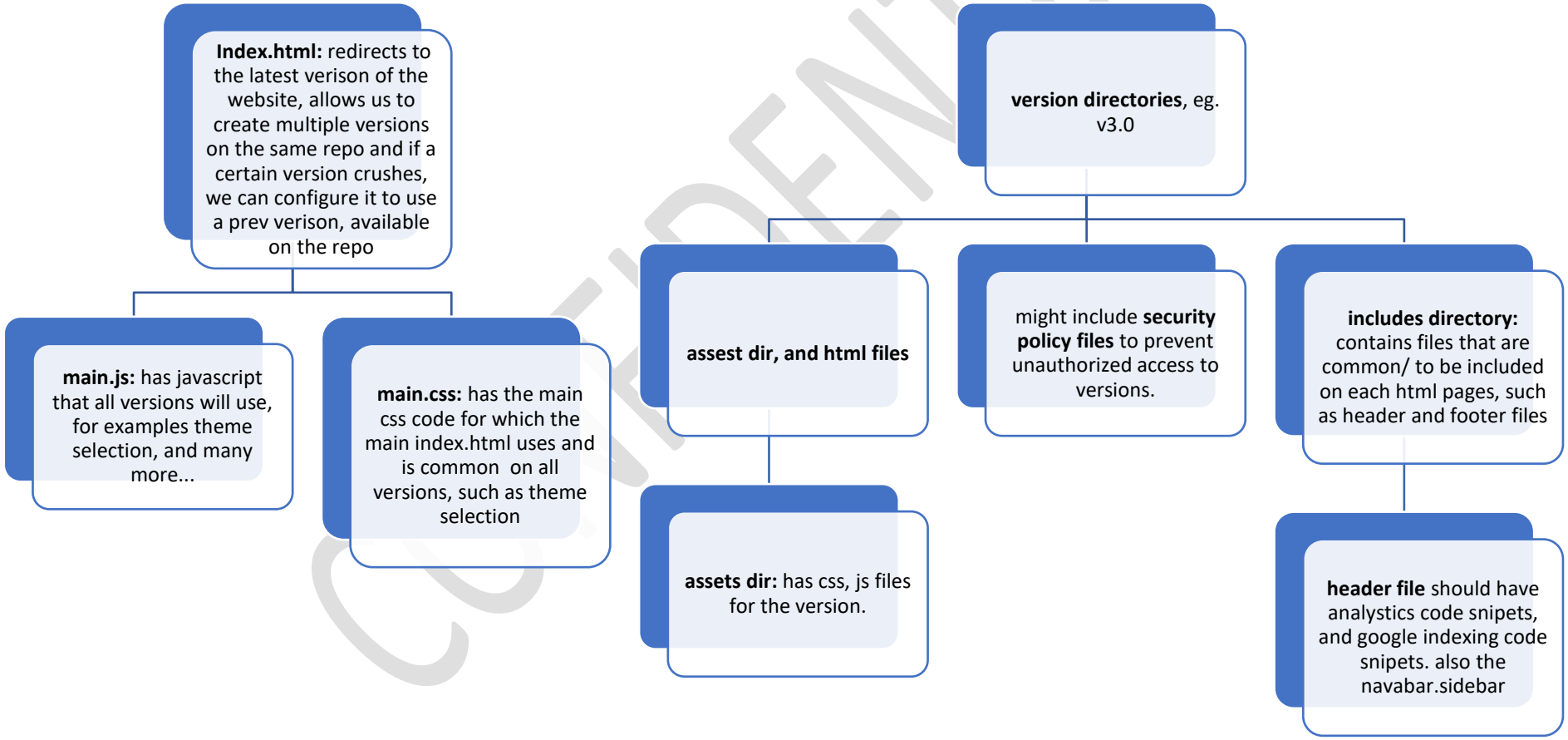
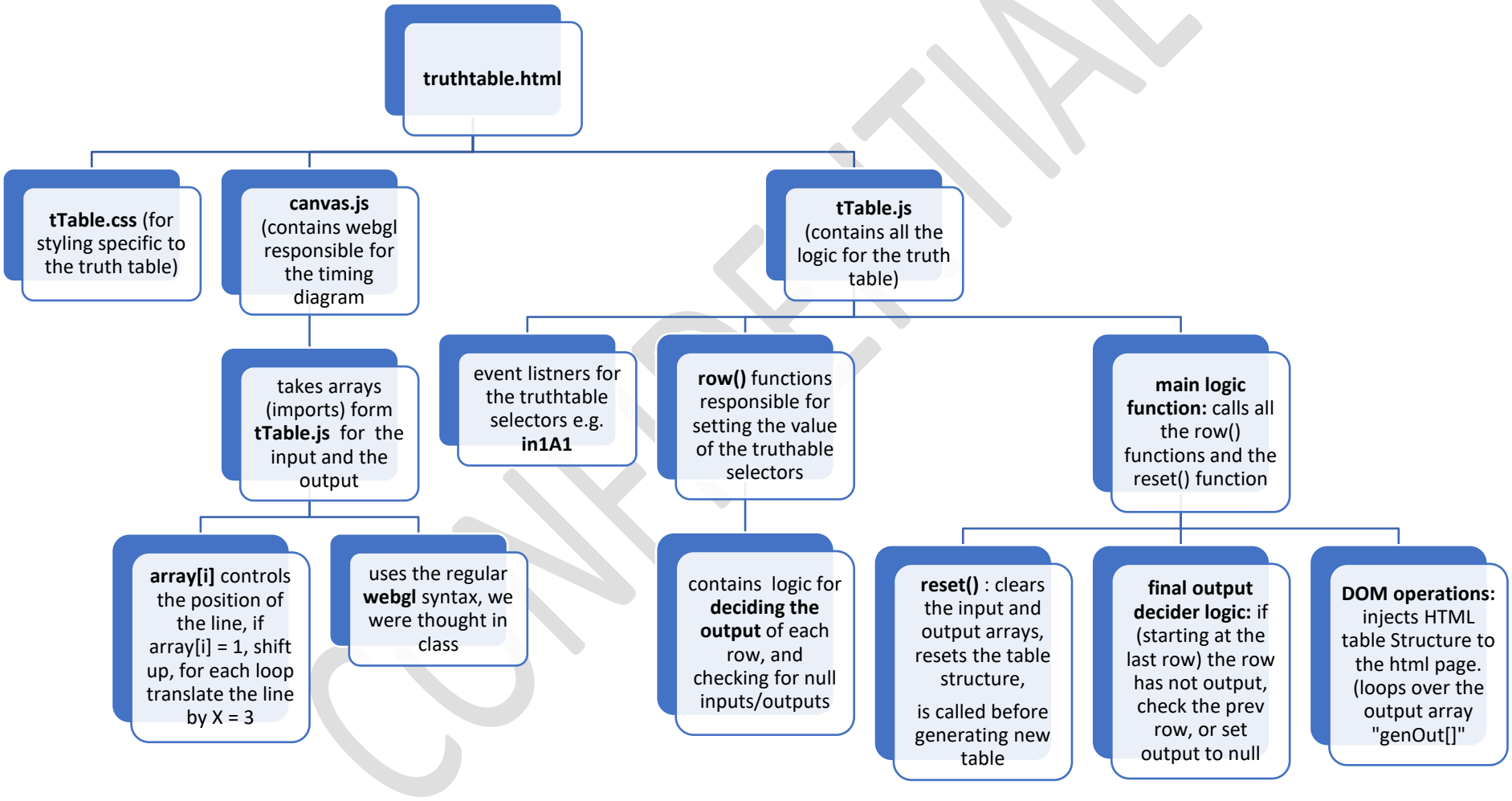
 by M. Mpotulo

PROJECT DIRECTORY HEIRACHY



 by M. Mpotulo

TRUTH-TABLE GENERATOR LOGIC HEIRACHY



 by M. Mpotulo

CODE EXPLANATION

1. SELECTOR VARIABLES

```
5     //selctor inputs
6     //row 1
7     const in1A1 = document.querySelector("#in1A1")
8     const in2A1 = document.querySelector("#in2A1")
9     const in1C1 = document.querySelector("#in1C1")
10    const in2C1 = document.querySelector("#in2C1")
```

- References all the selectors on the Combinational logic designer table.
- Values corresponding to the selectors are declared almost the same way, but in lower cases, e.g. **in1a1 for in1A1**

2. OUTPUT VALUES

```
61    //outputs
62    var B1o = null, B2o = null, B3o = null, B4o = null, B5o = null, B6o = null
63    var D1o = null, D2o = null, D3o = null, D4o = null, D5o = null, D6o = null
64
65    var outputR1 = null, outputR2 = null, outputR3 = null
66    var outputR4 = null, outputR5 = null, outputR6 = null
67    var output = null
```

Var B1o to D6o are for the output of each gate on the table, and var outputR1 to outputR6 are outputs for each row on the table.

3. GENERATOR ARRAYS

```
76      //truthable variables
77      export var genOut = []
78      export var genInA = [], genInB = [], genInC = [], genInD = []
79
```

- This are used to actually generate the truth table; they are predefined in the main logic function.
- They are exported since they are also used in the canvas.js to generate the timing diagram.
- genOut's value is not defined, it takes its values from the final output from the rows combined.

4. ROW OUTPUT INDICATOR BUTTONS

```
80      //row output buttons
81      const row1btn = document.querySelector("#row1btn")
82      const row2btn = document.querySelector("#row2btn")
83      const row3btn = document.querySelector("#row3btn")
84      const row4btn = document.querySelector("#row4btn")
85      const row5btn = document.querySelector("#row5btn")
86      const row6btn = document.querySelector("#row6btn")
```

- these are used to indicate if a certain row has an output,
- If true, the button will turn green,
- If not, it will turn red.
- Its useful in development mode, since it will help with debugging error on the truth table.

5. ROW FUNCTIONS

```
94    /**********************Row 1**********************/
95  > function row1() {
262   }···
263   /**********************Row 2**********************/
264 > function row2() {
447   }···
448   /**********************Row 3**********************/
449 > function row3() {
643   }···
```

- they handle the logic of each function independently of the other row,
- At the end of each function, they set the row output.
- On each row() there are input checker arrays, which are responsible for checking if a certain output is selected.
- Separating these functions will make it easier for us to debug errors.
- They are all called on the main function truthtable()

6. SETTING ARRAYS BASED ON NUMBER OF INPUTS SELECTED

```
1378        if (no_inputs.value == 2) {
1379            genInA = [0, 0, 1, 1]
1380            genInB = [0, 1, 0, 1]
1381            genInC = ["", "", "", ""]
1382            genInD = ["", "", "", ""]
1383        } else if (no_inputs.value == 3) {
1384            genInA = [0, 0, 0, 0, 1, 1, 1, 1]
1385            genInB = [0, 0, 1, 1, 0, 0, 1, 1]
1386            genInC = [0, 1, 0, 1, 0, 1, 0, 1]
1387            genInD = ["", "", "", "", "", "", "", ""]
1388        } else if (no_inputs.value == 4) {
1389            genInA = [0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1]
1390            genInB = [0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1]
1391            genInC = [0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1]
1392            genInD = [0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1]
1393        }
```

- sets the truth table arrays based on the number of inputs chosen,
- This allows flexibility of the truth able and prevents displaying unrelated input values on the truth table.
- They are predefined and count from 1-15 (in binary), check lines 1388-1393.
- Maybe hiding the whole column might help, other than defining other inputs as empty arrays.

7. FINAL OUTPUT DECIDER LOGIC

```
1410            /*******************set main output (using the last row with an output)********************/
1411
1412            if (outputR6 == null) { //if row 6 has no output, then check row 5
1413
1414                if (outputR5 == null) { //if row 5 has no output, then check row 4
1415
1416                    if (outputR4 == null) { //if row 4 has no output, then check row 3
1417
1418                        if (outputR3 == null) { //if row 3 has no output, then check row 2
1419
1420                            if (outputR2 == null) { //if row 2 has no output, then check row 1
1421
1422                                if (outputR1 == null) {//if row 1 has not output, set the output to null
1423                                    output = null
1424                                } else { //end if row 1
1425                                    output = outputR1
1426                                }
1427                            } else { //end if row 2
1428                                output = outputR2
1429                            }
1430                        } else { //end if row 3
1431                            output = outputR3
1432                        }
1433                    } else { //end if row 4
1434                        output = outputR4
1435                    }
1436                } else { //end if row 5
1437                    output = outputR5
1438                }
1439            } else { //end if row 6
1440                output = outputR6
1441            }
```

- this checks each row separately if it has an output, starting at the last row.
- If it doesn't it checks the prev row, it does this till it finds a row with an output,
- If no row has an output, it set the final **output** to null.
-

8. INPUT SELECTION CHECKER

```
1496        //diplay red for the column that is not picked
1497        if (aCheck.includes(1)) {
1498            console.log("A is Chosen");
1499            colA.style.backgroundColor = "transparent"
1500        } else {
1501            console.log("A is not chosen");
1502            colA.style.backgroundColor = "red"
1503            if (no_inputs.value == 4) {
1504                genInA = ["", "", "", "", "", "", "", "", "", "", "", "", "", "", "", ""]
1505            } else if (no_inputs.value == 3) {
1506                genInA = ["", "", "", "", "", "", "", ""]
1507            } else if (no_inputs.value == 2) {
1508                genInA = ["", "", "", ""]
1509            }
1510        }
```

- now those checking arrays come in play
- If a certain input is no chosen it indicate by having a red background
- Lines 1503-1509, hides the values of that certain input from the truth table,
- Even if the user selected number of inputs to be 4, but if C/D are not chosen, they will not show on the truth table.
- Hiding the whole column is not wise on this case as they user might have mistakenly not selected that certain input.

9. DOM OPERATIONS (GENERATING THE HTML TRUTH TABLE

```
1565    //display the output
1566    for (let i = 0; i < genOut.length; i++) {
1567        tbody.innerHTML += `
1568        <tr class="tt_row" id="row${i + 1}">
1569          <td class="tt_data colA">
1570            ${genInA[i]}
1571          </td>
1572          <td class="tt_data colB">
1573          ${genInB[i]}
1574          </td>
1575          <td class="tt_data colC">
1576          ${genInC[i]}
1577          </td>
1578          <td class="tt_data colD">
1579          ${genInD[i]}
1580          </td>
1581          <td class="tt_data colE">
1582          ${genOut[i]}
1583          </td>
1584        </tr>
1585        `
1586    }
```

- now the most important part, showing the final truth table to the user,
- This loops over the generated output array and pushes the html structure to the html document.
- How each cell, the values are taken from the generated arrays, e.g., genInA = []
- And finally, when this is done, the canvas function is from the canvas.js document is called.
- Instead of exporting the functions, they can be passed as parameters when calling the function.