

Notice that in line 2 of the table, there is no space between the two variable names, so there is no space in the output. In line 3, a space is echoed between the two variables.

In some echo statements, PHP can't tell the variable name from the other information around it. In cases where this could be confusing, you need to enclose the variable name in curly braces. For example, suppose you use the following statements:

```
$type = "bird";  
echo "Keep the $typecage clean";
```

Rather than the desired output, you get the following message:

```
Notice: Undefined variable: typecage in testvar.php on line 6
```

After notifying you of the problem, the following output is displayed:

```
Keep the clean
```

To make this code work correctly, you need to use the following echo statement:

```
echo "Keep the {$type}cage clean";
```

With this statement, the output is the following:

```
Keep the birdcage clean
```

Using Variable Variables

PHP allows you to use dynamic variable names, called *variable variables*. You can name a variable by using the value stored in another variable. That is, one variable contains the name of another variable. For example, suppose you want to construct a variable named `$city` with the value `Los Angeles`. You can use the following statement:

```
$name_of_the_variable = "city";
```

This statement creates a variable that contains the name that you want to give to a variable. Then you use the following statements:

```
$$name_of_the_variable = "Los Angeles";
```

Note the extra dollar sign (\$) character at the beginning of the variable name. This indicates a variable variable. This statement creates a new variable with the name that is the value in `$name_of_the_variable`, resulting in the following: