



Appium e Python: iniciando automação de teste mobile

Maria Clara
Python Nordeste 2019

21.Julho.2019

Como será nosso papo - teoria

O workshop

Me apresentar e apresentar a estrutura do nosso workshop

Introdução

Um pouco sobre qualidade de software

Automação de testes

Como funciona e por onde devemos começar?

Como será nosso papo - HANDS ON!

Setup do ambiente

Vamos baixar as ferramentas necessárias, setar variáveis de ambiente e instalar alguns pacotes

Reconhecendo o terreno

Vamos ver um pouco o rostinho do Appium e criar um dispositivo android emulado (massa, né?)

Nosso primeiro teste <3

Vamos identificar elementos na tela e brincar um pouco com eles!

Como será nosso papo - finalizando...

Retrospectiva

Curtiram? Quais as dificuldades durante o workshop?

Próximos passos...

Algumas sugestões de estudo

Recebidos <3

(Surpresa!)

Um pouco sobre mim



Maria Clara Bezerra

Mestre em Ciência da Computação - UFPE-CIn
Graduação em Sistemas de Informação - UPE (Caruaru)
Certificação: BSTQB/CTFL
Eng de testes e team leader no CESAR Recife



CESAR Recife



mcsb@cesar.org.br



/clarabez



/maria-clara-bezerra/

CESAR Recife

(Onde estou há 7 meses)

Projeto Motorola

SIDIA Samsung -
Manaus/AM

(1 ano e 2 meses)

SEL BR - Customização de
operadora para LATAM

FADE/UFPE-CIn -
Motorola Parternership

(2 anos e 10 meses)

Cloud team
Platform team (com 6
meses na sede da motorola
em Chicago/IL/USA)



Qualidade de software

"Grau de conformidade de um sistema, componente ou processo com os respectivos requisitos".

(CFTL, IEEE)

Teste de sw no dia a dia

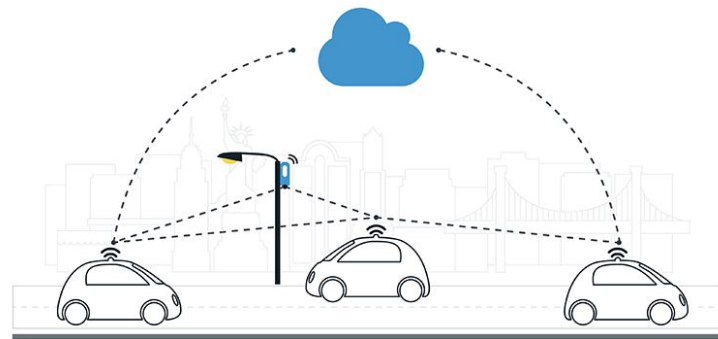
Qual a criticidade de teste de software?



Dinheiro



PSN



Carros autônomos

Etapas de desenvolvimento de:

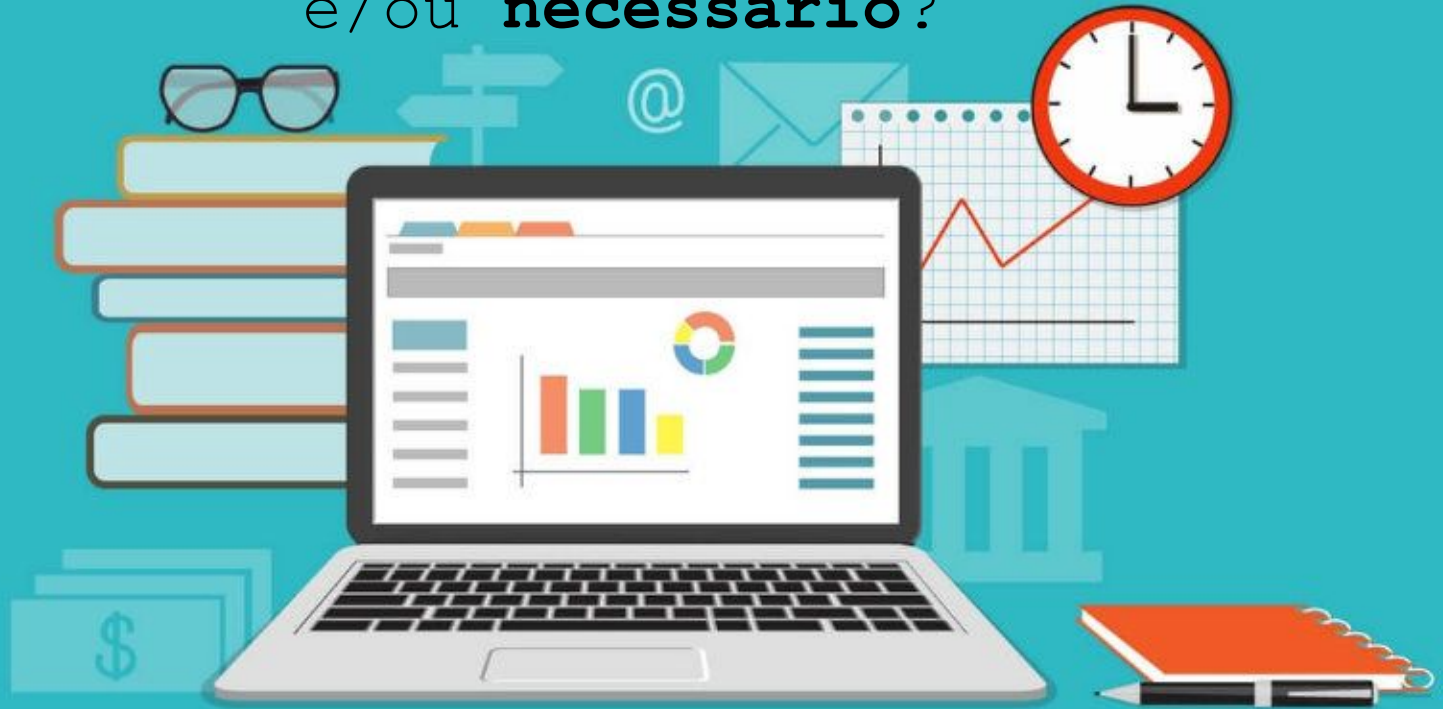
Software

-
1. Análise de requisitos
 2. Design
 3. Desenvolvimento
 4. Testes
 5. Manutenção
-

Testes

-
1. Planejamento
 2. Monitoramento e Controle
 3. Análise
 4. Modelagem
 5. Implementação
 6. Execução
 7. Conclusão
-

Por que o teste de software é **importante**
e/ou **necessário**?





E automação de testes?

"Grau de conformidade de um sistema, componente ou processo com os respectivos requisitos".

Só que agora utilizando de ferramentas e tecnologias para essa verificação. Mas existem benefícios e riscos. Temos que ter cuidado.

(CFTL, IEEE)

Benefícios da automação



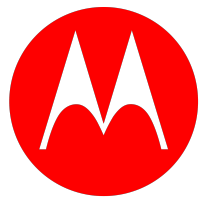
- **Redução do trabalho manual repetitivo (o time tem ganha tempo)**
- **Maior consistência e repetibilidade (máquinas não se distraem...)**
- **Avaliação mais objetiva**
- **Acesso mais fácil às informações sobre o teste executado.**

Riscos da automação



- Expectativas irreais para a ferramenta
- O tempo/custo/esforço para a implantação do ambiente podem ser subestimados
- O esforço da manutenção pode ser subestimada
- A ferramenta pode ser usada em demasia
- Cuidado para não negligenciar a interoperabilidade das ferramentas existentes no seu ambiente
- A solução é proprietária? Existe risco de perdermos o suporte?
- A solução é open-source? E se o projeto for descontinuado?

Automação mobile

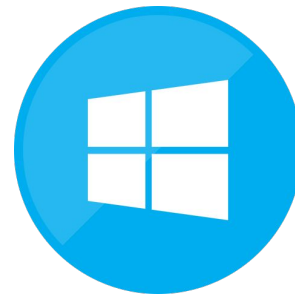


motorola



HUAWEI

NOKIA



Automação mobile



Calaba.sh



UI TESTING FOR ANDROID
espresso

Automação mobile



É uma ferramenta open-source para automação de aplicações nativas, mobile web e híbridas nas plataformas Android, iOS e Windows.

Utiliza **UIAutomator** e **Webdriver** em sua arquitetura.

Como será nosso papo - HANDS ON!

Setup do ambiente

Vamos baixar as ferramentas necessárias, setar variáveis de ambiente e instalar alguns pacotes

Reconhecendo o terreno

Vamos ver um pouco o rostinho do Appium e criar um dispositivo android emulado (massa, né?)

Nosso primeiro teste <3

Vamos identificar elementos na tela e brincar um pouco com eles!

Setup

Download:

- Android Studio e Android SDK - <https://developer.android.com/studio>
- PyCharm - <https://www.jetbrains.com/pycharm/>

Instalar (é bom não usar sudo):

```
>> brew install node
```

```
>> npm install -g appium
```

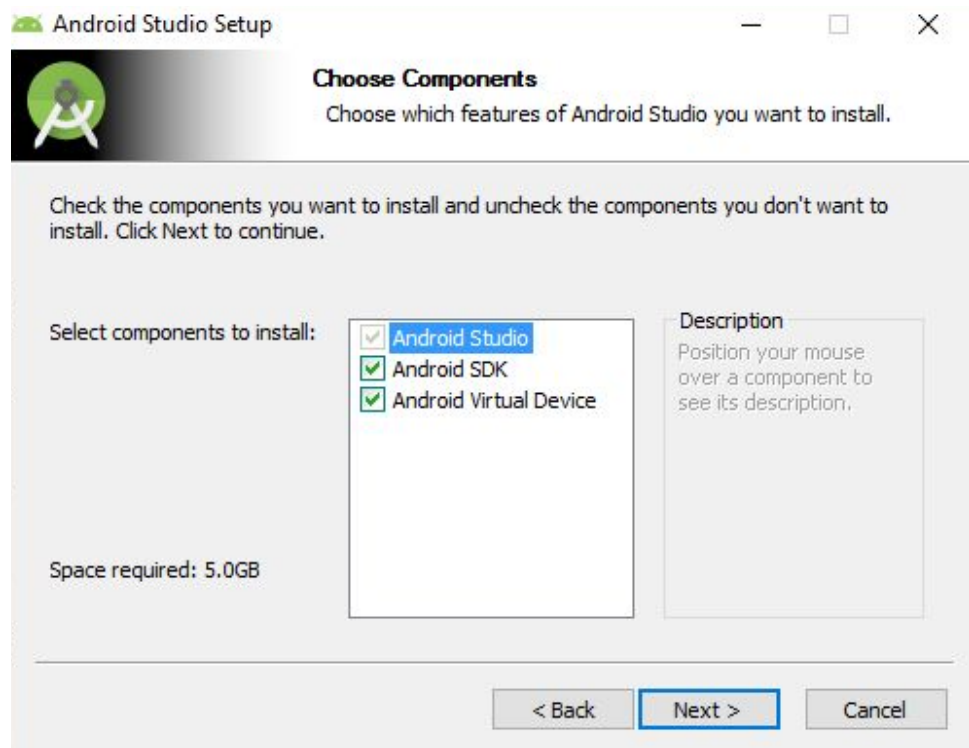
Issues conhecidas:

Problema de permissão?

```
>> sudo npm install -g appium --unsafe-perm=true --allow-root
```

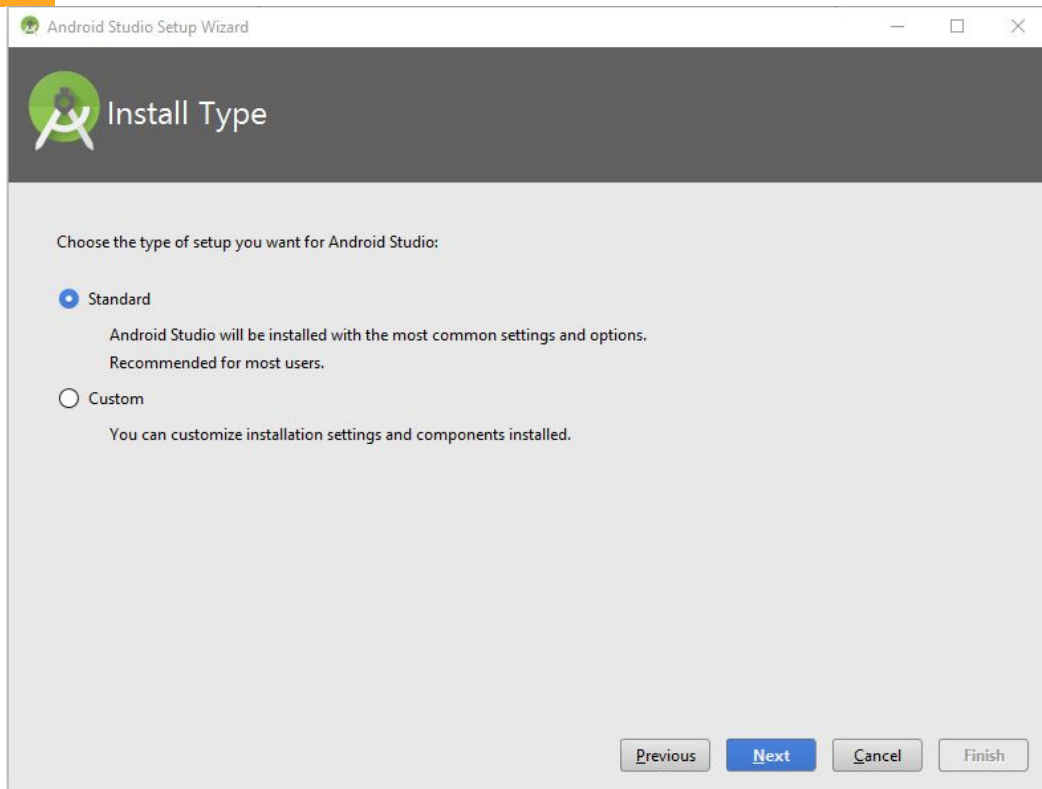
Setup

No começo do setup de instalação do Android Studio, temos que garantir que a opção "Android Virtual Device" está marcada.



Setup

A gente pode deixar a opção "Standard" marcada mesmo :)

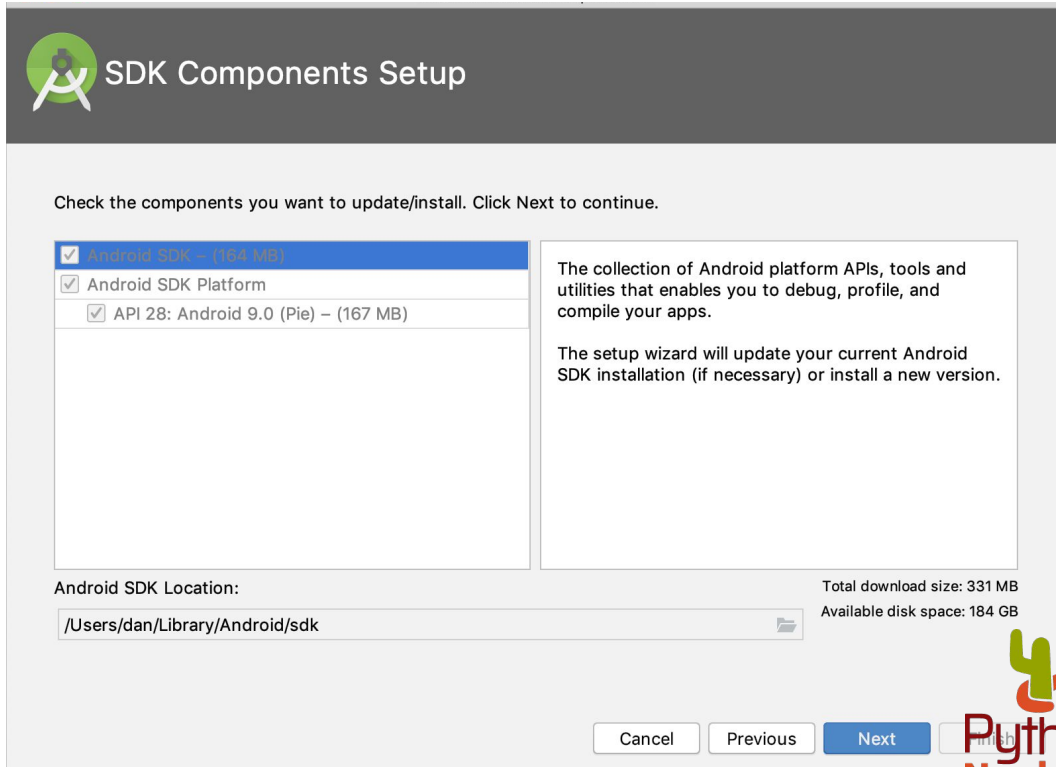


Setup

Agora é hora de já deixar garantido uma versão do Android vai ser baixada para ser instalada no nosso futuro device emulado. Então vamos marcar a opção do API que ele sugerir.

É o momento também de indicarmos o location do nosso sdk. Mas podemos setar mais adiante.

Na próxima tela pode aparecer mais de 1 Android version pra gente baixar. O Android P já é o suficiente <3



Setup

Variáveis de ambiente:

```
export ANDROID_HOME=/Users/sergei/Library/Android/sdk
export PATH=$ANDROID_HOME/platform-tools:$PATH
export PATH=$ANDROID_HOME/tools:$PATH
export PATH=$ANDROID_HOME/build-tools:$PATH
export JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_112.jdk/Contents/Home
export PATH=$JAVA_HOME/bin:$PATH
```

Validando se tá certinho:

```
echo $ANDROID_HOME
echo $JAVA_HOME
```

Dica:

Onde tá meu JAVA?

```
>> which java
```

Setup

Será que isso tudinho funcionou?



```
>> npm install -g appium-doctor --android
```

```
>> appium-doctor
```

Setup

Instanciar um device emulado com o Android Studio

- Baixar uma versão de Android pro device manager

Alguns comandos ADB (Android Debug Bridge)

```
>> adb devices
```

```
>> adb reboot
```

Agora nosso device tá prontinho! :)

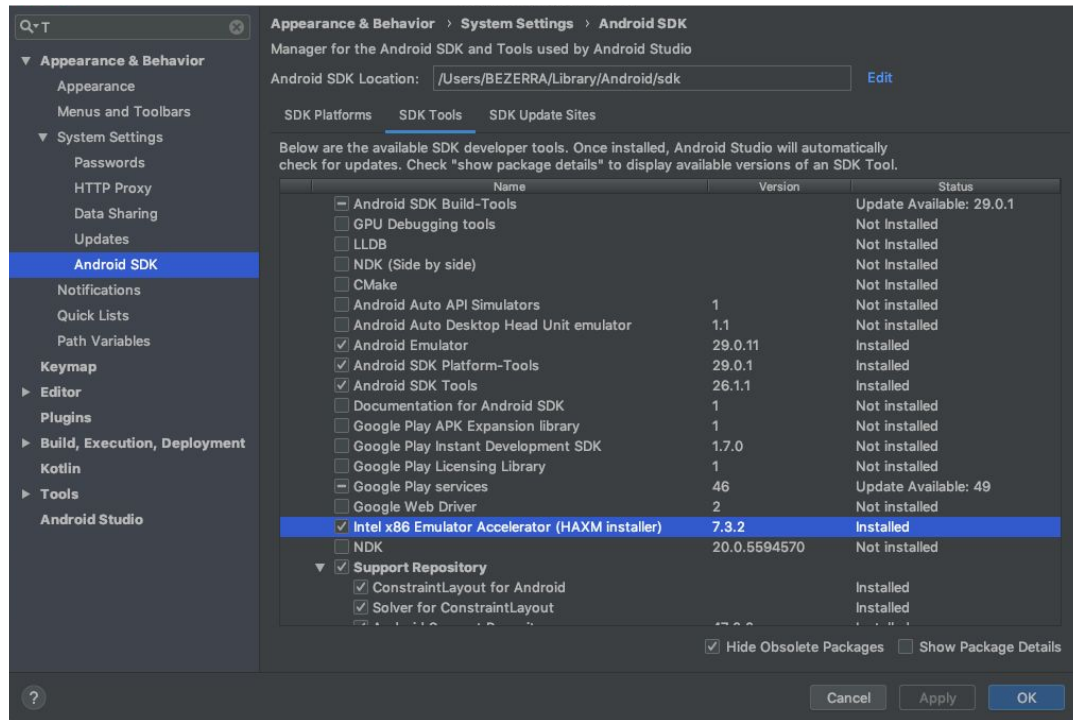


Setup

Alguns probleminhas que podem acontecer:

Mensagem de erro com algo tipo:
"emulator: ERROR: x86 emulation requires hardware acceleration"

Vá em: Tools > SDK Manager > Android SDK > SDK tools > Marque a opção "Intel x86 Emulator Accelerator..." > OK



Setup - Checklist

- **Download** das ferramentas Android Studio e Pycharm
- **Configuração** das variáveis de ambiente
- **Configuração** do Android Emulator
- Alguns **comandos** básicos de ADB
- Alguns possíveis **problemas** durante essa etapa





Bora instalar uma aplicação no nosso device?



Setup

Appium desktop download

- Opção mais simples de iniciar o serviço

"Desired Capabilities" no Appium:

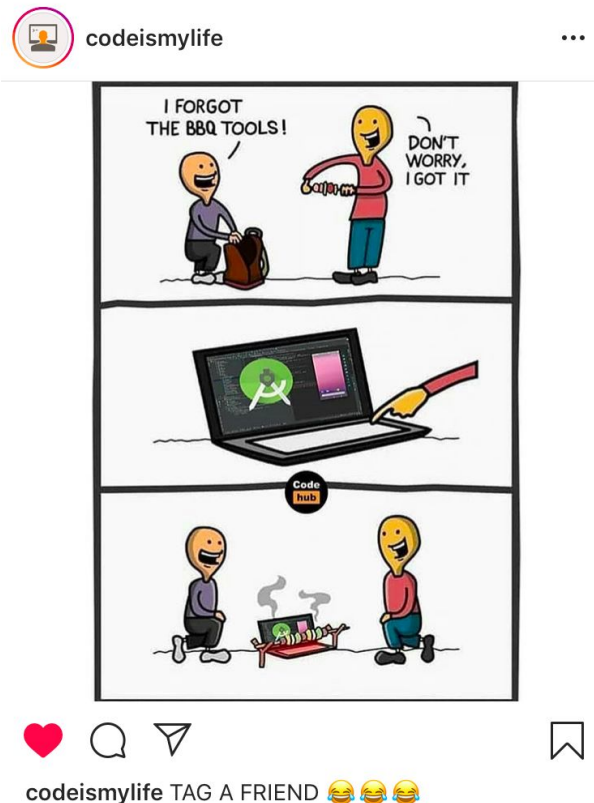
- Maneira de identificar a aplicação em teste

Instalar uma aplicação no device emulado

- `>> adb install Name.apk`

Abrir a aplicação através do Appium

- O começo de tudo! <3



Setup

Agora vamos identificar dois elementos importantes:

- AppPackage: Pacote geral da aplicação
- AppActivity: Cada activity é uma tela da aplicação

```
>> adb shell dumpsys window windows | grep -E 'mCurrentFocus'
```

Setup

Desired Capabilities via python:

```
from appium import webdriver
```

```
desired_cap = {  
    "platformName": "Android",  
    "deviceName": "Appium1",  
    "appPackage": "com.b2w.americanas",  
    "appActivity": "com.b2w.americanas.activity.MainActivity"  
}
```

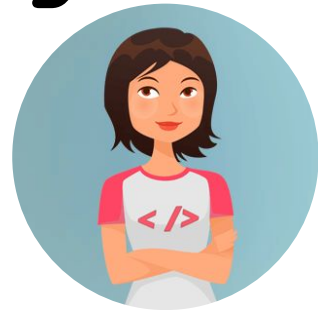
```
driver = webdriver.Remote('http://localhost:4723/wd/hub', desired_cap)
```

Bora melhorar esse código?



Bora melhorar esse código?

- **pytest**: um framework para facilitar nossa vida em automação de testes
- **Page Object Model (POM)**: padrão de projeto para automação



Como instalar o pytest?

```
>> pip install -U pytest
```

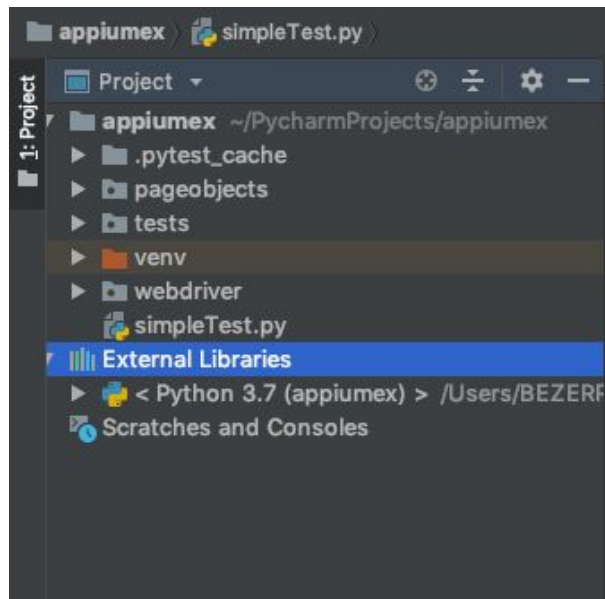
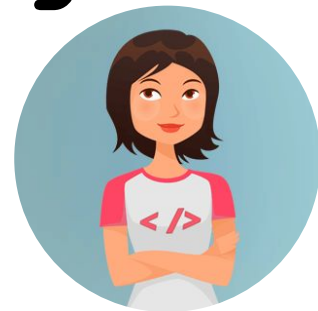
ou

```
>> easy_install -U pytest
```

Bora melhorar esse código?

Bora criar os seguintes folders dentro do nosso projeto:

- pageobjects
- tests
- webdriver



Bora melhorar esse código?

WEBDRIVER

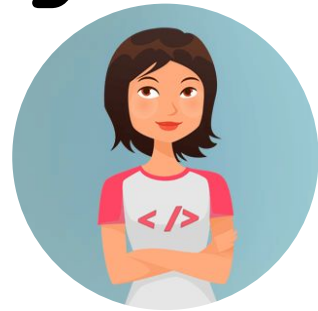
```
from appium import webdriver
```

```
class Driver:
```

```
    def __init__(self):
```

```
        desired_cap = {
            "platformName": "Android",
            "deviceName": "Appium1",
            "appPackage": "com.novapontocom.casasbahia",
            "appActivity": "br.com.viavarejo.feature.home.HomeActivity"
        }
```

```
        self.instance = webdriver.Remote('http://localhost:4723/wd/hub', desired_cap)
```



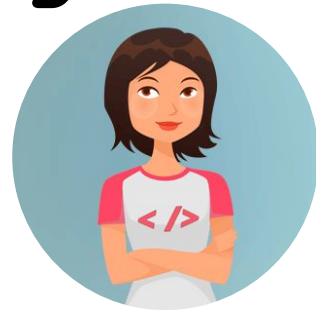
Bora melhorar esse código?

PAGEOBJECT

```
from appium.webdriver.common.mobileby import MobileBy
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
```

```
class CasasBahiaScreen:
    def __init__(self, driver):
        self.driver = driver
        self.home = WebDriverWait(self.driver.instance, 5).until(EC.visibility_of_element_located((
            MobileBy.XPATH, '//android.widget.FrameLayout[@content-desc="Conta"]/android.widget.ImageView')))
        self.produtos = WebDriverWait(self.driver.instance, 5).until(EC.visibility_of_element_located((
            MobileBy.XPATH, '//android.widget.FrameLayout[@content-desc="Produtos"]/android.widget.ImageView')))
```

```
def go_produtos(self):
    self.produtos.click()
```



Bora melhorar esse código?

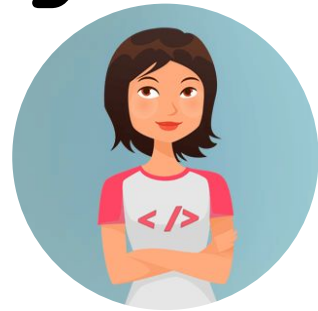
TESTS

```
import unittest
from webdriver.webdriver2 import Driver
from pageobjects.casasbahiascreen import CasasBahiaScreen
```

```
class CasasBahiaMelhorado(unittest.TestCase):
    def setUp(self):
        self.driver = Driver()
```

```
    def test_AppLaunch(self):
        launch = CasasBahiaScreen(self.driver)
        launch.go_produtos()
```

```
if __name__ == '__main__':
    suite = unittest.TestLoader().loadTestsFromTestCase(CasasBahiaMelhorado)
```



Sugestão para próximos passos

- **PEP8** - Ótimo padrão para escrita e revisão de código
- **PyTest** - Framework do Python para automação de testes
- **Unittest** - Framework para testes unitários
- **MoT** - Comunidade de teste de software de Recife

Referências

1. [Appium official page](#)
2. [Android Official page](#)
3. [Appium official repository on Github](#)
4. [Pytest official page](#)
5. [Evozi apk-downloader](#)
6. [Meetups of Ministry of Testing - Recife](#)
7. [Syllabus Apostila CTFL - BSTQB](#)
8. [Blog do CESAR School](#)
9. [PEP8 - Padrão oficial](#)
10. [Pytest](#)
11. [Page Object Model \(POM\)](#)
12. [Unittest](#)



www.cesar.org.br

Rua Bione. 220 | Cais do Apolo |
Bairro do Recife Recife/PE | CEP:
50.030-390 - Brasil
contato@cesar.org.br
+55 81 3425.4700



C . e . S . A . R
centro de estudos e sistemas
avançados do recife



mas vê gente neste mesmo futuro.



Obrigada! <3



Appium e Python: iniciando automação de teste mobile

Maria Clara
Python Nordeste 2019

mcsb@cesar.org.br

21.Julho.2019