**How to edit source files of an entire Visual Studio solution using DXCore**

July 4th, 2011

Usually, simple edits of text files are accomplished using the *TextDocument* object, which represents an open source file inside the IDE. The text document object is easy to access through the **Documents** [DXCore service](http://www.skorkin.com/2010/09/dxcore-services-list/" \o "DXCore overall services list" \t "_blank). It has lots of useful methods for editing a text like *InsertText*, *DeleteText* and *SetText*, which take a [source code coordinates](http://www.skorkin.com/2011/05/dxcore-source-code-and-source-tree-elements-coordinate-inside-source-files/) and a new text for replacement as a parameters. However, to use a text document object, it is required for the file to be opened inside the Visual Studio environment. If a file is closed, there’s no *TextDocument* object assigned to the file and you simply can’t use its methods. In case you are going to edit closed and/or multiple files, there’s a better way – the **FileChange** object (in the *DevExpress.CodeRush.Core.Replacement* namespace).

As the name says, the **FileChange** object represents a single change to a file. A file can be closed inside IDE and not even included into a project, so you can change any text file in your system. The object takes a few parameters:

* a path to a file
* a source range or a source point
* a text for insertion or replacement

If you pass a source point – the specified text will be inserted at the specified position. But if you pass a source range – the text inside a file will be replaced with the new one. The *FileChange* class has the corresponding properties:

|  |  |
| --- | --- |
| **Property** | **Description** |
| Data | Data, associated with a change. Used internally. |
| Order | The order of this change in a particular file. The value is set automatically. However, if several changes are made at the same position, you can set this property to manage the order the changes are applied. |
| Path | A full path to a file. |
| Range | A source range of the text inside a file for replacement. If a source range has two equal source points, the text will be inserted without deletion. |
| Text | A new text to insert or replace inside a file. |

The file change object should be added into a special collection to be applied – the **FileChangeCollection**. Then, you can apply a collection of file changes by using the [File DXCore service](http://www.skorkin.com/2011/02/dxcore-services-file/):

CodeRush.File.ApplyChanges(collection);

Note, if you have only a single file change, you can call the *CodeRush.File.ChangeFile* method instead of creating a new *FileChange* object. The *ChangeFile* method inserts, deletes or replaces the text inside the specified source file.

**Usage sample**

As an example, illustrating how to edit source files, let’s create a [new DXCore plug-in](http://www.skorkin.com/2010/08/how-to-create-a-new-dxcore-plug-in/), and add the [Action DXCore control](http://www.skorkin.com/2010/09/dxcore-components-action/), that will modify all files of the current project by inserting a file header at the top (as a comment), if it doesn’t exist.

When the action is executed, we get the current project instance and enumerate all of its files. We check the name of every file, so it has a valid extension and it is not a designer file, nor should it already contain a header. If the file is OK, then we create a new *FileChange* object, specify the path to a file, a new insertion point (1,1), a file header text) and add it to a collection of file changes. When all files are enumerated, we apply the collection, so all files of the project now have a file header. Here’s the code of the plug-in:

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53 | public partial class PlugIn1 : StandardPlugIn  {    private const string HEADER =  @"//-----------------------------------------------------------------------  // <copyright file=""{0}"" company=""My Company"">  //     Copyright (c) My Company, Inc. All rights reserved.  // </copyright>  //-----------------------------------------------------------------------  ";      private void action1\_Execute(ExecuteEventArgs ea)    {      ProjectElement activeProject = CodeRush.Source.ActiveProject;      if (activeProject == null)        return;        FileChangeCollection fileChangeCollection = new FileChangeCollection();      foreach (SourceFile sourceFile in activeProject.AllFiles)      {        if (!IsValidFile(sourceFile))          continue;          if (ContainsFileHeader(sourceFile))          continue;          string filePathAndName = sourceFile.Name;        SourcePoint insertionPoint = new SourcePoint(1, 1);        string newHeader = String.Format(HEADER, Path.GetFileName(filePathAndName));        FileChange fileChange = new FileChange(filePathAndName, insertionPoint, newHeader);        fileChangeCollection.Add(fileChange);      }      CodeRush.File.ApplyChanges(fileChangeCollection);    }      private bool IsValidFile(SourceFile sourceFile)    {      const string EXT = ".CS";      const string DESIGNER = ".DESIGNER";        string fileNameWithoutExtension = Path.GetFileNameWithoutExtension(sourceFile.Name);      if (Path.GetExtension(fileNameWithoutExtension).ToUpper() == DESIGNER)        return false;        string fileExtension = Path.GetExtension(sourceFile.Name);      return fileExtension.ToUpper() == EXT;    }      private bool ContainsFileHeader(SourceFile sourceFile)    {      LanguageElement firstChild = sourceFile.FirstChild;      return firstChild is Comment;    }  } |

The source of a plug-in is [attached](http://www.skorkin.com/files/2011/07/TestFileChanges.zip) (7,464 bytes, C#, Visual Studio 2010).