USING OBJECTS IN PHP

Unlike C++ and Java, PHP is not a true object-oriented language; however, PHP does support object-oriented programming concepts that allow programmers to write code in an object-oriented fashion.

Objects, which are packages of code composed of data and functions that make use of the data, are the core features of object-oriented programming. In PHP, an object's members, or properties, are made up of the data for the object. An object also has member functions, or methods, that are used to manipulate the object.

A class is PHP code that serves as a template for creating an object, which is also referred to as an instance of the class. It is possible to create many objects using the same class. Once created, each object can operate independently from other objects.

To create a new object, you assign a new instance of a class to a variable. For example, you can assign the `dir` class to a variable to create a `dir` object. The `dir` class is a built-in PHP class used to access the contents of a directory and is instantiated by specifying the path of the directory to be searched. When creating an object from a class you defined, you must include the `new` keyword. For more information about defining and using classes, see page 162.

Once you create a new object, you can access the properties and methods of the object. For example, the `path` property of the `dir` object contains the path to the directory specified when the object was instantiated. The `read` method returns the names of the files in the directory one at a time. The `close` method is used to close the directory once all the files have been retrieved.

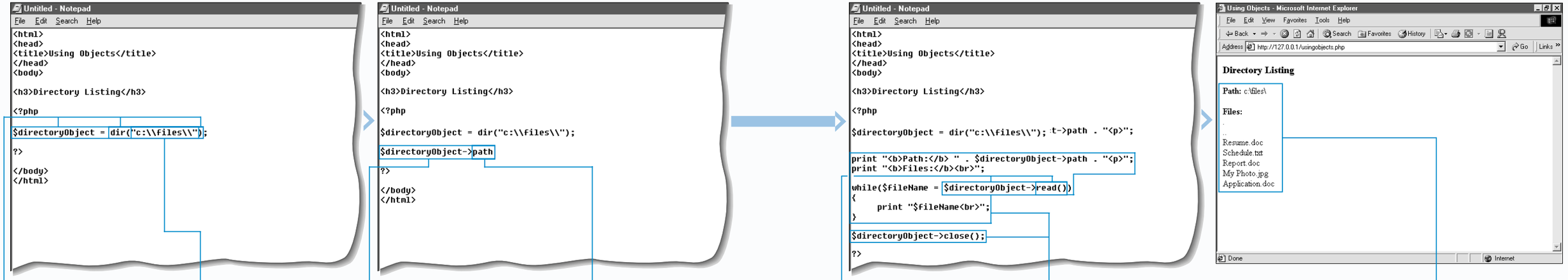
To access an object's properties and methods, you use the variable that represents the object followed by the member access operator (`->`). You then indicate the property or method you want the object to access.

Extra

The key difference between traditional and object-oriented programming is the way data and the functions that process the data are organized. Using traditional programming methods, an application is generally created as one large program, with no division between the functions of the application. For example, a shopping cart program may be designed as a single program containing customer information, item inventory and shipping functions. A slight change made to one portion of the program may greatly affect the rest of the program, which can make the program difficult to manage. Using object-oriented programming, however, allows you to structure and organize code in your applications into distinct modules, or objects. The variables and functions that deal with a specific feature, such as shipping, can be maintained separately from the other components. This can make the application easier to manage, especially if there are multiple programmers working on the same application.

One of the benefits of object-oriented programming is known as data hiding, or the black box concept. Data hiding makes classes easier to use by hiding the fields and methods of the classes from other parts of the program. The program then has to know only how to access the class, not the internal workings of the class. Data hiding is often used in programs to protect classes from tampering and to ensure that the methods of the classes are used are originally intended. A programmer can modify and maintain the code within the class without affecting the programs that use the class. This also helps ensure that objects developed by multiple people are compatible.

USING OBJECTS IN PHP



1 To create an object, type a variable name for the object followed by **=**.

2 Type the name of the class that you want use to create the object, followed by **()**.

3 Between the parentheses, type any arguments needed to create the object.

Note: In this example, the path of the directory is included as an argument.

4 To access a property of the object, type the name of the variable that contains the object, followed by **->**.

5 Type the name of the property you want to access.

■ Repeat steps 4 and 5 for each property you want to access.

6 To call a method of the object, type the name of the variable that contains the object followed by **->**.

7 Type the name of the method you want to call followed by **()**.

■ Type any arguments for the method between the parentheses.

8 Repeat steps 6 and 7 for each method you want to call.

9 Type the code that uses the accessed properties and the called methods of the object.

10 Display the PHP page in a Web browser.

■ The Web browser displays the result of using the properties and methods of an object.

DEFINE A CLASS

You may define a class that will serve as a template for an object. Classes are defined using the keyword `class` followed by the class name. The class name must begin with a letter followed by any combination of letters, numbers or underscore characters. Class names are case-insensitive, but it is good programming practice to use a consistent format for naming classes you define.

The name of the class is followed by a pair of braces { }. The code between the braces is referred to as the body of the class and may consist of properties, which are the data for the object, and methods, which are the structures that contain the code for specific actions. For information about defining methods, see page 164.

You can define properties for a class you create by using the keyword `var` followed by a name for the property preceded by a dollar sign ($). You may then assign an initial value to the property or leave the property unset. The value you assign to the property can be any data type, such as integer, floating-point number, string, array or boolean. You may define as many properties in a class as needed. All properties should be declared at the beginning of the class definition.

Once a class has been defined, you can create an object from the class. You create a variable to store the object and then use the `new` keyword followed by the name of the class to create an object.

You can also access the properties of the class in the object. To access the properties, use the variable that represents the object followed by the member access operator (->). You can then indicate the property you want to access. When accessing a property, the property name must be typed exactly as it was specified in the class definition, without the dollar sign ($).

Extra

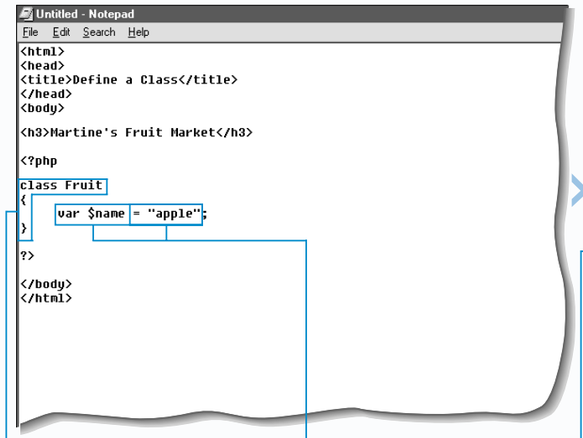
When assigning an initial value to a property of an object, you must indicate a literal value. An error will occur if you attempt to set the initial value of a property using a value obtained from a function, another variable or an expression with operators. If a value needs to be obtained from a function or an expression, you should use a method to assign the value to the property. The following example shows property declarations that will cause errors.

Example:

```
class Fruit
{
      var $name = "Granny" . "Smith";
      var $type = $name;
      var $soldOn = time();
      var $quantity = 1 + 1;
}
```

Although PHP allows you to set the property of an object using the assignment operator (=), this practice is not recommended and should be avoided. In object-oriented programming, only an object's methods should be used to access and alter the object's properties. This helps protect classes from tampering, which may cause an object to work improperly.

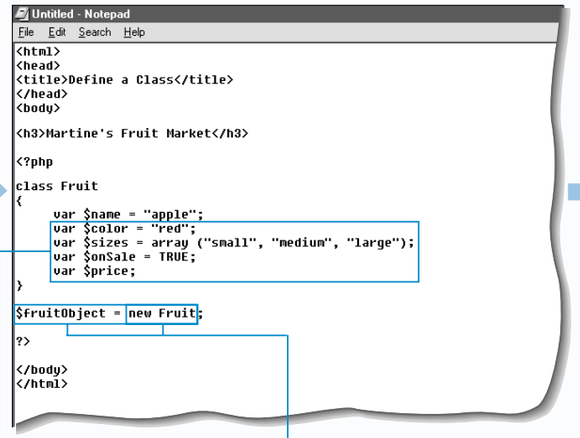
DEFINE A CLASS



1 Type **class** followed by the name of the class you want to create. Then type the opening and closing braces for the class.

2 To define a property of the class, type **var** followed by **$** and the name of the property.

3 To assign a value to the property, type **=** followed by the value you want to assign.

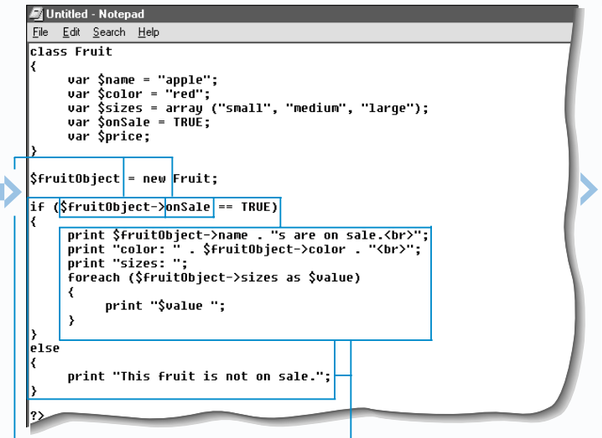


4 Repeat steps 2 and 3 for each property you want to define.

CREATE AN OBJECT USING A CLASS

5 Type the variable name for the object followed by **=**.

6 Type **new** followed by the name of the class you want to use to create the object.



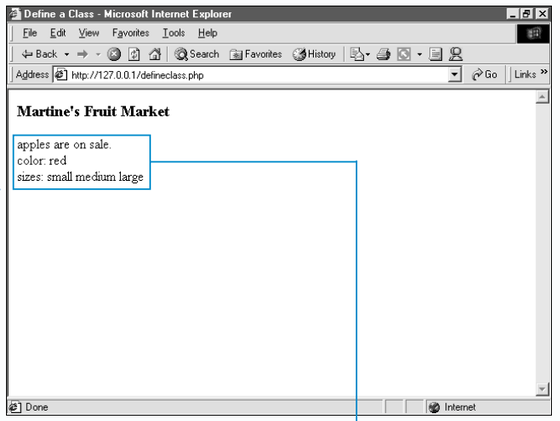
ACCESS PROPERTIES OF AN OBJECT

7 Type the name of the variable that contains the object followed by **->**.

8 Type the name of the property you want to access.

9 Type the code that uses the accessed property.

10 Repeat steps 7 to 9 for each property you want to access.



11 Display the PHP page in a Web browser.

■ The Web browser displays the result of defining and using a class.

DEFINE A METHOD

After defining a class, you can define the methods for the class. You can use methods to perform any number of tasks that relate to an object. For example, you may want to define a method that sets the properties of an object or displays the current values of the object's properties.

A method is created using the `function` keyword followed by the name of the method and a set of parentheses. You can set up a method to accept arguments. In the `function` statement, you define one or more variables between the parentheses that follow the method name. Each variable represents an argument you will pass to the method. You can use the variables you create in the `function` statement within the body of the method. When specifying the arguments of a method, you may specify a default value for an argument of the method. You use the assignment operator (=) to assign a default value to the argument.

The body of a method is enclosed in braces {}. Within the body of a method, you can use the special variable `$this` to set or access the value of a property. The special variable `$this` is followed by the member access operator (->) and the name of the property. To set a new value, you then include the assignment operator (=) followed by the new value for the property.

Similar to functions, methods may return a value using a `return` statement. When accessing a property of an object, you would ideally use a method that returns the value of the property rather than referring to the property directly. This practice of accessing the properties of an object through a method does not have to be strictly adhered to, but it is a more proper way of writing object-oriented programs.

Extra

You can set the values for an object's properties by using a constructor method. A constructor method is a special type of method that is always executed each time the class is accessed and an object is created. This makes constructor methods useful for performing initialization tasks for a new object. A constructor method must have the same name as the class for which it is the constructor. You can also call a constructor method after an object has been created.

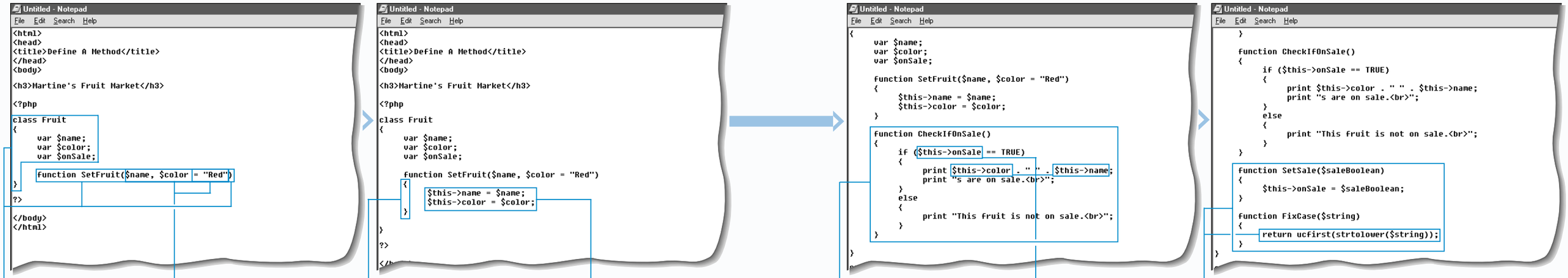
TYPE THIS:

```
class Fruit
{
      var $name;
      var $color;
      var $saleDate;
      function Fruit($name, $color)
      {
            $this->name = $name;
            $this->color = $color;
            $this->saleDate = time();
      }
      function PrintName()
      {
            print "<b>Fruit:</b> " . $this->name . "<br>";
      }
}
$fruitObject = new Fruit("apple", "red");
$fruitObject->PrintName();
$fruitObject->Fruit("mango", "green");
$fruitObject->PrintName();
```

RESULT:

Fruit: apple
Fruit: mango

DEFINE A METHOD



1 Type the code that defines a class.

2 To define a method, type **function** followed by the name of the method you want to define. Then type **()**.

3 To allow the method to accept arguments, type the names of the variables you want to store the arguments. Separate each argument with a comma.

4 To set a default value for an argument, type **=** followed by the default value after the argument.

5 Type the opening and closing braces of the method body. The code you want the method to execute is placed between the braces.

6 To have the method set the value of a property, type **$this->** followed by the name of the property. Then type **=** followed by the value or variable you want to use.

7 To define a method that accesses properties of the class, repeat steps 2 to 5.

8 To have the method access the value of a property, type **$this->** followed by the name of the property you want to access.

9 Repeat steps 2 to 5 for any other methods you want to create.

10 To return a value from a method, type **return** in the body of the method, followed by the information you want the function to return.

■ You can now call a method in the PHP script. See page 166 to call a method.

CALL A METHOD

After creating an object in your script, you can call the object's methods. Calling a method tells PHP to access and execute the code in the method.

To call a method, you type the name of the object followed by the member access operator (->) and the name of the method where you want to execute the code. The method name is followed by a set of parentheses. If the method was set up to accept arguments, you specify the arguments you want to pass to the method between the parentheses.

A method can be accessed from within the body of another method in the class definition. This is useful for defining a method that is to be used only within the class itself, such as a method used to perform internal calculations. This type of method is referred to as a private method. Private methods typically contain code that is repeatedly used in a class. Private methods are also useful for breaking up methods containing a large amount of code, which improves the readability of the code.

The special variable `$this` must be used when calling a private method within the body of another method. The special variable `$this` is followed by the member access operator (->), the name of the method and a set of parentheses. You may place any required arguments between the parentheses.

Unfortunately, PHP does not make any distinctions between the different methods that you define. There is no mechanism in PHP that will allow you to restrict the access of certain methods to within the class itself. This makes it possible to misuse code. To help avoid misuse, you should include clear, descriptive comments in your code to label private methods so they are not used outside the class definition.

Extra

You can define a method that returns the value of a property or calls a method by specifying the name of the property or method as an argument. The name of the property or method is stored in a variable, which is used to access the value of the property or call a method in the body of another method. The variable name is used in place of a literal property or method name. This is useful when you want to dynamically access an object's property or call an object's method.

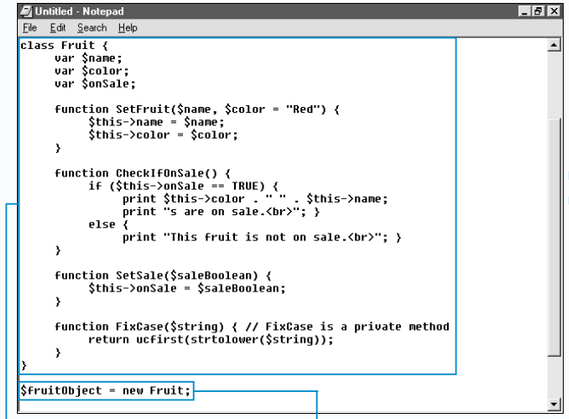
TYPE THIS:

```
class Fruit
{
      var $name;
      var $color;
      function Fruit($name, $color)
      {
            $this->name = $name;
            $this->color = $color;
      }
      function GetProperty($propertyName)
      {
            return $this->$propertyName;
      }
}
$fruitObject = new Fruit("Apple", "Red");
print $fruitObject->GetProperty("color") . " ";
print $fruitObject->GetProperty("name");
```

RESULT:

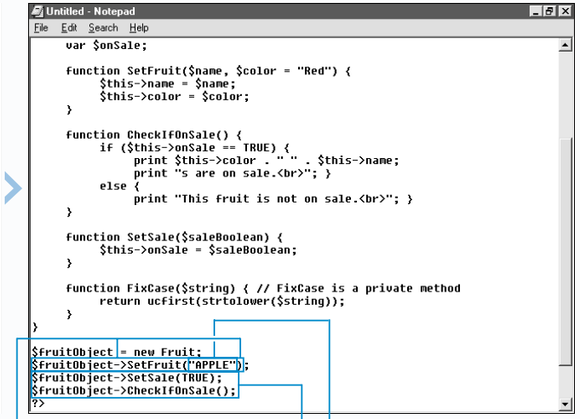
Red Apple

CALL A METHOD



1 Type the code that creates a class and its methods.

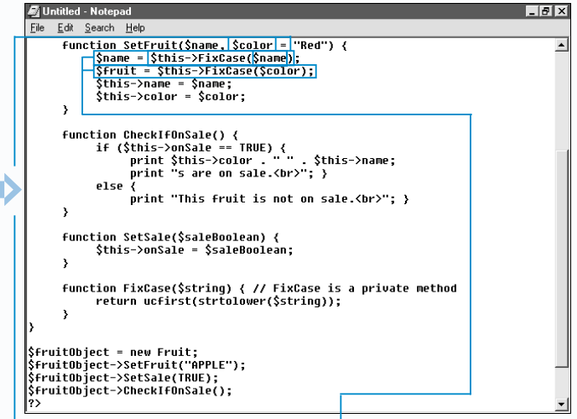
2 Type the code that creates an object.



3 To call a method of the object, type the name of the object followed by **->**. Then type the name of the method followed by **()**.

4 Between the parentheses, type the arguments you want to pass to the method. Separate each argument with a comma.

5 Repeat steps 3 and 4 for each method you want to call.

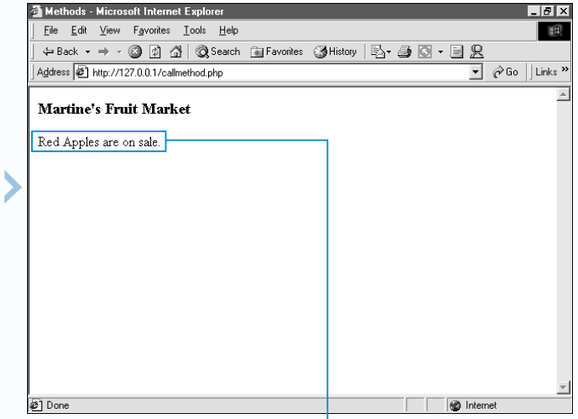


6 To call a private method within another method of the class, type **$this->** followed by the name of the method. Then type **()**.

7 Between the parentheses, type the arguments you want to pass. Separate each argument with a comma.

8 Type the code that uses the result of calling the private method.

9 Repeat steps 6 to 8 for each private method you want to call.



10 Display the PHP page in a Web browser.

■ The Web browser displays the results of calling the methods of an object.

EXTEND A CLASS

If a class you are defining is related to a class you have previously defined, you can make the new class an extension of the original class. This allows you to re-use properties and methods of the original class without having to retype the code in the new class. When you extend a class, the base class is usually referred to as the parent class, while the new class is called the child class.

When defining a class you want to use as a child class, you must use the `extends` keyword to specify the name of the class that will act as the parent class. When the child class is used to create an object, the resulting object inherits the properties and methods of the parent class in addition to any new properties and methods indicated in the child class.

If the parent class has a constructor method, the constructor method is not automatically called when a new instance of the child class is created. If you want the constructor method to execute for each instance of the child class, a constructor method must be defined for the child class. In the constructor method of the child class, you must call the constructor method of the parent class using the special variable `$this` and indicate the appropriate arguments. For more information about the constructor methods, see the top of page 165.

When creating a child class, you can override a method in the parent class that you do not want to be available when the child class is accessed. To override a method in the parent class, you must create a method in the child class that has the same name as the method you want to override. When an object is created using the child class, the method in the child class will be available instead of the method in the parent class.

Extra

A class you defined as a child class can be used as the parent class of another class. This allows you to create a chain of child classes and parent classes. Although there is no limit to the number of child classes that can be created from other child classes, you should try to limit the chain of extensions to prevent your code from becoming too confusing. You should also place information about the parent class in the child class definition to prevent potential conflicts from occurring.

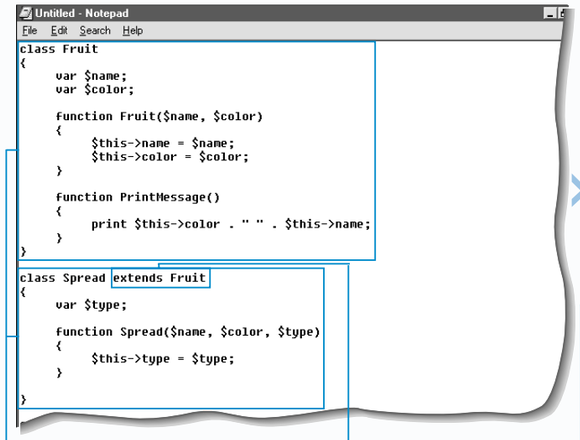
Example:

```
class Spread extends Fruit
{
      // Inherited from Fruit
      // Properties: name, color
      // Methods: Fruit, PrintMessage
      . . .
}

class Sandwich extends Spread
{
      // Inherited from Fruit and Spread
      // Properties: name, color, type
      // Methods: Fruit, PrintMessage
      . . .
}
```

It is good programming practice to design classes so they perform only a general set of tasks. Specialized methods can then be added by extending the parent class. This prevents the parent class from becoming bloated and maximizes the re-usability of code while maintaining the code's speed and efficiency. When a class becomes too large, more memory and resources are used by an object of the class. Extending a class also maintains the integrity of the parent class. This ensures that new modifications will not have any adverse effects on the rest of the program.

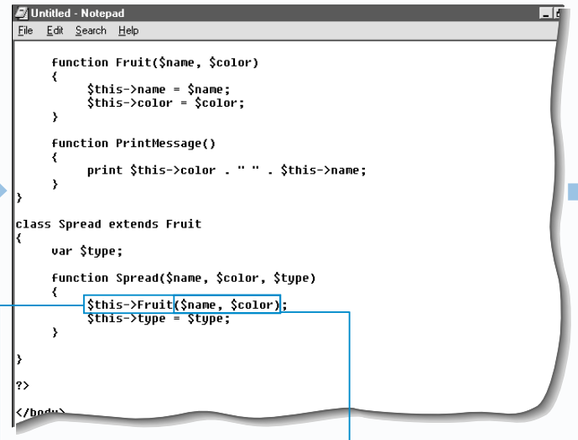
EXTEND A CLASS



1 Type the code that defines a class you want to be able to extend to another class.

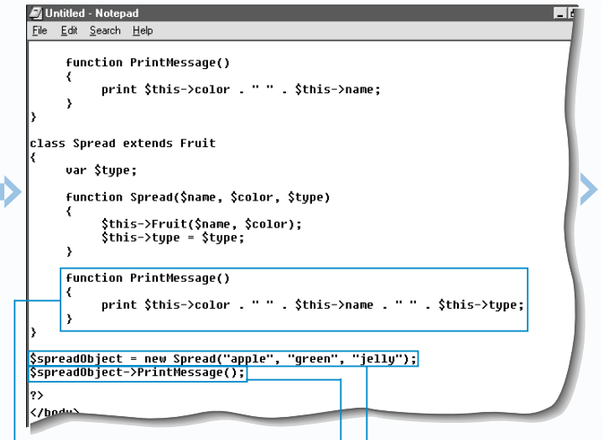
2 Type the code that defines a class you want to use as an extension of another class.

3 In the class definition, type **extends** followed by the name of the class you want to use as the parent class.



4 To call the constructor method of the parent class within the constructor method of the child class, type **$this->** followed by the name of the constructor method from the parent class.

5 Enclose any arguments for the method in parentheses.

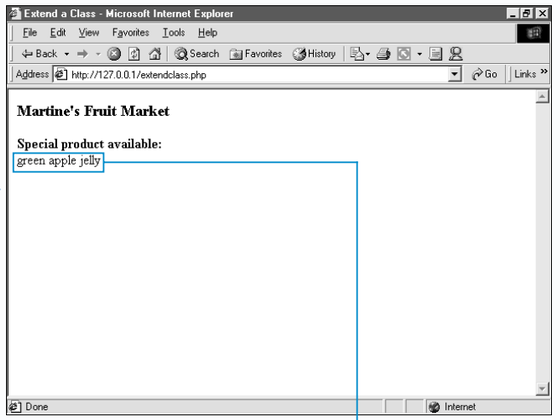


6 In the child class, you may define a method that overrides one of the methods in the parent class.

Note: The method must have the same name as the method in the parent class that it overrides.

7 Type the code that creates an object using the child class.

8 Type the code that calls a method of the object.



9 Display the PHP page in a Web browser.

■ The Web browser displays the result of creating an object of a child class and accessing its methods.

WORK WITH EXTERNAL DATA IN CLASSES

A class can access functions, constants and variables that were created outside the class definition. This is useful for re-using data in your PHP script.

A function that is located outside a class definition can be called in a method by simply specifying the name of the function and any arguments the function requires. This is useful when you already have a library of useful functions that you do not want to retype as private methods of a class. Some functions also will not work properly if they are within a class definition. For example, when using the `usort` function in a method, you need to pass a comparison function located outside of the class definition as an argument for the comparison function to be used properly.

To access a constant defined outside the class definition, you need to specify only the name of the constant. The constant definition must be placed before the class definition in the PHP script.

You can also access variables that are defined outside a class definition. The `global` keyword is used to make an external variable accessible to a method of a class. This is useful in cases where passing values to a class through a method is not practical, such as when you want to process an undetermined number of values from a form.

You should keep in mind that a class becomes less portable when it is dependent on external data. You should use external data only when necessary. When using external data, you may also want to place comments in the class definition to indicate how constants, global variables or functions are being accessed. Comments will help avoid confusion and make potential problems easier to track.

Extra

You can place a class definition in an include file to make the class accessible to multiple PHP pages. You can name the include file using the same name as the class followed by the .inc extension. When placing a class definition in an include file, you must remember to enclose the class definition within the `<?php` and `?>` delimiters. As with functions, classes can be declared only once in a script. To include a class file in a script, you can use the `include_once` statement.

Example:

```
include_once("Fruit.inc");
include_once("Vegetable.inc");
```

As the complexity of your Web site increases, you may find it more practical to use include files to organize constants, functions and classes used in your PHP pages. You must keep track of any dependencies that your PHP pages have on those constants, functions and classes. As a general rule, in a PHP page, you should specify the include files for constants first, followed by the include files for functions and then the include files for class definitions.

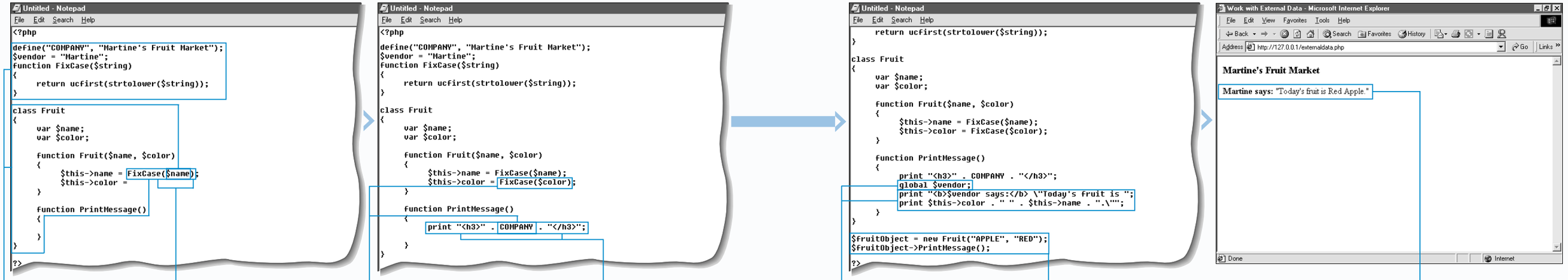
Example:

```
// Constants
include_once("color_constants.inc");
include_once("font_constants.inc");

// Functions
include_once("text_formatting_functions.inc");

// Classes
include_once("Fruit.inc");
include_once("Vegetable.inc");
```

WORK WITH EXTERNAL DATA IN CLASSES



1 Type the code that creates any constants, variables and functions that will be used in a class.

2 Type the code that defines a class.

3 To call a function created outside the class definition, type the name of the function followed by ().

4 Between the parentheses, type any arguments for the function.

5 Repeat steps 3 and 4 for each function you want to call in the class.

6 To use a constant defined outside the class definition, type name of the constant.

7 Type the code that uses the constant.

8 To access a variable defined outside the class definition, type **global** followed by the name of the variable.

9 Type the code that uses the variable.

10 Type the code that creates an object and calls a method of the object.

11 Display the PHP page in a Web browser.

The Web browser displays the result of using external data in a class.

GET INFORMATION ABOUT AN OBJECT

PHP provides several functions that allow you to obtain information about an object. You can use the functions to examine the attributes of an object without having to manually read through its class definition. This can save you time and effort, especially if you are working with include files. You can also use the functions to automate some tasks, such as generating documentation for your programs. This can help you avoid conflicts when naming methods and properties.

You can use the `get_class` function to obtain the class name of an object. The `get_class` function takes the variable that contains the instance of the object you want to check as its argument. The function returns the name of the class the object belongs to.

You can obtain a list of the properties of an object by using the `get_object_vars` function. To use this function, you must specify the variable that contains the instance of the object you want to check. The `get_object_vars` function returns an associative array containing the object's properties, including any properties inherited from parent classes. The keys of the array contain the property names and the values of the array contain the current values of the properties. Properties that do not have an assigned value are not returned in the array.

To obtain the methods of an object, use the `get_class_methods` function. To use the `get_class_methods` function, you need to specify the name of the class from which you want to get the list of methods. If you do not know the class name, you can use the `get_class` function to determine the class name of the object and then use the result in the `get_class_methods` function. The `get_class_methods` function returns an array of methods in the specified class, including any methods inherited from parent classes.

Extra

You can use the `get_class_vars` function to obtain the default values of the properties of a specified class. The function returns an associative array whose keys are the property names and whose values are the default property values. Properties that do not have a default value specified are not included in the array.

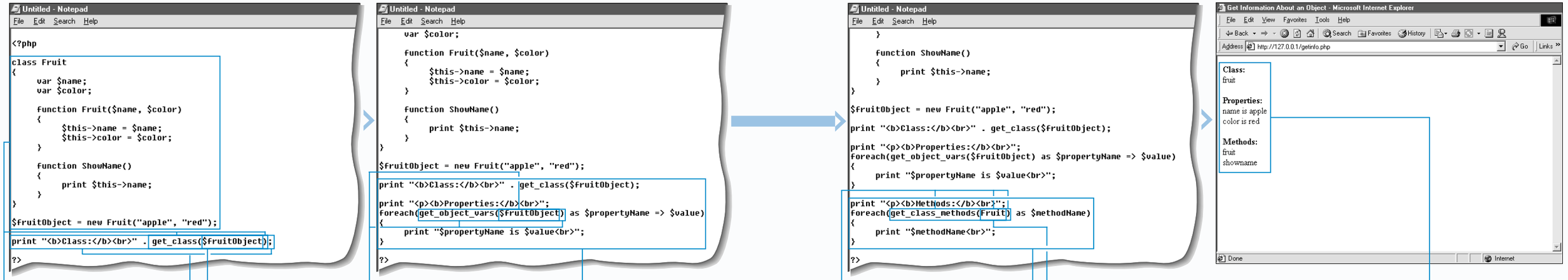
TYPE THIS:

```
class Fruit
{
    var $name = "apple";
    var $color = "red";
    var $size = "large";
    var $onSale;
}
$fruitObject = new Fruit;
print "<b>Default property values of Fruit:</b><br>";
foreach(get_class_vars("Fruit") as $propertyName => $defaultValue)
{
    print "$propertyName is $defaultValue<br>";
}
```

RESULT:

Default property values of Fruit:
name is apple
color is red
size is large

GET INFORMATION ABOUT AN OBJECT



1 Type the code that defines a class and creates an instance of the class.

2 To determine which class an object belongs to, type **get_class()**.

3 Between the parentheses, type the name of the object you want to check.

4 Type the code that uses the `get_class` function.

5 To determine the properties of an object, type **get_object_vars()**.

6 Between the parentheses, type the name of the object you want to check.

7 Type the code that uses the `get_object_vars` function.

8 To determine the methods of an object, type **get_class_methods()**.

9 Between the parentheses, type the name of the class from which the object was instantiated.

10 Type the code that uses the `get_class_method` function.

■ If you do not know the name of the class, use the `get_class` function as the argument.

11 Display the PHP page in a Web browser.

■ The Web browser displays the results of retrieving information about the object.

CHECK FOR CLASSES AND METHODS IN A SCRIPT

There are several functions available that you can use to check for the presence of classes and methods in your script. These functions are useful for troubleshooting large, complex PHP scripts. Using functions to test for certain conditions can save you from having to examine every line of code in a script.

The `get_declared_classes` function is used to determine the classes that are currently being used in the script. There are no arguments required when calling the `get_declared_classes` function. The `get_declared_classes` function returns an array containing the names of the classes being used in the script.

When viewing the array elements returned by the `get_declared_classes` function, you may see classes that you have not defined. These are built-in classes in PHP. You may want to keep these built-in classes in mind so you do not inadvertently define any classes with the same name as the built-in classes.

You can use the `class_exists` function to check if a particular class has been defined in the script. When using the `class_exists` function, you must specify the name of the class you want to check, enclosed in quotation marks, as an argument. The `class_exists` function will return a value of true if the specified class has been defined in the script.

The `method_exists` function is used to determine if a particular method belongs to an object. The `method_exists` function takes two arguments—the name of the variable that contains the object and the name of the method you want to check, enclosed in quotation marks. The `method_exists` function will return a value of true if the specified method is part of the class used to create the object.

Extra

The `is_subclass_of` function is used to determine if an object was created using a child class of a specified parent class. The `is_subclass_of` function takes two arguments—the name of the variable that stores the object and the name of the parent class you want to check, enclosed in quotation marks. If the object was created using a child class of the specified parent class, the function will return a value of true. The parent class being checked does not have to be an immediate parent of the child class. If the parent class being checked is a number of levels above the child class, the function will still return a value of true.

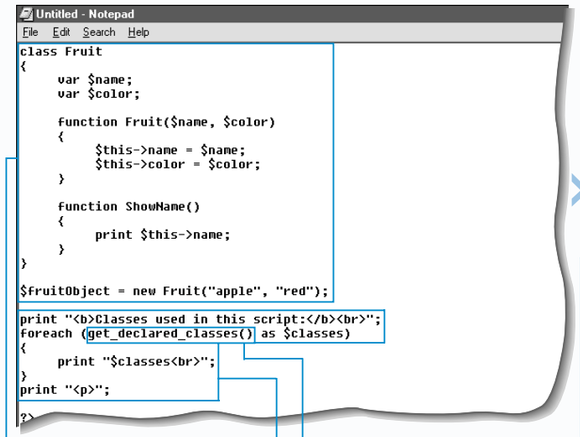
TYPE THIS:

```
class Spread extends Fruit
{
      var $type;
      function Spread($name, $color, $type)
      {
            $this->Fruit($name, $color);
            $this->type = $type;
      }
}
$spreadObject = new Spread("apple", "red", "jelly");
if (is_subclass_of($spreadObject, "Fruit") == TRUE)
{
      print "Fruit is a parent of Spread.";
}
else
{
      print "Fruit is not a parent of Spread.";
}
```

RESULT:

Fruit is a parent of Spread.

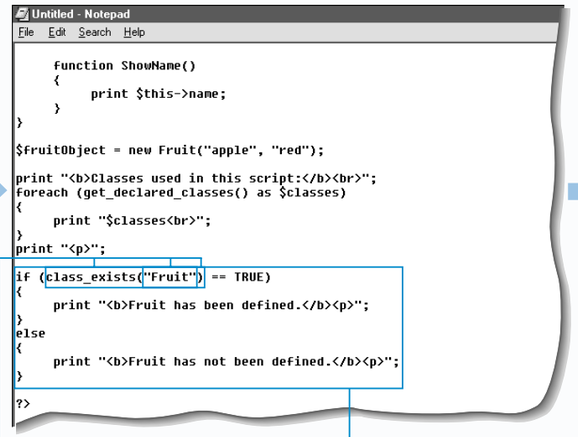
CHECK FOR CLASSES AND METHODS IN A SCRIPT



1 Type the code that defines a class and creates an instance of the class.

2 To get a list of classes used in the script, type **get_declared_classes**().

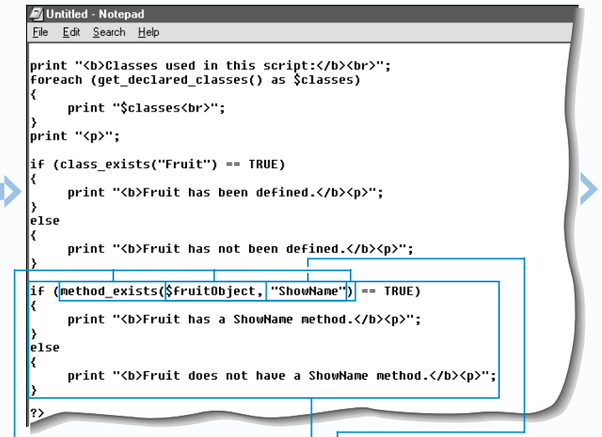
3 Type the code that uses the `get_declared_classes` function.



4 To check if a class has been defined, type **class_exists**().

5 Between the parentheses, type the name of class you want to check, enclosed in quotation marks.

6 Type the code that uses the `class_exists` function.

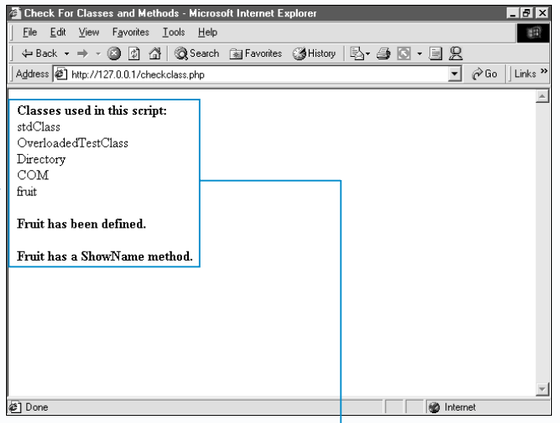


7 To check if a specific method exists in an object, type **method_exists**().

8 Between the parentheses, type name of the object followed by a comma.

9 Type the name of the method you want to check, enclosed in quotation marks.

10 Type the code that uses the `method_exists` function.



11 Display the PHP page in a Web browser.

■ The Web browser displays the results of checking for classes and methods in the script.