**<html>**

-css link

-body-

-폼 : 로그인창에 들어가는 인풋은 2개 (입력창text/전송창submit)

-폼 : 메모창에 들어가는 인풋은 2개 (입력창text/전송창submit)

(폼 자체에 id를 달아줌)

(이때 메모창에 옵션으로 required : 빈칸이면 다음으로 넘어가지 않음 / placeholder : 힌트)

-div : 미리 들어가있는 메모창의 공간 + 메모입력시 들어갈 div 공간

-스크립트 : js 파일만큼 src로 연결

**<js> login, memo, background, clock 4가지 js로 분류 각 1~4로 명칭**

**(1).**

- const loginForm = document.getElementById("login-form"); (id가 login-form인 html 부분을 참조해 loginForm으로 정의)

- const loginInput = document.querySelector("#login-form input");

- const greeting = document.querySelector("#greeting"); (querySelector는 id 참조시 #붙여야하며 첫번째 태그를 들고와서 정의)

- const memoFormShow = document.querySelector("#memo-form");

-loginForm.addEventListener("submit",onLoginSummit);

(html의 로그인 부분인 login-form을 참조한 loginForm에서 submit 실행시 onLoginSummit 이라는 function을 실행시킨다.

- function onLoginSummit(event) { (위의 addEventListener로 발생조건을 매개변수 event로 둠)

event.preventDefault(); (event 발생시 – submit은 누르면 창이 새로고침되거나 새창이 연결되는데 그것을 막아줌)

const username = loginInput.value; (loginInput의 입력한 value 값을 username으로 명명)

greeting.innerHTML = `반갑습니다 ${username}님`; (id가 greeting인 곳에 `반갑습니다 ${username}님`의 형태로 html에 표시해줌)

loginForm.classList.add("hidden"); (event 발생시 미리 작성해둔 class인 hidden을 loginForm 적용시킨다.)

memoFormShow.classList.remove("hidden"); (event 발생시 미리 작성해둔 class인 hidden을 memoFormShow에 적용시킨다.)

}

**(2).**

-const memoBoard = document.querySelector("#memo-board"); (위와 동일 참조해서 명명)

-const memoForm = document.querySelector("#memo-form");

-const memoInput = document.querySelector("#memo-form input");

-const savememo = ["미리 만들어진 메모입니다","미리 만들어진 메모입니다2","한번 더 추가합니다"];

(배열로서 위의 3가지 문구를 savememo라 명명)

*-for* (let m of savememo) { (for문으로 savememo를 요소를 m으로서 들고옴)

    let memo = document.createElement("div"); (memo는 createElement(“div”)라는 기능을 띈다고 명명 = div를 생성한다)

   memo.innerHTML = m; (memo는 html에 m을 적는다.)

memoBoard.appendChild(memo); (메모의 각 문구요소들을 memoBoard부분으로 옮겨준다)

}

-memoForm.addEventListener("submit", memoFormSubmit); (memoForm의 submit 실행시 memoFormSubmit 시행)

-function memoFormSubmit (event) {

    -event.preventDefault();

    -console.log(memoInput.value); (그냥 console 확인용)

-let memo = document.createElement("div"); (memo는 div를 생성한다고 명명)

-memoBoard.appendChild(memo); (memoBoard에 memo라는 div를 추가해줌)

-memo.innerHTML = memoInput.value; (추가된 memo요소를 html에 추가)

-memoInput.value = ""; (memo를 html에 값 추가해준이후 창에 글 남아있지 않도록 공백으로 만들어줌)

-console.dir(memoInput); (그냥 확인용)

-memoInput.autofocus = true; (굳이 안넣어도 될듯한 옵션)

}

**(3).**

-const images = ["0.jpg","1.jpg"]; (각 이미지의 이름을 나열한 배열을 images가 정의)

-const choseImage = images[Math.floor(Math.random()\*images.length)]; (images의 배열수만큼의 랜덤수의 정수값만 빼와서 choseImage라 정의)

-const bodyBackground = document.querySelector("body"); (html의 body 참조를 bodyBackground라 정의)

-bodyBackground.style.background = `url(./img/${choseImage})`; (위의 bodyBackground를 style.background로 `url(./img/${choseImage})`

형식으로 choseImage요소의 이름에 해당하는 파일 모두를 들고옴

**(4).**

const clock = document.querySelector("#clock"); (id가 clock인 부분을 참조해서 clock으로 정의)

function getClock() { (get clock() 의 기능을 정의)

    const date = new Date(); (현재의 날짜,시간등을 가져와 date로 정의)

const hours = String(date.getHours()).padStart(2,"0"); (위의 정의된 date의 시간부분만 빼와 padStart로 두자리(공백은0으로채움)로 만듬)

const minutes = String(date.getMinutes()).padStart(2,"0") (위와 동일)

const seconds = String(date.getSeconds()).padStart(2,"0") (위와 동일)

clock.innerHTML = `${hours}:${minutes}:${seconds}`; (clock을 `${hours}:${minutes}:${seconds}` 형태로서 html에 기입)

}

getClock(); (창을 켜자마자 function이 적용되도록 함)

setInterval(getClock, 1000); (getClock function의 작용을 반복함 1000ms 만큼 (1000ms = 1초))