

# Compiler Apache httpd 2.2.x sous Windows



par Fabien Faille ([Tutoriels Web](#))

Date de publication : 03 juillet 2008

Dernière mise à jour :


Comment compiler Apache httpd 2.2.9 sous Windows. Ce tutoriel est basé sur la documentation officielle et se veut un peu plus détaillé et moins théorique.

---

|   |   |
|---|---|
| I - Introduction.....                             | 3 |
| I-A - Préambule.....                              | 3 |
| I-B - Pré-requis.....                             | 3 |
| I-B-1 - Connaissances.....                        | 3 |
| I-B-2 - Environnement de développement.....       | 3 |
| I-B-3 - Sources.....                              | 3 |
| II - zlib.....                                    | 4 |
| II-A - Préparation.....                           | 4 |
| II-B - Compilation.....                           | 4 |
| III - OpenSSL.....                                | 5 |
| III-A - Préparation.....                          | 5 |
| III-B - Compilation du code assembleur (ASM)..... | 5 |
| III-C - Préparation, le retour.....               | 5 |
| III-D - Compilation.....                          | 6 |
| IV - Apache.....                                  | 7 |
| IV-A - Préparation.....                           | 7 |
| IV-B - Compilation.....                           | 7 |
| V - Conclusion.....                               | 9 |
| V-A - Épilogue.....                               | 9 |
| V-B - Remerciements.....                          | 9 |

## I - Introduction

### I-A - Préambule

Ce tutoriel vous explique comment compiler Apache 2.2.9 sous Windows. Nous allons, pour ce faire, utiliser les outils de développement de Microsoft dans leur version Express, ainsi que Perl et Awk. Ce tutoriel est basé sur la  **documentation Apache** et se veut un peu plus détaillé et moins théorique.









### I-B - Pré-requis

#### I-B-1 - Connaissances

Aucune connaissance spécifique à ces outils ne seront cependant nécessaires, les notions de base en compilation et en utilisation de la ligne de commande devraient suffire.

#### I-B-2 - Environnement de développement

La liste suivante vous présente les outils nécessaires à la compilation d'Apache et de ses dépendances sous Windows :

- Microsoft Visual C++ 2008 Express Edition ( [info](#),  [télécharger](#))
- Microsoft Windows SDK v6.1 ( [info](#),  [télécharger](#))
- Awk ( [info](#),  [télécharger](#))
- Perl ( [info](#),  [télécharger](#))

Certaines attentions sont à apporter afin de faciliter la suite de ce tutoriel :

- Microsoft Windows SDK v6.1  
Dans le dossier C:\Program Files\Microsoft SDKs\Windows\v6.1\Include, dupliquez le fichier **WinResrc.h** et nommez-le **winres.h**  
Note : cette modification est inutile pour la compilation d'Apache mais solutionne des problèmes de compilation récurrents, dus aux évolutions des environnements de développement et des SDKs fournis par Microsoft.
- Awk  
L'exécutable doit être nommé **awk.exe** et disponible dans le PATH.  
À copier dans le dossier des binaires de Visual C++, par exemple C:\Program Files\Microsoft Visual Studio 9.0\VC\bin.

#### I-B-3 - Sources

Cette liste vous propose des liens vers les codes sources d'Apache et de ses dépendances. Je vous recommande de toujours télécharger les codes sources (comme n'importe quel outil, logiciel, etc.) depuis le site de l'éditeur.

- Apache 2.2.9 ( [info](#),  [télécharger](#))
- zlib 1.2.3 ( [info](#),  [télécharger](#))
- OpenSSL 0.9.8h ( [info](#),  [télécharger](#))

## II - zlib

### II-A - Préparation

La zlib est utilisée pour la compilation du module **mod\_deflate** ainsi que pour OpenSSL. La compilation de la zlib est relativement simple et ressemble plus à une formalité qu'à autre chose. Admettons que nous travaillons dans un dossier C:\httpd\_build fraîchement créé pour l'occasion, nous décompresserons l'archive de la zlib dans C:\httpd\_build\zlib.

### II-B - Compilation

Ouvrez une 'Invite de commandes de Visual Studio 2008' et rendez-vous dans le dossier précédemment cité. Pour lancer la compilation, entrez la commande suivante :

```
nmake -f win32\Makefile.msc
```

La compilation est relativement rapide. Exécutez la commande suivante pour vérifier que la compilation a produit un code correct :

```
nmake -f win32\Makefile.msc test
```

La commande doit retourner quelque-chose comme suit :

```
zlib version 1.2.3 = 0x1230, compile flags = 0x55
uncompress(): hello, hello!
gzread(): hello, hello!
gzgets() after gzseek: hello!
inflate(): hello, hello!
large_inflate(): OK
after inflateSync(): hello, hello!
inflate with dictionary: hello, hello!
    echo hello world | minigzip | minigzip -d
hello world
```

## III - OpenSSL

### III-A - Préparation

OpenSSL est utilisé pour la compilation du module **mod\_ssl** ainsi que pour **ApacheBench/SSL**. Décompressons tout d'abord l'archive des sources dans un autre sous-dossier de notre répertoire de travail, prenons C:\httpd\_build\OpenSSL. Rendons-nous dans le dossier C:\httpd\_build\OpenSSL\crypto\sha\asm et éditons le fichier **sha1-586.pl** : en ligne **152**, il faut retirer le second argument de la déclaration pour obtenir le code suivant :


```
&function_begin("sha1_block_data_order");
```

Ce fichier modifié, nous pouvons passer à la configuration de la compilation. En ligne de commandes depuis le dossier C:\httpd\_build\OpenSSL, entrez la commande suivante :

```
perl Configure no-mdc2 no-rc5 no-idea enable-zlib VC-WIN32 -I../zlib
```


Cette commande nous permet de désactiver les pans de code qui sont brevetés, d'indiquer que nous souhaitons compiler avec Microsoft Visual C++ pour une plateforme 32bits et de donner le chemin vers la zlib précédemment compilée.

### III-B - Compilation du code assembleur (ASM)

Le code source d'OpenSSL comporte de multiples versions des mêmes fonctions codées en (au moins) deux langages : l' **assembleur** et le **C**. Nous suivrons ici les conseils de la documentation d'Apache et nous réaliserons la compilation des fonctions codées en assembleur. Les versions codées en C ne seront pas incluses dans les binaires résultants.

```
ms\do_masm.bat
```

### III-C - Préparation, le retour

Le **makefile** ( **info**) a été généré à l'étape précédente et doit être modifié pour que la compilation soit un succès. Ouvrir le fichier **ntdll.mak** qui se trouve dans le dossier **ms**, chercher la ligne contenant **zlib1.lib** et remplacer par **C:/httpd\_build/zlib/zdll.lib** (attention, ce sont bien des slashes : /) afin d'obtenir quelque chose comme suit :

```
$(SHLIB_EX_OBJ) $(CRYPTOOBJ) wsock32.lib gdi32.lib advapi32.lib user32.lib C:/httpd_build/zlib/  
zdll.lib
```

Nous reste à modifier le fichier qui commande la suite de tests pour enlever les parties relatives aux pans de code désactivés. Ouvrir le fichier **test.bat** et enlever les trois lignes suivantes :

```
echo ideatest  
ideatest  
if errorlevel 1 goto done
```

## III-D - Compilation

Attaquons-nous à la compilation des fonctions qui sont uniquement présentes en C. Depuis la ligne de commandes dans le dossier C:\httpd\_build\OpenSSL, entrez la commande suivante :

```
nmake -f ms\ntdll.mak
```

Cette étape est relativement longue mais doit se terminer sans message d'erreur. Une fois la compilation terminée, passons les tests pour vérifier le bon fonctionnement de nos binaires OpenSSL en entrant la commande suivante :

```
nmake -f ms\ntdll.mak test
```

Tous les tests vont défiler et si tout va bien, la dernière ligne sera :

```
passed all tests
```

## IV - Apache

### IV-A - Préparation

Nous approchons du but. Décompressons l'archive des sources d'Apache dans un nouveau sous-dossier de notre répertoire de travail, admettons C:\httpd\_build\httpd. Apache détectera automatiquement lors de sa compilation la présence de la zlib et d'OpenSSL à condition que ceux-ci se trouvent dans le sous-dossier **srclib** d'Apache et qu'ils soient bien nommés **zlib** et **OpenSSL**. Copions donc les dossiers au bon endroit, sans faire un simple déplacement, afin de pouvoir les réutiliser lors de la sortie de la prochaine version d'Apache ;)

Notre répertoire de travail et ses sous-dossiers devraient donc ressembler à ceci :

```
C:\httpd_build\ --- zlib\ ...
                  |- OpenSSL\ ...
                  |- httpd\ --- ...
                     |- srclib\ --- ...
                        |- zlib\ ...
                        |- OpenSSL\ ...
```

### IV-B - Compilation

Diverses commandes et options de compilation sont disponibles et la ligne de commandes à adopter va différer selon vos besoins.

#### Syntaxe de la commande de compilation

```
nmake -f Makefile.win option1=valeur options2=valeur ... commande
```

#### Les commandes de compilation

|                       |   |
|-----------------------|---|
| <code>_apacher</code> | compile Apache en Release   |
| <code>_apached</code> | compile Apache en Debug   |
| <code>installr</code> | compile et installe Apache en Release                                       |
| <code>installd</code> | compile et installe Apache en Debug   |
| <code>clean</code>    | supprime (le plus possible) de fichiers générés                             |
| <code>_cleanr</code>  | supprime (le plus possible) de fichiers générés par une compilation Release |
| <code>_cleand</code>  | supprime (le plus possible) de fichiers générés par une compilation Debug   |
| <code>_browse</code>  | parcourt le fichier d'informations de compilation                           |

#### Les options de compilation

| Option                  | Valeur par défaut |
|-------------------------|-------------------|
| <code>INSDIR</code>     | \Apache22         |
| <code>PORT</code>       | 80                |
| <code>SSLPORT</code>    | 443               |
| <code>DOMAINNAME</code> | example.com       |
| <code>SERVERNAME</code> | www.example.com   |
| <code>SERVENAME</code>  | admin@example.com |

Nous partons du principe que nous souhaitons installer notre Apache dans le dossier C:\httpd\_build\Apache22. Depuis la ligne de commandes, dans le dossier C:\httpd\_build\httpd, entrer la commande suivante :

```
nmake /f Makefile.win INSDIR="C:\httpd_build\Apache22" installr
```


Comme pour OpenSSL, la compilation est relativement longue mais devrait se terminer sans encombres.

## V - Conclusion

### V-A - Épilogue

Nous sommes finalement arrivés à la fin de ce petit tutoriel. Vous disposez à présent d'un Apache qui présentera certainement des performances légèrement supérieures au binaire livré sur le site et ce, uniquement en faisant appel à des outils gratuits. Vous pouvez par ailleurs utiliser ces binaires pour compiler d'autres projets liés comme PHP.

### V-B - Remerciements

Une spéciale-dédicace à  **Guillaume Rossolini** qui guide, conseille, relit, etc. Merci à l'équipe Developpez.com qui m'accueille pour ce premier tuto sur leur site. Un p'tit clin d'oeil à l'équipe **php-dev-win** et plus particulièrement à Pierre qui a motivé ces lignes.