영 진 전 문 대 학 교

글로벌시스템융합과 - GSC

PHP – Cookie and Session

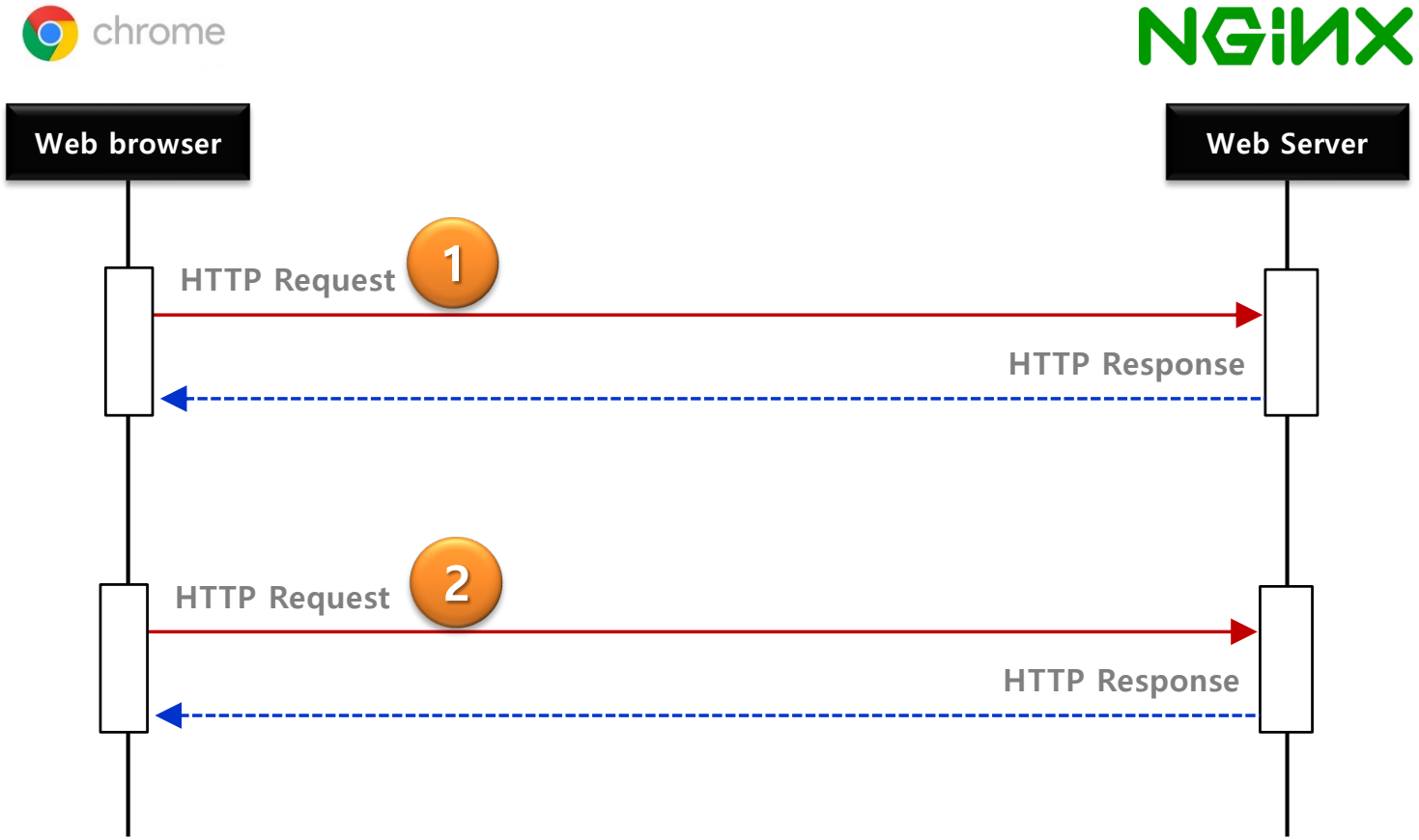
정영철 교수

글로벌시스템융합과

A.I WITH BETTER LIFE

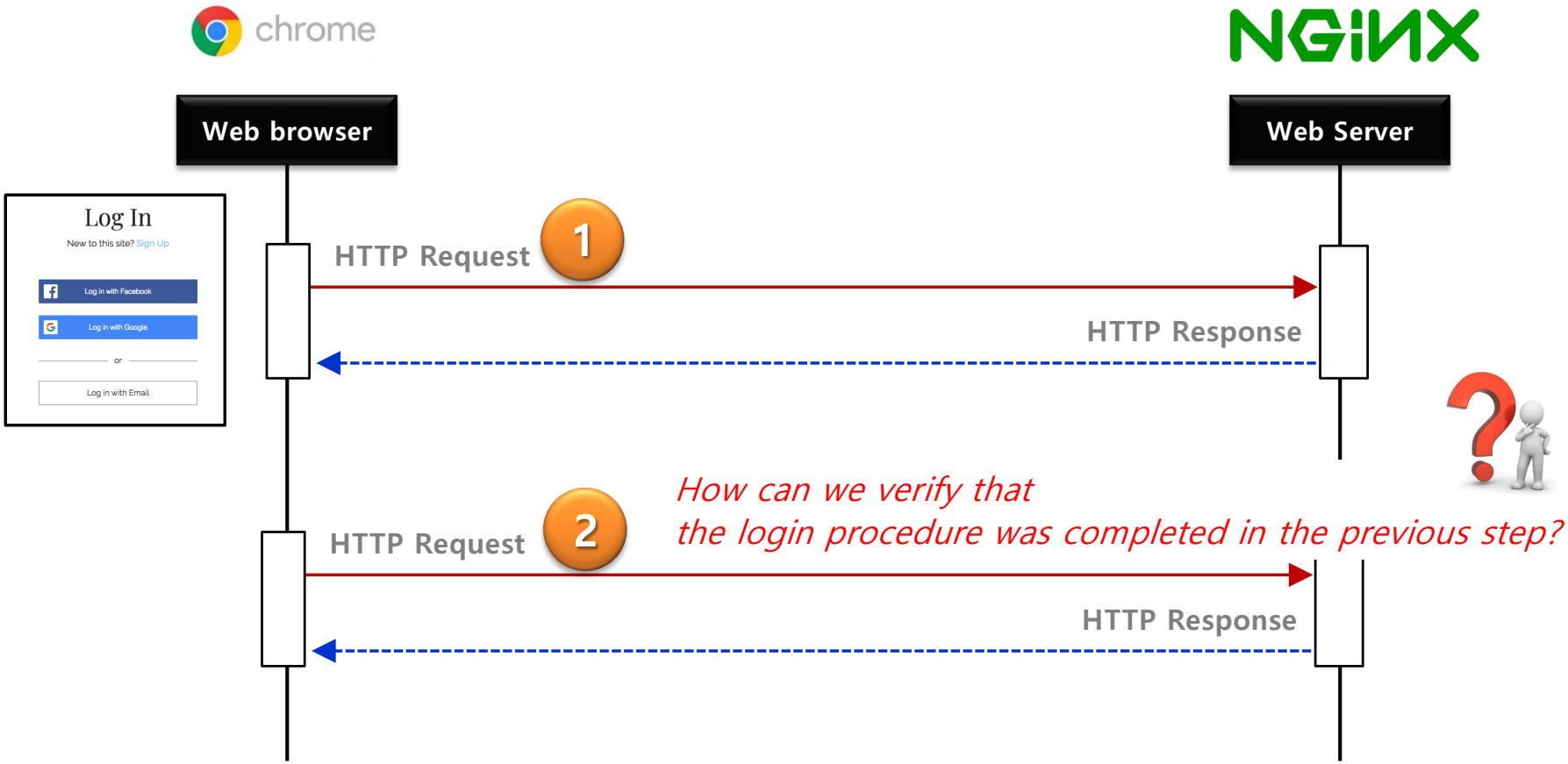
What was one of the key features of the web service?

Stateless



What are the pros and cons when using stateless mode?

- ✓ 장점 (Pros)
  - 웹 서버에 단순하고 부하가 적음
- ✓ 단점 (Cons)
  - 클라이언트와 서버 간의 세션 상태를 유지할 수 없음



Cookie(쿠키)

- **쿠키(Cookie)란?**
  - **웹 서버가 웹 브라우저를 통해 클라이언트(사용자)의 로컬 저장소(디스크)에 저장하는 작은 데이터 조각**
  - Key-Value(변수-값) 형태로 구성됨
  - 생성된 쿠키는 이후 클라이언트가 해당 서버로 요청 시, 자동으로 HTTP Request Header에 포함되어 전송됨

- **사용 절차**

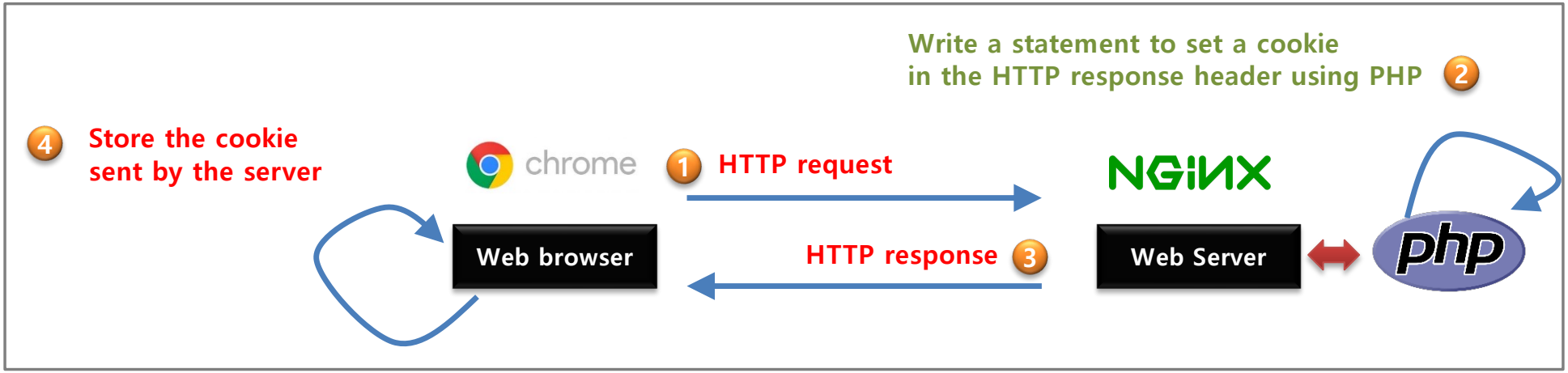
  ① **쿠키 저장**
  - PHP에서 setcookie() 함수를 사용하여 브라우저에 쿠키를 저장

  ② **쿠키 전달**
  - 브라우저가 해당 사이트에 다시 요청할 때
  - 저장된 쿠키를 Request Header에 자동 포함하여 서버로 전송

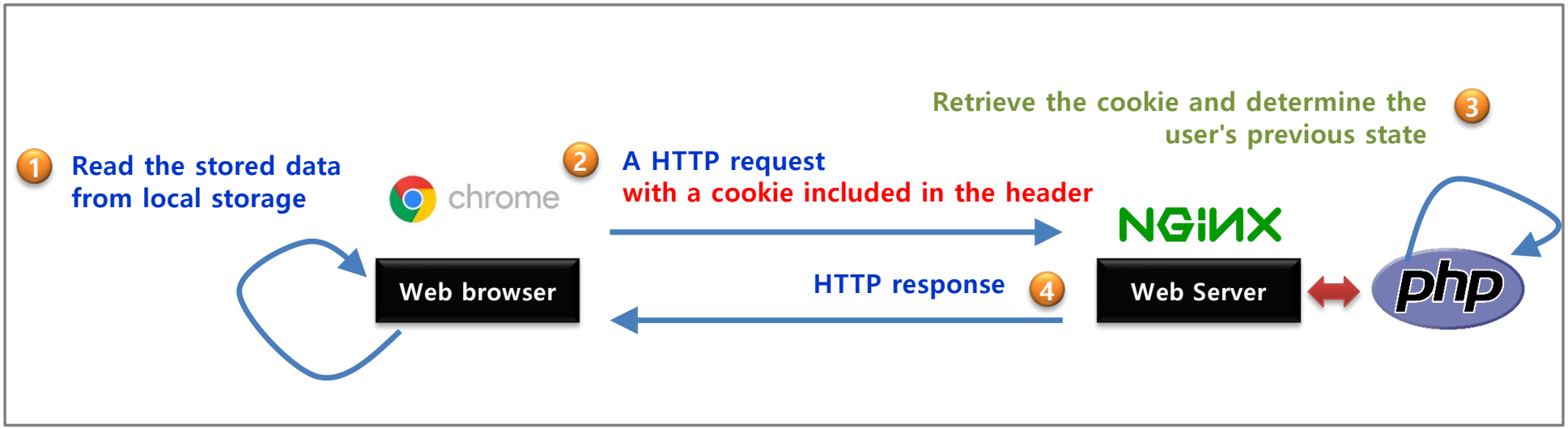
- **주요 활용 용도**
  - 사용자 인증 (Authentication, 예: 로그인 상태 유지)
  - 웹 사이트 방문자 추적 (Tracking)
  - 개인화된 정보 제공 (Personalization, 예: 사용자 설정 저장)

Cookie 사용 절차 : 쿠키 생성 & 쿠키 전달

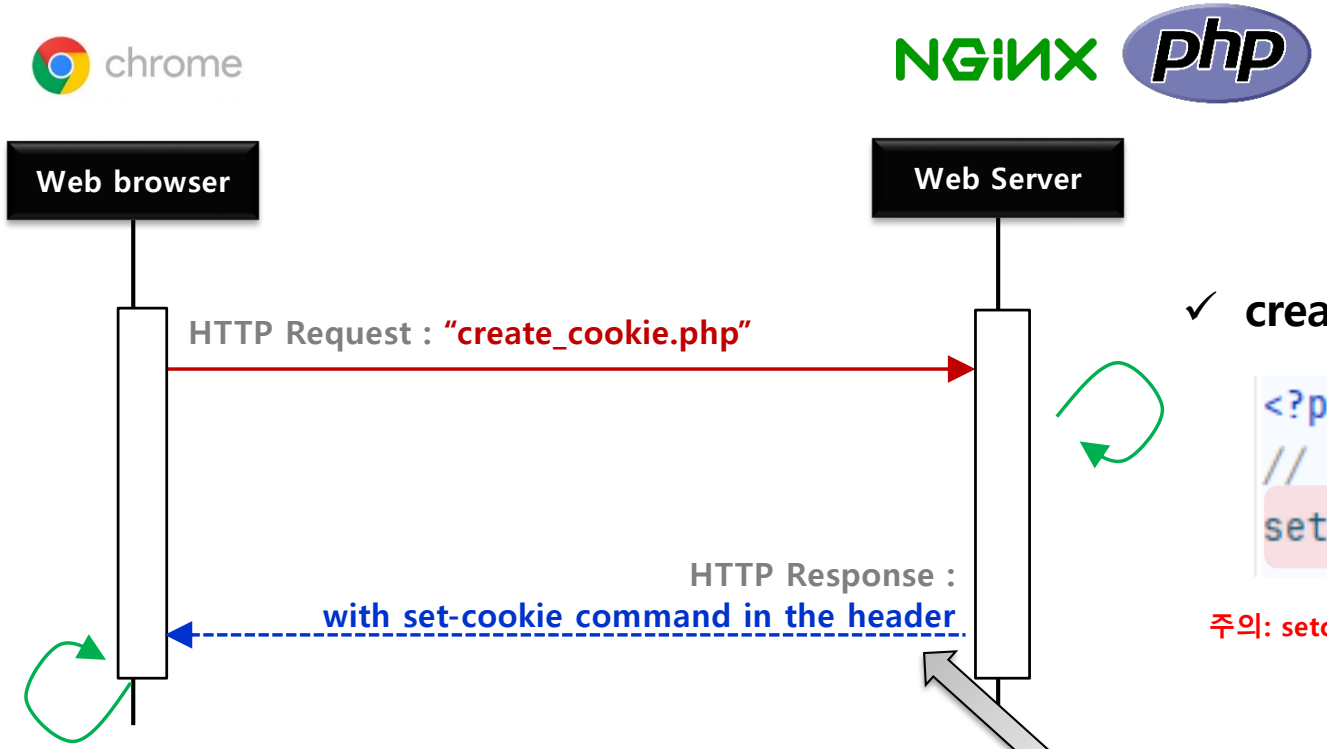
✓ 쿠키 저장 : 웹 서버 (Server) → 웹 브라우저 (Client)



✓ 쿠키 전달 : 웹 브라우저 (Client) → 웹 서버 (Server)



PHP 환경 쿠키 저장 절차

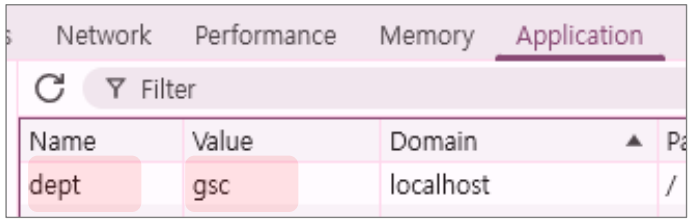


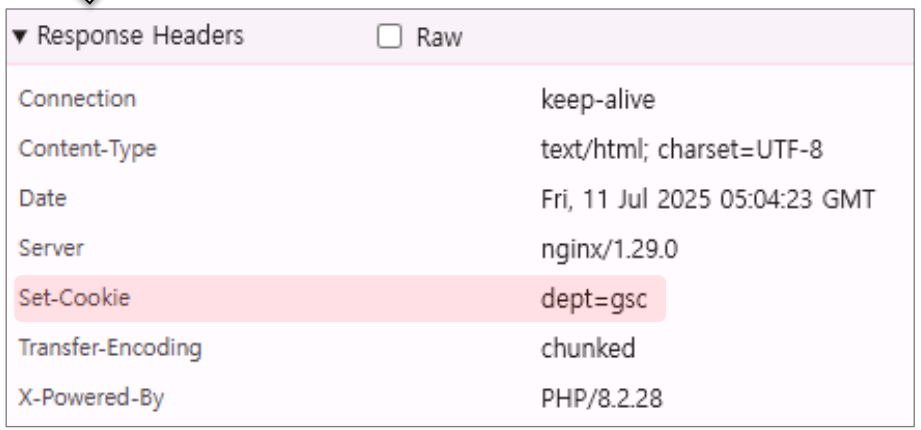
✓ **create_cookie.php**

```php
<?php
// 쿠키 생성
setcookie('dept', 'gsc');
```

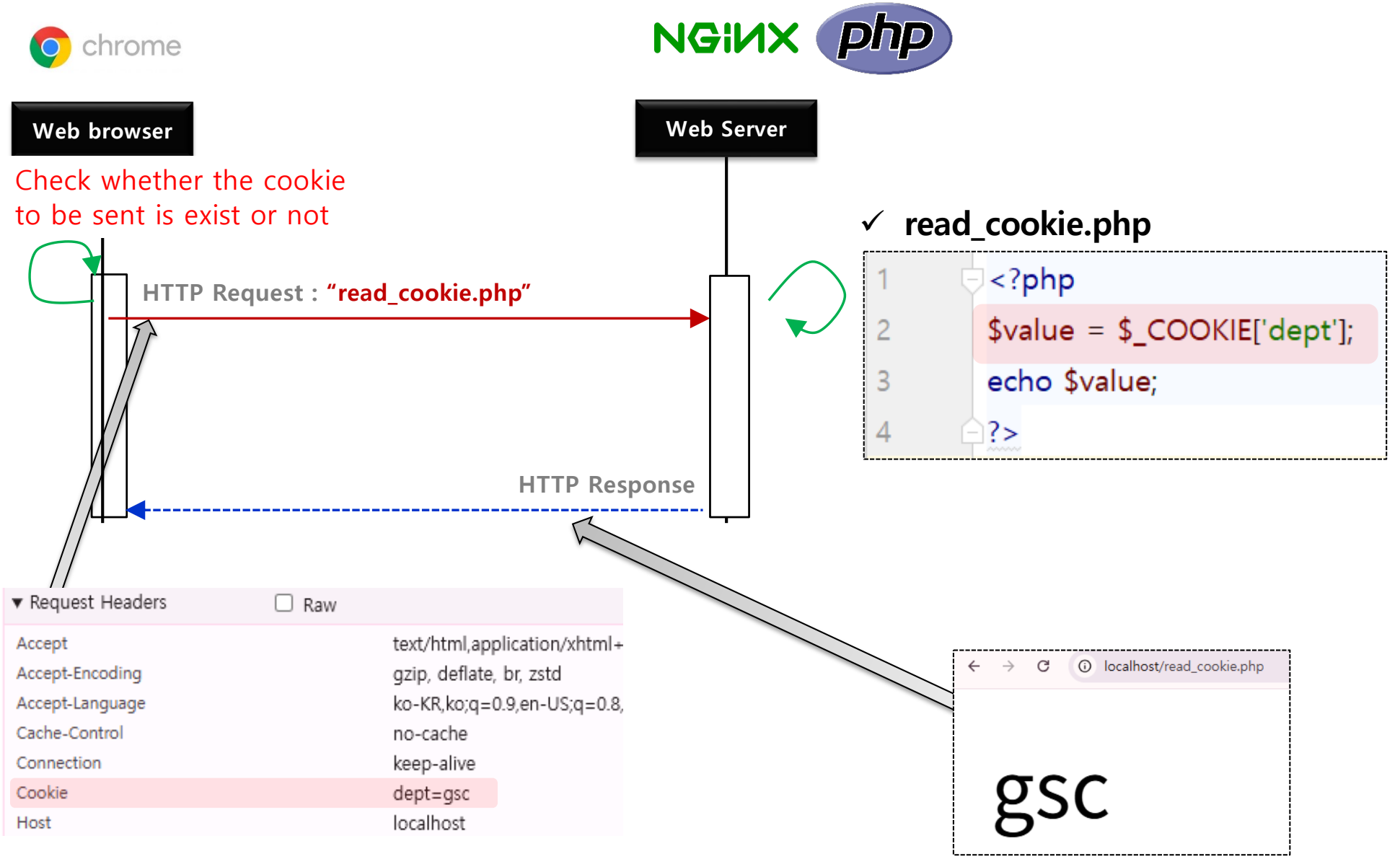
주의: setcookie 함수는 PHP 출력 코드 제일 상단 위치

Store the requested cookie info. at local storage





PHP 환경 쿠키 사용 절차



PHP Cookie 생성 및 읽기

- **setcookie()**
  - **역할**
    - **서버가 클라이언트(웹 브라우저)에 쿠키 생성을 요청하는 함수**
  - **동작 방식**
    - HTTP 응답(Response) 헤더에 Set-Cookie 명령이 포함됨
    - 브라우저는 이를 수신한 후 로컬 저장소(local storage) 또는 내부 쿠키 저장소에 해당 정보를 저장
  - **주의 사항**
    - **setcookie()는 출력 전에만 사용 가능**
    - HTML이 출력되기 전, 즉 PHP 스크립트 최상단에 위치해야 함
    - 이미 출력된 경우, headers already sent 오류 발생

- **$_COOKIE[ ]**
  - **역할**
    - **클라이언트로부터 전송된 쿠키 값을 서버가 읽기 위한 PHP 전역 배열**
  - **형태**
    - $_COOKIE['쿠키이름']
  - **동작**
    - 클라이언트가 이전에 받은 쿠키를 HTTP 요청(Request)에 함께 전송
    - 서버는 이 쿠키 값을 $_COOKIE 배열을 통해 접근 가능

Cookie 생성 : setcookie() function (1)

- 쿠키 생성 요청 (서버 → 클라이언트)
- 자료 구조: "이름=값" 형식의 문자열 데이터

setcookie (쿠키 이름, 쿠키 값, 만료 시간, 경로, 도메인, 보안, HTTP only, samesite);

```php
// 쿠키 설정 (브라우저 닫을 때까지 유지)
setcookie("user", "Alice");

// 만료 시간 지정 (1시간 뒤까지 유효)
setcookie("user", "Alice", time() + 3600);
```

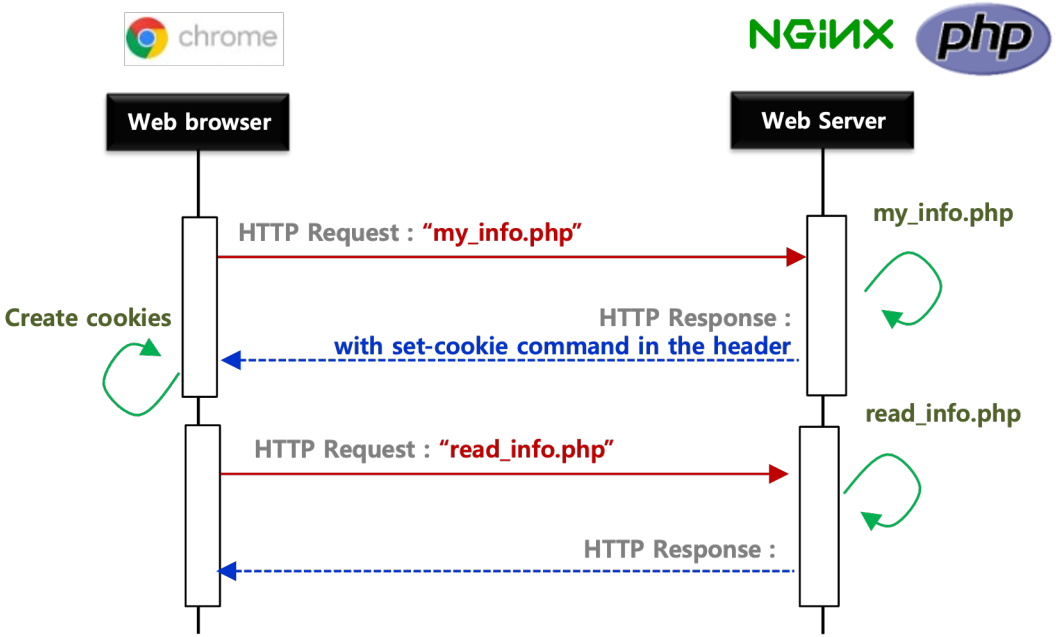
```php
setcookie(
  "user",            // name
  "ycjung",          // value
  time() + 3600,     // expires (1시간 후 만료)
  "/",               // path
  "example.com",     // domain
  true,              // secure (HTTPS 전용)
  true               // httponly (JS 접근 차단)
);

// SameSite 설정은 배열로 전달 (PHP 7.3+)
setcookie("session", "abc123", [
  'expires' => time() + 3600,
  'path' => '/',
  'domain' => 'example.com',
  'secure' => true,
  'httponly' => true,
  'samesite' => 'Lax'
]);
```

Cookie 생성 : setcookie() function (2)

매개변수	설명	예시값	실무 팁
name	쿠키 이름 (식별자)	"user_id"	필수, $_COOKIE['name']으로 접근
value	저장할 값	"ycjung"	문자열만 가능 복잡한 구조는 json_encode() 추천
expires	**쿠키 만료 시간 (UNIX 타임스탬프)**	time() + 3600 (1시간 후)	생략 시 브라우저 종료 시 삭제됨
path	쿠키가 유효한 URL 경로	"/", "/app"	**경로 하위**에서만 쿠키 전송됨
domain	쿠키가 유효한 도메인	"example.com"	**서브도메인** 간 공유 가능 (예: .example.com)
secure	HTTPS 연결에서만 쿠키가 전송되도록 설정	true 또는 false	https 사용 시 true 설정
httponly	JavaScript에서 접근 차단 (XSS 방지 목적)	true 또는 false	XSS 대응 위해 항상 true 사용 권장
samesite (PHP 7.3+)	타 사이트 요청 시 쿠키 전송 제한 (CSRF 대응)	"Lax", "Strict", "None"	CSRF 방어를 위해 기본값은 "Lax" 권장

Cookie 생성 : setcookie() function (3)



✓ **my_inof.php**

```php
<?php

$userInfo = [
    'age' => 23,
    'gsc' => 'Global System Convergence',
    'university' => 'Yeungjin Univ',
    'position' => 'professor'
];

setcookie('name', 'ycjung', time() + 3600);
setcookie('user_info', json_encode($userInfo), time() + 3600);
```

✓ **read_inof.php**

```php
<?php
if (isset($_COOKIE['user_info'])) {
    // 1단계: URL 디코딩 (setcookie가 내부적으로 인코딩했기 때문)
    $decodedString = urldecode($_COOKIE['user_info']);

    // 2단계: JSON 문자열 → PHP 배열로 복원
    $userInfo = json_decode($decodedString, associative: true); // true → 연관 배열

    // 3단계: 값 출력
    echo "학과(GSC): " . htmlspecialchars($userInfo['gsc']) . "<br>";
    echo "직책: " . htmlspecialchars($userInfo['position']) . "<br>";
    echo "대학: " . htmlspecialchars($userInfo['university']) . "<br>";
    echo "언어: " . htmlspecialchars($userInfo['language']) . "<br>";
} else {
    echo "쿠키가 설정되어 있지 않습니다.";
}
```

setcookie 함수 : 만료 시간 (1)

setcookie (쿠키 이름, 쿠키 값, 만료 시간, 경로, 도메인, 보안 , HTTP only, samesite);

✓ **만료 시간은 쿠키가 언제까지 남아있어야 하는지 알려주는 것.**

항목	설명
time()	• 현재 시간 (UNIX 타임스탬프, 초 단위)
+ 3600	• 현재 시간으로부터 3600초 = 1시간 유지
생략 시	• **브라우저 세션 종료 시 삭제됨** (세션 쿠키)

```php
// 브라우저 닫을 때까지 유지 (세션 쿠키)
setcookie('session_cookie', 'temp');

// 7일 동안 유지되는 쿠키
setcookie('login_cookie', 'user123', time() + (86400 * 7)); // 86400초 = 1일
```

✓ **쿠키 삭제 시 만료 시간을 과거로 지정**

```php
setcookie('login_cookie', '', time() - 3600); // 즉시 삭제
```

setcookie 함수 : 만료 시간 (2)

✓ **create_cookie_time.php**

```php
<?php

setcookie('team', 'Samsung Lions', time() + 5);
```

✓ **read_cookie_time.php**

```php
<?php

echo date( format: DATE_RFC2822)."<br>";
echo $_COOKIE['team'];
```

Fri, 11 Jul 2025 05:34:36 +0000
Samsung Lions

12초 뒤 실행

Fri, 11 Jul 2025 05:34:48 +0000

Warning: Undefined array key "team" in **/var/www/html/read_cookie_time.php** on line **4**

setcookie 함수 : 경로 (path)

setcookie (쿠키 이름, 쿠키 값, 만료 시간, 경로, 도메인, 보안 , HTTP only, samesite);

- ✓ 쿠키 적용 경로 (폴더) 설정
- ✓ '/' 인 경우 도메인 내 모든 디렉토리의 PHP 파일에서 쿠키 사용 가능
- ✓ '/foo/' 인 경우 foo 디렉토리와 하위 디렉토리 내 PHP 파일에서만 쿠키 사용 가능
- ✓ 경로가 지정되지 않으면 현 파일이 존재하는 디렉토리 내에서만 쿠키 사용 가능

✓ localhost/test/myPage.php

```
2    // 현 디렉토리  내 파일만 쿠키 적용
3    setcookie('FOO', "BAR", 0);
4    ?>
```

✓ localhost/test/myPage.php

```
1    <?php
2    // 현 도메인 내 모든 디렉토리 쿠키 적용
3    setcookie('FOO', "BAR", 0, '/');
4    ?>
```

✓ localhost/usageCookie.php

```
echo $_COOKIE['FOO'];
```

✓ localhost/usageCookie.php

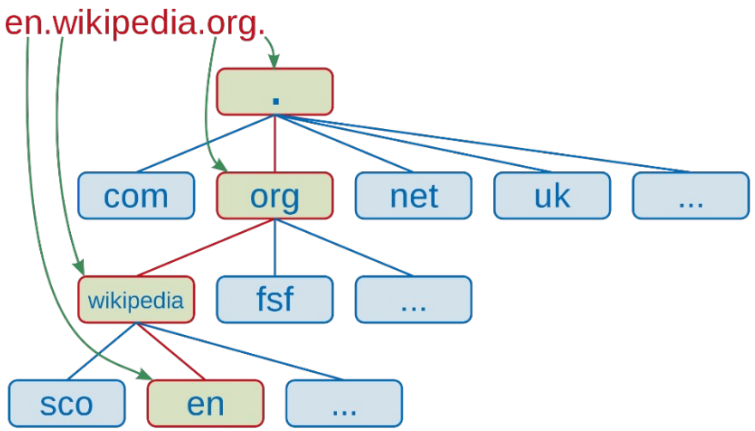
```
echo $_COOKIE['FOO'];
```

setcookie 함수 : 도메인 (domain)

setcookie (쿠키 이름, 쿠키 값, 만료 시간, 경로, 도메인, 보안 , HTTP only, samesite);

- ✓ 서브 도메인 간에는 쿠키가 기본적으로 공유되지 않음
- ✓ 서브 도메인간 쿠키 공유 시 사용

```php
<?php
// example.com 하위 도메인 내 "FOO" 쿠키 사용 가능
// 예) www.example.com      infocom.example.com
setcookie('FOO', "BAR", 0, ".example.com");
?>
```



domain 값	쿠키 전송 대상
생략 or 없음	현재 도메인에서만 예: www.example.com → sub.example.com ❌
"example.com"	example.com, www.example.com, sub.example.com 모두 가능
".example.com"	위와 동일
"sub.example.com"	sub.example.com에만 전송 다른 하위 도메인에는 ❌

setcookie 함수 : 보안 (Secure)

setcookie (쿠키 이름, 쿠키 값, 만료 시간, 경로, 도메인, 보안 , HTTP only, samesite);

- ✓ Secure 값은 Boolean 형
- ✓ True 일 경우, HTTPS 프로토콜에서만 쿠키 사용 가능

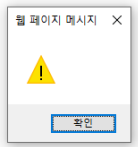
✓ myPage.php

```php
<?php
// Secure on
// HTTPS 사용시 만 쿠키 전송
setcookie('SECURE', "ON", 0, "", "", true);
?>
```

✓ prtCookie.php

```php
<?php
echo $_COOKIE['SECURE'];
?>
<script>
    alert(document.cookie);
</script>
```

Notice: Undefined index: SECURE in **C:\AutoSet9\public_html\prtCookie.php** on line **2**



setcookie 함수 : HTTP only (1)

setcookie (쿠키 이름, 쿠키 값, 만료 시간, 경로, 도메인, 보안 , HTTP only, samesite);

- ✓ **HTTP 프로토콜 통신 절차를 통해서만 쿠키 생성**
- ✓ **XSS(Cross-Site Scripting) 방지 목적**
- ✓ **예) JavaScript를 통한 쿠키 접근 방지**

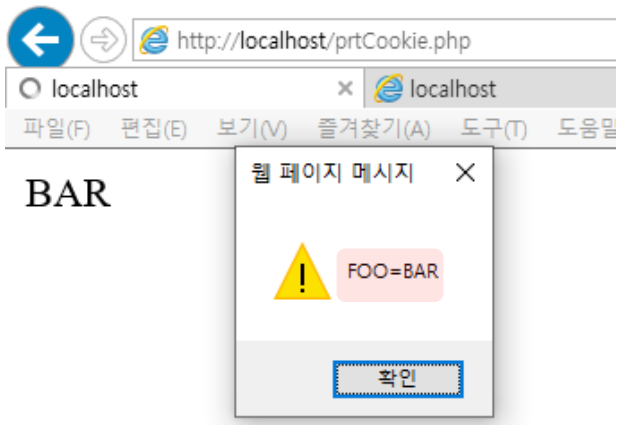
  document.cookie

✓ **prtCookie.php**

```php
<?php
echo $_COOKIE['FOO'];
?>
<script>
    alert(document.cookie);
</script>
```

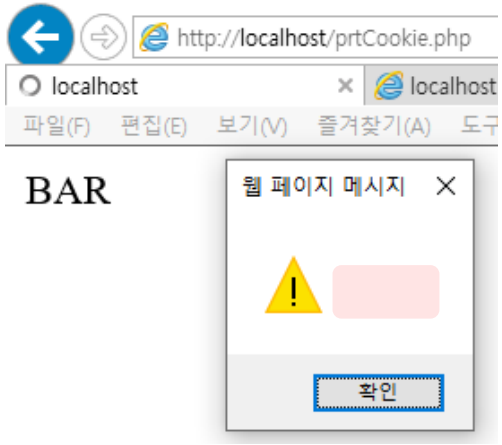
✓ **myPage.php**

```php
<?php
// 즉 HTTP only false
setcookie('FOO', "BAR", 0, "","", false, false);
?>
```



✓ **myPage.php**

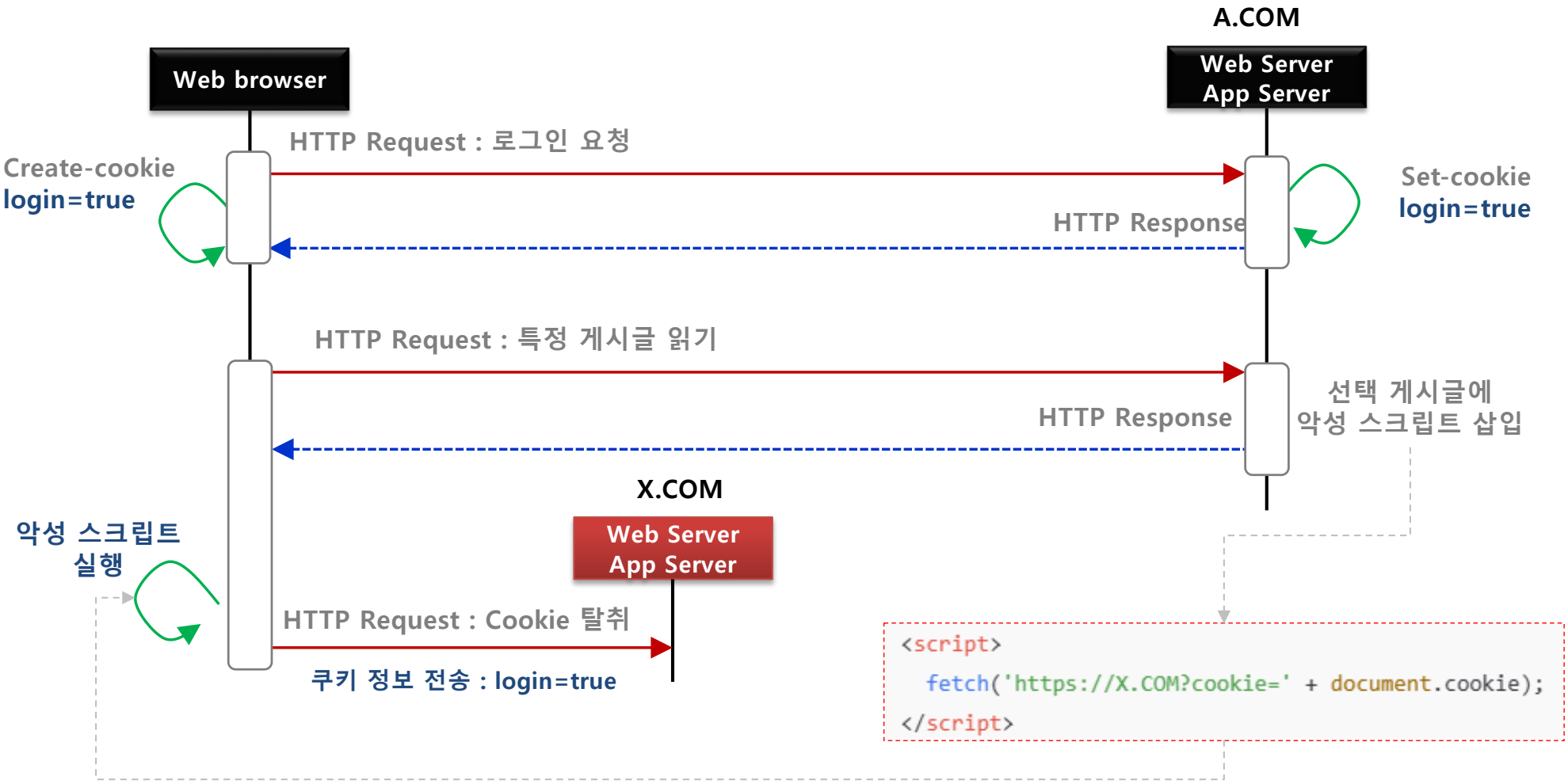
```php
<?php
// 즉 HTTP only true
setcookie('FOO', "BAR", 0, "","", false, true);
?>
```



setcookie 함수 : HTTP only (2)

✓ XSS란?

- XSS (Cross-Site Scripting)은 공격자가 웹사이트에 악성 JavaScript 코드를 삽입
- 해당 코드를 다른 사용자의 브라우저에서 실행되도록 유도하여, 개인정보, 세션 쿠키, 로그인 정보 등을 탈취하는 공격 방식
- 웹사이트 자체는 신뢰받는 사이트지만, 내부에서 사용자 입력을 검증 없이 출력하는 취약점이 있을 때 발생



setcookie 함수 : samesite (1)

setcookie (쿠키 이름, 쿠키 값, 만료 시간, 경로, 도메인, 보안 , HTTP only, samesite);

- ✓ SameSite는 쿠키의 교차 출처(cross-site) 전송을 제한하기 위한 속성
- ✓ CSRF(Cross-Site Request Forgery) 같은 보안 공격을 막기 위한 브라우저 보안 정책 중 하나
- ✓ setcookie() 함수에 이 옵션을 지정하면 언제 쿠키를 전송할지를 브라우저가 제어할 수 있음

값	설명
Strict	• **가장 제한적**: 같은 사이트 요청에서만 쿠키 전송
Lax	• **적절한 균형**: 같은 사이트 + 일부 안전한 요청(GET 방식)에서는 전송
None	• **제한 없음**: 타 사이트에서도 쿠키 전송 허용 (단, Secure 옵션 필수 → HTTPS 환경 필요)

- **None 사용 시 Secure 옵션 필수!**

```
setcookie("sessionid", "abc123", [
  "samesite" => "None",
  "secure" => true  // HTTPS 연결에서만 동작
]);
```

setcookie 함수 : samesite (2)

✓ CSRF란?

- CSRF (Cross-Site Request Forgery)는 사용자의 인증 정보를 악용해 원하지 않은 동작을 수행하는 공격 방식


```
<!-- X.COM에 숨겨진 계정 삭제 요청 -->
<form action="https://A.COM/delete-account" method="POST" style="display:none">
  <input type="hidden" name="confirm" value="yes">
  <input type="submit">
</form>

<script>
  document.forms[0].submit(); // 페이지 로드 시 자동 실행
</script>
```

✓ 공격 절차

① 사용자가 웹사이트 A.com에 로그인한 상태로 세션 쿠키를 보유 중

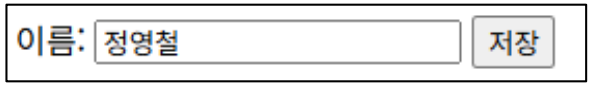
② 사용자가 악성 사이트 X.com에 방문

③ X.com 사이트가 사용자 모르게 A.com 사이트로 요청 전송

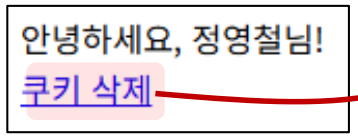
Lab 1: 사용자 이름 기억

```php
<?php
// 쿠키가 있는지 확인
if (isset($_COOKIE['username'])) {
    $username = htmlspecialchars($_COOKIE['username']);
    echo "안녕하세요, {$username}님!<br>";
    echo "<a href='delete_cookie.php'>쿠키 삭제</a>";
} else {
    echo "<form method='post' action='set_cookie.php'>
            이름: <input type='text' name='username'>
            <button type='submit'>저장</button>
          </form>";
}
```

- **index.php**



- **set_cookie.php**

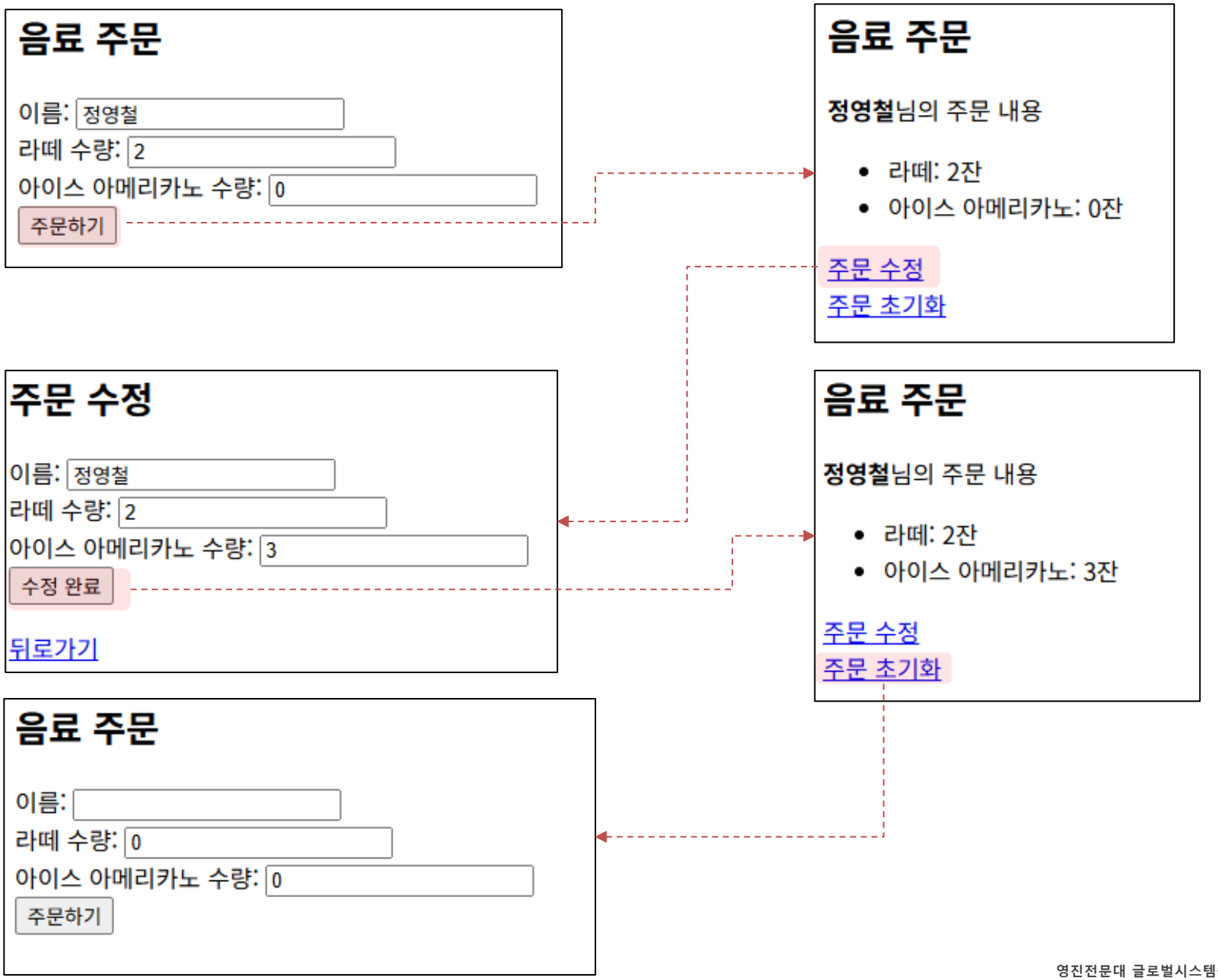


- **delete_cookie.php**

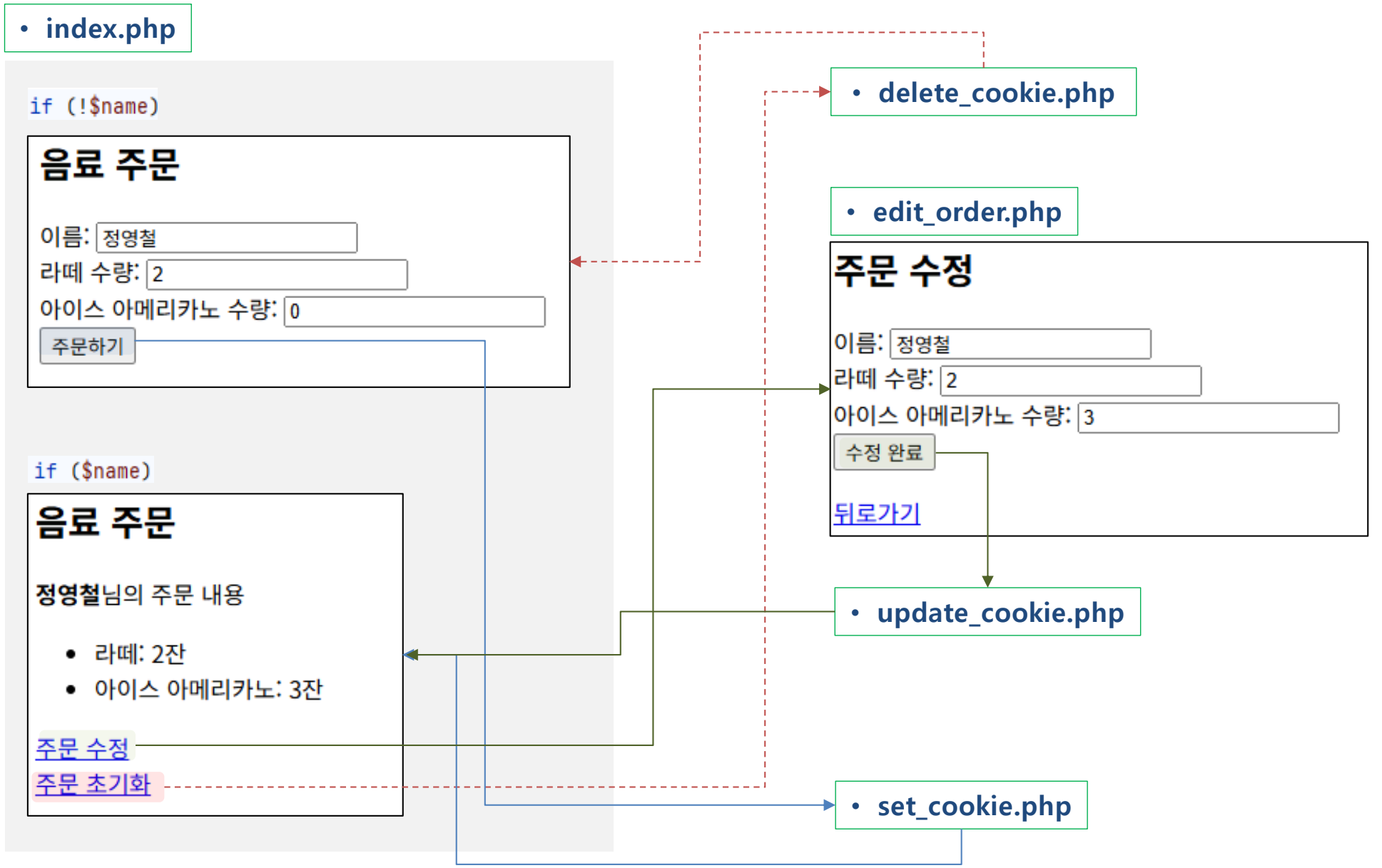
```php
<?php
setcookie('username', '', time() - 3600, '/');
header( header: "Location: index.php");
exit;
```

```php
<?php
if (isset($_POST['username'])) {
    $username = trim($_POST['username']);
    setcookie('username', $username, time() + 3600, '/');
    header( header: "Location: index.php");
    exit;
}
```

Lab 2: 음료 주문하기 : 플로우 설계



Lab 2: 음료 주문하기 : 페이지 분할



Session

Session (세션)이란?

- **Cookie의 문제점**
  - 쿠키는 서버가 클라이언트(브라우저)에 저장시키는 텍스트 기반 데이터
  - **쿠키 값은 클라이언트 측에 노출되기 때문에 위·변조 및 탈취 위험 존재**
    - 예: 로그인 쿠키 탈취 시 세션 하이재킹 발생 가능
  - 쿠키는 모든 HTTP 요청 시마다 자동 전송됨 → 불필요한 네트워크 트래픽 발생

- **세션(Session)의 개념**
  - 쿠키의 보안 문제를 보완하기 위해 도입된 **서버 중심의 상태 관리 방식**
  - **클라이언트는 오직 "세션 ID"만을 쿠키로 보관**
  - **실제 사용자 정보는 서버의 세션 저장소에만 저장됨**

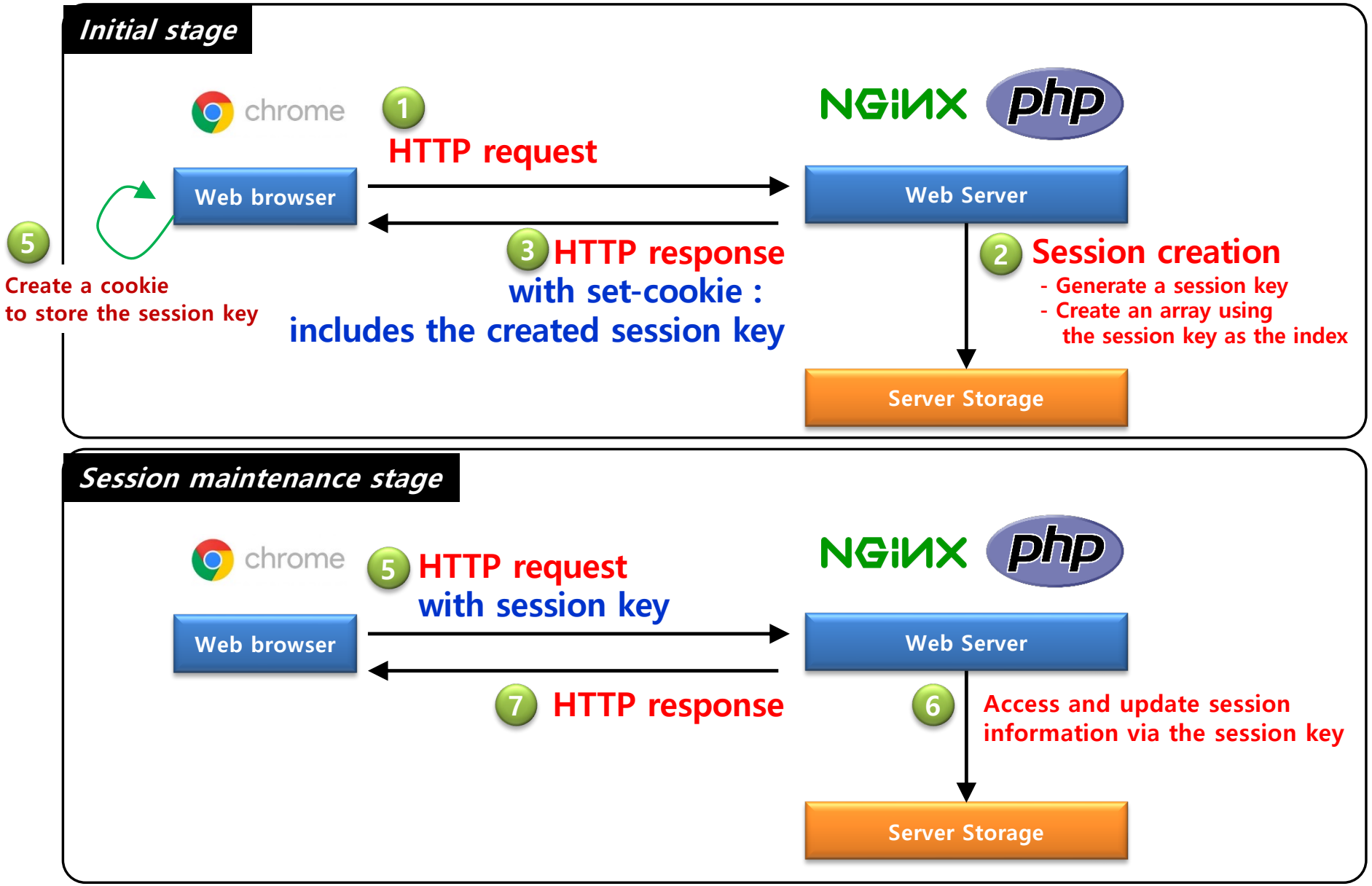
- **동작 방식**

단계	설명
① 클라이언트가 접속하면	• 서버는 고유한 세션 ID(key)를 생성하여 쿠키로 전송
② 클라이언트는 이후 요청마다	• 해당 세션 ID 쿠키를 함께 전송
③ 서버는 이 키를 바탕으로	• 서버 내부의 세션 저장소에서 **사용자 정보를 조회**

Cookie vs Session 비교

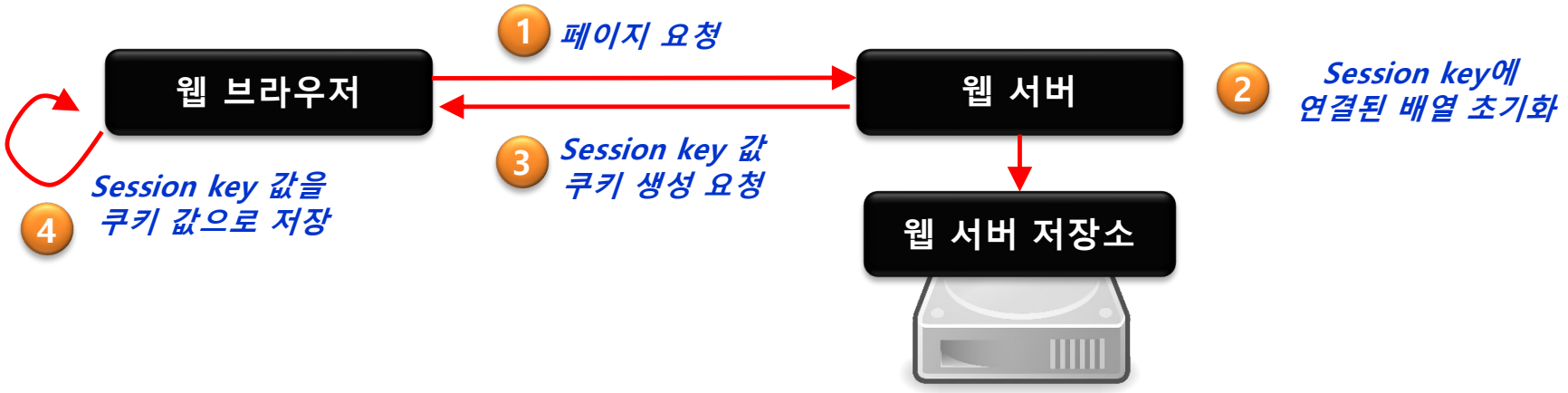
항목	Cookie	Session
저장 위치	**클라이언트 (브라우저)**	**서버 (서버 측 메모리 또는 파일 등)**
보안 수준	상대적으로 낮음 (노출 가능성)	상대적으로 높음 (서버에서만 보관)
사용 목적	간단한 데이터 보관 (기억하기 등)	로그인 상태, 장바구니 등 **중요한 상태 관리**
만료 시점	클라이언트 설정값에 따라 유동적	브라우저 종료 or 서버 설정 시간 초과
용량 제한	도메인당 4KB 이내	서버 상황에 따라 자유로움

Session : 동작 절차 (1)

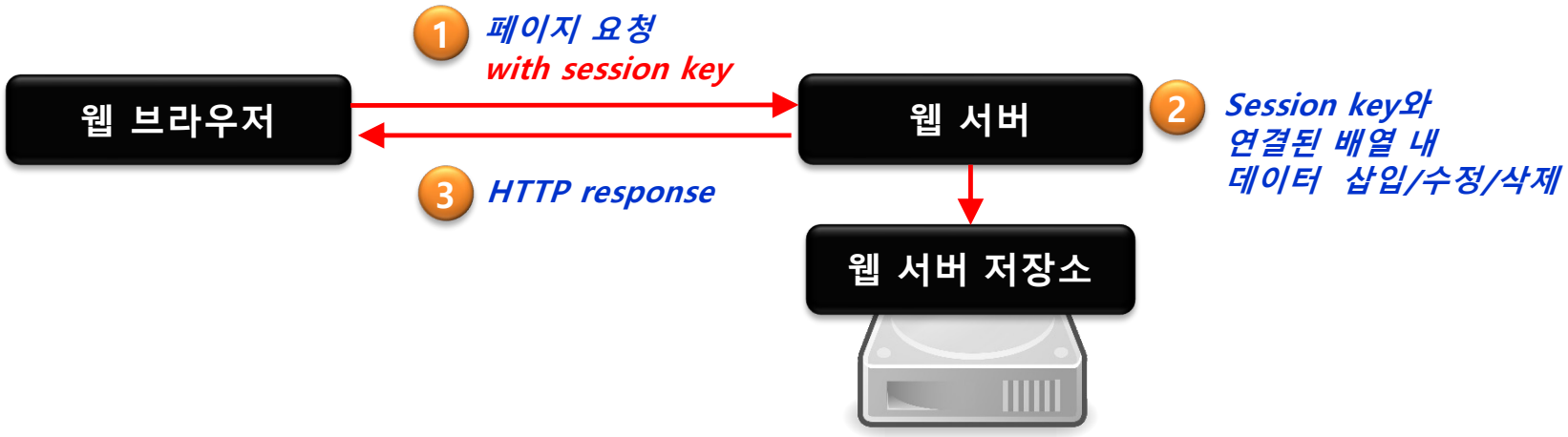


Session : 동작 절차 (2)

- 초기 접속



- 세션 생성 후 재 접속



세션 동작 절차 (1)

Web browser

Web Server

HTTP Request : "session_set.php"

▼ Response Headers Raw

Connection	keep-alive
Content-Type	text/html; charset=UTF-8
Date	Mon, 14 Jul 2025 02:28:38 GMT
Server	nginx/1.29.0
Transfer-Encoding	chunked
X-Powered-By	PHP/8.2.28

HTTP Response

- session_set.php

```
1  <?php
2  //session_start();
```

HTTP Request : "session_set.php"

▼ Response Headers Raw

Cache-Control	no-store, no-cache, must-revalidate
Connection	keep-alive
Content-Type	text/html; charset=UTF-8
Date	Mon, 14 Jul 2025 03:15:25 GMT
Expires	Thu, 19 Nov 1981 08:52:00 GMT
Pragma	no-cache
Server	nginx/1.29.0
Set-Cookie	PHPSESSID=3dc458c8796774966de9e7a8f99c6d01; path=/
Transfer-Encoding	chunked
X-Powered-By	PHP/8.2.28

HTTP Response

setcookie : session key

- session_set.php

```
1  <?php
2  session_start();
```

session_start()는
Body 내 컨텐츠 출력 전 호출

Create cookie

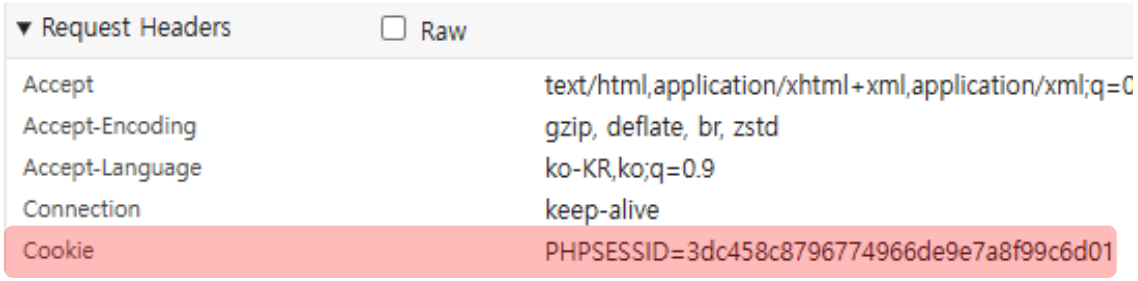
Name	Value
PHPSESSID	3dc458c8796774966de9e7a8f99c6d01

세션 동작 절차 (2)

Web browser

Web Server

HTTP Request : "session_set.php" + session key



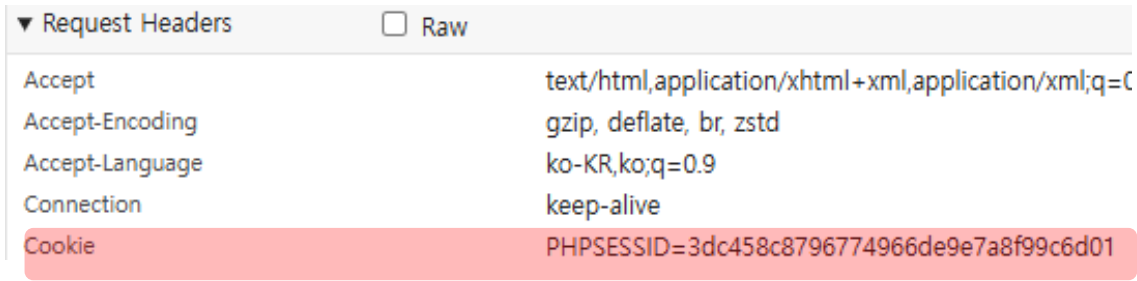
HTTP Response

- session_set.php

```php
<?php
session_start();

$_SESSION['id'] = "ycjung";
$_SESSION['name'] = "Youngchul Jung";
```

HTTP Request : "session_read.php" + session key



HTTP Response

- session_read.php

```
Array ( [id] => ycjung
[name] => Youngchul
Jung )
```

Session 관련 설정 옵션 : php.ini

- **Session 관련 환경 설정 옵션 : php.ini**

설정 옵션	설명	예시/기본값
session.auto_start	• PHP 시작 시 자동으로 세션 시작 여부	Off (기본값)
session.name	• 세션 쿠키 이름 (기본: PHPSESSID)	PHPSESSID
session.save_path	• 세션 데이터를 저장할 디렉토리	/tmp (리눅스 기준)
session.save_handler	• 세션 저장 방식 (files, redis, memcached 등)	files
session.gc_maxlifetime	• 세션 데이터의 유효 시간 (초 단위)	1440초 = 24분
session.cookie_lifetime	• 쿠키 자체의 유효 시간 (0이면 브라우저 종료 시 삭제)	0
session.cookie_path	• 세션 쿠키가 유효한 경로	/
session.cookie_domain	• 세션 쿠키가 유효한 도메인	생략 시 현재 도메인
session.cookie_secure	• HTTPS에서만 세션 쿠키 전송 여부	Off (보안 시 On 권장)
session.cookie_httponly	• JavaScript에서 document.cookie 접근 금지	On 권장
session.use_cookies	• 세션 ID 전달 방식에 쿠키 사용 여부	1 (사용함)
session.use_only_cookies	• 쿠키만 사용하고 URL에는 세션 ID 노출하지 않음	1 (보안상 권장)

Session 변수 등록 및 사용 하기

✓ **mysession.php**

```php
<?php
// 세션 시작 : 세션 기능을 사용 하기전 반드시 선 실행
session_start();

// 현 세션 내 데이터 저장
// 웹 버 측에 저장
$_SESSION['name']  = "Youngchul Jung";
$_SESSION['age']    = 22;
$_SESSION['univ']   = "Yeungjin Univ";
```

✓ **sessionprt.php**

```php
<?php
session_start();

// 현 세션에 저장 된 세션 값 출력
echo $_SESSION['name']."<br>";  // Youngchul Jung
echo $_SESSION['age']."<br>";   // 22
echo $_SESSION['univ']."<br>";  // Yeungjin Univ

// 현 세션 ID 값 출력 : s5k4k7losgno1av1sshs3o27m7
echo session_id();
```

✓ $_SESSION 슈퍼 글로벌 전역 변수를 활용하여 현 세션 Key 값과 연계 된 Session 배열 값에 사용자 정보 저장

Session 변수 제거하기 : 전체 삭제

✓ **mysession1.php**

```php
<?php
// 세션 시작 : 세션 기능을 사용 하기전 반드시 선 실행
session_start();

// 웹 서버 측에 저장
$_SESSION['name']  = "Youngchul Jung";
$_SESSION['age']    = 22;
$_SESSION['univ']   = "Yeungjin Univ";
```

✓ **mysession2.php**

```php
<?php
session_start();

// 현 세션에 저장 된 변수 값 출력
foreach($_SESSION as $key => $value)
    echo $key.":".$value."<br>";
```

name:Youngchul Jung
age:22
univ:Yeungjin Univ

✓ **mysession3.php**

```php
<?php
session_start();
session_destroy(); // $_SESSION 배열 모든 값 삭제
```

✓ **mysession2.php**

```php
<?php
session_start();

// 현 세션에 저장 된 변수 값 출력
foreach($_SESSION as $key => $value)
    echo $key.":".$value."<br>";
```

Session 변수 제거하기 : 부분 삭제

✓ **mysession1.php**

```php
<?php
// 세션 시작 : 세션 기능을 사용 하기전 반드시 선 실행
session_start();

// 웹 서버 측에 저장
$_SESSION['name']  = "Youngchul Jung";
```

✓ **mysession2.php**

```php
<?php
session_start();

// 현 세션에 저장 된 변수 값 삭제
unset($_SESSION['name']);
```

✓ **mysession3.php**

```php
<?php
session_start();

// 현 세션에 저장 된 세션 값 출력
echo $_SESSION['name']."<br>";  // error undefined index
```

클라이언트↔ 서버 간 세션 삭제 예제

```php
<?php
// 1. 세션 시작 (기존 세션이 있으면 불러옴)
session_start();

// 2. 세션 변수 초기화 (메모리 상의 데이터 제거)
$_SESSION = [];

// 3. 서버의 세션 파일 삭제 (세션 자체 파기)
session_destroy();

// 4. 클라이언트 측 쿠키 삭제
if (ini_get("session.use_cookies")) {
    // 현재 세션 쿠키의 설정 정보 가져오기
    $params = session_get_cookie_params();

    // 세션 쿠키 삭제 (만료 시간: 과거)
    setcookie(session_name(), '', [
        'expires' => time() - 3600,              // 쿠키 만료
        'path' => $params['path'],               // 경로 맞춰야 삭제됨
        'domain' => $params['domain'],           // 도메인 맞춰야 삭제됨
        'secure' => $params['secure'],           // HTTPS 사용 여부
        'httponly' => $params['httponly'],       // JavaScript 접근 방지
        'samesite' => $params['samesite'] ?? 'Lax' // PHP 7.3+용 옵션 (옵션)
    ]);
}
```

- 쿠키를 제대로 삭제하려면 path, domain을 정확히 지정해야 한다

Lab 3: Lab 2 쿠키 버전을 Session 버전으로 구현

- order_form.php: 입력 폼
- set_session.php: 세션 저장
- order_view.php: 확인 화면
- order_edit.php: 수정 폼
- clear_session.php: 세션 초기화

Extra Slides

header() 함수란?

- PHP에서 HTTP 헤더를 전송할 때 사용하는 함수
- 주로 페이지 이동, 파일 다운로드, 캐시 제어 등의 목적으로 사용
- 주의: header()는 반드시 출력 전에 호출해야 합니다 (공백, HTML 등 출력이 있으면 안 됨)

```php
header(string $header, bool $replace = true, int $response_code = 0);
```

매개변수	설명
$header	전송할 헤더 문자열
$replace	기존에 동일한 헤더가 있으면 덮어쓸지 여부 (기본값: true)
$response_code	상태 코드 (예: 200, 404 등)

📌 **페이지 리디렉션 (Redirect)**

```php
header("Location: home.php");
exit;  // 꼭 exit로 스크립트 종료!
```

📌 **파일 다운로드 강제 처리**

```php
header("Content-Disposition: attachment; filename=sample.txt");
header("Content-Type: text/plain");
readfile("sample.txt");
exit;
```

Q/A

감사합니다

주문식교육의 산실

영진전문대학교