Dom and DOm manipulation(study docs)

**Document object model:**

**Structured representation of html document** ,allows javascript to access html elements and styles to manipualte them

change text,html attributes,even css styles

dom is automatically created by browser as soon as page loads

in this tree each html element is an object

document is an  special type of object in dom

it acts as entry point to dom

it has a spl function called querySelector by which we can select any elements in the html document

so basicaly dom is represntation of html document

Steps to follow:

step 1:select the element where the event is going to occur

step2:now we can call addeventListen method on that element

step3:we neeed to pass in the type of the event in this case its a simple click

step4:as reaction to the click event we should tell the event listener what it should do using a function(event handler)

function can be passed as paramter

that function will be called as soon as the event occurs(*callback*)

**every dom element is an object in js**

console.log(typeof document.querySelector('.number')); //object

each object has its own properties and methods,using that we can manipulate the dom

in JavaScript, each DOM (Document Object Model) element is represented as an object. When you interact with elements on a web page using JavaScript, you are typically working with these objects. These objects expose various properties and methods that allow you to manipulate and interact with the elements on the page.

For example, when you use **document.querySelector** or **document.getElementById** to select an element, you get back a JavaScript object that represents that DOM element. You can then access properties like **innerHTML**, **textContent**, **style**, and methods like **addEventListener** on these objects to read or modify their content, appearance, and behavior.

**manipulating styles in css**

  document.querySelector('body').style.backgroundColor = '#60b347';

   document.querySelector('.number').style.width = '30rem';

**manipulating input elements:**

**document.querySelector('.guess').value = '';**

**Modal Window:**

This JavaScript code sets up event listeners and handles interactions with a modal dialog. Here's a summary of the techniques and concepts used in the code:

1. 'use strict': This is a pragma that enforces stricter parsing and error handling in JavaScript.

2. Querying DOM Elements:

- `document.querySelector()`: Used to select elements with specific class names and store them in variables.

- `document.querySelectorAll()`: Used to select all elements with a particular class and store them in a NodeList.

Code:

const modal = document.querySelector('.modal');

const overlay = document.querySelector('.overlay');

const btnCloseModal = document.querySelector('.close-modal');

const btnsOpenModal = document.querySelectorAll('.show-modal');

console.log(btnsOpenModal);

3. Event Listeners:

- `addEventListener()`: Used to attach event listeners to DOM elements. Event listeners are set up to respond to user interactions.

Code:

for (let i = 0; i < btnsOpenModal.length; i++) {

  btnsOpenModal[i].addEventListener('click', openModal);

}

btnCloseModal.addEventListener('click', closeModal);

overlay.addEventListener('click', closeModal);

document.addEventListener('keydown', function (*e*) {

  // if (e.key === 'Escape') {

  //   console.log('esc was pressed');

  // }

  if (e.key === 'Escape' && !modal.classList.contains('hidden')) closeModal();

});

4. Functions:

- The code defines functions (`openModal` and `closeModal`) to handle actions when the modal is opened or closed. These functions modify the CSS classes of the modal and overlay elements.

const openModal = function () {

  console.log('button clicked');

  modal.classList.remove('hidden'); //dont have to use the . for classes here

  overlay.classList.remove('hidden');

};

const closeModal = function () {

  modal.classList.add('hidden');

  overlay.classList.add('hidden');

};

6. Event Object:

- The code utilizes the event object (e.g., `e` in the keydown event listener) to respond to specific events. In this case, it listens for the 'keydown' event to detect when the 'Escape' key is pressed.

document.addEventListener('keydown', function (*e*) {

  // if (e.key === 'Escape') {

  //   console.log('esc was pressed');

  // }

  if (e.key === 'Escape' && !modal.classList.contains('hidden')) closeModal();

});

7. Conditional Statements:

- Conditional statements are used to check if the 'Escape' key is pressed and if the modal is currently open. If both conditions are met, the `closeModal` function is called.

8. Keyboard Event Handling:

- The code uses the 'keydown' event to listen for key presses, specifically the 'Escape' key. When the 'Escape' key is pressed, it triggers the `closeModal` function to close the modal.

9. CSS Classes:

- The code adds and removes the 'hidden' class to show or hide the modal and overlay elements.

  modal.classList.remove('hidden'); //dont have to use the . for classes here

  overlay.classList.remove('hidden');

Overall, the code sets up event listeners to open and close a modal when buttons are clicked, and it also allows users to close the modal by pressing the 'Escape' key. The use of functions, event handling, and conditional statements makes this code interactive and user-friendly.

**Pig game:go through the code entirely**