The continuing story of Vim

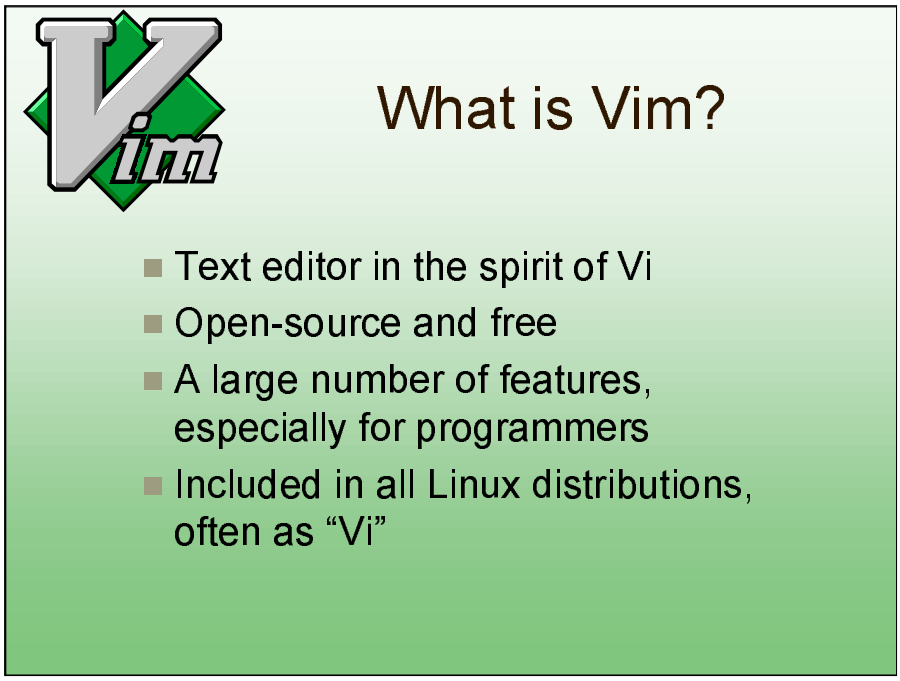
Bram Moolenaar
www.moolenaar.net

Presentation given by Bram Moolenaar at Linux2000.nl, October 10 2000.

I can do this presentation in Dutch or English. If there is someone who doesn't understand Dutch I'll use English.

I am the main author of Vim. Many others helped to make what it is now. I could never have done this without their help and support.

There will be time to ask questions at the end of the presentation.

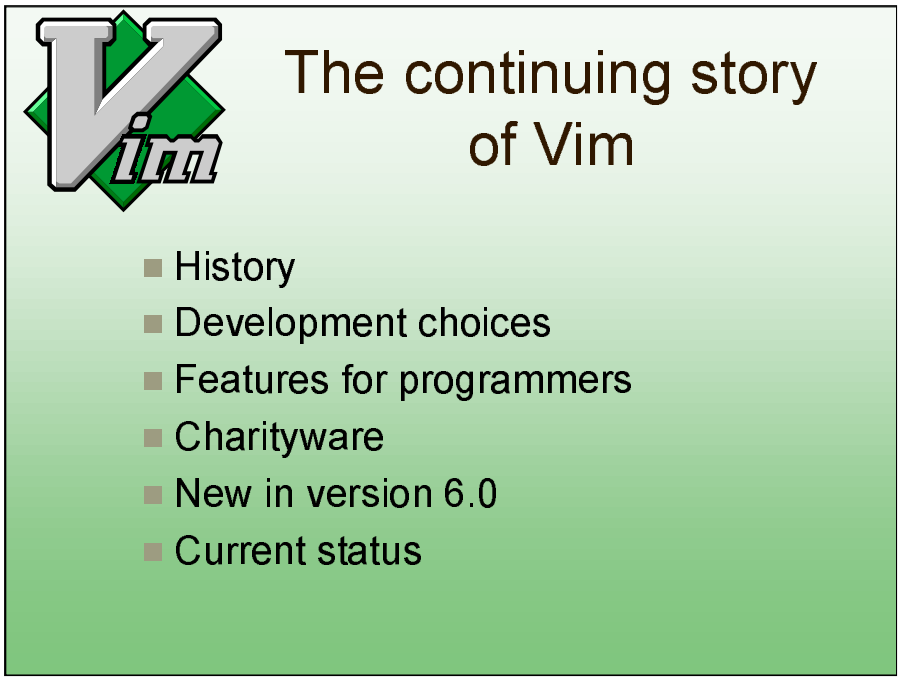


Question: Who of you have never used Vim? Who is using Vim every day?

For those of you who don’t know what Vim is. Perhaps some people typed “Vi” and didn’t know it was actually Vim they were using.

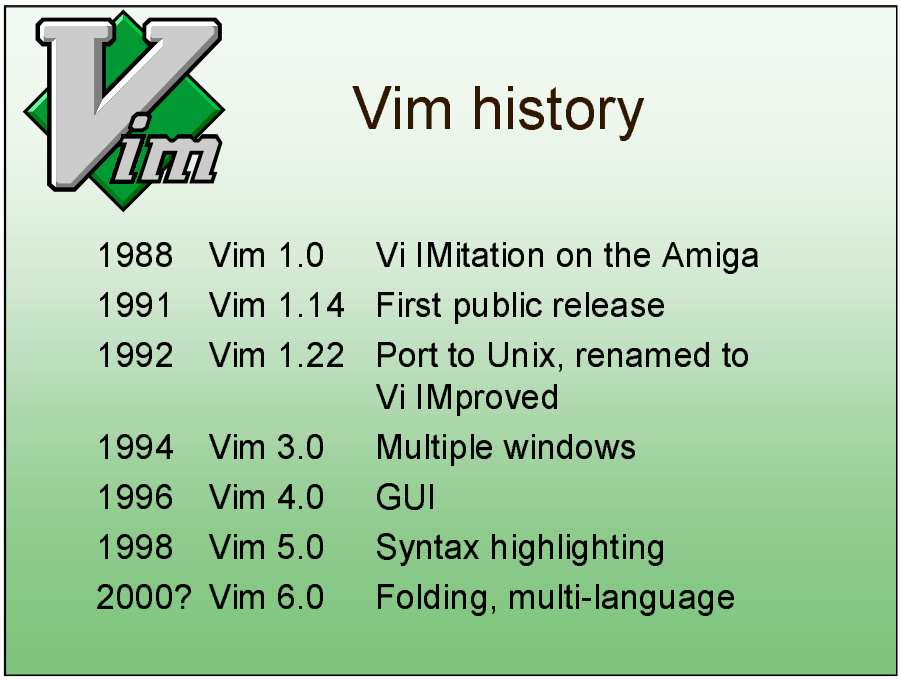
Vim is 99% Vi compatible, but has many more features. Just one feature is already a reason to switch from Vi to Vim: multi-level undo.

I don’t really know a good reason to keep using Vi instead of Vim.



In this presentation I will give an overview of the developments that Vim has been going through.

Since this is a mixed audience, I have tried to include something interesting both for non-Vim users and people who know Vim and want to know about the latest developments.



When I had bought an Amiga computer, I wanted to use the editor I learned on Unix. Since there was no good Vi for the Amiga, I started with the best that was available (a program called Stevie) and started improving it.

At first the goal was to imitate Vi. Later new functionality was added, then it was renamed from Vi IMitation to Vi IMproved.

Ports to various operating systems have been done by different people. The port to Unix was a milestone, since Vim started competing with Vi. Currently Vim runs on more than a dozen different operating systems.

Over time Vim has grown constantly. It has become a complex program. Adding functionality is taking much more effort now. But the list of features people ask to be added only keeps getting longer.

Vim 6.0 is still being developed. More about that later.

The first goal of Vim is to reproduce the behavior of Vi. And then add functionality to IMprove it.

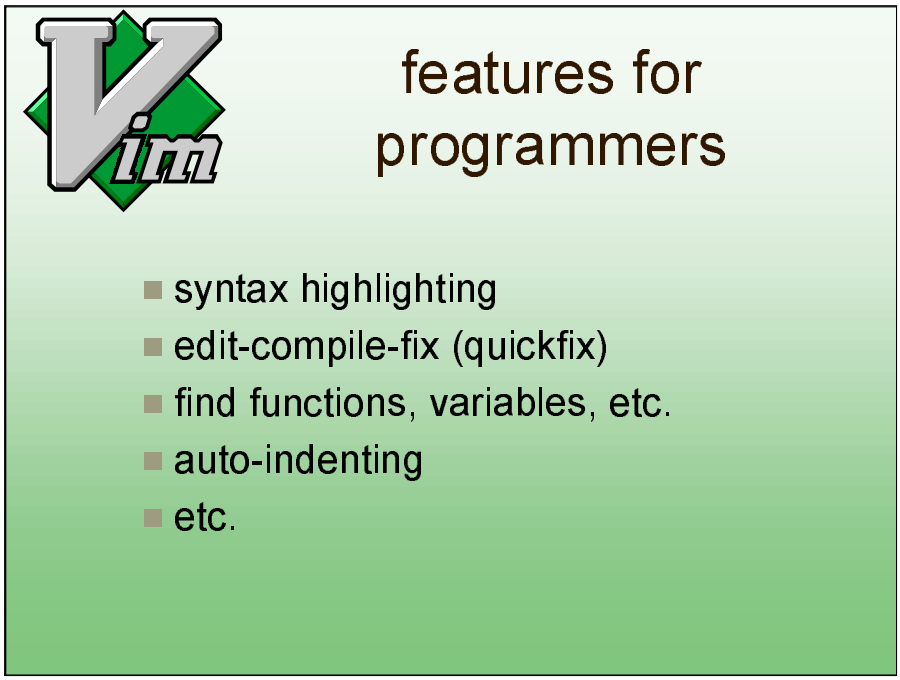
But sometimes the Vi behavior wasn't very good. Then Vim offers the option of doing it the Vi way or the IMproved Vim way.

Example: In Vi "u" toggles between undo and redo. Vim offers multi-level undo. Either Vi compatible, by using CTRL-R to repeat the undo or redo, or IMproved, using "u" for undo and CTRL-R for redo. The last is easier to use, but not Vi compatible.

Since there are many Vim users, there is always somebody that would like to have an option for every choice. But that results in a long, hard to use list of options. I try to find a balance between enough flexibility and not too many options.

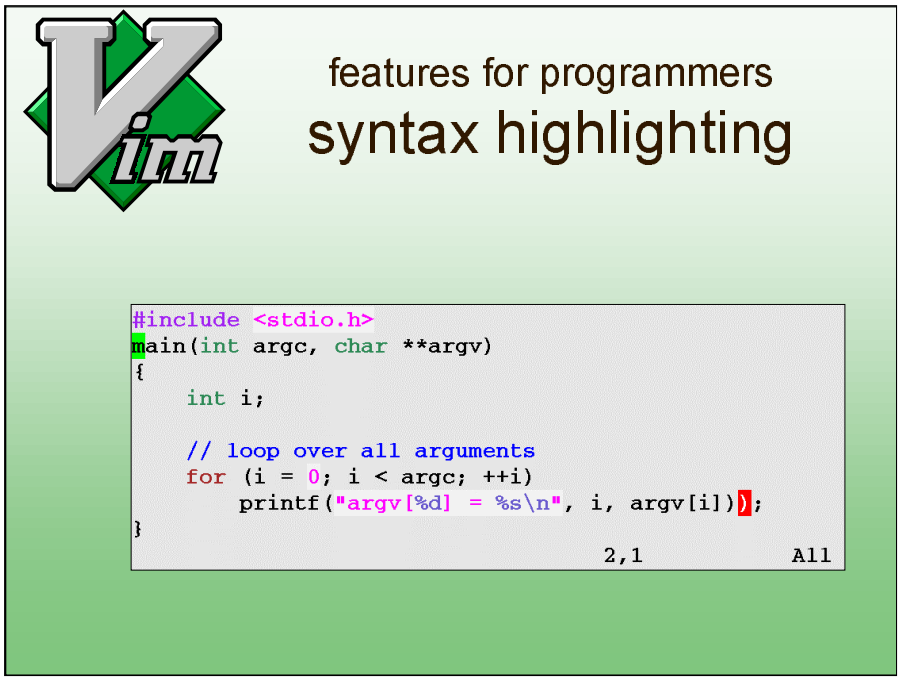
Nvi uses the "." command to repeat undo. Unfortunately this is both not Vi compatible (in Vi the "." repeats the "x", not the "u") and not easy to use. This is clearly not a Vim choice.

-- subject break --



Vim has a long list of features. To give you some impression of what it can do I have selected only a few.

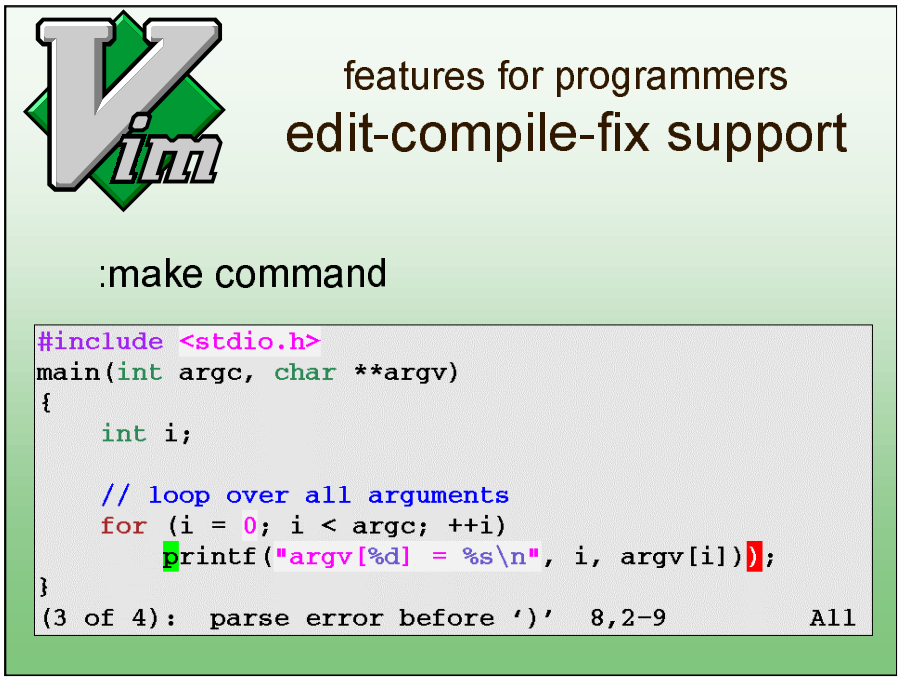
Vim is often used for programming. And I am a programmer myself. Therefore Vim has many useful features to support programmers.



Color helps to quickly find the text you're looking for. Especially to distinguish comments and commands.

Vim's syntax highlighting can also detect errors, like an unbalanced brace. This shows the error right when you are typing. Then you can quickly fix it.

Syntax highlighting is very flexible. You can just highlight a few items, like comments, or use it to check most of the syntax.

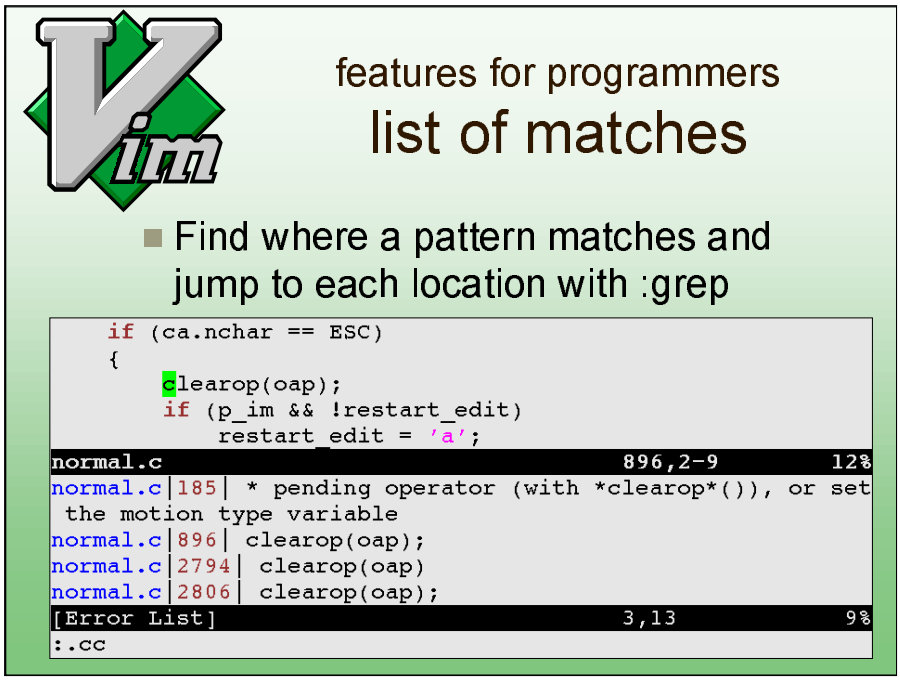


The compiler can be started from within Vim. Mostly you would use the ":make" command for this.

The error messages are parsed and Vim jumps to the right file, line (and column if it is mentioned) for the error. This speeds up fixing reported errors a lot.

The command to run the compiler and the format of error messages can be chosen freely, to make this work for your specific compiler.

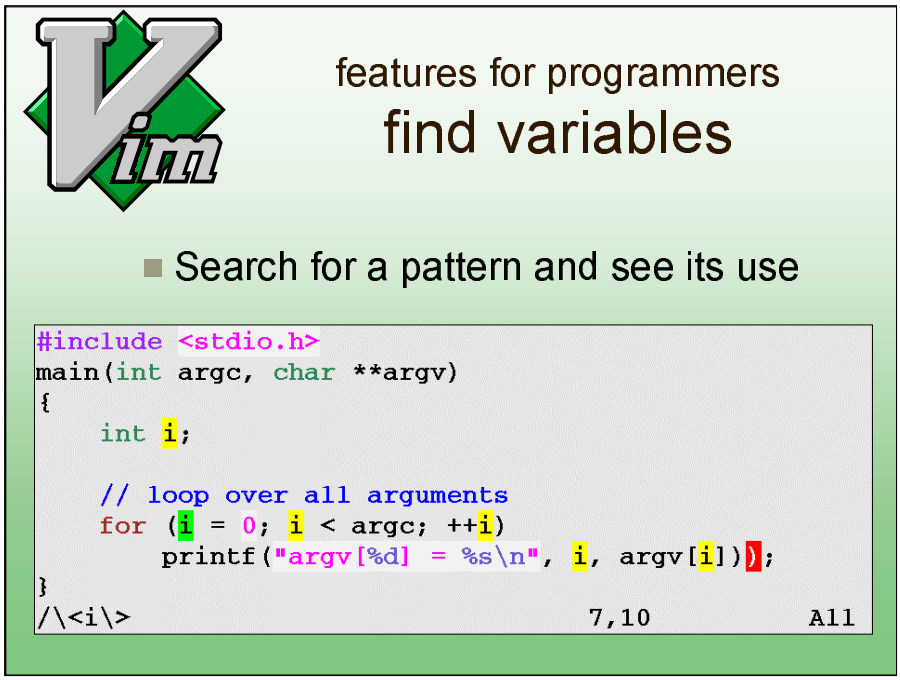
Quickfix has been present in Vim from the very early days. It has been improved several times. Recently multi-line error messages have been added. And recognizing the gcc message that it changes directory. New in Vim 6.0 is recognizing the column number from a pointer in a message.



A useful variant of ":make" is ":grep". You can use it to find all locations where a variable or function is used.

A useful application is when the arguments for a function change. Use ":grep" to find all locations where the function is used. Then jump to each location to correct the function call.

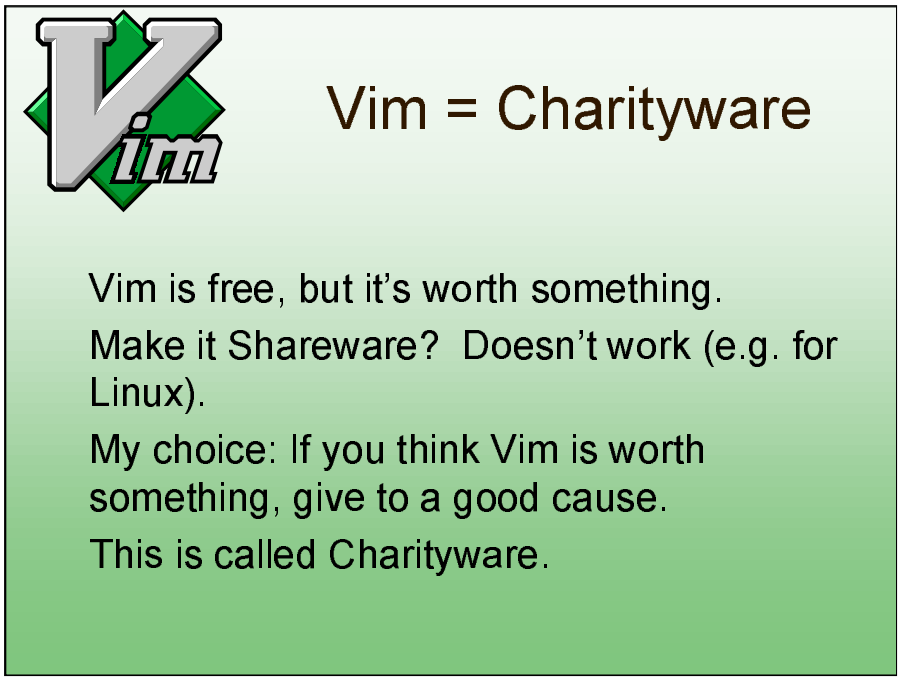
Listing the matches in a separate window is a new feature of Vim 6.0.



When searching for a pattern (here "i"), all matches are highlighted. This can be used to quickly see where a variable is used. It can be switched on and off, of course.

There are many more features for programmers, these are just a few examples.

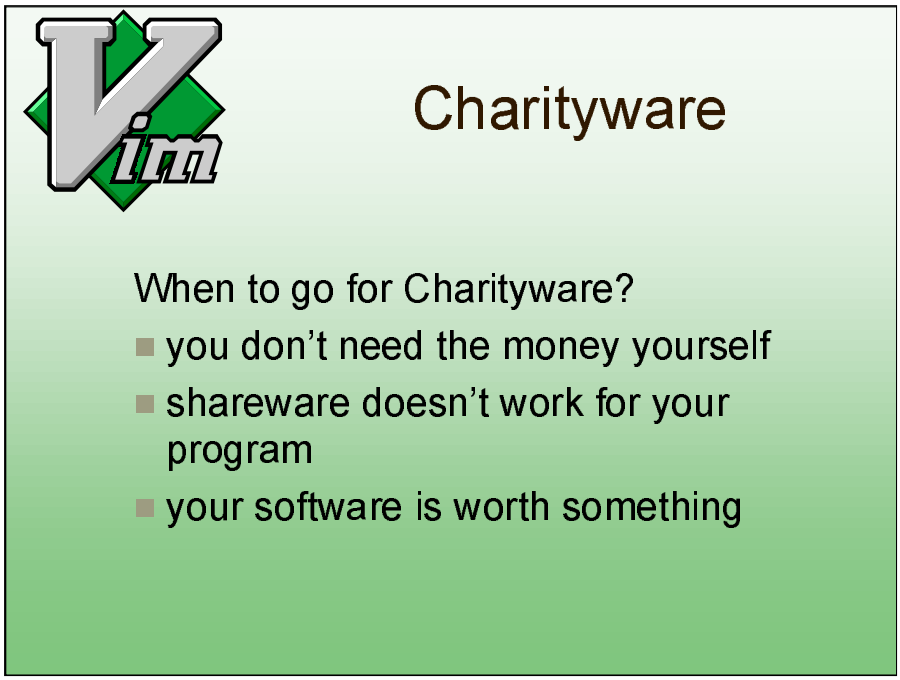
-- subject break --



Many people have expressed that Vim is better than most commercially available editors. Would it be possible to ask money for using Vim? Probably not. At least not for when it's used as Vi in Linux distributions. And I don't really need the money myself.

My solution is to ask people to donate money to a good cause. This is called Charityware. Not only does this raise money, it also brings awareness about third-world problems to many people.

Does it really work? Yes, I have received many donations and found sponsors for children. The wider Vim is spread, the more positive reactions I get.

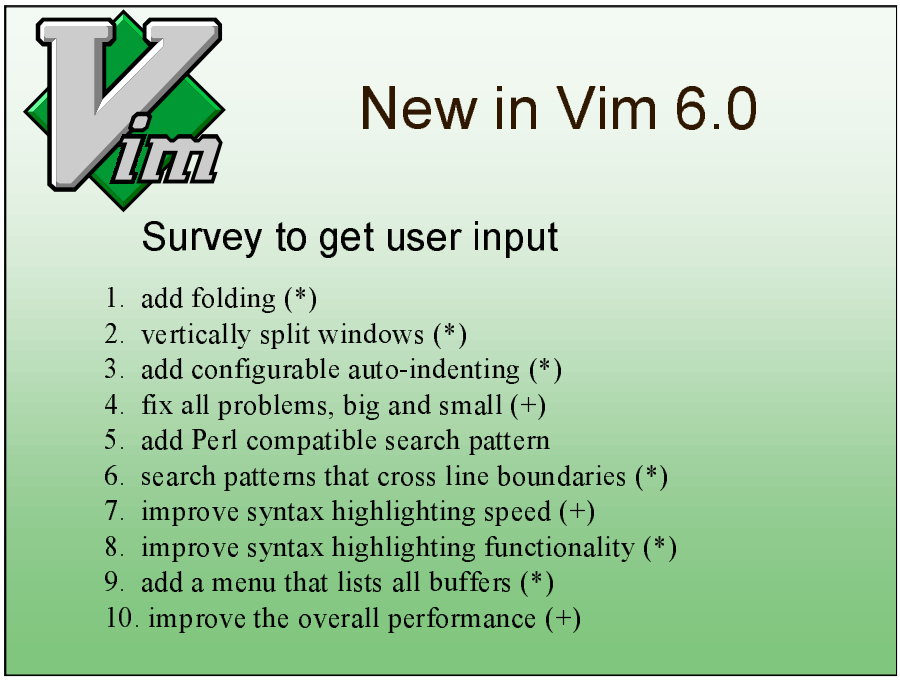


If you write open-source software yourself, you could consider making it Charityware too. You can use these three guidelines to make the decision.

If you decide to make your program Charityware, you should select a good cause that you ask people to donate to. This gives the idea a personal touch.

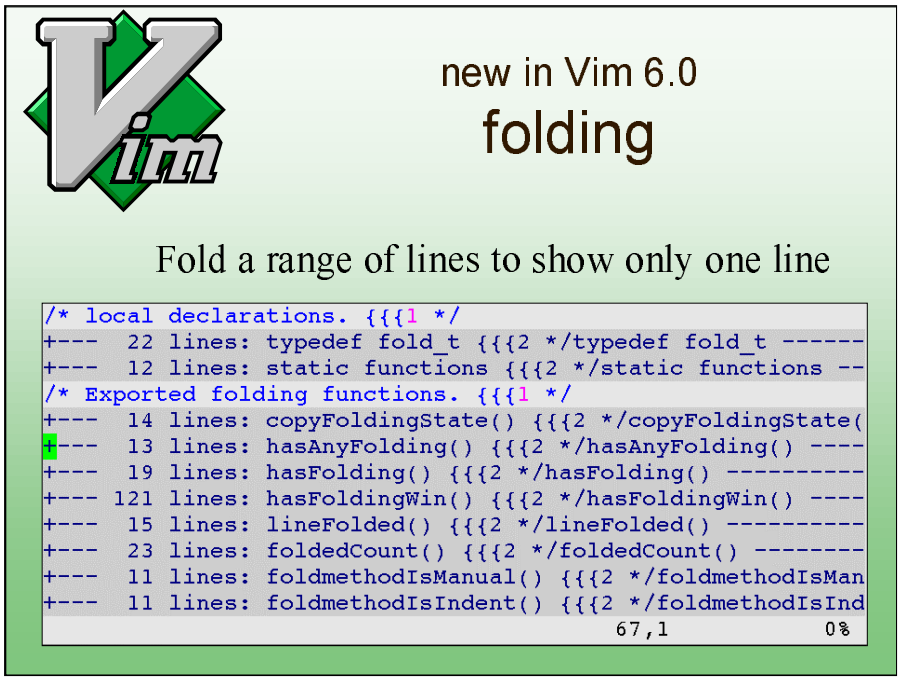
For Vim I have selected the Kibaale Children's Centre in the south of Uganda. This is in an area that has been hit hard by AIDS. Many parents die, leaving their children behind. I have worked at this center for a year. That is how I know the work done there does really help the children.

Ask me for more information about the center and how to help them.



When I started working on Vim 6.0 I had to choose which features to add. To allow all Vim users to influence this choice a survey was done. A long list of features was presented, and people could give points to each feature. This is the resulting top-ten.

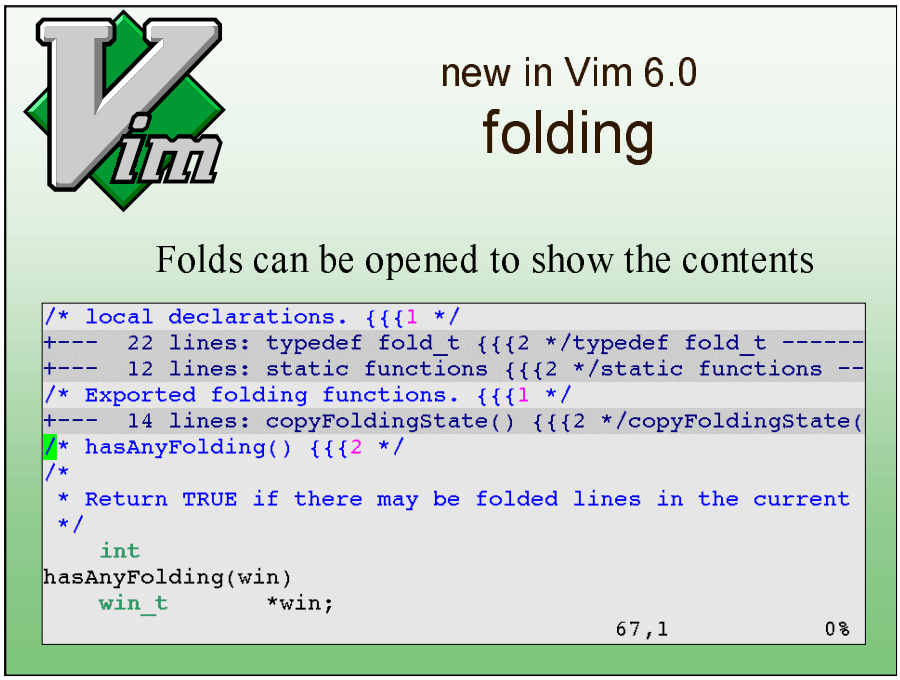
The items marked with a (+) are continuous activities. The items marked with a (*) have been implemented in Vim 6.0. Only one item is left out.



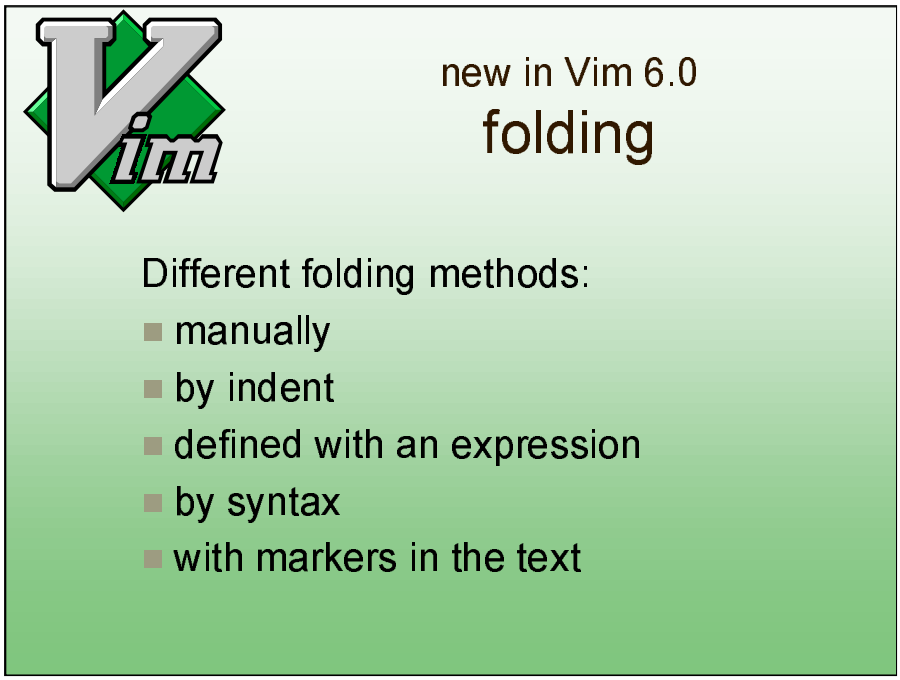
Folding is a major new feature in Vim 6.0.

A fold contains a range of lines. When the fold is closed, the lines are replaced with a single line that indicates what the fold contains.

The example shows one fold for each function. This makes it easy to locate a function. Then open the fold to see its contents.



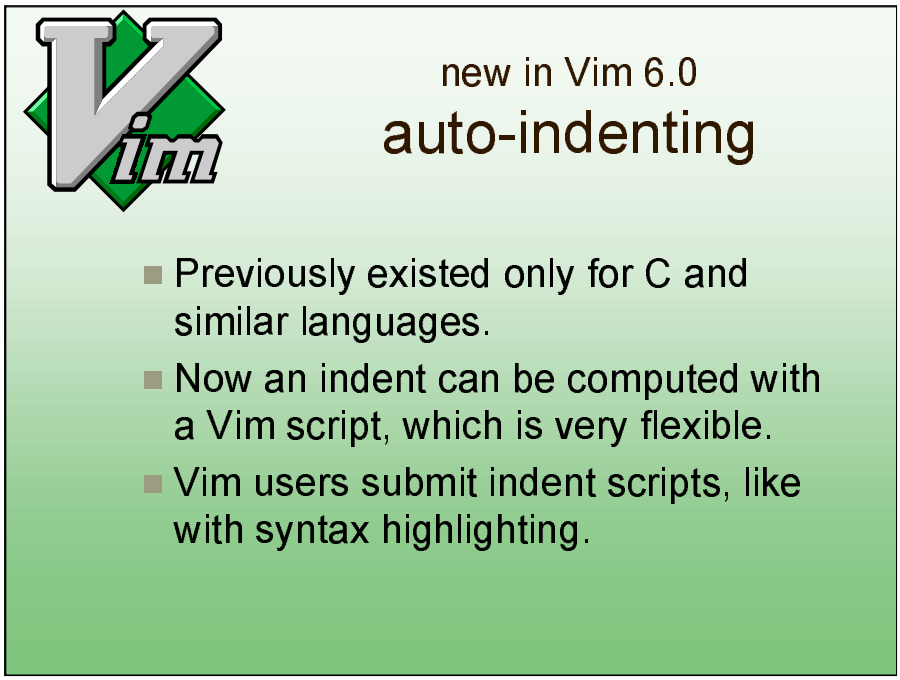
When a fold is opened you can work on the contents.



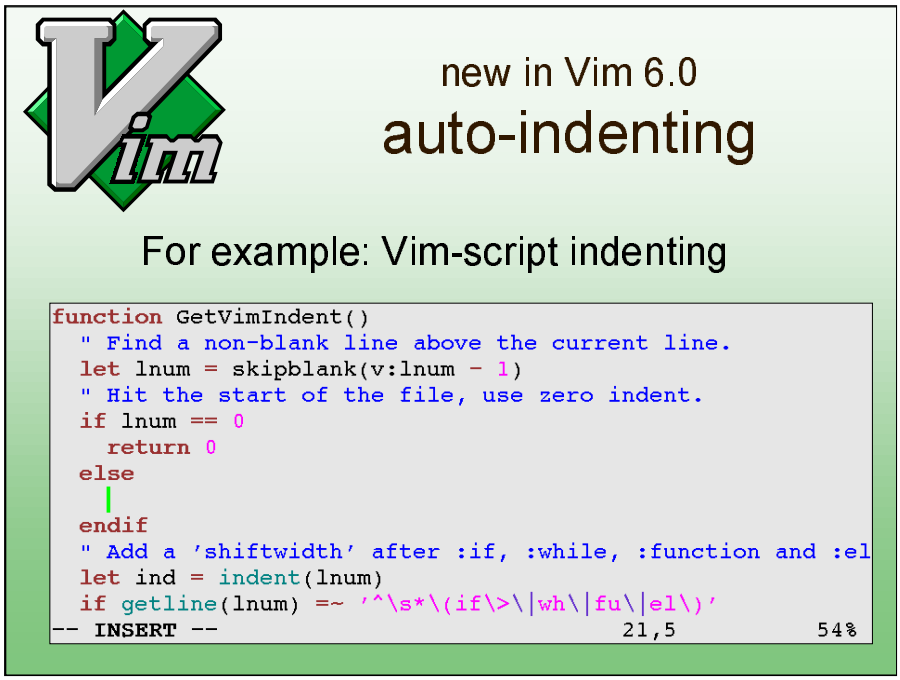
Why so many different ways to define folds? Because there is not an obvious way that suits all users.

Defining folds with markers works most accurate. It's possible to tell in which fold a comment is included. But not everybody wants to add markers in his text.

Using folding takes a little while to get used to. Once you know it, you can do your work faster, because you have a better overview.

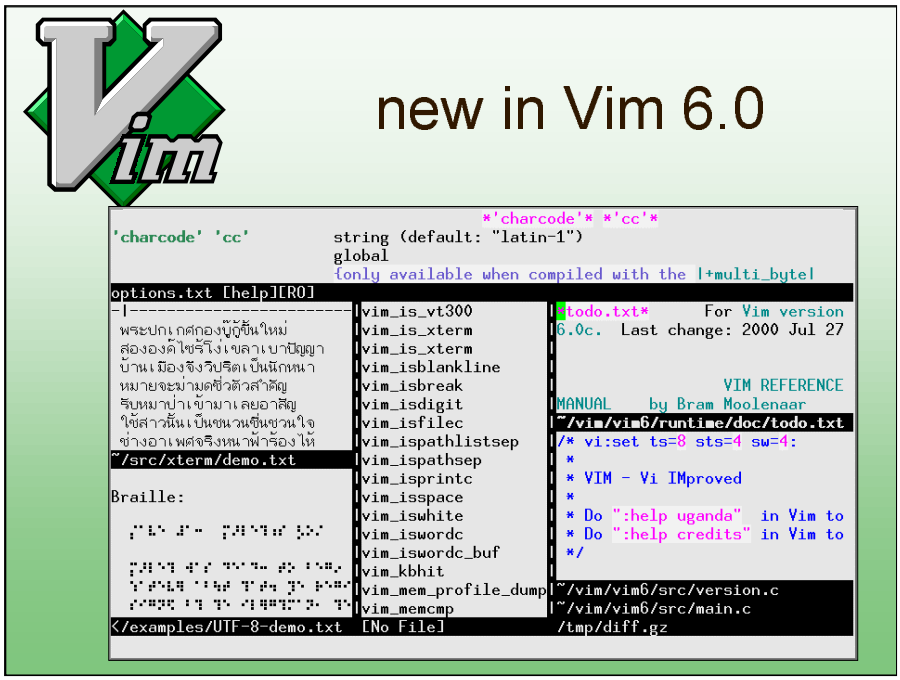


One drawback of using a Vim script for indenting is that it won't be very fast. I could not think of another solution that is faster and still flexible enough.



When using the “o” command on the “else” to open a new line, Vim automatically adds two spaces of indent.

When typing an “endif” statement, it starts with too much indent, this is corrected as soon as the statement is recognized.

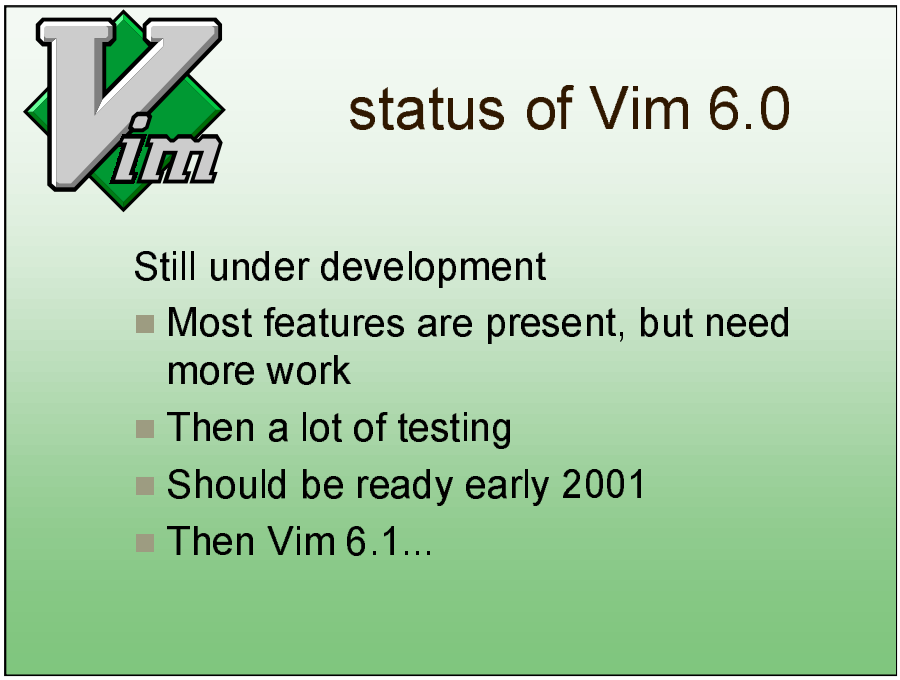


Other features that are part of Vim 6.0:

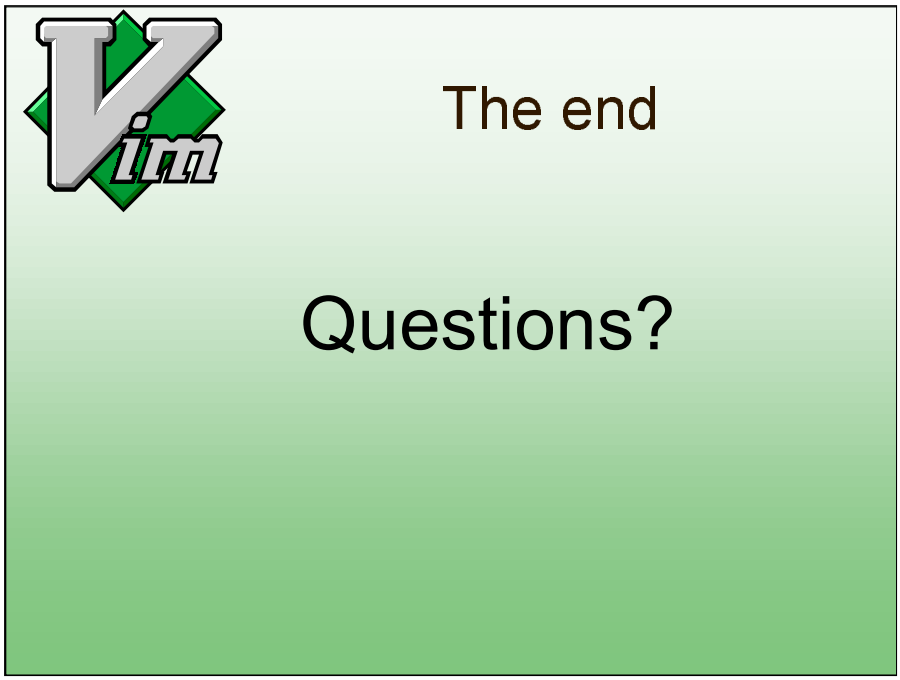
- UTF-8 support: Use many languages at the same time (the screen dump shows Thai and Braille)
- Multi-language support: translated messages and menus
- Vertical window split
- Improved performance for syntax highlighting
- Easier to use (plugins, settings files)

To see the new features, download the Vim 6.0 alpha-test release.

-- subject break --



Work on Vim continues. The complexity makes it progress only slowly now. I was hoping to finish version 6.0 in this year, but it looks like it will be early next year.



I will be around for more information.

I have some extra info about Charityware and helping orphans in Uganda.