Blocks

```
with gradio.Blocks():
```

Description

Blocks is Gradio's low-level API that allows you to create more custom web applications and demos than Interfaces (yet still entirely in Python).

Compared to the Interface class, Blocks offers more flexibility and control over: (1) the layout of components (2) the events that trigger the execution of functions (3) data flows (e.g. inputs can trigger outputs, which can trigger the next level of outputs). Blocks also offers ways to group together related demos such as with tabs.

The basic usage of Blocks is as follows: create a Blocks object, then use it as a context (with the "with" statement), and then define layouts, components, or events within the Blocks context. Finally, call the launch() method to launch the demo.

Example Usage

```
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)
```

).launch()

Initialization

Parameter	Description
`theme` *Theme | str | None* **default: None**	A Theme object or a string representing a theme. If a string, will look for a built-in theme with that name (e.g. "soft" or "default"), or will attempt to load a theme from the HF Hub (e.g. "gradio/monochrome"). If None, will use the Default theme.
`analytics_enabled` *bool | None* **default: None**	Whether to allow basic telemetry. If None, will use GRADIO_ANALYTICS_ENABLED environment variable or default to True.
`mode` *str* **default: "blocks"**	A human-friendly name for the kind of Blocks or Interface being created. Used internally for analytics.
`title` *str* **default: "Gradio"**	The tab title to display when this is opened in a browser window.
`css` *str | None* **default: None**	Custom css as a string or path to a css file. This css will be included in the demo webpage.
`js` *str | None* **default: None**	Custom js or path to js file to run when demo is first loaded. This javascript will be included in the demo webpage.
`head` *str | None* **default: None**	Custom html to insert into the head of the demo webpage. This can be used to add custom meta tags, scripts, stylesheets, etc. to the page.

Demos

```
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.
    """)
```

Methods

launch

```
gradio.Blocks.launch(...)
```

Description

Launches a simple web server that serves the demo. Can also be used to create a public link used by anyone to access the demo from their browser by setting share=True.

Example Usage

```
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"))
```

Agruments

Parameter	Description

Parameter	Description
`inline` *bool* | *None* **default: None**	whether to display in the interface inline in an iframe. Defaults to True in python notebooks; False otherwise.
`inbrowser` *bool* **default: False**	whether to automatically launch the interface in a new tab on the default browser.
`share` *bool* | *None* **default: None**	whether to create a publicly shareable link for the interface. Creates an SSH tunnel to make your UI accessible from anywhere. If not provided, it is set to False by default every time, except when running in Google Colab. When localhost is not accessible (e.g. Google Colab), setting share=False is not supported.
`debug` *bool* **default: False**	if True, blocks the main thread from running. If running in Google Colab, this is needed to print the errors in the cell output.
`max_threads` *int* **default: 40**	the maximum number of total threads that the Gradio app can generate in parallel. The default is inherited from the starlette library (currently 40).
`auth` *Callable* | *tuple[str, str]* | *list[tuple[str, str]]* | *None* **default: None**	If provided, username and password (or list of username-password tuples) required to access interface. Can also provide function that takes username and password and returns True if valid login.
`auth_message` *str* | *None* **default: None**	If provided, HTML message provided on login page.

Parameter	Description
`prevent_thread_lock` *bool* **default: False**	If True, the interface will block the main thread while the server is running.
`show_error` *bool* **default: False**	If True, any errors in the interface will be displayed in an alert modal and printed in the browser console log
`server_name` *str | None* **default: None**	to make app accessible on local network, set this to "0.0.0.0". Can be set by environment variable GRADIO_SERVER_NAME. If None, will use "127.0.0.1".
`server_port` *int | None* **default: None**	will start gradio app on this port (if available). Can be set by environment variable GRADIO_SERVER_PORT. If None, will search for an available port starting at 7860.
`height` *int* **default: 500**	The height in pixels of the iframe element containing the interface (used if inline=True)
`width` *int | str* **default: "100%"**	The width in pixels of the iframe element containing the interface (used if inline=True)
`favicon_path` *str | None* **default: None**	If a path to a file (.png, .gif, or .ico) is provided, it will be used as the favicon for the web page.
`ssl_keyfile` *str | None* **default: None**	If a path to a file is provided, will use this as the private key file to create a local server running on https.

Parameter	Description
`ssl_certfile` *str | None* **default: None**	If a path to a file is provided, will use this as the signed certificate for https. Needs to be provided if ssl_keyfile is provided.
`ssl_keyfile_password` *str | None* **default: None**	If a password is provided, will use this with the ssl certificate for https.
`ssl_verify` *bool* **default: True**	If False, skips certificate validation which allows self-signed certificates to be used.
`quiet` *bool* **default: False**	If True, suppresses most print statements.
`show_api` *bool* **default: True**	If True, shows the api docs in the footer of the app. Default True.
`allowed_paths` *list[str] | None* **default: None**	List of complete filepaths or parent directories that gradio is allowed to serve (in addition to the directory containing the gradio python file). Must be absolute paths. Warning: if you provide directories, any files in these directories or their subdirectories are accessible to all users of your app.
`blocked_paths` *list[str] | None* **default: None**	List of complete filepaths or parent directories that gradio is not allowed to serve (i.e. users of your app are not allowed to access). Must be absolute paths. Warning: takes precedence over `allowed_paths` and all other directories exposed by Gradio by default.

Parameter	Description
`root_path` *str | None* **default: None**	The root path (or "mount point") of the application, if it's not served from the root ("/") of the domain. Often used when the application is behind a reverse proxy that forwards requests to the application. For example, if the application is served at "https://example.com/myapp", the `root_path` should be set to "/myapp". Can be set by environment variable GRADIO_ROOT_PATH. Defaults to "".
`app_kwargs` *dict[str, Any] | None* **default: None**	Additional keyword arguments to pass to the underlying FastAPI app as a dictionary of parameter keys and argument values. For example, `"docs_url": "/docs"`
`state_session_capacity` *int* **default: 10000**	The maximum number of sessions whose information to store in memory. If the number of sessions exceeds this number, the oldest sessions will be removed. Reduce capacity to reduce memory usage when using gradio.State or returning updated components from functions. Defaults to 10000.
`share_server_address` *str | None* **default: None**	Use this to specify a custom FRP server and port for sharing Gradio apps (only applies if share=True). If not provided, will use the default FRP server at https://gradio.live. See https://github.com/huggingface/frp for more information.
`share_server_protocol` *Literal[('http', 'https')] | None* **default: None**	Use this to specify the protocol to use for the share links. Defaults to "https", unless a custom share_server_address is provided, in which case it defaults to "http". If you are using a custom share_server_address and want to use https, you must set this to "https".

queue

`gradio.Blocks.queue(···)`

Description

By enabling the queue you can control when users know their position in the queue, and set a limit on maximum number of events allowed.

Example Usage

```
with gr.Blocks() as demo:
    button = gr.Button(label="Generate Image")
    button.click(fn=image_generator, inputs=gr.Textbox(), outputs=gr.Image())
demo.queue(max_size=10)
demo.launch()
```

Agruments

Parameter	Description
`status_update_rate` *float* | *Literal['auto']* **default: "auto"**	If "auto", Queue will send status estimations to all clients whenever a job is finished. Otherwise Queue will send status at regular intervals set by this parameter as the number of seconds.
`api_open` *bool* | *None* **default: None**	If True, the REST routes of the backend will be open, allowing requests made directly to those endpoints to skip the queue.
`max_size` *int* | *None* **default: None**	The maximum number of events the queue will store at any given moment. If the queue is full, new events will not be added and a user will receive a message saying that the queue is full. If None, the queue size will be unlimited.
`concurrency_count` *int* | *None* **default: None**	Deprecated. Set the concurrency_limit directly on event listeners e.g. btn.click(fn, ..., concurrency_limit=10) or gr.Interface(concurrency_limit=10). If necessary, the total number of workers can be configured via `max_threads` in launch().

Parameter	Description
`default_concurrency_limit` *int	None	Literal['not_set']* **default: "not_set"**	The default value of `concurrency_limit` to use for event listeners that don't specify a value. Can be set by environment variable GRADIO_DEFAULT_CONCURRENCY_LIMIT. Defaults to 1 if not set otherwise.

integrate

`gradio.Blocks.integrate(···)`

Description

A catch-all method for integrating with other libraries. This method should be run after launch()

Agruments

Parameter	Description
`comet_ml` *<class 'inspect._empty'>* **default: None**	If a comet_ml Experiment object is provided, will integrate with the experiment and appear on Comet dashboard
`wandb` *ModuleType	None* **default: None**	If the wandb module is provided, will integrate with it and appear on WandB dashboard
`mlflow` *ModuleType	None* **default: None**	If the mlflow module is provided, will integrate with the experiment and appear on ML Flow dashboard

load

`gradio.Blocks.load(block, ···)`

Description

This listener is triggered when the Blocks initially loads in the browser.

Agruments

Parameter	Description
`block` *Block* | *None* ***required***	
`fn` *Callable* | *None* | *Literal['decorator']* **default: "decorator"**	the function to call when this event is triggered. Often a machine learning model's prediction function. Each parameter of the function corresponds to one input component, and the function should return a single value or a tuple of values, with each element in the tuple corresponding to one output component.
`inputs` *Component* | *list[Component]* | *set[Component]* | *None* **default: None**	List of gradio.components to use as inputs. If the function takes no inputs, this should be an empty list.
`outputs` *Component* | *list[Component]* | *None* **default: None**	List of gradio.components to use as outputs. If the function returns no outputs, this should be an empty list.
`api_name` *str* | *None* | *Literal[False]* **default: None**	defines how the endpoint appears in the API docs. Can be a string, None, or False. If set to a string, the endpoint will be exposed in the API docs with the given name. If None (default), the name of the function will be used as the API endpoint. If False, the endpoint will not be exposed in the API docs and downstream apps (including those that `gr.load` this app) will not be able to use this event.

Parameter	Description
`scroll_to_output` *bool* **default: False**	If True, will scroll to output component on completion
`show_progress` *Literal[('full', 'minimal', 'hidden')]* **default: "full"**	If True, will show progress animation while pending
`queue` *bool | None* **default: None**	If True, will place the request on the queue, if the queue has been enabled. If False, will not put this event on the queue, even if the queue has been enabled. If None, will use the queue setting of the gradio app.
`batch` *bool* **default: False**	If True, then the function should process a batch of inputs, meaning that it should accept a list of input values for each parameter. The lists should be of equal length (and be up to length `max_batch_size`). The function is then ***required*** to return a tuple of lists (even if there is only 1 output component), with each list in the tuple corresponding to one output component.
`max_batch_size` *int* **default: 4**	Maximum number of inputs to batch together if this is called from the queue (only relevant if batch=True)
`preprocess` *bool* **default: True**	If False, will not run preprocessing of component data before running 'fn' (e.g. leaving it as a base64 string if this method is called with the `Image` component).
`postprocess` *bool* **default: True**	If False, will not run postprocessing of component data before returning 'fn' output to the browser.

Parameter	Description
`cancels` *dict[str, Any] | list[dict[str, Any]] | None* **default: None**	A list of other events to cancel when this listener is triggered. For example, setting cancels=[click_event] will cancel the click_event, where click_event is the return value of another components .click method. Functions that have not yet run (or generators that are iterating) will be cancelled, but functions that are currently running will be allowed to finish.
`every` *float | None* **default: None**	Run this event 'every' number of seconds while the client connection is open. Interpreted in seconds. Queue must be enabled.
`trigger_mode` *Literal[('once', 'multiple', 'always_last')] | None* **default: None**	If "once" (default for all events except `.change()`) would not allow any submissions while an event is pending. If set to "multiple", unlimited submissions are allowed while pending, and "always_last" (default for `.change()` event) would allow a second submission after the pending event is complete.
`js` *str | None* **default: None**	Optional frontend js method to run before running 'fn'. Input arguments for js method are values of 'inputs' and 'outputs', return should be a list of values for output components.
`concurrency_limit` *int | None | Literal['default']* **default: "default"**	If set, this is the maximum number of this event that can be running simultaneously. Can be set to None to mean no concurrency_limit (any number of this event can be running simultaneously). Set to "default" to use the default concurrency limit (defined by the `default_concurrency_limit` parameter in `Blocks.queue()`, which itself is 1 by default).
`concurrency_id` *str | None* **default: None**	If set, this is the id of the concurrency group. Events with the same concurrency_id will be limited by the lowest set concurrency_limit.

Parameter	Description
`show_api` *bool* **default: True**	whether to show this event in the "view API" page of the Gradio app, or in the ".view_api()" method of the Gradio clients. Unlike setting api_name to False, setting show_api to False will still allow downstream apps to use this event. If fn is None, show_api will automatically be set to False.

Guides

- Blocks And Event Listeners
- Controlling Layout
- State In Blocks
- Custom CSS And JS
- Using Blocks Like Functions

← TabbedInterface

Row →