Navigation

> visit

visit navigates to a particular path. Pass a string or use one of Rails' path helpers.

```
visit '/blog'
visit blogs_path
```

> click_on

click_on will click an anchor tag, button, or input with type submit. Pass a string containing the anchor text.

```
click_on 'Sign in'
click_on 'Submit'
```

Page Interaction and Scoping

> has_css?

has_css? returns a boolean value reporting whether a specific selector is present on the page.

```
page.has_css?('nav[data-role="primary-navigation"] li', text: 'FAQ')
```

> within

within will scope interaction to within a particular selector. Useful if you're looking for content in a particular area.

```
within('footer') { expect(page).to have_content('Copyright') }
```

> has_content?

has_content? returns a boolean value reporting whether specific content is present on the page.

```
page.has_content?('Sign in')
```

> find

find returns a single Capybara::Node::Element instance from the page.

```
page.find('.todos li:first-child')
```

> all

all returns an array of Capybara::Node::Element instances from the page.

```
page.all('.todos li:nth-of_type(odd)')
```

Page Assertions

Note: All page assertions can be nested within `within` any number of times.

> have_css

`have_css` asserts that a certain selector is present on the page. `have_css` accepts CSS3 and is incredibly powerful.

```
expect(page).to have_css('header')
expect(page).to have_css('table#records + .pagination a[rel="next"]')
```

> have_content

`have_content` asserts that certain text is present on the page.

```
expect(page).to have_content('What are you looking for?')
```

Node Interactions

> click

`click` triggers a click on a `Capybara::Element`. Works with Javascript drivers.

```
find('article a.title').click
```

> trigger

`trigger` allows triggering of custom events. Works with Javascript drivers.

```
find('input[name="post[title]"]').trigger('focus')
```

> visible?

`visible?` returns a boolean value reporting if the `Capybara::Element` is visible. Works with Javascript drivers.

```
find('.navigation').visible?
```

Form Interactions

> fill_in

`fill_in` fills in fields for you. Pass the label text or the name of the input.

```
fill_in 'Title', with: 'I love Rails!'
fill_in 'post[title]', with: 'I love Rails!'
```

> check

`check` checks a checkbox. Pass the label text.

```
check 'I accept the terms of the site'
check 'I am thirteen years of age or older'
```

> uncheck

`uncheck` unchecks a checkbox. Pass the label text.

```
uncheck 'Admin access?'
```

> select

`select` selects an option from a select tag.

```
select 'Moderate', from: 'Political Party'
select 'MA', from: 'State'
```

> click_button

`click_button` will press a `button` or `input[type='submit']`

```
click_button 'Create My Account'
click_button 'Save Record'
```

Debugging

> save_and_open_page

`save_and_open_page` will save the current page (typically to `Rails.root/tmp`) and attempt to open the HTML in the default web browser.