Nerodia Documentation

Release 0.5.0

Lucas Tierney

Jan 23, 2019

Contents

Nerodia is a Python port of the Watir ruby gem. https://github.com/watir/watir

CHAPTER 1

Installation

1.1 Supported Python Versions

- Python 2.7
- Python 3.4+

1.2 Installing

If you have pip on your system, you can simply install or upgrade:

```
pip install -U nerodia
```

Alternately, you can download the source distribution from PyPI (e.g. nerodia-1.0.0.tar.gz), unarchive it, and run:

```
python setup.py install
```

CHAPTER 2

Examples

2.1 Navigate to a Website

```
from nerodia.browser import Browser

browser = Browser(browser='firefox')
browser.goto('watir.com')

browser.close()
```

2.2 Perform a Google Search

```
from nerodia.browser import Browser

browser = Browser(browser='firefox')
browser.goto('google.com')

search_input = browser.text_field(title='Search')
search_input.value = 'nerodia'
browser.button(value='Google Search').click()

browser.close()
```

2.3 Select a Checkbox

```
from nerodia.browser import Browser

browser = Browser(browser='firefox')
browser.goto('the-internet.herokuapp.com/checkboxes')

checkbox1 = browser.checkbox()
checkbox1.set()

browser.close()
```

2.4 Elements in Frames

```
from nerodia.browser import Browser

browser = Browser(browser='firefox')
browser.goto('the-internet.herokuapp.com/iframe')

print(browser.iframe().p().text)
print(browser.link(css='#page-footer a').text)

browser.close()
```

Result:

```
> Your content goes here.
> Elemental Selenium
```

CHAPTER 3

Differences from Watir

The goal of this project is to be as close to Watir as possible. In terms of functionality, it is equivalent; however, there are some syntax differences due to the nature of Python.

3.1 Containers

The following containers cannot be used because either the singular or plural version is reserved by Python.

Watir	Nerodia
a	link
as	links
del	delete
dels	deletes
i	ital
is	itals

3.2 Locators

The following locators cannot be used because they are reserved by Python.

Watir	Nerodia
class	class_name
for	N/A*

**This locator is only possible via the below options.*

Alternatively, if you are only using one locator you can pass them as individual arguments:

```
browser.div('class', 'spam')
```

A third option is to use a dictionary and unpack into the container:

```
locator = {'class': 'spam', 'index': 1}
browser.div(**locator)
```

3.3 Blocks

Since Python does not have blocks, alternate methods are required.

3.3.1 Context

For cases where we want to perform some actions inside of a different browser context without completely switching to that context, we use the context manager.

Consider the following Window switching Watir code:

```
browser.window(title: 'Spam and Ham!').use do
  browser.button(id: 'close').click
end
```

In Nerodia, the equivalent would be:

```
with browser.window(title='Spam and Ham!'):
    browser.button(id='close').click()
```

The same would go for frames.

3.3.2 Waits

For waits, we need to use `lambdas` or `closures`.

Consider the following wait Watir code:

```
btn = browser.button(id: 'btn')
btn.wait_until(timeout: 2, interval: 0.5) { btn.enabled }
btn.click
```

In Nerodia, the equivalent would be:

```
btn = browser.button(id='btn')
btn.wait_until(timeout=2, interval=0.5 method=lambda e: e.enabled)
btn.click()
```

Also, `while` is reserved in Python. Therefore, the Nerodia equivalent of Watir's `Wait.while` is `Wait.until_not`