

Html2Slideshow

[Howto](#) [The Gui in general](#) [Language-setting](#)
[The main dialog](#) [Shortcuts](#) [Example](#)

Howto

For an explanation of what Html2Slideshow is, please read the text of the Html2Slideshow-Homepage on <http://www.uplawski.de/html2slideshow.html>. However, the referenced page is only available in German, at the time of this writing (end of March 2009).

In the document, you are currently reading, I describe, how the graphical [Shoes](#)-surface for the Html2Slideshow-program is used. If you want to use Html2Slideshow on the command-line of a Unix-shell or in a DOS-window, instead, the argument -h or --help will print the following usage-information:

```
Usage: ruby html2slideshow.rb [options]
       or html2slideshow [options]
```

Specific options:

-d, --debug [true FALSE]	Switch on debug-mode. Will generate log-messages during processing
-s, --source SOURCE DIR	Read source-files from this directory (html- or image-files)
-t, --target [TARGET DIR]	Write slideshow-files to this sub-directory (if not provided, SOURCE DIR is used)

Common options:

-h, --help	Show this message
--version	Show version

The GUI in general



You need Shoes. Without Shoes, you will not be able to use HTML2Slideshow over the graphical user interface and have to stick with the command-line. Get the graphics toolkit and all the background-information you may need, from the Shoes-homepage: <http://shoooes.net/>

The generator and its GUI exist, thus, well separately from each other to allow different uses of the program. While `html2slideshow.rb` is the main Ruby-script, that can be used on the command-line, `html2slideshowDialog.rb` contains the definition of the Shoes-dialog, explained below.

I believe, that you know already how to start a Shoes-application, but just for completeness, do it like this:

```
$>shoes html2slideshowDialog.rb
```

So far, two options are not accessible to the console-program:

- The command-line version uses a hard-coded list of JPEG-extensions: **.jpg*, **.jpeg* and **.jp2*. Thus, your slideshow can only contain JPEGs or Jpeg2000-files, respectively. In the GUI, you can choose some more file-formats to include in your slideshows.
- The log-file, too, is hardcoded as [path to the html2slideshow-folder]/html2slideshow.log, while, in the GUI, you may name any other file, to log messages to.

Both features are explained further below.

The user-interface consists of a selection of dialogs, but for the most general task of creating slideshow-files from original (X)HTML-sources, the first or *main* dialog offers all the functionality you need.

Language-setting



I have included an easy way to adapt the GUI to languages other than English. The current release contains reliable translations to German and French. I will explain first, how you enable these two language-versions, then how you can enable the program to use additional languages.

Switching languages

Usually, you want the messages, captions and other text which appears on the surface, to be written in the language, that your operating system is already configured to use. Unfortunately, this setting is found at different places, depending on your current operating system. Under **Linux** Html2Slideshow is able to find the system-wide setting in the environment-variable LANG and will use it by default. So, if you are using Linux and want the program to speak to you in either German or French, no changes should be necessary.

Should you want to actively change the language of Html2Slideshow and know, that the program supports the new language, there is yet another, easy way to enforce that change. Go to the folder, where Html2Slideshow resides and create a new text-file LANG (without file-extension), if it does not accidentally exist already. The contents of the file is one single line with the two-letter country-code for the new setting. Initially, apart from 'en', only 'fr' or 'de' make sense. But you are free to write the code in capital letters, too.

Adding languages

All translations reside in one text-file, that you find in the program-folder. The file is named **"translations"**. Here is an exemplary section of its

contents.

```
Success:
  de: Erledigt
  fr: Finis

XX Options:
  de: XX Optionen
  fr: Options XX

current log-file shall be XX:
  de: Aktuelles Logfile XX
  fr: Fichier log sera XX

Image types:
  de: Bildformate
  fr: Formats graphiques

selected types are XX:
  de: ausgewählte Formate sind XX
  fr: Formats choisis sont XX
```

Each block starts with the English version of a text to display, followed by a colon. Each translation of this text starts in a new line and is prepended with the two-letter country-code and a colon.

xx is a wildcard, which allows a text to be parameterized by the program. The position of this wildcard may vary in a translation but it should stay intact.

When you add your own translation or modify the existing text, please keep it free of special characters, especially colons (other than those at the end of the english original or the language-codes), because these control the program-behavior and your text would be clipped at their position.

Special characters in the translation

Lower down in the translations-file, you find two sections which need further explanation.

The first contains text, which will be dynamically set or changed on the generated slideshow-pages. The *JavaScript*-routines behind this functionality cannot handle any special characters, like e.g. French accents (like in é). Instead, in this section, the escape-sequence of each such character must be used. The JavaScript-function `unescape()` will see to making them palatable to the browser.

The very last paragraph in the file contains static text for display on the slideshow-pages. The special characters there need to be masked as the typical named entities in HTML-syntax, like `´` for **é**.

When you are done and have added a new language-code and translation to all text-blocks in the file, do not forget to change the file `LANG` in the program directory, if you are using Windows or if the language is not pre-set in the environment-variable `LANG`.

The main dialog



When you start the program, this green dialog is what you see, at first.



The simplicity of the Shoes toolkit results in some small inconveniences, but they are almost entirely restricted to the appearance of the surface. In the screenshot above, you notice a less than optimal alignment of the GUI-elements (and there is more to come...). This does anyway not impair the functionality of the program.

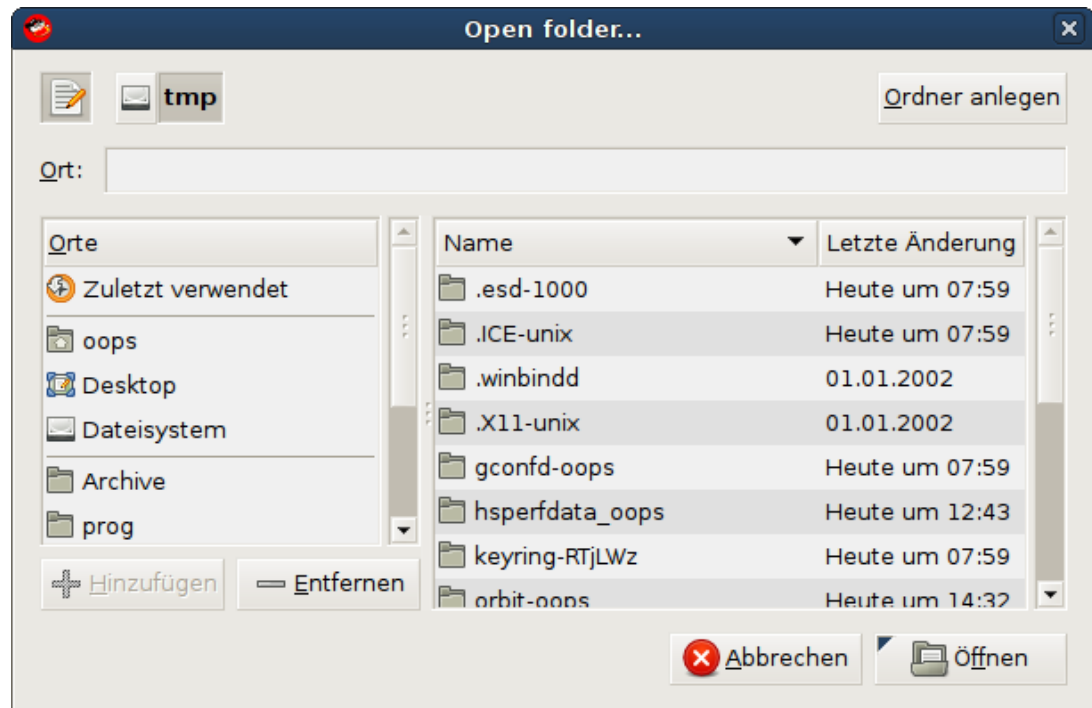
I will now explain each element from top left to bottom right:

Source-directory

This will be a folder (directory) on your hard-disk, where either image-files or (X)HTML-source files are located. You may type the full path to this directory in the text-field or choose a path from the file-system, after you clicked on the small button to the right of it.

Button

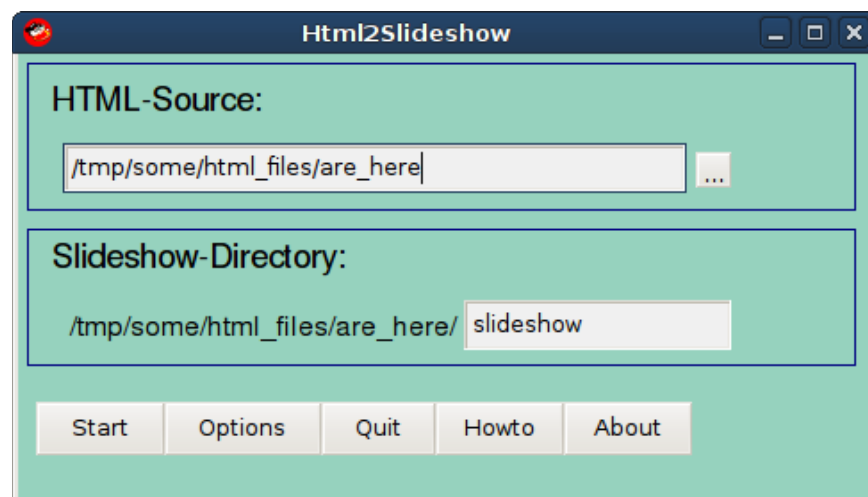
This button opens a file-dialog. You can specify the folder from the file-system, where either the images for your slideshow are found or the HTML source-files, referencing those images, will later be read.



Slideshow-Directory

Initially, this text field is disabled and cannot be altered. Only after you have either entered a source folder or chose one from the file-dialog, you can specify a *sub-directory* where you want the resulting slideshow-files be saved. A default folder "slideshow" is entered automatically, but can be modified. The field can even be cleared, in case that you want the source-folder to contain the slideshow-files, as well. Otherwise, the sub-folder will be created, if it does not already exist, when you click the "Start"-button (see below).

The main dialog after the source-folder has been entered (here, manually via the keyboard):

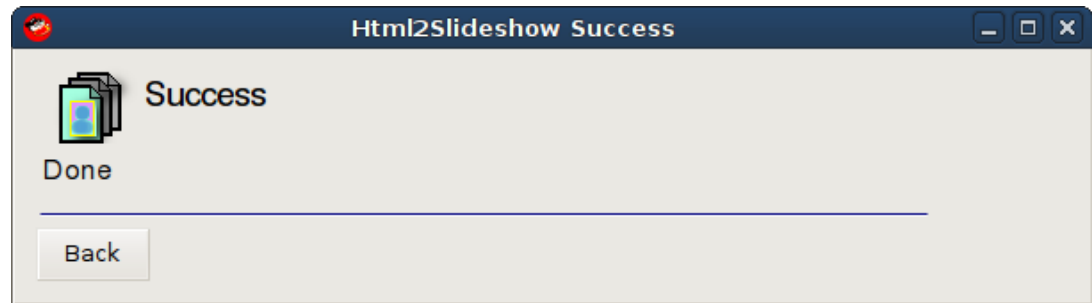


Should the line in the middle of the dialog become too long, it will simply wrap. Maybe try that out, before the surprise makes you assume a program-error. :-D

Button "**Start**"

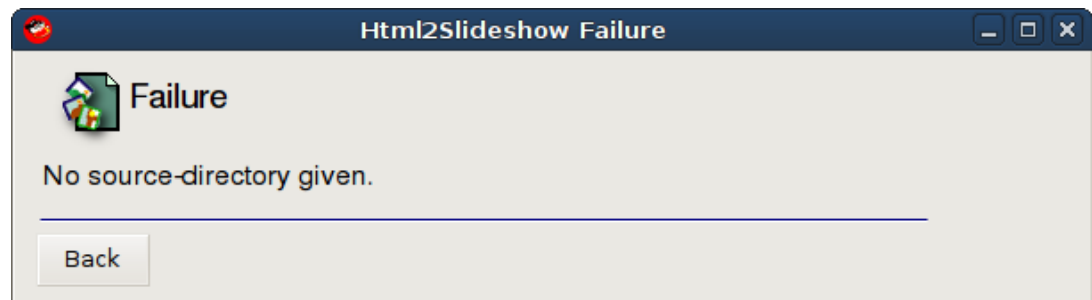
Click this button any time, to create a new set of slideshow-files from the HTML-files in the source-folder.

Provided, that all works well, that there have really been readable HTML-files in that folder and the target-folder could be created and/or written to, you will be informed by HTML2Slideshow:



That is quite a huge window for my frugal artwork and the word "Done", but the same dialog is used on other occasions and I have not yet found out, how the dimension of such a window can be **swiftly** adapted to its current use. Let us ignore that for now, okay? Thank you. ;-)

Should something go wrong, likewise, the message will be a different one:

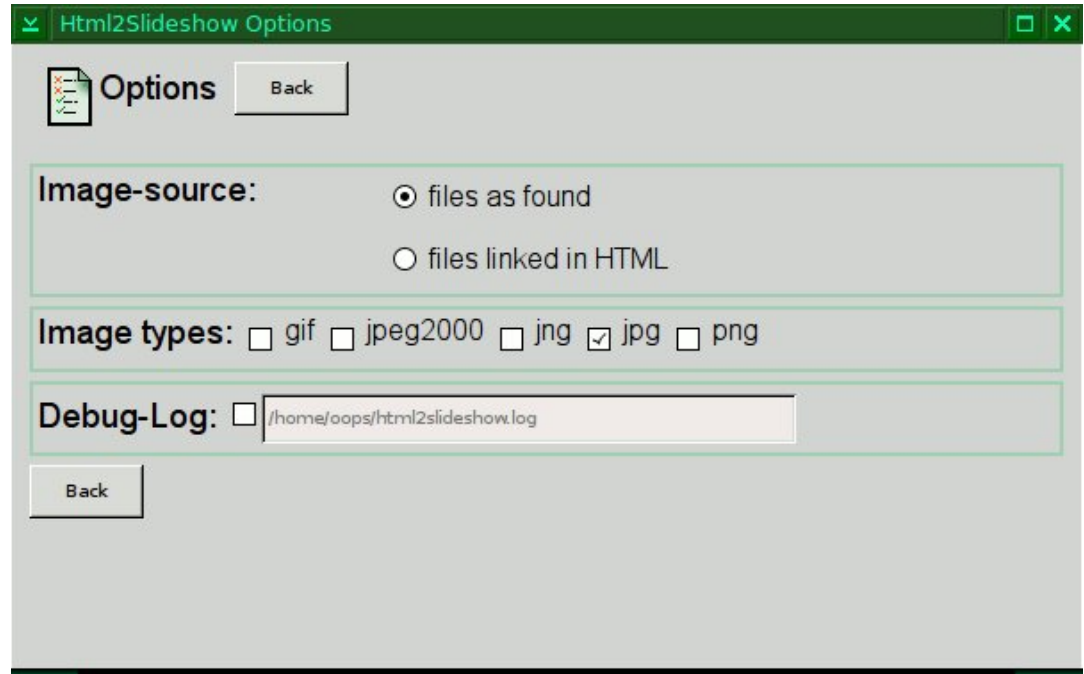


Please use the "Back"-button to return to the main-window. Do not try to close the window by use of the **x**-button in the window-frame. Because I need to avoid several dialogs being open at the same time, the triggering buttons will always stay disabled as long as a dialog is opened. Shoes does not allow me another way to keep track of open dialogs, should you not use the "Back" button. Otherwise, the program needs to be restarted, to activate all buttons again.

Button "**Options**"

This button opens yet another dialog, where you can further control the program behavior to some extent. For the time being, three option sets are available, others may follow, should I find

time and enthusiasm:



By default, the program will use all the image-files of suitable format, which it finds inside the source-folder. But you can select the second option to make it read the path to those images from HTML-files, instead. A HTML-file may reference an image like this:

```
<a href="/sub_folder/my_photo.jpg"></a>
```

Html2Slideshow will only use the URL to the linked file and ignores, if there is an additional thumbnail shown on the page. Choosing image-types, you can create slideshows from PNG, GIF, JNG or JPEG2000 image-files. By clicking the respective check-box, you configure Html2Slideshow to look for a selection of file-extensions, e.g. *.jpg, *.jpeg, *.JPG, *.JPEG in case of jpg, which is also the default.

The debug-option is used to write log-messages to a file for later consultation. As a default, a file "*html2slideshow.log*" in the current user's home-folder is used, but you can name a different file, after you activate the option by a click on the checkbox. The file will be created, if it does not exist and additional contents is appended, otherwise. As long as the checkbox is not activated, though, no log will be written. Be careful to choose a file, that is not needed for other purposes and none that would be destroyed, when text is added to its end!

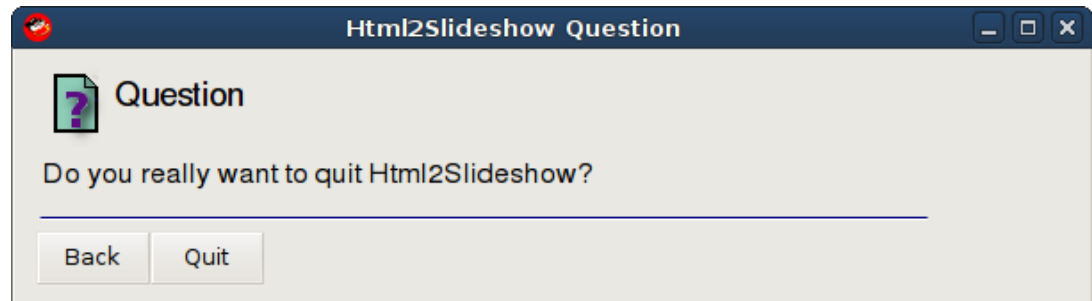
Here, too, the "Back"-button will bring you back to the main dialog. All options have immediately been applied.

Button "Quit"

Does, what it says. With the "Quit"-button you exit the program. In the current version, the configuration is lost each time, when you

close the main dialog and thus leave Html2Slideshow. This means, that before you can generate the next slideshow or replace an existing one, you will have to choose your folders and options once again. This will probably change when the program is updated some time, so that your current settings can be saved and restored in the next session.

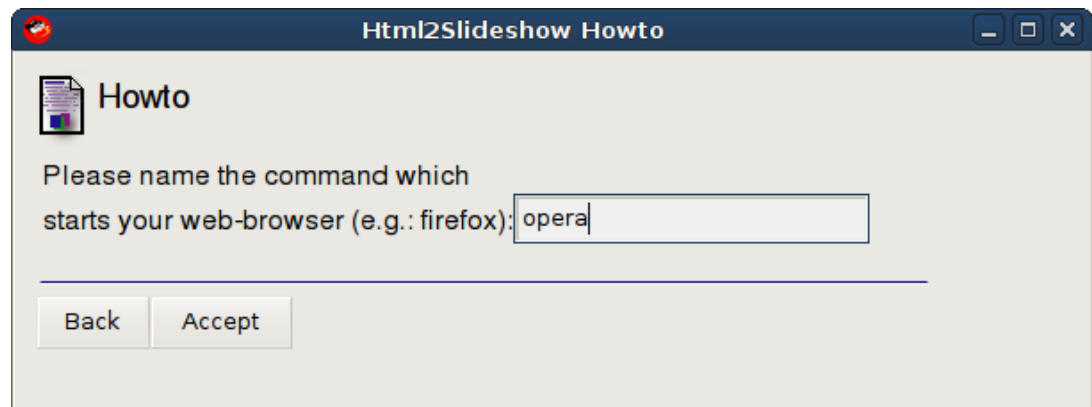
Also, in the current version and due to the non-permanent configuration, each time you click "Quit" in the main dialog, the confirmation dialog from the next screenshot pops up:



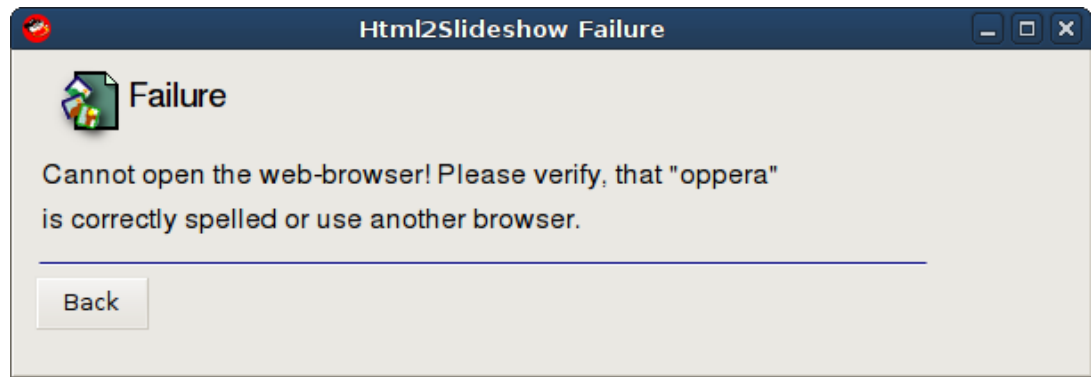
I plan to endow this dialog with a checkbox and allow the users to skip over the confirmation.

Button "**Howto**"

To open the HTML-page, which you are currently reading, any time in a web-browser of your choice, you can click the "Howto"-button in the main dialog. You are asked for the command, which opens the browser. Either type the path to the executable or, if the latter is already in the path for executable files on your system, just the name of the browser:

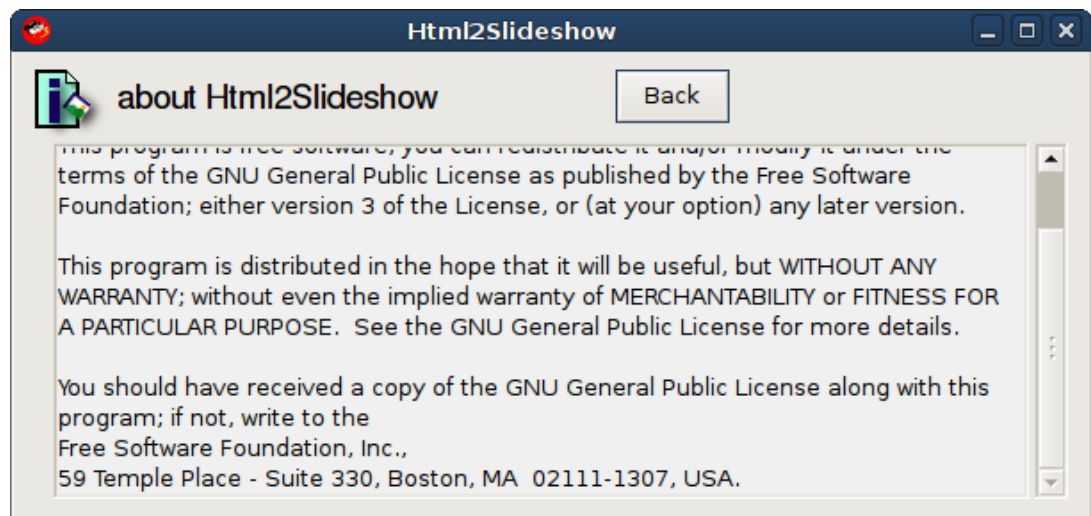


Afterwards, "Accept" will either open the page in your browser or a failure-message is displayed. In that case, you can return to the previous dialog with the "Back"-button and alter the browser-command.



Button "**About**"

This button displays the copyright-information and short version of the GNU-GPL.



Keyboard shortcuts



For each button, that you see on the surface, a keyboard-shortcut exists, which can alternatively trigger the same action as the button does. A list of these shortcuts is not necessary, as each one is the combination of the "Alt"-key with the first letter of the button-caption. For the [Howto-dialog](#) this means, that "**Alt+b**" closes the dialog, while "**Alt+a**" should bring up the web-browser of your choice.

In addition, the shortcut "**Alt+q**" is available in each dialog to quit the application completely. Use the same shortcut again, when asked to confirm that you want to close down Html2Slideshow.

Example



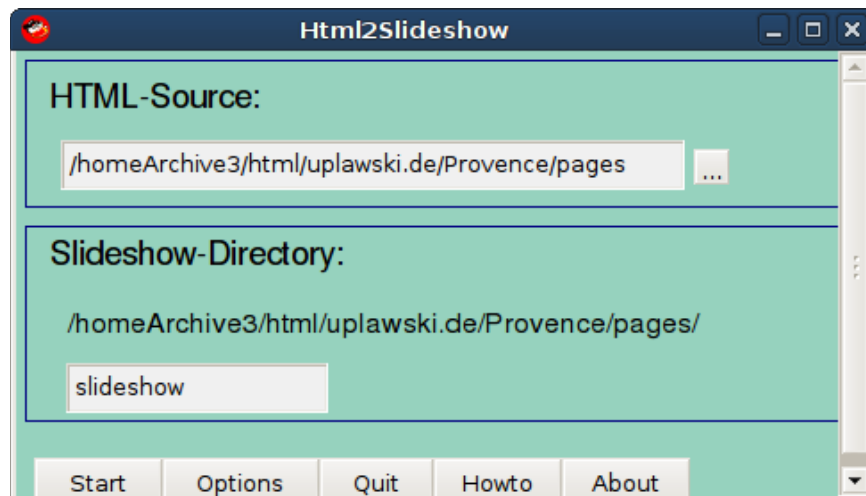
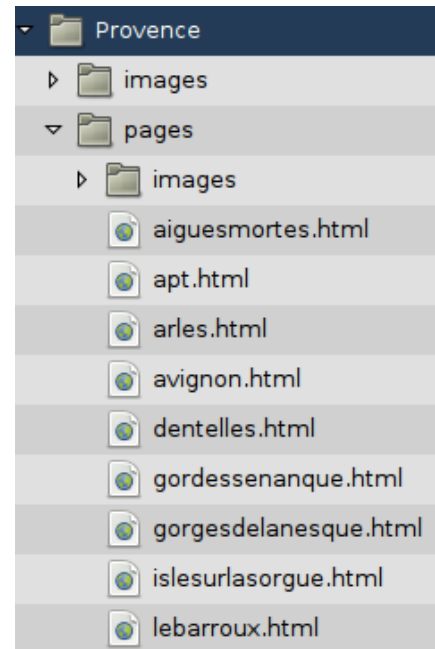
I am using my own site at <http://www.uplawski.de/Provence> as an example. The idea for my first C++-version of Html2Slideshow did also

arise during the creation of those pages.

The local directory with all the inside pages, at first looks like in the screenshot to the right.

What I want, is a slideshow-file for each of those html-files, should they contain references to photos. Html2Slideshow will create those files and spares me the trouble to copy JavaScript and CSS-code in the process.

In the main dialog of the program, I enter the path to the source-folder:



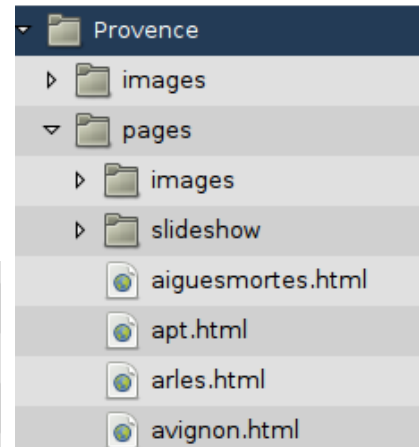
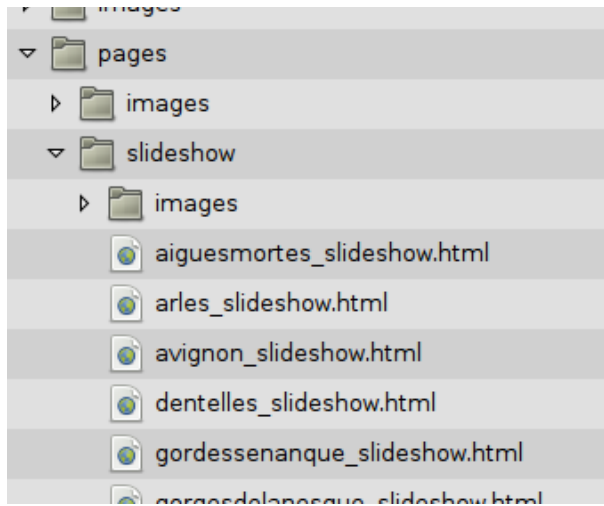
Due to the line-break in the path to the (will-be) slideshow-folder, a slider appears to the right of my dialog. This may or may not happen, depending on your display-configuration and the length of the path-entries, you choose.

Because I want to use the photos referenced in the HTML-source of my web-pages, I have to choose the option "*Image-source: files linked in HTML*", as explained [above](#).

Next, I just click the "Start"-button and a moment later, all is done. You can, though, use the [options-dialog](#) to alter the program-behavior, prior triggering the generator.

To the right, you see part of the result, as the new sub-folder "*slideshow*" has been added below `.../Provence/pages`.

Inside that folder, I find the new slideshow-files along with the CSS- and JavaScript-files, that they will use:



[Html2Slideshow-Homepage](#)