

# Documentación: Calculadora TAP con Flet

Este documento detalla el funcionamiento de una interfaz gráfica para una calculadora básica desarrollada en Python utilizando la librería **Flet**. El código actual implementa la visualización, botones numéricos del 1 al 4, un botón de limpieza y la lógica para capturar los clics.

## Parte 1: Configuración en Mac

Antes de tocar el código, necesitamos preparar tu computadora. Flet es una librería de Python, por lo que requerimos tener Python instalado.

### 1. Verificar Python

Abre tu Terminal (puedes buscarla presionando Command + Espacio y escribiendo "Terminal") y escribe:

**python3 –version**

Si ves algo como Python 3.10.x (o superior), estás listo. Si no, descárgalo desde [python.org](https://python.org).

### 2. Instalar Flet

En la misma terminal, ejecuta el siguiente comando para descargar e instalar la librería necesaria:

**pip3 install flet**

## **Parte 2: Cómo correr el código**

1. **Crear el archivo:** Abre tu editor de código favorito (como VS Code) o un editor de texto simple.
  2. **Pegar el código:**

The screenshot shows a code editor interface with a dark theme. The left sidebar contains various icons for file operations, search, and navigation. The top bar includes tabs for 'main2.py' and 'Main.py', a search bar with the text 'venv', and a tab labeled 'BLACKBOX'. The main area displays Python code using the Flet library to create a mobile-style calculator application. The code defines a main page with a title, window dimensions, and a container for the display and buttons. It handles button clicks to append digits or operators to the display, clear the display, and update the screen. A grid layout is used for the numeric and operator buttons. The right side of the interface features a vertical panel titled 'Build with Agent' containing AI-related controls and status messages.

```
1 import flet as ft
2
3 def main(page: ft.Page):
4
5     page.title = "Calculadora TAP"
6     page.window_width = 250 # Ancho estrecho, estilo móvil
7     page.window_height = 400 # Altura definida
8     page.padding = 20 # Espacio interno para que nada toque los bordes
9
10    display = ft.Container(
11        content=ft.Text("0", size=30), # El texto inicial es "0"
12        bgcolor=ft.colors.BLACK12, # Fondo gris oscuro
13        border_radius=8, # Bordes redondeados
14        alignment=ft.alignment.Alignment(1, 0), # Alinear texto a la derecha
15        padding=10, # ... otras configuraciones de tamaño
16        width=10,
17        height=70,
18    )
19
20    def on_button_click(e):
21
22        if display.content.value == "0":
23            # Si hay un 0 solito, lo reemplazamos por el nuevo número
24            display.content.value = e.control.content.value
25        else:
26            # Si ya hay números, agregamos el nuevo al final (concatenar)
27            display.content.value += e.control.content.value
28        page.update() # IMPORTANTE! Refrescar la pantalla
29
30    def on_clear_click(e):
31
32        display.content.value = "0" # Reiniciar a 0
33        page.update()
34
35    grid = ft.GridView(
36        runs_count=2, # Define que habrá 2 columnas
37        spacing=10, # Espacio entre botones
38        run_spacing=10,
39        width=10,
40        height=200,
41        expand=False
42    )
43
44    grid.controls.append(
45        grid.controls.append(
```

The screenshot shows a Python IDE interface with a dark theme. The top bar includes tabs for 'Bienvenido', 'main2.py', and 'Main.py'. A search bar is labeled 'venv'. The left sidebar contains various icons for file operations like copy, paste, find, and refresh. The main code editor displays Python code for a Flet application. The code defines a function 'main' that creates a grid with four buttons. Each button has a different color and text ('1', '2', '3', '4') based on its index. The 'on\_click' event for each button is set to a function named 'on\_button\_click'. At the bottom of the code, there is a line for 'clear\_button'. The status bar at the bottom shows the line number (107), column (1), spaces (4), encoding (UTF-8), file type (Python), version (3.14.2 (venv)), and other system information.

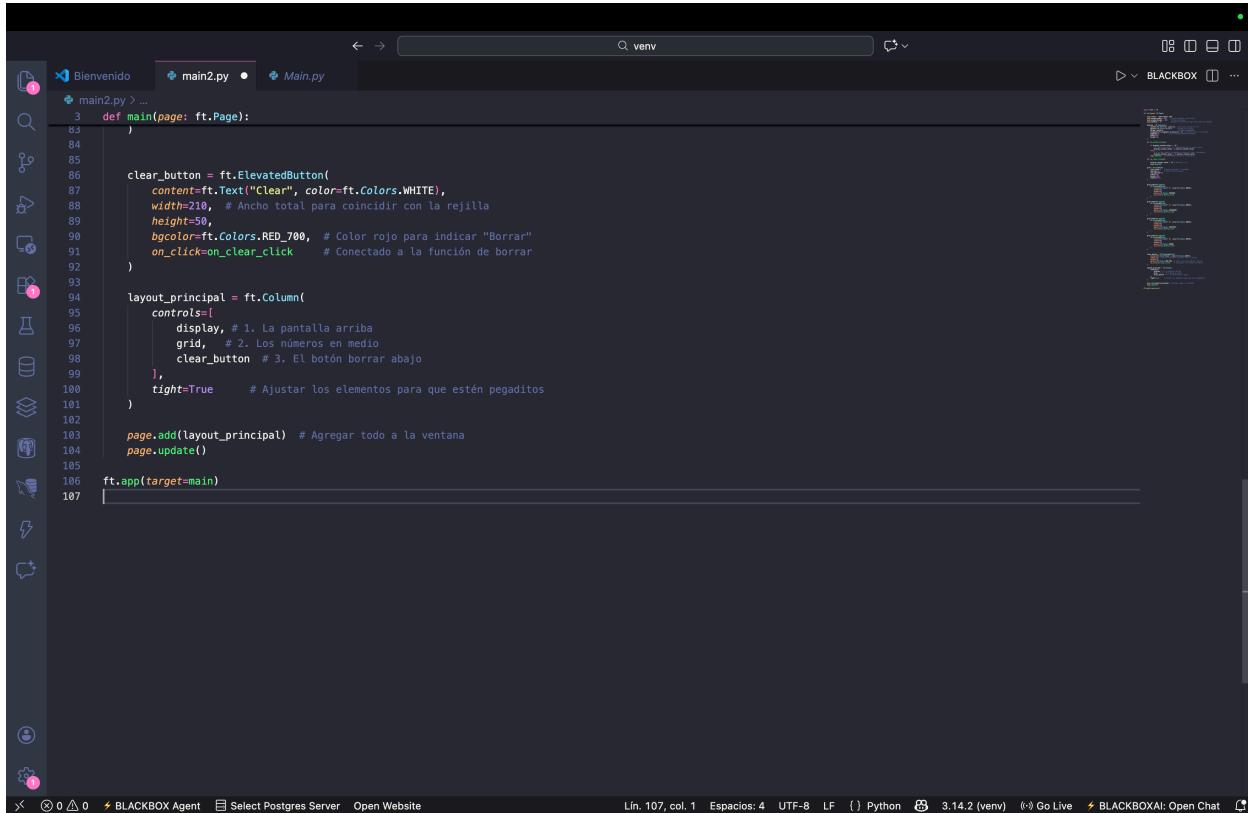
```
def main(page: ft.Page):
    grid.controls.append(
        ft.ElevatedButton(
            content=ft.Text(" 1 ", color=ft.Colors.WHITE),
            width=100,
            height=50,
            bgcolor=ft.Colors.PRIMARY,
            on_click=on_button_click
        )
    )

    grid.controls.append(
        ft.ElevatedButton(
            content=ft.Text(" 2 ", color=ft.Colors.WHITE),
            width=100,
            height=50,
            bgcolor=ft.Colors.SECONDARY,
            on_click=on_button_click
        )
    )

    grid.controls.append(
        ft.ElevatedButton(
            content=ft.Text(" 3 ", color=ft.Colors.WHITE),
            width=100,
            height=50,
            bgcolor=ft.Colors.TERTIARY,
            on_click=on_button_click
        )
    )

    grid.controls.append(
        ft.ElevatedButton(
            content=ft.Text(" 4 ", color=ft.Colors.WHITE),
            width=100,
            height=50,
            bgcolor=ft.Colors.ERROR,
            on_click=on_button_click
        )
    )

    clear_button = ft.ElevatedButton(
```



```
3  def main(page: ft.Page):
4
5      page.title = "Calculadora TAP"
6      page.window_width = 250      # Ancho estrecho, estilo móvil
7      page.window_height = 400     # Altura definida
8      page.padding = 20           # Espacio interno para que nada toque los bordes
```

3. **Guardar:** Guarda el archivo con el nombre que deseas por ejemplo: calculadora.py (es importante que termine en .py).
4. **Ejecutar:** Desde la terminal, navega a la carpeta donde guardaste el archivo y escribe:

**python3 calculadora.py**

Al hacer esto, se abrirá una ventana nativa en tu Mac con la interfaz de la calculadora.

## Parte 3: Explicación del Código

A continuación, desglosamos el código por secciones funcionales para entender qué hace cada bloque.

### 1. Configuración de la Ventana (main)



```
1  import fastapi as ft
2
3  def main(page: ft.Page):
4
5      page.title = "Calculadora TAP"
6      page.window_width = 250      # Ancho estrecho, estilo móvil
7      page.window_height = 400     # Altura definida
8      page.padding = 20           # Espacio interno para que nada toque los bordes
```

Esta sección define cómo se verá la "caja" de la aplicación al abrirse.

- **Explicación:** Estamos configurando la aplicación para que tenga dimensiones fijas, similares a las de un teléfono celular antiguo o una calculadora de bolsillo.

## 2. La Pantalla de Visualización (display)

Aquí definimos el área donde aparecen los números.

```
10     display = ft.Container(  
11         content=ft.Text("0", size=30), # El texto inicial es "0"  
12         bgcolor=ft.Colors.BLACK12,    # Fondo gris oscuro  
13         border_radius=8,           # Bordes redondeados  
14         alignment=ft.alignment.Alignment(1, 0), # Alinear texto a la derecha  
15         padding=10,               # ... otras configuraciones de tamaño  
16         width=210,  
17         height=70,  
18     )
```

- **Concepto clave:** Usamos un Container (una caja) para darle color de fondo y bordes redondeados al texto.
- **Alineación (1, 0):** En Flet, 0,0 es el centro. 1 en el eje X significa "todo a la derecha". Esto es típico en calculadoras (los números entran por la derecha).

## 3. El "Cerebro" de la Calculadora (Eventos)

Estas son las funciones que deciden qué pasa cuando el usuario toca algo.

### A. Función on\_button\_click (Escribir números)

```
20     def on_button_click(e):  
21  
22         if display.content.value == "0":  
23             # Si hay un 0 solito, lo reemplazamos por el nuevo número  
24             display.content.value = e.control.content.value  
25         else:  
26             # Si ya hay números, agregamos el nuevo al final (concatenar)  
27             display.content.value += e.control.content.value  
28         page.update() # ¡IMPORTANTE! Refrescar la pantalla
```

- **Lógica:** Esta función evita que escribas "01" o "05". Si la pantalla solo muestra el cero inicial, lo borra y pone el número presionado. Si ya tienes un "5" y presionas "2", los une para formar "52".
- **page.update():** Sin esta línea, los datos cambiarían internamente, pero el usuario no vería el cambio en la pantalla.

## B. Función on\_clear\_click (Borrar todo)

```

30     def on_clear_click(e):
31
32         display.content.value = "0" # Reiniciar a 0
33         page.update()
34

```

- **Función:** Restaura la calculadora a su estado inicial.

## 4. La Rejilla de Botones (GridView)

En lugar de colocar botones uno debajo del otro, usamos una rejilla para organizarlos.

```

35     grid = ft.GridView(
36         runs_count=2,    # Define que habrá 2 columnas
37         spacing=10,      # Espacio entre botones
38         run_spacing=10,
39         width=210,
40         height=200,
41         expand=False
42     )

```

- **Estructura:** El código añade 4 botones (1, 2, 3, 4). Como definimos runs\_count=2, se acomodarán así automáticamente:

```
[ 1 ][ 2 ]
[ 3 ][ 4 ]
```

- **Colores:** Cada botón tiene un color distinto (PRIMARY, SECONDARY, etc.) para diferenciarlos visualmente. Todos están conectados a la función on\_button\_click.

## 5. El Botón "Clear"

Este botón está fuera de la rejilla numérica porque ocupa todo el ancho.

```
86     clear_button = ft.ElevatedButton(
87         content=ft.Text("Clear", color=ft.Colors.WHITE),
88         width=210, # Ancho total para coincidir con la rejilla
89         height=50,
90         bgcolor=ft.Colors.RED_700, # Color rojo para indicar "Borrar"
91         on_click=on_clear_click # Conectado a la función de borrar
92     )
```

## 6. El Ensamblaje Final (layout\_principal)

Finalmente, tomamos todas las partes sueltas y las metemos en una columna vertical.

```
94     layout_principal = ft.Column(
95         controls=[
96             display, # 1. La pantalla arriba
97             grid, # 2. Los números en medio
98             clear_button # 3. El botón borrar abajo
99         ],
100        tight=True # Ajustar los elementos para que estén pegaditos
101    )
102
103    page.add(layout_principal) # Agregar todo a la ventana
104    page.update()
105
106 ft.app(target=main)
```

## 7. Resultado del código

Cuando ejecutes el código se debe de ver algo así.

