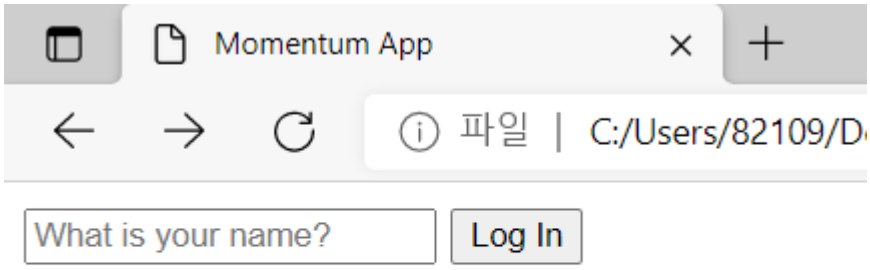
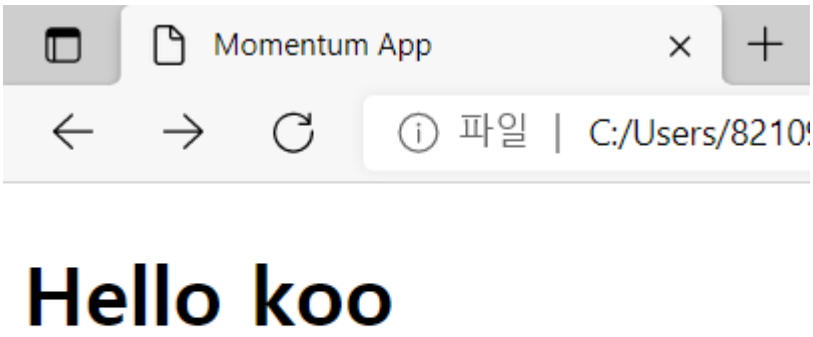
Vanilla-JS day5

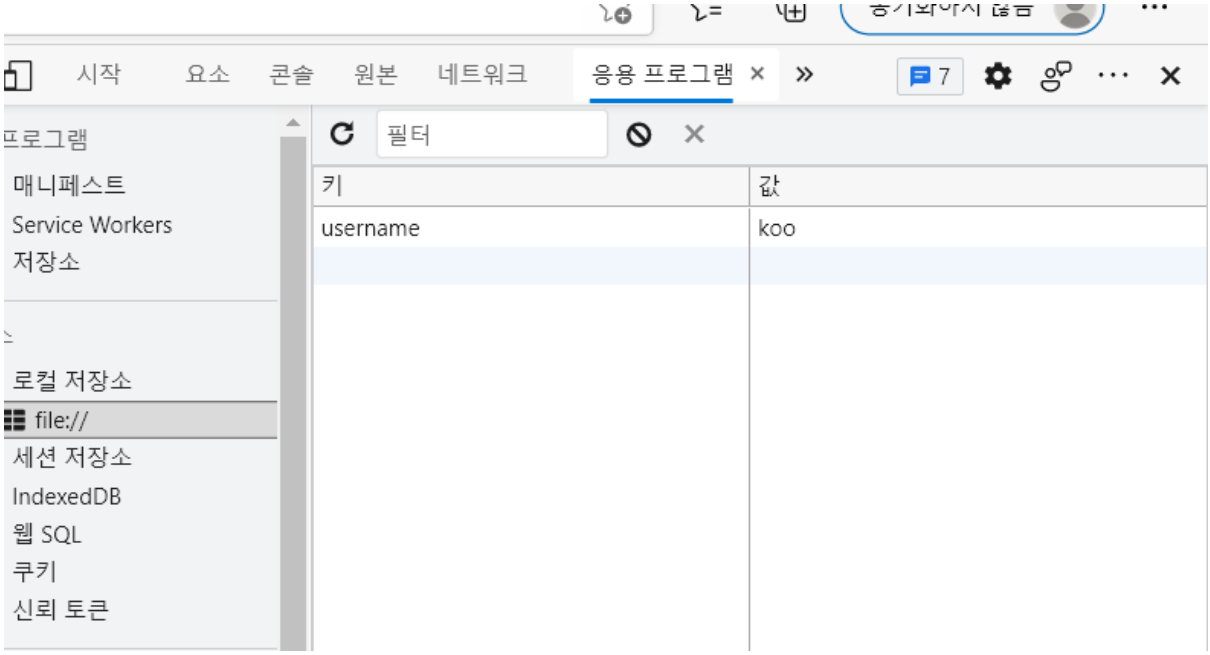
Login-Form 완성 예시



- 사진과 같이 유저의 이름을 입력받는 Input창과 Log-In button을 생성한다.



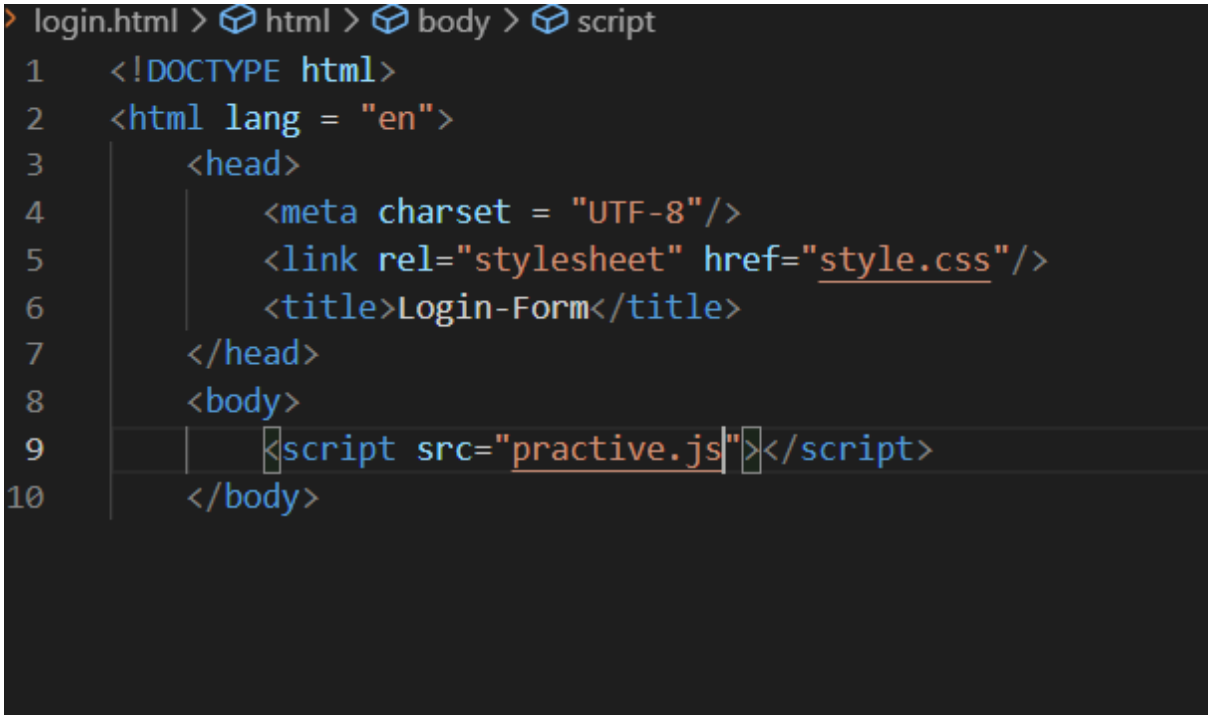
- 사용자가 input에 이름을 입력하고 Enter 혹은 Button 클릭시 위와 같이 문구를 출력한다.



- 입력한 정보는 브라우저의 Local Storage에 key와 value로 저장된다.

완성 예시와 같은 기능 구현하기

1. HTML 기본 구조 생성



Login-Form 생성에 앞서서 HTML 파일과 Css, Javascript 파일을 연결하는 기본적인 HTML구조를 만든다.

- <head> 태그에서 <link> 태그를 통해 css 연결
- <body> 태그에서 <script>태그를 통해 Javascript를 연결

2. Input과 Button 생성

완성예시 첫번째 사진과 같이 input과 button을 생성한다.

이때, input과 button을 Form 태그 안에 생성한다.

▼ Form 태그에 생성하는 이유

Form은 입력된 정보를 한번에 전달 한다. 이를 활용하여 입력된 사용자 정보를 한번에 어딘가로 전달하기 위해 Form을 사용.

```html
<!DOCTYPE html>
<html lang = "en">
    <head>
        <meta charset = "UTF-8"/>
        <link rel="stylesheet" href="style.css"/>
        <title>Login-Form</title>
    </head>
    <body>
        <form id ="login-form">
            <input required maxlength="15"
            type="text"
            placeholder="What is your name?"></input>
            <button>Log-In</button>
        </form>
        <script src="practive.js"></script>
    </body>
```

사진을 보면 <form>태그와 <input>, <button>이 추가된 모습을 볼수 있다.

이때, required max length="15" 속성을 사용하여 사용자가 입력을 안하거나 입력을 남용하는 것을 방지하였다.

3. Form태그의 submit 이벤트 방지

2번 과정을 마치면 브라우저에 아래와 같은 요소가 생성된다.

이때, 사용자가 이름을 입력하고 Log-In 버튼을 누르거나 Enter를 누르면

페이지가 새로고침된다.

왜냐하면 Form은 버튼을 누르거나 Enter를 누르면 **정보를 submit(제출) 후에 자동으로 새로고침하는 이벤트를 가지기 때문이다.**

이러한 이유로 인해 사용자가 입력하는 데이터가 손실되는 문제가 발생 함으로 Form의 새로고침 이벤트를 막아 주어야한다.

함수의 기본 기능을 막는함수는 preventDefault()이다.

```
JS practive.js > ...
1   const loginForm = document.querySelector("#login-form");
2
3   function onLoginSubmit(event){
4       event.preventDefault();
5       console.log("hi");
6   }
7
8   loginForm.addEventListener("submit", onLoginSubmit);
9
10
```

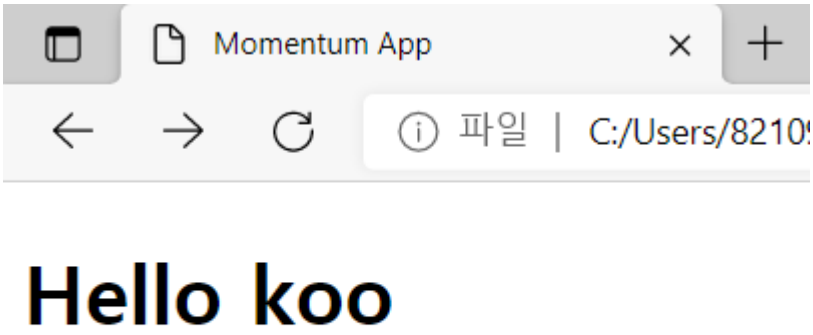
사진과 같이 form태그의 정보를 가져오고 form태그의 "submit" 이벤트

발생시 preventDefault() 함수를 실행함 으로써 "submit"가 가져오는

새로고침 이벤트를 방지 할 수 있다.

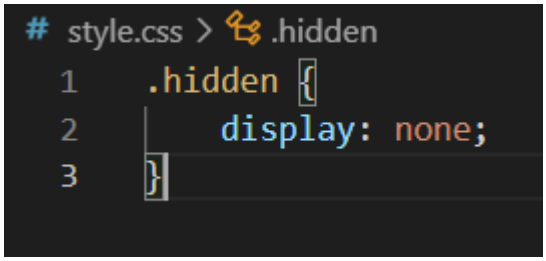
3. Form태그 Input 값 출력하기

form 태그의 submit를 방지 함으로써 아래와 같이 출력 결과를 얻을 수 있는 기초가 마련되었다.

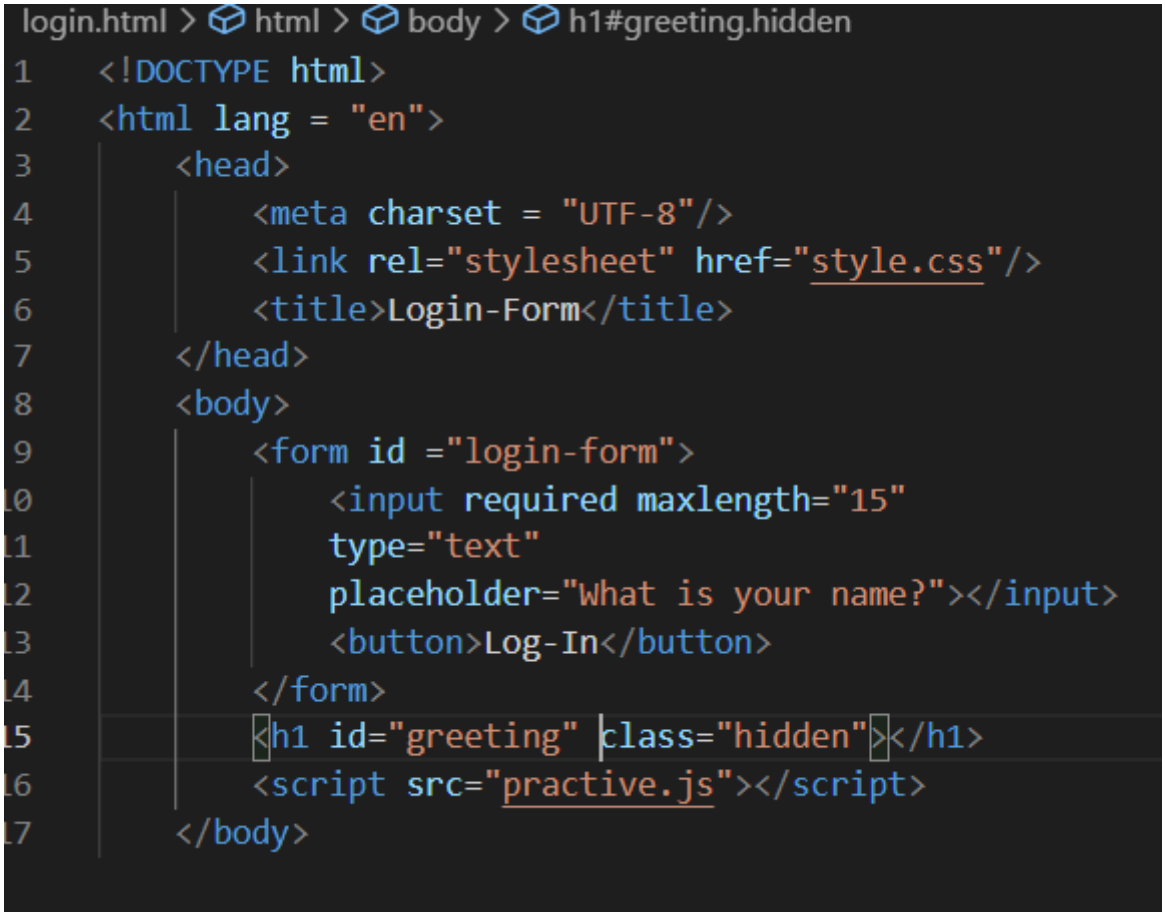


위 화면과 같은 기능구현을 위한 Logic은 input값을 화면에 출력시키고

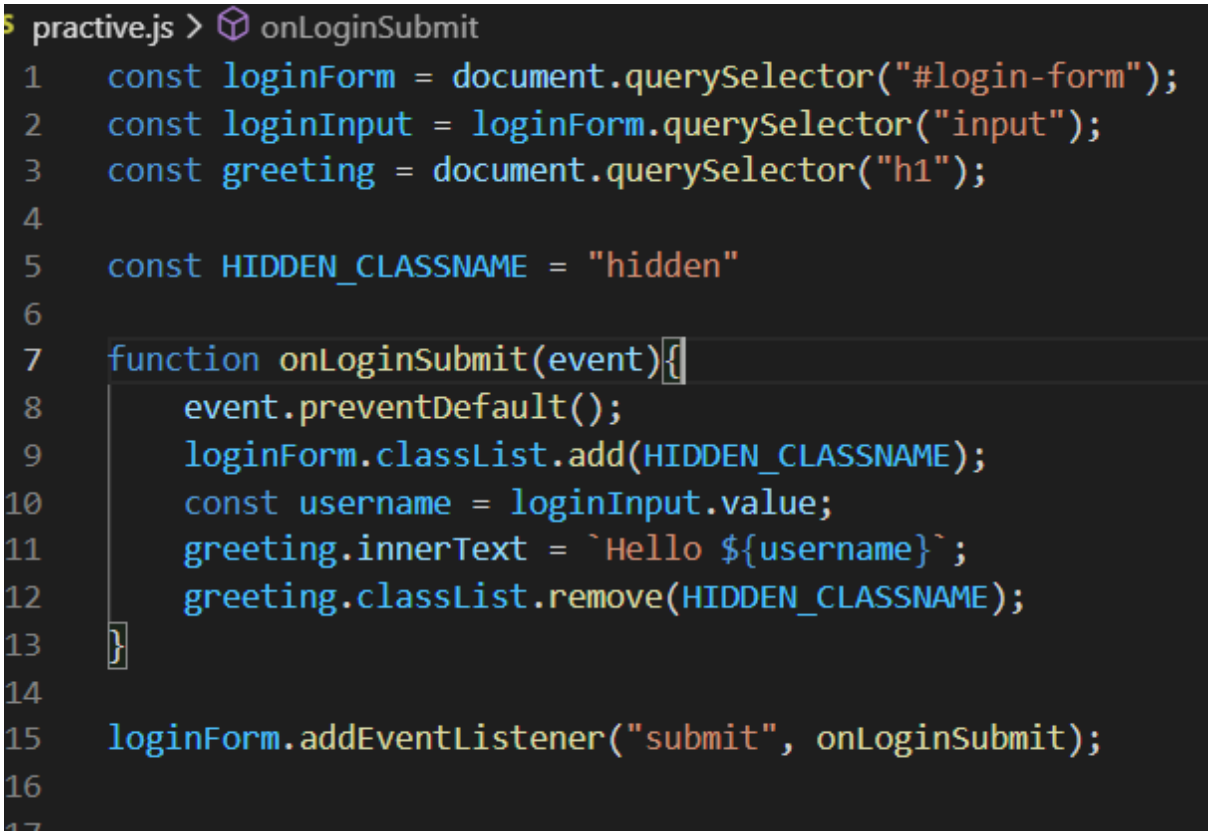
input, button 태그는 브라우저에서 숨기는 것이다.



우선, CSS에 사진과 같이 hidden class를 생성 후 display : none을 해준다.



다음으로, HTML에서 input값을 출력 시켜줄 <h1> 태그 생성후 css의 hidden 클래스를 적용시켜 브라우저에 숨긴다.



마지막으로 Javascript로 와서 HTML의 input, h1 태그의 정보를 가져오는 변수를 생성한다. 그 후 onLoginSubmit 함수에서 9번째 줄과 같이 form 함수에 hidden 클래스를 추가하여 브라우저에서 숨기고 10번째 줄에 받아온 사용자 이름을 11~ 12번째 줄과 같이 h1태그에 text로 넣어주고 h1태그의 hidden 클래스를 삭제함으로써 사용자 이름을 브라우저에 출력 할 수 있다.

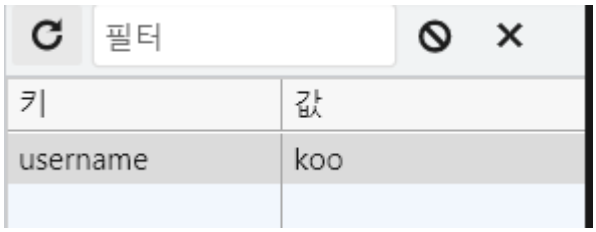
4. 사용자 이름의 저장과 불러오기

유튜브를 볼때, 조정한 볼륨 값을 새로고침해도 유지되듯이 **입력한 사용자의 이름을 저장하고 웹 페이지를 새로 고침해도 유지되야 할 것이다.**

Local Storage를 사용하여 이를 구현 해보자.

```
practive.js > onLoginSubmit
1   const loginForm = document.querySelector("#login-form");
2   const loginInput = loginForm.querySelector("input");
3   const greeting = document.querySelector("h1");
4
5   const HIDDEN_CLASSNAME = "hidden"
6
7   function onLoginSubmit(event){
8       event.preventDefault();
9       loginForm.classList.add(HIDDEN_CLASSNAME);
10      const username = loginInput.value;
11      localStorage.setItem("username", username);
12      greeting.innerText = `Hello ${username}`;
13      greeting.classList.remove(HIDDEN_CLASSNAME);
14  }
15
16  loginForm.addEventListener("submit", onLoginSubmit);
17
```

우선, 11번째 줄과 같이 localStorage.setItem()을 통해 입력된 사용자 이름을 아래와 같이 저장할 수 있다.



이제 저장된 값은 새로고침을 하여도 유지된다. 이렇게 저장된 값을 다시 불러오는 방법은 다음과 같다.

우선, localStorage에 저장된 값이 있는지 확인 후, form 태그나 h1 태그를

사용해야 하기 때문에, 9번째 줄과 같이 form 태그에도 hidden 클래스를 추가하여 숨긴다.

practive.js > ...

```
1   const loginForm = document.querySelector("#login-form");
2   const loginInput = loginForm.querySelector("input");
3   const greeting = document.querySelector("h1");
4
5   const HIDDEN_CLASSNAME = "hidden"
6
7   function onLoginSubmit(event){
8       event.preventDefault();
9       loginForm.classList.add(HIDDEN_CLASSNAME);
10      const username = loginInput.value;
11      localStorage.setItem("username", username);
12      paintGreetings(username);
13  }
14
15  function paintGreetings(username){
16      greeting.innerText = `Hello ${username}`;
17      greeting.classList.remove(HIDDEN_CLASSNAME);
18  }
19
20  const savedUsername = localStorage.getItem("username")
21
22  if(savedUsername === null){
23      loginForm.classList.remove(HIDDEN_CLASSNAME);
24      loginForm.addEventListener("submit", onLoginSubmit);
25  } else {
26      paintGreetings(savedUsername);
27  }
28
```

22,25 번째 줄과 같이 if else를 통해 localStorage에 저장된 값이 있는지

확인한다. 저장 값이 없을 경우 23~24번째 줄과 같이 HTML의 form태그의 hidden 클래스를 제거하여 브라우저에 보이게하고 submit 이벤트 함수를

대기시킨다.

반면, 저장 된 값이 있는 경우에는 15~18번째 줄의 paintGreetings 함수를

통해서 저장된 username 값을 불러와 출력하고 HTML의 h1태그의 hidden

클래스를 제거하여 브라우저에 보이게한다.

위 과정을 통해 최종적으로 완성예시와 같은 Login-Form을 완성 시킬 수 있다.