jinyPHP 3권 - 01.필터

**01. 필터**

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

안정적인 프로그램의 동작을 위해서는 기반이 되는 정확한 값의 안전한 데이터가 중요합니다. 잘못된 데이터들을 프로그램의 예상치 않은 오동작을 발생시킬 수 있습니다.

프로그램 코드를 작성하다 보면 데이터의 정확한 유효성을 체크하는 작업을 자주합니다. 데이터를 일정한 패턴으로 검사를 하는 것은 그렇게 간단하지 않습니다. PHP는 데이터의 유효한 형식을 분석하고 패턴을 검사할 수 있는 필터 함수들을 제공합니다.

또한, 웹서비스 용도로 많이 사용하는 PHP 언어의 특성상 데이터는 외부의 HTML 폼 입력을 통하여도 많이 입력이 됩니다. 불특정한 다수의 사람들로부터 데이터를 입력을 받을 때 이러한 데이터의 유효성과 패턴은 더욱더 중요합니다.

일부 악의적인 사용자가 잘못된 코드로 프로그램의 오류를 발생시키거나, SQL 쿼리 기호들을 썩어서 입력함으로써 데이터베이스의 내용을 해킹할 수도 있습니다.

PHP는 제공되는 필터 내부함수를 통하여 간단하게 데이터의 패턴을 검사하여 유효성을 확인 할 수 있습니다. 필터 함수 기능들은 PHP 5.2.0부터는 기본적으로 설치가 되어 있습니다. 하지만, 필터 기능과 동작은 php.ini의 환경설정에 의해 약간의 영향을 받습니다 ini 설정은 다음과 같습니다:

**filter.default**

filter.default 환경설정은 모든 $ \_GET, $ \_POST, $ \_COOKIE, $ \_REQUEST 및 $ \_SERVER 데이터를 필터링 합니다. 필터 이름을 기본적으로 사용허용합니다.

**filter.default\_flags**

filter.default\_flags 환경설정은 기본 필터로 설정이 된 경우 적용할 필터를 지정합니다. 이전 버전과의 호환성을 위해 기본적으로 FILTER\_FLAG\_NO\_ENCODE\_QUOTES로 설정됩니다.

**01.1 필터의 유용성**

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

필터는 데이터의 유효한 값을 간단하게 패턴 형태로 유효성 검사합니다. 필터 기능을 자주 적용하는 분야로는 웹을 데이터값을 입력 받는 분야일 것입니다. 웹서비스와 같은 응용서비스는 다양한 타입의 수많은 외부 데이터를 입력 받아 처리합니다.

* HTML 폼 입력값
* 쿠키값
* API 데이터
* 서버 변수들
* 데이터베이스 결과값

위의 같은 값들은 외부로 입력 받을 수 있는 데이터의 유형들 입니다. 이러한 외부입력과 연관되는 데이터들을 항상 유효성 검사를 같이 하여 사용을 하는 것을 권장합니다. 유효성 검사 없이 데이터를 바로 사용을 할 경우에는, 악의적인 사용자로부터 입력 받은 데이터 값들이 프로그램의 문제를 발생하거나 해킹에 노출되게 됩니다.

**01.1.1 필터목록**

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

PHP의 필터들은 다양한 형태의 패턴들을 검사할 수 있습니다. 패턴을 검사 유형은 필터이름으로 구분을 합니다. 내부함수 filter\_list()는 PHP 에서 지원하는 내장필터 목록을 출력합니다.

**|내부함수|**

|  |
| --- |
| array **filter\_list** ( void ) |

내부함수 filter\_list()를 사용할때는 전달되는 입력 매개변수는 없습니다. 내장된 필터의 목록들을 배열형태로 반환을 합니다.

또한 내장된 필터명은 개발자들이 알기 쉽게 정의한 이름입니다. 이름들은 내부적으로 처리하는 고유의 ID값들로 다시 매칭이 되어 있씁니다. 내부함수 filter\_id()는 입력된 필터명에 대해서 고유의 상수 ID 값을 확인할 수 있습니다.

**|내부함수|**

|  |
| --- |
| int **filter\_id** ( string $filtername ) |

**예제파일) filter-01.php**

|  |
| --- |
| <?php  foreach (**filter\_list()** as $id =>$filter) {  echo "필터명 =" . $filter . ', ID=' . **filter\_id($filter)** . '<br>';  }  ?> |

**화면출력)**

필터명 =int, ID=257

필터명 =boolean, ID=258

필터명 =float, ID=259

필터명 =validate\_regexp, ID=272

필터명 =validate\_domain, ID=277

필터명 =validate\_url, ID=273

필터명 =validate\_email, ID=274

필터명 =validate\_ip, ID=275

필터명 =validate\_mac, ID=276

필터명 =string, ID=513

필터명 =stripped, ID=513

필터명 =encoded, ID=514

필터명 =special\_chars, ID=515

필터명 =full\_special\_chars, ID=522

필터명 =unsafe\_raw, ID=516

필터명 =email, ID=517

필터명 =url, ID=518

필터명 =number\_int, ID=519

필터명 =number\_float, ID=520

필터명 =magic\_quotes, ID=521

필터명 =callback, ID=1024

위의 실습코드는 내장된 필터의 목록과 필터명에 대한 고유한 ID값을 확인하는 예제 입니다. 이처럼 다양한 형태의 데이터 유효성을 검사할 수 있는 필터들을 제공합니다.

**01.1.2 유효성검사**

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

내부함수 filter\_var()는 입력된 데이터를 지정한 필터형태의 패턴으로 데이터 유효성을 검사합니다.

**|내부함수|**

|  |
| --- |
| mixed filter\_var ( mixed $variable [, int $filter = FILTER\_DEFAULT [, mixed $options ]] ) |

함수의 매개변수 입력 값으로 두개의 값을 전달받습니다. 첫번째는 데이터 값은 검사유형 입니다. filter\_var() 함수는 지정된 필터로 단일 변수만 필터링합니다.

**01.2 필터상수**

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

PHP는 filter\_var() 함수를 통하여 필터링을 하기 위한 패턴유형을 상수명으로 제공합니다. 알기 쉬운 패턴코드들을 상수이름으로 정의하여 제공을 하는 것인 코드를 작성하는 측면에서 매우 유용합니다.

**01.2.1 정수 확인 : FILTER\_VALIDATE\_INT**

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

내부 필터함수 filter\_var()는 입력되는 데이터 값이 정수형 타입인지를 확인할 수도 있습니다. 함수를 호출할 때 두번째 인자값으로 필터유형 **FILTER\_VALIDATE\_INT**를 사용하면 됩니다. 값의 정수 여부 판단은 논리값으로 반환을 합니다.

**예제파일) filter-02.php**

|  |
| --- |
| <?php  $int = 100;    if (!filter\_var($int, **FILTER\_VALIDATE\_INT**) === false) {  echo("정수가 유효합니다");  } else {  echo("정수가 아닙니다");  }  ?> |

**화면출력)**

정수가 유효합니다

위의 실험은 입력 변수 $int의 데이터 값이 정수값 인지를 필터를 통하여 확인을 합니다. 만일 정수의 값이면 ture 반환값을 출력합니다.

수학에서 정수는 양수,0,음수 3가지로 구분을 합니다. ilter\_var()는 양수와 음수만 정수값을 판별할 수 있습니다. 즉, 0의 값은 정수로 인식을 하지 못합니다. 따라서 이와 관련된 부가적인 코드를 같이 작성을 해야 합니다. 정수값 필터 사용시 === 0 조건과 || 연결을 결합하여 사용하는 것을 권장합니다.

**예제파일) filter-03.php**

|  |
| --- |
| <?php  $int = 0;    if (filter\_var($int, FILTER\_VALIDATE\_INT) === 0 ||  !filter\_var($int, FILTER\_VALIDATE\_INT) === false) {  echo("Integer is valid");  } else {  echo("Integer is not valid");  }  ?> |

**화면출력)**

Integer is valid

위 실험은 0의 정수값을 같이 판별을 하는 예제코드 입니다.

**5.2.2 정수 확인 및 범위**

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

컴퓨터에서 정수값을 사용할 때 표현을 할 수 있는 최대치의 범위가 있습니다. 또한 프로그램에서 입력의 임계치등의 범위가 필요할 경우가 있습니다. 이러한 경우에는 FILTER\_VALIDATE\_INT 값과 같이 세번째 매개변수를 통하여 범위값을 같이 전달할 수 있습니다.

배열 값을 응용하여 정수값 판별과 값의 범위 여부 내에 있는지 확인 할 수 있습니다.

**예제파일) filter-04.php**

|  |
| --- |
| <?php  $int = 122;  $min = 1;  $max = 200;  $range = array("options" => array("min\_range"=>$min, "max\_range"=>$max);    if (filter\_var($int, FILTER\_VALIDATE\_INT, $range)) === false) {  echo("변수 값이 유효한 범위 내에 있지 않습니다.!");  } else {  echo("변수 값이 유효한 범위 내에 있습니다.!");  }  ?> |

**화면출력)**

변수 값이 유효한 범위 내에 있습니다.!

위의 실험은 필터함수 filter\_var()를 사용하여 입력되는 데이터가 정수유형 인지와 1에서 200 사이의 값을 가지는지를 확인합니다.

그외 참고하여 같이 사용할 수 있는 필터상수는 다음과 같습니다:

* **FILTER\_VALIDATE\_BOOLEAN :** 논리값을 검사합니다.
* **FILTER\_VALIDATE\_FLOAT :** float의 유효성을 검사합니다.
* **FILTER\_SANITIZE\_NUMBER\_FLOAT :** 숫자, + - 및 .e를 제외한 모든 문자를 제거합니다.
* **FILTER\_SANITIZE\_NUMBER\_INT :** 숫자와 + -를 제외한 모든 문자를 제거합니다.

**01.2.3 IP주소 유효성 : FILTER\_VALIDATE\_IP**

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

필터함수 filter\_var()는 인터넷 IP 주소의 유형을 검사할 수 있습니다. 필터 상수명으로 FILTER\_VALIDATE\_IP 를 사용하게 되면 입력한 데이터가 IP 주소인지를 검사할 수 있습니다.

**예제파일) filter-05.php**

|  |
| --- |
| <?php  $ip = "127.0.0.1";    if (!**filter\_var**($ip, FILTER\_VALIDATE\_IP) === false) {  echo("$ip 는 유효한 IP주소 입니다.");  } else {  echo("$ip 는 유효하지 IP주소 입니다.");  }  ?> |

**화면출력)**

127.0.0.1 는 유효한 IP주소 입니다.

**5.2.4 IPv6 주소 유효성**

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

IP에 대해서 IPv6 주소 타입 및 유효성을 같이 검사를 할 수 있습니다. IPv6 주소를 검사할때는 FILTER\_VALIDATE\_IP와 FILTER\_FLAG\_IPV6 두개의 값을 같이 이용을 합니다.

**예제파일) filter-06.php**

|  |
| --- |
| <?php  $ip = "2001:0db8:85a3:08d3:1319:8a2e:0370:7334";    if (!filter\_var($ip, **FILTER\_VALIDATE\_IP, FILTER\_FLAG\_IPV6**) === false) {  echo("$ip 는 유효한 IPv6 address");  } else {  echo("$ip 는 유효하지 않는 IPv6 address");  }  ?> |

**화면출력)**

2001:0db8:85a3:08d3:1319:8a2e:0370:7334 는 유효한 IPv6 address

**01.2.5 쿼리 스트링과 URL 유효성 검사 : FILTER\_VALIDATE\_URL**

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

검사유형 FILTER\_VALIDATE\_URL 와 FILTER\_FLAG\_QUERY\_REQUIRED 두개의 값을 통하여 URL 유효성과 이후 입력되는 GET 쿼리 스트링 데이터 값의 유효성 까지 함께 확인을 할 수 있습니다.

**예제파일) filter-07.php**

|  |
| --- |
| <?php  $url = "http://www.hojin.io";    if (!filter\_var($url, FILTER\_VALIDATE\_URL, FILTER\_FLAG\_QUERY\_REQUIRED) === false) {  echo("$url 유효한 URL");  } else {  echo("$url 유효하지 않은 URL");  }  ?> |

**화면출력)**

http://www.hojin.io 유효하지 않은 URL

다음 실험은 filter\_var() 함수를 사용하여 문자열을 삭제합니다. 다음과 같이 문자열에서 모든 HTML 태그와 ASCII 값이 127보다 큰 모든 문자를 제거합니다.

**예제파일) filter-08.php**

|  |
| --- |
| <?php  $str = "<h1>Hello World!</h1>";    $aaa = filter\_var($str, FILTER\_SANITIZE\_STRING, FILTER\_FLAG\_STRIP\_HIGH);  echo $aaa;  ?> |

**화면출력)**

Hello World!

**01.2.6 이메일 주소 검사**

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

내부함수 filter\_var()를 통하여 이메일의 유효성을 검사할 수 있습니다. 이메일의 유효성을 검사할때는 필터상수 FILTER\_VALIDATE\_EMAIL를 사용합니다.

**예제파일) filter-09.php**

|  |
| --- |
| <?php  $email = "infohojin@naver.com";    // 이메일에 포함된 부정한 글자들은 제거를 합니다.  $email = filter\_var($email, FILTER\_SANITIZE\_EMAIL);    // e-mail 유효성  if (!filter\_var($email, FILTER\_VALIDATE\_EMAIL) === false) {  echo("$email is a valid email address");  } else {  echo("$email is not a valid email address");  }  ?> |

**화면출력)**

james.lee@example.com is a valid email address

그외 참고하여 같이 사용할 수 있는 필터상수는 다음과 같습니다:

* **FILTER\_SANITIZE\_EMAIL :** 부정한 글자를 제거합니다.
* **FILTER\_VALIDATE\_EMAIL :** 전자 메일 주소 유효성 검사

**01.2.7 문자열 체크 : FILTER\_SANITIZE\_STRING**

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

입력한 값을 재출력하는 기능에서 잘못 입력되는 html 코드들은 오동작을 발생하게 됩니다. 내부함수 filter\_var()를 통하여 입력된 문자열에서 HTML 코드를 제거할 수 있습니다.

**예제파일) filter-10.php**

|  |
| --- |
| <?php  $str = "<h1>Hello World!</h1>";  $newstr = filter\_var($str, FILTER\_SANITIZE\_STRING);  echo $newstr;  ?> |

**화면출력)**

Hello World!

필터상수 FILTER\_SANITIZE\_STRIPPED은 FILTER\_SANITIZE\_STRING의 별칭 입니다.

**01.2.8 URL 유효성 검사**

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

필터상수 FILTER\_SANITIZE\_URL 는 입력값의 URL 유효성 검사를 확인할 수 있습니다.

**예제파일) filter-11.php**

|  |
| --- |
| <?php  $url = "https://www.jinyphp.com";    // Remove all illegal characters from a url  $url = filter\_var($url, FILTER\_SANITIZE\_URL);    // Validate url  if (!filter\_var($url, FILTER\_VALIDATE\_URL) === false) {  echo("$url is a valid URL");  } else {  echo("$url is not a valid URL");  }  ?> |

**화면출력)**

https://www. hojin.io is a valid URL

**01.2.9 정규표현**

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

필터상수 FILTER\_VALIDATE\_REGEXP는 입력된 데이터의 정규 표현식 유효성을 검사합니다.

**01.2.10 특수문자**

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

다음과 같은 필터상수를 통하여 입력되는 데이터의 특수문자들을 처리할 수 있습니다:

* **FILTER\_SANITIZE\_ENCODED :** 특수 문자를 제거 / 인코딩합니다.
* **FILTER\_SANITIZE\_SPECIAL\_CHARS :** 특수 문자를 제거합니다.
* **FILTER\_UNSAFE\_RAW :** 별도의 작업없이, 선택적으로 특수 문자만 제거 / 인코딩합니다.
* **FILTER\_SANITIZE\_FULL\_SPECIAL\_CHARS**

**HTML 특수문자코드표**

|  |  |  |  |
| --- | --- | --- | --- |
| **표현문자** | **숫자표현** | **문자표현** | **설명** |
| **-** | **&#00;-&#08;** | **-** | **사용하지 않음** |
| **space** | **&#09;** | **-** | **수평탭** |
| **space** | **&#10;** | **-** | **줄 삽입** |
| **-** | **&#11;-&#31;** | **-** | **사용하지 않음** |
| **space** | **&#32;** | **-** | **여백** |
| **!** | **&#33;** | **-** | **느낌표** |
| **"** | **&#34;** | **&quot;** | **따옴표** |
| **#** | **&#35;** | **-** | **숫자기호** |
| **$** | **&#36;** | **-** | **달러** |
| **%** | **&#37;** | **-** | **백분율 기호** |
| **&** | **&#38;** | **&amp;** | **Ampersand** |
| **'** | **&#39;** | **-** | **작은 따옴표** |
| **(** | **&#40;** | **-** | **왼쪽 괄호** |
| **)** | **&#41;** | **-** | **오른쪽 괄호** |
| **\*** | **&#42;** | **-** | **아스트릭** |
| **+** | **&#43;** | **-** | **더하기 기호** |
| **,** | **&#44;** | **-** | **쉼표** |
| **-** | **&#45;** | **-** | **Hyphen** |
| **.** | **&#46;** | **-** | **마침표** |
| **/** | **&#47;** | **-** | **Solidus (slash)** |
| **0 - 9** | **&#48;-&#57;** | **-** | **0부터 9까지** |
| **:** | **&#58;** | **-** | **콜론** |
| **;** | **&#59;** | **-** | **세미콜론** |
| **<** | **&#60;** | **&lt;** | **보다 작은** |
| **=** | **&#61;** | **-** | **등호** |
| **>** | **&#62;** | **&gt;** | **보다 큰** |
| **?** | **&#63;** | **-** | **물음표** |
| **@** | **&#64;** | **-** | **Commercial at** |
| **A - Z** | **&#65;-&#90;** | **-** | **A부터 Z까지** |
| **[** | **&#91;** | **-** | **왼쪽 대괄호** |
| **\** | **&#92;** | **-** | **역슬래쉬** |
| **]** | **&#93;** | **-** | **오른쪽 대괄호** |
| **^** | **&#94;** | **-** | **탈자부호** |
| **\_** | **&#95;** | **-** | **수평선** |
| **`** | **&#96;** | **-** | **Acute accent** |
| **a - z** | **&#97;-&#122;** | **-** | **a부터 z까지** |
| **{** | **&#123;** | **-** | **왼쪽 중괄호** |
| **|** | **&#124;** | **-** | **수직선** |
| **}** | **&#125;** | **-** | **오른쪽 중괄호** |
| **~** | **&#126;** | **-** | **꼬리표** |
| **-** | **&#127;-&#159;** | **-** | **사용하지 않음** |
|  | **&#160;** | **&nbsp;** | **Non-breaking space** |
| **¡** | **&#161;** | **&iexcl;** | **거꾸로된 느낌표** |
| **￠** | **&#162;** | **&cent;** | **센트 기호** |
| **￡** | **&#163;** | **&pound;** | **파운드** |
| **¤** | **&#164;** | **&curren;** | **현재 환율** |
| **￥** | **&#165;** | **&yen;** | **엔** |
| **|** | **&#166;** | **&brvbar;** | **끊어진 수직선** |
| **§** | **&#167;** | **&sect;** | **섹션 기호** |
| **¨** | **&#168;** | **&uml;** | **움라우트** |
| **ⓒ** | **&#169;** | **&copy;** | **저작권** |
| **ª** | **&#170;** | **&ordf;** | **Feminine ordinal** |
| **≪** | **&#171;** | **&laquo;** | **왼쪽 꺾인 괄호** |
| **￢** | **&#172;** | **&not;** | **부정** |
| **­** | **&#173;** | **&shy;** | **Soft hyphen** |
| **?** | **&#174;** | **&reg;** | **등록상표** |
| **&hibar;** | **&#175;** | **&macr;** | **Macron accent** |
| **°** | **&#176;** | **&deg;** | **Degree sign** |
| **±** | **&#177;** | **&plusmn;** | **Plus or minus** |
| **²** | **&#178;** | **&sup2;** | **Superscript two** |
| **³** | **&#179;** | **&sup3;** | **Superscript three** |
| **´** | **&#180;** | **&acute;** | **Acute accent** |
| **μ** | **&#181;** | **&micro;** | **Micro sign (Mu)** |
| **¶** | **&#182;** | **&para;** | **문단기호** |
| **·** | **&#183;** | **&middot;** | **Middle dot** |
| **¸** | **&#184;** | **&cedil;** | **Cedilla** |
| **¹** | **&#185;** | **&sup1;** | **Superscript one** |
| **º** | **&#186;** | **&ordm;** | **Masculine ordinal** |
| **≫** | **&#187;** | **&raquo;** | **오른쪽 꺾인 괄호** |
| **¼** | **&#188;** | **&frac14;** | **4분의 1** |
| **½** | **&#189;** | **&frac12;** | **2분의 1** |
| **¾** | **&#190;** | **&frac34;** | **4분의 3** |
| **¿** | **&#191;** | **&iquest;** | **거꾸로된 물음표** |
| **A** | **&#192;** | **&Agrave;** | **Capital A, grave accent** |
| **A** | **&#193;** | **&Aacute;** | **Capital A, acute accent** |
| **A** | **&#194;** | **&Acirc;** | **Capital A, circumflex accent** |
| **A** | **&#195;** | **&Atilde;** | **Capital A, tilde** |
| **A** | **&#196;** | **&Auml;** | **Capital A, dieresis or umlaut mark** |
| **A** | **&#197;** | **&Aring;** | **Capital A, ring (Angstrom)** |
| **Æ** | **&#198;** | **&AElig;** | **Capital AE diphthong (ligature)** |
| **C** | **&#199;** | **&Ccedil;** | **Capital C, cedilla** |
| **E** | **&#200;** | **&Egrave;** | **Capital E, grave accent** |
| **E** | **&#201;** | **&Eacute;** | **Capital E, acute accent** |
| **E** | **&#202;** | **&Ecirc;** | **Capital E, circumflex accent** |
| **E** | **&#203;** | **&Euml;** | **Capital E, dieresis or umlaut mark** |
| **I** | **&#204;** | **&Igrave;** | **Capital I, grave accent** |
| **I** | **&#205;** | **&Iacute;** | **Capital I, acute accent** |
| **I** | **&#206;** | **&Icirc;** | **Capital I, circumflex accent** |
| **I** | **&#207;** | **&Iuml;** | **Capital I, dieresis or umlaut mark** |
| **Ð** | **&#208;** | **&ETH;** | **Capital Eth, Icelandic** |
| **N** | **&#209;** | **&Ntilde;** | **Capital N, tilde** |
| **O** | **&#210;** | **&Ograve;** | **Capital O, grave accent** |
| **O** | **&#211;** | **&Oacute;** | **Capital O, acute accent** |
| **O** | **&#212;** | **&Ocirc;** | **Capital O, circumflex accent** |
| **O** | **&#213;** | **&Otilde;** | **Capital O, tilde** |
| **O** | **&#214;** | **&Ouml;** | **Capital O, dieresis or umlaut mark** |
| **×** | **&#215;** | **&times;** | **Multiply sign** |
| **Ø** | **&#216;** | **&Oslash;** | **width="130"Capital O, slash** |
| **U** | **&#217;** | **&Ugrave;** | **Capital U, grave accent** |
| **U** | **&#218;** | **&Uacute;** | **Capital U, acute accent** |
| **U** | **&#219;** | **&Ucirc;** | **Capital U, circumflex accent** |
| **U** | **&#220;** | **&Uuml;** | **Capital U, dieresis or umlaut mark** |
| **Y** | **&#221;** | **&Yacute;** | **Capital Y, acute accent** |
| **Þ** | **&#222;** | **&THORN;** | **Capital Thorn, Icelandic** |
| **ß** | **&#223;** | **&szlig;** | **Small sharp s, German (sz ligature)** |
| **a** | **&#224;** | **&agrave;** | **Small a, grave accent** |
| **a** | **&#225;** | **&aacute;** | **Small a, acute accent** |
| **a** | **&#226;** | **&acirc;** | **Small a, circumflex accent** |
| **a** | **&#227;** | **&atilde;** | **Small a, tilde** |
| **a** | **&#228;** | **&auml;** | **Small a, dieresis or umlaut mark** |
| **a** | **&#229;** | **&aring;** | **Small a, ring** |
| **æ** | **&#230;** | **&aelig;** | **Small ae diphthong (ligature)** |
| **c** | **&#231;** | **&ccedil;** | **Small c, cedilla** |
| **e** | **&#232;** | **&egrave;** | **Small e, grave accent** |
| **e** | **&#233;** | **&eacute;** | **Small e, acute accent** |
| **e** | **&#234;** | **&ecirc;** | **Small e, circumflex accent** |
| **e** | **&#235;** | **&euml;** | **Small e, dieresis or umlaut mark** |
| **i** | **&#236;** | **&igrave;** | **Small i, grave accent** |
| **i** | **&#237;** | **&iacute;** | **Small i, acute accent** |
| **i** | **&#238;** | **&icirc;** | **Small i, circumflex accent** |
| **i** | **&#239;** | **&iuml;** | **Small i, dieresis or umlaut mark** |
| **ð** | **&#240;** | **&eth;** | **Small eth, Icelandic** |
| **n** | **&#241;** | **&ntilde;** | **Small n, tilde** |
| **o** | **&#242;** | **&ograve;** | **Small o, grave accent** |
| **o** | **&#243;** | **&oacute;** | **Small o, acute accent** |
| **o** | **&#244;** | **&ocirc;** | **Small o, circumflex accent** |
| **o** | **&#245;** | **&otilde;** | **Small o, tilde** |
| **o** | **&#246;** | **&ouml;** | **Small o, dieresis or umlaut mark** |
| **÷** | **&#247;** | **&divide;** | **Division sign** |
| **ø** | **&#248;** | **&oslash;** | **Small o, slash** |
| **u** | **&#249;** | **&ugrave;** | **Small u, grave accent** |
| **u** | **&#250;** | **&uacute;** | **Small u, acute accent** |
| **u** | **&#251;** | **&ucirc;** | **Small u, circumflex accent** |
| **u** | **&#252;** | **&uuml;** | **Small u, dieresis or umlaut mark** |
| **y** | **&#253;** | **&yacute;** | **Small y, acute accent** |
| **þ** | **&#254;** | **&thorn;** | **Small thorn, Icelandic** |
| **y** | **&#255;** | **&yuml;** | **Small y, dieresis or umlaut mark** |

**01.2.11 함수적용**

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

다음과 같은 필터 상수들은 함수를 별도의 함수를 적용하거나 호출을 할 수 있습니다.

* **FILTER\_SANITIZE\_MAGIC\_QUOTES:** addslashes() 함수를 적용합니다.
* **FILTER\_CALLBACK :** 데이터 필터링을 위한 사용자 정의 함수를 호출합니다.

**01.3 필터함수**

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

PHP는 필터기능을 좀더 다양하게 처리를 하기 위해서 추가로 몇 개의 내부함수들을 제공합니다. 추가 함수들을 이용하면 복잡한 필터처리들을 보다 쉽게 구현을 할 수 있습니다.

**|내부함수|**

|  |
| --- |
| bool **filter\_has\_var** ( int $type , string $variable\_name ) |

내부함수 filter\_has\_var()는지정된 타입유형의 변수가 있는지 확인합니다. 타입의 종류는 다음과 같습니다 :

* INPUT\_GET
* INPUT\_POST
* INPUT\_COOKIE
* INPUT\_SERVER, 또는 INPUT\_ENV.

**예제파일) filter-12.php**

|  |
| --- |
| <?php  echo "INPUT\_GET = ";  echo filter\_has\_var(INPUT\_GET, 'aaa') ? 'Yes' : 'No';  ?> |

**화면출력)**

INPUT\_GET = Yes

위의 실습은 타입유형의 변수 존재 유무를 확인합니다. filter-12.php**?aaa=test** 형태로 GET 데이터를 추가하여 스크립트를 호출합니다. GET 방식으로 데이터를 전달과 변수명이 aaa 이기 때문 filter\_has\_var() 함수는 결과를 ‘**참’** 으로 출력합니다.

**|내부함수|**

|  |
| --- |
| mixed **filter\_input** ( int $type , string $variable\_name [, int $filter = FILTER\_DEFAULT [, mixed $options ]] ) |

내부함수 filter\_input()는 양식 입력등 처럼 외부 변수를 읽어와서 선택적으로 필터링합니다.

**예제파일) filter-13.php**

|  |
| --- |
| <?php  $search\_html = filter\_input(INPUT\_GET, 'search', FILTER\_SANITIZE\_SPECIAL\_CHARS);  $search\_url = filter\_input(INPUT\_GET, 'search', FILTER\_SANITIZE\_ENCODED);  echo "입력한 검색키워드는 $search\_html 입니다.<br>";  echo "<a href='?search=$search\_url'>재검색</a>";  ?> |

위의 실험은 filter-13.php**?search=jiny** 형태로 GET 데이터를 추가하여 스크립트를 호출합니다.

**|내부함수|**

|  |
| --- |
| mixed **filter\_input\_array** ( int $type [, mixed $definition [, bool $add\_empty = true ]] ) |

내부함수 filter\_input\_array()는 필터를 처리하기 위한 배열 정보를 이용하여 다수의 외부변수를 일괄적으로 필터링합니다. 이 함수는 filter\_input ()을 반복적으로 호출하지 않고도 많은 값을 검색 할 때 유용합니다.

**예제파일) filter-14.php**

|  |
| --- |
| <?php  error\_reporting(E\_ALL | E\_STRICT);  $args = array(  'name' => FILTER\_SANITIZE\_ENCODED,  'age' => FILTER\_VALIDATE\_INT,  'country' => FILTER\_SANITIZE\_ENCODED  );  $myinputs = filter\_input\_array(INPUT\_GET, $args);  var\_dump($myinputs);  ?> |

**화면출력)**

array(3) { ["name"]=> string(4) "jiny" ["age"]=> int(18) ["country"]=> string(5) "korea" }

위의 실험은 filter-14.php**?name=jiny&age=18&country=korea** 형태로 GET 데이터를 추가하여 스크립트를 호출합니다.

**|내부함수|**

|  |
| --- |
| mixed **filter\_var\_array** ( array $data [, mixed $definition [, bool $add\_empty = true ]] ) |

내부변수 filter\_var\_array()는 여러개의 변수를 필터링합니다. 이 함수는 filter\_var ()를 반복적으로 호출하지 않고도 많은 값을 검색하는 데 유용합니다. 아래 예제는 공식사이트에 있는 예제입니다.

**예제파일) filter-15.php**

|  |
| --- |
| <?php  error\_reporting(E\_ALL | E\_STRICT);  $data = array(  'product\_id' => 'libgd<script>',  'component' => '10',  'versions' => '2.0.33',  'testscalar' => array('2', '23', '10', '12'),  'testarray' => '2',  );  $args = array(  'product\_id' => FILTER\_SANITIZE\_ENCODED,  'component' => array('filter' => FILTER\_VALIDATE\_INT,  'flags' => FILTER\_FORCE\_ARRAY,  'options' => array('min\_range' => 1, 'max\_range' => 10)  ),  'versions' => FILTER\_SANITIZE\_ENCODED,  'doesnotexist' => FILTER\_VALIDATE\_INT,  'testscalar' => array(  'filter' => FILTER\_VALIDATE\_INT,  'flags' => FILTER\_REQUIRE\_SCALAR,  ),  'testarray' => array(  'filter' => FILTER\_VALIDATE\_INT,  'flags' => FILTER\_FORCE\_ARRAY,  )  );  $myinputs = filter\_var\_array($data, $args);  var\_dump($myinputs);  echo "\n";  ?> |

**화면출력)**

array(6) { ["product\_id"]=> string(17) "libgd%3Cscript%3E" ["component"]=> array(1) { [0]=> int(10) } ["versions"]=> string(6) "2.0.33" ["doesnotexist"]=> NULL ["testscalar"]=> bool(false) ["testarray"]=> array(1) { [0]=> int(2) } }