Client

```
gradio.Client(src, ...)
```

Description

The main Client class for the Python client. This class is used to connect to a remote Gradio app and call its API endpoints.

Example Usage

```
from gradio_client import Client

client = Client("abidlabs/whisper-large-v2")  # connecting to a Hugging Face Space
client.predict("test.mp4", api_name="/predict")
>> What a nice recording! # returns the result of the remote API call

client = Client("https://bec81a83-5b5c-471e.gradio.live")  # connecting to a temporary Gradio share URL
job = client.submit("hello", api_name="/predict")  # runs the prediction in a background thread
job.result()
>> 49 # returns the result of the remote API call (blocking call)
```

Initialization

Parameter	Description

ımeter	Description
`src` *str* ***required***	Either the name of the Hugging Face Space to load, (e.g. "abidlabs/whisper-large-v2") or the full URL (including "http" or "https") of the hosted Gradio app to load (e.g. "http://mydomain.com/app" or "https://bec81a83-5b5c-471e.gradio.live/").
`hf_token` *str | None* **default: None**	The Hugging Face token to use to access private Spaces. Automatically fetched if you are logged in via the Hugging Face Hub CLI. Obtain from: https://huggingface.co/settings/token
`max_workers` *int* **default: 40**	The maximum number of thread workers that can be used to make requests to the remote Gradio app simultaneously.
`serialize` *bool* **default: True**	Whether the client should serialize the inputs and deserialize the outputs of the remote API. If set to False, the client will pass the inputs and outputs as-is, without serializing/deserializing them. E.g. you if you set this to False, you'd submit an image in base64 format instead of a filepath, and you'd get back an image in base64 format from the remote API instead of a filepath.
`output_dir` *str | Path* **default: "/tmp/gradio"**	The directory to save files that are downloaded from the remote API. If None, reads from the GRADIO_TEMP_DIR environment variable. Defaults to a temporary directory on your machine.
`verbose` *bool* **default: True**	Whether the client should print statements to the console.
`auth` *tuple[str, str] | None* **default: None**	

Methods

predict

```
gradio.Client.predict(args, ...)
```

Description

Calls the Gradio API and returns the result (this is a blocking call).

Example Usage

```
from gradio_client import Client
client = Client(src="gradio/calculator")
client.predict(5, "add", 4, api_name="/predict")
>> 9.0
```

Agruments

Parameter	Description
`args` *<class 'inspect._empty'>* ***required***	The arguments to pass to the remote API. The order of the arguments must match the order of the inputs in the Gradio app.
`api_name` *str | None* **default: None**	The name of the API endpoint to call starting with a leading slash, e.g. "/predict". Does not need to be provided if the Gradio app has only one named API endpoint.
`fn_index` *int | None* **default: None**	As an alternative to api_name, this parameter takes the index of the API endpoint to call, e.g. 0. Both api_name and fn_index can be provided, but if they conflict, api_name will take precedence.

submit

```
gradio.Client.submit(args, ...)
```

Description

Creates and returns a Job object which calls the Gradio API in a background thread. The job can be used to retrieve the status and result of the remote API call.

Example Usage

```
from gradio_client import Client
client = Client(src="gradio/calculator")
job = client.submit(5, "add", 4, api_name="/predict")
job.status()
>> <Status.STARTING: 'STARTING'>
job.result()  # blocking call
>> 9.0
```

Agruments

Parameter	Description
`args` *<class 'inspect._empty'>* ***required***	The arguments to pass to the remote API. The order of the arguments must match the order of the inputs in the Gradio app.
`api_name` *str | None* **default: None**	The name of the API endpoint to call starting with a leading slash, e.g. "/predict". Does not need to be provided if the Gradio app has only one named API endpoint.
`fn_index` *int | None* **default: None**	As an alternative to api_name, this parameter takes the index of the API endpoint to call, e.g. 0. Both api_name and fn_index can be provided, but if they conflict, api_name will take precedence.
`result_callbacks` *Callable | list[Callable] | None* **default: None**	A callback function, or list of callback functions, to be called when the result is ready. If a list of functions is provided, they will be called in order. The return values from the remote API are provided as separate parameters into the callback. If None, no callback will be called.

view_api

`gradio.Client.view_api(...)`

Description

Prints the usage info for the API. If the Gradio app has multiple API endpoints, the usage info for each endpoint will be printed separately. If return_format="dict" the info is returned in dictionary format, as shown in the example below.

Example Usage

```
from gradio_client import Client
client = Client(src="gradio/calculator")
client.view_api(return_format="dict")
>> {
    'named_endpoints': {
        '/predict': {
            'parameters': [
                {
                    'label': 'num1',
                    'type_python': 'int | float',
                    'type_description': 'numeric value',
                    'component': 'Number',
                    'example_input': '5'
                },
                {
                    'label': 'operation',
                    'type_python': 'str',
                    'type_description': 'string value',
                    'component': 'Radio',
                    'example_input': 'add'
                },
                {
                    'label': 'num2',
                    'type_python': 'int | float',
                    'type_description': 'numeric value',
                    'component': 'Number',
                    'example_input': '5'
                },
            ],
            'returns': [
```

```
                {
                    'label': 'output',
                    'type_python': 'int | float',
                    'type_description': 'numeric value',
                    'component': 'Number',
                },
            ]
        },
        '/flag': {
            'parameters': [
                ...
                ],
            'returns': [
                ...
                ]
            }
        }
    'unnamed_endpoints': {
        2: {
            'parameters': [
                ...
                ],
            'returns': [
                ...
                ]
            }
        }
    }
}
```

Agruments

Parameter	Description
`all_endpoints` *bool* | *None* **default: None**	If True, prints information for both named and unnamed endpoints in the Gradio app. If False, will only print info about named endpoints. If None (default), will print info about named endpoints, unless there aren't any -- in which it will print info about unnamed endpoints.

Parameter	Description
`print_info` *bool* **default: True**	If True, prints the usage info to the console. If False, does not print the usage info.
`return_format` *Literal[('dict', 'str')] | None* **default: None**	If None, nothing is returned. If "str", returns the same string that would be printed to the console. If "dict", returns the usage info as a dictionary that can be programmatically parsed, and ***all endpoints are returned in the dictionary*** regardless of the value of `all_endpoints`. The format of the dictionary is in the docstring of this method.

duplicate

`gradio.Client.duplicate(from_id, ...)`

Description

Duplicates a Hugging Face Space under your account and returns a Client object for the new Space. No duplication is created if the Space already exists in your account (to override this, provide a new name for the new Space using `to_id`). To use this method, you must provide an `hf_token` or be logged in via the Hugging Face Hub CLI.
 The new Space will be private by default and use the same hardware as the original Space. This can be changed by using the `private` and `hardware` parameters. For hardware upgrades (beyond the basic CPU tier), you may be required to provide billing information on Hugging Face: https://huggingface.co/settings/billing

Example Usage

```
import os
from gradio_client import Client
HF_TOKEN = os.environ.get("HF_TOKEN")
client = Client.duplicate("abidlabs/whisper", hf_token=HF_TOKEN)
client.predict("audio_sample.wav")
>> "This is a test of the whisper speech recognition model."
```

Agruments

Parameter	Description
`from_id` *str* ***required***	The name of the Hugging Face Space to duplicate in the format "`username`/`space_id`", e.g. "gradio/whisper".
`to_id` *str | None* **default: None**	The name of the new Hugging Face Space to create, e.g. "abidlabs/whisper-duplicate". If not provided, the new Space will be named "`your_HF_username`/`space_id`".
`hf_token` *str | None* **default: None**	The Hugging Face token to use to access private Spaces. Automatically fetched if you are logged in via the Hugging Face Hub CLI. Obtain from: https://huggingface.co/settings/token
`private` *bool* **default: True**	Whether the new Space should be private (True) or public (False). Defaults to True.
`hardware` *Literal[('cpu-basic', 'cpu-upgrade', 't4-small', 't4-medium', 'a10g-small', 'a10g-large', 'a100-large')] | SpaceHardware | None* **default: None**	The hardware tier to use for the new Space. Defaults to the same hardware tier as the original Space. Options include "cpu-basic", "cpu-upgrade", "t4-small", "t4-medium", "a10g-small", "a10g-large", "a100-large", subject to availability.
`secrets` *dict[str, str] | None* **default: None**	A dictionary of (secret key, secret value) to pass to the new Space. Defaults to None. Secrets are only used when the Space is duplicated for the first time, and are not updated if the duplicated Space already exists.
`sleep_timeout` *int* **default: 5**	The number of minutes after which the duplicate Space will be puased if no requests are made to it (to minimize billing charges). Defaults to 5 minutes.

Parameter	Description
`max_workers` *int* **default: 40**	The maximum number of thread workers that can be used to make requests to the remote Gradio app simultaneously.
`verbose` *bool* **default: True**	Whether the client should print statements to the console.

deploy_discord

`gradio.Client.deploy_discord(...)`

Description

Deploy the upstream app as a discord bot. Currently only supports gr.ChatInterface.

Agruments

Parameter	Description
`discord_bot_token` *str	None* **default: None**	This is the "password" needed to be able to launch the bot. Users can get a token by creating a bot app on the discord website. If run the method without specifying a token, the space will explain how to get one. See here: https://huggingface.co/spaces/freddyaboulton/test-discord-bot-v1.
`api_names` *list[str	tuple[str, str]]	None* **default: None**	The api_names of the app to turn into bot commands. This parameter currently has no effect as ChatInterface only has one api_name ('/chat').

Parameter	Description
`to_id` *str	None* **default: None**	The name of the space hosting the discord bot. If None, the name will be gradio-discord-bot-`random-substring`
`hf_token` *str	None* **default: None**	HF api token with write priviledges in order to upload the files to HF space. Can be ommitted if logged in via the HuggingFace CLI, unless the upstream space is private. Obtain from: https://huggingface.co/settings/token
`private` *bool* **default: False**	Whether the space hosting the discord bot is private. The visibility of the discord bot itself is set via the discord website. See https://huggingface.co/spaces/freddyaboulton/test-discord-bot-v1