

Dans ce document vous y trouverez mes avancées dans le développement d'un plugin pour Orthanc.

Analyse du plugin SDK (Documentation) :

Attention : durant le développement en C++, il y aura quelques règles de style à respecter : <https://code.google.com/p/orthanc/wiki/CodingStyle>

On peut aussi trouver ici un exemple d'un plugin :

<https://code.google.com/p/orthanc/source/browse/Plugins/Samples/Basic/Plugin.c>

Vous pouvez trouver ici un exemple d'un plugin installé, de plus dans cette page on peut trouver la documentation du plugin SDK qu'on utilisera pour le notre:

<http://www.codeproject.com/Articles/797118/Implementing-a-WADO-Server-using-Orthanc>

Dependencies

The sample code depends on 4 components:

- The **Orthanc Plugin SDK** from Orthanc $\geq 0.8.0$.

Cliquez sur le lien comme au dessus pour accéder à la documentation du plugin SDK.

Une fois le document télécharger, cliquez sur le document et cela mènera un une page HTML :



Orthanc Plugin SDK

Documentation of the plugin interface of Orthanc


Main Page	Modules	Classes	Files
-----------	---------	---------	-------

Orthanc Plugin SDK Documentation

This C/C++ SDK allows external developers to create plugins that can be loaded into Orthanc to extend its functionality. Each Orthanc plugin must expose 4 public functions with the following signatures:

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.

The name and the version of a plugin is only used to prevent it from being loaded twice.

Generated on Tue Jul 15 2014 17:01:35 for Orthanc Plugin SDK by  1.8.1.2

4 fonctions à insérer :

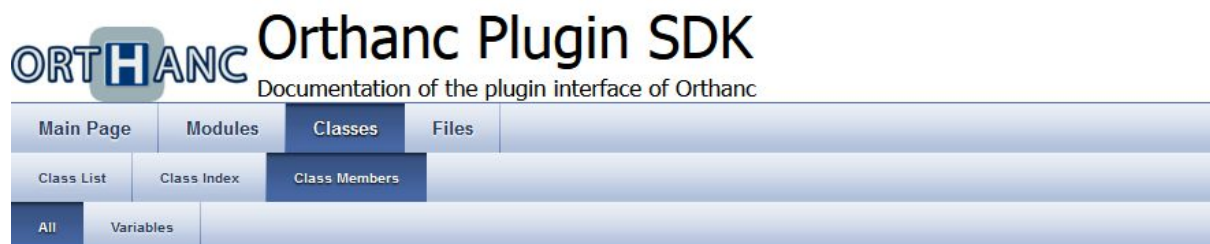
1. `int32_t OrthancPluginInitialize(const OrthancPluginContext* context)`: This function is invoked by Orthanc when it loads the plugin on startup. The plugin must store the context pointer so that it can use the plugin services of Orthanc. It must also register all its callbacks using `OrthancPluginRegisterRestCallback()`.

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.

Dans la documentation du plugin SDK, on trouve dans la catégorie **module**, une aide afin de d'effectuer l'interface en C. Attention cette partie est importante car il y existe certaines fonctions à utiliser "obligatoirement". Vous les trouverez à la fin de cette même page.

Puis on a la catégorie **files** dans laquelle on peut trouver le fichier suivant :
OrthancCPlugin.h (décrit le prototype des fonctions)

Dans la catégorie **classes** : On y trouve comme dit son nom les différentes classes avec leur rapide description. Dans celui-ci se trouve une sous- partie plus intéressante, **Class Members** qui liste toutes les class membres du plugin d'Orthanc.



Here is a list of all documented class members with links to the class documentation for each member:

- body : [OrthancPluginHttpRequest](#)
- bodySize : [OrthancPluginHttpRequest](#)
- data : [OrthancPluginMemoryBuffer](#)
- getCount : [OrthancPluginHttpRequest](#)
- getKeys : [OrthancPluginHttpRequest](#)
- getValues : [OrthancPluginHttpRequest](#)
- groups : [OrthancPluginHttpRequest](#)
- groupsCount : [OrthancPluginHttpRequest](#)
- method : [OrthancPluginHttpRequest](#)
- size : [OrthancPluginMemoryBuffer](#)

.....