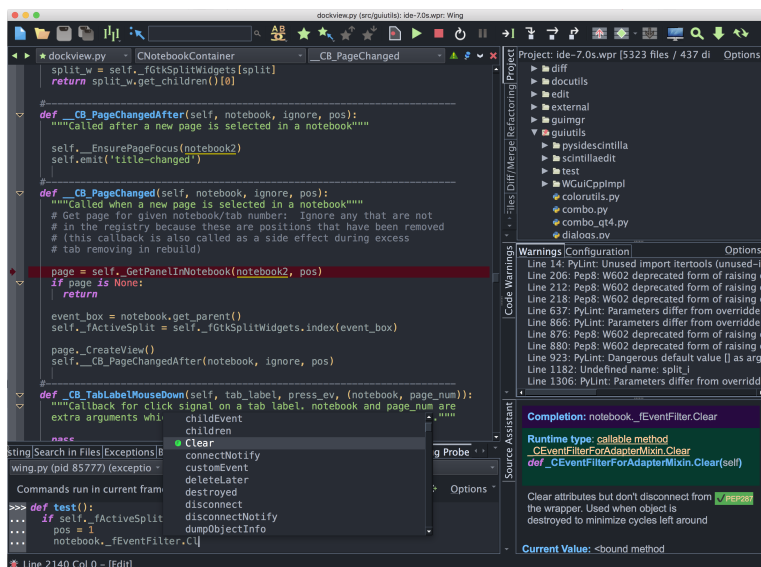




Wing 101 Quick Start Guide



This is a minimalist guide for getting started quickly with Wing 101. For a more in-depth introduction, try the [Tutorial](#).

Wing 101 was designed to be the simplest Python IDE possible that fully supports students learning to program for the first time. If you already know how to program, or if you are using Wing for development outside of introductory course work, you should use Wing Personal or Wing Pro instead. You will be much more productive, and chances are you will have no problem understanding and using the additional features provided by those products.

Let's get started with Wing 101!

Install Python

If you don't already have Python on your system, install it now. Two good options are:

- Obtain the standard Python distribution from python.org
- Use [Anaconda](#) for seamless access to many third party Python libraries. See [Anaconda package lists](#) for a list of the available libraries.

See [Supported Python Versions](#) for other options.

You may need to restart Wing after installing Python, so that it can recognize the new installation.

Configure Python

Wing finds the latest installed Python and uses that with your code. If you want to change which Python is used, set the **Python Executable** using **Configure Python** in the **Edit** menu.

Basic Configuration

You are now ready to start working with code, but may want to make a few configuration changes first:

Display Colors - The **User Interface > Color Palette** preference chooses the colors used in the editor or the whole user interface if the **User Interface > Use Color Palette Throughout the UI** preference is enabled.

Key Bindings - Wing can emulate VI/Vim, Visual Studio, Emacs, Eclipse, XCode, MATLAB, and Brief editors, as selected from **Keyboard Personality** in the **Edit** menu or with the **User Interface > Keyboard > Personality** preference.

Tab Key - The default tab key action depends on the selected keyboard personality and in some cases file type, context, and whether or not there is a selection in the editor. This can be changed from the **User Interface > Keyboard > Tab Key Action** preference.

There are many other options in **Preferences**.

Navigating Code

Wing 101 provides several ways to navigate and search your code:

Source Index menus at the top of the editor provide quick access to other parts of a source file.

Goto-definition is available from the **Source** menu, and by right-clicking on symbols in the editor and , **Python Shell**. Use the forward/back history buttons at the top left of the editor to return from the point of definition.

See [Navigating Source](#) for details on the above.

Toolbar search is a quick way to search the current file. [Details](#)

Search in the **Tools** menu shows the **Search** tool, which provides incremental text, wildcard, and regular expression search and replace in selections and the current file or documentation page. [Details](#)

Editing Code

Wing 101's editor is designed to speed up the process of writing and modifying Python code, and to reduce the incidence of coding errors. Its features include:

Auto-completion in Wing's editor and , **Python Shell** speeds up typing and reduces coding errors. The auto-completer uses **Tab** by default for completion, but this can be changed in the **Editor > Auto-completion > Completion Keys** preference. This feature is disabled by default in Wing 101. [Details](#)

Multiple Selections can be made with **Multiple Selections** in the **Edit** menu, the multiple selections toolbar item, and by pressing **Ctrl+Alt** (or **Command+Option** on OS X) while making a selection with the mouse. Once multiple selections have been made, edits made will be applied to all the selections at once. [Details](#)

Quick Selection operations in the **Edit > Select** menu allow selecting whole statements, blocks, or scopes before copying, editing, or searching through them. [Details](#)

Debugging Code

Wing's debugger is a powerful tool for finding and fixing bugs, understanding unfamiliar code, and writing new code interactively. You can launch code from the **Debug** menu or toolbar, [from the Python Shell](#), or from outside of the IDE either [on the same machine](#) or [on another host](#).

[Breakpoints](#) can be set by clicking on the breakpoint margin to the left of the editor. [Stepping operations](#) are in the **Debug** menu and toolbar.

The **Stack Data** tool is used to inspect or change program data. Hovering the mouse over a [symbol in the editor](#) shows the value for that symbol in a tooltip, if available on the active debug stack.

Debug process I/O is shown in the **Debug I/O** tool, or [optionally in an external console](#).

Other Features

Several other features are available to assist in your Python coding:

Python Shell -- Wing's **Python Shell** lets you try out code in an independent sandbox process. To enable debugging, click the bug icon in the top right of the **Python Shell**. The shell provides auto-completion if it has been enabled in preferences. [Details](#)

Preferences in the **Edit** menu (or **Wing101** menu on OS X) gives you control of the overall layout and color of the IDE, among many other options. Right click on tool and editor tabs for layout options, or drag tabs to move them or create new splits. Right-click on the toolbar to configure which tools are visible or to add your own. See [Customization](#) for details.

Further Reading

As you work with Wing 101 on your own software development projects, the following resources may be useful:

- [Wing Support Website](#) which includes a Q&A support forum, mailing lists, documentation, links to social media, and other information for Wing users.
- [Wing Reference Manual](#) which documents all the features in detail.
- Wing displays a useful tip at startup, to help you continue learning about the feature set over time. These are also accessible from **Wing Tips** in the **Help** menu.

Thanks for using Wing 101!