On souhaite tout d'abord obtenir un plugin de base, nous souhaitons pouvoir rechercher les tags par plusieurs champs. Pour le moment ces champs seront :

- Patient
- ResponsiblePerson
- ResponsiblePersonRole
- Breed
- Species
- Study
- Series
- Instances

Certaines explications afin de coder un plugin sont ici :
http://www.codeproject.com/Articles/797118/Implementing-a-WADO-Server-using-Orthanc

Voici donc les 4 composants obligatoires :

Dependencies

The sample code depends on 4 components:

- The Orthanc Plugin SDK from Orthanc ≥ 0.8.0.
- The CImg library is used to convert the DICOM images from PNG (as provided by Orthanc) to JPEG (as required by WADO).
- The JsonCpp library is used to parse the JSON files that are provided by the Orthanc core.
- The project is built with CMake.

Furthermore, when targeting Windows, the sample code will statically link against libpng, libjpeg and zlib. These components are required by CImg to convert from PNG to JPEG. As consequence, this sample code also illustrates how to compile all these third-party libraries using CMake.

Cependant un de ces composants ne sont pas obligatoire pour nous : CImg library ne nous sert pas encore sauf si l'on souhaite faire un viewer.

Si vous souhaitez compiler Orthanc à partir de la CMakeLists.txt, voici la doc :
https://drive.google.com/open?id=1nDV52v7TnsaaSFz4vZuZd267_eG7UDRuzQ5nk99J-us

Voici un autre lien menant vers l'analyse du plugin SDK, ainsi que d'autres informations utiles :
https://docs.google.com/document/d/11s4icq3KLAoNiPeSC2Z4zZBMxdNPDl9g4fE3HGQwtus/edit

Suite à l'écriture de ces fonctions importantes, je me suis demandé où était appelé le fichier Plugin.cpp pour que cela fonctionne. Suite à une recherche, je pense que le fichier est appelé depuis le CMakelists.txt de WebViewer :

```
177 add_library(OrthancWebViewer
178   SHARED
179   ${CORE_SOURCES}
180   ${AUTOGENERATED_SOURCES}
181   ${CMAKE_SOURCE_DIR}/Plugin/Plugin.cpp
182
```

C'est pour cela qu'il faudra que je me forme sur l'écriture d'un CMakeLists.txt afin de pouvoir build le projet convenablement. (Tuto : http://florian-goujeon.developpez.com/cours/cmake/initiation/ ou toutes les commandes Cmake : http://www.cmake.org/cmake/help/v3.0/index.html)

Obj : Reprendre le CMakelist.txt du webviewer et l'adapter pour compiler notre plugin 4vet.

- Je pense que je dois réfléchir au fichier qu'on souhaite garder pour le mettre dans notre plugin, pour cela je souhaite ne garder pour l'instant les fichiers qui ne servent que pour l'exécution du plugin et donc les fichiers liés aux 4 fonctions obligatoires du plugin qui sont :

4 fonctions à insérer :

1. **int32_t OrthancPluginInitialize(const OrthancPluginContext* context)**: This function is invoked by Orthanc when it loads the plugin on startup. The plugin must store the context pointer so that it can use the plugin services of Orthanc. It must also register all its callbacks using OrthancPluginRegisterRestCallback().
2. **void OrthancPluginFinalize()**: This function is invoked by Orthanc during its shutdown. The plugin must free all its memory.
3. **const char* OrthancPluginGetName()**: The plugin must return a short string to identify itself.
4. **const char* OrthancPluginGetVersion()** The plugin must return a string containing its version number.

- Je pense aussi tester l'exemple d'un plugin de base (Orthanc/Plugins/Samples/Basic), le CMakeLists.txt me parait plus simple à comprendre pour l'instant.

TEST DE COMPILATION DU SAMPLE BASIC D'ORTHANC

(N'oubliez pas d'utiliser pour mieux comprendre la signification des termes : http://www.cmake.org/cmake/help/v3.0/index.html)

N'oubliez pas tout d'abord de changer le plugin dans le fichier de Configuration.json d'Orthanc :

```
35
36  // List of paths to the plugins that are to be loaded into this
37  // instance of Orthanc (e.g. "./libPluginTest.so" for Linux, or
38  // "./PluginTest.dll" for Windows). These paths can refer to
39  // folders, in which case they will be scanned non-recursively to
40  // find shared libraries.
41  "Plugins" : [
42  //"/home/tt/Bureau/4vet/build/libOrthancWebViewer.so"
43  "/home/tt/Bureau/Plugins/Samples/Basic/build/libPluginTest.so"
44  ],
```

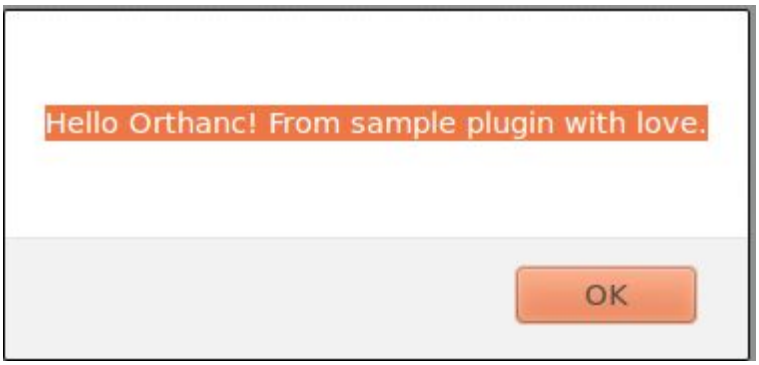
Faites attention à ne pas mettre deux plugins à la fois, cela n'a pas fonctionné pour moi.

Il vous suffit tout simplement de compiler comme pour Orthanc en suivant ce tutoriel : https://orthanc.googlecode.com/hg/LinuxCompilation.txt

Voici la partie de code que l'on voit lorsqu'on lance Orthanc avec sa configuration :

```
388   /* Declare several properties of the plugin */
389   OrthancPluginSetRootUri(context, "/plugin/hello");
390   OrthancPluginSetDescription(context, "This is the description of the sample plugin that can be seen in
    Orthanc Explorer.");
391   OrthancPluginExtendOrthancExplorer(context, "alert('Hello Orthanc! From sample plugin with love.');");
```

Vous devriez donc obtenir ceci :



Voici son CMakeLists.txt (dans le dossier Plugins) qui est comme le dit son nom Basique :

cmake_minimum_required(VERSION 2.8)

project(Basic)

Set a CMake, cache or environment variable to a given value.
set(SAMPLES_ROOT ${CMAKE_SOURCE_DIR}/..) # CMAKE_SOURCE_DIR : The path to the top level of the source tree.
include(${SAMPLES_ROOT}/Common/OrthancPlugins.cmake) # Inclusion du fichier OrthancPlugins.cmake contenant d'autres instructions

add_library(PluginTest SHARED Plugin.c) # Ajoute la librairie PLuginTest au Plugin.c, "SHARED libraries are linked dynamically and loaded at runtime" : http://www.cmake.org/cmake/help/v3.0/command/add_library.html?highlight=shared, cette fonction crée le fichier libPluginTest.so qu on utilise dans Configuration.json d Orthanc

CONCLUSION :

On possède donc un code plus simple à comprendre et adapter que celui du plugin Webviewer. Une fois notre plugin 4vet de base afficher (accés par le bouton 4vet) , nous pourrons nous concentrer sur la base de données.

J'ai installé Netbeans sur Linux le 02/09/2015 en suivant ce turoriel : http://doc.ubuntu-fr.org/netbeans

Ensuite les étapes sont les même que pour windows pour build et compiler, dont j'ai fait un tutoriel : https://drive.google.com/open?id=1yVgyl00uX5MIh04c1mlIKB3eRFsp5jFuuB3xarT4qvk

On a donc réussi à afficher un bouton 4vet : https://docs.google.com/document/d/13w9l0ohoOQGldEnThsG8k6nx6kH2olyz_4OR3PHMow4/edit

Mais je me demande où est-ce que la page explorer.html est appelé afin qu'elle s'affiche dans Orthanc ?

Résultat recherche :
OrthancExplorer.js est appelé depuis le CMakeLists.txt du webviewer :

```
90 EmbedResources(
91   ORTHANC_EXPLORER  ${CMAKE_SOURCE_DIR}/Resources/OrthancExplorer.js
92   JAVASCRIPT_LIBS   ${JAVASCRIPT_LIBS_DIR}
93   ${EMBEDDED_RESOURCES}
94   )
```

Cependant je pense qu'on a besoin de la bibliothèque Js pour que cela fonctionne c'est pour cela qu'il y a cela :

```
60 include(${CMAKE_SOURCE_DIR}/Resources/CMake/GdcmConfiguration.cmake)
61 include(${CMAKE_SOURCE_DIR}/Resources/CMake/LibJpegConfiguration.cmake)
62 include(${CMAKE_SOURCE_DIR}/Resources/CMake/JavaScriptLibraries.cmake)
```

J'ai alors prit le soin de rajouter les fichiers manquants qui se faisaient appeler :
JavaScriptLibraries.cmake et OrthancExplorer.js (qui fait appelle lui aussi à 4vet.html)

Cependant lors de la compilation en changeant notre fichier Cmakelists.txt de 4vet_01, une erreur apparaît, voici donc notre fichier modifié /Plugins/Samples/Basic/CMakeLists.txt (mais ne fonctionne pas encore) :

```
1  cmake_minimum_required(VERSION 2.8)
2
3  project(Basic)
4
5
6  # Set a CMake, cache or environment variable to a given value.
7  set(SAMPLES_ROOT ${CMAKE_SOURCE_DIR}/..) # CMAKE_SOURCE_DIR : The path to the top
8  include(${SAMPLES_ROOT}/Common/OrthancPlugins.cmake) # Inclusion du fichier Ortha
9  #include(${SAMPLES_ROOT}/Resources/CMake/JavaScriptLibraries.cmake) #Va chercher
10
11
12 set(EMBEDDED_RESOURCES
13     4VET  ${CMAKE_SOURCE_DIR}/WebApplication
14     )
15
16 #Fonction appelant le fichier Orthanc_Explorer.js qui est notre bouton.
17 EmbedResources(
18   ORTHANC_EXPLORER  ${CMAKE_SOURCE_DIR}/Resources/OrthancExplorer.js
19   JAVASCRIPT_LIBS   ${JAVASCRIPT_LIBS_DIR}
20   ${EMBEDDED_RESOURCES}
21   )
22
23 add_library(PluginTest SHARED Plugin.c) # Ajoute la librairie PLuginTest au Plugi
```

Cependant en mettant en commentaire ceci :

#Fonction appelant le fichier Orthanc_Explorer.js qui est notre bouton.
EmbedResources(

```
ORTHANC_EXPLORER  ${CMAKE_SOURCE_DIR}/Resources/OrthancExplorer.js
JAVASCRIPT_LIBS   ${JAVASCRIPT_LIBS_DIR}
${EMBEDDED_RESOURCES}
)
```

Je remarque qu'une erreur est tjrs là, je pense donc que l'erreur provient de ce fichier : JavaScriptLibraries.cmake

Voici ce que j'ai observé :
set(BASE_URL "http://www.montefiore.ulg.ac.be/~jodogne/Orthanc/ThirdPartyDownloads/WebViewer")

Je pense que le pb vient de cette commande qui fait que je ne peux pas accéder à celui-ci. Le reste du code étant des paquets téléchargés dans le dossier ThirdPartyDownloads, j'ai juste rajouter ce dossier dans notre dossier 4vet_01 mais je pense que je dois aussi rajouter une ligne disant d'aller chercher dans ce dossier nos paquets lors de la construction avec Cmake (**Pas sur**)

J'ai pense avoir trouvé l'erreur :

```
44 set(ORTHANC_ROOT ${CMAKE_SOURCE_DIR}/Orthanc)
45 include(CheckIncludeFiles)
46 include(CheckIncludeFileCXX)
47 include(CheckLibraryExists)
48 include(FindPythonInterp)
49 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/Compiler.cmake)
50 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/AutoGeneratedCode.cmake)
51 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/DownloadPackage.cmake)
52
53 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/BoostConfiguration.cmake)
54 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/GoogleTestConfiguration.cmake)
55 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/JsonCppConfiguration.cmake)
56 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/LibPngConfiguration.cmake)
57 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/ZlibConfiguration.cmake)
58 include(${CMAKE_SOURCE_DIR}/Orthanc/Resources/CMake/SQLiteConfiguration.cmake)
59
60 include(${CMAKE_SOURCE_DIR}/Resources/CMake/GdcmConfiguration.cmake)
61 include(${CMAKE_SOURCE_DIR}/Resources/CMake/LibJpegConfiguration.cmake)
62 include(${CMAKE_SOURCE_DIR}/Resources/CMake/JavaScriptLibraries.cmake)
63
```

Cela doit être ici que le CMakeLists.txt de webviewer permet de télécharger les différents paquets surtout avec include DownloadPackage.cmake(l. 51)

En compilant le construisant le projet avec Netbeans voici l'erreur qu'il me marque :

CMake Error at CMakeLists.txt:18 (EmbedResources):
Unknown CMake command "EmbedResources".

-- Configuring incomplete, errors occurred!
make: * [cmake_check_build_system] Error 1**