The Agent

```
require 'rubygems'
require 'mechanize'

agent = WWW::Mechanize.new

# disable keep_alive, when running into
# problems with session timeouts or getting an
# EOFError
agent.keep_alive = false

# Setting user agent:
agent.user_agent = 'Friendly Mechanize Script"

# Using one of the predefined user agents:
# 'Mechanize', 'Mac Mozilla', 'Linux Mozilla'.
# 'Windows IE 6', 'iPhone', 'Linux Konqueror',
# 'Windows IE 7', 'Mac FireFox', 'Mac Safari',
# 'Windows Mozilla'
agent.user_agent_alias = 'Mac Safari'

# To verify server certificates:
# (A collection of certificates is available
# here: http://curl.haxx.se/ca/ )
agent.ca_file = 'cacert.pem'

# Don't follow HTTP redirects
agent.redirect_ok = false

# Follow refresh in meta tags
agent.follow_meta_refresh = true

# Enable logging
require 'logger'
agent.log = Logger.new('mechanize.log')
```

Navigation/History

```
# load a page
agent.get('http://the.internet.net')

# Go back to the last page:
agent.back

# Follow a link by its text
agent.link_with(:text => 'click me').click

# Backup history, execute block and
# restore history
agent.transact do
  ...
end
```

Accessing page elements

```
# The current page
agent.page

# The HTML page content
page.body

# forms, links, frames
page.forms, page.links, frames

# Selecting by criteria follows the pattern:
# page.element(s)_with(:criteria => value)
# The plural form (.elements) returns an
# array, the singular form (.element) the
# first matching element or nil. Criteria
# is an attribute symbol and value may be
# a string or a regular expression. If no
# criteria attributr is given, :name will
# be used. e.g.:
page.form_with(:name => 'formName')
page.form_with('formName')
page.links_with(:text => /[0-9]*/
```

Ruby / Mechanize

http://mechanize.rubyforge.org/mechanize/

Nokogiri

http://nokogiri.org/

Hello Mechanize!

```
require 'rubygems'
require 'mechanize'

agent = WWW::Mechanize.new
agent.get('http://rubyforge.org/')

agent.page.forms.first.words = 'mechanize'
agent.page.forms.first.click_button

agent.page.link_with(:text => /WWW::Mechanize/).click
agent.page.link_with(:text => 'Files').click

links = agent.page / 'strong/a'
version = links.find do |link|
  link['href'] =~ /shownotes.*release_id/
end.text

puts "Hello Mechanize #{version}!"
```

Forms

```
# Submitting a form without a button:
form.submit

# Submitting a form with the default button
form.click_button()

# Submitting a form with a specific button
form.click_button(form.button_with(:name => 'OK')

# Form elements
form.fields, form.buttons, form.file_uploads, form.radio_buttons,
form.checkboxes

# Form elements can be selected just like page elements
# form.element(s)_with(:criteria => value)
# e.g.:
form.field_with(:name => 'password')
form.field_with('password')
form.checkboxes(:value => /view_.*/)

# Field values can also be selected directly by their name
form.password = 'secret'

# Setting field values
# field       : .value = 'something'
# checkbox    : .(un)check / .checked = true|false
# radio_button: .(un)check / .checked = true|false
# file_upload : .file_name = '/tmp/upload.dat'
# e.g.:
form.field_with('foo').value = 'something'
form.checkbox_with(:value => 'blue').uncheck
form.radio_buttons[3].check

# Select lists / drop down fields:
form.field_with('color').option[2].select
form.field_with('color').options.find{|o| o.value == 'red'}.select
form.field_with('color').select_none
form.field_with('color').select_all
```

Parsing the page content

```
# Selecting elements from the documents DOM
nodes = agent.page.search('expression')
nodes = agent.page / 'expression'

# Selecting the first matching element or nil
node = agent.page.at('expression')

# 'expression' might be an XPath or CSS selector
nodes = agent.page.search('//h2/a[@class="title"]')
nodes = agent.page.search('.h2 a.title')

# navigating the document tree:
node.parent
node.children

# node content and attributes
node.text
node.inner_html
node.attributes['width']

# found nodes, can be searched the same way
rows = agent.page / 'table/tr'
value = rows[0].at('td[@class="value"]').text
```