VOLTAR

Criando uma tela de login recebendo dados do usuário

Criar um exemplo de tela de login que recebe dados digitados pelo usuário.

NESTE TÓPICO

Marcar tópico

Olá alunos,

Utilizando a parte gráfica (GUI) do Python, o módulo Tkinter, vamos construir uma tela de login com os recursos da parte procedural da linguagem, com a criação de funções denominadas.

Do módulo Tkinter, vamos utilizar os objetos, widgets, frames e containers para a construção da tela de login.

CONSTRUINDO A TELA DE LOGIN

Vamos iniciar, invocando a biblioteca do **Tkinter**:

```
1.  from tkinter import *
```

O que significa que da biblioteca Tkinter vamos utilizar todas as suas funções, com o curinga asterisco.

Continuando com o código:

```
1.  from tkinter import *
2.
3.  tela = Tk() #instancia a classe 'Tk()' através da variável 'tela'
4.  tela.title("TELA DE LOGIN")
5.  tela["bg"] = "light green"
6.  tela.geometry("400x250") #define o tamanho da tela em width (largura) x height (altu
    ra)
7.  Principal(tela) #instância da classe Principal, passamos a variável tela como parâme
    tro
8.  tela.mainloop() #método obrigatório para carregar a tela e para que os eventos ocorr
    am
```

Na linha **3**, criamos uma variável **tela** para instanciar a classe **Tk()** (tkinter) com todas as funções que importamos. É por esta linha que o interpretador vai iniciar a execução da aplicação.

Depois utilizamos a função **title** para dar um título para a tela (linha **4**).

Nas linhas **5** e **6**, definimos uma cor para a tela e o tamanho dela (a medida é em pixels).

Na linha **7**, chamamos a classe **Principal**, passando como parâmetro, a variável **tela**.

Na linha **8**, a função para que tudo seja executado recursivamente, o **loop** (a tela e seus eventos), até que o usuário encerre a aplicação.

Continuando...

```
class Principal: #cria a classe principal
    def __init__(self, master=None):  #cria o método inicial e que será filho da classe Principal
        self.fontePadrao = ("Arial", "10")
        self.Container1 = Frame(master) #definimos o container pai com um frame
         self.Container1["pady"] = 10 #posiciona o container na tela a partir do top e bottom
        self.Container1["bg"] = "light green" # define uma cor de fundo
        self.Container1.pack() #para exibir os widgets do container

        self.Container2 = Frame(master) #cria um container para label e text field nome
        self.Container2["padx"] = 20 #posiciona o container na tela a partir de left e right
        self.Container2["bg"] = "light green"
        self.Container2.pack()

        self.Container3 = Frame(master) #cria um container para label e text field senha
        self.Container3["padx"] = 20
        self.Container3["bg"] = "light green"
        self.Container3.pack()

        self.Container4 = Frame(master) #cria um container para o botão AUTENTICAR e a label MENSAGEM
        self.Container4["pady"] = 20
        self.Container4["bg"] = "light green"
        self.Container4.pack()

        self.titulo = Label(self.Container1, text="Dados do usuário:")
        self.titulo["font"] = ("Arial", "10", "bold")
        self.titulo["bg"] = "light green"
        self.titulo.pack()

        self.nomeLabel = Label(self.Container2,text=" Nome:", font=self.fontePadrao)
        self.nomeLabel["bg"] = "light green"
        self.nomeLabel.pack(side=LEFT) #define o lado da exbição

        self.nome = Entry(self.Container2) #utiliza o método Entry (input) para receber dados via teclado
        self.nome.focus() #determina a posição do cursor na caixa de texto
        self.nome["width"] = 30
        self.nome["font"] = self.fontePadrao
        self.nome.pack(side=LEFT)

        self.senhaLabel = Label(self.Container3, text="Senha:", font=self.fontePadrao)
        self.senhaLabel["bg"] = "light green"
        self.senhaLabel.pack(side=LEFT)
        self.senha = Entry(self.Container3)
        self.senha["width"] = 30
        self.senha["font"] = self.fontePadrao
        self.senha["show"] = "*"
        self.senha.pack(side=LEFT)

        self.autenticar = Button(self.Container4)
        self.autenticar["text"] = "AUTENTICAR"
        self.autenticar["font"] = ("Calibri", "10","bold")
        self.autenticar["width"] = 12
        self.autenticar["bg"] = "white"
        #self.autenticar["command"] = self.verificaSenha
        self.autenticar.pack()

        self.sair = Button()
        self.sair["text"] = "SAIR"
        self.sair["font"] = ("Calibri", "10","bold")
        self.sair["width"] = 5
        self.sair["command"] = quit
        self.sair.pack(side=TOP)

        self.mensagem = Label(self.Container4, text="", font=self.fontePadrao)
```

```
64.          self.mensagem["bg"] = "light green"
65.          self.mensagem.pack()
66.
67.  tela = Tk() #instancia a classe 'Tk()' através da variável 'tela'
68.  tela.title("TELA DE LOGIN")
69.  tela["bg"] = "light green"
70.  tela.geometry("400x250") #define o tamanho da tela em width (largura) x height (altu
     ra)
71.  Principal(tela) #instância da classe Principal, passamos a variável tela como parâme
     tro
72.  tela.mainloop() #método obrigatório para carregar a tela e para que os eventos ocorr
     am
```

Na linha **1**, criamos a classe Principal, que foi chamada pela linha **71**, a partir daí, até a linha **65**, tudo vai pertencer a esta classe (preste atenção na endentação).

Na linha **2**, criamos o método **def** que pertence à classe Principal (**master=none**), com o nome **init**, os sinais de "**__**" significa que é um método privado.

Utilizamos a partir daí o objeto padrão do Python que é o **self** para fazer referencias com todos os objetos da tela.

A partir daí (da linha **4** até a linha **22**), criamos os **Container1, 2, 3 e 4**, passando vários parâmetros: cor de fundo, posição. E determinamos cada container como um widget (objeto) tipo **Frame.**

A partir da linha **24**, criamos um widget tipo **Label**, que apresentará um título na tela, que pertence ao **Container1**.

Da linha **29** até a linha **31**, criamos um **Label** e uma caixa de texto para receber o nome do usuário, que pertence ao **Container2**.

Da linha **39** até a linha **46**, criamos um **Label** e uma caixa de texto para receber a senha do usuário, que pertence ao **Container3**.

Da linha **48** até a linha **54**, criamos um widget tipo **Button** que chamará o método **verificaSenha** (será implementado a seguir), que pertence ao **Container4**.

Da linha **56** até a linha **61**, criamos um widget tipo **Button** que ao ser clicado, encerrará a aplicação, pois passamos o parâmetro "**command"= quit**, este botão não pertence a nenhum container.

Da linha **63** até a linha **65**, criamos outro **Label**, que deixará uma mensagem em branco, a mensagem será preenchida posteriormente pelo método **verificaSenha**, este **Label** pertence ao **Container4**.

Veja no vídeo abaixo como ficou a nossa tela:

Agora vamos completar a aplicação, criando um método para verificar a senha:

```
1.  #Método para verificar senha
2.      def verificaSenha(self):
3.          usuario = self.nome.get() #apanha os dados digitados pelo usuário no text fi
    eld 'nome'
4.          senha = self.senha.get()
5.          if usuario == "Denilson" and senha == "123":
6.              self.mensagem["text"] = "Autenticado!"
7.              self.nome["fg"] = "gray"
8.              self.senha["fg"] = "gray"
9.              self.sair.focus_force()
10.         else:
11.             self.mensagem["text"] = "Usuário e/ou senha incorretos!"
12.             self.senha.delete(0,END)
13.             self.nome.delete(0,END)
14.             self.nome.focus()
```

Na linha **2**, criamos o método **verificaSenha**, passando como argumento o objeto padrão **self**.

Na linha **3** e **4**, criamos as variáveis **nome** e **senha** que recebe o nome e a senha digitado pelo usuário.

Na linha **5**, abrimos um bloco **if** para verificar se o usuário e a senha estão corretas. Se estiverem, o método **mensagem** é chamado e exibirá o texto **Autenticado!**. Em seguida o nome e a senha mudam a cor da letra para cinza (**gray**). E por último coloca o foco no botão **sair** (linha **9**).

Senão, chama o método **mensagem** para exibir "**Usuário e/ou senha incorretos!**", limpa os campos do nome e da senha, utilizando a função: **delete(0,END)** e muda o foco do cursor para o campo **nome** (linha **14**).

Observação: A função **delete(n,m)** é acompanhada de parâmetros que determinada o que você quer apagar, lembrando que a primeira posição inicia com zero (0). Então passamos como valores de parâmetros: **(0,END)**, ou seja, vou apagar todo o texto do campo digitado pelo usuário, do 0 até o fim (END). Exemplo: se você colocar delete(1,4) e o usuário digitar: **uninove**, será apagado da segunda até a quinta posição e ficaria assim: **u ve**.

Vamos colocar a linha de comando: **self.autenticar["command"] = self.verificaSenha** no botão AUTENTICAR. Quando este botão for clicado, ele chamará o método **verificaSenha**.

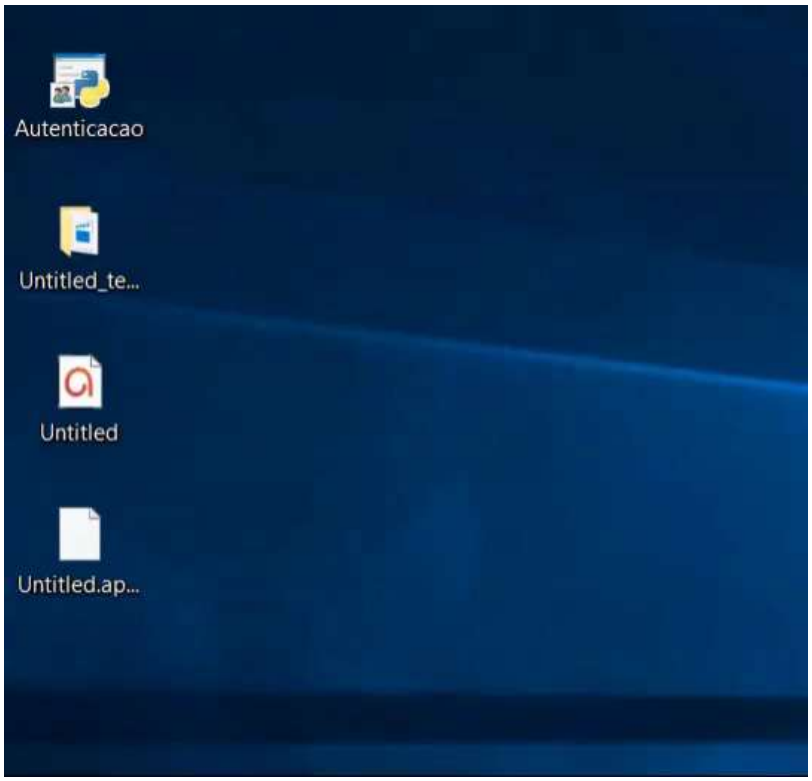
Mude o nome e a senha do bloco **if**, criando o login com seu nome e senha! (linha **5**).

Agora, vejam como ficou o código final:

```
from tkinter import *

class Principal: #cria a classe principal
    def __init__(self, master=None):  #cria o método inicial e que será filho da cla
sse Principal
        self.fontePadrao = ("Arial", "10")
        self.Container1 = Frame(master) #definimos o container pai com um frame
         self.Container1["pady"] = 10 #posiciona o container na tela a partir do top
e bottom
        self.Container1["bg"] = "light green" # define uma cor de fundo
        self.Container1.pack() #para exibir os widgets do container

        self.Container2 = Frame(master) #cria um container para label e text field n
ome
        self.Container2["padx"] = 20 #posiciona o container na tela a partir de left
e right
        self.Container2["bg"] = "light green"
        self.Container2.pack()

        self.Container3 = Frame(master) #cria um container para label e text field s
enha
        self.Container3["padx"] = 20
        self.Container3["bg"] = "light green"
        self.Container3.pack()

        self.Container4 = Frame(master) #cria um container para o botão AUTENTICAR e
a label MENSAGEM
        self.Container4["pady"] = 20
        self.Container4["bg"] = "light green"
        self.Container4.pack()

        self.titulo = Label(self.Container1, text="Dados do usuário:")
        self.titulo["font"] = ("Arial", "10", "bold")
        self.titulo["bg"] = "light green"
        self.titulo.pack()

        self.nomeLabel = Label(self.Container2,text=" Nome:", font=self.fontePadrao)
        self.nomeLabel["bg"] = "light green"
        self.nomeLabel.pack(side=LEFT) #define o lado da exbição

        self.nome = Entry(self.Container2) #utiliza o método Entry (input) para rece
ber dados via teclado
        self.nome.focus() #determina a posição do cursor na caixa de texto
        self.nome["width"] = 30
        self.nome["font"] = self.fontePadrao
        self.nome.pack(side=LEFT)

        self.senhaLabel = Label(self.Container3, text="Senha:", font=self.fontePadra
o)
        self.senhaLabel["bg"] = "light green"
        self.senhaLabel.pack(side=LEFT)

        self.senha = Entry(self.Container3)
        self.senha["width"] = 30
        self.senha["font"] = self.fontePadrao
        self.senha["show"] = "*"
        self.senha.pack(side=LEFT)

        self.autenticar = Button(self.Container4)
        self.autenticar["text"] = "AUTENTICAR"
        self.autenticar["font"] = ("Calibri", "10","bold")
        self.autenticar["width"] = 12
        self.autenticar["bg"] = "white"
        self.autenticar["command"] = self.verificaSenha
        self.autenticar.pack()

        self.sair = Button()
        self.sair["text"] = "SAIR"
        self.sair["font"] = ("Calibri", "10","bold")
        self.sair["width"] = 5
        self.sair["command"] = quit
```

```
64.          self.sair.pack(side=TOP)
65.
66.          self.mensagem = Label(self.Container4, text="", font=self.fontePadrao)
67.          self.mensagem["bg"] = "light green"
68.          self.mensagem.pack()
69.
70.      #Método para verificar senha
71.      def verificaSenha(self):
72.          usuario = self.nome.get() #apanha os dados digitados pelo usuário no text fi
     eld 'nome'
73.          senha = self.senha.get()
74.
75.          if usuario == "Denilson" and senha == "123":
76.              self.mensagem["text"] = "Autenticado!"
77.              self.nome["fg"] = "gray"
78.              self.senha["fg"] = "gray"
79.              self.sair.focus_force()
80.          else:
81.              self.mensagem["text"] = "Usuário e/ou senha incorretos!"
82.              self.senha.delete(0,END)
83.              self.nome.delete(0,END)
84.              self.nome.focus()
85.
86.  tela = Tk() #instancia a classe 'Tk()' através da variável 'tela'
87.  tela.title("TELA DE LOGIN")
88.  tela["bg"] = "light green"
89.  tela.geometry("400x250") #define o tamanho da tela em width (largura) x height (altu
     ra)
90.  Principal(tela) #instância da classe Principal, passamos a variável tela como parâme
     tro
91.  tela.mainloop() #método obrigatório para carregar a tela e para que os eventos ocorr
     am
```

Para ver a aplicação executando, vejam o vídeo abaixo:



Este é um exemplo de como podemos validar um login, dá para incrementar, apanhando os dados de login de uma tabela.

SAIBA MAIS...

Dê uma olhada nos links abaixo para saber mais sobre a linguagem Python:

https://www.python.org/doc/ (https://www.python.org/doc/)

https://wiki.python.org/moin/PythonBooks (https://wiki.python.org/moin/PythonBooks)

Neste tópico vimos como criar uma tela de login, recebendo os dados do usuário e validando-os.

Quiz

Exercício Final

Criando uma tela de login recebendo dados do usuário

INICIAR

Referências

SUMMERFIELD, M. *Programação em Python 3*: Uma introdução completa à linguagem Python. Rio de Janeiro Alta Books, 2012. 495 p.

MENEZES, N. N. C. *Introdução à programação com Python:* algoritmos e lógica de programação para iniciantes. 2. ed. São Paulo: Novatec, 2014. 328 p.

SWEIGART, AL. *Automatize tarefas maçantes com Python:* programação prática para verdadeiros iniciantes. São Paulo: Novatec, 2015. 568 p.

PYTHON, doc. Disponível em: <https://www.python.org/doc/>. Acesso em: Junho/2018.

PYTHON, books. Disponível em: <https://wiki.python.org/moin/PythonBooks>. Acesso em: Junho/2018.

Avalie este tópico

ANTERIOR
Programação em GUI com Tkinter

Índice

PRÓXIMO
Criando script de cavalo de tróia para logging em te...

Biblioteca (https://www.uninove.br/conheca-a-uninove/biblioteca/sobre-a-biblioteca/apresentacao/)
Portal Uninove (http://www.uninove.br)
Mapa do Site

® Todos os direitos reservados

Ajuda? (https://ava.uninove.br/seu/AVA/topico/topico.php?idCurso=)