HTTP and the Fetch API

HyperionDev

Muhammad Zahir Junejo

Lecture - Housekeeping

- ❑ The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.
  - ❑ Please review Code of Conduct (in Student Undertaking Agreement) if unsure
- ❑ No question is daft or silly - **ask them!**
- ❑ Q&A session at the end of the lesson, should you wish to ask any follow-up questions.
- ❑ Should you have any questions after the lecture, please schedule a mentor session.
- ❑ For all non-academic questions, please submit a query: www.hyperiondev.com/support

Lecture Objectives

1. What is HTTP(S)?
2. The HTTP Request-Response Cycle
3. Anatomy of an HTTP Request
4. Anatomy of an HTTP Response
5. Introducing the Fetch API
6. Making GET Requests with Fetch
7. Handling Responses with Fetch

What is HTTP(S)?

- HTTP(S), Hypertext Transfer Protocol (Secure), is the foundation of data communication on the World Wide Web. It defines how messages are formatted and transmitted and how web servers and browsers should respond to various commands.
- HTTP is the standard protocol for transferring hypertext, typically in the form of HTML files, between a web server and a browser.
- HTTPS is the secure version of HTTP, encrypting data to ensure privacy and security during transmission.

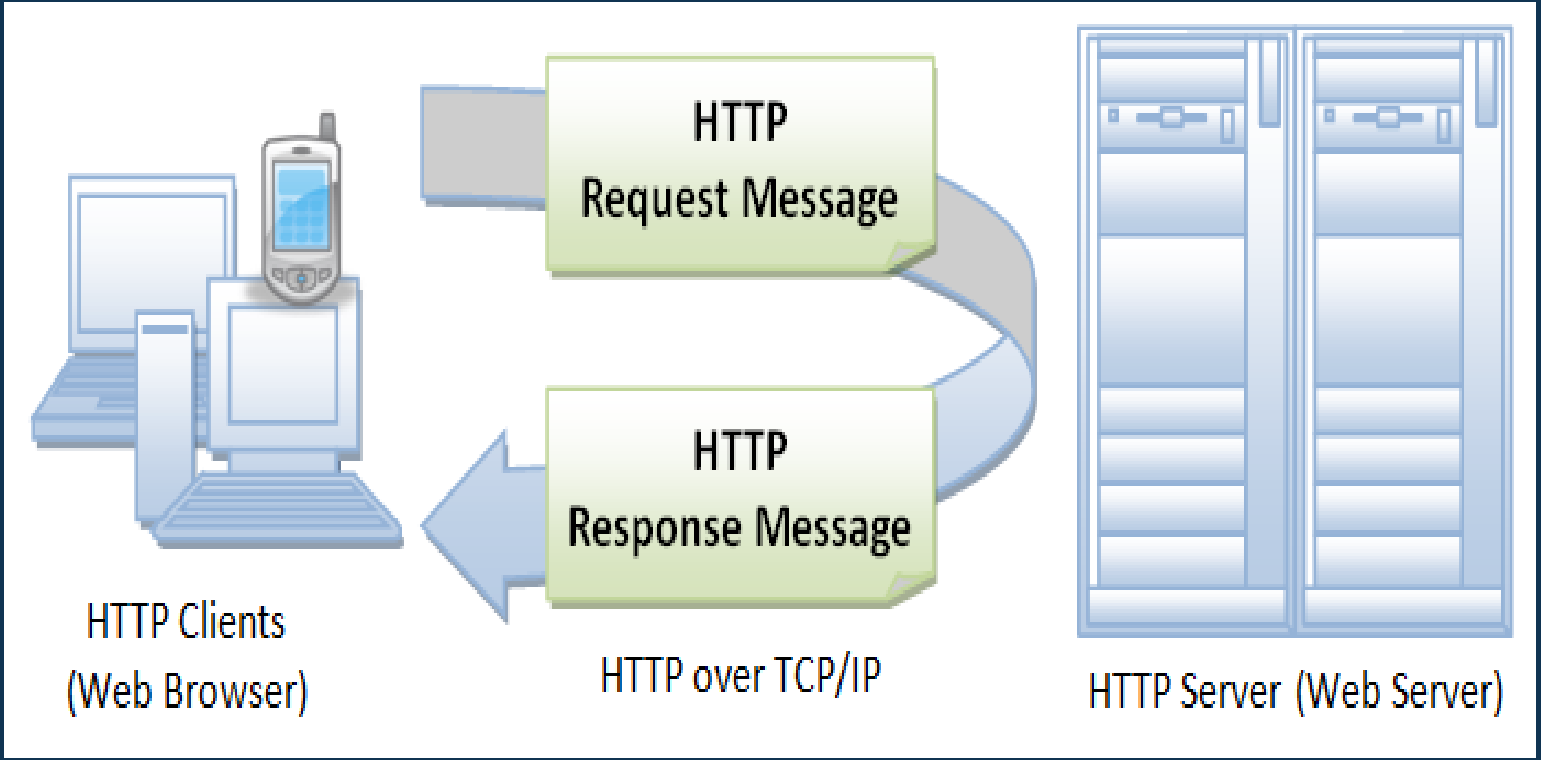
What is HTTP(S)?

- ❑ HTTP(S), Hypertext Transfer Protocol (Secure), is the foundation of data communication on the World Wide Web. It defines how messages are formatted and transmitted and how web servers and browsers should respond to various commands.
- ❑ HTTP is the standard protocol for transferring hypertext, typically in the form of HTML files, between a web server and a browser.
- ❑ HTTPS is the secure version of HTTP, encrypting data to ensure privacy and security during transmission.

The HTTP Request-Response Cycle

- HTTP operates as a request-response protocol, where clients (e.g., browsers) send requests to servers, and servers respond with data. This cycle forms the basis of web communication.
- Client (Browser) sends an HTTP Request to the Server.
- Server processes the request and generates an HTTP Response.
- The Response is sent back to the Client.
- The Client processes the Response and displays content.

The HTTP Request-Response Cycle



Anatomy of an HTTP Request

- ❑ HTTP Requests consist of several components:
  - ❑ Request Method (GET, POST, etc.)
  - ❑ URL
  - ❑ Headers (optional)
  - ❑ Body (optional, e.g., for POST requests)

GET /example-page HTTP/1.1

Host: www.example.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/58.0.3029.110 Safari/537.36

Anatomy of an HTTP Response

- HTTP Responses also consist of several components:
  - Status Code (e.g., 200 OK, 404 Not Found)
  - Headers (optional)
  - Body (contains the actual content)

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 1234

<!DOCTYPE html>
<html>
<!-- ... -->
</html>
```

Introducing the Fetch API

- ❑ The Fetch API is a modern JavaScript API for making HTTP requests in a more flexible and promise-based way. It replaces older methods like XMLHttpRequest.
- ❑ Fetch is based on Promises, allowing for cleaner and more concise asynchronous code.
- ❑ It supports a wide range of HTTP methods (GET, POST, PUT, DELETE, etc.).
- ❑ Fetch is built into modern browsers, making it readily available for web developers.

Making Requests with Fetch

❑ Making requests with Fetch is straightforward and involves specifying the URL you want to request.

❑ Example:

```
// Making a GET request with Fetch
fetch('https://api.example.com/data')
    .then(response => response.json())
    .then(data => {
        // Process the data
    })
    .catch(error => {
        // Handle errors
    });
```

References

- https://developer.mozilla.org/en-US/docs/Web/HTTP
- https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API
- https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch

Questions and Answers

Thank You!