CS6252 Web Technologies II
PHP Tidbits: Variable Scope and the filter_input Function

Scope of Variables

The scope of a variable is the content within which the variable is defined. Generally, a PHP variable has one of the following four kind of scopes:

- Local
- Static
- Global
- Superglobal

Local Variables

A variable that is declared within a function has local scope and can only be accessed within that function. Note that a variable is declared when it is first assigned a value. A local variable is deleted when the corresponding function has completed execution. In turn, each time the function is called the variable the created again.

```
<?php
    function localVariableTest() {
        $x = 1;  // $x is a local variable and can only be used within this function
        echo “The value of x is $x”;
        $x++;    // pointless statement as $x is deleted after this statement
    }

    localVariableTest(); // prints “The value of x is 1”
    localVariableTest(); // prints “The value of x is 1”
    echo “The value of x is $x”;
            // This echo statement will cause an error as $x does not exist.
?>
```

Static Variables

A static variable is a variable that is declared within a function and preceded by the keyword *static*. Like a local variable, the variable can only be accessed in the function in which it is created. However when the function has been executed, the variable is not deleted and it retains the value. The assignment that is part of the variable declaration will only be executed the first time the function is executed.

```
<?php
    function staticVariableTest() {
        static $x = 1;  // $x is a static variable
        echo “The value of x is $x”;
        $x++;
    }

    staticVariableTest(); // prints “The value of x is 1”
    staticVariableTest(); // prints “The value of x is 2”
    echo “The value of x is $x”;
            // This echo statement will cause an error as $x does not exist.
?>
```

Global Variables
A variable that is declared outside a function has global scope and can only be accessed outside of a function (unless you use the keyword *global*).

```
<?php
    $x = 1; // $x is a global variable
    function globalVariableTest() {
        echo “The value of x is $x”;
            // This echo statement will cause an error as $x does not exist.
    }

    echo “The value of x is $x”; // prints “The value of x is 1”
    globalVariableTest(); // causes an error
?>
```

Superglobal Variables
PHP maintains some built-in superglobal variables. Most of these variables are associate arrays, like the $_POST, $_GET, $GLOBALS, $_COOKIE, and $_SESSION array, for example. A superglobal variable is available in every scope. They can be accessed outside and within functions. The array $_POST, respectively $_GET, contains the form values that are passed in by the POST method, respectively $_GET method. The associative array $GLOBALS references all global variables. It can be used to access a global variable within a function:

```
<?php
    $x = 1; // $x is a global variable
    function superglobalVariableTest() {
        echo “The value of x is ” . $GLOBALS[x];
    }

    echo “The value of x is $x”; // prints “The value of x is 1”
    superglobalVariableTest(); // prints “The value of x is 1”
?>
```

The filter_input Function

If a user of an application enters data, the input should always be validated. That means, the application should verify that the user entered valid data. The filter_input function provides an easy way to obtain and check the values referenced by some superglobal arrays, like the values of the $_GET and $_POST array and of the $_COOKIE array. For example, the function call

```
<?php
   $someAge = filter_input(INPUT_POST, ‘age’, FILTER_VALIDATE_INT);
?>
```

If the variable age is not set in the $_POST array, the call to filter_input returns NULL. If the variable age is set, but does not contain an integer value, then the function call returns FALSE. Otherwise, the function returns the value of the variable age as specified in the $_POST array.

Syntax
The syntax of the input_filter function is as follows:

```
filter_input(input_type, variable, filter, options)
```

where

input_type	specifies the array from which a variable value is to be returned. The parameter value INPUT_POST accesses the $_POST array, the value INPUT_GET accesses the $_GET array, and the value INPUT_COOKIE accesses the array $_COOKIE, for example.
variable	specifies the variable name that is referenced in the specified array and whose value is to be returned by the function call.
filter	is an optional parameter that is used to validate the variable value that is specified by the first two parameters. If the filter parameter is omitted, no validation is applied. Available PHP filters are:

FILTER_VALIDATE_BOOLEAN	Returns TRUE for "1", "true", "on" and "yes", returns a FALSE value otherwise
FILTER_VALIDATE_EMAIL	Validate value as e-mail
FILTER_VALIDATE_FLOAT	Validate value as float
FILTER_VALIDATE_INT	Validate value as integer
FILTER_VALIDATE_URL	Validate value as URL, optionally with required comp

options	specifies additional optional restrictions that are applied with the filter. For example, when applying the filter FILTER_VALIDATE_INT, the options parameter can restrict the range of a valid integer value.

Return Value
In case one of the above filters is applied, the filter_input function returns NULL is the variable is not set. The function returns FALSE if the variable valid is invalid according to the filter. Otherwise, the filter_input function returns the validated value.