

10주차 강의

자바웹프로그래밍(1)

강사 : 최도현

트렌드 분석

• 웹 서비스 및 개발 트렌드

기술 트렌드 - 세션



웹 보안과 세션

• 웹 데이터 통신에서 세션이란?

- 사용자가 서버에 접속한 하나의 상태
 - 사용자 인증(식별) 이후 세션 생성

• 지난 시간 쿠키와 비교하여

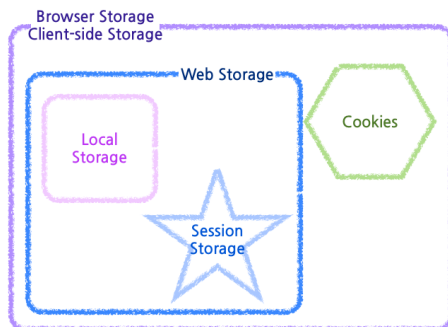
- 쿠키는 로컬에 저장 – 해킹 위험 존재
 - 세션은 서버에서 처리/저장, 비교적 안전?

• 쿠키는 장시간 신뢰 할 수 없다.

- 해결책? 세션 ID(식별자)를 발급받아 유지
- 중요한 데이터 저장은 세션

• 웹 브라우저 로그아웃 / 종료시에

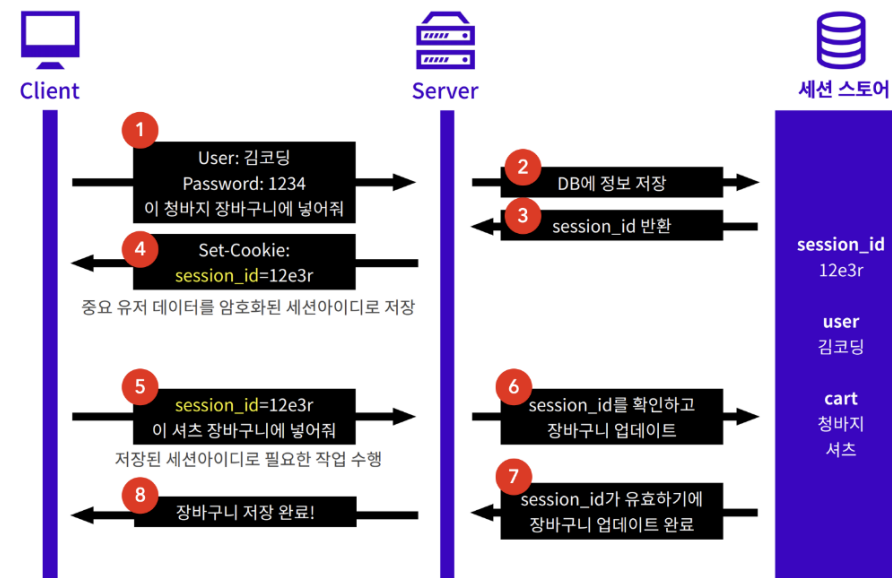
- 웹 브라우저만 종료해도 삭제됨
- 세션 ID는 보안 정책에 따라 삭제된다.



순	점검항목	순	점검항목
1	운영체제 명령 실행	12	크로스사이트 리퀘스트 변조
2	SQL 인젝션	13	자동화공격
3	XPath 인젝션	14	파일업로드
4	디렉토리 인덱싱	15	경로추적 및 파일다운로드
5	정보누출	16	관리자페이지 노출
6	악성콘텐츠	17	위치공개
7	크로스사이트 스크립트(XSS)	18	데이터 평문전송
8	악한 문자열 강도(브루트포스)	19	쿠키 변조
9	불충분한 인증 및 인가	20	웹 서비스 메소드 설정 공격
10	취약한 비밀번호 복구	21	URL/파라미터 변조
11	불충분한 세션 관리		

출처: 행정안전부

중요 세션 정보는 서버에 저장



PART1

- 데이터 저장(쿠키)

데이터 저장 – 쿠키(아이디 저장하기)



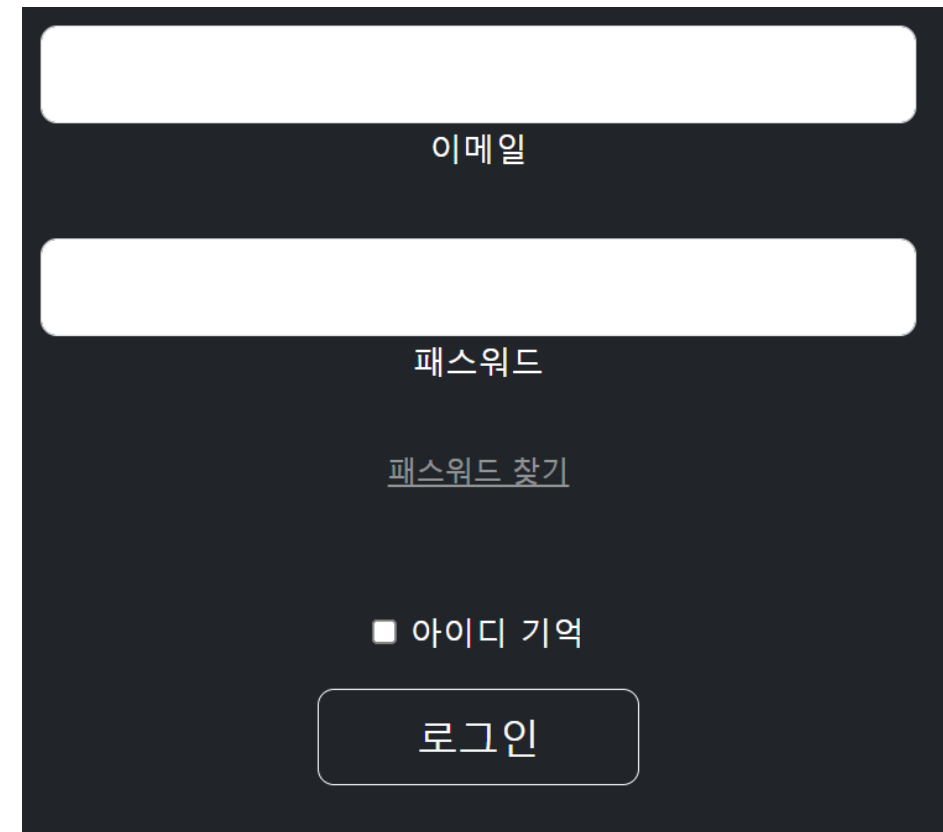
Cookie

데이터 저장 - 쿠키(id 저장하기)

- Login 폴더의 login.html을 이동한다.
 - 위치 확인, 아이디 체크 박스를 추가한다.

```
<div class="checkbox mb-3">  
  <label>  
    <input type="checkbox" value="remember-me" id="idSaveCheck"> 아이디 기억  
  </label>  
</div>
```

- 로그인 페이지 새로고침 하여 페이지 확인
 - 아직 큰 변화가 없다.
- 체크를 하면 이메일이 저장되어야 한다.
 - login.js에 아이디 저장 기능을 구현한다.



A dark-themed login form with two white input fields. The first field is labeled '이메일' (Email) and the second is labeled '패스워드' (Password). Below the password field is a link labeled '패스워드 찾기' (Find Password). At the bottom, there is a checkbox labeled '아이디 기억' (Remember ID) and a white '로그인' (Login) button.

데이터 저장 - 쿠키(id 저장하기)

- js 폴더의 login.js 에 추가 기능을 구현한다.

- 기존 check_input () 함수를 수정한다.

- 아이디 체크 id를 불러온다.

- 체크 박스가 체크상태 이면

- 쿠키를 생성, 1일

- 체크박스가 체크 상태 x 이면

- 쿠키를 삭제(추가 구현 필요)

```
// 전역 변수 추가, 맨 위 위치  
const idsave_check = document.getElementById('idSaveCheck');
```

```
// 검사 마무리 단계 쿠키 저장, 최하단 submit 이전
```

```
if(idsave_check.checked == true) { // 아이디 체크 o  
    alert("쿠키를 저장합니다.", emailValue);  
    setCookie("id", emailValue, 1); // 1일 저장  
    alert("쿠키 값 :" + emailValue);  
}
```

```
else{ // 아이디 체크 x  
    setCookie("id", emailValue.value, 0); //날짜를 0 - 쿠키 삭제  
}
```

- 쿠키 함수는 재활용한다. Why?

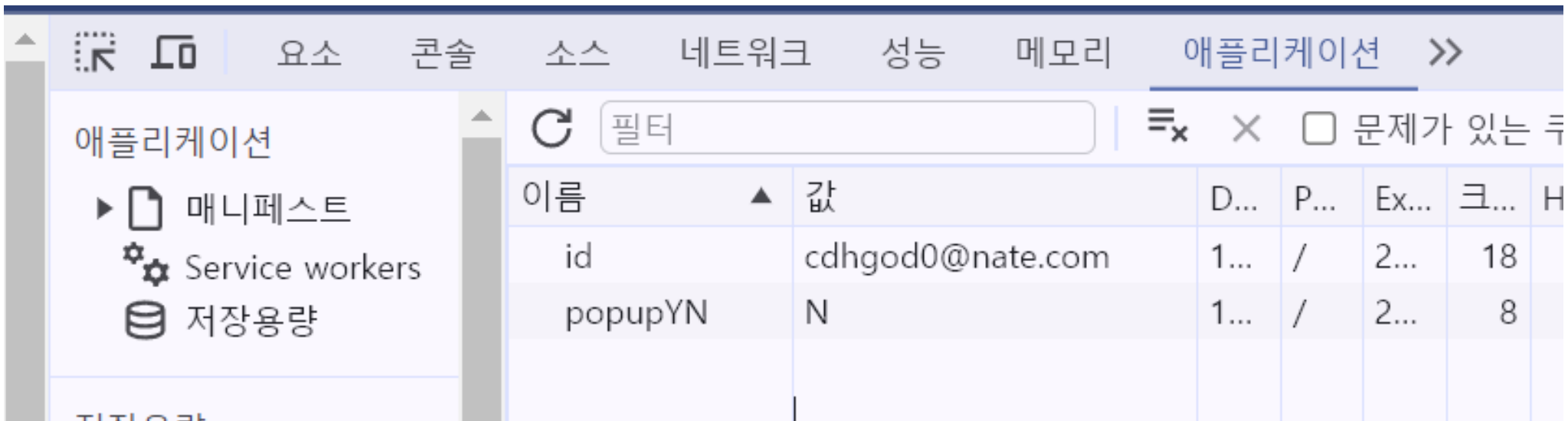
- 기존 popup.js의 쿠키 set/get함수를 재사용한다.

- Login.js에 재사용 한다. 함수 이름도 같다.

- 수정해야 하는 부분 : popupYN 이름을 id 로 변경해야 한다.

데이터 저장 - 쿠키(id 저장하기)

- 로그인 페이지를 열자.
 - 아이디 입력, 체크 박스 체크후 로그인 해보자.



- 먼저 쿠키가 잘 생성되는지 확인

데이터 저장 - 쿠키(id 저장하기)

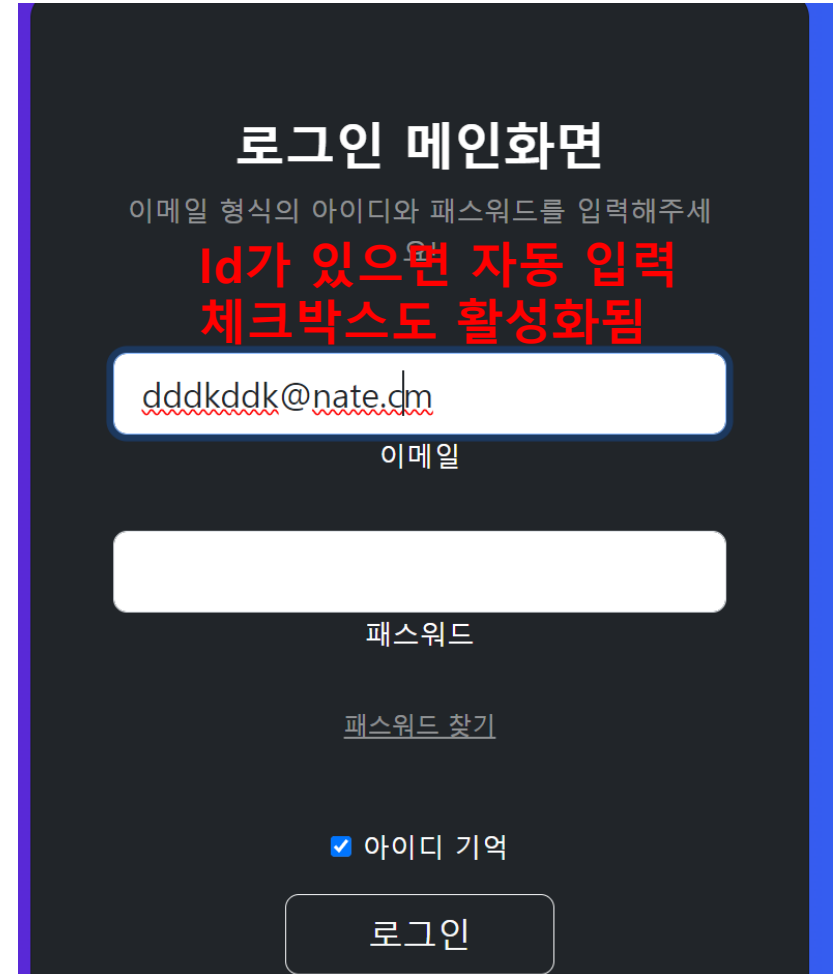
- 기타 함수를 추가 추가한다. (login.js 추가)
 - 로그인 페이지 - 아이디 자동 삽입

```
function init(){ // 로그인 폼에 쿠키에서 가져온 아이디 입력
  const emailInput = document.getElementById('typeEmailX');
  const idsave_check = document.getElementById('idSaveCheck');
  let get_id = getCookie("id");
```

```
  if(get_id) {
    emailInput.value = get_id;
    idsave_check.checked = true;
  }
}
```

- 마지막 init 로딩 부분 추가(login.html 수정)

```
<body class="text-center" onload="init();" >
```



The image shows a login page titled "로그인 메인화면" (Login Main Screen). Below the title is a subtitle "이메일 형식의 아이디와 패스워드를 입력해주세요" (Please enter your ID and password in email format). A red text overlay says "Id가 있으면 자동 입력 체크박스도 활성화됨" (If ID exists, the auto-input checkbox is also activated). The form has two input fields: "이메일" (Email) with the value "dddkddk@nate.com" and "패스워드" (Password). Below the password field is a link "패스워드 찾기" (Find Password). There is a checkbox labeled "아이디 기억" (Remember ID) which is checked. At the bottom is a "로그인" (Login) button.

응용 문제 풀기 - NOW!!!!

- 로그인/로그아웃 횟수 쿠키 저장하기
 - login.js에 추가 구현한다.
 - 로그인 login_count() 함수
 - 쿠키 이름 : login_cnt
 - 로그아웃 logout_count() 함수
 - 쿠키 이름 : logout_cnt
 - 기능 구현
 - 버튼을 클릭할 때마다 횟수(정수)를 증가
 - 기존 쿠키의 카운트 값을 얻는다.
 - 쿠키의 값을 + 1 업데이트 한다.
- 실습 결과 확인 - Q / A
 - 참고 : 기존 set/get 함수를 조금 수정해서 재활용한다.



PART1

• 데이터 저장(세션)

데이터 저장 - 로그인(세션)



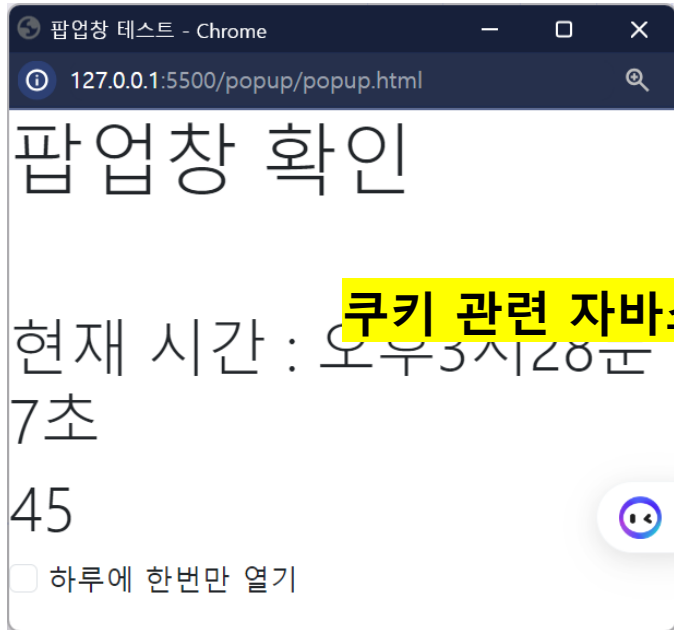
Cookie



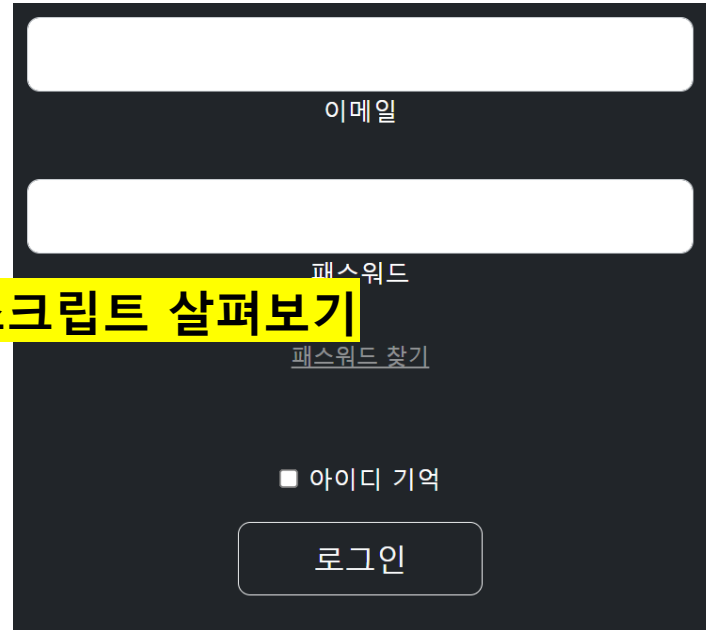
Session

지난주 내용 살펴보기

- 데이터 저장 – 쿠키
 - 팝업창 상태, ID 저장 등 쿠키 활용



쿠키 관련 자바스크립트 살펴보기



항목	태그 이름/설명
쿠키 저장소 대신 사용 할 수 있는 영구적인 저 장소는?	
자바스크립트에서 지원 하는 쿠키 객체의 이름 은?	
쿠키를 삭제하려면 무 엇을 수정해야 하는가?	
쿠키를 설정할 때 보안 옵션 설정은?	
다른 도메인에서 쿠키 전송을 차단 하는 정책 을 무엇이라 하는가?	
쿠키 내부의 키(KEY)에 저장할 수 있는 데이터 타입/자료구조는?	

내 홈페이지 확인

- 생각해보자. 데이터를 저장할 필요한 곳은?
 - 데이터를 저장 구현 = 일정기간 유지
 - LOL.com 메인창 다시 열어보기
- 일반적인 웹 사이트 기능 예)
 - 팝업 페이지
 - X 일 동안 팝업을 보지 않기
- 로그인 id
 - 이전에 입력한 id를 기억한다.
- 로그인 유지 기능(세션 : 오늘)
 - 메인화면을 열어도 로그인 상태로 열림
 - 일정 시간 이후 자동 로그아웃

로그인 메인화면

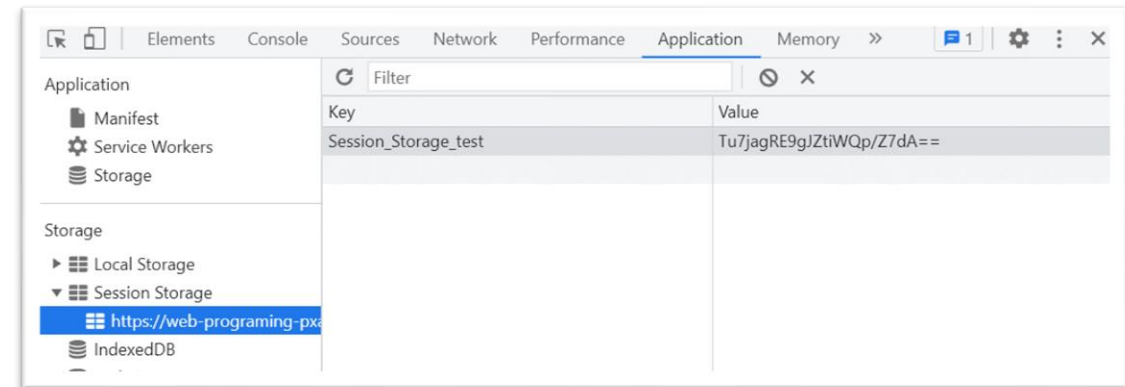
이메일 형식의 아이디와 패스워드를 입력해주세요!

이메일

패스워드

[패스워드 찾기](#)

☒ 아이디 기억



데이터 저장 - 세션(로그인)

- 세션 스토리지 로그인 구현하기
 - 기존 로그인 기능의 위치와 동작을 확인한다.
 - 기존 login.js 파일을 열어보자.
 - 버튼 클릭 후 check_input() 함수를 호출했다.
- Js 폴더에 session.js를 추가한다.
 - 세션 set/get 함수를 추가 구현한다.
 - 세션 지원 검사 - 웹 브라우저 지원 ok
 - sessionStorage 객체를 활용한다.
 - 다양한 메서드가 준비되어 있음
 - 세션이 없는 경우 새로 생성한다.
 - 키, 값으로 설정된다. (문자열 only)
 - 임시로 값을 id를 사용하자.

Method	Description
length	스토리지에 저장된 데이터의 길이
key(int)	해당 인덱스 위치에 있는 데이터(문자열)을 반환
getItem(key)	해당 키 값에 해당하는 데이터(문자열)을 반환
setItem(key, value)	해당 Key 값에 대한 데이터를 저장
removeItem(key)	해당 Key 값에 해당하는 데이터를 삭제
clear()	스토리지에 저장된 모든 데이터를 삭제

```
function session_set() { //세션 저장
    let session_id = document.querySelector("#typeEmailX");
    if (sessionStorage) {
        sessionStorage.setItem("Session_Storage_test", session_id.value);
    } else {
        alert("로컬 스토리지 지원 x");
    }
}
```

```
function session_get() { //세션 읽기
    if (sessionStorage) {
        return sessionStorage.getItem("Session_Storage_test");
    } else {
        alert("세션 스토리지 지원 x");
    }
}
```

참고 : 함수의 SCOPE

- 자바스크립트 객체를 지원, BUT
 - 현, 실습까지는 대부분 함수지향으로 구현.
 - 규모가 아직 작음, 코드 재사용 X

- 함수 SCOPE에 영향을 많이 받음
 - 블록 {} 내에서 생성
 - 내부 변수들은 함수 내에서 접근 가능
 - VAR = 함수 SCOPE
 - LET, CONST = 블록 SCOPE
- 즉, 함수를 추가/접근 할 때!
 - 대부분 함수는 분리가 원칙
 - 함수 내부에서 함수 호출은 OK

새로운 함수 추가할 때 → 중첩 정의는 피하자.

```
// 2-4-1
var a = 1;
function outer() {
  console.log(a);

  function inner() {
    console.log(a);
    var a = 3;
  }

  inner();

  console.log(a);
}
outer();
console.log(a);
```

0. 전역 실행컨텍스트 생성 [GLOBAL]

1. 변수 a 선언
2. 함수 outer 선언 [GLOBAL > outer]
3. 변수 a에 1 할당
4. outer 함수 호출 → outer 실행컨텍스트 생성
5. 함수 inner 선언 [GLOBAL > outer > inner]
6. outer scope에서 a 탐색 → global scope에서 a 탐색 → 1 출력
7. inner 실행컨텍스트 생성
8. 변수 a 선언
9. inner scope에서 a 탐색 → undefined 출력
10. 변수 a에 3 할당
11. inner 실행컨텍스트 종료
12. outer scope에서 a 탐색 → global scope에서 a 탐색 → 1 출력
13. outer 실행컨텍스트 종료

결과 : 1 → undefined → 1 → 1

데이터 저장 - 세션(로그인)

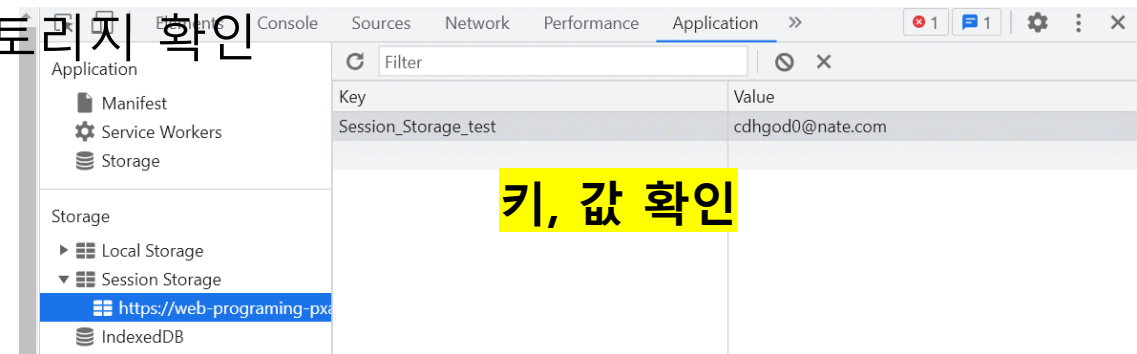
- 기존 login 함수를 수정한다.
 - 앞서 구현한 세션 set 함수를 호출

```
//생략...
if(id.value.length === 0 || password.value.length === 0){
    alert("아이디와 비밀번호를 모두 입력해주세요.");
}else{
    session_set(); // 세션 생성
    form.submit();
}
//생략....
```

- 로그인을 수행해본다.
 - 아이디, 패스워드 입력 후 로그인
 - F12 개발자 모드 --> Application → 세션 스토리지 확인
- 쿠키와 세션의 차이점을 기억하자!
 - 쿠키 : 모든 탭 공유(제한 없음)
 - 세션 : 하나의 탭 공유(범위가 제한)

	쿠키 Cookie	로컬 스토리지 Local Storage	세션 스토리지 Session Storage
매번 요청마다 서버로 전송?	O	X	X
용량제한	4KB	모바일: 2.5MB 데스크탑: 5MB ~ 10MB	모바일: 2.5MB 데스크탑: 5MB ~ 10MB
어떻게 얻나요	document.cookie	window.localStorage	window.sessionStorage
영구적인가?	유효기간이 있음	사용자가 지우지 않는 한 영구적	윈도우나 브라우저 탭을 닫으면 제거
저장방식	Key-value	Key-value	Key-value

참고: <https://www.zerocho.com/category/HTML&DOM/post/5918515b1ed39f00182d3048>



데이터 저장 - 세션(로그인)

- 세션 check 함수를 추가한다.
 - 로그인 → 로그인 페이지에 다시 접속 하는 경우

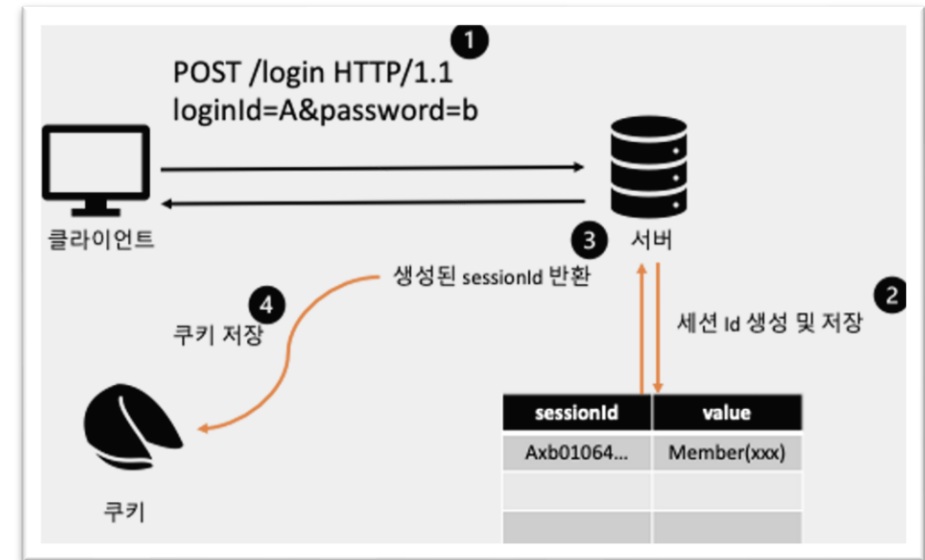
```
function session_check() { //세션 검사
  if (sessionStorage.getItem("Session_Storage_test")) {
    alert("이미 로그인 되었습니다.");
    location.href='../login/index_login.html'; // 로그인된 페이지로 이동
  }
}
```

- 이미 로그인 한 경우 = 세션이 유지됨

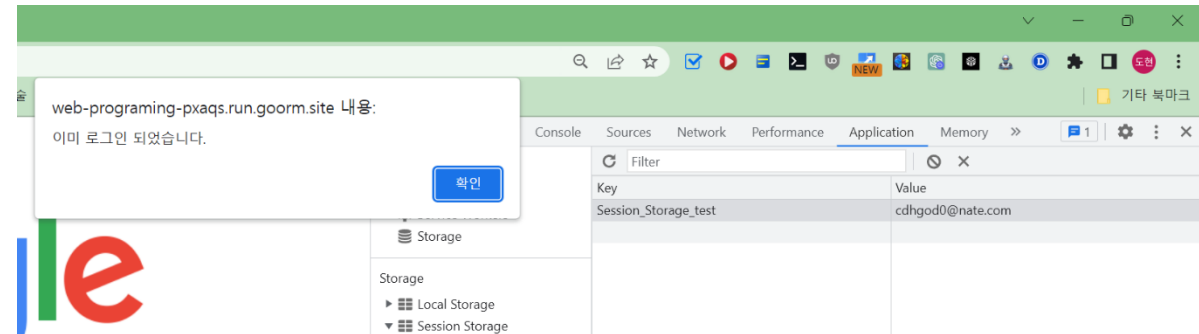
- 기존 init 함수를 수정한다.

```
function init(){ // 로그인 폼에 쿠키에서 가져온 아이디 입력
  // 생략.....

  if(get_id) {
    id.value = get_id;
    check.checked = true;
  }
  session_check(); // 세션 유무 검사
}
```



로그인 유지 기능 예) - 세션 id만 쿠키로 저장



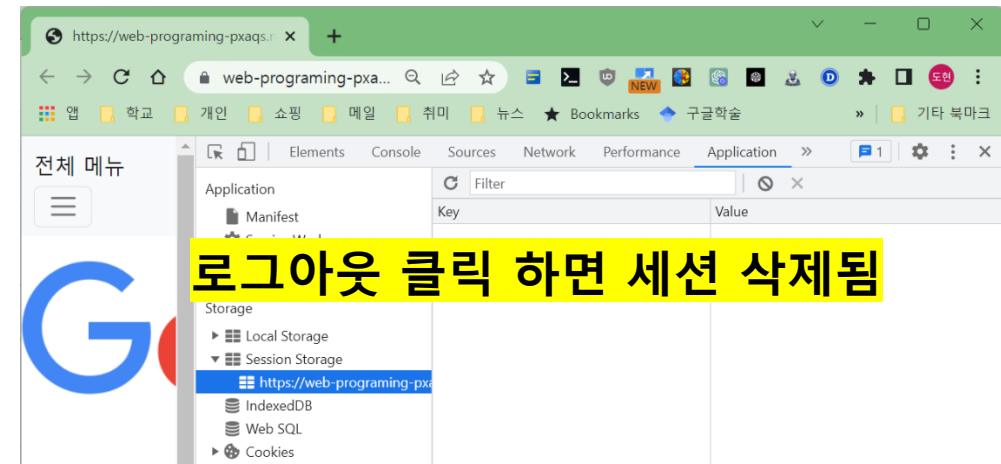
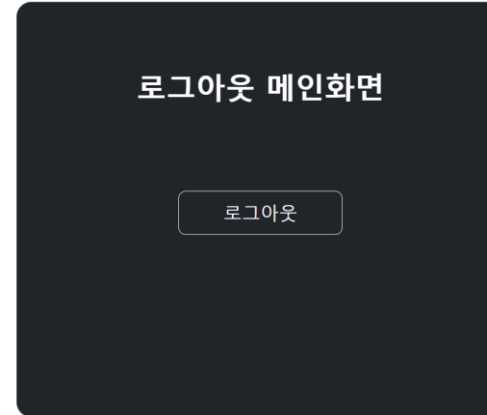
응용 문제 풀기 – NOW!!!!

- 세션 스토리지 로그아웃 구현하기
 - Js 폴더의 login.js를 수정한다.
 - 세션 del 함수를 추가 구현한다.

```
function session_del() { //세션 삭제
  if (sessionStorage) {
    sessionStorage.removeItem("Session_Storage_test");
    alert("로그아웃 버튼 클릭 확인 : 세션 스토리지를 삭제합니다.");
  } else {
    alert(" 세션 스토리지 지원 x");
  }
}
```

- 로그아웃 페이지 열고 수정하기
 - Js 폴더의 login.js를 연동한다.
 - logout 함수를 작성하고 세션 삭제를 구현한다.

```
function logout(){
  session_del(); // 세션 삭제
  location.href='../index.html';
}
```



10주차 연습문제

- 쿠키 관련 문제
 - 로그인 실패 횟수가 x번인 경우 로그인 제한
- login.js을 추가 구현한다.
 - 로그인 login_failed() 함수 구현
 - 쿠키에 로그인 실패를 카운팅한다.
 - 3번 이상인 경우 로그인을 아예 제한한다.
 - 실패횟수와 로그인이 제한 상태를 화면에 출력한다.
- 실습 결과 확인 – Q / A



로그인

로그인 가능 횟수를 초과했습니다. 4분 간 로그인 할 수 없습니다.

☐ 로그인 유지

[ID/PW 찾기](#) | [회원가입](#)

Q & A

- 다음주 할일
 - LOL 웹 사이트 구현(JS)
 - 로그인 창 : 세션

- 무엇이든 물어보세요!

