XEP Component for ColdFusion MX version 1.0

Table of Contents

1. Overview

XEP Component for ColdFusion MX is built around RenderX XEP XSL FO formatter, which transforms XML documents into PDF or PostScript. It enables to use XEP formatter in web-applications. XEP Component for ColdFusion MX is a high-quality page layout engine. Through this product it is possible to make web-applications more attractive and more functional as well as to create documents completely meeting your initial requirements.

2. Prerequisites

To install ColdFusion MX XEP Component, the following products should be installed:

- XEP 4.0;
- Sun JDK 1.2.2 or higher;
- ColdFusion MX 6.1.

3. Installation

3.1. GUI Installation

- Extract a README.txt and xep.jar files from the xep.zip file. If you have multiple VMs on you computer, choose one of them.

- Run Setup from JAR file using Java VM. If your Java VM supports Java 2 (JDK/JRE 1.2 or higher), you can run JAR by typing the following command on the prompt:

```
java -jar xep.jar
```

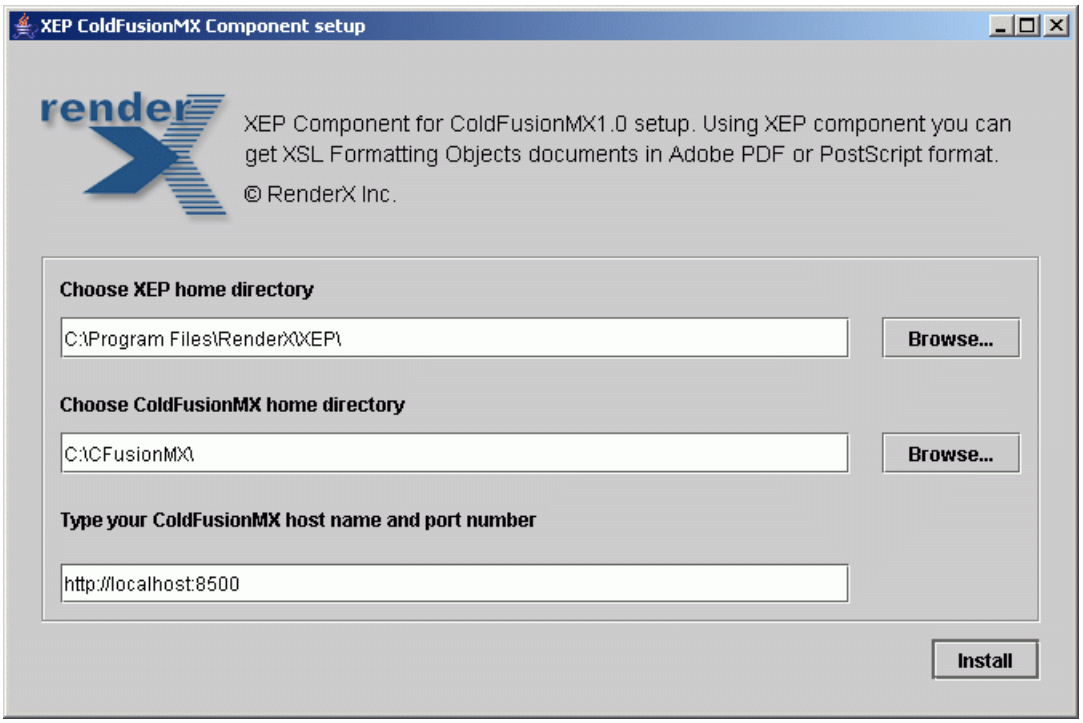
- Alternatively, for Java machines, which don't support executable JAR files, you can call Setup class from JAR using standard invocation methods.

  E.g. for Microsoft Java, the syntax will be as follows:

```
jview /cp xep.jar Setup
```

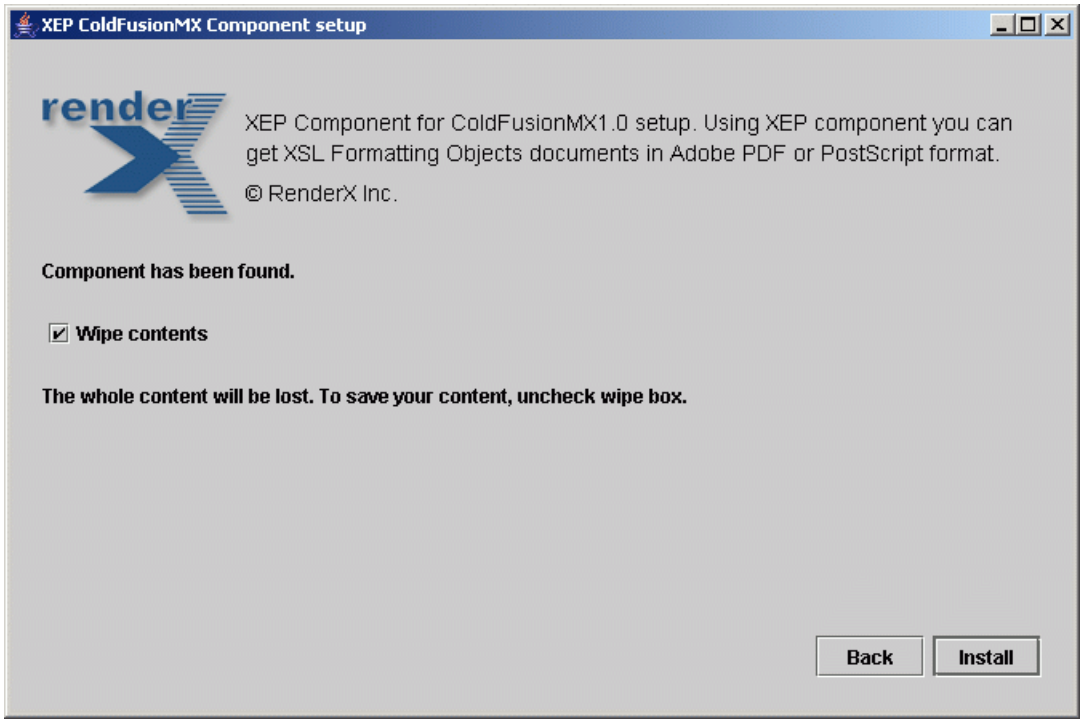
- You should specify both XEP and ColdFusion MX home directory paths by typing the path in the edit box, or choosing the directory by pressing "Browse" button; on systems that have no Swing support, the "Browse" button is not available, so you should type the path manually. Type server's URL in appropriate edit box in the following way http://<host_name>:<port_number> (by default it will be http://localhost:8500). (see Figure 1).

Figure 1.



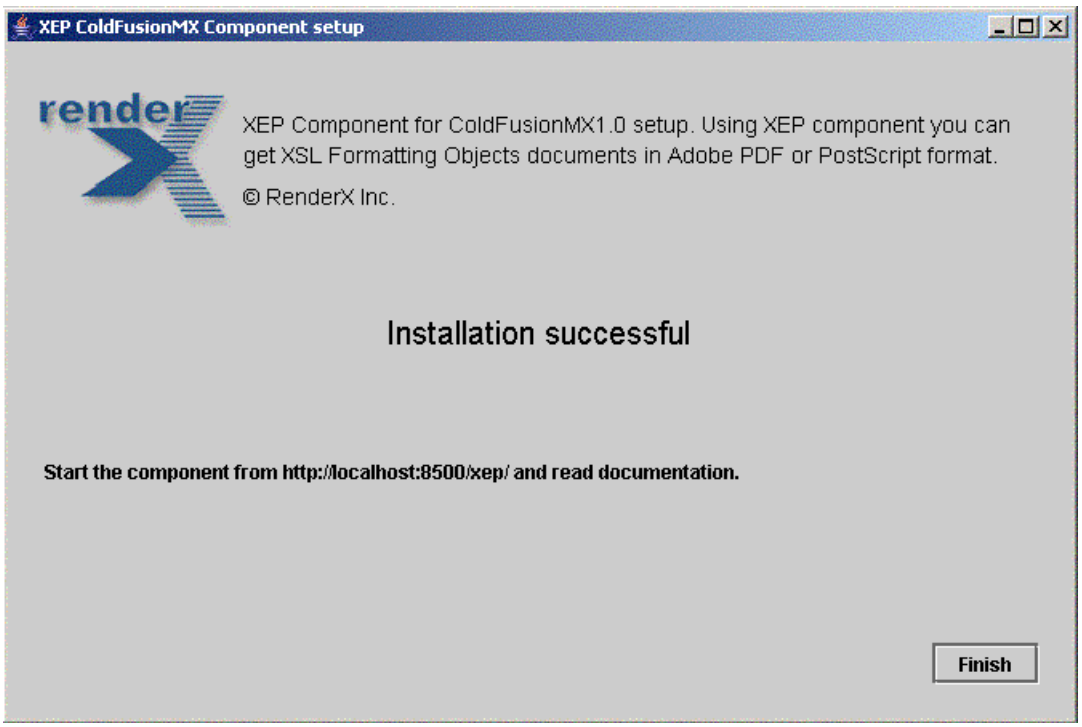
- If you have already installed XEP Component, after specifying the path to the XEP home directory, you should see the Wipe Screen (see Figure 2). Wiping of the component, means removing folders created by previous installation, including files created by user in that folders. By default wipe check-box is checked. To save the content of the directories and to install just the component un-check the wipe check-box.

Figure 2.



- In case of successful installation you should get the following message: "Installation successful"(see Figure 3).

Figure 3.



3.2. Installation from Console

To install XEP Component for ColdFusion MX from Console it is necessary to run

```
java -jar xep.jar -c
```

and execute the following steps:

- Type full path to XEP home directory to continue installation or QUIT to exit.
- Type full path to ColdFusion MX home directory to continue installation or QUIT to exit.
- Type host name and port number to continue installation or QUIT to exit (by default: http://local-host:8500)

The system copies all files to the specified location and sets up necessary configuration files.

4. Running XEP Component for ColdFusion MX

After installation, the copy of your XEP Component for ColdFusion MX is ready to use. The XEP Component contains the following directories:

$CFusionMX_HOME/wwwroot/WEB-INF/classes/com/

the component's classes, where $CFusionMX_HOME/wwwroot is a root of the ColdFusion MX installation directory;

$CFusionMX_HOME/wwwroot/xep/

root directory of XEP Component, where there are configuration files, documentation and samples of demonstrating applications of the XEP Component.

File configuration of XEP Component is $CFusionMX/wwwroot/xep/xep.properties and includes the following configuration parameters:

CONFIG=$XEP_HOME/xep.xml

Path to XEP configuration file, where $XEP_HOME is path to XEP home directory. This parameter is installed during the installation.

BYTE_LIMIT=1048586

Upload limitation. By default 1048586 (1024x1024) is specified.

com.renderx.sax.entityresolver=com.sun.resolver.tools.CatalogResolver
com.renderx.jaxp.uriresolver=com.sun.resolver.tools.CatalogResolver

Resolution of External Entities and URIs. By default these parameters are commented.

As XEP Component is written on Servlet, by default, ColdFusion MX does not detect changes of servlets, so to see the changes, you must restart the server. To prevent the restart of the server each time after the reinstallation of a new application written on servlets, you need to edit the jrun-web.xml file in the WEB-INF directory where ColdFusion is installed. This procedure reconfigures the ColdFusion MX for automatically reload changed servlets: in jrun-web.xml file you may find a pair of

```
<jrun-web-app>
```

tags. Within those elements, add the following line

```
<reload>true</reload>
```

and restart ColdFusion Server, after which the component will be activated.

Open a homepage of XEP Component by the following way: if your server is set on the local machine, type in address bar of a browser the following URL:

```
 http://localhost:8500/xep/
```

otherwise type

```
http://<host_name>:<port_number>/xep/
```

where you may find several examples of XEP Component applications.

Form HTTP request to run XEP Component. The full synopsis of the HTTP request includes the following parameters:

```
http://<host_name>:<port_number>/servlet/
      com.renderx.xepx.cf.Formatter?
      (xml=<source>&xsl=<stylesheet>|fo=<source>)
      [&format=(PDF|PostScript)]
      {&( |xsl_|pdf_|ps_)<name>=<value>}
```

xml, xsl

Source XML and transformation stylesheet.

fo

XSL FO document.

format

The PDF and PostScript output formats are supported, the default format is PDF.

<name>=<value>

Options for XEP (no prefix), PDF generator (prefixed with pdf_), PostScript generator (prefixed with ps_) and XSL transformation parameters (prefixed with xsl_). Valid options are described in RenderX XEP documentation.

In order to get a clear idea of using XEP Components, read the next section.

5. Sample applications.

You can find several applications using XEP Component, included with installation. Opening application at: *http://<host_name>:<port_number>/xep/* will guide you to the following examples.

- Library - XEP Component is adaptable in designing of electronic libraries. The sample presents the e-library, where you get an opportunity to download a desired book either in PDF or PostScript formats.
- SVG Viewer - Support for both XSL FO and SVG makes XEP a powerful tool for data representation. The sample demonstrates a formatting of XSL FO file generated from SVG into either PDF or PostScript through the upload form using XEP Component and shows how to pass some backends options to the component.
- Application Form - the example of an application form. Fill it online and get it in PDF format.

5.1. Library

Steps of using library:

1. Open a web page of e-library.
2. Choose a book and select PDF or PS output format.
3. Get the book in a previously selected format.

4. Print the book or get it saved for offline use.

Organizitaion of electronic libraries is realized through the following way. All book sources are stored in XML/XSL or FO formats. The code of the page *http://<host_name>:<port_number>/xep/library.cfm* is written on ColdFusion using HTML.

```
...
<cfset essayPDF="xml=$CFusionMX_HOME/wwwroot/xep/books/essay.xml&xsl=$CFusion-
MX_HOME/wwwroot/xep/books/essay.xsl&format=PDF">
<cfset essayPS="xml=$CFusionMX_HOME/wwwroot/xep/books/essay.xml&xsl=$CFusion-
MX_HOME/wwwroot/xep/books/essay.xsl&format=PostScript">
<cfset storyPDF="xml=$CFusionMX_HOME/wwwroot/xep/books/story.hml&xsl=$CFusion-
MX_HOME/wwwroot/xep/books/story.xsl&format=PDF">
<cfset storyPS="xml=$CFusionMX_HOME/wwwroot/xep/books/story.xml&xsl=$CFusion-
MX_HOME/wwwroot/xep/books/story.xsl&format=PostScript">
...
<cfoutput>
 <table>
  <tr>
   <td>
    Mike Dude, <i>"Manet - Still Life"</i> - get the book as
   <a href="http://<host_name>:<port_number>/servlet/com.renderx.xepx.cf.Format-
ter?#essayPDF#">PDF</a> or
   <a href="http://<host_name>:<port_number>/servlet/com.renderx.xepx.cf.Format-
ter?#essayPS#">PS</a>
   </td>
  </tr>
  <tr>
   <td>
    Rudyard Kipling, <i>"How The Camel Got His Hump"</i> - get the book as
   <a href="http://<host_name>:<port_number>/servlet/com.renderx.xepx.cf.Format-
ter?#storyPDF#">PDF</a> or
   <a href="http://<host_name>:<port_number>/servlet/com.renderx.xepx.cf.Format-
ter?#storyPS#">PS</a>
   </td>
  </tr>
 </table>
</cfoutput>
...
```

To call the component it is enough to form correspondent URL as shown above.

5.2. SVG Viewer

On the submit form HTTP request is generated, which besides XML/XSL or FO sources includes additional options as well.

Each field of the form is xml, xsl, fo, option or output format parameter, where the fieldname is a parameter name and field value is value of that parameter.

The code of the page *http://<host_name>:<port_number>/xep/upload.cfm* is written on ColdFusion using HTML.

```
...
<cfform action="/servlet/com.renderx.xepx.cf.Formatter" method="GET" enctype="mul-
tipart/form-data">
...
<input type="file" size="40" name="fo">
...
<input name="format" type="radio" value="PDF" checked>
...
<input name="format" type="radio" value="PostScript">
...
<input type="checkbox" checked name="validate">
...
<input type="checkbox" checked name="pdf_UNICODE_ANNOTATIONS">
...
<input type="checkbox" checked name="ps_DROP_UNUSED_DESTINATIONS">
...
</cfform>
```

The following options are used in the form.

Core Options

`VALIDATE`

Boolean value (*`true`*/*`false`*). Controls validation of input. In non-validating mode, XEP runs faster and takes less memory; however, less errors are intercepted, and the results of formatting are less predictable for malformed input. We discourage setting this switch to *`false`* until your stylesheets are thoroughly debugged.

Default: *`true`*

`DISCARD_IF_NOT_VALID`

Boolean value (*`true`*/*`false`*). Controls termination of processing upon unsuccessful validation.

Default: *`true`*

PDF Parameters for Output Generators

`COMPRESS`

Boolean value. Turns PDF compression on (*`true`*) and off (*`false`*). Useful only to debug generated PDF code

Default: *`true`*

`LINEARIZE`

Boolean value. Turns PDF linearization (also known as Web optimization) on (*`true`*) and off (*`false`*). Useful for PDF documents published on the Web.

Default: *`false`*

`UNICODE_ANNOTATIONS`

Boolean value. Controls whether PDF annotations (bookmarks, document info, etc) can be stored as two-byte Unicode strings. If it is set to *`false`*, all annotations are stored in a special single-byte encoding (`PDFEncoding`); all characters outside its character set will be replaced by bullets.

Default: *`true`*

`DROP_UNUSED_DESTINATIONS`

Boolean value. If it equals to *`false`*, all `id` attributes generate a named destination in the PDF output, with the name equal to the value of the attribute. If it is *`true`*, named destinations are created only for those `id` attributes that are referenced by an `internal-destination` attribute on some `<fo:basic-link>` or `<rx:bookmark>` element inside the document.

Default: *`true`*

PS Parameters for Output Generators

`CLONE_EPS`

Boolean value. Controls treatment of embedded EPS graphics. If *`true`*, EPS image contents are repeated for each occurrence of the image in the document. If *`false`*, EPS images are treated like other image formats: the contents is written once in the document prolog section. This saves space, but not all EPS images can be used this way.

Default: *`true`*

`UNICODE_ANNOTATIONS`

Boolean value. Controls whether PDF annotations embedded in PostScript (pdfmarks) are stored as two-byte Unicode strings, or single-byte strings in PDFEncoding.

Default: *`true`*

`DROP_UNUSED_DESTINATIONS`

Boolean value. If it equals to *`false`*, all `id` attributes generate a named destination (as a pdf-mark), with the name equal to the value of the attribute. If it is *`true`*, named destinations are

created only for those `id` attributes that are referenced by an `internal-destination` attribute on some `<fo:basic-link>` or `<rx:bookmark>` element inside the document.

Default: *`true`*

More information about XEP formatter's options you can find at "XEP 4.0 Reference for Java"

5.3. Application Form

Steps of using application form:

1. Find a registration page
2. Fill in the form and submit it.
3. Get an application form in PDF format

Let's see how the app.cfm is organized.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "ht-
tp://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
 <head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
  <title>Please Register</title>
  <style type="text/css"> </style>
 </head>
 <cfif isDefined('form.submit')>

/--Reading form's fields and writing final FO file of the application--/
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/name.fo" vari-
able="stringName">
  </cflock>
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/birthdate.fo"
variable="stringDate">
  </cflock>
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/birthdate.fo"
variable="stringDate">
  </cflock>
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/occupation.fo"
variable="stringOccupation">
```

```
  </cflock>
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/address.fo" vari-
able="stringAddress">
  </cflock>
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/phone.fo" vari-
able="stringPhone">
  </cflock>
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/cell.fo" vari-
able="stringCell">
  </cflock>
  <cflock name="CLock" type="ReadOnly" timeout="30">
   <cffile action="read" file="$CFusionMX_HOME/wwwroot/xep/fo/final.fo" vari-
able="stringFinal">
  </cflock>
  <cfset final=stringName & #XMLFormat(form.name)# & " " & #XMLFormat(form.last-
name)# &
    stringDate & #XMLFormat(form.birthday)# & stringOccupation & #XML-
Format(form.occupation)# &
    stringAddress & #XMLFormat(form.address)# & stringPhone & #XML-
Format(form.phone)# &
    stringCell & #XMLFormat(form.cell)# & stringFinal>
  <cffile action="write" file="$CFusionMX_HOME/wwwroot/xep/fo/file.fo" output=#fi-
nal#>
  </cffile>

/--Creation and sending HTTP request--/
  <cfset urlstr="http://localhost:8500/servlet/com.renderx.xepx.cf.Format-
ter?fo=$CFusionMX_HOME/wwwroot/xep/fo/file.fo">
  <cflocation url=#urlstr#>
 <cfelse>

/--Application form--/
  <cfform action="app.cfm"  method="post" enctype="application/x-www-form-urlen-
coded">
   <table border="0" align="center" bgcolor="white">
    <tr> <td> </td> </tr>
    <tr>
     <td>
      <div align="center">
       <span style="color:#993333; font-family: arial, helvetica, sans-
serif;font-weight : bold;font-style : italic; font-size:20px">
        Demo for XEP ColdFusion MX Component.
```

```
       </span>
      </div>
     </td>
    </tr>
    <tr> <td> </td> </tr>
    <tr>
     <td>
      <div style="color:navy; font-family: arial, helvetica, sans-serif;font-
weight : bold;font-style : italic; font-size:14px">
       Please fill in the form required data to get the Application Form in PDF
 format.
      </div>
     </td>
    </tr>
    <tr> <td> </td> </tr>
   </table>
   <table border="1" align="center">
    <tr>
     <td>
      <table width="350"  border="0" align="center" cellpadding="0" cellspa-
cing="0" bgcolor="white">
       <tr bgcolor="#336699">
        <td height="30" colspan="3"> <div align="center"> <span style="font-
size: 20pt; color: white"> <b> <i>Application Form </i> </b> </span> </div>
</td>
       </tr>
       <tr> <td> </td> </tr>
       <tr>
       <td width="120"> <div align="right">First Name:<span style="color: red">
 *</span> </div> </td>
        <td> </td>
        <td width="200"> <cfinput name="name" type="text" message="You must
enter your name." required="yes"> </td>
       </tr>
       <tr> <td> </td> </tr>
       <tr>
        <td> <div align="right">Last Name:<span style="color: red"> *</span>
</div> </td>
        <td> </td>
        <td> <cfinput name="lastname" type="text" message="You must enter your
 surname." required="yes"> </td>
       </tr>
         <tr> <td> </td> </tr>
       <tr>
       <td> <div align="right">Date of Birth:<span style="color: red"> *</span>
```

```
 </div> </td>
        <td> </td>
        <td> <cfinput name="birthday" type="text" validate="eurodate" mes-
sage="You must enter your date of birth." required="yes"> </td>
       </tr>
         <tr> <td> </td> </tr>
       <tr>
        <td> <div align="right">Occupation:</div> </td>
        <td> </td>
        <td> <cfinput name="occupation"type="text" required="no"> </td>
       </tr>
         <tr> <td> </td> </tr>
       <tr>
       <td> <div align="right">Address:<span style="color: red"> *</span> </div>
 </td>
        <td> </td>
        <td> <cfinput name="address" type="text" message="You must enter your
address." required="yes" > </td>
       </tr>
         <tr> <td> </td> </tr>
       <tr>
        <td> <div align="right">Phone:</div> </td>
        <td> </td>
        <td> <cfinput name="phone" type="text" required="no"> </td>
       </tr>
         <tr> <td> </td> </tr>
       <tr>
        <td> <div align="right">Cell:</div> </td>
        <td> </td>
        <td> <cfinput name="cell" type="text" required="no"> </td>
       </tr>
       <tr>
        <td colspan="3"> </td>
       </tr>
       <tr>
        <td colspan="3"> <div align="center">
         <input type="submit" name="Submit" value="Get PDF">
        </div> </td>
       </tr>
       <tr>
        <td colspan="3"> </td>
       </tr>
      </table>
     </td>
    </tr>
```

```
   </table>
  </cfform>
 </cfif>
```