Getting Started with ColdFusion

ColdFusion 4.0 for Windows® NT, Windows 95/98, and Solaris

Allaire Corporation

Copyright Notice

Part number: AA-GSTRT-RK

Contents

CHAPTER 1

Welcome To ColdFusion

ColdFusion is a rapid application development system for professional developers who want to create dynamic Web applications and interactive Web sites. It provides the fastest way to integrate browser, server, and database technologies into powerful Web applications. With ColdFusion, you can build everything from online stores to sophisticated business systems.

Developing applications with ColdFusion does not require coding in a traditional programming language; instead, you build applications by combining standard HTML with a straightforward server-side markup language, the ColdFusion Markup Language (CFML).

Contents

Product Features

This release marks a significant milestone in the evolution of ColdFusion as a development system for building scalable Web applications that integrate browser, server, and database technologies.

The focus of our development work has been in four major areas: rapid development, scalable deployment, open integration, and total security. Each of these areas is highlighted below.

Rapid Development

ColdFusion 4.0 continues to enhance the speed of development and ease-of-use that have been the hallmark of the development system from its beginning. ColdFusion 4.0 increases development productivity by integrating ColdFusion Studio more closely with ColdFusion Server, extending the visual tools, and expanding the functionality of the tag-based server scripting language, CFML.

New Feature Highlights

- **Two-way Visual Programming** – ColdFusion Studio 4.0 includes new, more powerful visual programming tools including a WYSIWYG design mode and enhanced visual database tools.
- **Dynamic State Simulation** – The ColdFuson Studio IDE supports establishing state for pages so developers can preview the interactions between pages that rely on multiple variables.
- **Dynamic Page Quality Assurance** – New tools support validating links, configuring dynamic page previewing and validating CFML grammar in pages.
- **One-step Deployment** – New features extend the site- and page-management features to support flexible deployment of complex applications to multiple servers, making the process of moving from development to deployment simple and straightforward.
- **Site Visualization** – ColdFusion Studio 4.0 supports the ability to visualize sites and see how pages are linked to each other across a system.
- **CFML Script** – CFML has been extended to support traditional scripting syntax for complex data processing on the server using branching and looping.

Scalable Deployment

ColdFusion has already reached a point where it is being used to deliver very large volume sites and applications servicing tens of thousands of users. With the 4.0 release, ColdFusion provides powerful new features that significantly enhance scalability.

New Feature Highlights

- **Load Balancing –** ColdFusion 4.0 supports native load balancing giving developers the ability to deploy large volume applications in high performance clusters that scale to meet any user demands. (Enterprise Edition only.)
- **High Availability –** ColdFusion 4.0 supports the creation of multi-server clusters with automatic fail-over if any server goes down – providing the infrastructure for deploying large volume high-availability sites. (Enterprise Edition only.)
- **Open State Repository** – State information can now be moved out of the registry into a pluggable external data source so servers can be easily configured for clustering and load balancing.
- **Advanced Thread Pooling** – The Web application server offers sophisticated thread pooling using I/O completion ports and tight integration with web server APIs.
- **Integration with NT Performance Monitor** – ColdFusion Server is fully integrated with the NT Performance Monitor for increased manageability and tuning.

Open Integration

ColdFusion offers better integration with server systems including mail, web servers and directories than any other IRAD system. With the 4.0 release, this integration has been extended to support Extensible Markup Language (XML) and enterprise technologies.

- **Automatic XML Parsing** – ColdFusion Server supports automatic parsing of XML data into CFML variables and the translation of CFQUERY record sets into XML.
- **Native Database Drivers** – ColdFusion 4.0 supports native database connectivity for Oracle and Sybase. (Enterprise Edition only.)
- **CORBA** – ColdFusion 4.0 extends its integration with component standards by supporting Common Object Request Broker Architecture (CORBA) and possibly Enterprise JavaBeans. (Enterprise Edition only.)
- **ColdFusion Extensions (CFX)** – ColdFusion 4.0 supports the creation of more complex CFXs (formerly called "Custom Tags") making it possible to extend ColdFusion with components created with CFML, C/C++, COM, CORBA, JavaBeans, JavaScript and VBScript.

Total Security

ColdFusion 4.0 has extended previous security features to enable a much greater range of flexibility and control over security both for development and deployment.

- **Open Authentication System** – Developers can leverage a wide range of different user authentication systems in their applications from within ColdFusion including Windows NT security, LDAP directories, and proprietary user and group databases.
- **Advanced Remote Development Security** – The Remote Development Services (RDS) used by ColdFusion Studio allow for user and group security configuration for all resources including files and databases using a configurable backend authentication system that integrates with existing user and group databases.
- **Server Sandbox Deployment** – With the server sandbox, server administrators can control what resources (files, databases and components) an application has access to when it is running on a server. This allows server administrators to deploy multiple applications on the same server without creating the risk that one application will access another application's resources.

Learning About Web Development and ColdFusion

Web application development is such a new field and requires such a mix of emerging and established technologies that meeting the documentation needs of ColdFusion users is quite a challenge. The skills required to build and deploy dynamic Web content range across HTML, databases, graphic arts, networking, a slew of scripting and programming languages, and even writing.

We have tried to present information on ColdFusion development and supporting technologies so that you can pursue topics of interest to you and integrate them into your overall learning process.

While it is certainly possible for an individual to master all these skills, the team approach has quickly become the only realistic development model for delivering complex applications and we address issues such as building and maintaining Web projects and working with version source control.

We also include pointers to many resources, both print and online, that provide additional information about ColdFusion and supporting technologies.

New to Web development?

The ColdFusion Markup Language is a tag-based language that integrates with HTML to provide greatly enhanced functionality for Web sites. The skills you are building in HTML and Web site development are a solid foundation for ColdFusion development.

ColdFusion Studio, our integrated development environment (IDE), offers many powerful features for building and maintaining Web sites. With its powerful user assistance, embedded help, and online documentation, you can use Studio to learn HTML, to develop and test Web sites, and to develop dynamic content with CFML.

New to ColdFusion?

Getting Started with ColdFusion presents a quick tour of a ColdFusion application. *Developing Web Applications* is a good place to start learning about building ColdFusion applications.

If you want access to experienced ColdFusion developers, you can participate in the Allaire Online Forums, where you can post messages and read replies on all subjects relating to ColdFusion. Check out the Forums at http://forums.allaire.com.

In addition, our Web site http://www.allaire.com is a tremendous resource for learning about ColdFusion programming techniques. You'll find Tech Notes, a Knowledge Base, as well as Allaire Alive, a repository of online video training sessions that focus on a specific area in ColdFusion. Check out the Developer site at http://www.allaire.com/developer for articles, technical notes, and the ColdFusion DevCenter.

Experienced Web developer?

You'll probably want to get going with your project, so take a look at the *Developing Web Applications* chapters on setting up data sources, managing input and output, the application framework, Java forms, and programming variables. *Getting Started with ColdFusion* includes a complete application with lots of working code samples that you can drop in to quickly prototype a project. If you want to integrate COM, CORBA, custom tags, CF API tags, LDAP, CFML scripting, or XML data exchange into your applications, see ../../Advanced_ColdFusion_Development/contents.htm*Advanced ColdFusion Development*/a.

Developer Resources

Allaire Corporation is committed to setting the standard for customer support in developer education, technical support, and professional services. Our Web site is designed to give you quick access to the entire range of online resources.

Allaire Developer Services	
Resource	**Description**
Allaire Web site http://www.allaire.comwww.allaire.com/a	General information about Allaire products and services.
Technical Support http://www.allaire.com/supportwww.allaire.com/support/a	Allaire offers a wide range of professional support programs. This page explains all of the available options.

Allaire Developer Services (Continued)	
Resource	**Description**
Professional Education http://www.allaire.com/educationwww.allaire.com/education/a	Information about classes, on-site training, and online courses offered by Allaire.
Developer Community http://www.allaire.com/developerwww.allaire.com/developer/a	All of the resources you need to stay on the cutting edge of ColdFusion development, including online discussion groups, Knowledge Base, Component Exchange, Resource Library, technical papers and more.
Allaire Alliance http://www.allaire.com/partnerswww.allaire.com/partners/a	The growing network of solution providers, application developers, resellers, and hosting services creating solutions with ColdFusion.

Developing Applications in ColdFusion Studio

ColdFusion Studio is a special version of HomeSite, Allaire's award-winning HTML editor. HomeSite's strengths in Web page creation have been enhanced with powerful tools specifically designed for ColdFusion development.

All of the components of dynamic page creation and site management are accessible from Studio.

- View your data sources
- Quickly build SQL statements to insert in CFQUERY
- Access the complete HTML and CFML tag sets from the Tag Chooser
- Edit code from tag-specific editors or from the Tag Inspector
- Render pages with internal or external browsers and visually edit page elements in Design mode
- Create projects to group your application pages and support files for easy maintenance and uploading
- Quickly make global changes to files using extended search and replace
- Save code blocks for re-use as snippets
- Build ColdFusion expressions from the complete set of CF functions, constants, operators, and variables available in the Expression Builder
- Debug application code
- View your site's structure in the Visualizer
- Validate HTML and CFML code
- Verify links for individual files or entire projects
- Enable version source control of your files for team development

About ColdFusion Documentation

The documentation set is designed to provide support for all components of the ColdFusion development system. Both the print and online versions are organized to allow you to quickly locate the information you need.

Documentation set

The documentation set contains:

../../Getting_Started_with_ColdFusion/contents.htm*Getting Started with ColdFusion*/a — Covers system installation and basic configuration, describes the components of the ColdFusion development system, and introduces the ColdFusion Markup Language (CFML).

../../Administering_the_App_Server/contents.htm*Administering ColdFusion Server*/a — Describes configuration options for maximizing performance, managing data sources, setting security levels, and a range of development and site management tasks.

../../Developing_Web_Apps/contents.htm*Developing Web Applications with ColdFusion*/a — Presents the fundamentals of ColdFusion application development and deployment, including data sources, user interfaces, and Web technologies. The development tools in ColdFusion Studio are covered in detail.

../../Advanced_ColdFusion_Development/contents.htm*Advanced ColdFusion Development*/a — Gives an overview of CFML elements such as functions, expressions, arrays, scripting, and XML data exchange. Also discusses custom tags, CF API tags, integrating object technologies, and site management.

../../CFML_Language_Reference/contents.htm*CFML Language Reference*/a — Provides the complete syntax, with example code, of all CFML elements.

Quick Reference Card — An online (Acrobat) guide to CFML.

Documentation distribution

The ColdFusion CD-ROM contains the complete document set. The setup program installs the document set by default.

The print manuals are available in Adobe Acrobat (PDF) format from the `dochome.htm` page in the `/cfdocs` directory of your Web root. If the files are not available locally, you get them from our Web site at http://www.allaire.com/products/COLDFUSION/Documentation.cfm.

You can also access the documentation in HTML from both of these locations.

Reading online documentation

You can open the online documents in a number of ways:

- From your browser, click the ColdFusion Documentation link on the Welcome to ColdFusion page. Each page contains links to other documents and a search window.
- In ColdFusion Studio, click the Help tab in the Resources area to open the help tree. You can expand the list to select topics by title.

Using Online Help In ColdFusion Studio

Studio's innovative online documentation system provides a variety of help options:

- Help References — The complete documentation set is available in HTML
- Dialog Help — Inline help for all HTML and CFML tags in the Tag Chooser and tag editors
- Tag Tips — Display pop-up help for tag syntax
- Tag Insight — Opens a selectable list of attributes for the current tag

To set Tag Tip and Tag Insight options, go to the Tag Help tab in Options (F8).

The default display of the online documentation is in the internal browser.

To change the help display:

1. Click the External Browser button on the main toolbar to open the current document in the default external browser. If multiple browsers are detected on the system during setup, a drop-down list of available browsers is generated automatically.
2. Click the Toggle Resources button on the main toolbar (F9) to toggle the display of the Resources area.

Searching Help References

Full-text search is available for the entire Help References set.

To use full-text search for Help References:

1. Click the Help Search button on the Help toolbar to open the search dialog.
2. Type in the search text and click a search criterion button. Search text is saved in the drop-down list for future use.
3. Set the scope of the search by selecting All References or use CTRL + mouse click to choose individual references. You can search down to the page level.
4. Click Search. The results display in the Resources area. Double-click on a result reference to open the document.
5. Click the Help References button to return to document tree.

6. Click the Results button to return to the last search results.

Tip: You can extend the online documentation in Studio by adding your own HTML files. Just copy a folder to the Help directory under the ColdFusion Studio directory. Press F5 to refresh the Help reference list. You can now browse and search these files in the Help References.

Documentation conventions

When reading, please be aware of these formatting cues:

- Code samples, filenames, and URLs are set in a distinct font
- Notes and tips are identified by bold type in the margin
- Bulleted lists present options and features
- Numbered steps indicate procedures
- Toolbutton icons are generally shown with procedure steps
- Menu levels are separated by the greater than (>) sign
- Text for you to type in is set in *italics*

Contacting Allaire

Corporate headquarters

Allaire Corporation
One Alewife Center
Cambridge, MA 02140

Tel: 617.761.2000
Fax: 617.761.2001

Web site: http://www.allaire.com

Technical support

Telephone support is available Monday through Friday 8 A.M. to 8 P.M. Eastern time (except holidays)

Toll Free: 888.939.2545 (U.S. and Canada)
Tel: 617.761.2100 (outside U.S. and Canada)

Postings to the http://forums.allaire.comColdFusion Support Forum/a can be made at any time.

Sales

Toll Free: 888.939.2545
Tel: 617.761.2100
Fax: 617.761.2101

Email: mailto:sales@allaire.comsales@allaire.com/a
Web: http://www.allaire.com/storewww.allaire.com/store/a

CHAPTER 2

Installing and Configuring ColdFusion

This chapter contains setup information and two brief examples to illustrate the creation of a ColdFusion application.

Contents

System Requirements

ColdFusion is supported in both Microsoft Windows and Sun Solaris environments. This section describes system requirements for each environment.

Windows 95 and 98, and Windows NT

To install and use ColdFusion, your system must meet the following requirements:

- **OS:** Windows NT 3.51 (or higher) or Windows 95/98
- **Processor**: Intel 486 or higher (Pentium recommended)
- **Disk space**: 30MB
- **Memory**: 32MB
- **Web server**: A World Wide Web server (supporting NSAPI, ISAPI, or WSAPI recommended)
- **Other**: A CD-ROM drive

Solaris (Enterprise edition only)

ColdFusion for Solaris has the following system requirements:

- **OS**: Solaris 2.5.1 or 2.6 running on Sparc
- **Memory**: 128 MB
- **Disk space**: 120 MB for a full install
- **Web server**: Netscape Fast Track/Enterprise versions 2.01 or 3.0, 3.51 or Apache 1.2 or 1.3
- **Other**: A CD-ROM drive

Clustering and load-balancing requirements

The Solaris clustering and load-balancing installation option has the following system requirements, in addition to the general system requirements outlined above.

- Netscape Enterprise Server 3.0 or greater.
- Root access privileges on each Solaris Server.
- All systems participating in the SmartCluster, including the systems running the ClusterCATS Explorer, must have access to every other system using both the IP address and the name of the system.

Windows Installation

Before installing ColdFusion, check that your Web server is installed and running. You can do this by loading a page in your Web browser using the HTTP protocol.

To check that your Web server is running:

1. Open a page in your Web browser using HTTP. For example, if the file myindex.htm is in your Web document root directory, load the page in your browser using the localhost URL:

   ```
   http://127.0.0.1/myindex.htm
   ```

   If your Web server is not running, you'll receive an error message.

2. You can also check your Services Control Panel or Web server administrative utility to manage the state of your Web server.

We recommend that you close all open applications before running setup. This will ensure proper installation of the ODBC 3.0 drivers. During the setup process, the setup wizard prompts you for the following information:

- A directory for ColdFusion program files
- The ColdFusion components you want to install
- The Web server you want to configure for ColdFusion

Note Because of a problem in the way Verity handles indexing, ColdFusion must be installed to a directory path that contains no spaces in the path name. This restriction is necessary so that ColdFusion can properly access Verity collections.

To install ColdFusion for Windows:

1. Log in to your system either using the administrator's account or using a login that is part of the Administrator's group.
2. Run `setup.exe` on the ColdFusion CD-ROM.
3. Read the ColdFusion license agreement.
4. Follow the instructions on your screen.
5. The setup wizard polls your computer and displays a list of the Web servers it detects. Select the server you want ColdFusion to run on. ColdFusion examples and documentation install below the selected server's root directory. If you select the Other Server option, you can type in a path or browse the tree to select a directory. You must run Setup again if you want to change to a different server.
6. Accept the default program folder name or type in a new one. The setup wizard installs the ColdFusion files.
7. If your system has a previous version of ColdFusion, the setup wizard replaces ColdFusion program files but does not remove or change any existing application pages (template files) that may be present.

8. ColdFusion updates the Windows registry and the ColdFusion services start. If you chose to install the ODBC Desktop Drivers, a warning screen appears. Close all open applications before installing the drivers.
9. Click Finish to complete the installation.
10. The ColdFusion Welcome page opens in your browser. Read the Release Notes for the latest information about configuring ColdFusion, product fixes, and enhancements.
11. To verify your installation, open the ColdFusion Administrator:

    `http://`*`hostname`*`/CFIDE/Administrator/index.cfm`

    Where *hostname* is the name of the server hosting ColdFusion.

If ColdFusion is not properly installed after following the steps above, please call Allaire Customer Service for help installing ColdFusion.

Uninstalling ColdFusion: Windows

To uninstall ColdFusion, click the Uninstall ColdFusion 4.0 icon in your ColdFusion program group. The InstallShield uninstaller automatically removes ColdFusion from your system.

Configuring the Apache Web Server for Windows

To configure the Apache Web Server version 1.3 for use with ColdFusion, you need to:

- Get a copy of the Apache Web server from the Apache HTTP Server Project web site at http://www.apache.org/.
- Read the Apache documentation on support for Win32 platforms: http://www.apache.org/docs/windows.html.
- Copy an Apache module to your Apache modules directory.
- Add a line to a configuration file.

It is assumed in the following procedure that your Apache installation is found in `c:\Apache`.

To configure Apache for ColdFusion:

1. If a version of the Apache Web server is running, shut down the web server.
2. Copy the module `cfusion\bin\ApacheModuleColdFusion.dll` to your Apache modules directory, typically: `c:\Apache\modules\ApacheModuleColdFusion.dll`.
3. Edit the `c:\Apache\conf\httpd.conf` configuration file to contain the following line:

```
LoadModule coldfusion_module
    modules/ApacheModuleColdFusion.dll
```

4. Restart the Apache Web server.

Solaris Installation

ColdFusion is distributed as a Solaris package file. You can use a number of Solaris package utilities, such as `pkgadd`, `pkgrm`, and `pkginfo` to manage the ColdFusion package file, as well as any other package files on your system.

Before installing ColdFusion please note the following considerations:

- You must be logged in as root to run the `pkgadd` utility.
- There are three versions of the netscape NSAPI plugin. The installation script will ask you which version of Netscape you are running if you choose to automatically configure a Netscape server.
- By default the package file installs ColdFusion into the `/opt` directory. If you want to install ColdFusion into a different directory, you must create that directory before running the `pkgadd` utility.
- Note the location and version number of your installed web server. The `pkgadd` utility prompts you for this information.

To install ColdFusion on Solaris:

1. Load the ColdFusion CD-ROM into your CD drive. If the Solaris Volume Manager is active, you won't need to mount the CD.
2. If necessary, mount the CD-ROM on `/cdrom/cdrom0`.
3. Use the following shell command to start the installation process:

```
pkgadd -d /cdrom/cdrom0
```

Solaris returns the names of the package files on the specified volume. Select the ColdFusion package file.

During the installation, ColdFusion services are started and a shell script is invoked to restart your Web server. If you are using Netscape, the installation will try and run the stop and start scripts to restart your web server.

ColdFusion can also be started using the ColdFusion start-up script, which is found in the `/etc/inet.d` directory.

Netscape and Solaris 2.6

Netscape does not certify Enterprise Server 3.0 or FastTrack Server 3.01 to work on Solaris 2.6. These servers are certified by Netscape to work on Solaris 2.5 or 2.5.1.

Note **ColdFusion requires Solaris 2.5.1 or higher.**

Netscape Enterprise 3.5.1 has been certified for Solaris 2.6. Several NSAPI functions, which didn't work with ColdFusion in earlier releases of Netscape Enterprise server, have been fixed in Enterprise 3.5.1.

Allaire has successfully tested Enterprise/FastTrack 2.01 and Enterprise 3.5.1 on Solaris 2.6 with ColdFusion.

Upgrading from ColdFusion 3.1.x: Solaris

When upgrading from ColdFusion 3.1 you may need to edit the package installation defaults file:

```
/var/sadm/install/admin/default
```

This file controls the behavior of the Solaris packaging commands. The value of the instance setting may need to be changed from `quit` or `unique` to `overwrite`.

Setting the value of the instance to `overwrite` allows the existing version of ColdFusion to be overwritten by the new version. See the admin(4) manual page for further information.

Uninstalling ColdFusion: Solaris

Note that the uninstall script removes your `odbc.ini` file. You may want to preserve your `odbc.ini` file by saving it to a new location. In addition, uninstalling ColdFusion removes all Verity collections, stored by default in `<installdir>/coldfusion/verity`, from your system.

Use the `pkgrm` utility to remove ColdFusion from your system as follows:

```
pkgrm cfusion
```

Updates to Netscape Configuration Files: Solaris

The ClusterCATS installation process updates the Netscape `magnus.conf` and `obj.conf` configuration files, as shown below:

obj.conf

Sections changed by BrightTiger are surrounded by start and end BT comment blocks.

```
# Netscape Communications Corporation - obj.conf
# You can edit this file, but comments and formatting changes
# might be lost when the admin server makes changes.

#Start BT:  Add Init routines and Entry points
Init fn=load-modules shlib="<ClusterCATS Install Dir>/btcats/
teserver.so"
funcs="btcats_server_init,btcats_nsapi_AuthTrans,btcats_nsapi_NameTrans,
btcats_ErrorFixup"
Init fnInit fn="btcats_server_init"
#End BT:

Init fn="flex-init" access="/local1/netscape/https-skagway/logs/access"
format.access="%Ses->client.ip% - %Req->vars.auth-user% [%SYSDATE%]
\"%Req->reqpb.clf-request%\" %Req->srvhdrs.clf-status% %Req-
>srvhdrs.content-length%"
Init fn="load-types" mime-types="mime.types"
```

```
<Object name="default">
#Start BT: Entry Points
NameTrans fn=btcats_nsapi_NameTrans
AuthTrans fn=btcats_nsapi_AuthTrans
Error fn=btcats_ErrorFixup reason="BrightTiger"
#End BT:

NameTrans fn="pfx2dir" from="/ns-icons" dir="/local1/netscape/ns-icons"
NameTrans fn="pfx2dir" from="/mc-icons" dir="/local1/netscape/ns-icons"
NameTrans fn="document-root" address="192.168.0.220" root="/local1/
netscape/root-wrangle"
NameTrans fn="document-root" address="192.168.0.173" root="/local1/
netscape/root-kodiak"
NameTrans fn="document-root" root="/local1/netscape/root-skagway"

PathCheck fn="unix-uri-clean"
PathCheck fn="check-acl" acl="default"
PathCheck fn="find-pathinfo"
PathCheck fn="find-index" index-names="index.html,home.html"
ObjectType fn="type-by-extension"
ObjectType fn="force-type" type="text/plain"
Service method="(GET|HEAD)" type="magnus-internal/imagemap"
fn="imagemap"
Service method="(GET|HEAD)" type="magnus-internal/directory" fn="index-
common"
Service method="(GET|HEAD)" type="*~magnus-internal/*" fn="send-file"
AddLog fn="flex-log" name="access"
</Object>

<Object name="cgi">
ObjectType fn="force-type" type="magnus-internal/cgi"
Service fn="send-cgi"
</Object>
```

magnus.conf

Sections changed by BrightTiger are surrounded by start and end BT comment blocks.

```
ServerRoot /local1/netscape/https-skagway
ServerID https-skagway
ServerName skagway.brighttiger.com
#Start BT:  Remove Address line from magnus.conf.  This will cause
#the Netscape Server to listen on INADDR_ANY
#Address 192.168.0.24
#End BT:
Port 80
LoadObjects obj.conf
RootObject default
ErrorLog /local1/netscape/https-skagway/logs/errors
PidLog /local1/netscape/https-skagway/logs/pid
User root
MtaHost localhost
NntpHost name-of-news-server
```

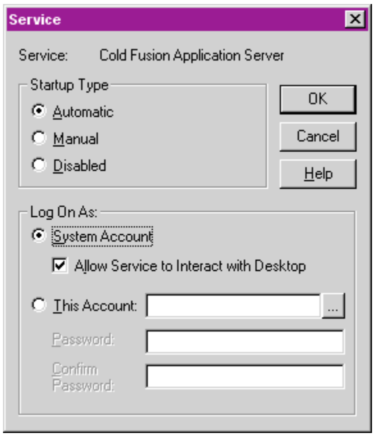
```
DNS off
Security off
Ciphers +rc4,+rc4export,+rc2,+rc2export,+des,+desede3
SSL3Ciphers
+rsa_rc4_128_md5,+rsa_3des_sha,+rsa_des_sha,+rsa_rc4_40_md5,+rsa_rc2_40_
md5,-rsa_null_md5
ACLFile /local1/netscape/httpacl/generated.https-skagway.acl
ClientLanguage en
AdminLanguage en
DefaultLanguage en
AcceptLanguage off
RqThrottle 128
```

Changing the ColdFusion User Account (Windows NT)

By default, ColdFusion runs under the host System account. However, running under this account, ColdFusion application pages may not have the access rights needed to interact with remote data sources, other application pages, COM objects, and so on. For this reason, you may need to run ColdFusion under a specific account created for that purpose.

To change the ColdFusion user account:

1. Open the Services Control Panel and select the ColdFusion Application Server service.
2. Click the Startup button to open the Service dialog. Use this dialog to set startup options for the ColdFusion Application Server service.



Service: Cold Fusion Application Server

Startup Type
[✓ Automatic]
Manual
Disabled

Log On As:
[✓ System Account]
[✓ Allow Service to Interact with Desktop]
This Account:
Password:
Confirm Password:

3. In the Log On As section, click the This Account radio button and enter the necessary account information.

Configuring the Netscape Web Server for Solaris

The ColdFusion NSAPI plugin has been tested with Netscape Enterprise Web Server versions 2.01 and 3.0, and 3.5.1, and the Netscape FastTrack Web Server 2.01. The installation script prompts you to automatically configure either of these servers. If you do not want the script to configure the Netscape plugin, you can follow procedures in this section to configure your Netscape server manually.

Choosing the right plugin version

Netscape plugins for ColdFusion are installed in the `<installdir>/coldfusion/webserver/nsapi` directory. Use the following table to select the proper plugin file.

ColdFusion Plugins for Netscape

Use this Plugin	With this Netscape Server Product	With this Version of Solaris
coldfusion2x.so	Netscape Enterprise Server 2.01 Netscape FastTrack 2.01	2.51 2.6
coldfusion3x.so	Netscape SuiteSpot Enterprise Server 3.0 Netscape FastTrack Server 3.0 Netscape Enterprise Server 3.51	2.51
coldfusion35.so	Netscape Enterprise Server 3.51	2.6

Note The Netscape 3.0 servers will not work on Solaris 2.6 with ColdFusion. You should use version 3.51, which Netscape certifies for use on Solaris 2.6.

The following procedure assumes you have the Enterprise 3.0 server installed in `/usr/netscape/suitespot` on the machine named `smurf`.

To configure the Netscape plugin:

1. Copy the appropriate plugin (see table above) to a directory in your server directories. Name it `coldfusion.so`.

   ```
   mkdir /usr/netscape/suitespot/plugins/coldfusion
   cp /opt/coldfusion/webserver/nsapi/coldfusion3x.so \
       /usr/netscape/suitespot/plugins/coldfusion/coldfusion.so
   ```

2. Edit the `/usr/netscape/suitespot/https-smurf/config/mime.types` file to add a new ColdFusion type. Add the line:

   ```
   type=magnus-internal/cold-fusion exts=cfm,dbm
   ```

3. Edit the `/usr/netscape/suitespot/https-surf/config/obj.confobj.conf` file to add new service and init directives. Add the following all on one line:

```
Init fn="load-modules" shlib="/usr/netscape/suitespot/
plugins/coldfusion/coldfusion.so" funcs="DoCFRequest"
```

4. Add the following line in the default Object:

   ```
   Service fn="DoCFRequest" method="(GET|POST)"
   type="magnus-internal/cold-fusion"
   ```

5. Stop and re-start the web server.

   If you run the Netscape Server Manager (Netscape's browser based administrator) you may get a warning about edits to the Netscape configuration files. This is normal. Follow the instructions and click on the Apply button to reload the new configuration files.

6. Test the setup by accessing the ColdFusion Administrator pages:

   ```
   http://localhost/CFIDE/administrator/index.cfm
   ```

If your server does not restart, check to make sure the pathname for the plugin you specified in the `obj.conf` file is correct for your installation and that it is on a single line.

Configuring the Apache Web Server for Solaris

ColdFusion has been tested with Apache version 1.2.x and 1.3. To obtain Apache free of charge, go to the Apache group web site at `http://www.apache.org`.

You can build the ColdFusion module into your Apache web server binary. This method is a very efficient method, since the server does not have to start a new process for every ColdFusion request.

Adding the ColdFusion module to Apache 1.2.x

The ColdFusion module can be found in the installation directory (usually `/opt`) under the `coldfusion/webserver/apache` directory.

In the following procedure, it is assumed that your Apache installation is found in `/usr/local/etc/httpd` and you installed ColdFusion in `/opt`.

To add the ColdFusion module:

1. Copy the module (`mod_coldfusion.a`) to your Apache source directory.

   ```
   cp /opt/coldfusion/webserver/apache/mod_coldfusion.a \
       /usr/local/etc/httpd/src/mod_coldfusion.a
   ```

2. Edit the `/usr/local/etc/httpd/src/Configuration` file to contain the following line:

   ```
   Module coldfusion_module mod_coldfusion.a
   ```

3. If you are using the Sun C compiler, edit the `EXTRA_LIBS=` line to include the C++ library:

   ```
   EXTRA_LIBS=-lC
   ```

If you are using the gcc compiler, you should add the absolute path of the C++ library to this line. You should include the version number of the library and use the highest one available.

For example:

```
EXTRA_LIBS=/usr/lib/libC.so.5
```

4. Run the Configure script to regenerate the configuration.

```
./Configure
```

5. Run `make` to build a new Apache httpd executable.
6. Install the new httpd executable in your installation directory and restart httpd.

You should now be able to access the ColdFusion Administrator with the following URL:

```
http://localhost/CFIDE/administrator/index.cfm
```

Adding the ColdFusion module to Apache 1.3.1

Allaire ColdFusion includes a shared object or file that Apache 1.3 can load at startup time. In order to do this, you must have configured and built Apache with the `mod_so` module. This module is not built into Apache by default. Consult the Apache documentation for details, but to configure this module in to the Apache build, you can run:

```
$ ./configure --enable-module=so <other apache options>
   $ make
   $ make install
```

Once you have `mod_so` configured into your Apache binary, you can then follow these steps:

1. Copy the ColdFusion module to the Apache modules directory:

```
cp /opt/coldfusion/webserver/apache/mod_coldfusion.so \
    /usr/local/apache/libexec
```

2. Edit your `httpd.conf` file to include the following directive:

```
LoadModule coldfusion_module libexec/mod_coldfusion.so
```

3. Restart Apache.

Using Scripts to Start and Stop ColdFusion (Solaris)

Two scripts are provided to start and stop ColdFusion processes:

```
<installdir>/colfusion/bin/start
<installdir>/colfusion/bin/stop
```

Note You must be logged in with root privileges to run these scripts.

In addition, the ColdFusion installation installs the following scripts to start and stop ColdFusion during system boot and shutdown:

```
/etc/init.d/coldfusion
/etc/rc1.d/K19coldfusion
/etc/rc3.d/S25coldfusion
```

ColdFusion runs the following processes on the system:

- `cfexec` — Starts/stops the other processes and manages page scheduling
- `cfserver` — The application server process
- `cfrdsservice` — Provides system support for the Administrator as well as security and debugging services for ColdFusion Studio

In addition, the following processes run if you've installed the ClusterCATS for ColdFusion option:

- `ipaliasd` — Provides IP failover capability for ColdFusion Application Server
- `dbeng50` — Provides Database services for clustering ColdFusion servers
- `reqmgr` — Processes ClusterCATS operations as root

Refer to ../../Administering_the_App_Server/contents.htm*Administering ColdFusion Server*./a

CHAPTER 3

ColdFusion Product Overview

This chapter provides a high-level summary of the ColdFusion web application server architecture, and features.

Contents

General Description

First released in 1995, ColdFusion has become the leading cross-platform Web application server. ColdFusion was the first Web application server on Windows NT, and as a pioneer in the application server market, ColdFusion has established a broad base of support with thousands of customers from small consulting groups to multinational corporations. Used by almost half of the Fortune 500, ColdFusion provides the fastest way to build and deploy scalable Web applications that integrate browser, server and database technologies.

The ColdFusion 4.0 Web application server includes all the technology that development teams need to quickly create and deliver open, scalable applications. The application server includes a powerful server integrated development environment (IDE), remote administration tools, and extensions that can connect to new technologies, legacy systems or business logic.

The latest release, version 4.0, enhances development productivity, scalability, integration and security. New features include interactive debugging, clustering, native database drivers and advanced security. As with previous releases, ColdFusion 4.0 maintains a commitment to rapid development and ease-of-use, while adding advanced functionality to support complex, large-volume applications.

ColdFusion 4.0 Application Server Components

ColdFusion Server — A high-performance, scalable, open platform for delivering Web applications that range from simple database driven pages to full e-commerce solutions on intranets, extranets and the Internet.

ColdFusion Studio — An integrated development environment with an array of highly-productive visual tools for creating ColdFusion applications, including the award winning Allaire HomeSite HTML editor.

ColdFusion Administrator — A complete management console for application, application sever and server cluster administration.

ColdFusion Extensions (CFX) — An open XML-based framework for extending ColdFusion with new server components and connectivity to enterprise systems using COM, CORBA, C/C++, VBScript, JavaScript or CFML.

Meeting the Development Challenges

ColdFusion was designed from the ground up to leverage the unique characteristics of the Web platform. It gives IT organizations the technology to overcome development challenges and build a new generation of business systems for online commerce, process automation, information publishing and an array of other applications.

Rapid Development

Meeting the challenge of increased demand for productivity, ColdFusion includes a range of innovative technologies for rapid Web application development.

Integrated Development Environment

ColdFusion optimizes developer productivity with a complete rapid application development (RAD) environment that includes visual programming, database and debugging tools. Based on the award-winning HTML authoring technology in Allaire HomeSite, ColdFusion Studio provides a suite of visual tools specifically designed to enhance application development.

Team Development Services

ColdFusion enables team development in large projects with developers and servers distributed across multiple locations. ColdFusion Studio supports shared project management and server-side source control. Based on user accounts, ColdFusion Studio clients can access databases and files on remote ColdFusion Servers via HTTP with SSL encryption, so team development against hosted servers is straightforward to configure and manage.

Tag-based server scripting

ColdFusion uses a powerful, comprehensive server-side scripting environment with an easy-to-learn tag-based syntax that cleanly integrates with HTML and Extensible Markup Language (XML). The ColdFusion Markup Language (CFML) creates a complete environment for building page-based applications that use COM or CORBA components.

Scalable deployment

Answering the need to handle large user volumes, ColdFusion delivers applications that can scale to handle the most demanding Internet or intranet sites.

High performance application delivery

ColdFusion Server runs as a high-performance multi-threaded service. Advanced features such as just-in-time page compilation and database connection caching, make ColdFusion an effective platform for delivering high-performance Web applications.

Server clustering

With the release of version 4.0, ColdFusion Server now offers native support for deploying multi-server clusters with dynamic load balancing and automatic server fail over. Clusters give ColdFusion the ability to scale to meet the needs of the most demanding sites.

Flexible server administration

ColdFusion 4.0 includes a browser-based Administrator for the server and Windows management console for clusters. ColdFusion Server publishes important real-time performance statistics and a number of logs that support the process of server tuning and management. All of the administrative tools can be deployed remotely.

Open integration

Overcoming the challenges created by technology chaos in the Internet market, ColdFusion tightly integrates with critical Internet and enterprise technologies across platforms to enhance development productivity.

Advanced Database Connectivity

ColdFusion offers the most advanced database development features available for creating robust database applications. The application server supports connectivity through ODBC, OLE-DB and native database drivers. Remote database development tools make building queries and database applications quick and easy.

Internet Technology Integration

ColdFusion 4.0 is tightly integrated with the full range of Internet protocols and technologies making it straightforward to use these technologies in applications. The application server includes native integration with e-mail servers, directories, file servers, distributed Web servers and object middleware including COM and CORBA. Extensions are available to connect to popular online payment servers and other technologies.

Enterprise Extensibility

ColdFusion is extensible with a wide range of technologies, including enterprise object standards, to support new functionality or connect to legacy systems. New components can be built with COM, CORBA, C/C++, VBScript, JavaScript and CFML.

Complete security

Confronting security risks, ColdFusion provides security on every level from development through deployment, accelerating the process of creating and delivering applications.

Secure development

ColdFusion 4.0 team development services provide the infrastructure for secure remote development across intranets and extranets. With the ColdFusion Remote Development Services, server administrators can create developer accounts using existing NOS authentication. Then developers can access files and databases through ColdFusion Studio over HTTP.

Secure deployment

ColdFusion 4.0 provides a complete set of services for building and deploying highly secure applications on intranets, extranets and the Internet. Server administrators can control security at run-time and developers can leverage the security services for authentication and access control within applications. Using server sandbox security, companies can safely host multiple applications on the same server.

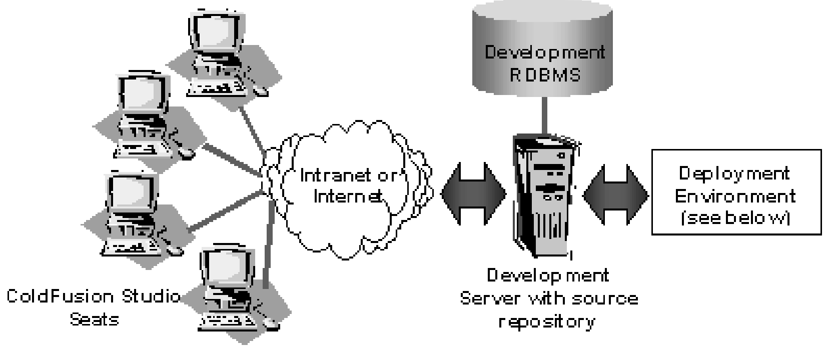
ColdFusion Technical Overview

Development architecture

Using the visual database and programming tools in ColdFusion Studio, developers create applications by building pages that combine the ColdFusion tag-based server scripting language, the ColdFusion Markup Language (CFML), with HTML and other Web technologies. The applications are then delivered with ColdFusion Server.

ColdFusion Studio includes the complete array of visual tools for building applications. The basic editor is based on the award winning HTML authoring tool, Allaire HomeSite. This editor supports visual page layout, color coded editing, sophisticated find and replace and a host of other productivity-enhancing features. In addition to the basic editor, ColdFusion Studio includes visual database tools, integration with source control, support for remote development and an interactive debugger.

Each copy of ColdFusion Studio includes a single-user version of ColdFusion Server so individual developers can create applications at their desktop. But most ColdFusion development projects are done in a team environment, so ColdFusion supports a set of team development services that offer remote development against a ColdFusion Server configured for development.


Development RDBMS
Intranet or Internet
Deployment Environment (see below)
ColdFusion Studio Seats
Development Server with source repository

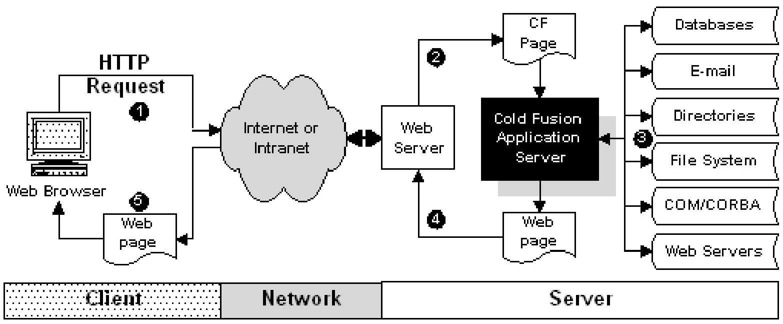
Unlike Web development systems based on legacy client/server technology, ColdFusion uses a uniquely Web-centric application architecture. Like a static Web site, a ColdFusion application is a collection of pages and components. But unlike a static Web site, the pages in a ColdFusion application are scripted with a tag-based server scripting language, CFML.

Written in ColdFusion Studio, and entirely processed on the server, CFML controls application logic, backend integration and dynamic page generation. Its tag-based syntax makes it the ideal environment for building applications that use HTML and XML as well as other Web standards. With more than 60 tags and 200 functions, CFML is the most advanced language available for server-side Web application scripting.

New developers will find CFML's clean syntax and tight integration with HTML easy-to-learn and use. At the same time, experienced developers will be able to take advantage of the productivity gain as well as the advanced features like structured exception handling, regular expressions, and the easy extensibility with COM, CORBA and C/C++.

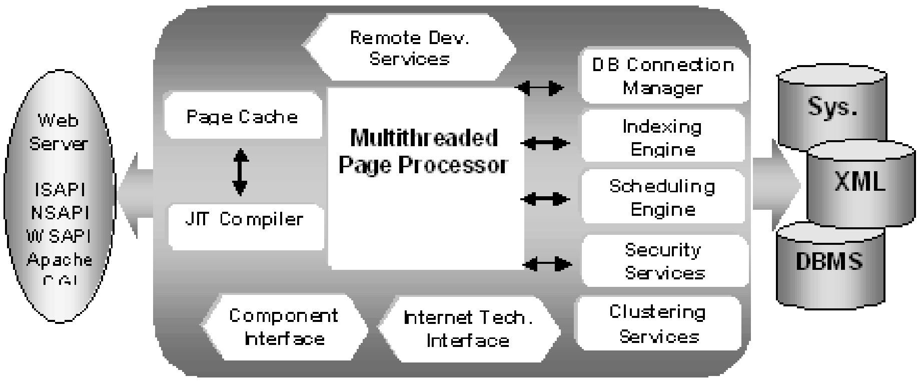
Deployment architecture

ColdFusion Web applications are delivered with ColdFusion Server. When a page in a ColdFusion application is requested by a browser, ColdFusion Server processes the CFML, interacts with backend systems and dynamically generates a Web page that is returned to the browser. The diagram below shows how ColdFusion works when a Web browser invokes a ColdFusion page.



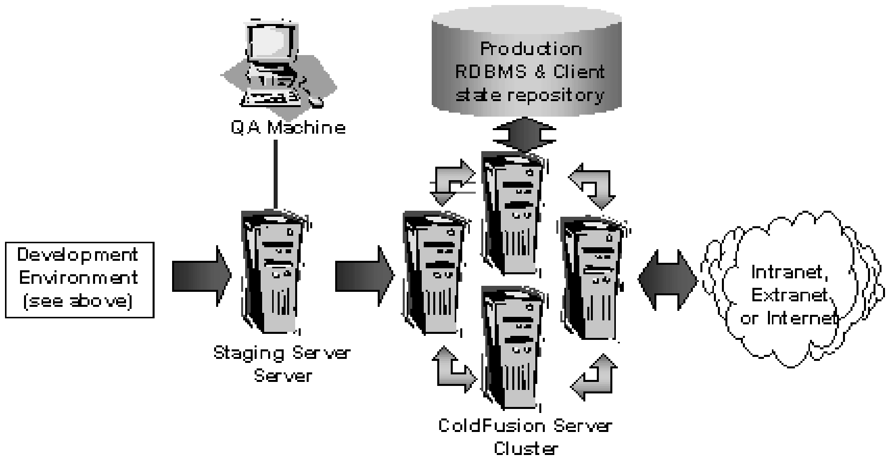
1. When a user clicks a “Submit” button on a form or a hypertext link on a page, the user’s Web browser sends an HTTP request to the Web server via the Internet or an intranet.
2. The Web server passes the data submitted by the client and the appropriate page to ColdFusion Server through a server API.
3. ColdFusion reads the data from the client and processes the CFML in the page. Based on the CFML, the server interacts with database servers, the file system, SMTP servers and potentially other applications and extensions through the ColdFusion API or through COM/DCOM.
4. ColdFusion dynamically generates a Web page that is returned to the Web server.
5. The Web server returns the HTML page to the user’s browser.

ColdFusion Server provides all of the necessary functionality for delivering Web applications. Each server includes a number of different services that handle page processing, security, state management, connection management, and a variety of other functions. Because this technology is provided by the application server, developers can focus on the problem of creating applications and easily take advantage of all the functionality already provided by the server.



Deploying Clusters

ColdFusion Server 4.0 offers native support for deploying applications in a clustered environment. A typical cluster deployment might use three to ten production servers to offer greater performance and availability for large scale e-commerce or enterprise applications.



When ColdFusion is deployed in a clustered environment end user requests are automatically distributed to the optimal server in the cluster based on load. If any server in the cluster fails, another server will automatically take over the IP address of the failed server, handling any new inbound requests and giving end-users high-availability.

Unlike other cluster technology, a ColdFusion cluster has no single point of failure. Any machine in the cluster can handle load balancing and fail over. Also, ColdFusion uses smart clusters, so every machine in the cluster is aware of the load being placed on the other ColdFusion Servers. As a result, load balancing is optimized to deliver the very best performance at the lowest cost. Finally, extending a ColdFusion cluster is as simple as adding a new server, and there is no limit to the number of servers a cluster will support.

CHAPTER 4

Quick Start to ColdFusion

Use this chapter to get a quick feel for creating ColdFusion application pages. Create a few simple application pages to get you started and then take a tour of a full-scale working email client example application installed with ColdFusion.

Contents

Creating Simple Dynamic Pages

Creating dynamic Web pages with ColdFusion is as straightforward as creating ordinary Web pages. The following sections will step you through two simple examples designed to introduce you to ColdFusion. If you're feeling more adventurous, take a look at the sections that deal with the example applications installed with ColdFusion in

Example 1: A simple dynamic page

This example shows how ColdFusion uses a form field variable entered into an HTML form.

Using ColdFusion Studio, create two files in the `\cfdocs` directory located in your Web server document root directory:

- `hellopage1.cfm`
- `hellopage2.cfm`

1. In the `hellopage1.cfm` file, type the following HTML and CFML, or copy and paste into a new document if you are viewing this online:

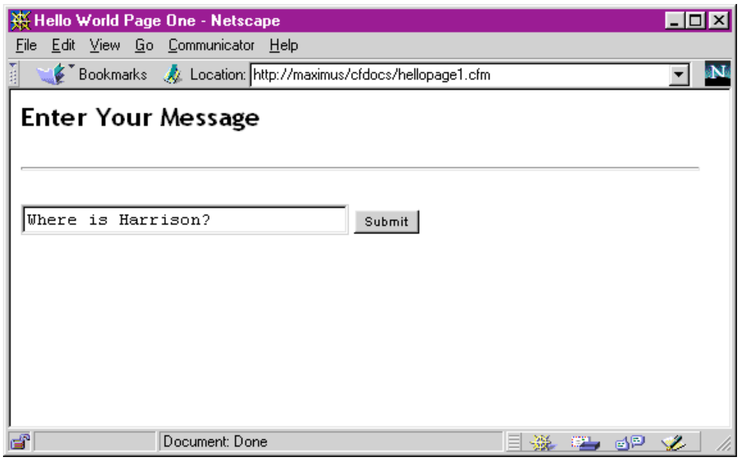
```
<HTML>
<HEAD>
<TITLE>Hello World Page One</TITLE>
</HEAD>
<BODY>

<H2>Enter Your Message</H2>
<HR>

<FORM ACTION="hellopage2.cfm" METHOD="post">
<INPUT TYPE="text" SIZE="30" NAME="HelloText">
<INPUT TYPE="submit" VALUE="Submit">
</FORM>

</BODY>
</HTML>
```

2. In the `hellopage2.cfm` file type the following HTML and CFML and save:

```
<HTML>
<HEAD>
<TITLE>Hello World Page Two</TITLE>
</HEAD>
<BODY>

<CFOUTPUT>
<H2>#Form.HelloText#</H2>
</CFOUTPUT>

</BODY>
</HTML>
```

3. Make sure that ColdFusion and your Web server are running.
4. Open a Web browser and enter the following URL to open your new ColdFusion page:

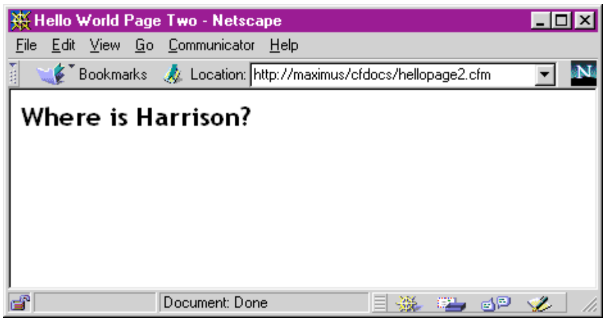
   `http://127.0.0.1/cfdocs/hellopage1.cfm`


Hello World Page One - Netscape
File Edit View Go Communicator Help
Bookmarks
Location: http://maximus/cfdocs/hellopage1.cfm

Enter Your Message

Where is Harrison?

5. In your browser, enter text into the form and click Submit. The page that appears displays the text you just entered.


Hello World Page Two - Netscape
File Edit View Go Communicator Help
Bookmarks
Location: http://maximus/cfdocs/hellopage2.cfm

Where is Harrison?

Explanation

When the form was submitted in the first page, a variable was created (Form.HelloText) and passed to the second page where it was resolved and displayed.

Example 2: A database-driven page

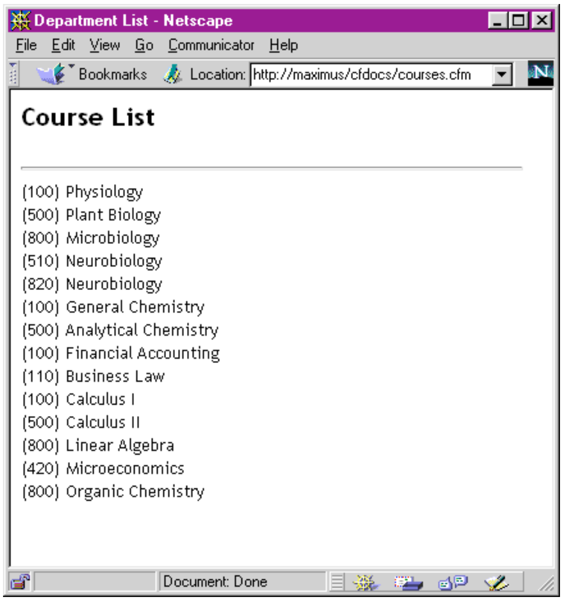
The following example shows you how to create an application page using data from the ColdFusion cfsnippets data source.

1. Create a file called `courses.cfm` in the `\cfdocs\` directory located in your Web server root document directory.
2. Type the following HTML and CFML in the `courses.cfm` file and save:

   ```
   <!--- Start of Database Query --->
   <CFQUERY NAME= "CourseList" DATASOURCE= "cfsnippets">
   SELECT * FROM CourseList
   </CFQUERY>
   <!--- End of Query --->

   <HTML>
   <HEAD>
   <TITLE>Department List</TITLE>
   </HEAD>

   <BODY>

   <H2>Course List</H2>
   <HR>

   <!--- Start of Output Block --->
   <CFOUTPUT QUERY="CourseList">
   (#CourseNumber#)</B> #CourseName#<BR>
   </CFOUTPUT>
   <!--- End of Output Block --->
   </BODY>
   </HTML>
   ```

   Note the use of the comment convention used in ColdFusion pages: `<!---` and `--->`. ColdFusion comments use a form similar to HTML comments, but with an additional hyphen.
3. Make sure that ColdFusion and your Web server are running.
4. Open a Web browser and enter the following URL to open your new application page:

   `http://127.0.0.1/cfdocs/courses.cfm`
5. The page that appears shows a list of course numbers and names.


Department List - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://maximus/cfdocs/courses.cfm
Document: Done

Course List

(100) Physiology
(500) Plant Biology
(800) Microbiology
(510) Neurobiology
(820) Neurobiology
(100) General Chemistry
(500) Analytical Chemistry
(100) Financial Accounting
(110) Business Law
(100) Calculus I
(500) Calculus II
(800) Linear Algebra
(420) Microeconomics
(800) Organic Chemistry

Explanation

In this example, the CFQUERY tag used the SQL SELECT statement to retrieve all the data from the CourseList table. The CFOUPUT tag then referenced the CourseNumber and CourseName columns returned by the query, for display.

Once again, here's the query part:

```
<!--- Start of Database Query --->
<CFQUERY NAME= "CourseList" DATASOURCE= "cfsnippets">
SELECT * FROM CourseList
</CFQUERY>
<!--- End of Query --->
```

Here's the output part:

```
<CFOUTPUT QUERY="CourseList">
(#CourseNumber#)</B> #CourseName#<BR>
</CFOUTPUT>
```

ColdFusion Example Applications

You can learn a lot more about ColdFusion by taking a look at the example applications installed with ColdFusion Server. Four fully functional example applications are installed that you can raid for working code. The examples are as follows:

- A simple personnel management application. Allows you to search and view a list of employee names or sort by department. Use the ColdFusion pages in this application to learn basic techniques used for database transactions, as well as other ColdFusion fundamentals.
- A full-featured POP3 email client. Use this application to actually manage your email. Or, better yet, check out how simple it is using CFML to integrate POP and SMTP services in Web applications. Slightly more advanced, shows some basic error handling techniques.
- An all-but-live ecommerce application pitching windsurfing hardware. Includes a behind-the-scenes facility for viewing CFML code. An advanced ColdFusion application, shows,
- A sophisticated content management application modeled after the content management system that hosts the Allaire Web site.

To run any of the example applications, open the Example application home page in your `cfdocs` directory. By default this is:

`hostname``/cfdocs/examplehome.htm`

The following sections provide a detailed walk-through of the email client application so you can see exactly how ColdFusion application pages fit together and how discreet blocks of ColdFusion Markup Language (CFML) code work.

Crazy Cab Example Application

The Crazy Cab example application is a full featured POP3 email client you can use to manage email messages. In addition to basic email handling features, it allows you to attach external files to email messages. In the pages that follow, we step through the ColdFusion code that comprises the Crazy Cab application.

In addition to Crazy Cab, be sure to check out the other example applications (still being developed at the time of this writing) installed with ColdFusion.

Note The example code that follows consists of selections from the relevant application pages, excluding as much HTML code, such as FONT tags as necessary to make the code as readable as possible. If you want to see the actual pages, open any of the Crazy Cab pages, installed in `cfdocs/exampleapp/email/*.cfm`, in ColdFusion Studio for closer inspection.

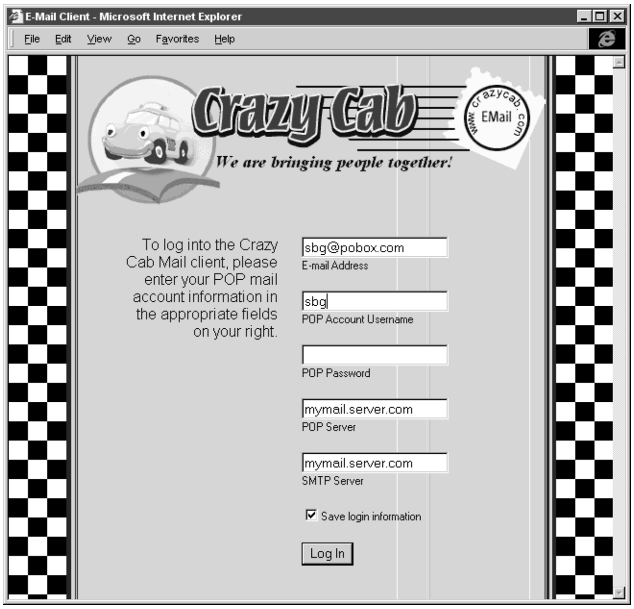
To launch Crazy Cab:

1. Open the Example Application home page:

   `http://`*`servername`*`/cfdocs/examplehome.htm`

   Where *servername* is the name of the server where ColdFusion Server is installed.
2. Click the Crazy Cab link. The login page for Crazy Cab appears, which prompts you for your email address, mail server name, and so on. Enter the required

information and click OK. If you have any email waiting for you, it appears in the message listing.


E-Mail Client - Microsoft Internet Explorer
File Edit View Go Favorites Help
Crazy Cab
We are bringing people together!
www.crazycab.com EMail
Log In

To log into the Crazy Cab Mail client, please enter your POP mail account information in the appropriate fields on your right.

sbg@pobox.com
E-mail Address

sbg
POP Account Username

POP Password

mymail.server.com
POP Server

mymail.server.com
SMTP Server

☑ Save login information

Crazy Cab is a relatively simple ColdFusion application that consists of only 15 ColdFusion pages and a handful of graphics files. To start with, we'll take a look at the login page, `login.cfm`, which checks to see if a user has already entered Crazy Cab email account information.

The login page: What happens first

To start with in `login.cfm`, the ColdFusion CFINCLUDE tag is used to reference the `_header.cfm` application page, which encapsulates a set of graphics files used for this and other pages in Crazy Cab.

```
<CFINCLUDE TEMPLATE="_header.cfm">
```

Next, a CFLOOP block is employed to iterate processing through a list of items, in this case, a set of form variables used to store information about you and your email account. If you entered this information once before in Crazy Cab and chose the Save login information option, this information is stored as a cookie on your system. If ColdFusion finds this cookie when you return, it populates the form with the information you provided. You'll see how ColdFusion manages cookies in Crazy Cab later on.

```
<CFLOOP LIST="Email,Username,POPserver,SMTPserver"
INDEX="CurrItem">
    <CFIF ISDEFINED("Cookie.#CurrItem#")>
        <CFSET "#CurrItem#" = EVALUATE("Cookie." & CURRITEM)>
    <CFELSE>
```

```
        <CFSET "#CurrItem#" = "">
    </CFIF>
</CFLOOP>
```

The CFLOOP block loops over the specified formfield variables. For each one, if a value is found in a cookie variable, ColdFusion returns the value of that item. If a value is not found, ColdFusion creates an empty string for the CurrItem variable. If the cookie exists, the individual values are used to populate the login form.

```
<CFIF ISDEFINED("Cookie.SaveInfo")>
    <CFSET CHECKED = "CHECKED">
<CFELSE>
    <CFSET CHECKED = "">
</CFIF>
```

The IF/ELSE block above does the following: If the cookie variable Cookie.SaveInfo exists, it creates a variable called CHECKED with the value "CHECKED," otherwise, it creates the variable but gives it an empty string value. The SaveInfo cookie variable is used in the form to indicate whether the form information should be saved or not.

The login form: login.cfm

This part of `login.cfm` is the HTML form used to collect information about your email account. Note how the form field variables are referenced in the CFLOOP and CFIF statements above. Note also that the form data is submitted to `auth.cfm`, which we'll examine momentarily.

```
<FORM ACTION="auth.cfm" METHOD="POST">

<CFOUTPUT>

<P><INPUT TYPE="TEXT" NAME="Email" VALUE="#Email#"><BR>
Email Address</P>

<P><INPUT TYPE="TEXT" NAME="Username" VALUE="#Username#"><BR>
POP Account Username</P>

<P><INPUT TYPE="PASSWORD" NAME="Password"><BR>
POP Password</P>

<P><INPUT TYPE="TEXT" NAME="POPserver" VALUE="#POPserver#"><BR>
POP Server</P>

<P><INPUT TYPE="TEXT" NAME="SMTPserver" VALUE="#SMTPserver#"><BR>
SMTP Server</P>

<P><INPUT TYPE="CHECKBOX" NAME="SaveInfo" #CHECKED#>
Save login information</P>

</CFOUTPUT>

<P><INPUT TYPE="SUBMIT" VALUE="Log In"></P>

</FORM>
```

```
</TD>
</TR>
</TABLE>
```

To view complete page code listings, see the files contained in the `cfdocs/exampleapp/email` directory on your ColdFusion application server.

Creating cookies: auth.cfm

When the form in the `login.cfm` page is submitted, form variables are created and passed to the `auth.cfm` page. Let’s take a look at what happens next.

Here’s the first part of `auth.cfm`. Notice the reference to the _header.cfm page in the CFINCLUDE tag.

```
CFINCLUDE TEMPLATE="_header.cfm">
```

Recall that it is used to embed a series of graphics at the top of every Crazy Cab page.

Next, a series of CFSET tags, which are used to create ColdFusion variables, change the form variables passed from the form in `login.cfm` into session variables. Session variables are useful here because they persist for the entire user session; they don’t need to be recreated for other pages in the Crazy Cab application.

```
<CFSET Session.Email = Form.Email>
<CFSET Session.Username = Form.Username>
<CFSET Session.Password = Form.Password>
<CFSET Session.POPserver = Form.POPserver>
<CFSET Session.SMTPserver = Form.SMTPserver>
```

Next, CFIF, CFCOOKIE, and CFLOOP tags are used to find out if the SaveInfo variable was created. SaveInfo is created only if you check the Save login information box on the `login.cfm` page. The CFCOOKIE tag creates the cookie based on the value of the Expires variable and populates it with information submitted in the `login.cfm` form.

```
<CFIF IsDefined("Form.SaveInfo")>
    <CFSET Expires = "NEVER">
<CFELSE>
    <CFSET Expires = "NOW">
</CFIF>

<CFCOOKIE NAME="SaveInfo" VALUE="TRUE" EXPIRES="#Expires#">

<CFLOOP LIST="Email,Username,POPserver,SMTPserver"
    INDEX="CurrItem">

    <CFCOOKIE NAME="#CurrItem#"
        VALUE="#evaluate('Form.' & CurrItem)#"
        EXPIRES="#Expires#">

</CFLOOP>
```

Retrieving mail

While the upper part of auth.cfm is dealing with cookies, a small JavaScript is used to load `refresh.cfm`, which does the job of retrieving email messages. Text in the page lets the user know that mail is being retrieved from the POP server.

```
<FONT FACE="Helvetica" SIZE="-1"><B>Attempting to retrieve
messages.<BR>
This may take a few moments.</B></FONT>

<SCRIPT LANGUAGE="JavaScript">
location.replace('refresh.cfm')
</SCRIPT>
```

Populating the list of messages: refresh.cfm

The `refresh.cfm` page referenced in the JavaScript in `auth.cfm` is responsible for actually retrieving mail messages and checking for errors.

First, a CFTRY block is opened, inside of which CFPOP attempts to retrieve email messages.

```
<CFTRY>
<CFPOP ACTION="GETHEADERONLY"
    NAME="Messages"
    SERVER="#Session.POPserver#"
    TIMEOUT="120"
    USERNAME="#Session.Username#"
    PASSWORD="#Session.Password#">
```

If errors are encountered, CFCATCH is used to inform the user of any errors, such as invalid account information, or any other error generated by the POP mail server.

```
<CFCATCH TYPE="ANY">
    <CFINCLUDE TEMPLATE="_header.cfm">

    <P>Oops...!</P>

    <CFIF CFCatch.Detail CONTAINS "password">
        The username and/or password you supplied was invalid!
    <CFELSEIF CFCatch.Detail CONTAINS "timeout">
        Crazy Cab is tired of waiting for
        <CFOUTPUT>#Session.POPserver#</CFOUTPUT> to respond.
    <CFELSE>
        Crazy Cab has encountered the following error:</P>

<P>
        <TABLE BORDER="1"
            CELLPADDING="5"
            CELLSPACING="0"
            BGCOLOR="#FFFFCC">
        <TR>
        <TD><TT><CFOUTPUT>
            <B>#CFCatch.Message#</B>
            <BR><BR>#CFCatch.Detail#</CFOUTPUT>
        </TT></TD>
        </TR>
        </TABLE>
</P>
    </CFIF>

    <P>Please <A HREF="login.cfm">log in</A> again.

    <CFINCLUDE TEMPLATE="_footer.cfm">

<CFABORT>

</CFCATCH>

</CFTRY>
```

At the bottom of the `refresh.cfm` page a session variable is created to store the messages retrieved by CFPOP. The session.messages variable is only created following a successful message retrieval. Last, a CFLOCATION tag is used to open `messagelist.cfm`, which presents email messages in a list.

```
<CFSET Session.Messages = Messages>
<CFLOCATION URL="messagelist.cfm" ADDTOKEN="NO">
```

Listing email messages with messagelist.cfm

In `messagelist.cfm`, a CFIF block tests for the existence of the session.messages variable, and if not found, attempts to retrieve messages again by opening the refresh.cfm file.

Session variables allow ColdFusion to successfully resolve the session.username variable in the message letting the user know that messages are waiting to be read.

```
<CFINCLUDE TEMPLATE="_header.cfm">

<!--- Make sure messages are loaded --->
<CFIF NOT ISDEFINED("Session.Messages")>
    <CFLOCATION URL="refresh.cfm" ADDTOKEN="NO">
</CFIF>

<TABLE BORDER="0"
    CELLSPACING="2"
    CELLPADDING="25"
    ALIGN="CENTER"
    BGCOLOR="#660000">
<TR>
    <TD COLSPAN="3" BGCOLOR="FFffcc">
    <!--- Display login information --->
    <CFOUTPUT>
        <B>Hello #Session.Username#,</B><BR>
        Welcome to your Crazy Cab inbox. You have
        #Session.Messages.RecordCount#</B> new message(s) on
        <B>#Session.POPserver#</B> waiting to be read.
    </CFOUTPUT>
    </P>
    </TD>
</TR>
</TABLE>
```

Next, the email messages are listed. Since the page uses the HTML TABLE tag to position elements, it might be hard to see what's happening.

```
    <TABLE BORDER="0" CELLSPACING="0" CELLPADDING="0">
    <TR>
        <TD COLSPAN="5">

    <!-- Column headers -->
        <TABLE BORDER="0"
            CELLSPACING="2"
            CELLPADDING="3"
            ALIGN="CENTER"
            BGCOLOR="#660000">

        <TR>
            <TD BGCOLOR="660000"><B>From</B></TD>
            <TD BGCOLOR="660000"><B>Subject</B></TD>
            <TD BGCOLOR="660000"><B>Date</B></TD>
        </TR>

    <!-- Output messages -->
        <CFOUTPUT QUERY="Session.Messages">
```

In this output block, the CFOUTPUT tag uses the QUERY attribute, which references the session.messages variable. The data returned by CFPOP earlier was stored in the session.messages variable, and this data takes the form of a query object, which means it can be referenced by CFOUTPUT just like data returned by the CFQUERY tag. Individual elements of the messages, such as the To, From, CC, and Subject elements are treated just like columns returned in a query to a database. ColdFusion outputs your email messages using the table rows and session variable references to position each element.

Notice in the following block how the email message's From element is wrapped in an HTML A tag. This creates a link to the `viewmsg.cfm` ColdFusion page, which is used to show the message. The ? character is used to pass the unique message number in the URL indicating which message to open. Notice also, how the ColdFusion HTMLEditFormat function is used to format the email address of the sender. Without it, the browser would attempt to interpret the raw HTML.

```
    <TR>
        <TD BGCOLOR="FFffcc">
        <A HREF="viewmsg.cfm?Msg=#MessageNumber#">
        #HTMLEditFormat(From)#
        </A></TD>
```

This same technique is used to wrap the other message details, allowing the user to click on any part of the returned message information in order to view it. In this chunk, the ColdFusion Trim function is used to trim white space from the beginning and end of the message subject string. Also, if no subject exists, the string "(no subject)" replaces the Subject element of the message. Here is the message Subject chunk:

```
<TD BGCOLOR="FFffcc">
    <A HREF="viewmsg.cfm?Msg=#MessageNumber#">
    <CFIF TRIM(SUBJECT) IS "">
    (no subject)
    <CFELSE>#Subject#</CFIF>
    </A>
    </TD>
```

At the bottom of the page, graphic links allow you to refresh the message list by opening the `refresh.cfm` page. There is also a link you can use to compose a message using `compose.cfm`. See Composing a message: compose.cfm below, for details.

```
<TD BGCOLOR=CC9900 VALIGN="middle">
    <A HREF="compose.cfm">
    <IMG SRC="images/compose.gif"
        WIDTH=29
        HEIGHT=25
        BORDER=0
        ALT=""
        ALIGN="left">
    </A>
    <A HREF="compose.cfm">Compose</A></TD>

<TD BGCOLOR=660000>
    <A HREF="refresh.cfm">
    <IMG SRC="images/refresh.gif"
        WIDTH=25
        HEIGHT=24
        BORDER=0
        ALT=""
        ALIGN="left"></A>
    <A HREF="refresh.cfm">Refresh</A></TD>

</TR>
</TABLE>
```

Composing a message: compose.cfm

The Crazy Cab `compose.cfm` page is a simple HTML form with text input elements you use to enter message text, email addresses of recipients, and so on. At the bottom of the form, the HTML input type FILE allows you to select an attachment to send with the message.

```
<CFINCLUDE TEMPLATE="_header.cfm">

<FORM ACTION="send.cfm"
    METHOD="POST"
    ENCTYPE="multipart/form-data">

<CFOUTPUT>
```

```
<TABLE BORDER="0"
    CELLPADDING="3"
    CELLSPACING="0">
<TR>
    <TD ALIGN="RIGHT"><B>To</B></TD>
    <TD><INPUT TYPE="TEXT" NAME="To" SIZE="30"></TD>
</TR>

<TR>
    <TD ALIGN="RIGHT"><B>cc</B></TD>
    <TD><INPUT TYPE="TEXT" NAME="cc" SIZE="30"></TD>
</TR>

<TR>
    <TD ALIGN="RIGHT"><B>Subject</B></TD>
    <TD><INPUT TYPE="TEXT" NAME="Subject" SIZE="30"></TD>
</TR>
</TABLE>

<P><TEXTAREA NAME="MessageBody"
    COLS="50"
    ROWS="20"
    WRAP="VIRTUAL">
</TEXTAREA></P>

</CFOUTPUT>

<P><B>Attachment</B><BR>
<INPUT TYPE="FILE" NAME="AttachFile"></P>

<P><INPUT TYPE="SUBMIT" VALUE="Send Message"></P>

</FORM>

<CFINCLUDE TEMPLATE="_footer.cfm">
```

Viewing a message: viewmsg.cfm

Crazy Cab uses the `viewmsg.cfm` file for a number of purposes. To start, `viewmsg.cfm` creates a temporary directory for inbound file attachments:

```
CFSET TempDir = GetTempDirectory() & "CrazyCab\">
<CFTRY>
    <CFDIRECTORY ACTION="CREATE" DIRECTORY="#TempDir#">
    <CFCATCH TYPE="ANY">
    </CFCATCH>
</CFTRY>

<!--- Create inbound temp directory if it doesn’t already exist --->
<CFSET AttachDir = TempDir & "inbound\">
<CFTRY>
    <CFDIRECTORY ACTION="CREATE" DIRECTORY="#AttachDir#">
```

```
        <CFCATCH TYPE="ANY">
        </CFCATCH>
    </CFTRY>
```

Note the use of CFTRY and CFCATCH to trap errors that might occur during the attempt to create these directories. In the next block, the CFPOP tag is used to retrieve an individual message. It uses the value of the URL.Msg variable to select the correct message to display and a number of session variables to communicate with the POP server. Remember that when you click on an individual message in the message list page, a URL variable is passed to viewmsg.cfm identifying which message is desired.

```
<CFPOP ACTION="GETALL"
    NAME="Message"
    MESSAGENUMBER="#URL.Msg#"
    SERVER="#Session.POPserver#"
    USERNAME="#Session.Username#"
    PASSWORD="#Session.Password#"
    ATTACHMENTPATH="#AttachDir#">
```

Then, a custom tag is used, CF_WRAP to control line width and how lines are wrapped in the message output. The CFSET tag is used to define a variable to hold the contents of the message body. The variable is used as an attribute passed to the custom tag, which (as you'll see) returns the message body ready for output.

```
<CFSET Body = Message.Body>
<CF_Wrap VARIABLE="Body" WIDTH=50>
```

The message details, such as the From, To, CC, and Subject elements are then displayed, preceded by the option to show the full message header.

```
<CFIF IsDefined("URL.FullHeaders")>
    <CFOUTPUT>#HTMLCodeFormat(Message.Header)#</CFOUTPUT>
<CFELSE>

<!---Show me the message! --->
<CFOUTPUT QUERY="Message">
    <P><B>Date:</B> #Date#<BR>
    <B>From:</B> #HTMLEditFormat(From)#<BR>
    <B>To:</B> #HTMLEditFormat(To)#<BR>
    <B>cc:</B> #HTMLEditFormat(cc)#<BR>
    <B>Subject:</B> #HTMLEditFormat(Subject)#<BR>

<A HREF="viewmsg.cfm?Msg=#URL.Msg#&FullHeaders=On">
    Click here for full headers
</A></P>

</CFOUTPUT>
</CFIF>
```

Next, the message body is displayed, returned by the custom tag as a variable called Body. The HTMLCodeFormat function replaces HTML code with escaped characters, for example &gt; for the greater than sign (>) and encloses the message content in the <PRE> and </PRE> tags.

```
<CFOUTPUT>
```

```
<PRE>--------</PRE>
#HTMLCodeFormat(Body)#
</CFOUTPUT>
```

Crazy Cab summary

Although there are several other pages that perform additional email handling tasks, what you've seen here should give you a very clear notion of how ColdFusion interacts with a POP server.

The following list details what we've examined in the Crazy Cab example application:

- The use of CFSET to define variables.
- Several different variable scopes, that is, contexts in which a variable applies. For example, Crazy Cab uses form variables: Form.*variablename* and session variables: Session.*variablename*.
- The use of the CFPOP tag to retrieve email from a POP server and to retrieve individual messages.
- The manner in which ColdFusion uses form variables passed from an HTML form.
- The use of CFTRY/CFCATCH tags to deal with errors that might be passed from the POP server.