

macromedia®

COLDFUSION® MX 7

CFML Quick Reference



Trademarks

1 Step RoboPDF, ActiveEdit, ActiveTest, Authorware, Blue Sky Software, Blue Sky, Breeze, Breezo, Captivate, Central, ColdFusion, Contribute, Database Explorer, Director, Dreamweaver, Fireworks, Flash, FlashCast, FlashHelp, Flash Lite, FlashPaper, Flex, Flex Builder, Fontographer, FreeHand, Generator, HomeSite, JRun, MacRecorder, Macromedia, MXML, RoboEngine, RoboHelp, RoboInfo, RoboPDF, Roundtrip, Roundtrip HTML, Shockwave, SoundEdit, Studio MX, UltraDev, and WebHelp are either registered trademarks or trademarks of Macromedia, Inc. and may be registered in the United States or in other jurisdictions including internationally. Other product names, logos, designs, titles, words, or phrases mentioned within this publication may be trademarks, service marks, or trade names of Macromedia, Inc. or other entities and may be registered in certain jurisdictions including internationally.This product includes code licensed from RSA Data Security.

Third-Party Information

This guide contains links to third-party websites that are not under the control of Macromedia, and Macromedia is not responsible for the content on any linked site. If you access a third-party website mentioned in this guide, then you do so at your own risk. Macromedia provides these links only as a convenience, and the inclusion of the link does not imply that Macromedia endorses or accepts any responsibility for the content on those third-party sites.

Copyright © 2005 Macromedia, Inc. All rights reserved. U.S. Patents Pending. This manual may not be copied, photocopied, reproduced, translated, or converted to any electronic or machine-readable form in whole or in part without written approval from Macromedia, Inc. Notwithstanding the foregoing, the owner or authorized user of a valid copy of the software with which this manual was provided may print out one copy of this manual from an electronic version of this manual for the sole purpose of such owner or authorized user learning to use such software, provided that no part of this manual may be printed out, reproduced, distributed, resold, or transmitted for any other purposes, including, without limitation, commercial purposes, such as selling copies of this documentation or providing paid-for support services. Part Number ZCF70M300

Project Management: Randy Nielsen
Editing: Linda Adler

Production Management: Patrice O'Neill

Media Design and Production: John Francis, Adam Barnett

Special thanks to Sawako Gensure, Seungmin Lee, Takashi Koto, Nozomi Kugita, Masayo Noda, Hiroshi Okugawa, Bowne Global Solutions

First Edition: January 2005

Macromedia, Inc.
600 Townsend St.
San Francisco, CA 94103

CONTENTS

CFML tags 5
CFML functions **24**
Array functions 24
Conversion functions 24
Date and time functions 24
Decision functions 25
Display and formatting functions 26
Dynamic evaluation functions 26
Extensibility functions 26
International functions 27
List functions 27
Mathematical functions 28
Other functions 28
Query functions 28
Security functions 29
String functions 29
Structure functions 30
System functions 31
XML functions 31
Gateway interface **32**
Methods to implement 32
IM Gateway CFC incoming message methods 32
IM Gateway message sending commands 32
Application.cfc **32**
Methods to implement 32
ColdFusion variables **32**
Variable scope 32
Caller scope 33
Client variables 33
Server variables 33
Application and session variables 33
Custom tag variables **33**
Request variable 33
Form variable 34
ColdFusion tag-specific variables **34**
ColdFusion query variables 34
CFCATCH variables 34
CFDIRECTORY variables 35
CFERROR variables 35
CFFILE ACTION=Upload variables 35
CFFTP error variables 36
CFFTP ReturnValue variable 36
CFFTP query object columns 36
CFHTTP variables 36
CFLDAP variables 36

CFPOP variables 36
CFQUERY and CFSTOREDPROC variables 37
CFREGISTRY variables 37
CFSEARCH variables 37
Standard CGI variables 37
Request 37
Server 37
Client 38
CGI environment variables 38
CGI server variables 38
CGI client variables 38
CGI client certificate variables 38

CFML tags

cfabort

```
<cfabort
  showError = "error_message">
```

cfapplet

```
<cfapplet
  appletSource = "applet_name"
  name = "form_variable_name"
  height = "height_in_pixels"
  width = "width_in_pixels"
  vSpace = "space_above_and_below_in_pixels"
  hSpace = "space_on_each_side_in_pixels"
  align = "alignment_option"
  notSupported = "message_to_display_for_nonJava_browser"
  param_1 = "applet_parameter_name"
  param_2 = "applet_parameter_name"
  param_n = "applet_parameter_name">
```

cfapplication

```
<cfapplication
  name = "application_name"
  loginStorage = "cookie" or "session"
  clientManagement = "yes" or "no"
  clientStorage = "datasource_name" or "Registry" or
    "Cookie"
  setClientCookies = "yes" or "no"
  sessionManagement = "yes" or "no"
  sessionTimeout = #CreateTimeSpan(days, hours, minutes,
     seconds)#
  applicationTimeout = #CreateTimeSpan(days, hours,
     minutes, seconds)#
  setDomainCookies = "yes" or "no"
  scriptProtect = "none", "all", or list>
```

cfargument

```
<cfargument
  name="string"
  type="data type"
  required="yes" or "no"
  default="default value"
  displayname="descriptive name"
  hint="extended description">
```

cfassociate

```
<cfassociate
  baseTag = "base_tag_name"
  dataCollection = "collection_name">
```

cfbreak

```
<cfassociate
  baseTag = "base_tag_name"
  dataCollection = "collection_name">
```

cfcache

```
<cfcache
  action = "action"
  directory = "directory_name"
  timespan = "value"
  expireURL = "wildcarded_URL_reference"
  username = "username"
  password = "password"
  port = "port_number"
  protocol = "protocol">
```

cfcalendar

```
<cfcalendar
  name = "name of calendar"
  height = "height"
  width = "width"
  selectedDate = "date"
  mask = "returned date format"
  startRange = "first disabled date"
  endRange = "last disabled date"
  disabled = "true", "false", or no attribute value
  mask = "character pattern"
  dayNames = "days-of-the-week labels"
  monthNames = "month labels"
  visible = "Yes" or "No"
  enabled = "Yes" or "No"
  tooltip = "Tip text"
  onChange = "actionscript to invoke">
```

cfcase

```
<cfcase
  value = "value or delimited set of values"
  delimiters = "delimiter characters">
```

cfcatch

```
<cfcatch type = "exceptiontype">
  Exception processing code here
</cfcatch>
```

cfchart

Syntax 1

```
<!--- This syntax uses an XML file or string to specify the
    chart style. --->
<cfchart
  style = "XML string or filename">
</cfchart>
```

Syntax 2

```
<!--- This syntax uses the attributes of the cfchart tag to
specify the chart style. --->
<cfchart
  backgroundColor = "Hex value or Web color"
  chartHeight = "integer number of pixels"
  chartWidth = "integer number of pixels"
  dataBackgroundColor = "Hex value or Web color"
  font = "font name"
  fontBold = "yes" or "no"
  fontItalic = "yes" or "no"
  fontSize = "integer font size"
  foregroundColor = "Hex value or Web color"
  format = "flash" or "jpg" or "png"
  gridlines = "integer number of lines"
  labelFormat = "number, currency, percent, date"
  markerSize = "integer number of pixels"
  name = "String">
  pieSliceStyle = "solid, sliced"
  scaleFrom = "integer minimum value"
  scaleTo = "integer maximum value"
  seriesPlacement = "default, cluster, stacked, percent"
  show3D = "yes" or "no"
  showBorder = "yes" or "no"
  showLegend = "yes" or "no"
  showMarkers = "yes" or "no"
  showXGridlines = "yes" or "no"
  showYGridlines = "yes" or "no"
  sortXAxis = "yes" or "no"
  tipBGColor = "hex value or web color"
  tipStyle = "MouseDown, MouseOver, none"
  title = "title of chart"
  url = "onClick destination page"
  xAxisTitle = "title text"
  xAxisType = "scale or category"
  xOffset = "number between -1 and 1"
  yAxisTitle = "title text"
  yAxisType = "scale or category"
  yOffset = "number between -1 and 1"
</cfchart>
```

cfchartdata

```
<cfchartdata
  item = "text"
  value = "number">
```

cfchartseries

```
<cfchartseries
  colorlist = "list">
  itemColumn="queryColumn"
  markerStyle="style"
  paintStyle="plain, raise, shade, light"
  query="queryName"
  seriesColor="Hex value or Web color"
  seriesLabel="Label Text"
  type="type"
  valueColumn="queryColumn"
  dataLabelStyle="style"
</cfchartseries>
```

cfcol

```
<cfcol
  header = "column_header_text"
  width = "number_indicating_width_of_column"
  align = "left" or "right" or "center"
  text = "column_text">
```

cfcollection

```
<cfcollection
  action = "action"
  collection = "collection_name"
  path = "path_to_verity_collection"
  language = "language"
  name = "queryname"
  categories = "yes" or "no">
```

cfcomponent

```
<cfcomponent
  extends ="anotherComponent"
  output = "yes" or "no"
  style = "rpc" or "document"
  namespace = "default service namespace"
  serviceportname = "port element name"
  porttypename = "porttype element name"
  bindingname = "binding element name"
  wsdlfile = "path to hard-coded wsdl file"
  displayname = "text string">
  hint = "text string">
  Variable declarations
  <cffunction ...>
    ...
  </cffunction>

  <cffunction ...>
    ...
  </cffunction>
</cfcomponent>
```

cfcontent

```
<cfcontent
  type = "file_type"
  deleteFile = "yes" or "no"
  file = "filename"
  variable = "variablename"
  reset = "yes" or "no">
```

cfcookie

```
<cfcookie
  name = "cookie_name"
  value = "text"
  expires = "period"
  secure = "yes" or "no"
  path = "url"
  domain = ".domain">
```

cfdefaultcase

```
<cfdefaultcase>
```

cfdirectory

```
<cfdirectory
  action = "directory action"
  directory = "directory name"
  name = "query name"
  filter = "list filter"
  mode = "permission"
  sort = "sort specification"
  newDirectory = "new directory name"
  recurse = "yes" or "no">
```

cfdocument

```
<cfdocument
  format = "PDF" or "FlashPaper"
  filename = "filename"
  overwrite = "yes" or "no"
  name = "output variable name"
  pagetype = "page type"
  pageheight = "page height in inches"
  pagewidth = "page width in inches"
  orientation = "portrait/landscape"
  margintop = "number"
  marginbottom = "number"
  marginleft = "number"
  marginright = "number"
  unit = "in" or "cm"
  encryption = "128-bit" or "40-bit" or "none"
  ownerpassword = "password"
  userpassword = "password"
  permissions = "permission list"
  fontembed = "yes" or "no"
  backgroundvisible = "yes" or "no"
  scale = "percentage less than 100">
```

```
    HTML and CFML code
  </cfdocument>
```

cfdocumentitem

```
  <cfdocument ...>
  Syntax 1
  <cfdocumentitem type = "pagebreak"/>
  Syntax 2
  <cfdocumentitem
    type = "header" or "footer">
    Header/footer text
  </cfdocumentitem>

  </cfdocument>
```

cfdocumentsection

```
  <cfdocument ...>

    <cfdocumentsection
      margintop = "number"
      marginbottom = "number"
      marginleft = "number"
      marginright = "number">
      HTML, CFML, and cfdocumentitem tags
    </cfdocumentsection>

  </cfdocument>
```

cfdump

```
  <cfdump
    var = "#variable#"
    expand = "yes" or "no"
    label = "text"
    top = "number of rows or levels">
```

cfelse

```
  <cfif expression>
      HTML and CFML tags
    <cfelseif expression>
      HTML and CFML tags
    <cfelse>
      HTML and CFML tags
  </cfif>
```

cfelseif

```
  <cfif expression>
      HTML and CFML tags
    <cfelseif expression>
      HTML and CFML tags
    <cfelse>
      HTML and CFML tags
  </cfif>
```

cferror

```
  <cferror
    type = "a type"
    template = "template_path"
    mailTo = "email_address"
    exception = "exception_type">
```

cfexecute

```
  <cfexecute
    name = "application name"
    arguments = "command line arguments"
    outputFile = "output filename"
    variable = "variable name"
    timeout = "timeout interval">
    ...
  </cfexecute>
```

cfexit

```
  <cfexit
    method = "method">
```

cffile

```
  <cffile
    action = "append"
    file = "full_path_name"
    output = "string"
    addNewLine = "yes" or "no"
    attributes = "file_attributes_list"
    mode = "mode"
    charset = "charset_option" >
```

```
<cffile
  action = "copy"
  source = "full_path_name"
  destination = "full_path_name"
  mode = "mode"
  attributes = "file_attributes_list">

<cffile
  action = "delete"
  file = "full_path_name">

<cffile
  action = "move"
  source = "full_path_name"
  destination = "full_path_name"
  mode = "mode"
  attributes = "file_attributes_list"
  charset = "charset_option">

<cffile
  action = "read"
  file = "full_path_name"
  variable = "var_name"
  charset = "charset_option" >

<cffile
  action = "readBinary"
  file = "full_path_name"
  variable = "var_name">

<cffile
  action = "rename"
  source = "full_path_name"
  destination = "path_name"
  mode = "mode"
  attributes = "file_attributes_list">

<cffile
  action = "upload"
  fileField = "formfield"
  destination = "full_path_name"
  nameConflict = "behavior"
  accept = "mime_type/file_type"
  mode = "permission"
  attributes = "file_attribute_or_list">
  result = "result_name"

<cffile
  action = "write"
  file = "full_path_name"
  output = "content"
  mode = "permission"
  addNewLine = "yes" or "no"
  attributes = "file_attributes_list"
  charset = "charset_option" >
```

cfflush

```
<cfflush
  interval = "integer number of bytes">
```

cfform

```
<cfform
  name = "name"
  action = "form_action"
  method = "POST" or "GET"
  format = "HTML" or "Flash" or "XML"
  skin = "Flash or XSL skin"
  style = "style specification"
  preserveData = "yes" or "no"
  onSubmit = "javascript"
  scriptSrc = "path"
  codeBase = "URL"
  archive = "URL"
The following attributes are supported in Flash and XML only
  width = "pixels or percent"
  height = "pixels or percent"
The following attributes are supported in Flash only
  onError = "ActionScript code"
  wMode = "window" or "transparent" or "opaque"
  accessible = "yes" or "no"
  preloader = "yes" or "no"
  timeout = "seconds"
```

```
  The following attributes are supported in HTML and XML only
    class = "form class"
    enctype = "Internet media type"
    id = "HTML id"
    onload = "load event script"
    onreset = "reset event script"
    target = "target window or frame">
    ...
  </cfform>
```

cfformgroup

```
  <cfformgroup
    type = "group type"
    label = "label"
    style = "style specification"
    selectedIndex = "page number">
    width = "pixels"
    height = "pixels"
    enabled = "Yes" or "No"
    visible = "Yes" or "No"
    OnChange = "ActionScript expression"
    toolTip = "Tip text"
  ...ColdFusion forms controls
  </cfformgroup>
  OR
  <cfformgroup
    type = "repeater"
    query = "query object"
    startrow = "row number"
    maxrows = "integer">
    ...ColdFusion forms controls
  </cfformgroup>
```

cfformitem

```
  <cfformitem
    type = "hrule, vrule, or spacer"
    style = "style specification"
    width = "pixels"
    height = "pixels"
    visible = "Yes" or "No"/>
  OR
  <cfformitem
    type= "html or text"
    style = "style specification"
    width = "pixels"
    height = "pixels"
    visible = "Yes" or "No"
    enabled = "Yes" or "No"
    tooltip = "Tip text"
    bind = "bind expression">
    ...text
  </cfformitem>
```

cfftp

cfftp: Opening and closing FTP server connections

```
  <cfftp
    action = "action"
    username = "name"
    password = "password"
    server = "server"
    timeout = "timeout in seconds"
    port = "port"
    connection = "name"
    proxyServer = "proxy server"
    retryCount = "number"
    stopOnError = "yes" or "no"
    passive = "yes" or "no">
```

cfftp: Connection: file and directory operations

```
  <cfftp
    action = "action"
    username = "name"
    password = "password"
    name = "query_name"
    server = "server"
    ASCIIExtensionList = "extensions"
    transferMode = "mode"
    failIfExists = "yes" or "no"
    directory = "directory name"
    localFile = "filename"
    remoteFile = "filename"
    item = "directory or file"
    existing = "file or directory name"
```

```
    new = "file or directory name"
    proxyServer = "proxy server"
    passive = "yes" or "no">
    result = "result_name"
```

cfftp action = "listDir"

```
<cfftp
  action = "listDir"
</cfftp>
```

cffunction

```
<cffunction
  name = "methodName"
  returnType = "dataType"
  roles = "securityRoles"
  access = "methodAccess"
  description = "function description"
  output = "yes" or "no"
  displayName = "name"
  Hint = "hint text">
```

cfgraph *See* **cfchart**

cfgraphdata *See* **cfchartdata**

cfgrid

```
<cfgrid
  name = "name"
  format = "applet" or "Flash" or "xml"
  height = "integer"
  width = "integer"
  query = "query_name"
  selectMode = "mode"
  insert = "yes" or "no"
  delete = "yes" or "no"
  font = "column_font"
  fontSize = "size"
  italic = "yes" or "no"
  bold = "yes" or "no"
  textColor = "web color"
  gridLines = "yes" or "no"
  rowHeight = "pixels"
  colHeaders = "yes" or "no"
  colHeaderFont = "font_name"
  colHeaderFontSize = "size"
  colHeaderItalic = "yes" or "no"
  colHeaderBold = "yes" or "no"
  colHeaderTextColor = "web color"
  bgColor = "web color"
  maxRows = "number"
The following attributes work in Flash format only
  style= "style specification"
  enabled = "Yes" or "No"
  visible = "Yes" or "No"
  toolTip = "Tip text"
  onChange = "ActionScript"
The following attributes work in applet and XML format only
  autoWidth = "yes" or "no"
  vSpace = "integer"
  hSpace = "integer"
  align = "value"
  sort = "yes" or "no"
  href = "URL"
  hrefKey = "column_name"
  target = "URL_target"
  appendKey = "yes" or "no"
  highlightHref = "yes" or "no"
  onValidate = "javascript_function"
  onError = "text"
  gridDataAlign = "position"
  rowHeaders = "yes" or "no"
  rowHeaderAlign = "position"
  rowHeaderFont = "font_name"
  rowHeaderFontSize = "size"
  rowHeaderItalic = "yes" or "no"
  rowHeaderBold = "yes" or "no"
  rowHeaderTextColor = "web color"
  colHeaderAlign = "position"
  selectColor = "web color"
  notSupported = "text"
  pictureBar = "yes" or "no"
  insertButton = "text"
  deleteButton = "text"
  sortAscendingButton = "text"
```

```
    sortDescendingButton = "text">
  Zero or more cfgridcolumn and cfgridrow tags
  </cfgrid>
```

cfgridcolumn

```
  <cfgridcolumn
    name = "column_name"
    header = "header"
    width = "column_width"
    type = "type"
    display = "yes" or "no"
    select = "yes" or "no"
    font = "column_font"
    fontSize = "size"
    italic = "yes" or "no"
    bold = "yes" or "no"
    textColor = "web color" or "expression"
    bgColor = "web color" or "expression"
    dataAlign = "position"
  The following attribute applies to Flash format only
    mask= "format mask"
  The following attributes apply to applet format only
    href = "URL"
    hrefKey = "column_name"
    target = "URL_target"
    headerFont = "font_name"
    headerFontSize = "size"
    headerItalic = "yes" or "no"
    headerBold = "yes" or "no"
    headerTextColor = "web color"
    headerAlign = "position"
    numberFormat = "format"
    values = "Comma-separated strings and/or numeric range"
    valuesDisplay = "Comma-separated strings and/or numeric
        range"
    valuesDelimiter = "delimiter character">
```

cfgridrow

```
  <cfgridrow
    data = "col1, col2, ...">
```

cfgridupdate

```
  <cfgridupdate
    grid = "gridname"
    dataSource = "data source name"
    tableName = "table name"
    username = "data source username"
    password = "data source password"
    tableOwner = "table owner"
    tableQualifier = "qualifier"
    keyOnly = "yes" or "no">
```

cfheader

```
  <cfheader
    name = "header_name"
    value = "header_value"
    charset="charset">
  OR
  <cfheader
    statusCode = "status_code"
    statusText = "status_text">
```

cfhtmlhead

```
  <cfhtmlhead
    text = "text">
```

cfhttp

```
  <cfhttp
    url = "server_URL"
    port = "port_number"
    method = "method_name"
    proxyServer = "hostname"
    proxyPort = "port_number"
    proxyUser = "username"
    proxyPassword = "password"
    username = "username"
    password = "password"
    userAgent = "user_agent"
    charset = "character encoding"
    resolveURL = "yes" or "no"
    throwOnError = "yes" or no"
    redirect = "yes" or "no"
    timeout = "timeout_period"
    getasbinary = "yes or no"
```

```
    multipart = "yes or no"
    path = "path"
    file = "filename"
    name = "queryname"
    columns = "query_columns"
    firstrowasheaders = "yes" or "no"
    delimiter = "character"
    textQualifier = "character"
    result = "result_name"
    cfhttpparam tags [optional for some methods]
  </cfhttp>
```

cfhttpparam

```
  <cfhttpparam
    type = "transaction type"
    name = "data name"
    value = "data value"
    file = "filename"
    encoded = "yes" or "no"
    mimeType = "MIME type designator">
```

cfif

```
  <cfif expression>
    HTML and CFML tags
  <cfelseif expression>
    HTML and CFML tags
  <cfelse>
    HTML and CFML tags
  </cfif>
```

cfimport

```
  <cfimport
    taglib = "taglib-location"
    prefix = "custom">
```

cfinclude

```
  <cfinclude
    template = "template_name">
```

cfindex

```
  <cfindex
    collection = "collection_name"
    action = "action"
    type = "type"
    title = "title"
    key = "ID"
    body = "body"
    custom1 = "custom_value"
    custom2 = "custom_value"
    custom3 = "custom_value"
    custom4 = "custom_value"
    category = "category_name"
    categoryTree = "category_tree"
    URLpath = "URL"
    extensions = "file_extensions"
    query = "query_name"
    recurse = "yes" or "no"
    language = "language">
    status = "status">
```

cfinput

```
  <cfinput
    name = "name"
    type = "input type"
    label = "text"
    style = "style specification"
    required = "yes" or "no"
    mask = "masking pattern"
    validate = "data type"
    validateAt= one or more of "onBlur", "onServer",
       "onSubmit"
    message = "text"
    range = "min_value, max_value"
    maxlength = "number"
    pattern = "regexp"
    onValidate = "script name"
    onError = "script name"
    size = "integer"
    value = "initial value"
    bind = "bind expression"
    checked
    disabled = "true" or "false" or no attribute value
    src = "image URL"
    onKeyUp = "JavaScript or ActionScript"
```

```
    onKeyDown = "JavaScript or ActionScript"
    onMouseUp = "JavaScript or ActionScript"
    onMouseDown = "JavaScript or ActionScript"
    onChange = "JavaScript or ActionScript"
    onClick = "JavaScript or ActionScript"
    firstDayOfWeek = "day name"
    dayNames = "days-of-the-week labels"
    monthNames = "month labels"
    enabled = "Yes" or "No"
    visible = "Yes" or "No"
    toolTip = "Tip text"
    height = "number of pixels"
    width = "number of pixels">
```

cfinsert

```
<cfinsert
  dataSource = "ds_name"
  tableName = "tbl_name"
  tableOwner = "owner"
  tableQualifier = "tbl_qualifier"
  username = "username"
  password = "password"
  formFields = "formfield1, formfield2, ...">
```

cfinvoke

Syntax 1

```
<!--- This syntax invokes a method of a component. --->
<cfinvoke
  component = "component name or reference"
  method = "method name"
  returnVariable = "variable name"
  argumentCollection = "argument collection"
  ...>
OR
```

Syntax 2

```
<!--- This syntax can invoke a method of a component only
  from within the component. --->
<cfinvoke
  method = "method name"
  returnVariable = "variable name"
  argumentCollection = "argument collection"
  ...>
OR
```

Syntax 3

```
<!--- This syntax invokes a web service. --->
<cfinvoke
  webservice = "URLtoWSDL_location"
  method = "operation_name"
  username = user name"
  password = "password"
  timeout = "request timeout in seconds"
  proxyServer = "WSDL proxy server URL
  proxyPort = "port on proxy server"
  proxyUser = "user id for proxy server"
  proxyPassword = "password for proxy server"
  servicePort = "WSDL port name"
  inputParam1 = "value1"
  inputParam2 = "value2"
  ...
  returnVariable = "var_name"
  ...>
OR
```

Syntax 4A

```
<!--- This syntax invokes a component.
This syntax shows instantiation with the cfobject tag.
This cfinvoke syntax applies to instantiating a component
with the cfobject tag and to instantiating a component
with the CreateObject function. --->
<cfobject
  component = "component name"
  name = "mystringname for instantiated object">
<cfinvoke
  <!--- Value is object name, within number signs. --->
  component = "#mystringname for instantiated component#"
  method = "method name"
  returnVariable = "variable name"
  argumentCollection = "argument collection"
  ...>
OR
```

Syntax 4B

```
<!--- This syntax invokes a web service.
This syntax shows instantiation with the cfobject tag.
This cfinvoke syntax applies to instantiating a web service
with the cfobject tag and to instantiating a web service
with the CreateObject function. --->
<cfobject
  webservice = "web service name"
  name = "mystringname for instantiated object"
  method = "operation_name">
<cfinvoke
  <!--- value is object name, within number signs. --->
  webservice = "#my stringname for instantiated web
   service#"
  timeout = "request timeout in seconds"
  proxyServer = "WSDL proxy server url
  proxyPort = "numeric port on proxy server"
  proxyUser = "string user id for proxy server"
  servicePort = "WSDL port name"
  proxyPassword = "string user password for proxy server">
```

cfinvokeargument

```
<cfinvokeargument
  name="argument name"
  value="argument value"
  omit = "yes" or "no">
```

cfldap

```
<cfldap
  server = "server_name"
  port = "port_number"
  username = "name"
  password = "password"
  action = "action"
  name = "name"
  timeout = "seconds"
  maxRows = "number"
  start = "distinguished_name"
  scope = "scope"
  attributes = "attribute, attribute"
  returnAsBinary = "column_name, column_name"
  filter = "filter"
  sort = "attribute[, attribute]..."
  sortControl = "nocase" and/or "desc" or "asc"
  dn = "distinguished_name"
  startRow = "row_number"
  modifyType = "replace" or "add" or "delete"
  rebind = "yes" or "no"
  referral = "number_of_allowed_hops"
  secure = "multi_field_security_string"
  separator = "separator_character"
  delimiter = "delimiter_character">
```

cflocation

```
<cflocation
  url = "url"
  addToken = "yes" or "no">
```

cflock

```
<cflock
  timeout = "timeout in seconds "
  scope = "Application" or "Server" or "Session"
  name = "lockname"
  throwOnTimeout = "yes" or "no"
  type = "readOnly" or "exclusive ">
  <!--- CFML to be synchronized --->
</cflock>
```

cflog

```
<cflog
  text = "text"
  log = "log type"
  file = "filename"
  type = "message type"
  application = "yes" or "no">
```

cflogin

```
<cflogin
  idletimeout = "value"
  applicationToken = "token"
  cookieDomain = "domain"
  ...
  <cfloginuser
    name = "name"
```

```
    password = "password-string"
    roles = "roles">
  ...>
</cflogin>
```

cfloginuser

```
<cfloginuser
  name = "name"
  password = "password-string"
  roles = "roles">
```

cflogout

```
<cflogout>
```

cfloop

cfloop: index loop

```
<cfloop
  index = "parameter_name"
  from = "beginning_value"
  to = "ending_value"
  step = "increment">
  HTML or CFML code
</cfloop>
```

cfloop: conditional loop

```
<cfloop
  condition = "expression">
  ...
</cfloop>
```

cfloop: looping over a date or time range

```
<cfloop
  from = "start_time"
  to = "end_time"
  index = "current_value"
  step = "increment">
</cfloop>
```

cfloop: looping over a query

```
<cfloop
  query = "query_name"
  startRow = "row_num"
  endRow = "row_num">
</cfloop>
```

cfloop: looping over a list or file

```
<cfloop
  index = "index_name"
  list = "list_items"
  delimiters = "item_delimiter">
  ...
</cfloop>
```

cfloop: looping over a COM collection or structure

```
<cfloop collection = #FFUNC# item = "file2">
  <cfoutput>#file2.name# <br> </cfoutput>
</cfloop>
```

cfmail

```
<cfmail
  to = "recipient"
  from = "sender"
  cc = "copy_to"
  bcc = "blind_copy_to"
  subject = "msg_subject"
  replyto = "reply_to_addr"
  failto = "fail_message_addr"
  username = "user name"
  password = "password"
  wraptext = "column number"
  charset = "character encoding"
  type = "msg_type"
  mimeattach = "path"
  query = "query_name"
  group = "query_column"
  groupcasesensitive = "yes" or "no"
  startrow = "query_row"
  maxrows = "max_msgs"
  server = "serverspecs"
  port = "port_id"
  mailerid = "headerid"
  timeout = "seconds"
  spoolenable = "yes" or "no"
  debug = "yes" or "no">
```

```
(Optional) Mail message body and/or cfhttpparam tags
</cfmail>
```

cfmailparam

```
<cfmail
  to = "recipient"
  subject = "msg_subject"
  from = "sender"
  ...more attributes... >
  <cfmailparam
    file = "file-name"
    type ="media type"
    contentID = "content ID"
    disposition = "disposition type">
  OR
  <cfmailparam
    name = "header-name"
    value = "header-value" >
  ...
</cfmail>
```

cfmailpart

```
<cfmail
  ... >
  (Optional cfmailparam entries)
  <cfmailpart
    type="mime type"
    charset="character encoding"
    wraptext="number"
  >
  Mail part contents
  </cfmailpart>
  ...
</cfmail>
```

cfmodule

```
<cfmodule
  template = "path"
  name = "tag_name"
  attributeCollection = "collection_structure"
  attribute_name1 = "valuea"
  attribute_name2 = "valueb"
  ...>
```

cfNTauthenticate

```
<cfNTauthenticate
  username="username"
  password="password"
  domain="nt_domain"
  result="result variable"
  listGroups = "yes" or "no"
  throwOnError = "yes" or "no">
```

cfobject

cfobject: COM object

```
<cfobject
  type = "com"
  action = "action"
  class = "program_ID"
  name = "text"
  context = "context"
  server = "server_name">
```

cfobject: component object

```
<cfobject
  name = "variable name"
  component = "component name">
```

cfobject: CORBA object

```
<cfobject
  type = "corba"
  context = "context"
  class = "file or naming service"
  name = "text"
  locale = "type-value arguments">
```

cfobject: Java or EJB object

```
<cfobject
  type = "Java"
  action = "Create"
  class = "Java class"
  name = "object name">
```

cfobject: web service object

```
<cfobject
  webservice= "http://....?wsdl" or "name set in
     Administrator"
  name = "myobjectname">
```

cfobjectcache

```
<cfobjectcache
  action = "clear">
```

cfoutput

```
<cfoutput
  query = "query_name"
  group = "query_column"
  groupCaseSensitive = "yes" or "no"
  startRow = "start_row"
  maxRows = "max_rows_output">
</cfoutput>
```

cfparam

```
<cfparam
  name = "param_name"
  type = "data_type"
  default = "value"
  max = "value"
  min = "value"
  pattern = "regular expression">
```

cfpop

```
<cfpop
  server = "servername"
  port = "port_number"
  username = "username"
  password = "password"
  action = "action"
  name = "queryname"
  messageNumber = "number"
  uid = "number"
  attachmentPath = "path"
  timeout = "seconds"
  maxRows = "number"
  startRow = "number"
  generateUniqueFilenames = "yes" or "no"
  debug = "yes" or "no">
```

cfprocessingdirective

```
<cfprocessingdirective
  pageencoding = "page-encoding literal string" />
OR
<cfprocessingdirective
  suppressWhiteSpace = "yes" or "no"
  pageEncoding = "page-encoding literal string">
  CFML tags
</cfprocessingdirective>
```

cfprocparam

```
<cfprocparam
  type = "in" or "out" or "inout"
  variable = "variable name"
  value = "parameter value"
  CFSQLType = "parameter datatype"
  maxLength = "length"
  scale = "decimal places"
  null = "yes" or "no">
```

cfprocresult

```
<cfprocresult
  name = "query_name"
  resultSet = "1-n"
  maxRows = "maxrows">
```

cfproperty

```
<cfproperty
  name="name"
  type="type"
  required="boolean"
  default="default value"
  displayname="descriptive name"
  hint="extended description">
```

cfquery

```
<cfquery
  name = "query_name"
  dataSource = "ds_name"
  dbtype = "query"
```

```
    username = "username"
    password = "password"
    maxRows = "number"
    blockFactor = "blocksize"
    timeout = "seconds"
    cachedAfter = "date"
    cachedWithin = "timespan"

    Either of the following:
      debug = "yes" or "no"
    OR
      debug
  >
    result = "result_name"
  </cfquery>
```

cfqueryparam

```
  <cfquery
    name = "query_name"
    dataSource = "ds_name"
    ...Other attributes...>
    SQL STATEMENT column_name =
    <cfqueryparam value = "parameter value"
      CFSQLType = "parameter type"
      maxLength = "maximum parameter length"
      scale = "number of decimal places"
      null = "yes" or "no"
      list = "yes" or "no"
      separator = "separator character">
    AND/OR ...additional criteria of the WHERE clause...
    </cfquery>
```

cfregistry

cfregistry action = "getAll"

```
<cfregistry
  action = "getAll"
  branch = "branch"
  type = "data type"
  name = "query name"
  sort = "criteria">
```

cfregistry action = "get"

```
<cfregistry
  action = "get"
  branch = "branch"
  entry = "value"
  variable = "variable"
  type = "data type">
```

cfregistry action = "set"

```
<cfregistry
  action = "set"
  branch = "branch"
  entry = "key or value"
  type = "value type"
  value = "data">
```

cfregistry action = "delete"

```
<cfregistry
  action = "delete"
  branch = "branch"
  entry = "key or value">
```

cfreport

Syntax 1

```
<!--- Syntax 1 - Use this syntax with the ColdFusion Report
Builder. --->
<cfreport
    template = "report definition filename"
    format = "PDF or FlashPaper" or "excel"
    name = "cf variable"
    filename = "output filename"
    query = "query variable"
    overwrite = "yes" or "no"
    encryption = "128-bit" or "40-bit" or "none"
    ownerpassword = "password"
    userpassword = "password"
    permissions = "permission list"
    cfreportparam tags [optional]
</cfreport>

OR
```

Syntax 2

```
<!--- Syntax 2 - Use this syntax with Crystal Reports. --->
<cfreport
    report = "report_path"
    dataSource = "ds_name"
    type = "type"
    timeout = "number of seconds"
    orderBy = "result_order"
    username = "username"
    password = "password"
    formula = "formula">
</cfreport>
```

cfreportparam

```
<cfreportparam
  name = "data name"
  value = "data value">
```

cfrethrow

```
<cfrethrow>
```

cfreturn

```
<cfreturn
  expr>
```

cfsavecontent

```
<cfsavecontent
  variable = "variable name">
  the content
</cfsavecontent>
```

cfschedule

```
<cfschedule
  action = "update"
  task = "taskname"
  operation = "HTTPRequest"
  file = "filename"
  path = "path_to_file"
  startDate = "date"
  startTime = "time"
  url = "URL"
  port = "port_number"
  publish = "yes" or "no"
  endDate = "date"
  endTime = "time"
  interval = "seconds"
  requestTimeOut = "seconds"
  username = "username"
  password = "password"
  proxyServer = "hostname"
  proxyPort = "port_number">
  proxyUser = "username"
  proxyPassword = "password"
  resolveURL = "yes" or "no"

<cfschedule
  action = "delete"
  task = "TaskName">

<cfschedule
  action = "run"
  task = "TaskName">
```

cfscript

```
<cfscript>
  cfscript code here
</cfscript>
```

cfsearch

```
<cfsearch
  name = "search_name"
  collection = "collection_name"
  category = "category[,category2,...]"
  categoryTree = "tree_location"
  status = ""
  type = "criteria"
  criteria = "search_expression"
  maxRows = "number"
  startRow = "row_number"
  suggestions = "suggestion_option"
  contextPassages = "number_of_passages"
  contextBytes = "number_of_bytes"
  contextHighlightBegin = "html_string"
  contextHighlightEnd = "html_string"
```

```
    previousCriteria = "criteria"
    language = "language">
```

cfselect

```
<cfselect
  name = "name"
  label = "label"
  style = "style specification"
  size = "integer"
  required = "yes" or "no"
  message = "text"
  onError = "text"
  multiple = "yes" or "no"
  query = "queryname"
  value = "text"
  display = "text">
  group = "query column name
  queryPosition = "above" or "below"
  selected = "value or list"
  onKeyUp = "JavaScript or ActionScript"
  onKeyDown = "JavaScript or ActionScript"
  onMouseUp = "JavaScript or ActionScript"
  onMouseDown = "JavaScript or ActionScript"
  onChange = "JavaScript or ActionScript"
  onClick = "JavaScript or ActionScript"
  visible = "Yes" or "No"
  enabled = "Yes" or "No"
  tooltip = "tip text"
  height = "number of pixels" Flash only
  width = "number of pixels" Flash only
  >
  Zero or more HTML option tags
</cfselect>
```

cfset

```
<cfset
  var variable_name = expression>
```

cfsetting

```
<cfsetting
  enableCFoutputOnly = "yes" or "no"
  showDebugOutput = "yes" or "no"
  requestTimeOut = "value in seconds" >
```

cfsilent

```
<cfsilent>
...
</cfsilent>
```

cfslider

```
<cfslider
  name = "name"
  label = "text"
  range = "min_value, max_value"
  scale = "uinteger"
  value = "integer"
  onValidate = "script_name"
  message = "text"
  onError = "text"
  height = "integer"
  width = "integer"
  vSpace = "integer"
  hSpace = "integer"
  align = "alignment"
  lookAndFeel = "motif" or "windows" or "metal"
  vertical = "yes" or "no"
  bgColor = "color"
  textColor = "color"
  font = "font_name"
  fontSize = "integer"
  italic = "yes" or "no"
  bold = "yes" or "no"
  notSupported = "text">
```

cfstoredproc

```
<cfstoredproc
  procedure = "procedure name"
  dataSource = "ds_name"
  username = "username"
  password = "password"
  blockFactor = "blocksize"
  debug = "yes" or "no"
  returnCode = "yes" or "no"
  result = "result_name">
```

cfswitch

```
<cfswitch
  expression = "expression">
    One or more cfcase tags
    Zero or one cfdefaultcase tags
</cfswitch>
```

cftable

```
<cftable
  query = "query_name"
  maxRows = "maxrows_table"
  colSpacing = "number_of_spaces"
  headerLines = "number_of_lines"
  HTMLTable
  border
  colHeaders
  startRow = "row_number">
  ...
</cftable>
```

cftextarea

```
<cftextarea
name = "name"
label = "text"
required = "yes" or "no"
style = "style specification"
validate = "data type"
validateAt= one or more of "onBlur", "onServer", "onSubmit"
message = "text"
range = "min_value, max_value"
maxlength = "number"
pattern = "regexp"
onValidate = "script name"
onError = "script name"
disabled = "true" "false" or no attribute value
value = "text"
onKeyUp = "JavaScript or ActionScript"
onKeyDown = "JavaScript or ActionScript"
onMouseUp = "JavaScript or ActionScript"
onMouseDown = "JavaScript or ActionScript"
onChange = "JavaScript or ActionScript"
onClick = "JavaScript or ActionScript"
visible = "Yes" or "No"
enabled = "Yes" or "No"
tooltip = "Tip text"
height = "number of pixels" Flash only
width = "number of pixels" Flash only
>
   Text
</cftextarea>
```

cfthrow

Syntax 1

```
<cfthrow
  type = "exception_type "
  message = "message"
  detail = "detail_description "
  errorCode = "error_code "
  extendedInfo = "additional_information"
  object = "java_except_object">
```

Syntax 2

```
<cfthrow
  object = #object_name#>
```

cftimer

```
<cftimer
  label= "text"
  type = "inline" or "outline" or "comment" or "debug" >
  CFML statements
</cftimer>
```

cftrace

```
<cftrace
  abort = "Yes" or "No"
  category = "string"
  inline = "Yes" or "No"
  text = "string"
  type = "format"
  var = "variable_name"
</cftrace>
```

cftransaction

```
<cftransaction
  action = "begin" or "commit" or "rollback"
  isolation = "read_uncommitted" or "read_committed" or
    "repeatable_read" >
</cftransaction>
```

cftree

```
<cftree name = "name"
  format="applet", "flash", xml, or "object"
  required = "yes" or "no"
  delimiter = "delimiter"
  completePath = "yes" or "no"
  appendKey = "yes" or "no"
  highlightHref = "yes" or "no"
  onValidate = "script_name"
  message = "text"
  onError = "text"
  lookAndFeel = "motif" or "windows" or "metal"
  font = "font"
  fontSize = "size"
  italic = "yes" or "no"
  bold = "yes" or "no"
  height = "integer"
  width = "integer"
  vSpace = "integer"
  hSpace = "integer"
  align = "alignment"
  border = "yes" or "no"
  hScroll = "yes" or "no"
  vScroll = "yes" or "no"
  style= "style specification"
  enabled = "Yes" or "No"
  visible = "Yes" or "No"
  tooltip = "tip text"
  onChange = "ActionScript"
  notSupported = "text">

</cftree>
```

cftreeitem

```
<cftreeitem
  value = "text"
  display = "text"
  parent = "parent_name"
  img = "filename"
  imgopen = "filename"
  href = "URL"
  target = "URL_target"
  query = "queryname"
  queryAsRoot = "yes" or "no"
  expand = "yes" or "no">
```

cftry

```
<cftry>
  Code that might throw an exception
  One or more cfcatch blocks
</cftry>
```

cfupdate

```
<cfupdate
  dataSource = "ds_name"
  tableName = "table_name"
  tableOwner = "name"
  tableQualifier = "qualifier"
  username = "username"
  password = "password"
  formFields = "field_names">
```

cfwddx

```
<cfwddx
  action = "action"
  input = "inputdata"
  output = "result variable name"
  topLevelVariable = "top-level variable name for
     JavaScript"
  useTimeZoneInfo = "yes" or "no"
  validate = "yes" or "no" >
```

cfxml

```
<CFXML
  variable="xmlVarName"
  caseSensitive="yes" or "no">
```

CFML functions

Array functions

ArrayAppend(*array, value*)
ArrayAvg(*array*)
ArrayClear(*array*)
ArrayDeleteAt(*array, position*)
ArrayInsertAt(*array, position, value*)
ArrayIsEmpty(*array*)
ArrayLen(*array*)
ArrayMax(*array*)
ArrayMin(*array*)
ArrayNew(*dimension*)
ArrayPrepend(*array, value*)
ArrayResize(*array, minimum_size*)
ArraySet(*array, start_pos, end_pos, value*)
ArraySort(*array, sort_type* [, *sort_order*])
ArraySum(*array*)
ArraySwap(*array, position1, position2*)
ArrayToList(*array* [, *delimiter*])
IsArray(*value* [, *number*])
ListToArray(*list* [, *delimiters*])

Conversion functions

ArrayToList(*array* [, *delimiter*])
BinaryDecode(*string, binaryencoding*)
BinaryEncode(*binarydata, encoding*)
CharsetDecode(*string, encoding*)
CharsetEncode(*binaryobject, encoding*)
Hash(*string* [, *algorithm* [, *encoding*]])
LCase(*string*)
ListToArray(*list* [, *delimiters*])
ToBase64(*string or binary_object* [, *encoding*])
ToBinary(*string_in_Base64* or *binary_value*)
ToScript(*cfvar, javascriptvar, outputformat, ASFormat*)
ToString(*value* [, *encoding*])
URLDecode(*urlEncodedString* [, *charset*])
URLEncodedFormat(*string* [, *charset*])
Val(*string*)
XmlFormat(*string*)
XmlParse(*xmlText* [[, *caseSensitive*], *validator*])
XmlTransform(*xml, xsl* [, *parameters*])

Date and time functions

CreateDate(*year, month, day*)
CreateDateTime(*year, month, day, hour, minute, second*)
CreateODBCDate(*date*)
CreateODBCDateTime(*date*)
CreateODBCTime(*date*)
CreateTime(*hour, minute, second*)
CreateTimeSpan(*days, hours, minutes, seconds*)
DateAdd("*datepart*", *number*, "*date*")
DateCompare("*date1*", "*date2*" [, "*datePart*"])
DateConvert("*conversion-type*", "*date*")
DateDiff("*datepart*", "*date1*", "*date2*")
DateFormat("*date*" [, "*mask*"])
DatePart("*datepart*", "*date*")
Day("*date*")

DayOfWeek("*date*")
DayOfWeekAsString(*day_of_week*)
DayOfYear("*date*")
DaysInMonth("*date*")
DaysInYear("*date*")
FirstDayOfMonth(*date*)
GetHttpTimeString(*date_time_object*)
GetTickCount()
GetTimeZoneInfo()
Hour(*date*)
IsDate(*string*)
IsLeapYear(*year*)
IsNumericDate(*number*)
LSDateFormat(*date* [, *mask*])
LSIsDate(*string*)
LSParseDateTime(*date/time-string*)
LSTimeFormat(*time* [, *mask*])
Minute(*date*)
Month(*date*)
MonthAsString(*month_number*)
Now()
ParseDateTime(*date/time-string* [, *pop-conversion*])
Quarter(*date*)
Second(*date*)
TimeFormat(*time* [, *mask*])
Week(*date*)
Year(*date*)

Decision functions

DirectoryExists(*absolute_path*)
FileExists(*absolute_path*)
IIf(*condition*, *string_expression1*, *string_expression2*)
IsArray(*value* [, *number*])
IsBinary(*value*)
IsBoolean(*value*)
IsCustomFunction(*name*)
IsDate(*string*)
IsDebugMode()
IsDefined("*variable_name*")
IsK2ServerABroker()
IsK2ServerDocCountExceeded()
IsK2ServerOnline()
IsLeapYear(*year*)
IsNumeric(*string*)
IsNumericDate(*number*)
IsObject(*value*)
IsQuery(*value*)
IsSimpleValue(*value*)
IsStruct(*variable*)
IsUserInRole("*role_name*")
IsValid(*type*, *value*)
isValid("*range*", *value*, *min*, *max*)
isValid("*regex*" or "*regular_expression*", *value*, *pattern*)
IsWDDX(*value*)
IsXML(*value*)
IsXmlAttribute(*value*)
IsXmlDoc(*value*)
IsXmlElem(*value*)

IsXmlNode(*value*)
IsXmlRoot(*value*)
LSIsCurrency(*string*)
LSIsDate(*string*)
LSIsNumeric(*string*)
StructIsEmpty(*structure*)
StructKeyExists(*structure*, "*key*")
YesNoFormat(*value*)

Display and formatting functions

CJustify(*string*, *length*)
DateFormat("*date*" [, "*mask*"])
DecimalFormat(*number*)
DollarFormat(*number*)
FormatBaseN(*number*, *radix*)
GetLocale()
GetLocaleDisplayName([*locale* [, *inLocale*]])
HTMLCodeFormat(*string* [, *version*])
HTMLEditFormat(*string* [, *version*])
LJustify(*string*, *length*)
LSCurrencyFormat(*number* [, *type*])
LSDateFormat(*date* [, *mask*])
LSEuroCurrencyFormat(*currency-number* [, *type*])
LSIsCurrency(*string*)
LSIsDate(*string*)
LSNumberFormat(*number* [, *mask*])
LSParseCurrency(*string*)
LSParseDateTime(*date/time-string*)
LSParseEuroCurrency(*currency-string*)
LSParseNumber(*string*)
LSTimeFormat(*time* [, *mask*])
NumberFormat(*number* [, *mask*])
ParagraphFormat(*string*)
RJustify(*string*, *length*)
StripCR(*string*)
TimeFormat(*time* [, *mask*])
YesNoFormat(*value*)

Dynamic evaluation functions

DE(*string*)
Evaluate(*string_expression1* [, *string_expression2* [, …]])
IIf(*condition*, *string_expression1*, *string_expression2*)
SetVariable(*name*, *value*)

Extensibility functions

CreateObject
  COM object: CreateObject(*type*, *class*, *context*, *serverName*)
  Component object: CreateObject(*type*, *component-name*)
  CORBA object: CreateObject(*type*, *context*, *class*, *locale*)
  Java or EJB object: CreateObject(*type*, *class*)
  Web service object: CreateObject(*type*, *urltowsdl* [, *portname*])
GetGatewayHelper(*gatewayID*)
ReleaseComObject(*objectName*)
SendGatewayMessage(*gatewayID*, *data*)
ToScript(*cfvar*, *javascriptvar*, *outputformat*, *ASFormat*)

International functions

DateConvert("*conversion-type*", "*date*")
GetEncoding(*scope_name*)
GetHttpTimeString(*date_time_object*)
GetLocale()
GetLocaleDisplayName([*locale* [, *inLocale*]])
GetTimeZoneInfo()
LSCurrencyFormat(*number* [, *type*])
LSDateFormat(*date* [, *mask*])
LSEuroCurrencyFormat(*currency-number* [, *type*])
LSIsCurrency(*string*)
LSIsDate(*string*)
LSIsNumeric(*string*)
LSNumberFormat(*number* [, *mask*])
LSParseCurrency(*string*)
LSParseDateTime(*date/time-string*)
LSParseEuroCurrency(*currency-string*)
LSParseNumber(*string*)
LSTimeFormat(*time* [, *mask*])
SetEncoding(*scope_name, charset*)
SetLocale(*new_locale)*

List functions

ArraySort(*array, sort_type* [, *sort_order*])
ArrayToList(*array* [, *delimiter*])
Asc(*string*)
Chr(*number*)
CJustify(*string, length*)
Compare(*string1, string2*)
CompareNoCase(*string1, string2*)
Decrypt(*encrypted_string, key* [, *algorithm* [, *encoding*]])
Encrypt(*string, key* [, *algorithm* [, *encoding*]])
Find(*substring, string* [, *start*])
FindNoCase(*substring, string* [, *start*])
FindOneOf(*set, string* [, *start*])
FormatBaseN(*number, radix*)
GetClientVariablesList()
LCase(*string*)
Left(*string, count*)
Len(*string or binary object*)
ListAppend(*list, value* [, *delimiters*])
ListChangeDelims(*list, new_delimiter* [, *delimiters*])
ListContains(*list, substring* [, *delimiters*])
ListContainsNoCase(*list, substring* [, *delimiters*])
ListDeleteAt(*list, position* [, *delimiters*])
ListFind(*list, value* [, *delimiters*])
ListFindNoCase(*list, value* [, *delimiters*])
ListFirst(*list* [, *delimiters*])
ListGetAt(*list, position* [, *delimiters*])
ListInsertAt(*list, position, value* [, *delimiters*])
ListLast(*list* [, *delimiters*])
ListLen(*list* [, *delimiters*])
ListPrepend(*list, value* [, *delimiters*])
ListQualify(*list, qualifier* [, *delimiters*] [, *elements*])
ListRest(*list* [, *delimiters*])
ListSetAt(*list, position, value* [, *delimiters*])
ListSort(*list, sort_type* [, *sort_order*] [, *delimiters*])
ListToArray(*list* [, *delimiters*])

```
ListValueCount(list, value [, delimiters ])
ListValueCountNoCase(list, value [, delimiters ])
ReplaceList(string, list1, list2)
```

Mathematical functions

```
Abs(number)
ACos(number)
ArrayAvg(array)
ArraySum(array)
ASin(number)
Atn(number)
BitAnd(number1, number2)
BitMaskClear(number, start, length)
BitMaskRead(number, start, length)
BitMaskSet(number, mask, start, length)
BitNot(number)
BitOr(number1, number2)
BitSHLN(number, count)
BitSHRN(number, count)
BitXor(number1, number2)
Ceiling(number)
Cos(number)
DecrementValue(number)
Exp(number)
Fix(number)
FormatBaseN(number, radix)
IncrementValue(number)
InputBaseN(string, radix)
Int(number)
Log(number)
Log10(number)
Max(number1, number2)
Min(number1, number2)
Pi()
Rand([ algorithm ])
Randomize(number [, algorithm ])
RandRange(number1, number2 [, algorithm ])
Round(number)
Sgn(number)
Sin(number)
Sqr(number)
Tan(number)
```

Other functions

```
CreateUUID()
DeleteClientVariable("name")
GetBaseTagData(tagname [, instancenumber ])
GetBaseTagList()
GetBaseTemplatePath()
GetClientVariablesList()
PreserveSingleQuotes(variable)
URLSessionFormat(request_URL)
WriteOutput(string)
```

Query functions

```
IsQuery(value)
QueryAddColumn(query, column-name [, datatype], array-name)
QueryAddRow(query [, number ])
QueryNew(columnlist [, columntypelist ])
```

QuerySetCell(*query, column_name, value [, row_number]*)
QuotedValueList(*query.column [, delimiter]*)
ValueList(*query.column [, delimiter]*)

Security functions

Decrypt(*encrypted_string, key [, algorithm [, encoding]]*)
Encrypt(*string, key [, algorithm [, encoding]]*)
GetAuthUser()
GenerateSecretKey(*algorithm*)
GetTempDirectory()
Hash(*string [, algorithm [, encoding]]*)
IsUserInRole("*role_name*")

String functions

ColdFusion MX supports the Java UCS-2 representation of Unicode character values 0–65535. (Earlier releases supported ASCII values.)

String-processing functions process any of these characters (including ASCII 0 (NUL) characters), and continue counting subsequent characters of the string, if any. (In earlier releases, some string-processing functions processed the ASCII 0 (NUL) character, but did not process subsequent characters of the string.)

Asc(*string*)
BinaryDecode(*string, binaryencoding*)
BinaryEncode(*binarydata, encoding*)
CharsetDecode(*string, encoding*)
CharsetEncode(*binaryobject, encoding*)
Chr(*number*)
CJustify(*string, length*)
Compare(*string1, string2*)
CompareNoCase(*string1, string2*)
DayOfWeekAsString(*day_of_week*)
Decrypt(*encrypted_string, key [, algorithm [, encoding]]*)
Encrypt(*string, key [, algorithm [, encoding]]*)
Find(*substring, string [, start]*)
FindNoCase(*substring, string [, start]*)
FindOneOf(*set, string [, start]*)
FormatBaseN(*number, radix*)
GenerateSecretKey(*algorithm*)
GetToken(*string, index [, delimiters]*)
Hash(*string [, algorithm [, encoding]]*)
HTMLCodeFormat(*string [, version]*)
HTMLEditFormat(*string [, version]*)
Insert(*substring, string, position*)
JavaCast(*type, variable*)
JSStringFormat(*string*)
LCase(*string*)
Left(*string, count*)
Len(*string or binary object*)
LJustify(*string, length*)
ListValueCount(*list, value [, delimiters]*)
ListValueCountNoCase(*list, value [, delimiters]*)
LSIsDate(*string*)
LSIsNumeric(*string*)
LSParseCurrency(*string*)
LSIsCurrency(*string*)
LSParseDateTime(*date/time-string*)

LSParseEuroCurrency(*currency-string*)
LSParseNumber(*string*)
LTrim(*string*)
Mid(*string*, *start*, *count*)
MonthAsString(*month_number*)
ParagraphFormat(*string*)
ParseDateTime(*date/time-string* [, *pop-conversion*])
REFind(*reg_expression*, *string* [, *start*]
[, *returnsubexpressions*])
REFindNoCase(*reg_expression*, *string* [, *start*]
[, *returnsubexpressions*])
RemoveChars(*string*, *start*, *count*)
RepeatString(*string*, *count*)
Replace(*string*, *substring1*, *substring2* [, *scope*])
ReplaceNoCase(*string*, *substring1*, *substring2*
[, *scope*])
REReplace(*string*, *reg_expression*, *substring* [, *scope*])
REReplaceNoCase(*string*, *reg_expression*, *substring*
[, *scope*])
ReplaceList(*string*, *list1*, *list2*)
Reverse(*string*)
Right(*string*, *count*)
RJustify(*string*, *length*)
RTrim(*string*)
SpanExcluding(*string*, *set*)
SpanIncluding(*string*, *set*)
StripCR(*string*)
ToBase64(*string or binary_object* [, *encoding*])
ToBinary(*string_in_Base64 or binary_value*)
ToString(*value* [, *encoding*])
Trim(*string*)
UCase(*string*)
URLDecode(*urlEncodedString* [, *charset*])
URLEncodedFormat(*string* [, *charset*])
Val(*string*)
Wrap(*string*, *limit* [, *strip*])
XmlFormat(*string*)

Structure functions

Duplicate(*variable_name*)
IsStruct(*variable*)
StructAppend(*struct1*, *struct2*, *overwriteFlag*)
StructClear(*structure*)
StructCopy(*structure*)
StructCount(*structure*)
StructDelete(*structure*, *key* [, *indicatenotexisting*])
StructFind(*structure*, *key*)
StructFindKey(*top*, *value*, *scope*)
StructFindValue(*top*, *value* [, *scope*])
StructGet(*pathDesired*)
StructInsert(*structure*, *key*, *value* [, *allowoverwrite*])
StructIsEmpty(*structure*)
StructKeyArray(*structure*)
StructKeyExists(*structure*, "*key*")
StructKeyList(*structure* [, *delimiter*])
StructNew()
StructSort(*base*, *sortType*, *sortOrder*, *pathToSubElement*)
StructUpdate(*structure*, *key*, *value*)

System functions

DirectoryExists(*absolute_path*)
Duplicate(*variable_name*)
ExpandPath(*relative_path*)
FileExists(*absolute_path*)
GetBaseTemplatePath()
GetContextRoot()
GetCurrentTemplatePath()
GetDirectoryFromPath(*path*)
GetEncoding(*scope_name*)
GetException(*object*)
GetFileFromPath(*path*)
GetFunctionList()
GetHttpRequestData()
GetLocale()
GetLocaleDisplayName([*locale* [, *inLocale*]])
GetMetaData(*object*)
GetMetricData(*mode*)
GetPageContext()
GetProfileSections(*iniFile*)
GetProfileString(*iniPath*, *section*, *entry*)
GetTempDirectory()
GetTempFile(*dir*, *prefix*)
GetTemplatePath *See* GetBaseTemplatePath
GetTickCount()
SetEncoding(*scope_name*, *charset*)
SetLocale(*new_locale)*
SetProfileString(*iniPath*, *section*, *entry*, *value*)
WriteOutput(*string*)

XML functions

AddSOAPRequestHeader(*webservice*, *namespace*, *name*, *value* [, *mustunderstand*])
AddSOAPResponseHeader(*namespace*, *name*, *value* [, *mustunderstand*])
GetSOAPRequest()
GetSOAPRequestHeader(*namespace*, *name* [, *asXML*])
GetSOAPResponse(*webservice*)
GetSOAPResponseHeader(*webservice*, *namespace*, *name* [, *asXML*])
IsSOAPRequest()
IsXML(*value*)
IsXmlAttribute(*value*)
IsXmlDoc(*value*)
IsXmlElem(*value*)
IsXmlNode(*value*)
IsXmlRoot(*value*)
IsWDDX(*value*)
ToString(*value* [, *encoding*])
XmlChildPos(*elem*, *childName*, *N*)
XmlElemNew(*xmlObj* [, *namespace*], *childName*)
XmlFormat(*string*)
XmlGetNodeType(*xmlNode*)
XmlNew([*caseSensitive*])
XmlParse(*xmlText* [[, *caseSensitive*], *validator*])
XmlSearch(*xmlDoc*, *xPathString*)
XmlTransform(*xml*, *xsl* [, *parameters*])
XmlValidate(*xmlDoc* [, *validator*])

Gateway interface

Methods to implement

```
GatewayName([String id[, String configFile]])
String getGatewayID()
GatewayHelper getHelper()
int getStatus()
String outgoingMessage(coldfusion.
    eventgateway.CFEvent cfmessage)
void restart()
void setCFCListeners(String[]  listeners)
void setGatewayID(String id)
void start()
void stop()
```

IM Gateway CFC incoming message methods

onAddBuddyRequest onAddBuddyRequest(CFEvent)
onAddBuddyResponse onAddBuddyResponse(CFEvent)
onBuddyStatus onBuddyStatus(CFEvent)
onIMServerMessage onIMServerMessage(CFEvent)
onIncomingMessage onIncomingMessage(CFEvent)

IM Gateway message sending commands

submit
accept
decline
noact

Application.cfc

Methods to implement

```
onApplicationEnd
onApplicationStart
onError
onRequest
onRequestEnd
onRequestStart
onSessionEnd
onSessionStart
```

ColdFusion variables

ColdFusion MX returns variables, such as those returned in a `cfdirectory` or `cfftp` operation. A variable is usually referenced by *scoping* it according to its type: naming it according to the code context in which it is available; for example, Session.varname, or Application.varname.

You use the `cflock` tag to limit the scope of CFML constructs that modify shared data structures, files, and CFXs, to ensure that modifications occur sequentially. For more information, see *ColdFusion MX Developer's Guide*.

Variable scope

ColdFusion MX supports the Variables scope. Unscoped variables created with the `cfset` tag acquire the Variables scope by default. For example, the variable created by the statement `<CFSET linguist = Chomsky>` can be referenced as `#Variables.linguist#`.

Caller scope

ColdFusion MX supports the Caller scope as a structure.

Client variables

The following client variables are reserved:

```
Client.CFID
Client.CFToken
Client.HitCount
Client.LastVisit
Client.TimeCreated
Client.URLToken
```

Server variables

Use the Server prefix to reference server variables, as follows:

```
Server.ColdFusion.ProductName
Server.ColdFusion.ProductVersion
Server.ColdFusion.ProductLevel
Server.ColdFusion.SerialNumber
Server.ColdFusion.SupportedLocales
Server.ColdFusion.AppServer
Server.ColdFusion.Expiration
Server.ColdFusion.RootDir
Server.OS.Name
Server.OS.AdditionalInformation
Server.OS.Version
Server.OS.BuildNumber
```

Application and session variables

To enable application and session variables, use the `cfapplication` tag or Application.cfc. Reference them as follows:

```
Application.myvariable
Session.myvariable
```

To ensure that modifications to shared data occur in the intended sequence, use the `cflock` tag.

The predefined application and session variables are as follows:

```
Application.ApplicationName
Session.CFID
Session.CFToken
Session.URLToken
```

Custom tag variables

A ColdFusion custom tag returns the following variables:

```
ThisTag.ExecutionMode
ThisTag.HasEndTag
ThisTag.GeneratedContent
ThisTag.AssocAttribs[index]
```

A custom tag can set a Caller variable to provide information to the caller. The Caller variable is set as follows:

```
<cfset Caller.variable_name = "value">
```

The calling page can access the variable with the `cfoutput` tag, as follows:

```
<cfoutput>#variable_name#</cfoutput>
```

Request variable

Request variables store data about the processing of one page request. Request variables store data in a structure that can be passed to nested tags, such as custom tags, and processed once.

To provide information to nested tags, set a Request variable, as follows:

```
<CFSET Request.field_name1 = "value">
<CFSET Request.field_name2 = "value">
```

```
<CFSET Request.field_name3 = "value">
...
```

Each nested tag can access the variable with the `cfoutput` tag, as follows:

```
<CFOUTPUT>#Request.field_name1#</CFOUTPUT>
```

Form variable

ColdFusion supports the Form variable FieldNames. FieldNames returns the names of the fields on a form. You use it on the action page associated with a form, as follows:

```
Form.FieldNames
```

ColdFusion tag-specific variables

Some ColdFusion tags return data as variables. For example, the `cffile` tag returns file size information in the FileSize variable, referenced as CFFILE.FileSize.

The following tags return data that can be referenced in variables:

```
cfcatch
cfdirectory
cferror
cffile
cfftp
cfhttp
cfindex
cfldap
cfpop
cfquery
cfregistry
cfsearch
cfstoredproc
```

ColdFusion query variables

A ColdFusion tag that returns a query object supports the following variables, where *queryname* is the value of the `name` attribute:

```
queryname.CurrentRow
queryname.RecordCount
queryname.ColumnList
```

CFCATCH variables

Within a `cfcatch` block, the active exception properties can be accessed as the following variables:

```
CFCATCH.Type
CFCATCH.Message
CFCATCH.Detail
CFCATCH.ErrNumber
CFCATCH.NativeErrorCode
CFCATCH.SQLState
CFCATCH.LockName
CFCATCH.LockOperation
CFCATCH.MissingFileName
CFCATCH.TagContext
CFCATCH.ErrorCode
CFCATCH.ExtendedInfo
```

Within a `cfcatch` block, database exception properties can be accessed as the following variables:

```
CFCATCH.QueryError
CFCATCH.SQL
CFCATCH.Where
CFCATCH.Datasource
```

Within a `cfcatch` block, undefined variable exception properties can be accessed as the following variable:

```
CFCATCH.Name
```

Within a `cfcatch` block, syntax and parsing exception properties can be accessed as the following variables:

```
CFCATCH.TokenText
CFCATCH.Snippet
```

```
CFCATCH.Column
CFCATCH.KnownColumn
CFCATCH.Line
CFCATCH.KnownLine
```

CFDIRECTORY variables

The `cfdirectory` tag, with `action=list`, returns a query object as follows, where *`queryname`* is the `name` attribute value:

```
queryname.Name
queryname.Size
queryname.Type
queryname.DateLastModified
queryname.Attributes
queryname.Mode
```

CFERROR variables

When `cferror` generates an error page, the following error variables are available if `type="request"` or `"exception"`.

```
Error.Diagnostics
Error.MailTo
Error.DateTime
Error.Browser
Error.GeneratedContent
Error.RemoteAddress
Error.HTTPReferer
Error.Template
Error.QueryString
```

The following error variables are available if `type="validation"`.

```
Error.ValidationHeader
Error.InvalidFields
Error.ValidationFooter
```

Any `cfcatch` variable that applies to exception type can be accessed within the Error scope, as follows:

```
Error.Type
Error.Message
Error.Detail
Error.ErrNumber
Error.NativeErrorCode
Error.SQLState
Error.LockName
Error.LockOperation
Error.MissingFileName
Error.TagContext
Error.ErrorCode
Error.ExtendedInfo
```

Note: You can substitute the prefix `CFERROR` for `Error`, if `type = "Exception"`; for example, `CFERROR.Diagnostics`, `CFERROR.Mailto`, or `CFERROR.DateTime`.

CFFILE ACTION=Upload variables

File variables are read-only. Use the `CFFILE` prefix to reference file variables; for example, `CFFILE.ClientDirectory`. The `File` prefix is deprecated in favor of the `CFFILE` prefix.

```
CFFILE.AttemptedServerFile
CFFILE.ClientDirectory
CFFILE.ClientFile
CFFILE.ClientFileExt
CFFILE.ClientFileName
CFFILE.ContentSubType
CFFILE.ContentType
CFFILE.DateLastAccessed
CFFILE.FileExisted
CFFILE.FileSize
CFFILE.FileWasAppended
CFFILE.FileWasOverwritten
CFFILE.FileWasRenamed
CFFILE.FileWasSaved
CFFILE.OldFileSize
CFFILE.ServerDirectory
CFFILE.ServerFile
```

```
CFFILE.ServerFileExt
CFFILE.ServerFileName
CFFILE.TimeCreated
CFFILE.TimeLastModified
```

CFFTP error variables

When you use the `cfftp stoponerror` attribute, these variables are populated:

```
CFFTP.Succeeded
CFFTP.ErrorCode
CFFTP.ErrorText
```

CFFTP ReturnValue variable

Some `cfftp` file and directory operations provide a return value, in the variable CFFTP.ReturnValue. Its value is determined by the results of the `action` attribute. When you specify any of the following actions, `cfftp` returns a value:

```
GetCurrentDir
GetCurrentURL
ExistsDir
ExistsFile
Exists
```

CFFTP query object columns

When you use the `cfftp` tag with the `listdir` action, `cfftp` returns a query object, where *`queryname`* is the `name` attribute value, and *`row`* is the row number of each file or directory entry:

```
queryname.Name[row]
queryname.Path[row]
queryname.URL[row]
queryname.Length[row]
queryname.LastModified[row]
queryname.Attributes
queryname.IsDirectory
queryname.Mode
```

CFHTTP variables

A `cfhttp get` operation can return text and binary files. Files are downloaded and the contents stored in a variable or file, depending on the MIME type, as follows:

```
CFHTTP.FileContent
CFHTTP.MimeType
CFHTTP.Header
CFHTTP.ResponseHeader[http_hd_key]
CFHTTP.StatusCode
```

CFLDAP variables

The `cfldap action=query` tag returns information about the LDAP query, as follows:

```
queryname.CurrentRow
queryname.RecordCount
queryname.ColumnList
```

CFPOP variables

The `cfpop` tag returns the following result columns, depending on the `action` attribute value and the use of other attributes, such as `attachmentpath`, where *`queryname`* is the `name` attribute value:

```
queryname.Date
queryname.From
queryname.Body
queryname.Header
queryname.MessageNumber
queryname.ReplyTo
queryname.Subject
queryname.CC
queryname.To
queryname.CurrentRow
queryname.RecordCount
```

queryname.ColumnList
queryname.Attachments
queryname.AttachmentFiles

CFQUERY and CFSTOREDPROC variables

The cfquery tag returns information about the query in this variable:

CFQUERY.ExecutionTime

The cfquery tag uses the query name to scope the following data about the query:

queryname.CurrentRow
queryname.RecordCount
queryname.ColumnList

The cfstoredproc tag returns the following variables:

CFSTOREDPROC.ExecutionTime
CFSTOREDPROC.StatusCode

CFREGISTRY variables

The cfregistry tag returns a query record set that you can reference after executing the GetAll action, as follows, where *queryname* is the name attribute value:

queryname.Entry
queryname.Type
queryname.Value

CFSEARCH variables

A cfsearch operation returns the following variables, where *searchname* is the name attribute value:

searchname.URL
searchname.Key
searchname.Title
searchname.Score
searchname.Custom1 and Custom2
searchname.Summary
searchname.RecordCount
searchname.CurrentRow
searchname.RecordsSearched
searchname.ColumnList

Standard CGI variables

The CGI variables that are available for your use vary with the web server and configuration. This section lists the CGI 1.1 variables that some web servers create when a CGI script is called. Some of the following variables may not be available.

Request

CGI.AUTH_TYPE
CGI.CONTENT_LENGTH
CGI.CONTENT_TYPE
CGI.METHOD
CGI.PATH_INFO
CGI.PATH_TRANSLATED
CGI.QUERY_STRING
CGI.REMOTE_ADDR
CGI.REMOTE_HOST
CGI.REMOTE_USER
CGI.REQUEST_METHOD
CGI.SCRIPT_NAME

Server

CGI.GATEWAY_INTERFACE
CGI.SERVER_NAME
CGI.SERVER_PORT
CGI.SERVER_PROTOCOL
CGI.SERVER_SOFTWARE

Client

```
CGI.CERT_ISSUER
CGI.CERT_SUBJECT
CGI.CLIENT_CERT_ENCODED
CGI.HTTP_ACCEPT
CGI.HTTP_IF_MODIFIED_SINCE
CGI.HTTP_USER_AGENT
```

The `CERT_ISSUER, CERT_SUBJECT, CLIENT_CERT_ENCODED` variables are available only when you use client certificates.

CGI environment variables

When a browser makes a request to a server, the web server and the browser create environment variables. In ColdFusion MX, these variables are referred to as *`CGI environment variables`*. They take the CGI prefix regardless of whether the server uses a server API or CGI to communicate with the ColdFusion server.

Environment variables contain data about the transaction between the browser and the server, such as the IP Address, browser type, and authenticated username. You can reference CGI environment variables for a given page request anywhere in the page. CGI variables are read-only.

Note: The environment variables available to applications depend on the browser and server software.

CGI server variables

The server creates the following common CGI environment variables (some of these are not available with some servers):

```
SERVER_SOFTWARE
SERVER_NAME
GATEWAY_INTERFACE
SERVER_PROTOCOL
SERVER_PORT
REQUEST_METHOD
PATH_INFO
PATH_TRANSLATED
SCRIPT_NAME
QUERY_STRING
REMOTE_HOST
REMOTE_ADDR
AUTH_TYPE
REMOTE_USER
AUTH_USER
REMOTE_IDENT
CONTENT_TYPE
CONTENT_LENGTH
```

CGI client variables

The browser creates and passes in the request header the following common CGI environment variables:

```
HTTP_REFERER
HTTP_USER_AGENT
HTTP_IF_MODIFIED_SINCE
```

CGI client certificate variables

ColdFusion MX makes available the following client certificate data. These variables are available when running Microsoft IIS 4.0 or Netscape Enterprise under SSL if your web server is configured to accept client certificates.

```
CERT_SUBJECT
CERT-ISSUER
CLIENT-CERT-ENCODED
```