# Assignment 5

## Exercise 1 – Events

What is event bubbling and event capturing? How can you specify what method to use? How can you stop the propagation of events? What is the difference between `this`, `event.target` and `event.currentTarget`? What happens if parent elements have event listeners attached? What happens if parent and child elements have event listeners attached? Demonstrate the different behaviour with the following examples:

a) Extend the example in `ex1/events.html` for demonstrating bubbling, capturing and `stopPropagation` with event listeners defined on the outer and additionally on the inner div. Create multiple files for demonstrating the various options.

b) The file `ex1/eventsParameter.html` contains the function `clicked(e,mode)`. Use this function as the event listener for the outer and inner div. However, now you need to pass the parameter `mode = 'log'` in order to create console output. How can you pass parameters to functions used as event handlers?

c) The script in the file `ex1/events2.html` should display the first name and last name of the person as content of the corresponding div, when the div is clicked. Unfortunately, it does not work. Explain why this happens and provide two solutions:
   a. Using an arrow function for calling the actual event handler [ MDN ].
   b. Using the `bind()` method [ MDN ].

## Exercise 2 – JSON & HTTP Requests

This exercise only works when using a web server. Therefore, place the provided folder `ex2` in the document root of your web server (see Assignment Sheet 1).
The folder contains the files `gallery.html`, `gallery.js` and `gallery.json`.
What is an asynchronous request? How does it differ from a synchronous request? How are asynchronous requests supported with JS´s XMLHttpRequest?
Implement the method `loadImages(domElement, jsonURI)` in `gallery.js` that requests the file `jsonURI` via an asynchronous XMLHttpRequest. For each image in the JSON file, a div element within `domElement` should be created. Within each div there should be an image element with the image url as source. Below the image there should be the description.

Test the implementation with the page `gallery.html`.

## Exercise 3 – Working with Promises

This exercise only works when using a web server. Therefore, place the provided folder `ex3` in the document root of your web server. The folder contains the files `galleryA.htm`, `galleryA.js`, `galleryB.htm`, `galleryB.js` and `gallery.json`. In the subfolder `details`, there is one json file containing the description of each image.

a) Write a method `loadImages(domElement, jsonURL)` in `galleryA.js` that requests the file `jsonURL` via the JS fetch API.
The json file contains a `descURL` property for each image. The `descURL` refers to another json file that contains one single image description.

   Like in Assignment 2, create a div for each image containing the image element and the description. However, now the description needs to be loaded with an additional fetch request from the description json file of each picture.
   In case some description file cannot be opened the description field should have the value “error, description could not be loaded”.
   The order of images shown on the page `galleryA.html` must be the same as in the gallery.json file.
   **For part A implement the requests using “then” and “catch” of promises.**

a) Create an alternative implementation and store it in `galleryB.js`. In this implementation use async functions, await and try / catch blocks.

## Exercise 4 – Your own Promises

The folder `ex4` contains the file `ex4.html`. The HTML file contains the buttons start, accept and reject. When the user clicks on the start button, the method `handleWithPromise` is called already including call-back functions for accept and reject. Implement the function `handleWithPromise`. The function should first set the status text to “Promise pending. Please click on a button to accept or reject the promise.” As soon as the user clicks on the accept button, the promise should be accepted. When the user clicks on the reject button the promise should be rejected.

## Exercise 5 – ES6 classes

What are ES6 Classes? How do they differ from implementing classes via constructors and prototypes? Implement the class hierarchy from Exercise 4 in Assignment sheet 4 using ES6 classes.