PHP 异常处理 PHP MySQL 简介

PHP 过滤器

PHP 过滤器用于验证和过滤来自非安全来源的数据，比如用户的输入。

什么是 PHP 过滤器？

PHP 过滤器用于验证和过滤来自非安全来源的数据。

测试、验证和过滤用户输入或自定义数据是任何 Web 应用程序的重要组成部分。

PHP 的过滤器扩展的设计目的是使数据过滤更轻松快捷。

为什么使用过滤器？

几乎所有的 Web 应用程序都依赖外部的输入。这些数据通常来自用户或其他应用程序（比如 web 服务）。通过使用过滤器，您能够确保应用程序获得正确的输入类型。

您应该始终对外部数据进行过滤！

输入过滤是最重要的应用程序安全课题之一。

什么是外部数据？

- 来自表单的输入数据
- Cookies
- Web services data
- 服务器变量
- 数据库查询结果

函数和过滤器

如需过滤变量，请使用下面的过滤器函数之一：

- filter_var() - 通过一个指定的过滤器来过滤单一的变量
- filter_var_array() - 通过相同的或不同的过滤器来过滤多个变量
- filter_input - 获取一个输入变量，并对它进行过滤
- filter_input_array - 获取多个输入变量，并通过相同的或不同的过滤器对它们进行过滤

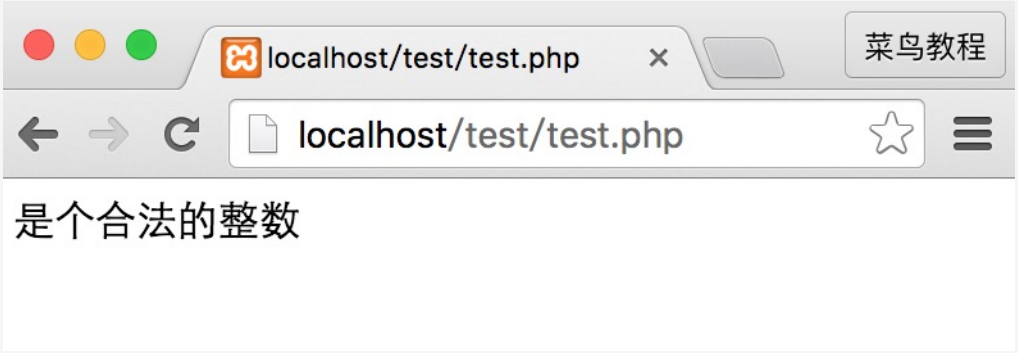
在下面的实例中，我们用 filter_var() 函数验证了一个整数：

实例

```
<?php
$int = 123;
if(!filter_var($int, FILTER_VALIDATE_INT))
{
```

```
echo("不是一个合法的整数");
}
else
{
echo("是个合法的整数");
}
?>
```

上面的代码使用了 "FILTER_VALIDATE_INT" 过滤器来过滤变量。由于这个整数是合法的，因此上面的代码将输出：



如果我们尝试使用一个非整数的变量（比如 "123abc"），则将输出："Integer is not valid"。

如需查看完整的函数和过滤器列表，请访问我们的 PHP Filter 参考手册。

Validating 和 Sanitizing

有两种过滤器：

Validating 过滤器：

- 用于验证用户输入
- 严格的格式规则（比如 URL 或 E-Mail 验证）
- 如果成功则返回预期的类型，如果失败则返回 FALSE

Sanitizing 过滤器：

- 用于允许或禁止字符串中指定的字符
- 无数据格式规则
- 始终返回字符串

选项和标志

选项和标志用于向指定的过滤器添加额外的过滤选项。

不同的过滤器有不同的选项和标志。

在下面的实例中，我们用 filter_var() 和 "min_range" 以及 "max_range" 选项验证了一个整数：

实例

```
<?php
$var=300;
```

```php
$int_options = array(
"options"=>array
(
"min_range"=>0,
"max_range"=>256
)
);
if(!filter_var($var, FILTER_VALIDATE_INT, $int_options))
{
echo("不是一个合法的整数");
}
else
{
echo("是个合法的整数");
}
?>
```

就像上面的代码一样，选项必须放入一个名为 "options" 的相关数组中。如果使用标志，则不需在数组内。

由于整数是 "300"，它不在指定的范围内，以上代码的输出将是：

```
不是一个合法的整数
```

如需查看完整的函数和过滤器列表，请访问我们的 PHP Filter 参考手册。您可以看到每个过滤器的可用选项和标志。

验证输入

让我们试着验证来自表单的输入。

我们需要做的第一件事情是确认是否存在我们正在查找的输入数据。

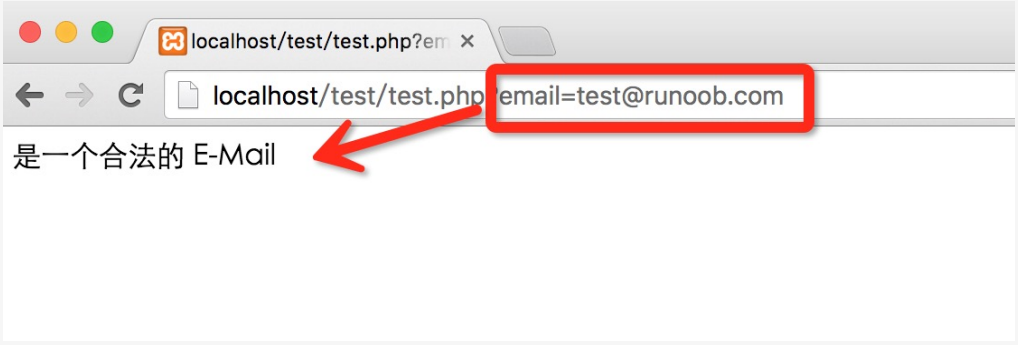
然后我们用 filter_input() 函数过滤输入的数据。

在下面的实例中，输入变量 "email" 被传到 PHP 页面：

实例

```php
<?php
if(!filter_has_var(INPUT_GET, "email"))
{
echo("没有 email 参数");
}
else
{
if (!filter_input(INPUT_GET, "email", FILTER_VALIDATE_EMAIL))
{
echo "不是一个合法的 E-Mail";
}
else
{
echo "是一个合法的 E-Mail";
}
}
?>
```

以上实例测试结果如下：



实例解释

上面的实例有一个通过 "GET" 方法传送的输入变量 (email)：

1. 检测是否存在 "GET" 类型的 "email" 输入变量
2. 如果存在输入变量，检测它是否是有效的 e-mail 地址

净化输入

让我们试着清理一下从表单传来的 URL。

首先，我们要确认是否存在我们正在查找的输入数据。

然后，我们用 filter_input() 函数来净化输入数据。

在下面的实例中，输入变量 "url" 被传到 PHP 页面：

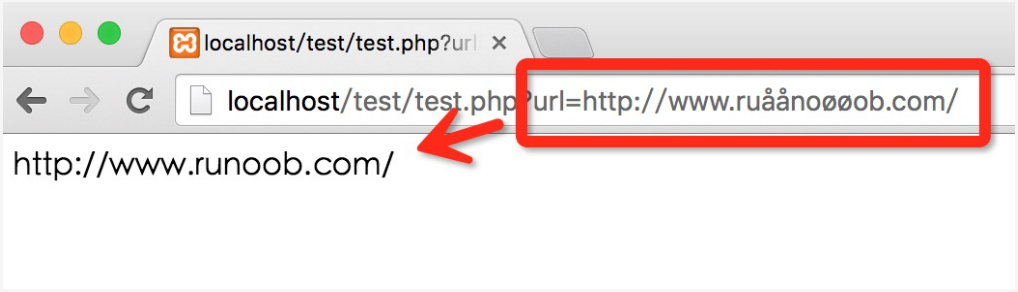
```
<?php
if(!filter_has_var(INPUT_GET, "url"))
{
    echo("没有 url 参数");
}
else
{
    $url = filter_input(INPUT_GET,
    "url", FILTER_SANITIZE_URL);
    echo $url;
}
?>
```

实例解释

上面的实例有一个通过 "GET" 方法传送的输入变量 (url)：

1. 检测是否存在 "GET" 类型的 "url" 输入变量
2. 如果存在此输入变量，对其进行净化（删除非法字符），并将其存储在 $url 变量中

假如输入变量是一个类似这样的字符串："http://www.ruåånoøøob.com/"，则净化后的 $url 变量如下所示：



过滤多个输入

表单通常由多个输入字段组成。为了避免对 filter_var 或 filter_input 函数重复调用，我们可以使用 filter_var_array 或 the filter_input_array 函数。

在本例中，我们使用 filter_input_array() 函数来过滤三个 GET 变量。接收到的 GET 变量是一个名字、一个年龄以及一个 e-mail 地址：

实例

```php
<?php
$filters = array
(
"name" => array
(
"filter"=>FILTER_SANITIZE_STRING
),
"age" => array
(
"filter"=>FILTER_VALIDATE_INT,
"options"=>array
(
"min_range"=>1,
"max_range"=>120
)
),
"email"=> FILTER_VALIDATE_EMAIL
);
$result = filter_input_array(INPUT_GET, $filters);
if (!$result["age"])
{
echo("年龄必须在 1 到 120 之间。<br>");
}
elseif(!$result["email"])
{
echo("E-Mail 不合法<br>");
}
else
{
echo("输入正确");
}
?>
```

实例解释

上面的实例有三个通过 "GET" 方法传送的输入变量 (name、age 和 email)：

1. 设置一个数组，其中包含了输入变量的名称和用于指定的输入变量的过滤器
2. 调用 filter_input_array() 函数，参数包括 GET 输入变量及刚才设置的数组
3. 检测 $result 变量中的 "age" 和 "email" 变量是否有非法的输入。（如果存在非法输入，在使用 filter_input_array() 函数之后，输入变量为 FALSE。）

filter_input_array() 函数的第二个参数可以是数组或单一过滤器的 ID。

如果该参数是单一过滤器的 ID，那么这个指定的过滤器会过滤输入数组中所有的值。

如果该参数是一个数组，那么此数组必须遵循下面的规则：

- 必须是一个关联数组，其中包含的输入变量是数组的键（比如 "age" 输入变量）
- 此数组的值必须是过滤器的 ID ，或者是规定了过滤器、标志和选项的数组

使用 Filter Callback

通过使用 FILTER_CALLBACK 过滤器，可以调用自定义的函数，把它作为一个过滤器来使用。这样，我们就拥有了数据过滤的完全控制权。

您可以创建自己的自定义函数，也可以使用已存在的 PHP 函数。

将您准备用到的过滤器的函数，按指定选项的规定方法进行规定。在关联数组中，带有名称 "options"。

在下面的实例中，我们使用了一个自定义的函数把所有 "_" 转换为 "."：

实例

```
<?php
function convertSpace($string)
{
return str_replace("_", ".", $string);
}
$string = "www_runoob_com!";
echo filter_var($string, FILTER_CALLBACK,
array("options"=>"convertSpace"));
?>
```

上面代码的结果如下所示：

```
www.runoob.com!
```

实例解释

上面的实例把所有 "_" 转换成 "." ：

1. 创建一个把 "_" 替换为 "." 的函数

2. 调用 filter_var() 函数，它的参数是 FILTER_CALLBACK 过滤器以及包含我们的函数的数组

PHP 异常处理 PHP MySQL 简介