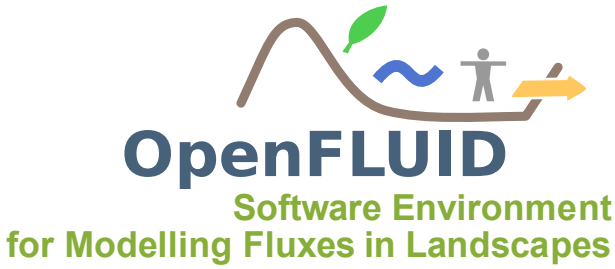


TP1 : Création d'un simulateur

Objectifs:	Créer la structure d'un simulateur et sa signature, construire et compiler pour une utilisation avec le moteur
Pré-requis:	TP0

Les développements s'appuient sur la suite de compilation/construction/test/packaging CMake. CMake est disponible sur http://www.cmake.org .

1 Créer un simulateur

Il existe 2 façons de créer le code source d'un simulateur:

- Ecrire le code source "à la main", à partir d'un code vierge : long, fastidieux, source d'erreurs, ...
- utiliser l'environnement de développement OpenFLUID : facile, assisté, intégré à l'environnement OpenFLUID, ...

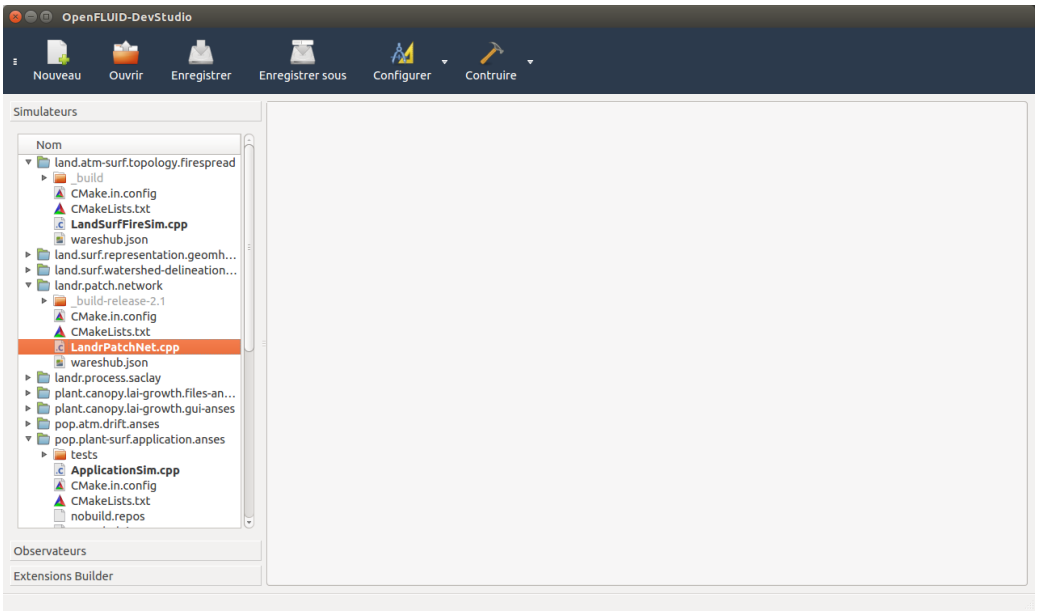
Nous allons utiliser cette 2ème possibilité pour cet exercice.

1.1 Lancement de l'environnement de développement OpenFLUID

L'environnement de développement OpenFLUID est disponible sous deux formes:

- intégré dans l'interface OpenFLUID-Builder,
- en tant qu'outil indépendant : OpenFLUID-DevStudio.

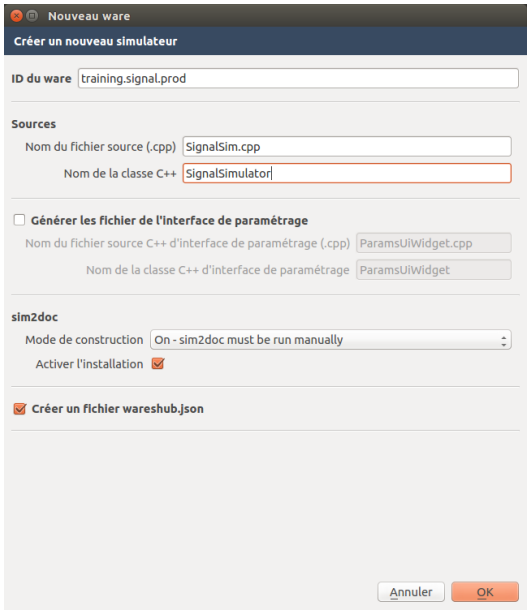
Dans un premier temps, nous utiliserons l'interface d'OpenFLUID-DevStudio puis dans la suite des TP l'interface intégrée à OpenFLUID-Builder. Lancer `OpenFLUID-DevStudio` à partir de l'icône sur votre bureau ou à partir de la liste des programmes de l'ordinateur.



1.2 Création d'un nouveau simulateur

Nous allons tout d'abord générer le code source "vide" d'un simulateur. Dans OpenFLUID-DevStudio, aller dans Fichier/ Nouveau ware/ Simulateur.

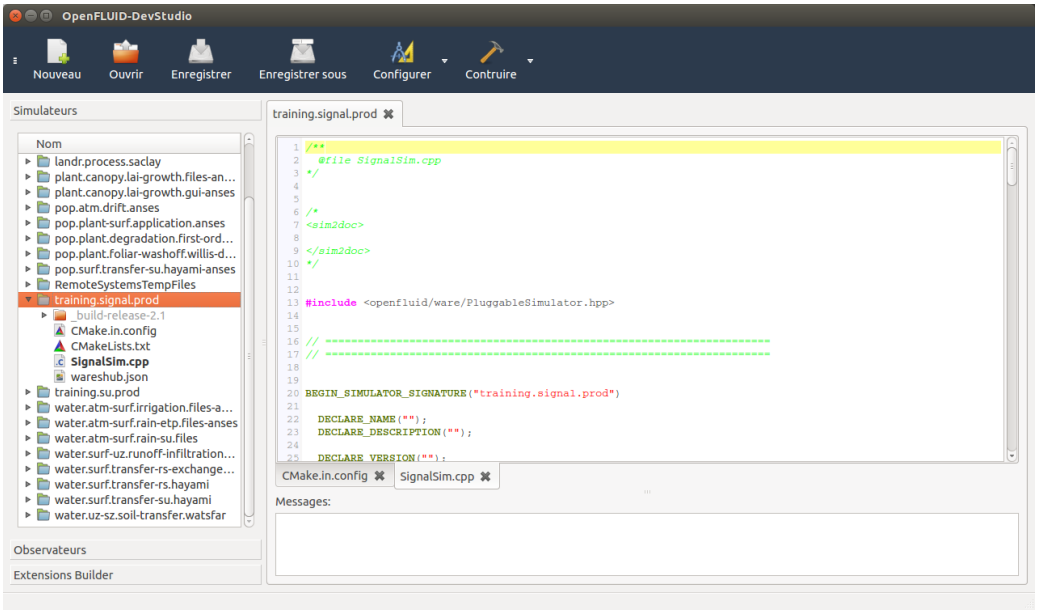
Dans la fenêtre qui s'ouvre, dans *ID du Ware* indiquer le nom du simulateur `training.signal.prod`. Dans *Nom du fichier source (.cpp)* indiquer `SignalSim.cpp` comme nom de fichier, dans *Nom de la classe C++* indiquer `SignalSimulator` comme nom de classe C++ qui contiendra le simulateur. Laisser les autre options par défaut et cliquer sur OK.



Vous devriez obtenir un dossier `training.signal.prod` contenant 4 fichiers :

- `SignalSim.cpp` : fichier source du simulateur,

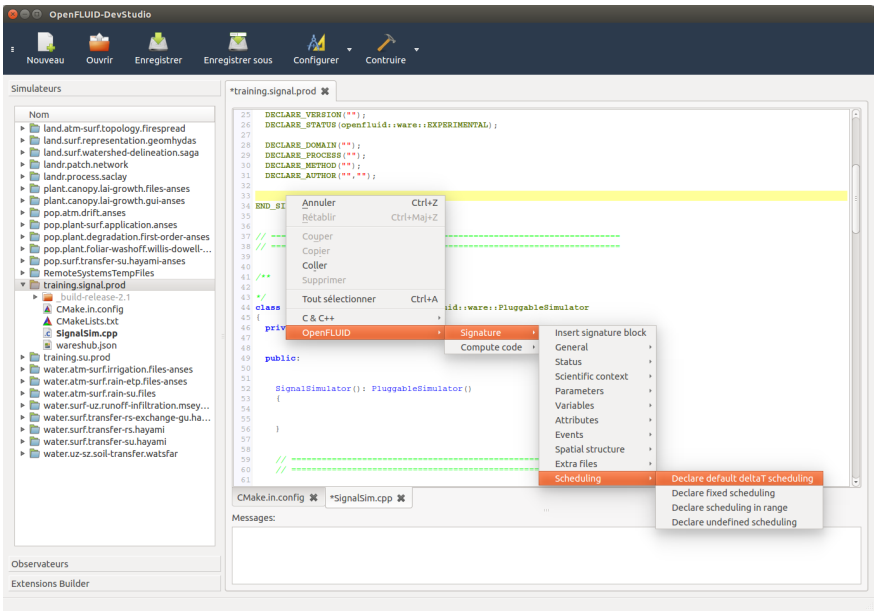
- `CMakeLists.txt` : fichier du système de construction du simulateur (à priori, à ne pas modifier),
- `CMake.in.config` : fichier de configuration de la construction du simulateur (à priori, à ne pas modifier),
- `wareshub.json` : fichier qui contient les métadonnées du simulateur (non utilisé durant la formation).



1.3 Mise à jour de la signature du simulateur

La signature du simulateur comporte des champs facultatifs qui permettent de renseigner sur la description, le statut, l'auteur... Remplir le champ `DECLARE_AUTHOR` en indiquant votre nom et un email. Vous pouvez également remplir les champs `DECLARE_NAME` et `DECLARE_DESCRIPTION` afin d'indiquer des informations sur le nom complet et la description du simulateur.

Nous allons indiquer également que le simulateur fonctionnera avec le pas de temps par défaut de la simulation à l'aide de l'instruction `DECLARE_SCHEDULING_DEFAULT`. Pour écrire cette instruction, passer de préférence par le menu contextuel que propose OpenFLUID-DevStudio afin d'éviter les erreurs: pour cela, faire un clic-droit à l'endroit souhaité, puis aller dans OpenFLUID/ Signature/ Scheduling et sélectionner `Declare default deltaT scheduling`.

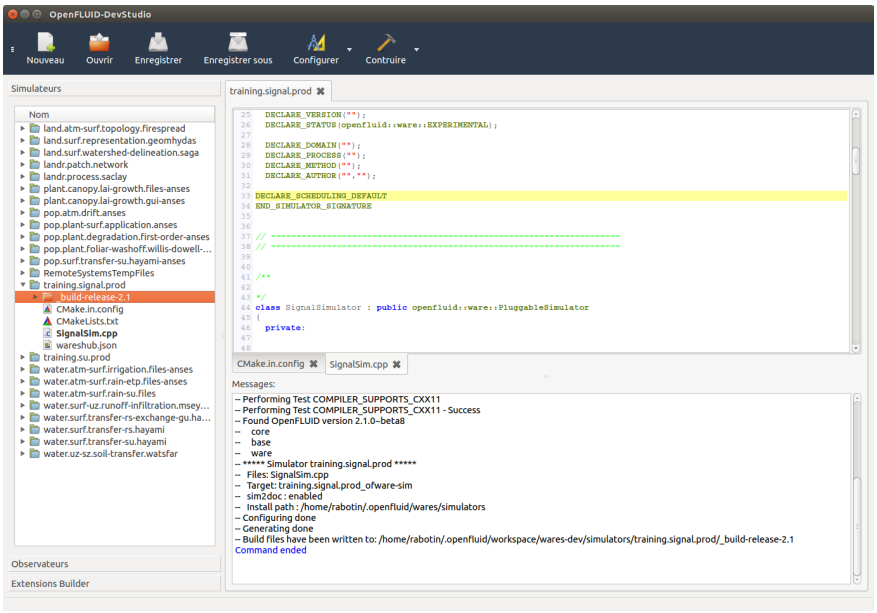


Placer cette instruction avant la fin de la signature signalée par l'appel à `END_SIMULATOR_SIGNATURE`. La signature de votre simulateur doit ressembler à cela :

```
BEGIN_SIMULATOR_SIGNATURE("training.signal.prod");
  DECLARE_NAME("");
  DECLARE_DESCRIPTION("Training␣simulator");

  DECLARE_VERSION("");
  DECLARE_STATUS(openfluid::ware::EXPERIMENTAL);

  DECLARE_DOMAIN("");
  DECLARE_PROCESS("");
  DECLARE_METHOD("");
  DECLARE_AUTHOR("Doe␣J.","doe@foobar.org");

  //Scheduling
  DECLARE_SCHEDULING_DEFAULT;
END_SIMULATOR_SIGNATURE
```

2 Configurer, construire et installer le simulateur

Pour configurer le simulateur, cliquer sur l'icône `Configurer`. Le résultat de la configuration apparaît dans la partie *Messages* et un nouveau dossier `build-release-2.1` est apparu dans le dossier du simulateur.



Cette étape ne sera à faire qu'une seule fois.

A l'icône `Construire`, choisir *Construire et installer* et cliquer sur l'icône `Construire`. La construction et l'installation du simulateur s'effectuent. Si des erreurs sont présentes dans le code, celles-ci seront notifiées lors de la construction.

Note: A chaque modification du code du simulateur, il faut sauvegarder les modifications du fichier et construire pour appliquer les modifications

.

Pour vérifier que le simulateur a été correctement construit et installé, exécuter la commande suivante dans un terminal:
`openfluid report simulators --list`
Le simulateur devrait alors apparaître dans la liste.