**What is httpServerResponse.statusCode?**

This property controls the status code that will be delivered to the client when the headers are flushed.

The **httpServerResponse.statusCode** is an internal application programming interface of class ServerResponse within the HTTP module.

**Types of status codes**

[**Status codes**](https://developer.mozilla.org/en-US/docs/Web/HTTP/Status) are a set of universal web development codes utilized to create apps all over the world.

* Informational responses (100–199)
* Successful responses (200–299)
* Redirects (300–399)
* Client errors (400–499)
* Server errors (500–599)

**How to use httpServerResponse.statusCode**

In our discussion, we’ll see some instances of how to use some of these status codes.

**1. 200 OK**

The **HTTP 200 OK response code** implies a successful request.

Here is an example of this kind of HTTP response.

* **GET**: The response contains an entity that corresponds to the resource requested.
* **POST**: An entity that describes or contains the action’s result.
* **HEAD**: In the response, the entity-header fields related to the requested resource are sent without any message-body.

Example:

    res.status(200).json({

code: 200,

success: true,

message:” successfully”

data:response | {}

});

**2. 201 Created**

This indicates that the request was completed, and a new resource was created as a result. The URI(s) sent in the response entity can refer to the newly generated help, with a Location header field providing the most particular URI for the resource.

    res.status(201).json({

code: 201,

success: true,

message:”successfully”

data:response | {}

});

**3. 204 No Content**

This response is sent when the server has completed the request and does not need to return an entity-body, but wants updated metainformation. The answer may contain additional or updated metadata in entity-headers, which should be associated with the requested variant if present.

    res.status(204).json({

code: 204,

success: true,

message : “No Content”

data:response | {}

});

**4. 400 Bad Request**

When the server is unable to understand the request due to incorrect syntax, this response is returned.

    res.status(400).json({

code: 400,

success: true,

message:”Bad Request”

data:response | {}

});

**5. 401 Unauthorized**

Even though the HTTP standard specifies “unauthorized,” this response indicates “unauthenticated” logically. To get the requested response, the client must first authenticate itself.

This response code can also be used to ensure the user has the proper access to the resource.

    res.status(401).json({

code: 401,

success: true,

message:”unauthorized”

data:response | {}

});

**6. 403 Forbidden**

This response is gotten when the server refuses to provide the requested resource because the client does not have access permissions to the material; it is unauthorized. Unlike 401, the server is aware of the client’s identity.

    res.status(403).json({

code: 403,

success: true,

message:”Forbidden

data:response | {}

});

**7. 404 Not Found**

This means the requested resource is unavailable on the server, or maybe the URL has been mistyped in the browser. This can also indicate that the endpoint is correct, but the resource does not exist in an API. This response may be used instead of 403 to conceal the existence of a resource from an unauthorized client.

    res.status(404).json({

code: 404,

success: true,

message:”Not Found”

data:response | {}

});

**8. 500 Internal Server Error**

The server ran into an unanticipated problem that stopped it from completing the request.

    res.status(500).json({

code:500,

success: true,

message:” Internal Server Error”

data:response | {}

});

**9. 503 Service Not Available**

This happens due to a sudden overload or server maintenance and it means the server cannot handle the request.

    res.status(503).json({

code:503,

success: true,

message:” Service Not Available”

data:response | {}

});

**10. 504 Gateway Timeout**

When the server is operating as a gateway and cannot promptly respond, this error response is returned.

res.status(504).json({

code:504,

success: true,

message:” Gateway Timeout”

data:response | {}

});

There are many different status codes for HTTP responses, and we’ve gone through a few of them here. For more, see the resources section of this page.