

# Test Driven Design using

ZenTest



by: Ryan Davis, Seattle.rb

# Old Skool

1. Blake it run.

2. Make it right!

3. Make it fast.

# Old Skool is too Old

- aka “WaterFall”
- Exemplifies the “Over the Wall” QA mentality of yesteryear:
  - Do engineering (for too long), then send an amorphous blob of gunk to QA for testing in a vacuum.
  - Lacks tight feedback loops necessary to close bugs fast enough.
  - ...is most often the thing cut to hit ship dates.
- These are **real**, even **prevalent**.
- I worked on one just *6 months ago!*

# New Skool: Test Driven Design

*1. Make it right.*

2. Make it run.

3. Make it fast.



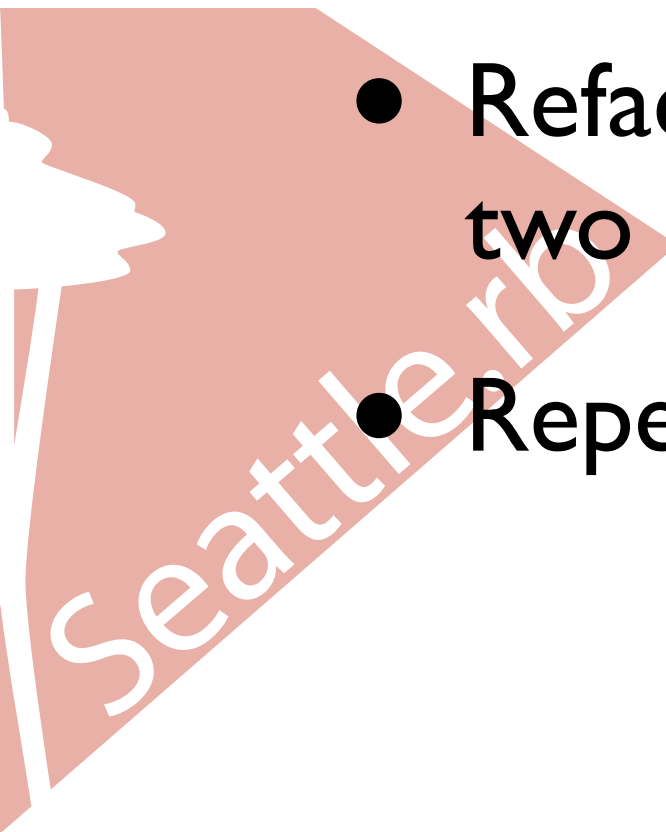
# New Skool: Test Driven Design

- aka “Test First”
- Drives design decisions through testing.
  - Ensures things *are* testable.
  - Provides a fast and tight feedback loop.
- Attacks the problem at the specification level.
- Doesn't wait until the last minute to see if things work.



# TDD Mantra

- Go Red - produce failing testcase
  - Failures should make you *happy!*
- Go Green - implement and pass test
- Refactor - clean up - made safer by previous two steps
- Repeat until done



# Our Driving Philosophy

- Make it: Right, Run, Fast (*enough*).
- Simplicity & Clarity
  - Otherwise, you don't *fully* understand.
- Speed comes ***last***, and only objectively.
- Balance size and speed.
- Accelerate Maturity by splitting from parents.
- ***Have fun!*** Otherwise, what's the point?



# A *Very* Quick Intro to TDD

- A Special Thanks to Steven Baker:
  - He ***nailed it*** at Canada on Rails.
  - Rewriting these slides felt like a violation of DRY.





```
class AccountTest < Test::Unit::TestCase
  def test_new_account_should_have_zero_balance
    account = Account.new

    assert_equal 0, account.balance
  end
end
```

E

1) Error:

test\_new\_account\_should\_have\_zero\_balance:

NameError: uninitialized constant Account

./test/unit/account\_test.rb:6:in

`test\_new\_account\_should\_have\_zero\_balance'

1 tests, 0 assertions, 0 failures, 1 errors

```
class CreateAccount < ActiveRecord::Migration
  def self.up
    create_table "accounts", :force => true do |t|
      t.column :balance, :integer, :default => 0
    end
  end
end
```

```
class Account < ActiveRecord::Base
end
```

Started

.

Finished in 0.055268 seconds.

1 tests, 1 assertions, 0 failures, 0 errors

```
class DepositTest < Test::Unit::TestCase
  def test_should_update_balance_when_making_deposit
    account = Account.new
    account.deposit(10)
    assert_equal 10, account.balance
  end
end
```

.E

1) Error:

```
test_should_update_balance_when_making_deposit(DepositTest):
NoMethodError: undefined method `deposit' for #<Account...>
  ./test/unit/deposit_test.rb:8:in
`test_should_update_balance_when_making_deposit'
```

2 tests, 1 assertions, 0 failures, 1 errors

```
class Account < ActiveRecord::Base

  def deposit(amount)
    end

end
```

.F

1) Failure:

test\_should\_update\_balance\_when\_making\_deposit:  
<10> expected but was  
<0>.

2 tests, 2 assertions, 1 failures, 0 errors

```
class Account < ActiveRecord::Base

  def deposit(amount)
    self.balance += amount
  end

end
```

Started

..

Finished in 0.242602 seconds.

2 tests, 2 assertions, 0 failures, 0 errors

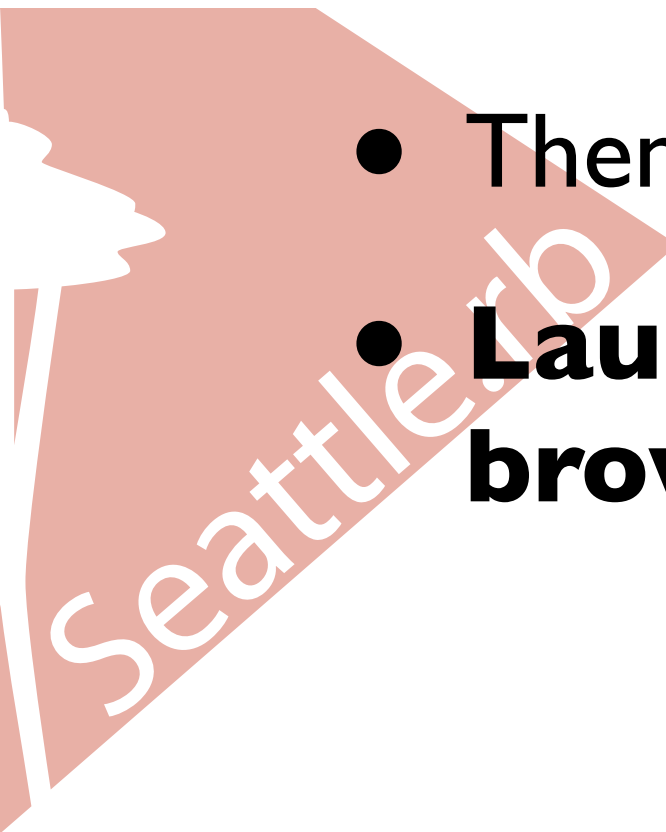
It *really is* that simple...





# No Peekie TDD

- Rubyholic.com is a ruby group registration system written in about a week.
- Went to private beta -- all bugs found were from holes in our tests.
- Then went to designers to make it pretty.
- **Launched without ever loading in a browser.**

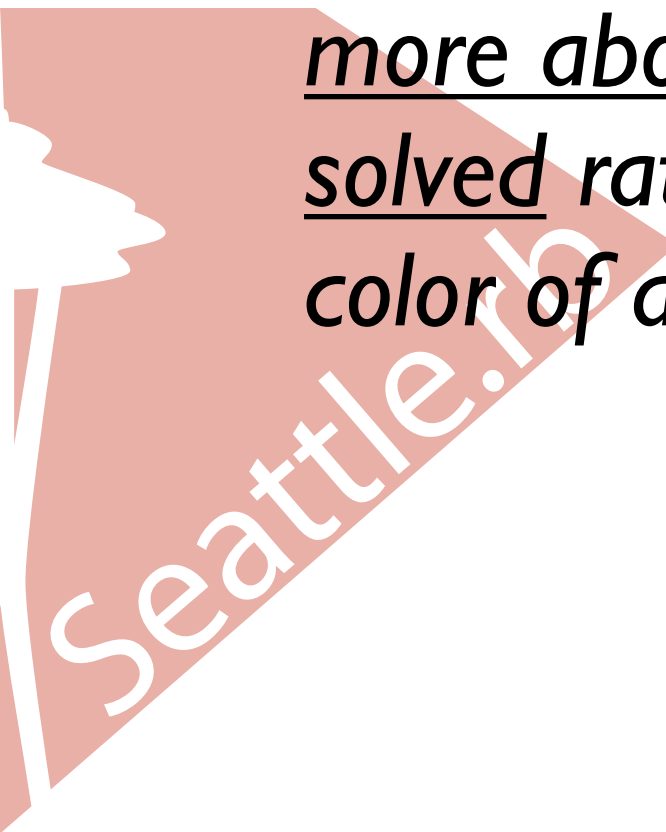




# No Peekie TDD

*“The revolutionary part of this is that it speeds development by helping you develop without needing to open your web browser! I find myself thinking more about the functional issues that need to be solved rather than the placement of an image or the color of a link.”*

- Geoffrey Grossenbach







# ZenTest

- Suite of 5 tools:
  - ZenTest - Accelerate TDD and audit code.
  - unit\_diff - Clarify errors using diff.
  - autotest - Test feedback on turbo.
  - multiruby - Execute multiple rubies.
  - Test::Rails - Enhanced rails testing library.

# zentest

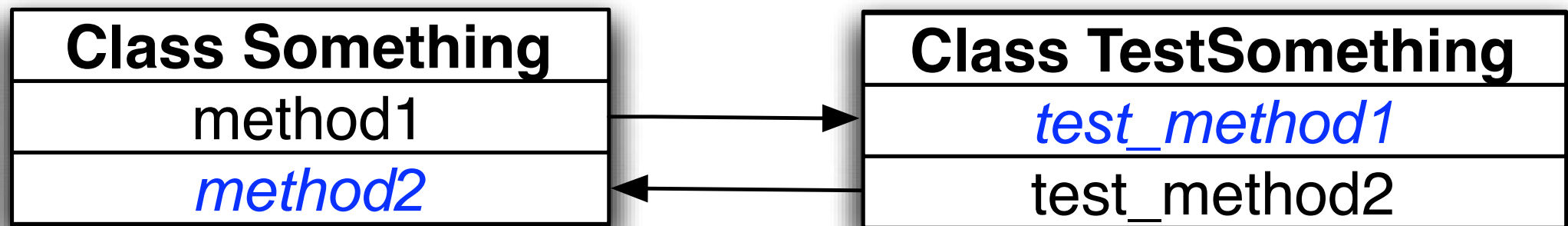
- Bidirectional auditing and code generation.
- Can automatically evaluate generated code to accelerate TDD.
- Measures assertion to method ratio.

ZenTest



# ZenTest

## Generating Tests





ZenTest

# ZenTest in Action

```
575 % cat test_factorial.rb
```

# unit\_diff

## Illuminating Issues

ZenTest

& ./test.rb

...

1) Failure:

test\_conditional4(TestTypeChecker)

[./r2ctestcase.rb:1068:in `test\_conditional4'

./r2ctestcase.rb:1055:in `test\_conditional4'1.

<t(:if,

t(:call,

t(:lit,

==,

t(:arg]

t(:return

t(:if,

t(:call

t(:lit

:>,

t(:arg

t(:return, t(:lit, 3, Type.long), Type.void),

t(:return, t(:lit, 4, Type.long), Type.void), Type.void),

Type.void)> expected but was

<t(:if,

t(:call,

Or this:

& ./test.rb | unit\_diff

...

1) Failure:

test\_conditional4(TestTypeChecker)

12c10

< :>,

---

> :<,



# unit\_diff in Action

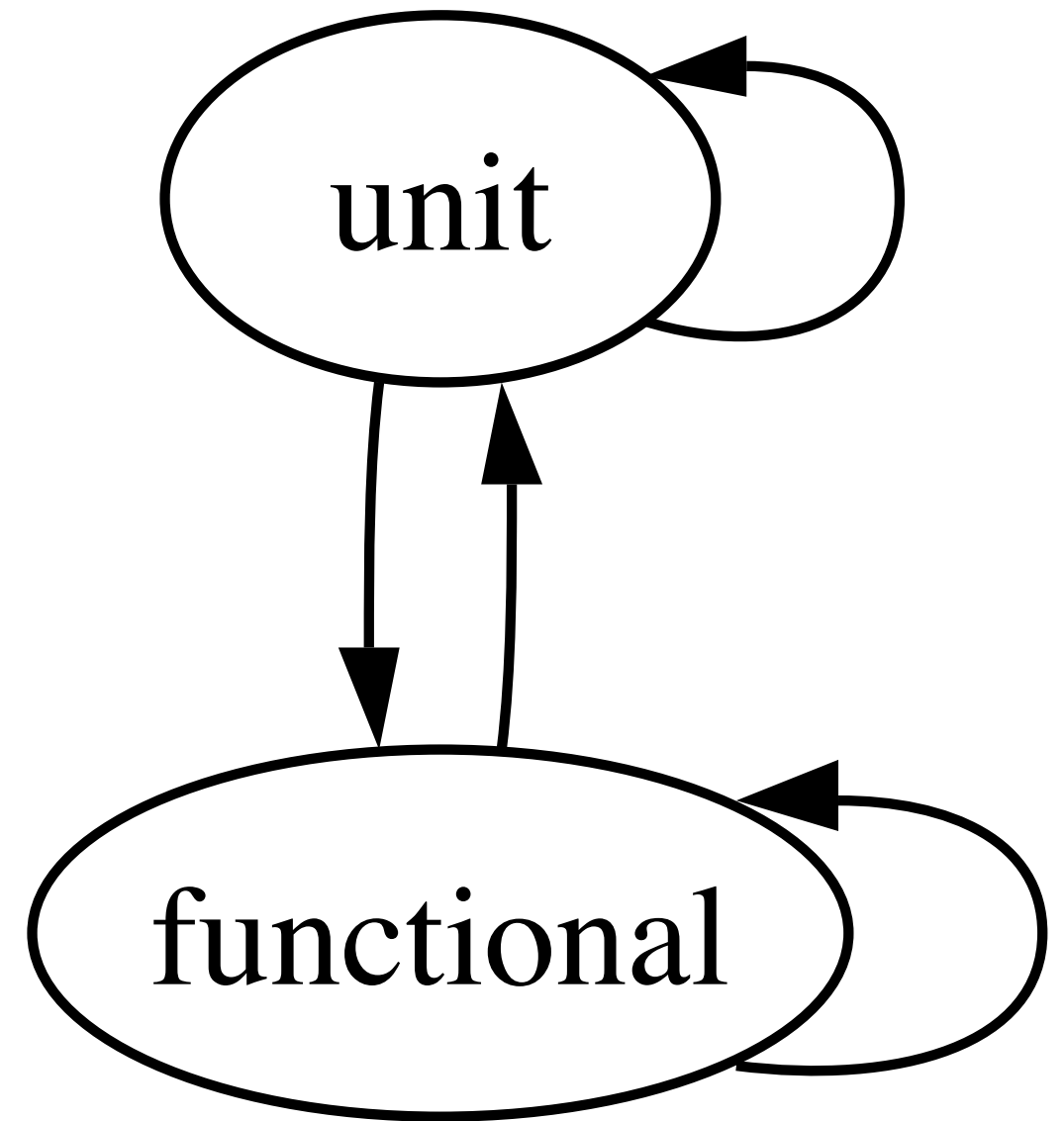
```
$ make
ruby -w -I:../ParseTree/dev/lib:../ParseTree/dev/test:../RubyInline/de
v test_all.rb
Loaded suite test_all
Started
.....
.....
.....
.....
.....
Finished in 3.96993 seconds.

377 tests, 841 assertions, 0 failures, 0 errors
$ █
```



# autotest

- Intelligent Fast Feedback - run the right tests every time you save.
- Focus on Failures - reruns failed tests until they pass.



# AutoTest in Action

```
File Edit Options Buffers Tools P4 Help
```

```
assert_equal("test_method", @tester.normal_to_test("method"))
assert_equal("test_method_equals", @tester.normal_to_test("method="))
assert_equal("test_spaceship", @tester.normal_to_test("<="))
assert_equal("test_times", @tester.normal_to_test("*"))
assert_equal("test_times2", @tester.normal_to_test("**"))
assert_equal("test_unary_minus", @tester.normal_to_test("@-"))
assert_equal("test_unary_plus", @tester.normal_to_test("@+"))
assert_equal("test_class_index", @tester.normal_to_test("self.[]"))
end
```

```
-uu-:--F1 test zentest.rb 59% (507,0) (Ruby Abbrev)-----
```

```
Loaded suite -e
```

```
Started
```

```
.....
```

```
Finished in 0.263702 seconds.
```

```
44 tests, 140 assertions, 0 failures, 0 errors
```

```
# Passed
```

```
# All passed
```

```
# Waiting for changes
```

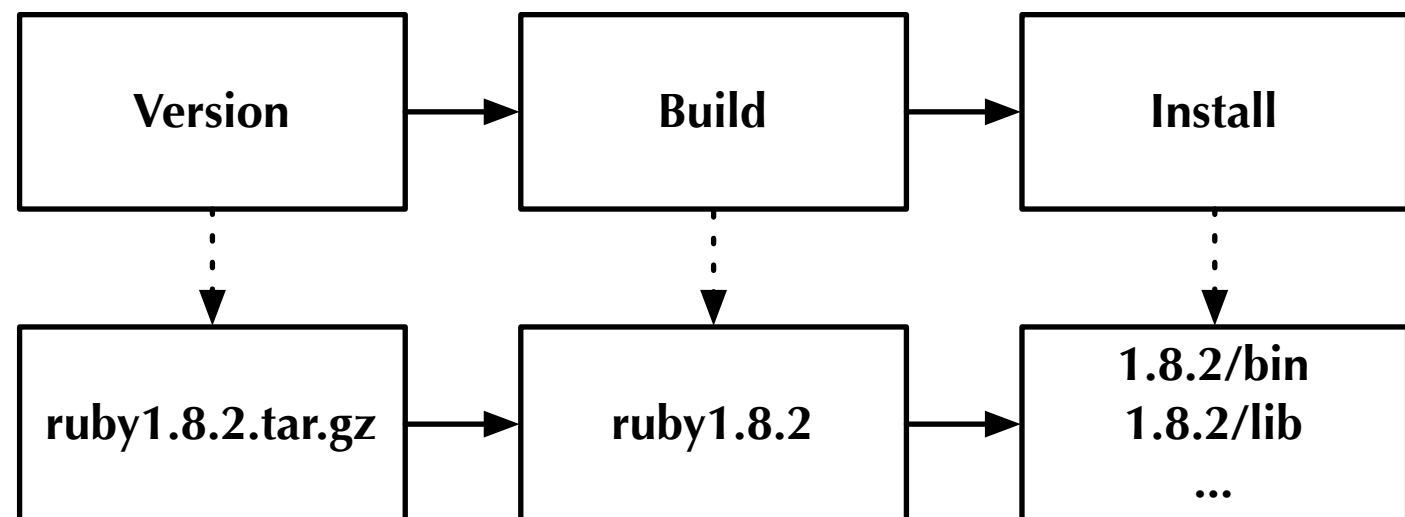
```
-uu1:**-F1 autotest Bot (366,0) (Shell:run Shell-Compile)-----
```

```
Quit
```



# multiruby

- Multiruby runs your command through multiple versions of ruby.
- Automatically configures, builds, and privately installs any number of ruby versions.
  - Just drop in a tarball to add a version.
  - Essential for extension writers and compatibility testing.





# MultiRuby in Action

```
511 % multiruby -w -Ilib:bin:../../RubyInline/dev test/test_all.rb
```

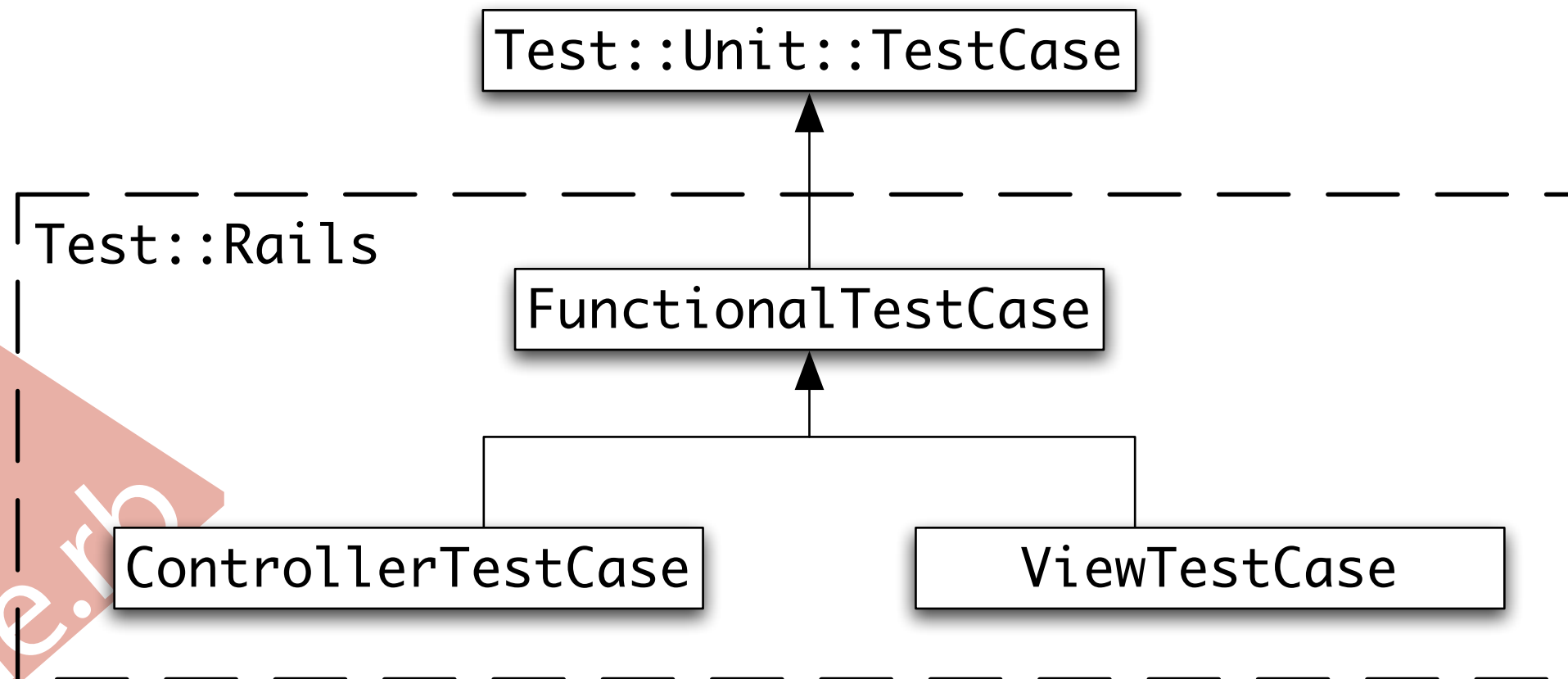
# Test::Rails

- Build Industrial-Strength Rails Code
  - Decouple controller tests from view tests.
  - Automatically audit consistency between the two.
- Test single partials / AJAX actions
- Extended suite of assertions

ZenTest



# Test::Rails



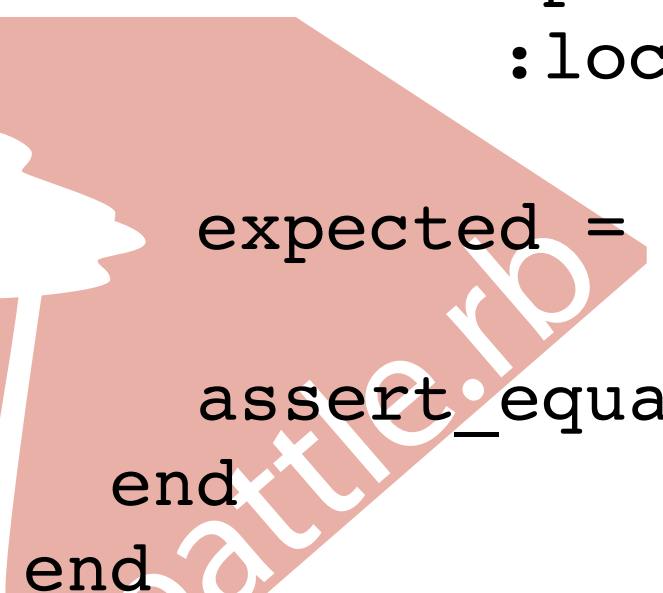
# Test::Rails in “Action”

```
class RootControllerTest < Test::Rails::ViewTestCase
  fixtures :points

  def test_point_partial
    render :partial => 'point',
           :locals  => { :point => points(:home) }

    expected = "<mk lat=\"-122.0\" lon=\"47.0\" id=\"100\"/>"

    assert_equal expected, @response.body
  end
end
```

A red, semi-transparent logo for 'Seattle.rb' is positioned in the bottom-left corner. It features a stylized white outline of the state of Washington, with the text 'Seattle.rb' written in a white, sans-serif font across it.



**Thank You**

seattle.rb

<http://www.zenspider.com/seattle.rb>