

# Shiny para Python

Creación de Dashboards Interactivos con Shiny  
para Python

Eduardo Valero Vilella

Universidad de Murcia

25/04/2022

UNIVERSIDAD DE  
MURCIA

# Índice

- ▶ 1. Introducción
- ▶ 2. Set up del entorno
- ▶ 3. Primeros Pasos
- ▶ 4. Shiny Core vs Express
- ▶ 5. Creación del Dashboard
- ▶ 6. Publicación

## 1. Introducción

## Objetivos del trabajo

UNIVERSIDAD DE  
MURCIA

- ▶ Introducir la librería Shiny para Python
  - ▶ Crear un dashboard interactivo
  - ▶ Hacer un deploy de nuestra aplicación

# 1. Introducción

## Contexto y Motivación

UNIVERSIDAD DE  
MURCIA

¿Qué rol cumple Shiny en el ecosistema de Python?

- ▶ Dashboards y aplicaciones web interactivos
- ▶ Visualizaciones avanzadas
- ▶ Sin código front-end

CSS  
IS  
AWESOME



# 1. Introducción

Shiny

¿Qué es Shiny?

- ▶ Originalmente librería para R
- ▶ Desarrollada por Posit, creadores de RStudio
- ▶ Similar a **Plotly Dash** y **Streamlit**
- ▶ Sintaxis simple
- ▶ Integración con bibliotecas comunes de Python
- ▶ Integración con **Ipywidgets**



plotly



pandas



## 2. Set up del entorno

### Instalación de librerías

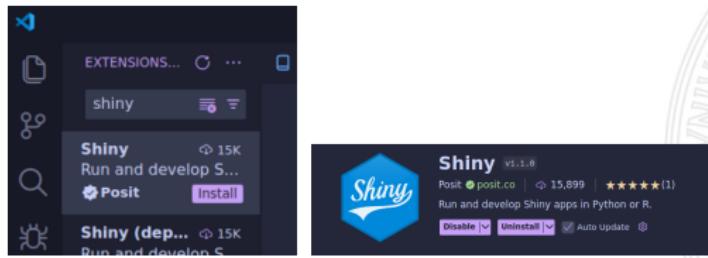
- ▶ Conda: Python 3.11.10
- ▶ Instalación shiny: `pip install shiny`
- ▶ Instalación de otras librerías:
  - ▶ `pip install pandas shinywidgets plotly...`
  - ▶ `pip install -r requirements.txt`
- ▶ Dataset de demostración:  
`pip install palmerpenguins`



## 2. Set up del entorno

### Entorno de trabajo

- ▶ Visual Studio Code
- ▶ Extensión de Shiny para VSCode



# 3. Primeros pasos

## Creando el proyecto

shiny create

- ▶ Plantillas: -template basic-app-plot
- ▶ Modo: -m [express|core]

```
● └[$] < shiny create -t basic-app -m express
... Creating Basic app Shiny app...
? Enter destination directory: .
✓ Created Shiny app at .

→ Next steps:
- Open and edit the app file: app.py
- Run the app with 'shiny run app.py' from the app directory.
```

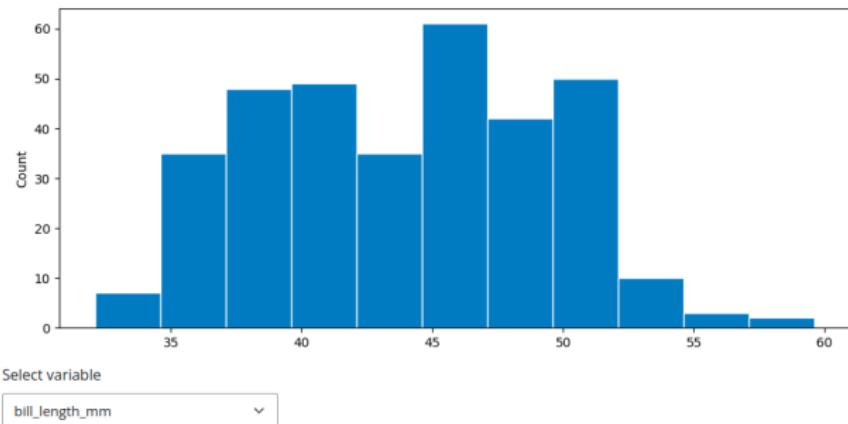
### 3. Primeros pasos

#### Ejecutando la app

shiny run app.py

- ▶ Autoreload: -reload -launch-browser app.py
- ▶ VSCode plug-ing

Basic Shiny app



# 4. Shiny Core vs Express

## Shiny Core

```
1 import seaborn as sns
2
3 # Import data from shared.py
4 from shared import df
5 from shiny import App, render, ui
6
7 # User interface (UI) definition
8 app_ui = ui.page_fixed(
9     # Add a title to the page with some top padding
10    ui.panel_title(ui.h2("Basic Shiny app", class_="pt-5")),
11    # A container for plot output
12    ui.output_plot("hist"),
13    # A select input for choosing the variable to plot
14    ui.input_select(
15        "var", "Select variable", choices=["bill_length_mm", "body_mass_g"]
16    ),
17 )
18
19
20 # Server function provides access to client-side input values
21 def server(input):
22     @render.plot
23     def hist():
24         # Histogram of the selected variable (input.var())
25         p = sns.histplot(df, x=input.var(), facecolor="#007bc2", edgecolor="white")
26         return p.set(xlabel=None)
27
28
29 app = App(app_ui, server)
```

# 4. Shiny Core vs Express

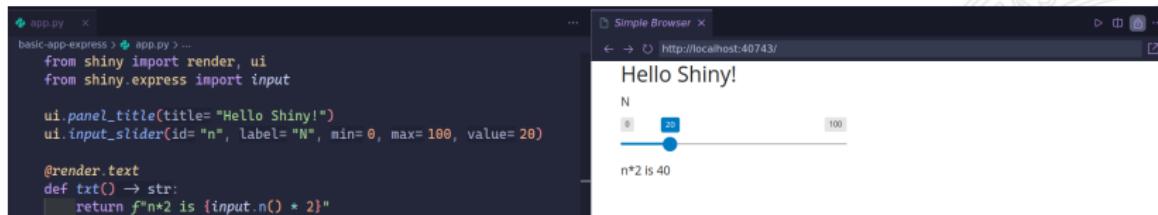
## Shiny Express

```
1 import seaborn as sns
2
3 # Import data from shared.py
4 from shared import df
5 from shiny.express import input, render, ui
6
7 # Page title (with some additional top padding)
8 ui.page_opts(title=ui.h2("Basic Shiny app", class_="pt-5"))
9
10
11 # Render a histogram of the selected variable (input.var())
12 @render.plot
13 def hist():
14     p = sns.histplot(df, x=input.var(), facecolor="#007bc2", edgecolor="white")
15     return p.set(xlabel=None)
16
17
18 # Select input for choosing the variable to plot
19 ui.input_select("var", "Select variable", choices=["bill_length_mm", "body_mass_g"])
```

# 5. Creación del Dashboard

## Plantilla Básica

```
shiny create -t basic-app -m express
```



The terminal window shows the contents of `app.py`:

```
app.py  X
...
basic-app-express > app.py > ...
from shiny import render, ui
from shiny.express import input

ui.panel_title(title="Hello Shiny!")
ui.input_slider(id="n", label="N", min=0, max=100, value=20)

@render.text
def txt() -> str:
    return f"n*2 is {input.n() * 2}"
```

The browser window titled "Simple Browser" displays the application at `http://localhost:40743/`. The page title is "Hello Shiny!". It features a slider labeled "N" with a value of 20, ranging from 0 to 100. Below the slider, the text "n\*2 is 40" is displayed.

# 5. Creación del Dashboard

## Datos Reactivos

- ▶ Carga reactiva de datos con `shiny.reactive`
- ▶ Dataset: Palmer Penguins
- ▶ Añadimos datos para la localización

```
@reactive.calc  
def datos() -> pd.DataFrame
```



# 5. Creación del Dashboard

## Sintaxis básica

- ▶ Contenedores usando with
- ▶ Elementos de la interfaz en shiny.ui
- ▶ Configuración de la página ui.page\_opts()
- ▶ Markdown

```
with ui.contenedor:  
    Contenido
```

```
with ui.card():  
    ui.card_header()  
    ui.markdown()
```

# 5. Creación del Dashboard

## Sintaxis básica

- ▶ Creamos la carpeta static para archivos estáticos
- ▶ Usamos los tipos de Shiny
- ▶ Decoradores para renderizar gráficos

```
from shiny.types import ImgData
@render.image
def penguins_image -> ImgData
```

# 5. Creación del Dashboard

## Sintaxis básica

### Dashboard de Pingüinos

#### Descripción de datos

##### 1. Datos generales

- Especie: *Adelie*, *Gentoo* o *Chinstrap*
- Peso (gramos)
- Año de nacimiento

##### 2. Descripción del pico

- Longitud (mm)
- Grosor vertical (mm)

##### 3. Descripción de las aletas:

- Longitud.

##### 4. Localización:

- Isla
- Coordenadas



# 5. Creación del Dashboard

## Layouts: Columnas

- ▶ Anidar de contenedores
- ▶ layout\_columns: espacio total de 12 columnas

```
with ui.contenedor_1:  
    with ui.contenedor_2:  
        with ui.contenedor_3:  
            Contenido A  
  
    with ui.contenedor_4:  
        Contenido B
```

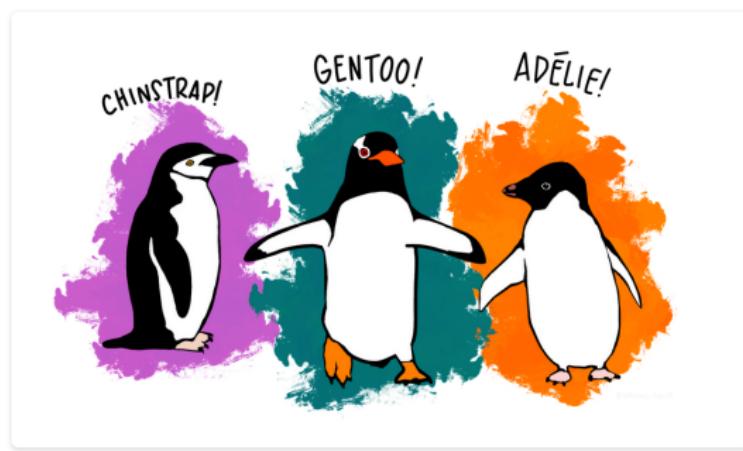
# 5. Creación del Dashboard

## Layouts: Columnas

### Dashboard de Pingüinos

#### Descripción de datos

1. Datos generales
  - Especie: *Adelie*, *Gentoo* o *Chinstrap*
  - Peso (gramos)
  - Año de nacimiento
2. Descripción del pico
  - Longitud (mm)
  - Grosor vertical (mm)
3. Descripción de las aletas:
  - Longitud.
4. Localización:
  - Isla
  - Coordenadas



# 5. Creación del Dashboard

## ValueBox

- ▶ Visualizar datos importantes
- ▶ *Showcase*: imágenes o iconos
- ▶ `render.ui` para HTML general

```
with ui.value_box(showcase):  
    "Título"  
    @render.ui  
    def numero_pinguinos() -> int:
```



# 5. Creación del Dashboard

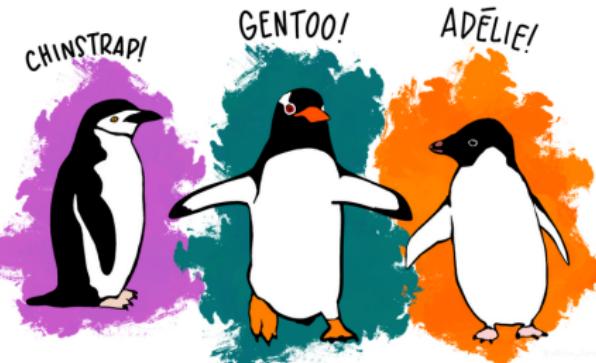
## ValueBox

UNIVERSIDAD DE  
MURCIA

### Dashboard de Pingüinos

#### Descripción de datos

1. Datos generales
  - Especie: *Adelie*, *Gentoo* o *Chinstrap*
  - Peso (gramos)
  - Año de nacimiento
2. Descripción del pico
  - Longitud (mm)
  - Grosor vertical (mm)
3. Descripción de las aletas:
  - Longitud.
4. Localización:
  - Isla
  - Coordenadas



Número de pingüinos

344



Tamaño medio de alas

201 mm



Peso máximo

4.202 g

# 5. Creación del Dashboard

## Visualizaciones dinámicas y reactivas

- ▶ Card con sidebar para filtros
- ▶ Cards en pantalla completa
- ▶ Cada `ui.input_*` tiene un identificador
- ▶ Extraemos el valor con `shiny.express.input`

```
ui.input_select(id = "var")... @render.plot  
def histograma() -> Axes:  
    valor = input.var()
```

# 5. Creación del Dashboard

## Visualizaciones dinámicas y reactivas



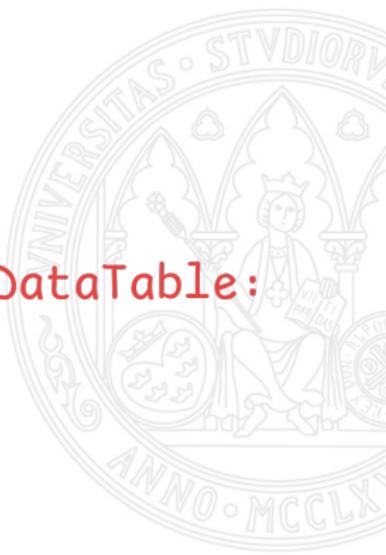
# 5. Creación del Dashboard

## Tablas de Datos

- ▶ Visualización de nuestros datos en forma de tabla
- ▶ DataGrid o DataTable
- ▶ Filtros de datos

```
@render.data_frame
```

```
def tabla_datos() -> DataGrid | DataTable:  
    return render.DataGrid(df)
```



# 5. Creación del Dashboard

## Tablas de Datos

- o Isla
- o Coordenadas



Número de pingüinos  
344

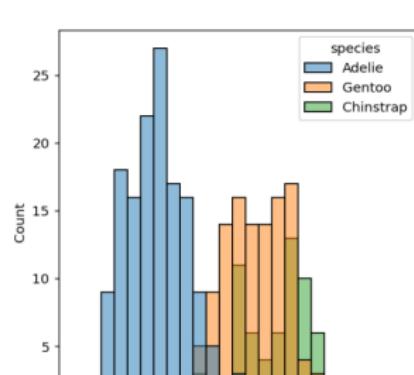


Tamaño medio de alas  
201 mm



Peso máximo  
4.202 g

Histograma



species	island	bill_length_mm	bill_depth_mm	flipper_length_mm
Adelie	Torgersen	39.1	18.7	181
Adelie	Torgersen	39.5	17.4	186
Adelie	Torgersen	40.3	18	195
Adelie	Torgersen			
Adelie	Torgersen	36.7	19.3	193
Adelie	Torgersen	39.3	20.6	190
Adelie	Torgersen	38.9	17.8	181
Adelie	Torgersen	39.2	19.6	195
Adelie	Torgersen	34.1	18.1	193
Adelie	Torgersen	42	20.2	190
Adelie	Torgersen	37.8	17.1	186

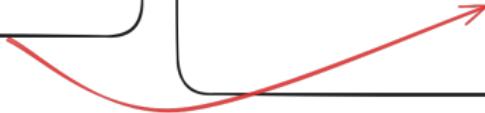
# 5. Creación del Dashboard

## Organización mediante Tabs

- ▶ Muchos datos en la misma página
- ▶ Creamos tabs separados
- ▶ Cambiamos el `panel_title` por `page_opts`

```
with ui.contenedor:  
    Contenido
```

```
with ui.nav_panel("título"):
```



# 5. Creación del Dashboard

## Organización mediante Tabs

Dashboard de Pingüinos   Descripción del Conjunto de Datos   Visualización general



Número de pingüinos  
344

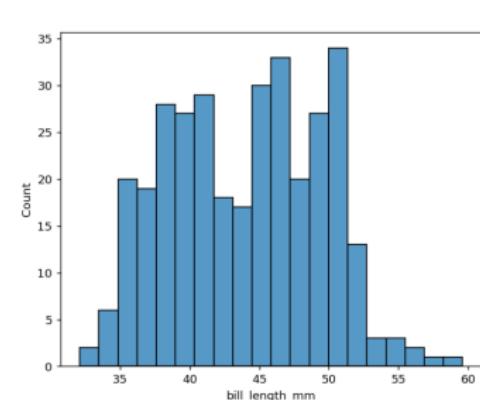


Tamaño medio de alas  
201 mm



Peso máximo  
4.202 g

Histograma



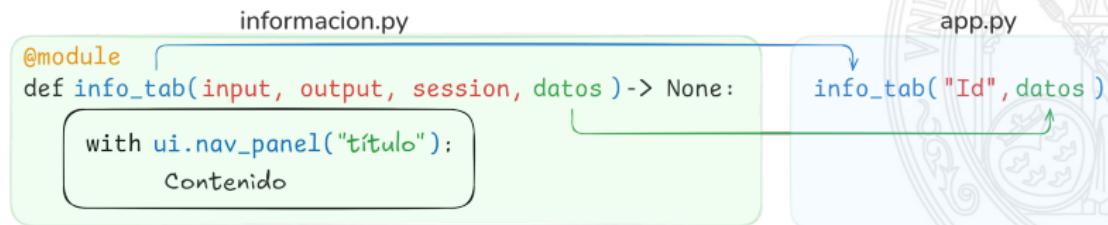
species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
Adelie	Torgersen	39.1	18.7	181	3750
Adelie	Torgersen	39.5	17.4	186	3800
Adelie	Torgersen	40.3	18	195	3250
Adelie	Torgersen				
Adelie	Torgersen	36.7	19.3	193	3450
Adelie	Torgersen	39.3	20.6	190	3650
Adelie	Torgersen	38.9	17.8	181	3625
Adelie	Torgersen	39.2	19.6	195	4675
Adelie	Torgersen	34.1	18.1	193	3475
Adelie	Torgersen	42	20.2	190	4250
Adelie	Torgersen	37.8	17.1	186	3300
Adelie	Torgersen	37.8	17.3	180	3700
Adelie	Torgersen	41.1	17.6	182	3200
Adelie	Torgersen	38.6	21.2	191	3800

Viewing rows 1 through 14 of 344

# 5. Creación del Dashboard

## Modularización

- Modularizamos el código
- Cada panel a un archivo
- shiny.express.module
- input, output y session



# 5. Creación del Dashboard

## Mapas interactivos

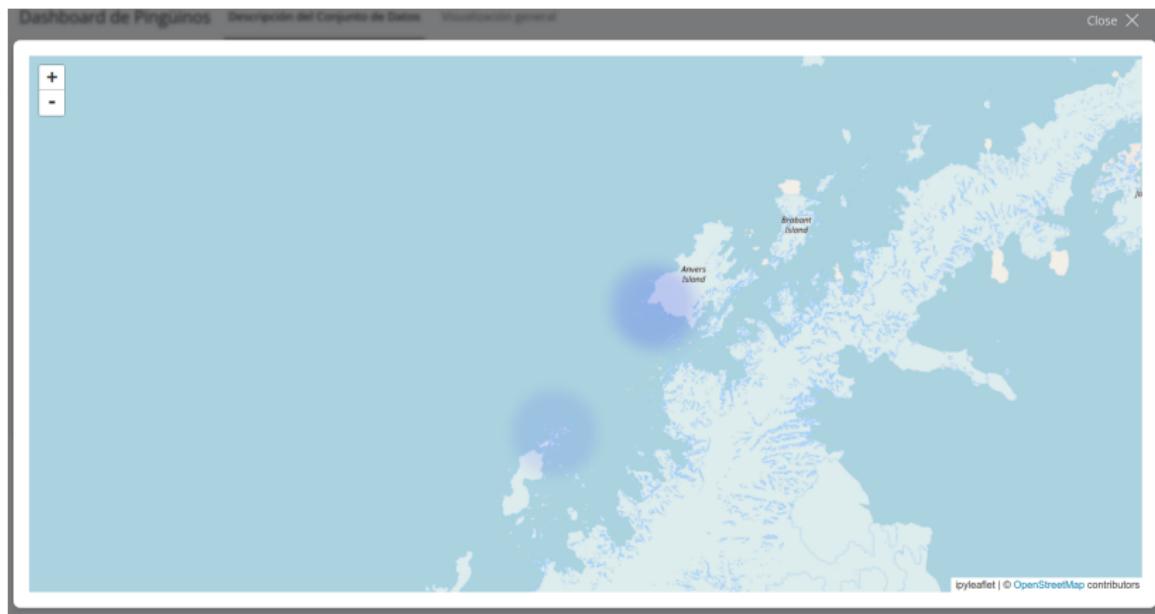
- ▶ ipyleaflet para crear mapas interactivos
- ▶ shiny\_widgets con visualizaciones interactivas
- ▶ Para uso general usamos render\_widget

```
@render_widget  
def leaflet_map() -> Map:
```

## 5. Creación del Dashboard

UNIVERSIDAD DE  
MURCIA

## Mapas interactivos



# 5. Creación del Dashboard

## Visualizaciones dinámicas e interactivas

- ▶ Algunas de las librerías más conocidas tienen su propio motor de render
- ▶ Más rápidos y eficientes



```
@render_plotly  
def plotly_plot() -> Figure:
```

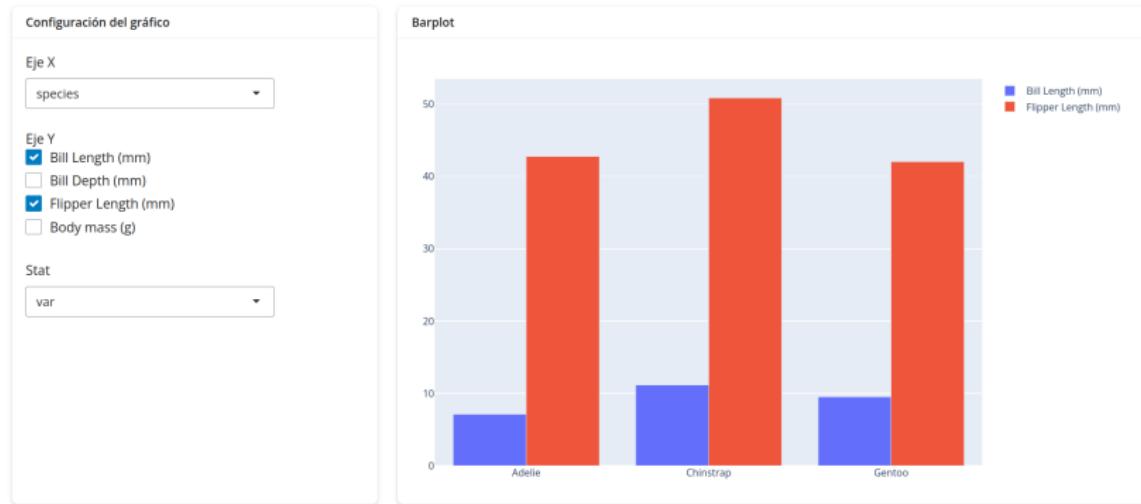
# 5. Creación del Dashboard

## Visualizaciones dinámicas e interactivas



# 5. Creación del Dashboard

## Visualizaciones dinámicas e interactivas



# 6. Publicación

## ShinyLive

UNIVERSIDAD DE  
MURCIA

- ▶ Shiny app como web estática
- ▶ Muy flexible
- ▶ Limitaciones:
  - ▶ Solo paquetes compatibles con Pyodide
  - ▶ Seguridad



# 6. Publicación

## Servidores Privados

- ▶ Posit Connect: automático, github
- ▶ Shiny Server: open-source, configuración compleja
- ▶ Servidores cloud comunes



Google Cloud



# 6. Publicación

## Servidores en la nube

- ▶ Método más rápido
- ▶ Hugging Face Spaces
- ▶ Shinyapps.io
  - ▶ Registro gratuito en shinyapps.io
  - ▶ Paquete: `pip install rsconnect-python`
  - ▶ Configuración token: `rsconnect add`
  - ▶ Deploy: `rsconnect deploy shiny`



# 6. Publicación

## Links

- ▶ Dashboard Publicado  
[https://valevil.shinyapps.io/tad\\_shiny/](https://valevil.shinyapps.io/tad_shiny/)
- ▶ Repositorio  
[https://github.com/valevil27/TAD\\_Shiny](https://github.com/valevil27/TAD_Shiny)