#macro statement

Under certain circumstances, you might find that you often repeat several lines of a *VTL* code in multiple areas inside your template. To solve this problem, Velocity provides the **#macro** directive that allows you to repeat a segment of *VTL* codes. The first thing you have to do is to declare or create the macro itself, for example:

```
#macro (HelloWorld)
Hello World! It is such a beautiful world!
#end
```

Then within the template, you can call this macro and it will print "Hello World! It is such a beautiful world!"

To call this macro, for example, type:

```
#HelloWorld()
```

When the macro in the above example is passed on to the Velocity Template Engine, "Hello World! It is such a beautiful world!" will be printed out as a result. You may wonder why this is important, but imagine having to print a note with 10 lines of text in different parts of the document. The Velocity macro helps you save the space and time. It also allows you to pass the value from a variable on to a method parameter as you would using a Java method. To do this you have to create a macro, for example:

```
#macro( myMacro $color $somelist )
#foreach( $something in $somelist )
$something is $color.
#end
#end
```

In the above example, you have created a macro called "**myMacro**" that accepts two variables, the first variable is supposed to be a string and the second variable called "**somelist**" is supposed to be a list (note that when creating a macro that accepts variables always remember what types of variables it accepts, otherwise an error will occur during report generation). To call **myMacro** type, for example:

```
#set ($list = ["A Rose", "Blood", "Strawberry"])
#set ($color = "red")
#myMacro($color $list)
```

The above example shows that "myMacro" has been called and there are two variables in the brackets, **$color** and **$list**, separated by a space. The result of the example:

```
A Rose is red.
Blood is red.
Strawberry is red.
```