Dynamically built arrays are not required to have values for every position in the array. If the dynamic array shown previously is passed into the `updateRecords` method, records 0 and 2 would be updated with the new information. The value in position 1 in the `dogs` array would remain untouched.

Take a moment to look at these methods. There are only two XML tags that have been coded in the methods (`dogs` and `dog`). Even those two could have been retrieved from the XML file. However, the assumption that these tags will always exist in a valid dog XML file makes logical sense. By dynamically pulling all of the other tags (`dog_name`, `dog_weight`, `dog_color`, and `dog_breed`) from the XML file, changes can be made to the file without causing any code changes. Additional tags can be added, removed, and/or changed.

Let's put it all together.

Example 6-1. The dog_data.php file

```
<?php
class dog_data
{
private $dogs_array = array(); //defined as an empty array initially
private $dog_data_xml = "";
function __construct() {
        libxml:use_internal_errors(true);
        $xmlDoc = new DOMDocument();
        if ( file_exists("e5dog_applications.xml") )        {
        $xmlDoc->load( 'e5dog_applications.xml' );
        $searchNode = $xmlDoc->getElementsByTagName( "type" );
                foreach( $searchNode as $searchNode )
                {
                        $valueID = $searchNode->getAttribute('ID');
                        if($valueID == "datastorage")
                        {
                                $xmlLocation = $searchNode->getElementsByTagName( "location" );
                                $this->dog_data_xml = $xmlLocation->item(0)->nodeValue;
                                break;
                        }
                } }
        else { throw new Exception("Dog applications xml file missing or corrupt"); }
$xmlfile = file_get_contents($this->dog_data_xml);
        $xmlstring = simplexml:load_string($xmlfile);

        if ($xmlstring === false) {
                $errorString = "Failed loading XML: ";
                foreach(libxml:get_errors() as $error) {
                        $errorString .= $error->message . " " ;  }
                throw new Exception($errorString); }
        $json = json_encode($xmlstring);
        $this->dogs_array = json_decode($json,TRUE);
        }
function __destruct()
{
        $xmlstring = '<?xml version="1.0" encoding="UTF-8"?>';
            $xmlstring .= "\n<dogs>\n";
            foreach ($this->dogs_array as $dogs=>$dogs_value) {
```