

Trabalho de python™

Professor: Ricardo Guedes

Alunos: Paulo Mateus, Gabriela Valentim, Samara Oliveira



Bibliotecas utilizadas

Turtle: Nossa velha amiga tat burra ;D

Math: Ajudar nos desenhos (math.sin...)

Random: Sortear os prédios



```
3
4 import math
5 import random
6 import turtle
7
8 import sys
9
10 from tkinter import *
11 from tkinter import messagebox
12 from tkinter import colorchooser
13
14 import tkinter.ttk as ttk
15
16 from cidadeBela.cores import cores
17 from cidadeBela.matematica import matematica
18
19 from cidadeBela.tetos import tetos
20 from cidadeBela.portas import portas
21 from cidadeBela.janelas import janelas
22
```



Bibliotecas utilizadas

Tkinter: Janelas gráficas

As vezes é preciso “criar”...

Matemática: Poupar trabalho de preguiçoso?

Cores: Formatação de cores





Bibliotecas utilizadas

... criar por organização!

Janelas: Janelas dos prédios

Portas: Portas dos prédios

Tetos: Tetos dos prédios





IDE utilizada: PyScripter



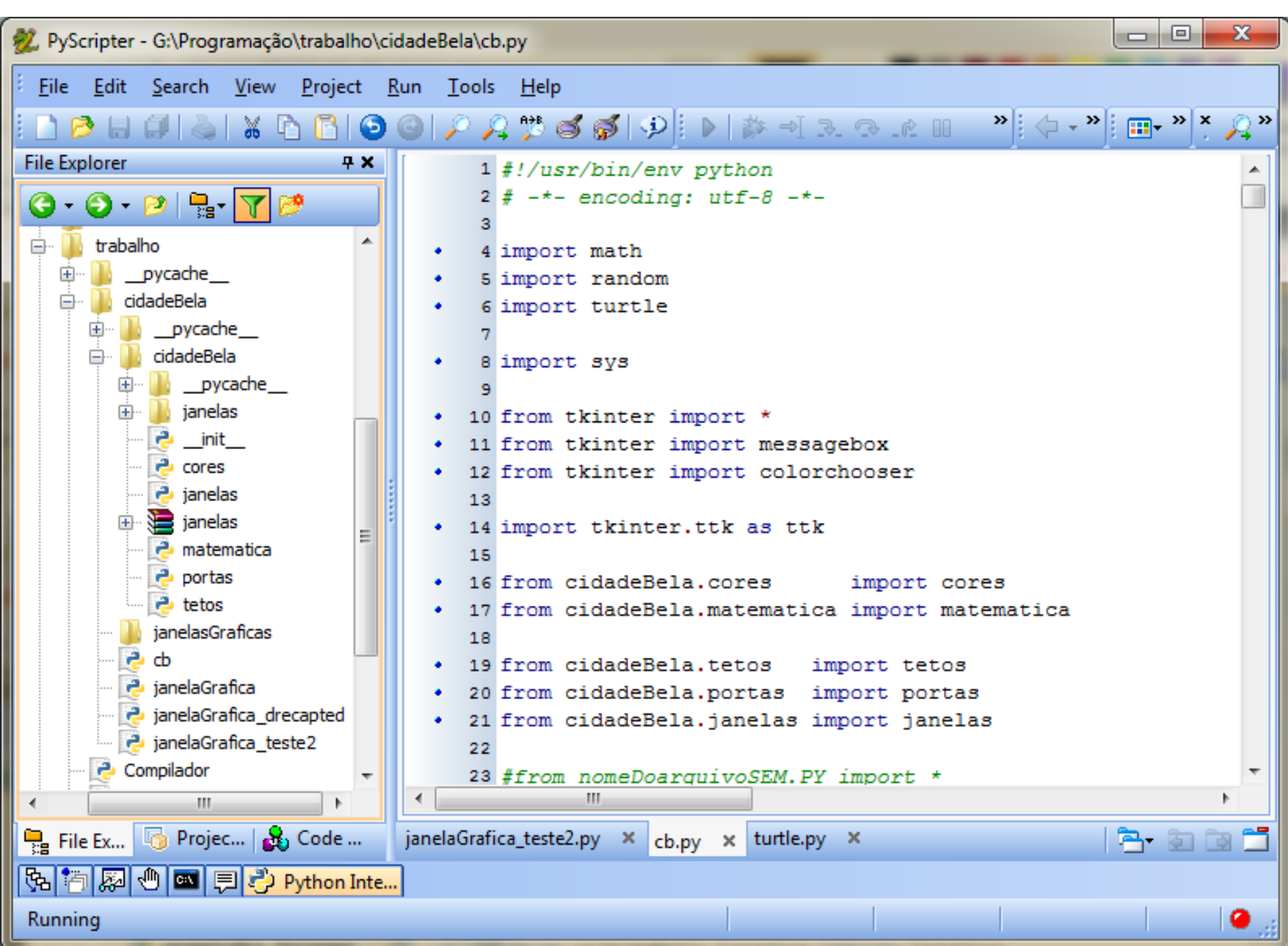
Download:

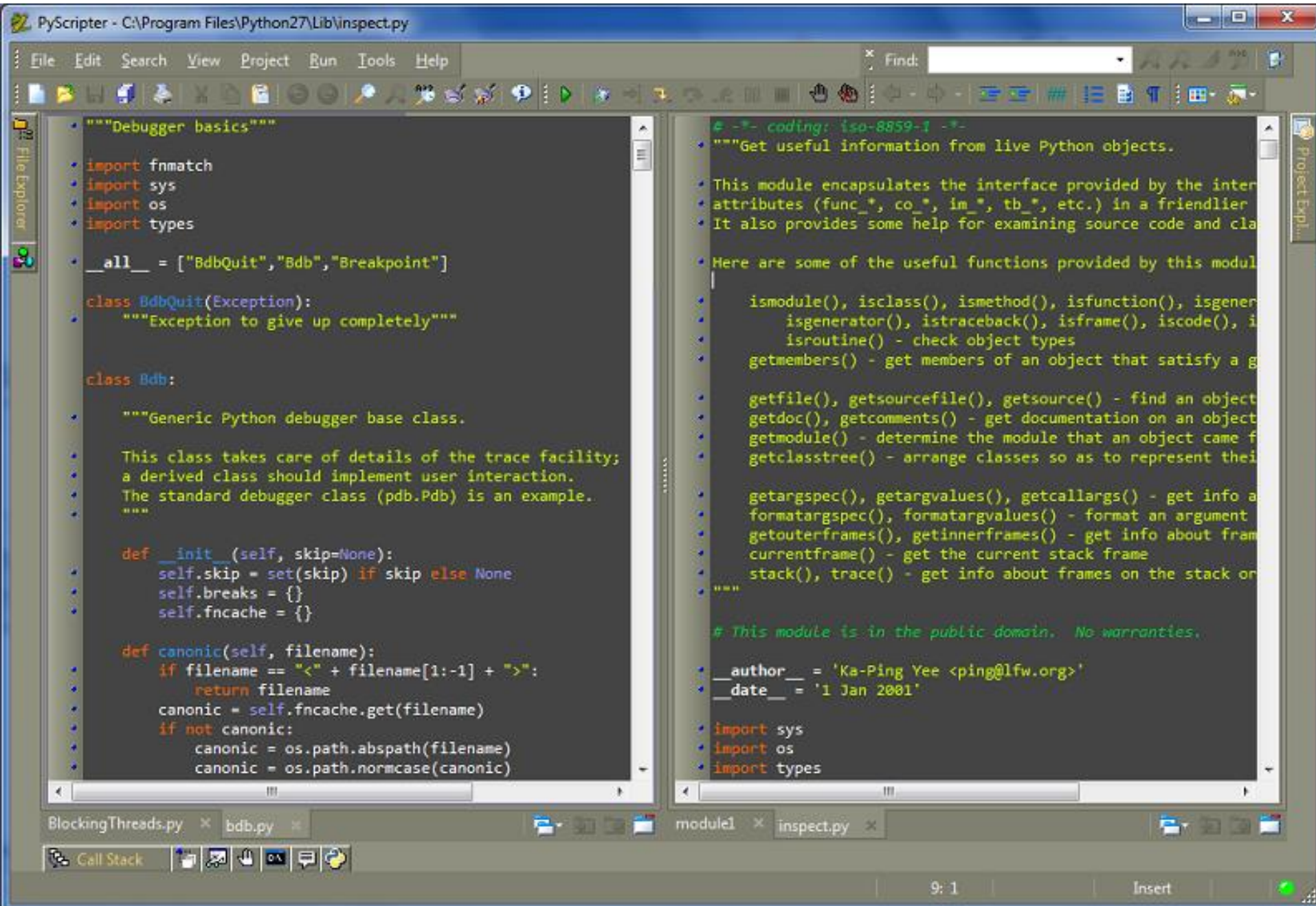
<http://code.google.com/p/pyscripter/downloads/list>

Estilos:

<http://code.google.com/p/pyscripter/wiki/Customization>









cb.py

Classe para criar cidades!

```
>>> cb = cidadeBela()
```





Métodos

Janelas

janelafixar()

janelaLimpar()

janelaFechar()

janelaErro()

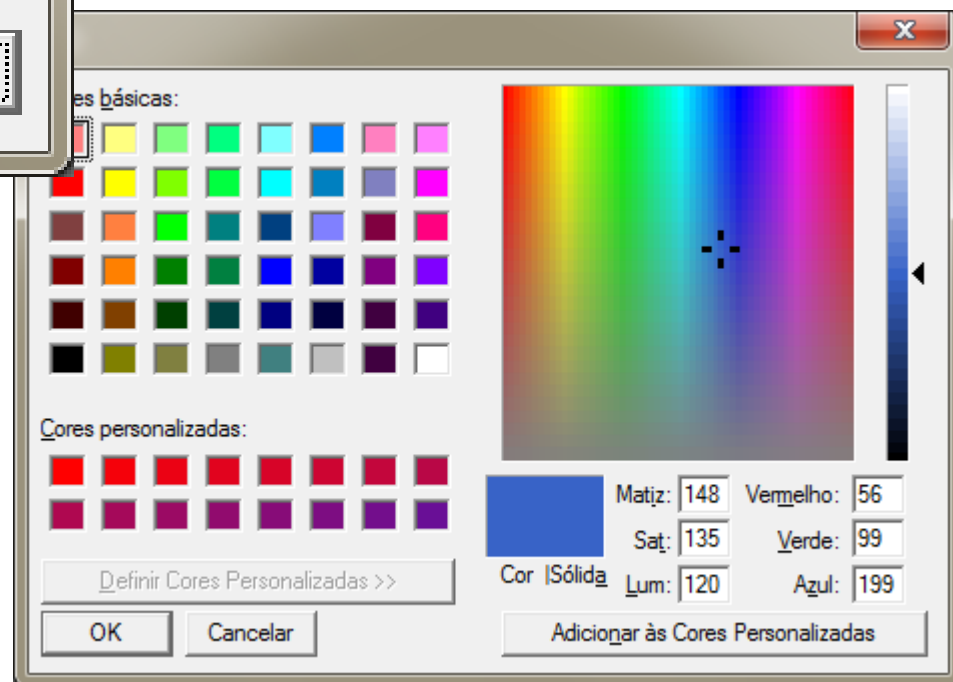
janelaCores()

janelaSobre()





janelaErro()



janelaCores()

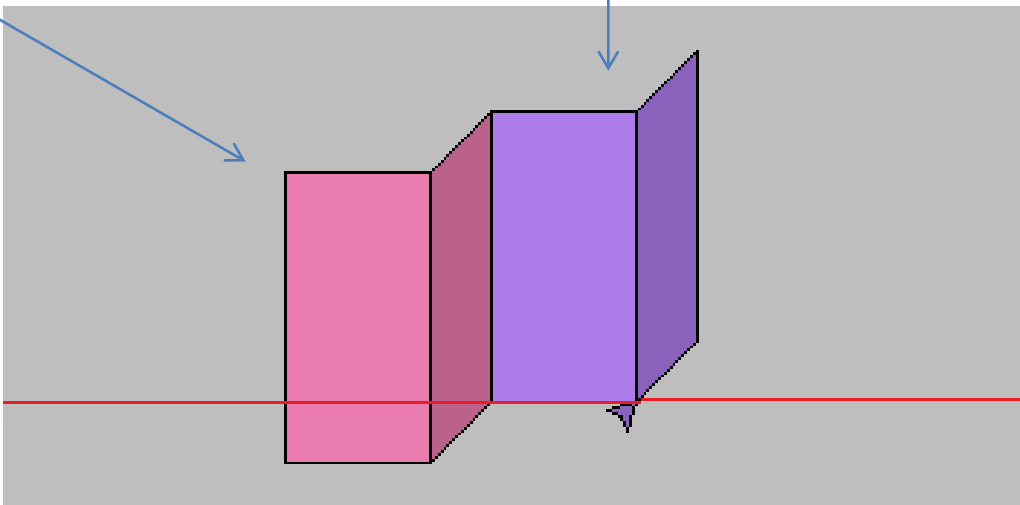


Métodos

Blocos principais

`paralelepipedo()`

`paralelepipedoInv()`





Métodos

Funções 'Construtoras'

hotel()

hotelInv()

casa()

casaInv()





Métodos

Fazendo cidades

cidade() Desatualizado

cidade_2()





Métodos

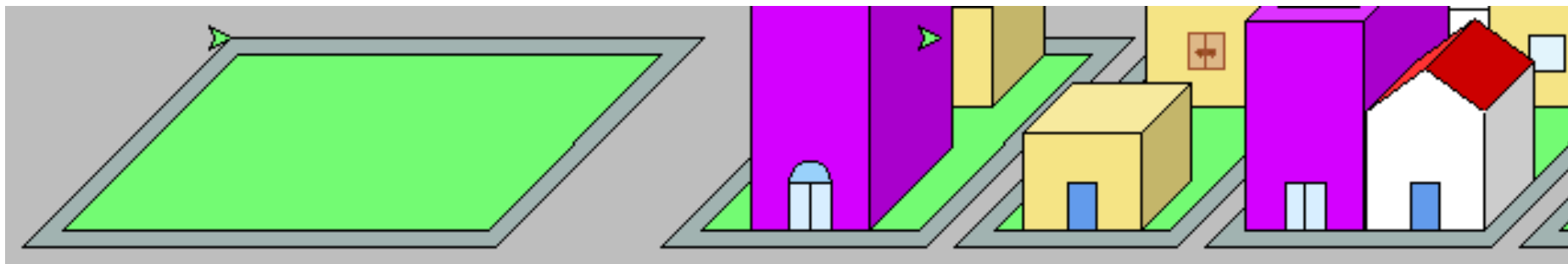
Fazendo cidades

quarteirao()

linhaQuarteiros()

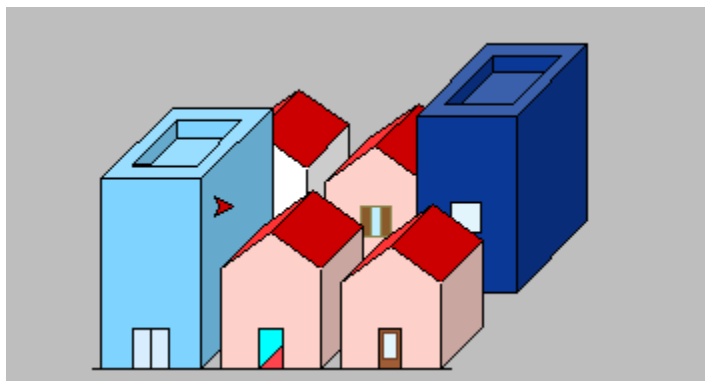
quarteiraoPredios()





quarteirao()

linhaQuarteiros()



quarteiraoPredios()



Métodos

Outros métodos

tamanhoPredios()

sortearPredio()

formatarElem1()

getPos()

verificarPos()





Atributos

prédios

cb.predios

cb.prediosProc

Cores

cb.coresHotel

cb.coresCasa

cb.coresLoja

Janelas

cb.janelasHotel

cb.janelasCasa

cb.janelasLoja

cb.janelasTodas

Tetos

cb.tetosHotel

cb.tetosCasa

cb.tetosLoja

cb.tetosTodas

Portas

cb.portasHotel

cb.portasCasa

cb.portasLoja

cb.portasTodas





cores.py

Manuseando cores

```
>>> cores = cores()
```





Métodos

O segredo da sombra...

converterDEC2HEX()

converterHEX2DEC()

clarear()

clarearDEC2HEX()

escurecer()

escurecerDEC2HEX()





Métodos

Outros métodos

janelaCores()

degradeQuad1()

sortearcor()





matematica.py

2 == 3?

```
>>> matematica = matematica()
```





Métodos

Ângulos

sen() cos() tg()

Em vez de usar:

```
>>> math.sin(math.radians(angulo))
```

Usa:

```
>>> matematica.sen(angulo)
```





Métodos

Módulo e afins... (enchendoLinguica = True)

modulo()

eNegativo()

ePositivo()





Métodos

Um pouco de Random em matematica...

sortearPeso()

Exemplo:

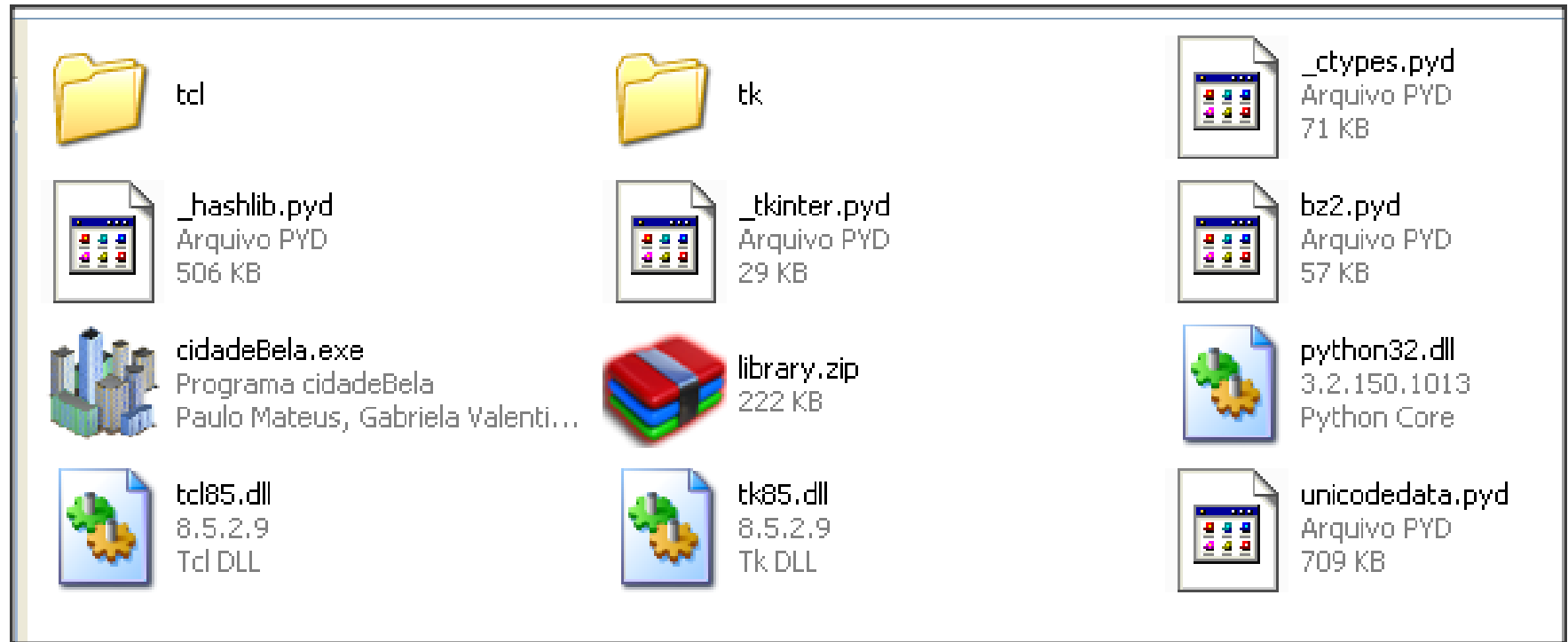
```
>>> matematica.sortearPeso([["bola", 2], ["ovo", 3]])
```

Retorna: (2/5 chances de enviar "bola" e 3/5 chances de enviar "ovo")





Compilar





Compilar

Bibliotecas utilizadas

`cx_freeze:`

Responsável por compilar

`win32api:`

É preciso da `win32api` para `cx_freeze` funcionar





Compiler

Links

cx_freeze:

<http://cx-freeze.sourceforge.net/>

win32api:

<http://sourceforge.net/projects/pythonce/>







Compilar

Para compilar seus projetos

Altere os seguintes arquivos

	<code>setup.bat</code> Arquivo em lotes do MS-DOS 1 KB		<code>setup.py</code> Python File 2 KB
---	--	---	--

(Disponíveis para download)



C:\WINDOWS\system32\cmd.exe

```
C:\Documents and Settings\Mateus\Desktop\cidadeBela\Trabalho>call setup.py build

running build
running build_exe
creating directory build\exe.win32-3.2
copying C:\Arquivos de programas\python32\lib\site-packages\cx_Freeze\bases\Win32GUI.exe -> build\exe.win32-3.2\cidadeBela.exe
copying C:\WINDOWS\system32\python32.dll -> build\exe.win32-3.2\python32.dll
```



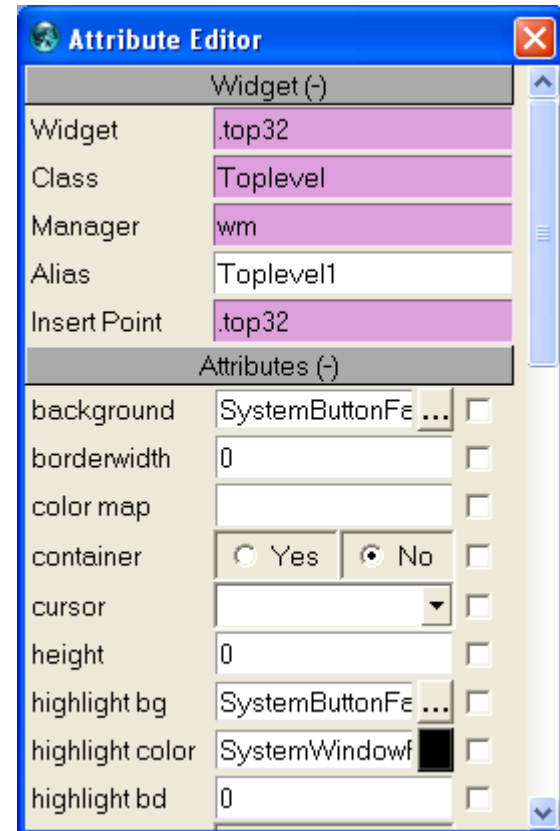
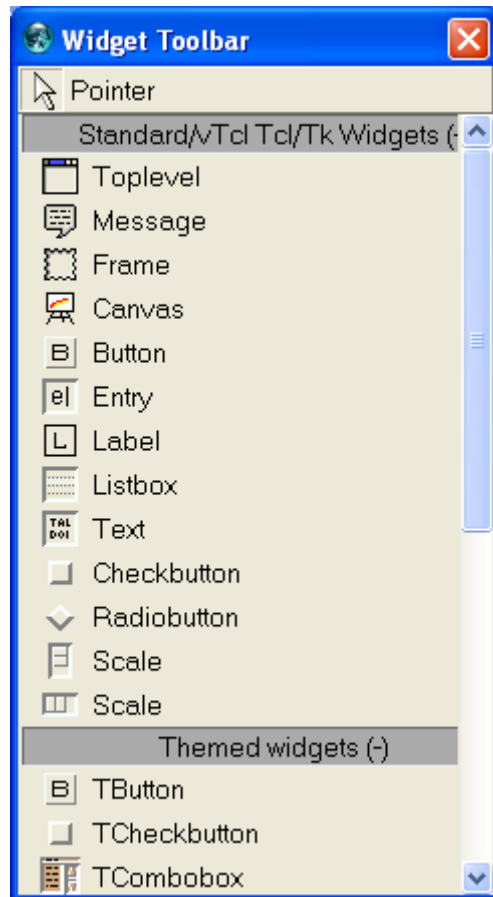
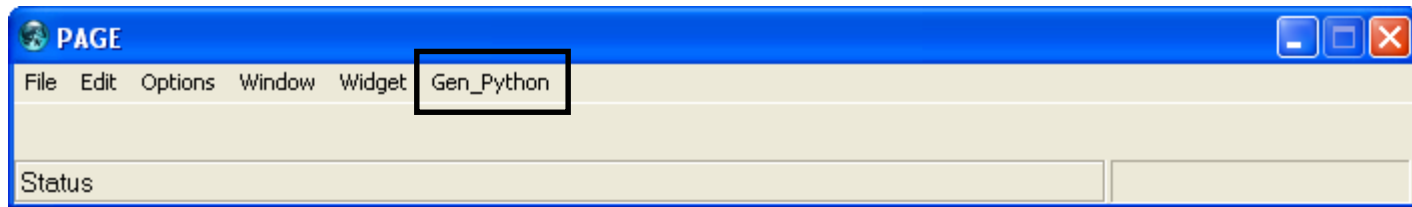
Janela gráfica: **GUI utilizada**

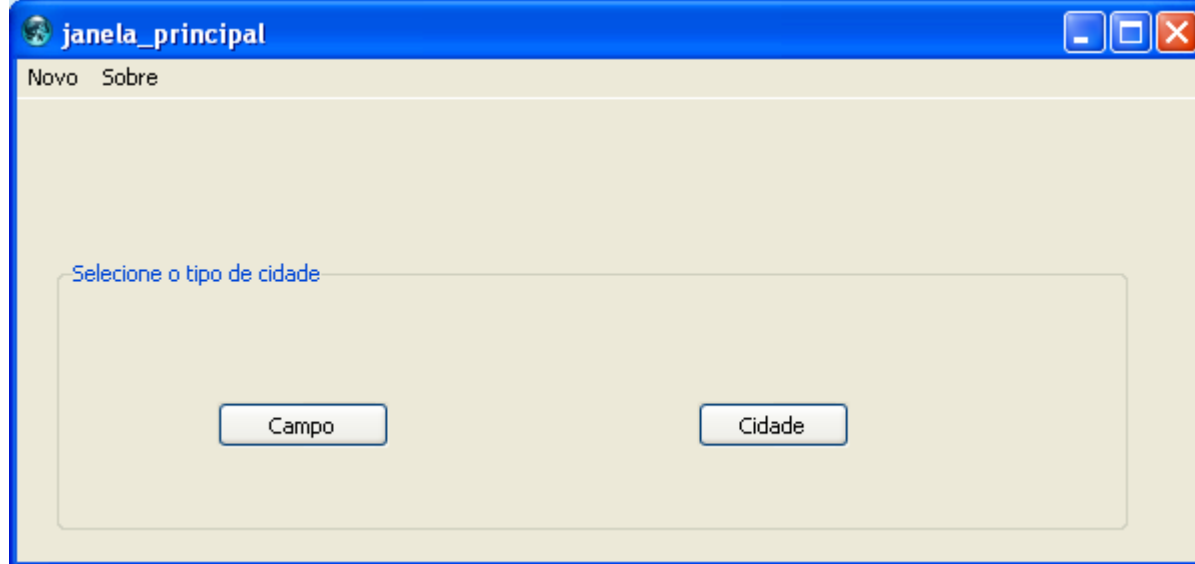


Download:

<http://sourceforge.net/projects/page/>





A screenshot of a Python Console window titled "Python Console". The window has a blue title bar with standard window controls. Below the title bar is a menu bar with one item: "Generated Python". The main area of the window is a light beige color with a vertical scrollbar on the right side. The console displays the following Python code:

```
#!/usr/bin/env python
# -*- python -*-

import sys

py2 = py30 = py31 = False
version = sys.hexversion
if version >= 0x020600F0 and version < 0x03000000 :
    py2 = True    # Python 2.6 or 2.7
    from Tkinter import *
    import ttk
elif version >= 0x03000000 and version < 0x03010000 :
    py30 = True
    from tkinter import *
    import ttk
elif version >= 0x03010000:
    py31 = True
    from tkinter import *
    import tkinter.ttk as ttk
else:
    print ("""
You do not have a version of python supporting ttk widgets..
You need a version >= 2.6 to execute PAGE modules.
""")
```



Janela gráfica: Rodar TCL



Download:

<http://www.activestate.com/tcl-dev-kit/downloads>





Instaladores

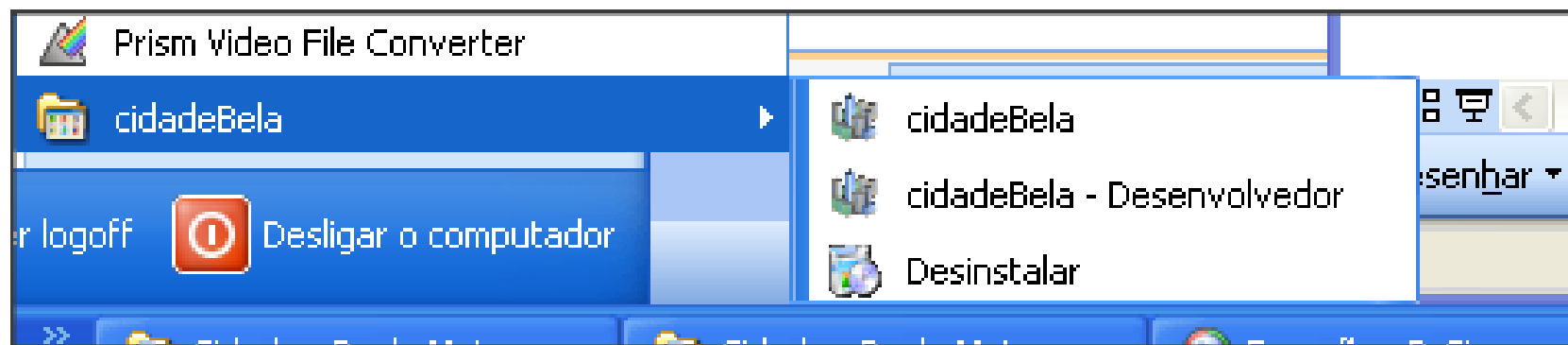
Inno Setup:

<http://www.jrsoftware.org/isinfo.php>

CreateInstall:

<http://www.createinstall.com/cifree/>







Dúvida?

```
>>> help()
```





Obrigado!

```
>>> turtle.bye()
```

