

# DWA\_03.4 Knowledge Check\_DWA3.1

---

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

```
const createUniqueId = () => {
  const random1 = Math.floor(Math.random() * 1000000000000000)
  const random2 = Math.floor(Math.random() * 1000000000000000)
  const timestamp = new Date().getTime()
  return `${random1}-${timestamp}-${random2}`
}

/**
 * A factory function that creates an order object literal representing an
 * actual order in the app state. It is wrapped in a factory function instead of
 * just being created directly since several values are created automatically
 * such as a unique ID and the creation date of the order.
 *
 * @param {object} props
 * @returns {object}
 * @prop {string} title
 * @prop {string} id -- A unique value generated by {@link createUniqueId} |
 */
export const createOrderData = (props) => {
  const { title, table, column } = props

  return {
    title,
    table,
    column,
    id: createUniqueId(),
    created: new Date(),
  }
}
```

---

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

```
JS scripts.js 9+, M X JS data.js JS view.js
IWA_18 > JS scripts.js > [?] handleHelpToggle
1  // @ts-check
2
3  import { html, updateDraggingHtml, createOrderHtml, moveToColumn } from "../view.js";
4  import { COLUMNS, createOrderData, updateDragging, state } from "../data.js";
5
6  const {add, other, help, edit, columns} = html
7
8  /**
9   * A handler that fires when a user drags over any element inside a column. In
10   * order to determine which column the user is dragging over the entire event
11   * bubble path is checked with `event.path` (or `event.composedPath()` for
12   * browsers that don't support `event.path`). The bubbling path is looped over
13   * until an element with a `data-area` attribute is found. Once found both the
14   * active dragging column is set in the `state` object in "data.js" and the HTML
15   * is updated to reflect the new column.
16   *
17   * @param {object} event
18   */
19  const handleDragOver = (event) => {
20    event.preventDefault();
21    const path = event.path || event.composedPath()
22    let column = null
23
24    for (const element of path) {
25      const { area } = element.dataset
26      if (area) {
27        column = area
28        break;
29      }
30    }
31
32    if (!column) return
33    updateDragging({ over: column })
```

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

```
JS scripts.js 9+, M X JS data.js M JS view.js
IWA_18 > JS scripts.js > [?] handleHelpToggle
1  // @ts-check
2
3  import { html, updateDraggingHtml, createOrderHtml, moveToColumn } from "../view.js";
4  import { COLUMNS, createOrderData, updateDragging, state } from "../data.js";
5
6  const {add, other, help, edit, columns} = html
```

```

    },
    //.....
    function handleAddToggle(event) {
        const { target } = event;
        const isAddOrderButton = target === other.add;

        if (isAddOrderButton) {
            add.overlay.open = true;
        } else {
            add.overlay.open = false;
            add.form.reset();
        }

        other.add.focus();
    };

```

---

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

```

* @param {object} props
* @returns {object}
* @prop {string} title
* @prop {string} id -- A unique value generated by {@link createUniqueId} |
*/
export const createOrderData = (props) => {
    const { title, table, column } = props

```

Making use of the @link creatUniqeld was something i picked up during the course.

---