Toward Better Error Handling

Detecting errors at several levels, via old and new ways

Charlie Arehart
Systemanage
Our Practice Makes You Perfect ™
carehart@systemanage.com

The Challenge

- Errors in your code are inevitable
  - Are your users seeing an error message?
  - What’s the message say?
  - Can they really do anything with the information?
  - Is there perhaps information they shouldn’t see?
  - Could you handle the error more effectively?
  - Are you even aware that the error’s happening?

Topics

- Seminar covers:
  - Administrator settings affecting error msgs
  - Old style error handling with CFERROR
  - Request error handling
  - Validation error handling
  - Next generation error handling
    - Application-level exception handling: CFERROR
    - Site-wide error handling
    - Code-level error handling: CFTRY

Who Am I?

- A programmer by trade, trainer at heart
  - With nearly 20 years IT experience (5 in CF)
  - Monthly contributor to ColdFusion Developer Journal; columnist in Java Developer Journal
  - Speaker at user groups, conferences worldwide; Seminar developer/presenter
  - Macromedia Certified Instructor & Developer
- Practical training and hi-gear consulting

Goals

- Explain some often missed features
  - And some frequently misunderstood
- Enable you to improve error handling
  - Provide better error msgs to users
  - Reduce security risks
  - Cause you to be notified upon errors, log them
- Solve problems not otherwise easily solved
- All done very easily
  - just a few minutes work

Four Possible Error Results

- Traditional CF error message
  - Plain black on white
- Message you format and control
  - Using site layout, but no CF code
- Message you format and control
  - And that can do any CF code (log, email)
- Handling the error in a way that hides it from the user
  - May not show any error at all (log, email)
CF Error Processing

That pesky CF error message:

- Message is great for developers!
  - Rather detailed error message
  - What line of code, in what template
  - Datasource name, generated SQL
  - Great for Debuging

Clearly, Allaire had developers in mind!

Studio tip: Goto line of code with Ctrl-G

The Good News

The Bad News

Message stinks for end users

- White on black not likely your site's color scheme!
- Where are your nav bars?
- What are they to do with the info?

The Security Risk

Is there any security risk in the information shown?

- What can a skilled hacked do knowing:
  - Physical path to templates
  - Datasource Name
  - SQL Statements showing table schema

The Impact

Do you even know the error occurred?

- Are you just hoping they'll report it?
- Will they really know what to do with info?

Even if they report it, who'll hear the complaints?

- The end user's boss, co-workers
- Your company's customer service dept?
- Maybe your CEO? Your boss?
- Clearly, you should know about it ASAP

The Solution

Many developers never think about this

- Or don’t know how to solve
- Options exist to limit, modify, or hide entirely the error message
  - Can arrange to be emailed on error
  - Or write errors to an app-specific log
- Some are new as of 4.5, not well known
  - Must pay attention to new releases
  - Don't rely on "common knowledge". Study!
Part I: Administrator Settings

- 3 changes can be made in CF Admin to affect default error msg
  - Can ease reporting of errors by users
  - Limit security risk of info provided
  - Later topics of site-wide error handling, CFERROR, CFTRY are important too
  - But should still address these settings

Defining “Admin email” Error Handling Address

- If “admin email” addr defined in CF admin, will be shown on default error msg

  Please inform the site administrator that this error has occurred (be sure to include the contents of this page in your message to the administrator).

  Hyperlink will open email to that addr
  - Have to hope user will follow the instructions, especially “include the contents”

How to and Notes

- In 5 Admin, set it using new “tools” menu
  - Then Logs and Statistics>Logging Settings
  - Input text prompt “Administrator e-mail”
  - In 4.5, use Logging>Settings to get prompt
- Notes:
  - This addr can be overridden with CFERROR
  - If not defined, line simply won’t be shown
  - Not suitable if no single user should be notified
  - Be sure to change if that person leaves!
  - Value also available in error.mailto variable

Msg Text Security Risk

- Should hide SQL and Datasource Name, and Template Path, in production
  - If not otherwise hiding error msg using CFERROR/CFTRY later
  - Concern is not just hackers but also other developers on same server
  - Can use template path to access code (CFFILE, CFCONTENT, etc.)
  - Unless you've implemented CF Basic/Adv Security
  - Could use DSN and schema to code SQL
  - Unless you've implemented DB security

“Show SQL”: How To

- Easy to stop (simple checkbox in CF Admin)
  - Problem is finding the place to set it
  - Under debugging options!
  - In CF 5 Admin, use “Server” tab
  - Then Debug Settings>Debug Options
  - In 4.5, Miscellaneous>Debugging
  - Uncheck “Show SQL and Datasource Name”
  - Will remove SQL statement and DSN from msg

“Show SQL”: Notes

- Curious behavior:
  - SQL and DSN will still show in error msg, regardless of that setting, if user is allowed to see Debug info
  - May explain why it’s set where it is
  - Also means there’s no reason not to turn it off if developers see debug info
  - Those who should see it will, those who shouldn’t won’t
"Hide Template Path": How to

- Also a simple checkbox in CF Admin
  - Also set under debugging options
  - Uncheck "Display the template path in error messages"
  - Last line of the message no longer indicates the template name and its path

"Hide Template Path": Notes

- Unlike "show SQL" option, once unchecked, is not shown
  - Not even to debugging users
  - Likely want it on in dev/test environments
- One other curiosity
  - Template name and path STILL SHOWN to any user regardless of setting
  - When the error is a compilation error
  - The specific sequence of files included or processed is:
    C:\inetpub\wwwroot\somedir\sometemplate.cfm

Part II: Old Style Error Handling with CFERROR

- CF has long had CFERROR tag
  - Typically placed in application.cfm
  - Declares template to be used to present errors
  - You control format, choosing what details to show
  - Can show site colors, nav bars
  - Two forms: one for request errors, one for validation errors (form processing)
- Problems:
  - Template can’t execute CF code!

CFERROR Type=”Request”

- In application.cfm, might code:
  `<cferror type="REQUEST" template="error_handler.cfm">`
- In error_handler.cfm, might code:
  `<table>...<!--- code for nav bar--->[<table>
  `<h2>An Error Has Occurred</h2>
  `<br>Details: #error.diagnostics#`
  `<a href="#error.mailto#">#error.mailto#</a>`

- Note available "error" variables
  - Do not need to use CFOUTPUT to show them
  - CANNOT use any CF tags, such as CFINCLUDE
  - Can set mail recipient with CFERROR MAILTO

More Attractive Example

CF Form Validation Errors

- May know that CF supports built-in server-side form validation
  - Use hidden fields with fieldname_val
  - As in username_required, age_integer
  - You optionally specify message to show
  - CF automatically generates validation error message displayed on form submission
Sample Form Validation

• Might have login page with code:
  <form action="login.cfm" method="post">
  Please login:<br>
  <table>
    <tr><td>Username: </td><td><input type="Text" name="username"></td></tr>
    <input type="Hidden" name="username_required" value="You must enter a username">
    <tr><td>Password: </td><td><input type="Password" name="password"></td></tr>
    <input type="Hidden" name="username_password" value="You must enter a password">
  </table>
  <input type="Submit" value="Login">
  </form>

Sample Form Validation

• CF shows, again, plain B&W message

Form Entries Incomplete or Invalid
One or more problems exist with the data you have entered.
• You must enter a username
Use the Back button on your web browser to return to the previous page and correct the listed problems.

CFERROR Type="Validation"

• Can improve on that with CFERROR
• In application.cfm, might add:
  <cferror type="VALIDATION" template="error_formval.cfm">
• In error_handler.cfm, might code:
  <h2>Error processing your submission</h2>
  Details: #error. invalidfields#
  Please email #error.mailto#
• Again, note "error" variables
  • Adds new "validationfields" variable & others
  • Same limits on use of CF tags

Problems Remain

• Couldn’t use CFINCLUDE for navbar
  • Could lead to trouble keeping updated
• Couldn’t fire off email to admin/developer
  • MAILTO attribute does NOT cause auto email!
  • Some used Javascript to fire off another CF template to send email
  • New approaches discussed later relieve that need

Problems Cont.

• Other problems due to not being allowed to execute CF tags:
  • In previous example, error.mailto was blank when displayed to user
  • Hadn’t been set at admin level, nor in CFERROR
  • Couldn’t use CFIF to show conditionally

Part III: Next Generation Error Handling

• CF 4.5 added improved error handling
  • Users no longer need suffer the boring default error message display
  • Many developers still don’t know about these!
• Can still use CFERROR, but new TYPE="exception"
  • Can use any CF tags in error handling template
  • Can now send email, log message, etc.
• Can also trap errors at site level
• Can also trap errors at code level
CFERROR Type="Exception"

- In application.cfm, might code:
  `<cferror type="Exception" template="error_handler2.cfm">`

- In error_handler2.cfm, might code:
  - CFINCLUDE of navbar in header file
  - Message indicating error has occurred and that admin has been notified
  - Send email to the admin/use developer
  - Can use same error variables as before

**Error Diagnostic Variables**

- Available Error Handling Variables:
  - Error.Diagnostics
  - Error.DateTime
  - Error.Template
  - Error.QueryString
  - Error.HTTPReferer
  - Error.Browser
  - Error.RemoteAddress
  - Error.MailTo
  - Available in both old and new style CFERROR

**Notes**

- Overrides old-style Type="request"
  - If both defined, TYPE="exception" runs
- Not all errors fire Type="exception"
  - Should keep TYPE="request" in case
- Can also use EXCEPTION="...", attribute to handle different exceptions
  - Can even create multiple CFERROR lines to handle different exceptions separately
  - If no EXCEPTION attr defined, catches "any"

**Capturing Var Scopes Prior to CF 5**

- Prior to CF5, could display/email/log all session vars using following code:
  `<cfif isDefined('session.cfid')>
  <cfoutput>
  CFQUERY to insert to DB
  Use CFLOG, as of CF 5, to write text log file
  In email or log, might capture current value of variables such as
  - Session, application, cookie, client, form, url, cgi, etc.
  - Could greatly assist in debugging
  - Could use CFDUMP Var="#session#" in CFS
  </cfoutput>
  </cfif>`
Capturing Var Scopes Prior to CF 5 (cont)

• Can use that approach for all but “server” scope variables
  • Note need to test for whether values exist and are simple enough to print
  • Showing structures within structures takes just a little more code
  • CFDUMP is such a nice addition
  • Handles this for us!
• Client vars can be captured with getclientvariableslist() function

Site-wide Error Handling

• If no CFERROR defined, or error not caught by defined EXCEPTION types
  • CF can be told to catch all errors site-wide
• Involves defining an error handling template just as in TYPE="exception"
  • Has the power to use all tags
    • In CF 5 Admin, use "Server" tab
    • Then Server Settings>Settings
    • In 4.5, Server>Settings
  • Text prompt "Site-wide Error Handler"

Site-wide Notes

• Can use same approach as CFERROR TYPE="exception"
  • Can use any CF tags
  • Note, however, that the filename specification in admin must be complete path, not web-relative
  • Can place this file anywhere on server
  • Be sure to test it. Filename not validated

Site-wide Notes (cont.)

• Be aware this impacts all users in all apps! Don’t use it just for “your” app
  • Only makes sense if a site-wide error handler is appropriate
  • Are all apps really related to each other?
    • Such that one handler is appropriate for all
  • Won’t be executed if error caught at lower-level
  • But then some errors will still get by CFERROR, even if both types are set
  • Again, this makes a good backup against traditional CF B&W message

Code-level Error Handling: TRY/CATCH

• Most modern languages support try/catch logic
  • Used to surround code that may cause an error
    • Database processing that may fail integrity checks
    • Or if database is unavailable
    • Attempting CFHTTP to remote site not responding
    • Calling COM or Java object, not responding
    • Any sort of code that might fail
  • Can catch and handle that specific error
Error Handling Hierarchy

CFTRY/CFCATCH
- These are implemented as nested tag pairs:
  - CFTRY
    - <!--- some code you want to monitor --->
    - <CFCATCH>
      - <!--- code to handle any error --->
    - </CFCATCH>
    - </CFTRY>
- Control transfers to CATCH if error

CFTRY/CFCATCH Example

```cfml
<CFTRY>
  <CFQUERY DATASOURCE="#request.dsn#" NAME="test">
    SELECT * FROM MyTable
  </CFQUERY>
  <CFCATCH>
    An Error has occurred while selecting records from MyTable. <p>
    The details of the error are: <br>
    <CFOUTPUT>&#CFCATCH.MESSAGE#<p>&#CFCATCH.DETAIL#</CFOUTPUT>
  </CFCATCH>
</CFTRY>
```

Control Flow
- If no error:
  - None of code in CFCATCH is executed
  - Processing continues after </CFTRY>
  - Kind of adds capability to check "return code" for CF processing

Control Flow (cont.)
- If error occurs:
  - Statements in CFCATCH executed
  - Normal CF error message NOT displayed
  - No CF error message written to CF logs
  - Processing continues after </CFTRY>
  - Unless you CFABORT within CFCATCH
- Can execute any CF tags, functions in CFCATCH
  - CFMAIL, CFLOG, CFQUERY to insert log, etc.
  - Previous "error" variables not available

CFCATCH Variables
- Instead, have several new variables
  - CFCatch.Type – exception type caught
  - CFCatch.Message – error message
  - CFCatch.Detail – detailed error msg
  - CFCatch.TagContext – tag stack
  - If enabled in Admin, in Debugging settings
  - A few other vars for specific error types
  - See documentation for more details
- Of course, have all the normal CF vars
CATCHing Multiple Errors

- Simple example catches any error
- Can instead test for several types of errors, with multiple CFCATCHes
  - Each having a different TYPE, such as:
    - Application
    - Database
    - Template
    - Security
    - Etc.
  - Can even declare a customtype
    - Useful with CFTHROW. See docs for more

CFTRY/CFCATCH Notes

- Don’t try to wrap entire program in CFTRY
- CFTRY won’t catch syntax/compile errors
  - Is a runtime error handler
- Do use it for unavoidable DB errors
- Think twice about using for avoidable DB errors, such as catching Insert duplicates
  - May be better to test for dup and avoid insert
- CFMX adds try/catch in CFSCRIPT

CFTHROW/CFRETHROW

- CFTHROW permits programmatically passing control to an error handler
  - May be throwing it within a CFTRY to associated CFCATCH
  - May be throwing it outside CFTRY, to be caught by CFERROR, site-wide handler, or shown as standard CF error message
- CFRETHROW used within CFCATCH to handle and then pass up to other handler in hierarchy above

Summary

- Admin settings:
  - Consider “administrator email address”
  - Consider “Show SQL andDatasource name” and “show template path” options
- Site-wide error handler
- CFERROR
  - Old types: request and validation
  - New type: exception (“monitor” deprecated)
- CFTRY/CFCATCH, CFTHROW

Bottom Line

- Your users should see more effective and interesting messages
  - With normal site appearance
  - Options to take normal site links
  - You as developer should be notified when errors occur!
    - Either by email
    - Or by log created when error occurs

Resources

- Still more than we could cover here!
- My 4-part series of articles in CFDJ
  - “Toward Better Error Handling”, 2001
  - Some more details not covered here
- All articles online at www.systemanage.com
- Admin settings covered in “Administering CF” Manual
- Tags in “CFML Lang Reference”, and “Developing CF Apps” manuals
Thank you!

- Questions now, or later?
- Contact: Charlie Arehart, Systemanage
  - carehart@systemanage.com
  - 301 604 8399
- Also more info on:
  - One day seminars, one day consulting, speaking at your user group, and more