 $\_GET 和 $\_POST 变量用于检索表单中的信息，比如用户输入。

PHP 能把来自 HTML 页面中的表单元素**自动**变成可供 PHP 脚本使用。

<form action="welcome.php" method="post">

Name: <input type="text" name="name"><br>

E-mail: <input type="text" name="email"><br>

<input type="submit">

</form>

<?php echo $\_POST["fname"]; ?>

$\_GET 是通过 URL 参数传递到当前脚本的变量数组。

$\_POST 是通过 HTTP POST 传递到当前脚本的变量数组。

GET 可用于发送非敏感的数据。

**注释：**绝不能使用 GET 来发送密码或其他敏感信息！

## 提示：开发者偏爱 POST 来发送表单数据。

## 表单验证

应该在任何可能的时候对用户输入进行验证（通过客户端脚本）。浏览器验证速度更快，并且可以减轻服务器的负载。

如果用户输入需要插入数据库，您应该考虑使用服务器验证。在服务器验证表单的一种好的方式是，把表单传给它自己，而不是跳转到不同的页面。这样用户就可以在同一张表单页面得到错误信息。用户也就更容易发现错误了。

action="<?php echo htmlspecialchars($\_SERVER["PHP\_SELF"]);?>"

当提交此表单时，通过 method="post" 发送表单数据。

### **什么是 $\_SERVER["PHP\_SELF"] 变量？**

$\_SERVER["PHP\_SELF"] 是一种超全局变量，它返回当前执行脚本的文件名。

因此，$\_SERVER["PHP\_SELF"] 将表单数据发送到页面本身，而不是跳转到另一张页面。这样，用户就能够在表单页面获得错误提示信息。

### **什么是 htmlspecialchars() 函数？**

htmlspecialchars() 函数把特殊字符转换为 HTML 实体。这意味着 < 和 > 之类的 HTML 字符会被替换为 &lt; 和 &gt; 。这样可防止攻击者通过在表单中注入 HTML 或 JavaScript 代码（跨站点脚本攻击）对代码进行利用。

在用户提交该表单时，我们还要做两件事：

1. （通过 PHP trim() 函数）去除用户输入数据中不必要的字符（多余的空格、制表符、换行）
2. （通过 PHP stripslashes() 函数）删除用户输入数据中的反斜杠（\）

<?php// 定义变量并设置为空值

$name = $email = $gender = $comment = $website = "";

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {//检查表单是否使用 $\_SERVER["REQUEST\_METHOD"] 进行提交

$name = test\_input($\_POST["name"]);

$email = test\_input($\_POST["email"]);

$website = test\_input($\_POST["website"]);

$comment = test\_input($\_POST["comment"]);

$gender = test\_input($\_POST["gender"]);

}

function test\_input($data) {

$data = trim($data);//去除用户输入数据中不必要的字符

$data = stripslashes($data);//删除用户输入数据中的反斜杠

$data = htmlspecialchars($data);//把特殊字符转换为 HTML 实体

return $data;

}

?>