

Python Fundamentos



Sobre o Instrutor

Alexandre dos Reis Proença



Graduado Técnico em Eletrônica pela Universidade Tecnológica Federal do Paraná 2003.

Graduado Analista de sistemas pela Universidade Positivo 2013.

Experiência no mercado de TI (+10 anos).

Experiência em Python (core/web 5+ anos).

Experiência em infraestrutura de redes (+5 anos)

Desenvolvimento em RESTful Web services (Django, Bottle, Tornado + 3 anos).

Experiência com métodos ágeis (Scrum, Kanban +2 anos).

Experiência com configuração de Webserver (Nginx e Apache +2 anos).

Experiência com bancos de dados NoSQL (MongoDB +2 anos).

Conhecimento estratégias de "Caching" (pub/sub Redis, memcached +2 anos).



Conhecendo a turma

- Nome
- Idade
- Estuda ?
- Período ?
- Trabalho ?
- Programa ?
- Qual é o seu nível de envolvimento com Python ?
- Qual a sua expectativa com o curso ?



Sobre o Curso

Tópicos

Sobre o Python

Variáveis

Coleções de Dados

Algoritmos

Entrada e Saída
de Dados

Formatação de
Strings

Algebra Booleana

Estruturas de
Decisão

Funções

Tipos de Dados
Primitivos

Estruturas de
Repetição

Classes



O que Vamos Precisar?

- Uma conta no slack.com
- Uma conta no github.com
- Uma [IDE](https://www.jetbrains.com) www.jetbrains.com



Um pouco mais sobre o curso?

- O que eu deveria saber?

Conhecimentos básicos sobre matemática. conhecimento inicial sobre programação não se faz necessário.

- O que eu vou aprender?

Implementar e utilizar algoritmos básicos, utilizando a linguagem Python. Identificar os algoritmos necessários para a resolução de problemas específicos, usando a linguagem Python

- Qual versão do Python vamos usar?

Neste curso usamos Python3.x

- Como deve me portar?

Mantenha-se focado, celular desligado ou vibracall, evite conversas Durante a explicação.



Por que Programar?

- É divertido
Fazer programas = resolver quebra-cabeças
Programação como arte
- É flexível
Programas feitos por terceiros nem sempre resolvem seu problema
- É útil
Programação como ferramenta
Pode ser aplicado a quase qualquer atividade
(Artes Gráficas, Pesquisas, Cálculos, Simulações, Automação)

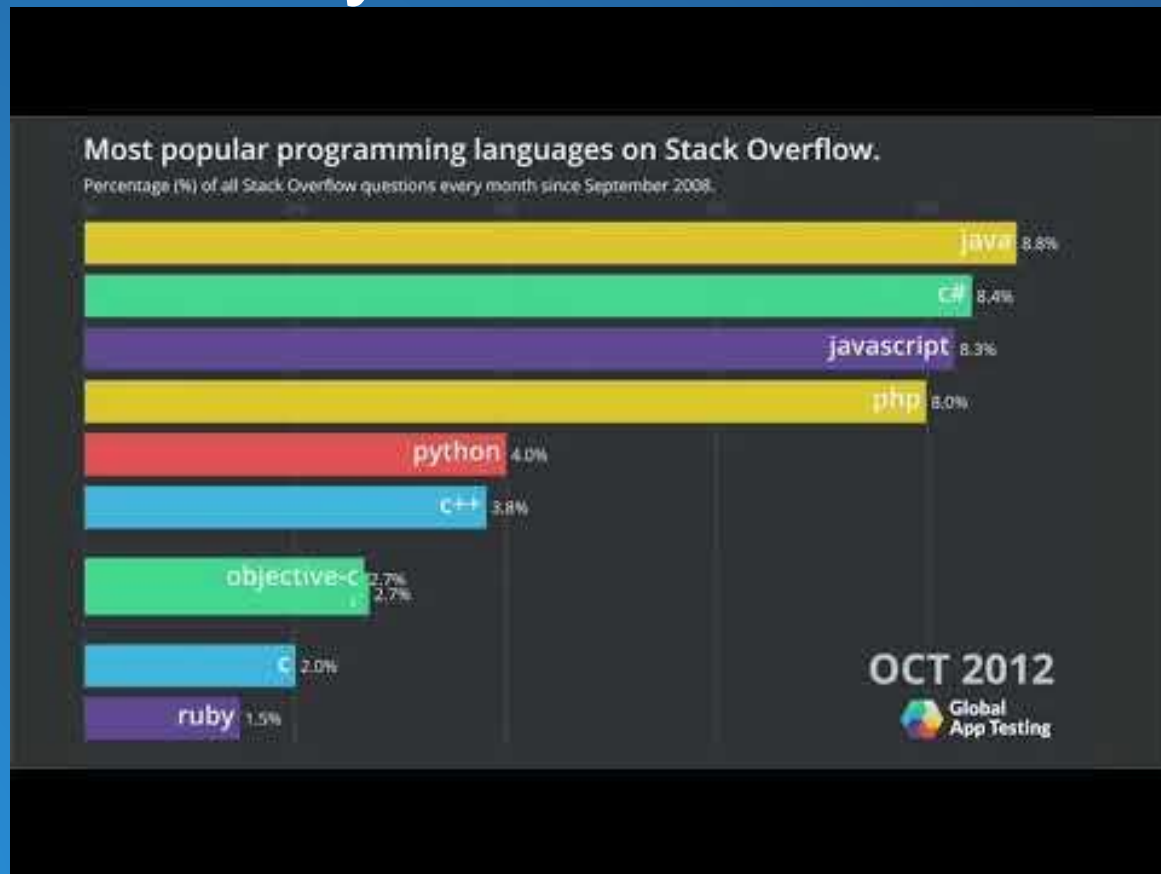


Por que Python?

- Simples
- Fácil de aprender
- Muitos recursos
- Multi-plataforma
- Comunidade grande
- Grátis!
- Grande procura de profissionais



Como o Python está atualmente?

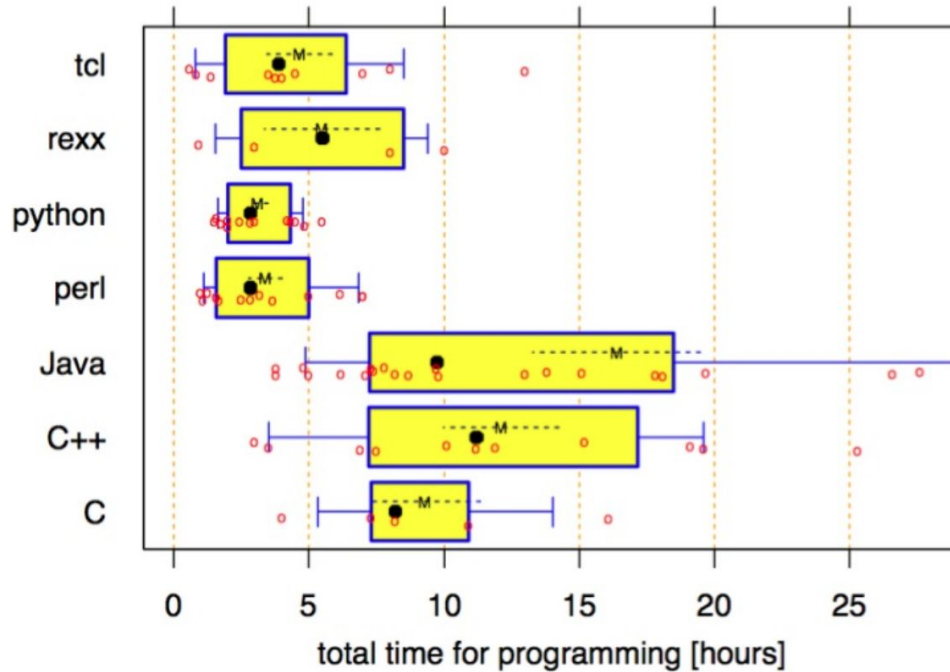


QUEM USA PYTHON ?



PRODUTIVIDADE PYTHON!

- Tempo de desenvolvimento



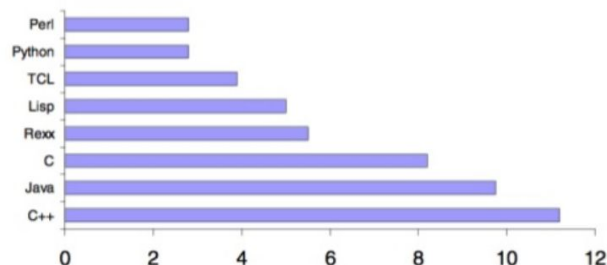
Prechelt, L. An empirical comparison of C, C++, Java, Perl, Python, Rexx, and Tcl. IEEE Computer

PRODUTIVIDADE PYTHON!

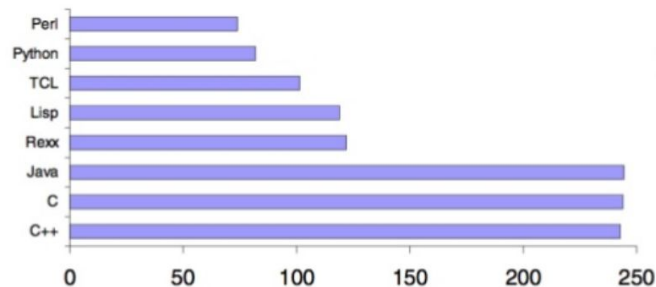
- Tempo para ser produtivo

Recortar slide

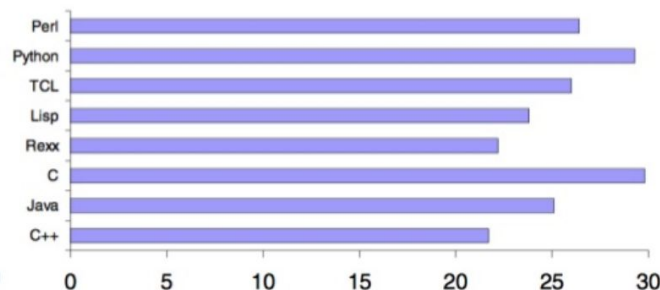
Média de Horas para Resolver o Problema



Média de Linhas de Código



Linhas de Código por hora



Características da Linguagem

- Alto nível
- Interpretada
- Script
- Tipagem dinâmica e forte
- Multiparadigma
- Fácil leitura do código e
- Exige poucas linhas de código se comparado ao mesmo programa em outras linguagens.
- Processamento de textos
- Conexão com Bancos de Dados
- Processamento concorrente e paralelo



Python é interpretado ou compilado ?

Interpretada, com alguma compilação. CPython compila** código fonte Python para bytecode, e depois *interpreta este bytecode, executando conforme progride.

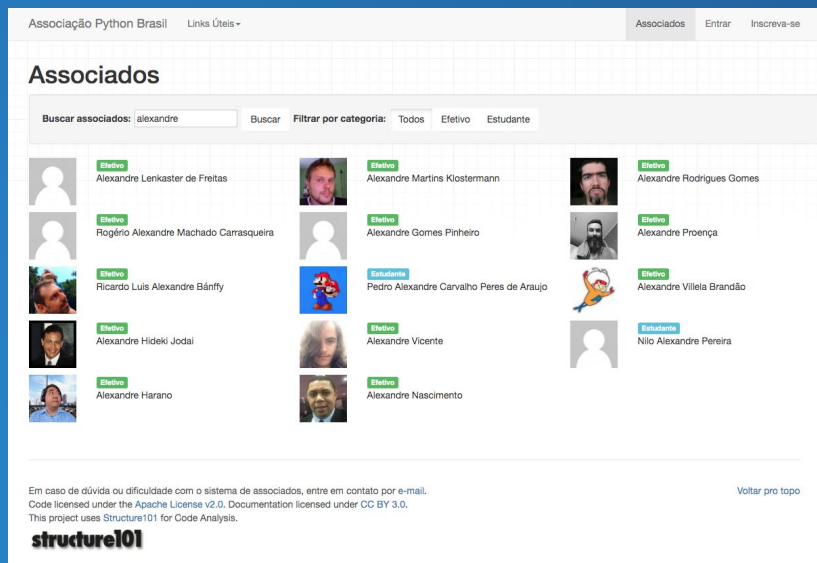
* Nota: isto não é 'compilação' no sentido tradicional da palavra. Tipicamente, diríamos que 'compilação' é pegar uma linguagem de alto nível e convertê-la para código de máquina. Mas é uma certa 'compilação'.



Comunidade Python

Python Brasil

A Associação Python Brasil (APyB) foi formada em abril de 2007 com a meta de apoiar as comunidades relacionadas à linguagem Python e suas tecnologias derivadas.



The screenshot shows the 'Associados' (Members) page of the Associação Python Brasil website. The page has a header with the site name and navigation links. Below the header, there is a search bar and a filter section. The main content area displays a grid of member profiles, each with a profile picture, a status label (e.g., 'Eletivo', 'Estudante'), and the member's name. The footer contains contact information, licensing details, and the 'structure101' logo.

Associação Python Brasil Links Úteis - Associados Entrar Inscreva-se

Associados

Buscar associados: alexandre Buscar Filtar por categoria: Todos Eletivo Estudante

Nome	Status
Alexandre Lenkaster de Freitas	Eletivo
Rogério Alexandre Machado Carrasqueira	Eletivo
Ricardo Luis Alexandre Bânffy	Eletivo
Alexandre Hideki Jodai	Eletivo
Alexandre Harano	Eletivo
Alexandre Martins Klostermann	Eletivo
Alexandre Gomes Pinheiro	Eletivo
Pedro Alexandre Carvalho Peres de Araujo	Estudante
Alexandre Vicente	Eletivo
Alexandre Nascimento	Eletivo
Alexandre Rodrigues Gomes	Eletivo
Alexandre Prouença	Eletivo
Alexandre Villela Brandão	Eletivo
Nilo Alexandre Pereira	Estudante

Em caso de dúvida ou dificuldade com o sistema de associados, entre em contato por e-mail.
Code licensed under the Apache License v2.0. Documentation licensed under CC BY 3.0.
This project uses Structure101 for Code Analysis.

structure101 Voltar pro topo

Comunidades Locais



GruPy-RN

Grupo de usuários do Rio Grande do Norte

Telegram GitHub Facebook
Google Groups



PythonOnRio

Grupo de usuários do Rio de Janeiro

Site Facebook Telegram Twitter
Yahoo Groups



GruPy-PB

Grupo de usuários da Paraíba

Google Groups



Py013

Grupo de usuários da Baixada Santista DDD 013 (Litoral de SP)

Facebook Github



GruPy Blumenau

Grupo de usuários de Blumenau SC

Site Facebook Telegram Twitter
Slack



GruPy-ABC

Grupo de usuários do Grande ABC (São Paulo)

Site Github Telegram



Python Florianópolis

Grupo de usuários de Florianópolis

Site Google groups Google Plus
Github Facebook Twitter Slack



PyTché

Grupo de usuários do Rio Grande do Sul

Google Groups Meetup Telegram



Pug-PE

Grupo de usuários de Pernambuco

Google Groups Site



GruPy-SP

Grupo de usuários de São Paulo

Google Groups Github Facebook
Twitter Slack



GruPy-PR

Grupo de usuários do Paraná

Google Groups Github Mobilize
Slack Telegram



Pug-SC

Grupo de usuários de Santa Catarina

Yahoo Groups



GruPy-ES

Grupo de usuários do Espírito Santo

Google Groups



GruPy-AI

Grupo de usuários de Alagoas

Google Groups



GruPy-RO

Grupo de usuários de Rondônia

Google Groups



GruPy-BA

Grupo de usuários da Bahia

Google Groups Meetup

Documentação oficial do Python



The screenshot shows the official Python documentation website. At the top, there is a navigation bar with the Python logo, a 'Donate' button, a search bar, and links for 'About', 'Downloads', 'Documentation', 'Community', 'Success Stories', and 'News'. The main content area features a large heading 'Browse the docs online or download a copy of your own.' followed by a paragraph stating that Python's documentation, tutorials, and guides are constantly evolving. Below this, there are two buttons for 'Python 3.x Docs' and 'Python 2.x Docs', and a link to 'Documentation Releases by Version'. To the right of the text is an illustration of a folder with gears. At the bottom, there are three columns: 'Beginner' with links to 'Beginner's Guide', 'Python 2 or 3?', and 'Python FAQs'; 'Moderate' with links to 'Python Periodicals' and 'Python Books'; and 'Advanced' with links to 'Python Packaging User Guide', 'In-development Docs', and 'Guido's Essays'.

python™

Donate Search

About Downloads Documentation Community Success Stories News

Browse the docs online or download a copy of your own.

Python's documentation, tutorials, and guides are constantly evolving. Get started here, or scroll down for documentation broken out by type and subject.

Python 3.x Docs Python 2.x Docs

See also [Documentation Releases by Version](#)

Beginner

- Beginner's Guide
- Python 2 or 3?
- Python FAQs

Moderate

- Python Periodicals
- Python Books

Advanced

- Python Packaging User Guide
- In-development Docs
- Guido's Essays

Interpretador do Python

- Um interpretador é um programa que executa outros programas
- O interpretador lê o código Python e executa as instruções que ele contém
- É uma camada de software entre seu programa e código e o Hardware do computador
- Estada disponível em plataformas Windows Mac Linux/Unix
- Implementado em várias linguagens mas a padrão é C (CPython)
- CPython compila seu código fonte Python para bytecode
- Este bytecode é então executado na Máquina Virtual CPython



Instalando o Python



The screenshot shows the Python.org website with the Python logo and the word "python" in a serif font. Below the logo is a navigation bar with links: "About", "Downloads", "Documentation", and "Community". The "Downloads" link is highlighted. Below the navigation bar, the heading "Download the latest version for Windows" is displayed in a bold, yellow font. To the left of this heading is a large red arrow pointing right. Below the heading is a yellow button with the text "Download Python 3.7.1". Below the button, there is a link for "Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other". Below that is a link for "Want to help test development versions of Python? Pre-releases". At the bottom, there is a link for "Looking for Python 2.7? See below for specific releases".

python™

About Downloads Documentation Community

Download the latest version for Windows

Download Python 3.7.1

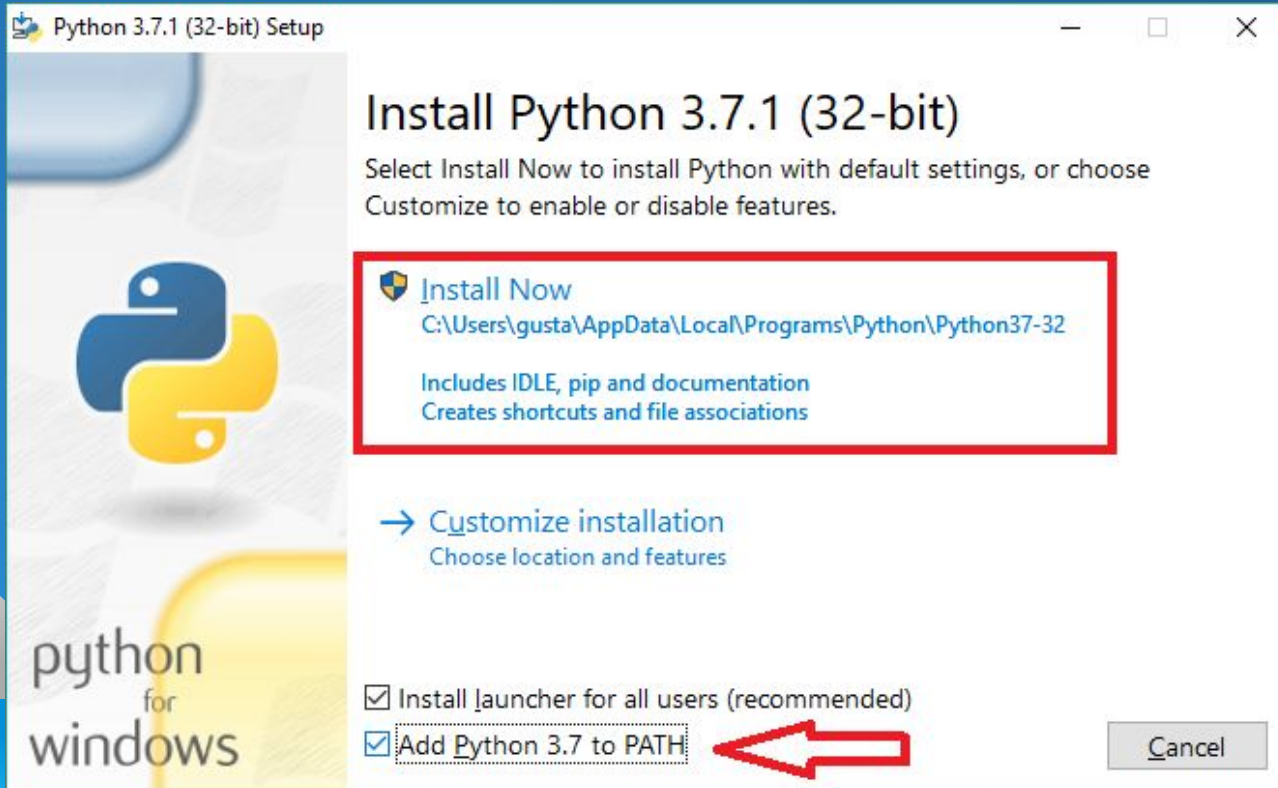
Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases

Looking for Python 2.7? See below for specific releases



Instalando o Python no Windows



Modo interativo do Python

IDLE é o ambiente de desenvolvimento Python padrão. Seu nome é um acrônimo de Integrated DeveLopment Environment (“Ambiente de Desenvolvimento Integrado”). Ele funciona bem em ambas as plataformas Linux/Unix e Windows.

Este modo lhe possibilita "conversar" diretamente com o interpretador, declarando variáveis, funções, classes, e fazendo tudo que poderia fazer em um programa qualquer.

<https://www.youtube.com/watch?v=pzmEjNS0kw8>



Python como calculadora

```
>>> # Um comentário é precedido do caracter "#" ...  
# Comentários são ignorados pelo interpretador ... 10+5 15  
>>> 10-15 # Comentários podem aparecer também após código -5  
>>> 10*3 # Multiplicação  
>>> 10**2 # Exponenciação  
>>> 10**0.5 # Raiz  
>>> 10/3 # Divisão  
>>> 10//3 # Retorna somente a parte inteira  
>>> 10%3 # Resto de divisão inteira simbolizado por %
```



Conceitos básicos sobre indentação

A indentação é uma característica peculiar na linguagem.

Enquanto que os blocos são delimitados explicitamente em C, Java e PHP por **chaves** e em Pascal e Fortran por palavras-chave como then e endif, em **Python blocos são delimitados por espaços ou tabulações** formando uma indentação visual; não existem símbolos de “abre” e “fecha”.

Ao se usar um editor de texto comum, é fácil haver erros de indentação, sendo conveniente configurar o editor para a indentação do Python. As IDEs que suportam Python têm, em geral, a função de indentação automática.



Conceitos básicos sobre indentação

```
#include <stdio.h>

int main ()
{
    int valor = 1;
    if (valor > 1)
        if (valor > 7)
            printf("valor alto\n");
    else
        printf("valor igual %d\n", valor);
    return 0;
}
```

```
valor = 1
if valor > 1:
    if valor > 7:
        print("valor alto ")
else:
    print("valor igual a ", valor)
```


Conceitos básicos sobre indentação

```
valor = 1
if valor > 1:
    if valor > 7:
        print("valor alto ")
    else:
        print("valor igual a ", valor)
```

Convenção de nomes no Python

Os padrões de codificação são um conjunto de diretrizes para produzir código atualizável e escalonável.

E como você é bom em segui-los pode decidir a longevidade de sua carreira no desenvolvimento de software.

Lembre-se de que a convenção de nomenclatura é apenas uma das facetas de muitos padrões de codificação do Python. Você pode ler sobre os outros fatores da documentação do [PEP8](#).



Convenção de nomes no Python

Como começar?

- Não use nomes que pareçam muito gerais ou prolixos.

O que é ruim?

- `user_list`, `moveInts`, `swapNums`, `dict_to_store_defns_of_a_word`

O que é bom?

- `user_info`, `move_integers`, `swap_numbers`, `word_definitions`.

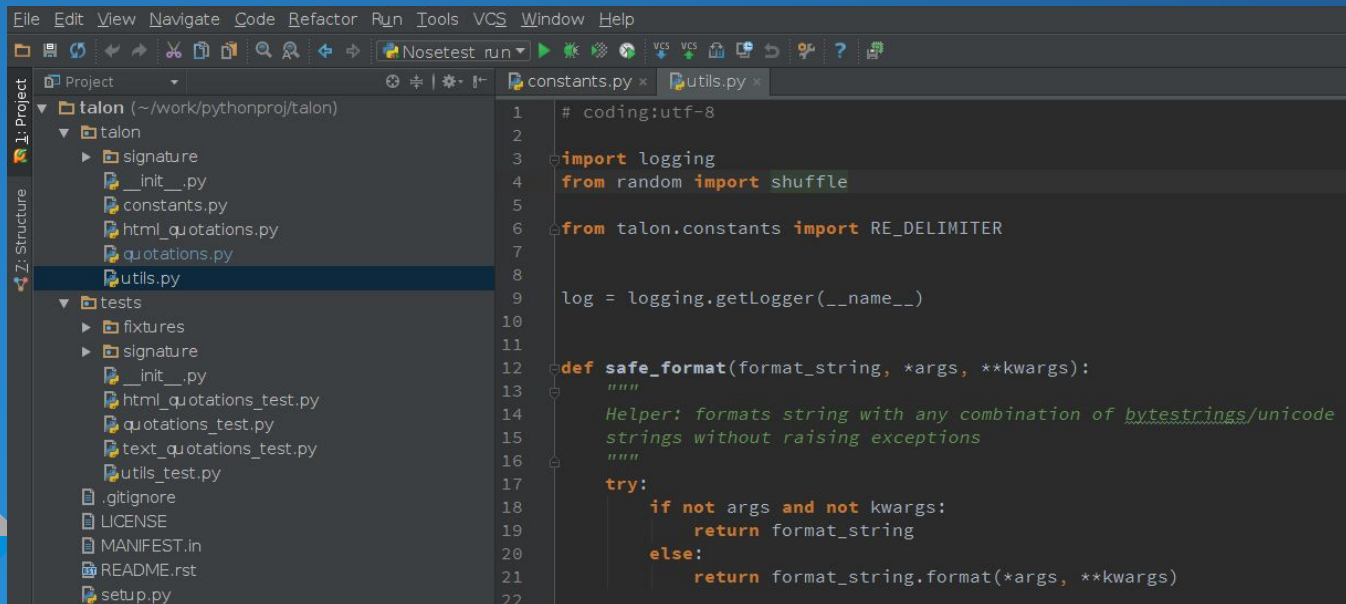
Observações:

- Use verbos para nomear funções ou métodos (pegar, checar, put, get)
- Use Substantivos para nomear variáveis, atributos ou objetos (mulher, gato, canetas, items, carro)
- Evite usar adjetivos como nomes (verde, lindo, alto, antigo)
- Não seja ingênuo e nomeie coisas como "X", "Y" ou "Z."
- Use o CamelCase apenas em nomes das Classes
- Use sempre uma nomenclatura mais semântica possível (`user_info`, `move_integers`) Ex:
 - `cor_verde = '#236647'`
 - `nivel_alto = '3v'`
 - Variável = Substantivo + substantivo ou adjetivo
 - Metodo = verbo + substantivo (+ adjetivo quando necessário)



O Primeiro Programa

Para escrever e executar nosso primeiro programa vamos precisar baixar e instalar um programa que chamamos de IDE, este curso usa o versão gratuita do Pycharm desenvolvido pela JetBrains.



The screenshot shows the PyCharm IDE interface. On the left, the 'Project' and 'Structure' toolbars are visible. The 'Project' toolbar shows the project name 'talon' and its path '~/.work/pythonproj/talon'. The 'Structure' toolbar shows the project's file structure, including a 'tests' directory with various test files. The main editor window displays the code for 'utils.py'. The code includes a logging setup and a 'safe_format' function that formats strings without raising exceptions.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Project Structure
talon (~/.work/pythonproj/talon)
├── talon
│   ├── signature
│   ├── __init__.py
│   ├── constants.py
│   ├── html_quotations.py
│   ├── quotations.py
│   └── utils.py
├── tests
│   ├── fixtures
│   ├── signature
│   ├── __init__.py
│   ├── html_quotations_test.py
│   ├── quotations_test.py
│   ├── text_quotations_test.py
│   └── utils_test.py
├── .gitignore
├── LICENSE
├── MANIFEST.in
├── README.rst
└── setup.py

1 # coding:utf-8
2
3 import logging
4 from random import shuffle
5
6 from talon.constants import RE_DELIMITER
7
8
9 log = logging.getLogger(__name__)
10
11
12 def safe_format(format_string, *args, **kwargs):
13     """
14     Helper: formats string with any combination of bytestrings/unicode
15     strings without raising exceptions
16     """
17     try:
18         if not args and not kwargs:
19             return format_string
20         else:
21             return format_string.format(*args, **kwargs)
22     except:
```