Converting Fusebox 2 to FW1 - Creating Scaffolding

Posted At : February 28, 2019 3:19 PM | Posted By : Mark Kruger
Related Categories: ColdFusion, Coldfusion Tips and Techniques

A quick Muse post with a bit of code. I know I know it's been a while and my coding has suffered. Still, some of you may find this useful.

I'm converting a Fusebox 2 application to FW1 and I wrote a simple script to automate some of the shell files I need. FW1 uses a URL convention that looks similar to Fuesbox. In Fusebox you have a 2 part URL param that dictates what your code is supposed to do. For example, *fuseaction=reports.users* would logically be a "circuit" called "reports" - think of it as collection of code or an application within a suite of applications. The second part of the fuseaction dictates which report exactly is supposed to be run.

FW1 is not dissimilar to this approach although it tends to introduce complexity for complexity sake at times (don't @ me). In FW1 an "action" parameter dictates which controller to run which in turn calls services and views as needed to set up a page or action. So FW1 may have *action=reports.users* - it looks quite familiar.

Since my Fusebox application is well organized I created a script that builds off the circuit and creates the necessary FW1 files. For each circuit I am going to create:

- circuitName.cfc in the FW1 /controllers folder.
- circuitName.cfc in the FW1 /model/services folder.
- circuitName.cfc in the FW1 /model/DAO folder.
- A folder named for the circuit in the FW1 /views folder with a placeholder file within (default.cfm) for my eventual content.

The script is pretty easy and it's designed to be run within the fusebox application where the FW1 application is accessable via file operations. First some setup:

```
<!--- MAK: location of my F1 files. --->
<cfset targetFW1 = "E:\eclipse-ws\new-fw1\trunk\">

<!--- MAK: Set up our target Dirs for FW1. --->
<cfset controllerDir = targetFW1 & "controllers\">
<cfset serviceDir = targetFW1 & "model\services\">
<cfset DAODir = targetFW1 & "model\DAO\">
<cfset viewDir = targetFW1 & "views\">
```

Next I created some placeholder files for controller, services and DAO. I'm going to read those files into variables.

```
<cffile action="read" file="#templateDir#dao.txt" variable="daoContent">
<cffile action="read" file="#templateDir#service.txt" variable="servicesContent">
<cffile action="read" file="#templateDir#controller.txt" variable="controllerContent">
```

Then I'm going to use my Fusebox applications structure called "Fusebox.circuits" and loop through it taking action on my plan.

```
<!--- MAK: loop through them and check them out. --->
<cfloop collection="#fusebox.circuits#" item="circ">

    <!--- MAK: Does the controller exist? --->
```

```
    <cfif NOT fileExists(controllerDir & circ & ".cfc")>
        <cffile action="write" file="#controllerDir##lcase(circ)#.cfc"
output="#controllerContent#">
    </cfif>
    <!--- MAK: Does the services file exist? --->
    <cfif NOT fileExists(serviceDir & circ & ".cfc")>
        <cffile action="write" file="#serviceDir##lcase(circ)#.cfc"
output="#servicesContent#">
    </cfif>
    <!--- MAK: Does the DAO file exist? --->
    <cfif NOT fileExists(DAODir & circ & ".cfc")>
        <cffile action="write" file="#DAODir##lcase(circ)#DAO.cfc" output="#daoContent#">
    </cfif>
    <!--- MAK: Creat the Directory in the view along with a default.cfm --->
    <cfif NOT directoryExists(viewDir & lcase(circ))>
        <cfdirectory action="create" directory="#viewDir##lcase(circ)#">
        <cffile action="write" file="#viewDir##lcase(circ)#\default.cfm" output="<h4>Hello
World</h4>">
    </cfif>
</cfloop>
```

That's it. The end result is matching DAO, Controller and service files and view folders. Of course I may delete some of them or merge or whatever as my FW1 application takes shape, but having a matching convention with Fusebox let's me examine code from one into the other without a lot of fuss.

Follow Up

I created a script that handles the "second" part of fuseaction and places an CFM in the views folder as a placeholder. Basically "reports.users" should result in a /views/reports/users.cfm file containing HTML. This is where the eventual display code will be housed.