Chapter

11

통신

11.1 소켓 통신

소켓 통신이란 네트워크상에서 여러 다른 컴퓨터들의 프로세스 간의 통신 채널을 말합니다.

| 내장 함수 |

```
resource fsockopen ( string $hostname [, int $port = -1 [, int &$errno [, string
&$errstr [, float $timeout = ini_get("default_socket_timeout") ]]]] )
```

내장 함수 fsockopen()은 유닉스/리눅스 기반의 도메인 소켓을 오픈합니다. 도메인 서버와 포트를 응용하여 파일 포인트 형태로 반환합니다. 또한 접속 타임아웃도 함께 설정할 수 있습니다.

fsockopen() 함수를 통하여 오픈한 포인터는 fgets(), fgetss(), fputs(), fclose(), feof() 등과 같이 파일 처리하는 것과 유사하게 사용할 수 있습니다.

예제 파일 | **fsockopen.php**

```
<?php
    $fp = fsockopen("www.example.com", 80, $errno, $errstr, 30);
    if (!$fp) {
        echo "$errstr ($errno)<br />\n";
    } else {
        $out = "GET / HTTP/1.1\r\n";
        $out .= "Host: www.example.com\r\n";
        $out .= "Connection: Close\r\n\r\n";
        fwrite($fp, $out);

        while (!feof($fp)) {
            echo fgets($fp, 128);
        }
        fclose($fp);
    }

?>
```

| 내장 함수 |

resource **pfsockopen** (string $hostname [, int $port = -1 [, int &$errno [, string &$errstr [, float $timeout = ini_get("default_socket_timeout")]]]])

내장 함수 pfsockopen()은 지속적인 인터넷 또는 Unix 도메인 소켓을 연결합니다.

예제 파일 | **pfsockopen.php**

```
<?php

    $host = gethostbyaddr($_SERVER['REMOTE_ADDR']);

    $host = 'www.example.com';
    $service_uri = '/cgi/proACT';
    $vars ='code=23&act=TESTing';

    //# HTTP 요청 헤더 구성
    $header = "Host: $host\r\n";
```

```
    $header .= "User-Agent: PHP Script\r\n";
    $header .= "Content-Type: application/x-www-form-urlencoded\r\n";
    $header .= "Content-Length: ".strlen($vars)."\r\n";
    $header .= "Connection: close\r\n\r\n";

    $fp = pfsockopen("ssl://".$host, 443, $errno, $errstr);
    if (!$fp) {
        echo "$errstr ($errno)<br/>\n";
        echo $fp;
    } else {
        fputs($fp, "POST $service_uri  HTTP/1.1\r\n");
        fputs($fp, $header.$vars);
        fwrite($fp, $out);

        while (!feof($fp)) {
            echo fgets($fp, 128);
        }
        fclose($fp);
    }

?>
```

11.2 AJAX

AJAX는 Asynchronous JavaScript XML의 약자입니다. AJAX는 페이지의 재로딩 없이 비동기/동기 방식으로 페이지의 일부분을 갱신할 수 있는 웹 기술입니다.

보통 웹 페이지는 url 주소에 따라 웹 서버에서 페이지를 로딩하여 화면을 출력합니다. 특정한 페이지로 이동하거나, FROM 요소 등을 통하여 데이터를 입력 시 url 주소가 변경이 됩니다. 이때 브라우저는 페이지를 다시 로딩하여 화면을 재구성하게 됩니다.

PHP와 같이 자바스크립트의 AJAX 방식을 혼용하여 사용할 수 있습니다. url로 페이지 전체를 다시 로딩하는 방식이 아닌 페이지 일부분만 로딩하여 내용을 변경할 수 있습니다.

AJAX는 인터넷 표준을 기반을 따릅니다.

- XMLHttpRequest object(서버와 비동기 방식으로 데이터 교환)
- JavaScript/DOM(정보 표시 및 상호작용)
- CSS
- XML

11.2.1 Javascript & JQUERY

AJAX를 사용하기 위해서는 자바스크립트의 XMLHttpRequest 기능을 사용해야 합니다. 자바스크립트의 XMLHttpRequest 코드들은 코드가 복잡하고 브라우저마다 상호 특성에 따라 다르게 동작할 수 있습니다.

요즘 들어 AJAX를 쉽게 처리하기 위해서 자바스크립트 라이브러리 등을 많이 이용하여 사용합니다. 대표적으로 JQuery가 인기가 높습니다. JQuery는 복잡한 자바스크립트의 코드들을 쉽게 처리할 수 있으며 AJAX 통신 코드 또한 포함되어 있습니다(JQuery는 https://jquery.com/ 사이트를 참조). 또는 AJAX용으로 간결하게 제작된 AXIOS라는 라이브러리도 있습니다.

11.2.2 Javascript 예제

AJAX를 처리하기 위해서는 2개의 스크립트 파일이 필요로 합니다. 첫 번째 스크립트는 브라우저 접속 시 보여지는 기본 화면입니다. 두 번째 스크립트는 어떠한 동작이 있을 경우 AJAX로 처리되는 스크립트입니다.

예제 파일 | **ajax.php**

```
<html>
<head>
    <script>
    function ajax(str) {
        if (str.length == 0) {
            document.getElementById("txtHint").innerHTML = "";
            return;
```

```
        } else {
            var xmlhttp = new XMLHttpRequest();
            xmlhttp.onreadystatechange = function() {
                if (this.readyState == 4 && this.status == 200) {
                    document.getElementById("txtHint").innerHTML = this.
                     responseText;
                }
            };
            xmlhttp.open("GET", "ajax_email.php?q=" + str, true);
            xmlhttp.send();
        }
    }
    </script>
</head>

<body>

    <p><b>아래 이메일 주소를 입력해 주세요:</b></p>
    <form>
        email: <input type="text" onkeyup="ajax(this.value)">
    </form>
    <p>확인: <span id="txtHint"></span></p>
</body>
</html>
```

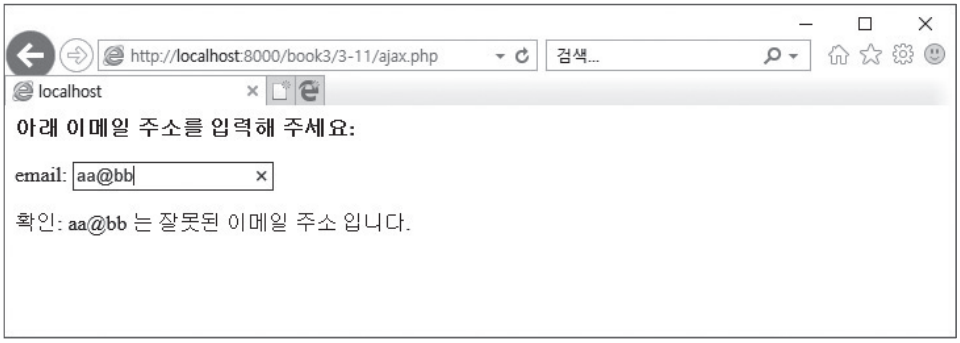
예제 파일 | **ajax_email.php**

```
<?php
   $email = $_GET['q'];

   if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
      echo "$email 정상적인 이메일 주소입니다.";
   } else {
      echo "$email 는 잘못된 이메일 주소입니다.";
   }

?>
```

화면 출력



위의 예제는 전형적인 자바스크립트의 XMLHttpReques 함수를 이용한 AJAX 처리입니다. 먼저 ajax.php로 접속 시 이메일 주소를 입력하는 폼이 하나 출력됩니다.

input 필드에 키를 한 글자씩 입력할 때마다 ajax_email.php 파일을 비동기 AJAX로 호출하여 이메일의 유효성을 체크합니다. 입력되는 이메일 문자는 GET 방식으로 전달합니다.

출력된 문자열 메시지를 자바스크립트의,

```
document.getElementById("txtHint").innerHTML = this.responseText;
```

처리로 DOM 내용을 갱신하여 출력합니다.

11.3 JQuery 예제

JQuery 라이브러리를 이용하면 좀 더 쉽게 AJAX를 처리할 수 있습니다. 다음 예제는 JQuery와 POST 방식의 전송 처리입니다.

JQuery를 사용하기 위해서는 라이브러리 파일을 스크립트로 삽입해야 합니다. 여기서는 CDN 방식으로 삽입해 보도록 하겠습니다.

예제 파일 | **ajax2.php**

```
<html>
<head>
    <!-- JQuery CDN -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/
    jquery.min.js"></script>

    <script>
        $(document).ready(function() {
            $("#email").keydown( function() {
                // AJAX 호출
                $.ajax({
                    url:'ajax_post.php',
                    type:'post',
                    data:$('form').serialize(),
                    success:function(data) {
                        $('#txtHint').html(data);
                    }
                });
            });

        });

    </script>
</head>

<body>
    <p><b>아래 이메일 주소를 입력해 주세요:</b></p>
    <form name='login' method='post' enctype='multipart/form-data'>
        email: <input type="text" name="email" id="email">
    </form>
    <p>확인: <span id="txtHint"></span></p>
</body>
</html>
```

예제 파일 | **ajax_post.php**

```
<?php
   $email = $_POST['email'];
```

```
3
4       if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
5           echo "$email 정상적인 이메일 주소입니다.";
6       } else {
7           echo "$email 는 잘못된 이메일 주소입니다.";
8       }
9
10  ?>
```