jinyPHP 1권 - 19. 웹 페이지

**19. 웹 페이지**

PHP는 초기 시작은 웹 사이트를 쉽게 개발할 수 있는 C 언어 스타일의 서버 사이드 스크립트입니다.

PHP는 다양 데이터를 처리하여 웹 사이트로 화면을 출력할 수 있습니다. 전 세계 수많은 웹 사이트는 이미 PHP로 개발, 운영되고 있습니다.

**19.1 페이지 출력**

기존 웹 페이지는 HTML 마크업과 .htm 확장자를 가지고 있습니다. 또한 이러한 웹 페이지들을 웹 서버는 클라이언트 브라우저로 전달하고 화면을 출력하게 됩니다.

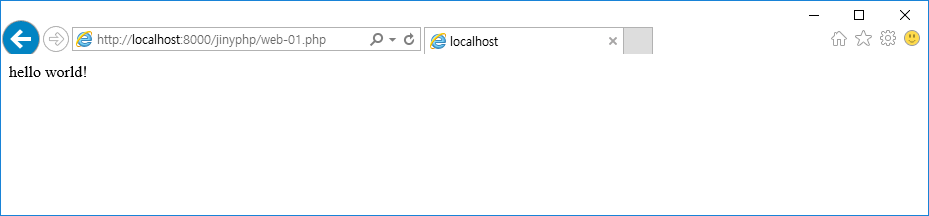
PHP는 HTML과 궁합이 잘 맞는 개발 언어 중 하나입니다. 기존 HTML 페이지 안 어디에서든지 <?php ~~ ?>를 삽입하게 되면 웹 서버가 PHP 코드를 인식하여 동적으로 처리합니다.

지금까지 배운 문법과 구조를 쉽게 기존 페이지에 적용할 수 있습니다.

**예제 파일** web-01.php

|  |
| --- |
| <html>  <head>    </head>  <body>  **<?php**  echo "hello world!";  **?>**  </body>  </html> |

**화면 출력)**



위 예제를 보면 웹 서버가 web-01.php html 페이지를 브라우저로 전송하기 전에 <?php ~ ?> 영역을 스크립트 처리하여 브라우저로 전송하게 됩니다.

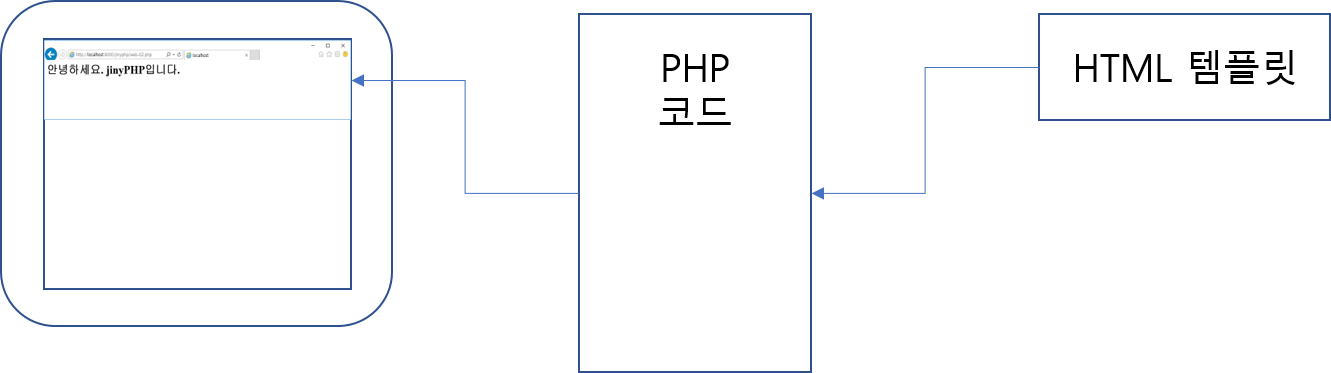
**19.2 템플릿 작업**

PHP와 HTML이 혼용되어 있는 코드가 있는 반면 순수한 PHP 코드도 있습니다.

예전에는 HTML 페이지 안에 PHP코드를 삽입하여 개발하는 형태로 작업을 많이 했습니다. 하지만 HTML 안에 PHP 코드가 들어 있으면 가독성이 떨어지고 유지보수도 힘듭니다. 또한 디자이너와 협업하여 개발하기에도 어려운 부분이 있었습니다.

요즘 트렌드는 MVC 패턴이라고 해서 화면에 보이는 view 영역 (HTML)과 데이터와 연산을 처리하는 M/C(PHP)를 분리하여 작업하는 것을 선호합니다.

PHP와 HTML을 분리하여 가장 간단하게 웹 페이지를 표현하는 방법은 HTML을 템플릿화하여 처리하는 것입니다. 화면에 보이는 부분을 HTML 파일로 저장해 놓고, PHP에서 파일로 읽어 화면에 출력하는 것입니다.



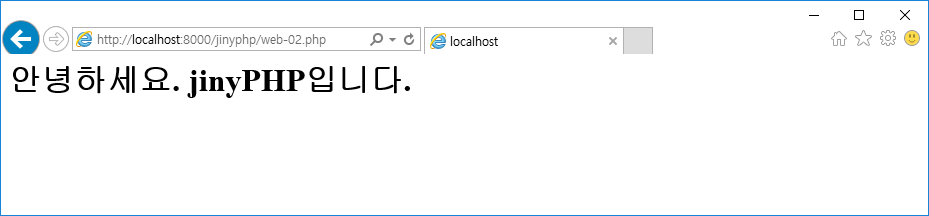
**예제 파일** web-02.php

|  |
| --- |
| <?php  class JinyFiles {  public function fileLoad($filename)  {  if ($fp = fopen($filename, "r")){  if ($fp){  while (!feof ($fp)) $buffer .= fgets($fp, 4096);  fclose($fp);  return $buffer;  }  } else {  echo "파일을 읽어올 수 없습니다.";  }  }  }    $temp = new JinyFiles();  echo $temp->fileLoad("./temp.htm");  ?> |

**예제 파일** temp.htm

|  |
| --- |
| <html>  <head>    </head>  <body>  <h1>안녕하세요. jinyPHP입니다.</h1>  </body>  </html> |

**화면 출력)**



위의 예는 간단한 html 뷰와 코드를 분리한 예제입니다. 외부에 별도로 생성한 view용 html 파일을 읽어서 화면에 출력합니다. 이러한 방식을 통해 php 코드와 html 을 분리하여 관리합니다.

**19.3 코드 치환**

템플릿을 이용하여 HTML 파일 그대로 내용을 다시 재출력하는 것입니다.

하지만 이러한 방식의 문제는 동적인 데이터를 가지고 있는 HTML경우에는 데이터 처리가 어렵다는 것입니다. 그래서 이전에는 html 코드 안에 <?php ~ ?>를 삽입하여 코드를 처리했습니다.

템플릿에서 동적으로 처리하고자 하는 내용이 있다고 하면 **치환 키워드**를 만들어 사용합니다. 기존 html 파일에 특정한 키워드를 만들어 놓고, PHP에서 화면을 출력하기 전에 치환 키워드를 동적 데이터로 변경하는 방식입니다.

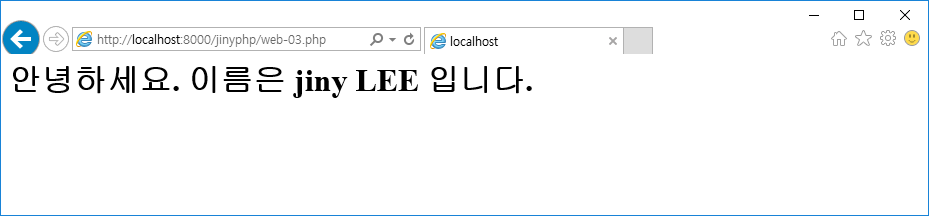
**예제 파일** web-03.php

|  |
| --- |
| <?php  class JinyFiles {  public function fileLoad($filename)  {  if ($fp = fopen($filename, "r")){  if ($fp){  while (!feof ($fp)) $buffer .= fgets($fp, 4096);  fclose($fp);  return $buffer;  }  } else {  echo "파일을 읽어올 수 없습니다.";  }  }  }    $temp = new JinyFiles();  $body = $temp->fileLoad("./temp-02.htm");  // 치환 키워드를 변경합니다.  $name = "jiny LEE";  $body = str\_replace("{(userName)}",$name,$body);  echo $body;  ?> |

**예제 파일** temp-02.htm

|  |
| --- |
| <html>  <head>    </head>  <body>  <h1>안녕하세요. 이름은 {(userName)} 입니다.</h1>  </body>  </html> |

**화면 출력**



위의 예제를 보면 템플릿 파일을 읽어서 변수에 저장합니다. 변수는 html 본문 코드를 담고 있습니다.

문자열을 치환하는 내부 함수를 통해 치환 키워드를 찾아서 데이터로 변경한 후 화면에 출력합니다.

str\_replace() 함수는 문자열의 일치되는 내용을 찾아서 다른 문자로 바꿔주는 함수입니다.

|  |
| --- |
| $body = str\_replace(치환 문자, 대체 문자, $body); |

가변적으로 처리해야 하는 내용은 템플릿에서 치환 문자로 넣어둡니다. PHP는 템플릿 파일을 읽은 후에 치환 문자를 실제적인 데이터로 변경 후 화면에 출력하면 가변적인 화면을 구현할 수 있습니다.

**19.4 폼 입력**

====================

HTML의 <form> 태그는 브라우저를 통해 사용자 입력을 받아서 서버로 전송하는 기능을 합니다. 폼 요소는 HTML 에서 예전부터 사용을 해오던 클라이언트와 서버 간 통신 방법 중 하나입니다.

폼 태그를 사용하는 형식은 아래와 같습니다.

<form >

폼 요소…

<submit>

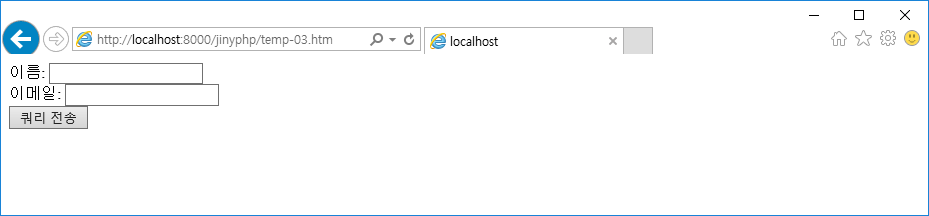
</form>

HTML에서 폼 요소들은 <form></form> 태그로 감싸져 있습니다. 또한, 폼 태그 안에는 폼 정보를 서버로 전송하는 버튼 역할의 <submit> 요소를 추가로 하나 더 가지고 있습니다. 서브밋(submit) 버튼을 클릭하면 폼의 데이터를 서버로 전송합니다.

**예제 파일** temp-03.htm

|  |
| --- |
| <html>  <body>  <form action="web-04.php" method="post">  이름: <input type="text" name="name"><br>  이메일: <input type="text" name="email"><br>  <input type="submit">  </form>    </body>  </html> |

**화면 출력**



위의 예처럼 temp-03.htm 파일은 폼 태그를 가지고 있는 웹 페이지입니다. 이름과 이메일을 입력받을 수 있는 입력 창과 전송 확인을 할 수 있는 서브밋 버튼을 하나 출력합니다.

폼 태그는 다양한 속성을 설정할 수 있습니다. 그 중에서 action이라는 설정 부분이 있습니다. action 속성은 폼 페이지에서 사용자가 전송 버튼을 클릭했을 때 서버 쪽에서 데이터를 받아 처리할 수 있는 스크립트의 URL입니다. action은 입력한 데이터를 전달하는 페이지 주소를 설정하는 속성입니다.

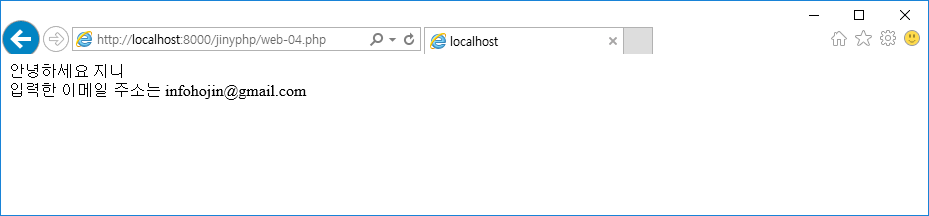
만일 action=”web-04.php” 형태로 설정했다면 클릭 시 web-04.php 파일로 데이터와 함께 페이지를 처리한다는 의미입니다.

그럼 데이터를 처리하는 web-04.php 파일의 내용을 확인해 보도록 하겠습니다.

**예제 파일** web-04.php

|  |
| --- |
| <html>  <body>  안녕하세요 <?php echo $\_POST["name"]; ?><br>  입력한 이메일 주소는 <?php echo $\_POST["email"]; ?>  <body>  <html> |

**화면 출력**



위의 폼 요소를 가지고 있는 페이지에서 간단한 정보를 input 태그 입력창에 데이터를 입력한 후 전송(submit)을 클릭하면 web-04.php 파일로 url이 함께 이동합니다. 브라우저 상단 주소란에 보면 URL 주소가 변경되는 것을 확인할 수 있습니다.

폼 서브밋으로 페이지를 이동할 경우에는 url 변경과 같이 이전 페이지 form 요소의 데이터 또한 함께 전송합니다. 전송된 데이터의 타입 POST, GET 방식을 체크하여 폼 요소의 데이터를 읽어 올 수 있습니다. 입력한 데이터가 함께 전달된 데이터를 읽어서 화면에 출력이 가능합니다.

**19.5 폼 데이터**

HTML 폼에서 전송 클릭을 하면 입력한 데이터가 서버로 전송됩니다. 보통 웹 페이지에서 서버로 데이터를 전송하는 방법은 크게 두 가지가 있습니다. GET 방식과 POST 방식입니다. 폼(form) 태그에서는 데이터를 전송하는 방식을 사용자가 직접 속성값을 통해 지정할 수 있습니다.

**method="post"** 또는 **method="get"**

형태로 적어 주시면 됩니다. 지정한 전송 방식 POST 또는 GET으로 설정할 수 있습니다.

**19.5.1 GET**

GET 방식의 전송은url 페이지에서 데이터를 조회하는 용도로 많이 사용합니다. GET방식은 데이터를 전송 시에 URL 쿼리스트링 방식으로 전달하는 것을 말합니다. 즉 데이터가 URL 주소 형태로 변경되어 데이터를 전달합니다.

예) jinyphp.com/test.php?name=hojin&country=korea

실행 스크립트 뒤에 ? 기호를 추가하여 데이터의 유무를 확인합니다. 이름=값 형태의 한 쌍의 데이터로 이루어지며, 여러 데이터가 있을 경우에는 & 문자로 연결합니다.

이처럼 URL로 표현되는 GET 방식의 데이터 전송은 누구나 알 수 있는 데이터 표현식입니다. URL주소만 보면은 사용자가 입력한 데이터가 무엇인지, 데이터 변수는 어떻게 되는지 직관적으로 확인할 수 있습니다. 이러한 점 때문에 GET 방식은 중요하지 않은 데이터를 전송할 경우에 많이 사용합니다. 주로 데이터를 조회하는 데 필요한 필터링 정보만 전달합니다. 만일, 중요한 개인정보와 같은 데이터일 경우 GET 방식으로 전송하면 안 됩니다.

GET 방식은 2천여 개의 문자들로 데이터를 전송하기에 충분한 데이터 양이 아닙니다. 하지만 GET 방식은 데이터가 URL로 표시되기 때문에 페이지의 북마크나 공개 되도 되는 데이터 값을 호출할 때 유용합니다.

**19.5.2 POST**

POST는 GET과 달리 전송되는 데이터 양의 제한이 없습니다. 또한 비공개적으로 데이터를 전달하기 때문에 GET 방식보다 좀 더 안전합니다. POST 방식은 URL에 데이터를 표시하지 않습니다.

하지만 URL 형태로 처리되지 않기 때문에 폼을 서브 및 처리를 후에 history back 과 같은 처리 시 북마크가 정상적으로 동작하지 않을 수 있습니다.

또한 POST는 대량의 데이터를 처리할 수 있는 점을 이용하여 바이너리 같은 파일도 전송이 가능합니다. 특히 서버에 파일을 업로드할 수 있는 멀티 파트 바이너리와 같은 고급 기능을 지원합니다. 파일 업로드 기능을 사용할 때는 POST 방식으로 이용해야 합니다.

**19.5.3 글로벌 변수**

PHP언어는 HTML폼 요소로 전달되는 데이터를 쉽게 처리 할 수 있는 글로벌 슈퍼변수를 제공합니다. 글로벌 슈퍼변수를 이용하면 쉽게 전달되는 데이터를 접근하고 읽어 올 수 있습니다.

GET으로 전송할 경우에는 슈퍼변수 $\_GET 배열 사용이 가능합니다. 또한 POST로 전송할 경우에는 슈퍼변수 $\_POST 배열 사용이 가능합니다.

**19.5.4 REQUEST**

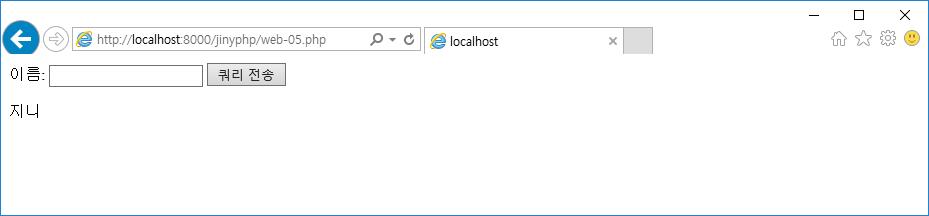
폼의 데이터가 전달될 때 POST로 전달된 것인지 GET 방식으로 전달된 것인지 확인이 필요합니다. 전달 방식에 따라서 $\_GET과 $\_POST를 구별해야 하기 때문입니다.

$\_SERVER["REQUEST\_METHOD"] 글로벌 변수는 폼의 전송 method 속성 값을 확인할 수 있습니다.

**예제 파일** web-05.php

|  |
| --- |
| <html>  <body>  <form method="post" action="<?php echo **$\_SERVER['PHP\_SELF']**;?>">    이름: <input type="text" name="username">  <input type="submit">  </form>  <?php    if (**$\_SERVER["REQUEST\_METHOD"] == "POST"**) {  // input 필드의 데이터를 가지고 옵니다.  $name = $\_POST['username'];  if (empty($name)) {  echo "이름이 입력되지 않았습니다";  } else {  echo $name;  }  }  ?>  </body>  </html> |

**화면 출력)**



위의 예제는 폼의 처리 데이터 유형이 POST인지를 확인하고 POST 값을 읽어와 처리합니다.

**19.6 $\_GET**

$\_GET은 PHP의 글로벌 슈퍼변수로 웹 개발 시 URL로 주어지는 폼의 데이터 값을 쉽게 글로벌 변수를 통해 받아 올 수 있는 특수 변수입니다.

브라우저에서 URL 입력 시 아래의 도메인 주소와 같이 ?와 & 기호를 함께 사용한 것을 본 적이 있을 것입니다. 통상적으로 html 페이지를 URL로 호출할 때 임의의 데이터를 같이 포함하여 전달할 수 있는데, 이렇게 전달되는 변수들을 GET 방식의 데이터 전달이라고 합니다.

http://www.jinyphp.com/list.php?board=notice&uid=1

위의 URL 주소를 보면 list.php 파일명 뒤에 ? 기호를 만나면 ? 뒤의 문자열은 GET 형태의 데이터를 전달한다는 의미입니다.

? 뒤의 문자열을 다시 정리해 보면,

boad=notice&uid=1

변수명 = 값 & 변수명 = 값 & 변수명 = 값

변수명과 데이터는 =로 연결되어 한 세트로 설정되며, 여러 변수를 연결할 때는 &로 연결할 수 있습니다.

즉, &가 많을수록 전달되는 변수들이 많다고 보면 됩니다.

이렇게 URL을 이용하여 전송된 GET 방식의 데이터는 PHP의 슈퍼 글로벌 변수 $\_GET 변수명으로 데이터를 가지고 올 수 있습니다.

$\_GET은 어레이 변수로 뒤에 ['변수명'] url로 입력된 변수명을 입력하면 됩니다.

http://www.jinyphp.com/list.php**?board=notice&uid=1** 으로 접속하면

$\_GET['**board**'] => "notice" 값을 읽어올 수 있습니다.

$\_GET['**uid**'] => "1" 값을 읽어올 수 있습니다.

PHP로 웹 페이지를 개발할 때 GET 방식의 데이터 전달은 아주 많이 사용하는 방법이고, 쉽게 PHP에 데이터를 전달할 수 있습니다.

페이지 만들어 보겠습니다.

예제파일: list.php

|  |
| --- |
| <html>  <body>  게시판 목록을 출력합니다.  <?php  echo "게시판 코드 = " . $\_GET['notice'] . "<br>";  echo "게시판 번호 = " . $\_GET['uid'] . "<br>";  ?>  </body>  </html> |

위의 예제를 실행하면 입력된 GET 값을 출력할 수 있습니다.

웹 페이지에서 GET 방식의 데이터 전송은 URL 방식 이외 form 태그의 **method="get"** 설정으로 데이터를 전달할 수 있습니다.

**19.7 $\_POST**

$\_POST는 PHP의 글로벌 변수로 웹 개발 시 가장 많이 사용하는 변수입니다. $\_POST는 HTML에서 데이터를 전달하는 방식으로 form에서 method="post"로 설정한 후에 submit을 통해 전달되는 데이터를 받을 수 있습니다.

GET 방식의 데이터 전달 방식은 URL에 자신이 전송하고자 하는 변수명과 데이터 값이 함께 표기되어 보안적인 면에서 취약점이 노출됩니다.

이런 점을 보완하고자 대부분의 웹 페이지에서 데이터를 전송할 때 POST 방식을 주로 이용합니다. 로그인 페이지, 회원가입 등 민감한 데이터일수록 POST 방식을 선호합니다.

PHP에서 POST 방식의 데이터는 PHP의 슈퍼 글로벌 변수 $\_POST 변수명으로 데이터를 가져올 수 있습니다.

$\_POST는 어레이 변수로 뒤에 ['변수명']으로 form에서 정의된 이름을 변수명으로 입력하면 됩니다.

예를 들면, 폼문에서 <input type=text name=username > 형태로 이름을 지정한 후 submit하면 PHP에서는 $\_POST['username'] 형태로 form의 입력된 데이터 값을 읽어올 수 있습니다.

$\_POST 글로벌 변수는 어레이 변수로 여러 개의 폼의 데이터를 담아서 한 번에 전달할 수 있습니다.

**폼 입력 예제** write.php

|  |
| --- |
| <html>  <body>  <form method="post" action="<?php echo $\_SERVER['PHP\_SELF'];?>">    Name: <input type="text" name="username">  <input type="submit">  </form>  <?php  if ($\_SERVER["REQUEST\_METHOD"] == "POST") {  // input 필드의 데이터를 가지고 옵니다.  $name = $\_POST['username'];  if (empty($name)) {  echo "이름이 입력되지 않았습니다";  } else {  echo $name;  }  }  ?>  </body>  </html> |

위의 write.php 예제를 실행하면 이름이 username으로 설정된 input 값을 submit한 경우에 입력된 값을 출력합니다.

form method="post" 폼의 메서드를 POST로 설정합니다. 그리고 submit할 때 실행할 페이지 동작 action=을 직접 write.php로 써 줄 수도 있지만 PHP의 슈퍼 글로벌 변수 $\_SERVER['PHP\_SELF']를 참조하여 현재의 자기 자신의 페이지로 설정합니다.

action 값을 직접 입력하는 방식 이외에 $\_SERVER['PHP\_SELF']를 사용하면 싱글 페이지 제작 시 별도로 매번 수정하지 않아도 자기 자신을 호출하기 때문에 편리합니다.

$\_SERVER["REQUEST\_METHOD"] 는 form 입력 시 METHOD 값을 확인할 수 있는 변수입니다. 만일 METHOD가 POST일 때 username input 값을 읽어옵니다. 읽어온 값을 확인해서 내용이 있으면 echo문을 이용하여 화면에 출력하고, 내용이 없는 경우는 "이름이 입력되지 않았습니다"를 표시합니다.