Iago Rubio Sanfiz
Michèle Garoche

CSSED User Manual

for versions 0.3.1 - 0.3.0

CSSED User Manual

for versions 0.3.1 - 0.3.0

by Iago Rubio Sanfiz and Michèle Garoche

Revision History

Revision 1.14 10/02/2005 Revised by: MG
Updated to cssed versions 0.3.1 - 0.3.0, improved stylesheets.
Revision 1.13 29/12/2004 Revised by: IRS
Added instructions for installation.
Revision 1.12 24/08/2004 Revised by: MG
Updated to cssed version 0.2-1 - Added jadetex.cfg, scripts for images, customized dsssl and css stylesheets.
Revision 1.11 10/04/2004 Revised by: IRS
Updated to cssed version 0.1-5.
Revision 1.8 31/12/2003 Revised by: IRS
Initial release for cssed version pre0.1-2.

Web site: http://cssed.sourceforge.net

Project page: http://sourceforge.net/projects/cssed

This manual has been written with GNU emacs and bluefish in SGML DocBook format with customized stylesheets derived from NWalsh and dsssl stylesheets. The PostScript, PDF and HTML outputs are produced with customized scripts via OpenJade. All images have been created with screen shots and modified with the Gimp and ImageMagick.

Built on December 16, 2020.

73 pages
Published February, 13th 2005

Table of Contents

List of Figures

Introduction

What is cssed?

cssed is a small program for editing CSS style sheets, it is developed as an Open Source project under GPL. You can download it from the cssed site (http://cssed.sourceforge.net) or from the Source Forge's Project page (http://sourceforge.net/projects/cssed) .

It tries to ease the edition and creation of style sheets to format the output and improve accessibility in web developing.

What is CSS?

CSS is the acronym for Cascading Style Sheets, and it is a w3c.org recommendation. CSS is intended to fix some caveats in HTML formatting. HTML has been created to store data in a convenient manner, but not specifically to present it in a visual manner. Further extensions to HTML gave it the ability to perform rendering tasks, and to embed color and rendering information in some HTML tags attributes.

This finally leads to many problems rendering the information stored in the HTML documents:

- Each time a color or rendering attribute changes in a set of HTML documents, each of its tag attributes must be changed.
- The *face* attribute of the font tag only allows for one font name, so if this font is not present in the client's browser it uses the default font, but still processes all the tag's attributes. If a developer uses the My Font font with a size of 8 - that is fine displayed in a browser - the default font with a size 8 will be used if My Font is not present; and it is not uncommon that the default font with size 8 makes the site impossible to browse.
- HTML must be further extended to allow speech based browser for visual impaired as it has no aural attributes to deal with that kind of browsers.
- The formatting attributes into an HTML document cannot be changed on a browser basis or to any other media than the web browser; so HTML formatted sites used to look really bad when printed, and were not fully browser independent. When you read in a website "This site have been optimized for this browser and this version with those screen dimensions", actually the webmaster is telling you "This site will be well rendered only in the browser and on the screen I used to develop it".

CSS fixes all those problems and others, allows developers to easily change the entire document style with changes in just one document, to show each browser a different page style improving user's browsing experience, to set rules for speech machines, PDA, mobile phone, TV, print and even terminal output.

One of the best things of CSS is that it is small and easy to learn. Much more easy than learning HTML for example.

Further information

- You can find tons of information and links at the w3c.org (http://www.w3.org/Style/CSS/) CSS site
- CSS1 specification (http://www.w3.org/TR/CSS1)
- CSS2 specification (http://www.w3.org/TR/CSS2)
- Position is everything (http://www.positioniseverything.net/), a site to learn how to roundabout browser bugs to get the correct rendering

Who must read this manual

This is the cssed application User's Manual. It is not a developer guide to the application, nor a CSS manual. Just read it if you are a cssed user.

New in this release

Plugins

The major change in this release the introduction of plugins, which add functionalities to cssed. They are loadable/unloadable at run time via the Plugins menu, and distributed as separate packages.

Official plugins

Available plugins are:

- cssed-filebrowser-plugin: adds a file browser a la gnome2
- cssed-findinfiles-plugin: adds a tab to the footer panel, which allows you to search for a term from a given directory
- cssed-quicksearch-plugin: adds a search field to the toolbar, for use within the active document
- cssed-tag-plugin: adds a new tab to the footer panel, to create and browse tags created with exuberant-ctags
- cssed-vte-plugin: adds a console to the footer panel, from which you can execute any command line tool

Unofficial plugins

Another plugin not included in the official release is:

- cssed-plugindoc-plugin: adds an item to the plugin menu to browse the documentation for plugins, in case you want to write your own plugin.

Addition to the menus

Another important change is addition to the menus, improving the user's experience.

File menu

- File⟶Close
- File⟶Close all
- File⟶Revert
- File⟶Save all

View menu

These changes are effective in version 0.3.1:

- A new menu which allows you to create, load, or modify documents lists, i.e. lists of documents you want to open quickly in a cssed session. It consists of:
  - View⟶Document list⟶Opened documents
  - View⟶Document list⟶Open from file
  - View⟶Document list⟶Blank document list

- Toolbar submenu and its contents have been moved to this menu.

Document menu

- A new submenu (version 0.3.1) in the Set EOL mode menu to convert line endings from one system mode to another one:
  - Document⟶Set EOL mode⟶Convert EOLs⟶To Mac (CR)
  - Document⟶Set EOL mode⟶Convert EOLs⟶To Unix (LF)
  - Document⟶Set EOL mode⟶Convert EOLs⟶To Doc / Windows (CR/LF)

- A new menu which allows you to change the encoding of the document with:
  - Document⟶Force Encoding⟶Default
  - Document⟶Force Encoding⟶UTF8
  - Document⟶Force Encoding⟶DBCS

- Changes to the highlighting menu to introduce the possibility of highlighting other types of documents than just CSS ones. The document is highlighted according to its extension, but you can change it if not appropriate. Only the CSS highlighting is defined in Preferences and applied immediately. Highlighting for other types of document are currently hardcoded; this will change in a future release. The new items are the following:
  - Document⟶Highlighting⟶Plain text
  - Document⟶Highlighting⟶CSS
  - Document⟶Highlighting⟶HTML/PHP
  - Document⟶Highlighting⟶SH
  - Document⟶Highlighting⟶C/CPP
  - Document⟶Highlighting⟶Apache/.htaccess
  - Document⟶Highlighting⟶PERL
  - Document⟶Highlighting⟶Python
  - Document⟶Highlighting⟶XML
  - Document⟶Highlighting⟶Diff/patch
  - Document⟶Highlighting⟶Makefile

- A Character set menu for switching between various character sets, especially useful when dealing with Asian languages:
  - Document⟶Character set⟶ANSI
  - Document⟶Character set⟶East Europe
  - Document⟶Character set⟶GB 2312
  - Document⟶Character set⟶Hangul
  - Document⟶Character set⟶Shift JIS

- A new Tab background colour menu (in version 0.3.1) which allows you to change the background colour of the document tabs for easier recognition:
  - Document⟶Tab background colour⟶Revert to original colour
  - Document⟶Tab background colour⟶Red
  - Document⟶Tab background colour⟶White
  - Document⟶Tab background colour⟶Blue
  - Document⟶Tab background colour⟶Yellow
  - Document⟶Tab background colour⟶Green

CSS menu

These changes are effective in version 0.3.1:

- Validate and dump, Validate only, Scan selector have been moved to this menu.

Help menu

- A new menu to browse the cssed web site (Help⟶Cssed web site)
- Another one to access the online documentation for the latest release via one of the proposed browsers (Help⟶Cssed documentation on line).

This feature is for Mac OS X users only. It is experimental and will be extended to other systems in a future release.

Other changes

Other changes affect the user interface:

- The scratch pad in the footer panel has been split into two pads: a temporary one on the left side, and a permanent one on the right side.

Chapter 1
Installation

You can download the latest version of cssed from Source Forge's Project page (http://sourceforge.net/projects/cssed) or from the cssed site (http://cssed.sourceforge.net).

From both sites, you can download it in binary (compiled) or source form.

Easy installation

Here are the instructions to install cssed either from binary or source within a package manager system.

RPM Users

Binary installation

To install a binary RPM of cssed, you need to download the latest RPM package (named `cssed-VERSION-BUILD.rpm`) and use the following commands as root:

```
rpm -ivh cssed-VERSION-BUILD.rpm
```

Apt or Yum RPM installation

To install cssed RPMs via apt or yum, take a look at the instructions on Dag's repository (http://dag.wieers.com/home-made/apt/FAQ.php), especially the instructions on how to use the repository (http://dag.wieers.com/home-made/apt/FAQ.php#B1).

Once configured your prefered package manager, you only need to issue those commands to install cssed on your system:

```
yum install cssed
```

Or:

```
apt-get cssed
```

Debian users

Debian users must download the *.deb file called `cssed_VERSION_ARCH.deb` and use the command:

```
dpkg --install cssed_VERSION_ARCH.deb
```

Gentoo users

Gentoo users have no need to perform any of the steps above, just use the command:

```
emerge cssed
```

You can get further information at the portage site (http://www.gentoo.org/doc/en/portage-user.xml).

Mac OS X users

The package is available via Fink (http://fink.sourceforge.net/).

As of version 0.3.0, cssed will no more be updated on the 10.2.x tree, the 0.2.1 release remains available for 10.2.x users in the unstable branch. For 10.3.x users it is available in both source and binary form (when updated), with the usual delay of two months before it is available on the stable tree, provided that all required dependencies are in that tree.

To install the binary on Mac OS X:

1. Install Apple's X11, or the appropriate XFree86/Xorg packages, be it the case.
2. Open a terminal and type:

   `sudo apt-get update` to update the packages list.
3. Enter your password when asking for, then install cssed with:

   ```
   sudo apt-get install cssed
   ```

To compile the source on Mac OS X:

1. Install Apple's X11 and X11SDK, or the appropriate XFree86/Xorg packages, be it the case.
2. Enable Fink unstable tree (you may use the stable tree on the 10.3 branch)

   a. Open a terminal and type:

      ```
      sudo pico /sw/etc/fink.conf
      ```

   b. Enter your password when asking for, change the Trees lines as follows:

      ```
      Trees: local/main stable/main unstable/main stable/crypto unstable/crypto local/bootstrap
      ```

   c. Save the result (be aware that the above line should not be split).
3. Update your Fink installation with:

   ```
   fink selfupdate
   ```
4. Install cssed:

   ```
   fink install cssed
   ```

Windows users

You need to install GTK-2.2 or GTK-2.4 runtime libraries, prior to install cssed.

The better place to download the GTK runtime libraries is at gimp-win home page (http://gimp-win.sourceforge.net/)

Once installed, pick the `cssed-$VERSION-win32.exe` file from the Source Forge project's page (http://sourceforge.net/projects/cssed), double click on it and follow the instruction provided by the application's installer.

Building cssed from source

To install cssed from source:

1. Download the tarball package (named `cssed-VERSION-BUILD.tar.gz`).
2. Uncompress it with the command:

   ```
   tar xfz cssed-VERSION-BUILD.tar.gz
   ```
3. Once this has been done, you must move to the created directory.
4. To install cssed type **./configure**, **make** and **make install** to install the standard version of cssed on your system.

Configure options

Here are the non standard configure options:

`--with-terminal` : Adds an (experimental and unfinished) terminal window to the footer notebook.

`--with-ipc-queue` : Adds an ipc queue for open signals (when available on the system).

`--with-help-menus` : Adds a submenu item to the help menu to access the cssed web site and the online documentation.

`--with-debugging-output` : This is for testers to put on the standard output lots of debugging information. It decreases speed so it is not recommended if you are not a tester.

Mac OS X users: you are invited to consult the section How to compile myself (http://fink.sourceforge.net/faq/usage-general.php?#compile-myself) on the Fink's web site.

Building from CVS

You can get the latest development version of cssed, following the instructions at the Source Forge's Project page (http://sourceforge.net/projects/cssed).

Once you have got a working checkout of cssed from CVS, you can update it with the command:

```
cvs update
```

Take into account that each time new files are added some builds script may change. If your build script stops working after a CVS update, you can rebuild it with two scripts in the root folder of the cssed distribution:

- autogen.sh: it uses a sort of brute force rebuilding every build script in the source tree.
- bootstrap: it tries to rebuild only the needed files.

To run it, simply cd into the root cssed distribution folder and type:

```
./autogen.sh
```

or:

```
./bootstrap
```

Both work and choosing one of them is up to you.

Mac OS X users: do not to use bootstrap as it may mess up the source tree.

Plugins installation

You can download the plugins from the cssed site (http://cssed.sourceforge.net).

The cssed-plugindoc-plugin is available here (http://micmacfr.homeunix.org/cssed/cssed-plugindoc-plugin-0.1.tar.gz). For Mac OS X users, it is in the Fink distribution.

Plugins installation is done the same way the main application is. You should have cssed installed beforehand.

Mac OS X users: you can install plugins without cssed installed, the built process will install it for you.

Chapter 2
User interface

This section describes the interface of cssed. For description of plugins interface, see Chapter 4.

Description

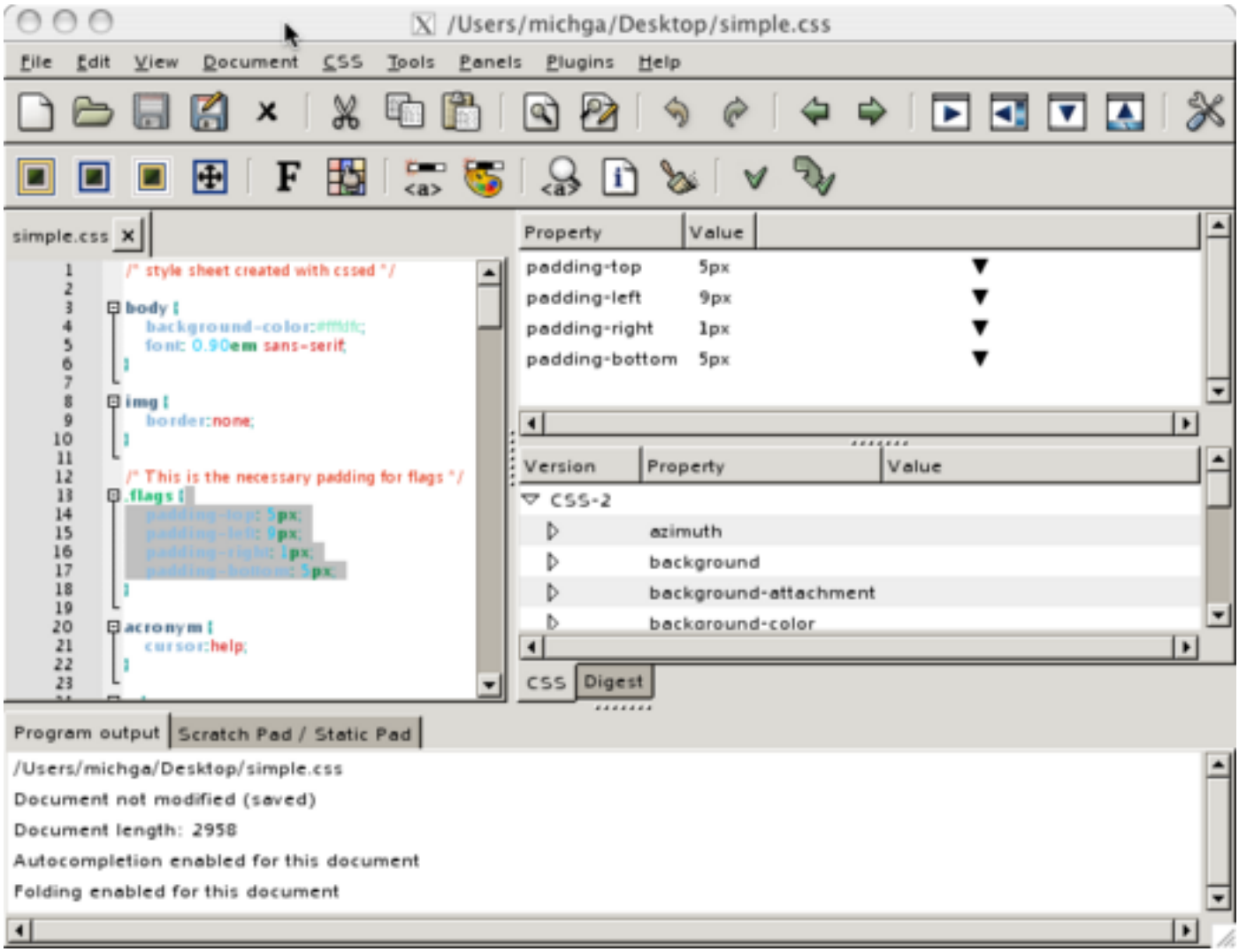


Figure 2-1. cssed user interface

The cssed user interface is divided into three main sections:

- The menu and tool bars: give quick access to the cssed main functions.
- The editor notebook: is located in the upper right corner of cssed interface and contains all editor windows.
- The side panel: Located on the right side of cssed interface, it comprises two tabs. The first one contains the selector scanner and the CSS property's tree view, the second one contains the digest.
- The program output, the scratch pad/static pad, and optionally the terminal: are in the footer of cssed interface.

Additionally, you will find tips just by pausing the mouse over most elements of the interface, and contextual menus are available in the document window and in the scratch pad/static pad.

The menu and tool bars

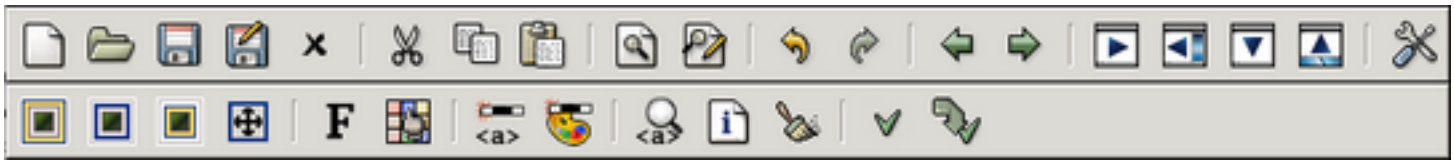


Figure 2-2. Menu and tool bars

The menu and tool bars are common to other editors. From the menu, you can perform most cssed tasks; you will use the tool bars to perform the most common ones. All menus have got menu accelerator keys, but not all items in menus.

Mac OS X users: the menu accelerator keys are not available on your system.

Here we will inspect the menus, and let you know the possible equivalence in the tool bars.

The File menu

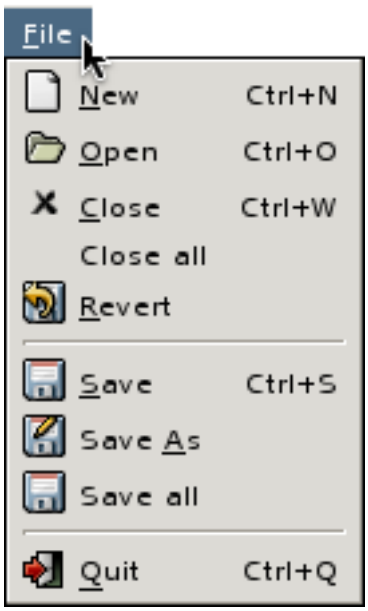


Figure 2-3. File menu

This is where you perform actions on files:

- New, Open
- Close, Close all
- Revert
- Save, Save as, Save all
- Quit

Those actions, but Close all, Revert, Save all, and Quit, are accessible from the first part of the main tool bar.

Figure 2-4. Equivalence of the File menu in the main tool bar

Notice that the Close action applies to the current document.

Notice also the accelerator keys for most common actions in this menu.

The Edit menu

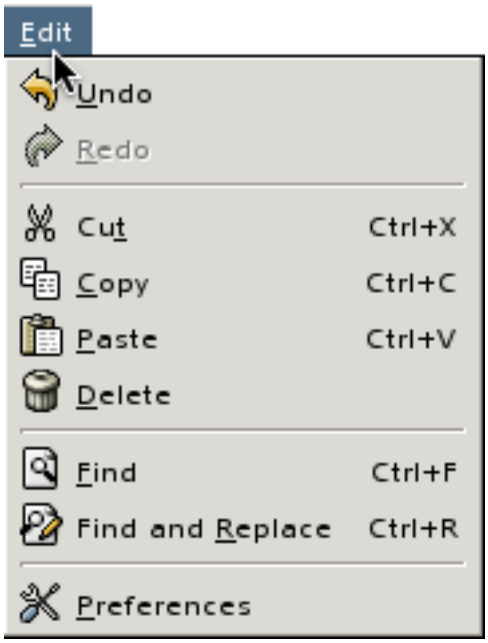


Figure 2-5. Edit menu

This is where you perform actions on documents:

- Undo and Redo
- Cut, Copy, Paste and Delete
- Find, Find and Replace

In addition, this is where you access cssed preferences.

Those actions, but Delete, are accessible from the 2nd, 3rd, 4th and last part of the main tool bar.

Figure 2-6. Equivalence of the first part of the Edit menu in the main tool bar

Figure 2-7. Equivalence of the last part of the Edit menu in the main tool bar

Once again, notice the accelerator keys for most actions in the Edit menu. Notice that **Ctrl**-**Z** for undo works, though not signaled in the menu.

The View menu

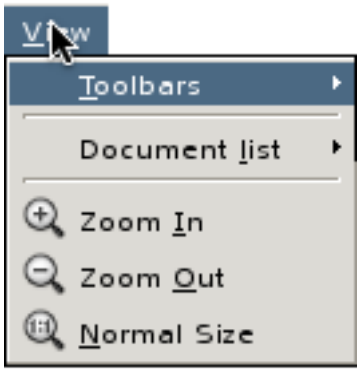


Figure 2-8. View menu

This is where you can change the visual display of the cssed window.

From here you can:

- Hide / Show the main and secondary tool bars
- Change the zoom factor of the current document
- Create, modify and load documents lists which will ease grouping and opening documents

Those actions have no equivalence in the tool bars.

The Document menu

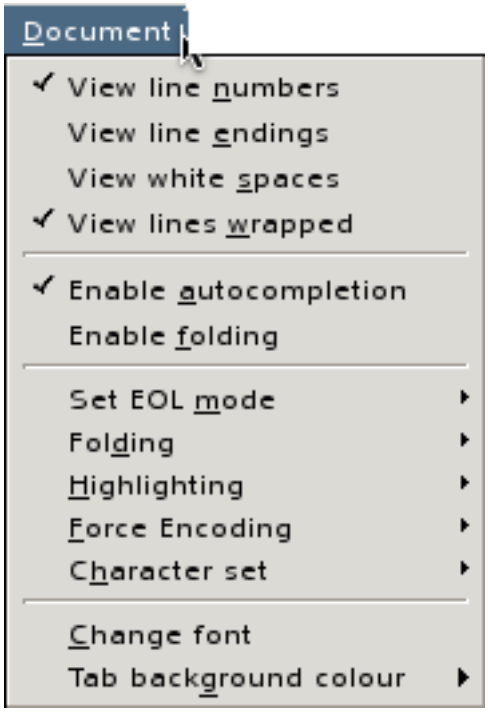


Figure 2-9. Document menu

All actions in this menu are related to the current document. From there, you can:

- Show/hide line numbers, line endings, and white spaces
- Enable/disable line wrapping, auto completion, and folding
- Set or change the EOL mode of the current document

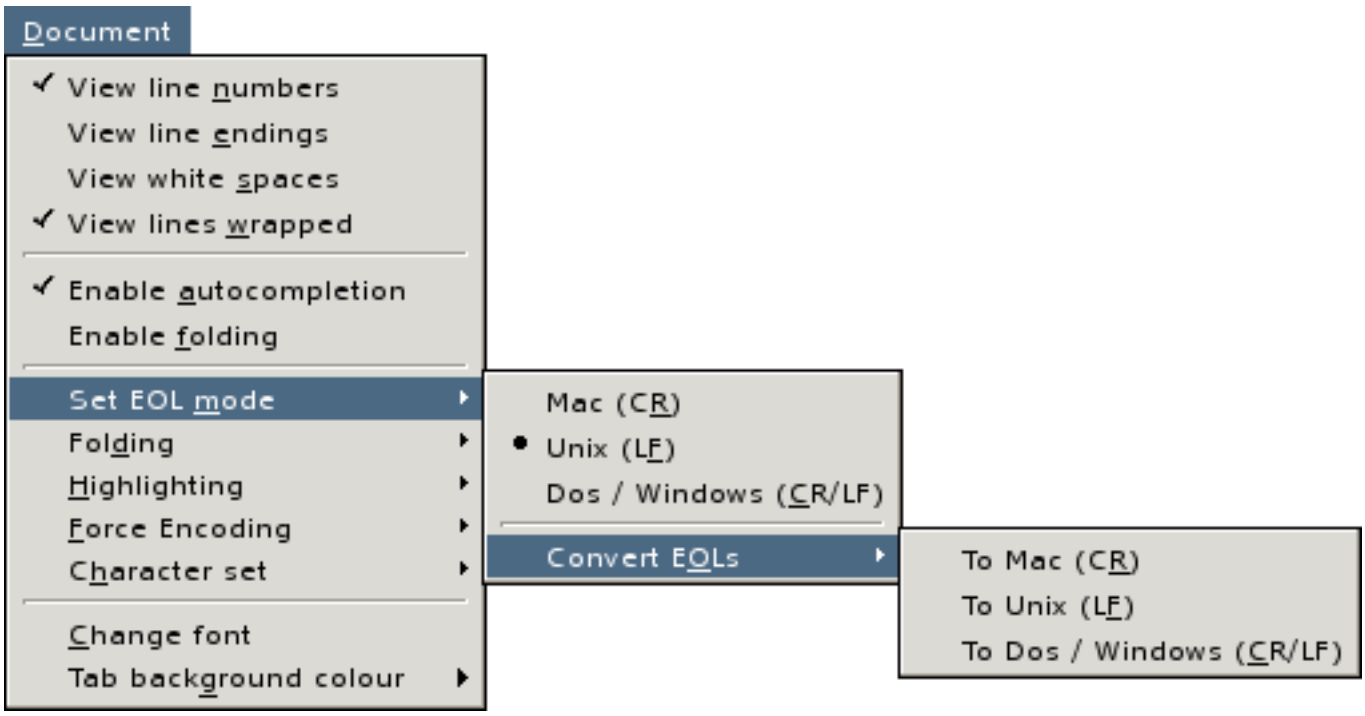


Figure 2-10. Set or change EOL mode

- Fold/unfold all properties

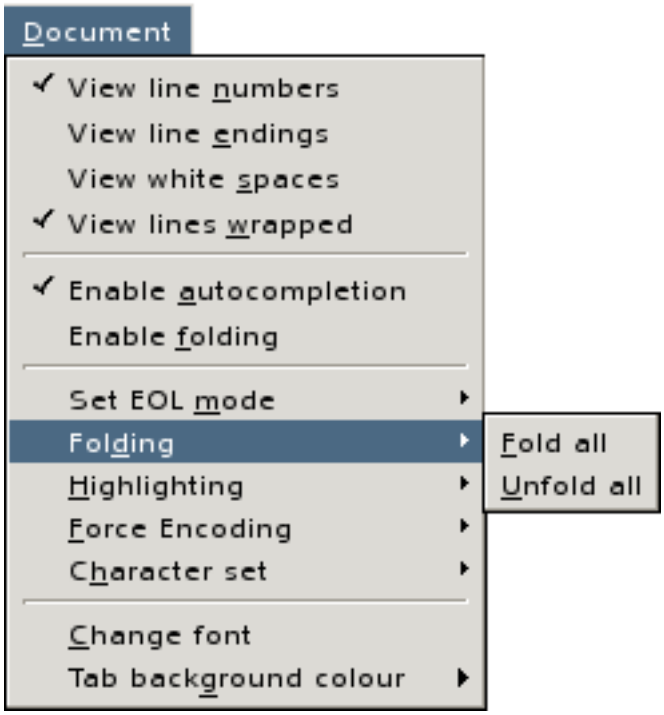


Figure 2-11. Folding/unfolding all properties

- Apply syntax highlighting to the document, according to the type of document

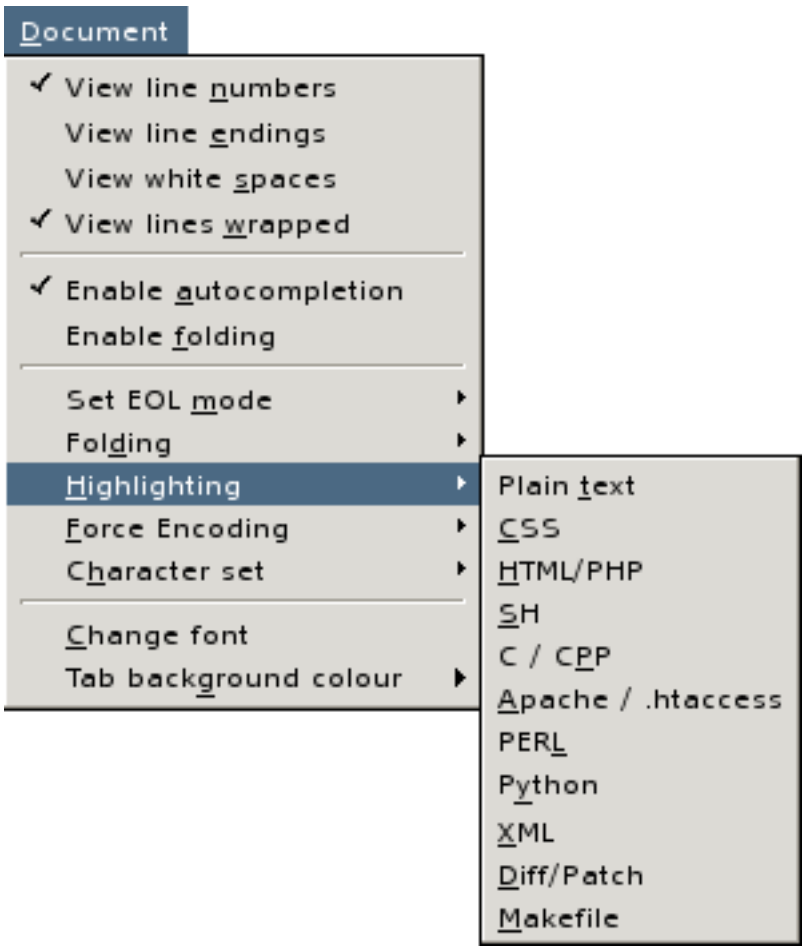


Figure 2-12. Apply highlighting

- Force the encoding of the document

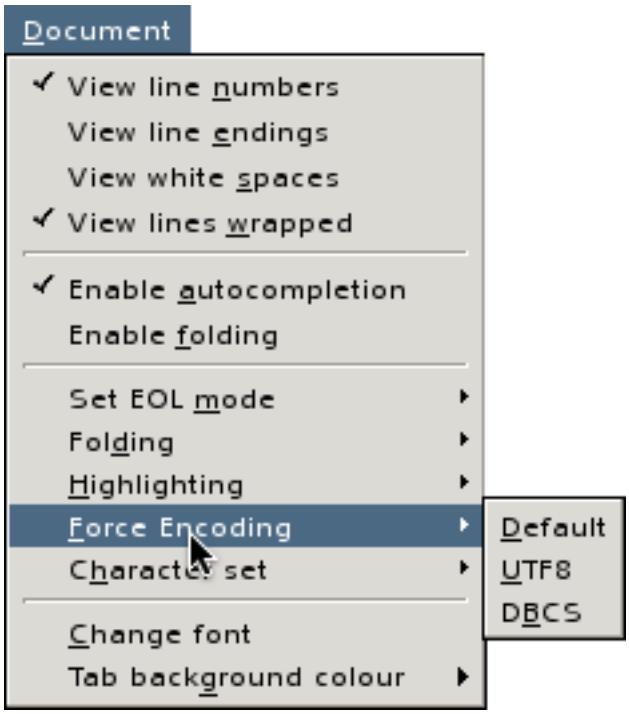


Figure 2-13. Force encoding

Use it with caution, and do not mix several encodings in the same document.

- Change the character set

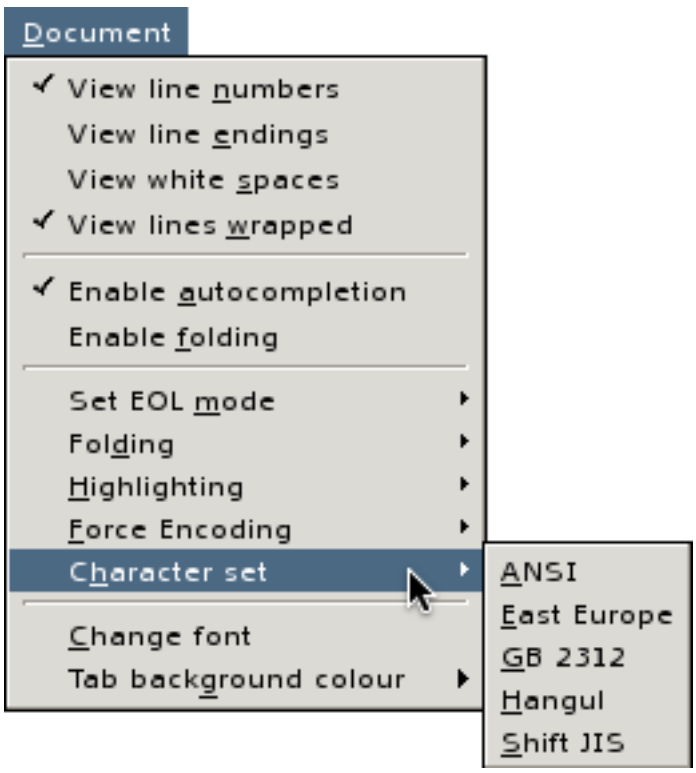


Figure 2-14. Character set

- Change the font of the current document.
- Change the background color of the document tab

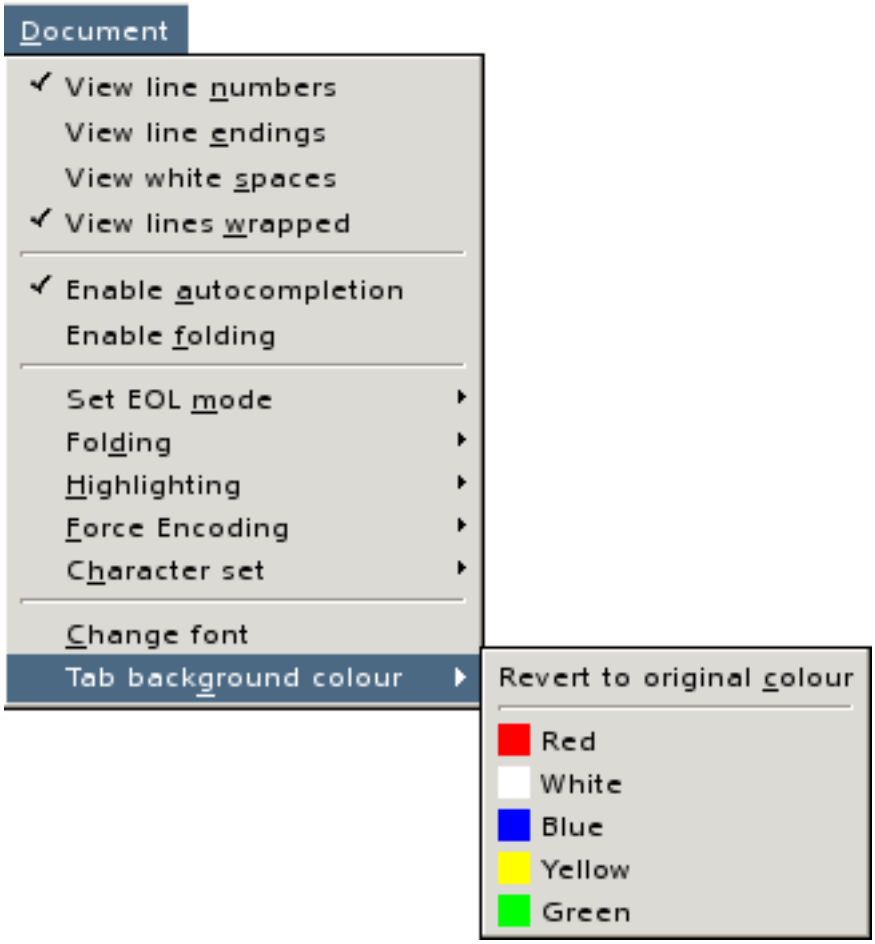


Figure 2-15. Tab background color

There is no equivalence for those actions in the tool bars.

The CSS menu

Figure 2-16. CSS menu

This is where you will find all the dialogs to add properties, attributes, and values to your documents, as well as validate and scan them. The menu is divided into three parts:

- Dialogs, which in turn are divided into three parts:

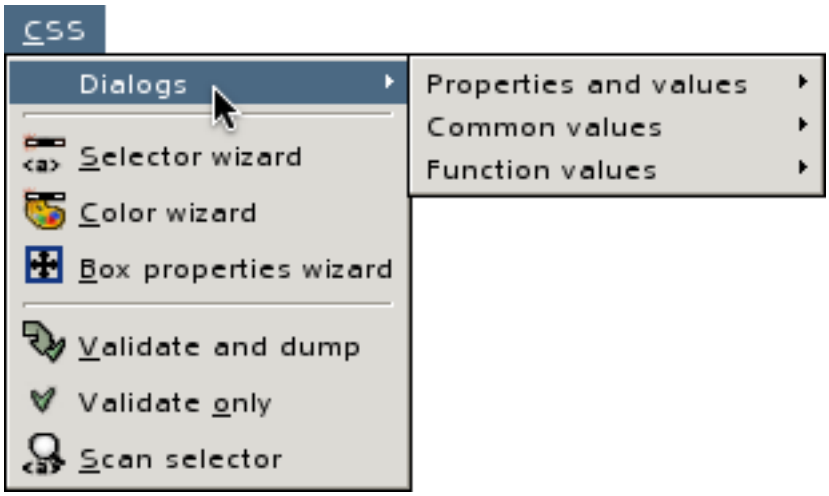


Figure 2-17. CSS Dialogs menu

- Properties and values: This is where you will find all the CSS dialogs for properties and values.

  It contains 66 dialogs to ease CSS creation either aural, visual, or paged.

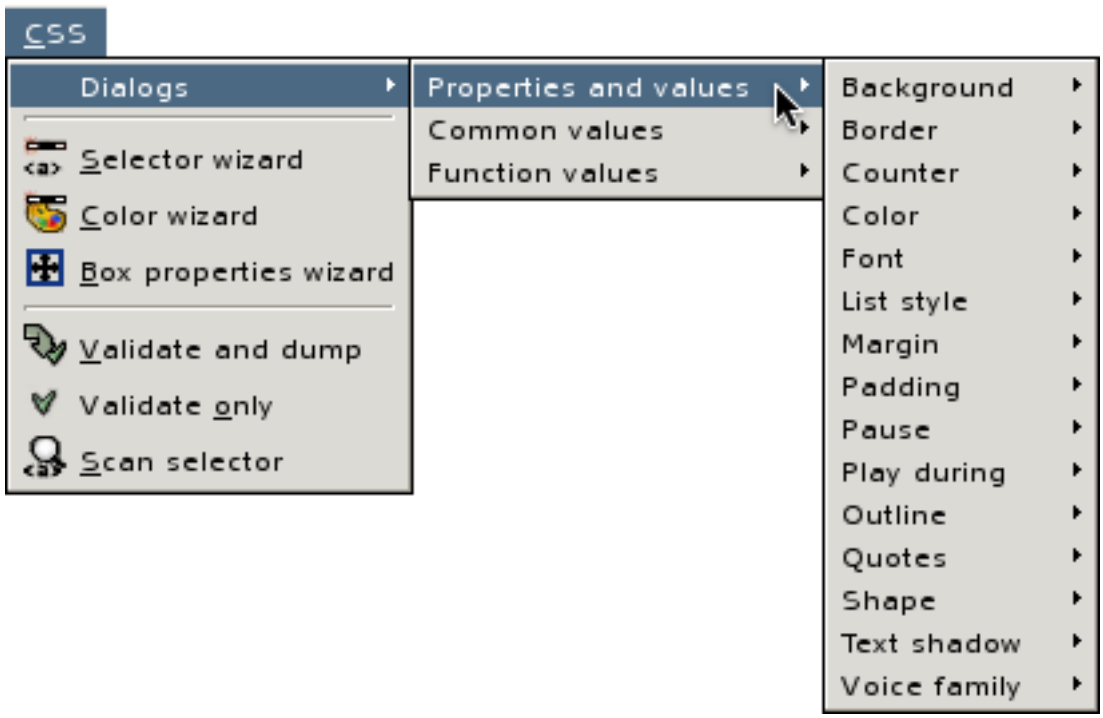


Figure 2-18. CSS Properties and Values menu

Each item leads to another cascade of submenus, to finally end up with access to a given dialog.

- Here you can see the path to the border width style and color item:

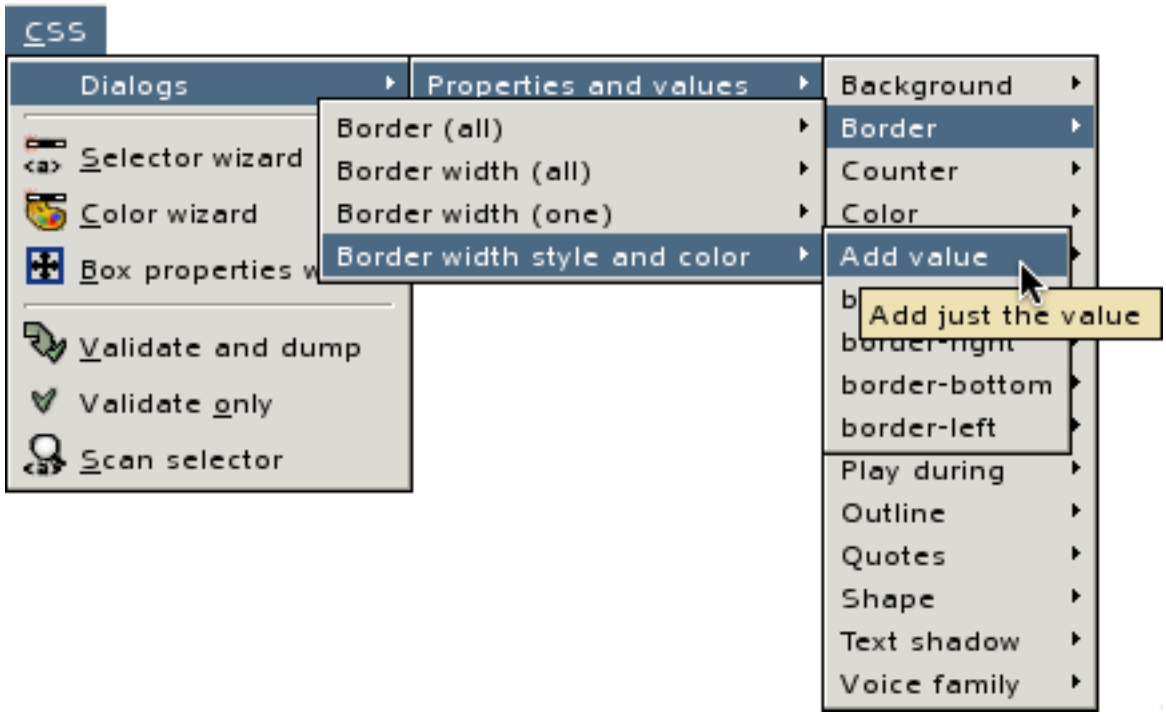


Figure 2-19. Border width style and color item in the CSS menu

The most commonly used dialogs are replicated in the first three parts of the secondary tool bar.

Notice also the balloon help on most of the menus items to make clear which action the menu triggers.

Figure 2-20. Equivalents of CSS properties and values dialogs in the secondary tool bar

- Margin all dialog

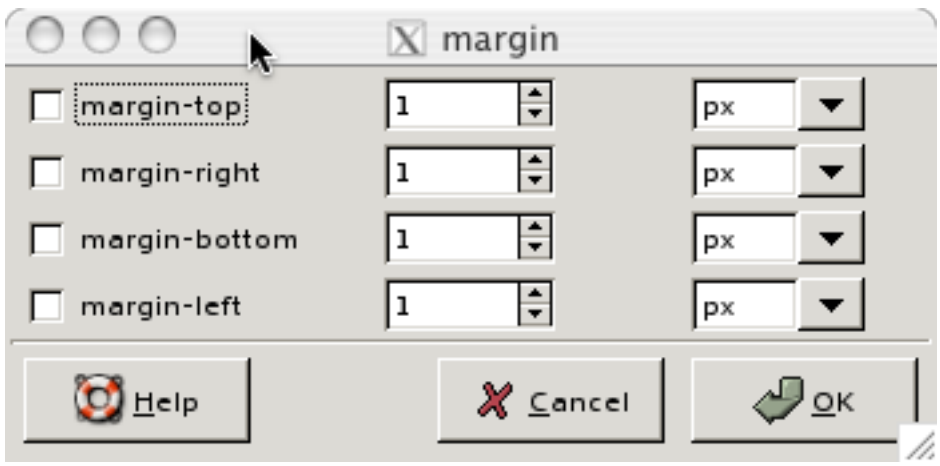


Figure 2-21. Margin all dialog

- Border all dialog

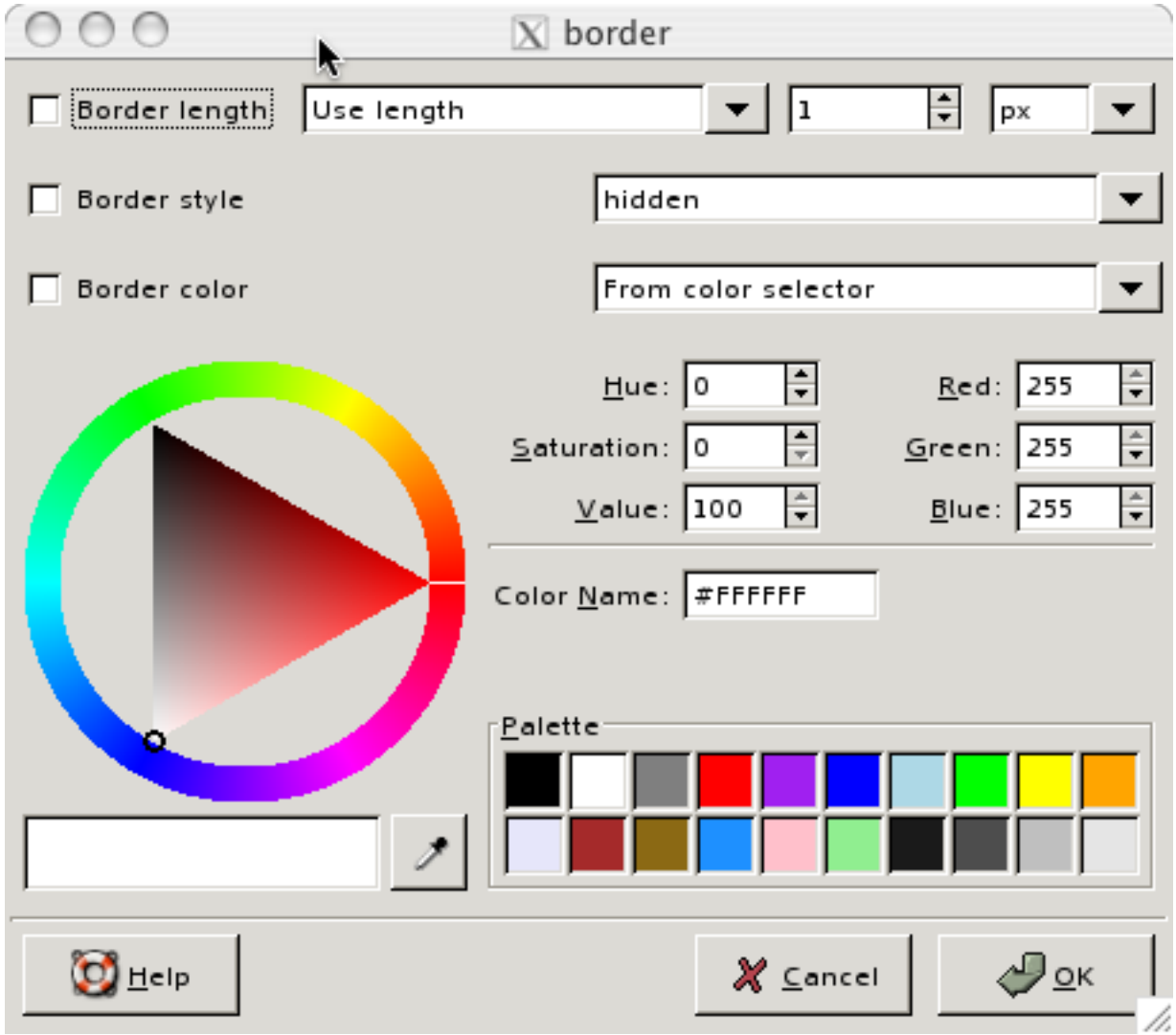


Figure 2-22. Border all dialog

- Padding all dialog

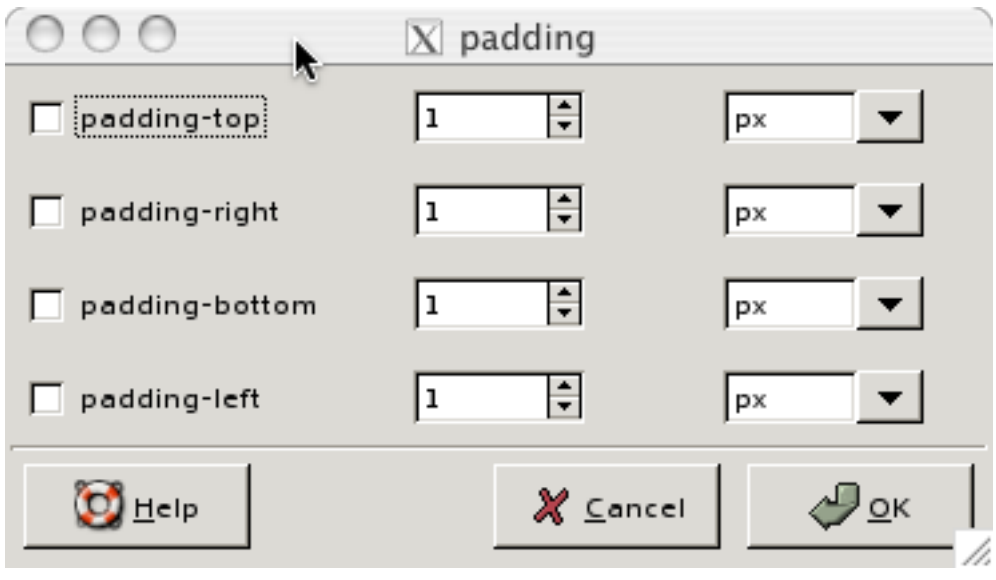


Figure 2-23. Padding all dialog

- Font dialog

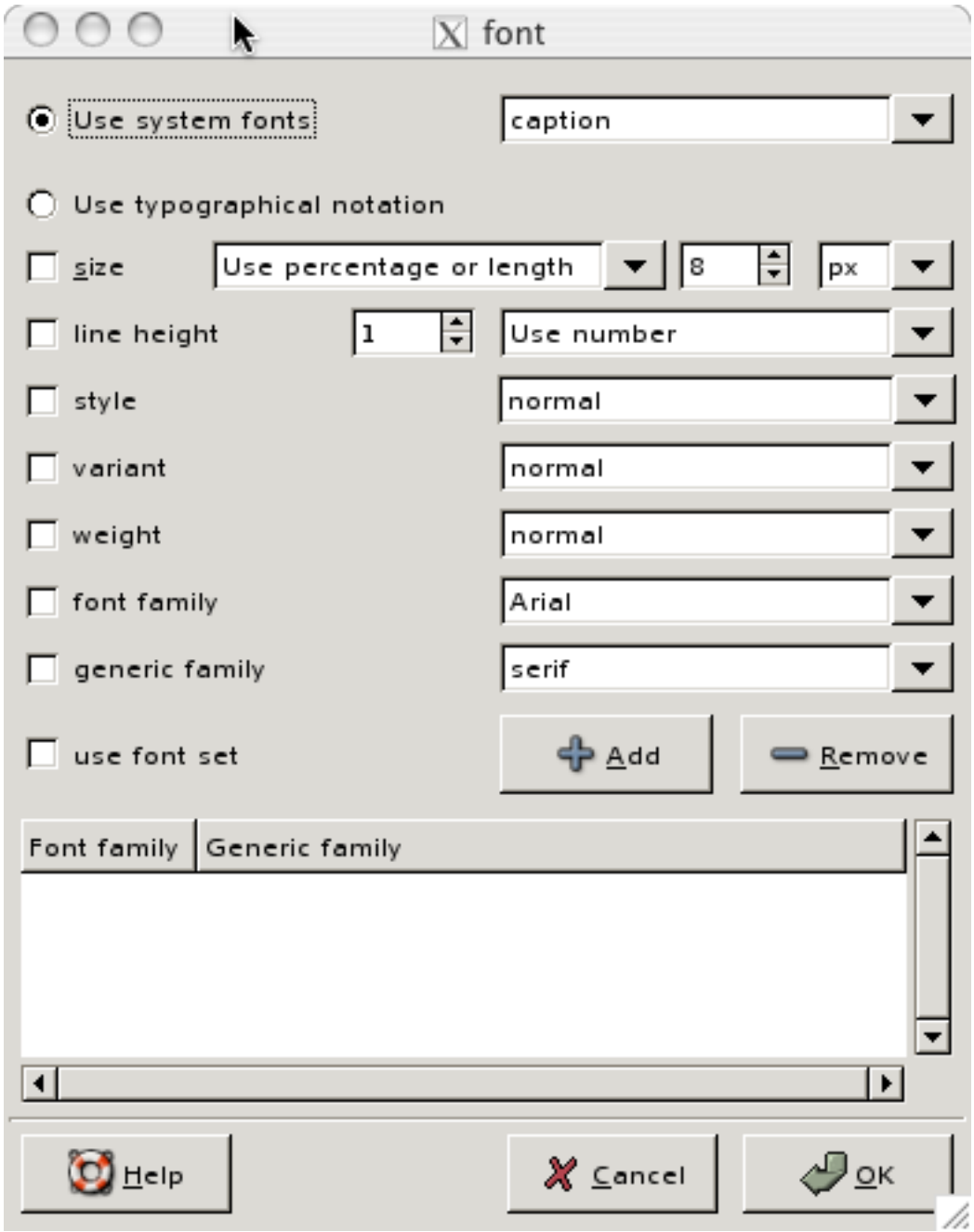


Figure 2-24. Font dialog

- Color dialog

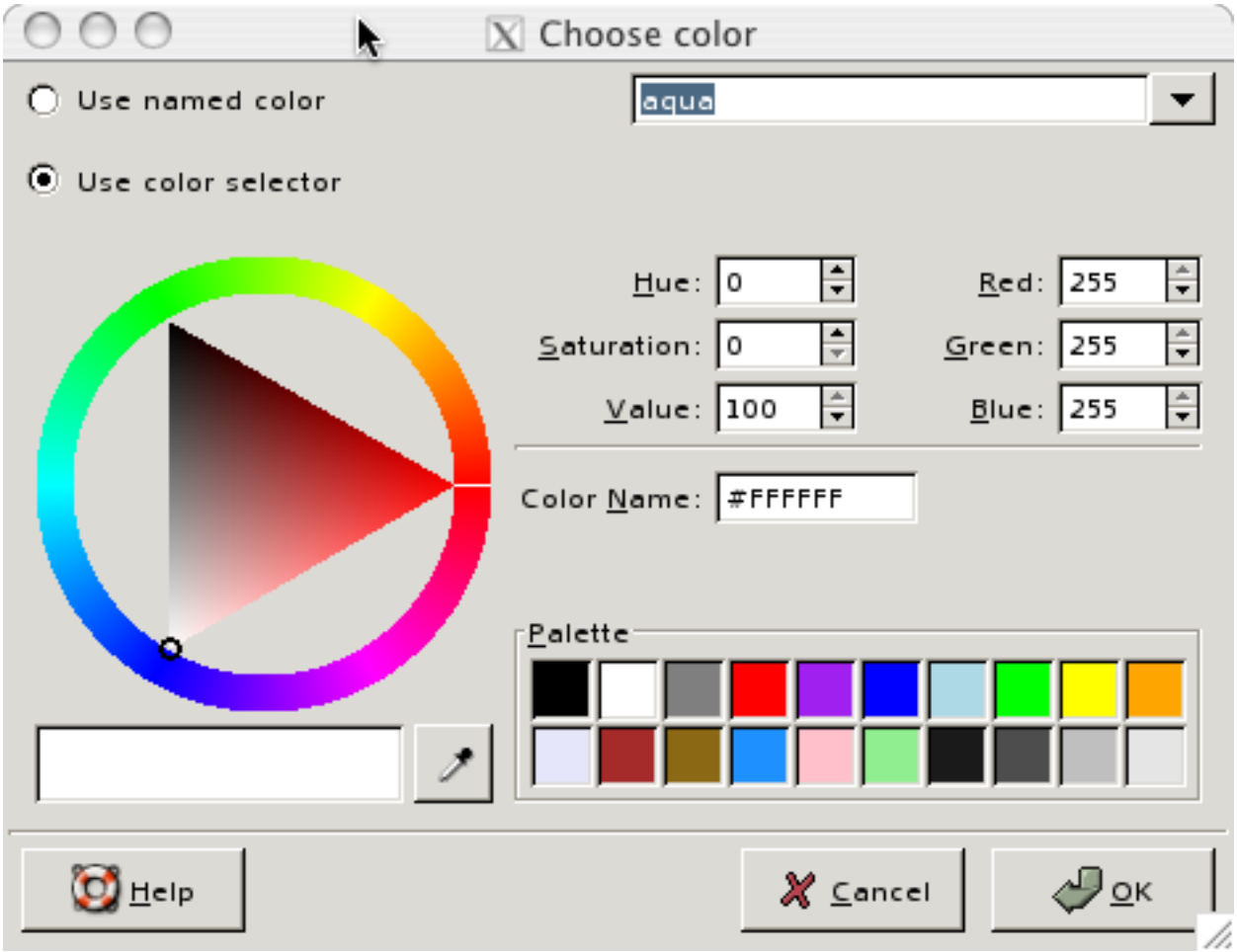


Figure 2-25. Color dialog

- Common values: from there you can insert values for units, strings, and identifiers

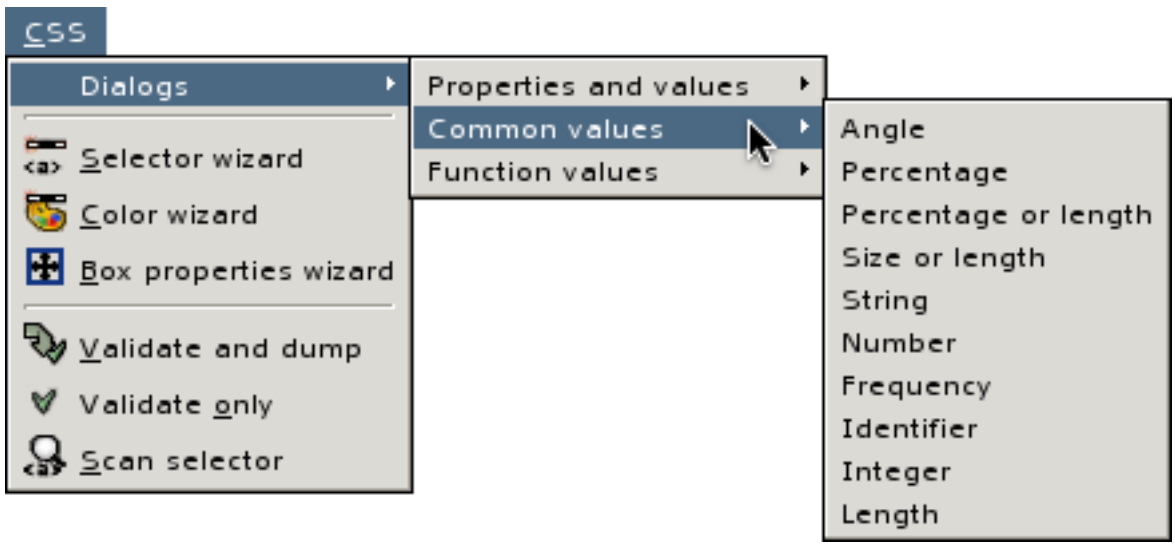


Figure 2-26. CSS Common Values menu

There is no equivalent for those items in the tool bars.

- Function values: those dialogs allow you to insert values for generated contents, i.e. url() and attr()

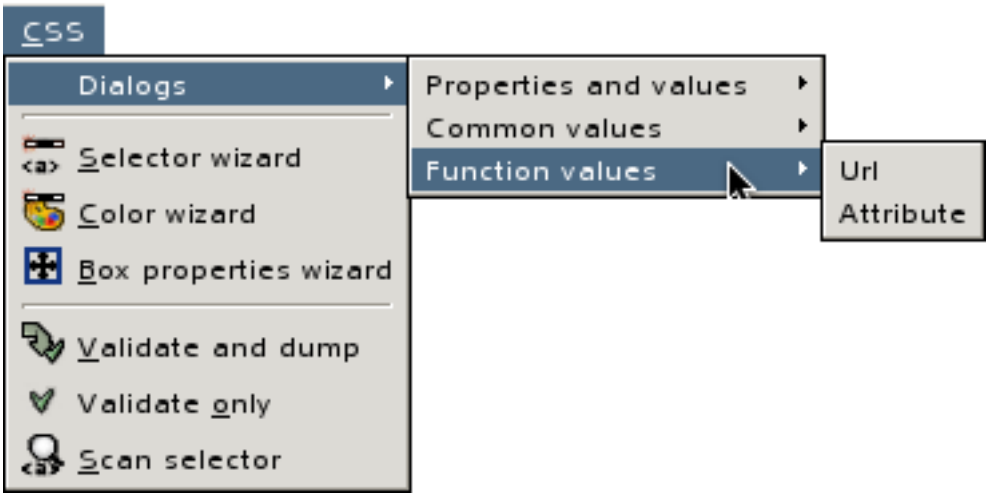


Figure 2-27. CSS Function Values menu

There is no equivalent for this item in the tool bars.

- Wizards: box properties, selector, and color for complex rules

  This is where you go when you have complex properties to build.

  - The box properties wizard allows you to set values for margin, padding, border, and box size properties.

    It is made of three parts:

    - The first part details all properties, so that you can assign different values to each property.

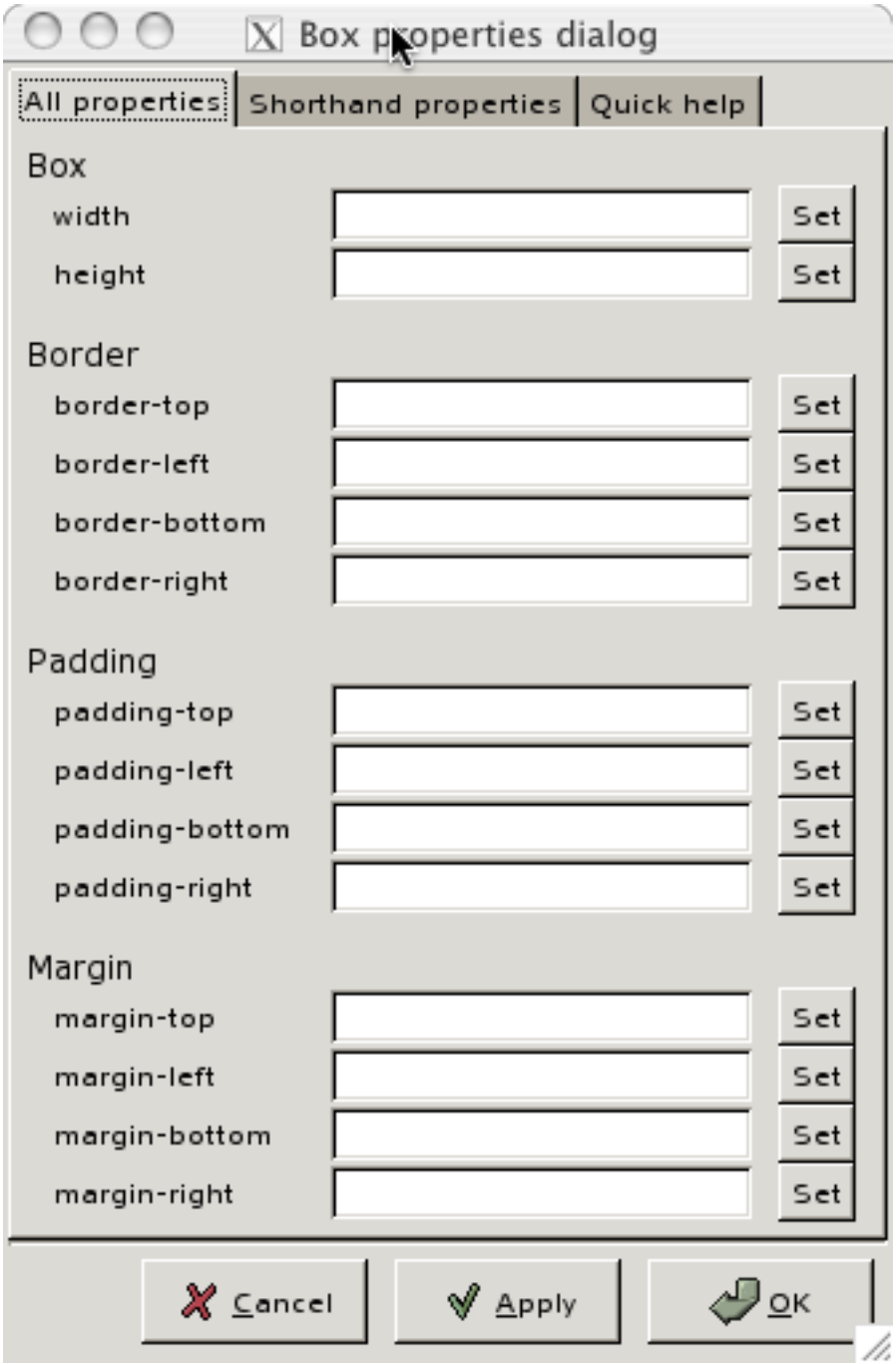


Figure 2-28. All properties in Box Properties wizard

- The second part is devoted to shorthand properties, this way you can set the same value to any sub-property of a major property (e.g. same color to each side of a border).

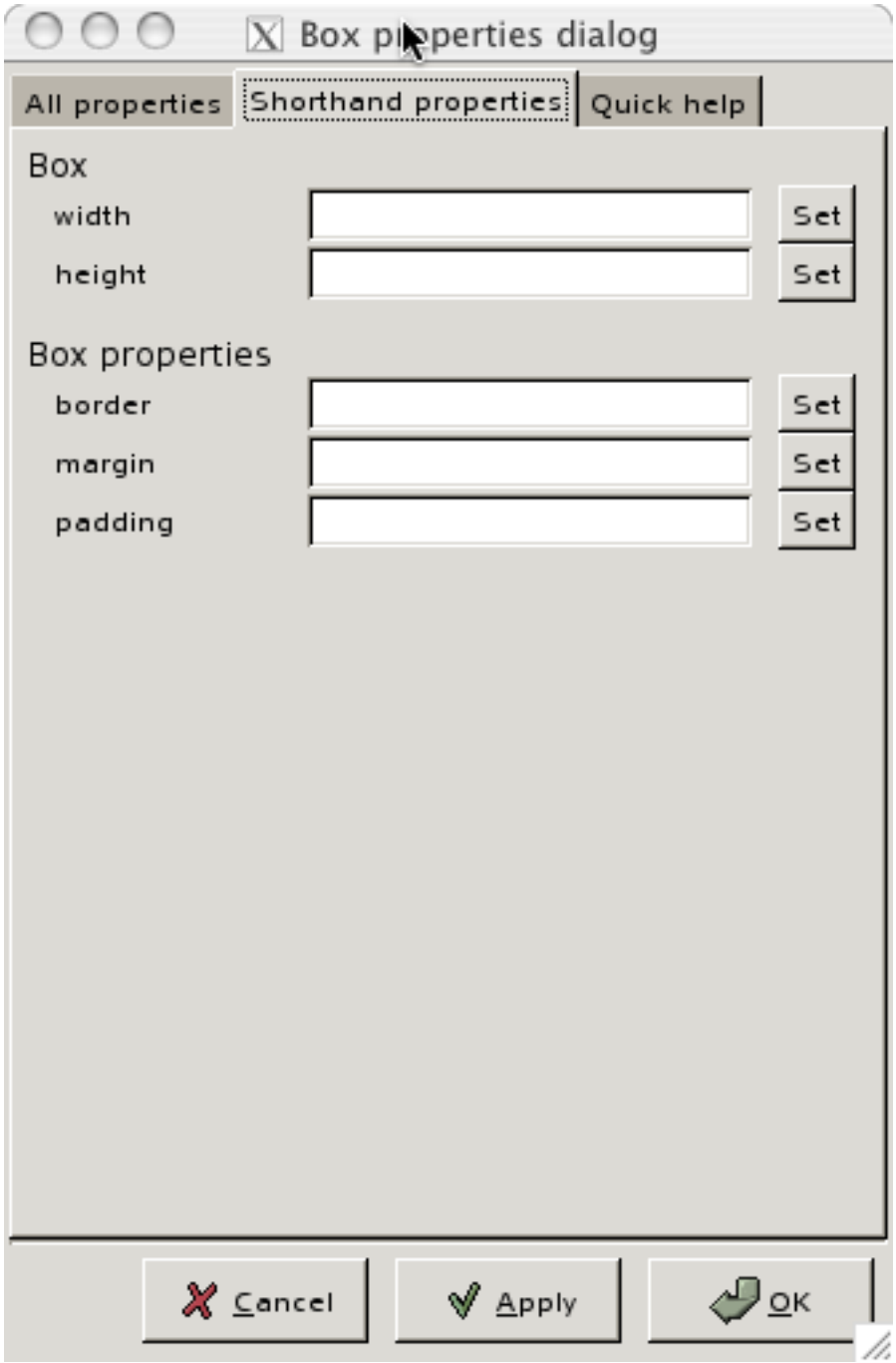


Figure 2-29. Shorthand properties in Box Properties wizard

- The third part is a quick help for you to locate quickly the various concepts in a CSS box.

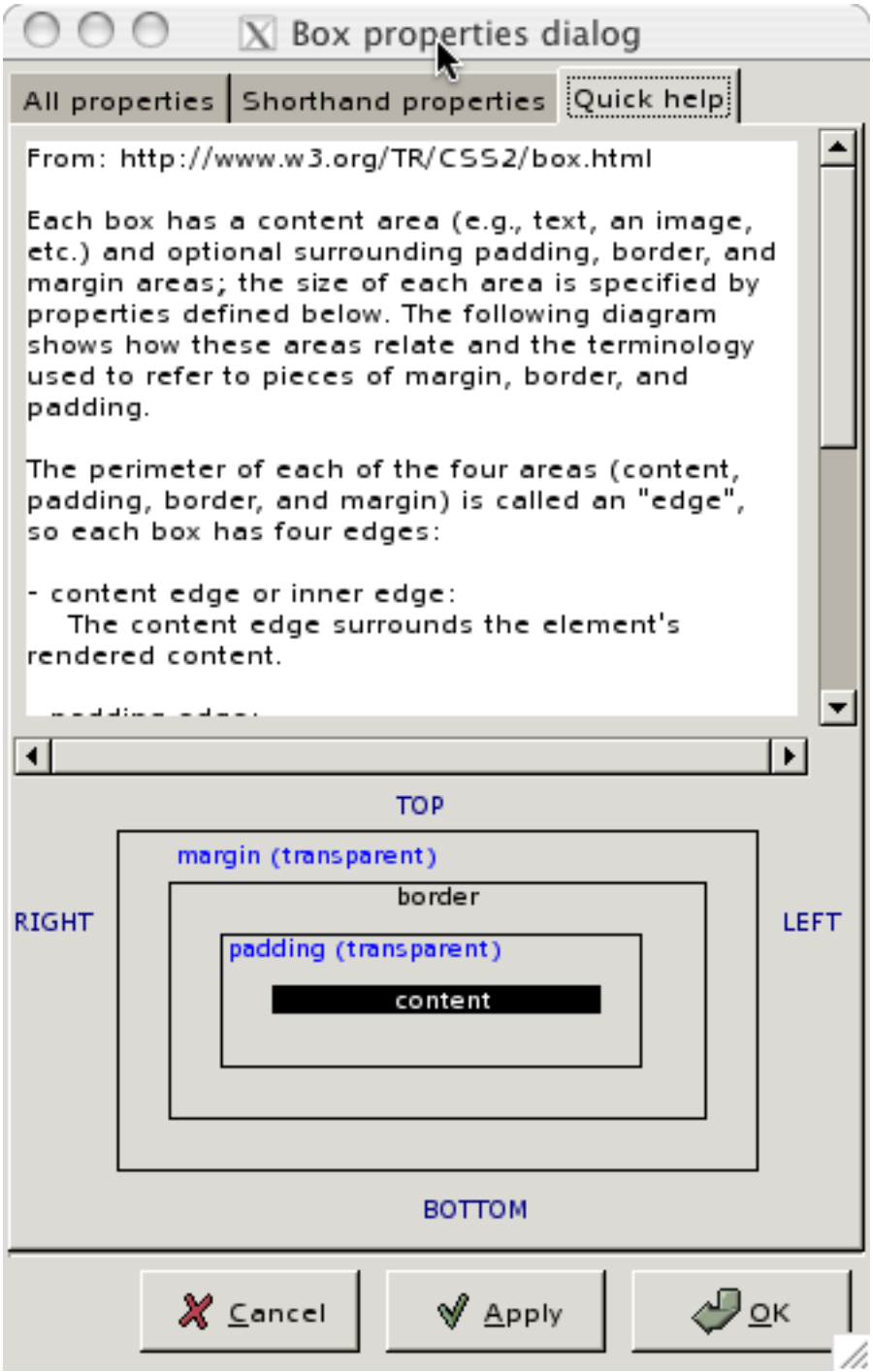


Figure 2-30. Quick help in Box Properties wizard

- The selector wizard helps you to construct particular or general rules for:
  - single selectors

Figure 2-31. Single selector wizard

- child selectors

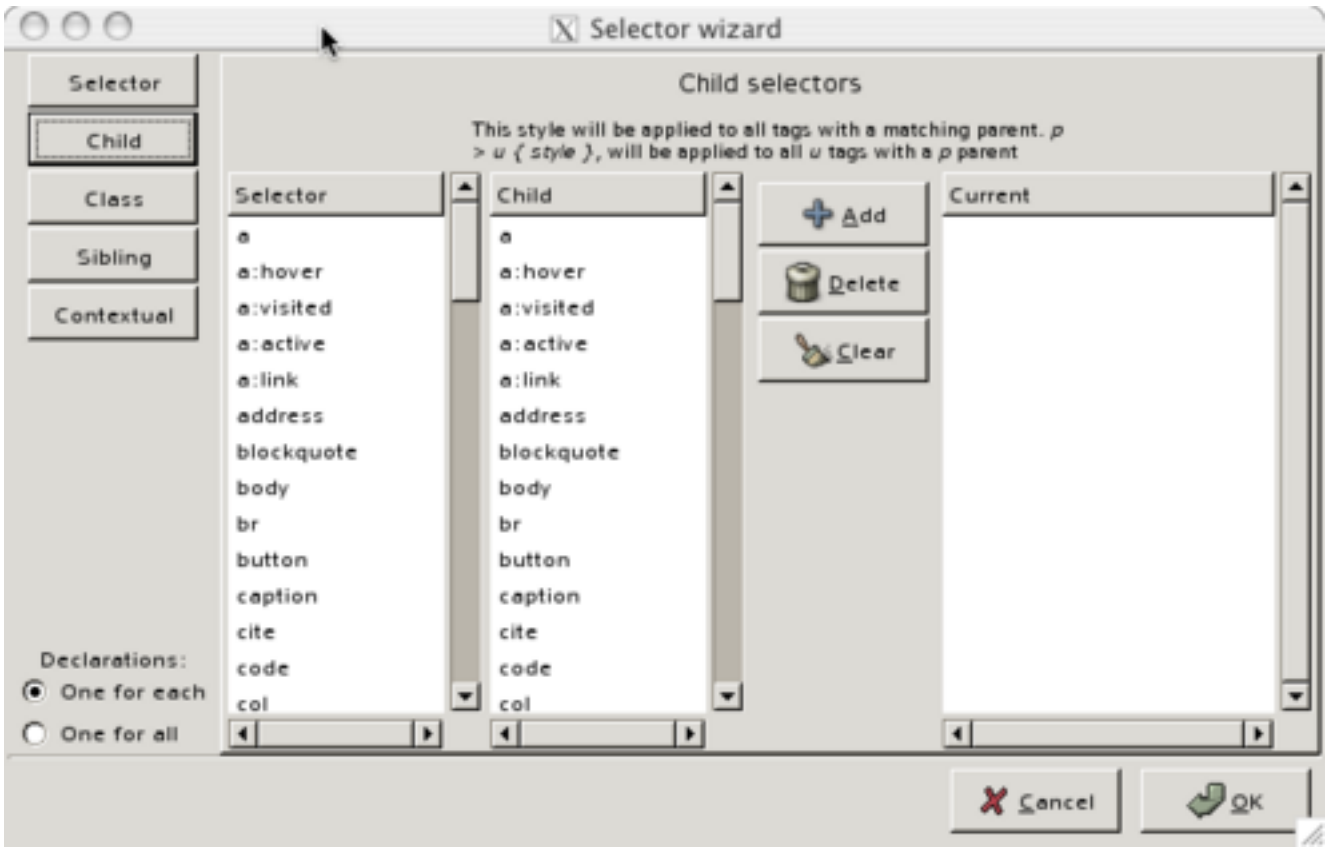


Figure 2-32. Child selector wizard

- class selectors

Figure 2-33. Class selector wizard

- sibling selectors

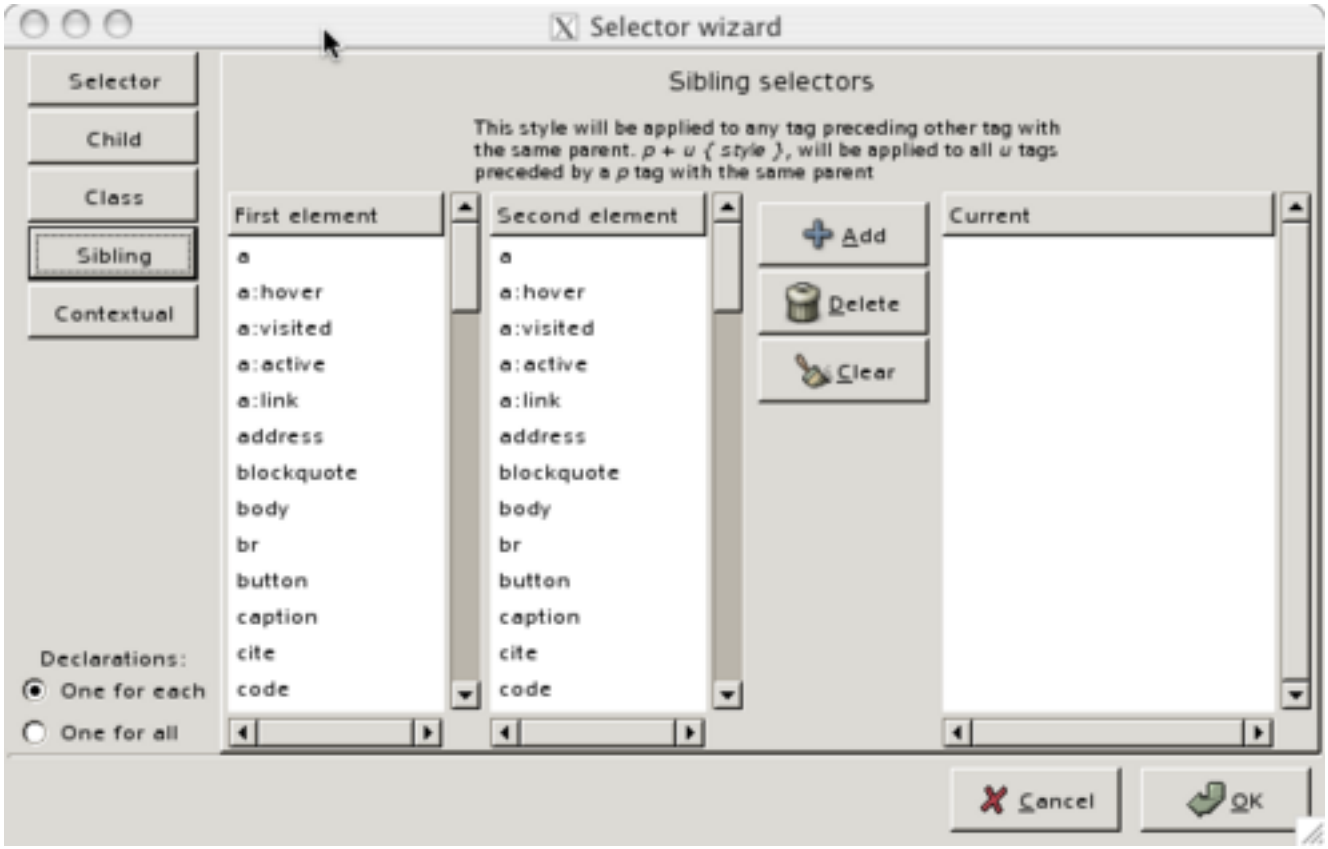


Figure 2-34. Sibling selector wizard

- contextual selectors

Figure 2-35. Contextual selector wizard

- With the color wizard, you can set a value for any property involving color.

Figure 2-36. Color wizard

There are equivalents in the secondary tool bar for those wizards.

Figure 2-37. Equivalents of wizards in the secondary tool bar

- Miscellaneous actions performed on the current document:
  - Validate and dump (version 0.3.1): checks if the document is well-formed and parseable by a web browser, and dumps the analysis into the program output.
  - Validate only (version 0.3.1): checks if the document is well-formed and parseable by a web browser. Errors are listed in the program output tab of the footer panel.
  - Scan selector (version 0.3.1): lists in the upper part of the side panel all properties and values of the selector, your mouse pointer is in. From there you can change/add values and properties.

All items have equivalents in the 4th and 5th parts of the secondary tool bar.

Figure 2-38. Equivalents of validate and scan menu items in the secondary tool bar

The Tools menu

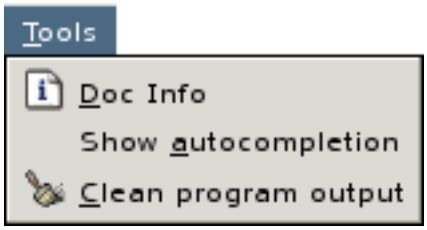


Figure 2-39. Tools menu

This is where you perform actions on the current document. The menu contains the following items:

- Doc info: gives some information about the current document, such as its location, its size in characters, its saved/modified state. The output is listed in the program output tab of the footer panel.
- Show autocompletion: displays the auto completion popup menu. Mostly useful when you have disabled automatic auto completion. The popup menu appears where your mouse pointer is in the current document.
- Clean program output: clears the program output in the footer panel.

All items, but the show autocompletion one have equivalents in the 4th part of the secondary tool bar.

Figure 2-40. Equivalents of tools menu items in the secondary tool bar

The Panels menu

Figure 2-41. Panels menu

From there you can show/hide the side and footer panels.

Observe that the icons which have only a blank background matches the hide action, whereas the icons with a blue rectangle matches the show action. Similarly, horizontal arrows represent the side panel, while vertical arrows represent the footer panel.

All those items have equivalents in the 6th part of the main tool bar

Figure 2-42. Equivalents of the panels menu items in the main tool bar

The Plugins menu

Figure 2-43. Plugins menu

From there, you access a plugins dialog window, where you can see which plugins are installed, and consequently load or unload them.

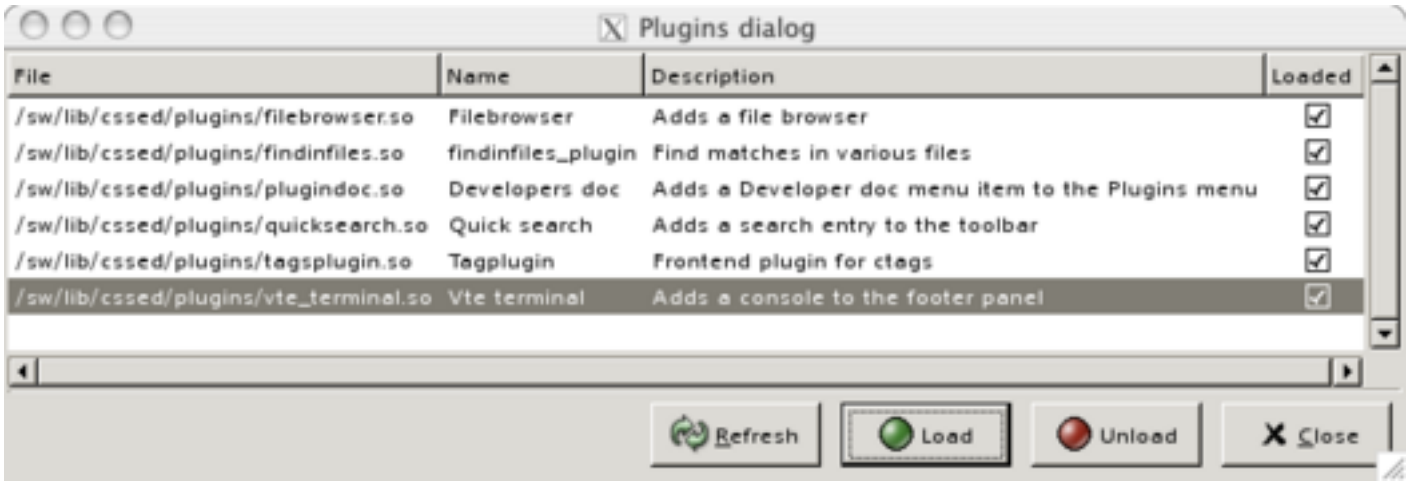


Figure 2-44. Plugins dialog

Some plugins may add new items to the plugin menu. In this case, they will be described in Chapter 4.

There is no equivalent in the tool bars for this menu.

The Help menu

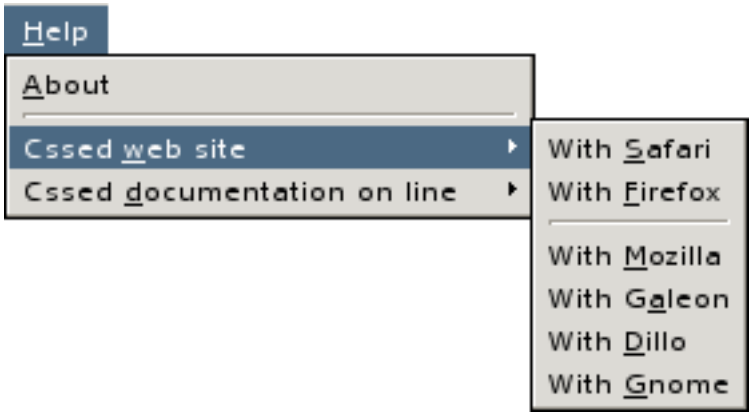


Figure 2-45. Help menu

The About item in this menu leads to a four tabs window with information about cssed: author, developers, translators and their details (in translated version), and a dedicace for this release.

The other items are only available on Mac version at the moment. They give access to the cssed web site and the online documentation. Some common browsers on this system are provided, as well as some others available on the X11 layer.

There is no equivalent in the tool bars for this menu.

Navigation through bookmarks

With those icons you access quickly any temporary bookmark you have previously defined just by clicking in the margin in front of the line you wanted to bookmark.

Figure 2-46. Navigation through bookmarks

Those items are located in the fifth part of the main tool bar. They are not present in the menus.

The editor notebook

The editor notebook stores all documents opened in cssed. It always contains a document, so if you close all documents in cssed a blank one will be shown.

When several documents are open, you can change the current document by clicking on the small tab above the document.

Figure 2-47. Editor notebook

The side panel

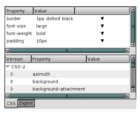


Figure 2-48. Side panel

It is divided in two tabs:

- The CSS tab, where there is specific CSS stuff, divided in two parts:
  - The selector scanner that helps in validating syntax in a single selector, and lets you edit values from the list either manually or by double-clicking on the arrows.

    Figure 2-49. Selector scanner

  - The CSS tree view where all the existing CSS2 properties and values are shown and can be selected for insertion in the document by double-clicking on a value.

    Figure 2-50. CSS tree view

- The Digest tab, which allows you to parse the current document by selectors.

  It offers a convenient way to access quickly a given selector in a big document.

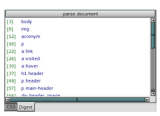
  

  Figure 2-51. Digest

The program output, the scratch and static pads, and the terminal

Those are located in the footer panel.

- The program output is used to show the user information about what is happening in cssed while performing a given action.

  Here you can see the output of a validate and dump action on the current document.

Figure 2-52. Program output

- The scratch pad and static pad are just text buffers to be used as a dragable clipboard from/to the editor notebook or for notes.

  It is divided into a temporary part: the scratch pad, and a permanent one: the static pad. Drag and drop may be performed between both pads.

Figure 2-53. Scratch and static pads

- The terminal (optional) is very handy to perform external operations while editing a document, such as moving files, running a make, or checking the rendering in a browser.

  Here it is a mere ls command which has been executed.

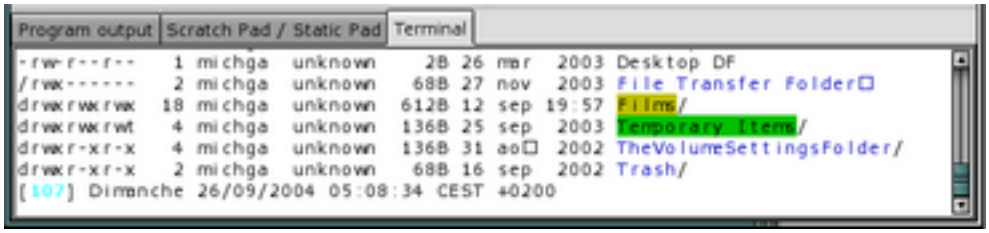


Figure 2-54. Terminal

Chapter 3
Using cssed

I will explain here how to use cssed ... Well ... Really I will explain how I use it. As in any other application there are various ways to perform editing tasks, and mine is not mandatory. I am sure when you will know cssed you will run it the way you think fit, so just read this if you do not know how to work with cssed, or you feel lost when working with it.

Launching cssed

You can launch cssed from:

- the command line with:
  - `cssed`

    This will launch cssed and open a new document in cssed.

  - `cssed foo`

    This will launch cssed and open the file `foo` from within cssed.

- GNOME, by clicking on the cssed icon, providing that you have customized GNOME panel to add a launcher for cssed.
- KDE, by clicking on the cssed icon, provided that you have customized the KDE menu to include cssed in, say, Development submenu, and added cssed to the applications launcher.
- the Nautilus manager, by dragging and dropping a file onto cssed notebook editor, providing that you have at least version 2.2 of GNOME installed.
- Emacs/XEmacs, providing you have installed the **emacs launcher** you will find on the cssed web site (`http://cssed.sourceforge.net/extras.php`).

Configuring cssed

See the Section called *Configuring with the dialog* in Chapter 6.

The editor window

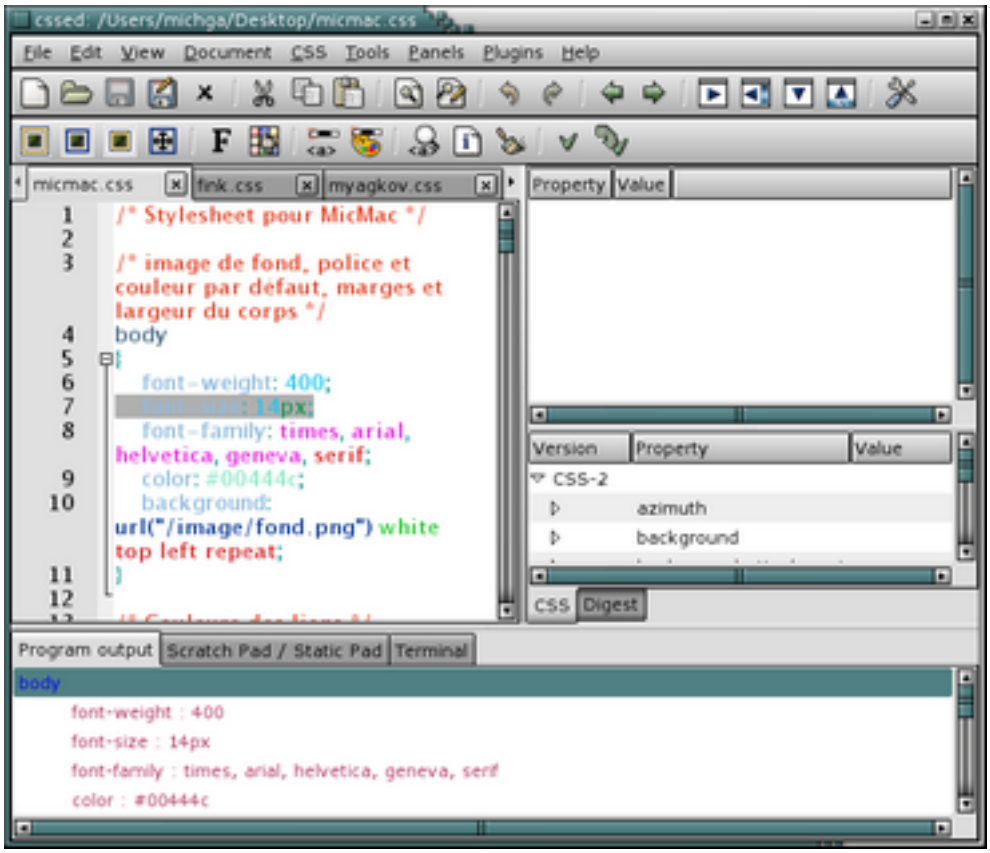


Figure 3-1. Several files open in the editor window

The editor window is the cssed documents container. Any document open in cssed, will own an editor window where you can see, and change document contents.

Each open document is displayed in a tab at the top of cssed editor window. If there are several open documents, and there is no more room for tabs, the new tab will be hidden and two small buttons facing right and left, will appear on the right side of the documents tabs. With those two buttons you can browse through all open documents.

If you get gtk 2.4.9 and above installed, you will notice that the small arrows are no more grouped together. To retrieve a more efficient usage in all applications, you may want to customize gtk by creating or modifying the .gtkrc file in your home directory. Here is how to do this on Mac (the name of the rc file may vary on other system, but the lines to insert are the same):

Getting grouped arrows

1. Open the `.gtkrc-2.0` file in a terminal window with pico:

   ```
   pico .gtkrc-2.0
   ```

2. Put those lines in it:

   ```
   style "default"
   {
    GtkNotebook::has_secondary_forward_stepper = 1
    GtkNotebook::has_secondary_backward_stepper = 1
   }
   class "GtkWidget" style "default"
   ```

A quicker access is provided by the contextual pop-up menu, which opens when you right-click on a tab. It displays all the open tabs in cssed, and you can choose any of them to make it the active document.

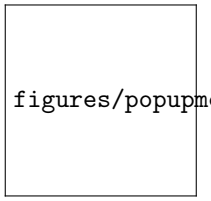


Figure 3-2. The editor window contextual menu

The path to the current document appears in the cssed window title bar, providing that an existing document has been opened or a new document has been saved.

Setting document preferences

Apart from the global settings (discussed in the Section called *Configuring with the dialog* in Chapter 6), each document open in cssed can have its own preferences and behave in a different way. The Document menu shows and lets you set the current document settings. This menu changes each time you switch to another document to reflect its current settings.

Here are the settings you can set:

View line numbers

On activation, the line numbers are shown in the right margin of cssed.

View line endings

On activation, the end of line characters are made visible. It is useful to know what kind of document you are editing Unix (LF), Mac (CR) or DOS/Windows (CR and LF). [1]

View white spaces

On activation, the white space characters are shown in a graphical way.

View lines wrapped

On activation, when the lines are larger than the editor window width, they are wrapped at the right margin and continue at the left margin. In this mode the horizontal scroll bar is never shown.

Enable autocompletion

On enabling, auto completion is enabled for the current document. It allows you to choose quickly properties and values from a popup menu. More on this thema in the Section called *Auto completion*.

Enable folding

On enabling, a **-** appears in the left margin and all properties are unfold. Clicking on it switches the **-** to a +, and fold the given property; you unfold it again by clicking on the +. You can also fold/unfold all properties from the item folding within the Document menu. When you disable folding again, the properties are automatically unfold.

Figure 3-3. Folding all on the current document

1. The acronyms used to name end of lines are:

- CR - Carriage Return
- LF - Line Feed
- CR/LF - Uses both characters Carriage Return and Line Feed

Set EOL mode

This entry converts the end of line characters between the most commonly used types. You can convert a Windows document in a Unix one for example.

Highlighting

This entry allows you to change the syntax highlighting chosen by cssed (based on the file extension) once the file is open.

The syntax highlighting for CSS files is based on what is set in preferences (see the Section called *Configuring with the dialog* in Chapter 6). For other files it is hard-coded at the time being, but will be configurable in a future release).

Here you can see a shell script highlighted:

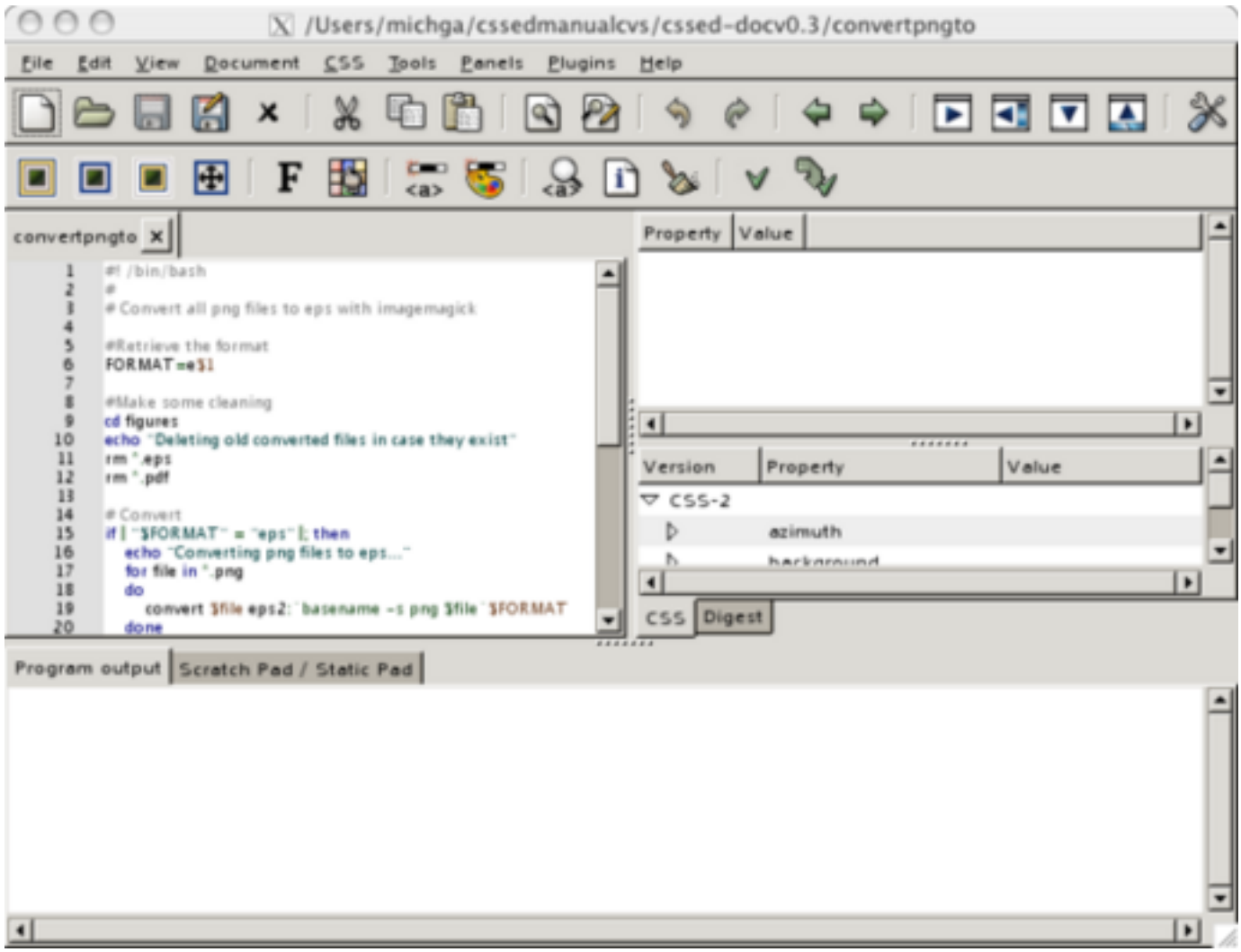


Figure 3-4. Highlighting a shell script

Force encoding

The default encoding for new document is UTF-8. You can force the encoding either for a new document or for an already existing document. See the Section called *Forcing the encoding* for an in-depth discussion.

Character set

This menu is aimed at people who need to deal with Asian languages. Some explanation about its functionalities will be found in the Section called *Setting the character set*.

Change Font

This entry lets you change the font for the current document.

Remember that when you change the document settings, they are only changed for the current document.

You can see below the document settings in action.

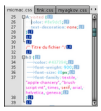


Figure 3-5. Document settings in action

Note line numbers, folding enabled, line endings set to Unix though made on a Mac, lines wrapping, white spaces (... for spaces, arrows for tabulations), and highlighting.

Auto completion

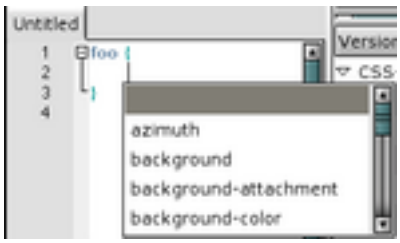
Auto completion is one of the most used features in cssed. It lets you start typing a property name and see all possible completions. Once you have selected a property, you can apply it one of its values from a pop menu.

To use auto completion, you should first activate it in the Document menu (See the Section called *Setting document preferences*).

Usage of autocompletion

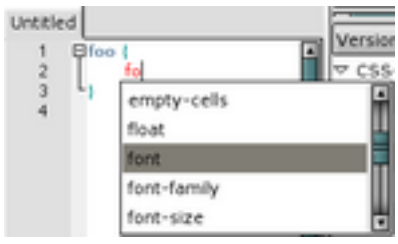
1. Type the declaration name and the opening curly brace.

   As soon as you will type the opening brace, you will be presented with the list of all possible CSS properties.



2. Use one of the following techniques to search for the desired property:

   a. Type the first letters of the property, if you know it. The list will scroll up or down automatically to the properties whose names begin with the given letters.



   b. Use the scrollbar to navigate the list.

3. Select the desired property by:

   a. either double-clicking on it

   b. or clicking on it and press **return**

   The property will be written in the document.

4. Once the property is written in the document, type the sentence separator character **:**, a pop menu with the possible values for this property will be shown. You can then select the desired value the same way you did it for the property with the auto completion list.

A value embedded in parentheses leads to a dialog. Just click on it to access the dialog, fill it in, and click on **OK**.

5. Type a **;** to complete the process.
6. When you have added all the properties you want to the declaration, hit the **Esc** key to hide the autocompletion list and use the combination keys **Ctrl-l** to delete the new line added by cssed. See the explanation in the Section called *Auto completion annoyances*.

If you choose a value by error, just delete it as well as the sentence separator character **:**, and enter the sentence separator character **:** again to trigger the value's pop menu.

To quickly switch auto completion on and off, you can use the accelerator key **Alt-D-a**. With this accelerator key you can use auto completion for a single item, and then toggle it again to off.

Some users do not like the auto completion cycle. To let them use just the auto completion list, the accelerator keys **Alt-T-a** will show this list even if auto completion is disabled for the current document. No trigger key will be further used to show the auto completion list, or auto completion menu if auto completion is disabled in the current document, even if you start the auto completion cycle showing the list with **Alt-T-a**.

Mac OS X users: remember you cannot use accelerator keys.

Auto completion annoyances

The auto completion is not finished yet, and there are some situations where it can be annoying to the user. The auto completion list is shown each time the opening brace is typed. This way when you type a media at-rule - for example - the auto completion list is shown even when a property into this at-rule is bad CSS.

Use the **Esc** key to hide the auto completion list at any time.

The auto completion pop menu works by reading the word from the start of the line to the sentence separator, and looking for a matching string in the CSS tree view (View: the Section called *The tree view*). This way it only will work with CSS formated with a sentence per line and not when you put various sentences in the same line, even when it is valid CSS.

To get auto completion to work with those requirements, cssed always adds an end of line character when you type the sentence end character **;**.

This way - when using auto completion - you always must delete the last line added by cssed.

To do it quickly use the accelerator keys **Ctrl-l**.

You can also use **Alt-;** instead of **;** to avoid to continue the auto completion cycle.

Selecting blocks of text

In cssed, there are multiple ways of selecting blocks of text.

Kind of blocks

Any chunk of text

Push the left mouse button, select the block with the mouse, and release the button.

A whole line

Move the mouse pointer to the beginning of the line (the cursor changes to a pointer) and click.

Or:

Click on a line number, provided that you have activated the line numbers (See the Section called *Setting document preferences*).

Multiple lines

Put the cursor at the start of the text to be selected, push the **Shift** key and click on the line number you want to be the end of block.

The whole text

Use **Ctrl-A**.

Or:

Push **Ctrl** and click on any number in the margin, provided that you have activated the line numbers (See the Section called *Setting document preferences*).

Indenting blocks of text

In cssed the behavior of the tab key when indenting a selected text block differs if there is one line of text selected, or if the block of text selected contains multiple lines:

- When one line is selected, the **Tab** key deletes this line and adds a tab character.

  To indent a single line, move the mouse pointer before it and then use the **Tab** key. It takes the end of line character into account.

- When multiple lines are selected, the **Tab** key indents the whole block by one tab.

  To indent multiple lines, you can just select the last character of the first line to be indented up to the first character of the last line to be indented.

If you want to indent a block of code, be sure it contains at least an end of line character with one of the above tips.

Forcing the encoding

- Default behaviour: as already stated, the default file encoding for any new document is UTF-8. When opening a file, cssed tries to recognize the UTF-8 encoding and open the file with that encoding if it is valid UTF-8. On the contrary, it will open the file with the ISO-xxxx encoding, which best matches.
- Forcing: if you want to use another encoding, either with a new file or when opening an already existing file, force the encoding just before writing any text

  If you want to write JIS or Hangul, you must force the encoding to DBCS on each document you create or open.

You may want to check the encoding of the file `foo` with:

```
file foo.
```

Take into acount that **file** always reports ASCII encoding when there is no accented character in the file.

Do not mix encodings in a given file, you will end up with bad rendering.

Setting the character set

This is only needed for Asian languages. The supported character sets are the following:

- ANSI: American standard, generally used with ASCII encoding
- Western Europe: accented characters, generally used with ISO-xxxx or UTF-8 encodings
- GB2312: used with DBCS and UTF-8 encodings
- Hangul: Native Korean, used with DBCS and UTF-8 encodings
- Shift JIS: Japanese, used with DBCS and UTF-8 encodings

ANSI and Western Europe are recognized when typing characters.

To use input methods in cssed, you will need to proceed as follows:

Using input method in cssed

1. Set the font to the Asian font of your choice
2. Set the encoding to DBCS or UTF-8
3. Set the character set to GB2312, Hangul, or Shift JIS

The tree view

The CSS tree view is located in the side panel (See the Section called *The side panel* in Chapter 2) and allows you to see all properties and values in the current CSS definition file (See the Section called *CSS definition file* in Chapter 6).

Using the tree view

Navigating through the properties

Use the scroll bar on the right side to browse the tree.

Displaying the values

Click on the arrow in the Version column to unfold the values of a given property. Click again to fold them.

Adding properties to the document

Double-click on the desired value of any property. This will add the property/values pair, following by an end of line character. This way you can quickly insert various properties without typing the end of line character yourself.

A value embedded in parentheses leads to a dialog. Just double-click on it to access the dialog, fill it in, and click on **OK**.

The selector scanner

The selector scanner is not finished yet, but is now a working feature.

To use the selector scanner, you should follow these steps:

1. First place the caret [2] between the opening and closing braces of a selector or identity definition.
2. Select the scan selector tool in the menu Tools⟶Scan selector (**Alt-T-s**).

   The current declaration is scanned and shown in the list at the top of the side panel (See the Section called *The side panel* in Chapter 2).
3. Edit the values by either:

   a. Clicking twice the Value column to edit directly the value.

   b. Double-clicking the down triangle facing icon in the third column to choose the value from a pop up menu.

It is planned that this list will be a drop target for the CSS tree view to quickly insert or replace new property/value pairs.

The digest

The digest, located in the side panel too (see the Section called *The side panel* in Chapter 2) is very handy when working with huge documents.

Using the digest

1. Just click on the digest tab to access it.
2. Click on the parse document title to get the document folded by declaration.
3. Double-click on the desired line in the digest panel to get access to it in the document.

   You will see the document scrolling up/down to the first line of the declaration, which will be marked with a chevron in the margin.

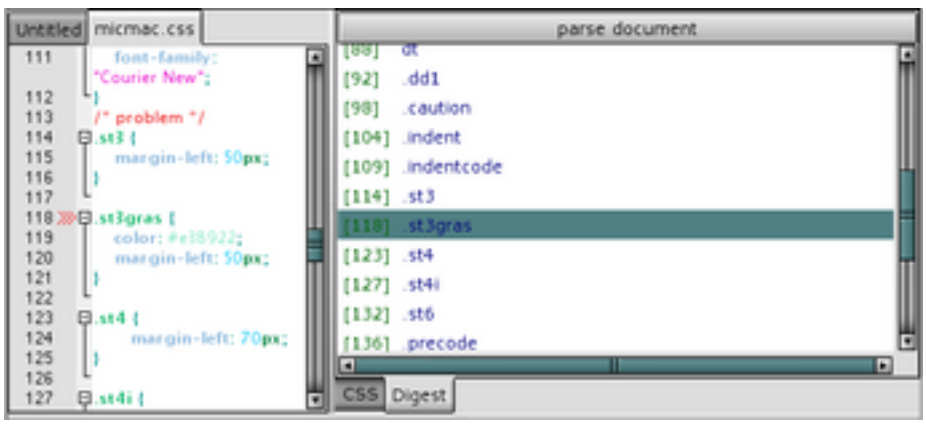


Figure 3-6. Accessing a declaration from the digest

The scratch and static pads

The scratch and static pads (see the Section called *The program output, the scratch and static pads, and the terminal* in Chapter 2 are one of those silly features that you can love or hate. Those are nothing but text buffers that can be used as a drag & drop target/source to the editor windows, as a second clipboard, or simply for notes.

It was added to cssed because some years ago I used an application with a similar scratch pad. I started to work with this application without using it at all. As times went on, I began to use it to do a lot of things: as a second editable clipboard, to put notes about what I was doing while editing multiple documents, and even to copy/edit and paste a buffer in various documents.

2. The caret is the blinking line that shows you the location of the text insertion point.

The clipboard do not let us easily edit its contents, so I finally get in love with the scratch pad. It is the easy explanation about why cssed has a "scratch pad".

Notice that the pads are divided into two parts. As their names suggest, the scratch pad is temporary, while the static pad is permanent.

Use the scratch pad while editing files, then store in static pad what you want to have at hand permanently. You can drag and drop between the pads.

The program output window

To avoid excessive use of warning dialog boxes, cssed uses a list at the application's footer to show relevant but not critical output (see the Section called *The program output, the scratch and static pads, and the terminal* in Chapter 2). It is used by the CSS dialogs to warn about problems or default values in the dialog boxes.

It is also used by the validator to show its output.

You can clear this output window with the menu item Tools⟶Clean program output.

The CSS menu

From this menu you can access all dialogs - just the dialogs - used in the tree view or auto completion menu, to ease the insertion of complex CSS values.

From this menu you can also access the *Selector Wizard*, the *Color Wizard*, and the *Box Properties Wizard*.

The selector Wizard

The selector wizard lets you add one or various empty selectors at one time. This way you can start quickly a new style sheet with the most common selectors. It only applies to style sheets used with HTML documents.

There are different kinds of selectors you can add from this dialog. Those selectors' types can be selected with the buttons at the upper left corner of the dialog. You can only add a type of selectors at a time.

When you select a selector set - a group of selectors in the rightmost list of this dialog - you can choose between create a declaration for each, or use the same declaration for all by checking the radio buttons at the lower right corner of the dialog.

The types of selectors that can be found are:

- Single selector - selector button: it adds a single selector. The style will be applied to any tag matching the selector tag.
- Child selectors: here you can set the style of any selector with a matching parent. The parent selector will be chosen from the first list, the child selector from the second. For example: $p > b$ - select p in the first list and b in the second - will apply the style to any b tag with a p tag parent.
- Class selectors: it sets the style of any selector identified by the *class* attribute. For example, *p.test* will match any p tag with *class="test"* attribute.

```
<p class="test">this  will be shown with the p.test style</p>
```

- Sibling selectors: with sibling selectors you can apply style to one tag preceded by other tag in the same parent. If you want to apply style - for example - to any p tag preceded by a *h1* tag, you must choose in the Sibling pane, h1 in the first list and p in the second list. This will produce the selector *h1 + p*.

```
<div><h1>header</h1><p>this  will be shown with the h1 + p style</p>
```

- Contextual selectors: contextual selectors, let you apply a style to any tag with a specific ancestor. For example, selecting p in the first list and b in the second, will apply style to all b tags within a p tag.

The syntax validator

You can validate the syntax of your CSS document from the menu Tools⟶Validate and dump or Tools⟶Validate only.

The validator uses a subset of the libcroco library modified to get its output in cssed.

Libcroco is a parsing and layout engine used to parse and show CSS styled output. Since it works exactly as a user agent - parsing until fatal error and avoiding some invalid values - we can know how our style sheet will be read from a CSS2 compliant UA.

In the menu Tools⟶Validate and dump you can check this behavior. It will show in the output window what it is reading and when it stops parsing.

The menu Tools⟶Validate only will only show the validation results.

This validator is unfinished, and some strange behavior may occur.

Especially if you use empty declarations, the validator will fail even if it must be ignored as it is valid CSS.

Chapter 4
Plugins

To be written.

Chapter 5
Accelerator keys

Under this generic name, we will present you the various ways to perform quickly a given action in the current version of cssed.

There are editor window accelerator keys (EWA keys) and menu accelerator keys (MA keys), as well as icons accessibles from the tool bars and some generic combination of keys. The later are listed at the bottom of the table.

Note that the menu accelerator keys described below match only the English version.

Mac OS X users could not use the menu accelerator keys.

Table 5-1. Shortcuts

Action	EWA keys	MA keys	Icons
Open the File menu		**Alt-F**	
Create a new file	**Ctrl-N**	**Alt-F-N**	
Open a file	**Ctrl-O**	**Alt-F-O**	
Close the current file	**Ctrl-W**	**Alt-F-C**	
Close all files			
Revert to the last saved state		**Alt-F-R**	
Save the current file	**Ctrl- S**	**Alt-F-S**	
Save the current file as		**Alt-F-A**	
Save all files			
Quit cssed	**Ctrl-Q**	**Alt-F-Q**	
Open the Edit menu		**Alt-E**	
Undo the last action	**Ctrl-Z**	**Alt-E-U**	
Redo the last undone action		**Alt-E-R**	
Cut the selection	**Ctrl-X**	**Alt-E-t**	
Copy the selection	**Ctrl-C**	**Alt-E-C**	
Paste the selection	**Ctrl-V**	**Alt-E-P**	
Delete the selection	**Backspace/suppr**	**Alt-E-D**	
Find	**Ctrl-F**	**Alt-E-F**	

Action	EWA keys	MA keys	Icons
Find and replace	**Ctrl-R**	**Alt-E-R**	
Preferences		**Alt-E-P**	
Open View menu		**Alt-V**	
Open Toolbar submenu		**Alt-V-T**	
Open Document list submenu		**Alt-V-l**	
Create a document list from opened documents		**Alt-V-l-O**	
Open a document list		**Alt-V-l-f**	
Create an empty document list		**Alt-V-l-B**	
Zoom in the current document		**Alt-V-I**	
Zoom out the current document		**Alt-V-O**	
Normal size on the current document		**Alt-V-N**	
Open the Document menu		**Alt-D**	
Show/hide line numbers		**Alt-D-n**	
Show/hide line endings		**Alt-D-e**	
Show/hide white spaces		**Alt-D-s**	
Wrap/unwrap lines		**Alt-D-w**	
Enable/disable auto completion		**Alt-D- a**	
Enable/disable folding		**Alt-D-f**	
Open the set EOL mode submenu		**Alt-D-m**	
Set Mac EOL mode		**Alt-D-m-R**	
Set Unix EOL mode		**Alt-D-m-F**	
Set DOS EOL mode		**Alt-D-m-C**	
Open the Convert EOLs submenu		**Alt-D-m-O**	
Convert to Mac EOL mode		**Alt-D-m-O-R**	
Convert to Unix EOL mode		**Alt-D-m-O-F**	
Convert to DOS EOL mode		**Alt-D-m-O-C**	
Open the Folding submenu		**Alt-D-d**	
Fold all properties		**Alt-D-d-F**	
Unfold all properties		**Alt-D-d-U**	
Open the Highlighting submenu		**Alt-D-H**	
Highlight plain text		**Alt-D-H-P**	
Highlight CSS		**Alt-D-H-C**	
Highlight HTML/PHP		**Alt-D-H-H**	
Highlight shell scripts		**Alt-D-H-S**	
Highlight C/C++		**Alt-D-H-P**	
Highlight Apache configuration files		**Alt-D-H-A**	
Highlight PERL scripts		**Alt-D-H-L**	
Highlight Python scripts		**Alt-D-H-y**	
Highlight XML		**Alt-D-H-X**	
Highlight diff/patch		**Alt-D-H-D**	
Highlight Makefile		**Alt-D-H-M**	
Change the font of the current document		**Alt-D-C**	
Open the CSS menu		**Alt-C**	
Open the Tools men		**Alt-T**	
Scan the current selector		**Alt-T-S**	

Action	EWA keys	MA keys	Icons
Give information about the document		**Alt-T-D**	
Show auto completion		**Alt-T- a / Alt-A**	
Clean the program output		**Alt-T-C**	
Validate and dump the current document		**Alt-T-V**	
Validate the current document		**Alt-T-o**	
Open the Panels menu		**Alt-P**	
Show the footer panel		**Alt-P-f**	
Hide the footer panel		**Alt-P-p**	
Show the side panel		**Alt-P-S**	
Hide the side panel		**Alt-P-H**	
Show the Help menu		**Alt-H**	
Show the About box		**Alt-H-A**	
Delete the current line	**Ctrl-L**		
Duplicate the current line	**Ctrl-D**		
Select all text	**Ctrl-A**		
Select the current line		**Alt-I**	

Chapter 6
Configuration

To configure css default document's settings, you can used the configuration dialog in cssed tool bar (the last button on it), or configure it by hand.

Configuring with the dialog

The configuration dialog have three tabs to let you configure cssed.

- The document settings tab: let you configure what settings will have a document when opened.
- The syntax highlighting tab: let you configure the colors of the colored CSS in the document.
- The font tab: let you change the default text editor font face and size. The styles will be ignored as they are used by the syntax highlighting,

The document settings tab

Here you can enter the settings that will have any opened document by default. To change a setting, just check or uncheck the desired setting, then validate with the OK button.

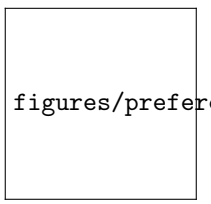


Figure 6-1. The Document settings tab

- View line numbers: when checked, new documents will have the line numbers margin expanded.
- View line endings: when checked, the End Of Line character/s will be displayed.
- View white spaces: when checked, the white space characters (tab, space,...) will be displayed.
- Wrap long lines: when checked, lines that reach the right margin of the editor window will be wrapped.
- Enable autocompletion: when checked, auto completion will be activated by default.
- Enable folding: when checked, the folding feature of the editor window will be activated.

The highlighting tab

This tab let you choose your preferred color scheme for syntax highlighting.

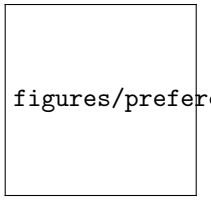


Figure 6-2. The highlighting tab

To change the foreground or background colors of any style, click on the color well at the right side of each token identifier. You can also choose to have the token's face in bold style, checking a box at the row end. Once you have finished, do not forget to validate with the OK button.

The tokens you can choose are:

- *Default* - it is the default style color
- *Tag* - style of tags (selectors) as p, u, etc.
- *Pseudoclass* - style of valid pseudo classes (:hover, etc.)
- *Unknown Pseudoclass* - style of invalid pseudo classes.
- *Operator* - style of CSS operators as braces, coma, etc.
- *Valid property* - a property known by cssed. The property is the left part of the declaration line.
- *Invalid property* - a property unknown to cssed.
- *Value* - a value known by cssed. The value is the right part of the declaration line.
- *Unknown Value* - a value unknown to cssed.
- *Important* - the CSS important keyword.
- *Directive* - a CSS directive as @media, @page.
- *Double quoted string* - a string enclosed between double quotes.
- *Single quoted string* - a string enclosed between single quotes.
- *Class* - a CSS class (selector) as p.style. It will match the tag's class="" attribute.
- *Id* - a CSS id as #123. It will match the tag's id="" attribute.
- *Function* - a CSS function as url(), rgb(), attr(), ...
- *Number* - an integer or floating point number.
- *Valid unit* - a unit recognized by the CSS2 standard (pt,px,em, ...).
- *Color* - a named color (blue, red, green).
- *Hexadecimal color* - a color in hexadecimal notation (0x000000, 0x000).
- *Attribute match* - an attribute matching expression (as img[width=200]).
- *Language* - a language matching expression (as img:lang(en)).
- *Comment* - a CSS comment /* */.

The font tab

This tab let you choose the editor font family and size

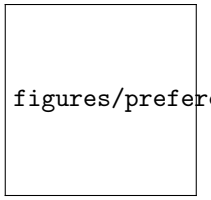


Figure 6-3. The font tab

In this tab the style choice is ignored as it is used for the syntax highlighting. To change the font, just select the font family and the size and click the Apply or OKbuttons.

The current font is indicated on the left side at the bottom of the dialog.

The configuration file

To configure cssed by hand, open the file `$HOME/.cssed/cssed-cfg.xml` and edit it in your preferred text editor.

Here is an example of a customized configuration file:

```
<cssed-conf>
   <document>
   <linenumbers show="true" />
   <autocompletion enabled="true" />
   <lineendings show="false" />
   <whitespaces show="false" />
   <folding enabled="true" />
   <lineswraped show="true" />
    <font name="!Lucida Grande" size="9" />
  </document>
   <window x="569" y="45" width="727" height="532">
    <notebook height="302" />
  <sidebar width="240" />
  <treeview height="111" />
 </window>
 <highlighting>
  <style name="DEFAULT" fore="0x993300" back="0xffffff" bold="false" />
   <style name="TAG" fore="0xa6896a" back="0xffffff" bold="true" />
   <style name="PSEUDOCLASS" fore="0xa0a0d9" back="0xffffff" bold="false" />
   <style name="UNKNOWN_PSEUDOCLASS" fore="0xe414eb" back="0xffffff" bold="false" />
   <style name="OPERATOR" fore="0xa8ad39" back="0xffffff" bold="true" />
   <style name="IDENTIFIER" fore="0xdfbc91" back="0xffffff" bold="true" />
   <style name="UNKNOWN_IDENTIFIER" fore="0x00ff" back="0xffffff" bold="false" />
   <style name="VALUE" fore="0xe11fea" back="0xffffff" bold="false" />
   <style name="COMMENT" fore="0xb0beb" back="0xffffff" bold="false" />
   <style name="IMPORTANT" fore="0xe441d1" back="0xffffff" bold="true" />
   <style name="DIRECTIVE" fore="0xb18bc8" back="0xffffff" bold="false" />
   <style name="DOUBLESTRING" fore="0xcf17e9" back="0xffffff" bold="false" />
   <style name="SINGLESTRING" fore="0x3a3ad4" back="0xffffff" bold="false" />
   <style name="CLASS" fore="0x76bd22" back="0xffffff" bold="true" />
   <style name="ID" fore="0xe61b72" back="0xffffff" bold="true" />
   <style name="VALID_VALUE" fore="0x2525d9" back="0xffffff" bold="false" />
   <style name="FUNCTION" fore="0x993300" back="0xffffff" bold="false" />
   <style name="NUMBER" fore="0xfaca13" back="0xffffff" bold="false" />
   <style name="UNIT" fore="0x569120" back="0xffffff" bold="true" />
   <style name="COLOR" fore="0x21c917" back="0xffffff" bold="false" />
   <style name="HEXACOLOR" fore="0xc7e989" back="0xffffff" bold="false" />
   <style name="ATTR_MATCH" fore="0x9f9fe6" back="0xffffff" bold="false" />
   <style name="LANGUAGE" fore="0x67eedf" back="0xffffff" bold="false" />
 </highlighting>
</cssed-conf>
```

You can see that the document settings have a value of *true* or *false*, the name of the font is preceded by an exclamation mark, and the foreground and background colors are described as hexadecimal values.

You can also change the size of the various elements of the window in this file.

CSS definition file

The CSS definition file is the core of cssed. It is installed in the `prefix/share/cssed/data` folder and named `cssed-def.xml`, together with the default configuration file named `cssed-cfg.xml`, and the dtd file named `cssed-def.dtd`. It has all valid CSS properties and values.

All properties and values in cssed will be taken from this file. It is used by the tree view (View: the Section called *The tree view* in Chapter 3), and by auto completion (View: the Section called *Auto completion* in Chapter 3).

Its format is really easy XML. It has three tags only:

- <*cssed-def*></*cssed-def*> - It is the root node and can contain one or more <*element*> tags.
- <*element*> </*element*> - It defines a CSS property as "border" or "margin" it can contain one or more <*value*> tags.
- <*value* /> - It defines one of the possible values for the element and it is an empty tag (it cannot contain any other tags).

The *cssed-def* element has a property *implementation* that will be shown as the root node in the tree view (View: the Section called *The tree view* in Chapter 3). Valid examples are:

```
<csssed-def implementation="CSS-2"></cssed-def>
<csssed-def implementation="My CSS"></cssed-def>
```

The element node has an attribute name that must contain the property's name. Valid examples are:

```
<element name="azimuth"></element><element name="azimuth"></element>
```

The *value* node has two attributes. The first named *type* is right now unused but it is mandatory to set it to the values *string* or *fixed-string* for compatibility with further versions of cssed. The second one, *name*, must contain the value itself. If type is equal to *string*, it means that is a complex - not fixed - value and must be dialog driven. The dialog name must be enclosed between braces. If type is equal to *fixed-string*, it means that the string in the name attribute is a possible value for the property and will be inserted directly. Valid examples are:

```
<value type="string" name="(angle)" />
<value type="fixed-string" name="none" />
<value type="fixed-string" name="red" />
```

In the first example a dialog named "*(angle)*" will be used to drive property's value insertion. In the second and third ones, the values *none* and *red* will be directly inserted as valid property's value.

A complete example of cssed configuration file can be:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csssed-def implementation="CSS-2">
<element name="azimuth">
<value type="string" name="(angle)" />
<value type="fixed-string" name="none" />
</element>
</cssed-def>
```

DTD

The Document Type Definition of the cssed-def.xml file is the following:

```
<!ELEMENT csssed-def ( element+ ) >
<!ATTLIST csssed-def implementation NMTOKEN #REQUIRED >

<!ELEMENT element ( value+ ) >
<!ATTLIST element name ID #REQUIRED >
```

```
<!ELEMENT value EMPTY >
<!ATTLIST value name CDATA #REQUIRED >
<!ATTLIST value type ( fixed-string | string ) #REQUIRED >
```

Built in dialogs

There are some dialogs that can be used right now in the css definition file. Below you will find a list of them. The dialog name is between parenthesis - as it must be in the definition file itself - the *"Url"* points to the page of the CSS2 specification that defines the dialog's value, and the *"Used by"* field lists the properties that use this dialog.

- **(angle)**: insert an angle.

  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-angle

  Used by: *azimuth*.

- **(attr)**: insert the content of one tag's attribute.

  Url: http://www.w3.org/TR/CSS2/generate.html#propdef-content

  Used by: *content*.

- **(background)**: let you select various values to a *background* property.

  Url: http://www.w3.org/TR/CSS2/colors.html#propdef-background

  Used by: *background*.

- **(border)**: add value to a *border* property. Set the value for all borders.

  Url: http://www.w3.org/TR/CSS2/box.html#propdef-border

  Used by: *border*.

- **(border-width)**: add the width to one border.

  Url: http://www.w3.org/TR/CSS2/box.html#propdef-border

  Used by: *border-top-width*, *border-right-width*, *border-bottom-width*, and *border-left-width*.

- **(border-width-style-color)**: set the width style and color of one border.

  Url: http://www.w3.org/TR/CSS2/box.html#propdef-border

  Used by: *border-top*, *border-right*, *border-bottom*, *border-left*.

- **(increment-reset)**: set/reset the increment of a *counter* property.

  Url: http://www.w3.org/TR/CSS2/generate.html#counters

  Used by: *counter-increment*, *counter-reset*.

- **(counter)**: set a counter to a *content* property.

  Url: http://www.w3.org/TR/CSS2/generate.html#counters

  Used by: *content*.

- **(string)**: add a string as value for a property.

  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-string

Used by: *content*, *quotes*, and *text-align*.

- **(uri)**: set a url as value for a property with the *url()* function.
  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-uri
  Used by: *background-image*, *image*, *content*, *cue*, *cue-after*, *cue-before*, *cursor*, and *list-style-image*.

- **(font)**: fill in the values of one border property.
  Url: http://www.w3.org/TR/CSS2/box.html#border-properties
  Used by: *border*.

- **(font-family)**: set the font's family.
  Url: http://www.w3.org/TR/CSS2/fonts.html#propdef-font-family
  Used by: *font-family*.

- **(font-size)**: set the size of a font.
  Url: http://www.w3.org/TR/CSS2/fonts.html#propdef-font-size
  Used by: *font-size*.

- **(identifier)**: set an identifier as value for a property.
  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-identifier
  Used by: *page*.

- **(integer)**: add an integer number value to a property.
  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-integer
  Used by: *windows*, *z-index*, *font-size-adjust*, and *orphans*.

- **(length)**: set a length as value for a property.
  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-length
  Used by: *marker-offset*, and *word-spacing*.

- **(list-style)**: set the style of the markers in a list. It can be numerical, graphical, or alphabetical.
  Url: http://www.w3.org/TR/CSS2/generate.html#propdef-list-style
  Used by: *list-style*.

- **(margin)**: set all margin values (top, left, bottom and right).
  Url: http://www.w3.org/TR/CSS2/box.html#propdef-margin
  Used by: *margin*.

- **(margin-width)**: set the width of one margin.
  Url: http://www.w3.org/TR/CSS2/box.html#value-def-margin-width
  Used by: *margin-top*, *margin-right*, *margin-bottom*, and *margin-left*.

- **(number)**: set a real number as value for a property. It is used by aural properties.

  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-number

  Used by: *pitch-range*, *richness*, *speech-rate*, *stress*, and *volume*.

- **(padding)**: set all paddings in a box element.

  Url: http://www.w3.org/TR/CSS2/box.html#propdef-padding

  Used by: *padding*.

- **(pause)**: set a pause in an aural property.

  Url: http://www.w3.org/TR/CSS2/aural.html#propdef-pause

  Used by: *pause*.

- **(percentage)**: set a percentage as value for a property.

  Url: http://www.w3.org/TR/CSS2/syndata.html#value-def-percentage

  Used by: *pause*, and *volume*.

- **(percentage-length)**: it is used by properties that can take as value one percentage or a numerical length.

  Url: http://www.w3.org/TR/CSS2/syndata.html#values

  Used by: *right*, *text-indent*, *top*, *width*, *vertical-align*, *background-position*, *border*, *bottom*, *height*, *max-height*, *max-width*, *min-height*, *min-width*, *padding-top*, *padding-right*, *padding-bottom*, and *padding-left*.

- **(play-during)**: set a sound to be played and the repeat or mixing properties of this sound, in aural style sheets.

  Url: http://www.w3.org/TR/CSS2/aural.html#propdef-play-during

  Used by: *play-during*.

- **(quotes)**: set what characters will be used as quotes. Commonly used to set the quotes based in the language being used. It works with the *content* property when it sets its values to *open-quote* or *close-quote*.

  Url: http://www.w3.org/TR/CSS2/generate.html#propdef-quotes

  Used by: *quotes*.

- **(shape)**: set the rectangle of a *clip* property, with the *rect()* function.

  Url: http://www.w3.org/TR/CSS2/visufx.html#value-def-shape

  Used by: *clip*.

- **(size-length)**: it is used by properties that can get as value a size number, or a length identified by a number and the unit used to define the length.

  Url: http://www.w3.org/TR/CSS2/syndata.html#values

  Used by: *size*.

- **(text-shadow)**: set the x and y offset and blur of a shadow in the text.

  Url: http://www.w3.org/TR/CSS2/text.html#propdef-text-shadow

  Used by: *text-shadow*.

- **(voice-family)**: add a voice set, and generic voice family to aural style sheets.

  Url: http://www.w3.org/TR/CSS2/aural.html#propdef-voice-family

  Used by: *voice-family*.

Chapter 7
Development

State

cssed is in Alpha state now.

Some features have already been implemented, others are planned.

Additional features may be added via the plugable interface.

Implemented features

- CSS syntax validation.
- CSS, plain text, HTML/PHP, sh, C/CPP, Apache/.htaccess, Perl, Python, xml, Diff/patch, Makefile syntax highlighting.
- Forcing encoding: UTF8, DBCS, ISO-xxx.
- Switching character sets.
- Search and replace.
- XML based CSS definition file (easy to customize).
- Auto completion.
- Dialog driven CSS property insertion.
- More than 30 dialogs for complex (non static) CSS properties and values.
- Full CSS2 support.
- Show or hide line numbers, white spaces or line endings.
- Line ending conversion between Windows, Mac and Unix end of line characters.
- Markers used as bookmarks for easy navigation through a document.
- MDI interface (notebook based).
- CSS validation.
- CSS selector, id, class and complete style wizards.
- Color wizard.
- Box wizard.
- XML based configuration file.
- Digest.
- Info on document.
- Zooming.
- Folding/unfolding.
- See also Chapter 4 for additional features.

Planned features

- Preview in browser.

- Preview window to check styles.
- Create style sheets from HTML.

Getting involved

If you want to help, request it on the source forge's project page, or via the mailing list cssed-devel@lists.sourceforge.net

The list web interface is at http://lists.sourceforge.net/lists/listinfo/cssed-devel

You can also send a mail with the word *help* in message's body to *cssed-dev-request@lists.sourceforge.net*. If your mail user agent or web interface add a signature to your mail, use the words *help* and in the following line type *end*.

You can get more help about the use of the mailing list interface in the mailman page at http://www.list.org/mailman-member/index.html