

Contents

- [1 Overview](#)
- [2 Setup](#)
- [3 Creating a diff file \(command-line approach\)](#)
- [4 Creating a diff file \(graphical approach\)](#)
- [5 Applying a patch \(command-line approach\)](#)
- [6 Applying a patch \(graphical approach\)](#)
- [7 See also](#)

Overview

A diff is either a file that describes the differences between two files, or a utility that produces the said file. Here, a diff refers to the file. It is often called a patch since it can be applied with the patch program.

Diffs are often used in software development to describe the changes made to the current version of the software. The diff file (the changes) can also be used to update the former version of the software by using it as input to the patch program. This is referred to as applying the patch.

These instructions describe how to create diff files and how they can be used with the patch program. Both command-line and graphical approaches are presented.

Setup

To create diff files (patches) with the Windows command line, you need to obtain the diff program for Windows. Get it from here: <http://gnuwin32.sourceforge.net/packages/diffutils.htm> (download the complete package, except sources). Installing the program should be straightforward, but there might occur a slight hiccup concerning the language that the program uses. By default, the diff program automatically detects the native language set to your computer. If it is not English and you don't want to use diff in your native language, disable the native language support by setting the LANG and LANGUAGE environment variables to "en".

To apply patches with the Windows command line, you need to obtain the patch program for Windows. Download it from <http://gnuwin32.sourceforge.net/packages/patch.htm>. There should be nothing worth mentioning in the installation.

After you have installed both diff and patch programs, make sure that they are added to your path (Control Panel > System > Advanced > Environment Variables > Path). You can check this by opening the command line and typing "diff -v". If the output is something like the following, diff has been installed correctly and can be found from the path:

```
diff (GNU diffutils) 2.8.7
Written by Paul Eggert, Mike Haertel, David Hayes,
Richard Stallman, and Len Tower.
```

Using_Diffs

Copyright (C) 2004 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

To create patches and apply them graphically, download the TortoiseSVN revision control software from <http://tortoisesvn.tigris.org/> and install it. You can also use the [Subclipse](#) plug-in for Carbide.c++ to create and apply patches with Carbide.c++. For more information, refer to [Using Subclipse with Carbide.c++](#).

Creating a diff file (command-line approach)

Let's assume there are two text files, `oldFile.txt` and `newFile.txt`. The file contents are as follows:

`oldFile.txt:`

```
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
```

`newFile.txt:`

```
Line 1
Line 2
Line 4
Line 5
Line 6
Line x
```

If we want to see the differences between those two files, we would type in the command line:

```
diff -u oldFile.txt newFile.txt # -u tells that the output is in the unified diff format
```

The output looks like this:

```
--- oldFile.txt 2008-10-07 16:25:02.388856900 +0300
+++ newFile.txt 2008-10-07 16:24:55.841730500 +0300
@@ -1,6 +1,6 @@
   Line 1
   Line 2
-  Line 3
   Line 4
   Line 5
   Line 6
+  Line x
```

The `-u` option above tells diff program to output in the unified diff format. This format is the preferred one, since it produces somewhat smaller output and works better with the patch program than the regular diff format.

Now, if we want to create a diff file (a patch) that describes the differences between those two files, we would type in the command line:

Using_Diffs

```
diff -u oldFile.txt newFile.txt > diffFile.diff
```

From now on, the file `diffFile.diff` is called a patch.

If we had a complete directory structure with changes, we would use the `-r` option to tell `diff` that it should recursively compare also the subdirectories:

```
diff -ur oldDir newDir > diffFile.diff
```

Creating a diff file (graphical approach)

Like in the command-line approach, to create a diff file with TortoiseSVN, you'll need to have "an old file" and "a new file". With TortoiseSVN, the new file is the working copy and the old file is the working base. Let's assume you have checked out a revision from SVN and made some changes to it. To create a diff file that describes the differences between the working copy and the working base, follow these instructions:

1. Open the Windows Explorer.
2. Right-click on the working copy folder (it should be colored red if there really are changes).
3. Select `TortoiseSVN > Create patch...` The following dialog summarizes the files that will be included in the patch file. Make sure that they are correct by double-clicking on them and checking the output of TortoiseMerge.
4. Press OK, browse to the destination folder, and type in the name of the patch file (e.g. "`diffFile.diff`"). After this, TortoiseSVN shows the diff file contents in color-coded, unified diff format.

From now on, the file `diffFile.diff` is called a patch.

Applying a patch (command-line approach)

Let's assume we have a patch file called `diffFile.diff`. It describes the changes in `newFile.txt` in relation to `oldFile.txt` (naturally, the changes could be much more significant, like complete directory structures, but let's keep this example simple). To update (patch) the `oldFile.txt` so that it is equal with the `newFile.txt`, we use the diff file as input to the patch program, like this:

```
patch < diffFile.diff
```

The command above patches `oldFile.txt` with changes described in `diffFile.diff`.

Applying a patch (graphical approach)

Let's assume we have a patch file called `diffFile.diff`. It was produced with TortoiseSVN, and it describes the differences between two files, `newFile.txt` and `oldFile.txt` (naturally, the changes could be much more significant, like complete software revisions, but let's keep this example simple). To update (patch) the `oldFile.txt` so that it is equal with the `newFile.txt`, follow these instructions:

1. Open the Windows Explorer.
2. Right-click on the `diffFile.diff`.

Using_Diffs

3. Select TortoiseSVN > Apply patch. . . . If the patch file was in a directory from where the `oldFile.txt` cannot be found, TortoiseSVN prompts you to browse to the folder from where it can be found. Otherwise, or after you browse for the correct folder, the File patches dialog opens. Make sure that patching will execute correctly by double-clicking on `oldFile.txt` and skimming through the visual output of TortoiseMerge.
4. When you are ready to continue, right-click on the File patches dialog and select Patch all. TortoiseSVN patches the `oldFile.txt` for you.

See also

- [Wikipedia article about the diff program](#)
- [Wikipedia article about the patch program](#)
- <http://gnuwin32.sourceforge.net/>: Various Unix utilities ported to Windows
- <http://tortoisesvn.tigris.org/>: TortoiseSVN, a revision control software