

What's New in RSpec?

Who am I?

Geoffrey Dagley

McKinney Station

What is RSpec?

- Behavior Driven Development (BDD) Framework
- “RSpec provides a Domain Specific Language with which you can express executable examples of the expected behaviour of a system.”
- <http://rspec.info>

What's New?

- Shared examples
- Nested specs
- Stories

Shared Examples

- Previously

```
describe "All things", :shared => true
```

- Now

```
shared_examples_for "All things"
```

Shared Examples

(as modules)

- Now

```
share_as :AllThings
```

- And later

```
describe "Something" do  
  include AllThings  
  ... #all your other specs  
end
```

- SEE `rspec/examples/shared_example_group_example.rb`

Nested Example Groups

- Better organization of specs
- Define setup for all nested example groups
- Define additional setup in each example group
- `see rspec/examples/
stack_spec_with_nested_example_groups.rb`

Stories

- Specs are to Unit testing what Stories are Integration testing.
- BDD at a higher level.
- Conversational like specs.
- Started life as `rbehave` by Dan North.
- SEE `http://rspec.info/documentation/stories.html`

Parts of a Story

- **Title** - One line to describe the story
- **Narrative** - Description of user, role, and benefit

As a role

I want feature

So that value

- **Scenario** - A story can have many scenarios
 - **Given** - The context of the scenario (setup)
 - **When** - The event in the scenario (usually I event)
 - **Then** - The outcome (verification of the event behaviour)
 - **And** - Additional context or outcomes

Codifying the Story

```
Story "title", "narrative" do
  Scenario "description" do
    Given "some context" do
      # some setup
    end
    When "some event" do
      # trigger the event
    end
    Then "some outcome" do
      # verify the outcome
    end
  end
end
end
```

Simplifying the Story

- Put code elsewhere and leave description intact

```
class SomeSteps < Spec::Story::StepGroup
  steps do |some|
    some.given("some context") do
      # some setup
    end
    some.when("some event") do
      # trigger the event
    end
    some.then("some outcome") do
      # verification of the event
    end
  end
end
end
```

Simplifying the Story

(continued)

```
steps = SomeSteps.new
Story "title", "narrative", :steps => steps do
  Scenario "description" do
    Given "some context"
    When "some event"
    Then "some outcome"
  end
end
```

The Story in Plain Text

Story: title

narrative

Scenario: description

Given some context

When some event

Then some outcome

Running the Story

```
run_story do |runner|  
  runner.steps << SomeSteps.new  
  runner.load #PATH/TO/PLAIN_TEXT_STORY  
end
```

Questions?

- <http://rspec.info>
- <http://www.mckinneystation.com>