

# Premiers pas en ColdFusion

par Ludovic Lefort ([Site web](#)) ([Blog](#))

Date de publication : 08/08/2007

Cet article est adressé aux personnes n'ayant jamais développé en ColdFusion et désirant apprendre les rudiments de ce langage.

- 0 - Présentation de ColdFusion
- 1 - Le langage
  - 1.1 - Le célèbre "Hello world !"
  - 1.2 - Déclaration et utilisation de variables
  - 1.3 - Les opérateurs de comparaison
  - 1.4 - Les opérateurs conditionnels
  - 1.5 - La boucle FOR
- 2 - Application.cfm et onrequestend.cfm
- 3 - Gestion d'erreur
- 4 - Utilisation de formulaire HTML
  - 4.1 - Exemple de GET
  - 4.2 - Exemple de POST
- 5 - Envoyer un mail
- 6 - Travailler avec une base de données
  - 6.1 - Lire les données d'une table
  - 6.2 - Exécuter une commande SQL

## 0 - Présentation de ColdFusion

ColdFusion est un langage à base de balises (comme  **HTML** ou  **XML**) créé en 1995 et actuellement supporté par Adobe.

La force de ce langage et évidemment l'argument commercial de Adobe est la rapidité de développement des pages.

ColdFusion est un langage payant, la dernière version à ce jour est ColdFusion 8.

Pour une présentation plus complète je vous invite à consulter [ce lien](#).

## 1 - Le langage

### 1.1 - Le célèbre "Hello world !"

Une fois n'est pas coutume notre premier exemple affichera simplement le texte : "Hello world!"

```
<cfoutput>
Hello world !
</cfoutput>
```

La balise **<cfoutput>** permet d'écrire sur une page.

### 1.2 - Déclaration et utilisation de variables

En coldfusion comme dans pas mal d'autres langages les variables ne sont pas typées.

La balise **<cfset>** permet de définir et d'attribuer une valeur à une variable.

```
<cfset i=1>
<cfset str="toto">
```

Affichons maintenant les valeurs de nos deux variables :

```
Ma première variable est un entier et sa valeur est : #i#<br>
Ma deuxième variable est une string et sa valeur est : #str#
```

En entourant la nom de ma variable par le symbole # je spécifie que je veux afficher le contenu de ma variable.

Si j'avais omis le # seul le nom de la variable aurait été affichée.

Vous aurez remarqué que la balise **<cfoutput>** accepte les balises  **HTML**.

### 1.3 - Les opérateurs de comparaison

- eq est égal
- neq est différent
- gt plus grand que
- lt plus petit que

### 1.4 - Les opérateurs conditionnels

Les opérateurs conditionnels de ColdFusion sont **<cfif>**, **<cfelse>** et **<cfelseif>**

```
<cfif a gt b>
  <cfoutput>#a# est plus grand que #b#</cfoutput>
</cfif>
```

Nous allons maintenant ajouter un **<cfelse>**

```
<cfif a gt b>
  <cfoutput>#a# est plus grand que #b#</cfoutput>
<cfelse>
  <cfoutput>#a# est plus petit que #b#</cfoutput>
</cfif>
```

Et pour finir un exemple complet: **<cfif>** **<cfelseif>** **<cfelse>**

```
<cfif a gt 10>
  <cfoutput>#a# est plus grand que 10</cfoutput>
<cfelseif a gt 5>
  <cfoutput>#a# est plus grand que 5</cfoutput>
<cfelse>
  <cfoutput>#a# est plus petit que 5</cfoutput>
</cfif>
```

## 1.5 - La boucle FOR

Une boucle peut être définie grâce à la balise **<cfloop>**

```
<cfloop index="i" from="1" to="10" step="1">
  <cfoutput>#i#<br></cfoutput>
</cfloop>
```

Cette boucle affiche les nombres de 1 à 10 l'un en dessous de l'autre.

## 2 - Application.cfm et onrequestend.cfm

Ces deux noms de fichiers sont des noms réservés reconnu par le serveur ColdFusion.

Lors de l'exécution d'une page, le serveur commence par regarder si un fichier nommé Application.cfm existe dans le même répertoire, s'il le trouve il commence par exécuter le code qu'il contient.

De la même manière si le serveur trouve un fichier nommé onrequestend.cfm il l'exécute juste après la page demandée.

Dans cet exemple, je vais utiliser la page application.cfm afin de définir un header commun à toutes les pages

```
<cfapplication name="test">  
<strong>HEADER FROM APPLICATION.CFM</strong><br>
```

La première ligne définit un nom pour que l'application soit reconnue de manière unique auprès du serveur.

Elle est obligatoire si l'on travaille avec des variables de session.

De la même manière je vais créer un fichier nommé onrequestend.cfm qui définira un bas de page commun.

```
<br>  
<strong>FOOTER FROM ONREQUESTEND.CFM</strong>
```

### 3 - Gestion d'erreur

ColdFusion offre la possibilité d'intercepter et de gérer les erreurs pouvant survenir sur une page grâce aux balises **<cftry>** et **<cfcatch>**


Le code risquant de provoquer une exception doit être placé dans une balise **<cftry>**

Le code placé dans la balise **<cfcatch>** sera exécuté uniquement si une exception survient.

```
<cfset a = 2>
<cfset b = 0>
<cftry>
  <cfoutput>#a / b#</cfoutput>
<cfcatch>
  <cfoutput>Erreur : Division par zéro !</cfoutput>
</cfcatch>
</cftry>
```

## 4 - Utilisation de formulaire HTML

### 4.1 - Exemple de GET

Dans cet exemple nous allons créer une page nommée "formGet.cfm" qui contient un form  HTML pointant vers la page elle-même.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
<head>
  <title>Untitled</title>
</head>

<body>
<cfif isdefined("url.maValeur")>
  <cfoutput>La valeur passée est : #url.maValeur#</cfoutput>
<cfelse>
  <form name="formGet" action="formGet.cfm" method="GET">
    <input type="Text" name="maValeur"><br>
    <input type="Submit" value="Ok">
  </form>
</cfif>
</body>
</html>
```

Examinons ce code :

Nous commençons par un **<cfif isdefined("url.maValeur")>** qui permet de tester si la variable url.maValeur est définie

**url** est un objet permettant d'accéder très facilement aux paramètres passés par l'url.

Ensuite si la variable n'est pas définie, on affiche un formulaire, dans le cas contraire on affiche la valeur passée par l'url.

### 4.2 - Exemple de POST

Nous allons maintenant faire la même chose que dans le point 5.1 mais en utilisant un POST.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
<head>
  <title>Untitled</title>
</head>


<body>
<cfif isdefined("form.maValeur")>
  <cfoutput>La valeur passée est : #form.maValeur#</cfoutput>
<cfelse>
  <form name="formPost" action="formPost.cfm" method="POST">
    <input type="Text" name="maValeur"><br>
    <input type="Submit" value="Ok">
  </form>
</cfif>
```

```
</body>  
</html>
```

Le fichier est quasi identique. Seule l'action dans le form change et la manière de récupérer la valeur

Dans un post la valeur n'est pas passée par l'url on n'utilise donc pas l'objet **url** mais **form**

## 5 - Envoyer un mail

Nous allons maintenant décrire comment faire pour envoyer un mail depuis le site en utilisant le client  SMTP du serveur


Encore une fois ColdFusion nous a prémâché le travail en intégrant la balise **<cfmail>**


```
<cfmail from="toto@toto.com" to="titi@titi.com" subject="Sujet du mail">  
Contenu du mail  
à envoyer  
</cfmail>
```

Rien de plus simple ...

## 6 - Travailler avec une base de données

Voici un point très important qui fait toute la puissance de ColdFusion : La facilité avec laquelle on peut travailler avec une base de données.

Avant de pouvoir travailler avec une base de données il faut commencer par créer une connexion vers la db dans votre  **IDE**.

La manipulation étant différente suivant l' **IDE** utilisé, je ne peux que vous conseiller de vous référer à l'aide de votre application.

### 6.1 - Lire les données d'une table

Nous allons dans cet exemple afficher le résultat d'une requête **SELECT** à l'aide de la balise **<cfdump>**

**<cfdump>** permet d'afficher n'importe quel type de variable (simple ou complexe) sur une page.

Parfait pour le debug.

```
<CFQUERY NAME="bookList" DATASOURCE="cfbookclub">
  SELECT * from BOOKS
</CFQUERY>

<cfdump var="#booklist#">
```

Voici comment macromedia justifie le prix de Coldfusion : vous affichez le contenu d'une base de données en quelques lignes de code !

**<cfdump>** étant plutôt indiqué pour le debug nous allons faire la même chose au travers d'un tableau  **HTML** :

```
<CFQUERY NAME="bookList" DATASOURCE="cfbookclub">
  SELECT * from BOOKS
</CFQUERY>

<table border="1">
<cfoutput query="booklist">
  <tr>
    <td>#booklist.BOOKID#</td>
    <td>#booklist.TITLE#</td>
  </tr>
</cfoutput>
</table>
```

Comment balayer toutes les lignes d'une query ? Avec **<cfoutput>**; tout simplement !

En effet la balise **<cfoutput>** accepte un argument **query** qui permet de spécifier quelles données parcourir.

Ensuite les champs récupérés de la db sont accessibles à partir d'un objet portant le même nom que votre query.

### 6.2 - Exécuter une commande SQL

L'envoi d'une commande tel que INSERT, UPDATE, DELETE se fait exactement de la même manière que le SELECT.

```
<CFQUERY NAME="bookList" DATASOURCE="cfbookclub">  
  UPDATE books SET author='toto' where id='4'  
</CFQUERY>
```

Merci à **kikof** pour la relecture de cet article.

