

Tumult Whisk User Manual



Version 2.5
February 1, 2021



Copyright © 2010-2021 Tumult Inc. All rights reserved.
Handcrafted in San Francisco.

Table of Contents

1. Tumult Whisk Overview	page 3
2. User Interface	page 5
3. Editing	page 15
4. Previewing	page 18
5. Developer Tools	page 20
6. PHP	page 21
7. Watched Files	page 23
8. Validation	page 25
9. Code Snippets	page 26
10. Color Swatches	page 29
11. Style Sheets	page 30
12. Tips & Tricks	page 31
13. Troubleshooting	page 32
14. App Security	page 35
15. Keyboard Shortcuts	page 42
16. Version History	page 45

Tumult Whisk Overview

You are currently reading the documentation for **Tumult Whisk v2**.

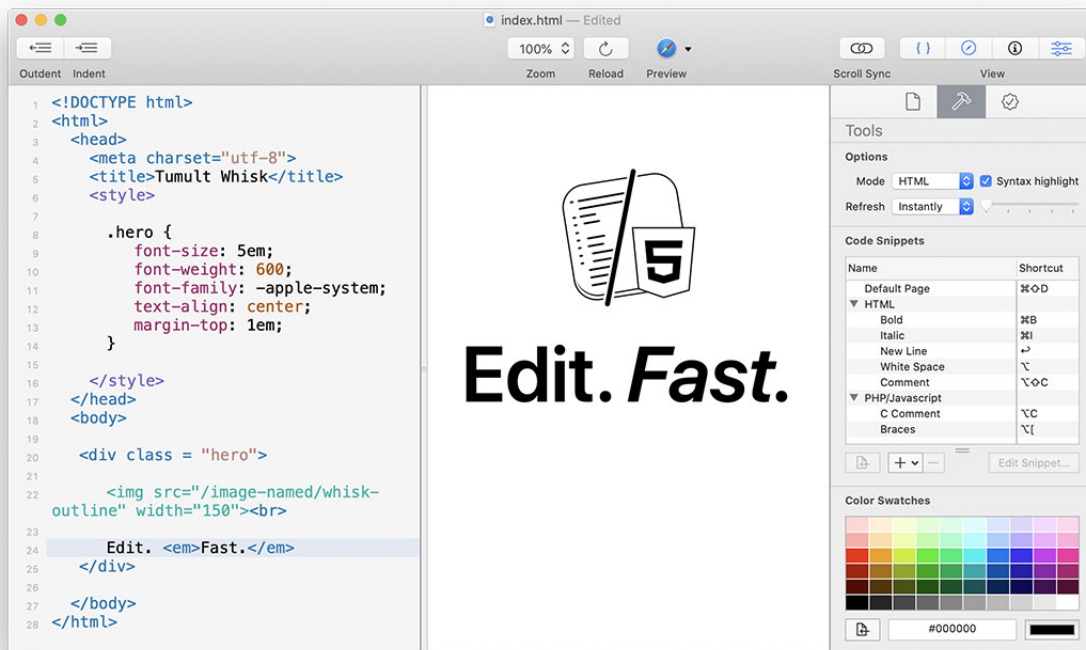
This documentation is also available as a [PDF Book](#) .

For an older version, read the [HyperEdit v1 Documentation](#) .

Tumult Whisk is a lightweight HTML and PHP editor with a live preview that updates as you type.





Whisk is an essential app in your toolkit for HTML, PHP, CSS, and Javascript programming. It is great for learning web development, testing snippets of code, or building entire web pages.

Whisk's integrated preview pane displays the web page live as you type. Whisk breaks the tedious cycle of writing HTML, saving the file, then reloading and viewing the page in the browser. Combining the writing phase with the viewing phase clarifies the effects of your changes and speeds up the overall process of making a web page. W3C-based validation will red-underline any mistakes. It uses the same rendering engine found in Safari, so it is not only standards compliant, but also very fast.



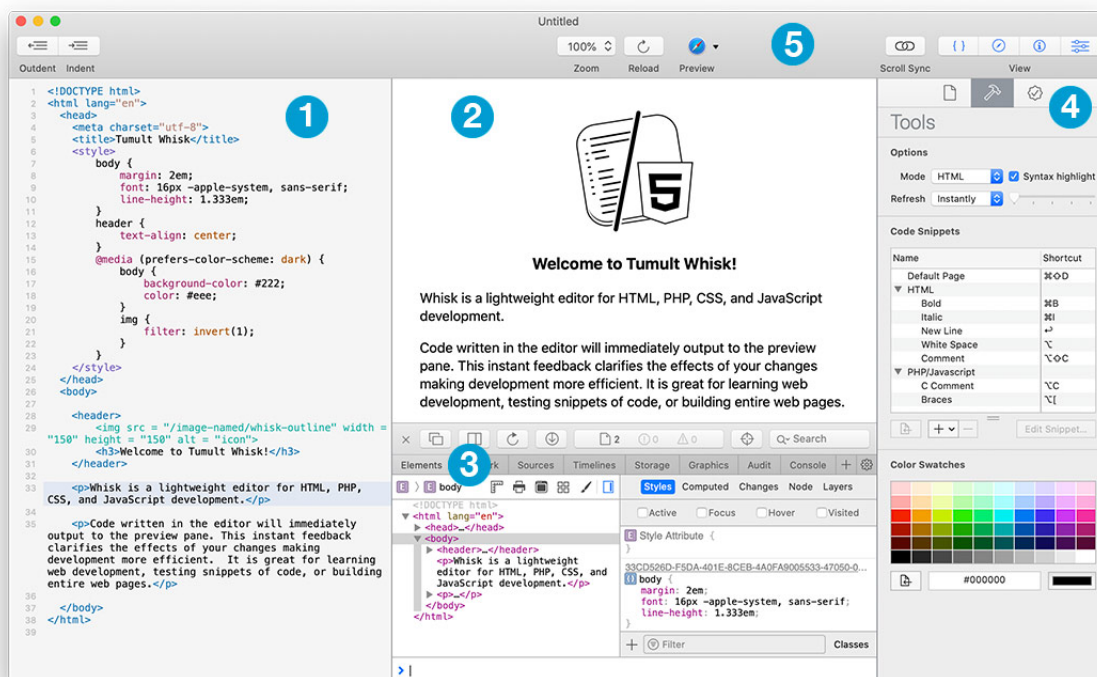
Whisk's main window: edit on the left, preview on the right

Although Whisk assumes you have a basic understanding of HTML, it is still a very useful tool for learning more advanced HTML, as well as PHP, CSS, and Javascript. Through direct manipulation, the changes seen instantly from the result of an action are recognized easier. For more on these languages, please visit documentation available for them:

- HTML: <https://developer.mozilla.org/docs/Web/HTML> 
- CSS: <https://developer.mozilla.org/docs/Web/CSS> 
- JavaScript: <https://developer.mozilla.org/docs/Web/JavaScript> 
- PHP: <https://www.php.net/docs.php> 

User Interface

Main Window



Whisk's main window

1. **Source Editor:** Write code here. Read [Editing](#) documentation.
2. **Web Preview:** Live HTML preview. Read [Previewing](#) documentation.
3. **Developer Tools:** Inspect elements, view the console logs, see network requests, etc. Read [Developer Tools](#) documentation.
4. **Inspector:** Configure the document. Read [Inspector](#) documentation.
5. **Toolbar:** Quick access to common functions and configurability. Read [Toolbar](#) documentation.

Customization

Split Positioning

The split between the Source Editor and Web Preview can be changed by grabbing the divider and dragging.

Double-clicking the divider (or pressing Command-Shift-Backslash) will change from a Widescreen Layout to a Portrait Layout with the Web Preview placed below the Source Editor.

The toolbar's view item or the View menu can also configure which elements are shown.

Web Preview in Separate Window

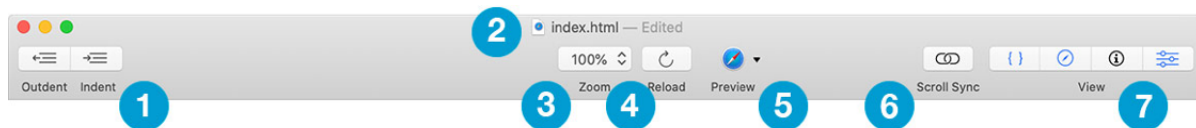
You can tear off the Web Preview and display it in a secondary window by choosing the **View > Detach Web Preview** menu item. This is especially useful if you are using multiple displays and want one display to show the Source Editor and the other to have the Web Preview.

Saving Window Settings

Whisk will save all window layout and options related to your document. Because web files (like HTML, PHP, CSS, etc.) are text only and do not contain metadata, Whisk maintains its own database of your document settings that are tied to the file path of the document.

- If you wish to make all new documents use the current document settings, you can set this via the **View > Save Window Layout and Settings** menu item.
- Likewise you can apply the stored settings to the current document via the **View > Apply Saved Window Layout and Settings** menu item.
- To reset what is saved to Whisk's default, choose the **View > Restore Default Window Layout and Settings** menu item.

Toolbar

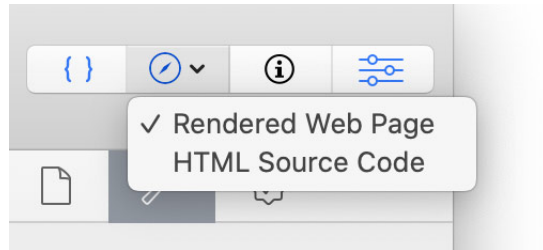


Main window toolbar

1. **Outdent & Indent:** Increases or decreases leading whitespace for selected lines the Source Editor. By default it will use tabs.
2. **Title:** Shows document name and editing status. Click to modify the name or command-click the title to reveal the path.
3. **Zoom:** Magnifies or minifies the Web Preview. On macOS 11+, this behaves similarly to Safari by not increasing the viewport width along with contents. To change Source Editor text size, see the [Interface Preferences](#).
4. **Reload:** Reloads the Web Preview.
5. **Preview:** Clicking the primary icon will open that browser and preview to it. The drop down will show a list of browsers to preview in. For more information, read the [Previewing](#) chapter.
6. **Scroll Sync:** This will lock the scroll for the Source Editor and the Web Preview together, so that the proportionate amount is scrolled the same in both. It is most useful for mostly text documents

where the mapping of HTML code to output can be easily correlated. More complex layouts or code can produce unpredictable results.

7. **View:** Toggle visibility of specific views in the main window. From left to right: Source Editor, Web Preview, Developer Tools, Inspector. If the current editing mode is PHP, the Web Preview control can be pressed and held to reveal a menu to choose between showing the Rendered Web Page and HTML Source Code.



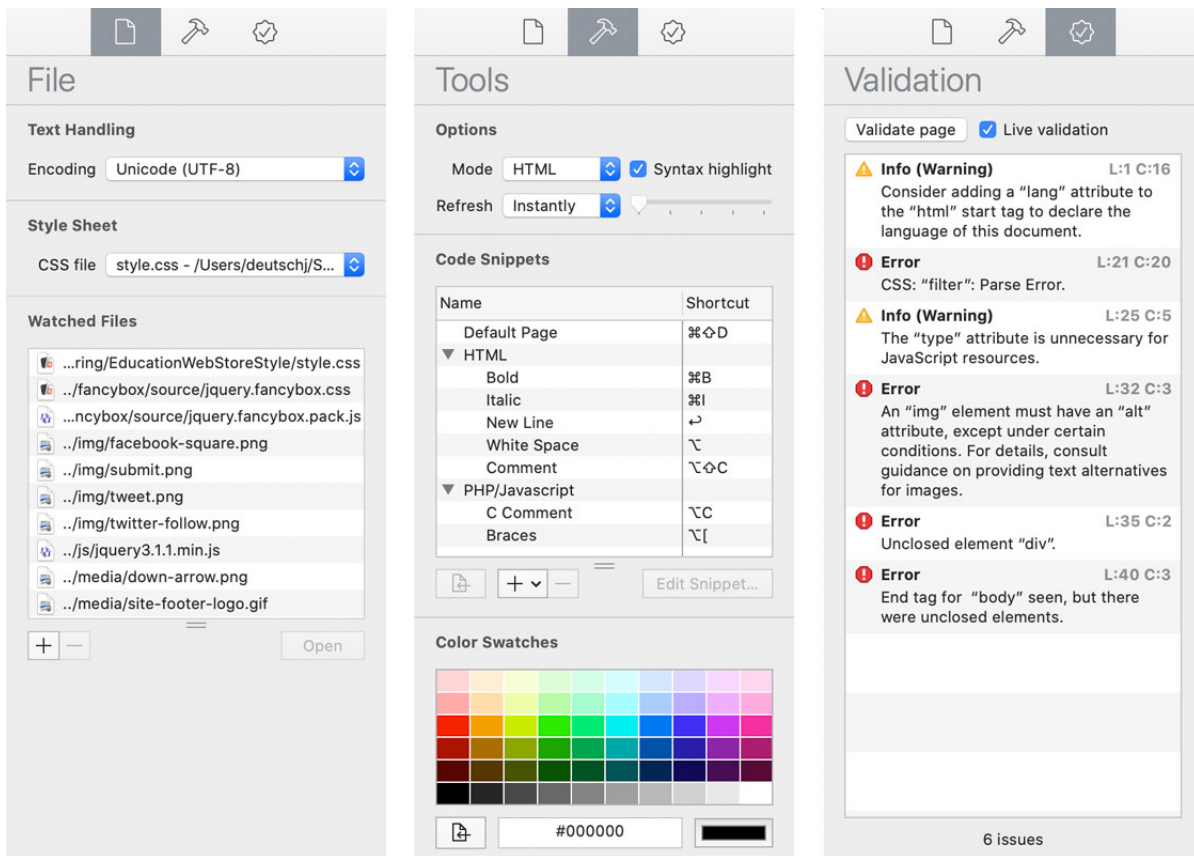
Web Preview drop down menu for PHP

There is also a **Save** toolbar button that can be added via the **View > Customize Toolbar...** menu item.

Tab Bar

On macOS 10.14 and later, Whisk supports using the macOS document tab bar to open several documents within the same window. This can be accessed via the **View > Show Tab Bar** menu item. It will also show up if the System Preferences' Dock Pref Pane's "Prefer tabs when opening documents" setting is "Always."

Inspector



Whisk's File, Tools, and Validation Inspectors

File Inspector

This inspector can be triggered by hitting Command-1.

Text Handling

The **Encoding** setting represents the character set that will be used to write the file. If the document has been previously saved, then it will also open using the selected encoding. Otherwise it will attempt to open using the **Save encoding** setting from the [General Preferences](#). If this encoding does not work, it will attempt to autodetect the text encoding, and then will set this as the value.

For more in depth information, please read the section on [Text Encodings](#).

Style Sheet

This allows you to choose a CSS Style Sheet which will apply to the HTML code. Please see the [Style Sheets](#) chapter for in depth information about this feature.

Watched Files

Watched Files will list all files that will automatically trigger a reload of the web preview. The list contains those that are detected through load mechanisms as well as manually added files. Please see the [Watched Files](#) chapter for in depth information about this feature.

Tools Inspector

This inspector can be shown by hitting Command-2.

Options

The Options area contains general settings related to what type of document you are working with and how often it should be reloaded.

Modes:

- **HTML:** Any text in the Source Editor will be rendered as-is in the Web Preview. By default the refresh is set to Instantly.
- **PHP:** Text entered in the Source Editor is intended to be PHP code. It is run through the binary path specified in the [General Preferences](#). By default the refresh is set to Instantly.
- **JavaScript:** Text entered in the Source Editor has no rendering in the Web View, therefore refreshing is disabled.
- **CSS:** Text entered in the Source Editor has no rendering in the Web View, therefore refreshing is disabled.
- **None:** There is no syntax highlighting. Text entered in the Source Editor has no rendering in the Web View, therefore refreshing is disabled.

Syntax Highlighting determines whether the Source Editor colors the text. Actual colors can be configured in the [Interface Preferences](#).

TIP

If you have a very large document, turning off Syntax Highlighting can improve editing performance.

There are three different Refresh timings:

1. **Instantly:** The Web Preview is refreshed immediately after every keystroke/text change.
2. **Delayed:** The Web Preview is refreshed after a duration from the last keystroke/text change. This uses the values of the slider located on the left, where each tick mark represents one second. A delayed refresh is useful to avoid a certain amount of distraction while working but still wanting to be able to quickly see the effects of your changes. It may also be useful in situations where the Web Preview uses a lot of system resources to render and is hindering editing responsiveness.
3. **Manually:** The Web Preview is only refreshed when hitting the Reload button or Command-R. This is useful when the content may have adverse side effects in an intermediate representation (like

PHP writing to the filesystem) or from network requests.

Code Snippets

These are bits of code that can be easily inserted to the working document by double-clicking or using custom keyboard modifiers.

Whisk opens documents with an empty page, but the "Default Page" code snippet, triggered via Command-Shift-D, is a good starting point for new documents.

Read the [Code Snippets](#) chapter of the documentation for more information.

Color Swatches

Color Swatches provide quick access to your most commonly used colors. They are globally saved so all documents have access to them.

Read the [Color Swatches](#) chapter of the documentation for more information.

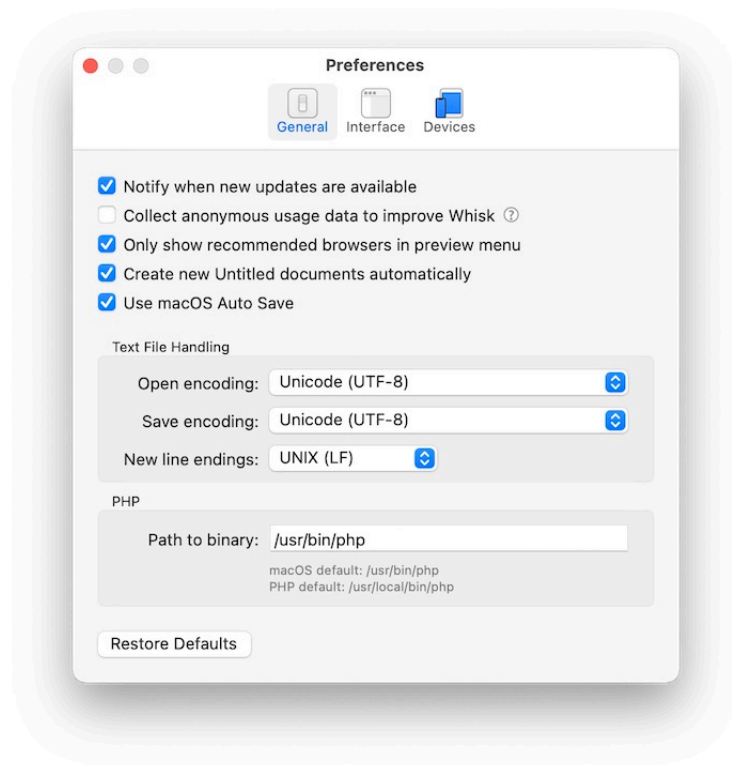
Validation Inspector

The Validation Inspector provides a list of issues and errors in your document. You can do single validation report via the **Validate page**, or constantly update the issue list by switching on the **Live validation** checkbox.

Read the [Validation](#) chapter of the documentation for more information.

Preferences

General Preferences



Whisk's General Preferences

- **Notify when new updates are available:** If checked, on startup Whisk will always look for new updates on launch. It is recommended to have this setting checked, otherwise you may be missing critical updates that include bug fixes or new features. *(This option is only shown on the Tumult Store version of Whisk. The Mac App Store version uses the App Store updating mechanisms which can be configured through the System Preferences)*
- **Collect anonymous usage data to improve Whisk:** This setting currently does not do anything; data is not presently transmitted. It is reserved for possible future use to help determine product direction. You can click the **?** for details on the collection policy and read Whisk's [Privacy Policy](#).
- **Only show recommended browsers in preview menu:** Whisk maintains a whitelist of browsers to show in the preview toolbar menu item because querying macOS for applications which can handle "http" URLs or "html" files can often result in non-browsers showing up in the list. However new browsers may come out which are not immediately on Whisk's whitelist or there may be applications you want to preview to which Whisk would not consider a traditional browser. If this is the case, check this box for the preview menu to show all options available on the system.
- **Create new Untitled documents automatically:** When checked, Whisk will create an Untitled document if it is activated without any other documents open.
- **Use macOS Auto Save:** When checked, Whisk will make use of the modern macOS [Auto Save and Versions](#) document capabilities, which have existed since OS X 10.7 Lion. This adds Duplicate, Rename, Move To, and Revert To menu items. Save As is accessible by holding the option key. Because multiple versions of your document are automatically saved and can be reverted via the Time Machine interface, it is recommended to keep this setting enabled. It is

provided as there are some situations such as using [Panic Transmit](#)'s "Edit In" feature where automatic saves can lead to unintended effects of uploading invalid files. If changed, you must quit and relaunch Whisk for it to fully take effect.

Text File Handling

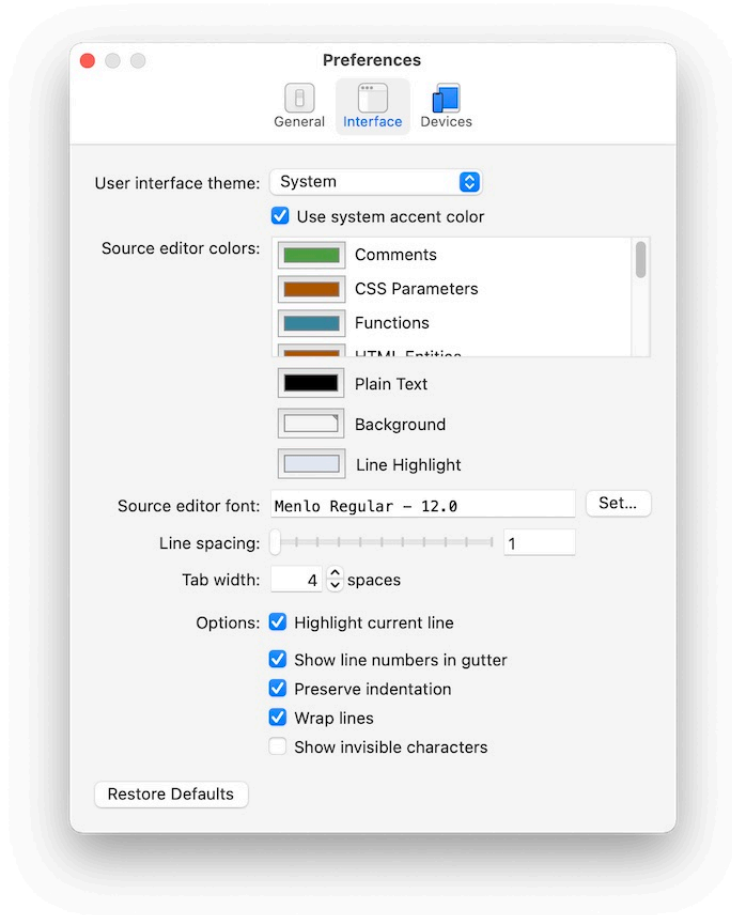
Read the [Text Encodings](#) section for information about the Open and Save encoding settings.

New line endings determines which type of control character marks the end of lines. It is only used when a newline is created through editing in Whisk; existing line endings are not changed. A "newline" is an invisible character which instructs the computer to start a new line of text. In the early days of computing, DOS, Mac, and UNIX all had different invisible characters/sequences that they used. This setting is for more [historic purposes](#), and generally should not be modified.

PHP

Path to binary indicates the command line binary that should be called when rendering PHP code. This defaults to the macOS installation location, but can be modified if you install via homebrew, MAMP, or other means. For more information, read the [PHP](#) documentation chapter.

Interface Preferences



- **User interface theme:** Whisk can run in light or dark themes. On macOS 10.14 and later, a "System" option is available that uses the respective theme depending on the setting chosen in the System Preferences General Pref Pane.
- **Use system accent color:** On macOS 10.14 and later, an accent color can be chosen in the System Preferences General Pref Pane. This is used for certain highlights in Whisk, but may not always be desired.
- **Colors:** These change the syntax highlighting as well as text, background, and line highlight colors. Each is tied to a theme, so you can have independent colors on the Dark and Light themes.
- **Source editor font:** Changes the font and size for all documents. If a document is open, you can see the changes live.
- **Line spacing:** Changes the distance between lines for all documents. If a document is open, you can see the changes live.
- **Tab width:** The distance between tab stops, measured in the number of equivalent space characters. If a document is open, you can see the changes live.
- **Highlight current line:** Lightly colors the background behind the line that has the text cursor. If there is a selection, this is temporarily not drawn in lieu of the selection. The color can be configured via the "Line Highlight" color above.
- **Show line numbers in gutter:** This will show a number next to each line in the Source Editor. Disabling can sometimes improve performance on large files.
- **Preserve indentation:** If this setting is checked, the leading whitespace (spaces, tabs) of the previous line will be created as the leading whitespace for a new line.
- **Wrap lines:** Determines if lines that are wider than the view of the Source Editor should be displayed below, or if unchecked, cause a horizontal scrollbar to appear.
- **Show invisible characters:** With this checked, the Source Editor will lightly print any characters that represent whitespace or normally non-printed characters in the file. It is useful for finding "gremlins" which might be unexpected data in a file.

Devices Preferences



Whisk's Devices Preferences

Whisk can preview your document to iOS devices using the [Hype Reflect](#) application. Devices are chosen via the Preview toolbar menu item.

After a device is previewed to, it will stay in the Preview menu, even when the Hype Reflect application is not running. **Clear Recently Previewed Devices** will reset these choices, so it will then only show active devices.

File Access Preferences

The Preference Pane will only show up on the Mac App Store version of Whisk. For more information, see the [App Sandboxing](#) chapter of the documentation.

Editing

Editing basics

Keyboard Manipulation

Whisk's Source Editor is a standard macOS text view. As such, all common [keyboard shortcuts](#) will work. [Code Snippets](#) can also be inserted via shortcuts.

Syntax Highlighting

Whisk provides syntax highlighting for the following languages:

- HTML
- PHP
- JavaScript
- CSS

The editing language and render mode for [previewing](#) can be set via the [Tools Inspector](#) in the Options area. You can also disable syntax highlighting.

Colors and showing invisible characters can be adjusted in the [Interface Preferences](#).

Indentation

When making a new line, Whisk will preserve the leading indentation of the previous line. This can be configured in the [Interface Preferences](#).

You can use the **Outdent** and **Indent** toolbar buttons, **Text > Shift Right/Shift Left** or Command-] and Command-[to adjust the leading indentation of the current selected lines. If multiple lines of text are selected, the tab and shift-tab keys will also perform this function.

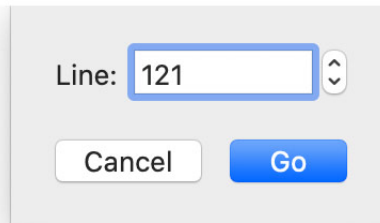
Commenting

Use the **Text > Comment Selection** or Command-/ to toggle comments on and off for individual lines. This is context aware, so HTML code will make `<!-- -->` comments, CSS will use `/* */`, and JavaScript leads lines with `//` comments. Commenting shortcuts are also available as [Code](#)

Lines

Whisk will by default show line numbers in the gutter on the left. This can be disabled in the [Interface Preferences](#).

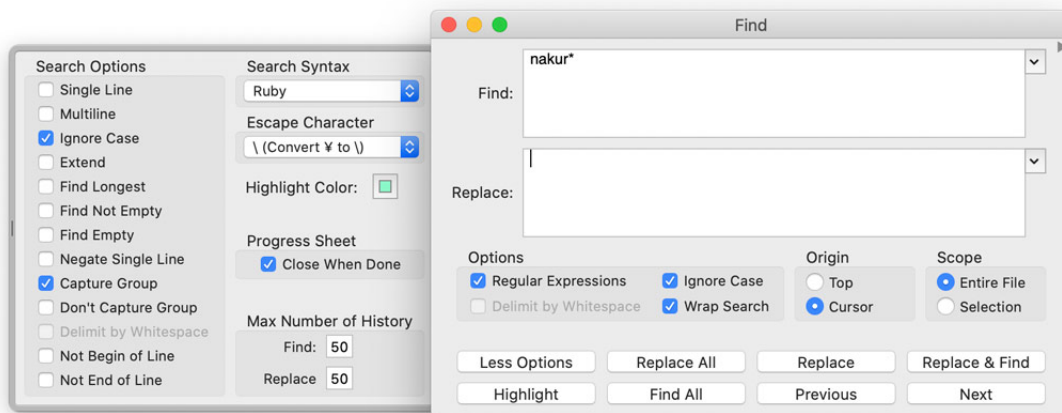
To jump immediately to a specific line, you can choose the **Text > Go to Line...** menu item (or Command-L) and choose a line number to select.



Whisk's Go to Line sheet

Find and Replace

Whisk uses [OgreKit](#) to provide Find/Replace functionality. This powerful find panel allows for entering regular expressions, highlighting multiple results, and much more.



Whisk's Find and Replace Panel

Spell Checking

Automatic spell checking is disabled, as this is often inappropriate for source code. You can choose the **Edit > Spelling and Grammar** menu to configure if/when the document is checked.

TIP

You can quickly spot-check spelling by hitting Command-; (semicolon). It will check starting at the current text cursor position.

Configuring the editor

All other features and options for the Source Editor are configured in the [Interface Preferences](#).

Text Encodings

Text Encoding, otherwise known as [character encoding](#) or charsets, is how computers convert between on-screen text and their underlying data representation on disk. A language like English has very different characters than one like Japanese, so a different encoding is needed for computers to efficiently store the text. In the past, there were encodings used for both different languages and different operating systems, since the operating system creator often decided how they wanted to represent the characters on disk. In the last two decades, the world has slowly converged on a single standard, [Unicode](#), which can represent nearly every major language. As computers have become more powerful and we interact globally through the internet, efficient storage is less of a concern than compatibility. Specifically, Unicode's [UTF-8](#) representation has become the most common and recommend text encoding.


By default, Whisk will open and save files using **UTF-8**. You probably will not need to change this.

However, older documents may have made use of different encodings. You can adjust the text encodings used to open and save documents in the [General Preferences](#). The [File Inspector](#) allows you to set the text encoding for writing the file. This setting is preserved in Whisk's database tied to the file path.

Previewing

Under-the-hood, Whisk uses a HTTP server to show the contents in previews (both in-app and external). This allows for closer simulation to what content will look like on a server. It only allows connections from your localhost machine by default; see the [Preview Server App Security](#) documentation for more details.

In-App Web Preview

Whisk's Web Preview is powered by [WebKit](#) , the same rendering engine found within Safari. Because of this, it is fast and standards compliant.

The Options section in the [Tools Inspector](#) allows you to edit when you want it to refresh. You can manually reload via the **Reload** toolbar button, or Command-R.

To debug the Web Preview, you can show the [Developer Tools](#) to get access to the Web Inspector.

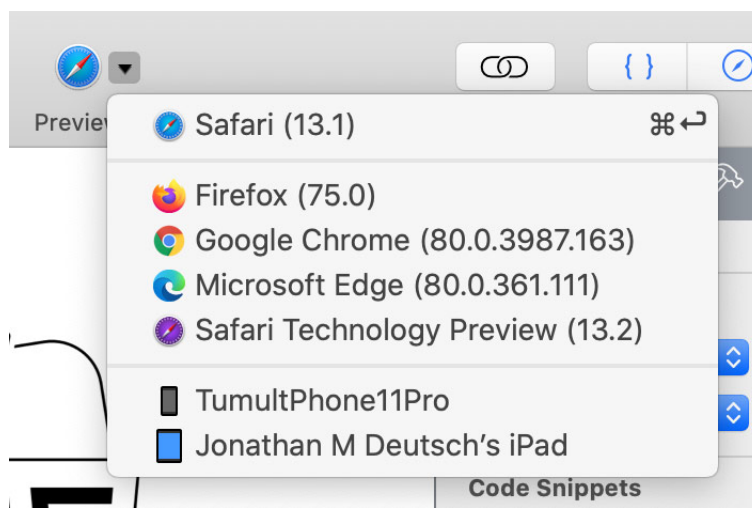
Browser Preview

Desktop Browsers

Whisk can show your document in browsers installed on your computer. It is a good idea to test for compatibility in at least these three browsers:

- Safari
- Chrome
- Firefox

To preview to a browser, hit the **Preview** button in the Toolbar. The Toolbar has a pop up button that can reveal other browsers. The last one chosen becomes the default for the browser preview, and can be quickly shown via Command-Return.



The Preview Toolbar Button Pop Up Menu

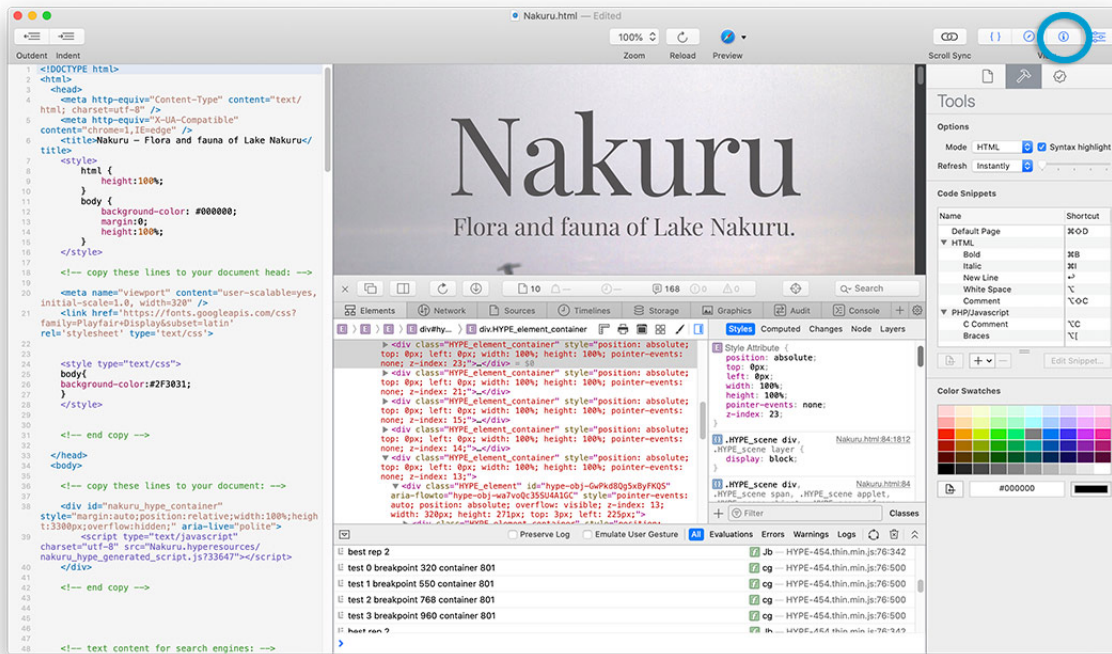
iOS Devices

Whisk can also preview to iOS devices directly using the [Hype Reflect](#) application. This was originally developed as part of [Tumult Hype](#). Read the [Hype Reflect documentation](#) to learn more.

Developer Tools

Whisk provides full access to the Safari Web Inspector. You can get JavaScript console log output, inspect elements, view network requests, and do much more. Read [Apple's Safari Web Inspector Documentation](#) for more information.

To show the Developer Tools, click the corresponding button in the toolbar or choose **View > Show Developer Tools**.



Document with the Developer Tools open

If the Web Preview is narrow, the Developer Tools will open in a separate window. Otherwise it will be embedded in the Web Preview.

TIP

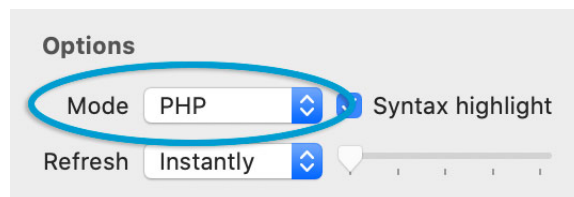
When the Developer Tools is open, Whisk will inject a `<script>` tag into the HTML sent to the web preview requesting a file called `whisk-console-delay.js`. This file is blank; its only purpose is to help workaround a bug in the console where it can incorrectly appear blank if a reload happens quickly.

PHP

Activating PHP Rendering

Because macOS 10.3 and later comes with the command-line version of PHP out-of-the-box, you can use Whisk to run and display the results of your PHP code live. It is excellent for testing smaller amounts of code before insertion into a large site or testing non-complex pages.

A file with a `.php` extension renders as PHP by default. To activate PHP rendering on other/untitled documents, simply change the **Mode** in the Options section of the **Tools Inspector** to **PHP** and it will automatically be rendered. It can also be activated from the main menu via **Preview > Mode > PHP**.



Change the mode to PHP in the Tools Inspector

The PHP editing mode can also be set for all new documents by selecting it and then choosing the **View > Save Window Layout and Settings** menu.

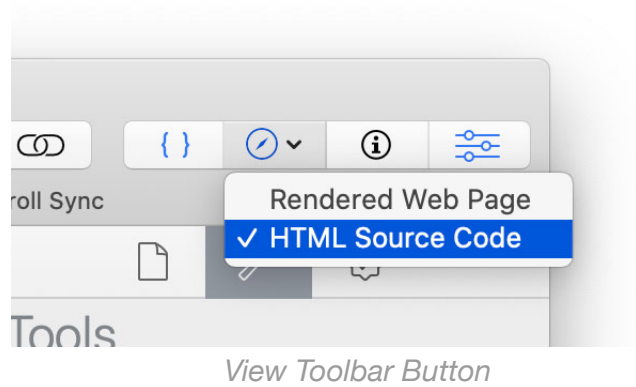
TIP

When Whisk is set to refresh instantly, it will report parse errors very often. Therefore setting a delay of 1 second is recommended.

Viewing HTML Code Output

By default, the PHP mode will render the page in the Web Preview. But it is often useful to look at the HTML Source instead to see what the code is outputting.

The HTML source generated by PHP can be shown by long-pressing the Web Preview toolbar button and choosing **HTML Source Code**:



It can also be viewed by pressing Command-Shift-Right Arrow, or by choosing the **Preview > PHP Output > HTML Source Code** menu.

Configuring PHP

Whisk's [General Preferences](#) allows setting the path to the PHP binary. This defaults to the macOS installation location, but can be modified if you install via homebrew, MAMP, or other means.

Troubleshooting

See the [PHP Troubleshooting](#) page for more information.

Watched Files

Whisk's Watched Files feature makes sure that the Web Preview is reloaded whenever any files from the page are changed. It automatically creates a list of resources used by the page, and watches your filesystem to see if any changes have been made. When they have, Whisk will reload the page.

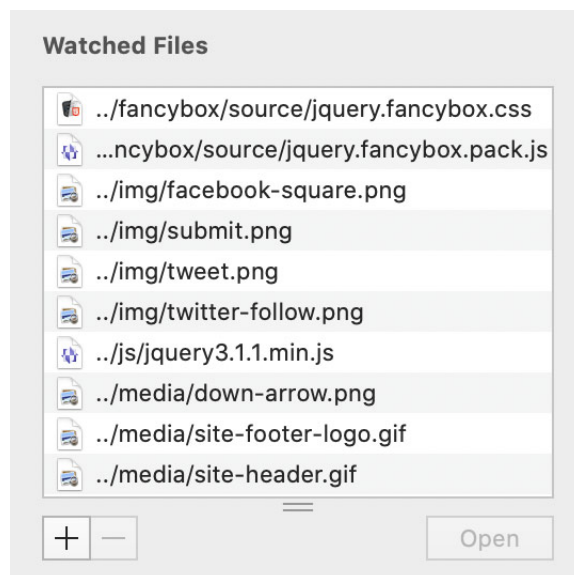
This means with Watched Files you can externally edit images, CSS files, JavaScript files, PHP include files, or other embedded data, and refresh the Web Preview automatically when those files are saved.

TIP

If your document's Reload setting in the Options section of the Tools Inspector is set to **Manually**, watched files will not automatically reload the page. Change to Instantly or Delayed instead.

Adding Watched Files

Watched Files are located in the [File Inspector](#).




Watch Files

Whisk will automatically monitor resources requested by your HTML page. If it finds a corresponding file on disk, it will automatically be added to the Watched Files.

WARNING

The Mac App Store version of Whisk must have file access permissions for Watched Files to work properly. For more information, see the [App Sandboxing](#) chapter of the documentation.

There are many cases where automatic adding will not find a resource that you may wish to be watched for reloads. Most commonly this might be PHP include files, but could also be data files that are requested via XMLHttpRequests or other dynamic resources. To manually add your own files or folders to watch for changes, either hit the  button and choose them in the Open Panel or drag the files directly into the table view.

Instantaneous Updates to Files Open in Whisk

Whisk will reload a page's Web Preview if there are changes on disk to a file. There's a special mode that can update even faster: if the file is open in Whisk, then any changes (even unsaved ones) will trigger a reload. Instead of using the on-disk representation, Whisk will use the in-memory representation for the file.

This is especially useful for editing external CSS files; changes you make in the Source Editor will be reflected immediately in the other document's Web Preview.

WARNING

The in-memory representation is only used for assets referenced by the HTML page; PHP resources are not used in this manner.

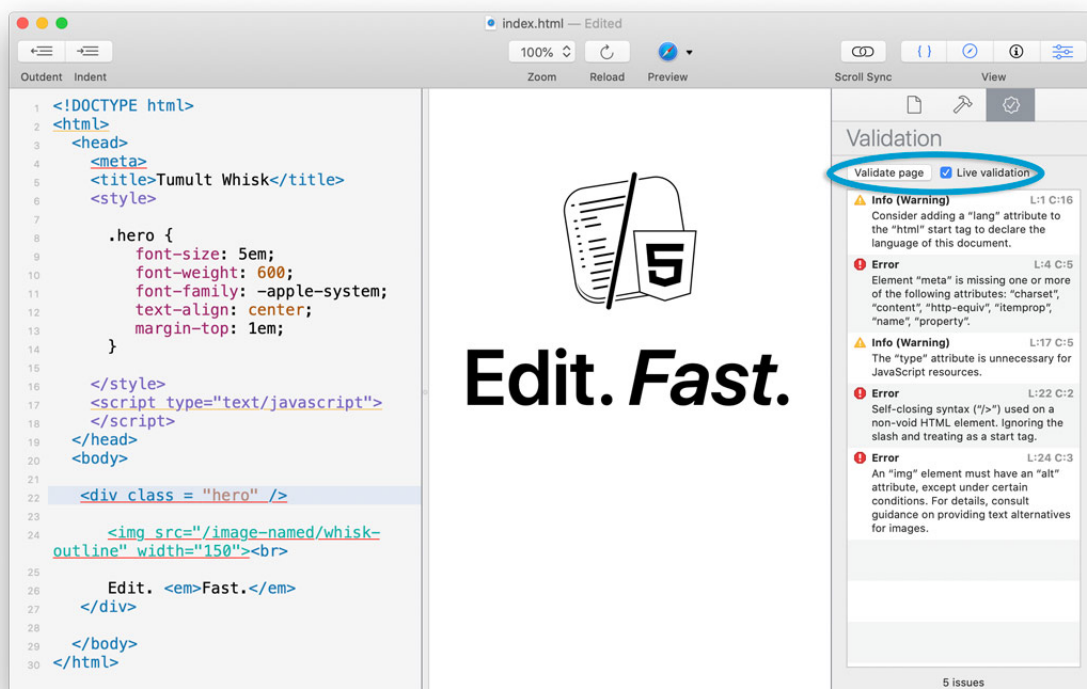
The Watched Files section in the Inspector provides an **Open** button to quickly open files in Whisk. If they cannot be edited in Whisk, they will be opened with the file's preferred editor.

Validation

Whisk offers live HTML5, CSS, XHTML5, and SVG validation.

Under-the-hood, this is same engine as the W3C validator: [The Nu Html Checker \(v.Nu\)](#) .

Live validation is enabled by checking the **Live validation** box in the [Validation Inspector](#). You can also validate in non-live manner by click **Validate page** or choosing **Text > Validate Page** (Command-K). Issues are shown in the Validation Inspector table view and yellow or red underlines will appear in the Source Editor when the validation has encountered a mistake.



A document with some issues.

Error descriptions are reported in the table, and clicking on them will select the offending line of text. The table can be temporarily cleared via the **Text > Clear Validation** menu item.

The first validation needs to start the validation process, so may take about 5 seconds before results are shown. Progress is displayed at the bottom of the Validation Inspector.

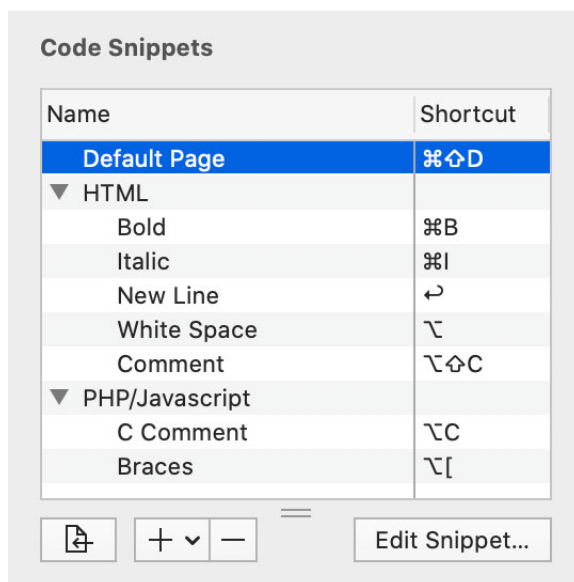
You can quickly move between errors via the **Text > Next Error** and **Text > Previous Error** menu items, or Control-Up/Down.

Code Snippets

Code snippets allow for quick insertion of reusable code and tags using keystroke combinations. Any keystroke that is not currently used by the system can be used for your snippet.

Using Code Snippets

Code snippets are found in the **Tools Inspector**:



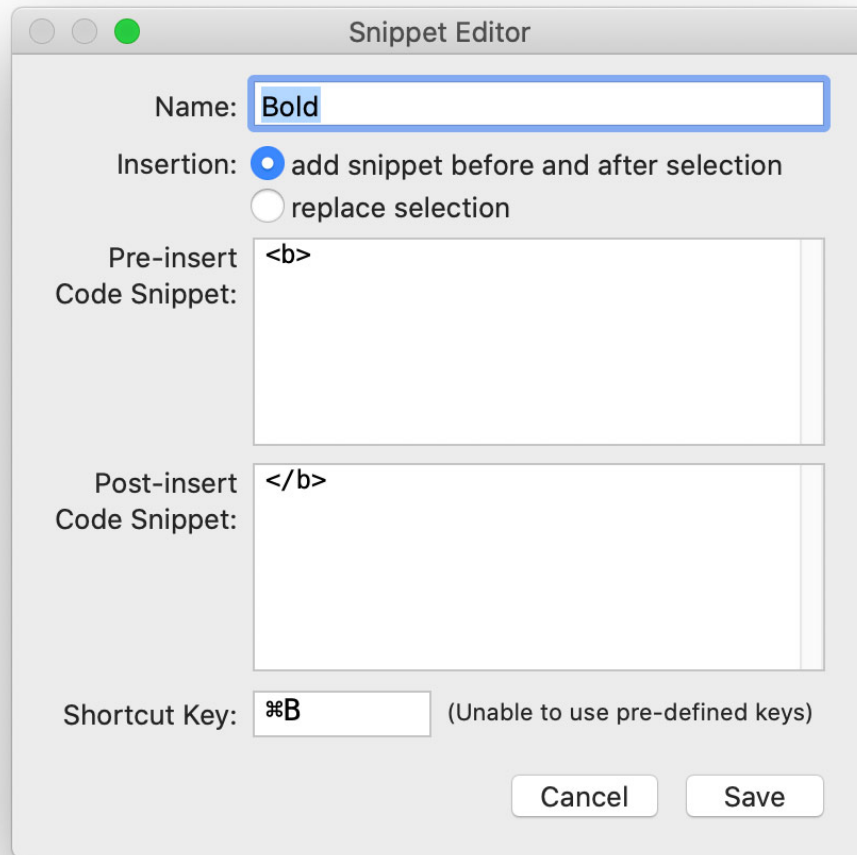
The Code Snippets section

Snippets can be inserted into the Source Editor using 4 different methods:

1. Use the Keyboard Shortcut associated with the Code Snippet
2. Click the Insert button
3. Double-click the snippet
4. Drag the snippet to the Source Editor

Editing Code Snippets

Clicking the **+** pop down and choosing **New Snippet...** or clicking the **Edit Snippet...** button will bring up the Code Snippet Editor.



Code Snippet Editor

- **Name:** The description that will appear in the Code Snippets table.
- **Insertion:** Whether the selected text should be preserved by placing the snippet's pre- and post-insert code snippet before and after the text, or if the selected text should be replaced by the pre-insert Code Snippet.
- **Pre-insert Code Snippet:** The code that should be placed before a selection, or the code to replace the entire selection if the insertion option is set to do so.
- **Post-insert Code Snippet:** The code that should be placed after a selection (only if insertion radio button is not set to replace the selection).
- **Shortcut Key:** The keystroke combination that will insert this snippet into the document. Command-keys are not required, so even regular letters can be used. For example, the "enter" key (on the numeric keypad) is by default set to generate a `
`. Pressing delete will clear the shortcut key.

WARNING

If it is used by the system or Whisk already, it will probably beep (and the shortcut field will not be updated). Simply choose another combination.

Magic Variables

These strings can go in the snippet code, and will be automatically replaced when inserting into the Snippet:

- `${title}` : The filename, without extension, of the Document. If it has not been saved, then "Untitled" will be used.
- `${charset}` : The current text encoding name. The use case is use this in a meta charset tag, which is what the Default Page snippet does.
- `${caret}` and `$/caret` : These are only used if the **replace selection** option for insertion is chosen. They mark the new selection range.

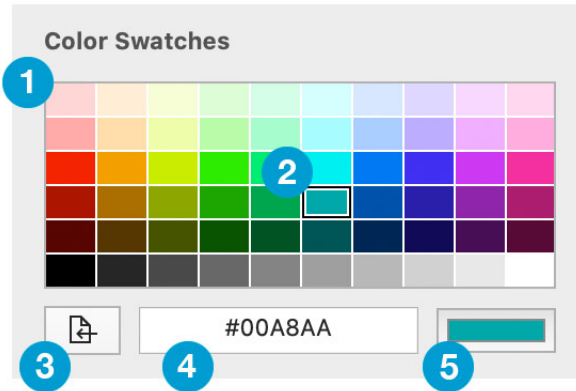
Grouping Code Snippets

Groups of one level can be created for organizational purposes. To create a group, press the `+` pop down button and choose **New Group**.

Double-clicking a group will allow you to rename it (whereas double-clicking a snippet will insert it into the document). Simply drag and drop snippets or groups to rearrange them.

Color Swatches

The **Color Swatches** section in the **Tools Inspector** lets you visually find web values for colors, save commonly used colors, and insert them into your document.



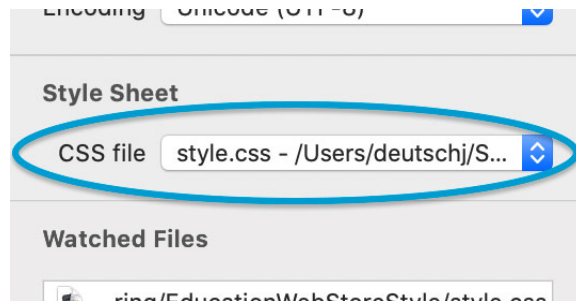
1. **Swatches:** These are all the swatches available for saving colors. They come pre-populated with the rainbow.
2. **Selected Swatch:** Clicking on a particular swatch will fill in its value into the Color Text Field and Selected Color Well. Double-clicking on the swatch will insert it into the document.
3. **Insert Button:** This will insert the color's value from the Text Field into the cursor position of your document.
4. **Color Text Field:** This field represents the text representation of the color from the Selected Color Well. It is bidirectional, so you can also type in this field to update the color in the well. It accepts 3-digit hex, 6-digit hex, 8-digit hex, `rgb()`, `rgba()`, and named colors.
5. **Selected Color Well:** Clicking this will open the standard macOS Color Picker as a Pop Over window. When a color is selected, it will show up in the well and show the text value in the Color Text Field. The color can be added to the swatches by dragging from the well to the desired color swatch.

TIP

To quickly test different colors in your document, select the text for the color in the Source Editor. When the **Insert Button** is used, it will replace the selected text with the new color and also preserve the selection, so each new insertion will let you immediately try the new color.

Style Sheets

Style sheets can be attached to documents without using `<link>` or `<style>` tags. This is useful for writing blog entries or forum postings, where you want to test how it will look when published. To attach a style sheet to the document, select **Choose...** from the CSS file pop-up button in the Style Sheets section of the File Inspector, and then select your file.



An attached style sheet in the options palette

The style sheet choice is associated with the file path in Whisk's file settings database, so reopening the file will preserve the selection as long as the file has not moved.

The style sheet can also be set for all new documents by selecting it and then choosing the **View > Save Window Layout and Settings** menu.

To clear the list of style sheets, choose **Clear Menu** from the CSS file pop-up button.

Style Sheets are automatically added as **Watched Files**, so if you have one open changes will be immediately reflected in the document.

Tips & Tricks

Quickly populate default HTML/PHP code for a page

Whisk's [Code Snippets](#) can hold commonly used bits of code, including entire pages. While Whisk by default creates blank documents, you can insert the "Default Page" as a good starting point. The default keyboard shortcut is Command-Shift-D. The code can be modified by editing the Snippet if it does not meet your needs.

Performance Tips for large documents

- Make sure that the refresh in the options palette is set to either a delay or to manually.
- Turn off gutter line numbering in the editor preferences.
- Turn off Syntax Highlighting by unchecking the Text:Syntax Highlighting menu item.

Tab to inserts spaces

By default, Whisk does not insert spaces when the tab key is pressed. However, because any key can be used to for inserting a snippet, the tab key can be overridden. Simply create a new snippet, and in the Pre-insert Snippet Code, type as many spaces as you would like to enter. In the Shortcut Key, press the tab key once, and then save the snippet. Now, the tab key will insert spaces!

WARNING

Shifting blocks of text right will still insert tabs.

Send text through a command line program

Whisk currently uses the command line to evaluate PHP code. This program can be manually changed by editing the Path to PHP binary field in the advanced preferences. For example, the path could be changed to `/usr/bin/wc`, and when the PHP mode is selected, it will instead give a line, word, and character count.

WARNING

The path field does not accept arguments, but a shell script could be used to supply them to the program you wish to use.

Troubleshooting

For general questions about Tumult Whisk and pre-purchase questions, please visit our [FAQ](#) .

HTML

Why does my page not look like I expect?

Whisk's [Web Preview](#) uses WebKit, the same rendering engine found in Safari. For most purposes, the rendering should be identical to Safari. If elements are not rendering or being laid out as expected, it is recommended to check in Safari and also against other browsers using the [Browser Preview](#) feature.

How do I debug the web page?

To debug the rendering of the Web Preview, you can use the [Developer Tools](#) to inspect DOM elements.

JavaScript

How do I see console output or run the JavaScript debugger?

Debugging and seeing JavaScript output can be accomplished through the [Developer Tools](#).

Why do my alerts and prompts not work?

Some JavaScript features that spawn dialogs like `alert()` and `prompt()` do not activate in Whisk.

Why is my page not refreshing anymore?

If your JavaScript code has an infinite loop, the Web Preview will stop rendering. You should correct the fault, save the document, and then relaunch Whisk.

For this reason, it may be prudent to use a manual refresh when working with JavaScript code.

Sometimes infinite loops can be created unintentionally in an intermediate edit.

PHP Code

Why do my server-side variables not work?

Variables like `DOCUMENT_ROOT`, and other `$_SERVER` variables are not available in Whisk's environment. Whisk executes the PHP code through the command-line, and not Apache, therefore there is no server (and no server variables). These variables can be simulated manually in an include file if necessary.

Why did my page stop rendering?

If your PHP code has an infinite loop, the Web Preview will stop rendering. You should correct the fault, save the document, and then relaunch Whisk.

For this reason, it may be prudent to use a manual refresh when working with PHP code. Sometimes infinite loops can be created unintentionally in an intermediate edit.

Reporting Bugs


If you notice issues operating buttons, menus, or are experiencing issues specific to Whisk, please submit a bug report. Including your document and logs will help us find a solution for you quickly and potentially resolve the bug you're reporting for everyone.

To submit a bug report, chose the **Help > Report an Issue...** menu item.

When you do so, please also include your document in the Issue Reporter's Attachments section, and ensure that the "Send logs, preferences, and system information" checkbox is checked.

If you don't fill in the email field, your bug report may be helpful to us, but we won't know how to get back to you with a potential resolution.

Whisk Issue Reporter



We're sorry that you've encountered an issue with Whisk.

Reports sent from this form will reach the developers and directly lead to improvements in the product.

What was your problem?

Whisk did not do X as expected

Ex: "After multiple undos, Whisk cannot render the PHP preview"

Please elaborate (with steps to reproduce, if appropriate):

1. Create a new document

2. Do operation A

3. Do operation B

You'll notice C is not looking as it should. I would expect it to look like D. This happens every time I tried.

Please include as much detail as possible.

Ex: "I noticed that Whisk wouldn't preview my code anymore. Here's how I can reproduce:

1. Use the PHP editing mode

2. Write code with the require() directive

3. Undo several of them and redo a few more

Whisk beeps at me whenever I try to do this."

Attachments

Attach Open Whisk Documents

index.html

test3.html

+v

—

☒ Send logs, preferences, and system information

Attach Whisk documents, screenshots, and other files that may be beneficial when diagnosing the issue.

Contact information (optional)

Name: James T. Kirk

Email: captain@enterprise.com

We may contact you if we need more details.

Cancel

Submit Issue

Whisk's Issue Reporter

The more detail you provide, the easier we can find a fix!

34 of 52

App Security

Whisk is a modern macOS application. As such, it must utilize many platform security technologies to be distributed both on the Mac App Store and via direct download.

Usability and security are classically in conflict with each other: a 100% secure app would not do much. Data does not live in a vacuum. As a user you need to be informed to make the decision on risks and tradeoffs you are willing to accept by using software. This page documents the primary security technologies employed by Whisk and the access the app requires to maintain usability.

App Sandboxing

App Sandboxing [🔗](#) is a security technology Apple introduced in macOS 10.7 restricting the files and hardware an application is allowed to access. Applications must explicitly declare what abilities they need with a system called "entitlements." The App Sandbox can also grant access to files based on intent: if a file is chosen from an **File > Open...** panel or dragged in from the Finder, then the application is granted access.

It may seem that a lightweight editor like Whisk would not need file access beyond this, but Whisk would not be very useful in such a scenario. Consider that a `.html` file as part of a project with many assets. It was granted access by dragging onto the Whisk app icon in the Dock, and therefore gets opened by Whisk. It contains the following markup:

```
<img src = "../mySecondImage.jpg" />
```

In this case, the `../myImage.jpg` asset is a local file not granted access to Whisk, and it will fail to display – the page will appear broken in the Web Preview.

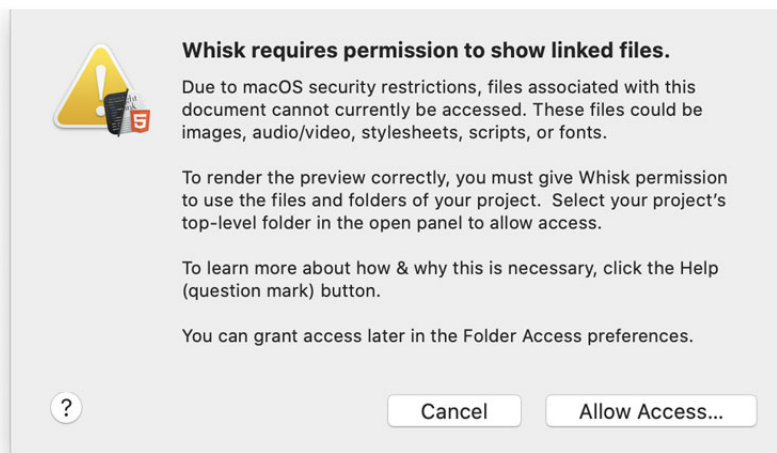
The situation gets more complex with PHP code. Here, PHP can call `require()` or `include()` to load associated files, along with code containing `fread()` and `fwrite()` to read and write files as your program directs.

The arbitrary nature of paths within code coupled with the rapid development Whisk seeks to enable means that individually granting access to required files would become a usability nightmare. Therefore Whisk needs broad filesystem access. This is achieved differently on the Tumult Store and Mac App store version of Whisk.

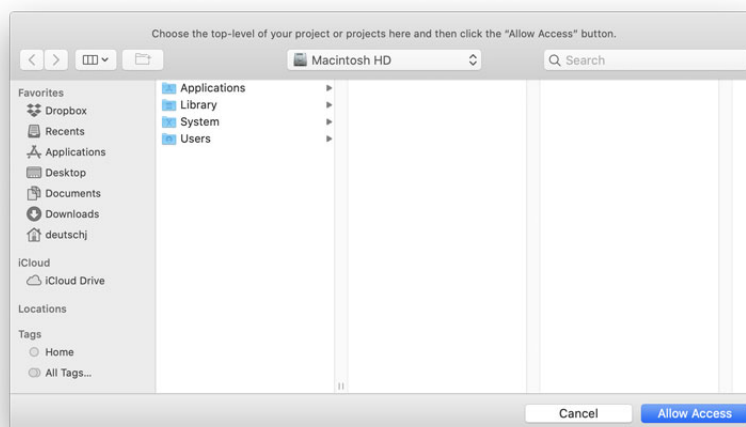
App Sandboxing on the Mac App Store

Since macOS 10.7.4, Apple has required App Sandboxing to be enabled for all applications submitted to the Mac App Store. It is highly unlikely the file system access entitlements Whisk needs (and that the Tumult Store version uses) would ever be approved by Apple's review process. Therefore to function, users are required to grant permissions in the implicit manner of opening a folder or volume.

If Whisk detects trying to preview a file it does not have permission to or run PHP, it will prompt for access:



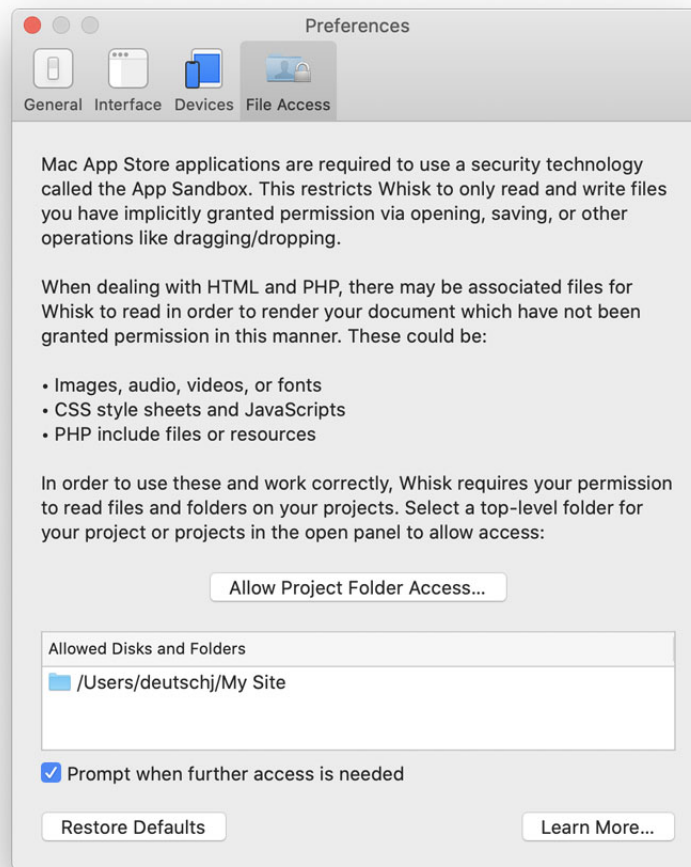
Clicking "Allow Access..." will show a typical Open Panel:



You can navigate to a top-level folder of your project or all projects and then click the "Allow Access" button to give Whisk access to those files. The further up you go, the less prompts you are likely to see in the course of using Whisk. If you navigate to the root of your disk, typically titled "Macintosh HD", and click the "Allow Access" button and then Whisk will function as expected without any

additional prompts.

The Mac App Store version has an additional **File Access** Preference Pane which shows all folders and volumes that you have granted permission to access.

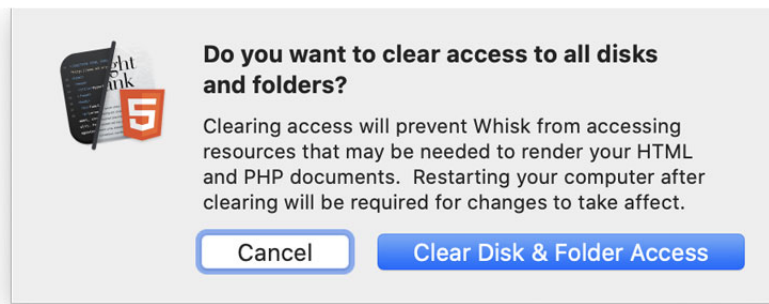


Whisk's File Access Preferences

You can grant access in this preference panel. This is especially useful if Whisk had not autodetected that outside assets are being used.

If you intend to never allow Whisk to have file system access, you can uncheck "Prompt when further access is needed" to never display the access dialog.

If you want to revoke Whisk's file system access, you can do so by clicking the "Restore Defaults" button. A confirmation is required:



Confirmation to revoke file system access

App Sandboxing on the Tumult Store

Direct distribution apps like the Tumult Store version of Whisk are not required by Apple to use the App Sandboxing technology. To improve user security and keep the two versions as close together as possible, the Tumult Store version of Whisk still enables App Sandboxing. To improve common usability, it has these two entitlements:

- The Tumult Store version of Whisk has a full access entitlement to read as your user account to the file system. This is for any files referenced in your code can be rendered by the web preview.
- The Tumult Store version of Whisk has a full access entitlement to write as your user account to the file system. This is for PHP code to use write commands.

While these may sound overly permissive, all non-sandboxed apps can read and write to the filesystem. Thus, this is the case for a vast majority of apps that are not distributed on Mac App Store. Whisk is not privileged beyond the user account in which it is run; standard UNIX file permissions, access control lists, and other macOS technologies will still restrict certain file operations.

Other Entitlements

Both the Tumult Store and Mac App Store versions of whisk also declare these entitlements:

- Read access to HyperEdit (aka Whisk 1.0) preferences and color swatches
- Network client access, as your HTML content may access online resources
- Network server access, because the preview uses a [web server](#) to create network conditions and host previews to browsers installed on your computer

macOS 10.15+ Catalina Files and Folder Access

Whisk is also subject to further OS-level file and folder protections. On macOS 10.15 Catalina and later, Whisk may prompt for access to your Documents, Desktop, or Downloads folders. You can audit this access via the System Preferences' Security Pref Pane under the Privacy tab's "Files and Folders"

section.

WKWebView-based Web Preview

Whisk's web preview renders content using the macOS component known as [WKWebView](#) . Under-the-hood, this is the same WebKit rendering engine that powers Safari. It is standards compliant and fast.

More specifically, WKWebView is built upon a technology called [WebKit2](#) . The salient feature of WebKit2 is that all web page content is rendered using a separate process than the host application. This improves security through isolation and the separate rendering process can be sandboxed more aggressively than the host application.

Preview Server

To more accurately simulate how your content would behave on its final server, Whisk uses its own web server to host content for the application's web preview and browser-based previews. (The alternative would be to use `file:///` -style URLs which are subject to different [CORS](#) rules and cannot make [XMLHttpRequests](#) .)

By default, Whisk's preview server only allows your machine (`localhost`) to connect; all other attempted connections are rejected. Therefore others on your network or on the wider internet cannot see the content you are working on.

Previews can be made to iPhones or iPads over the network using the [Hype Reflect](#) application. These are initiated by you within the Whisk application. This adds the IP address of the iOS device to the access list, and then the device is allowed to request the page. This access is reset when Whisk is relaunched.

Validation Process

For HTML/XHTML validation, Whisk uses [The Nu Html Checker \(v.Nu\)](#) . This is a Java-based program, and therefore Whisk contains a minimal version of the Java Runtime. The vnu process itself hosts a HTTP server. Whisk connects to this and makes queries with the HTML page and vnu returns a list of issues or errors. This server only accepts connections from your machine (`localhost`).

PHP

To render PHP, Whisk runs the command line `php` application. Its path is specified in Whisk's General Preferences.

The program runs with permissions that are inherited from the [App Sandbox](#). On the Mac App Store version of Whisk, you will need to [grant access](#) to use functionality like `require()`, `include()`, `fread()` or `fwrite()`.

WARNING

Be careful when using PHP with a live preview. Half-written lines of code may be syntactically correct but have seriously wrong consequences. It is often a good idea to change the web preview refresh to be manually triggered.

Direct Download Distribution

The Tumult Store direct download version of Whisk makes use of several OS technologies which ensure you are running a malware-free application which came from Tumult.

Code Signing

Whisk is [Code Signed](#) with Tumult's Developer ID certificate. This ensures that the app comes from Tumult and its contents have not been modified.

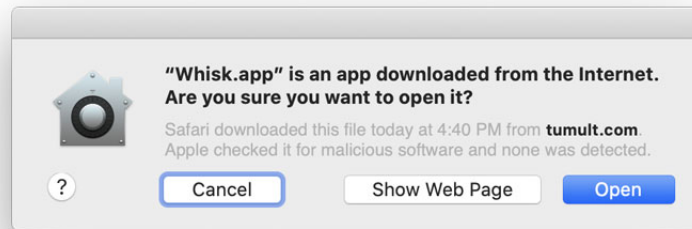
Notarization & Gatekeeper

Whisk is [notarized](#) by Apple. Notarization ensures known malware is not contained in the application. Notarization also imposes other strict app security requirements on Whisk, such as using a [hardened runtime](#).

The basic flow of notarization is:

- The code signed Whisk application is uploaded to Apple's servers
- Apple checks the application for malware and other issues which ensure its security
- If everything checks out, Apple's servers keep a record that the specific bits are okay
- When you launch the application for the first time, [Gatekeeper](#) looks online for this record, and on macOS 10.14.5 or later will only allow running if it is found.

On first launch of the Tumult Store version, Gatekeeper will present a dialog that looks like:



Gatekeeper dialog presented on first launch

The dialog and its wording has changed over the course of macOS releases, but all versions will list the source as coming from **tumult.com**. If it does not state this then the application may have been compromised. "Apple checked it for malicious software and none was detected" appears on some variants and indicates you are running an notarized app.

TIP

Notarization is only checked on macOS 10.14 or later. Prior to this, Gatekeeper will instead display a dialog on first launch regarding Whisk's code signing status alone.


Apple has a support document about how to [Safely open apps on your Mac](#) .

Updating

The Tumult Store version of Whisk uses a 3rd party library called [Sparkle](#) to assist in downloading and installing app updates. Sparkle is widely used by other macOS apps. Whisk updates are distributed over a secure HTTPS connection, and verified with an EdDSA signature.

Keyboard Shortcuts

Whisk is a native macOS app and supports common macOS shortcuts including Command - S to save, Command - V to Paste, Command - Comma to open Preferences, and so forth. This documentation only reflects keyboard shortcuts which are unique to Whisk.

Most keyboard shortcuts can be discovered in Whisk's menu. If a keyboard shortcut does not exist for a menu item, you can customize and add them via the [System Preferences Keyboard Pref Pane](#)  .


Modifiers:

- ⌘ : Command
- ⌥ : Option
- ⌃ : Control
- ⇧ : Shift

General

Action	Shortcut
Preview in External Browser	Command - Return

Text Editing

Whisk uses a standard macOS text view, which has a rich number of keyboard shortcuts, including many emacs key bindings. [This guide from Amsys](#)  has a good list.

[Code Snippets](#) can be inserted using keyboard shortcuts. Please refer to that documentation on how to use and change them.

Action	Shortcut
Outdent block	Command - [
Indent block	Command -]
Go to Line	Command - L

Web Preview

Action	Shortcut
Reload Page	Command - R
Zoom In	Command - Plus
Zoom Out	Command - Minus
Actual Size (100%)	Command - 0

PHP

Action	Shortcut
Show Rendered Web Page in Preview	Command - Option - Left
Show HTML Source Code in Preview	Command - Option - Right

HTML5 Validation

Action	Shortcut
Validate Page	Command - K
Clear Validation	Command - Shift - K
Show Next Error	Control - Down
Show Previous Error	Control - Up

User Interface

Inspector

Action	Shortcut
File Inspector	Command - 1

Tools Inspector	Command - 2
Validation Inspector	Command - 3

Window Management

Action	Shortcut
Show/Hide Source Editor	Command - Shift - 9
Show/Hide Web Preview	Command - Shift - 0
Show/Hide Developer Tools	Command - Shift - I
Show/Hide Inspector	Command - Shift - I
Switch between Widescreen and Landscape Layout	Command - Shift - \

Version History

2.5.0

February 1, 2021

- Apple Silicon support to run natively on "M1" Macs
- macOS 11 Bug Sur compatibility and UI adjustments
- New App Icons matching the macOS 11 Big Sur style
- Localizations for German, French, Spanish, Italian, Japanese, and Simplified Chinese
- Line Comment Selection via Command-Slash
- The tab key will indent code when there is a multi-line selection
- Preference for tab width, infinite tab stops without wrapping, tab stop sizing matches font size
- macOS 11.0+ only: Zoom will magnify like in Safari (page zoom that does not grow the width)
- Color Swatches can have opacity
- Option to disable macOS Auto Save
- Added an optional Save toolbar button
- Data collection policy window shows submission data
- Tweaked background and line highlight colors to match Hype and improve contrast
- Changing to PHP mode will clear out validator errors and changing back to HTML restarts validation
- Fixed issue where showing/hiding line numbers a few times would never show the line numbers
- Validation server Java process now properly terminates when Whisk quits
- Fixed an issue where the source editor font could be blank in preferences
- Restore Defaults in Interface Preferences will reset line spacing
- macOS 11.0+ only: Injected scroll sync code JavaScript does not pollute globals anymore
- Fixed issue where an error was being logged to the console if scrolling before scroll sync could begin
- Fixed issue where Restore Defaults in Interface Preferences would show "System" on systems that do not have that option
- Improved centering of color swatch hex field
- Snippets table scroll bar has better coloring on macOS 10.15 and below in dark theme
- Fixed issue where scrollers would show up as white in dark theme on macOS 10.12-10.13
- Removed table headers and vertical grid from Code Snippets table
- Tweaked visual when dragging Color Swatches
- Fixed a bug where non-RGB colors could not be chosen
- Removed the Colors toolbar icon, as it is unused
- Mac App Store version looks at receipt upon first launch for licenses

- Mac App Store version no longer uses IAP for Trial
- About Window tweaks
- Update copyrights and credits

2.0.1

May 24, 2020

- Scroll Sync will now work on macOS 10.11 - 10.14
- Scroll Sync will now work if there is no doctype declared
- Fixed issue where Hype Reflect previews would not show images and other local page resources
- Fixed bug where hitting return to accept Kanji with a Japanese input method always created a newline
- Showing HTML Source Code for PHP in Dark Theme will use a black background and white text
- Fixed issue where Watched Files table would flicker when modifying watched files
- Browser preview menu will now show multiple copies with the disk location if the name/version is identical
- Fixed issue where validator messages would only show a single line on macOS 10.11 - 10.12
- Fixed issue where Dev Tools may not show up if the Web Preview is closed
- Fixed issue where splitter resize view would not be shown if Dev Tools are open
- Fixed issue where detaching the Web Preview with Dev Tools open would result in a blank spot in the window where Dev Tools should go
- Fixed issue where preferences plist file could get unnecessarily large
- Email will now attempt to be automatically filled when trying to join Mailing List
- Trial purchase button no longer covers window titlebar buttons on on RTL languages
- Validate page button is disabled when starting up the validator
- Validation is disabled for JavaScript and PHP modes
- Zoom menu and Reload toolbar buttons are disabled when the web preview is not shown

2.0.0

May 4, 2020

- Renamed from HyperEdit to Whisk!
- Requires macOS 10.11 or later, supporting macOS 10.15 Catalina (and 64-bit)
- Complete UI refresh including integrated inspector and new icons
- Scroll Sync
- Dark theme
- Preview in different browsers and to Hype Reflect
- Watched Files automatically discovers associated files and reloads the preview when they change

- Updated syntax highlighting to color keywords added in current language versions
- Better macOS multi-document tab bar support
- Support for macOS Versions/Autosave feature
- HTML5 validation (replacing HTML4 validation)
- HTML and PHP editing will syntax highlight JavaScript and CSS code blocks
- Ability to save all document settings for new documents
- Document settings are preserved across launches
- New Preferences window where settings take effect immediately
- Zoom menu for Web Preview (replaces the smaller/larger text buttons)
- Web Preview uses WKWebView for improved performance and crashes/hangs don't take down the app
- Web Preview uses HTTP for more realistic rendering
- New default page is HTML5 and has areas for script/style
- Preference to change line highlight color
- Preference to change editor line height
- Go To Line panel is now a sheet and starts with the line currently on, has steppers
- Color picker uses a popover and can have alpha values
- Color swatches accepts rgba, hex, and color name values
- Code Snippets can have magic variables for `${caret}`, `${title}`, and `${charset}`
- Code Snippets button to insert into the document
- Context menu on splitter to change widescreen/portrait or detach
- Unicode (UTF-8) is now the default encoding
- In-App feedback/bug reports window
- New About window
- New Mailing List prompt
- Improved trial and purchase flow
- Free viewer mode if not purchased/no trial activated
- App Sandbox (Mac App Store version asks for disk access to bypass when needed)
- Notarized app for direct download/Tumult Store distribution
- .dmg for direct download/Tumult Store distribution
- Fixed issue where the line number would not show up on the last line
- Fixed issue where line highlighting would not always extend to the end of the window
- Fixed issue where the Dev Tools would steal focus on each key stroke
- Fixed issue where the Dev Tools may not show the first console logs on refresh
- Snippet Editor is now resizable
- Fixed issue indenting/outdenting from first character in a line
- Fixed Indent/outdent not working with undo
- Fixed issue where replace and undo/redo would not syntax highlight
- Fixed a common syntax highlighter crash

- New documents get a pre-populated file extension based on editing mode
- Added a tool tip for the refresh delay to make it clear it is in seconds
- Removed "smart" text insertions in the snippet editor
- The delete key will clear snippet keyboard shortcuts
- Better autodetection for text encodings to prevent documents from opening blank

1.6.1

January 6, 2011

(Mac App Store-only release)

- Support for the Mac App Store
- Option to highlight current line with insertion point
- Higher resolution app icon
- Fixed issue where HTML5 SQL databases could not be initialized
- Fixed bug where linked files would not be saved when hitting the plus button
- Syntax highlighter works better with international characters
- Syntax highlighter is much faster/memory efficient

1.6.0

April 30, 2008

- Improved Leopard compatibility
- Fixed many memory leaks
- Preview menu has browser icons
- New Javascript console using the WebKit Inspector (if Safari 3.1 is installed)
- Added preference to display invisible characters
- Added Text menu item to convert text to upper and lower case
- New toolbar icons (thanks Marc Edwards of iSlayer!)
- Added automatic checking for updates (via Sparkle, thanks Andy Matuschak!)
- Added ability to report crashes (via UKCrashReporter, thanks Uli Kusterer!)
- Fixed an issue saving window layout when the preview is collapsed
- Fixed vital information window ordering
- Fixed bug where the document was not being marked as dirty
- Support for localizations

1.5.3

November 29, 2007

- Fixed memory leaks
- Fixed issue where dragging an image to the editor would give the wrong width/height
- Option to continuously spell check
- Leopard Compatibility

1.5.2

September 18, 2006

- Universal Binary

1.5.1

March 6, 2006

- Fixed several encoding issues
- HTML entity coloring
- Files in the linked-file drawer can be opened by double-clicking on them
- HyperEdit does a better job of cleaning up temporary files
- Restored Mac OS X 10.2.8 support
- Color palette remembers its state
- Revert sheet is now properly dismissed
- Fixed a bug in properly loading files with a ".htm" extensions
- Various other bug fixes

1.5.0

June 20, 2005

- New color swatches palette allows inserting colors and saving favorites
- The web preview can be torn off into a separate window
- Linked files for a document are preserved
- The document can be previewed in different browsers
- Dragging images into the editor will generate an HTML image tag
- Snippets can be dragged to and from the editor
- Validation errors can be skimmed with control-up/down
- Preference to not create blank documents automatically
- Preference to not load the web preview when a file is opened
- Find/replace correctly updates the web preview
- Resizing the document will only resize the web preview
- Disabled "smart delete" when pasting

- Fixed bug where the Validator would think the user was not an administrator
- Fixed hangs and crashes related to Mac OS X 10.4 Tiger
- Other bug fixes and minor performance improvements

1.0.3

July 22, 2004

- Fixed bug where validation would not work, and say it was improperly installed
- HyperEdit Help (Command-?)
- Error skimming with Control-Up/Down
- Documents created with HyperEdit will now open with HyperEdit
- Find panel will find next with the return key
- New HyperEdit document icon and tweaked application icon
- Fixed some typos

1.0.2

June 15, 2004

- Much improved stability and many bugs fixed
- Fixed bug in syntax highlighting where colors would be shifted over, or incorrect
- Command-Shift-Right/Left is restored to its usual purpose
- Opens files properly from dragging onto the dock or application icon
- Does not use package installer anymore and overall size decreased
- Individual editor background color can be changed by dragging a color swatch into the line number gutter

1.0.1

May 20, 2004

- Fixed live validation by including appropriate libraries with the installer (optional)
- Fixed Mac OS X 10.2 support
- Added Latin-2 encoding
- Added link to visit HyperEdit website in the Help menu.

1.0.0

May 17, 2004

- Live HTML W3C-based validation with red underlines and an error drawer
- View HTML source code rendered from PHP
- Faster, more colorful, customizable Syntax Highlighter
- HTML and PHP rendering performance significantly increased
- Regular expression find and replace (thanks to OgreKit)
- Text Encoding engine improvements and support for more encodings
- Shifting right and left of text blocks
- Snippets palette supports groups, rearranging, window resizing
- Preview in default web browser
- Go To Line dialog box
- Preference to change line endings to Mac, UNIX, or Windows
- Preference to prevent line wrapping
- Toolbar items for shifting left/right, previewing in browser, and saving (Customize Toolbar to see them all)
- 50% split restoration via key combination Command-)
- Will remember if palettes were open or closed from the previous launch
- Split position will be saved with manual window position save
- New/revised toolbar and application icons
- Fixed bug where PHP documents greater than 8 KB would cause HyperEdit to hang
- Fixed bug where PHP errors would write a lot to the console.log
- Fixed bug where the refresh timer would fire after a document was closed causing HyperEdit to crash
- Fixed bug where the editor's scroll bar would disappear
- Fixed bug where the number view would cause a crash using 2-byte encodings
- Fixed bug where Undo would not refresh the preview pane

1.0b3

December 29, 2003

- New Code Snippets palette
- Startup Items in advanced preferences
- New Syntax Highlighting engine colors HTML, PHP, CSS, and Javascript
- Line numbers appear in the gutter
- Options drawer is now a palette
- Linked Files drawer revamped with file icons and customizable check interval
- Fully reveals and hides the preview by pressing Command-0
- Editing mode/Syntax highlighting determined by file extension
- Remembers window position, size, and split orientation
- PHP include/require statements work correctly

- PHP default path is set to /usr/bin/php
- PHP should not crash if nothing is installed
- Opens and edits files of any type
- Preserves indentation for new lines
- Spell checking is disabled by default

1.0b2

August 18, 2003

- PHP Integration
- Web view options drawer
- Linked files make the web view refresh when they are updated
- Document style sheets (always cached)
- Support for multiple text encodings
- Horizontal split view
- Tabbed Preferences
- Find/Replace Panel
- Preservation of indentation level
- Option to enable/disable cache
- Turned off syntax highlighting for PHP, CSS, and Javascript files
- Fixed toolbar item customization bug
- Fixed bug that did not report correct state of the Javascript console
- Donation notice
- New icon (thank you Jenni Lin)

1.0b1

July 23, 2003

- Initial Release