

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Testes de Software

**TBL 1, FASE 3 - VERIFICAÇÃO E VALIDAÇÃO NA API
SPACEX**

<https://github.com/VictorDeon/SpaceX-API>

Orientadores:

Professor: Ricardo Ajax
Monitora: Amanda Bezerra

Brasília, DF
14 de outubro de 2019





1. GRUPO 07 - INTEGRANTES E CONTRIBUIÇÕES

Tabela 1 — Integrantes e Contribuições ao TBL 1, fase 3

Integrante	Matrícula	Contribuição
Amanda Pires	15/0004796	100%
André Pinto	17/0068251	100%
Ivan Dobbin	17/0013278	100%
Leonardo Medeiros	17/0038891	95%
Lieverton Silva	17/0039251	100%
Renan Cristyan	17/0044386	85%
Welison Regis	17/0024121	100%
Wictor Girardi	17/0047326	85%

Fonte: dos autores, 2019.

2. WALKTHROUGH

O Walkthrough, técnica similar as inspeções, baseia-se em um procedimento realizado por um grupo de três a cinco pessoas com o objetivo de detectar erros em uma aplicação. O método utiliza pequenos casos de testes, representados por entradas do programa ou módulo, que são testados durante a duração do Walkthrough em que consiste na execução mental passo a passo da lógica do programa por parte do participante. (MAYERS, 2004)

Conforme Mayers (2004), o método de Walkthrough não é eficiente na detecção de erros de alto nível, como erros no processo de análise de requisitos. Porém, em outro contexto, o Walkthrough é performático em encontrar entre 30% e 70% de erros de lógica de implementação ou de código em programas típicos. Nesse sentido, o processo de Walkthrough é vantajoso, visto que o erro encontrado pode ser precisamente localizado no código. (MAYERS, 2004)

Além das vantagens citadas, o Walkthrough é um método interessante pois possibilita realizar os testes com pessoas que não sejam os próprios autores do artefato, o que corrobora com alguns princípios de Teste de Software definido por Mayers, como, por exemplo, o princípio 2: “Um programador deve evitar tentar testar seu próprio programa”. (MAYERS, 2004)

3. PSEUDO-CÓDIGO

a. SpaceX-API/spaceX.py

```
SpaceX(object)
    ▷ Menu to choice the consult.

NEXT_LAUNCH ← 1
LATEST_LAUNCH ← 2
UPCOMING_LAUNCHES ← 3
PAST_LAUNCHES ← 4

function run(cls)
```

- ▷ Shows the options for the user
- ▷ :return: chosen option

```
while true do
    print "0 que você deseja visualizar?"

    write "1) Próximo Lançamento"      ▷ next_launch
    write "2) Último Lançamento"       ▷ latest launch
    write "3) Próximos Lançamentos"    ▷ upcoming launches
    write "4) Lançamentos Passados"    ▷ past launches
    write "5) Sair"

    try
        write "Insira uma opcao: "
        option ← read
    except ValueError
        write "Você deve inserir somente números inteiros de
preferencia de 1 a 5"
        option ← 0
    end try

    if option < 1 or option > 5 then
        write "Essa opção não existe, por favor insira uma
opção válida."
    else if option == 5 then
        break
    else
        cls.show_result(option)
        write "Deseja sair da aplicação? (S/N): "
        answer ← read
        if lower answer start with 's' then
            break
        end if
    end if
end while
end function
```



```
function show_result(cls, option)
  ▷ Run a specific option inserted
  ▷ :param option: Option to visualize some datas
```

```
    if option == SpaceX.NEXT_LAUNCH then
      cls.next_launch()
    else if option == SpaceX.LATEST_LAUNCH then
      cls.latest_launch()
    else if option == SpaceX.UPCOMING_LAUNCHES then
      cls.upcoming_launches()
    else if option == SpaceX.PAST_LAUNCHES then
      cls.past_launches()
    else
      write "Opção invalida"
    end if
  end function
```

```
function close()
  ▷ Close the program.

  write "Finalizando o programa..."
end function
```

```
function next_launch()
  ▷ Returns the next launch
```

```
    connection ←
Connect("https://api.spacexdata.com/v3/launches/next")
```

```
    write connection.result
  end function
```

```
function upcoming_launches()
  ▷ Returns the closest upcoming launches
```

```
    connection ←
Connect("https://api.spacexdata.com/v3/launches/upcoming")
```



```
    for result in connection.result do
      write result
    end for
  end function

  function latest_launch()
    ▷ Returns the latest launch

    connection ←
Connect("https://api.spacexdata.com/v3/launches/latest")

    write connection.result
  end function

  function past_launches()
    ▷ Returns the past launches

    connection ←
Connect("https://api.spacexdata.com/v3/launches/past")

    for result in connection.result do
      write result
    end for
  end function
```

b. SpaceX-API/connections/api_connection.py

```
Connect(object)

  ▷ Connect to api

  function constructor(self, url, headers=None, params=None)

    ▷ Connection constructor

    if headers then
      self.headers ← headers
    else
```

```
        self.headers ← {'Accept': 'application/json'}
    end if

    try
        if params then
            self.response ← requests.get(url, self.headers,
params←params)
        else
            self.response ← requests.get(url, self.headers)
        end if

        self.result ← self.response.json()
    except requests.exceptions.RequestException
        write "Ocorreu um erro na comunicação com a API SpaceX"
    end try
end function
```

```
function result(self)
```

- ▷ Get a JSON result of request.
- ▷ :return: JSON result

```
if type(self.__result) == dict then
    return Launch(
        flight_number ← self.result.get('flight_number'),
        mission_name ← self.result.get('mission_name'),
        rocket ← self.result.get('rocket').get('rocket_name'),
        rocket_type←self.result.get('rocket').get('rocket_type'),
        launch_success←self.result.get('launch_success'),
        launch_date←self.result.get('launch_date_utc'),
        launch_year←self.result.get('launch_year')
    )
end if
launchs ← []
for result in self.result do
    launchs.append(
        Launch(
```

```
        flight_number ← result.get('flight_number'),
        mission_name ← result.get('mission_name'),
        rocket ← result.get('rocket').get('rocket_name'),
        rocket_type ←
result.get('rocket').get('rocket_type'),
        launch_success ← result.get('launch_success'),
        launch_date ← result.get('launch_date_utc'),
        launch_year ← result.get('launch_year')
    )
)

    return launches
end function

function response(self)

    ▷ Get the request response
    ▷ :return: response

    return self.response
end function
```

c. SpaceX-API/models/launch.py

```
Launch(object)
    ▷ SpaceX Launch

    function constructor(self, flight_number, mission_name, rocket,
        rocket_type, launch_success,
        launch_year, launch_date)

        ▷ Constructor

        self.flight_number ← flight_number
```



```
self.mission_name ← mission_name
self.launch_date ← launch_date
self.launch_year ← launch_year
self.rocket ← rocket
self.rocket_type ← rocket_type
self.launch_success ← launch_success
end function

function str(self)

  ▷ Give a object a string representation
  ▷ :return: string representation of object

  if self.__launch_success is not None then
    return "{0}\n{1}\n{2}\n{3}\n{4}\n{5}\n".format(
      self.flight_number, self.mission_name, self.rocket,
      self.launch_year, self.launch_date, self.launch_success
    )
  end if

  return "{0}\n{1}\n{2}\n{3}\n{4}\n".format(
    self.flight_number, self.mission_name, self.rocket,
    self.launch_year, self.launch_date
  )
end function

function flight_number(self)

  ▷ Get the flight number of launch.
  ▷ :return: flight number

  return "Número do Voo: {0}".format(self.flight_number)
end function

function mission_name(self)

  ▷ Get the mission name
```

```
    ▷ :return: mission name

    return "Missão: {0}".format(self.mission_name)
end function

function launch_date(self)
    ▷ Get the launch date in dd/mm/yyyy às hh:mm
    ▷ :return: launch date

    date ← datetime.strptime(self.__launch_date,
"%Y-%m-%dT%H:%M:%S.%fZ")

    return "Data de Lançamento (UTC):
{0}".format(date.strftime("%d/%m/%Y às %H:%M"))
end function

function launch_year(self)

    ▷ Get the launch year
    ▷ :return: launch year

    return "Ano de Lançamento: {0}".format(self.__launch_year)
end function

function rocket(self)

    ▷ Get the rocket
    ▷ :return: rocket

    return "Foguete: {0} ({1})".format(self.__rocket,
self.__rocket_type)
end function

function launch_success(self)

    ▷ Verify if the launch happened successfully!
    ▷ :return: Success message or Fail message.
    if self.__launch_success then
```

```
        return "Lançamento realizado com sucesso!"
    end if

    return "Lançamento falhou!"
end function
```

4. INSTANCIÇÃO DE PSEUDO-CÓDIGO

Entrada 1:

1. Executa main;
2. main chama run da SpaceX;
3. Começa um loop;
4. Imprime opções na tela;
5. Lê valor e atribui na variável option (usuário digitou 1);
6. Valida se o valor é inteiro;
7. Verifica se o valor está entre 1 e 5;
8. Executa show_result(option);
9. Imprime dados do próximo lançamento;
10. Imprime se deseja sair ou não;
11. Lê valor (usuário digitou 'S');
12. Verifica se a primeira letra começa com 's';
13. Quebra o loop;
14. Fim do programa.

5. DEFEITOS

5.1. PROBLEMA

Defeito de Software é qualquer imperfeição ou inconsistência no produto do **software** ou em seu processo, um **defeito** é também uma não conformidade. No programa utilizado, foram encontrados os seguintes defeitos:

- a. A aplicação não valida a entrada para finalizar o programa.
- b. No arquivo `spaceX.py`, na linha 24, inicia-se um laço de repetição infinito (`while(true)`).
 - i. Solução: Declarar uma variável para controle da duração do loop.
- c. A função `'time.sleep()'` aparece após o print da finalização do programa.
 - i. Solução: Remover as chamadas `'time.sleep()'`.
- d. No python3 não se deve mais haver herança de 'object' como era feito no python2.
- e. Nome de módulos em python devem seguir o padrão de nome snake_case
- f. Nas linhas 49 e 51 é feita a leitura do comando do usuário, sendo necessária observar se sua resposta é 'S' ou 'N', porém não é corretamente validada a resposta.
 - i. Solução: uma possível melhoria é definir que para a opção 'S' seja aceito apenas 'S/s' e que também seja verificado 'N/n', caso não esteja dentro das opções, informar o usuário.
- g. A aplicação quebra caso não exista comunicação com a internet. Não trata uma função que tem raise error.
 - i. Solução: utilizar uma função try except e informar ao usuário a falta de comunicação com a internet, invés de deixar a aplicação quebrar.

5.2. QUALIDADE

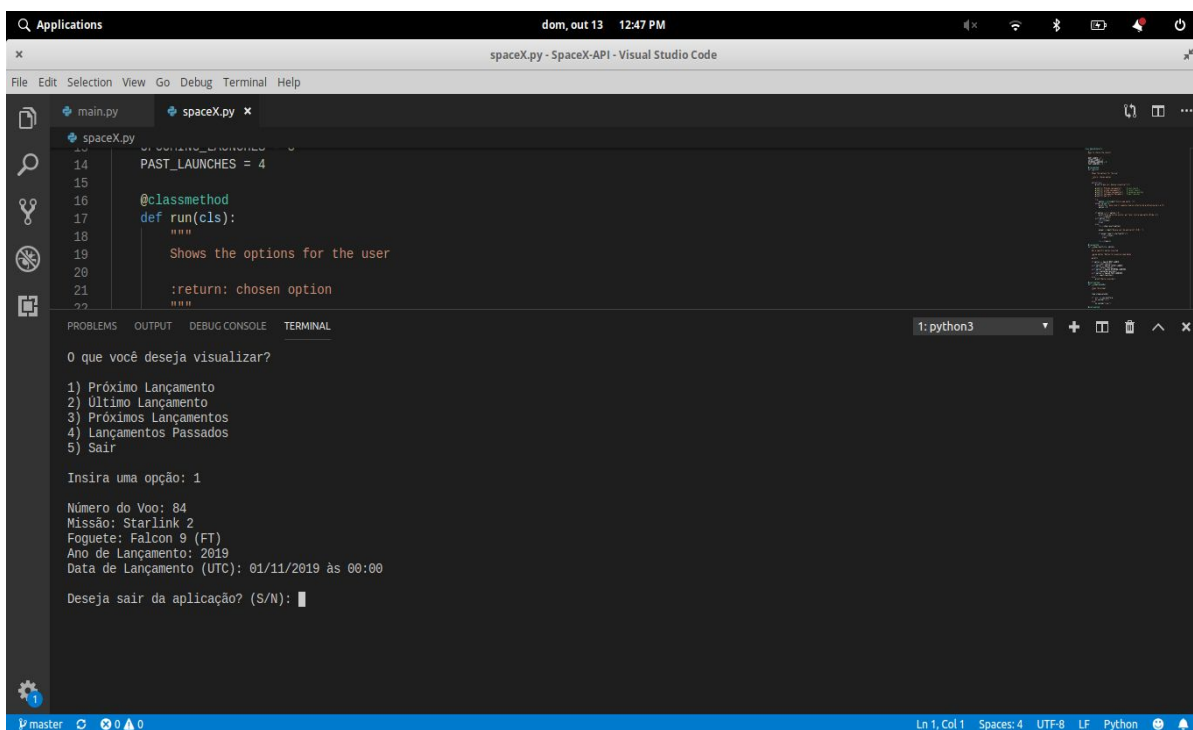
Para um código ser considerado de qualidade, ele tem que ter boa manutenibilidade (analisabilidade, modificabilidade, estabilidade, testabilidade, conformidade) manter os padrões definidos para aquele projeto, realizar com

eficiência (comportamento em relação ao tempo, utilização de recursos, conformidade) o que é solicitado pelo contexto do problema e possuir confiabilidade (maturidade, tolerância a falhas e recuperabilidade).

6. EVIDÊNCIAS DE EXECUÇÃO

A execução do código foi feita em uma máquina Linux, sistema operacional Elementary OS. Foi utilizada a versão 3.6.8 do Python e a IDE VsCode para a análise do programa. Abaixo, seguem as evidências da execução do projeto e exemplos de algumas entradas.

Entrada “1”:



```
0 que você deseja visualizar?
1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair

Insira uma opção: 1

Número do Voo: 84
Missão: Starlink 2
Foguete: Falcon 9 (FT)
Ano de Lançamento: 2019
Data de Lançamento (UTC): 01/11/2019 às 00:00

Deseja sair da aplicação? (S/N):
```

Qualquer entrada que comece diferente de “S” ou “s” é suficiente para permanecer na aplicação.



Entrada “2”:

```
0 que você deseja visualizar?
1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair

Insira uma opção: 2

Número do Voo: 83
Missão: Amos-17
Foguete: Falcon 9 (FT)
Ano de Lançamento: 2019
Data de Lançamento (UTC): 06/08/2019 às 22:52
Lançamento realizado com sucesso!

Deseja sair da aplicação? (S/N):
```

Entrada “3”:

```
0 que você deseja visualizar?
1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair

Insira uma opção: 3

Número do Voo: 84
Missão: Starlink 2
Foguete: Falcon 9 (FT)
Ano de Lançamento: 2019
Data de Lançamento (UTC): 01/11/2019 às 00:00
-----

Número do Voo: 85
Missão: Starlink 3
Foguete: Falcon 9 (FT)
Ano de Lançamento: 2019
Data de Lançamento (UTC): 01/11/2019 às 00:00
-----

Número do Voo: 86
Missão: JCSat 18 / Kacific 1
Foguete: Falcon 9 (FT)
Ano de Lançamento: 2019
Data de Lançamento (UTC): 11/11/2019 às 00:00
-----
```



Entrada “4”:

```
Q Applications dom, out 13 1:02 PM
x SpaceX.py - SpaceX-API - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
main.py SpaceX.py x
15
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: python3
0 que você deseja visualizar?
1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair
Insira uma opção: 4
Número do Voo: 1
Missão: FalconSat
Foguete: Falcon 1 (Merlin A)
Ano de Lançamento: 2006
Data de Lançamento (UTC): 24/03/2006 às 22:30
Lançamento falhou!
-----
Número do Voo: 2
Missão: DemoSat
Foguete: Falcon 1 (Merlin A)
Ano de Lançamento: 2007
Data de Lançamento (UTC): 21/03/2007 às 01:10
Lançamento falhou!
-----
Número do Voo: 3
Missão: Trailblazer
Foguete: Falcon 1 (Merlin C)
Ano de Lançamento: 2008
Data de Lançamento (UTC): 02/08/2008 às 03:34
Lançamento falhou!
```

Entrada “5”:

```
Q Applications dom, out 13 1:02 PM
x SpaceX.py - SpaceX-API - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
main.py SpaceX.py x
15
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: bash
0 que você deseja visualizar?
1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair
Insira uma opção: 5
Finalizando o programa...
amanda@amanda:~/Documents/desenvolvimento/workspace/SpaceX-API$
```



O programa faz a validação de entradas inválidas:

```
0 que você deseja visualizar?
1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair

Insira uma opção: q
Você deve inserir somente números inteiros de preferencia de 1 a 5
Essa opção não existe, por favor insira uma opção válida.
```

```
0 que você deseja visualizar?
1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair

Insira uma opção: 9
Essa opção não existe, por favor insira uma opção válida.
```


Sem internet:

```
andrelucax@andrelucax:/tmp/SpaceX-API$ python3 main.py
0 que você deseja visualizar?

1) Próximo Lançamento
2) Último Lançamento
3) Próximos Lançamentos
4) Lançamentos Passados
5) Sair

Insira uma opção: 1

Ocorreu um erro na comunicação com a API SpaceX
Traceback (most recent call last):
  File "main.py", line 13, in <module>
    main()
  File "main.py", line 9, in main
    SpaceX.run()
  File "/tmp/SpaceX-API/spaceX.py", line 47, in run
    cls.__show_result(option)
  File "/tmp/SpaceX-API/spaceX.py", line 67, in __show_result
    cls.__next_launch()
  File "/tmp/SpaceX-API/spaceX.py", line 107, in __next_launch
    print(connection.result)
  File "/tmp/SpaceX-API/connections/api_connection.py", line 40, in result
    if type(self.__result) == dict:
AttributeError: 'Connect' object has no attribute '_Connect__result'
andrelucax@andrelucax:/tmp/SpaceX-API$
```



7. REFERÊNCIAS

- [1] MYERS, G. J. **The Art of Software Testing**. 2.ed. New Jersey, USA. John Wiley & Sons, 2004.
- [2] CORMEN, T., LISERSON, C., RIVEST, R., et al. **Algoritmos Teoria e prática**. . 2nd Ed. S.I., Elsevier, 2002.