

DWA_03.4 Knowledge Check_DWA3.1

1. Please show how you applied a Markdown File to a piece of your code.


MARKDOWN

```
1 ~ # Hi 🙋, I'm Keamogetswe Suprise Kgakatsi
2 ~ ### A passionate Software Developer from South Africa
3 ~ ![:coding](https://cdn.dribbble.com/users/1859583/screenshots/4171367/coding-freak.gif)
4
5 ~ ![:banner](https://komarev.com/ghnvc/?username=kgakatsikeamogetswe&label=Profile%20views&color=0e75b6&style=flat)
6
7 ~ 🌱 I'm currently learning **Web Development**
8
9 ~ 🗨 Ask me about **HTML & CSS**
10
11 ~ 📧 How to reach me **stimulatekay11ew@gmail.com**
12
13 ~ ⚡ Fun fact **I think I am funny**
14
15 ~ ### Table of content
16 ~ | **FirstName** | **Second** |
17 ~ | --- | --- |
18 ~ | Keamogetswe | Kgakatsi |
19 ~ | Oratile | Mokgoatlheng |
20 ~
```

PREVIEW

Hi 🙋, I'm Keamogetswe Suprise Kgakatsi

A passionate Software Developer from South Africa



MARKDOWN

```
1 ~ # Hi 🙋, I'm Keamogetswe Suprise Kgakatsi
2 ~ ### A passionate Software Developer from South Africa
3 ~ ![:coding](https://cdn.dribbble.com/users/1859583/screenshots/4171367/coding-freak.gif)
4
5 ~ ![:banner](https://komarev.com/ghnvc/?username=kgakatsikeamogetswe&label=Profile%20views&color=0e75b6&style=flat)
6
7 ~ 🌱 I'm currently learning **Web Development**
8
9 ~ 🗨 Ask me about **HTML & CSS**
10
11 ~ 📧 How to reach me **stimulatekay11ew@gmail.com**
12
13 ~ ⚡ Fun fact **I think I am funny**
14
15 ~ ### Table of content
16 ~ | **FirstName** | **Second** |
17 ~ | --- | --- |
18 ~ | Keamogetswe | Kgakatsi |
19 ~ | Oratile | Mokgoatlheng |
20 ~
```

PREVIEW

Profile views 3

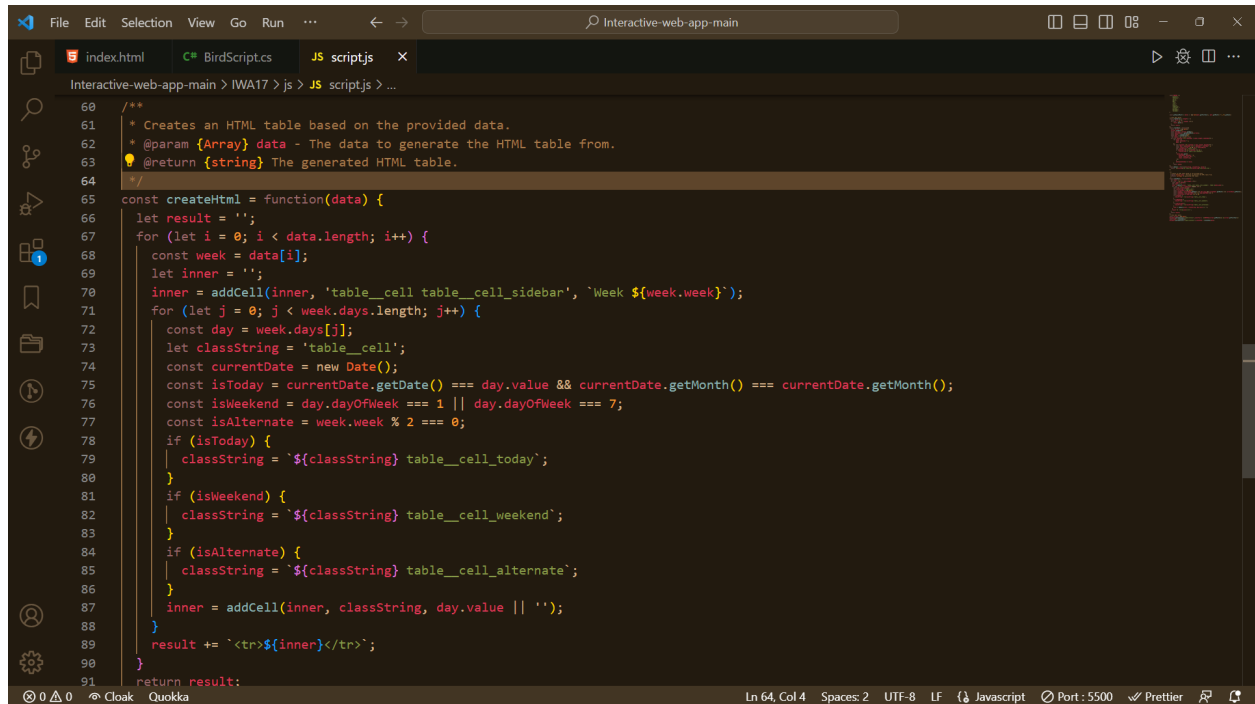
- 🌱 I'm currently learning Web Development
- 🗨 Ask me about HTML & CSS
- 📧 How to reach me stimulatekay11ew@gmail.com
- ⚡ Fun fact I think I am funny

Table of content

FirstName	Second
Keamogetswe	Kgakatsi
Oratile	Mokgoatlheng

```
const myStates = [
  "Phokeng",
  "Lefaragatlhe",
```

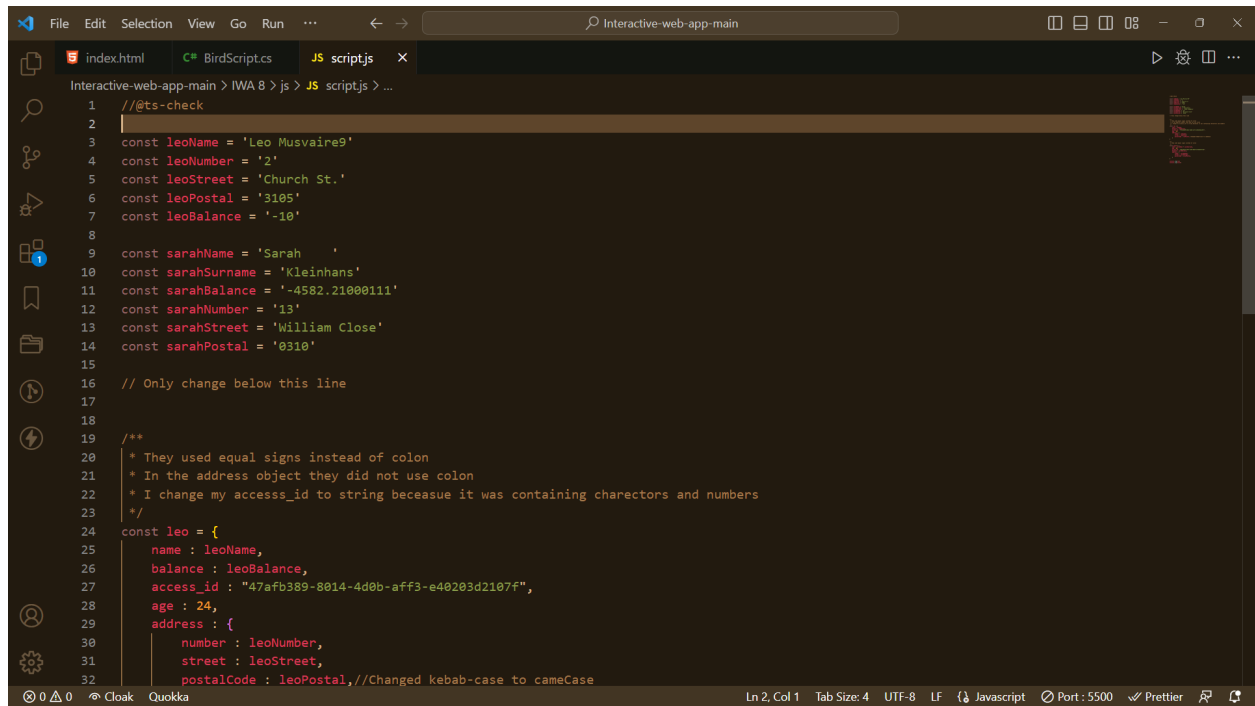
2. Please show how you applied JSDoc Comments to a piece of your code.



The screenshot shows a code editor with a dark theme. The file explorer on the left shows a project structure with files like index.html, BirdScripts, and script.js. The main editor area displays a JavaScript file named script.js. At the top, there are JSDoc comments for a function named createHtml. The comments include a description, a parameter annotation for 'data', and a return annotation. Below the comments is the implementation of the createHtml function, which iterates over an array of data and builds an HTML table string. The function uses various conditional checks to determine the class of each table cell based on the current date and the day of the week. The status bar at the bottom indicates the current line and column (Ln 64, Col 4), the number of spaces (2), the encoding (UTF-8), the line ending (LF), the language (JavaScript), the port (5500), and the formatter (Prettier).

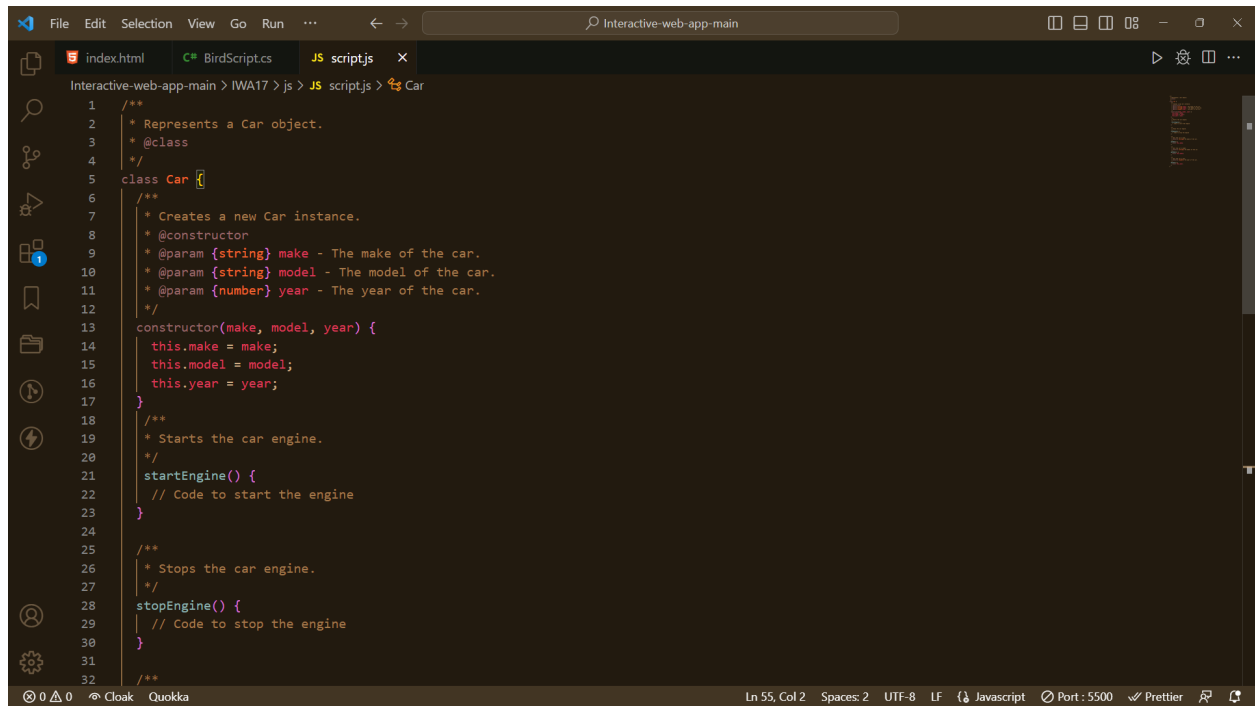
```
60 /**
61  * Creates an HTML table based on the provided data.
62  * @param {Array} data - The data to generate the HTML table from.
63  * @return {string} The generated HTML table.
64  */
65 const createHtml = function(data) {
66   let result = '';
67   for (let i = 0; i < data.length; i++) {
68     const week = data[i];
69     let inner = '';
70     inner = addCell(inner, 'table__cell table__cell_sidebar', `Week ${week.week}`);
71     for (let j = 0; j < week.days.length; j++) {
72       const day = week.days[j];
73       let classString = 'table__cell';
74       const currentDate = new Date();
75       const isToday = currentDate.getDate() === day.value && currentDate.getMonth() === currentDate.getMonth();
76       const isWeekend = day.dayOfWeek === 1 || day.dayOfWeek === 7;
77       const isAlternate = week.week % 2 === 0;
78       if (isToday) {
79         classString = `${classString} table__cell_today`;
80       }
81       if (isWeekend) {
82         classString = `${classString} table__cell_weekend`;
83       }
84       if (isAlternate) {
85         classString = `${classString} table__cell_alternate`;
86       }
87       inner = addCell(inner, classString, day.value || '');
88     }
89     result += `<tr>${inner}</tr>`;
90   }
91   return result;
}
```

3. Please show how you applied the @ts-check annotation to a piece of your code.



```
1 // @ts-check
2
3 const leoName = 'Leo Musvair9'
4 const leoNumber = '2'
5 const leoStreet = 'Church St.'
6 const leoPostal = '3105'
7 const leoBalance = '-10'
8
9 const sarahName = 'Sarah '
10 const sarahSurname = 'Kleinhans'
11 const sarahBalance = '-4582.2100111'
12 const sarahNumber = '13'
13 const sarahStreet = 'William Close'
14 const sarahPostal = '0310'
15
16 // Only change below this line
17
18
19 /**
20  * They used equal signs instead of colon
21  * In the address object they did not use colon
22  * I change my access_id to string because it was containing characters and numbers
23  */
24 const leo = {
25   name : leoName,
26   balance : leoBalance,
27   access_id : "47afb389-8014-4d0b-aff3-e40203d2107f",
28   age : 24,
29   address : {
30     number : leoNumber,
31     street : leoStreet,
32     postalCode : leoPostal, // Changed kebab-case to camelCase
```

4. As a BONUS, please show how you applied any other concept covered in the 'Documentation' module.



The screenshot shows a web browser window with the address bar displaying "Interactive-web-app-main". The browser has three tabs open: "index.html", "BirdScriptcs", and "JS scriptjs". The "JS scriptjs" tab is active, showing a JavaScript class definition for a Car object. The code is as follows:

```
1  /**
2   * Represents a Car object.
3   * @class
4   */
5  class Car {
6      /**
7       * Creates a new Car instance.
8       * @constructor
9       * @param {string} make - The make of the car.
10      * @param {string} model - The model of the car.
11      * @param {number} year - The year of the car.
12      */
13      constructor(make, model, year) {
14          this.make = make;
15          this.model = model;
16          this.year = year;
17      }
18      /**
19       * Starts the car engine.
20       */
21      startEngine() {
22          // Code to start the engine
23      }
24      /**
25       * Stops the car engine.
26       */
27      stopEngine() {
28          // Code to stop the engine
29      }
30  }
31  /**
32  */
```

The status bar at the bottom of the browser shows "0 0 0", "Cloak", "Quokka", "Ln 55, Col 2", "Spaces: 2", "UTF-8", "LF", "Javascript", "Port: 5500", "Prettier", and a bell icon.