RubyUnit: Unit Testing in Ruby

Premshree Pillai
Yahoo! Bangalore
ppillai@yahoo-inc.com

Unit Testing: The Idea

- User stories
- Limited number of user stories implemented at a time
- Unit tests written before code
- Tests form specification for code

Ruby and Testing: Looking at objects

```
Class Myclass
    ...
end

obj = Myclass.new

ObjectSpace.each_object(Myclass) { |x| p
x }
```

Returns `<Myclass:0x812e658>`

Ruby and Testing: Looking inside objects

```
Class Myclass
    def foo
        ...
    end
    def bar
        ...
    end
end

obj = Myclass.new
obj.methods
```

Returns ["foo", "bar"]

Ruby and Testing: Looking inside objects (contd.)

- `obj.respond_to?("method_to_test")`
- `obj.id`
- `obj.class`
- `obj.kind_of? Type_to_test`
- `obj.instance_of? Class_to_test`

Ruby and testing: Classes and Methods

- Classes
  - `Myclass.private_instance_methods`
  - `Myclass.protected_instance_methods`
  - `Myclass.public_instance_methods`
  - `Myclass.singleton_methods`
  - `Myclass.constants`
  - `Myclass.superclass.constants`
- Calling methods dynamically
  - *`Myclass_obj`*`.send("`*`method_name`*`", [,])`
  - *`Myclass_obj`*`.method("`*`method_name`*`", [,])`

Unit testing modules for Ruby

- Test ::Unit (bundled with Ruby)

```
require 'test/unit'
class TC_ClassToTest < Test::Unit::TestCase
   def test_method_to_test
     assert_equal("expected output",
                  "actual_output",
                  "some_string")
   end
end
```

Software to build the test suite skeleton available from Zen Spider

- RubyUnit

Mock objects

```
mock = Object.new
mock.extend(Test::Unit::Assert)
def mock.method_called_by_testee
     assert_equal("expected output",
"actual_output","some_string")
end

testObject.something_with(mock)
```

RubyUnit

- Initial setup:

```
require "runit/testcase"
require 'runit/cui/testrunner'
require 'runit/testsuite'
require "myclass" #Class Under Test (CUT)
```

- Test class:

```
class Testing_class < RUNIT::TestCase
      ...
end
```

Testing_class is the test class (TC)

RubyUnit (contd.)

- Methods of test class begin with `test_`

```
class Testing_class < RUNIT::TestCase
      def test_mytest1
            ...
      end
end
```

- Methods of RUNIT::TestCase

  - assert_fail(message)
  - assert(boolean, message="")
  - assert_equal(expected, actual, message="")
  - assert_equal_float(expected, actual, e, message="")
  - assert_same(expected, actual, message="")
  - assert_nil(obj, message="")
  - assert_not_nil(obj, message="")
  - assert_respond_to(method, obj, message="")

RubyUnit: RUNIT::TestCase methods

- Methods of RUNIT::TestCase (contd.)
  - `assert_kind_of(c, obj, message="")`
  - `assert_instance_of(c, obj, message="")`
  - `assert_match(str, re, message="")`
  - `assert_not_match(str, re, message="")`
  - `assert_exception(exception, message="") {block}`
  - `assert_no_exception(*arg) {block}`
  - `assert_operator(obj1, op, obj2, message="")`
  - `assert_send(obj1, op, *args)`

RubyUnit: Running Tests

- Running the tests
  - Running a specific test:

    ```
    RUNIT::CUI::TestRunner.run(Testing_class.new("some_test_method"))
    ```

  - Running the suite:

    ```
    RUNIT::CUI::TestRunner.run(Testing_class.suite)
    ```

RubyUnit: Sample template test code

```
if __FILE__ == $0
    require 'runit/testcase'
    require 'runit/cui/testrunner'
    require 'runit/testsuite'

    class  Testing_class < RUNIT::TestCase
        def test_method1
            mine = My_class.new
            ...
        end
        ...
    end

    RUNIT::CUI::TestRunner.run(Testing_class.suite)
end
```

Building an app using RubyUnit: code walkthrough

```
require 'runit/testcase'
require 'runit/cui/testrunner'
require 'runit/testsuite'

class  TC_NewlineToHtmlBreak < RUNIT::TestCase
     def test_NewlineToHtmlBreak
          nl2br = NewlineToHtmlBreak.new
          assert_equal("premshree\nis cool",
nl2br.convert, "\
          Newlines should have been converted to HTML
breaks")
     end
     RUNIT::CUI::TestRunner.run(TC_NewLineToHtmlBreak.su
ite)
end
```

Building an app using RubyUnit: code walkthrough (contd.)

```
class NewlineToHtmlBreak
    def convert(str)
        return str.gsub!("\n", "<br />")
    end
end
```

Building an app using RubyUnit: code walkthrough (contd.)

```
require 'runit/testcase'
require 'runit/cui/testrunner'
require 'runit/testsuite'
require 'NewlineToHtmlBreak'

class  TC_NewlineToHtmlBreak < RUNIT::TestCase
     def test_NewlineToHtmlBreak
          nl2br = NewlineToHtmlBreak.new
          assert_equal("premshree\nis cool",
nl2br.convert, "\
          Newlines should have been converted to HTML
breaks")
     end
     RUNIT::CUI::TestRunner.run(TC_NewLineToHtmlBreak.su
ite)
end
```

Unit testing web apps

- MVC architecture
  - Decoupling models and controllers – mock objects
  - Automated feeds to controllers (from views)
  - Method interception

  Ref: Ruby on Rails

References

- Ruby language: http://ruby-lang.org/
- Test::Unit: http://testunit.talbott.ws/
- RubyUnit: http://homepage1.nifty.com/markey/ruby/rubyunit/index_e.html
- ZenTest: http://www.zenspider.com/ZSS/Products/ZenTest
- Ruby on Rails: http://www.rubyonrails.com/
- HttpUnit: http://httpunit.sourceforge.net/
- XP: http://www.xprogramming.com/

Thank You!
Questions?