Free Goodies: Send Email From Components

Posted At : March 30, 2006 2:57 PM | Posted By : Steve
Related Categories: Mailer.cfc

I often need to send an email when a certain action is taken. The action is usually taken in a CFC. I have often seen email sent out when the CFC method is called, but this means you must remember to send the email each time you call the method.

Better, I think, to send the email from the method itself. Unfortunately, this means that you must pass into the component (preferably on initialization) all of the information you need to send an email message (generally, the mail server and the from address but potentially a username and password as well).

This can add several aguments to the initialization method of the component that aren't central to the task that the component performs.

To avoid this, I created a Mailer component that I initialize with this information. I then pass the Mailer component to my action component and use it to send email.

For example:

I will instantiate Mailer with my mail server and default from address (I can choose a different from address for any given message):

```
<cfset Mailer =
CreateObject("component","com.sebtools.Mailer").init("mail.example.com","robot@example.com")>
```

If I needed a username and password, I would specify that during initialization as well. Note that Mailer must be installed in the com/sebtools directory either under the web root or in the custom tags directory.

I am assuming that I already have a **Data Manager** component loaded in the variable "DataMgr" (just to help keep the example short).

Now, I initialize my UserMgr component:

```
<cfset UserMgr = CreateObject("component,"UserMgr").init(DataMgr,Mailer)>
```

Now, in order to update a user and send an email to that user to let them know their record has been updated, I want to be able to call that method like this (on a form action page):

```
<cfset UserMgr.saveUser(argumentCollection=Form)>
```

Here is the UserMgr component:

UserMgr

Breaking it down by method:

```
<cffunction name="init" access="public" returntype="any" output="no">
   <cfargument name="DataMgr" type="any" required="yes">
   <cfargument name="Mailer" type="any" required="yes">

   <cfset variables.DataMgr = arguments.DataMgr>
   <cfset variables.Mailer = arguments.Mailer>

   <cfreturn this>
</cffunction>
```

This is the init() method inializes and returns the component. It takes the DataMgr and Mailer as arguments and makes them available (via variables scope) to the rest of the component.

```
<cffunction name="getUser" access="public" returntype="query" output="no">
   <cfargument name="UserID" type="numeric" required="yes">

   <cfreturn variables.DataMgr.getRecord("users",arguments)>
</cffunction>
```

The getUser() method returns a recordset of the given user. It is just included for completeness.

```
<cffunction name="saveUser" access="public" returntype="numeric" output="no">
   <cfargument name="UserID" type="string" required="no">
   <cfargument name="FirstName" type="string" required="no">
   <cfargument name="LastName" type="string" required="no">
   <cfargument name="Email" type="string" required="no">

   <cfset var result = variables.DataMgr.saveRecord("users",arguments)>
   <cfset var qUser = getUser(result)>

   <cfif Len(qUser.Email)>
      <cfset variables.Mailer.send(qUser.Email,"Record Updated","Just to let you know, your record was
updated.")>
   </cfif>

   <cfreturn result>
</cffunction>
```

The saveUser() method saves the record and sends an email to the user. Any action could have been taken before or after calling the Mailer.send() method - this was just one example. Note that DataMgr was here only to keep the examples brief - it is not needed to use Mailer.

That's it!

Mailer can also handle dynamic data and multi-part email messages, but those are topics for another day (though a glance at the documentation should provide the answers).

Mailer is a free download and has documentation available.