


IMD0033 - Probabilidade

Aula 02 - Plataformas de Desenvolvimento

Ivanovitch Silva
Fevereiro, 2018



Agenda

- Python vs R
- Anaconda
- Meu primeiro notebook
- Controle de Versão
- Aquecimento



DataCamp
Learn data analysis for free,
interactively

DATA SCIENCE WARS



VS.



python

R and Python are waging war:
while both programming languages are gaining prominence
in the data analytics community, they are fighting
to become data scientists' language of choice.

Which side are you taking?



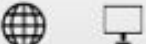







<https://goo.gl/2mQm2K>


If you come from a C.S./developer background, you'll probably feel more comfortable with Python. On the other hand, if you come from a statistics/analyst background, R will likely be more intuitive























R vs Python for Data Science

Summary of Modern Advances

elitedatascience.com

Language Rank	Types	Spectrum Ranking
1. C		100.0
2. Java		98.1
3. Python		98.0
4. C++		95.9
5. R		87.9
6. C#		86.7
7. PHP		82.8
8. JavaScript		82.2
9. Ruby		74.5
10. Go		71.9



Language Rank	Types	Spectrum Ranking
1. Python	 	100.0
2. C	  	99.7
3. Java	  	99.5
4. C++	  	97.1
5. C#	  	87.7
6. R		87.7
7. JavaScript	 	85.6
8. PHP		81.2
9. Go	 	75.1
10. Swift	 	73.7

IEEE Spectrum - Jul 2017 <https://goo.gl/HSPLWe>



Version 3.x (<https://www.python.org/downloads/>)



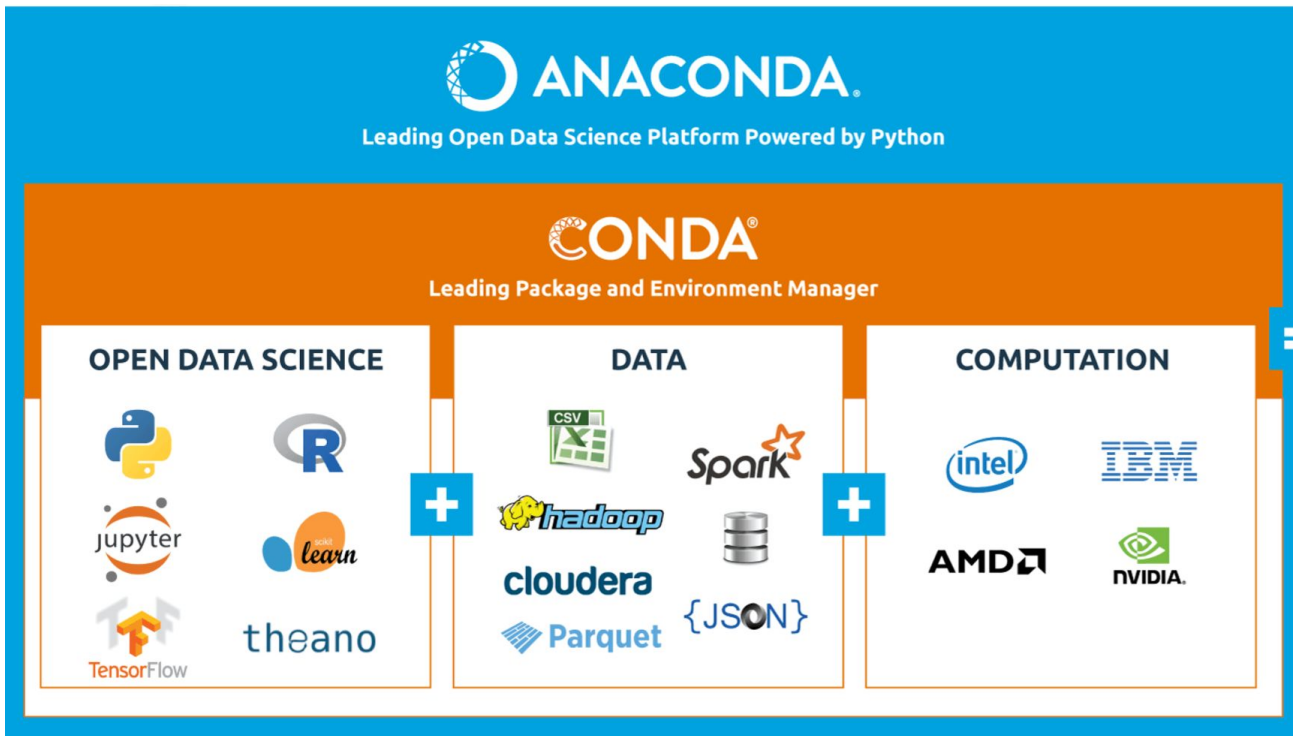
ANACONDA®

Modern open source analytics platform
powered by Python



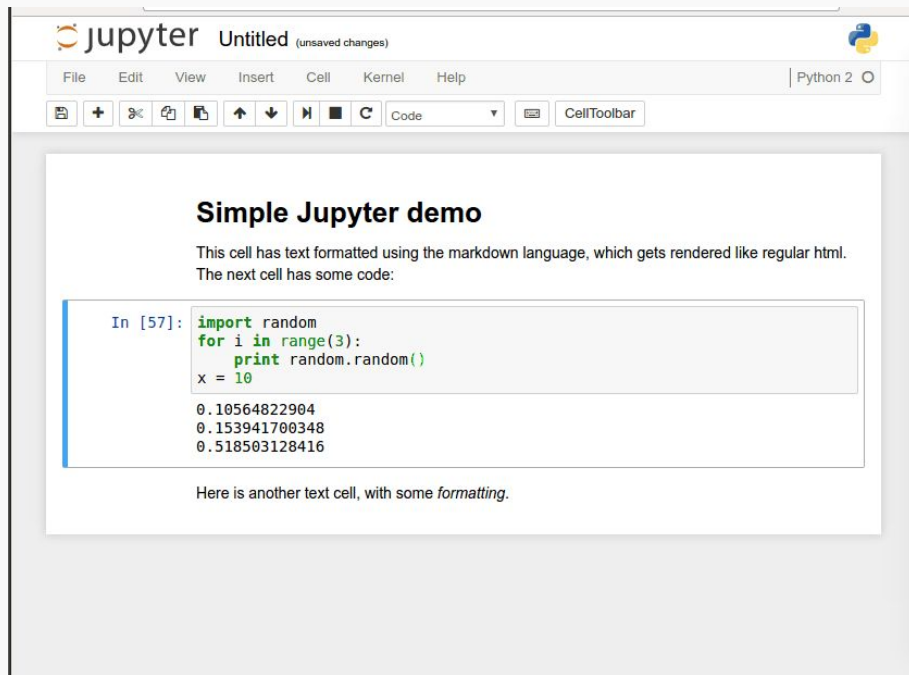
<https://www.continuum.io/downloads>

Por que Anaconda?



O que é um notebook Jupyter?

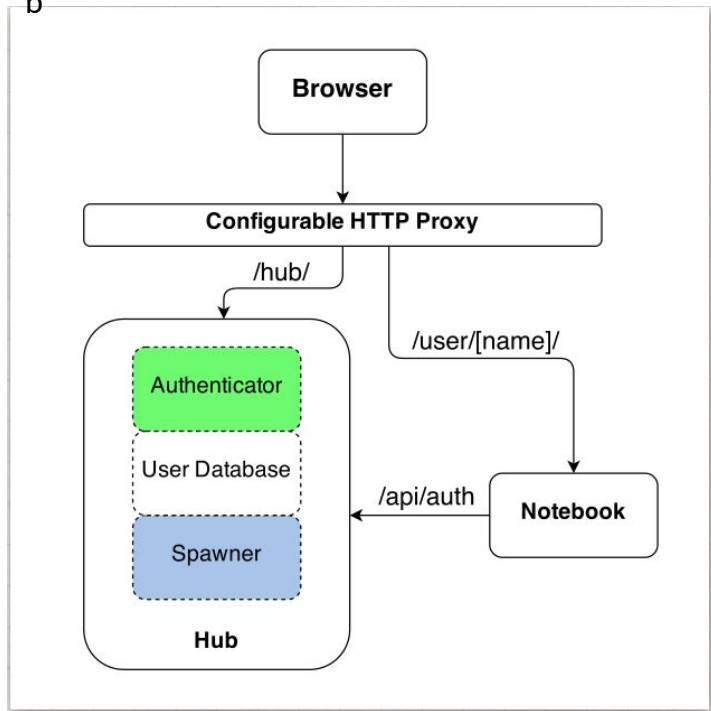
Uma mistura de código e elementos flexíveis (texto, figuras, links, equações, etc)



Além de JULia, PYThon e R (JUPYTER), o notebook Jupyter tem suporte para dezenas de outras linguagens..
<https://github.com/jupyter/jupyter/wiki/Jupyter-kernels>

JupyterHub

<https://github.com/jupyterhub/jupyterhub>



JupyterHub pode ser utilizado como um servidor de notebooks para uma turma de estudantes, curso corporativo, grupo de estudo ou pesquisa, etc.



ANACONDA CLOUD

Where packages, notebooks, and environments are shared.

Powerful collaboration and package management for open source and private projects.

Public projects and notebooks are always free.

Private plans start at \$7/month.

[Sign Up](#)[Sign In](#)

New to Anaconda Cloud? Sign up!



Use at least one lowercase letter, one numeral, and seven characters.



☒ I accept the [Terms & Conditions](#)

[Sign up!](#)

By clicking "Sign up!" you agree to our privacy policy and terms of service. We will send you account related emails occasionally.



Home



Environments



Projects (beta)



Learning



Community

Documentation

Developer Blog

Feedback



Applications on

base (root)

Channels



jupyterlab

0.31.5

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Launch



notebook

5.4.0

Web-based, interactive computing notebook environment to write and run human-readable docs while describing the data analysis.

Launch



qtconsole

4.3.1

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Launch

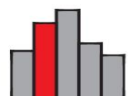


spyder

3.2.6

Scientific PYTHON Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Launch



glueviz

0.12.0

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Install



orange3

3.4.1

Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

Install

GitHub



Aprenda Git em poucos minutos

Introdução ao Git

script.py

```
if __name__ == "__main__":  
    print("Welcome to a script!")
```

você

```
import math  
print(10 + 10)  
if __name__ == "__main__":  
    print("Welcome to a script!")
```

colega

```
if __name__ == "__main__":  
    print("Welcome to a script!")  
    print("Here's my amazing contribution to this project!")
```

união

```
import math  
print(10 + 10)  
if __name__ == "__main__":  
    print("Welcome to a script!")  
    print("Here's my amazing contribution to this project!")
```

Instalando

Downloads



Older releases are available and the [Git source repository](#) is on GitHub.



GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

[View Logos →](#)

<https://git-scm.com/downloads>

Passo #1: criar um repositório (repo)

1. Criar o repositório

```
$ git init MyFirstRepo
```

2. Navegue no diretório `git init MyFirstRepo`

3. Execute `ls -la` para checar o conteúdo do diretório

Criando arquivos no diretório

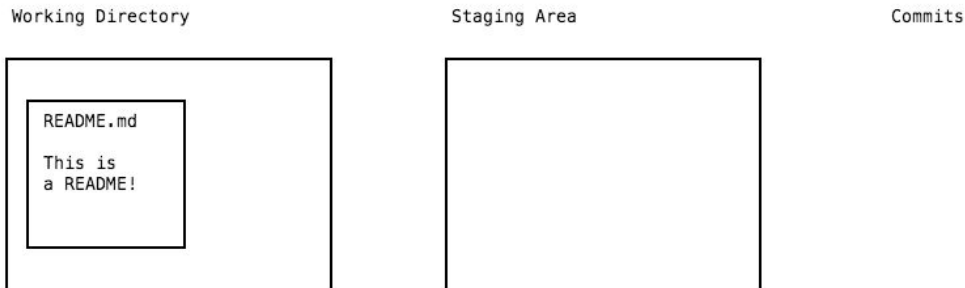
1. Create a file named `README.md` with the following content:

```
My first git project
```

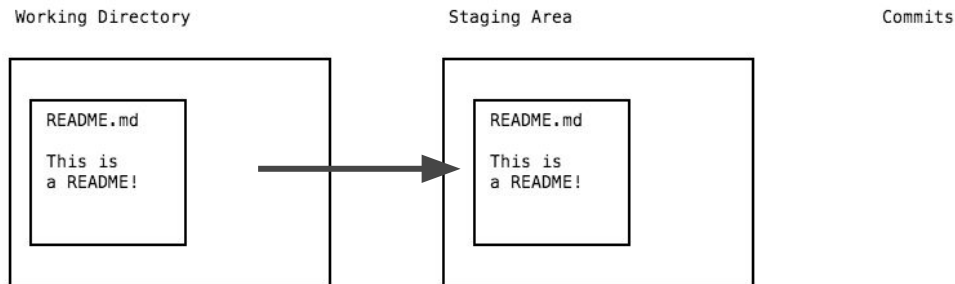
2. Create a file named `script.py` with this content:

```
if __name__ == "__main__":  
    print("10")
```

Verificando o estado do arquivo



Verifique o estado do arquivo: `git status`



`git add`

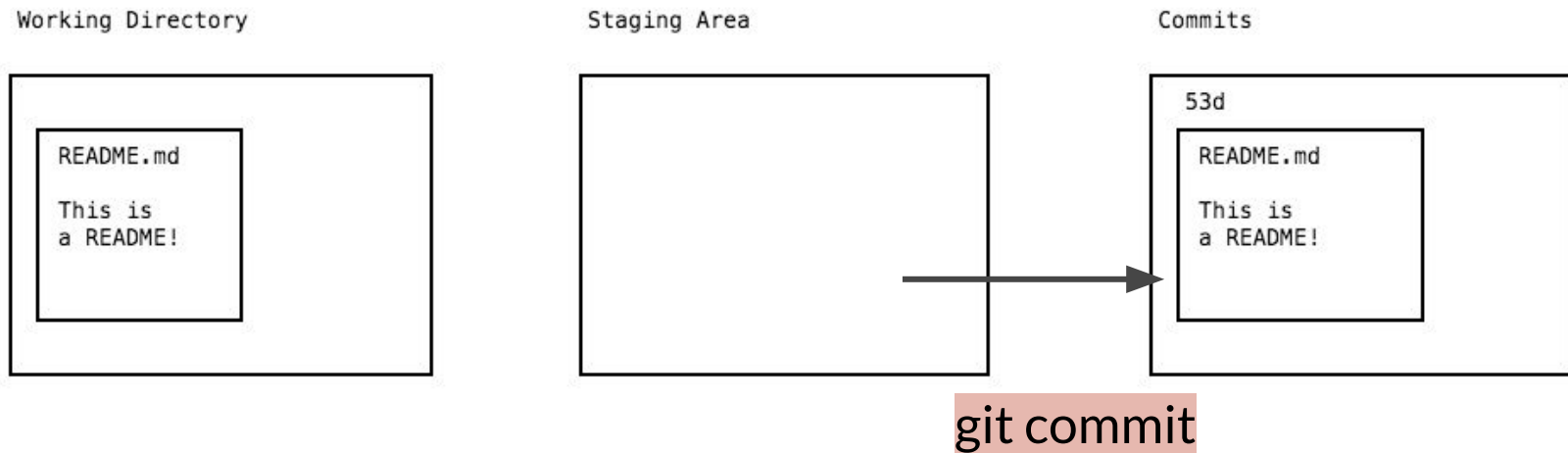
1. Check the status of the repo.
2. Add `script.py` to the staging area.
3. Add `README.md` to the staging area.

Configurando o autor das modificações

```
git config --global user.email "your.email@domain.com"
```

```
git config --global user.name "Your name"
```


Confirmando as modificações (commit)



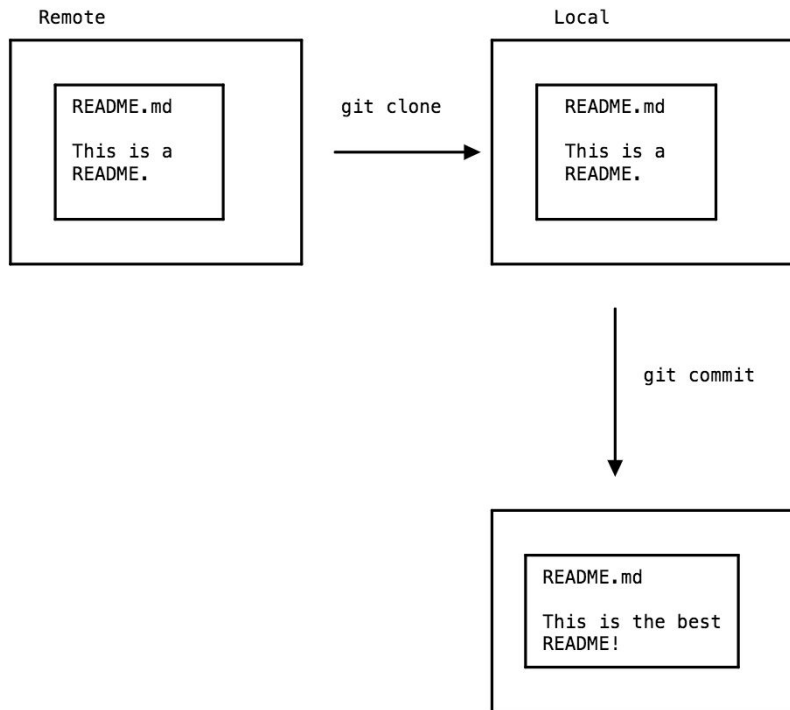
Type `git commit -m "Initial commit. Added script.py and README.md"` to make the first commit to the repository with an informative message.

Analizando o histórico de mudanças

Description:

1. Run `git log` to explore the commit history of the repository.

Repositórios remotos



- Compartilhar nosso código com outros.
- Construir um portfólio.
- Colaborar em projetos com múltiplos desenvolvedores.
- Permitir a cópia (download) e o uso de códigos feitos por terceiros

Repositórios remotos (exemplo)

Podemos facilmente copiar a solução de IA feito pela Amazon([Amazon Deep Learning](#)) a partir do GitHub:

- `git clone https://github.com/amznlabs/amazon-dsstne.git`

Repositórios remotos



1. Clonar o projeto "fast style transfer" a partir do Github para um repositório local
2. <https://github.com/lengstrom/fast-style-transfer>
3. Verifique o histórico de alterações com `git log`
4. Clonar o repositório do curso

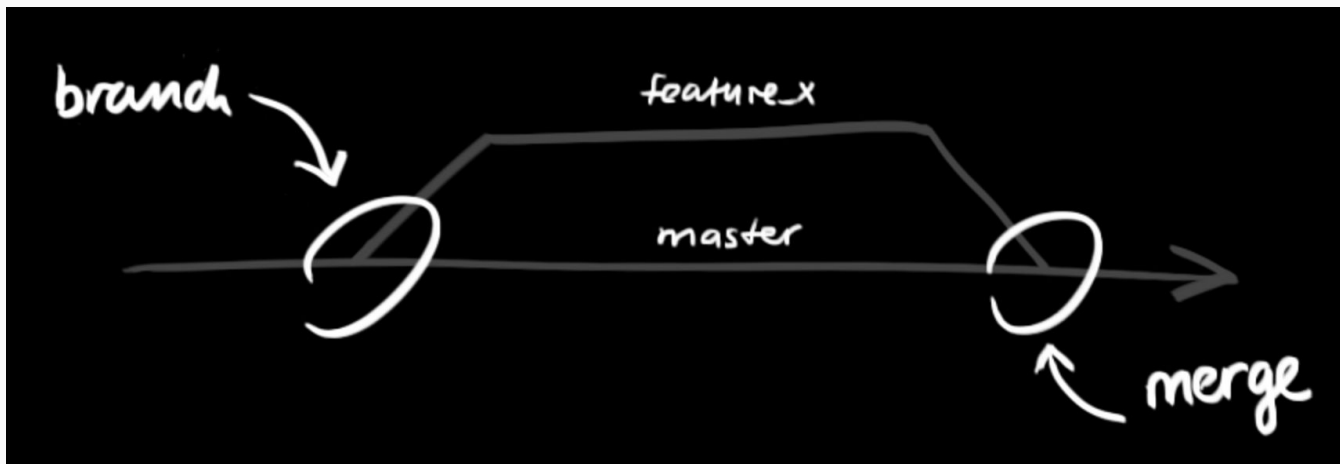
... vamos voltar para o nosso
repositório "MyFirstRepo"!!!

Integração com o Github

Criar uma conta no Github

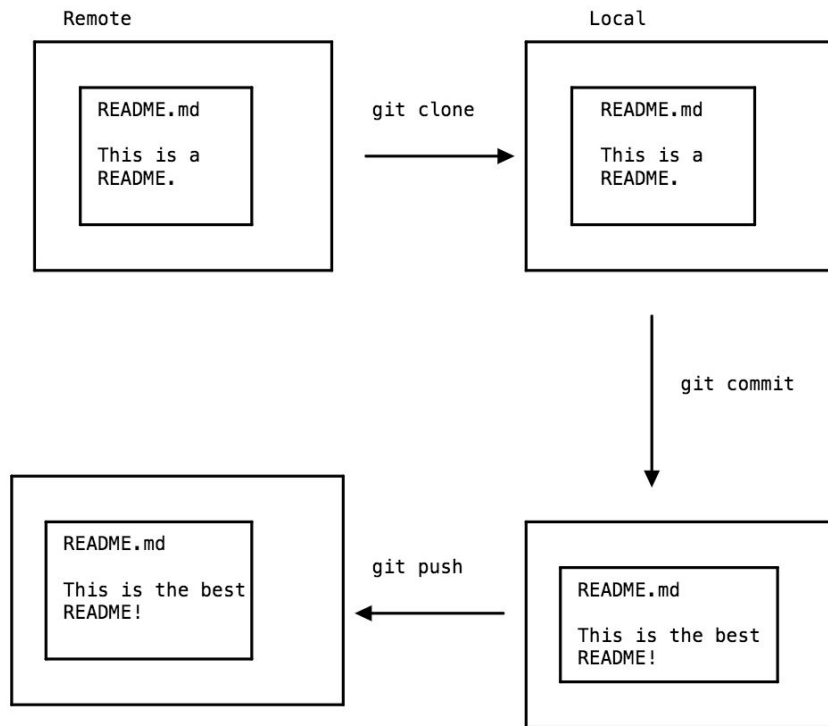
- Crie uma conta no Github. Selecione um nome de usuário, senha e um e-mail válido.
- Escolha um plano. Se você selecionou o plano gratuito (free plan), todos os seus repositórios serão públicos.
- Leia o guia de boas práticas em [GitHub [Hello World guide](#)].

Ramificando um repositório



- Cada repositório no Git consiste de um ou mais ramos (branches).
- Um ramo principal de um repositório é chamado de master.
- Utilize o comando `git branch` para visualizar o ramo atual do repositório

Atualizando o repositório remoto (pushing)



Pushing [exercício]

- Use o comando `git remote` para visualizar informações sobre o repositório remoto.
- Crie um repositório no Github

```
git remote add origin https://github.com/<your_github_user>/hello-world.git  
git push -u origin master
```

Estude os notebooks abaixo



Git and a Version Control - Introduction to Git.ipynb
Git and a Version Control - Git Remotes.ipynb
Git and a Version Control - Git Branches.ipynb



```
index.js
import React, { useState } from 'react';
import './index.css';
import './index.html';
import './index.js';

const App = () => {
  const [contacts, setContacts] = useState([]);
  const [name, setName] = useState('');
  const [phone, setPhone] = useState('');
  const [email, setEmail] = useState('');

  const handleSubmit = (e) => {
    e.preventDefault();
    setContacts([...contacts, { name, phone, email }]);
    setName('');
    setPhone('');
    setEmail('');
  };

  return (
    <div>
      <h1>React Form</h1>
      <form>
        <input type="text" value={name} onChange={e => setName(e.target.value)} />
        <input type="text" value={phone} onChange={e => setPhone(e.target.value)} />
        <input type="text" value={email} onChange={e => setEmail(e.target.value)} />
        <button type="button" value="Submit" />
      </form>
      <table>
        <caption>Contacts</caption>
        <thead>
          <tr>
            <th>Name</th>
            <th>Phone</th>
            <th>Email</th>
          </tr>
        </thead>
        <tbody>
          {contacts.map((contact) => (
            <tr>
              <td>{contact.name}</td>
              <td>{contact.phone}</td>
              <td>{contact.email}</td>
            </tr>
          ))}
        </tbody>
      </table>
    </div>
  );
};

export default App;
```

```
index.html
<!DOCTYPE html>
<html>
  <head>
    <title>React Form</title>
    <script src="index.js"></script>
  </head>
  <body>
    <div>
      <h1>React Form</h1>
      <form>
        <input type="text" value="" />
        <input type="text" value="" />
        <input type="text" value="" />
        <button type="button" value="Submit" />
      </form>
      <table>
        <caption>Contacts</caption>
        <thead>
          <tr>
            <th>Name</th>
            <th>Phone</th>
            <th>Email</th>
          </tr>
        </thead>
        <tbody>
          <tr>
            <td></td>
            <td></td>
            <td></td>
          </tr>
        </tbody>
      </table>
    </div>
  </body>
</html>
```


Aquecimento

- Versões do Python
- Tipos básicos
- Lista
- Arquivos e estruturas de repetição
- Estruturas condicionais
- Dicionários
- Funções e módulos

Notebook: "Warming_up.ipynb"

References

- <http://rogerdudler.github.io/git-guide/>
- <http://product.hubspot.com/blog/git-and-github-tutorial-for-beginners>
- <http://www.dataquest.io/>
- <http://www.datacamp.com/>
- <https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook#gs.2H4cgDM>
- <http://nbviewer.jupyter.org/github/jakevdp/PythonDataScienceHandbook/blob/master/notebooks/Index.ipynb>
- <https://www.youtube.com/watch?v=WVLhm1AMeYE&t=114s>