This line places the values of each error property into the `error_message` property. If all the updates were successful, the `$error_message` property would contain

```
"TRUE,TRUE,TRUE,TRUE"
```

Notice that each item passed includes a "," for separation except the last item. This is necessary to break apart the results of the string.

Example 3-13. The lab.php file calling a constructor

```
<?php
require_once("dog.php");
$lab = new Dog('Fred','Lab','Yellow','100');
list($name_error, $breed_error, $color_error, $weight_error) = explode(',', $lab);
print $name_error == 'TRUE' ? 'Name update successful<br/>' : 'Name update not
successful<br/>';
print $breed_error == 'TRUE' ? 'Breed update successful<br/>' : 'Breed update not
successful<br/>';
print $color_error == 'TRUE' ? 'Color update successful<br/>' : 'Color update not
successful<br/>';
print $weight_error == 'TRUE' ? 'Weight update successful<br/>' : 'Weight update not
successful<br/>';
// ------------------------------Set Properties-------------------------

...There are no other changes to lab.php below this line.
```

There is a slight change to the creation of the object on the third line of Example 3-13.

```
$lab = new Dog('Fred','Lab','Yellow','100');
```

You are now passing the initial values (`Fred`, `Lab`, `Yellow`,and `100`) into the object via the constructor. Otherwise, you would have to make four calls to set methods (`set_dog_name`, `set_dog_breed`, `set_dog_color`, and `set_dog_weight`) to accomplish the same thing. This allows you to use the `set` methods for updates that are needed after you have initially set up the object (`$lab`).

In order to determine if the updates to the four properties were successful, you must retrieve your values (`TRUE`, `TRUE`, `TRUE`,`TRUE`) from the `$error_message` property in the object. The `__toString` method in the `Dog` class allows you to do this by treating `$lab` as if it were a string. This allows you to use the `explode` method in a similar process as the output to the `get_properties` method.

```
list($name_error, $breed_error, $color_error, $weight_error) = explode(',', $lab);
```

This line of code will break the contents of `$lab` (`TRUE`, `TRUE`, `TRUE`,`TRUE`) by the commas and place each part into the properties `$name_error`, `$breed_error`, `$color_error`, and `$weight_error`. Each of these properties will now contain the string `'TRUE'`. You can then evaluate your messages to see if the updates were successful in a very similar technique that evaluated the results of the `set` methods.

```
print $name_error == 'TRUE' ? 'Name update successful<br/>' : 'Name update not
successful<br/>';
```

There are only a few minor differences between this format and the similar statements to evaluate the results of the `set` methods. Each of these code lines uses a different error message for evaluation (`$name_error`, `$breed_error`, `$color_error`, and `$weight_error`). Previously you used the same property (`$error_message`) for all the results from the set methods. You are now evaluating the string `'TRUE'` instead of the constant `TRUE` (the only difference in the code is the actual quotes).