# Utility Function: RoundToClosest()

Posted At : March 22, 2010 10:04 PM | Posted By : Jon Hartmann
Related Categories: ColdFusion, Utility Function

Its been a while! I got an email from a reader who wanted some help with a bit of image manipulation that has roused me back into CF coding! The particular bit of functionality he wants is rather complex, and so its been forcing me to make a bunch of utility functions to piece it together. First thing I ran into: rounding a number in CF is a little tricky. If you want to round to 2 decimal places, you should multiply the number by 100, call Round() and then divide by 100 again. What a waste. Luckily a little trickery with NumberFormat() can get us what we want.

```
<cffunction name="RoundToClosest" output="false" returntype="numeric">
    <cfargument name="Value" type="numeric" required="true" />
    <cfargument name="Places" type="numeric" required="false" default="0" />

    <cfset var local = StructNew() />

    <!--- Create the mask for NumberFormat --->
    <cfset local.mask = "0" />

    <!--- If we're doing decimals, format the mask correctly --->
    <cfif arguments.Places gt 0>
        <cfset local.mask = local.mask & "." & RepeatString("0", arguments.Places) />
    </cfif>

    <cfset local.returnValue = NumberFormat(arguments.Value, local.mask) />

    <!--- Special Checks to prevent -0 oddity --->
    <cfif local.returnValue eq "-0">
        <cfset local.returnValue = 0 />
    </cfif>

    <cfreturn local.returnValue />
</cffunction>
```

The code is pretty straight forward; the only bit of weirdness is right at the end. When testing this function against some very-near-to-zero-negative-numbers, the function started to return weird results. In particular, with the value -.612342345e-16, the function returned "-0". Since 0 cannot be negative, I specifically check for this and set it back to 0. I'm not sure if "-0" could cause computational errors or not, but I'd prefer not to find out the hard way.

More to come!