Velocity CheatSheet with Selected Tools

Cascade CMS utilizes version 1.7 of the Velocity Template Language developed and supported by Apache. Velocity is based on Java. What follows complements Apache's Velocity User Guide but is more tailored to the use of Velocity in the Cascade CMS app. A second cheatsheet (Cascade-Specific Velocity) focuses on variables and tools unique to the CMS.

Within Cascade, information to be displayed on a web page is stored within an XML file. Originally, the app presumed the use of the XPathTool in conjunction with an Index Block. The Index Block provided the means to generate the appropriate XML file; the XPathTool provided the means of reaching into the file to grab relevant information. The tool, XPath, was developed, and is supported, by W3 Schools; its use in Cascade involves only slight modifications. More about XPath can be found via these links: XPath Tutorial, XPath Cheatsheet. XPath and, within Cascade, the XPathTool provide very powerful means of culling the exact information desired from an XML file.

Note on syntax and terminology: *The code samples here are a mix of pure Velocity (as it exists outside of Cascade) and selected, older, velocity code as implemented in the context of the XPathTool. Occasionally, the syntax is not consistent between the two variants. Just learn it and live with it!*

In an XML file, information is stored within nodes ("<node>"); nodes are nested to organize the information. There is a deep connection between HTML and XML.

Basic Velocity Code

All statements in Velocity start with a "#" symbol; all variables start with a "$" sign.

Variable naming conventions:

All variables must start with a letter, either lowercase or uppercase. Otherwise they can contain numbers, and/or underscores.

```
$abc    $ABC    $a12    $oneTwo    $one_two
```

Variables are duck-typed dynamically. Use .class to determine the type of variable.

```
$string.class  ..... class java.lang.String
$number.class  ..... class java.lang.Integer  or  class java.lang.Double
$array.class  ..... class java.util.ArrayList
$hash.class ..... class java.util.LinkedHashMap
```

When a variable is used inside quote marks, we add curly braces: "${var}"

Comments

```
## This is a Velocity comment.
#* This is a multiline comment.
Line 2. *#
```


3423 Piedmont Road NE, Suite 520 · Atlanta, GA 30305 o: 678.904.6900 f: 678.904.6901 w: www.hannonhill.com

Assignment: #set($var =)

```
#set ($dir = "www" )
#set ($page = "index.html" )
```

Interpolation: #set ($path = "/$dir/$page")

$path ..... /www/index. html

Concatenation: #set($path = “/” + $dir + “/” + $page)

$path ..... /www/index. html

Directives (Loops and Macros are discussed below)

Break, Stop

```
#foreach($num in [1..4])
    #if($num == 3)
        #break  ## breaks the loop but executes any subsequent code in the file
        #stop   ## terminates execution entirely
    #end
    $num
#end
```

Evaluate (This Directive takes a string and evaluates it as code!)

```
#set($var1 = "peaches")
#set($num = "1")
#set($dynamicVar = "$var$num")
        ## $dynamicVar is now the string '$var1'.  Check with:
            Var1: $var1
            DynamicVar: $dynamicVar
            Variable Class: $dynamicVar.class
#evaluate($dynamicVar)  ## evaluates the string as a variable!

#macro(Greetings)
   Hello, Earthing!
#end
#set($macro = "#" + "Greetings")
#evaluate($macro) ..... Hello, Earthing!
```

Traversing XML nodes:

.getChild(“nodeName”).value or .getChildren(“nodeName”).values

Math Operations (+, -, *, /, %, >, <, ==)

```
#set ($value = $foo + 1)        ## Spacing around math operators is required.
#set ($value = $foo / $bar)     ## If $foo and $bar are integers, the result is an integer.
                                ## If not integers, the result is a rational number.
#set ($foo = $bar % 5)          ## Modulus operator.
```

If Statements

```
#if ($foo)  #end                        ## True if $foo is defined.
#if (!$foo)  #end                       ## True if $foo is not defined; False if defined.
#if ($foo && $foo.value)  #end          ## False as $foo.value cannot be evaluated.
#if ($foo && $bar)  #end                ## True if both $foo AND $bar are defined.
#if ($foo || $bar)  #end                ## True if either $foo OR $bar are defined.
```

String Manipulation

CHANGE CASE

```
#set($name = "SPONGE BOB")
#set($name = $name.toLowerCase())
$name  ……  sponge bob
$name.toUpperCase()  ……  SPONGE BOB
$name  ……  sponge bob
```

CONTAINS

```
#set($fruits = "Apple, Banana, Cherry")
#if( $fruits.contains( "ry"))  Fruit found!  #end  …… Fruit found!
```

EQUALS

```
#set($countryCode = 'ja')
#set($Europe = ['ee', 'ja', 'ku', 'aa'])
#foreach ($eu in $Europe)
    #if($eu.equals($countryCode))
        I found the country code "$countryCode" in $Europe!
    #end
#end
```

REPLACE ALL

```
#set($phrase = “must eat cake”)
$phrase.replaceAll("\s|\/", "_"))  ……  must_eat_cake
```

STARTS WITH, ENDS WITH

```
#set($images = ["a.png", "b.jpeg", "c.gif"])
#foreach($image in $images)
    #if($image.endsWith(".png")) $image #end  ……  a.png
    #if($image.startsWith("c")) $image #end     ……  c.gif
#end
```

SPLIT

```
#set($path = "www/news/article.html")
#set($paths = $path.split("/"))
$paths.class  ……  class [Ljava.lang.String;
#foreach($path in $paths)  $path  #end  ……  www
                                            news
                                            article.html
```

MISCELLANEOUS

```
#set($x = "")
#if($x.isEmpty())  EMPTY #end  ……  EMPTY
```

Node from an XML file listing pages & folders:

```
    <system-page id="d1e60d90ac1e000a7a9bed9f37008fb1" current="true">
$page.getName() …… system-page
$page.getAttribute("current").value  ……  true
```


3423 Piedmont Road NE, Suite 520 · Atlanta, GA 30305 o: 678.904.6900 f: 678.904.6901 w: www.hannonhill.com

Arrays

```
#set($array = ['abracadabra', 45, 'pig', 'Banana'])
$array.size()  ……  4

Find a given item:
        $array[0] or $array.get(0)  ……  abracadabra
        $array[-3]  ……  45

Add a new item:
        #set($x = $array.add(23))  ## Unless wrapped in the set directive, this returns "true".
        $array  ……  [abracadabra, 45, pig, Banana, 23]

Insert at a specific location:
        $array.add(1, 'test')
        $array  ……  [abracadabra, test, 45, pig, Banana, 23]

Change at a specific location:
        $array.set(1, 'bubble')
        $array  ……  [abracadabra, bubble, 45, pig, Banana, 23]

Remove an item:
        $array.remove()  Strings can be removed by identifying the string (.remove("string"));
                         integers are removed by identifying the position in the array.

Add all items of one array to another:
        #set($dummy = $array1.addAll($array2))
```

Looping Statements

```
<ul>
  #foreach ($item in $list)  ## generate a set of list elements.
        <li>$item</li>
  #end
</ul>

Foreach methods:
    $foreach.index      (starts at 0)
    $foreach.count      (starts at 1)
    $foreach.hasNext  (boolean)

    #foreach($number in [0..5])
        $number #if($foreach.hasNext),#end  ……  0, 1, 2, 3, 4, 5
    #end

    #set ($range = [$start..$end])  ## Range endpoints can be set as integer variables.
```

Hashmaps

```
#set($hashmap = {})
#set($whats-for-breakfast = {
          "eggs":"blah",
         "english muffin":"carbs but love",
         "cereal":"should have bought milk!"
         })

#foreach($key in $whats-for-breakfast.keySet())
        $key:  $whats-for-breakfast.get($key)  ……..  ## The key, value pairs.
#end

Hashmap: $whats-for-breakfast  ……..  {eggs=blah, english muffin=carbs but love,
                                       cereal=should have bought milk!}
Keyset: $whats-for-breakfast.keySet()  ……..  [eggs, english muffin, cereal]
Values: $whats-for-breakfast.values()  ……..  [blah, carbs but love, should have bought milk!]

Specific Value: $whats-for-breakfast.get("cereal")  ……..  should have bought milk!
                $whats-for-breakfast.eggs  ……..  blah
                #set($var = "english muffin")
                $whats-for-breakfast[$var]  ……..  carbs but love

Booleans:
        $whats-for-breakfast.containsKey("eggs")  ……..  true
        $whats-for-breakfast.containsValue("blah")  ……..  true

Add a Key:Value Pair
        #set($item = "toast")
        #set($comment = "only with jam!")
        #set($dummy = $whats-for-breakfast.put($item, $comment))

        $whats-for-breakfast  ……..  {eggs=blah, english muffin=carbs but love,
                                     cereal=should have bought milk!, toast=only with jam!}
```

Macros

```
#macro( macroName $n1 $n2 $n3)
   ## macro code
#end

To import a macro:
        #import(path to macro file)

To call  a macro:
        #macroName($v1 $v2 $v3)

A macro can call other macros and/or itself.
```

Basic Velocity Tools

Select List of Older Cascade-Specific, Velocity-Based, Tools

Velocity-Based Tool	Methods	Snippet
XPath Tool (2 methods)	$_XPathTool.selectSingleNode $_XPathTool.selectNodes	xp
Escape Tool (15 methods)	$_EscapeTool.xml($string) $_EscapeTool.html($string) $_EscapeTool.javascript($string)	esc
Serializer Tool (2 methods)	$_SerializerTool.serialize($collection, true) $_SerializerTool.toJson($collection, true)	ser
Date Tool (18 methods)	$_DateTool.getDate($date) $_DateTool.format("format", $date)[1]	date
Display Tool (11 methods)	$_DisplayTool.list(["a", "b"], ",") $_DisplayTool.plural($a.size(), "set", "sets")	---
Field Tool (1 method)	$_FieldTool.in()	---
List Tool (4 methods)	$_ListTool.removeNull() $_ListTool.reverse() $_ListTool.toList()	lis
Math Tool (19 methods)	$_MathTool.mod() $_MathTool.toInteger() $_MathTool.toNumber()	mat
Number Tool (7 methods)	$_NumberTool.currency() $_NumberTool.percent()	---
Regex Tool (1 method)	$_RegexTool.compile()	reg
Sort Tool (2 methods)[2]	$_SortTool.sort($collection) $_SortToo.addSortCriterion("selector", "language", "text | number | qname", "ascending | descending", "lowerfirst | upperfirst")	sor
String Tool (4 methods)	$_StringTool.generateUUID() $_StringTool.substringAfter() $_StringTool.substringBefore()	str

[1] Friendly date formats (such as MM/DD/YYYY) are available here.
[2] Invoking the .sort method alone automatically includes .addSortCriterion. Criteria not needed can be left as empty quotation marks.

3423 Piedmont Road NE, Suite 520 · Atlanta, GA 30305 o: 678.904.6900 f: 678.904.6901 w: www.hannonhill.com