# eval

The **eval** built-in is fairly controversial in the Python community. The reason for this is that eval accepts strings and basically runs them. If you were to allow users to input any string to be parsed and evaluated by eval, then you just created a major security breach. However, if the code that uses eval cannot be interacted with by the user and only by the developer, then it is okay to use. Some will argue that it's still not safe, but if you have a rogue developer, they can do a lot more harm doing other things than using eval.

Let's take a look at a simple example:

```
var = 10
source = 'var * 2'
print (eval(source))
#20
```

As you can see, we create two variables. The first is assigned the integer 10. The second has a string that has the same letters as the variable we just defined and it looks like we're going to multiply it by two. However it's just a string so it doesn't do anything. But with eval, we can make it do something! You can see it in action on the very next line where we pass the string into eval and we get a result. This is why people think that eval can be dangerous.

There is another built-in function called **exec** which supports the dynamic execution of Python code. It's a somewhat "dangerous" built-in too, but it doesn't have the bad reputation that eval does. It's a neat little tool, but use it with caution.