|  |
| --- |
|  |
| Application Type |
| Génération avec PowerDesigner |
|  |
| **Thomas Auguey** |
| **20/08/2015** |

|  |
| --- |
| Aide à l’implémentation basée sur la génération de code. |

SOMMAIRE

[Introduction 3](#_Toc433982214)

[Lexique 4](#_Toc433982215)

[Situations utiles 5](#_Toc433982216)

[Model physique de données 6](#_Toc433982217)

[Templates 6](#_Toc433982218)

[Model orienté objet 7](#_Toc433982219)

[Templates 7](#_Toc433982220)

[Stéréotypes 8](#_Toc433982221)

[Options de génération 8](#_Toc433982222)

[Persistance 8](#_Toc433982223)

# Introduction

Cette section est un support à l’outil de modélisation PowerDesigner (PowerAMC dans la version française). PowerDesigner est outil de support au développeur pour modéliser les modèles UML et générer le code de l’implémentation finale.

PowerDesigner permet de modéliser de nombreux schémas dont principalement:

* Les cas d’utilisations
* Les modèles physiques de données
* Les diagrammes de classes

Le principal point fort du logiciel étant de pouvoir exporter ces model dans des documents personnalisées par des Templates programmables (codage de l’implémentation, document de référence, …). On appréciera notamment la génération des diagrammes de classes en code compilable et les modèles de données en code SQL. L’inverse étant également possible pour les projets existants.

Notez que l’utilisation de cet outil de modélisation n’est pas indispensable mais permet un gain de temps considérable lors des phases de codages et de modélisations du projet.

# Lexique

ORM : Mapping objet-relationnel. Technique de programmation permettant de faire correspondre des données entre les systèmes de types incompatibles. Par exemple le mapping d’une base de données relationnelle en objets du langage de programmation.

SGBD : Système de gestion de base de données

# Situations utiles

PowerDesigner est utile dans plusieurs situations :

* Générer le code SQL depuis un modèle physique de données
* Générer le code Objet depuis un diagramme de classes
* Générer le modèle physique de données depuis une base de données

Vous développez une application cliente autour d’une base de données existante:

1. Générer un model physique de données à partir d’une connexion au SGBD
2. Convertir le model physique de données en model orienté objet (diagramme de classes)
3. Générer le code du model de données (application cliente)

Vous développez une application cliente et une base de données:

1. Modéliser un model conceptuel de données
2. Générer le model physique de données
3. Générer le model orienté objet (diagramme de classe)
4. Générer le code de la base de données (SGBD)
5. Générer le code du model de données (application cliente)

# Model physique de données

Model utiliser pour modéliser les tables d’une base de données.

## Templates

Ce tableau résume les Templates utilisables par le model :

|  |  |  |  |
| --- | --- | --- | --- |
| Langage | Version | Framework | Fichier Template |
| Visionner – Basic | 3 | - | Tpl\_Visionner.xdb |
| SQL Sever | Std 2012 | - | Tpl\_SQLServer2012.xdb |
| Postgres SQL | 9 | - | Tpl\_pgsql9.xdb |

# Model orienté objet

Model utiliser pour modéliser les classes d’une application.

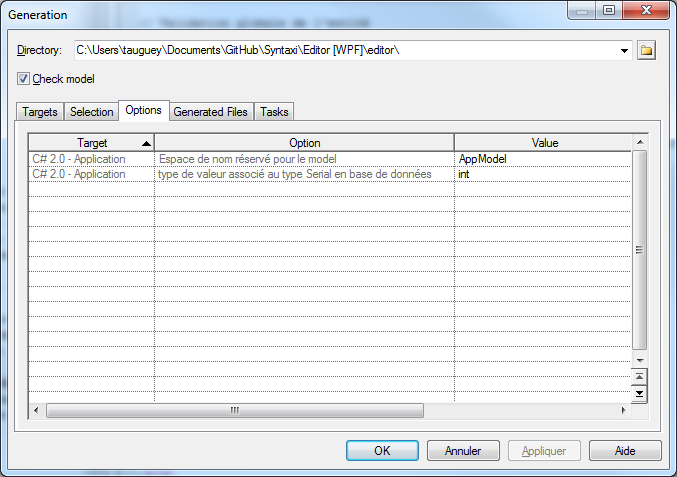
## Templates

Ce tableau résume les Templates utilisables par le model :

|  |  |  |  |
| --- | --- | --- | --- |
| Langage | Version | Framework | Fichier Template |
| Visionner – Basic | 3 | - | Tpl\_Visionner.xol |
| C# | 5 | - | Tpl\_CSharpApp.xol |
| C# | 5 | .Net | Tpl\_CSharpApp.xol |
| C++ | VC2013 | - | Tpl\_CPlusPlus.xol |
| C++ | VC2013 | wxWidget | Tpl\_CPlusPlus.xol |

## Options de génération

Les options de générations sont spécifiques au langage cible, elles permettent d’ajuster la génération du code. Chaque option est définit plus loin dans son model.

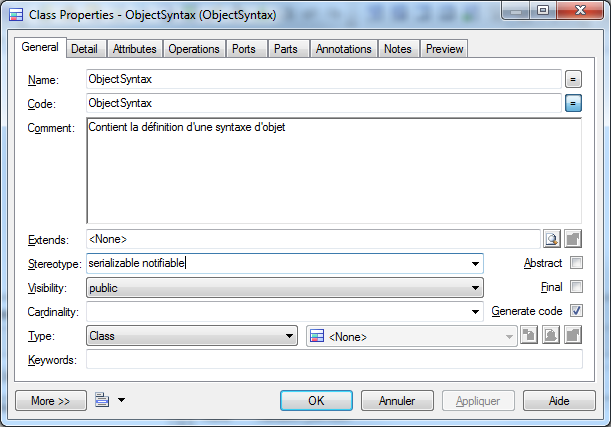


***Options de génération dans le dialogue « Générer »***

## Stéréotypes

Les stéréotypes sont utilisés dans les objets pour définir l’usage final dans le code. Par exemple, si les données d’une classes sont persistantes, sérialisables, validables, etc…

Les stéréotypes sont définit dans les propriétés de classe :



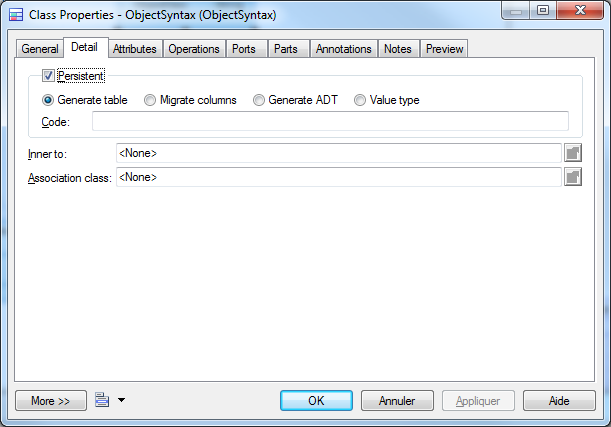
***Propriété d’une classe contenant les stéréotypes « serializable » et « validable »***

## Persistance

Il est possible de spécifier explicitement le nom des tables et des champs en correspondance avec la base de données. Par défaut le nom des tables et des champs est hérité des classes.

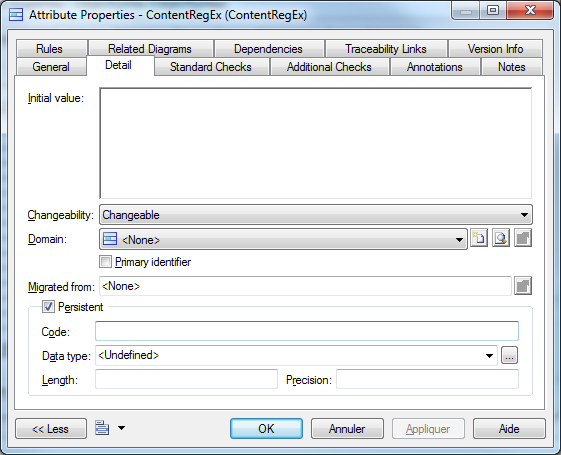
Spécifier le nom d’une table :

Rendez-vous dans le dialogue de propriétés de classe sous l’onglet « **Détail** » et éditer le champ « **Code** » sous le groupe « **Persistent** ».



Spécifier le nom d’un champ :

Rendez-vous dans le dialogue de propriétés de l’attribut sous l’onglet « **Détail** » et éditer le champ « **Code** » sous le groupe « **Persistant** ».



Classe d’association

Dans les types d’associations multiples (N-N) il est nécessaire de représenter la table de correspondance en base de données par une classe d’association.

