Playwright

Live de Python #222

Roteiro

1. Playwright

Uma introdução

2. Os objetos

Playwright, Browser, Page, Context ...

3. Locators

Interação com elementos

4 expects

Checando ações

picpay.me/dunossauro

apoia.se/livedepython

pix.dunossauro@gmail.com

Ajude o projeto <3

Ademar Peixoto, Adilson Herculano, Adriana Cavalcanti, Alexandre Harano, Alexandre Lima, Alexandre Souza, Alexandre Takahashi, Alexandre Villares, Alex Lima, Alynne Ferreira, Alysson Oliveira, Ana Carneiro, Andre Azevedo, André Rafael, Aquiles Coutinho, Arnaldo Turque, Aurelio Costa, Bruno Batista, Bruno Freitas, Bruno Guizi, Bruno Oliveira, Bruno Ramos, Caio Nascimento, Carina Pereira, Christiano Morais, Clara Battesini, Daniel Freitas, Daniel Haas, Danilo Segura, David Couto, David Kwast, Delton Porfiro, Dhyeives Rodovalho, Diego Farias, Diego Guimarães, Dilenon Delfino, Dino Aguilar, Diogo Paschoal, Douglas Bastos, Douglas Braga, Douglas Zickuhr, Dutofanim Dutofanim, Eliel Lima, Elton Silva, Emerson Rafael, Eneas Teles, Erick Ritir, Érico Andrei, Eugenio Mazzini, Euripedes Borges, Everton Silva, Fabiano Tomita, Fabio Barros, Fábio Barros, Fabio Castro, Fábio Thomaz, Fabricio Araujo, Felipe Rodrigues, Fernanda Prado, Fernando Rozas, Flávio Meira, Flavkaze Flavkaze, Gabriel Barbosa, Gabriel Mizuno, Gabriel Nascimento, Gabriel Simonetto, Geandreson Costa, Guilherme Cabrera, Guilherme Felitti, Guilherme Gall, Guilherme Ostrock, Guilherme Piccioni, Gustavo Dettenborn, Gustavo Pereira, Gustavo Suto, Heitor Fernandes, Henrique Junqueira, Hugo Cosme, Igor Taconi, Israel Gomes, Italo Silva, Jair Andrade, Jairo Jesus, Jairo Lenfers, Janael Pinheiro, João Paulo, João Rodrigues, Joelson Sartori, Johnny Tardin, Jonatas Leon, Jônatas Silva, José Gomes, Joseíto Júnior, Jose Mazolini, José Pedro, Juan Gutierrez, Juliana Machado, Júlio Gazeta, Julio Silva, Kaio Peixoto, Kaneson Alves, Leandro Miranda, Leonardo Mello, Leonardo Nazareth, Luancomputacao Roger, Lucas Adorno, Lucas Mello, Lucas Mendes, Lucas Nascimento, Lucas Oliveira, Lucas Simon, Lucas Teixeira, Lucas Valino, Luciano Silva, Luciano Teixeira, Luiz Junior, Luiz Lima, Luiz Paula, Luiz Perciliano, Maicon Pantoja, Maiquel Leonel, Marcelino Pinheiro, Marcelo Matte, Márcio Martignoni, Marcio Moises, Marco Mello, Marcos Gomes, Marco Yamada, Maria Clara, Marina Passos, Mateus Lisboa, Matheus Cortezi, Matheus Silva, Matheus Vian, Mauricio Nunes, Mrreinadogoes Mrreinadogoes, Murilo Andrade, Murilo Cunha, Murilo Viana, Natan Cervinski, Nathan Branco, Nicolas Teodosio, Osvaldo Neto, Patricia Minamizawa, Patrick Felipe, Paulo Braga, Paulo Tadei, Pedro Henrique, Pedro Pereira, Peterson Santos, P Muniz, Priscila Santos, Rafael Lopes, Rafael Rodrigues, Rafael Romão, Ramayana Menezes, Regis Tomkiel, Renato Veirich, Ricardo Silva, Riverfount Riverfount, Robson Maciel, Rodrigo Alves, Rodrigo Cardoso, Rodrigo Freire, Rodrigo Oliveira, Rodrigo Quiles, Rodrigo Vaccari, Rodrigo Vieira, Rogério Nogueira, Rogério Sousa, Ronaldo Silva, Ronaldo Silveira, Rui Jr, Samanta Cicilia, Thalles Rosa, Thiago Araujo, Thiago Bueno, Thiago Curvelo, Thiago Moraes, Thiago Oliveira, Thiago Salgado, Thiago Souza, Tiago Minuzzi, Tony Dias, Valcilon Silva, Valdir Tegon, Victor Wildner, Vinícius Bastos, Vinicius Stein, Vitor Luz, Vladimir Lemos, Walter Reis, Wellington Abreu, Wesley Mendes, William Alves, Willian Lopes, Wilson Neto, Wilson Rocha, Xico Silvério, Yury Barros

Obrigado você

Playwright

Uma introdução

Playwright

Uma biblioteca para manipulação do navegador em testes.

- Criada pelo time opensource da Microsoft
- Criada em Jan de 2020
- Original do JavaScript
- Baseada na arquitetura do Puppeteer (Sem webdriver)
- Tem suporte a programação assíncrona
- Integração com Selenium Grid

```
pip install playwright
```

Instalação

playwright install

Instalação dos browsers

Exemplo inicial

```python
# exemplo_00.py
from playwright.sync_api import sync_playwright

with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page()
    page.goto("http://playwright.dev")

    print(page.title())

    browser.close()
```

https://playwright.dev/python/docs/library#usage

Objetos

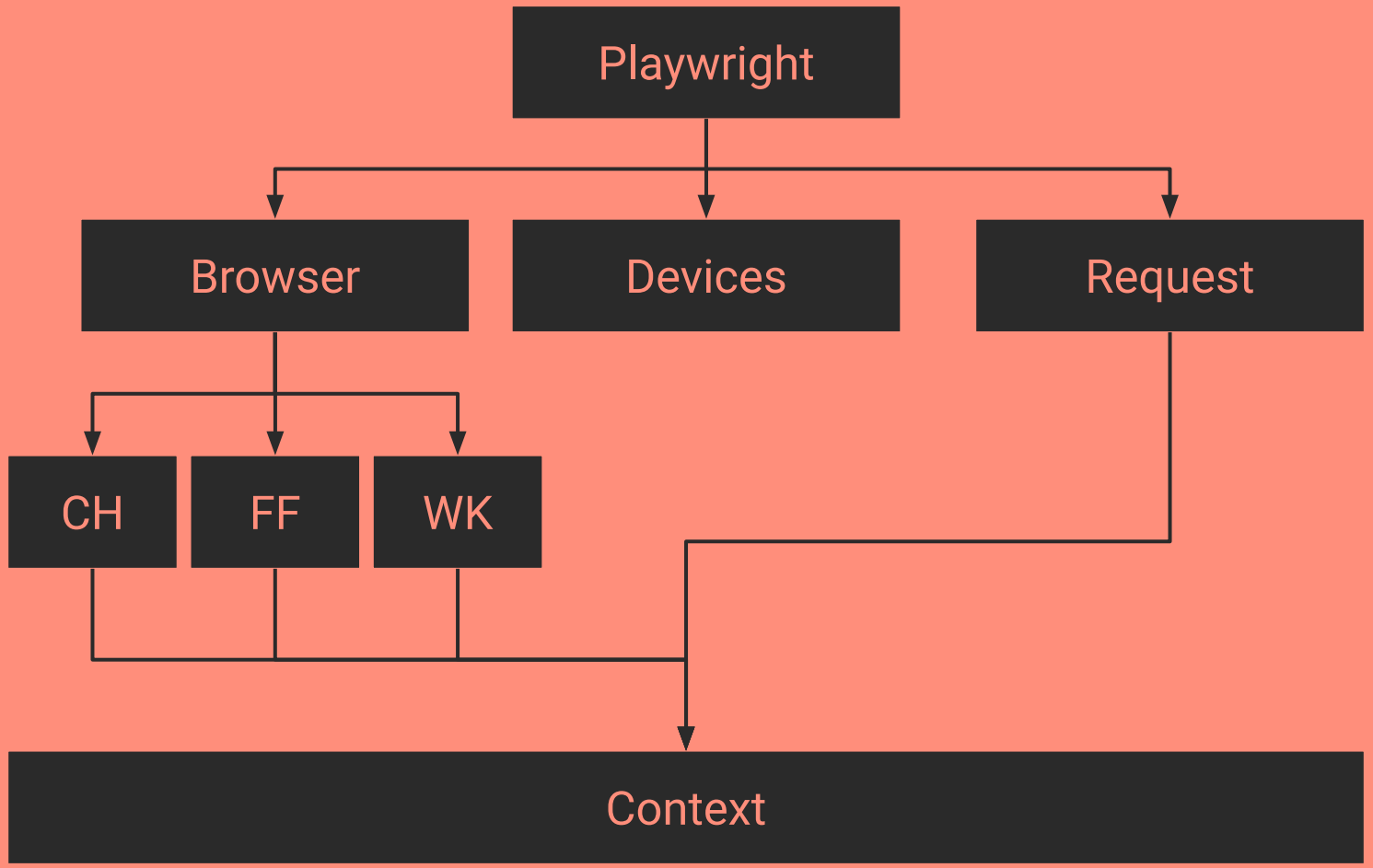
Entendendo o exemplo inicial

A composição de objetos

```python
# exemplo_00.py
from playwright.sync_api import sync_playwright

with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page()
    page.goto("http://playwright.dev")

    print(page.title())

    browser.close()
```

Playwright

O objeto Playwright



Browser x Context

```python
# exemplo_01.py
from playwright.sync_api import (
    Browser, BrowserContext, Page, Playwright, sync_playwright
)
from time import sleep

with sync_playwright() as p:
    p: Playwright

    browser: Browser = p.chromium.launch(headless=False)
    context: BrowserContext = browser.new_context(
        color_scheme='dark',
        record_video_dir='.',
        viewport={'width': 1280, 'height': 1024},
    )
    page: Page = context.new_page()
    page.goto('http://ddg.gg')

    sleep(3)

    browser.close()
```

https://playwright.dev/python/docs/api/class-browsercontext

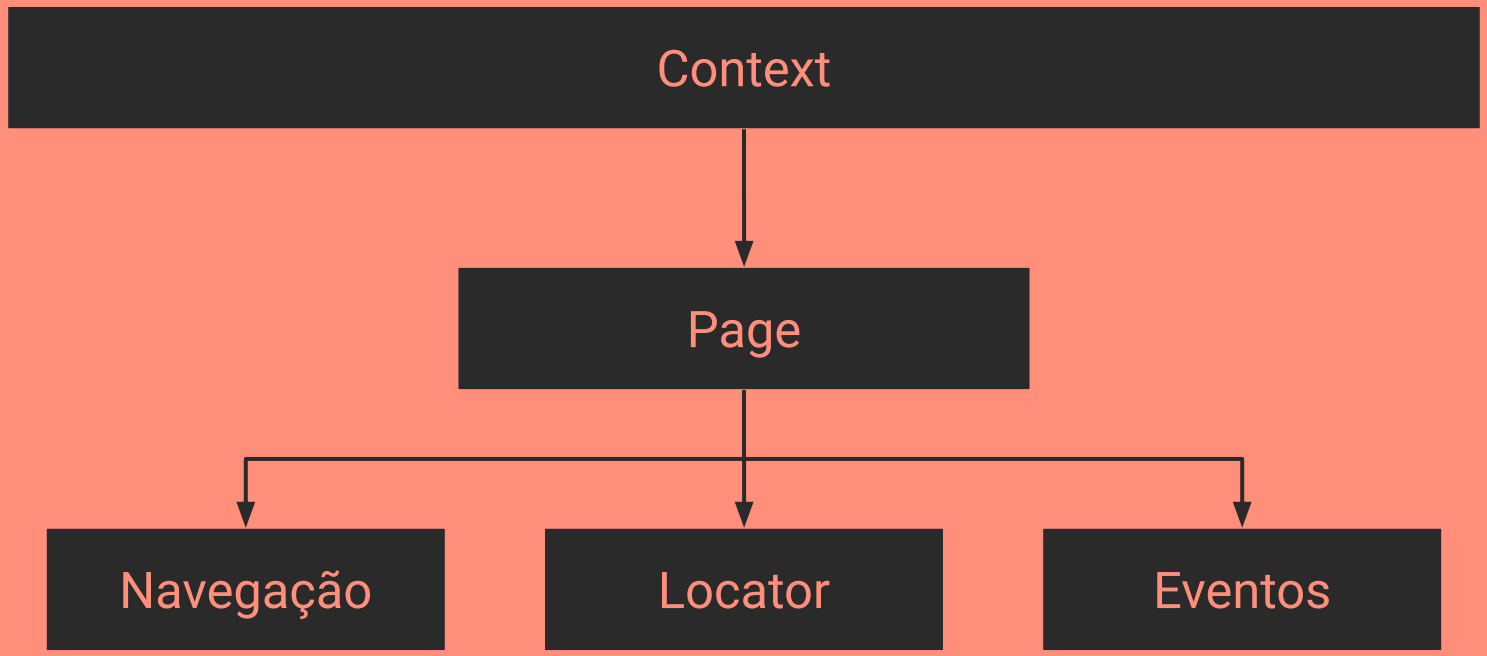
Devices

```python
# exemplo_02.py
from time import sleep
from playwright.sync_api import sync_playwright


with sync_playwright() as p:
    browser = p.chromium.launch(headless=False)

    context = browser.new_context(
        **p.devices['iPhone 6']
    )

    page = context.new_page()
    page.goto('http://ddg.gg')
    # ...
```

Request

```python
# exemplo_03.py
from playwright.sync_api import sync_playwright

with sync_playwright() as p:
    request = p.request.new_context()

    response = request.get('http://ddg.gg')

    print(response.status)
    print(response.status_text)
    print(response.text())
    print(response.json())
```

O restante do diagrama



Navegação

```python
# exemplo_04.py
from playwright.sync_api import sync_playwright
from time import sleep

with sync_playwright() as p:
    browser = p.chromium.launch(headless=False)
    page = browser.new_page()

    page.goto('http://ddg.gg')
    sleep(1)
    page.goto('http://google.com')
    sleep(1)
    page.go_back()
    sleep(1)
    page.go_forward()
```

Eventos

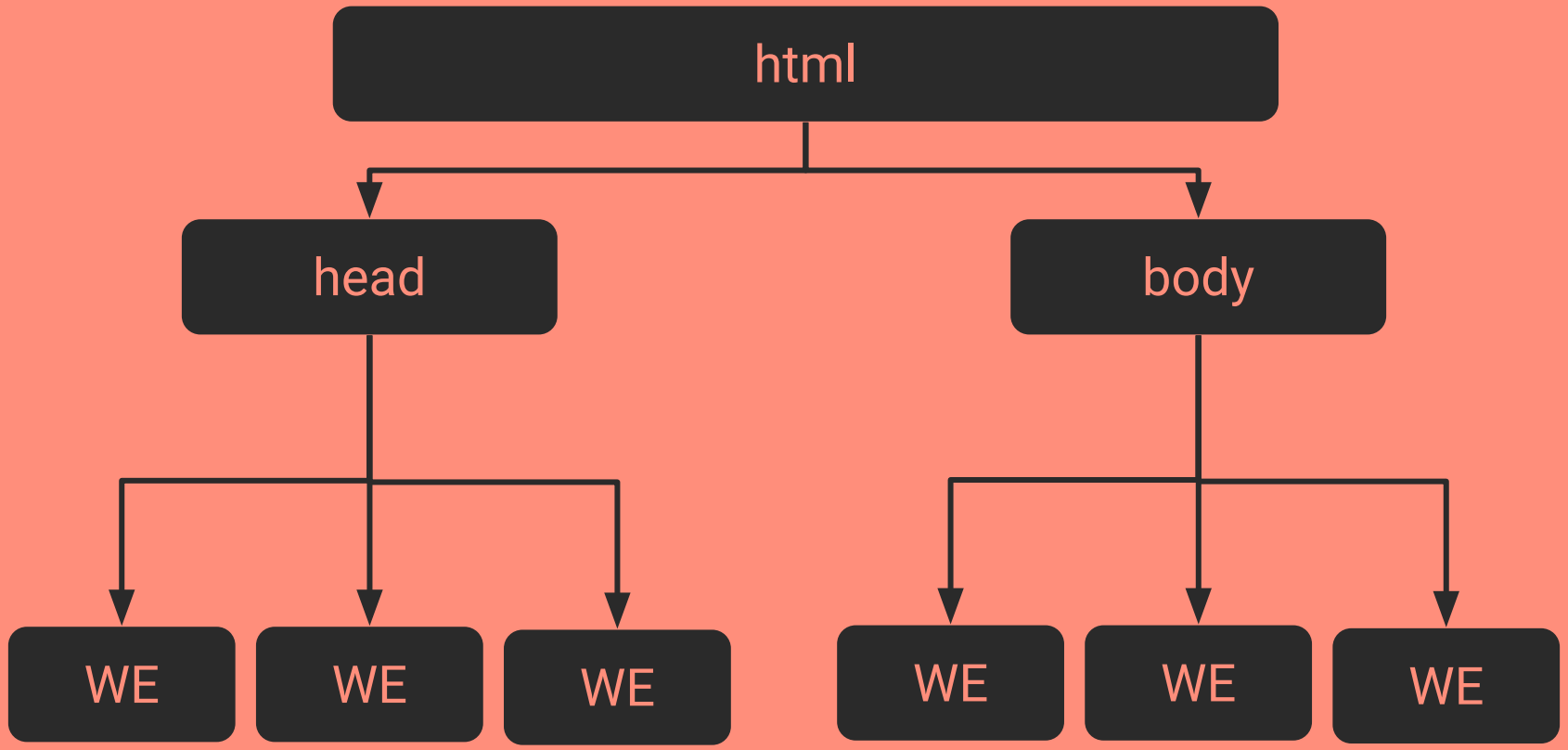
```python
# exemplo_05.py
from playwright.sync_api import sync_playwright
from time import sleep


def event_handler(request):
    print(request.url)


with sync_playwright() as p:
    browser = p.chromium.launch(headless=False)
    page = browser.new_page()

    page.on('request', event_handler)
    # ...
```

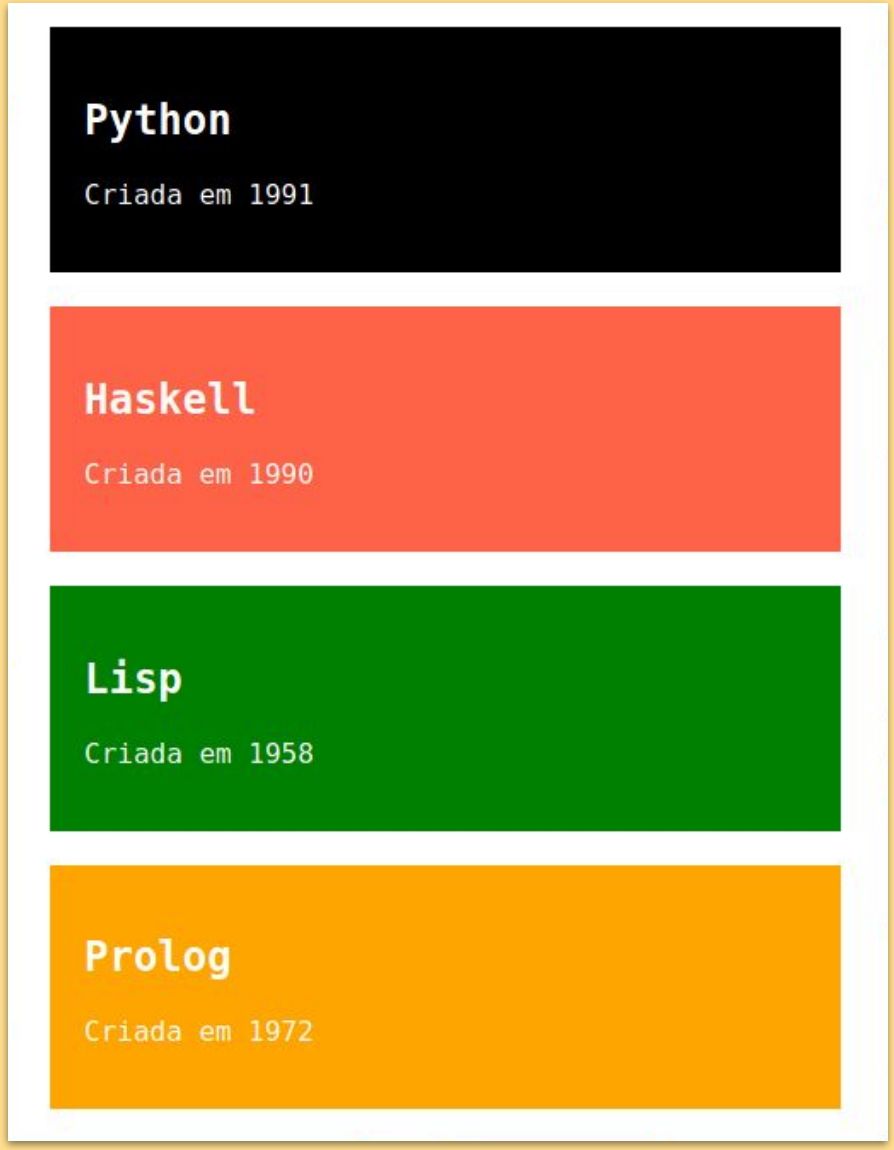
Locators

Interação com elementos

Locators

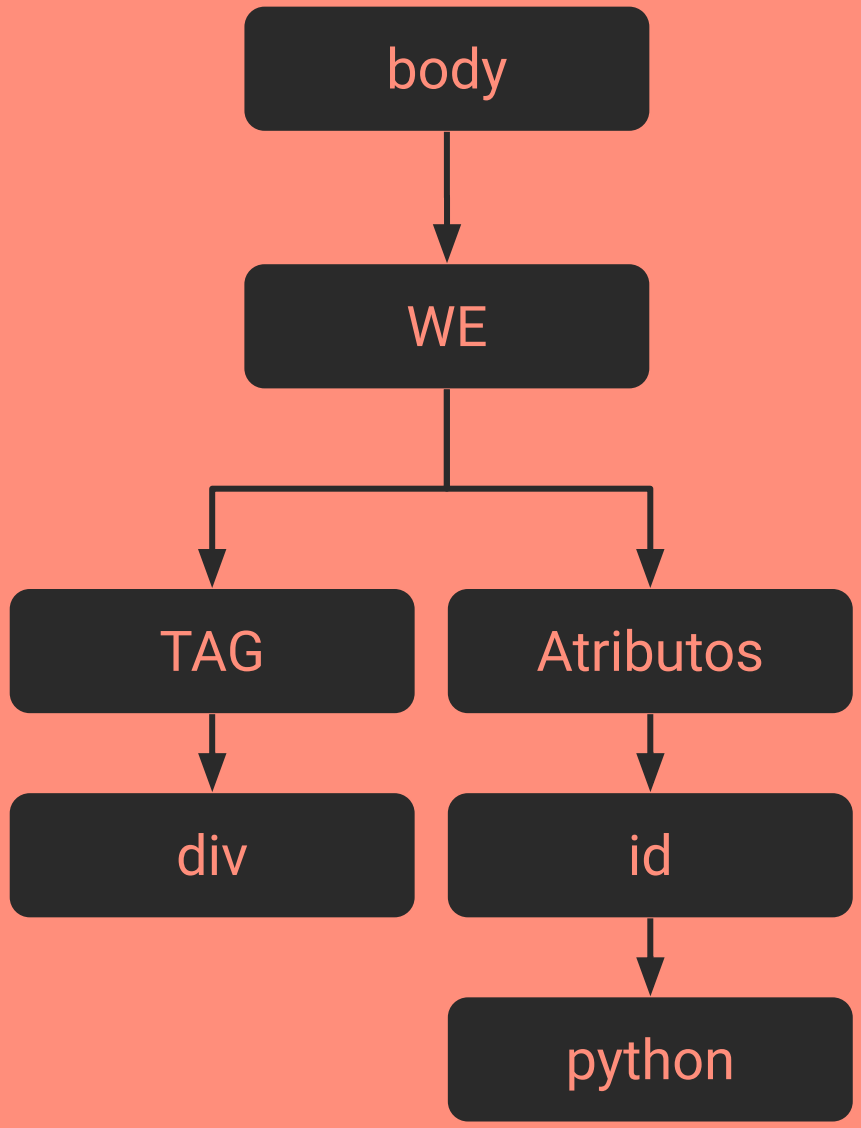
São formas de localizar elementos no DOM e interagir com eles.





https://selenium.dunossauro.live/aula_05_a.html

```html
<body>
  <div id="python">
    <h2>Python</h2>
    <p>Criada em 1991</p>
  </div>

  <div id="haskell">
    <h2>Haskell</h2>
    <p>Criada em 1990</p>
  </div>

  <div id="lisp">
    <h2>Lisp</h2>
    <p>Criada em 1958</p>
  </div>

  <div id="prolog">
    <h2>Prolog</h2>
    <p>Criada em 1972</p>
  </div>
</body>
```



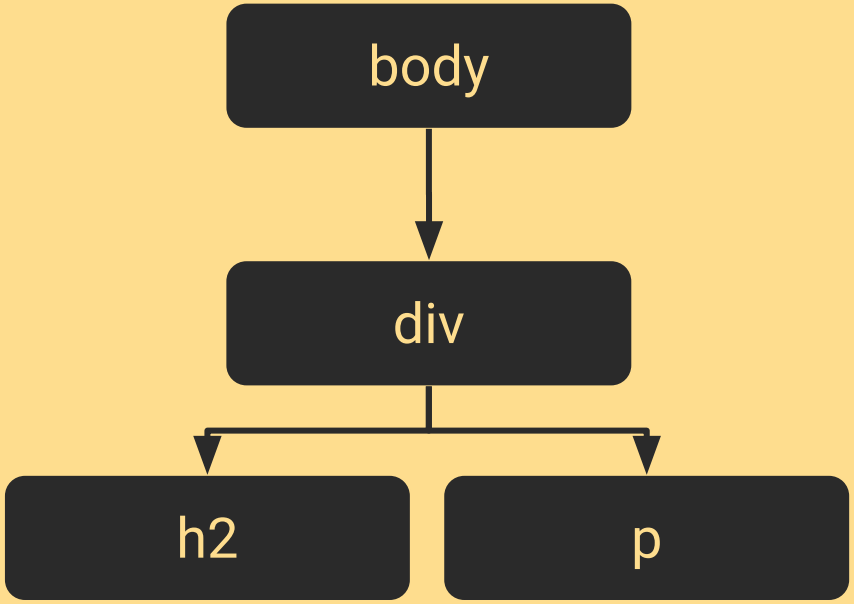

```
<body>
  <div id="python">
    <h2>Python</h2>
    <p>Criada em 1991</p>
  </div>

  <div id="haskell">
    <h2>Haskell</h2>
    <p>Criada em 1990</p>
  </div>

  <div id="lisp">
    <h2>Lisp</h2>
    <p>Criada em 1958</p>
  </div>

  <div id="prolog">
    <h2>Prolog</h2>
    <p>Criada em 1972</p>
  </div>
</body>
```

Acesso pelo atributo

```python
# exemplo_06.py
from playwright.sync_api import sync_playwright

with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page(base_url='https://selenium.dunossauro.live/')
    page.goto('aula_05_a.html')

    locator = page.locator('id=python')  # sugar
    print(locator.text_content())

    locator2 = page.locator('#python')  # CSS
    print(locator2.text_content())

    locator3 = page.locator('//*[@id="python"]')  # XPATH
    print(locator3.text_content())

    browser.close()
```

Acesso pela tag

```python
# exemplo_07.py
from playwright.sync_api import sync_playwright


with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page(base_url='https://selenium.dunossauro.live/')
    page.goto('aula_05_a.html')

    locator = page.locator('div')
    result = locator.nth(0).text_content()

    print(result)
    browser.close()
```



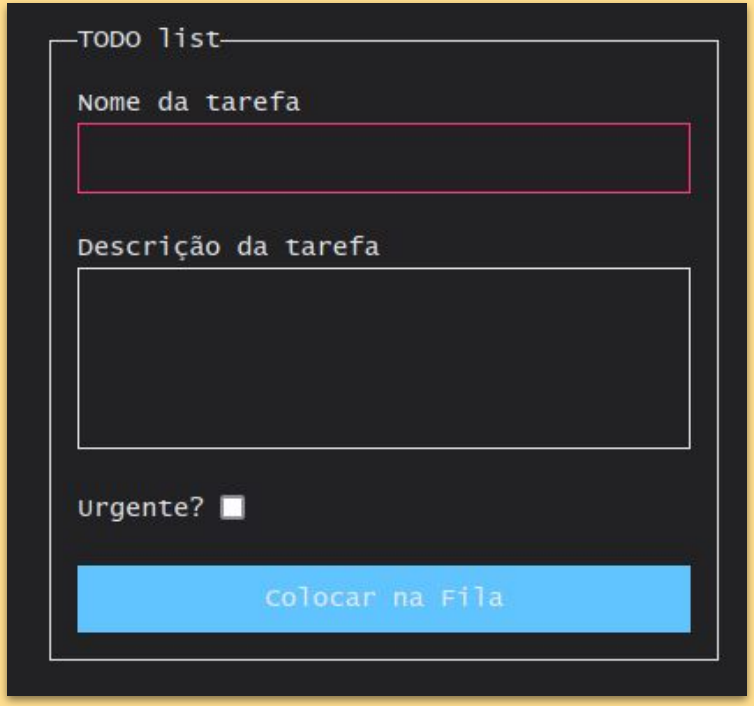

```python
# exemplo_08.py
l_h2 = page.locator('div#python > h2')
l_p = page.locator('div#python > p')
```

```html
<body>
  <div id="python">
    <h2>Python</h2>
    <p>Criada em 1991</p>
  </div>

  <div id="haskell">
    <h2>Haskell</h2>
    <p>Criada em 1990</p>
  </div>

  <div id="lisp">
    <h2>Lisp</h2>
    <p>Criada em 1958</p>
  </div>

  <div id="prolog">
    <h2>Prolog</h2>
    <p>Criada em 1972</p>
  </div>
</body>
```

Interação com os elementos

O locator tem dois métodos principais para interações:

- fill(): Envia texto para elemento
- click(): Clica no elemento



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

```html
<legend>TODO list</legend>

<div class="form-group">
  <label for="terefa">Nome da tarefa</label>
  <input id="todo-name" type="text"
         name="Tarefa" required="">
</div>

<div class="form-group">
  <label for="tarea">Descrição da tarefa</label>
  <textarea id="todo-desc" name="tarea"
            rows="5" cols="30">
  </textarea>
</div>

<div class="form-group">
  <label for="check">Urgente?</label>
  <input id="todo-next" type="checkbox" name="check">
</div>

<button id="todo-submit"
        type="button"
  class="btn btn-primary btn-block">
  Colocar na Fila
</button>
```

```python
# exemplo_09.py
from playwright.sync_api import sync_playwright

url = 'https://selenium.dunossauro.live/todo_list.html'

with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page()
    page.goto(url)

    page.locator('#todo-name').fill('Fazer Live 222')
    page.locator('#todo-desc').fill('Live sobre playwright')
    page.locator('#todo-next').click()
    page.locator('#todo-submit').click()

    page.screenshot(path='result.png', full_page=True)

    page.close()
```

Fazendo a checagem!

Expects

Expects

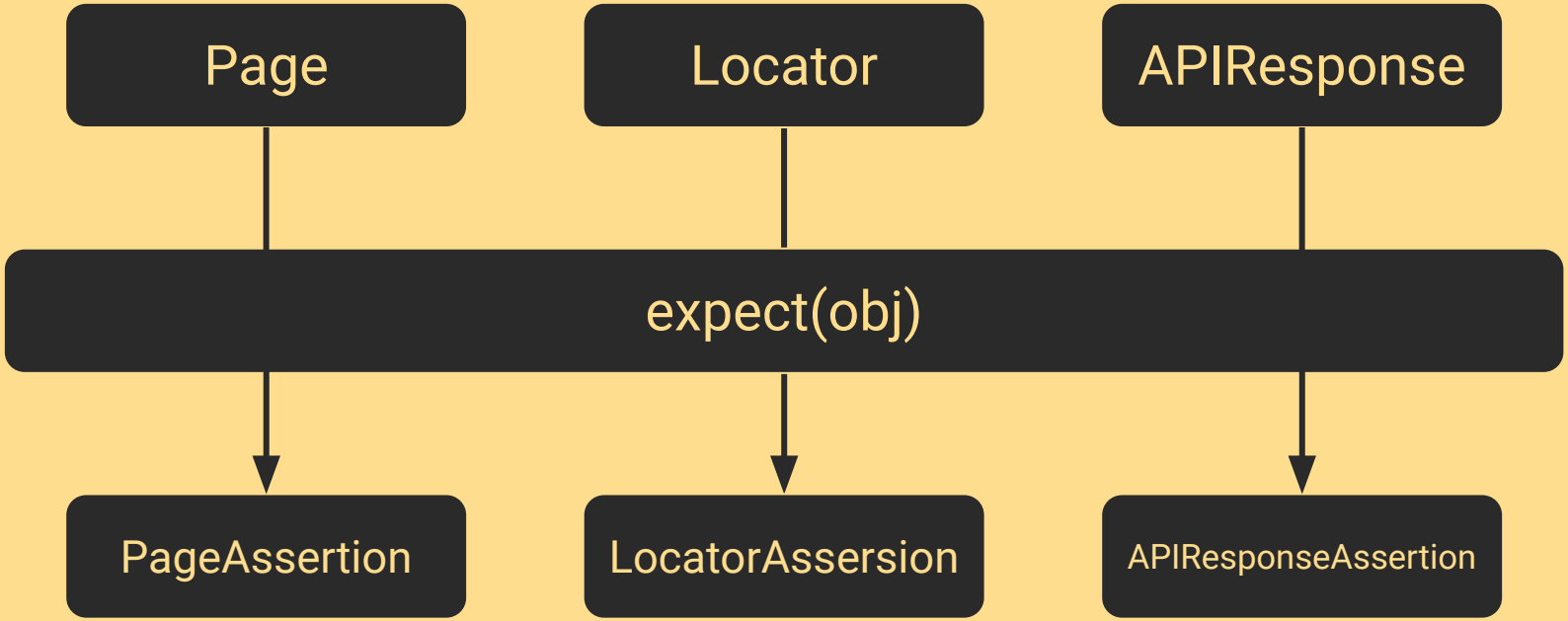
São forma de fazer validação de condições esperadas para ações.

Por exemplo, "Quando clicar no botão X, Então Y deve aparecer na tela"

E vamos checar se Y está mesmo na tela

Despacho

O expects do playwright funciona como uma forma de despacho. Ele vai oferecer métodos diferentes em relação ao objeto passado para ele



PageAssertion

```python
# exemplo_10.py
import re
from playwright.sync_api import expect, sync_playwright

with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page()

    page.goto('https://selenium.dunossauro.live/')

    expect(page).to_have_title('Curso de selenium com Python')
    expect(page).to_have_title(re.compile(r'Curso.*Python'))

    expect(page).to_have_url(re.compile(r'.*dunossauro.live'))

    page.close()
```

APIRequestAssertion

```python
# exemplo_11.py
from playwright.sync_api import expect, sync_playwright

with sync_playwright() as p:
    req = p.request.new_context()

    response = req.get('http://ddg.gg')

    expect(response).to_be_ok()
```

LocatorAssertion

```python
# exmeplo_12.py
# ...
with sync_playwright() as p:
    browser = p.chromium.launch()
    page = browser.new_page()
    page.goto(url)

    create_task(page)

    expect_header = expect(
      page.locator('.terminal-card > header')
    )
    expect_header.to_have_text('Fazer Live 222')
    expect_header.to_have_class('name')

    browser.close()
```

picpay.me/dunossauro

apoia.se/livedepython

pix.dunossauro@gmail.com

Ajude o projeto <3