

Click

```
click_button 'text on button'
```

```
click_button 'id'
```

```
click_button 'title'
```

```
click_button 'value'
```

```
click_on 'link_or_button_text'
```

```
click_link 'link_id'
```

```
click_link 'link_title'
```

```
click_link 'link_text'
```

```
click_on 'link_or_button_value'
```

```
find('css_selector').click
```

```
find(:xpath, 'xpath_selector').click
```

Type

```
fill_in 'name', with: 'text to type in field'
```

```
fill_in 'id', with: 'text to type in field'
```

```
fill_in 'label', with: 'text to type in field'
```

```
find('css_selector').set('text to type in field')
```

```
find(:xpath, 'xpath_selector').set('text to type in field')
```

Other Actions

```
select 'text of choice', :from => 'select_id'
```

```
select 'text of choice', :from => 'select_name'
```

```
select 'text of choice', :from => 'select_label_text'
```

```
check('name_of_checkbox')
```

```
check('id_of_checkbox')
```

```
check('text_of_checkbox_label')
```

```
uncheck('name_id_or_text_of_checkbox')
```

```
attach_file('name', 'path/to/file/to_upload.png')
```

```
attach_file('id', 'path/to/file/to_upload.png')
```

```
attach_file('label_text', 'path/to/file/to_upload.png')
```

```
choose('radio_button_name')
```

```
choose('radio_button_id')
```

```
choose('radio button label text')
```

Capy



Navigate

```
visit 'url'
```

```
page.driver.browser.switch_to.window(window_handle)
```

```
my_last_win_handle = page.driver.browser.window_handles.last
```

```
my_original_win_handle = page.driver.browser.window_handles.first
```

```
page.driver.browser.switch_to.alert.accept
```

Scoping

```
within('css_locator')
```

```
within(:xpath, 'xpath_locator')
```

```
within_frame('name')
```

```
within_frame('id')
```

```
within_frame('index')
```

Anything in the block is scoped by the within. e.g. use with block syntax:

```
within('#div_id') do
```

```
  click_button('button_id')
```

```
end
```

Verify

```
expect(page).to have_content('words on page')
```

```
expect(location).to have_content('words in an area')
```

```
expect(page_or_location).to_not have_content('unexpected words')
```

```
expect(current_url).to have_content('part/or_all/of_url')
```

```
expect(page).to have_selector('css_selector')
```

```
expect(page).to have_selector(:xpath, 'xpath_selector')
```

```
expect(page).to have_selector('css_selector', count: 12)
```