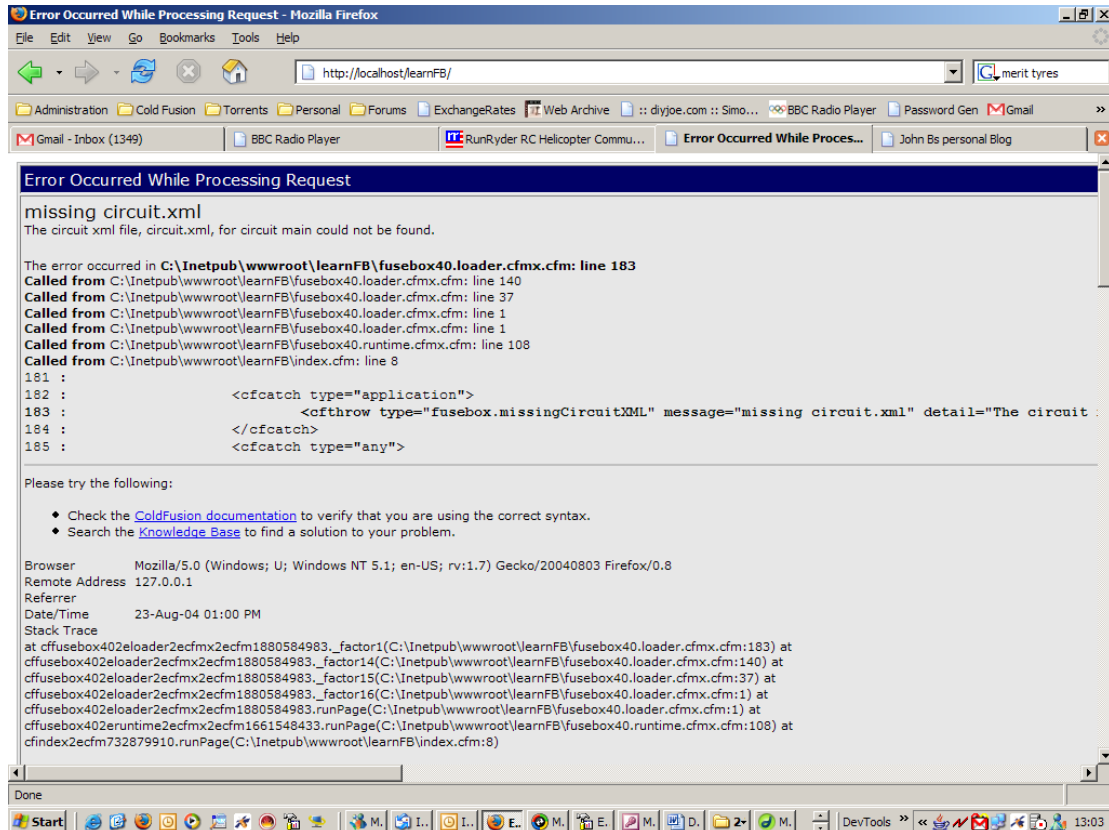


Lesson 1 – HelloWorld!

So you've decided to learn Fusebox - you've downloaded the core files, extracted them – pointed your browser at the folder and seen an error message like below...what next?

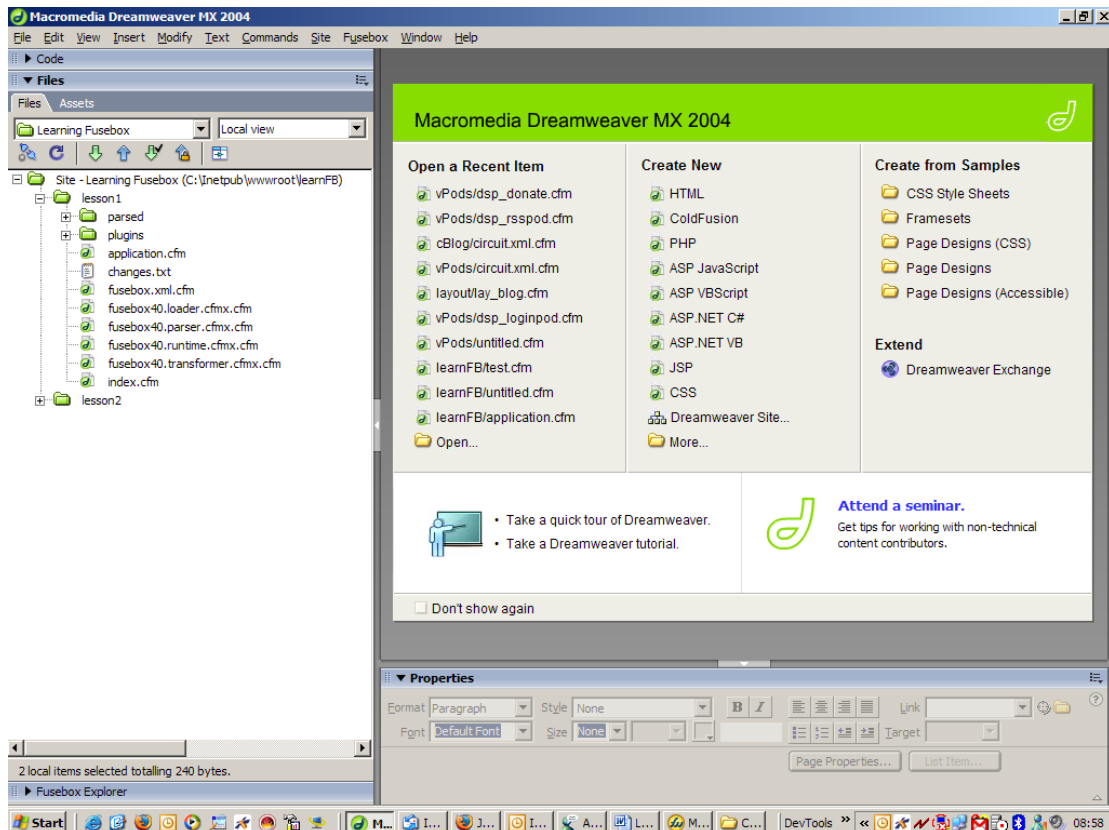


Results of running the core files, as extracted.

Extracting a heap of files that do nothing can be really frustrating especially if you have no idea what to do next as the core files certainly don't give any clues as to what you need to do to get an application up and running.

This first article in my 'LearnFB' series will get you up and running with a very simple 'hello world' Fusebox 4 application. Later articles will introduce Content Variables, XFA's, Layouts, dealing with CFCs and MVC. So without further ado, let's get started.

First off, download the core files from `http://www.fusebox.org`. I'm going to be working from folders corresponding to the lesson number beneath `c:\inetpub\wwwroot\LearnFB` accessible via my browser as `http://localhost/learnFB` so to avoid confusion you should do the same or amend URLs where necessary throughout.



the Dreamweaver MX 2004 workspace

Also, you'll notice I'm using Dreamweaver 2004 as my preferred code editor. I have a number of extensions installed in Dreamweaver, firstly Sandy Clarks Fusebox4 extension¹ and the Fusebox Explorer². Sandy's extension provides tag insight for the various XML files that are used throughout Fusebox4 and is darn useful when you forget some syntax.

So, now we've got our files extracted let's go ahead and make the 'Hello Wold' application.

In the folder create a file dsp_helloworld.cfm containing the following code;

```
<html>
<head>
<title>Hello world</title>
</head>

<body>
<h3>Hello World!!!</h3>
</body>
</html>
```

We've now got our dsp_ file that will display the message but we need to do a little more to get the application to work.

¹ <http://www.shayna.com/docs/dwmxfb4.zip>

² <http://www.cfopen.org/fuseboxexplorer>

Create another file called `circuit.xml.cfm` – this file uses XML to define the fuseactions within a particular circuit in your application.

Open the new file, remove all the contents and paste the following code into it,

```
<circuit access="public">
    <fuseaction name="helloworld">
        <include template="dsp_helloworld.cfm"/>
    </fuseaction>
</circuit>
```

At the moment you don't need to worry about what the code actually means, I'll come back to that later.

Next we need to edit the `fusebox.xml.cfm` file to assign an alias for our circuit as well as some other config parameters for the fusebox so go ahead and open the `fusebox.xml.cfm`.

At the top you'll see a chunk of code that defines the circuits in your application,

```
<circuits>
    <circuit alias="main" path="" parent="" />
</circuits>
```

since we're only making a really simple first application we don't need to worry about path and parent for the time being, but let's change the alias to LearnFB so it should now look like this;

```
<circuits>
    <circuit alias="LearnFB" path="" parent="" />
</circuits>
```

Underneath the circuit definition in the `fusebox.xml.cfm` file you'll see a whole heap of parameters, at the moment we're interested in the `defaultFuseaction` parameter – as it's name suggests this is the default fuseaction that is run if a fuseaction isn't specified in the URL. `defaultFuseaction` is used primarily when you hit an application for the first time. It takes a value using `circuit.fuseaction` notation – just before we named our circuit LearnFB and in our `circuit.xml.cfm` file we called the fuseaction `helloworld` so go ahead and change the line to look like;

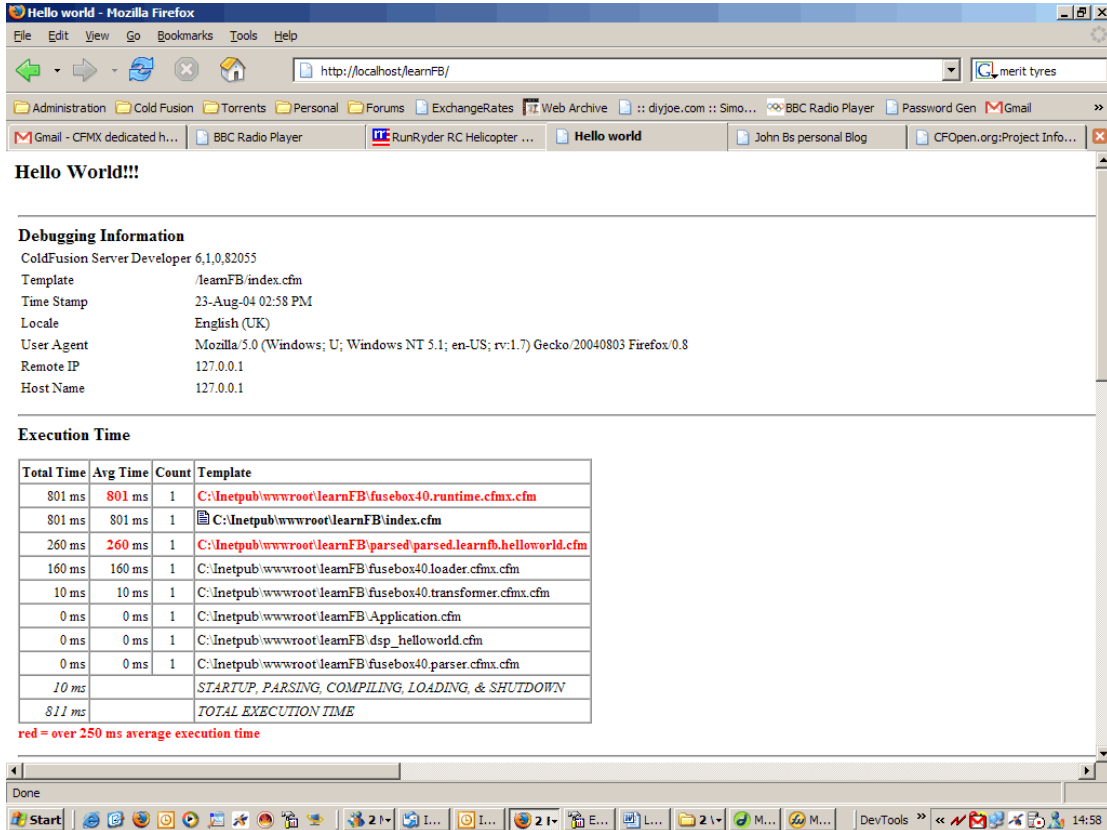
```
<parameter name="defaultFuseaction"
value="LearnFB.helloworld" />
```

Summarising what we've done so far,

1. created our `dsp_helloworld.cfm` file

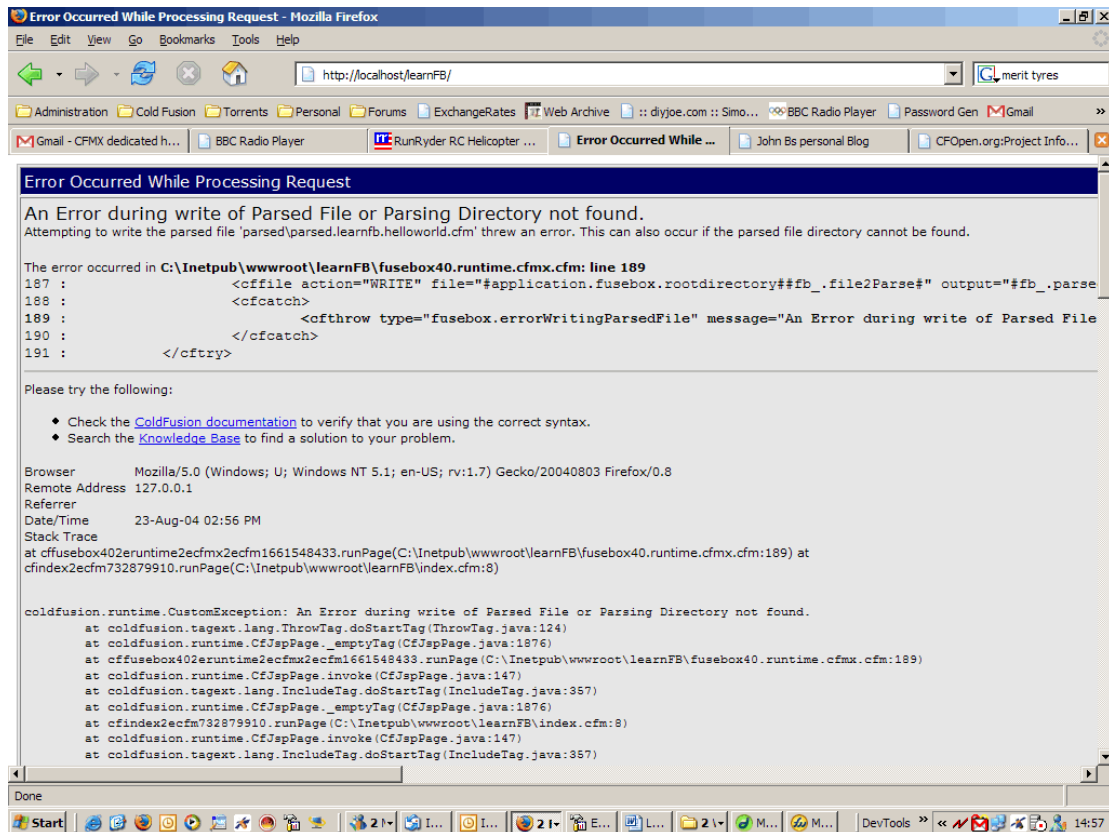
2. created out circuit.xml.cfm file and added our fuseaction
3. registered our circuit in the fusebox.xml.cfm file and changed the defaultFuseaction to our new circuits fuseaction.

Open up a browser and try running the application via <http://localhost/learnFB/lesson1> - if all goes well you should see 'Hello World'



Results of successful running of app

There's a chance that you won't see HelloWorld as you're expecting, instead a nasty error message presents itself;



Permissions error on the parsed folder.

If you see an error that refers to 'an error during wrote of parsed file' then you need to look into what permissions you have assigned to the parsed folder beneath your site. When a request is made for a fuseaction the core files actually parse the fusebox.xml.cfm and circuit.xml.cfm files and create a parsed file that is output to the 'parsed' folder, in this case you need to make sure that ColdFusion has permission to write to the parsed folder. By default on Windows CFMX is running as the SYSTEM user so change the permissions on the folder c:\inetpub\wwwroot\learnFB\parsed so that SYSTEM has Read, Write to the folder. Refresh your browser page and you should see the message, Hello World.

Notice when you used the address <http://localhost/LearnFB/lesson1> the fusebox sent you to your defaultfuseaction (as we set in fusebox.xml.cfm) without you have to explicitly specify the circuit alias and fuseaction. You can get to the same screen by typing the full address into your browser, <http://localhost/learnFB/lesson1/index.cfm?fuseaction=learnFB.helloworld> to see the same result.

Hello World!!!

Debugging Information
ColdFusion Server Developer 6,1,0,82055
Template /learnFB/index.cfm
Time Stamp 23-Aug-04 03:15 PM
Locale English (UK)
User Agent Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7) Gecko/20040803 Firefox/0.8
Remote IP 127.0.0.1
Host Name 127.0.0.1

Execution Time

Total Time	Avg Time	Count	Template
200 ms	200 ms	1	C:\inetpub\wwwroot\learnFB\index.cfm
190 ms	190 ms	1	C:\inetpub\wwwroot\learnFB\fusebox40\runtime.cfm.cfm
120 ms	120 ms	1	C:\inetpub\wwwroot\learnFB\fusebox40\loader.cfm.cfm
10 ms	10 ms	1	C:\inetpub\wwwroot\learnFB\fusebox40\parser.cfm.cfm
0 ms	0 ms	1	C:\inetpub\wwwroot\learnFB\Application.cfm
0 ms	0 ms	1	C:\inetpub\wwwroot\learnFB\dsp_helloworld.cfm
0 ms	0 ms	1	C:\inetpub\wwwroot\learnFB\fusebox40\transformer.cfm.cfm
0 ms	0 ms	1	C:\inetpub\wwwroot\learnFB\parsed_parsed\learnfb.helloworld.cfm
10 ms			STARTUP, PARSING, COMPILING, LOADING, & SHUTDOWN
210 ms			TOTAL EXECUTION TIME

red = over 250 ms average execution time

Explicitly requesting the Learnfb.helloworld circuit/fuseaction

Fusebox routes all requests through the index.cfm file – just as in the electricity wiring in your home, the fusebox handles all the electricity circuits in your home. This has its own advantages and disadvantages. One of the main advantages is we can easily lock down our application so that index.cfm is the only file that is allowed to execute.

We can do this by using the Application.cfm file in ColdFusion that executes before any other file in every request and making sure that the requested filename is index.cfm – if it's not we'll redirect them to index.cfm

```
<cfif right(cgi.script_name, Len("index.cfm")) NEQ
"index.cfm" AND right(cgi.script_name, 3) NEQ "cfc">
    <cflocation url="index.cfm" addtoken="no">
</cfif>
```

You can test this works by trying to access http://localhost/learnFB/lesson1/dsp_helloworld.cfm directly, you'll notice the URL change to index.cfm showing you that the code in Application.cfm has executed and redirected you to the fusebox.

So there, you've created your very first Fusebox4 application and protected the fusebox so all requests go through index.cfm – in the next lesson we'll add a second fuseaction and pass data between the two fuseaction.