jinyPHP 1권 - 13. 쿠키 세션

**13 쿠키 세션**

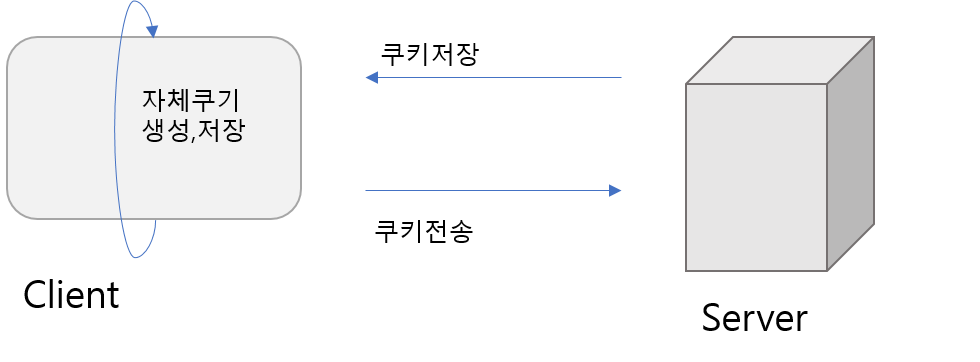
요즘 대부분의 웹 사이트들은 서버와 사용자 간 데이터를 주고 받습니다. 또한 일부 웹 사이트는 접속하는 사용자의 데이터를 임시로 저장하여 데이터 처리를 계속 하거나 사용자를 구별하기도 합니다. 예를 들어, 회원 로그인 상태, 쇼핑몰의 장바구니 등이 그렇습니다. 웹 서비스에서 임시 데이터를 저장하는 방법으로 자주 사용하는 기능은 쿠키 기능과 세션 기능입니다. 쿠키와 세션은 웹 사이트를 개발, 설계할 때 가장 중요하고 많이 사용하는 기능입니다.

**13.1 쿠키**

예전부터 가장 많이 사용하는 데이터 임시 저장 방법으로는 쿠키 기능을 사용합니다. 쿠키(cookie)는 웹 사이트를 접속한 사용자 임시 데이터를 브라우저에 저장하는 방법입니다.

쿠키는 매우 작은 4KB 이하의 데이터 크기입니다. 사용자의 접속된 컴퓨터에 저장됩니다. 4KB의 용량은 대략 300개 정도의 정보를 간단한 데이터를 저장할 수 있습니다.

쿠키는 주로 접속한 페이지의 상태 등을 저장을 할 때 사용합니다. 매번 사용자가 서버로 페이지 요청을 할 때 쿠키 정보도 같이 서버로 전송됩니다.



쿠키는 PHP 이외에 자바스크립트를 통해 클라이언트에서 직접 설정 및 변경도 가능합니다. 이러한 점으로 인하여 쿠키는 특정한 사용자 및 해킹 등에 의해서 조작될 수도 있습니다.

따라서 쿠키는 중요하지 않은 임시 데이터 정보를 저장할 때 적당합니다.

**13.1.1 쿠키의 저장**

쿠키는 접속한 사용자 컴퓨터에 저장되는 데이터 입니다. 그렇기 때문에 접속자 브라우저가 실행되고 있는 경우에만 존재합니다.

PHP 언어는 간단한 쿠키 내부 함수를 통해 쿠키 값을 설정할 수 있습니다. 생성된 쿠키 값은 접속 사용자 브라우저에게 쿠키를 전송하고 생성됩니다.

**|관련함수|**

|  |
| --- |
| bool **setcookie**(name, value, expire, path, domain, secure, httponly); |

PHP의 setcookie() 함수를 통해 쿠키를 설정하기 위해서 몇 개의 인자 값이 필요합니다. 또한 쿠키의 실행 결과 값을 논리값으로 반환할 수 있습니다.

* name = 입력 인자 중 name값은 필수 항목입니다.
* value = 쿠키의 이름과 관련된 값
* expire = 쿠키 유효 기간. expire를 기술하지 않으면 현재 웹 브라우저가 실행되는 동안만 유효함
* path = 쿠키를 전송할 요청 경로
* domain = 쿠키를 전송할 도메인
* secure
* httponly

그 외 나머지는 인자 파라미터는 옵션입니다.

**쿠키 이름 및 규칙**

쿠키 이름을 생성할 때 작명하는 이름은 RFC2109 규약을 따라서 만들어야 합니다.

쿠키 이름은 아스키 코드 알파벳 및 숫자만 포함할 수 있습니다. 즉 콤마(,), 세미콜론(;), 공백('') 등의 문자는 포함할 수 없습니다. 또한 $로 시작할 수 없습니다.

**예제 파일** cookie-01.php

|  |
| --- |
| <?php  $cookie\_name = "user";  $cookie\_value = "jiny";  if (**setcookie($cookie\_name, $cookie\_value, time() + (86400 \* 30), "/")**) {  // 86400 = 1 day  echo "쿠키 생성 성공";  } else {  echo "쿠키 생성 실패";  }  ?> |

**결과**

쿠키 생성 성공

위의 예제는 쿠키 생성에 대한 실험입니다. 예제 코드를 보면 쿠키 이름 "user"에 사용자 이름 "jiny"를 30일(86400 \* 30) 유효 기간으로 설정하는 단계입니다. 뒤의 "/"는 웹 사이트 전체에서 사용 가능하다는 의미입니다.

별도의 서브 경로에서만 지정하려고 하면 path를 변경하면 됩니다.

setcookie() 함수를 사용하면서 주의해야 할 점은 웹 사이트의 <html> 태그 출력 전에 먼저 쿠키 함수를 실행해야 됩니다.

쿠키 값은 전송 시 자동적으로 URL 인코딩하여 전송합니다. 또한 받을 때도 자동적으로 URL 인코딩되어 받습니다. 만일 인코딩하지 않으려면 setrawcookie() 함수를 이용하면 됩니다.

**13.1.2 쿠키 값 확인**

PHP는 setcookie() 함수로 생성한 쿠키를 **슈퍼 변수 $\_COOKIE**를 통해 확인할 수 있습니다.

$\_COOKIE는 배열 변수입니다. 생성한 쿠키명을 배열의 키로 입력하면 설정한 쿠키 값을 확인할 수 있습니다.

|문법|

**$\_COOKIE[쿠키명];**

PHP에서 쿠키 값을 이용하는 방법은 위처럼 매우 간단합니다. 다음 예제는 설정한 쿠키 값을 슈퍼변수를 통하여 확인하는 방법입니다.

**예제 파일** cookie-02.php

|  |
| --- |
| <?php  if (isset($\_COOKIE['user'])) {  echo $\_COOKIE['user'];  } else {  echo "쿠키가 없습니다.";  }  ?> |

**결과**

jiny

위의 예에서 보면 앞에서 설정한 쿠키 값을 슈퍼 변수를 통해 간단하게 읽어올 수 있습니다. 그냥 배열 값으로 읽으면 됩니다.

슈퍼 변수 $\_COOKIE는 다수의 쿠키 값의 데이터를 저장하고, 읽어올 수 있습니다. 비록 쿠키가 많은 양의 데이터를 저장할 수 는 없지만 필요한 개수만큼 저장하여 배열로 읽어오면 됩니다.

쿠키값 을 사용할 때는 프로그램 오류를 방지하기 위하여 쿠키 값을 확인 후에 처리하는 것이 좋습니다. 쿠키 값도 존재 여부를 isset()와 같은 함수를 통해 확인할 수 있습니다.

**13.1.3 쿠키 값 변경**

한 번 생성한 쿠키 값을 변경할 수 있습니다.

쿠키 값을 변경하는 방법은 쿠키를 생성하는 것도 동일합니다. 다만, 동일한 쿠키명으로 쿠키를 생성하게 되면 쿠키 값이 변경 갱신된 효과가 됩니다.

**예제 파일** cookie-03.php

|  |
| --- |
| <?php  if (isset(**$\_COOKIE['user']**)) {  echo $\_COOKIE['user'];  } else {  echo "쿠키가 없습니다.";  }  echo "<br>";  $cookie\_name = "user";  $cookie\_value = "jinyPHP";    if (**setcookie($cookie\_name, $cookie\_value, time() + (86400 \* 30), "/")**) {  // 86400 = 1 day  echo "쿠키 변경 성공";  } else {  echo "쿠키 변경 실패";  }  ?> |

**결과**

jiny

쿠키 변경 성공

쿠키를 새로 설정하여 변경했다고 해서 쿠키의 값이 바로 슈퍼변수에 저장되어 반영되지는 않습니다. 따라서 쿠키 설정 다음에 바로 다음에 변경된 쿠키 값을 확인할 수는 없습니다.

변경된 쿠키 값을 반영하기 위해서는 현재의 페이지 화면을 새로고침해야 하지만 새롭게 변경된 쿠키 값을 적용하여 확인할 수 있습니다.

이러한 쿠키 변경과 바로 사용할 수 없는 점은 코드 작성상 불편함이 있습니다. 약간의 편법으로 쿠키 값을 새로운 변수에 재할당하여 사용을 하면 이러한 불편함을 보완할 수 있습니다.

**예제 파일** cookie-04.php

|  |
| --- |
| <?php  if (isset($\_COOKIE['user'])) {  $\_\_cookie\_user = $\_COOKIE['user'];  } else {  echo "쿠키가 없습니다.";  }  echo "현재의 쿠키 값 = $\_\_cookie\_user<br>";  $cookie\_name = "user";  $\_\_cookie\_user = "jinyPHP";  if (setcookie($cookie\_name, $\_\_cookie\_user, time() + (86400 \* 30), "/")) {  // 86400 = 1 day  echo "쿠키 변경 성공";  } else {  echo "쿠키 변경 실패";  }  echo "쿠키 변경 후 다시 확인 = ".$\_COOKIE['user']."<br>";  echo "페이지 재로드하면 변경된 쿠키를 확인합니다. <br>";  echo "쿠키 값 별도에 저장 = ". $\_\_cookie\_user."해서 사용하면 편리합니다. ";  ?> |

**결과**

현재의 쿠키 값 = jiny

쿠키 변경 성공

쿠키 변경 후 다시 확인 = jiny

페이지 재로드하면 변경된 쿠키를 확인합니다.

쿠키 값 별도에 저장 = jinyPHP해서 사용하면 편리합니다.

위의 예제에서 쿠키 값을 바로 사용하는 것이 아니라 $\_\_cookie\_user 와 같이 별개의 변수를 이용하여 쿠키 값을 생성하고 값을 이용하여 처리하면 쿠키 값 반영에 대한 불편함을 해결할 수 있을 것입니다.

**13.1.4 쿠키 삭제 및 만료**

쿠키는 특이하게도 쿠키의 유효 기간이 있습니다. 유효시간이 지나면 쿠키의 값은 자동으로 소멸하거나 적용되지 않습니다.

PHP에서 setcookie() 함수를 사용하여 쿠키를 생성할 때 쿠키의 만료 시간을 함께 설정했습니다. 쿠키는 만료 시간이 지나면 자연스럽게 삭제됩니다.

또한 쿠키 값을 강제로 삭제하기 위해서는 쿠키 값의 만료 시간을 현재 시간 이전으로 변경 재설정하면 시간 만료로 인해 쿠기가 삭제됩니다.

예)

|  |
| --- |
| setcookie("user", "", **time() - 3600**); |

**13.1.5 쿠키 체크**

슈퍼변수 $\_COOKIE는 배열로 되어 있습니다. 배열 값 하나하나를 검사를 할 수도 있지만, 배열의 개수를 확인할 수 있는 count() 함수로 쿠키의 존재 여부를 확인할 수 있습니다.

**예제 파일** cookie-05.php

|  |
| --- |
| <?php  if (count($\_COOKIE) > 0) {  echo "Cookies 존재. <br>";  print\_r($\_COOKIE);  } else {  echo "Cookies 없음. <br>";  }  ?> |

**결과**

Cookies 존재.

Array ( [user] => jinyPHP )

위의 예는 쿠키 슈퍼변수의 개수를 파악하여 쿠키가 존재하는지 체크하는 간단한 예입니다.

**13.2 자바스크립트 쿠키**

앞서 설명한 것과 같이 쿠키는 접속한 사용자 브라우저에 저장되어 있는 매우 작은 데이터 파일 입니다. 접속자의 컴퓨터에 저장된 임시 데이터의 특성에 따라서 PHP 언어 이외에 클라이언트에서 실행 가능한 다른 언어로도 쿠키 값은 변경될 수 있습니다. 주로 가장 많이 사용하는 클라이언트 쿠키 조작 방법으로는 자바스크립트가 있습니다. 자바스크립트를 이용하여 쿠키 값을 설정, 변경, 삭제 등이 가능합니다.

다음은 간단한 자바스크립트 쿠키 소스 예제 입니다.

**자바스크립트 쿠키 읽기**

|  |
| --- |
| // 쿠키 값 읽기  function getCookie( cookieName ){  // @param cookieName 쿠키명  var search = cookieName + "=";  var cookie = document.cookie;  // 현재 쿠키가 존재할 경우  if( cookie.length > 0 ){  // 해당 쿠키명이 존재하는지 검색한 후 존재하면 위치를 리턴.  startIndex = cookie.indexOf( cookieName );  // 만약 존재한다면  if( startIndex != -1 ){  // 값을 얻어내기 위해 시작 인덱스 조절  startIndex += cookieName.length;  // 값을 얻어내기 위해 종료 인덱스 추출  endIndex = cookie.indexOf( ";", startIndex );  // 만약 종료 인덱스를 못 찾게 되면 쿠키 전체 길이로 설정  if( endIndex == -1) endIndex = cookie.length;  // 쿠키 값을 추출하여 리턴  return unescape( cookie.substring( startIndex + 1, endIndex ) );  } else {  // 쿠키 내에 해당 쿠키가 존재하지 않을 경우  return false;  }  } else {  // 쿠키 자체가 없을 경우  return false;  }  } |

**자바스크립트 쿠키 설정**

|  |
| --- |
| // 쿠키를 설정합니다.  function setCookie( cookieName, cookieValue, expireDate ){  // @param cookieName 쿠키명  // @param cookieValue 쿠키 값  // @param expireDay 쿠키 유효 날짜  var today = new Date();  today.setDate( today.getDate() + parseInt( expireDate ) );  document.cookie = cookieName + "=" + escape( cookieValue ) + "; path=/; expires=" + today.toGMTString() + ";";  } |

**자바스크립트 쿠키 삭제**

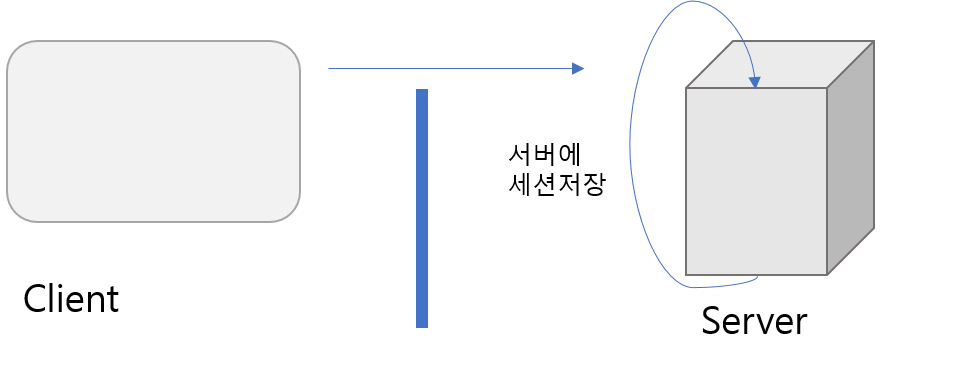
|  |
| --- |
| // 쿠키 삭제  function deleteCookie( cookieName ){  // @param cookieName 삭제할 쿠키명  var expireDate = new Date();  //어제 날짜를 쿠키 소멸 날짜로 설정한다.  expireDate.setDate( expireDate.getDate() - 1 );  document.cookie = cookieName + "= " + "; expires=" + expireDate.toGMTString() + "; path=/";  } |

**13.3 세션**

쿠키는 접속자 컴퓨터에 저장되는 임시 데이터입니다. 이렇게 외부에 임시 저장된 쿠키는 다른 사용자 또는 해킹에 의해서 조작될 수 있는 문제점들이 있습니다.

이런 이유로 쿠키는 회원 정보와 같은 중요한 내용을 담기에는 보안상 위험합니다. 웹 초기에는 쿠키의 변경으로 웹 사이트 해킹 등 문제점이 있었습니다.

세션(Sessions)은 이러한 문제점을 보완하기 위해 민감한 정보를 접속자의 컴퓨터가 아닌 서버에 저장하는 기술입니다.



세션은 접속자의 브라우저가 실행되고 있는 동안 유지됩니다.

**13.3.1 Session 시작하기**

PHP에서 세션을 사용하기 위해서는 세션의 사용을 스크립트 시작 전에 미리 알려줘야 합니다.

스크립트 실행 상단에 session\_start()함수를 호출합니다. 세션 시작을 호출하면 이후 PHP 스크립트 전반에서 세션 기능을 사용할 수 있습니다.

예제파일: session-01.php

|  |
| --- |
| <?php  // 세션 시작  session\_start();  ?> |

세션 시작 함수는 HTML태그 출력 전에 실행되어야 합니다.

**13.3.2 세션 값 설정**

PHP에서 세션 값을 설정하는 것은 쿠키 사용법 보다 더 매우 쉽습니다.

세션 또한 쿠키와 같이 슈퍼변수 배열 타입으로 되어 있습니다. $\_SESSION 배열변수는 PHP 스크립트 전반에서 별도의 특별한 지정이나 선언 없이도 사용할 수 있습니다.

**$\_SESSION[세션명] = 세션 값;**

세션 이름을 $\_SESSION 배열의 키 값으로 설정하고 값을 입력하면 됩니다.

**13.3.3 세션 값 불러오기**

설정한 세션 값은 $\_SESSION 슈퍼변수의 배열 키 값을 입력하면 됩니다.

**예제 파일** session-02.php

|  |
| --- |
| <?php  session\_start();    **$\_SESSION["user"]** = "jiny";  echo **$\_SESSION["user"];**  ?> |

**결과**

jiny

세션은 서버에 임시 데이터를 저장합니다. 또한 브라우저가 실행되고 있는 동안 유지됩니다.

설정한 세션 값은 여러 PHP스크립트 파일 에서도 값을 공유하여 사용할 수 있습니다.

**예제 파일** session-03.php

|  |
| --- |
| <?php  session\_start();    echo $\_SESSION["user"];  ?> |

**결과**

jiny

위의 예제는 session-01.php에서 설정한 세션 값을 session-02.php에서도 확인할 수 있습니다. PHP 스크립트 언어는 페이지 단위로 로딩 처리되기 때문에 A 페이지에서 작업한 내용을 B 페이지에 별도의 데이터 작업 없이 값을 전달하는 것으로 매우 편리합니다.

**13.3.4 세션의 동작 원리**

대부분 세션은 사용자 브라우저 접속 시 user-key 형태로 생성됩니다. 세션 키는 765487cf34ert8daccb38654f3a7e12처럼 알아볼 수 없는 형태를 가집니다.

만일 페이지가 변경되거나 새로운 창이 생성될 때 동일한 세션을 가지고 있으면 user-key에 접근할 수 있고, 아닌 경우에는 새로운 세션이 생성됩니다.

**13.3.5 세션 값 변경**

세션 값의 변경은 새로운 세션을 선언하는 것과 동일합니다. 세션을 설정하는 것처럼 슈퍼변수 값에 그냥 새로운 값을 저장하면 됩니다.

예제파일:session-04.php

|  |
| --- |
| <?php  session\_start();  $\_SESSION["user"] = "abcd";  ?> |

위의 예는 세션 user 의 값을 변경하는 것입니다. 만일 다른 페이지에서 설정한 값이 없다고 하면 처음으로 세션을 생성한 것으로 볼 수 있습니다.

**13.3.6 세션 삭제**세션 값을 삭제하기 위해서는 session\_unset() 함수와 session\_destroy() 함수를 이용하면 됩니다.

예제파일:session-05.php

|  |
| --- |
| <?php  // remove all session variables  session\_unset();  // destroy the session  session\_destroy();  ?> |

위 함수를 실행하면 모든 세션의 값을 제거하거나 파괴할 수 있습니다. 만일 개별 세션만 삭제하고자 하는 경우에는 세션 값에 null 또는 다른 값을 넣어주면 됩니다.

**13.3.7 세션 옵션**

세션의 설정 값은 PHP의 환경 설정 파일 php.ini에 정의되고 영향을 받습니다. 세션의 설정을 변경하기 위해서는 직접 php.ini를 수정해야 합니다. 하지만 일부 임시로 세션 설정을 변경하는 경우라면 직접 php.ini 파일을 수정하는 것은 어려울 것입니다. 따라서 PHP 7.x부터는 php.ini의 값을 프로그램에서 동적으로 임시 재설정하여 실행할 수 있습니다.

세션 시작 함수 session\_start() 호출 시 설정 배열 인자 값을 전달해주면 세션 환경을 변경하여 실행할 수 있습니다.

예제파일:session-06.php

|  |
| --- |
| <?php  session\_start([  'cache\_limiter' => 'private',  'read\_and\_close' => true,  ]);  ?> |

예를 들어 session.cache\_limiter 값을 'private'으로 변경하고 참조한 경우에 세션을 직접 종료할 수도 있습니다.