

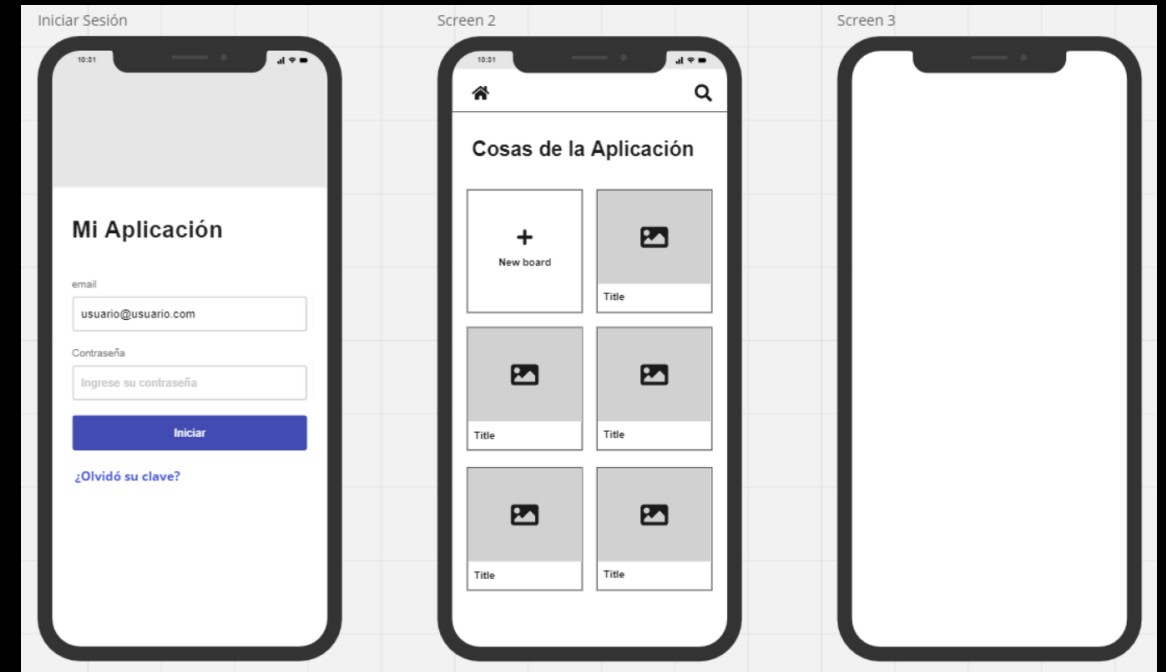
# Introducción al Desarrollo de Aplicaciones en Android

- Ing. Skrauba Axel

# Múltiples Pantallas en *Kivy*

A medida que las aplicaciones se vuelven más complejas, gestionar **varias pantallas** y **transiciones** puede ser todo un desafío.

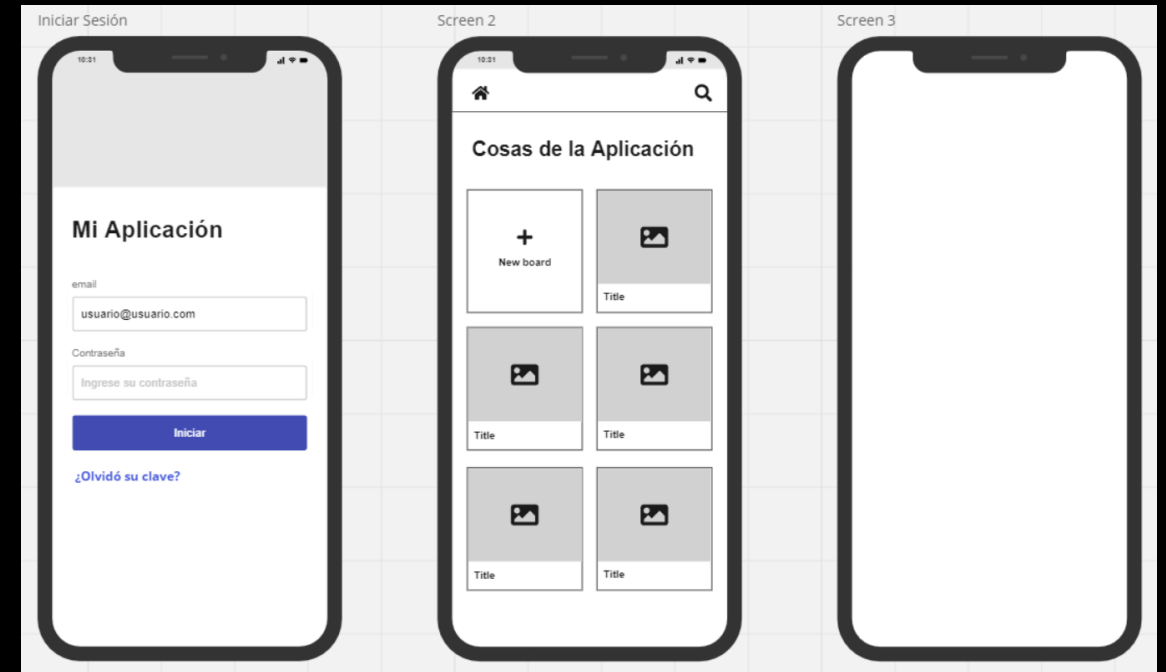
Aquí es donde entra en juego el **Screen Manager** de Kivy



# Screen Manager

Es una herramienta que facilita la creación y gestión de múltiples pantallas dentro de una aplicación.

Permite una navegación suave entre diferentes partes de la aplicación, como la pantalla de inicio, la configuración, el perfil de usuario, entre otras.




# Funcionamiento del *Screen Manager*

## Creación del Screen Manager

En Kivy, primero debemos crear una instancia de **ScreenManager**, que actuará como el administrador de todas nuestras pantallas.

Esto se hace típicamente en la función *build* de la aplicación principal.




```
1  from kivy.app import App
2  from kivy.uix.screenmanager import ScreenManager
3
4  class MyApp(App):
5      def build(self):
6          sm = ScreenManager()
7          # Agregar pantallas al Screen Manager aquí
8          return sm
```

# Funcionamiento del *Screen Manager*

## Definición de Pantallas

Creamos las pantallas que deseamos en nuestra aplicación, normalmente como subclases de **Screen**.

Cada pantalla representa una parte de la interfaz de usuario de la aplicación.



```
1  from kivy.uix.screenmanager import Screen
2
3  class MainScreen(Screen):
4      pass # Contenido de la pantalla principal
5
6  class SettingsScreen(Screen):
7      pass # Contenido de la pantalla de configuración
```

# Funcionamiento del *Screen Manager*

## Agregar Pantallas al Screen Manager

Después de crear las pantallas, las agregamos al **Screen Manager** utilizando el método **add\_widget**.

Esto les permite formar parte de la jerarquía del **Screen Manager**.

NOTA: iría en el **build()** de **App**.



```
1 sm = ScreenManager()
2 sm.add_widget(MainScreen(name='main'))
3 sm.add_widget(SettingsScreen(name='settings'))
```

# Funcionamiento del *Screen Manager*

## Control de la Pantalla Actual

**Screen Manager** realiza un seguimiento de cuál es la pantalla actual mediante la propiedad ***current***.

Para cambiar entre pantallas, simplemente actualizamos el valor de **current** al nombre de la pantalla que deseamos mostrar.



```
1 sm.current = 'main' # Cambiar a la pantalla principal
2 sm.current = 'settings' # Cambiar a la pantalla de configuración
```

# Funcionamiento del *Screen Manager*

## Transiciones entre Pantallas

**Screen Manager** también permite especificar transiciones animadas entre pantallas utilizando la propiedad *transition*.

Por ejemplo, puedes usar *SlideTransition*, *FadeTransition*, o personalizar tus propias transiciones.



```
1 from kivy.uix.screenmanager import SlideTransition
2
3 sm.transition = SlideTransition(direction='left')
```

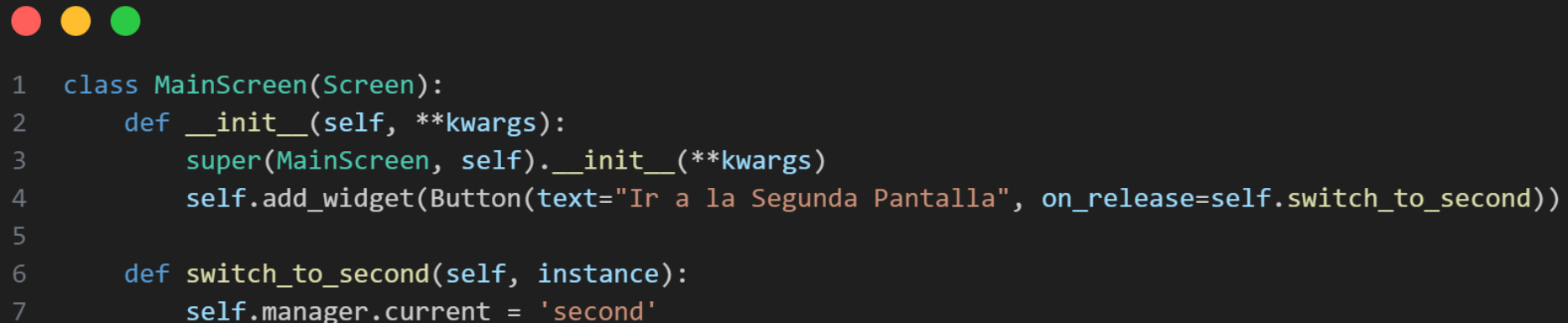


# Funcionamiento del *Screen Manager*

## Interacción entre Pantallas

Cada pantalla dentro del *Screen Manager* puede tener su propia lógica y elementos de interfaz de usuario.

Para interactuar entre pantallas, puedes utilizar métodos o eventos personalizados.



```
1 class MainScreen(Screen):
2     def __init__(self, **kwargs):
3         super(MainScreen, self).__init__(**kwargs)
4         self.add_widget(Button(text="Ir a la Segunda Pantalla", on_release=self.switch_to_second))
5
6     def switch_to_second(self, instance):
7         self.manager.current = 'second'
```

# Modularización



```
1  from kivy.app import App
2  from kivy.uix.screenmanager import ScreenManager, SlideTransition
3  from kivy.lang import Builder
4
5  Builder.load_file('main_screen.kv') # Cargar el archivo kv de la pantalla principal
6  Builder.load_file('second_screen.kv') # Cargar el archivo kv de la segunda pantalla
7  Builder.load_file('third_screen.kv') # Cargar el archivo kv de la tercera pantalla
8
9  # Clases de cada pantalla y su lógica
10 from main_screen import MainScreen
11 from second_screen import SecondScreen
12 from third_screen import ThirdScreen
13
14 class MyApp(App):
15     def build(self):
16         sm = ScreenManager(transition=SlideTransition(direction='left'))
17         sm.add_widget(MainScreen(name='main'))
18         sm.add_widget(SecondScreen(name='second'))
19         sm.add_widget(ThirdScreen(name='third'))
20         return sm
21
22 if __name__ == '__main__':
23     MyApp().run()
```

# Modularización

***Builder*** en *Kivy* es una herramienta que permite cargar definiciones de interfaz de usuario escritas en lenguaje de marcado *Kivy* (KV) desde archivos externos y luego usar esas definiciones en tu aplicación *Kivy*.

El propósito principal de ***Builder*** es separar la lógica de la interfaz de usuario de la lógica de la aplicación, lo que facilita el mantenimiento y la organización del código.

```
tu_proyecto/  
├── main.py  
├── GUI/  
│   ├── main_screen.kv  
│   ├── second_screen.kv  
│   └── third_screen.kv  
├── logica/  
│   ├── main_screen.py  
│   ├── second_screen.py  
│   └── third_screen.py
```

# Preguntas...

- Dudas
- Sugerencias
- Cuestiones
- Vamos al código...

# Referencias

- <https://kivy.org/doc/stable/api-kivy.uix.screenmanager.html>
- <https://kivy.org/doc/stable/api-kivy.lang.builder.html>