Assignment 3

Recommended readings:

- Lecture slides as starting literature
- MDN Links within slides for details, especially:
  - setInterval
  - clearInterval
  - replace
  - checkValidity

Note: All exercises must be solved with plain JavaScript and CSS not using additional libraries or frameworks.

Exercise 1 – DOM Operations

Given the following HTML code:

```
<body>
     <a href= https://en.wikipedia.org target="_new">
          <img src=" https://cdn4.iconfinder.com/data/icons/country-flag-
1/744/United_Kingdom-128.png"></a>
     <a href=" https://de.wikipedia.org" target="_new">
          <img src=" https://cdn2.iconfinder.com/data/icons/country-flag-
1/744/Austria-128.png"></a>
     <a href=" https://ja.wikipedia.org" target="_new">
          <img src=" https://cdn4.iconfinder.com/data/icons/country-flag-
1/744/Japan-128.png"></a>
</body>
```

Embed this JavaScript in your page. Open the page in your browser, and explain each line of the output in detail:

```
var link = document.getElementsByTagName("a")[1];

console.log(link.attributes["href"].nodeValue);
console.log(link.getAttributeNode("href").nodeValue);
console.log(link.href);

console.log(link.firstChild);
console.log(link.attributes[0].nodeType);
console.log(link.firstChild.nodeType);
console.log(link.parentNode.nodeName);
console.log(link.lastChild.parentNode.nodeName);
console.log(link.attributes.length);
```

Moreover, explain the consequence of this JavaScript code:

```
var link = document.getElementsByTagName("a").item(2);
link.removeAttribute("target");
```

Exercise 2 – Calculating Glass Prize

Given the glass order in the file *glassOrder.html* adopted from Assignment 2.2. The file contains an additional price input element. Implement the following functionalities in the file calculate.js. Do not change the html file.

- Add a read only price input element to the glass order form with label *price*.
- Implement a function *calculate(width,height,thickness)* that returns the price based on the formula:
    - prize = width * height * thickness / 64 if the glass is coloured, or
    - prize = width * height * thickness / 100 if the glass is in a shade of grey (hint: R, G, and B channels have the same value for shades of grey).

  All inputs are assumed to be in cm. You may need to convert the values accordingly.
- Implement an *update()* function that calculates the price and sets the value of the price input field accordingly if all required inputs are given and have valid values. Use the *checkValidity()* function for checking. If no valid values are given, the input field should show a hyphen (-).
- Attach the update function as an event handler for the relevant input elements such that update() is called whenever a value is changed.

Exercise 3 – Making a HTML Page Editable

Given any HTML document with sections and one h1 heading element as a child of each section. One example is the document of Exercise 2.4. Write a JavaScript *editable.js* fulfilling the following requirements, when referenced from the HTML document:

- Your code adds a form containing a select box, a text input, a textarea and an initially disabled submit button after the last section.
- The select box allows the user to select any section of the document based on the heading of the section. After selection, the text input element shows the heading of the selected section and the textarea shows the innerHTML of the selected section and the button is activated.
- When the button is clicked the content of the selected section is updated with the data of the input element and the textarea. If the heading is changed in the textarea and in the input field, only the value of the input field applies. Be sure to correctly handle cases where the user removes the heading element in the HTML source. Also be sure to update the content of the select box after a heading is changed.

Do not change the HTML document except adding the reference to your JavaScript file!

Exercise 4 – Image Gallery

Given the picture gallery from the lecture slides found in the folder *gallery*. Extend the gallery such that when a picture is clicked and shown in full window mode the user has controls for:

- “*previous*” showing the previous picture. If not available, the last one.
- “*next*” showing the next image. If not available, the first one.
- “*play/pause*” implementing an automatic slideshow showing the pictures in an endless loop until *pause* or *exit* is clicked. The duration for showing each picture should be equal to the number of seconds specified in an input field next to the buttons (default value: 3 seconds).
- “*exit*” for stopping an eventually running slideshow and closing the full window mode.

Important: Your script must be working with any HTML content following the format of the HTML document of the lecture.

Exercise 5 – Puzzle Game in JavaScript

The folder *puzzle* contains the HTML document *puzzle.html*. The page shows a grid with 4 fields and a list of 4 images. The goal of the game is to move each image of the list to the correct grid cell for solving the puzzle.

Moving images should be realized like in the lecture slides. If the user puts an image onto a field in the grid and clicks (the mouse position is the relevant position), then the image is removed and set as the background of the specified grid cell. If there is already another picture in the grid cell, then the previous picture is moved back to the list of pictures. When all pictures are in the correct position of the grid, the user has won and a message is shown; otherwise, a message of wrong placement is shown.

Note: The “grid” is actually realized by the position of some divs. The solution is encoded in the data attribute *“data-result”* of each div. Your solution should not be restricted to the grid. It should be sufficient that each picture is moved to the div with the matching data-result attribute.

While you should apply the general principal of the lecture for moving the images to the grid you can change the code such that the mouse is not positioned above the currently moved image during moving. This makes is easier to interact with the grid.