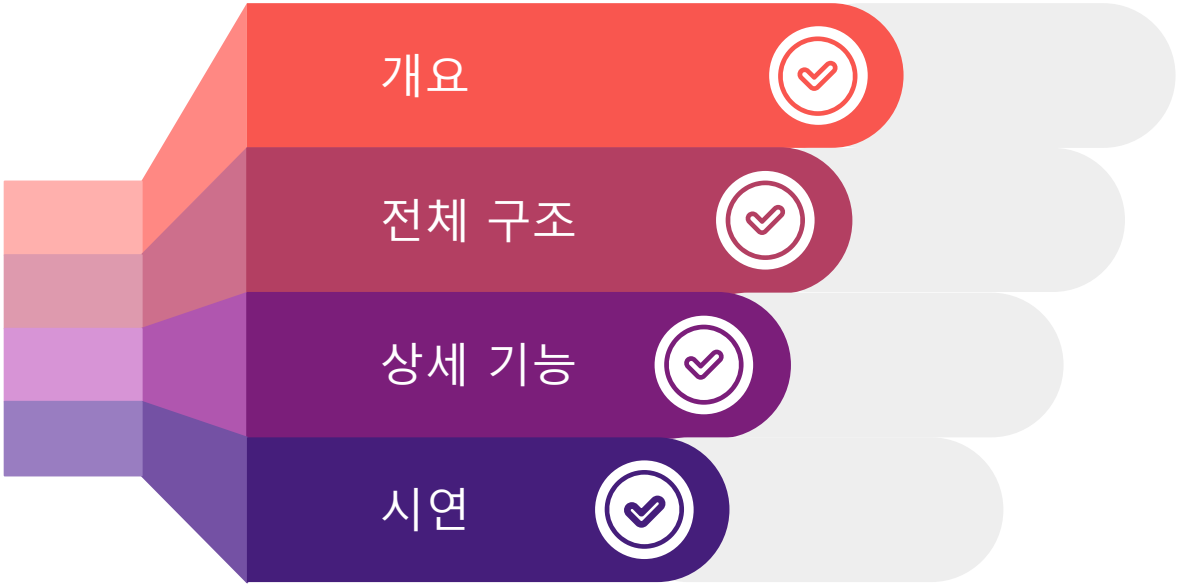
To-Do List

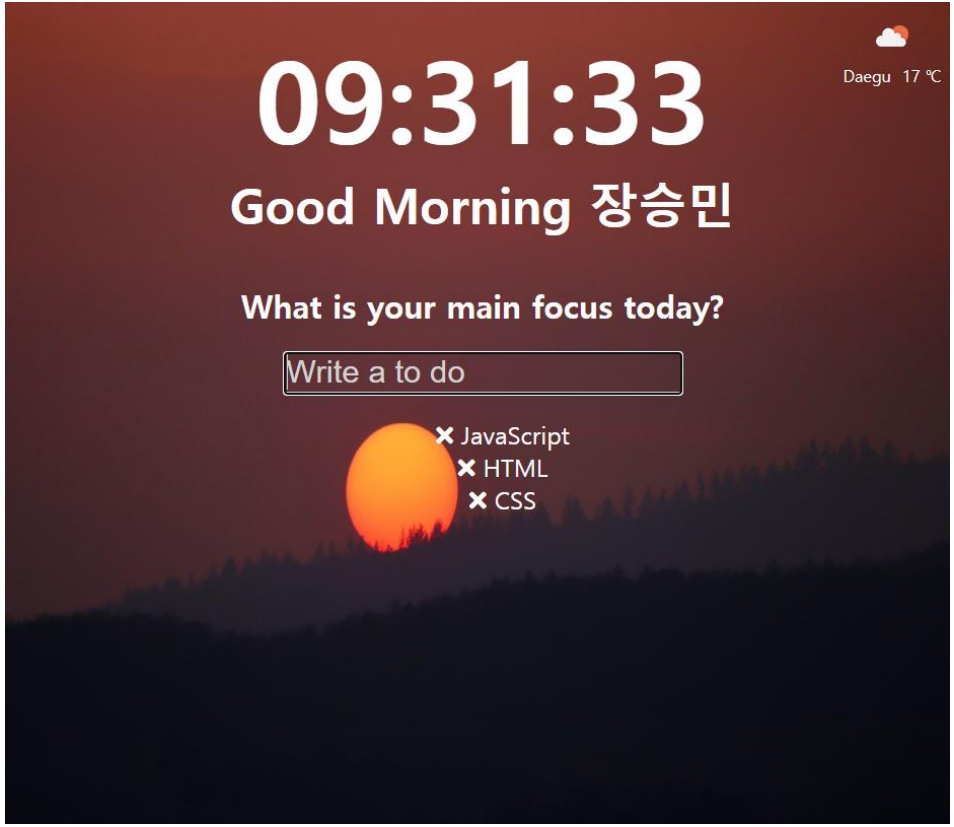
목차



개요 1

HTML, CSS, JavaScript

중요한 일, 사소한 일 등 해야 할 일들을 한눈에 볼 수 있게 기록할 수 있는 웹페이지를 제작하였습니다.



2 전체 구조

clock.js

weather.js

greeting.js

bg.js

todo.js

3 상세 기능

HTML

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="index.css">
    <link rel="stylesheet" href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css"
        integrity="sha384-AYmEC3Yw5cVb3ZcuHtOA93w35dYTsvhLPVnYs9eStHfGJvOvKxVfELGroGkvsg+p" crossorigin="anonymous" />
</head>

<body>
    <div class="js-clock">
        <h1>
            00:00
        </h1>
        <div class="weather">

            <img class="js-weather icon">
            <span class="js-weather place"></span>  <span class="js-weather temp"></span>
        </div>
    </div>
    <form class="js-form form">
        <input type="text" placeholder="What is your name?">
    </form>
    <h1 class="js-greetings greetings"></h1>

    <form class="js-toDoForm toDoForm">
        <h2>What is your main focus today?</h2>
        <input type="text" placeholder="Write a to do" />
    </form>
    <ul class="js-toDoList" style="list-style-type:none;">
    </ul>
    <script src="clock.js"></script>
    <script src="greeting.js"></script>
    <script src="todo.js"></script>
    <script src="bg.js"></script>
    <script src="weather.js"></script>
</body>

</html>
```

CSS

```css
body{
    background-color: #2c3e50;
}
.btn{
    cursor: pointer;
}
body{
    color: white;
}
.clicked{
    color: #7f8c8d;
}
.form,
.greetings{
    margin-top: 0;
    text-align: center;
    display: none;
}
.showing{
    display: block;
    font-size: 300%;
}

@keyframes fadeIn{
    from{
        opacity: 0;
    }
    to{
        opacity: 1;
    }
}


.bgImage{
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    z-index: -1;
    animation: fadeIn 0.5s linear;
}
.weather{
    position: absolute;
    top: 10px;
    right: 10px;
    text-align: center;
    width: 130px;
    height: 100px;
    font-size: medium;
}
.js-clock h1{
    text-align: center;
    font-size: 700%;
    margin-top: 20px;
    margin-bottom: 0;
}
.toDoForm{
    margin-top: 5%;
    text-align: center;
    font-size: 130%;
}
.toDoForm input{
    font-size: 150%;
    border-top-width:0 ;
    border-right-width:0 ;
    border-left-width:0 ;
    border-bottom: solid 2px lightgrey;
    /* border-bottom: white; */
    background-color: transparent;
    color: lightgray;
}
.toDoForm input::placeholder{
    color: lightgrey;
}
ul{
    font-size: 150%;
    text-align: center;
}
li{
    text-align: center;
}
```

JavaScript

clock.js

```
const clockContainer = document.querySelector('.js-clock'),
    clockTitle = clockContainer.querySelector('h1');

function getTime() {
    const date = new Date();
    const minutes = date.getMinutes();
    const hours = date.getHours();
    const seconds = date.getSeconds();
    clockTitle.innerText = `${
        hours < 10 ? `0${hours}`:hours}:${
        minutes<10?`0${minutes}`:minutes}:${
        seconds <10 ? `0${seconds}`:seconds}`;
}

function init() {
    getTime();
    setInterval(getTime, 1000);
}
init();
```

메인 화면에 현재 시각을 구현하는 스크립트

bg.js

```
const body = document.querySelector('body');

const IMG_NUMBER = 8;


function paintImage(imgNumber) {
    const image = new Image();
    image.src = `images/${imgNumber + 1}.jpg`;
    image.classList.add('bgImage')
    body.prepend(image);
}


function genRandom() {
    const number = Math.floor(Math.random() * IMG_NUMBER);
    return number;
}

function init() {
    const randomNumber = genRandom();
    paintImage(randomNumber);
}
init();
```

배경 이미지를 랜덤으로 구현하는 스크립트

greeting.js

```
const form = document.querySelector('.js-form'),
    input = form.querySelector('input'),
    greeting = document.querySelector('.js-greetings');

const USER_LS = 'currentUser',
    SHOWING_CN = 'showing';

function saveName(text){
    localStorage.setItem(USER_LS,text);
}

function handleSubmit(event){
    event.preventDefault();
    const currentValue =  input.value;
    paintGreeting(currentValue);
    saveName(currentValue);
}

function askForName() {
    form.classList.add(SHOWING_CN);
    form.addEventListener("submit",handleSubmit);
}
```

localStorge 에 사용자 이름을 저장하는 함수

입력된 사용자 이름을 localStroge에 저장하는 함수

입력 폼을 나타내는 함수

```
function paintGreeting(text) {
    form.classList.remove(SHOWING_CN);
    greeting.classList.add(SHOWING_CN);
    const date = new Date();
    const hour = date.getHours();
    var output ='';
    if(hour>5&&hour<12){
        output = 'Good Morning';
    }else if(hour>=12&&hour<17){
        output ='Good Afternoon'
    }else if(hour>=17&&hour<21){
        output='Good Evening';
    }else{
        output='Good Night';
    }
    greeting.innerText = `${output} ${text}`;
}

function loadName() {
    const currentUser = localStorage.getItem(USER_LS);
    if (currentUser === null) {
        askForName();
    } else {
        paintGreeting(currentUser);
    }
}

function init() {
    loadName();
}
init();
```

입력 폼을 지우고, 사용자 이름을 출력하는 함수

localStorge에 저장되어있는 사용자 이름이 없을 시 입력 폼을 나타내 입력을 받고, 저장된 값이 있을 시 사용자 이름을 출력함

weather.js

```
const weather_temp = document.querySelector('.temp');
const weather_place = document.querySelector('.place');
const weather_icon = document.querySelector('.icon');

const API_KEY ="a1770fc6f09f37a0b8ce75b814f238a1";
const COORDS = 'coords'



function getWeather(lat,lng){
    fetch(
        `https://api.openweathermap.org/data/2.5/weather?lat=${lat}&lon=${lng}&appid=${API_KEY}&units=metric`
        ).then(function(response){
            return response.json();
        }).then(function(json){
            const temperature = json.main.temp;
            const place = json.name;
            weather_temp.innerText =`${temperature} ℃`;
            weather_place.innerText =`${place}`;
            weather_icon.src = 'http://openweathermap.org/img/wn/'+ json.weather[0].icon+ '.png';
        })
}

function saveCoords(coordsObj){
localStorage.setItem(COORDS,JSON.stringify(coordsObj));
}
```

API로 날씨정보를 불러오는 함수

위치정보객체를 문자열형태의 JSON 형식으로 localStorage에 저장하는 함수

```
function handledGeoSuccess(position){
    const latitude = position.coords.latitude;
    const longitude = position.coords.longitude;
    const coordsObj = {
        latitude ,
        longitude
    };
    saveCoords(coordsObj);
    getWeather(latitude,longitude);
}

function handledGeoError(){
    console.log('Cant access geo location');
}

function askForCoords(){
    navigator.geolocation.getCurrentPosition(handledGeoSuccess,handledGeoError);
}

function loadCoords(){
    const loadedCords = localStorage.getItem(COORDS);
    if(loadedCords === null){
        askForCoords();
    }else{
        const parsedCoords = JSON.parse(loadedCords);
        getWeather(parsedCoords.latitude,parsedCoords.longitude);
    }
}


function init(){
    loadCoords();
}
init();
```

위치 정보 제공에 동의 했을 시 위치 정보를 저장하는 함수

위치 정보 제공을 거절 했을 시 실패 로그 작성

위치 정보 제공 동의 여부를 묻는 함수

localStorage에 저장된 위치 정보가 없다면 위치 정보 제공 동의를 묻고, 저장된 정보가 있다면 해당하는 위치의 날씨정보를 나타내는 함수

todo.js

```
const toDoForm = document.querySelector('.js-toDoForm'),
    toDoInput = toDoForm.querySelector('input'),
    toDoList = document.querySelector('.js-toDoList');

const TODOS_LS = 'toDos';

let toDos =[];
const XBTN = 'fas fa-times';


function deleteToDo(event){
    const btn = event.target;
    const li = btn.parentNode;
    toDoList.removeChild(li);
    const cleanToDos = toDos.filter(function(toDo){
        return toDo.id !== parseInt(li.id);
    })
    toDos = cleanToDos;
    saveToDos();
}

function saveToDos(){
    localStorage.setItem(TODOS_LS,JSON.stringify(toDos));
}
```

해당하는 todo 목록을 삭제하는 함수

todo 목록을 문자열 형태의 JSON 형식으로 localStorge에 저장하는 함수

```
function paintToDo(text){
    const li = document.createElement('li');
    const delBtn = document.createElement('i');
    const span = document.createElement('span');
    const newId = toDos.length +1

    delBtn.className=XBTN;
    delBtn.addEventListener('click',deleteToDo);

    span.innerText =' ' + text;
    li.appendChild(delBtn);
    li.appendChild(span);
    li.id = newId;
    toDoList.appendChild(li)
    const toDoObj ={
        text:text,
        id:newId
    }
    toDos.push(toDoObj);
    saveToDos();
}

function handleSubmit(event){
    event.preventDefault();
    const currentValue = toDoInput.value;
    paintToDo(currentValue);
    toDoInput.value='';
}

function loadToDos() {
    const loadedToDos = localStorage.getItem(TODOS_LS);
    if (loadedToDos !== null) {
        const parseToDos = JSON.parse(loadedToDos);
        parseToDos.forEach(function(toDo){
            paintToDo(toDo.text);
        });
    }
}

function init() {
    loadToDos();
    toDoForm.addEventListener('submit',handleSubmit)
}
init();
```

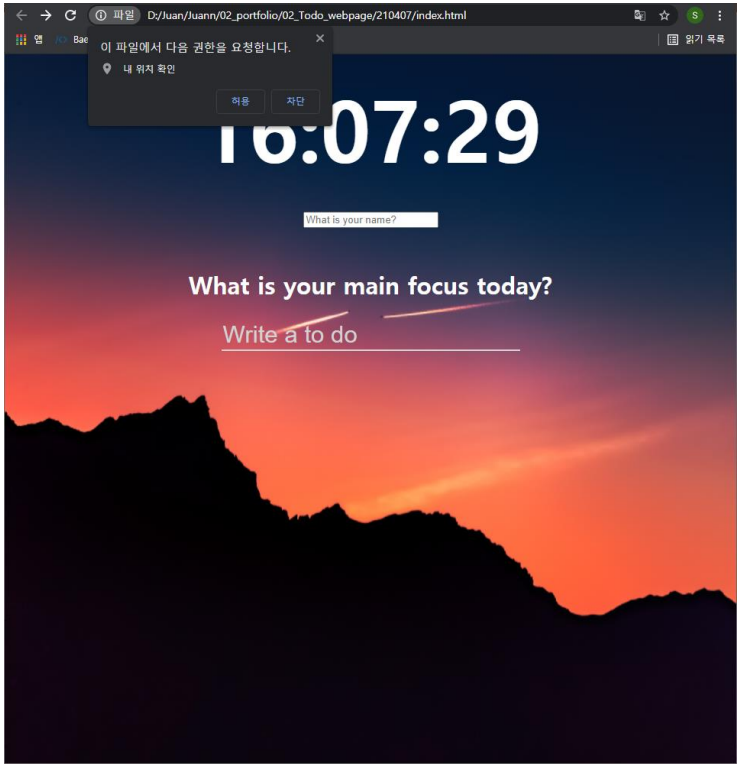
모든 todo 목록 들을 화면에 출력하는 함수

입력한 todo 목록을 제출 화면으로 출력하기 위한 함수로 전달하는 함수

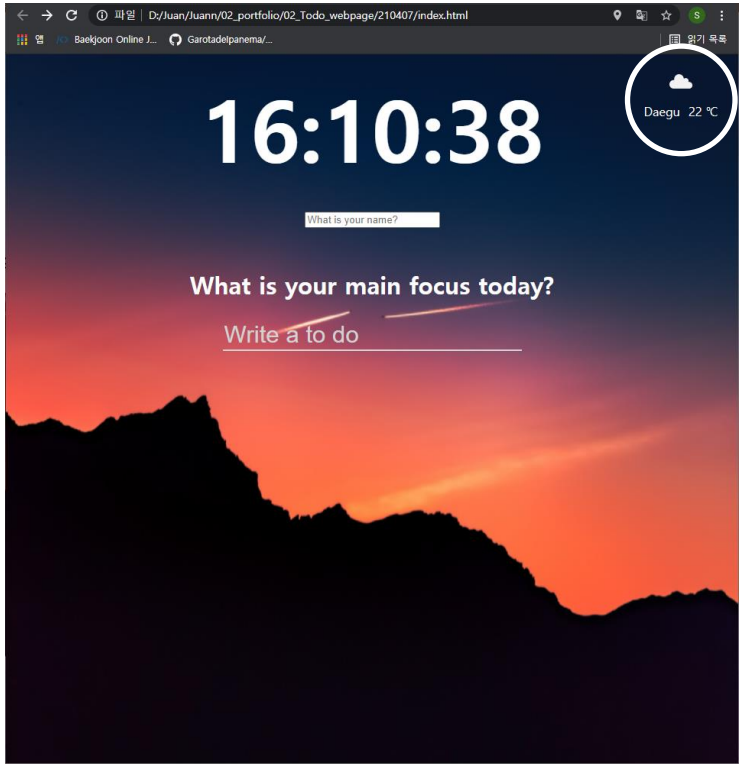
메인화면 로드 시 저장된 todo 목록들이 있으면 모든 todo 목록들을 출력하는 함수

시연 4

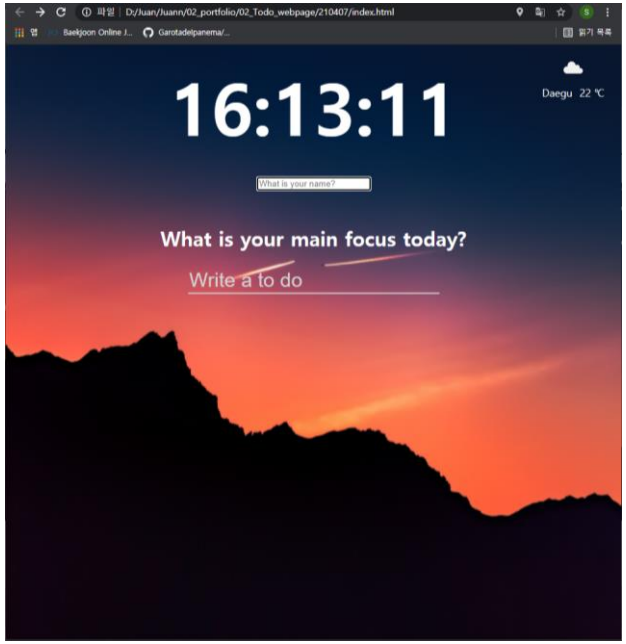
날씨 정보 API를 위해 위치 정보 제공 동의 요청



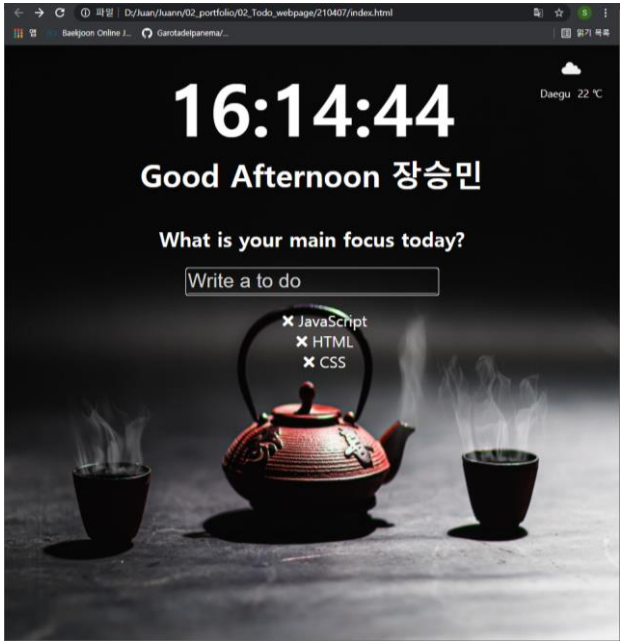
위치 정보 제공 동의 후 날씨 정보 출력



사용자 이름 저장



현재 시각에 따른 인사말 출력 – 배경화면 랜덤 변경 – todo 목록 저장



localStorge에 저장되어있는 todo 목록들, 사용자 이름, 현재 위치 좌표

Key	Value
toDos	[{"text":"JavaScript","id":1},{"text":"HTML","id":2},{"text":"CSS","id":3}]
currentUser	장승민
coords	{"latitude":35.8806693,"longitude":128.6215081}

THANK YOU