10 - Condições especiais para esperas

Roteiro

- Esperas built-in
- Eventos

picpay.me/dunossauro | apoia.se/livedepython

picpay.me/dunossauro | apoia.se/livedepython

```
dunossauro at babbage in ~/git/apoiase on master*
$ python apoiadores.py
ADRIANO FERRARI      ALBERTO TAVARES      Alex Lima            Alexandre Possebom   Alexandre Santos
Alexandre Sá         Alexandre Tsuno      Alexandre Villares   Alynne Ferreira      Alysson Oliveira
Amaziles Carvalho    And Past             Andre Machado        Andre Rodrigues      Athayr Athayr
Bernardo Fontes      Bruno Barros         Bruno Oliveira       Bruno Gaffuri        Bruno Guizi
Bruno Luna           Bruno Lira           Bruno Rocha          CARLOS SANTOS        Caio Nascimento
Carlos Cardoso       Carlos Augusto       Carol Souza          Cleber Santos        Cleiton Mittmann
Cleiton Lima         DYLAN MAKER          Davi Ramos           David Reis           David Silva
Davinir Junior       Dayham Soares        Diego Guimarães      Diego Ubirajara      Dilenon Delfino
Douglas Bastos       EDUARDO PAZZI        Edimar Fardim        Eduardo Guadanhim    Eduardo Nunes
Eliabe Silva         Eliakim Morais       Elias Soares         Elias Soares         Eliel Lima
Emerson Lara         Emerson Lara         Eugenio Mazzini      Eugenio Mazzini      Fabiano Silos
Fabiano Teichmann    Fabiano Gomes        Fabrício Coelho      Fausto Caldeira      Fernando Furtado
Filipe Cruz          FlavKaze FlavKaze    Franklin Silva       Fábio Serrão         Gabriel Simonetto
Gabriela Santiago    Geandreson Costa     Gladson Menezes      Gleison Oliveira     Guilherme Ramos
Helter Silva         Hemilio Lauro        Humberto Rocha       Hélio Neto           Hélio Neto
Isaac Ferreira       JONATHAN DOMINGOS    Jean Vetorello       Johnny Tardin        Jonatas Oliveira
Jonatas Simões       Jonatas Oliveira     Jorge Plautz         José Prado           Jovan Costa
João Lugão           João Coelho          Juan Gutierrez       Jucélio Silva        Júlia Kastrup
Kauan Alves          LEONARDO CRUZ        Leon Teixeira        Leonardo Rezende     Letícia Silva
Lucas Mendes         Lucas Nascimento     Lucas Neris          Lucas Valino         Lucas Polo
Magno Malkut         Maiquel Leonel       Marcello Benigno     Marcus Salgues       Maria Boladona
Maria Clara          Mateus Braga         Matheus Francisco    Melisa Campagnaro    Nilo Pereira
Nídio Dolfini        Octavio Sydow        Pablo Henrique       Patrick Corrêa       Patrick Gomes
Paulo Tadei          Pedro Alves          Rafael Galleani      Rafael Dias          Regis Santos
Renan Gomes          Renan Moura          Renato Santos        Rennan Almeida       Renne Rocha
Rhenan Bartels       Ricardo Schalch      Richard Nixon        Rodrigo Ferreira     Rodrigo Vaccari
Régis Tomkiel        Sérgio Passos        Thais Viana          Thiago Araujo        Tiago Cordeiro
Tyrone Damasceno     Valdir Junior        Valdir Silveira      Vergil Valverde      Vicente Marcal
Victor Geraldo       Vinícius Ferreira    Vitor Dutra          Vítor Zanoni         Wander Silva
Welington Carlos     Wellington Camargo   Welton Souza         William Oliveira     William Oliveira
Willian Lopes        Willian Lopes        Willian Rosa         Yros Aguiar          Falta você
```

Expected conditions

https://www.selenium.dev/selenium/docs/api/py/_modules/selenium/webdriver/support/expected_conditions.html

Expected conditions

São classes prontas para esperas “comuns”, “usuais”. E eu gosto de dividir elas por categorias (Não oficial)

- Existência do elemento
- Visibilidade do elemento
- Navegação
- Verificação de texto
- Janelas e frames

Expected conditions

São classes prontas para esperas “comuns”, “usuais”. E eu gosto de dividir elas por categorias (Não oficial)

- Existência do elemento
- Visibilidade do elemento
- Navegação
- Verificação de texto
- **Janelas e frames (aula de janelas)**

Existência do elemento

Existência do elemento

Talvez o tipo de espera mais comum. Saber se o elemento está na tela, ou existe na tela.

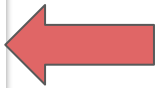
Isso evita de tentar executar uma operação em um elemento que pode ainda não estar disponível.

Checando com expected conditions

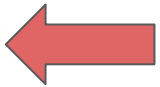
```
from selenium.webdriver import Firefox
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.common.by import By
from selenium.webdriver.support.expected_conditions import (
    presence_of_element_located
)

browser = Firefox()
browser.get('')

wdw = WebDriverWait(browser, 30)

locator = (By.CSS_SELECTOR, 'button.myclass')

wdw.until(
    presence_of_element_located(locator)
)

browser.find_element(*locator).click()
```

Checando com expected conditions

```
from selenium.webdriver import Firefox
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.common.by import By
from selenium.webdriver.support.expected_conditions import (
    presence_of_element_located
)

browser = Firefox()
browser.get('')

wdw = WebDriverWait(browser, 30)

locator = (By.CSS_SELECTOR, 'button.myclass')

wdw.until(
    presence_of_element_located(locator)
)

browser.find_element(*locator).click()
```



Import da espera pronta



Uso dessa espera

CODEEEEEEEEEEE

https://selenium.dunossauro.live/aula_10_a.html

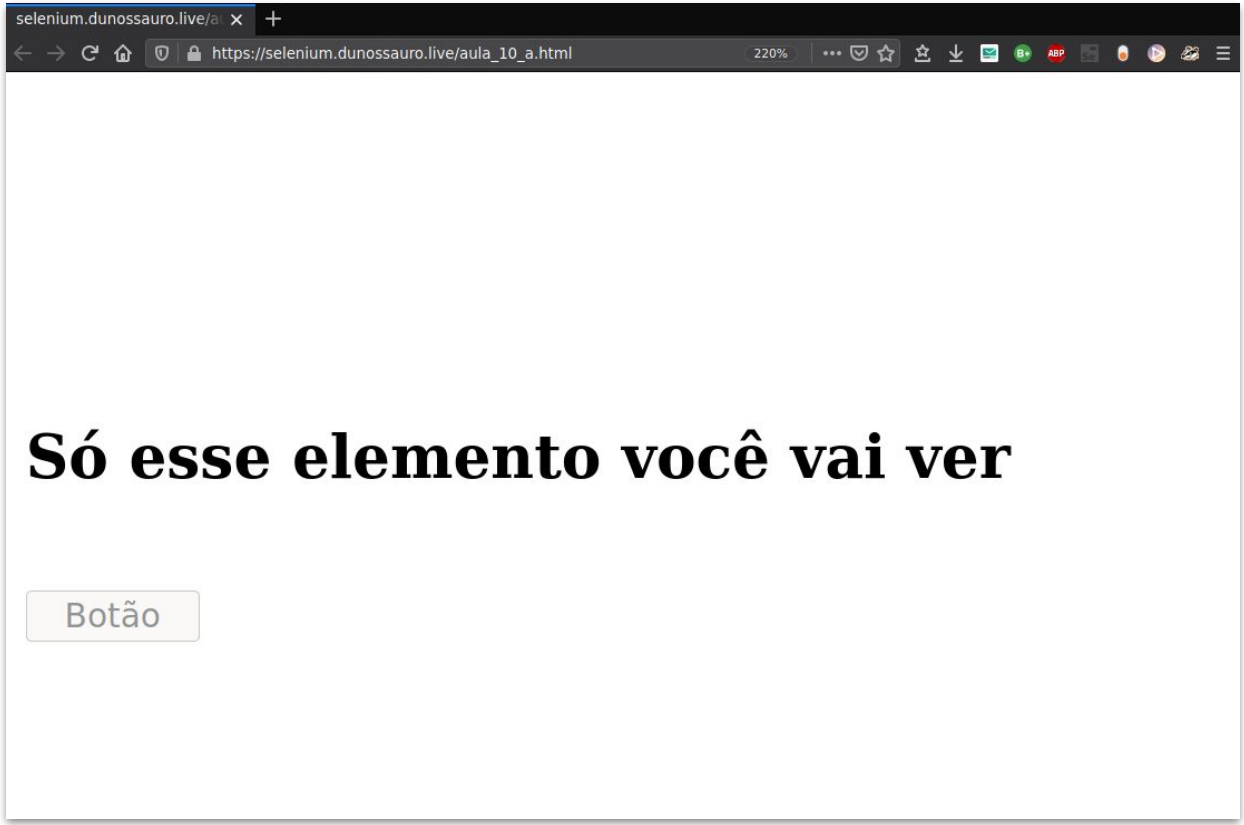
Visibilidade do elemento

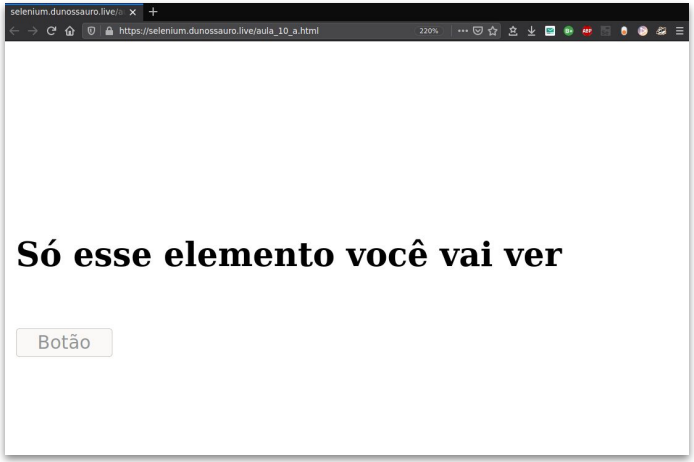
Visibilidade do elemento

Aqui basicamente queremos saber se o elemento está desenhado na tela ou não (aula de drawing/eventos) e também se ele está ativo ou não.

Exemplos:

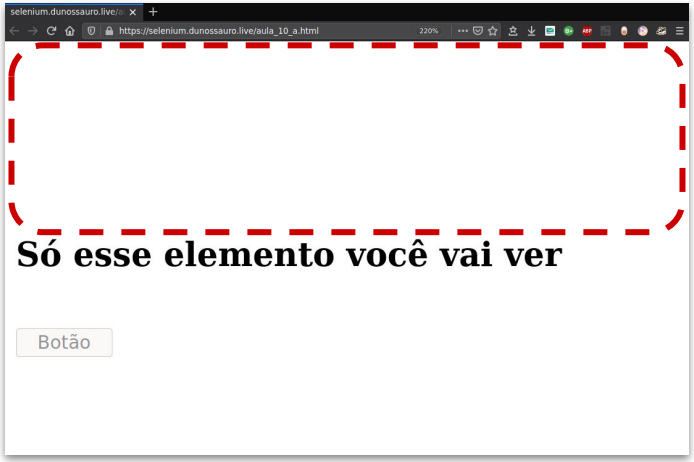
- O elemento pode não ter sido desenhado
- O elemento pode ter sido desenhado mas estar inativo


selenium.dunossauro.live/a
https://selenium.dunossauro.live/aula_10_a.html
220%
Só esse elemento você vai ver
Botão



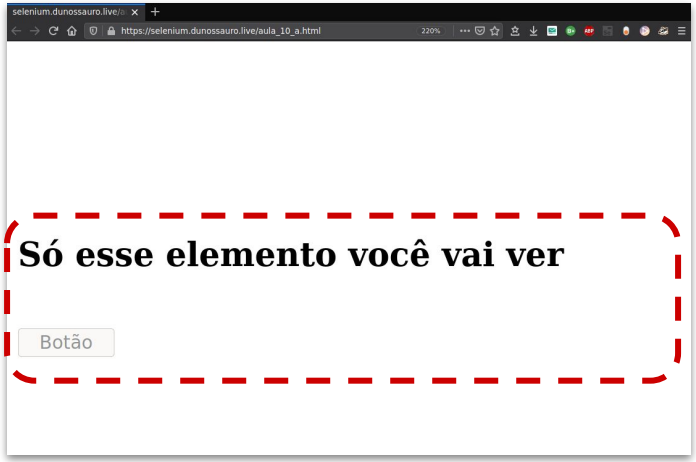

```css
body * {
  visibility: hidden;
}
body h2,
body button {
  visibility: visible;
}
```

```html
<body>
  <h1>Bem vindo ao site invisível</h1>
  <h1>Sério, é mesmo invisível</h1>
  <h2>Só esse elemento você vai ver</h2>
  <div>
    Essa é uma div
  </div>
  <button type="button" name="button" disabled>Botão</button>
</body>
```




```css
body * {
  visibility: hidden;
}
body h2,
body button {
  visibility: visible;
}
```

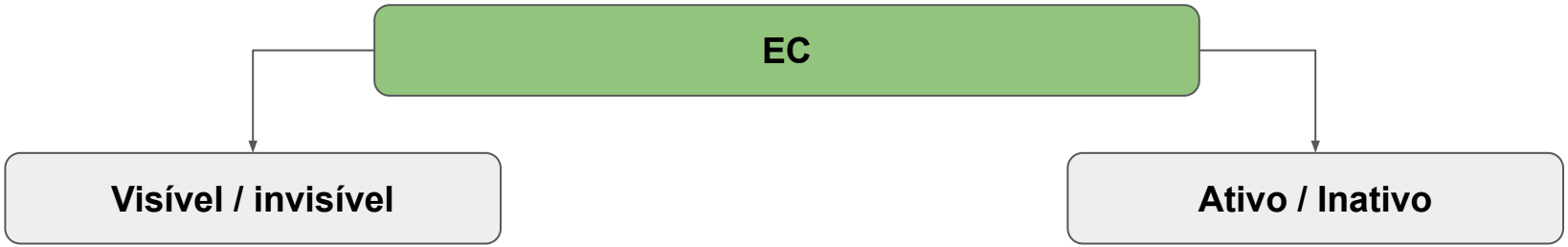
```html
<body>
  <h1>Bem vindo ao site invisível</h1>
  <h1>Sério, é mesmo invisível</h1>
  <h2>Só esse elemento você vai ver</h2>
  <div>
    Essa é uma div
  </div>
  <button type="button" name="button" disabled>Botão</button>
</body>
```



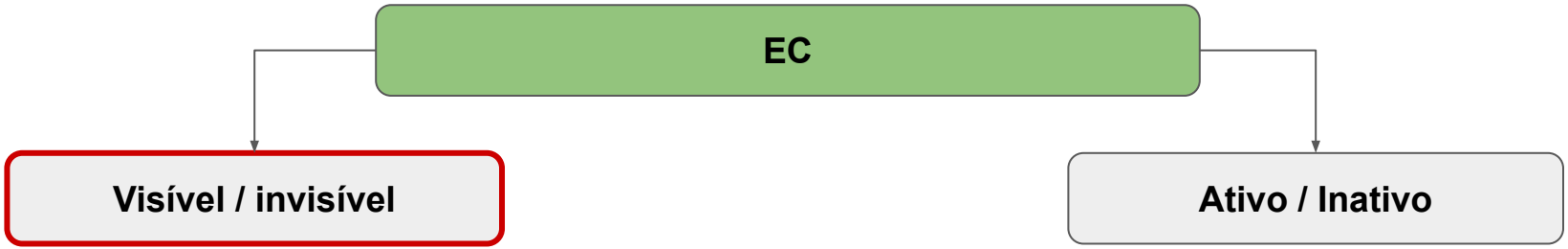

```css
body * {
  visibility: hidden;
}
body h2,
body button {
  visibility: visible;
}
```

```html
<body>
  <h1>Bem vindo ao site invisível</h1>
  <h1>Sério, é mesmo invisível</h1>
  <h2>Só esse elemento você vai ver</h2>
  <div>
    Essa é uma div
  </div>
  <button type="button" name="button" disabled>Botão</button>
</body>
```

Visibilidade do elemento



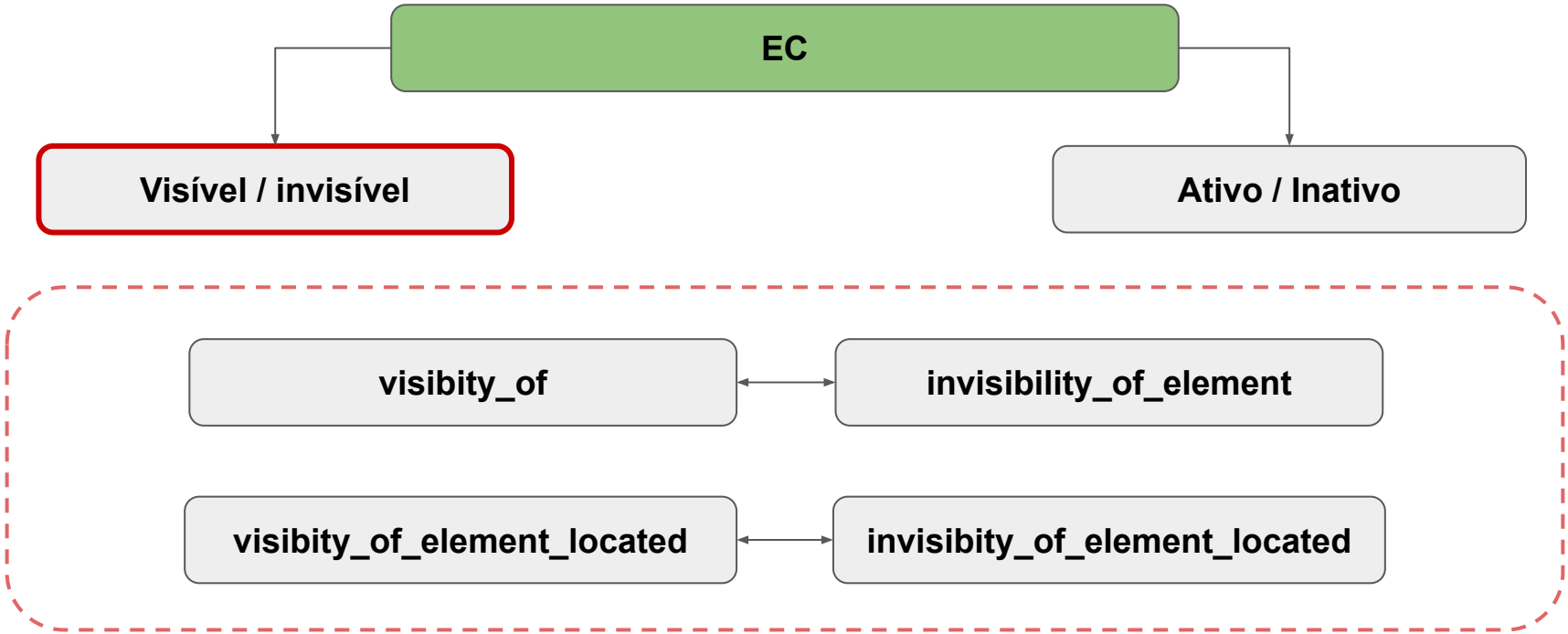
Visibilidade do elemento



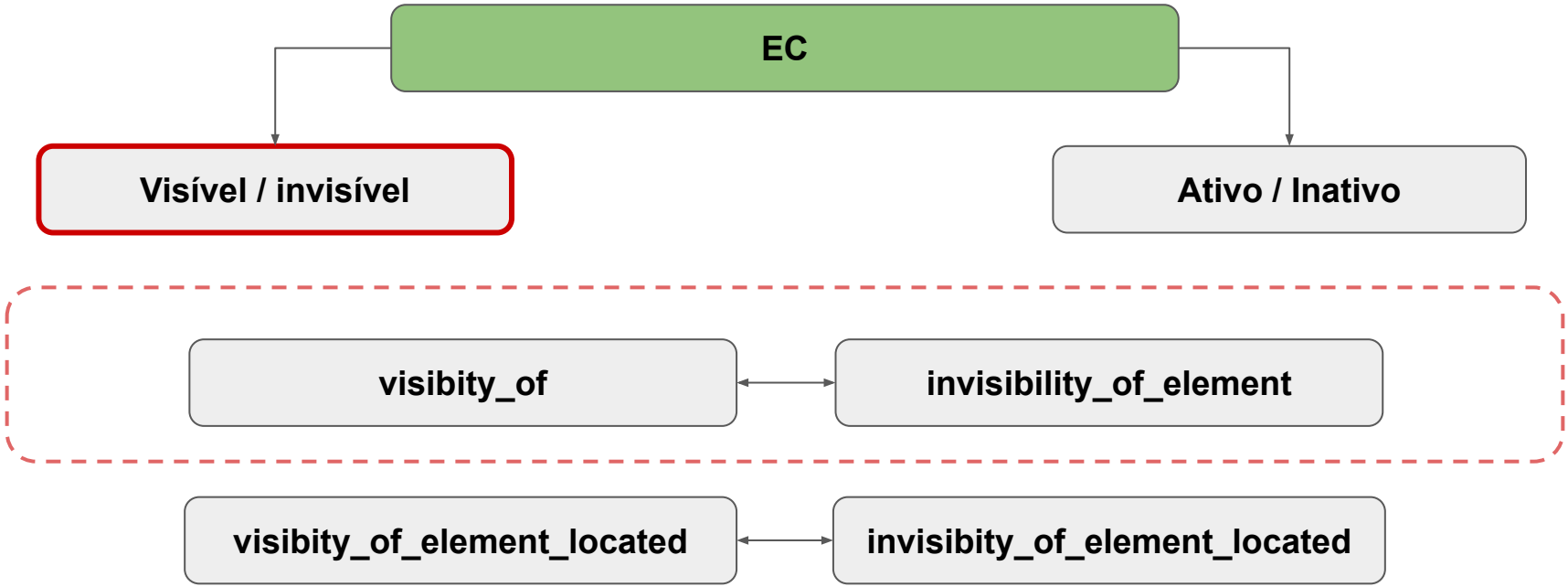

```
body * {
  visibility: hidden;
}
body h2,
body button {
  visibility: visible;
}
```

https://selenium.dunossauro.live/aula_10_b.html

Visibilidade do elemento



Visibilidade do elemento

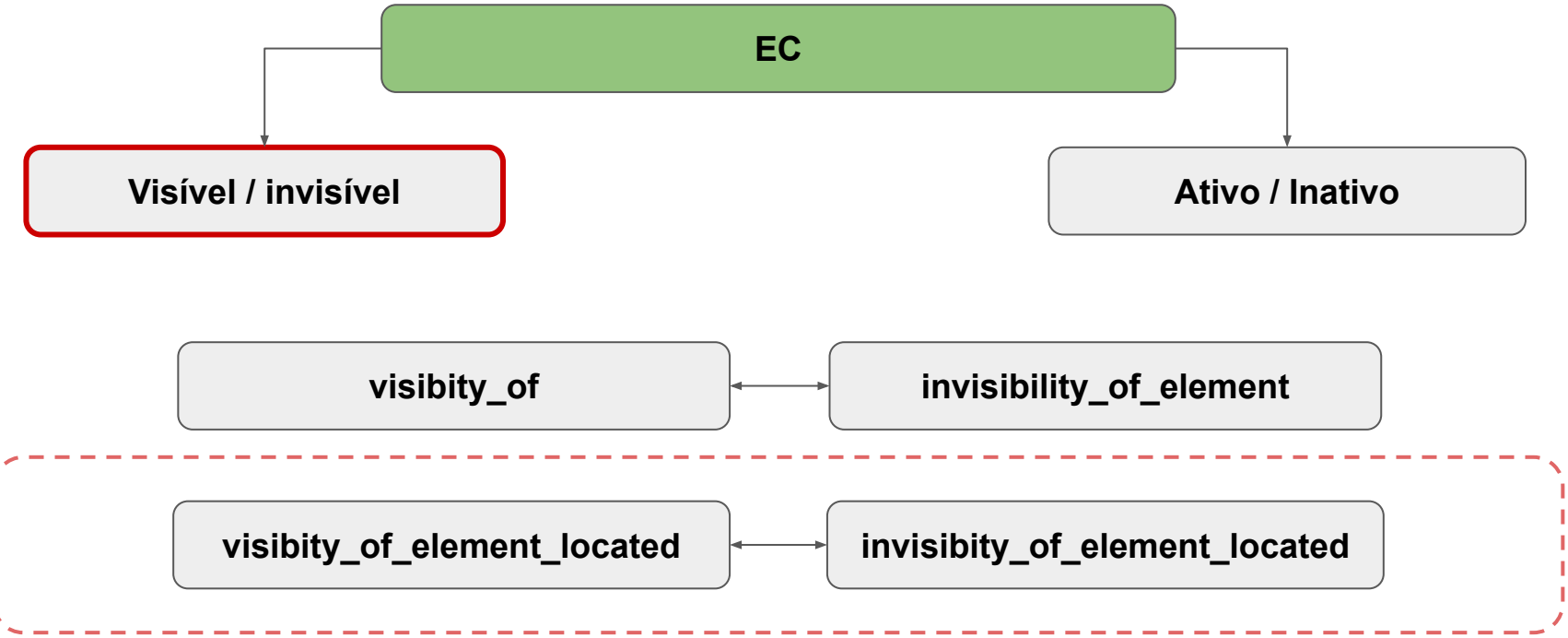


Visibilidade

```python
from selenium.webdriver import Firefox
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support.expected_conditions import (
    visibility_of,staleness_of,
    invisibility_of_element
)
f = Firefox()

f.get('URL')

w = WebDriverWait(f, 5)

element = f.find_element_by_tag_name('h1')
```

```python
w.until_not(
    visibility_of(element), 'Elemento está visível'
)
w.until(
    visibility_of(element), 'Elemento não está visível'
)
w.until(
    invisibility_of_element(element), 'Elemento está visível'
)
```

Visibilidade do elemento

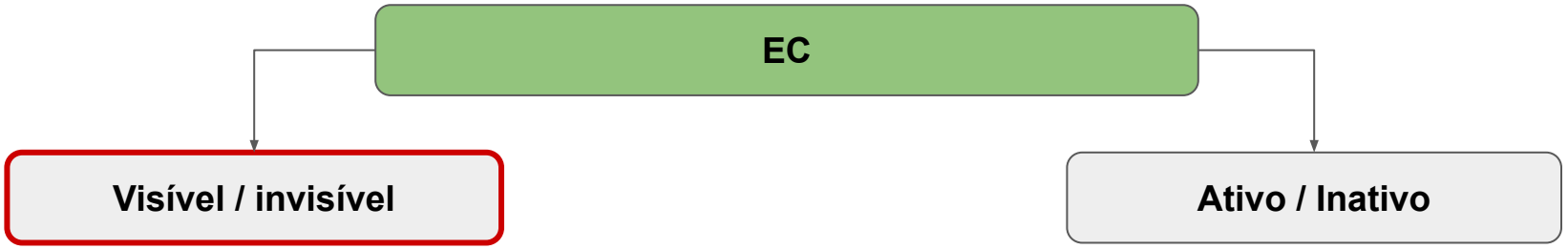


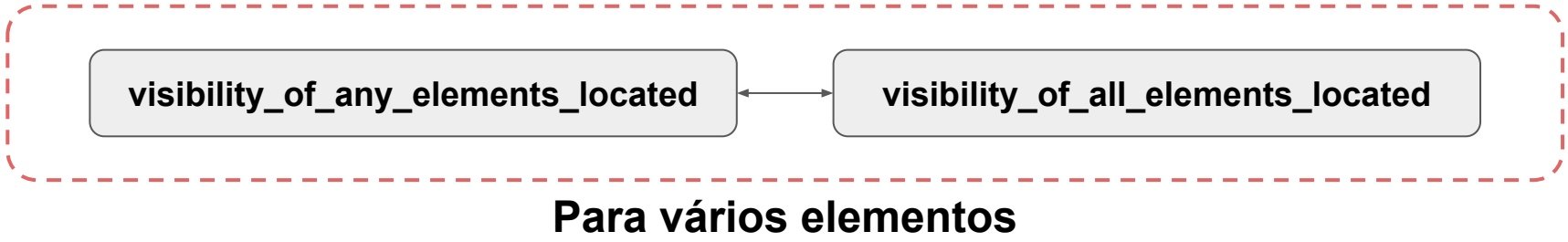
Visibilidade “locator”

```python
from selenium.webdriver import Firefox
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.common.by import By
from selenium.webdriver.support.expected_conditions import (
    visibility_of_element_located,
    invisibility_of_element_located
)
f = Firefox()

f.get('URL')

w = WebDriverWait(f, 5)
```

```python
element = (By.CSS_SELECTOR, 'h1')

w.until_not(
    visibility_of_element_located(element), 'Elemento está visível'
)
w.until(
    visibility_of_element_located(element), 'Elemento não está visível'
)
w.until(
    invisibility_of_element_located((By.CSS_SELECTOR, 'h1')),
    'Elemento está visível'
)
```

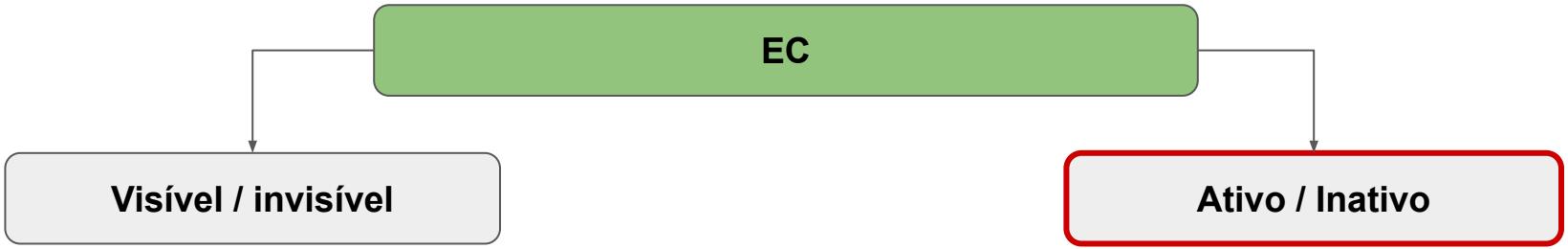
Visibilidade do elemento





Para vários elementos

Visibilidade do elemento

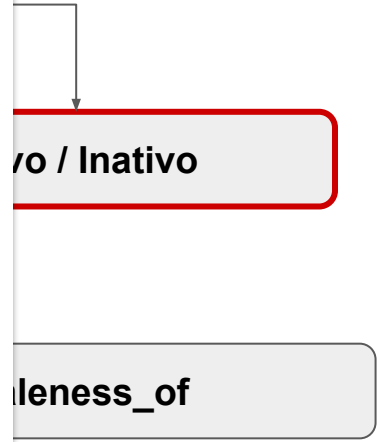



```
<body>

  <h1>Bem vindo ao site invisível</h1>
  <h1>Sério, é mesmo invisível</h1>
  <h2>Só esse elemento você vai ver</h2>
  <div>
    Essa é uma div
  </div>
  <button type="button" name="button" disabled>Botão</button>
```

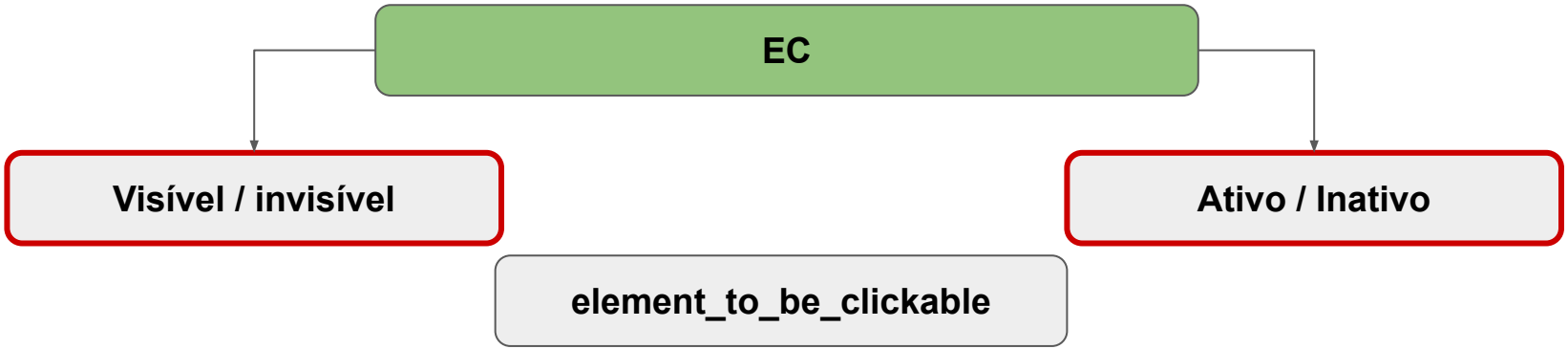
staleness_of

https://selenium.dunossauro.live/aula_10_b.html

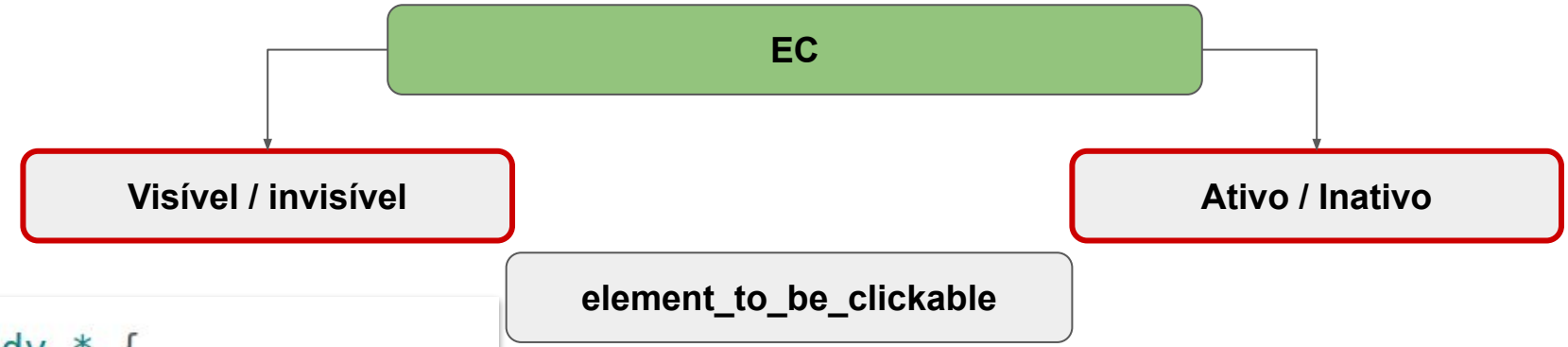
```python
from selenium.webdriver import Firefox
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support.expected_conditions import (
    staleness_of
)
f = Firefox()

f.get('url')

w = WebDriverWait(f, 5)

element = f.find_element_by_tag_name('button')


w.until(
    staleness_of(element), 'Elemento não está habilitado'
)
```



Visibilidade do elemento



Visibilidade do elemento

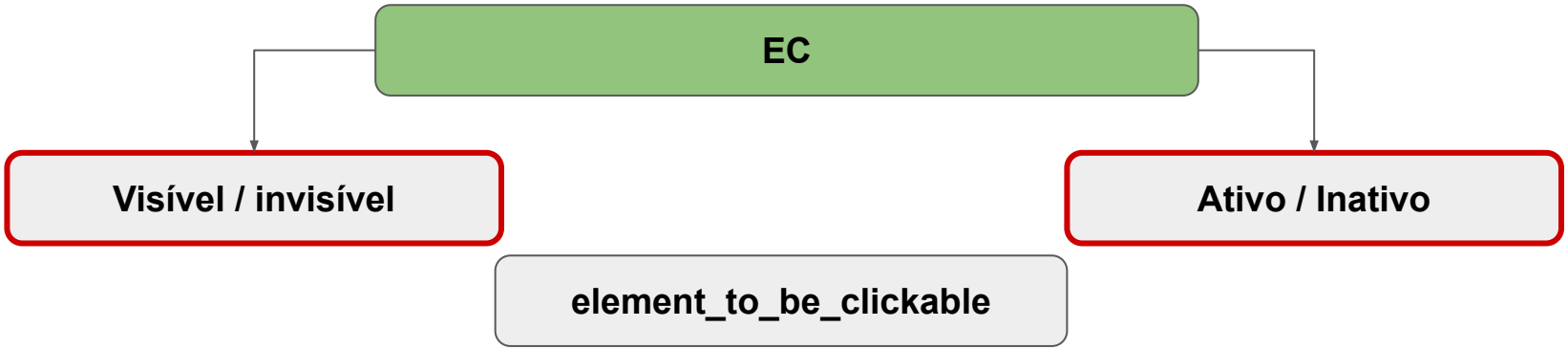



```css
body * {
  visibility: hidden;
}
body h2,
body button {
  visibility: visible;
}
```

```html
<button type="button" name="button" disabled>Botão</button>
```

https://selenium.dunossauro.live/aula_10_b.html

Visibilidade do elemento



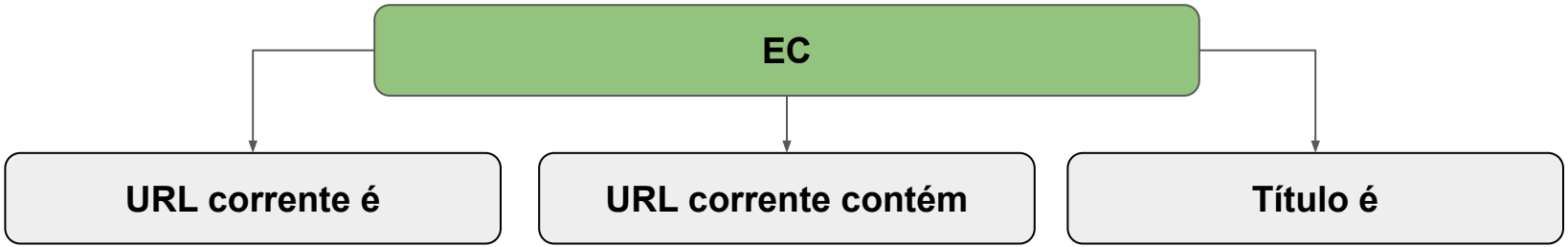

```
w.until(
    element_to_be_clickable(element), 'Elemento não é clicável'
)
```

CODEEEEEEEEEEE

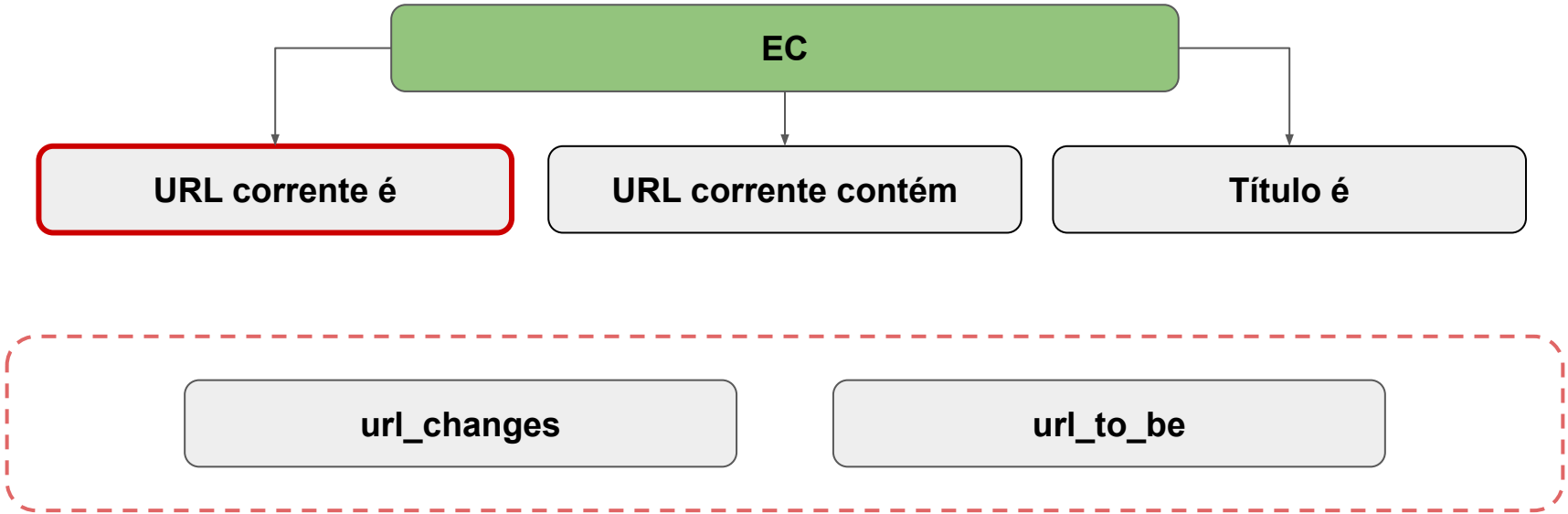
https://selenium.dunossauro.live/aula_10_b.html

Navegação

Esperas baseadas em navegação



Esperas baseadas em navegação



Esperas baseadas em navegação

```
class url_changes:
    def __init__(self, url):
        self.url = url

    def __call__(self, driver):
        return self.url != driver.current_url
```

```
class url_to_be:
    def __init__(self, url):
        self.url = url

    def __call__(self, driver):
        return self.url == driver.current_url
```

```
wdw.until(
    EC.url_to_be('http://minha_url.com')
)

wdw.until(
    EC.url_changes('http://minha_url.com')
)
```

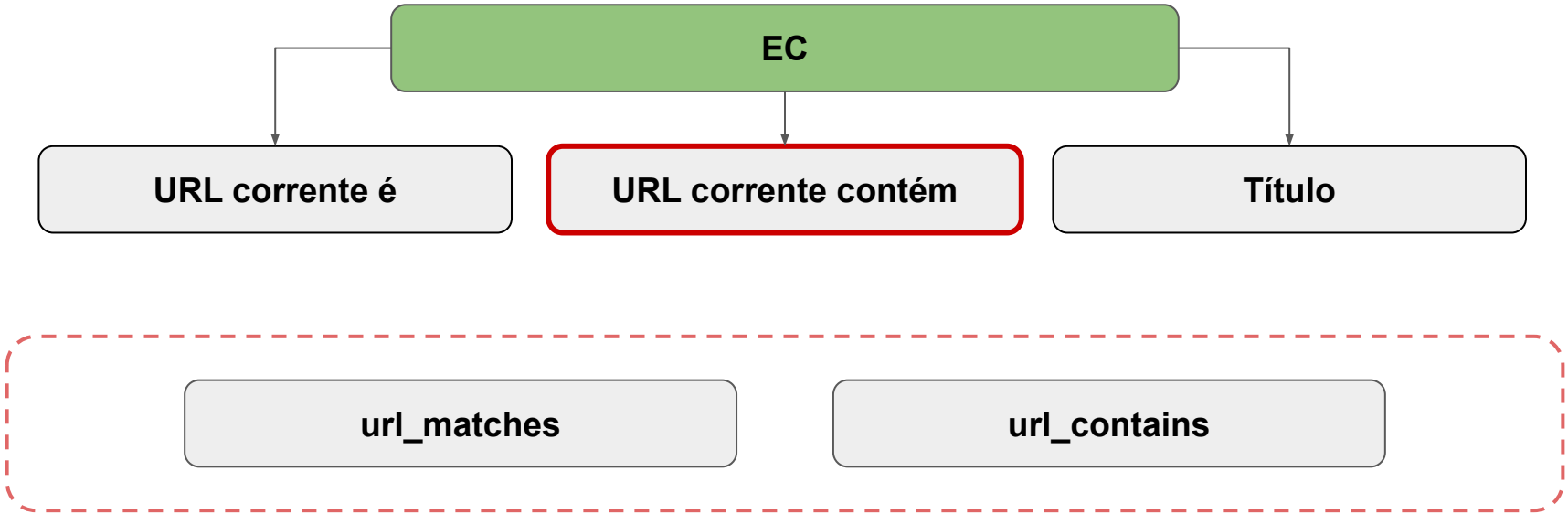
Esperas baseadas em navegação

```
wdw.until(
    EC.url_to_be('http://minha_url.com')
)

wdw.until(
    EC.url_changes('http://minha_url.com')
)
```

=

```
wdw.until_not(
    EC.url_changes('http://minha_url.com')
)

wdw.until_not(
    EC.url_to_be('http://minha_url.com')
)
```

Esperas baseadas em navegação



Esperas baseadas em navegação

```
class url_contains(object):
    def __init__(self, url):
        self.url = url

    def __call__(self, driver):
        return self.url in driver.current_url
```

```
class url_matches(object):
    def __init__(self, pattern):
        self.pattern = pattern

    def __call__(self, driver):
        import re
        match = re.search(self.pattern, driver.current_url)

        return match is not None
```

Esperas baseadas em navegação

```
class url_contains(object):
    def __init__(self, url):
        self.url = url

    def __call__(self, driver):
        return self.url in driver.current_url
```

String está na URL?

```
class url_matches(object):
    def __init__(self, pattern):
        self.pattern = pattern

    def __call__(self, driver):
        import re
        match = re.search(self.pattern, driver.current_url)

        return match is not None
```

Esperas baseadas em navegação

```
class url_contains(object):
    def __init__(self, url):
        self.url = url

    def __call__(self, driver):
        return self.url in driver.current_url
```

String está na URL?

```
class url_matches(object):
    def __init__(self, pattern):
        self.pattern = pattern

    def __call__(self, driver):
        import re
        match = re.search(self.pattern, driver.current_url)

        return match is not None
```

Regex casa com a url?

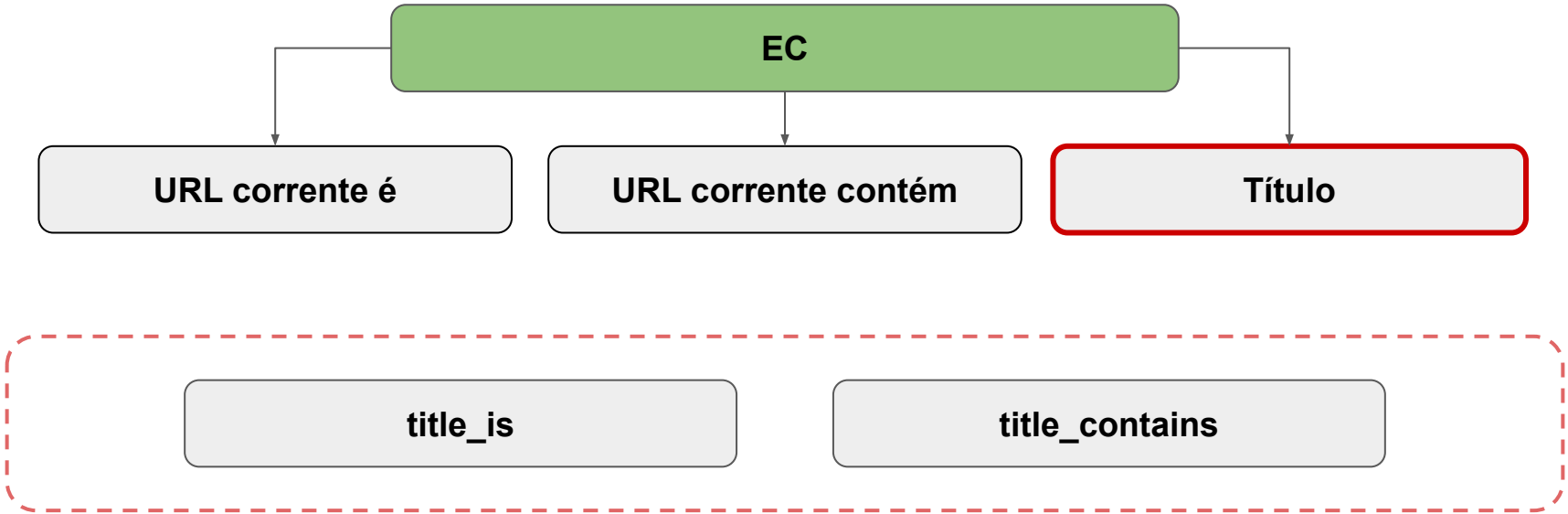
Esperas baseadas em navegação

```
class url_contains(object):
    def __init__(self, url):
        self.url = url

    def __call__(self, driver):
        return self.url in driver.current_url
```

```
class url_matches(object):
    def __init__(self, pattern):
        self.pattern = pattern

    def __call__(self, driver):
        import re
        match = re.search(self.pattern, driver.current_url)

        return match is not None
```

```
wdw.until(
    EC.url_contains('https')
)


wdw.until(
    EC.url_matches(r'selenium\.')
)
```

Esperas baseadas em navegação



Esperas baseadas em navegação

```
class title_is:
    def __init__(self, title):
        self.title = title

    def __call__(self, driver):
        return self.title == driver.title
```

```
class title_contains:
    def __init__(self, title):
        self.title = title

    def __call__(self, driver):
        return self.title in driver.title
```

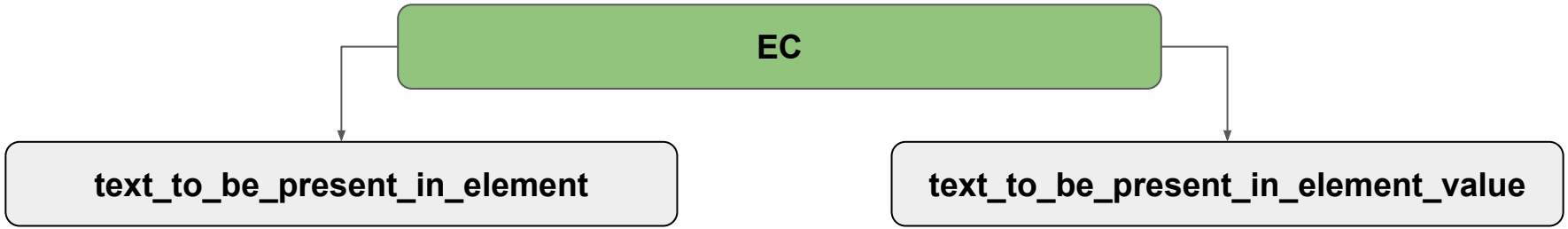
```
wdw.until(
    EC.title_contains('Aula')
)


wdw.until(
    EC.title_is('Aula 10b')
)
```

CODEEEEEEEEEEE

https://selenium.dunossauro.live/aula_10_c.html

Verificação de texto

Esperas por texto



Esperas por texto

```
class text_to_be_present_in_element(object):
    def __init__(self, locator, text_):
        self.locator = locator
        self.text = text_

    def __call__(self, driver):
        try:
            element_text = _find_element(driver, self.locator).text
            return self.text in element_text
        except StaleElementReferenceException:
            return False
```

```
class text_to_be_present_in_element_value
    def __init__(self, locator, text_):
        self.locator = locator
        self.text = text_

    def __call__(self, driver):
        try:
            element_text = _find_element(driver,
                                         self.locator).get_attribute("value")
            if element_text:
                return self.text in element_text
            else:
                return False
        except StaleElementReferenceException:
                return False
```

CODEEEEEEEEEEE

https://selenium.dunossauro.live/aula_10_d.html

Exercícios

- Refazer Exercício 3 (sim pela terceira vez)
- Refazer Exercício 10 (com as classes prontas)
- Exercício 11