14.2.5 Arrays and Expansions

Click one of the buttons to take you to that part of the video.

Arrays and Expansions 0:00-0:14

In this demonstration we show how to use bash variables that contain an array of values. We also show two shell expansions, parameter expansion and command substitution.

Indexed Arrays 0:15-0:43

In bash, there are are two types of arrays, indexed and associative. Let's look first at indexed arrays where each element in the array is referenced by a number.

We can create an indexed array by assigning a value to a variable element using brackets ‘[ ]', like this. We'll use the ‘declare -p' command to see the results. However, it's good practice to first declare the variable to be an array using the ‘declare -a' command.

Assign Multiple Values 0:44-1:10

We can declare an indexed array and give it multiple values like this using the ‘-a' option. Notice that when we run a ‘declare -p', each element is assigned a number beginning at zero (0). But, the numbers don't have to start with zero (0) and they don't have to be in sequence. That's a funny quirk with indexed arrays.

Add, Change, and Reference an Element 1:11-2:11

We can add a single element like this, and we can change a single element like this. However, there's a trick to referencing an array element. We must use a shell expansion called parameter expansion or variable expansion. It uses the dollar sign ($) braces ‘{}' construct.

To illustrate, let's use the echo command without variable expansion. At first this doesn't make sense. Black is the value of the first element and the bracket nine ([9]) is what we have in the command.

What happens is that bash parses the command and performs a substitution when it encounters the dollar sign($). But it parses only $colorarray, which expands to the first element in the array. Then bash interprets the bracket nine ([9]) as a literal and echos it to the shell.

To make this work we surround our element with braces. And that works the way we intended.

Associative Arrays 2:12-3:25

For an associative array, each element is given a name. The value is "associated" with the element name. We can better understand this by looking at an example.

We declare an associative array using the ‘-A' option. And for this array we're also using the ‘-i' option so that we can use shell arithmetic to assign them values. When we add an element we give it a name inside the brackets instead of a number. And we reference elements the same way.

Now, this is a bit more advanced. But, let's look at this script. We've declared the same scorearray and a couple of integer variables we'll use to compute an average. We're using a loop that uses the element names, one by one, to reference the element's value. This value is summed with this plus equals (+=) construct. When we're done with the loop, we get the average by dividing the sum by the number of elements. There are three key parts of this script, the list of element names, the reference to the value of each element, one by one, and the number of elements.

Parameter Expansions 3:26-3:58

Let's switch topics and look at parameter expansions. We saw the dollar sign ($) braces '{}' construct that we used to reference a single element of an array and the expansion that we used in our array script. Let's look at some variations to that.

Here's an expansion that returns a substring of 15 characters beginning at the 14th character. We start counting the characters at zero (0). And here's an expansion that substitutes one sting for another.

Command Substitution 3:59-4:43

Command substitution is like parameter expansion. It substitutes the output of a command for the command itself. Here's a quick example. We have a file named urls.txt that has a list of web sites. Here's its contents. Now let's substitute the results of this cat command in an echo command. Here's what that looks like. The output of the cat command is echoed to the shell. Now let's have some fun and see if we can get firefox to bring up all these URLs at one time using the same command substitution. And here they are, all three URLs are loaded in firefox.

Summary 4:44-4:49

That completes this demonstration. We showed how to use array variables, and we showed how to use two shell expansions, parameter expansions and command substitutions.
