# DFM-updater

## Objectif

automatiser les mises à jour répétitives sur les Forms ou le code d’une application. C++ Builder Exemple :

* Ajouter/modifier/supprimer des propriétés d’un contrôle
* Renommer des fonctions, variables
* Modifier le nombre ou le type des paramètres d’une fonction
* Ajouter de nouveaux évènements d’évènements
* Ajouter/modifier/supprimer du code C++
* Etc.

## Compilation et exécution

Le code JAVA est dans <https://srv-dev/svn/Génésis/trunk/Outils/DFM-updater/DFM-updater>

L’application :

* Peut être compilée en ligne de commande avec le script package.bat
* Peut être lancée en ligne de commande avec le script run.bat. Ex :

|  |
| --- |
| Run.bat c:\Borland\RH\_TRUNK\\*.dfm |

## Fonctionnement

La classe RhUiModernizer est le point d’entrée.

La méthode RhUiModernizer.run() énumère les fichiers passés en paramètre et traite chacun d’eux avec la méthode ProcessDfm().

ProcessDfm() utilise la classe DfmReaderWriter pour parser le fichier DFM et créer un arbre des contrôles et de leurs propriétés, modélisé par la classe DfmObject.

* Le format des DFMs dans Builder 4 étant binaire et propriétaire, DfmReaderWriter utilise l’outil convert.exe de Borland pour convertir du binaire vers du texte, et vice versa.
* L’emplacement de convert.exe est défini dans DfmReaderWriter.convert().

Le code C++ est parsé (sommairement) par la classe CppReaderWriter.

* Le code C++ est reformaté à l’aide de l’outil Astyle.exe, invoqué par CppReaderWrite.reformat().

La métode RhUiModernizer.updateDfmObjects() se charge de modifier le contenu des DFMs. Elle s’appuie sur une liste de règles de conversion, chaque règle dérivant de la classe AConversionRule.

Pour effectuer un traitement, une règle existante est réutilisée (ex : ChangePropertyValue permet de modifier la valeur d’une propriété d’un objet du DFM), ou une nouvelle règle dérivant de AConversionRule est créée.

Les méthodes abstraites de AConversionRule à implémenter :

* isApplicable() renvoie vrai si la règle doit être appliqué à l’un ou l’autre des objets passés en paramètre.
* doApply() applique les modifications aux objets passés en paramètre.

La métode RhUiModernizer.updateCppCode() se charge de modifier les fichiers cpp et h. Elle s’appuie également sur une liste de règles de conversion dérivant de AConversionRule.

## Exemple : Modifier la valeur d’une propriété

On souhaite modifier la valeur de la propriété BorderColor de tous les objets TColoredGroupBox.

## Les règles existantes