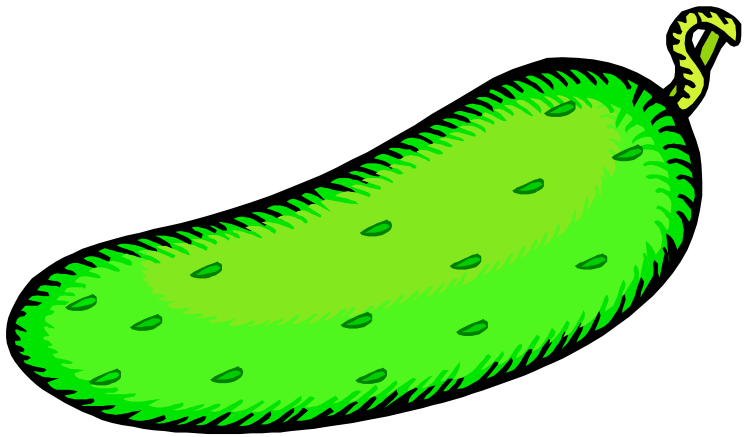
Introducing Cucumber & Capybara

(Engineering Software as a Service §7.5)



David Patterson

User stories => Acceptance Tests?

- Wouldn't it be great to automatically map 3x5 card user stories into tests for user to decide if accept the app?
- How would you match the English text to test code?
- How could you run the tests without a human in the loop to perform the actions?

Cucumber: Big Idea

- Tests from customer-friendly user stories
  - Acceptance: ensure satisfied customer
  - Integration: ensure interfaces between modules consistent assumptions, communicate correctly.
- Cucumber meets halfway between customer and developer
  - User stories not code, so clear to customer and can be used to reach agreement
  - Also not completely freeform, so can connect to real tests

Example User Story

```
Feature: User can manually add movie
```

1 Feature

```
Scenario: Add a movie
```

≥1 Scenarios / Feature

```
  Given I am on the RottenPotatoes home page
  When I follow "Add new movie"
  Then I should be on the Create New Movie page
  When I fill in "Title" with "Men In Black"
  And I select "PG-13" from "Rating"
  And I press "Save Changes"
  Then I should be on the RottenPotatoes home page
  And I should see "Men In Black"
```

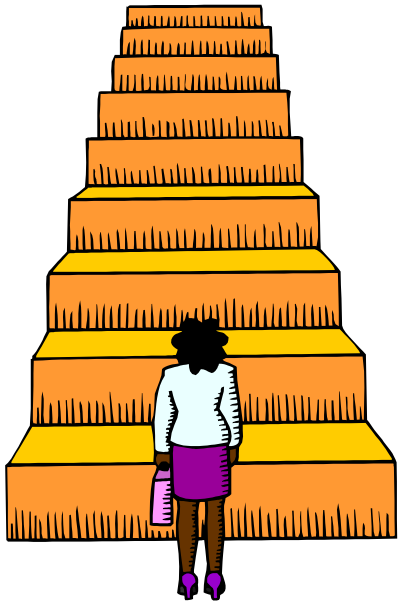
3 to 8 Steps / Scenario

Cucumber User Story, Feature, and Steps

- User story: refers to single feature
- Feature: ≥1 scenarios that show different ways a feature is used
  - Keywords `Feature` and `Scenario` identify respective components
  - Kept in `.feature` files
- Scenario: 3 - 8 steps that describe scenario
- Step definitions: Ruby code to test steps
  - Kept in `X_controller.rb` files

5 Step Keywords

1. `Given` steps represent state of world before event: preconditions
2. `When` steps represent event
   - e.g., simulate user pushing a button
3. `Then` steps represent expected postconditions; check if true
4. / 5. `And` & `But` extend previous step



Steps => Step Definitions via Regular Expressions

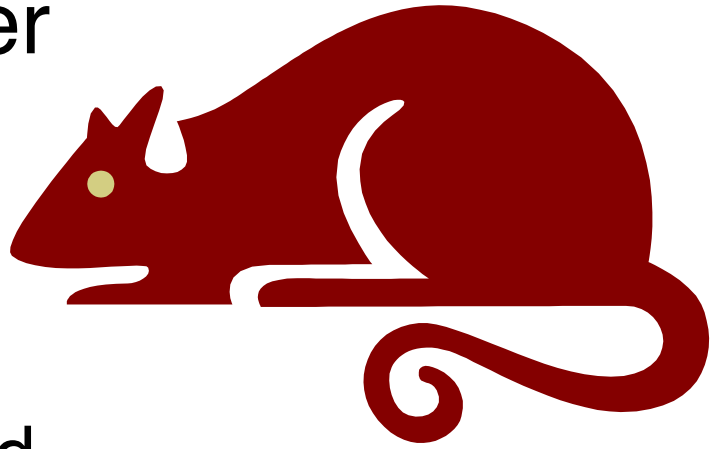
- ***Regexes match English phrases in steps of scenarios to step definitions!***
- `Given /^(?:|I )am on (.+)$/`
- `"I am on the Rotten Potatoes home page"`
- Step definitions (Ruby code) likely use captured string
  - `"Rotten Potatoes home page"`

More on “Cuke”

- Need to install Cucumber Gem
  - Just for test and development environment, not for production environment
- When install Cucumber, creates commonly used step definitions
- Need a test database to run app
- Then edit `.features` file to add features

Fake User to try Scenarios?

- Tool that pretends to be user to follow scenarios of user story
- Capybara simulates browser
  - Can interact with app to receive pages
  - Parse the HTML
  - Submit forms as a user would

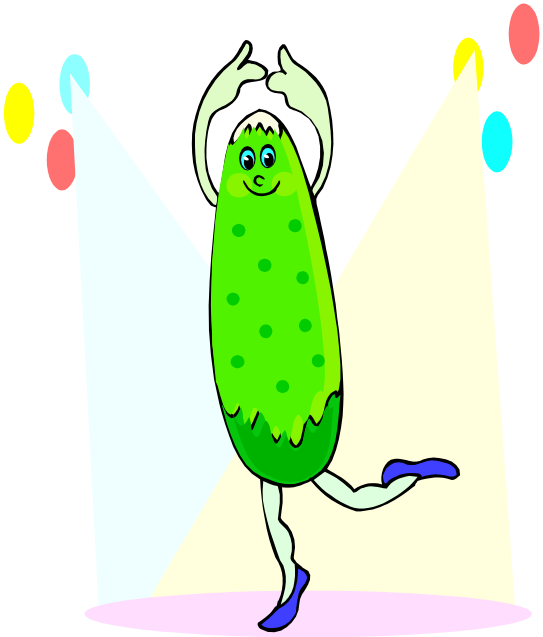


Which is FALSE about Cucumber and Capybara?

1. Step definitions are in Ruby, and are similar to method calls, while steps are in English and are similar to method definitions
2. A Feature has one or more Scenarios, which are composed typically of 3 to 8 Steps
3. Steps use `Given` for current state, `When` for actions, and `Then` for consequences of actions
4. Cucumber matches step definitions to scenario steps using regexes, and Capybara pretends to be user that interacts with SaaS app accordingly

Running Cucumber and Capybara

(Engineering Software as a Service §7.6)

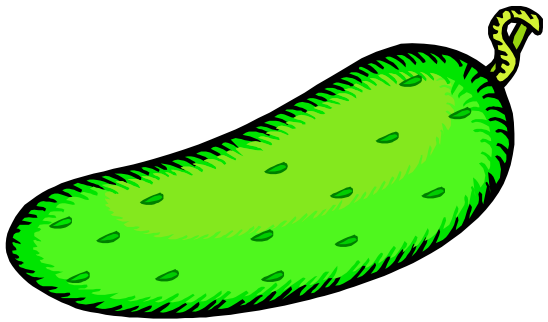


David Patterson

Red-Yellow-Green Analysis

- Cucumber colors steps
- Green for passing
- Yellow for not yet implemented
- Red for failing
  (then following steps are Blue)
- Goal: Make all steps green
  for pass

  (Hence green vegetable
  for name of tool)



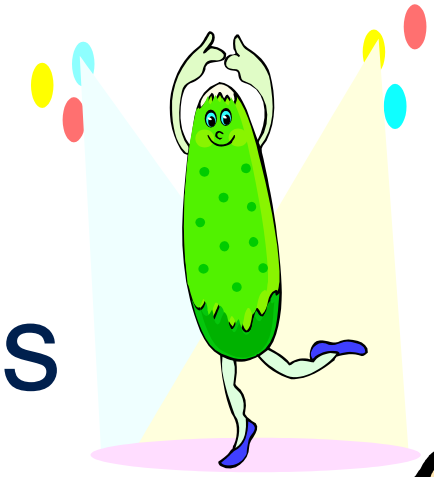
Demo

- Add feature to cover existing functionality
  - Note: This example is doing it in wrong order – should write tests first
  - Just done for pedagogic reasons
- (Or can look at screencast:

  http://vimeo.com/34754747)

Enhancing Rotten Potatoes Again

(Engineering Software as a Service §7.8)



David Patterson

Add a *real* new feature?

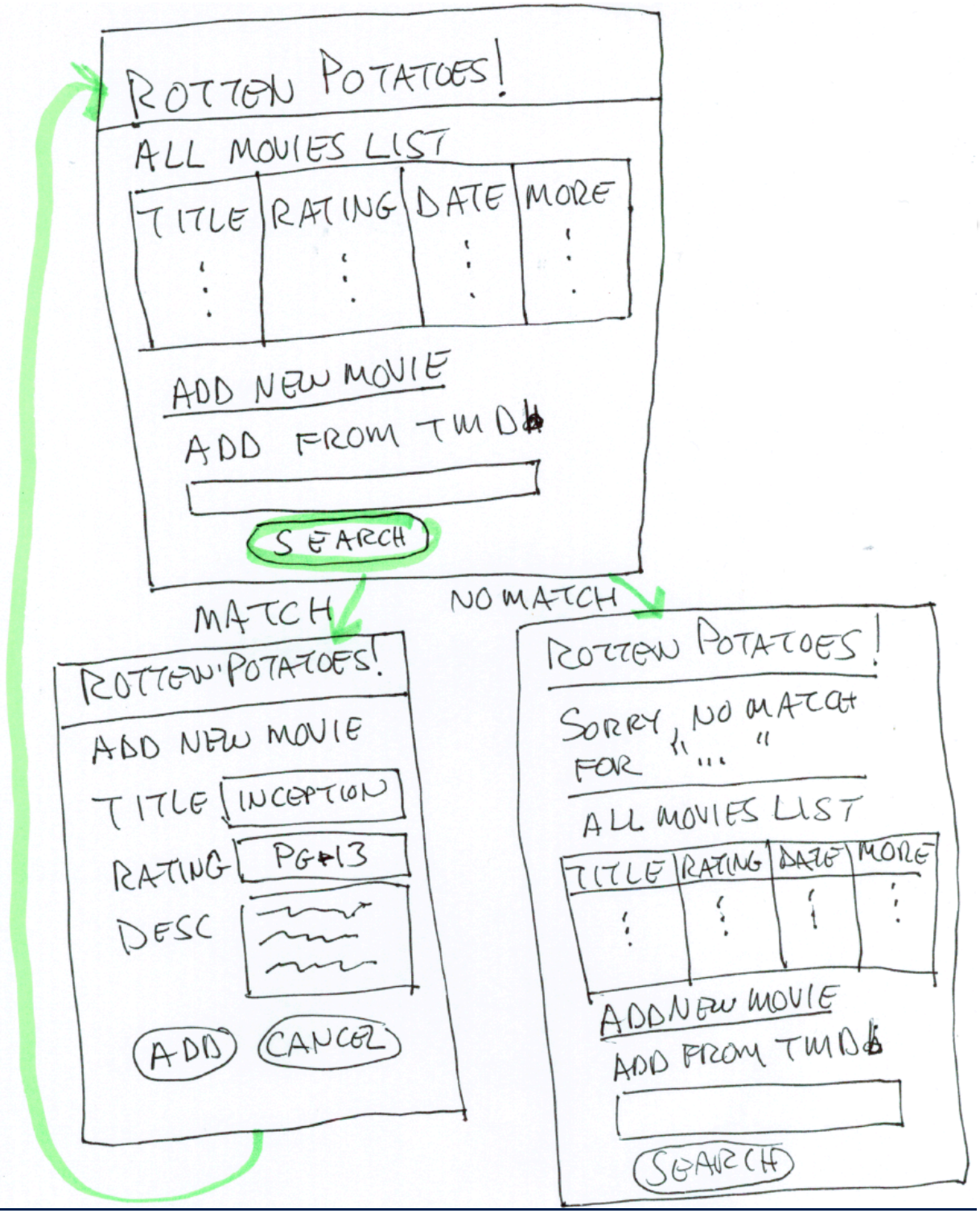
- What if add something harder?
  - e.g., includes form to fill in
  - e.g., needs a User Interface
  - e.g., needs to add route to connect view to controller
  - e.g., includes both a happy path and a sad path

Integrated with The Movie Database (TMDb)

- New Feature: Populate from TMDb, versus enter information by hand
- Need to add ability to search TMDb from Rotten Potatoes home page
- Need LoFi UI and Storyboard

Storyboar TMDb

- Figure 7.6 of *Engineering Software as a Service*



Search TMDb User Story
(Figure 7.7 in *Engineering SaaS*)

```
Feature: User can add movie by searching in The Movie
   Database (TMDb)
   As a movie fan
   So that I can add new movies without manual tedium
   I want to add movies by looking up their details in TMDb
Scenario: Try to add nonexistent movie (sad path)
   Given I am on the RottenPotatoes home page
   Then I should see "Search TMDb for a movie"
   When I fill in "Search Terms" with "Movie That Does Not
   Exist"
   And I press "Search TMDb"
   Then I should be on the RottenPotatoes home page
   And I should see "'Movie That Does Not Exist' was not
   found in TMDb."
```

Haml for Search TMDb page

(Figure 7.8 in *Engineering SaaS*)

```
-# add to end of app/views/movies/
  index.html.haml:

%h1 Search TMDb for a movie

= form_tag :action => 'search_tmdb' do

  %label{:for => 'search_terms'} Search Terms
  = text_field_tag 'search_terms'
  = submit_tag 'Search TMDb'
```

http://pastebin/18yYBVbC

Haml expansion last 2 lines

- This Haml:

```
= text_field_tag 'search_terms'
= submit_tag 'Search TMDb'
```

- Turns into this HTML:

```
<label for='search_terms'>Search Terms</
  label>
<input id="search_terms" name="search_terms"
  type="text" />
```

- `for` attribute of `label` tag matches `id` attribute of `input` tag, from `text_field_tag` helper (above)

Try Cucumber?

- If try Cucumber, it fails
- Missing the route
- Also `MoviesController#search_tmdb` is controller action that should receive form, yet not in `movies_controller.rb`
- Should use Test Driven Development (future lecture) to implement method search_tmdb
- Instead, to finish sad path, add fake controller method that always fails

Trigger Fake Controller when form is POSTed (Figure 7.9)

```
# add to routes.rb, just before or just after
  'resources :movies' :

# Route that posts 'Search TMDb' form
post '/movies/search_tmdb'
```

http://pastebin/FrfkF6pd

Fake Controller Method: Will Fail Finding Movie (Figure 7.9)

```
# add to movies_controller.rb, anywhere inside
#  'class MoviesController <
  ApplicationController':


def search_tmdb
  # hardwired to simulate failure
  flash[:warning] = "'#{params[:search_terms]}'
  was not found in TMDb."
  redirect_to movies_path
end
```

http:/pastebin/smwxv70i

Which statement is TRUE?

1. Usually you complete the Behavior Driven Design phase with Cucumber before starting the Test Driven Development phase with RSpec
2. Usually you code the sad paths first
3. A sad path can pass without having code written needed to make a happy path pass
4. None of the above is true

Running Rotten Potatoes Again

(Engineering Software as a Service §7.8)

David Patterson

Demo

- Add feature to search for movie in TMDb
  - Note: This will be a sad path, in that won't find it
  - Will use fake method
    (until future when implement it using TDD)
- (Or can look at screencast:

  http://vimeo.com/34754766)

Happy Path of TMDb

- Find an existing movie, should return to Rotten Potatoes home page
- But some steps same on sad path and happy path
- How make it DRY?
- `Background` means steps performed before *each* scenario

TMDb with 2 Scen: Background (Fig 7.10) http://pastebin/icQGrYCV

```
Feature: User can add movie by
   searching for it in The Movie
   Database (TMDb)
  As a movie fan
  So that I can add new movies
   without manual tedium
  I want to add movies by
   looking up their details in
   TMDb

Background: Start from the
   Search form on the home page
  Given I am on the
   RottenPotatoes home page
  Then I should see "Search TMDb
   for a movie"

Scenario: Try to add nonexistent
   movie (sad path)
  When I fill in "Search Terms"
   with "Movie That Does Not
   Exist"
  And I press "Search TMDb"
  Then I should be on the
   RottenPotatoes home page
  And I should see "'Movie That
   Does Not Exist' was not found
   in TMDb."

Scenario: Try to add existing
   movie (happy path)
  When I fill in "Search Terms"
   with "Inception"
  And I press "Search TMDb"
  Then I should be on the
   RottenPotatoes home page
  And I should see "Inception"
```

Cucumber Summary

- New feature => UI for feature, write new step definitions, even write new methods before Cucumber can color steps green
- Usually do happy paths first
- Background lets us DRY out scenarios of same feature
- BDD/Cucumber test behavior; TDD/RSpec in folllowing chapter is how write methods to make all scenarios pass

And in Conclusion

- Cucumber – “magically” maps 3x5 card user stories onto acceptance tests and integration tests for app