CTD
Intro
Week 14

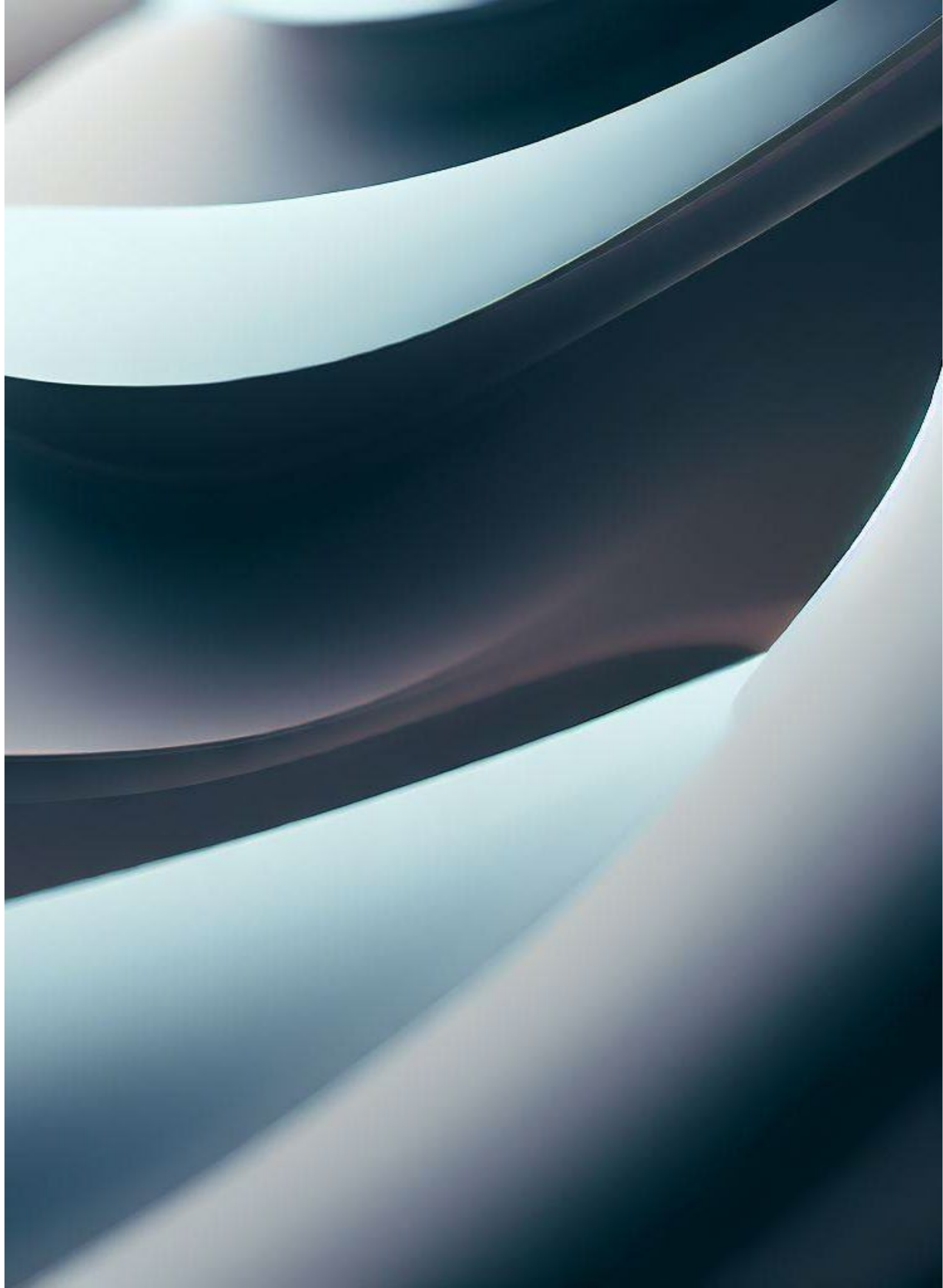
Asynchronous Programming

Promises

Forms

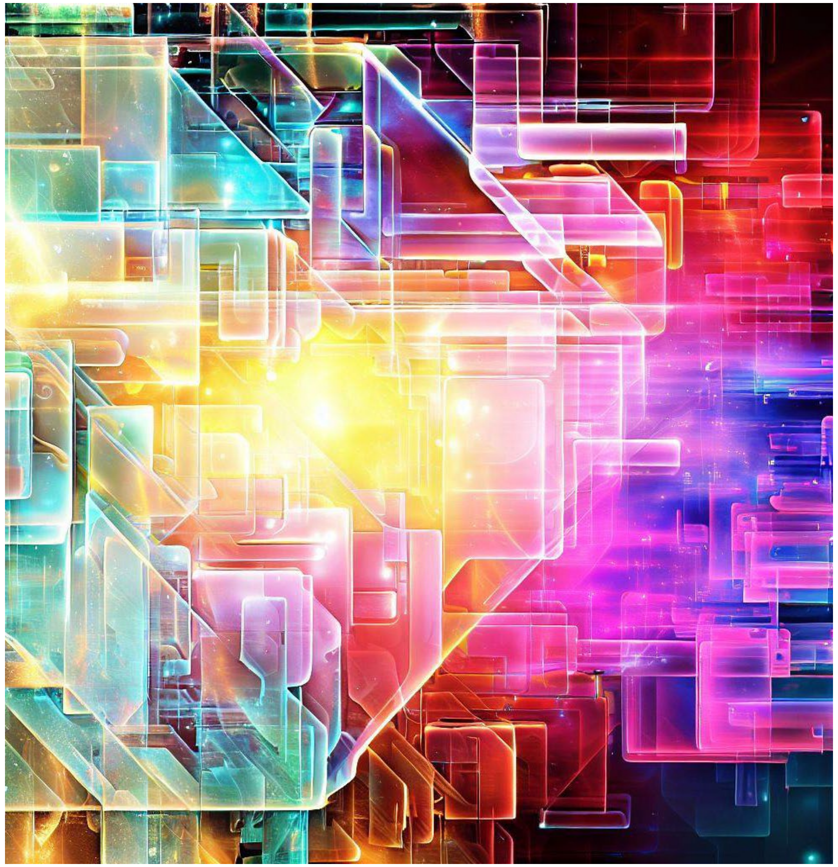
Forms

- Collect data from the user of a web page
- Originally designed to send data to the server
  - Many now interact with javascript in the client
    - As in this lesson
- Elements
  - <form> - contains the form
  - <label> - describe field(s)
    - for="idOfInputElement"
  - <input type=…>
    - Type of input, e.g. text, textbox, select, radio…
    - Validation, e.g. required=true
    - id matching label's for property



Asynchronous Javascript / Concurrency

- Ascychronous execution is powerful tool in javascript
  - There are many javascript functions which trigger asynchronous callback functions.
    - Such as addEventListener, setTimeout
- Concurrency is not (necessarily) parallelism
  - Javascript only supports a single thread of execution
  - The order of execution of asynchronous callback functions is not known
- When an event triggers a callback it is placed in a queue for execution
- When index.js is loaded, the code executes, possibly setting up asynchronous callback functions, such as event listeners.
  - After that individual callback functions are triggered asynchronously



Promises

- Convenient way to handle asynchronous execution
  - States: pending, fulfilled, rejected
- ’then’ method specifies callback function(s)
  - resolve, reject
  - Asynchronous, executed on state change
- Chaining
  - myPromise.then().then()… .catch()
  - Next ‘then’ executes after previous ‘then’ resolve/reject
  - ‘catch’ can be chained to handle errors
- Handle asynchronous results inside callback functions
  - Avoid the temptation to (try to) use await to make behavior synchronous
  - Await can only be used inside async functions
  - Async functions themselves behave asynchronously

Hints on hiding an empty list

- Initially empty
- Becomes non-empty asynchronously
- Can become empty again asynchronously
- How to temporarily hide an element
  - Boolean ‘hidden’ property
  - Vs setting display to none

Hints on implementing an edit button

- Bring the form back
- Store and then populate the current values
- Top-level functions to make edit and remove buttons
  - Defining their callbacks
  - They apply to their parentNode

Q&A and Demo