Insper

# Design de Software

Aula 14 – Tkinter

2016 – Engenharia

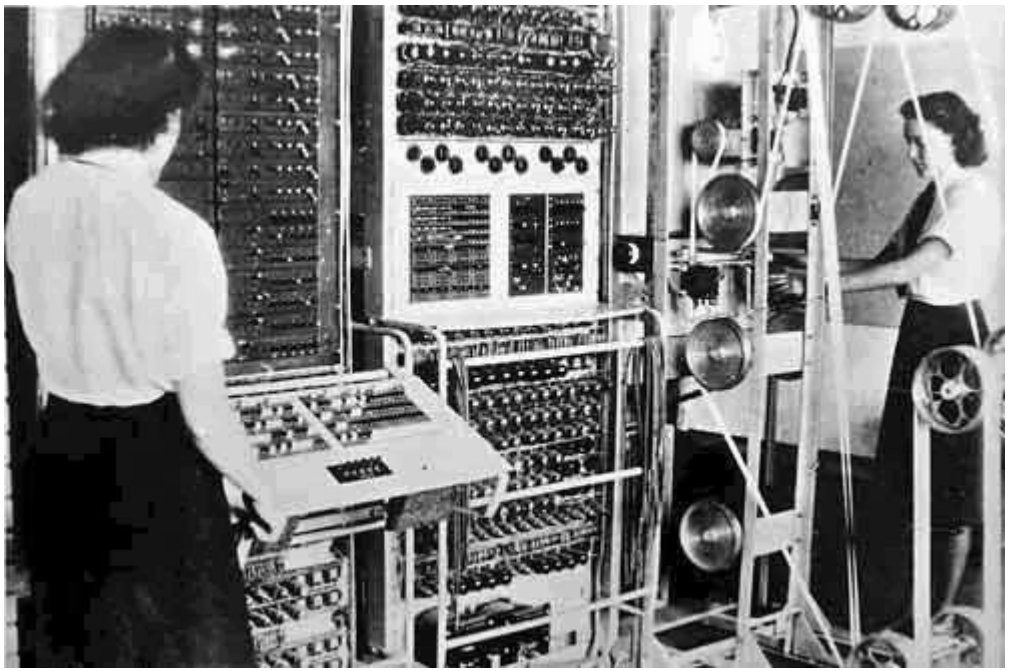
Fábio Ayres <fabioja@insper.edu.br>
Raul Ikeda <rauligs@insper.edu.br>

# Objetivo da Aula

- Programar interfaces visuais com Tkinter

# Interfaces de usuário: primórdios

No começo eram as luzinhas brilhantes e fitas de papel perfurado...

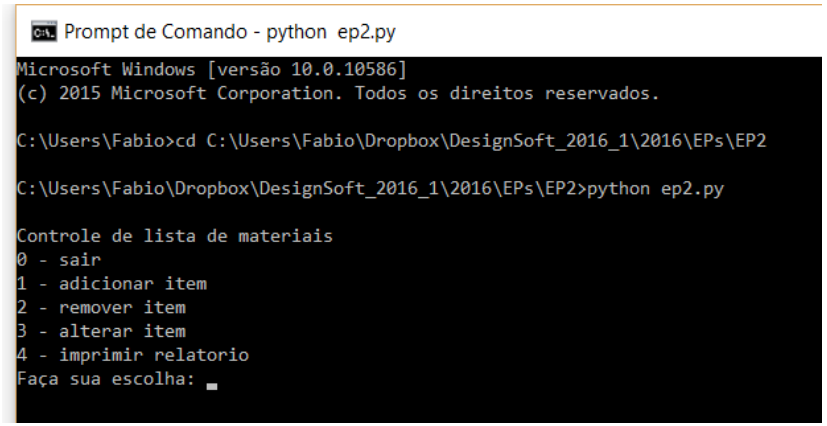


Fonte: Wikipedia: Colossus computer
https://en.wikipedia.org/wiki/Colossus_computer

# Interfaces de usuário: texto

- Em seguida, começaram a aparecer os terminais de texto
  - Muito mais fáceis de usar que fitinhas de papel e luzinhas!

By Jason Scott - Flickr: IMG_9976, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=29457452




**AINDA MUITO USADAS NA PRÁTICA!**

- Exemplo: acesso remoto a servidores!



Insper

# Interfaces gráficas

- Mas todos sabiam que teríamos que melhorar...


Ivan Sutherland : Sketchpad Demo (1/2)
https://youtu.be/USyoT_Ha_bA


Apple Lisa: inspirado no Xerox PARC Alto
https://youtu.be/-G9S-h2w2dU



Insper

# (Leitura cultural quase obrigatória)

**Triumph of the nerds**

- Baseado no livro "Accidental Empires", descrevendo o surgimento do Silicon Valley.
- Feito em 1996
  - Muita história rolou depois!
- O nome é uma paródia com a comédia "Revenge of the Nerds" (em português: "A Vingança dos Nerds").


https://youtu.be/yA54i2UifNc

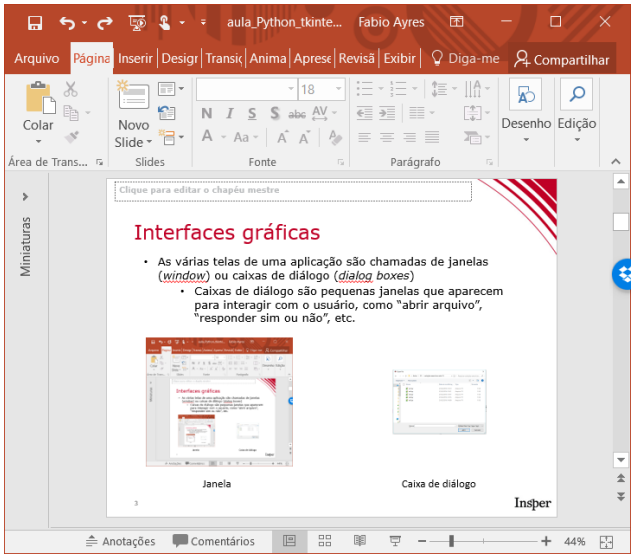
(hoje vale muito mais que meros US$100 milhões)



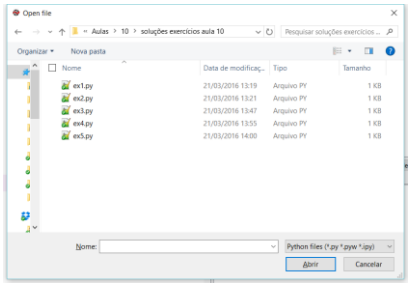
Insper

# Interfaces gráficas

- As várias telas de uma aplicação são chamadas de janelas (*window*) ou caixas de diálogo (*dialog boxes*)
    - Caixas de diálogo são pequenas janelas que aparecem para interagir com o usuário, como “abrir arquivo”, “responder sim ou não”, etc.




Janela




Caixa de diálogo



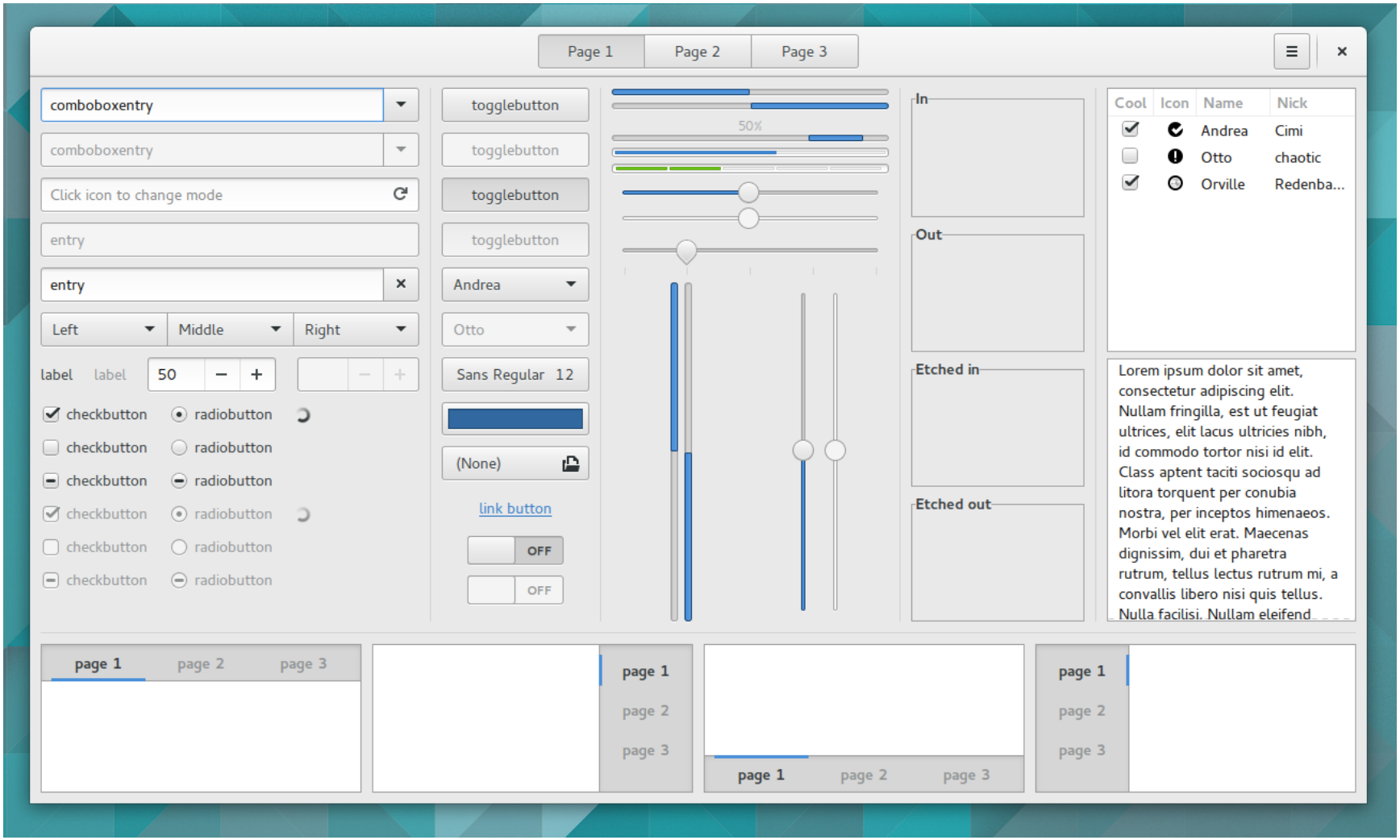
Insper

# Elementos de interfaces gráficas

- Os elementos internos de uma janela são genericamente chamados de *widgets* (https://pt.wikipedia.org/wiki/Widget)
- Tipos de widget em Tkinter:

| Alguns widgets basicos | Widgets avançados | Outros elementos |
|---|---|---|
| Frame | Listbox | Menus |
| Label | Scrollbar | Contextual Menus |
| Button | SizeGrip | Canvas |
| Checkbutton | Text | |
| Radiobutton | Progressbar | |
| Entry | Scale | |
| Combobox | Spinbox | |

# Elementos de interfaces gráficas




Fonte: https://en.wikipedia.org/wiki/GTK%2B

# Tkinter

- Tkinter é a biblioteca default do Python para o desenvolvimento de aplicações gráficas.
    - Baseada na velha interface Tk criada em 1991
    - Bibliotecas para desenvolvimento de sistemas também são chamadas de *frameworks*
- Mas não é a única, e certamente não é a mais bonita ou moderna!
    - wxPython: grande competidor do Tkinter como framework GUI mais popular do Python
    - PyQt: O poderoso framework Qt acessível via Python
    - Kivy: Framework moderno, ainda em desenvolvimento
        - Roda em desktops ou aplicativos *mobile*
        - Usa a filosofia moderna de separar código e design gráfico (ver a tríade web *HTML5+CSS+Javascript*)

# Tkinter

- Então, por que usaremos Tkinter nessa introdução?
    - Embora simples, apresenta os elementos essenciais da construção de interfaces gráficas
    - Já está instalado no seu computador!
- Desvantagens
    - Feio. Porém, se usarmos tkinter.ttk ao invés de tkinter, fica melhor: usa componentes nativos
    - Simples: se precisar de coisas mais avançadas (como widgets mais sofisticados ou telas de toque) use outros frameworks.

# Criar uma janela

Biblioteca padrão de interface gráfica do Python.

apelido

```
import tkinter as tk
```

Cria a janela

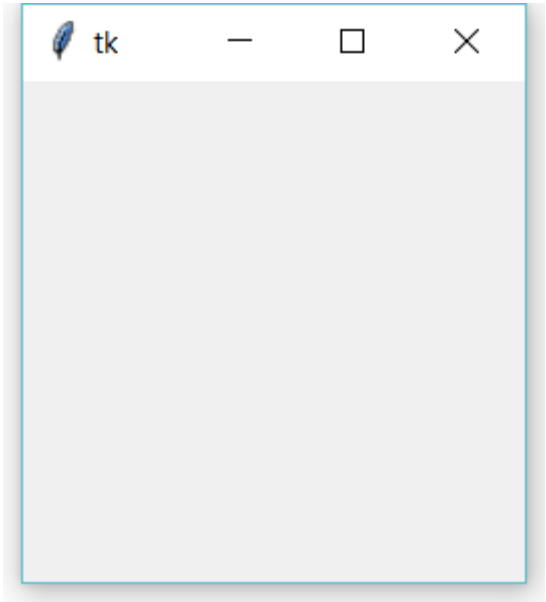
```
window = tk.Tk()
window.mainloop()
```

Põe o loop de eventos para rodar. Em outras palavras, começa a aplicação.

Nota:
Em Python 3 a biblioteca do Tkinter chama-se **tkinter**.
Em Python 2 chamava-se **Tkinter**.
Muitos tutoriais online ainda falam da versão Python 2, cuidado!
Troque **Tkinter** por **tkinter** nesses casos!

# Criar uma janela

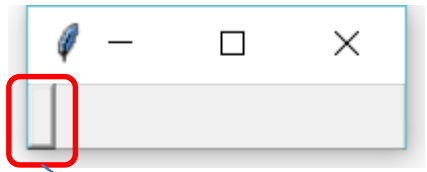
```
import tkinter as tk

window = tk.Tk()
window.mainloop()
```



# Adicionar um botão

```
import tkinter as tk

window = tk.Tk()

botão = tk.Button(window)
botão.grid()

window.mainloop()
```

Onde o botão vai aparecer.

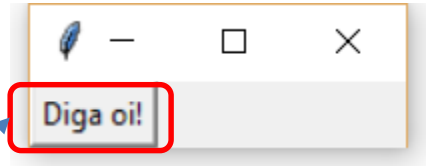


Que botão feio!
E não faz nada!

*Layout manager*:
posiciona e exibe o botão.
Receita de bolo por enquanto

# Adicionando propriedades

```
import tkinter as tk

window = tk.Tk()

botão = tk.Button(window)
botão.configure(text="Diga oi!")
botão.grid()

window.mainloop()
```




Modifica a propriedade **text** do botão

# Adicionando comportamento

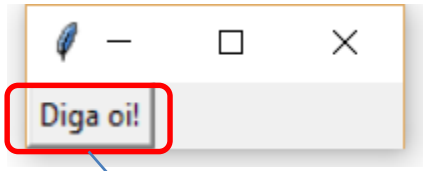
- Para adicionar comportamento, vamos usar *callbacks*
- Um *callback* é uma função que passamos à um widget, e que será executada quando interagirmos com o widget
- O nome vem do inglês *call back*, que significa (numa tradução livre) “ligar de volta”.
    - É como se você falasse com a secretária do widget, e deixasse um número de telefone para que o widget ligue de volta para você

# Adicionando um callback

```
import tkinter as tk

def diz_oi():
    print("Oi gente!")

window = tk.Tk()

botão = tk.Button(window)
botão.configure(text="Diga oi!")
botão.configure(command=diz_oi)
botão.grid()

window.mainloop()
```

Função a ser usada como *callback*

Adicionando o *callback*




Agora, quando clicamos no botão, o Python chama a função diz_oi()

# Organizando o código em classes

- Agora já está valendo organizar nosso aplicativo em uma classe! Veja este exemplo:

```
import tkinter as tk

class MeuAplicativo:

    def __init__(self):
        self.window = tk.Tk()
        self.window.title("Meu primeiro aplicativo!")

        botão = tk.Button(self.window)
        botão.configure(text="Diga oi!")
        botão.configure(command=self.diz_oi)
        botão.grid()

    def iniciar(self):
        self.window.mainloop()

    def diz_oi(self):
        print("Oi gente!")

app = MeuAplicativo()
app.iniciar()
```

# Projetando a interface

- Antes de continuar, vamos pensar no que gostaríamos de fazer. Vamos fazer um aplicativo onde
  - você escreve uma mensagem numa caixa de entrada de texto e, ...
  - ao apertar um botão, ...
  - essa mensagem é escrita numa outra parte da interface.

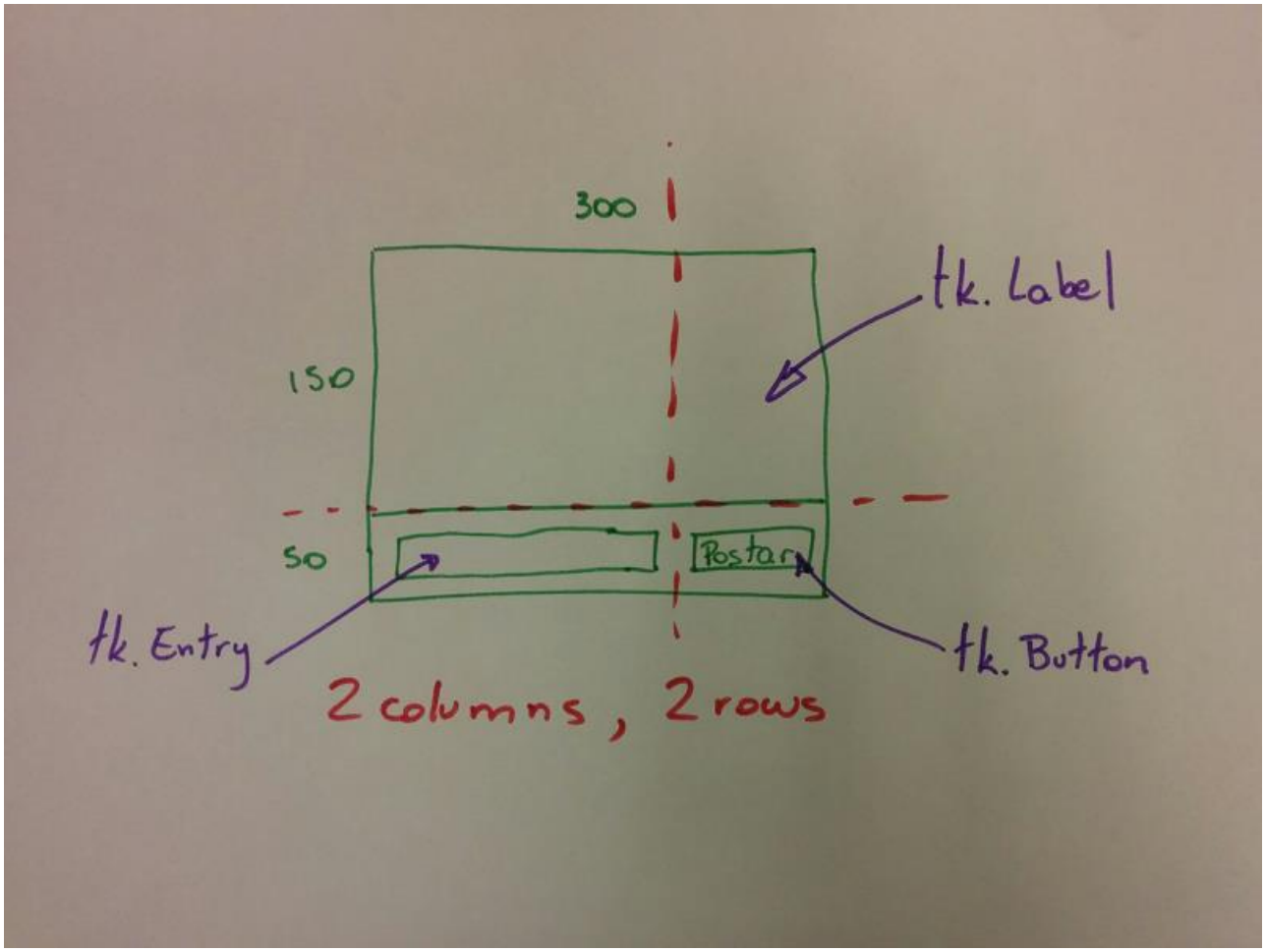
- A nossa primeira tarefa é desenhar esta interface
  - Em outros *frameworks* de interface gráfica existem editores de tela.

  - Tkinter não tem um bom editor, infelizmente. Usaremos uma ferramenta avançada, *wireless,* de alta resolução e portável...

    - Papel e caneta!

# Projetando a interface

# Projetando a interface






Insper

# Resultado

- Arquivo: aula_tkinter_big.py

# Aprendendo a aprender

- Vamos ver juntos algumas referencias interessantes de Tkinter:
  - http://www.tkdocs.com/tutorial/index.html
    - Aqui é o lugar certo para aprender Tk!
  - http://www.tcl.tk/man/tcl8.5/TkCmd/contents.htm
    - Documentação oficial do Tk, onde tudo começou. Medonha, mas é o que tem pra hoje. Não tem nada de Python, mas é "traduzível" para Python. Não tenha medo!
  - https://docs.python.org/3.5/library/tkinter.html
    - Documentação oficial do tkinter

# Exercícios

1. Estude o mecanismo de 'bind' (atrelar) de eventos à callbacks.
   - http://www.tkdocs.com/tutorial/concepts.html#events
   - http://effbot.org/tkinterbook/tkinter-events-and-bindings.htm
   - Nota: tem uns "lambda" misteriosos lá, que são funções feitas "on-the-fly". Troque aquilo por callbacks normais e siga em frente.

2. Agora, modifique o código de aula_tkinter_big.py para que a mensagem seja postada também quando o usuário pressionar Tab

3. Agora adicione um "bind" no widget de Label para que quando o mouse entre no Label, a mensagem seja apagada. Dica: estude o código em http://www.tkdocs.com/tutorial/concepts.html#events

# Exercícios

4. Canvas: Estude a documentação e o exemplo completo da seção "Creating items" em http://www.tkdocs.com/tutorial/canvas.html

5. Esse código de exemplo é muito feio! Usa variáveis globais, e não usa classes! O professor ficou tão nauseado com esse código que o reescreveu do jeito correto: veja o arquivo aula_tkinter_canvas.py.

6. Faça uma aplicação com um canvas e use os métodos do objeto do canvas para desenhar algo! Veja a documentação em http://infohost.nmt.edu/tcc/help/pubs/tkinter/web/canvas.html

7. Em https://gist.github.com/calebrob6/4022622 o autor fez um jogo de Pong em tkinter! Só que está feito em Python 2.7... Você consegue atualizá-lo para Python 3.5? Não é difícil!
   - Nota: o código é meio porco...

Insper

www.insper.edu.br