Writing Scripts - Keyword Arguments

An argument lets a script use additional information. Like local variables, a script has to declare what arguments it can take. For example, consider the message:

```
cash-register ring-up-item price 3.59 quantity 2.
```

This message takes two keyword arguments: price and quantity. A script for the message ring-up-item would have to declare these two arguments:

```
$ price p, quantity q.
```

You can think of this as declaring two local variables, p and q, which are set by the keyword arguments price and quantity. The only difference between arguments and local variables is that the script can't set an argument, its value comes from the message expression.

The whole script for ring-up-item is:

```
$ price p, quantity q.

$ cost.

cost := p * q.

sub-total := sub-total + cost.
```

You should add this script to cash-register. Do this in the browser:

1.
Caret on the left side of the window. An insert property note appears.

2.
Enter the name to be ring-up-item, and tap Script: Yes. The script window appears. You will not see the property appear in the browser window until after the next step.

3.
Enter the script above into the window and tap Save.

Now the cash-register can be sent the message ring-up-item. In the workspace, execute the expression:

```
cash-register ring-up-item price 3.59 quantity 2.
```

You should see the fields in the form view update to reflect the new purchases.

Keyword Argument Names

The names you can choose for both the keyword and variable names follow the same rules as for all variables: any combinations of letters, digits and dashes, starting with a letter. We could have chosen more explanatory names in the above expressions:

```
$ price how-much, quantity how-many.

$ cost.

cost := how-much * how-many.

sub-total := sub-total + cost.
```

Often the names of the keywords are the most descriptive names for the variables. In this example price and quantity are probably even more clear than how-much and how-many. This leads you to want to write the declaration like:

```
$ price price, quantity quantity.
```

You can write this, but there is a short cut: if you want the name of the variable to be the same as the name of the keyword, then you can just write the keyword. The whole script would now look like:

```
$ price, quantity.
```

```
$ cost.


cost := price * quantity.


sub-total := sub-total + cost.
```

»

You can see that the choice of the variable name is totally up to you. It can be the same, or different than the name of the keyword. Sometimes you want them to be the same because it makes the code clear, and the keyword is the best descriptive name. Sometimes you want them to be different because you use the variable often in the script and you'd like it to be short (like p and q). Choice of variable names is a personal style.

»

On the other hand, choice of keywords makes a bigger difference. Every time the message is sent, the keyword will be used identify the value. The keyword should always be descriptive. Consider the following two messages:

```
ring-up-item cash-register price 5 quantity 3


ring-up-item cash-register p 5 q 3.
```

The second is hard to read and remember. Unless you have the script open in a window, you're not likely to remember what p and q are. Choice of keyword is important and you should always give them some thought.