Simple Variables

Professor Don Colton

Brigham Young University Hawaii

Compare watching a DVD to playing a computer game. Both are highly visual. Both have sound. Both can be done for hours at a time.

But watching a DVD is the same every time you do it. Nothing you do will change what happens next. Playing a computer game is all about you being in control. Your main goal is to change what happens next.

Within the computer, when we print "Hello, World", it's like watching a DVD. Every time we run the program, the same thing will happen. Maybe that's good. But maybe we need more than that.

In order to print "Hello Fred" when talking to Fred, and "Hello Joe" when talking to Joe, we generally use a variable to hold the name of the person we are talking to. Then we print "Hello $name" and the computer inserts the appropriate name.

This handout provides the basics of working with simple variables.

1 What's Simple?

There are many forms of information. In this section we will focus on simple pieces of information. Examples would be a number (3.1415) or a name (Fred). Non-simple pieces of information might include things like a list (Bread, Milk, Sugar, Bananas). Those will be addressed in another handout.

A simple variable holds one piece of information.

In Perl, that information can be a number or a string. A string is a series of characters like a name (Fred) or a sentence (Hello, World!).

2 Variable Names

In Perl, variable names are prefixed by a dollar sign ($) that identifies the variable as simple (or more precisely, as a scalar). The dollar sign can be thought of as part of the variable name or not. We will think of it as a prefix to the variable name.

The first real character of the variable name comes right after the dollar sign. It must be a letter, A through Z or a through z. As a special case, the underscore character is considered to be a letter.

After the first character, digits are also allowed.

There is no specific maximum length for a variable name. You can make it as long as you like.

`$ABC` and `$abc` are not the same variable name. That is, the names are "case sensitive."

Things other than letters, digits, and underscores are not allowed in variable names. Specifically, spaces are not allowed.

3 Putting Something Into a Variable

The main way to put information into a variable is to use the assignment operator which looks like an equals sign.

```
$x = 5;
```

In this example, we have a variable named `$x` and we are copying the number `5` into that variable.

```
$x = "5";
```

In this example, we have a variable named `$x` and we are copying the string `"5"` into that variable. A string is a series of zero or more characters.

```
$x = "Hello, World!";
```

In this example, we have a variable named `$x` and we are copying the string `"Hello` World!", into that variable.

```
$x = <STDIN>;
```

In this example, we have a variable named `$x` and we are copying from the keyboard into that variable. When the program reaches this statement, it will wait for the user to type something and press Enter. The letters typed together with the Enter at the end will be copied into the variable.

```
chomp ( $x = <STDIN> );
```

In this example, we have a variable named `$x` and we are copying from the keyboard into that variable. When the program reaches this statement, it will wait for the user to type something and press Enter. The letters typed but NOT the Enter at the end will be copied into the variable. The Enter is removed by the `chomp` command.

Note that each time you put something into a variable, the computer forgets what was there before. Simple variables only hold one thing at a time.

4 Printing a Variable

The following command prints the words `"Hello` World!", without using a variable.

```
print "Hello, World!";
```

We can directly print the contents of a variable. The following commands print the words `"Hello` World!",.

```
$x = "Hello, World!";
print $x;
```

We can print the contents of a variable as part of a longer statement. The following commands print the words `"I shout Hello` World! every morning.",.

```
$x = "Hello, World!";
print "I shout $x every morning.";
```

5 Calculating with a Variable

We can use a variable to hold a number for calculation. The following program asks the user for a distance in inches. It stores that value in a variable (`$in`). Then it uses the number in a calculation.

```
print "How many inches? ";
chomp ( $in = <STDIN> );
$cm = 2.54 * $in;
print "$in inches equals $cm cm.";
```

6 Meaningful Names

The computer does not ascribe any particular meaning to your variables. You can call them whatever you want. If the variable is to hold a distance in inches, you could call it `$inches` or `$in` or `$aloha`. The computer does not care.

Humans (including programmers) naturally ascribe meaning to things they recognize. A human seeing a variable name of `$aloha` will probably be confused as to what is held in that variable.

It is strongly recommended that you create variable names that describe the information stored inside. Thus, `$inches` would be a good name for a variable that holds a distance in inches. `$aloha` would not be a good name for such a variable. When you create variables that are not meaningful, you increase the chances of creating programming errors in your code.

Short variable names like `$x` are handy for limited scopes but you should not rely on the value in `$x` in far distant parts of your program. Think about how we use pronouns (he, she, it). They are useful in localized contexts. They are problematic when referring to distant objects because the context is missing. Misunderstandings can easily happen. That would be like programming errors.

Highly recommended reading: Look up "Hungarian notation" on Wikipedia for a very helpful discussion of variable naming for real projects.

7 Summary

You should know the following things.

1. Simple variable names use a prefix of `$`, begin

with a letter (or underscore), continue with zero or more letters, digits, or underscores, and can be as long as you like.

2. Upper case letters are different than lower case letters. `$abc` is not the same as `$ABC`.

3. Simple variables can hold a number or a string.

4. Simple variables only hold one thing at a time. When you put something in, the prior contents are simply lost.

5. Simple variables can be used in print statements.

6. Simple variables can be used in calculations.

7. Variable names should be meaningful to prevent programming bugs.