1. Asynchronous Programming:
    - structured using callback functions
    - in current versios of JS there are:
            - callbacks
            - promises
            - async function
    - not the same this as concurrent or multi-threaded
    - JS code is generally single-threaded

    ```
    console.log('Hello');

    setTimeout(function(){
        console.log('Goodbye');
        }, 2000);

    console.load('Hello again!');
    ```

    1.1. Callbacks:
        - function passed into another function as an argument
        - then invoked inside the outer function to complete some kind of routine or action

2. Promises Basics: Objects holding asynchronous operations:
    - a promise is an asynchronous action that may complete at some point and produce a value
    - states:
            - pending - operation still running(unfinished);
            - fulfilled - operation finished(the result is available);
            - failed - operation failed(an error is present)
    - promises use the Promise class
    - promises can be resolved, pending or rejected

    ```
    new Promise(executor)
    ```

3. AJAX & Fetch API: connecting to a server via fetch API
    - AJAX - asynchronous JS and XML
    - background loading of dynamic content/data
    - examples of AJAX usage:
        - partial page rendering - load HTML fragment and show it in a
        - JSON service - loads json object and displays it

    - The FETCH API:
        - allows making network requests
        - uses Promises
        - enables a simpler and cleaner API

        ```
        fetch('./api/some.json')
            .then(function(response) {...})
            .catch(function(err) {...})
        ```

        - the response of a fetch() request is a Stream object
        - the reading of the stream happens asynchronously
        - when the json() method is called, Promise is returned
        - the response status is checked (should be 200) before parsing the response as JSON

4. ES6 Async/Await:
    - Async function returns a promise that can await other promises in a way that

looks synchronous
    - Contains "AWAIT" expression that:
        - is only valid inside async function
        - pauses the execution of that function
        - waits for the promise's resolution