



## What is ColdFiSH?

ColdFish is a syntax highlighter for use within ColdFusion applications. It is fast, efficient, easy to use, and allows for easy customization.

Support and other licenses available if needed.

## Contributors

If you're interested in contributing, just send me an email at [jason@delmore.info](mailto:jason@delmore.info).

## Installation

- 1 - Move the com folder to your webroot
- 2 – *(optional – do this if you want to use the tag syntax)* Move xl.cfm and xlparam.cfm from the CustomTags folder into the ColdFusion CustomTags folder (if you want to use the CustomTag)
- 3 – *(optional – do this if you want to see some examples run)* Move the examples folder into your webroot

## License

Copyright 2008 Jason Delmore  
All rights reserved.  
[jason@delmore.info](mailto:jason@delmore.info)

This program is free software: you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.  
You should have received a copy of the GNU Lesser General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

# Documentation

## Usage

Check out the examples file.

- [example.cfm](#) – Example code

## Component Docs

coldfish.com.jasondelmore.coldfish.coldfish

# Component coldfish

hierarchy:

path:

properties:

methods:

[WEB-INF.cftags.component](#)

coldfish.com.jasondelmore.coldfish.coldfish

/Library/WebServer/Documents/coldfish/com/jasondelmore/coldfish/coldfish.cfc

[bufferAppend\\*](#), [endActionscript\\*](#), [endBind\\*](#), [endCFSET\\*](#), [endComment\\*](#), [endHighlight\\*](#), [endXMLTag\\*](#), [endOneLineComment\\*](#), [endScript\\*](#), [endTag\\*](#), [endValue\\*](#), [formatFile](#), [formatLine\\*](#), [formatString](#), [getStyle](#), [getStyles](#), [init](#), [initializeColors\\*](#), [initializeKeywordMap\\*](#), [initializeVariables\\*](#), [keywordsearch\\*](#), [regionMatches\\*](#), [setInitialParser](#), [setStyle](#), [startActionscript\\*](#), [startBind\\*](#), [startCFSET\\*](#), [startComment\\*](#), [startHighlight\\*](#), [startXMLTag\\*](#), [startOneLineComment\\*](#), [startScript\\*](#), [startTag\\*](#), [startValue\\*](#), [substring\\*](#)

\* - private method

### formatFile

`public formatFile ( string filePath )`

This function accepts a file path, reads in the file and formats it into syntax highlighted HTML.

Parameters:

**filePath:** string, optional, filePath

### formatString

`public formatString ( string code )`

This function accepts a block of code and formats it into syntax highlighted HTML.

Parameters:

**code:** string, optional, code

### getStyle

`public getStyle ( string element )`

This function can be used to get the style used in conjunction with a type of language element.

Parameters:

**element:** string, optional, element

### getStyles

`public getStyles ( )`

This function returns all of the styles used for each type of language element.

### init

`public init ( )`

This function initializes all of the variables needed for the component.

### setInitialParser

`public setInitialParser ( )`

You can set the initial parser state with this. This is helpful for scripts that make initial parsing impossible (ie. Script, Actionscript)

### setStyle

`public setStyle ( string element, string style )`

This function can be used to set the style used in conjunction with a type of language element. The value submitted should be a valid CSS black; (i.e. 'color:black;background-color:yellow;')

Parameters:

**element:** string, optional, element

**style:** string, optional, style