Response Status Code

The same way you can specify a response model, you can also declare the HTTP status code used for the response with the parameter `status_code` in any of the *path operations*:

- `@app.get()`
- `@app.post()`
- `@app.put()`
- `@app.delete()`
- etc.

```
{!../../../docs_src/response_status_code/tutorial001.py!}
```

!!! note Notice that `status_code` is a parameter of the "decorator" method (`get` , `post` , etc). Not of your *path operation function*, like all the parameters and body.

The `status_code` parameter receives a number with the HTTP status code.

!!! info `status_code` can alternatively also receive an `IntEnum` , such as Python's http.HTTPStatus .

It will:

- Return that status code in the response.
- Document it as such in the OpenAPI schema (and so, in the user interfaces):

!!! note Some response codes (see the next section) indicate that the response does not have a body.

```
FastAPI knows this, and will produce OpenAPI docs that state there is no response
body.
```

About HTTP status codes

!!! note If you already know what HTTP status codes are, skip to the next section.

In HTTP, you send a numeric status code of 3 digits as part of the response.

These status codes have a name associated to recognize them, but the important part is the number.

In short:

- `100` and above are for "Information". You rarely use them directly. Responses with these status codes cannot have a body.
- **200** and above are for "Successful" responses. These are the ones you would use the most.
    - `200` is the default status code, which means everything was "OK".
    - Another example would be `201` , "Created". It is commonly used after creating a new record in the database.
    - A special case is `204` , "No Content". This response is used when there is no content to return to the client, and so the response must not have a body.

- **300** and above are for "Redirection". Responses with these status codes may or may not have a body, except for `304`, "Not Modified", which must not have one.
- **400** and above are for "Client error" responses. These are the second type you would probably use the most.
  - An example is `404`, for a "Not Found" response.
  - For generic errors from the client, you can just use `400`.
- **500** and above are for server errors. You almost never use them directly. When something goes wrong at some part in your application code, or server, it will automatically return one of these status codes.

!!! tip To know more about each status code and which code is for what, check the [MDN documentation about HTTP status codes](#).

Shortcut to remember the names

Let's see the previous example again:

```
{!../../../docs_src/response_status_code/tutorial001.py!}
```

`201` is the status code for "Created".

But you don't have to memorize what each of these codes mean.

You can use the convenience variables from `fastapi.status`.

```
{!../../../docs_src/response_status_code/tutorial002.py!}
```

They are just a convenience, they hold the same number, but that way you can use the editor's autocomplete to find them:

!!! note "Technical Details" You could also use `from starlette import status`.

```
**FastAPI** provides the same `starlette.status` as `fastapi.status` just as a
convenience for you, the developer. But it comes directly from Starlette.
```

Changing the default

Later, in the [Advanced User Guide](#){.internal-link target=_blank}, you will see how to return a different status code than the default you are declaring here.