

Introcution to JavaScript

Nils Twelker

March 2023

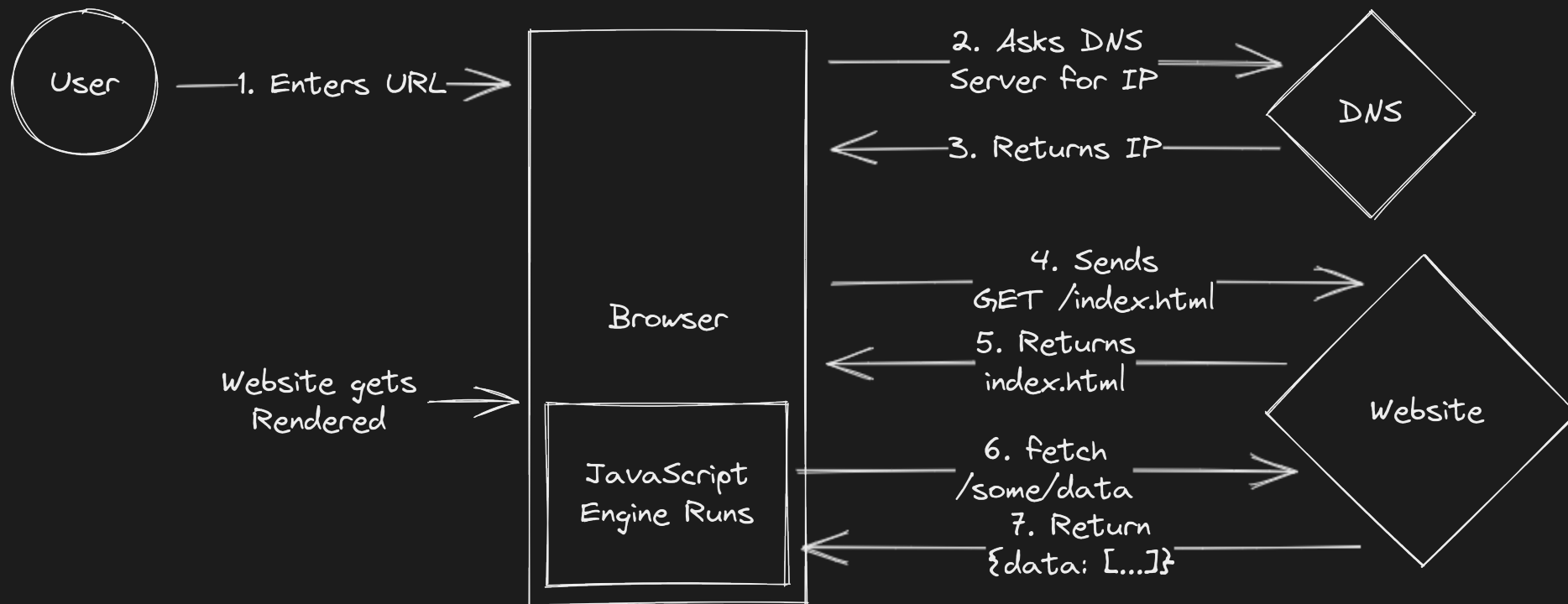
What learned we last Week?

- Common Events `click` , `dblclick` , `mouseover` , `keydown`
- `on<event>` Attribute
- Event Bubbling
- Dispatching Events (Custom Events)
- Displaying Lists `Item A`
- Displaying Tables `<table><tr><th>Name</th></tr></table>`
- Displaying Forms `<form><input type='text'></form>`

Goals of this week

- Server Client Communication
 - HTTP (Hypertext Transfer Protocol)
 - CRUD (Create, Read, Update, Delete)
- JSON (JavaScript Object Notation)
- AJAX (Asynchronous JavaScript and XML)
- Fetch API
- Promises
- Async/Await

Server Client Communication



HTTP (Hypertext Transfer Protocol)

Protocol for communication between a Web Client and a Web Server.

1. Web Client sends HTTP Request to Web Server.
2. Web Server processes Request.
3. Web Server sends HTTP Response to Web Client.

HTTP Request & Response

```
GET /index.html HTTP/2
Host: www.example.com
```

```
HTTP/2 200 OK
Content-Type: text/html
Content-Length: 1234

<html>
  <head>
    <title>Example</title>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

CRUD (Create, Read, Update, Delete)

Operations for persistent data.

- Create: POST
- Read: GET
- Update: PUT or PATCH
- Delete: DELETE

CRUD (Create)

Request

```
POST /users HTTP/2
Host: www.example.com
Content-Type: application/json

{
  "name": "John Doe",
  "age": 42
}
```

Response

```
HTTP/2 201 Created
Content-Type: application/json
Content-Length: 123

{
  "id": 123,
  "name": "John Doe",
  "age": 42
}
```

CRUD (Read)

Request

```
GET /users/123 HTTP/2  
Host: www.example.com
```

Response

```
HTTP/2 200 OK  
Content-Type: application/json  
Content-Length: 123  
  
{  
  "id": 123,  
  "name": "John Doe",  
  "age": 42  
}
```

CRUD (Update)

Request

```
PATCH /users/123 HTTP/2
Host: www.example.com
Content-Type: application/json

{
  "age": 43
}
```

Response

```
HTTP/2 200 OK
Content-Type: application/json
Content-Length: 123

{
  "id": 123,
  "name": "John Doe",
  "age": 43
}
```

CRUD (Delete)

Request

```
DELETE /users/123 HTTP/2  
Host: www.example.com
```

Response

```
HTTP/2 204 No Content
```

HTTP Status Codes

- 1xx: Informational
- 2xx: Success
- 3xx: Redirection
- 4xx: Client Error
- 5xx: Server Error

Some common Status Codes:

- 200 OK
- 201 Created
- 204 No Content
- 400 Bad Request
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found
- 500 Internal Server Error
- 503 Service Unavailable

JSON (JavaScript Object Notation)

Lightweight data-interchange format.

```
{  
  "name": "John Doe",  
  "age": 42,  
}
```

JSON Data Types

```
{  
  "string": "Hello World",  
  "number": 42,  
  "boolean": true,  
  "null": null,  
  "array": [1, 2, 3],  
  "object": {  
    "name": "John Doe",  
    "age": 42  
  }  
}
```

AJAX (Asynchronous JavaScript and XML)

Technique for asynchronous communication between a Web Client and a Web Server.

- Asynchronous: No page reload.
- JavaScript: Client side scripting language.
- XML: Data format. (JSON is more common today)

Fetch API

API for making HTTP Requests.

```
fetch("https://example.com/users/123")  
  .then(response => response.json())  
  .then(data => console.log(data))
```

Fetch API (POST)

```
fetch("https://example.com/users", {  
  method: "POST",  
  headers: {  
    "Content-Type": "application/json"  
  },  
  body: JSON.stringify({  
    name: "John Doe",  
    age: 42  
  })  
}).then(response => response.json())  
  .then(data => console.log(data))
```

Promises

Object that represents the eventual completion (or failure) of an asynchronous operation.

```
const promise = new Promise((resolve, reject) => {  
  setTimeout(() => {  
    resolve("Hello World")  
  }, 1000)  
})  
  
promise.then(data => console.log(data))
```

Async/Await

```
async function getData() {  
    const response = await fetch("https://example.com/users/123")  
    const data = await response.json()  
    return data  
}
```

```
const myData = await getData()  
console.log(myData)
```

Tasks and Points

Goal is to get 100 Points.

- `custom-events` (20 Points)
- `lists` (30 Points)
- `form` (30 Points)
- `tables` (40 Points)
- `more-events` (40 Points)
- `event-bubbling` (40 Points)
- `friend-list` (80 Points)