

# Instalação e Configuração do Ambiente de Desenvolvimento

*Autor do documento: Allan M. de Souza (monitor)*

*Professor do curso: Thiago Silva*

**Resumo:** Neste tutorial apresentaremos o ambiente de desenvolvimento que será utilizado durante o curso. Utilizaremos Python como a linguagem de programação, além de um conjunto de bibliotecas auxiliares para nos ajudar no desenvolvimento das aplicações. Ao final desse tutorial, você será capaz de criar e configurar seu próprio ambiente de desenvolvimento e terá conhecimento sobre as principais ferramentas que serão utilizadas, sendo elas, Python, Miniconda e Jupyter.

## Ferramentas

No decorrer do curso, muitas bibliotecas serão utilizadas para auxiliar no desenvolvimento das aplicações. Entretanto, com apenas uma única ferramenta, será possível instalar, configurar e gerenciar todas as bibliotecas adicionais necessárias. A ferramenta que permite isso é a Miniconda, a qual é um instalador minimalista que inclui apenas **conda**, Python, os pacotes dos quais eles dependem e um pequeno número de outros pacotes úteis, incluindo pip, zlib e alguns outros.

A Figura 1 apresenta uma visão geral do ambiente, onde serão instalados a Miniconda e o Python no seu computador. Em seguida, para adicionar as bibliotecas extras para auxiliar no desenvolvimento das aplicações, utilizaremos o gerenciador de pacotes **conda**, o qual realiza a instalação das bibliotecas desejadas de maneira simples e rápida. Exemplos dessa bibliotecas incluem: *jupyter*, *pandas*, *matplotlib*, *numpy*, *scikit-learn*, e etc.

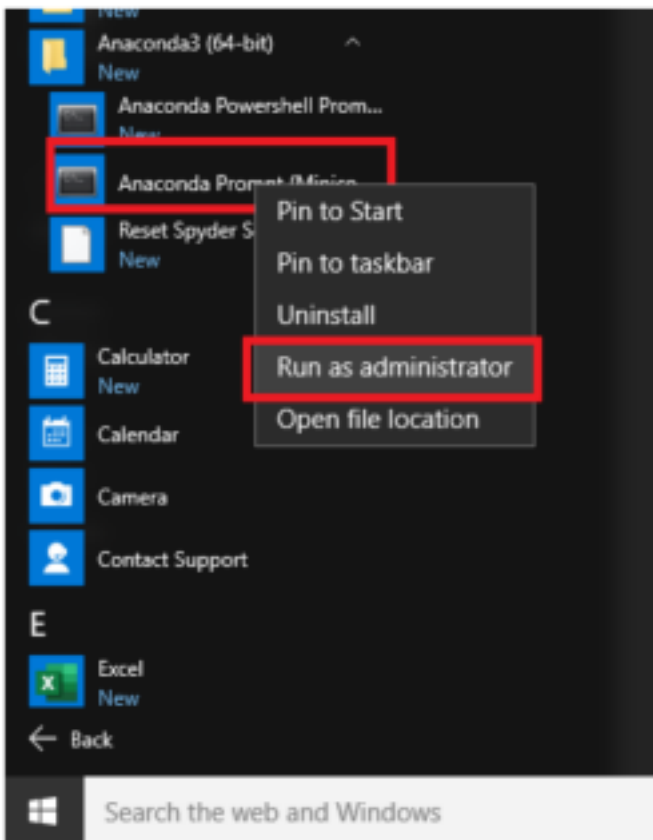


## Miniconda

Para fazer o download do mini conta, acesse o site [link] e selecione a versão correspondente a sua plataforma i.e., sistema operacional e arquitetura. A Figura 2 apresenta as últimas versões disponíveis.

Latest - Conda 4.10.3 Python 3.9.5 released July 21, 2021

Platform	Name	SHA256 hash
Windows	Miniconda3 Windows 64-bit	b33797064593ab2229a0335dc69003bba05cb56a20c2f2d3b1231213642e208a
	Miniconda3 Windows 32-bit	24f438e57ff2ef1ce3e93890d4e9d33f5090955f759f448d84a4018d3cd12d8b
MacOSX	Miniconda3 MacOSX 64-bit bash	706de9721f43e2c7d2803144c635f5f6a4823483536dc141cc82d8b927cd508
	Miniconda3 MacOSX 64-bit pkg	8fa371ae97218c3c085cd5f04b1f40156d1506a5b61d5c078f89d563fd416816
Linux	Miniconda3 Linux 64-bit	1ea2f885b4dbc3090862845560bc64271eb17085387a70c2ba3f29fff6f8d52f
	Miniconda3 Linux-aarch64 64-bit	4879820a10718743f945d88ef142c3a4b30dfc8e448d1ca00e019506374b773f
	Miniconda3 Linux-ppc64le 64-bit	fa92ee4773611f58ed9333f977432b6b64769292f605d518732183be1f3321fa
	Miniconda3 Linux-s390x 64-bit	1faed9abecf4a4dd44e0d8091fc2cdaa3394c53e077af1dad6b9d4aadbd90d8



Após fazer o download, execute o arquivo de instalação como administrador, para instalar o gerenciador no local que desejar. Utilize esse vídeo [link] como referência para realizar a instalação. Com o Miniconda instalado, é necessário atualizar o gerenciador de pacotes conda. Para isso, é necessário abrir o prompt de comando como administrador, chamado

**Anaconda Prompt (Miniconda3)**. O atalho deve estar disponível na pasta Anaconda3 no menu do botão “Iniciar”.

Para executá-lo como administrador, é necessário clicar com o botão direito em cima do atalho e então clicar em executar como administrador, mostrado na figura abaixo:

Para atualizar o gerenciador utilize o seguinte comando: **conda update conda**

```
Anaconda Prompt (Miniconda3)

(base) C:\Users\L>conda update conda
```

apertar o botão “Enter” e aceitar as sugestões de instalação sugeridas, apertando Enter.

```
idna                pkgs/main/win-64::idna-2.8-py37_0 --> pk
openssl             1.1.1d-he774522_4 --> 1.
pyparser            pkgs/main/win-64::pyparser-2.19-py37~ --> pk
requests            2.22.0-py37_1 --> 2.
setuptools           45.2.0-py37_0 --> 46
tqdm                 4.42.1-py_0 --> 4.

Proceed ([y]/n)? _
```

## Jupyter Notebook

O Jupyter Notebook é um aplicativo da web de código aberto que você pode usar para criar e compartilhar documentos que contêm código ativo, equações, visualizações e texto. Para instalar o Jupyter, utilizamos o **conda** com o seguinte comando:

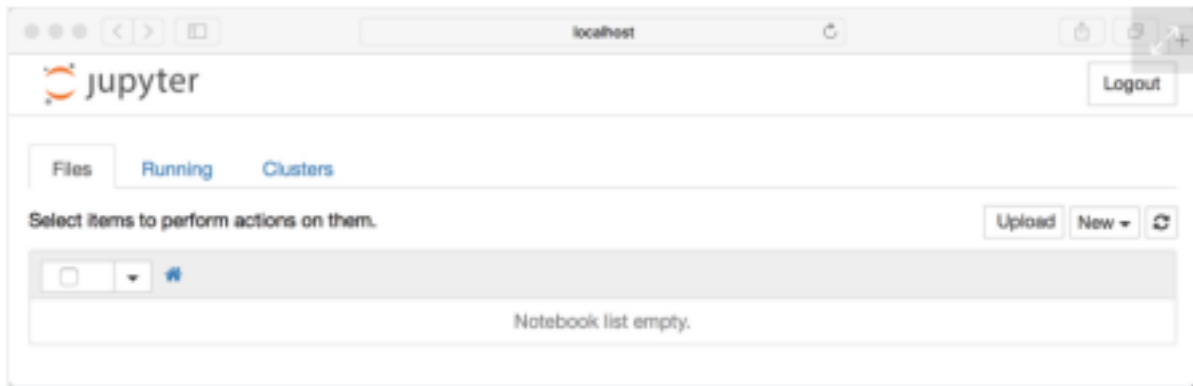
```
> conda install jupyter
```

Agora que você instalou o Jupyter, vamos aprender como usá-lo. Para começar, tudo que você precisa fazer é abrir seu terminal e ir para uma pasta de sua escolha. Eu recomendo usar uma pasta de documentos para o curso i.e., INF-1010. Em seguida, vá até esse local em seu terminal e execute o seguinte comando:

```
Shell

$ jupyter notebook
```

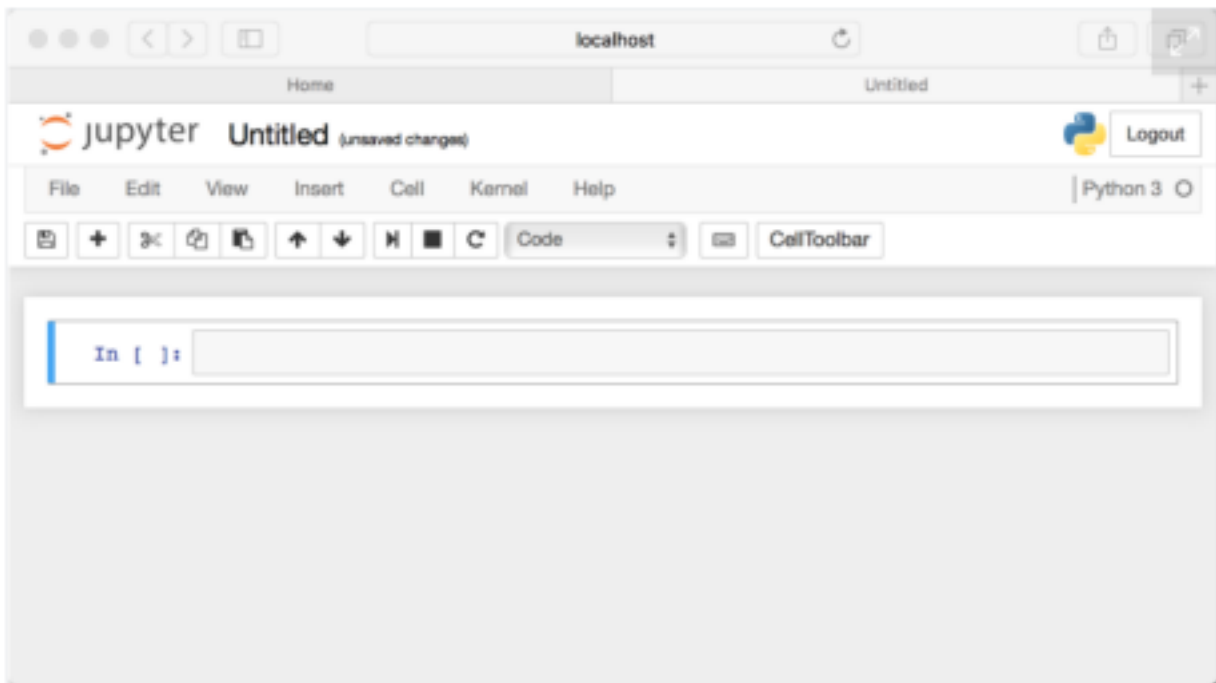
Isso iniciará o Jupyter e seu navegador padrão deve iniciar (ou abrir uma nova guia) no seguinte URL: <http://localhost:8888/tree>. Seu navegador agora deve ser parecido com isto:



Observe que agora você não está realmente executando um Notebook, mas apenas executando o servidor Notebook. Vamos criar um Notebook agora!

## Criando um Notebook

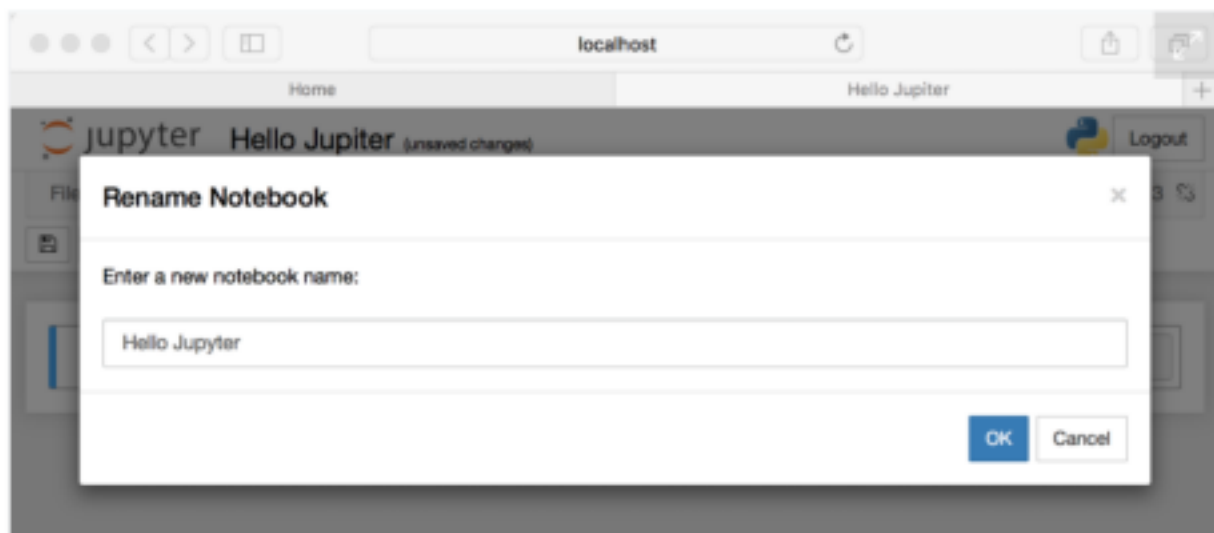
Agora que você sabe como iniciar um servidor Notebook, provavelmente deve aprender como criar um documento real do Notebook. Tudo que você precisa fazer é clicar no botão Novo (canto superior direito), e uma lista de opções será aberta, vamos escolher Python 3. Assim, sua página da web agora deve ter a seguinte aparência:



## Nomeando o Notebook

Você notará que no topo da página está a palavra Sem título. Este é o título da página e o nome do seu Notebook. Já que esse não é um nome muito descritivo, vamos mudá-lo! Basta

passar o mouse sobre a palavra Sem título e clicar no texto. Agora você deve ver uma caixa de diálogo no navegador intitulada Renomear Bloco de anotações. Vamos renomear este para Hello Jupyter:



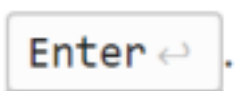
## Executando as Células

O padrão da célula de um Notebook é usar código sempre que você cria um pela primeira vez, e essa célula usa o kernel que você escolheu quando iniciou seu Notebook. Nesse caso, você iniciou o seu notebook com Python 3 como seu kernel, o que significa que você pode escrever código Python em suas células de código. Uma vez que seu Notebook inicial tem apenas uma célula vazia, o Notebook realmente não pode fazer nada.

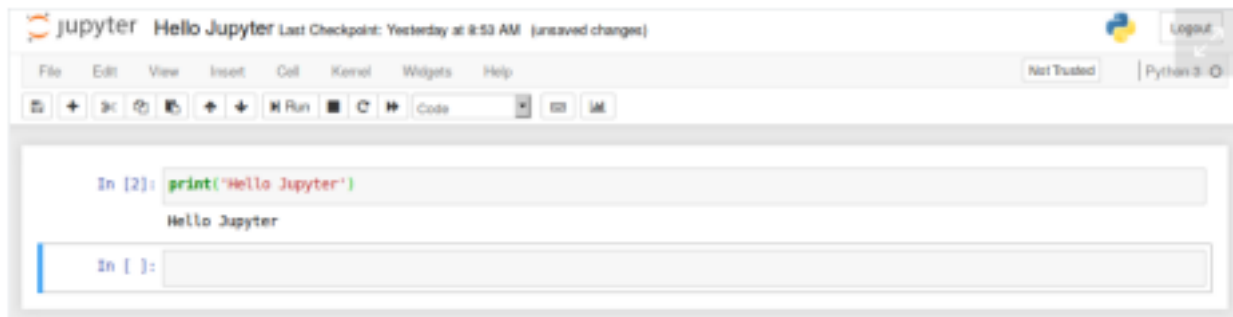
Portanto, para verificar se tudo está funcionando como deveria, você pode adicionar algum código Python à célula e tentar executar seu conteúdo. Vamos tentar adicionar o seguinte código a essa célula:

```
Python  
print('Hello Jupyter!')
```

Executar uma célula significa que você executará o conteúdo da célula. Para executar uma célula, você pode apenas selecionar a célula e clicar no botão Executar que está na fileira de botões na parte superior. Se preferir usar o teclado, basta pressionar

 Shift +  .

Quando executar o código acima, a saída deve ficar assim:



Se você tiver várias células em seu Bloco de anotações e executar as células em ordem, poderá compartilhar suas variáveis e importações entre células. Isso torna mais fácil separar seu código em blocos lógicos sem a necessidade de re-importar bibliotecas ou re-criar variáveis ou funções em cada célula.

Ao executar uma célula, você notará que há alguns colchetes próximos à palavra **In [ ]**: à esquerda da célula. Os colchetes serão preenchidos automaticamente com um número que indica a ordem em que você executou as células. Por exemplo, se você abrir um novo Notebook e executar a primeira célula na parte superior do Notebook, os colchetes serão preenchidos com o número 1.

## Ver o que está em execução

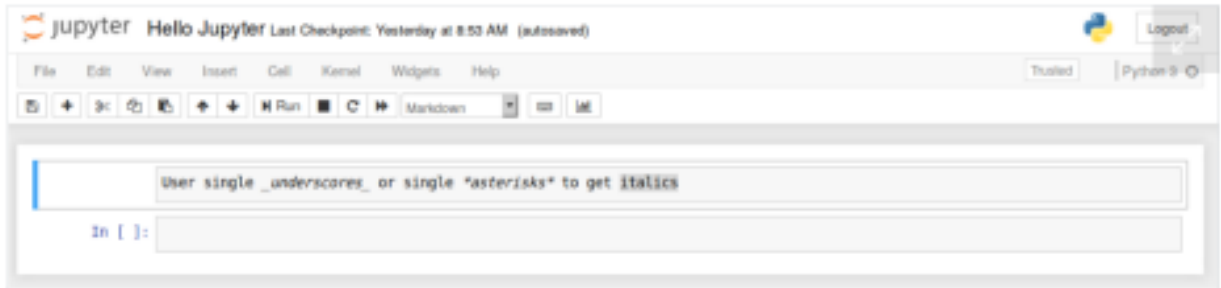
Na página inicial do seu servidor Jupyter (<http://localhost:8888/tree>) estão duas outras guias: **Running** e **Clusters**. A guia **Running** informará quais Notebooks e Terminais você está executando no momento. Isso é útil quando você deseja desligar o servidor, mas precisa ter certeza de que salvou todos os seus dados. Felizmente, os notebooks salvam automaticamente com bastante frequência, então você raramente perde dados. Mas é bom poder ver o que está funcionando quando você precisar. Outra coisa boa sobre esta guia é que você pode passar por seus aplicativos em execução e fechá-los lá.

## Tipos de células

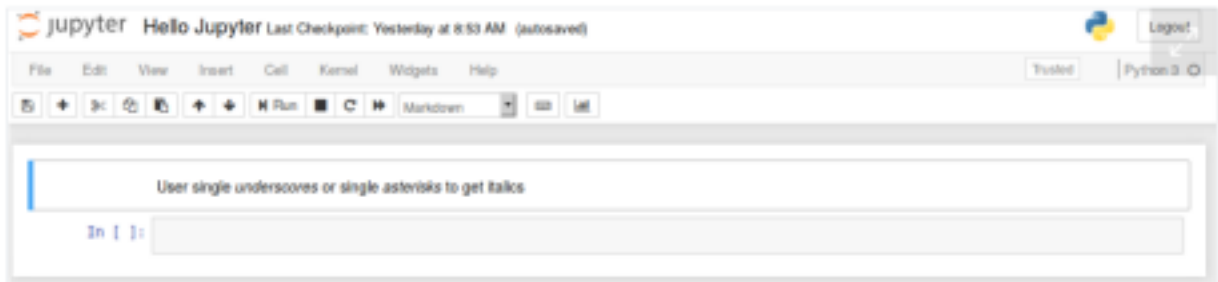
Os principais tipos de células que você usará são os tipos de células **Code** e **Markdown**. Você já aprendeu como as células de código funcionam, então vamos aprender como estilizar seu texto com **Markdown**.

O Jupyter Notebook suporta Markdown, que é uma linguagem de marcação que é um superconjunto do HTML. Este tutorial cobrirá alguns dos princípios básicos do que você pode fazer com o Markdown.

Defina uma nova célula para Markdown e, em seguida, adicione o seguinte texto à célula:

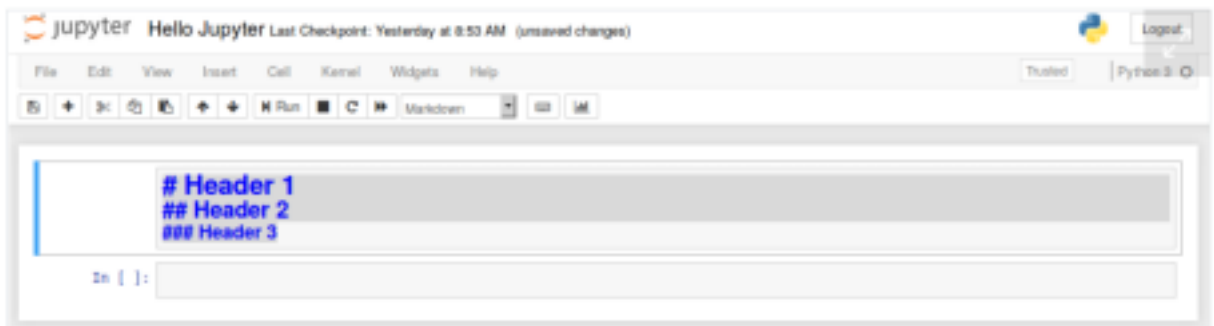


Quando você executar a célula, a saída deve ser semelhante a esta:

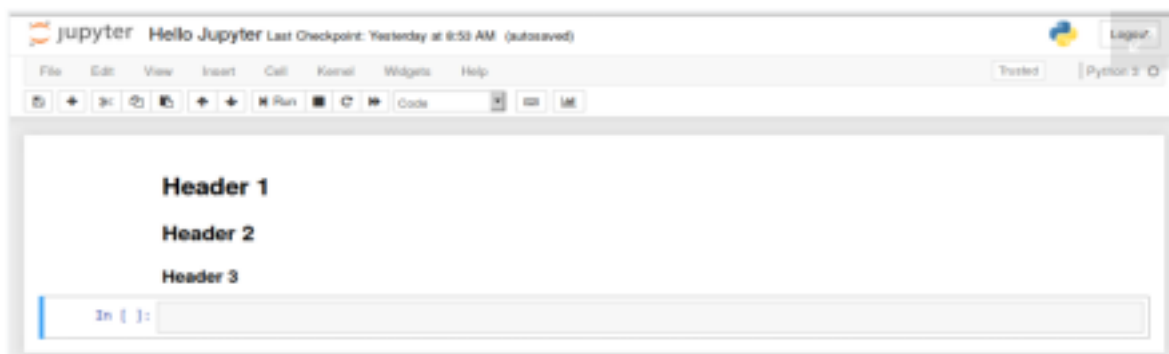


## Cabeçalhos

Criar cabeçalhos no Markdown também é bastante simples. Você apenas tem que usar **#** (i.e., **cerquilha**, **hashtag**). Quanto mais sinais de cerquilha você usar, menor será o cabeçalho. O Jupyter Notebook faz uma espécie de pré-visualização para você:



Então, ao executar a célula, você terá um cabeçalho bem formatado:



## Exemplos de códigos Markdown e Comandos

Para mais informações sobre de Markdown para editar o texto de seu notebook utilize o link (<https://docs.pipz.com/central-de-ajuda/learning-center/guia-basico-de-markdown#open>).

Além disso, o Jupyter Notebook possui um série de comandos que podem ser utilizados para navegar, editar, executar e gerenciar células, veja alguns desses comandos no seguinte link: <https://karloguidoni.com/til/cheatsheets/jupyter-notebook/>

## Google Colab

Essa é uma plataforma online do Google. Para acompanhar o curso você pode ter o ambiente na sua máquina, como informado acima, ou usar a plataforma fornecida pelo Google. Recomendamos ter o ambiente na sua máquina instalado, mesmo que pretenda utilizar o Google Colab.