Classes and Objects

A class is similar to a blueprint of a house. The blueprint contains a description (characteristics) of all the elements needed to construct the house. However, the blueprint is not the actual house itself. It describes what is possible if we hire a crew and construct the house. The blueprint is not considered to exist (as a house would exist). However, it describes the items needed to build the house (nails, drywall, and wood) and the process to build the house.

A *class* describes the characteristics (properties) of the module of code and the *actions* (methods or functions) that can occur in that code. However, it does not physically exist (within memory) until an instance of the class (called an object) is created. Once an instance is created, the characteristics and methods can be accessed. Classes and objects (when created properly) protect the characteristics (properties) from direct access. This provides the object the opportunity to verify that any request to change a value in a property is valid before the change occurs. This is commonly called *encapsulation*. To protect properties from direct access to the outside world, they should be declared using the `private` access type. `Private` access will only allow methods within the class the ability to change the values in the properties. `Set` **methods** (discussed later in this chapter) are used to change properties. `Get` **methods** (also discussed later in this chapter) are normally used to retrieve property values.

Creating a PHP Class

Let's begin by creating a basic structure for a class. You will create a `dog` class which will allow you to set some characters of the dog (size, breed, color, and name) and you will provide the ability for the dog to speak and to display the values saved in each property. You will create the dog class in a separate file (library) that can be loaded into the program (or any other program) when needed.

To create a PHP class, you use the `class` keyword and encapsulate all code within the class in {}.

Example 3-2. Basic class structure in the dog.php file

```
<?php
class Dog
{
// all code is placed here
}
?>
```

As seen in Example 3-2, the `class` keyword is lowercase. However, the name of the class, `Dog`, begins with an uppercase letter. PHP will allow you to create a class with a lowercase first letter. However, it is common practice to easily identify classes by the use of the uppercase first letter. The actual file name containing your class (`dog.php`) should also match the class name (`Dog`).

> *For more information on classes, methods, and properties visit php.net at:*
>
> *http://php.net/manual/en/classobj.examples.php.*
>
> *For videos, visit "the new boston" at:*
>
> *https://www.thenewboston.com/videos.php?cat=11&video=17175.*

Class names cannot include spaces. You should also avoid using special characters. However, the _ is permitted and commonly used to connect two words together (`set_name`). You many notice that some class names include two underscores (__) before the actual class name (`__Myclass`). However, this is not a recommended technique due to the existence of "magical" classes (we will look at two of these classes later in this chapter) that use this format.