FORM PROCESSING, USING COLDFUSION FOR DATABASE INTERACTION

Objectives

• Understand how to capture form output and test form variables for appropriate content
• Understand how DataSources are defined by the ColdFusion administrator.
• Understand how to access databases and retrieve data using ColdFusion.
• Understand how to update data stored in a database using ColdFusion.
Form processing

Data passed to ColdFusion scripts typically use POST method.
Retrieve value from form fields by using form namespace.

```html
<body>
<h1>Hello,
<cfoutput>#form.firstname# #form.lastname#</cfoutput>
</h1>
</body>

http://coldfusioncs.etsu.edu/pittares/demo/forms/formexample.htm
http://coldfusioncs.etsu.edu/pittares/demo/forms/form1test1.htm
```

GET parameters

Retrieve GET parameters (either generated by form or set manually) using URL namespace.

```html
<body>
<h1>Hello,
<cfoutput>#URL.firstname# #URL.lastname#</cfoutput>
</h1>
</body>

http://coldfusioncs.etsu.edu/pittares/demo/forms/formexample2.htm
```

Technically, both Form and URL qualifiers are not required. If a variable does not exist in the variables namespace, other namespaces will be searched. Using full qualifiers improves clarity of code and avoids inadvertent name collision.
Beware

If form value is not set by user, this will trigger an error:

http://coldfusioncs.etsu.edu/pittares/demo/forms/processform1test1.cfm

May need to turn off friendly error message in IE to see the above error screen (under "Advanced", "Browsing").

Verifying form data

`IsDefined("varname")` returns whether `varname` is defined in available namespaces.

Notice `varname` is in quotes.

```cstext
<cfif IsDefined("form.paymentmethod")>
<h1>You selected the payment method: <cfoutput>#form.paymentmethod#</cfoutput></h1>
<cfelse>
<h1>You can't get things without paying!</h1>
</cfif>
```

Test value of variable using EQUALS.

http://coldfusioncs.etsu.edu/pittares/demo/forms/formexample3.htm
http://coldfusioncs.etsu.edu/pittares/demo/forms/processformexample3.cfm
**cfabort**

cfabort stops processing of the current page at the location of the cfabort tag.

```xml
<cfif NOT IsDefined("form.paymentmethod")>
    <h1>You did not select a payment method</h1>
    <cfabort>
</cfif>
```

**Using cfswitch**

cfswitch can be used to test the same expression repeatedly for different values.

```xml
<cfswitch expression = "variable or expression">
    <cfcase value = "check1">
        Executed if expression is check1
    </cfcase>
    <cfcase value = "check2">
        Executed if expression is check2
    </cfcase>
    <cfdefaultcase>
        Executed if no other case matches expression
    </cfdefaultcase>
</cfswitch>
```

http://coldfusioncs.etsu.edu/pittares/demo/forms/formexample6.htm
http://coldfusioncs.etsu.edu/pittares/demo/forms/processformexample6.cfm

Don't forget pound signs!
Verifying form data

**Len(string)** returns length of string.

**Trim(string)** returns copy of string with leading and trailing whitespace removed.

**LTrim(string)** and **RTrim(string)** also exist.

```cftmpl
<cfset variables.firstname = Trim(form.firstname)>
<cfset variables.lastname = Trim(form.lastname)>
<cfif Len(firstname) LT 3 OR Len(lastname) LT 3>
    <h1>You didn't enter a valid name!</h1>
<cfelse>
    <h1>Hello, <cfoutput>#firstname# #lastname#</cfoutput></h1>
</cfif>
```

Other "Is" Functions

Wide array of "IS" functions exist. A sample:

- **IsBoolean(val)**
- **IsLeapYear(val)**
- **IsNumeric(val)**
- **IsNumericDate(val)**--checks both timestamps and strings like "12/14/68"

[http://coldfusioncs.etsu.edu/pittares/demo/forms/formexample5.htm](http://coldfusioncs.etsu.edu/pittares/demo/forms/formexample5.htm)
[http://coldfusioncs.etsu.edu/pittares/demo/forms/processformexample5.cfm](http://coldfusioncs.etsu.edu/pittares/demo/forms/processformexample5.cfm)
Database Connection

DataSources set up through Administrator's Control Panel.
Each DataSource given a name that we use to access it using ColdFusion.

http://coldfusioncs.etsu.edu/CFIDE/administrator/
Defining a Query

A query must be named and associated with a datasource.

```cfquery name="books" datasource="db2910test">
    select title, author from books order by title
</cfquery>
```

Any type of SQL statement can be placed inside cfquery block.

**cfquery causes the query to be executed.** The results are stored in the name specified in the cfquery tag.

Variables in query

The query itself may be a variable, or it may contain variables.

```cfset query="select * from books">
<cfquery name="books" datasource="db2910test">
    #query#
</cfquery>
```

```cfset table="books">
<cfquery name="books" datasource="db2910test">
    Select * from #books#
</cfquery>
```
Displaying query results

cfoutput displays query results:

cfoutput query attribute set to the name of the query.
Fields returned by the query referenced using #s.
cfoutpt automatically iterates over entire result set.

```html
<cfquery name="books" datasource="db2910test">
  select title, author from books order by title
</cfquery>
<h1>Booklist List</h1>
<cfoutput query="books">
<p><b>#$title#$</b><br />
#$author#$</p>
</cfoutput>
```

http://coldfusioncs.etsu.edu/pittares/demo/database/dbdemo1.cfm

Advanced Query Output

Output can be placed into lists, tables, etc. by using combination of HTML and output tags.

```html
<cfquery name="books" datasource="db2910test">
  select title, author from books order by title
</cfquery>
<h1>Booklist List</h1>
<table>
  <th colspan="2">Book list</th>
  <cfoutput query="books">
    <tr><td>#$title#$</td><td>#$author#$</td></tr>
  </cfoutput>
</table>
```

Be careful about what is inside and what is outside the iteration caused by cfoutput.

http://coldfusioncs.etsu.edu/pittares/demo/database/dbdemo3.cfm
Useful query-related variables

Queryname.recordcount returns the number of records returned by a query.

Currentrow, when used inside cfoutput, returns the current row/record number being processed.

Currentrow is simply a loop counter and has no relationship to data actually stored in the table.

http://coldfusioncs.etsu.edu/pittares/demo/database/dbdemo4.cfm

Database insertion

Database insertion can be done using cfquery.

<cfset name = "Robert">
<cfset address = "1234 Elm Street">
<cfquery name="test" datasource="pittares">
    insert into test values('#name#','address#','')
</cfquery>

If all form field names match up exactly with database field names, cfinsert can be used instead.

<cfinsert datasource="pittares" tablename="test">

Important: if submit button is named, that name will be attempted for insertion. Delete the name from the form.

http://coldfusioncs.etsu.edu/pittares/demo/database/dbdmoa.cfm
http://coldfusioncs.etsu.edu/pittares/demo/database/dbdemob.htm
cfinsert

cfinsert can be problematic with form fields like checkboxes which send through no data unless they are selected. To solve, test to see if form variable is set. If not, set it manually.

```cfml
<cfif not isdefined("Form.MyCheckbox")>
  <cfset Form.MyCheckbox = "No">
  <cfinsert datasource="pittares" tablename="test">

  Note that cfset can set variables in other namespaces (Form, in this example.)

Alternately can use cfparam--which will set the value of a variable if it's not already set.

```cfml
<cfparam name="Form.MyCheckbox" default="No">
```

cfparam example

```cfml
<cfset firstname = "Robert">
<cfparam name="firstname" default="NoFirstName">
<cfparam name="lastname" default="NoLastName">
<cfoutput>Hello, #firstname# #lastname#!</cfoutput>

http://coldfusioncs.etsu.edu/pittares/database/cfparamdemo.cfm
cfinsert

All fields transmitted to the CF script will attempt to be inserted into the table.

If a field is sent that is not in the table, this will result in an error.

If a field is not in the form that is in the table, that field will be set to null on the insertion. This may or may not be permitted based on table design.

cfupdate

Similar to cfinsert, only form data used to update record whose key matches the form/URL parameter.

Example..if table key field is named "emp_id" then a get/post variable "emp_id" needs to be sent containing the value of the record to be edited.

<cfupdate datasource="pittares" tablename="test">
Record deletion and Summary

There is no cfdelete tag. Deleting must be handled by the cfquery tag.

All SQL interaction can be done using cfquery. Use of cfinsert and cfupdate can be used as a time saving mechanism.

Keep in mind this does tightly pair the form design and the CF script. Any changes to table field names or structure will affect the html form (and vice versa).

Redirection using cflocation

<cflocation url="URL"> will redirect the user's browser to another page.

Unlike the PHP header function, this redirection can appear anywhere in a page, even after output has been written.

Redirection is "instantaneous," so user may not be able to read previous output.

[http://coldfusioncs.etsu.edu/pittares/database/cflocationdemo.cfm](http://coldfusioncs.etsu.edu/pittares/database/cflocationdemo.cfm)