Prevent Form Spam with SpamFilter.cfc

Posted At : August 28, 2007 2:27 PM | Posted By : Steve
Related Categories: SpamFilter, com.sebtools

I was told last week that a client's contact form has been getting a lot of spam - on the order of 30 a day. We had a simple spam prevention measure in place, but that clearly wasn't doing the job. It was time for something a little more sophisticated.

I wanted to create a solution that would be easy to implement on any site and wouldn't require any JavaScript - or any effort at all for the user. I didn't want to count on the behavior of spammers because that is easy to adjust. What isn't as easy for them to change, however, is their content.

I implemented my solution for this client and saw immediate success. The solution has been in place a little over a week. In that time, no spam has gotten through and we haven't had any reports that legitimate messages have been stopped.

SpamFilter.cfc is called on the action page of a form and given the form structure itself. From there it uses internal word and regular expression definitions (which you can change) to determine if the form post is spam. It has two methods that can be called to stop spam

The first option is the isSpam() method. It takes the form structure and returns a boolean value indicating whether or not SpamFilter.cfc thinks that the form is spam.

```
<cfset DataMgr = CreateObject("component","DataMgr").init(mydsn,"MSSQL")>
<cfset SpamFilter = CreateObject("component","SpamFilter").init(DataMgr)>

<cfif SpamFilter.isSpam(Form)>
   <!--- Handle spam --->
<cfelse>
   <!--- Handle form submission --->
</cfif>
```

The second option is the filter() method. It takes the form structure and throws an error if the form looks like spam. This can be useful if you have an error trapping mechanism on your forms that return the error to the user.

```
<cfset DataMgr = CreateObject("component","DataMgr").init(mydsn,"MYSQL")>
<cfset SpamFilter = CreateObject("component","SpamFilter").init(DataMgr)>

<cftry>
   <cfset SpamFilter.filter(Form)>
   <cfcatch>
      <!--- Handle spam --->
   </cfcatch>
</cftry>

<!--- Handle form submission --->
```

In either event, if the message looks like spam, you can take what action you deem necessary. For example, I generally send a message back to the browser to let the user know their post was blocked as spam.

SpamFilter.cfc has built-in methods to allow you to manage the words and regular expressions (and their weighting).

I have also set up a page on my web site that will house more spam words as I find them. To add those to your definitions, simply call the loadUniversalData() method. If you want SpamFilter.cfc to call that method internally when initialiazed, simply set the second argument of the init() method ("getNewDefs") to true.

SpamFilter.cfc does need a database and does require my free **DataMgr** component set in order to

work (**What is DataMgr?**). All of the tables created by SpamFilter.cfc have a "spam" prefix, which should make it safe to use with an existing datasource.

SpamFilter.cfc (now in beta) is open source and free for any use. Feel free to give it a try.