Java Application: HexEdit

Written by: Keith Fenske, http://kwfenske.github.io/
First version: Monday, 27 October 2008
Document revised: Saturday, 13 February 2010

Description

HexEdit is a Java 1.4 graphical (GUI) application to edit a file as a stream of hexadecimal digits, where each 8-bit byte is represented by two 4-bit "nibbles" (the hex digits). No meaning is attached to the digits, and hence to the contents of the file, so this editing is very raw. You can insert, delete, or replace digits or bytes, and you may view an approximate character equivalent of the digits in plain text (7-bit ASCII). The primary purpose of a hex editor is to patch or correct specific locations within a file without affecting the rest of the file, something that most word processors can't do. A secondary purpose is to view the exact content of files. (To generate a hex dump and save this dump to a file, see the DumpFile Java application.)

The application window has two portions. The top portion has standard Java buttons and options. The bottom portion has a hexadecimal dump with three regions: file offsets on the left, bytes in hexadecimal in the middle, and ASCII text on the right. The bottom looks like an old video terminal (quite deliberately), with a color scheme to match. The Java buttons and options are straightforward after some experimentation. Don't worry: no files are changed unless you click on the "Save File" button. In fact, files aren't saved by default, so if you click on the Exit button without saving, any changes will be lost. This may differ from what you expect with word processors or other types of editors.

File offsets are 8-digit hexadecimal numbers to show you where the start of a row (dump line) is located from the beginning of the file. Bytes are dumped (displayed) as two hexadecimal digits. The first digit is the high-order digit in the byte; the second digit is the low-order digit. All file bytes contain two digits, although during editing, you may see an odd number of nibbles in the file. (A zero digit is appended if necessary when writing a file.) Bytes are not grouped in any way, because different computers and programs have different ways of deciding which byte is the low-order byte in a word. The text region on the right is only an approximation; most binary data is not text. Bytes that are printable 7-bit ASCII characters are shown as text; anything else has a replacement character ("."). This does not mean that the program is limited to ASCII text. You can copy and paste any text in the local system's default encoding, even if that encoding uses 8-bit bytes or multiple bytes.

Other than the obvious scroll bar to move through the file, and the fun you can have resizing text by changing the number of bytes per line or the window size, your interaction is via the mouse and keyboard. Most features are provided by both methods.

The mouse can be used to position, select, scroll, or to open a context menu. To position the cursor (the location where data is inserted or deleted), click once with your primary mouse button, which is usually called a "left" click. To select all data from a previous location to a new location, press and hold the "Shift" key on your keyboard and then left click on the new location. (Release the Shift key when done.) You may also select by clicking the mouse button, holding the button down, moving the mouse to a new location, and releasing the mouse button. Selections must be entirely within the dump region or the text region. If your mouse has a scroll wheel, you can rotate that wheel to move up or down in the file. The final mouse feature is a context menu that will "pop up" if you click the right mouse button. Not all computers have a right button, so any button other than the primary button will be accepted, and if your computer has only one mouse button, then hold down the "Control" key while clicking the primary mouse button.

Regular text typed on the keyboard will be inserted into the file, or will replace a selection if a selection has been made. For the dump region, only the decimal digits "0" to "9" are accepted, along with the uppercase hexadecimal digits "A" to "F" and the lowercase hexadecimal digits "a" to "f" (and a few punctuation characters are ignored). For the text region, all printable characters are accepted, anything that Java can convert with the local system's default encoding.

Keyboard Shortcuts

There are numerous keyboard "shortcuts" or control codes. In the following table, "Control-X" means to press and hold the "Control" key, then press and release the "X" key, then release the Control key. Most of these key combinations are standard for editing applications, except that the unique nature of a hex dump changes the meaning of some keys.

Alt	Invokes regular menus. Officially, "Alt" stands for "Alternate" but nobody says it that way.
Alt-M	Shows the "Edit Menu" (copy, paste, find, replace, select, etc). This is the same as the pop-up menu for right mouse clicks.
Alt-O	Shows the "Open File" dialog box.
Alt-S	Shows the "Save File" dialog box.
Alt-X	Exits (closes) the program, no questions asked. Closing the main window has the same effect.
arrow keys	The left arrow moves the cursor one position to the left (one nibble for dumps, one byte for text). The "Shift" key combined with the left arrow key (press and hold Shift key, press and release left arrow key, release Shift key) selects one position to the left, or expands the current selection by one position. The right arrow goes one position to the right, the up arrow goes one line/row up, and the down arrow goes one line/row down.

Backspace	If there is a selection, then delete the selection. Otherwise for hex dumps, delete the nibble (digit) before the cursor, and for ASCII text, delete the character (byte) before the cursor. This key may be "Bksp" on some keyboards, or just a left arrow (not to be confused with the real left arrow key).
Control	Invokes keyboard shortcuts. May be "Ctrl" on some keyboards.
Control-A	Selects all data.
Control-C	Copies selected data to the clipboard. If the dump region is active, then text characters representing the hexadecimal digits are copied. If the text region is active, the bytes are converted to text characters using the local system's default encoding.
Control-F	Opens the "find" or "replace" dialog box.
Control-G	Goes to a specific file byte offset in hexadecimal (dialog box).
Control-N	Finds the next occurrence of the search string.
Control-R	Replaces the current selection (if any) with the replacement string.
Control-V	Pastes the clipboard as data. Similar rules as Control-C.
Control-X	Similar to Control-C except the selected data is deleted after it is copied to the clipboard. Usually referred to as a "cut" operation.
Control-Z	Reserved for an "undo" feature that hasn't been implemented yet.
Delete	Same as Backspace, except that the deletion is forward: the nibble or byte following the cursor. May be "Del" on some keyboards.
End	Goes to the end of the current line/row, which (surprise!) turns out to be the beginning of the next line/row. Control-End goes to the end of the file. May be combined with the Shift key to select.
Escape	Cancels any current selection. May be "Esc" on some keyboards.
F6	Makes the cursor active in the text region, while Shift-F6 makes the dump region active.
Home	Goes to the beginning of the current line/row, or if already at the beginning, to the previous line/row. Control-Home goes to the start of the file. May be combined with the Shift key to select.
Insert	Toggles between "insert" and "overwrite" mode. Insert mode adds new nibbles/bytes at the cursor location (vertical line). Overwrite mode replaces nibbles/bytes at the cursor (box outline). This key may be "Ins" on some keyboards.
Page Down	Goes down as many lines/rows as there are in the display, less one. May be combined with the Shift key to select.

Page Up	Opposite of Page Down: goes up. May be combined with Shift.
Tab	Used by Java to traverse components (jump from button to button, etc).

Apache License or GNU General Public License

HexEdit is free software and has been released under the terms and conditions of the Apache License (version 2.0 or later) and/or the GNU General Public License (GPL, version 2 or later). This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY, without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the license(s) for more details. You should have received a copy of the licenses along with this program. If not, see the http://www.apache.org/licenses/ and http://www.gnu.org/licenses/ web pages.

Installation

You must have the Java run-time environment (JRE) installed on your computer. HexEdit was developed with Java 1.4 and should run on later versions. It may also run on earlier versions, but this has not been tested. You can download the JRE from Oracle (formerly Sun Microsystems):

JRE for end users: http://www.java.com/getjava/
SDK for programmers: http://www.oracle.com/technetwork/java/
IDE for programmers: http://www.netbeans.org/

Once Java is installed, you need to put the program files for HexEdit into a folder (directory) on your hard drive. The name of the folder and the location are your choice, except it is easier if the name does not include spaces. Assume that files will go into a "C:\Java" folder. Then create the folder and unpack the Java *.class files into this folder (if you received the program as a ZIP file). The files look something like this:

ApacheLicense20.txt (12 KB, legal notice)
GnuPublicLicense3.txt (35 KB, legal notice)
HexEdit2.class (29 KB, executable program)
HexEdit2.doc (43 KB, this documentation in Microsoft Word format)
HexEdit2.gif (21 KB, sample program image)
HexEdit2.ico (87 KB, icon for Windows)
HexEdit2.jar (25 KB, archive file with same class files inside)
HexEdit2.java (176 KB, source code)
HexEdit2.manifest (1 KB, main class manifest for archive file)
HexEdit2.pdf (84 KB, this documentation in Adobe Acrobat format)
HexEdit2Data.class (3 KB, helper class for main program)
HexEdit2Text.class (12 KB)

HexEdit2User.class (2 KB)
RunJavaPrograms.pdf (60 KB, more notes about running Java)

To run the program on Windows, start a DOS command prompt, which is Start button, Programs, Accessories, Command Prompt on Windows XP/Vista/7. Change to the folder with the program files and run the program with a "java" command:

```
c:
cd \java
java  HexEdit2
```

The program name "HexEdit2" must appear exactly as shown; uppercase and lowercase letters are different in Java names. Some systems (Macintosh) will run a main "class" file by clicking on the class file name while viewing a directory in the file browser (Mac Finder). Many systems will run a "jar" file by clicking (or double clicking) on the jar file name (Windows Explorer). The command line is the only guaranteed way of running a Java program. Should you find this program to be popular, you can create a Start menu item or desktop shortcut on Windows XP/Vista/7 with a target of "java.exe HexEdit2" starting in the "C:\Java" folder.

One complication may arise when trying to run this program. Java looks for an environment variable called CLASSPATH. If it finds this variable, then that is a list of folders where it looks for *.class files. It won't look anywhere else, not even in the current directory, unless the path contains "." as one of the choices. The symptom is an error message that says:

```
Exception in thread "main" java.lang.NoClassDefFoundError: HexEdit2
```

To find out if your system has a CLASSPATH variable defined, type the following command in a DOS window:

```
set  CLASSPATH
```

To temporarily change the CLASSPATH variable to the current directory, use the following command line:

```
java  -cp  .  HexEdit2
```

To permanently change the CLASSPATH, you must find where it is being set. This is in Control Panel, System, Advanced, Environment Variables on Windows XP/Vista/7.

Removal or Uninstall

To remove this program from your computer, delete the installation files listed above. If the folder that contained the files is now empty, you may also delete the folder ... if you created the folder, of course, not the system. If you created desktop shortcuts or Start menu items, then delete those too. There are no hidden configuration or preference files, and no information is stored in the Windows system registry. You don't need an "uninstall" program.

Graphical Versus Console Application

The Java command line may contain options for the insert/overwrite typing mode, the number of input bytes per dump line, and a file name. See the "-?" option for a help summary:

```
java HexEdit2 -?
```

The command line has more options than are visible in the graphical interface. An option such as -u14 or -u16 is recommended because the default Java font is too small.

Restrictions and Limitations

There is an unusual circumstance where the mouse may point at the dump region but keyboard input is not accepted. You can force this by clicking anywhere on the dump region, then without moving the mouse, open the "Edit Menu" with the Alt-M key combination, and cancel the menu with the Esc (escape) key. The buttons on top now have keyboard focus, even though the mouse is pointing at the dump on the bottom. The dump region will regain focus as soon as you click or move the mouse.

To handle insertions and deletions, the entire data file is buffered in memory as a split array of 4-bit "nibbles" (two data nibbles per 8-bit file byte). To view a file, the Java heap size must be at least twice the size of the file. To edit a file, it must be four times. The default Java 1.4 virtual machine on Windows will allow editing of files over 10 megabytes, and you may increase the maximum heap size with the "-Xmx" option on the Java command line. This program is not recommended for files larger than 100 megabytes unless you have a fast computer and disk drive. The absolute maximum file size is one gigabyte, because nibbles in the file are counted with a signed 32-bit integer.

---

file: HexEdit2.doc 2021-10-26