



# Custom Tags, UDFs & CFCs

*Shlomy Gantz*  
*President, BlueBrick*

# About me

```
<CFSET CurrentTitle = "President, BlueBrick Inc.">
```

```
<CFSET experience_YY = 15 >
```

```
<CFSET experience_CF = 10>
```

```
<CFSET aTitles = arrayNew(1)>
```

```
<CFSET aTitles[1] = "Adobe Certified Instructor">
```

```
<CFSET aTitles[2] = "Adobe Community Expert">
```

```
<CFSET aTitles[3] = "Manager, NYFLEX user group">
```

```
<CFSET aTitles[4] = "Speaker, CFUNITED, Max..">
```

```
<CFSET aTitles[5] = "Author, CF Developer's  
Handbook, CFDJ">
```

```
<CFSET Mom = "Very Proud">
```

# Agenda

- ✓ <CFINCLUDE>
- ✓ Custom tags / <CFMODULE> / CFX
- ✓ UDFs
- ✓ CFCs
- ✓ Q & A

# CFINCLUDE

- ✓ Most basic form of re-use
- ✓ Only one attribute:
  - Template = “[path to file]”
- ✓ Any text file type can be included
- ✓ No protected scope

# Custom Tags

# Custom Tags

- ✓ Simply a .CFM template
- ✓ Use *CF\_filename* Syntax
- ✓ CF looks for files at:
  - Same directory as calling template
  - *C:\CFUSIONMX7\CustomTags* directory
  - *C:\CFUSIONMX7\CustomTags* sub directories
  - *Additional paths set in Administrator*

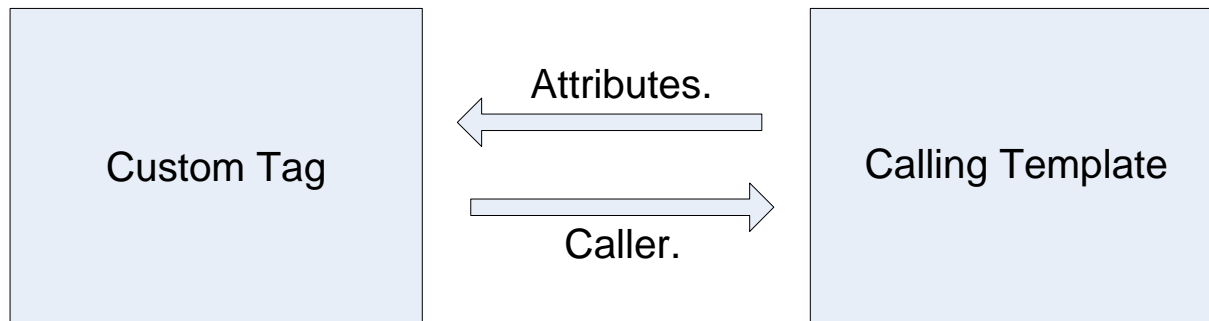


# Custom Tags

- ✓ Thousands of tags can be downloaded from the Developer's Exchange
- ✓ <http://www.adobe.com/cfusion/exchange/>
- ✓ Simply place the .cfm file in the *CustomTags* Directory

# Custom Tags – Creating

- ✓ Attributes scope
- ✓ Caller Scope



# CFMODULE

# CFMODULE

- ✓ Sometimes you have no access to the *CustomTags* Directory

```
<cfmodule
    template = "path"
    name = "tag_name"
    attribute_name1 = "valuea"
    attribute_name2 = "valueb"
    ...>
```

# CFMODULE

- ✓ Using the Template attribute you can refer to another .cfm template

```
<cfmodule template = "../ct/header.cfm">
```

=

```
C:\inetpub\wwwroot\cfunited07\ct\header.cfm
```

- ✓ Use a relative path from the calling template
- ✓ Use an absolute path from a ColdFusion mapping

# CFMODULE

- ✓ Using the Name attribute you can refer to a custom tag under *CustomTags*

```
<cfmodule template = "myct.header">
```

=

```
C:\CFUSIONMX7\CustomTags\myct\header.cfm
```

- Directory names are delimited with a period
- .CFM file extension is omitted

# Custom Tags (CFX)

- ✓ Custom Tags can also be developed in:
  - C++
  - Java
  
- ✓ CFX tags are installed via the CF administrator

# Custom Tags - Advanced

- ✓ Nested Tags
- ✓ Attributes / Caller scope

# User Defined Functions

# User Defined Functions

- ✓ User Defined Functions (UDFs) are simply functions added by developers
  
- ✓ You can create UDFs by:
  - Using a `<CFSCRIPT>` block
  - Using the `<CFFUNCTION>`

# UDFs - Creating

## ✓ <CFFUNCTION> Example:

```
<CFFUNCTION name="SalaryEnhancer">  
    <CFARGUMENT name="OrigSalary">  
    <CFRETURN arguments.OrigSalary*1.5>  
</CFFUNCTION>
```

```
<CFOUTPUT>#SalaryEnhancer(75000)#</CFOUTPUT>
```

# UDFs - Creating

✓ <CFSCRIPT> Example:

```
<CFSCRIPT>
function SalaryEnhancer(OrigSalary) {
    return arguments.OrigSalary*1.5;
}
</CFSCRIPT>
```

```
<CFOUTPUT>#SalaryEnhancer(75000)#</CFOUTPUT>
```

# UDFs - Creating

- ✓ UDFs accept arguments
- ✓ UDFs return one value
- ✓ UDFs can accept and return complex variables (arrays, structures, queries..)

# User Defined Functions

## ✓ CFFUNCTION TAG

```
<cffunction  
    name = "methodName"  
    returnType = "dataType"  
    output = "yes/no"  
...>
```

# User Defined Functions

## ✓ CFARGUMENT

```
<cfargument  
    name="ArgumentName"  
    type="data type"  
    required="Yes/No"  
    default="default value"  
...>
```

# User Defined Functions

✓ CFRETURN

<CFRETURN Expression>

# User Defined Functions

- ✓ Use the *var* keyword to create private variables

```
<CFSET var newSalary = 50000>
```

# UDFs – Pros & Cons

## ✓ Pros

- Explicitly return values
- Better Performance
- Easier maintenance

## ✓ Cons

- Have to be included on the page

# Components

# Components

- ✓ Introduced in ColdFusion MX 6
- ✓ Group UDFs together to create a ColdFusion Component (CFC)
- ✓ Can be secured as well as invoked remotely

# Components

- ✓ Advanced OO Development
  - Inheritance
  - Encapsulation
  - Persistence
  - Self-Documenting
  
- ✓ Components can have methods (UDFs) as well as properties

# Components - Creating

- ✓ **Static Components**
  - 1 or more UDFs
  - In a <CFCOMPONENT> tag block
  - Saved with a .cfc file extension
  
- ✓ **The name of the component is the name of the file**

# Components - Creating

## *SalaryManager.cfc*

```
<CFCOMPONENT>  
<CFFUNCTION name="SalaryEnhancer">  
    <CFARGUMENT name="OrigSalary">  
    <CFRETURN arguments.OrigSalary*1.5>  
</CFFUNCTION>  
</CFCOMPONENT>
```

# Components - Invoking

- ✓ Call component methods (UDFs)
  - `<CFINVOKE>` tag
  - `<CFINVOKE>` tag with `<CFINVOKEARGUMENT>` tags
  - `<CFOBJECT>/CreateObject()` with dot notation

# Components - Invoking

```
<CFINVOKE
```

```
    component="SalaryManager"  
    method="SalaryEnhancer"  
    returnVariable="newSalary"  
    origSalary="70000"  
  
/>
```

```
<CFOUTPUT>#newSalary#</CFOUTPUT>
```

# Components - Invoking

```
<CFINVOKE
    component="SalaryManager"
    method="SalaryEnhancer"
    returnVariable="newSalary">
    <CFINVOKEARGUMENT
        name="origSalary" value="70000">
    </CFINVOKE>

    <CFOUTPUT>#newSalary#</CFOUTPUT>
```

# Components - Invoking

```
<CFOBJECT
```

```
    component = "SalaryManager"  
    name = "mySalaryManager" >
```

```
<CFOUTPUT>
```

```
#mySalaryManager.SalaryEnhancer(70000)#
```

```
</CFOUTPUT>
```

# Components - Invoking

- ✓ You should store CFCs in their own directory.
- ✓ Use dot notation to specify the location relative to the webroot

```
<cfinvoke  
  component="cfunited07.components.SalaryManager"  
  method="SalaryEnhancer"  
  returnvalue="newSalary" />
```

C:\inetpub\wwwroot\cfunited07\components\SalaryManager.cfc

- ✓ The component's folder is called "package"

# Components – Addl Features

- ✓ Self Documentation
  - ✓ Web Services / WSDL
  - ✓ Dreamweaver Integration
  - ✓ . . . And much more
- 
- Go see Raymond Camden's 3 hour CFC extravaganza

# Additional Tags to look at...

✓ <CFOBJECT>

✓ <CFIMPORT>

# Q & A

[shlomy@bluebrick.com](mailto:shlomy@bluebrick.com)

<http://www.shlomygantz.com>