Tutorial 4: Modern JavaScript

Objectives

- Practice modern JavaScript techniques, including:
  - Populate & display data by creating HTML elements
  - Keyboard events
  - Classes
  - Different meaning of `this` and effect of `bind()`

Tutorial Exercises

Exercise 1: Cards (15 mins)

Create folder `tut04/cards/`, with 3 files:

- `card-sources.js`: contains an array of links for card images
- `cards.js`: populate cards with links to images from `card-sources.js`
- `index.html`: html for card board with references to .js files

Task 1: Prepare cards

- In `card-sources.js`, create an array of links for at least 5 cards. You can start with this one:

http://acbl.mybigcommerce.com/52-playing-cards/

Task 2: Show card board

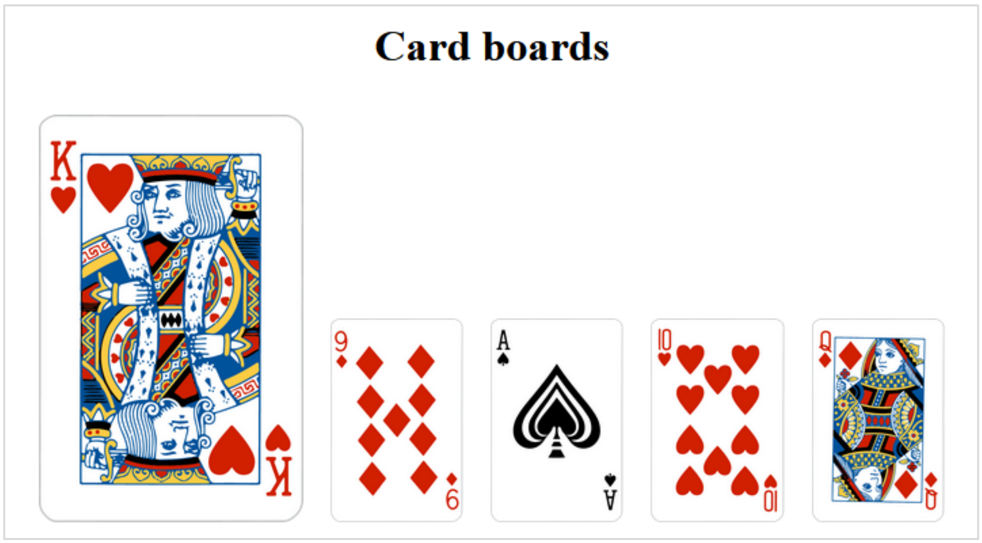
- Loop the array of images to create img elements add to the board view and display.

Task 3: Select card

- When user choose a card, it's enlarged by double of height.

Task 4: Change card

- User can click another card, but only 01 card enlarged at a time.



Exercise 2: Flash cards (45 mins)

Download & extract the *flashcards-starterpack* then rename to `tut04/flashcards/`, and continue with the tasks below.

- `words.js`: contains a dictionary of words
- `script.js`: populate words from `words.js` into flash cards

Task 1: Click events - Flip card

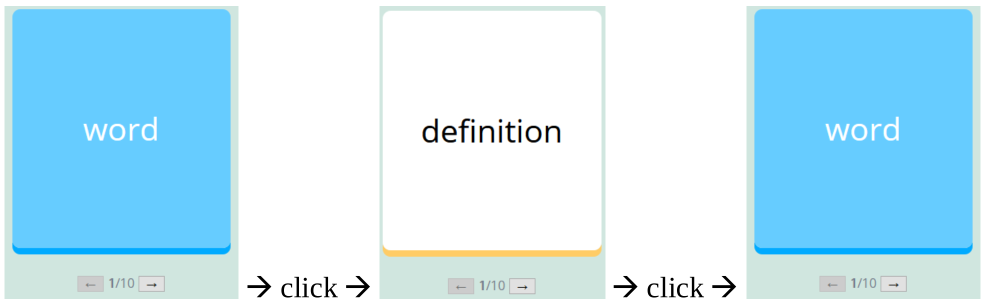
Run **index.html** in the browser, you will see a flashcard with a side for word & another for definition; however, normally you can see only 01 side at a time.

Hint: Remember the CSS class `hidden`?

Your task is to show a side & flip to another side when user clicks on the card.

- Listen for click event on flashcard boxes
- Create a handler function named `flipCard(event)` to toggle show/ hide the suitable side

Hint: CSS classList toggle class may be helpful



Task 2: Populate the cards

Loop the array of KOREAN words and create suitable HTML elements to display as a flashcard, add to the flashcard container and display.

- Create a function named `createCard(word, definition)` to return HTML card element from specified word & definition
- Create a function named `populateCards(container)` to loop the KOREAN words, invoke createCard to create card elements, add them into the container to display & return cards as array for navigating
- Update status bar with correct number of cards

You will see many cards at this step, use CSS class `hidden` to show only the card at the specified index (first card by default)

- Create a function named `setIndex(index)` to: check if valid index → hide card at the current index → show card at the specified index → enable/ disable navigating buttons based on the index

Task 3: Click events – navigate between flash cards

Create 2 handlers (functions) for click events on button: **Previous** & **Next** to navigate between cards.

Note: if current card is at index 0, button **Previous** should be disabled. Similar to button **Next** in case of showing the last card.

Hint: `previousSibling()` and `nextSibling()` may be useful.

Task 4: Keyboard events – navigate between flash cards

Handle keyboard events on 2 arrow keys (left & right) to navigate between cards.

Hint: Use the two handlers created from *Task 3*.

Exercise 3 Flash cards OOP (45 mins)

Create folder `tut04/flashcards-oop/`, and refactor flashcards program into classes.

Task 1: Classes

Create 3 file `flashcard.js`, `statusbar.js` and `app.js` for these 3 classes respectively:

- `FlashCard`: represents a flash card with word and definition.
- `StatusBar`: represents the status bar
- `App`: represents the application. *App* has a list of *FlashCard*, and the status bar.

Note: `main.js` is now to start the *App* with required parameters.

Hint: use the design of pattern 1 (pass the container element inside the class constructor)

Task 2: Communication between App & StatusBar? → NEXT WEEK

When user click on *StatusBar*, it need to notify the App to display the correct *FlashCard*. Use ***custom events*** to facilitate this.