

Automatic Markup Validation

Aron Atkins

aron@spotstory.com

617.335.2871

Boston Ruby Group

April 10, 2007

What's the story?

- Mechanical validation of many formats
 - XHTML/HTML
 - CSS
 - RSS
 - etc.
- Checking for legal markup reduces risk
 - Dangling div, td, etc.

Available Plugins

- `assert_valid_markup`
 - Uses W3 Validator to check XHTML/HTML
 - A local server can improve performance
- `assert_valid_asset`
 - Built upon `assert_valid_markup`
 - Adds CSS validation using W3
- RSS parser built into Ruby
 - Future could use public validators (ATOM, etc)
- Other XML formats? DTD?

Give a little helper ...

- Install `assert_valid_markup`
- Install the `test_validation_helper.rb`
 - <http://blog.spotstory.com/2007/04/09/automatic-markup-validation>
- `test/test_helper.rb`

```
require File.expand_path(  
  File.dirname(__FILE__) +  
  "/test/test_validation_helper.rb")
```

Don't be shy ...

- test/test_validation_helper.rb

```
alias :non_validating_get :get  
alias :non_validating_post :post
```

```
def get(path, parameters=nil, headers={}, flash=nil)  
  non_validating_get path, parameters, headers, flash  
  check_markup  
end
```

```
def post(path, parameters=nil, headers={}, flash=nil)  
  non_validating_post path, parameters, headers, flash  
  check_markup  
end
```

Validation Dispatch

- test/test_validation_helper.rb

```
def check_markup
  return if ! ENV['NO_VALIDATION'].nil?
  if @response.redirect?
  elsif javascript_response? || json_response? || kml_response?
  elsif xml_response?
    assert_rss_valid
    print "+"
  else
    assert_html_valid
    print "+"
  end
end
end
```

HTML Validation

- test/test_validation_helper.rb

```
def assert_html_valid
  begin
    assert_valid_markup
  rescue
    print "\nVALIDATION FAILED:\nHEADERS:\n"
    @response.headers.each { |hdr, val|
      print "\t#{hdr}: #{val}\n" }
    print "BODY:\n"
    print @response.body + "\n"
    raise # re-raise after showing the failing body.
  end
end
```

RSS Validation

- Using the Ruby builtin module

```
def assert_rss_valid
  assert_not_nil @response.body

  # Ignore unknown elements because the built-in Ruby RSS
  # support doesn't fully understand extensions.
  RSS::Parser.parse(@response.body, true, true)

  # Make sure all the objects have been processed.
  assert_no_match /cannot be processed/, @response.body
end
```

Get the code

- Posted on the Spotstory Blog:
<http://blog.spotstory.com/2007/04/09/automatic-markup-validation/>
-
- Email:
aron@spotstory.com