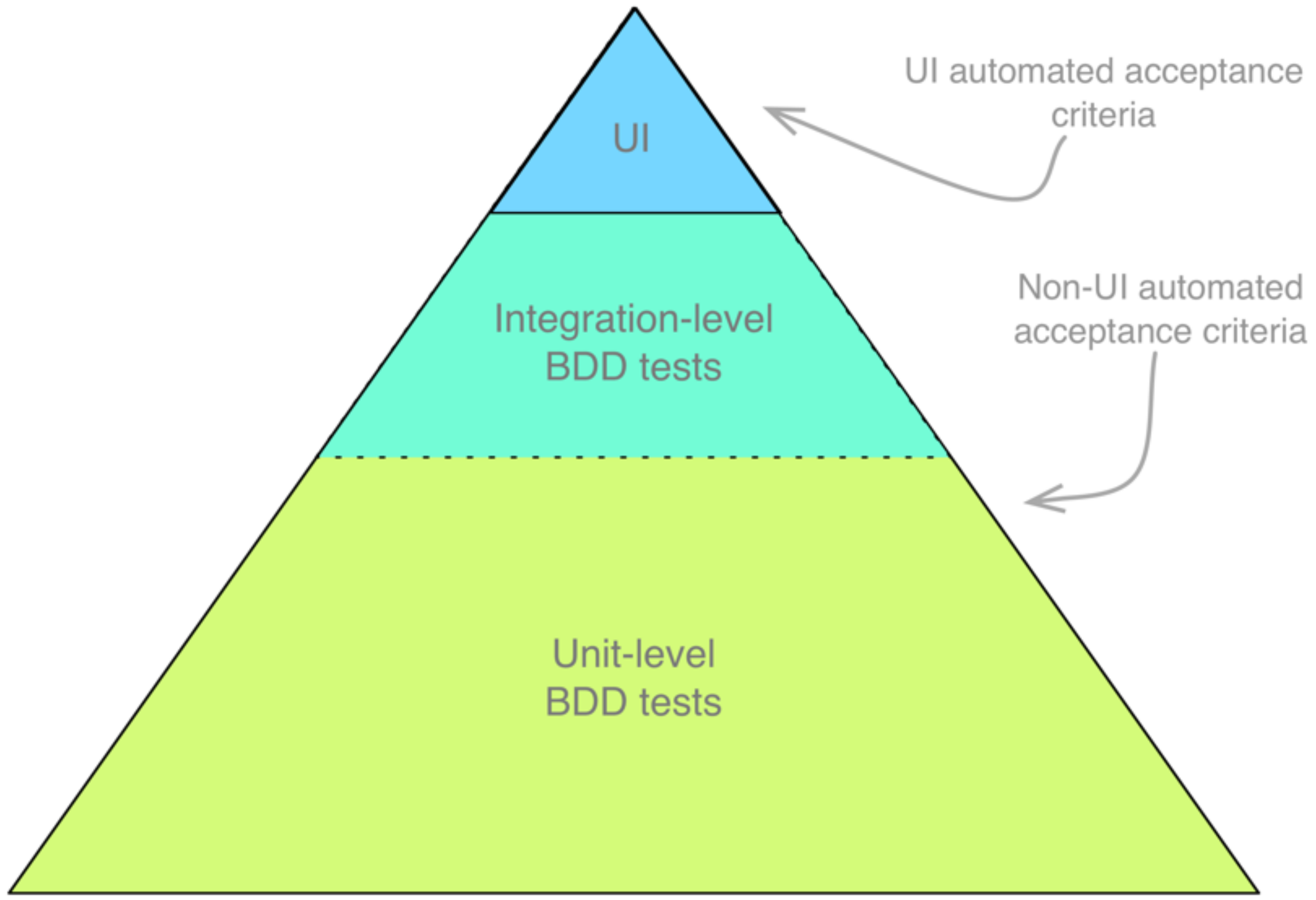
```
@Step
public int calculatePointsNeededBetween(String departure,
                                        String destination) {
    myAccountPage.selectDepartureCity(departure);
    myAccountPage.selectDestinationCity(destination);
    return myAccountPage.getCalculatedPoints();
}
```

Page/Component details



```
public int getCalculatedPoints() {
    return Integer.valueOf($(".requiredPoints").getText());
}
```

Tip #2 - Favour non-UI tests where possible


UI
Integration-level BDD tests
Unit-level BDD tests
UI automated acceptance criteria
Non-UI automated acceptance criteria

Tip #3 - Know when not to automate

This slide is left intentionally blank

Low-level BDD

- Technical documentation for other developers
- Implementation details that business owners may not be interested in
- Faster to write and easier to maintain

JUnit

N unit

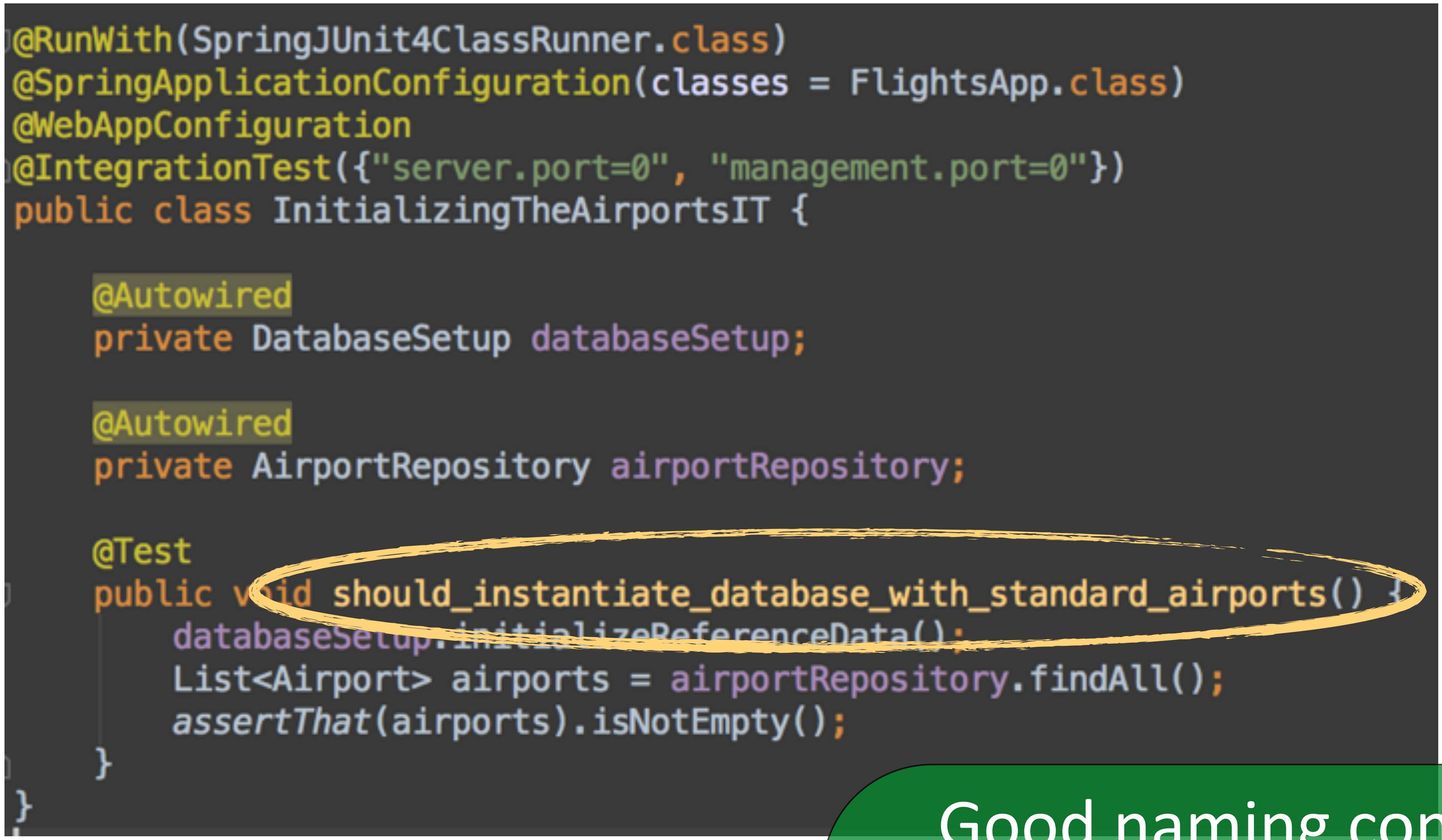
spock

RSpec

Jasmine

Low-level BDD

JUnit

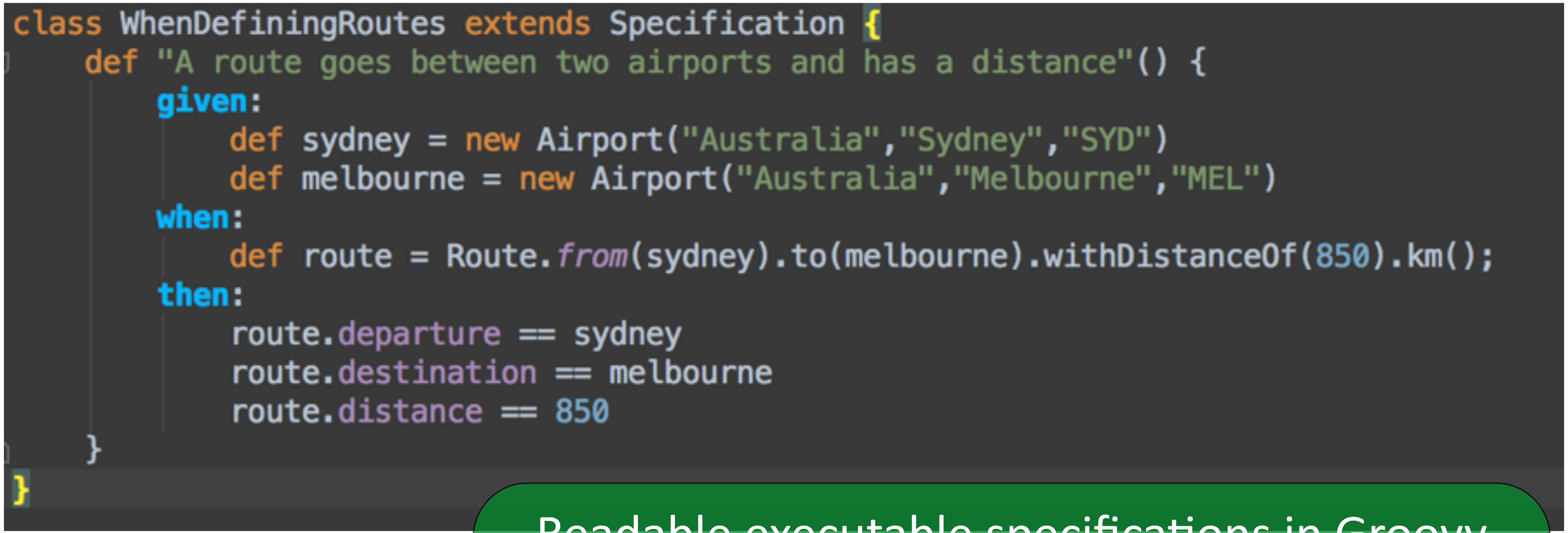


```java
@RunWith(SpringJUnit4ClassRunner.class)
@SpringApplicationConfiguration(classes = FlightsApp.class)
@WebAppConfiguration
@IntegrationTest({"server.port=0", "management.port=0"})
public class InitializingTheAirportsIT {

    @Autowired
    private DatabaseSetup databaseSetup;

    @Autowired
    private AirportRepository airportRepository;

    @Test
    public void should_instantiate_database_with_standard_airports() {
        databaseSetup.initializeReferenceData();
        List<Airport> airports = airportRepository.findAll();
        assertThat(airports).isNotEmpty();
    }
}
```

Good naming conventions are the first step towards BDD

Low-level BDD

spock

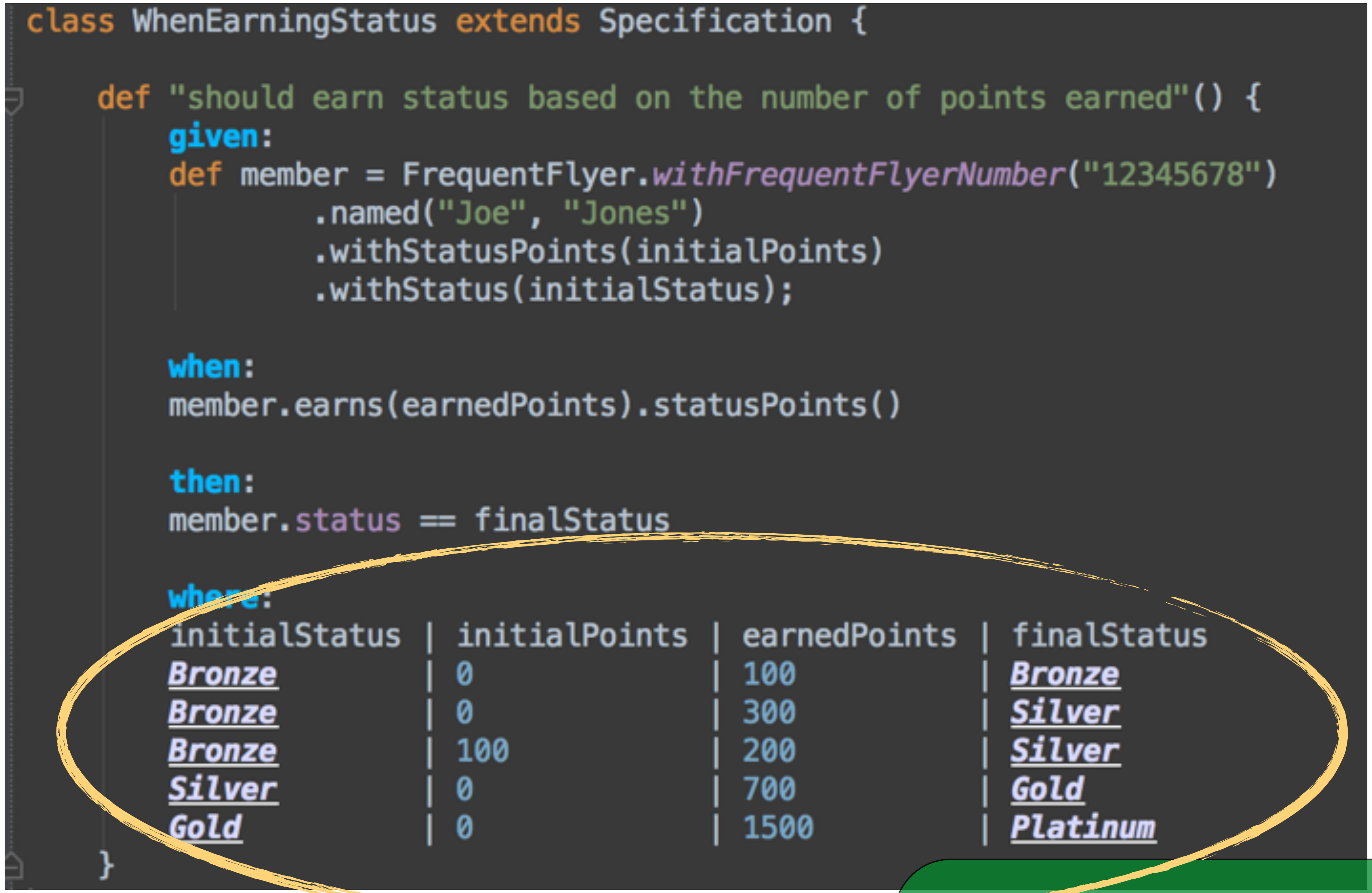


```
class WhenDefiningRoutes extends Specification {
    def "A route goes between two airports and has a distance"() {
        given:
            def sydney = new Airport("Australia","Sydney","SYD")
            def melbourne = new Airport("Australia","Melbourne","MEL")
        when:
            def route = Route.from(sydney).to(melbourne).withDistanceOf(850).km();
        then:
            route.departure == sydney
            route.destination == melbourne
            route.distance == 850
    }
}
```

Readable executable specifications in Groovy

Low-level BDD

spock

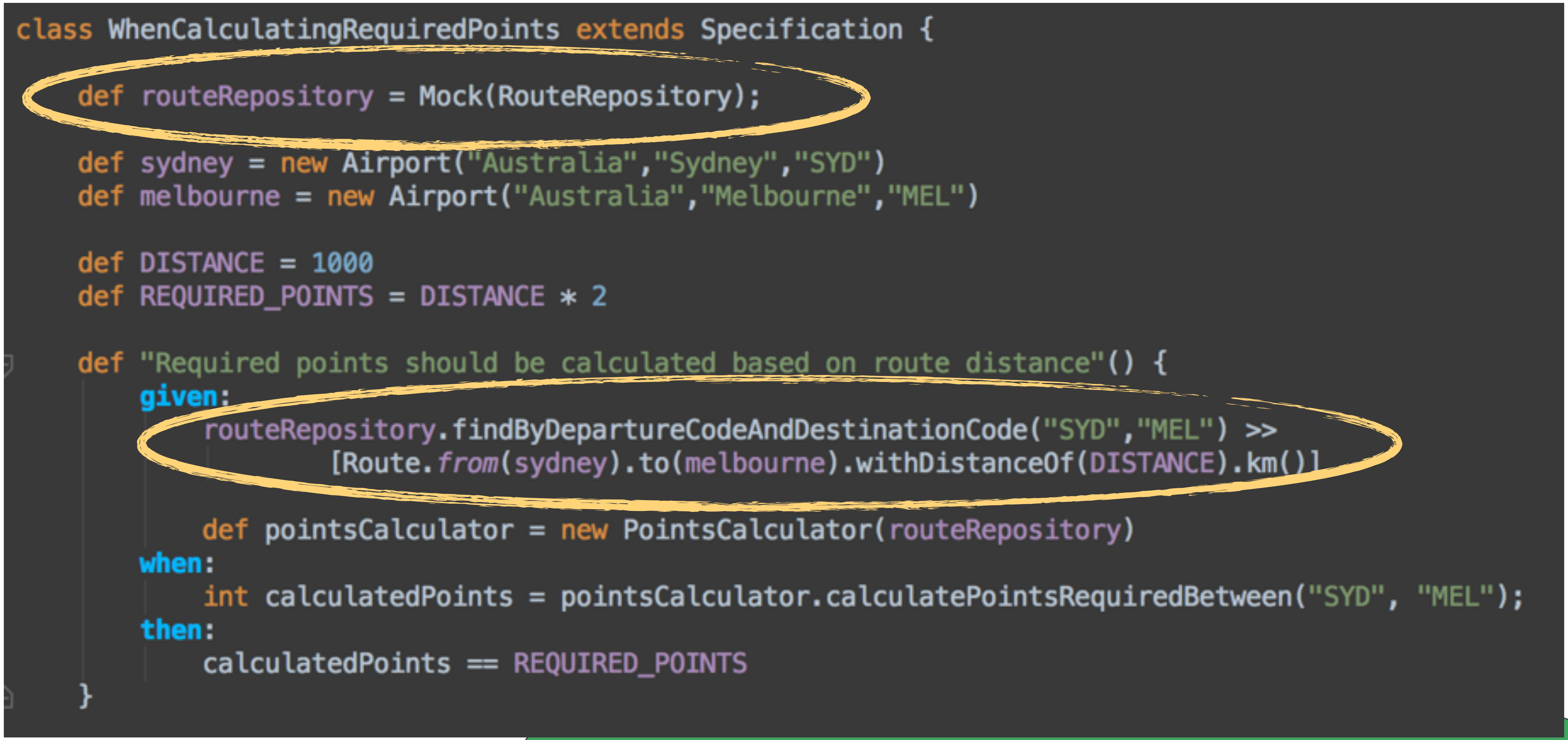


```
class WhenEarningStatus extends Specification {

    def "should earn status based on the number of points earned"() {
        given:
        def member = FrequentFlyer.withFrequentFlyerNumber("12345678")
                .named("Joe", "Jones")
                .withStatusPoints(initialPoints)
                .withStatus(initialStatus);

        when:
        member.earns(earnedPoints).statusPoints()

        then:
        member.status == finalStatus

        where:
        initialStatus | initialPoints | earnedPoints | finalStatus
        Bronze        | 0             | 100          | Bronze
        Bronze        | 0             | 300          | Silver
        Bronze        | 100           | 200          | Silver
        Silver        | 0             | 700          | Gold
        Gold          | 0             | 1500         | Platinum
    }
```

Great support for data-driven tests

Low-level BDD

spock

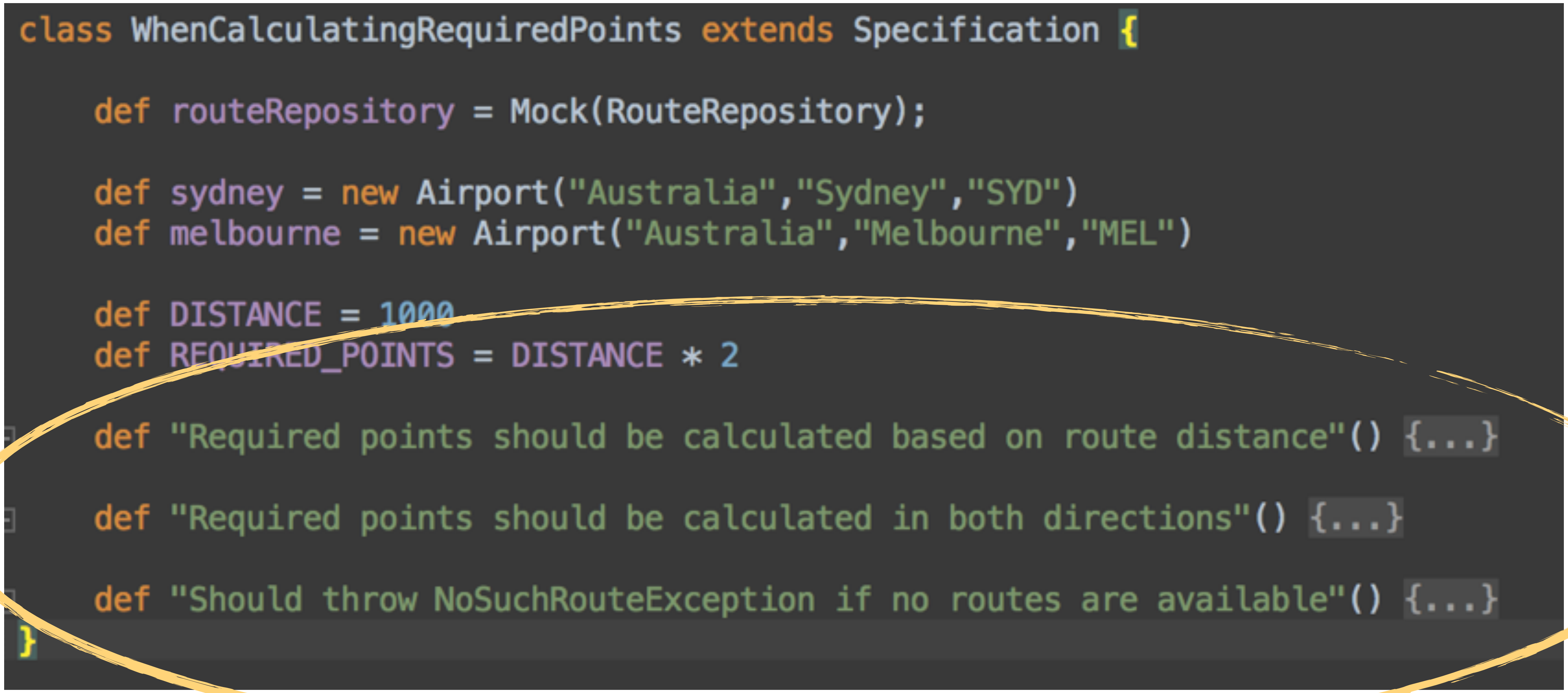


```
class WhenCalculatingRequiredPoints extends Specification {

    def routeRepository = Mock(RouteRepository);

    def sydney = new Airport("Australia","Sydney","SYD")
    def melbourne = new Airport("Australia","Melbourne","MEL")

    def DISTANCE = 1000
    def REQUIRED_POINTS = DISTANCE * 2

    def "Required points should be calculated based on route distance"() {
        given:
            routeRepository.findByDepartureCodeAndDestinationCode("SYD","MEL") >>
                    [Route.from(sydney).to(melbourne).withDistanceOf(DISTANCE).km()]

            def pointsCalculator = new PointsCalculator(routeRepository)
        when:
            int calculatedPoints = pointsCalculator.calculatePointsRequiredBetween("SYD", "MEL");
        then:
            calculatedPoints == REQUIRED_POINTS
    }
```

Powerful and light-weight stubbing and mocking

Low-level BDD

spock

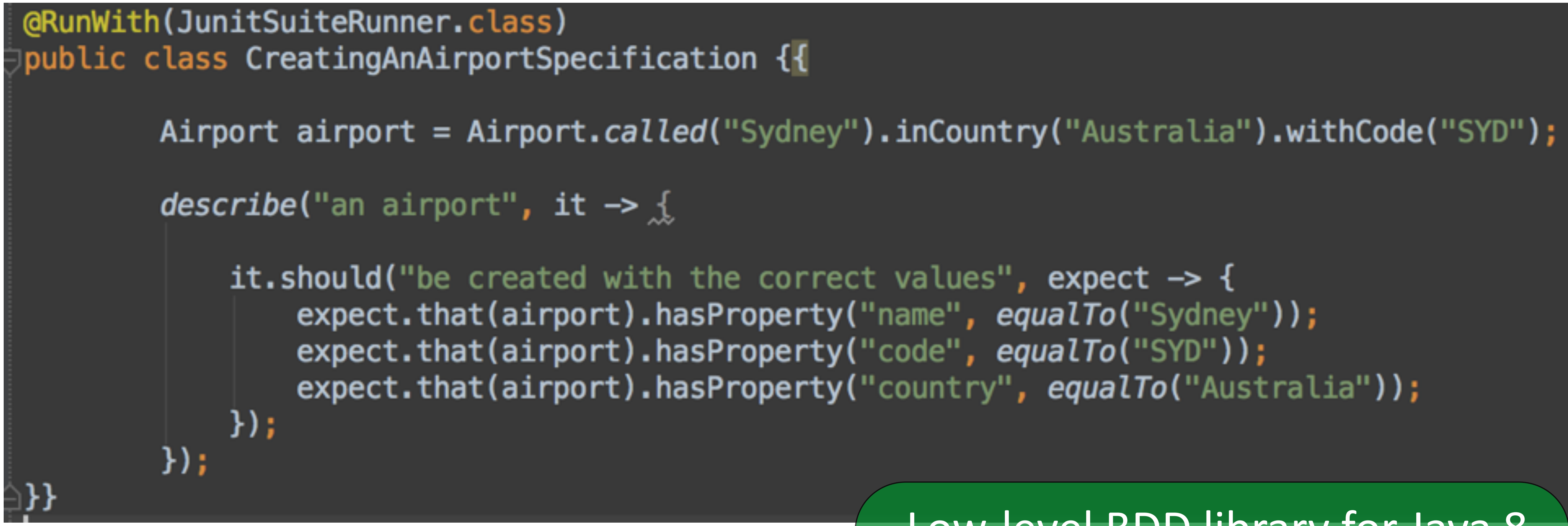


```
class WhenCalculatingRequiredPoints extends Specification {

    def routeRepository = Mock(RouteRepository);

    def sydney = new Airport("Australia","Sydney","SYD")
    def melbourne = new Airport("Australia","Melbourne","MEL")

    def DISTANCE = 1000
    def REQUIRED_POINTS = DISTANCE * 2

    def "Required points should be calculated based on route distance"() {...}

    def "Required points should be calculated in both directions"() {...}

    def "Should throw NoSuchRouteException if no routes are available"() {...}
}
```

Makes very readable specifications

Low-level BDD

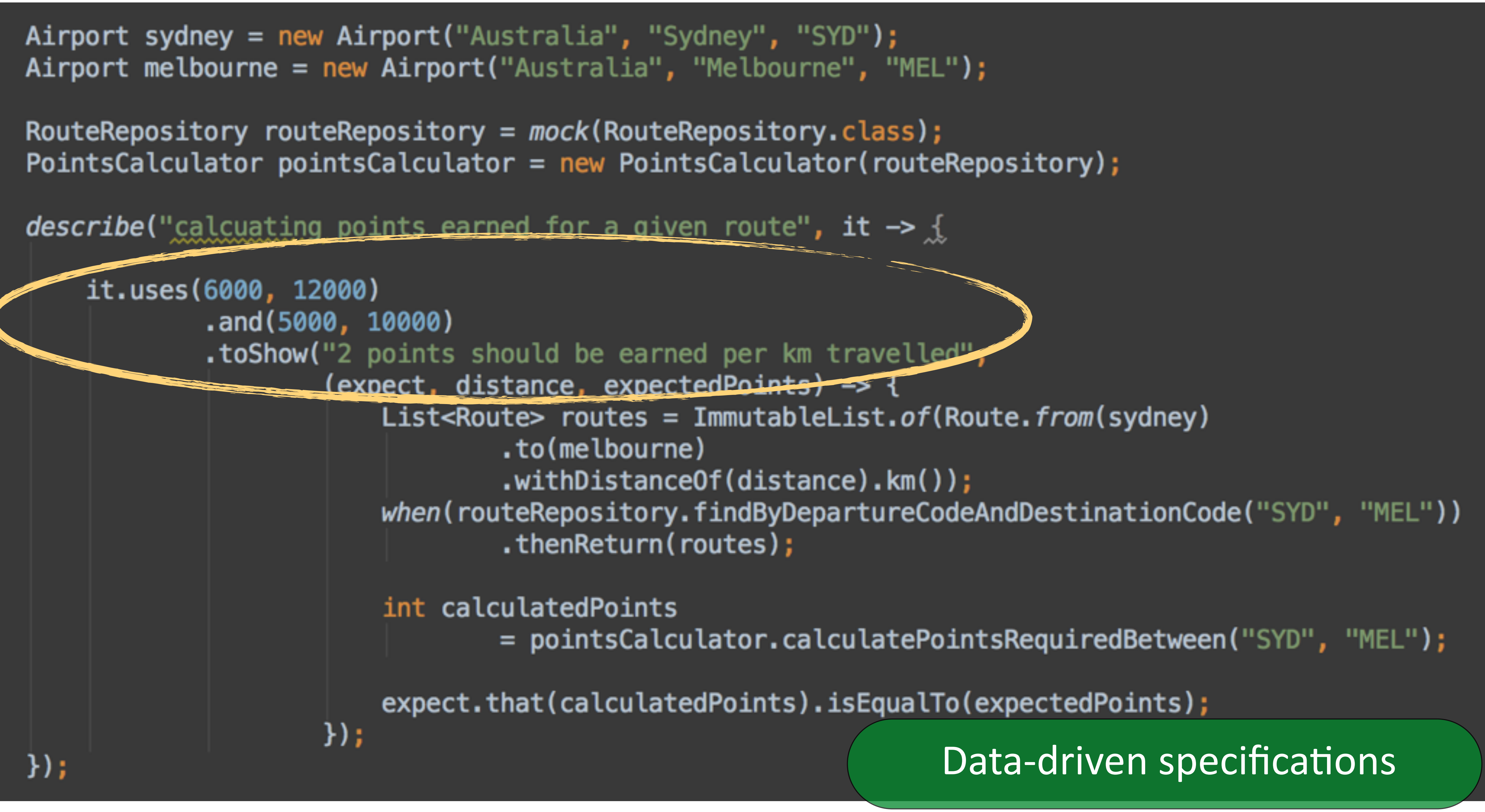
Lambda-Behave


```
@RunWith(JunitSuiteRunner.class)
public class CreatingAnAirportSpecification {{

        Airport airport = Airport.called("Sydney").inCountry("Australia").withCode("SYD");

        describe("an airport", it -> {

            it.should("be created with the correct values", expect -> {
                expect.that(airport).hasProperty("name", equalTo("Sydney"));
                expect.that(airport).hasProperty("code", equalTo("SYD"));
                expect.that(airport).hasProperty("country", equalTo("Australia"));
            });
        });
}}
```

Low-level BDD library for Java 8

Low-level BDD

Lambda-Behave

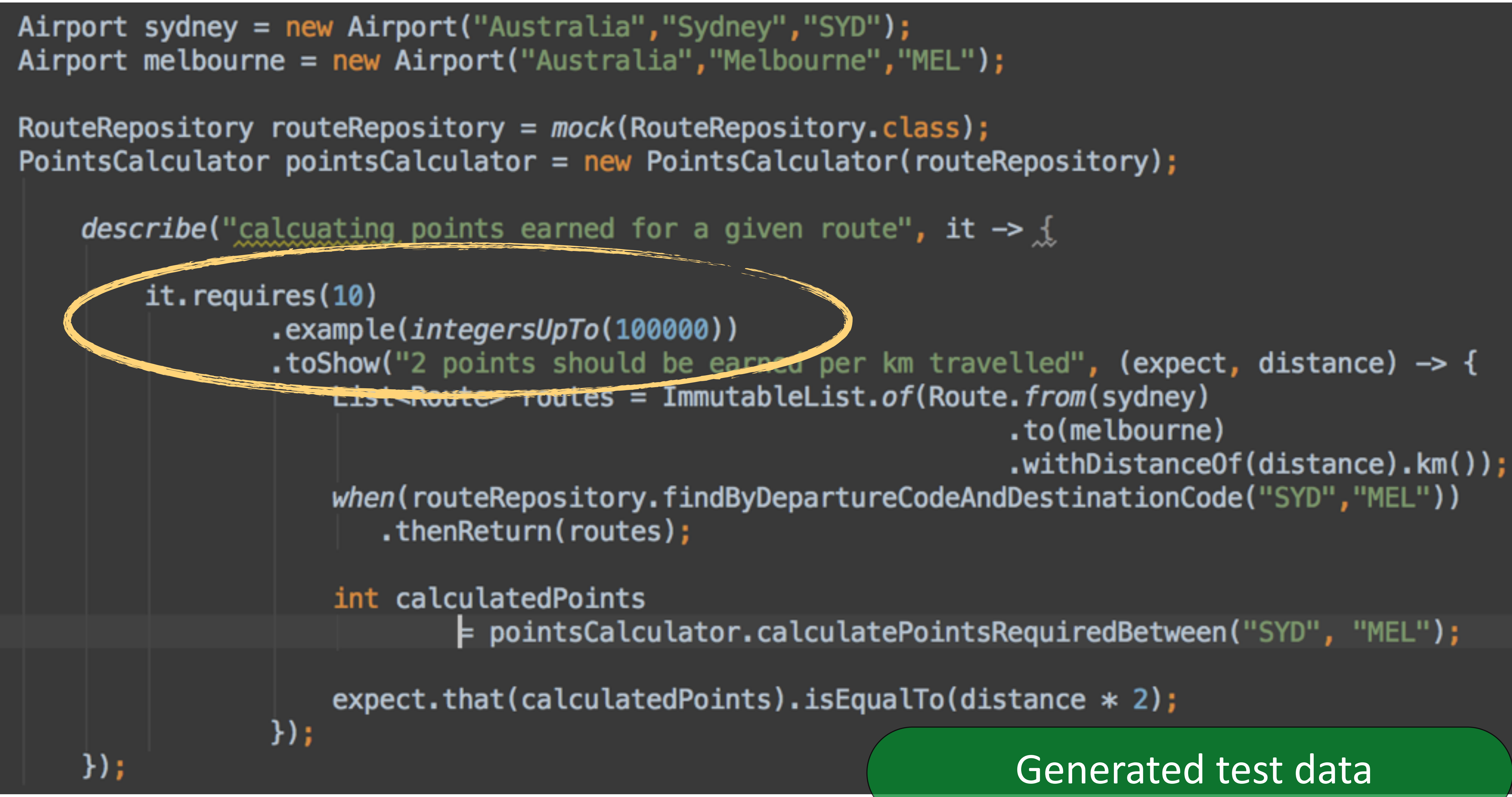


```
Airport sydney = new Airport("Australia", "Sydney", "SYD");
Airport melbourne = new Airport("Australia", "Melbourne", "MEL");

RouteRepository routeRepository = mock(RouteRepository.class);
PointsCalculator pointsCalculator = new PointsCalculator(routeRepository);

describe("calcuating points earned for a given route", it -> {

    it.uses(6000, 12000)
            .and(5000, 10000)
            .toShow("2 points should be earned per km travelled",
                    (expect, distance, expectedPoints) -> {
                        List<Route> routes = ImmutableList.of(Route.from(sydney)
                                .to(melbourne)
                                .withDistanceOf(distance).km());
                        when(routeRepository.findByDepartureCodeAndDestinationCode("SYD", "MEL"))
                                .thenReturn(routes);

                        int calculatedPoints
                                = pointsCalculator.calculatePointsRequiredBetween("SYD", "MEL");

                        expect.that(calculatedPoints).isEqualTo(expectedPoints);
                    });
});
```

Data-driven specifications

Low-level BDD

Lambda-Behave

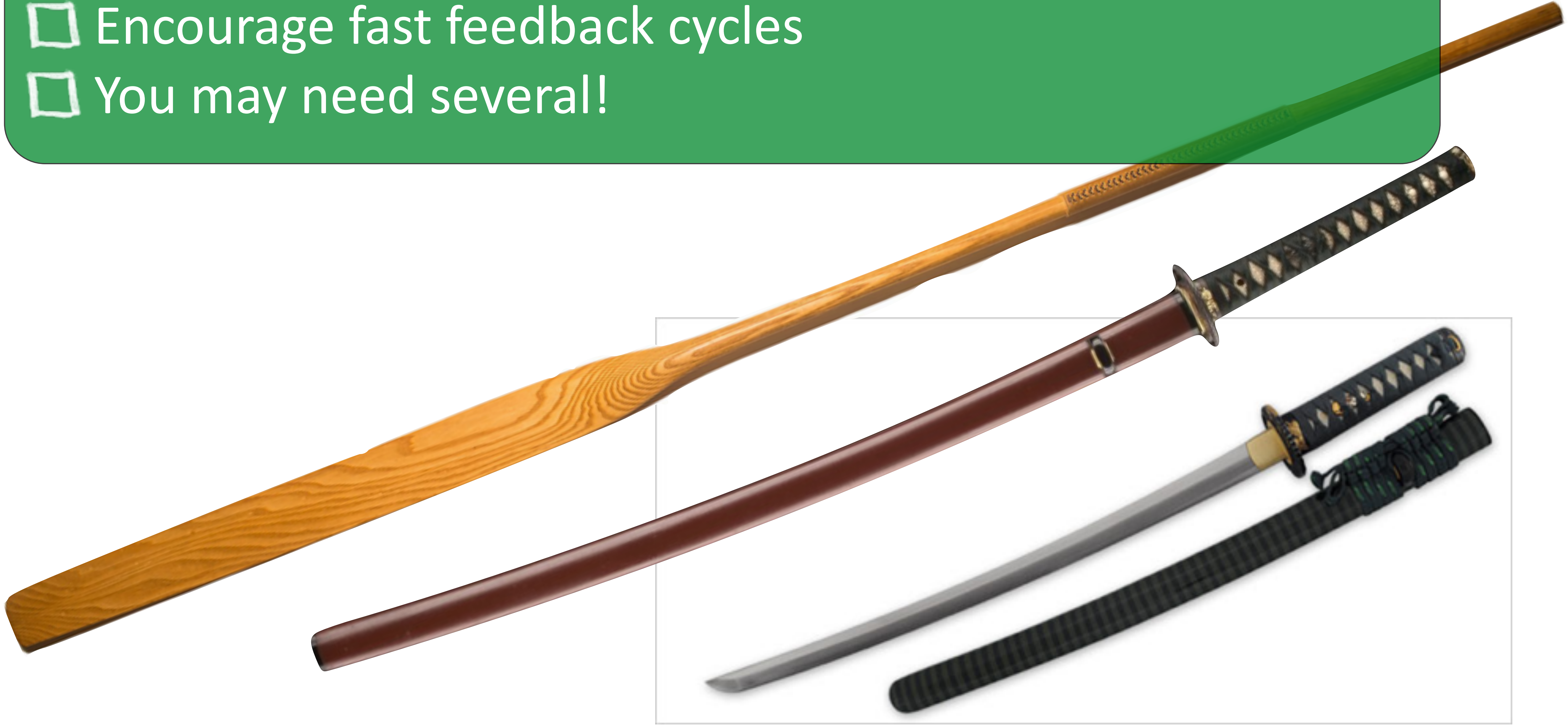


```
Airport sydney = new Airport("Australia","Sydney","SYD");
Airport melbourne = new Airport("Australia","Melbourne","MEL");

RouteRepository routeRepository = mock(RouteRepository.class);
PointsCalculator pointsCalculator = new PointsCalculator(routeRepository);

    describe("calcuating points earned for a given route", it -> {

        it.requires(10)
                .example(integersUpTo(100000))
                .toShow("2 points should be earned per km travelled", (expect, distance) -> {
                    List<Route> routes = ImmutableList.of(Route.from(sydney)
                                                                .to(melbourne)
                                                                .withDistanceOf(distance).km());
                    when(routeRepository.findByDepartureCodeAndDestinationCode("SYD","MEL"))
                        .thenReturn(routes);

                    int calculatedPoints
                            = pointsCalculator.calculatePointsRequiredBetween("SYD", "MEL");

                    expect.that(calculatedPoints).isEqualTo(distance * 2);
                });
    });
```

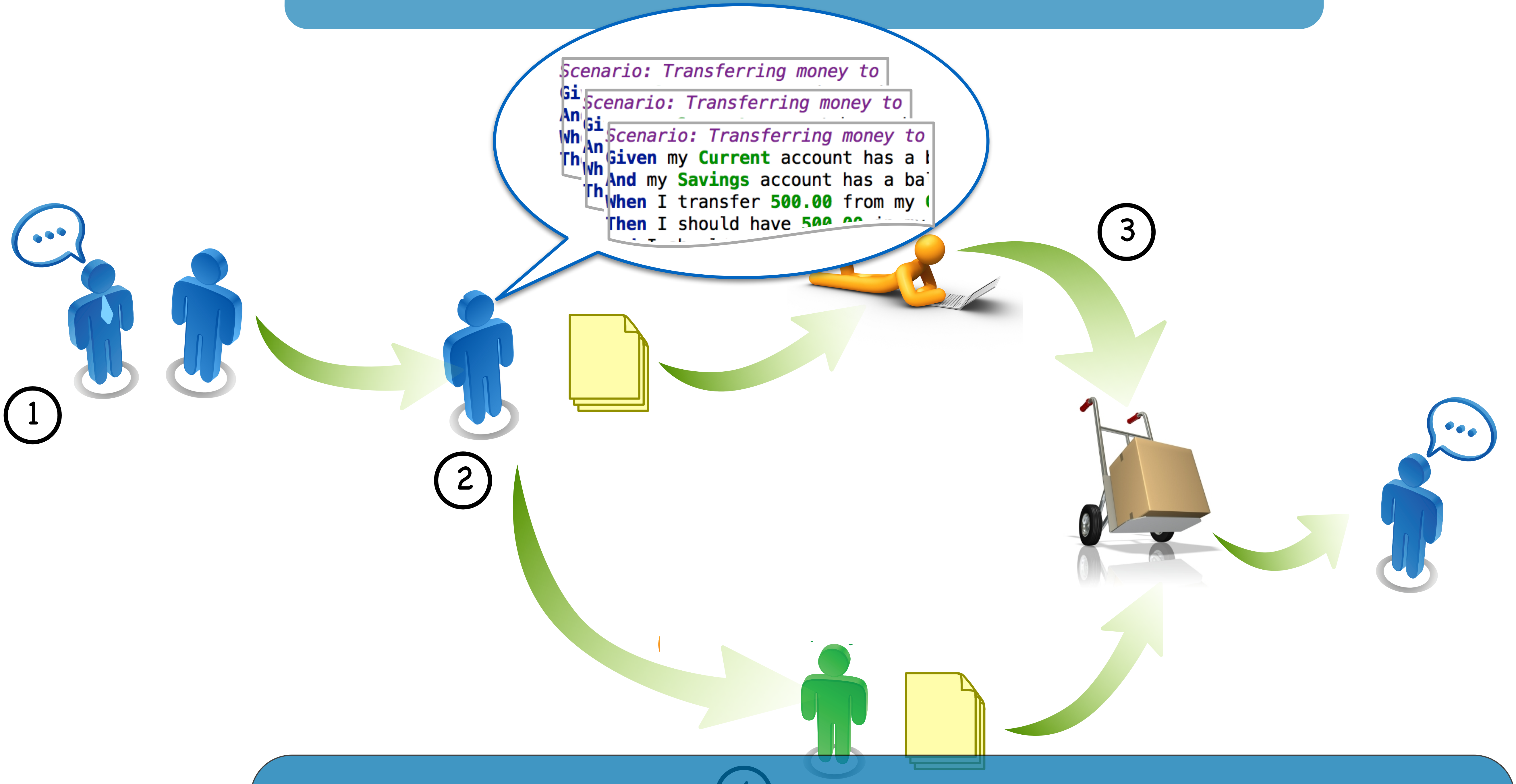
Generated test data

A low-level BDD tool should

- ☐ Be highly readable
- ☐ Be developer-friendly
- ☐ Make it easy to think in terms of specifications, not tests
- ☐ Encourage fast feedback cycles
- ☐ You may need several!



BDD Gotchas

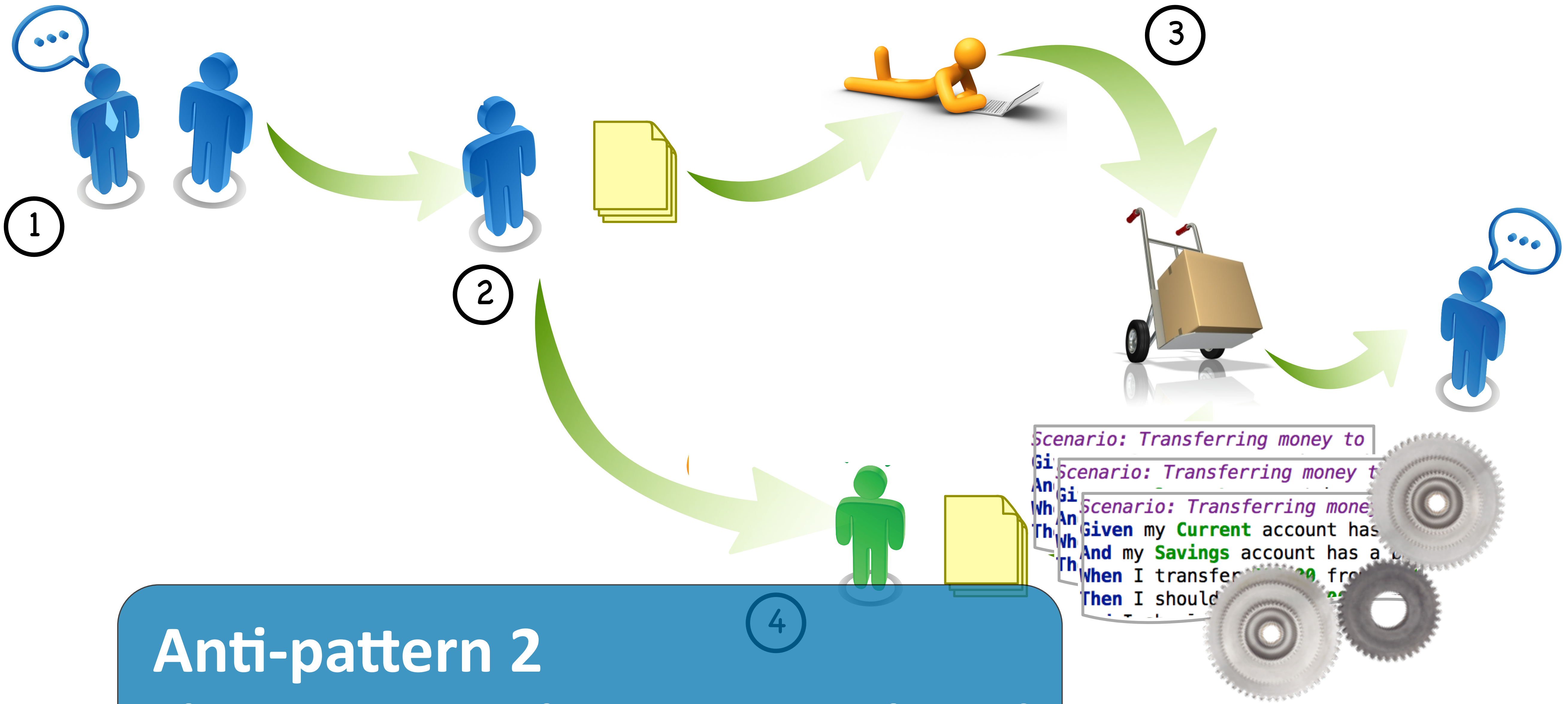


```
Scenario: Transferring money to
Given my Current account has a
And my Savings account has a ba
When I transfer 500.00 from my
Then I should have
```

Anti-pattern 1

The business analyst writes the scenarios and then gives them to the other team members.

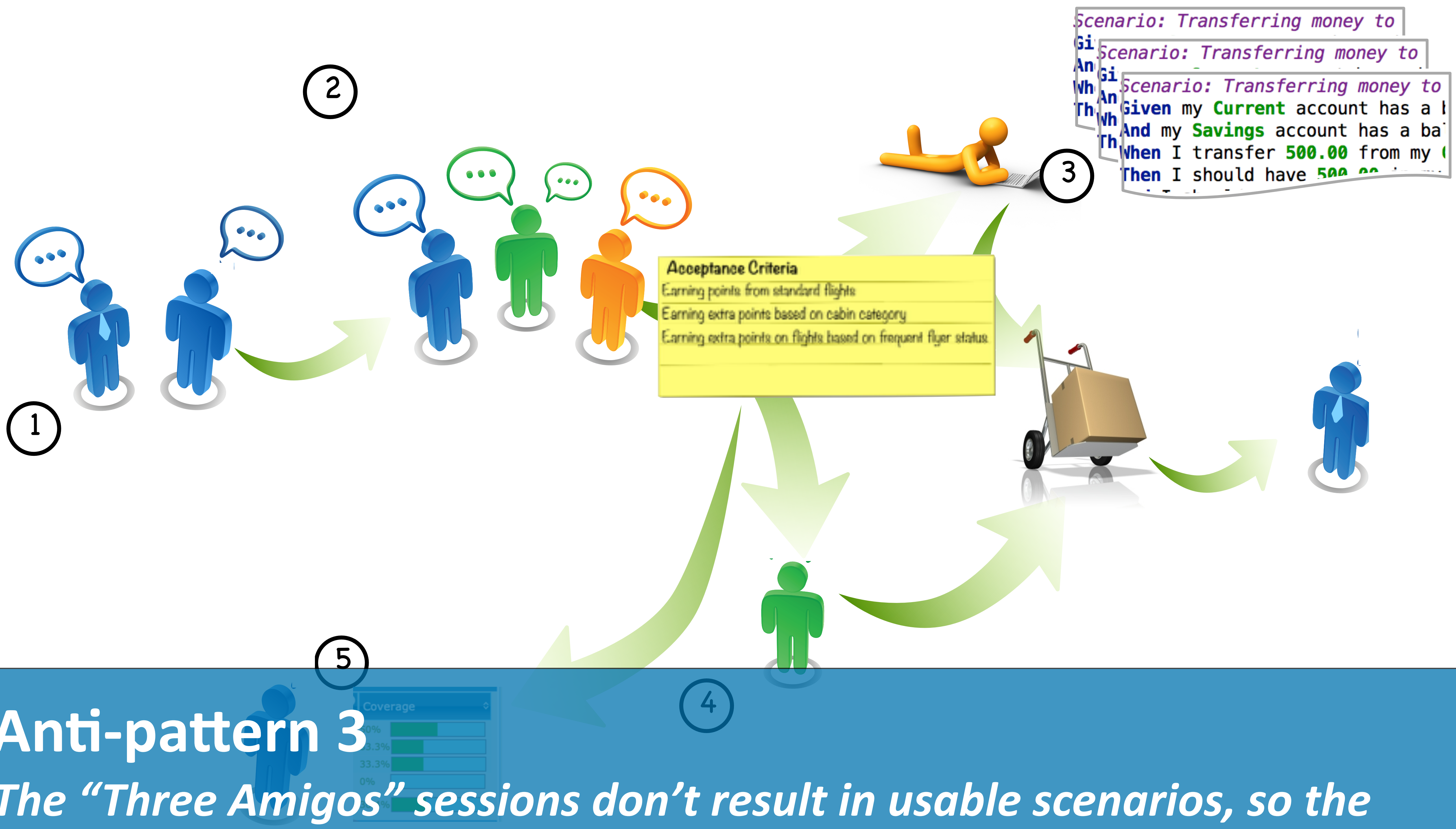
BDD Gotchas


1
2
3
4
Scenario: Transferring money to
Scenario: Transferring money t
Scenario: Transferring money
Given my Current account has
And my Savings account has a
When I transfer
Then I should

Anti-pattern 2

The tester writes the scenarios at the end to implement an automated test suite.

BDD Gotchas



Acceptance Criteria

Earning points from standard flights

Earning extra points based on cabin category

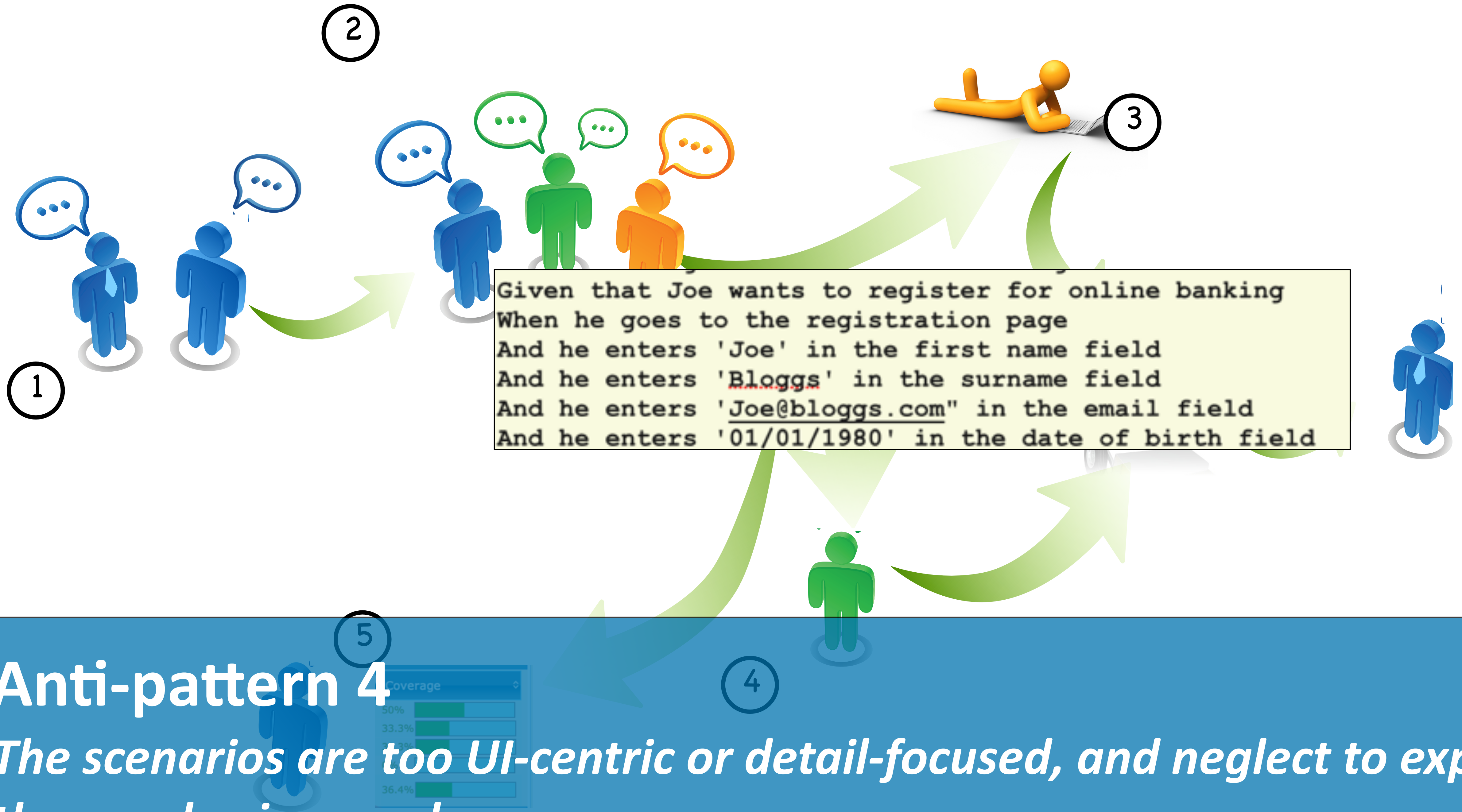
Earning extra points on flights based on frequent flyer status

```
Scenario: Transferring money to
Given my Current account has a
And my Savings account has a ba
When I transfer 500.00 from my
Then I should have 500.00
```

Anti-pattern 3

The "Three Amigos" sessions don't result in usable scenarios, so the developer invents them afterwards.

BDD Gotchas

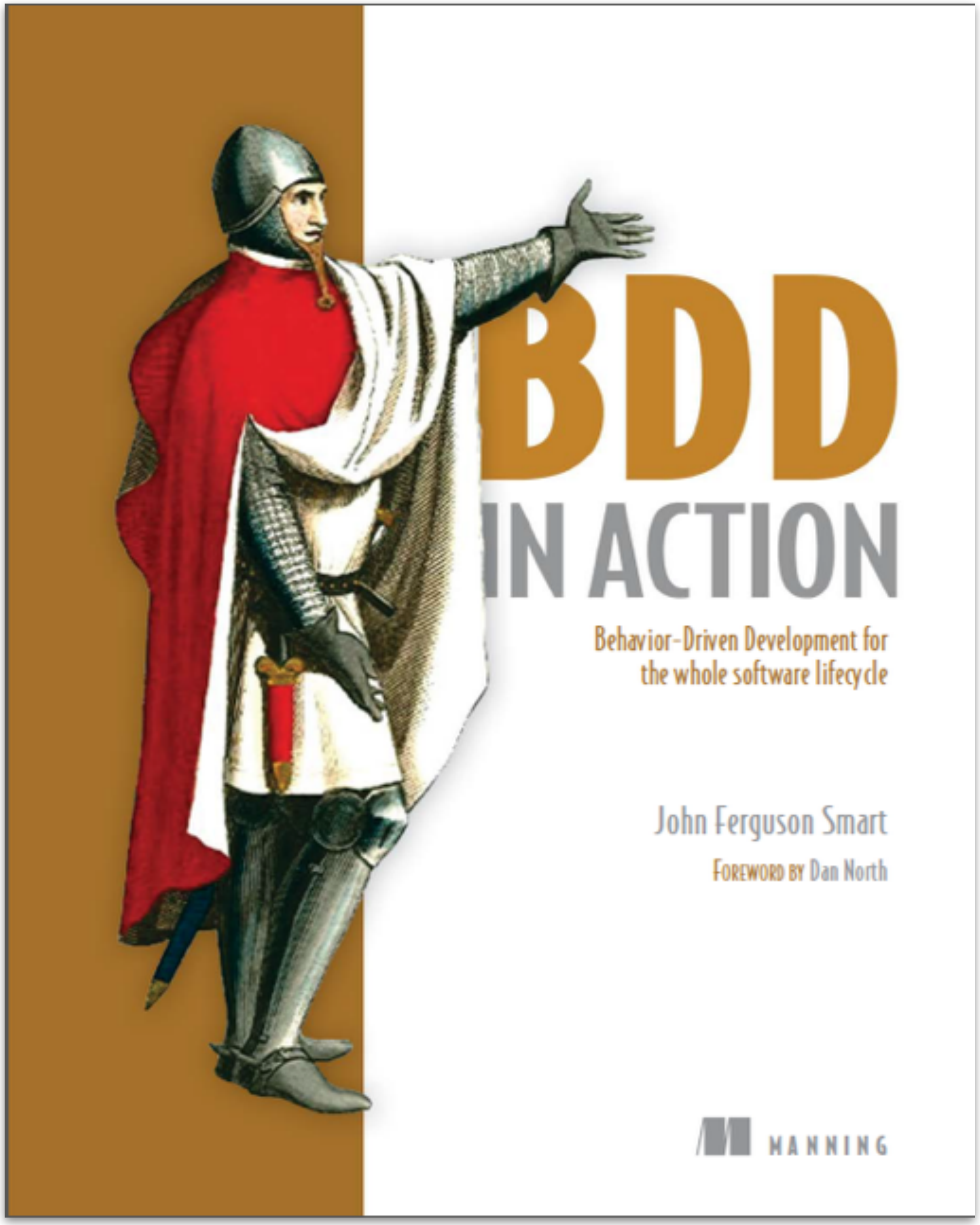


```
Given that Joe wants to register for online banking
When he goes to the registration page
And he enters 'Joe' in the first name field
And he enters 'Bloggs' in the surname field
And he enters 'Joe@bloggs.com" in the email field
And he enters '01/01/1980' in the date of birth field
```

Anti-pattern 4

The scenarios are too UI-centric or detail-focused, and neglect to express the core business value.

Book signings!


BDD IN ACTION
Behavior-Driven Development for the whole software lifecycle
John Ferguson Smart
Foreword by Dan North
MANNING

Digital Guru bookstore

Hilton Hotel, 2nd Level lobby

Thursday, 12:30-1:00pm

http://www.wakaleo.com

john.smart@wakaleo.com

wakaleo