**Solution: Handle errors**

(upbeat music) - [Man] To solve the challenge of fixing and handling errors, I'll start by finding out what needs to be fixed by clicking the broken run arrow to view the error list. The first error says that the wait millisecond VI contains an unwired or bad terminal. I'll click the Show Error button, and on my block diagram it looks like I forgot to connect this Constance. That's an easy fix, and that resolves that issue. The remaining error on my error list says that the While Loop condition terminal is not wired, and I'll get to that in a second. First I'll connect error wires between my three functions to pass along any unexpected errors that occur. Now, if there is an error it will get passed down that line so I'll need to handle it at the end. To do that I'll press Control + Space to bring up the quick drop search. I'll search for simple error handler, press Enter, and place that at the end. I'll connect to the error outline from the close file VI, to the input on the Simple Error Handler. Now, I do not want to display a pop up dialogue if an error does occur, so I'll Right Click and create a Constance for the dialogue type. And I'll select a value of no dialogue. The last thing to take care of is the condition terminal for the While Loop. When the 'Read From Text File' VI gets to the end of the file and doesn't have anything left to read, it will throw an error. And I can use that status value to stop the While Loop. To do that I'll add an 'Unbundle by name' Node by going to the functions pallet cluster menu. And selecting 'Unbundle by name'. And I'll place that inside the While Loop. I'll wire the error cluster coming from the read VI, to the unbundle by name. And I'll wire the Status Boolean, to the Condition Terminal. Now, when the Read VI throws an error, it will stop the loop. That resolves all the errors so I can close my error list, and run this VI. And in the text from file indicator, I see the numbers, zero's through nine displayed. Stopping a While Loop when an error occurs inside of it is a common enough thing that Lab View will actually let you wire a error cluster directly to the Loops Condition Terminal. I can do that by deleting the 'Unbundle By Name', and then just connecting a wire directly from this Error Cluster into the Condition Terminal. This will have the exact same effect as unbundling the status value from the Error Cluster. Also although it won't make a difference in this particular example, it's good practice to use a Shift Register when passing a chain of Error Clusters into an Out of a While Loop. To implement that, I can Right Click on the Error Terminal, select 'Replace with Shift Register', and notice that now my cursor is a Shift Register icon and I can click on the Incoming Terminal to identify that as the other half of the Shift Register. If I was not immediately stopping the loop when the error occurs, this Shift Register would ensure that the error information is not lost, and gets passed on from loop bit oration to the next.