

# New Caching Features in ColdFusion 9

a hands-on session @ BFusion 2009  
October 24, Bloomington, Indiana



**Instructor:**

Aaron S. West

[a.west@me.com](mailto:a.west@me.com)

<http://www.aaronwest.net>

These lab instructions and all code were written by Aaron S. West. Last revised: October 22, 2009

<b>Introduction</b>	<b>3</b>
Lab Description	3
Instructor Assumptions	3
Software Prerequisites - IMPORTANT, please read!	3
Important Note About the Walkthrough Exercises	4
<b>Walkthrough 1 - Cache a full page on both the client and server</b>	<b>4</b>
<b>Walkthrough 2 - Cache a page fragment on both the client and server</b>	<b>5</b>
<b>Walkthrough 3 - Cache a page fragment and display cached metadata</b>	<b>6</b>
<b>Walkthrough 4 - Clear a cached page fragment using &lt;cfcache action="flush"&gt;</b>	<b>6</b>
<b>Walkthrough 5 - &lt;cfcache&gt; and the difference between timespan and idletime</b>	<b>7</b>
<b>Walkthrough 6 - Using cachePut() and cacheGet()</b>	<b>8</b>
<b>Walkthrough 7 - Using cacheRemove()</b>	<b>9</b>
<b>Walkthrough 8 - Examining cache hits and cache misses with the use of cacheGetMetadata()</b>	<b>10</b>
<b>Walkthrough 9 - Dependent Caching in ColdFusion 9</b>	<b>11</b>
<b>Walkthrough 10 - Session-specific Object caching</b>	<b>12</b>
<b>Walkthrough 11 - Display info about the Object cache and Template cache</b>	<b>13</b>
<b>Walkthrough 12 - Removing all items in Object and Template cache</b>	<b>14</b>
<b>Walkthrough 13 - Using cacheGetProperties()</b>	<b>16</b>
<b>Walkthrough 14 - Using cacheSetProperties()</b>	<b>16</b>

# Introduction

This 90 minute hands-on session was written especially for BFusion 2009. The purpose of the lab is to expose students to the new caching features in ColdFusion 9 by walking through several hands-on exercises. Welcome to the lab, we hope you enjoy it!

## Lab Description

Learn how the new caching features in ColdFusion 9 work. From changes to the <cfcache> tag to brand new functions, you will gain a solid understanding of caching concepts through many hands-on exercises / walkthroughs.

## Instructor Assumptions

The following assumptions have been made of all lab attendees by the instructor.

- you are using a Windows or Mac computer
- you have ALL the software prerequisites (listed below) installed and functioning properly (VERY important)
- you are an intermediate (or above) developer with a good understanding of CFML

## Software Prerequisites - IMPORTANT, please read!

This lab requires you have certain software already installed and functioning properly on your Windows or Mac computer. Please review the following prerequisites *before* you show up for the lab. We will not have time to ensure your computer is working properly.

### 1. *Adobe ColdFusion 9*

ColdFusion 9 was released on October 4, 2009. You must install ColdFusion 9 on your laptop before arriving at the lab. I recommend attendees install ColdFusion using the multiserver option at install time. This will allow you to install multiple instances of ColdFusion should you ever want to do so. During the installation you will be asked to choose an external web server or the built-in web server. You may choose either but I will be working with the built-in web server for simplicity purposes.

<http://www.adobe.com/go/coldfusion>

### 2. *Adobe ColdFusion Builder*

ColdFusion Builder will be used to create and edit all CFML caching walkthroughs. You may download and install the standalone version of ColdFusion Builder or the Eclipse-

based plugin. At the time of this writing ColdFusion Builder was in public beta and available for download at:

<http://labs.adobe.com/technologies/coldfusionbuilder/>

3. *Lab / exercise files. All caching walkthrough code is available to attendees prior to coming to the lab. Hopefully you were e-mailed with a link to download this PDF as well as the lab slide deck and all walkthrough code. If not, please grab one of the thumb drives from me before we get started.*

### **Important Note About the Walkthrough Exercises**

The walkthrough exercises that make up the rest of the content of this lab are self-contained and may be reviewed individually. During the lab we will start with Walkthrough 1 and progress to the end. Given our 90 minute time limit we may not finish all the lab content. However, the instructions have been written so you can complete all walkthroughs on your own at a later date.

## **Walkthrough 1 - Cache a full page on both the client and server**

---

In this walkthrough we will create a new ColdFusion template that caches an entire Web page on both the client and server.

1. Open ColdFusion Builder standalone or the ColdFusion Builder perspective if you installed the Eclipse-based plugin
2. Close any open projects
3. Create a new ColdFusion Builder project
  1. File - New - ColdFusion Project
  2. Project name: CachingLab
  3. Project location: <your\_web\_root>/CachingLab
4. Copy the lab assets into your project
  1. Locate the “walkthroughs” folder distributed with this PDF or provided separately. If you don’t have this folder please ask Aaron for a thumb drive.
  2. Copy the “walkthroughs” directory from Windows Explorer or Mac’s Finder and paste the directory into the root of your CachingLab project.
5. Each walkthrough file is a completed version of the specific exercise
6. Create a new ColdFusion page in your CachingLab project
  1. File - New - ColdFusion Page
  2. Name: wt-1 (you do not have to type “.cfm” after the name)
  3. Press Finish
7. Type the following code into wt-1.cfm

```
<cfcache action="cache" timespan="#CreateTimeSpan(0,0,0,15)#">

<cfoutput>
The current date/time is: #Now()#
</cfoutput>
```

8. Run the ColdFusion page using the integrated Web browser in ColdFusion Builder, or your external Web browser.
9. Notice how the entire page is cached for 5 seconds. The date/time value displayed will be cached for 5 seconds every time the page is executed.

## Walkthrough 2 - Cache a page fragment on both the client and server

---

In this walkthrough we will create a new ColdFusion template that caches a page fragment on both the client and server.

1. Create a new ColdFusion page called wt-2.cfm
2. Type the following code into the page

```
<cfcache action="cache" timespan="#CreateTimeSpan(0,0,0,5)#">
    <cfoutput>This date/time IS cached: #Now()#<br /></cfoutput>
</cfcache>

<cfoutput>This date/time is not cached: #Now()#</cfoutput>
```

3. Run the ColdFusion page in a browser
4. Notice how the first date/time value gets cached and only changes every 5 seconds (as you continually press refresh)
5. The second date/time value changes with every page refresh

## Walkthrough 3 - Cache a page fragment and display cached metadata

---

In this walkthrough we will create a new ColdFusion template that caches a page fragment and displays the little bit of metadata ColdFusion 9 makes available with Template caches.

1. Create a new ColdFusion page called wt-3.cfm
2. Type the following code into the page

```
<cfcache action="cache" timespan="#CreateTimeSpan(0,0,0,10)#">
    <cfoutput>This date/time IS cached: #Now()#<br /></cfoutput>
</cfcache>

<cfoutput>This date/time is not cached: #Now()#</cfoutput>

<cfdump var="#GetAllTemplateCacheIDs()#">
```

3. Run the ColdFusion page in a browser
4. Notice the additional metadata displayed at the bottom of the page using the undocumented function GetAllTemplateCacheIDs()

## Walkthrough 4 - Clear a cached page fragment using <cfcache action="flush">

---

In this walkthrough we demonstrate the use of the "flush" action of the <cfcache> tag. We'll cache a page fragment and then use a hyperlink to instruct the page to clear just the Template cache for the current page.

1. Create a new ColdFusion page called wt-4.cfm
2. Type the following code into the page

```
<cfif isDefined("URL.flush")>
    <cfcache action="flush" expireurl="*wt-4.cfm*">
</cfif>
```

```
<cfcache action="cache" timespan="#CreateTimeSpan(0,0,0,45)#">
    <cfoutput>This date/time IS cached: #Now()#<br /></cfoutput>
</cfcache>

<cfoutput>This date/time is not cached: #Now()#</cfoutput>

<cdump var="#GetAllTemplateCacheIDs()#">
<br />
<a href="wt-4.cfm?flush=1">Flush Cache</a>
```

3. Run the ColdFusion page in a browser
4. Refresh the page a few times and notice the first date/time value does not change. The second date/time value will change with each refresh of the page.
5. Press the "Flush Cache" link and see the result of using action="flush"

## Walkthrough 5 - <cfcache> and the difference between timespan and idletime

---

This walkthrough demonstrates the difference between the timespan and idletime attributes of the <cfcache> tag.

1. Create a new ColdFusion page called wt-5.cfm
2. Type the following code into the page

```
<cfcache action="cache" timespan="#CreateTimeSpan(0,0,0,30)#"
idletime="#CreateTimeSpan(0,0,0,5)#">

<cfoutput>
The current date/time is: #Now()#
</cfoutput>
```

3. Run the ColdFusion page in a browser
4. The timespan attribute can be expressed as a decimal (in days) or an actual timespan value. This value represents how long an item will be cached without respect to the number of cache hits

5. The idleTime attribute is also expressed as a decimal or a timespan value. This value indicates how long a cache is kept around without getting any cache hits

## Walkthrough 6 - Using cachePut() and cacheGet()

---

This walkthrough demonstrates use of the Object cache in ColdFusion through two functions, cachePut() and cacheGet().

1. Create a new ColdFusion page called wt-6.cfm
2. Type the following code into the page

```
<cfset cachedData = cacheGet("wt-6-cache")>

<cfif isNull(cachedData)>
    Cache doesn't exist, so create it.<br />
    <cfset sleep(1000)>
    <cfset cachedData = "This date/time IS cached: #Now()#<br />
">
    <cfoutput>#cachedData#</cfoutput>
    <cfset cachePut("wt-6-cache", cachedData,
CreateTimeSpan(0,0,0,15))>
</cfif>

<cfoutput>This date/time is not cached: #Now()#</cfoutput>

<h4>Cached data</h4>
<cfdump var="#cachedData#">

<h4>Cached metadata</h4>
<cfdump var="#cacheGetMetadata('wt-6-cache')#">
```

3. Run the ColdFusion page in a browser
4. Notice how the first date/time value gets cached and only changes every 10 seconds. The second date/time value changes every page refresh.
5. This example is very similar to wt-3 except here we are using the new caching functions instead of the cfcache tag.



## Walkthrough 7 - Using cacheRemove()

---

This walkthrough shows how to remove an item from the Object cache using the new `cacheRemove()` function in ColdFusion 9.

1. Create a new ColdFusion page called `wt-7.cfm`
2. Type the following code into the page

```
<cfif isDefined("URL.flush")>
    <cfset cacheRemove("wt-7-cache")>
</cfif>

<cfset cachedData = cacheGet("wt-7-cache")>

<cfif isNull(cachedData)>
    Cache doesn't exist, so create it.<br />
    <cfset sleep(1000)>
    <cfset cachedData = "This date/time IS cached: #Now()#<br />
">
    <cfoutput>#cachedData#</cfoutput>
    <cfset cachePut("wt-7-cache", cachedData,
CreateTimeSpan(0,0,0,15))>
</cfif>

<cfoutput>This date/time is not cached: #Now()#</cfoutput>

<h4>Cached data</h4>
<cfdump var="#cachedData#">

<h4>Cached metadata</h4>
<cfdump var="#cacheGetMetadata('wt-7-cache')#">
<br />
<a href="wt-7.cfm?flush=1">Flush Cache</a>
```

3. Run the ColdFusion page in a browser
4. Refresh the template a few times and notice how the first date/time is cached and doesn't change. The second date/time changes with every page refresh.
5. Press the "Flush Cache" link and see the result of using `cacheRemove()`
6. This example is very similar to `wt-4` except here we are using the new `cacheRemove()` function instead of the `cfcache` tag.

## Walkthrough 8 - Examining cache hits and cache misses with the use of cacheGetMetadata()

---

In this walkthrough we will examine cache hits and cache misses by using the cacheGetMetadata() function. This function retrieves metadata for any Object level cached.

1. Create a new ColdFusion page called wt-8.cfm
2. Type the following code into the page

```
<cfset cacheItem1 = cacheGet('wt-8-item1')>
<cfset cacheItem2 = cacheGet('wt-8-item2')>
<cfset cacheItem3 = cacheGet('wt-8-item3')>

<cfif isNull(cacheItem1)>
    <cfset cachePut('wt-8-item1', 'This is the cache for item
1.')>
</cfif>

<!--- Section A. Uncomment for step 2. --->
<!---><cfif isNull(cacheItem2)>
    <cfset cachePut('wt-8-item2', 'This is the cache for item
2.')>
</cfif>--->

<!--- Section B. Uncomment for step 3. --->
<!---><cfset cachePut('wt-8-item2', 'Cache item 2 has been
updated!')>--->

Cache Item 1:
<cfdump var="#cacheGetMetadata('wt-8-item1')#">

Cache Item 2:
<cfdump var="#cacheGetMetadata('wt-8-item2')#">

Cache Item 3:
<cfdump var="#cacheGetMetadata('wt-8-item3')#">
```

3. Run the template as-is. On first run the hitcount for cache item 1 will be 0. Subsequent runs will increment the hitcount for cache item 1.
4. Uncomment section A and run the template. Cache item 2 will now be created and its hitcount will start at 0. Subsequent runs will increment the hitcount for cache item 2 as well as cache item 1.
5. Uncomment section B and run the template. Notice how the hitcount for cache item 2 is always 0 and the lastupdated property of cache item 2 changes with every run.
6. Comment out section B and run the template. Notice how the hitcount begins to increment with every run of the template. Also notice how the lastupdated property of the cache item 2 shows the last time the cachePut() function ran and the lasthit property changes with every run.

## Walkthrough 9 - Dependent Caching in ColdFusion 9

---

In this walkthrough we will look at the use of dependent caching. This is accomplished by using the dependsOn attribute of the <cfcache> tag.

1. Create a new ColdFusion page called wt-9.cfm
2. Type the following code into the page

```
<cfquery name="getOrders" datasource="cfartgallery">
    SELECT ORDERID
    FROM ORDERS
    ORDER BY ORDERDATE DESC
</cfquery>

<cfcache action="cache" id="wt-9-cache"
dependsOn="getOrders.RecordCount">
    <cfoutput>Some big heavy process that does something with
new orders...
    <cfset sleep(1000)>
    <br />
    Number of orders #getOrders.RecordCount# on #Now()#
    </cfoutput>
</cfcache>
```

3. Create a new ColdFusion page called wt-9-update.cfm
4. Type the following code into the page

```
<cfquery name="addOrder" datasource="cfartgallery">
```

```
INSERT INTO ORDERS (TAX, TOTAL, ORDERDATE, ORDERSTATUSID,
CUSTOMERFIRSTNAME, CUSTOMERLASTNAME, ADDRESS, CITY, STATE,
POSTALCODE, PHONE)
VALUES (10, 100, #CreateODBCDateTime(Now())#, 1, 'Aaron',
'West', '2009 ColdFusion Ave', 'Nashville', 'TN', '37203',
'415-555-9836')
</cfquery>

<!---<cfquery name="addOrder" datasource="cfartgallery">
    delete
    from ORDERS
    where ORDERID = 28
</cfquery>--->
```

5. Save both of the wt-9 pages
6. Run wt-9.cfm to cache the page fragment based on the number of current records
7. Run wt-9-update.cfm to add a record to the database
8. Rerun wt-9.cfm to see the cache refresh
9. Uncomment the delete query in wt-9-update.cfm, save the file and run it again to remove the row you previously inserted. NOTE: Your ORDERID value may be different than mine.
10. Rerun wt-9.cfm and see how the date/time value displayed is the same as in step 6 above.

## Walkthrough 10 - Session-specific Object caching

---

In this walkthrough we will illustrate the use of session-specific Object caching. Sessions must be enabled in Application.cfc or Application.cfm. If you are using my code examples, Application.cfc is included.

1. Create a new ColdFusion page called wt-10.cfm
2. Type the following code into the page

```
<cfset shoppingCartCache = session.URLToken & "ShoppingCart">

<cfset userShoppingCart = cacheGet(shoppingCartCache)>

<cfif isNull(userShoppingCart)>
```

```
<cfset userShoppingCart = [{item = "Pillow-  
#Right(CreateUUID(), 4)#", price = "29.99"}, {item = "Blanket-  
#Right(CreateUUID(), 4)#", price = "49.99"}]>  
<cfset cachePut(shoppingCartCache, userShoppingCart)>  
</cfif>
```

Session specific cache value:

```
<cfdump var="#userShoppingCart#">
```

3. Run the ColdFusion page in your default browser and notice the information displayed in the cfdump
4. Run the ColdFusion page in a second browser and notice the different information (UUIDs) in the displayed cfdump

## Walkthrough 11 - Display info about the Object cache and Template cache

---

This walkthrough shows how to display information about both the Object cache and the Template cache. This is accomplished via the documented `cacheGetAllIds()` function for Object cache and the undocumented `getAllTemplateCacheIds()` function for Template cache.

1. Create a new ColdFusion page called `wt-11.cfm`
2. Type the following code into the page

```
<cfset cachePut("wt-11-cache", "This will become a cached  
string.", CreateTimeSpan(0,0,0,5))>  
  
<h4>Object cache</h4>  
<cfloop index="cache" array="#cacheGetAllIds()#">  
    <cfdump var="#cacheGetMetadata(cache)#" label="Cache  
Metadata for #cache#">  
</cfloop>  
  
<cfcache action="cache" timespan="#CreateTimeSpan(0,0,0,15)#">  
    <cfoutput>This date/time IS cached: #Now()#<br /></cfoutput>  
</cfcache>
```

```
<cfoutput>This date/time is not cached: #Now()#</cfoutput>

<h4>Template cache</h4>
<cfdump var="#GetAllTemplateCacheIDs()#">
```

3. Run the ColdFusion page in a browser
4. Notice how there's information displayed about both the Object cache and Template cache items we created

## Walkthrough 12 - Removing all items in Object and Template cache

---

In this walkthrough we will add an item to the Object cache and the Template cache and then use the two functions from the previous walkthrough to remove all cached items.

1. Create a new ColdFusion page called wt-12.cfm
2. Type the following code into the page

```
<cfset cachePut("wt-11-cache", "This will become a cached
string.", CreateTimeSpan(0,0,2,0))>

<h4>Object cache</h4>
<cfloop index="cache" array="#cacheGetAllIds()#">
    <cfdump var="#cacheGetMetadata(cache)#" label="Cache
Metadata for #cache#">
</cfloop>

<cfcache action="cache" timespan="#CreateTimeSpan(0,0,2,0)#">
    <cfoutput>This date/time IS cached: #Now()#<br /></
cfoutput>
</cfcache>

<cfoutput>This date/time is not cached: #Now()#</cfoutput>

<h4>Template cache</h4>
<cfdump var="#GetAllTemplateCacheIDs()#">
```

3. Save wt-12.cfm

4. Create a new ColdFusion page called wt-12a.cfm
5. Type the following code into the page

```
<cfloop index="cache" array="#cacheGetAllIds()#">
    <cfset cacheRemove(cache)>
</cfloop>

<cfcache action="flush">
```

Done... now dumping any available Object and Template caches.

```
<h4>Object cache</h4>
<cfloop index="cache" array="#cacheGetAllIds()#">
    <cfdump var="#cacheGetMetadata(cache)#" label="Cache
Metadata for #cache#">
</cfloop>

<h4>Template cache</h4>
<cfdump var="#GetAllTemplateCacheIds()#">
```

6. Run wt-12.cfm in your browser and refresh it a few times
7. Notice how the first date/time (from the Template cache) does not change.
8. The second date/time changes with every page refresh.
9. We now have data cached in the Object cache and a page fragment cached in the Template cache
10. Run wt-12a.cfm and notice how all Object cache and Template cache is removed
11. If you want to really test this out return to wt-3.cfm and change the cached timespan to 60 seconds. Now open wt-4.cfm and change the cached timespan to 60 seconds.
12. Run wt-3.cfm and wt-4.cfm in a browser
13. Next, run wt-12.cfm in a browser
14. You've now created several different Object and Template caches
15. Now run wt-12a.cfm in a browser and notice how all cache is cleared
16. To further check this out, execute steps 12-15 again and notice how the cached create date/time is new in all cached metadata.

## Walkthrough 13 - Using cacheGetProperties()

---

In this walkthrough we will take a look at the cache properties for both Object cache and Template cache using the cacheGetProperties() function.

1. Create a new ColdFusion page called wt-13.cfm
2. Type the following code into the page

```
<cfdump var="#cacheGetProperties()#">
```

3. Run the ColdFusion page in a browser
4. Take a look at the properties for the Object cache (shown first) and the Template cache

## Walkthrough 14 - Using cacheSetProperties()

---

This walkthrough demonstrates the use of cacheSetProperties() to change the values stored in either the Object cache or Template cache. This example will change the Object cache properties.

1. Create a new ColdFusion page called wt-14.cfm
2. Type the following code into the page

```
<cfset myProps = StructNew(>
<cfset myProps.diskStore = "your_absolute_directory_path">
<cfset myProps.diskpersistent = "true">
<cfset myProps.eternal = "false">
<cfset myProps.maxelementsinmemory = "5000">
<cfset myProps.maxelementsondisk = "100000">
<cfset myProps.memoryevictionpolicy = "FIFO">
<cfset myProps.objecttype = "Object">
<cfset myProps.overflowtodisk = "true">
<cfset myProps.timetoidleconds = "86400">
<cfset myProps.timetolivesecond = "86400">
```

Before:



```
<cfdump var="#cacheGetProperties("Object")#">
```

```
<cfset cacheSetProperties(myProps)>
```

After:

```
<cfdump var="#cacheGetProperties("Object")#">
```

```
<cfset cachePut("wt-14-cache", "This is a cached string.")>
```

3. Run the ColdFusion page in your browser and notice the difference between the before and after sections
4. Run the page again and notice how the before section now shows the same data as the after section proving the settings were updated on the ColdFusion server
5. Reverse these changes by restarting ColdFusion server or altering the properties of the myProps structure and re-running the page