ChatInterface

`gradio.ChatInterface(fn, ···)`

Description

ChatInterface is Gradio's high-level abstraction for creating chatbot UIs, and allows you to create a web-based demo around a chatbot model in a few lines of code. Only one parameter is required: fn, which takes a function that governs the response of the chatbot based on the user input and chat history. Additional parameters can be used to control the appearance and behavior of the demo.

Example Usage

```python
import gradio as gr

def echo(message, history):
    return message

demo = gr.ChatInterface(fn=echo, examples=["hello", "hola", "merhaba"], title="Echo Bot")
demo.launch()
```

Initialization

Parameter	Description
`fn` *Callable* ***required***	The function to wrap the chat interface around. Should accept two parameters: a string input message and list of two-element lists of the form [[user_message, bot_message], ...] representing the chat history, and return a string response. See the Chatbot documentation for more information on the chat history format.

ımeter	Description
`chatbot` *Chatbot | None* **default: None**	An instance of the gr.Chatbot component to use for the chat interface, if you would like to customize the chatbot properties. If not provided, a default gr.Chatbot component will be created.
`textbox` *Textbox | None* **default: None**	An instance of the gr.Textbox component to use for the chat interface, if you would like to customize the textbox properties. If not provided, a default gr.Textbox component will be created.
`additional_inputs` *str | Component | list[str | Component] | None* **default: None**	An instance or list of instances of gradio components (or their string shortcuts) to use as additional inputs to the chatbot. If components are not already rendered in a surrounding Blocks, then the components will be displayed under the chatbot, in an accordion.
`additional_inputs_accordion_name` *str | None* **default: None**	Deprecated. Will be removed in a future version of Gradio. Use the `additional_inputs_accordion` parameter instead.
`additional_inputs_accordion` *str | Accordion | None* **default: None**	If a string is provided, this is the label of the `gr.Accordion` to use to contain additional inputs. A `gr.Accordion` object can be provided as well to configure other properties of the container holding the additional inputs. Defaults to a `gr.Accordion(label="Additional Inputs", open=False)`. This parameter is only used if `additional_inputs` is provided.
`examples` *list[str] | None* **default: None**	Sample inputs for the function; if provided, appear below the chatbot and can be clicked to populate the chatbot input.
`cache_examples` *bool | None* **default: None**	If True, caches examples in the server for fast runtime in examples. The default option in HuggingFace Spaces is True. The default option elsewhere is False.

meter	Description
`title` *str | None* **default: None**	a title for the interface; if provided, appears above chatbot in large font. Also used as the tab title when opened in a browser window.
`description` *str | None* **default: None**	a description for the interface; if provided, appears above the chatbot and beneath the title in regular font. Accepts Markdown and HTML content.
`theme` *Theme | str | None* **default: None**	Theme to use, loaded from gradio.themes.
`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.
`analytics_enabled` *bool | None* **default: None**	Whether to allow basic telemetry. If None, will use GRADIO_ANALYTICS_ENABLED environment variable if defined, or default to True.
`submit_btn` *str | None | Button* **default: "Submit"**	Text to display on the submit button. If None, no button will be displayed. If a Button object, that button will be used.

meter	Description
`stop_btn` *str | None | Button* **default: "Stop"**	Text to display on the stop button, which replaces the submit_btn when the submit_btn or retry_btn is clicked and response is streaming. Clicking on the stop_btn will halt the chatbot response. If set to None, stop button functionality does not appear in the chatbot. If a Button object, that button will be used as the stop button.
`retry_btn` *str | None | Button* **default: "🔄 Retry"**	Text to display on the retry button. If None, no button will be displayed. If a Button object, that button will be used.
`undo_btn` *str | None | Button* **default: "↩️ Undo"**	Text to display on the delete last button. If None, no button will be displayed. If a Button object, that button will be used.
`clear_btn` *str | None | Button* **default: "🗑️ Clear"**	Text to display on the clear button. If None, no button will be displayed. If a Button object, that button will be used.
`autofocus` *bool* **default: True**	If True, autofocuses to the textbox when the page loads.
`concurrency_limit` *int | None | Literal['default']* **default: "default"**	If set, this is the maximum number of chatbot submissions that can be running simultaneously. Can be set to None to mean no limit (any number of chatbot submissions can be running simultaneously). Set to "default" to use the default concurrency limit (defined by the `default_concurrency_limit` parameter in `.queue()`, which is 1 by default).

Demos

chatinterface_random_response chatinterface_streaming_echo

```python
import random
import gradio as gr

def random_response(message, history):
    return random.choice(["Yes", "No"])

demo = gr.ChatInterface(random_response)

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

Guides

Creating A Chatbot Fast

Sharing Your App

← Interface

TabbedInterface →