https://www.gradio.app/docs/gradio/blocks

Blocks

Blocks - это низкоуровневый API Gradius, который позволяет создавать больше пользовательских веб-приложений и демонстраций, чем интерфейсов (но все еще полностью на Python).

По сравнению с классом интерфейса, Blocks обеспечивает **большую гибкость** и контроль над:

(1) **расположением компонентов**,

(2) **событиями**, которые запускают выполнение функций,

(3) **потоками данных** (например, входы могут инициировать выходы, которые могут инициировать следующий уровень выходов).

Blocks также предлагает **способы группировки** связанных демонстраций, например, с помощью вкладок.

Основное использование Blocks заключается в следующем:

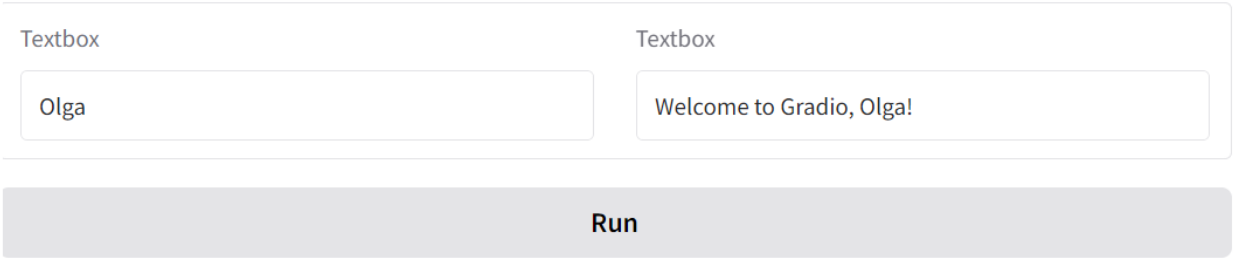
- ✓ создайте объект Blocks,
- ✓ затем используйте его в качестве контекста (с помощью инструкции "with"),
- ✓ затем определите макеты, компоненты или события в контексте Blocks.
- ✓ Наконец, вызовите метод launch() для запуска демонстрации.

https://www.gradio.app/docs/gradio/blocks

```
pip install gradio
```

Пример 1

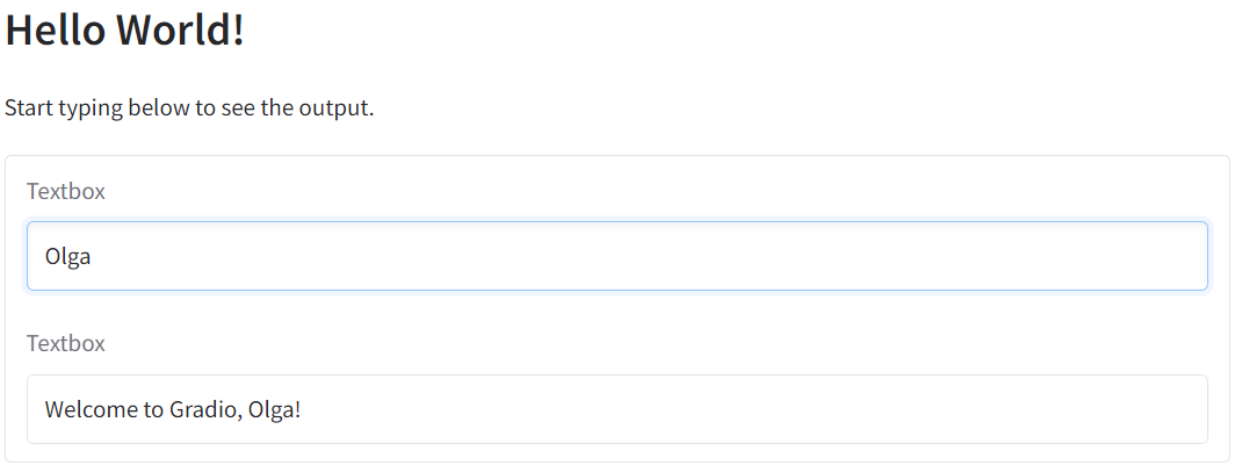
```
import gradio as gr

def update(name):
    return f"Welcome to Gradio, {name}!"

with gr.Blocks() as demo:
    gr.Markdown("Start typing below and then click
**Run** to see the output.")

    with gr.Row():
        inp = gr.Textbox(placeholder="What is your
name?")

        out = gr.Textbox()

    btn = gr.Button("Run")

    btn.click(fn=update, inputs=inp, outputs=out)

demo.launch()
```

https://www.gradio.app/docs/gradio/blocks

Пример 2

```
import gradio as gr


def welcome(name):
    return f"Welcome to Gradio, {name}!"


with gr.Blocks() as demo:
    gr.Markdown(
    """
    # Hello World!
    Start typing below to see the output.
    """)
    inp = gr.Textbox(placeholder="What is your name?")
    out = gr.Textbox()
    inp.change(welcome, inp, out)

if __name__ == "__main__":
    demo.launch()
```

https://www.gradio.app/docs/gradio/blocks

Пример 3

```
import numpy as np
import gradio as gr

def flip_text(x):
    return x[::-1]

def flip_image(x):
    return np.fliplr(x)

with gr.Blocks() as demo:
    gr.Markdown("Flip text or image files using this
demo.")
    with gr.Tab("Flip Text"):
        text_input = gr.Textbox()
        text_output = gr.Textbox()
        text_button = gr.Button("Flip")
    with gr.Tab("Flip Image"):
        with gr.Row():
            image_input = gr.Image()
            image_output = gr.Image()
        image_button = gr.Button("Flip")

    with gr.Accordion("Open for More!", open=False):
        gr.Markdown("Look at me...")
        temp_slider = gr.Slider(
            0, 1,
            value=0.1,
            step=0.1,
            interactive=True,
            label="Slide me",
        )

    text_button.click(flip_text, inputs=text_input,
outputs=text_output)
    image_button.click(flip_image, inputs=image_input,
outputs=image_output)

if __name__ == "__main__":
    demo.launch()
```

https://www.gradio.app/docs/gradio/blocks

Flip text or image files using this demo.

Flip Text | Flip Image

Textbox

А РОЗА УПАЛА НА ЛАПУ АЗОРА

Textbox

АРОЗА УПАЛ АН АЛАПУ АЗОР А

Flip

Flip text or image files using this demo.

Flip Text | Flip Image

Image

Image

Flip

Open for More!

Look at me...

Slide me

0,1

0 1

https://www.gradio.app/docs/gradio/blocks

Methods

gradio.Blocks.launch(···)

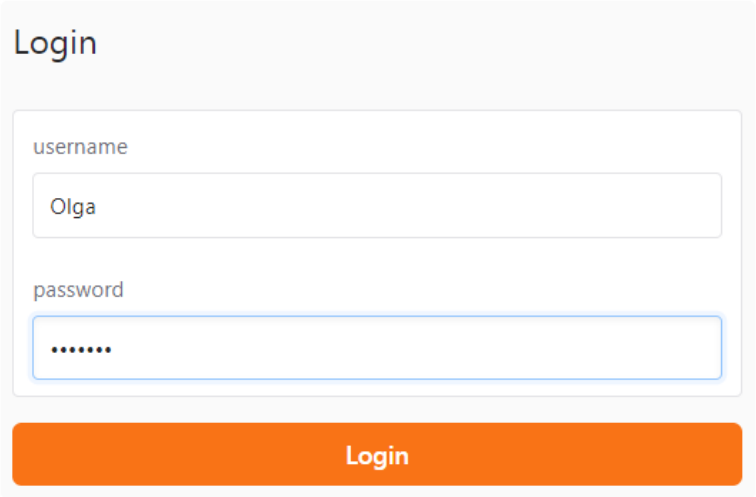
**Описание launch **

Запускает простой веб-сервер, который обслуживает демонстрацию.

Также может использоваться для создания общедоступной ссылки, по которой любой пользователь может получить доступ к демонстрации из своего браузера, установив значение **share=True**.

Пример 4

```
import gradio as gr

def reverse(text):
    return text[::-1]

with gr.Blocks() as demo:
    button = gr.Button(value="Reverse")
    button.click(reverse, gr.Textbox(), gr.Textbox())

demo.launch(share=True, auth=("username", "password"))
```



https://www.gradio.app/docs/gradio/blocks

gradio.Blocks.integrate(···)

Описание **integrate **

Универсальный **метод для интеграции с другими библиотеками**.

Этот метод следует запускать после launch().

gradio.Blocks.load(block, ···)

Описание **load **

Этот прослушиватель запускается, когда блоки изначально загружаются в браузер.

gradio.Blocks.unload(fn, ···)

Описание **unload **

Этот прослушиватель запускается, когда пользователь закрывает или обновляет вкладку, завершая сеанс пользователя.

Он **полезен для очистки ресурсов при закрытии приложения**.


```
import gradio as gr

with gr.Blocks() as demo:

    gr.Markdown("# When you close the tab, hello will
be printed to the console")
    demo.unload(lambda: print("hello"))

demo.launch()
```

When you close the tab, hello will be printed to the console

gradio.Blocks.queue(···)

Описание **queue **

Включив функцию очереди, вы можете контролировать, когда пользователи будут знать свое положение в очереди, и устанавливать ограничение на максимальное количество разрешенных событий.