|  | ***Frigcal*** — Personal Calendar GUI; No Login Required |
| --- | --- |

Top

| Summary: | A stand-alone desktop GUI, with basic calendar functionality, portable calendar files, and privacy. |
| --- | --- |
| Version: | [2.0.1](http://docs.google.com/docetc/DeveloperGuide.html#release201), August 2017 (see all version [history](http://docs.google.com/docetc/DeveloperGuide.html#s7)) |
| Author: | © M. Lutz, 2014-2017, [learning-python.com](http://learning-python.com) |
| License: | Provided freely, but with no warranties of any kind (see also [README.txt](http://docs.google.com/README.txt)) |
| Screenshots: | This program's GUI and scripts run on [Mac OS](http://docs.google.com/docetc/docimgs/macosx/index.html), [Windows](http://docs.google.com/docetc/docimgs/windows/index.html), and [Linux](http://docs.google.com/docetc/docimgs/linux/index.html) |
| Usage: | [Download](http://learning-python.com/frigcal.html) and [start](http://docs.google.com/README.txt) source code, Mac app, and Windows and Linux executables |

This is the most recent version of this document, styled for viewing on both desktop and mobile devices, and available both online and in the source-code package. In app and executable packages, you can replace the original version shipped by saving this page's HTML in the install folder with your browser.

***New***: see the separate how-to for using [Frigcal on Android](http://learning-python.com/using-tkinter-programs-on-android.html).

### Contents

* [**Introducing Frigcal**](#_30j0zll)
  + [Frigcal's Features](#_1fob9te)
  + [Frigcal for the Impatient](#_3znysh7)
* [**Using the GUI**](#_2et92p0)
  + [Window Layout](#_tyjcwt)
  + [Border Control Actions](#_1t3h5sf)
  + [Day-Grid Actions in 'mouse' Click Mode](#_lnxbz9)
  + [Day-Grid Actions in 'touch' Click Mode](#_35nkun2)
  + [Selection List Actions in Both Modes](#_1ksv4uv)
  + [Other GUI Actions and Usage Notes](#_44sinio)
* [**Using the Program**](#_1y810tw)
  + [Program Files](#_4i7ojhp)
  + [Data Files](#_2xcytpi)
  + [Configurations File](#_1ci93xb)
  + [Portability](#_2bn6wsx)
  + [Dependencies](#_qsh70q)
* [**Using Calendars**](#_3as4poj)

# Introducing Frigcal

|  |
| --- |

Welcome to ***frigcal***—a program that allows you to enter and manage your personal events in a GUI that's designed to be as easy to use as a traditional "refrigerator" calendar. Frigcal works the same on Mac OS, Windows, and Linux, per the screenshots above; is available as a Mac app, Windows and Linux executables, and portable source code; and stores your calendar events in a standard format that can be used across platforms and programs. Best of all, you don't need to login to an online account to use this program; with frigcal, your calendar events are your business, not content to be monetized by others.

This document is frigcal's user guide. It covers the basics of using the GUI, program, and calendar files. This section provides a quick first look at frigcal and its features, and the bare minimum on usage details. If you're looking for install and launch details, see the [README](http://docs.google.com/README.txt) file. If you're looking for developer-oriented details such as version history, see the [Developer Guide](http://docs.google.com/docetc/DeveloperGuide.html).

In this section:

* [Frigcal's Features](#_1fob9te)
* [Frigcal for the Impatient](#_3znysh7)

## Frigcal's Features

|  | The frigcal program implements basic but common calendar functionality in a standalone desktop GUI, which runs and stores your calendars locally on your computer. It's designed to mimic a monthly calendar of the sort you might tack up on your refrigerator (month images and all), with a fundamental emphasis on simplicity and privacy. Among its features: |
| --- | --- |

### ***General***

* Portable iCalendar files for storage of your events
* Multiple calendar files support, and new file creation
* Automatic backup of calendar files, with multiple backups management
* Unicode support for non-English text in both GUI and calendar files
* Portable, open-source program, for Mac, Windows, and Linux
* Fully offline operation that keeps your calendars private
* Provided as self-contained executables and portable source-code

### ***GUI***

* Natural month displays, with easy-to-use navigation
* Display and editing of persistent events, with summaries and descriptions
* Event color coding by both calendar name and event category
* Multiple resizable month-view windows, with independent or tandem navigation
* Month-images display option with customizable images
* Quick event overviews on mouse hovers via footer option
* Event cut, copy, and paste, across days and month windows
* Event selection lists for days with many events to display
* Configurable fonts and colors, event models, calendar paths, and more

Now that you've seen the highlights, you should also know that the frigcal program does not implement more advanced calendaring tools such as task scheduling, journaling, alarms, event invitations, or recurring or multiday events, though some can be emulated manually. For instance, copy-and-paste can be used for recurring and multiday events, and event description text suffices for basic date-based journal entries.

In exchange for this deliberately restricted utility, though, this program minimizes both code and GUI complexity; reduces the risk of data loss; and provides an open and portable interface sufficient for most personal calendar and journaling needs. It does what most of us need a calendar to do, without cluttering the GUI with loads of functionality that most of us will never use.

Perhaps most importantly, as an offline and open-source program, frigcal provides the utmost in *privacy*. It neither requires nor uses a network connection, account login, or proprietary system or device. It never uses your calendars for any purpose other than GUI display. And it provides its full source code so you can be sure that all its activity is above board. With frigcal, your calendar data remains your private asset—not raw material for advertisers (or worse) to covertly scan. Why settle for less in a tool you trust with the moments of your life?

## Frigcal for the Impatient

|  | The remainder of this document provides a comprehensive look at frigcal use, and is recommended reading for the best frigcal experience. If you'd rather dive right in now, though, here's the absolute minimum you need to know to get started: |
| --- | --- |

**Startup**

* Fetch a frigcal package from [this site](http://learning-python.com/frigcal.html) and unzip per the [README](http://docs.google.com/README.txt).
* Start frigcal by running its app, executable, or source-code script per the [README](http://docs.google.com/README.txt).
* Frigcal's appearance and behavior can optionally be customized in [frigcal\_configs.py](http://docs.google.com/frigcal_configs.py).

**Calendars and images**

* Frigcal makes a default calendar file for you on its first run, stored in its [Calendars](http://docs.google.com/Calendars) folder.
* You can store other existing calendar files in the Calendars folder to use them in frigcal.
* Search your calendars with the [searchcals.py](http://docs.google.com/searchcals.py) script: it's not part of the GUI itself.
* Month images are loaded from the [MonthImages](http://docs.google.com/MonthImages) folder: store 12 of your own photos there.
* To upgrade, copy over your frigcal\_configs.py, and Calendar and MonthImages folders.

**GUI: events**

* Single-click a day or event to highlight it.
* Double-click on a day's tile to add an event for that day.
* Double-Click on an event's tab on a day tile to view or edit that event.
* Right-click an event's tab to cut or copy that event (control-click works on Macs).
* Right-click on a day's tile to paste a previously copied event.
* Double-click on a day tile's number to open that day's events in a larger list.
* Events can be deleted after double- or right-clicking their event tab.
* Events can be moved by cut and paste, and replicated by copy and paste.

**GUI: window-border actions**

* Use PrevMo/NextMo (and up/down arrows) to move to other months.
* Use PrevYr/NextYr (and shift+up/down arrows) to move to other years.
* Use Today (and the Escape key) to return to the current month and day.
* Enter a date and press GoTo to jump to that date in the current window.
* Press Clone to create new month windows; Tandem makes them move in synch.
* Press the Footer toggle to view summary details of events you hover over.
* Press the Images toggle to open or close the image for the window's month.

**Saves and/or exits**

* Calendar saves happen only when you click the Main window's close button.
* On close, only changed calendars are saved after backups, and only if approved.
* On close, two dialogs allow you to save only, save and exit, or exit only.
* Calendars are loaded and saved in ICS format, portable to other programs.
* Calendar files are automatically backed-up on saves in the Calendar/Backups folder.
* The Backups folder retains a fixed number of backups, and prunes older files.
* A close in a Popup (Clone) window just closes that window and its image.

Did you get all that? If not, don't worry. In the remainder of this user guide, we'll explore the GUI, program, and calendar files in a bit more detail. Let's get started with the GUI—the main part of your frigcal experience. If you've already installed frigcal, it may help to work along as we go; if not, you may want to jump ahead to the usage basics [section](#_1y810tw) and come back when you're ready.

# Using the GUI

This section provides detailed coverage of the GUI's operation. If you're looking for startup details, see [Using the Program](#_1y810tw) instead. For information on managing your calendar files, see [Using Calendars](#_3as4poj).

For readers not working along yet, it may help to refer to a [screenshot](http://docs.google.com/docetc/docimgs/index.html). One main-window image is included inline below for reference, but there are both clickable thumbnails and links to relevant examples along the way—view this document with images enabled in your browser for the best experience, and open the embedded images in a new window for easier viewing. To save space, the referenced images are all from [Mac OS](http://docs.google.com/docetc/docimgs/macosx/index.html) only (technically, the frozen Mac app), but you can view their [Windows](http://docs.google.com/docetc/docimgs/windows/index.html) and [Linux](http://docs.google.com/docetc/docimgs/linux/index.html) equivalents in the screenshots folder.

Also keep in mind that this GUI was designed to be used on both [touch](#_35nkun2) and [mouse/keyboard](#_lnxbz9) devices, and this influences both its action triggers and behavior in general. The GUI's operation, for example, includes interactions both common to and differing between selectable usage modes. Unless you're using a touchscreen and find the default bindings awkward, though, you can probably use the default mouse-mode setting and skip the touch mode section here.

***Note to Mac OS users***: in this section, "*right-click*" on your platform means either a mouse right-click (if you're using a two-button mouse); a "control"-key + click combination (using a mouse or touchpad click); or a two-finger touchpad press (i.e., click). The control-click binding was added as an accommodation to Macs in version 2.0. Most of these are configurable, so see your System Preferences if events do not fire as described here. Also unique to your platform, some of your dialogs slide down instead of popping up, and you have a minimal top-of-screen [menu](http://docs.google.com/docetc/docimgs/macosx/mac-menu.png) for help and quit plus a clickable Dock icon for deiconify and quit (the latter is similar to app- and task-bars on other platforms).

In this section:

* [Window Layout](#_tyjcwt)
* [Border Control Actions](#_1t3h5sf)
* [Day-Grid Actions in 'mouse' Click Mode](#_lnxbz9)
* [Day-Grid Actions in 'touch' Click Mode](#_35nkun2)
* [Selection List Actions in Both Modes](#_1ksv4uv)
* [Other GUI Actions and Usage Notes](#_44sinio)



## Window Layout

|  |  | The GUI's top level is composed of one or more month view [windows](http://docs.google.com/docetc/docimgs/macosx/composite-2.png), which may be opened on the same or differing months, and moved to new months and years either independently or in tandem. Month windows in turn open dialogs for event view/edit, creation, list selection, cut/paste, and so on, as well as associated images. |
| --- | --- | --- |

[Month windows](http://docs.google.com/docetc/docimgs/macosx/main-window-may-footer.png) use a resizable display that always shows 6 weeks for consistency and clarity, with event summary text widgets positioned on days arranged in a *grid*, and control widgets placed around window *borders*. Both border controls and day-grid days, numbers, and events invoke actions when selected. An optional [footer](#3rdcrjn) display in the lower portion of the window gives event details on hovers or clicks. Event *summaries*—single-line text that provides short captions—appear on days, in the footer, in event [dialogs](http://docs.google.com/docetc/docimgs/macosx/event-create-window.png), and in the event selection list [dialog](http://docs.google.com/docetc/docimgs/macosx/selection-list.png). Event *descriptions*—multiline text large enough for full details and journal entries—appear in footers, and in event dialogs.

For ease of use on touch interfaces, there is no top-of-window *menu* or event *drag-and-drop*; window menus take space and can be cramped, and drag-and-drop can be nearly nondeterministic on a Windows tablet (there is a top-of-screen menu on Macs but it's minimal, and Mac's don't have touch screens anyhow). Instead, controls appear on window [borders](#_1t3h5sf), and event cut/copy/paste is fully supported as an alternative to drag-and-drop (see [cut/paste note](#z337ya) ahead). For keyboard use, whether physical or on-screen, a set of key bindings are equivalent to date navigation buttons (see [Date Navigation](#2s8eyo1) ahead).

Both month windows and event dialog windows may be freely resized and moved in the GUI. [Month](http://docs.google.com/docetc/docimgs/macosx/main-window-june.png) windows are non-modal (non-blocking); their initial size may be tailored by either absolute pixels or percentage of full screen size, and their initial position may be preset (see [Configurations File](#_1ci93xb)). Event [Create](http://docs.google.com/docetc/docimgs/macosx/event-create-window.png) and [Paste](http://docs.google.com/docetc/docimgs/macosx/event-paste-window.png), [View/Edit](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png), [Cut/Copy/Open](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy.png), and [Selection List](http://docs.google.com/docetc/docimgs/macosx/selection-list.png) dialog windows are all modal (blocking), and opened as needed in response to GUI actions described ahead.

## Border Control Actions

|  |  | The GUI operations in the table below—invoked by controls arranged around month window borders—are always available, regardless of your click mode setting in [frigcal\_configs.py](#_1ci93xb). |  |
| --- | --- | --- | --- |

| Control | Action | Equivalent Key |
| --- | --- | --- |
| PrevYr | Move to previous year in this window, or in all windows if Tandem. | Shift+Up-arrow |
| NextYr | Move to next year in this window, or in all windows if Tandem. | Shift+Down-arrow |
| PrevMo | Move to previous month in this window, or in all windows if Tandem. | Up-arrow |
| NextMo | Move to next month in this window, or in all windows if Tandem. | Down-arrow |
| Today | Move to current date in this window only. | Escape (Esc) |
| GoTo | Move to entered mm/dd/yyyy date in this window only. | Enter in text |
| Clone | Open a new month view window. | N/A |
| Tandem | Enable/disable synchronized moves for all month windows. | N/A |
| Images | Open/close month image associated with this window's month. | N/A |
| Footer | Open/close event summary text display at bottom of this window. | N/A |
| ? | Open this help document in a web-browser window. | N/A |
| *Window quit* | Close clone, or backup/save calendar files and/or exit program. | N/A |
| *Window minimize* | Minimize this month window and its image, later restored as a pair. | N/A |

The following list provides more details on border control actions.

**Quit—Save and/or Exit**: The main window's quit button—a.k.a. *close*, and rendered as an "X" at the top-right corner on Windows and Fedora Linux, and a small red circle at the top-left corner on Mac and Ubuntu Linux—is used for both kicking-off a calendar files backup/save operation, and ending the program. A quit button press on a **Clone** popup window, described [ahead](#17dp8vu) in this list, silently closes its window only. A quit button press on the main window (denoted by the "**Main**" in its title text) invokes a two-step process that lets you save changes and/or exit, and issues one or two dialogs:

1. The [first dialog](http://docs.google.com/docetc/docimgs/macosx/close-dialog-1-save.png) verifies ".ics" calendar files backup and save (and is skipped if no calendars have been changed).
2. The [second dialog](http://docs.google.com/docetc/docimgs/macosx/close-dialog-2-quit.png) verifies program exit (if all backups and saves worked)

Hence, after clicking the main window's quit button:

* To backup and save without exiting, confirm the first dialog and cancel the second.
* To exit without saving, cancel the first dialog then verify the second.
* To both save and exit, confirm both dialogs.

See [Using Calendars](#_3as4poj) for more on calendar saves and backups.

*Mac users*: you can also access the quit operation in your application (leftmost) [menu](http://docs.google.com/docetc/docimgs/macosx/mac-menu.png). In this form, Quit is the same as clicking the main window's close button, and invokes the two-step save-and/or-exit process. The Dock item's right-click Quit has the same effect.

**Date Navigation—Buttons and Keys**: These widgets appear at the top and bottom of the month [window](http://docs.google.com/docetc/docimgs/macosx/main-window-may-footer.png). They move the window's display (and its open [image](#26in1rg)) to:

* Previous/next month or year: **PrevMo**, **NextMo**, **PrevYr**, **NextYr**
* Today's date: **Today**
* An entered date: **GoTo**

The keys ***Up-arrow/Down-arrow*** and ***Shift+Up-arrow/Shift+Down-arrow*** also go to previous/next month and year, respectively, but are simply redundant with navigation buttons at window bottom, and not required of touch users, whose on-screen keyboards may be incompatibly large. (Left and right arrow keys are reserved for text edits in event summary fields.) Whether invoked by buttons or keys, previous/next month and year navigations are applied to all open month windows (and their images) if **Tandem** is enabled, per the next bullet item.

Also as conveniences for keyboard users: an ***Enter*** key press in the **GoTo** date input field is the same as pressing the **GoTo** button—moving to the entered date; and as of version 1.5 the ***Escape*** key (a.k.a. "Esc") is the same as pressing the **Today** button—moving to the current date. Whether invoked by buttons or keys, both of these operations always effect the currently active window only.

**Clone Button and Tandem Checkmark**: The **Clone** button opens a new, independent month view window, and selecting **Tandem** causes all open month windows (and their open [images](#26in1rg)) to navigate in sync on a previous/next month or year action (only) in any open month window. The Clone option supports [multiple month views](http://docs.google.com/docetc/docimgs/macosx/composite-2.png). Each month window can be open on a different month—open 2 to view the current and next or prior months together; open 3 to view the current, prior, and next months at once; open 6 to see half a year if your screen is up to the task.

Month windows move through months either independently or in synch. If **Tandem** is toggled *on*, all open windows respond to any previous/next month or year navigation, and thus move together. If **Tandem** is *off*, only a single window is moved when its navigation controls are used. A **Tandem** setting in any window is GUI-wide, and automatically propagated to all month windows. All open month windows also reflect *updates* made in any: if more than one are viewing the same month, all are updated immediately for any change made to their month in any window. As mentioned [earlier](#4d34og8), only a *quit* in the main window (denoted by the "**Main**" in its title text) ends the program; other windows close themselves only.

**Footer Checkmark**: This toggle shows/hides the event overview-text display area near the [bottom of the window](http://docs.google.com/docetc/docimgs/macosx/main-window-may-footer.png). This display is filled with event date, summary, and description when the mouse cursor enters an event's summary field, or the field is single-clicked/tapped (see click mode actions ahead in this section). Its text is not erased on mouse exit, as some text requires scrolling. Its text is also not erased on month changes, as its prior content may be useful (this is [configurable](#_1ci93xb) as of version 1.2.3). Footer text is *per-window*: each open month window can display a different event's data in its footer.

Because it gives more details than an event's summary widget, but less than the full View/Edit event dialog, the footer display is mostly a convenience, designed to minimize full dialog opens. As of version 1.2, the footer's text is *read-only*, as no changes to it would be applied; on some platforms you can still copy its text to paste elsewhere (via mouse highlighting plus Ctrl-C on Windows), but no edits are allowed.

Note that the footer's scrollbar may be difficult to reach without entering another event, and, due to month window size constraints, footer text may not be displayed at all both on small screens and in months with many events in any given day. To see more footer text, try expanding the month window, or changing the footer's font, size, and resizing options in the [configurations file](#_1ci93xb); or simply left-click an event to open the full event View/Edit dialog as needed.

**Images Checkmark**: This toggle shows/hides an image file associated with the window's displayed month in a [separate window](http://docs.google.com/docetc/docimgs/macosx/composite.png). Month images change automatically as you navigate through months, and are automatically minimized and restored with their month windows. Like footers, images are *per-window*: each open month window can display a different month image in its own image popup. As noted [earlier](#17dp8vu), selecting **Tandem** causes all open image windows to be updated simultaneously on month moves (i.e., previous/next month and year navigations) along with their month windows.

Images are loaded from the [images folder](#_2xcytpi), in which an image filename's relative sort order among the 12 image files present gives the image's relative month number (e.g., the first image file in the folder by ascending sort order is used for January). The image files folder's location defaults to MonthImages in the program's folder, and may be customized in the [configurations file](#_1ci93xb) (see its setting "imgpath"). To use images of your own, either change the default MonthImages folder's content in-place, or change the configurations file to point to a different folder where your images are stored.

To enable the optional images feature when using the frigcal source-code package (only), you must install the Pillow extension *unless* you are using PNG images with a Python using Tk 8.6+ (including standard Python 3.4+ on Windows), or GIF or PPM/PGM images with any Python 3.X (see [Dependencies](#_qsh70q)). This feature is basic by design; images are simply displayed in a separate window, synchronized with the associated month window. For speed no auto-resizing is performed, so this works best if images are all the same size, and small enough to display well in a popup window.

See also [Future Directions?](http://docs.google.com/docetc/DeveloperGuide.html#s6c) for ideas on this topic, and version [1.4](http://docs.google.com/docetc/DeveloperGuide.html#release14), [1.5](http://docs.google.com/docetc/DeveloperGuide.html#release15), and [1.6](http://docs.google.com/docetc/DeveloperGuide.html#release16) changes which make the images feature more usable.

**Help Button**: As of version 1.2, the "**?**" button at top right in month windows opens this document in a web browser from within the program. *Mac users*: you can also reach this document from the help entries in your top-of-screen [menu](http://docs.google.com/docetc/docimgs/macosx/mac-menu.png).

Additional GUI actions span click modes but are invoked by clicks in the day grid, and described in the next three sections. Among these, *cut and paste* operations invoked by right-clicks on events and days may be used to move or copy events to other days and month windows; and the *event selection list dialog* opened by left-clicks on day-number areas above events is useful when a day has too many events to display. See the next three sections for action triggers, and [Other GUI Actions and Usage Notes](#_44sinio) for more on cut-and-paste and event selection lists.

## Day-Grid Actions in 'mouse' Click Mode

|  |  | The additional event triggers and actions in the table below are defined for calendar days and events in the GUI's day-grid area, when your "clickmode" setting is 'mouse' in [frigcal\_configs.py](#_1ci93xb). This multi-click mode the preset default, and is generally suitable for devices with a mouse and keyboard. |  |
| --- | --- | --- | --- |

| Trigger | Action |
| --- | --- |
| Single left-click (tap) event | Select event field for summary text edits (not updated until Enter pressed). |
| Enter key in event summary text | Update event's possibly edited summary text only. |
| Single left-click (tap) day or day-number | Move the current-day shading to the selected day (other events move shading too). |
| Double left-click (tap) day | Open [Create](http://docs.google.com/docetc/docimgs/macosx/event-create-window.png) event dialog for new event on day. |
| Double left-click (tap) day-number | Open event [selection list](http://docs.google.com/docetc/docimgs/macosx/selection-list.png) dialog for all events in day [see [ahead](#_1ksv4uv) for actions]. |
| Double left-click (tap) event | Open [View/Edit](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) dialog for event. |
| Single right-click (tap+hold) event | Open [Cut/Copy/Open](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy.png) options [menu](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy-menu.png) for event. |
| Single right-click (tap+hold) day or day-number | Paste the latest cut/copy event in this day, by opening a prefilled [Create](http://docs.google.com/docetc/docimgs/macosx/event-paste-window.png) event dialog. |
| Mouse hover-in on event | Display [footer](#3rdcrjn) text if enabled. On tablets, single-press or stylus may activate this. |

## Day-Grid Actions in 'touch' Click Mode

|  |  | The additional event triggers and actions in the table below are defined for calendar days and events in the GUI's day-grid area, when your "clickmode" setting is 'touch' in [frigcal\_configs.py](#_1ci93xb). This single-click mode is generally suitable for devices with only touch screens, but some users may find it useful in other contexts. This mode must be enabled; if the former section's 'mouse' bindings suffice, you can skip this topic. |  |
| --- | --- | --- | --- |

| Trigger | Action |
| --- | --- |
| N/A | *There is no in-line edit of displayed summary text: use left-click* [*View/Edit*](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) *dialog.* |
| Single left-click (tap) day | Open [Create](http://docs.google.com/docetc/docimgs/macosx/event-create-window.png) event dialog for new event on day. |
| Single left-click (tap) day-number | Open event [selection list](http://docs.google.com/docetc/docimgs/macosx/selection-list.png) dialog for all events in day [see [ahead](#_1ksv4uv) for actions]. |
| Single left-click (tap) event | Open [View/Edit](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) dialog for event, and display its footer text if enabled. |
| Single right-click (tap+hold) event | Open [Cut/Copy/Open](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy.png) options [menu](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy-menu.png) for event. |
| Single right-click (tap+hold) day or day-number | Paste the latest cut/copy event in this day, by opening a prefilled [Create](http://docs.google.com/docetc/docimgs/macosx/event-paste-window.png) event dialog. |
| Mouse hover-in on event | Display [footer](#3rdcrjn) text if enabled, and if you have a hover action (e.g., stylus or laptop). |

## Selection List Actions in Both Modes

|  |  | The actions in the table below are available in the event selection lists popped up in response to day-number left-click actions in the preceding two sections. This popup intentionally mimics a single, selected day in a month-window day grid. For more details on selection lists, see the next section's first item. |  |
| --- | --- | --- | --- |

| Trigger | Action |
| --- | --- |
| Single left-click (tap) on event-list item | Open [View/Edit](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) dialog for the event clicked. |
| Single right-click (tap+hold) on event-list item | Open [Cut/Copy/Open](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy.png) options [menu](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy-menu.png) for the event clicked. |
| Create button | Open [Create](http://docs.google.com/docetc/docimgs/macosx/event-create-window.png) event dialog for new event on day. |

## Other GUI Actions and Usage Notes

|  |  | This section provides additional GUI-related interaction and usage details. Refer to the preceding sections for more details on the windows and actions described. |  |
| --- | --- | --- | --- |

**Event Selection Lists—Viewing More Events**: The physical size of day frames imposes a limit on the number of events displayed in month windows (just as in a real "refrigerator" calendar, this program's metaphor). As of version 1.3, you can view, edit, or cut/copy all events for a day—including any events off-screen—by **left-clicking** in the *day-number* area above events at the top of any day frame, to open the [event selection list dialog](http://docs.google.com/docetc/docimgs/macosx/selection-list.png). This list's items look and act nearly the same as events in a day frame in a month window: **single-left** and **single-right** clicks on an event in the list open [View/Edit](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) and [Cut/Copy](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy.png) dialogs for the event, respectively, and a **Create** button opens the [Create](http://docs.google.com/docetc/docimgs/macosx/event-create-window.png) event dialog (like a day left-click). For more details, see this dialog's [full documentation](http://docs.google.com/docetc/DeveloperGuide.html#eventselectionlist) in the developer release notes for version 1.3, where it was introduced.

**Cut/Copy/Paste—Moving Events**: Events may be moved and copied to other days and months as follows: **right-click** on an *event* to cut or copy (via a pop-up actions menu); then **right-click** on any *day* or *day number* to paste the most recently cut/copy event on that day, via a prefilled [Create](http://docs.google.com/docetc/docimgs/macosx/event-paste-window.png) dialog, which allows changes prior to pasting. The event right-click Cut/Copy menu dialog and menu appear at the spot clicked, and are captured [here](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy.png) and [here](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy-menu.png). Per the preceding note, right-clicks on event items in the event selection list dialog open the Cut/Copy menu too, just as they do on events in day frames (though pastes are invoked only on day frames).

**More on Pastes**: A cut/copy event can be pasted to a day in any open window, and as many times as you wish. That is, you can paste a cut or copied event multiple times to add it to many days—for instance, to manually propagate a *recurring* or *multiday* event. Right-click the event to [copy](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy-menu.png) it once, then right-click days to [paste](http://docs.google.com/docetc/docimgs/macosx/event-paste-window.png) it on as many days as desired. As for all updates, pastes show up immediately in all windows open on the target month.

**Open on Right-Click**: For convenience, the Cut/Copy menu opened by event right-clicks of the prior paragraphs also has an [Open](http://docs.google.com/docetc/docimgs/macosx/event-cut-copy-menu.png) option, which is equivalent to an event left-click, and pops up the full event [View/Edit](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) dialog. Hence, most work can be done with right-clicks alone (e.g., cut/copy/open in event, paste in day), plus day left-clicks to add new events.

**Event Changes and Deletes**: The event [View/Edit](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) dialog, opened on event left-clicks, allows you to view full event details; remove the event with Delete; or change most event fields and update the calendar for the new data with Update. You can also delete events with a Cut in the Cut/Copy event right-click menu; a Cut also saves the event for a possible later paste, while a View/Edit dialog's Delete does not. As of version 1.4, the event edit dialog's Cancel [verifies the window close](http://docs.google.com/docetc/docimgs/macosx/event-edit-cancel.png) if you've made any changes; Delete and Update do not (and neither does a right-click Cut), but these operations can't discard user inputs, and do not update calendar files immediately. To back out unintended deletes or updates, click the main month window's [close button](#4d34og8) (the "X" or red circle) and opt to not save changes.

**Changing an Event's Calendar**: The View/Edit dialog for an existing event lets you change most event details, but for implementation reasons does not allow you to change the calendar to which the event belongs. However, you *can* move an event to a different calendar via cut/paste—*cut* the event via event right-click, then *paste* it back to where it was via day right-click, changing its calendar in the Paste (prefilled Create) dialog that is issued on the paste.

**Event Colorization**: Event summary widget background color defaults to white, but can be configured in [frigcal\_configs.py](#_1ci93xb)—both for all events in a *calendar*, and for all events sharing a *category* name. Event category colors span calendars, and override any calendar color settings. To colorize events:

* Edit the category and/or calendar color dictionaries in the configurations [file](#_1ci93xb), giving either a color name or #*RRGGBB* hex-code string for color values.
* Enter event category and calendar names in the GUI's View/Edit and Create dialogs opened on event and day clicks, respectively (see prior note for calendar name changes). An empty category name in the GUI picks up either the calendar's color, or the color of a category-dictionary entry whose key is an explicit empty string (this can provide a default category).

New in version 1.7: both the widget background and foreground text can now be colorized, and the [pickcolor.py](http://docs.google.com/pickcolor.py) utility script can be used for custom color selections; see [frigcal\_config.\_base.py](http://docs.google.com/frigcal_configs_base.py) and release 1.7 developer [notes](http://docs.google.com/docetc/DeveloperGuide.html#release17) for more details.

**Summary Text Visibility**: Event summary widget text displayed on days is intended for brief captions only. In-depth notations are instead entered in an event's multiline *description* field. To see more event summary text, try expanding the window and/or changing the day label and other fonts in your [configurations file](#_1ci93xb). You can also see the full summary text in the Footer display (see [earlier](#3rdcrjn)), as well as in the full event View/Edit dialog window opened on event left-clicks (see click mode actions [earlier](#_lnxbz9) in this section). The event selection list dialog's wider display also shows more of the summary (see [earlier](#2jxsxqh) in this list).

**Other Visibility Notes**: This GUI may sometimes *resize* itself based on its content. You can resize any month window manually if its automatic sizing changes undesirably as you navigate to months with more or fewer events displayed. As mentioned earlier, many events per day may also preclude *footer* visibility in a given month, though the full View/Edit dialog shows complete details on demand (see [above](#3rdcrjn)).

**Trace Messages**: Program messages are printed to the console; if you don't care to see these, change the script's **trace** variable; redirect stdout in your shell; or rename the main script to "frigcal.pyw" or set a shortcut to it to be Run/Minimized to suppress the console window altogether. ***Update***: as of version 1.4, you may also run the new **frigcal-noconsole.pyw** script to suppress the console on Windows automatically. ***Update***: as of version 2.0, frigcal "frozen" app and executable formats do not display a console directly, though printed message may be viewed in other ways on some platforms; see [README.txt](http://docs.google.com/README.txt) for usage details.

**Using Unicode symbols and Emojis** frigcal always preserves all Unicode characters in your calendar files' data when they are loaded and saved. Still because *emojis*—Unicode symbols whose codepoints lie outside the BMP range—are not supported by the Tk library used by frigcal, they must be replaced with a � Unicode replacement character for display in the GUI. You also cannot enter them in the GUI's edit fields. For details, see the [version 1.7](http://docs.google.com/docetc/DeveloperGuide.html#release17) developer notes. Despite this limitation, your calendar events' text can still use any of the very many Unicode symbols and language [characters](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png) that fall in the BMP's U+0000..U+FFFF codepoint range, including these: ✓ ☓ ☑ ☒ ☀ ☼ ☉ ★ ☆ ☞ ☜ ☯ ⚐ ♡ ☮ 重 出 ж म ä ☺.

Beyond this section's coverage, there's no better way to learn how to operate a GUI than by running it live. The next section provides install and launch pointers to help get you started.

# Using the Program

|  |  | This section gives basic program usage details, covering its most-important files and user-visible aspects of its operation. it was originally written for (and still primarily addresses) the initial source-code-only frigcal distribution, but has been updated with notes about the standalone app and executable frigcals added in 2.0. For information on calendar files, see instead [Using Calendars](#_3as4poj); for help with using the GUI itself, see [Using the GUI](#_2et92p0). |
| --- | --- | --- |

In this section:

* [Program Files](#_4i7ojhp)
* [Data Files](#_2xcytpi)
* [Configurations File](#_1ci93xb)
* [Portability](#_2bn6wsx)
* [Dependencies](#_qsh70q)

## Program Files

***Version 2.0 update***: as of frigcal 2.0, the program is provided as "frozen" (self-contained) Mac app and Windows and Linux executables, in addition to its original source-code form. The new frozen formats do not require any extra software (including Python); install with a single download and unzip, and run with a simple app or executable click; and better support icons and the like. They also may not include all of the source-code files mentioned in this section, but this may not be important to you. If you're unsure about the choices, you probably want to install the frozen package for your platform if you consider yourself more of a user than a programmer. For frozen package usage details omitted here, see their coverage in the [README.txt](http://docs.google.com/README.txt) file.

This program's source files reside in its download packages' zip files, available at [this index page](http://learning-python.com/frigcal.html). You can view them and other package content both on [your computer](http://docs.google.com/) after extracting the zip file, and in an unzipped online copy maintained at [this site](http://learning-python.com/frigcal-products/unzipped).

To *start* the program, click or otherwise run the Mac app, Windows executable, Linux executable, or source-code package's scripts. For apps and executables, a simple click suffices to start the system, and won't be discussed further here (again, see the [README](http://docs.google.com/README.txt)). For the **source-code package**, run either of the following in the top-level folder (a.k.a. *directory*) of your copy's unzipped content:

* [**frigcal.py**](http://docs.google.com/frigcal.py), the main script, to launch the GUI on any platform.
* [**frigcal-noconsole.pyw**](http://docs.google.com/frigcal-noconsole.pyw) to launch the main script without a console window on Windows.

*Windows users*: You can launch either of these scripts by simply clicking their icons. If desired, drag them out to desktop shortcuts to make them easier to start, and set their shortcuts' icon to "icons\frigcal.ico" in the top-level folder (right-click on the shortcut to Properties). You can also launch from command lines in frigcal's folder: "frigcal.py", "py frigcal.py", or "py -3 frigcal.py".

*Linux users*: A "python3 frigcal.py" command line in frigcal's folder will likely launch the program as is, but you may wish to edit the script's initial "#!" line for your system's install paths and change and DOS end-lines to Unix end-line (see [ahead](#3whwml4)). Some Linux systems may require an extra install for tkinter support ([see here](#_1y810tw)).

*Mac users*: The prior Linux note generally applies to you too, though you can also launch Python scripts by clicking them in Finder (which runs them with the included Python Launcher), making desktop aliases to them (much like Windows shortcuts), and so on.

Other user-related source files located in this program's top-level directory and covered in full later in this section:

* Module [**frigcal\_configs.py**](http://docs.google.com/frigcal_configs.py) has user configurable options, in Python code.
* Script [**makenewcalendar.py**](http://docs.google.com/makenewcalendar.py) creates a new, empty, named calendar on demand.
* Script [**searchcals.py**](http://docs.google.com/searchcals.py) searches your calendars for a string and prints matching dates.
* Utility [**unicodemod.py**](http://docs.google.com/unicodemod.py) converts a calendar (or other text) file to a new Unicode encoding.
* Utility [**pickcolor.py**](http://docs.google.com/pickcolor.py) displays the #RRGGBB string of a selected custom color for configs.
* Utility [**docetc\fixeoln.py**](http://docs.google.com/docetc/fixeoln.py) converts configs (or other) file end-lines to Windows or Unix format.

The first of these two is user-editable and described [ahead](#_1ci93xb). Because the program creates a [default calendar](#2p2csry) automatically, the second of these is an optional step. The third file listed above is a frigcal 2.0 [enhancement](#23ckvvd), and last three are described in [version 1.7](http://docs.google.com/docetc/DeveloperGuide.html#release17) developer notes.

Other ".py" files in the unzipped download package are modules used only by the system itself, not its users. Of these, *programming-inclined users* might be interested in [\_\_sloc.py\_\_](http://docs.google.com/__sloc__.py), used for line-count metrics, and supplemental ".py" example files in the [docetc folder](http://docs.google.com/docetc) not used by the frigcal program. Also for programmers: by *code structure*, the program consists of a primary module with classes that mirror main GUI windows (months and dialogs), plus a module for non-GUI calendar file access, and a set of smaller utility and configuration modules and scripts. Browse the code in the [top-level folder](http://docs.google.com/) for more details.

## Data Files

***Version 2.0 update***: in frigcal's new "frozen" app and executable formats, the files and folders mentioned here are present, but may be embedded in a different install-folder structure. See the [README.txt](http://docs.google.com/README.txt) file for information on these packages' structures.

Calendar files and month image files are stored in folders, which by default are located in this program's top-level directory. These folders may be located elsewhere via [configurations file](#_1ci93xb) path settings. Calendar and backup folders are fully described ahead, in [Using Calendars](#_3as4poj); month image usage is also covered above in [Using the GUI](#26in1rg). In brief:

* The [**Calendars**](http://docs.google.com/Calendars) folder (or one you configure) stores calendar ".ics" files.
* The [**MonthImages**](http://docs.google.com/MonthImages) folder (or one you configure) stores month image files.
* The [**Calendars\Backups**](http://docs.google.com/Calendars) subfolder stores calendar ".ics" file automatic backups.

You can store arbitrary unrelated subfolders and files in any of these three folders. Specifically, the program uses only ".ics" files in the first and third of these, and only image files (of any type) in the second, and ignores any other items present. In the images folder, non-file items are skipped as of version [1.4](http://docs.google.com/docetc/DeveloperGuide.html#release14), and non-image files as of [1.5](http://docs.google.com/docetc/DeveloperGuide.html#release15).

## Configurations File

***Version 2.0 update***: as of frigcal 2.0, the configurations file described here has been split into *two parts* to better support **upgrades**. The new file [frigcal\_configs\_base.py](http://docs.google.com/frigcal_configs_base.py) contains "factory presets" for all configurable options as well as documentation on their values, and the file [frigcal\_configs.py](http://docs.google.com/frigcal_configs.py) is now meant to be used only for all user overrides of the base file's default settings. That is, replace the base file's settings by reassigning its names in the non-base file, and don't edit the base file itself. This way, you can more easily restore your customizations when upgrading to a new frigcal release in the future—just copy your "frigcal\_configs.py" user-edits file into the new install folder and your customizations should be good to go.

To configure this program's appearance and behavior to suit your tastes and usage, edit source-code file [**frigcal\_configs.py**](http://docs.google.com/frigcal_configs.py) in the program's top-level folder. These customizations are optional, but encouraged; frigcal's maker's preferences do not necessarily match yours!

This file supports a wide variety of user customizations—fonts, colors, window size, click modes, user directories for calendars and month images, and more. These settings, coded as simple Python assignments and loaded at program start-up, are described in full in the file's [comments](http://docs.google.com/frigcal_configs_base.py); are shipped with multiple example settings that demonstrate usage; and may be freely changed by users. Whenever possible, this system employs these Python-coded configurations (a.k.a. *configs*) instead of GUI devices, for both flexibility and interface simplicity.

There are more details on the configuration process in the configs file [itself](http://docs.google.com/frigcal_configs_base.py), but a few general usage points are worth noting here.

**Edit with care**: Errors in this file don't terminate the GUI (you'll get a pop-up message with error details), and omissions in this file apply defaults and produce detailed console messages. Still, users are advised to take care with edits: import the file separately to check it for errors, and make a backup of its shipped version. Both *IDLE* and [*PyEdit*](http://learning-python.com/pyedit.html) can be used to check for errors in the file (see the next point).

**Unicode encoding and text editors**: The configs file is shipped in UTF-8 Unicode encoding form as of version 1.7, to both support and demonstrate non-English settings text. Python loads this source-code format by default, and most modern text editors—including Python's *IDLE*, *Notepad* on Windows, *TextEdit* on Mac, and *gedit* and *vim* on Linux—support this common text file format, and usually automatically. The [*PyEdit*](http://learning-python.com/pyedit.htlm) text editor program, from frigcal's makers, also uses UTF-8 as its default encoding. See [version 1.7](http://docs.google.com/docetc/DeveloperGuide.html#configsunicode) developer notes for background details on the encoding change.

As fallback option, in the *very* unlikely event that users cannot edit the configs file in its shipped UTF-8 form, there is a plain ASCII equivalent in the docetc folder [here](http://docs.google.com/docetc/frigcal_configs_base--ascii.py). This file is identical to that in the script's main folder, sans three non-English category names which cannot be converted to narrower encodings like ASCII or Latin-1. If required for edits, copy it up to '..' and rename without its "--ascii" suffix.

You can change this file's content and encoding declaration to use Latin-1 instead, but the shipped UTF-8 version supports a much broader character range. Also note that the included [unicodemod.py](http://docs.google.com/unicodemod.py) utility can convert the shipped UTF-8 configs [file](http://docs.google.com/frigcal_configs_base.py) to a similarly general encoding like UTF-16, but the file's content precludes automatic conversion to narrower schemes like ASCII.

**Linux end-line conversion options**: Like all text files in the frigcal package, the configs file is shipped in either Windows (a.k.a. DOS) or Unix end-line format, depending on the host of the latest edits. This may be inconvenient for some users on other platforms. The portable [*PyEdit*](http://learning-python.com/pyedit.html) editor and *gedit* on Linux both handle Windows end-lines automatically (Mac's *TextEdit* handles them on input, but may mix end-line types on saves). Well-known Unix conversion techniques convert end-lines too, including the *dos2unix* shell command, and *vim* text-editor commands such as ":e ++ff=dos" and ":set ff=dos". Some commands may require an install, however, and editor techniques can be complex.

To address this, as of version 1.7 frigcal ships with a Python-coded end-line conversion script, [fixeoln.py](http://docs.google.com/docetc/fixeoln.py), which can be used to convert end-lines in text files of any Unicode encoding—ASCII, UTF-8, Latin-1, UTF-16, UTF-32, and others—when no other tools are available or convenient. The configs file can be converted from Windows to Linux end-line format by this script with a simple console/shell command of this form (the "utf8" at the end of this is the optional default encoding, which works for ASCII files too):

~/Downloads/frigcal/docetc$ **python fixeoln.py tounix ../frigcal\_configs.py utf8**

As of version 2.0, a companion script, [fixeoln-all.py](http://docs.google.com/docetc/fixeoln-all.py), can also be run to convert *every* text file in the package to the end-line format of your platform in a single run:

~/Downloads/frigcal/docetc$ **python fixeoln-all.py tounix .. utf8**

For more details, see the [version 1.7](http://docs.google.com/docetc/DeveloperGuide.html#configsendlines) developer note, the two scripts' docstrings, and the test [run log](http://docs.google.com/docetc/fixeoln_docetc/fixeoln-test.txt).

## Portability

***Version 2.0 update***: the new app and executable frigcal packages also muddle the portability story somewhat. frigcal is still portable to Mac, Windows, and Linux; but its apps and executables each run only on the single platform for which they are meant. The source-code package runs on all three platforms, but requires a separate install of a platform-specific Python on each target platform. This section summarizes general portability; see the [README](http://docs.google.com/README.txt) for per-package details.

Because of the programming language used for its implementation, this program is generally as portable as Python 3.X and its standard tkinter GUI toolkit—which run on most desktop-metaphor computers. In more detail regarding platform, language, and data portability, this program:

* Is known to run on [Mac OS](http://docs.google.com/docetc/docimgs/macosx/index.html) (El Capitan, Sierra, High Sierra) [Windows](http://docs.google.com/docetc/docimgs/windows/index.html) (7, 8, and 10), and [Linux](http://docs.google.com/docetc/docimgs/linux/index.html) (Ubuntu and Fedora, using Gnome).
* Is known to run under Python **3.3**, **3.4**, and **3.5** specifically, but any **Python 3.X** version is likely to suffice (see also the next section).
* Uses and generates ".ics" **iCalendar** text files in the **UTF-8 Unicode** encoding that are portable to other calendar programs which support this standard (see [Using Calendars](#_3as4poj) ahead).

## Dependencies

***Version 2.0 update***: the new Mac app and Windows and Linux executable formats of frigcal are *completely self-contained*: all dependencies are included, and no additional installs are required (including Python). This section pertains mostly to the source-code package, and can be skimmed or skipped by other-package users.

As shipped, the *source-code* version of this program works if you install just Python 3.X. In full detail, it relies on the following components:

1. [*Required*] **Python 3.X**, as well as its standard library's **tkinter** module for its GUI.  
   This program is written entirely in the **Python** programming language—specifically Python 3.X, where "X" means the latest in the version 3 line. If it's not already installed on your computer, Python 3.X can be fetched from the following site (for Windows, simply get and run the latest 3.X self-installing executable there):  
     
   [https://www.python.org/downloads](https://www.python.org/downloads/)  
     
   **tkinter** is included with many Pythons—including all those installed with the python.org Windows installers—so this is often not a separate fetch/install step. If tkinter doesn't work for you on Ubuntu Linux, try a "sudo apt-get install python3-tk tk" (and similar on Fedora). Mac OS users may wish to [read this](https://www.python.org/download/mac/tcltk/); it currently recommends installing a separate Tk 8.5, though 8.6 can be used with other Python [distributions](https://brew.sh/).  
   To date, this program has been developed and tested under Python **3.3**, **3.4**, and **3.5**; while compatibility is expected for future 3.X releases (and perhaps beyond the 3.X line), it naturally cannot be guaranteed. If you experience anomalies in later releases, try using 3.3 through 3.5.
2. [*Included*] The 3rd-party **icalendar** parser/generator package, which in turn requires **pytz**.  
   These platform-neutral third-party packages are both required to load and save calendar files. For convenience, open source versions of both are *included* with this program and used automatically, and need not be fetched or installed separately. If ever required, you can see and fetch these packages and their docs at:  
     
   <https://pypi.python.org/pypi/icalendar>   
   <https://pypi.python.org/pypi/pytz>   
   <http://icalendar.readthedocs.org>  
     
   frigcal owes thanks to these packages' developers for shortening development time and cost. ICS file parsing and generation are low-level tasks, but these systems have "just worked" for years of real-world frigcal calendaring.
3. [*Optional*] The 3rd-party **Pillow** (formerly known as PIL) package.  
   This image-processing package is required *only* if you wish to use the program's optional month [Images](#26in1rg) feature for some image types. frigcal still works without Pillow; if not installed, unsupported month images are simply disabled in the GUI (and you'll get a popup message if you select them). Because Pillow's installed code may vary per platform and Python version, it must be installed separately if some image types are used. Commonly used Pillow self-installers for Windows and Python 3.X are included and documented in frigcal's [dependencies](http://docs.google.com/dependencies) folder for convenience (click to run), but see the following if these don't apply to your platform or version, or fail on your machine:  
     
   <https://pypi.python.org/pypi/Pillow>  
     
   ***Update***: as of frigcal [version 1.6](http://docs.google.com/docetc/DeveloperGuide.html#release16), depending on your image file types and Python version, a Pillow install may **no longer be required** to use the optional month images display feature. Month images now display with *just Python itself* if you use either PNG images and a Python that uses Tk 8.6 or later (including Python 3.4 and later with the standard Windows installer at python.org), or GIF or PPM/PPG images and any Python 3.X. For all other combinations of image type and Python release, Pillow is still required for the image display feature, and you'll still get a popup if images are enabled and a month image cannot be displayed. A new popup per image failure replaces the former popup on Images toggle-on. Using PNG images and Tk 8.6+ (which is standard in Python 3.4+ on Windows) is generally recommended, as it provides both image quality and install simplicity.

This code may also work on **Python 2.X** with minor changes (e.g., tkinter imports would use Tkinter), but cross-version compatibility was dropped after grappling with it on the recent *mergeall* archive synchronization project—an admittedly more system-intensive program, which may represent a worst case for dual-version complexity (see [mergeall's homepage](http://learning-python.com/mergeall.html)).

# Using Calendars

|  |  | This final section lists calendar file usage details to help you manage your saved calendar events. Because the system makes a new calendar for you the first time you run it, most of the following items can be skipped or skimmed if you do not have existing calendar files you wish to use, and a single default and automatic calendar suffices for your goals. |
| --- | --- | --- |

**Format**: Calendar event data is stored in portable, program-neutral **iCalendar ".ics" text files**, using a text format defined by the iCalendar standard. This is the calendar world's equivalent to CSV or JSON files for data exchange, and uses an application-specific format similar in spirit to GEDCOM files in genealogy data. This format provides calendar data portability both to and from other calendar programs and platforms.

Additionally, frigcal both loads and saves calendar files using the common and general **UTF-8 Unicode encoding** scheme. This encoding handles simple ASCII text, but also allows calendar event descriptions to contain both non-English and special-character [Unicode text](http://docs.google.com/docetc/docimgs/macosx/event-view-edit-window.png), and is similarly portable among calendar programs—frigcal can use any existing UTF-8 iCalendar file, and creates UTF-8 iCalendar files which can be used by any other program that supports this scheme. Frigcal also provides an encoding converter, and can display any Unicode character supported by its underlying GUI library.

See the [earlier note](#3j2qqm3) as well as the [version 1.7](http://docs.google.com/docetc/DeveloperGuide.html#release17) developer release notes for more details on Unicode files and support. While frigcal handles all Unicode in calendar data, newer characters outside the Unicode BMP cannot presently be displayed by the Tk GUI toolkit used by frigcal, and are replaced for display only as of version 2.0 (alas, emojis await a new Tk).

**Location**: Your calendars are stored as text data files, one per calendar, in your *calendars folder*. This folder is either the [Calendars](#_2xcytpi) folder where this program resides (the default), or a folder whose path you name in [frigcal\_configs.py](#_1ci93xb) (see its setting "icspath"). For instance, a *travel* calendar's data is stored in a file of this form, where *calendars* refers to your default or configured calendar files folder:

*calendars*\travel.ics

Calendar files managed by frigcal in this folder work on all platforms, and should be usable in other calendar programs that support the iCalendar standard. Conversely, frigcal can use iCalendar files created by other programs if they are placed in this folder.

**Multiple Calendars**: This program supports viewing and editing multiple calendar ".ics" files. Each ".ics" file present in the calendars folder on startup is assumed to be a user calendar; the combination of all their events is displayed in the GUI. Event summaries displayed on days are *sorted* (i.e., ordered and grouped) by calendar filename first, and then creation date within calendar. If you use multiple calendars, you can choose calendar filenames to group day events as desired (e.g., the first filename by sort order gets precedence when screen space is limited, though the [event selection list](#_1ksv4uv) displays all).

**Save and Backup**: A calendar's data is saved to its ".ics" file *only* under four conditions—on program [quit](#4d34og8) requests; if its data has changed; if the save is verified by the user; and after a successful and automatic backup copy is made of the file's prior version. Files are not updated after each change. Rather, for speed and accuracy, calendar data is stored and updated in memory only, and saved to files later on program quit requests. On a quit (the main window's "X" or red-circle close button), a two-step dialog sequence confirms a *save* and then an *exit*, and either step can be confirmed or cancelled. Hence, you can backup and save at any time without exiting—click the main window's close button, and confirm the save dialog but cancel the exit dialog. If no data has changed, the save dialog does not appear on quit requests. Also, frigcal does not change your calendar file unless it was able to backup the prior version, per the next point.

**Backup Model**: The system maintains multiple backup copies of each calendar file, created when changed data is saved. Backups files show up in the automatically created [Backups](#_2xcytpi) subfolder of your calendars folder, with date/time name prefixes to make them unique. For instance, a backup of a *travel.ics* calendar is stored on saves in a file of this form:

*calendars*\Backups\date150405-time142907--travel.ics

Backup files are simply copies of iCalendar text files, and can be used directly as calendars; rename to strip their date/time prefixes if desired. To change the maximum number of backups retained for each ".ics" file, set its variable in [frigcal\_configs.py](#_1ci93xb) (the current default is 10, which suffices for most users)—the system will keep up to that many most-recent backup copies per file in the Backups subfolder. Older backups are pruned as needed when new backups are written, and saves overwrite calendar files only if a backup succeeds.

**Default Calendar**: By default, the program creates and uses one ".ics" file if none are present on initial (or later) startup. If you are starting from scratch and can group and colorize your events by category name instead of calendar file, this single, default, and automatic calendar will probably be adequate. frigcal creates your default calendar in your [calendars folder](#1pxezwc), at:

*calendars*\frigcal-default-calendar.ics

You can remove this calendar if you add others later and don't wish to use the default (see [Removing Calendars](#147n2zr) ahead). frigcal also generates a required initial event in the default calendar that looks [like this](http://docs.google.com/docetc/docimgs/macosx/default-calendar.png), which you may also later remove.

**Making New Calendars**: To make new calendars manually, run the [makenewcalendar.py](#_4i7ojhp) script to make any number of new ".ics" calendar files in your calendars folder. You can run this script from a command line, or by simply clicking its filename/icon on Windows. All calendars you create this way will be loaded on startup. This script respects any calendar folder settings in [frigcal\_configs.py](#_1ci93xb); new calendars created this way are stored wherever your calendar folder is located when the script is run. frigcal generates a required initial event in new calendars that looks similar to [this](http://docs.google.com/docetc/docimgs/macosx/default-calendar.png), which you may later remove.

**Using Existing Calendars**: To use existing calendars, simply copy any number of existing ".ics" files to your [calendars](#_2xcytpi) folder. All calendars you copy there will be loaded on program startup, and backed up on saves in the Backups subfolder. As noted earlier in this section, frigcal supports non-English and special-character *Unicode text* in calendar files and displays, and uses the UTF-8 encoding to load and save calendar files. No special handling is required for existing calendar files already in UTF-8 or ASCII encoding forms. Existing calendar files not in UTF-8 format may require a one-time conversion; see [version 1.7](http://docs.google.com/docetc/DeveloperGuide.html#unicodeedits) developer release notes for options for using existing non-UTF-8 files, including a provided encoding-converter script (also recommended [earlier](#_1ci93xb) for configuration files).

**Removing Calendars**: To remove a calendar, simply delete its ".ics" file from the [calendars](#_2xcytpi) folder. All remaining ".ics" files in that folder will be used on the next run. *Note*: if you remove a calendar file from your calendars folder, frigcal will *not* automatically remove any backups of that calendar from the [Backups](#_2xcytpi) subfolder. This is by design—backup files are small, may be useful, and are your property. If you're sure you'll never need them, also manually remove any backup [files](#49x2ik5) of a removed calendar when removing the calendar's main file.

**Empty Calendars**: If you attempt to save a calendar after removing its last event, frigcal will create a required sole event in the calendar because empty calendar files aren't allowed. This keeps the calendar active in the next session, and you may remove the added event later. If you wish to instead remove a calendar altogether, delete its file per the prior bullet.

**Using Outlook Calendars**: To view Outlook (or other calendar program) events: export calendars to ".ics" files if needed, and copy/move them to your frigcal calendars [folder](#1pxezwc). Event text may be viewed and edited and all existing calendar data is retained on saves. For simplicity, though, not all Outlook functionality is supported by this program; see the [introduction](#_1fob9te) earlier and Developer Guide's [Why frigcal?](http://docs.google.com/docetc/DeveloperGuide.html#s6) for more on program features and limitations.

**Performance**: Despite its text-based storage medium, this program performs well by most measures. As an example: for a very large ".ics" file used in testing (currently 876K bytes, 35K lines, and 12 years of events), the GUI can take from 5 to 10 seconds to appear on a Windows 7 Ultrabook and Window 8.1 tablet, due to the initial parsing step. Once the file is loaded, though, this program runs very quickly thereafter, except for a generally shorter multi-second delay to generate and write the file on exit. Memory usage is also reasonable by modern standards of reasonable—with over a decade of calendar data, this program takes no more space than a typical web page viewed in a typical browser.

***Updates***: The load for the example calendar described takes only 2 to 3 seconds on a newer MacBook Pro. Moreover, Python 3.5 now runs the initial calendar load almost twice as fast on Windows in use cases tested; see [version 1.6](http://docs.google.com/docetc/DeveloperGuide.html#release16) developer release notes.

**Handling Backup/Save Errors**: File backup and save errors should rarely (if ever) occur, and the system goes to great lengths to maintain multiple backups as a safeguard, as described [above](#49x2ik5). However, if you get a popup error-message dialog that begins with "*Error while backing up...*", it means that your calendar file was not overwritten in any way, as saves are cancelled on backup failure. If you instead receive an error-message dialog that begins with "*Error while generating...*", the system failed while trying to either generate your calendar's text, or write it to a file. Your simplest recourse in both cases is to simply ***try again***; calendar data is saved intact in memory until program exit (and changed calendars are still marked as changed after a backup or save failure), so a successful later save implies that a temporary system glitch was the error's likely cause.

If your calendars backup and save correctly on a retry, you can safely ignore a prior error. Otherwise, an error during *backups* warrants closing without a save and restarting, as your data file is unchanged. An error during *generation* instead may also be harmless if your calendar file is unmodified (per its mod time); this implies an *icalendar* library failure prior to opening or writing the file. In the worst case, a failure while writing to the calendar file may corrupt its data; in this unlikely event, copy the most recent version of the failed file from your Backups folder (stripping date and time from [its name](#49x2ik5)) up to your Calendars folder, and restart. You'll lose any updates in the failed session, but your prior calendar version's data should be intact. Repeat as needed with older versions in Backups, if the most recent is invalid. ***See also*** the startup and pruning failure error handling in [version 1.5](http://docs.google.com/docetc/DeveloperGuide.html#release15) developer notes.

**Calendar Search**: New in frigcal 2.0, the included script (or executable in some packages) [searchcals.py](http://docs.google.com/searchcals.py) allows you to search all your calendar files for a string. This script is run from a command line and is separate from the GUI to avoid complicating the interface, but its output gives dates in a format that can be directly cut-and-pasted into the GUI's GoTo field to jump to the matching date. For more details, plus examples of running on different platforms, see the documentation at the top of the new script, as well as the [README.txt](http://docs.google.com/README.txt) coverage of your frigcal package.

For more technical details on this program's usage of the iCalendar standard, see the Developer Guide's [iCalendar implementation](http://docs.google.com/docetc/DeveloperGuide.html#s5) notes. You'll also find example calendar files—including those used to generate the screenshots you've seen in this guide—in the shipped calendar folder's [2.0-examples](http://docs.google.com/Calendars/2.0-examples) subfolder; see its [README](http://docs.google.com/Calendars/2.0-examples/README.txt) for more details.

## ***Before You Go...***

If you like this program, you may also be interested in these other productivity tools brought to you by the makers of frigcal:

| [*mergeall*](http://learning-python.com/mergeall.html) | — | Backup and Mirror Your Stuff Your Way |
| --- | --- | --- |
| [*PyEdit*](http://learning-python.com/pyedit.html) | — | Edit Text. Run Code. Have Fun. |
| [*PyMailGUI*](http://learning-python.com/pymailgui.html) | — | Email Without the Evil |
| [*PyGadgets*](http://learning-python.com/pygadgets.html) | — | GUI Toys, Just for the Hack of It |

You can find these and other free software packages at the [programs site](http://learning-python.com/programs.html).

|  | [**Top**](#ihv636) | [**Code**](http://learning-python.com/frigcal-products/unzipped) | [**Page**](https://learning-python.com/frigcal.html) | [**News**](http://learning-python.com/post-release-updates.html) | [**Blog**](http://learning-python.com/posts.html) | [**Apps**](http://learning-python.com/programs.html) | [**Input**](mailto:lutz@learning-python.com) | ©*M.Lutz* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |