Ghini Documentation

Versión 1.0.x

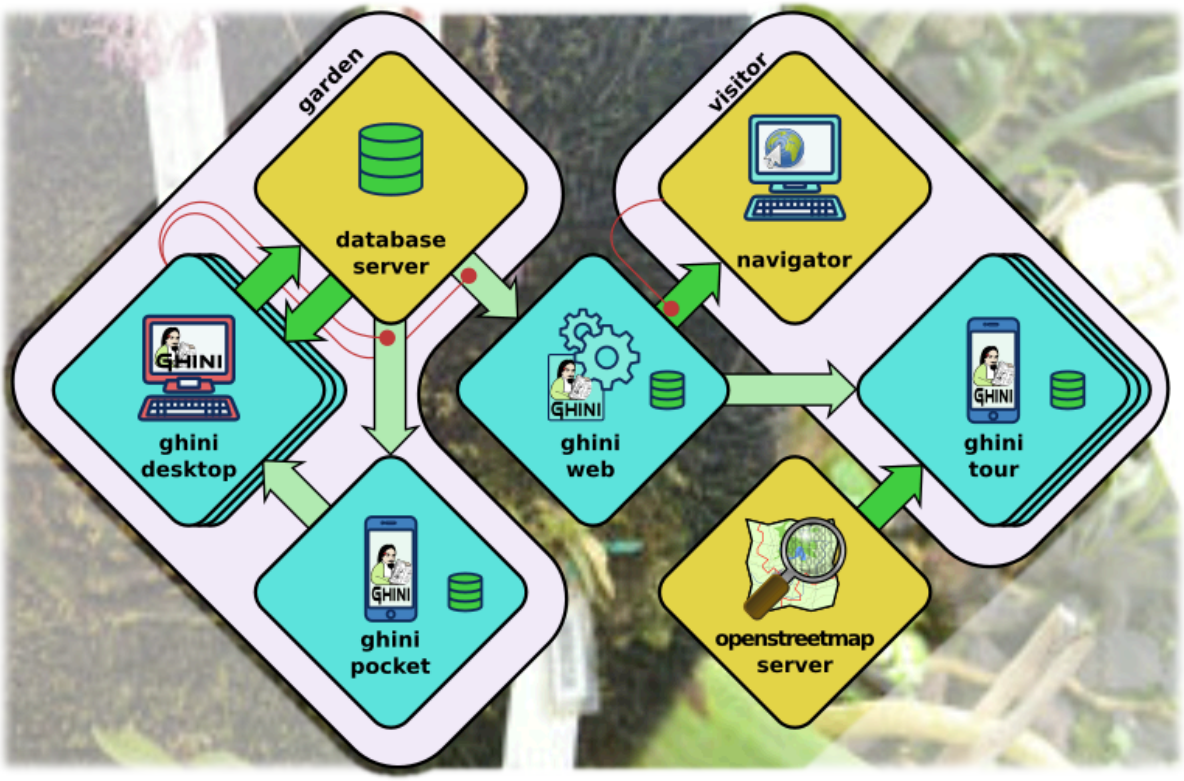
Mario Frasca

enero 11, 2023

Índice general

Ghini es un conjunto de programas para manejar una base de datos botánicas.

- **ghini.desktop** le permite crear y consultar una base de datos que representa objetos y eventos en su colección de plantas.
- **ghini.web** le ayuda a publicar partes de la base datos en el web.
- **ghini.pocket** coloca una instantánea de su base de datos en su dispositivo de mano.
- **ghini.tour** ofrece al visitante un mapa del jardín, y paneles informativos virtuales.



Esta documentación tiene como enfoque central ghini.desktop. Un capítulo final está dedicado al resto de la familia Ghini: *ghini.pocket*, *ghini.web*, *ghini.tour*, y la *data streams between software components*.

Ghini es codigo abierto y libre y es distribuido bajo la GPL, Licencia Pública GNU. Además, todo el software que ofrecemos como servicio, está cubierto por la licencia GNU Affero.

CAPÍTULO 1

Declaraciones

1.1 Objetivos y puntos resaltados de Ghini

¿Debe utilizar este software? Usted sabrá. Confiamos que si usted maneja una colección botánica, encontrará Ghini demasiado útil y esperamos que esta página le motive a ensayarlo.

Esta página muestra cómo el software Ghini atiende a las necesidades de un jardín botánico.

Si ya está convencido y todo lo que quiere es empezar, no más *instale el software* <*install.html*> _ y revise nuestras *recetas aportadas por los usuarios* <*use_cases.html*> _.

1.1.1 Jardín Botánico

Según la Wikipedia, »un jardín *botánico* es un jardín dedicado a la recolección, cultivo y exhibición de una amplia gama de plantas, identificadas con sus nombres botánicos« y todavía según la Wikipedia, »un *jardín* es un espacio planeado, generalmente al aire libre, destinado a la exhibición, cultivo y disfrute de las plantas y otras formas de la naturaleza.«

Así que tenemos en un jardín botánico tanto el espacio físico, el jardín, como su dinámica, las actividades a que se dedica el jardín, actividades que nos hace llamar al jardín un jardín botánico.

1.1.2 Software para Jardines Botánicos

En el otro extremo de nuestro razonamiento tenemos el *programa aplicativo* Ghini y nuevamente citando la Wikipedia, »un programa aplicativo es un programa diseñado para realizar un grupo coordinato de funciones, tareas o actividades a beneficio del usuario«, o, en definitiva, »diseñado para ayudar a las personas a realizar una actividad«.

Figura 1: **el jardín en cuanto jardín**

Figura 2: **actividades del jardín, relacionadas a la colección**

Datos y algoritmos dentro de Ghini han sido diseñados para representar el espacio físico y la dinámica de un jardín botánico.

Figura 3: **la estructura esencial de la base de datos Ghini**

En la figura anterior, una vista simplificada de la base de datos, los bloques resaltados son los relativos a los objetos fundamentes en la representación de la colección.

Distinguimos tres secciones principales en la base de datos. Leemos el gráfico de derecha a izquierda, empezando con la información relevante a la **Taxonomía**, pasando por la administración de la **Colección** y finalmente considerando el **Jardín** físico.

El elemento central de la base de datos Ghini es la `Accession` (Accesión). Seguir los enlaces a las otras tablas de la base de datos nos permite entender mejor su estructura:

Accesión conecta Planting a Species

> Una `Accession` representa el evento, abstracto, de la recepción de material botánico en el jardín, y conecta la `Plant` —grupo de plantas físicas ubicadas en una `Location` en el jardín— con la `Species`. De una `Accession` dependen cero o más `Plant` (0..n) y siempre está conectada a exactamente 1 `Species`. Cada `Plant` pertenece a una sola `Accession`, cada `Species` puede tener asociadas múltiples `Accession`.
>
> Una `Accession` permanece en la base de datos aunque todas sus `Planting` hayan sido removidas, vendida, o se hayan muerto. Al individuar la `Species` de una `Accession` se conecta en manera consistente todas sus `Planting` a la misma `Species`.

Accesiones como base de la historia de las plantas

> `Propagation` y `Contact` proporcionan material vegetal para el jardín; Esta información es opcional y coleccionistas más pequeño pueden preferir dejar esta a un lado. Un ensayo de `Propagation` puede ser infructuoso, mayoría de las veces resultará en exactamente una accesión, pero puede también producir ligeramente diferentes taxones, por lo que la base de datos permite cero o más `Accession` por `Propagation` (0..n). También un `Contact` puede dar cero o más `Accession` (0..n).

Accesiones y Verificas como opiniones informatas

> Especialistas pueden exprimir una opinión sobre la `Species` a que pertenece una `Accession`, formulando una `Verification`, poniendo su firma, y afirmando el nivel de confianza a aplicar.

Propagaciones como fuentes de Accesiones

> Si una `Accession` fue resultado de una `Propagation` exitosa realizada en el jardín, la `Propagation` realiza la conexión entre la nueva `Accession` y de sus `Plant` a la misma planta madre, sea la que ofreció la semilla, o el padre vegetativo.

Incluso después de la explicación anterior, los nuevos usuarios generalmente todavía preguntar ¿por qué necesitan pasar a través de una pantalla `accesión`, mientras que todo lo que quieran

es insertar una planta de la colección y otra vez: ¿Qué es este «accession» cosa de todos modos? Mayoría de las discusiones en la red no hace el concepto ningún clarificante. Uno de nuestros usuarios dio un ejemplo que estoy contento de incluir en la documentación de Ghini.

Caso de uso

1. A principios de 2007 conseguimos cinco plántulas de * Heliconia longa * (una Species de planta) de nuestro vecino (el Contact fuente). Puesto que era la primera adquisición del año, le pusimos nombre 2007.0001 (o sea les dimon un único código de Accession, con cantidad 5) y nos plantamos todos juntos en un único Location como una sola Plant, también con la cantidad de 5.

2. En el momento de la escritura, nueve años más tarde, la Accession 2007.0001 tiene 6 Plant distintas, cada una en un diferente Location en nuestro jardín, que se dieron vegetativamente (asexualmente) de las 5 plantas originales. Nuestra intervención sólo fue dividir, mover y por supuesto escribir esta información en la base de datos. La cantidad total de la planta está por encima de 40.

3. Nuevas Plant obtenidas por Propagation sexual (asistida) entran en nuestra base de datos bajo diferentes códigos de Accession, en que nuestro jardín es la fuente de Contact y donde sabemos cuál de nuestras Plant el padre de semilla.

los tres casos anteriores se traducen en varias historias de uso:

1. activar el menú Insertar → accesión, verificar la existencia y corrección de la Species *Heliconia longa*, especificar la cantidad inicial de la Accession; añadir su Plant en la Location deseada.

2. Editar Plant para corregir la cantidad de plantas vivas, repetir este proceso tantas veces como sea necesario.

3. Editar Plant dividiendo en las Location diferentes, esto produce distintas Plant bajo la misma Accession.

4. Editar “” siembra “” para agregar una (semilla) “” propagación “”.

5. Editar Plant para actualizar el estado de la Propagation.

6. activar el menú Insertar → Accesión para asociar una accesión a un intento exitoso de Propagation; agregar la nueva Plant en la Location deseada.

En particular la capacidad de dividir un Plant en varias Location diferentes y que todas queden uniformemente asociadas a una Species, o la posibilidad de mantener información sobre Plant que se han ido de la colección, ayudar a justificar la presencia del nivel de abstracción Accession.

1.1.3 Hypersimplified view

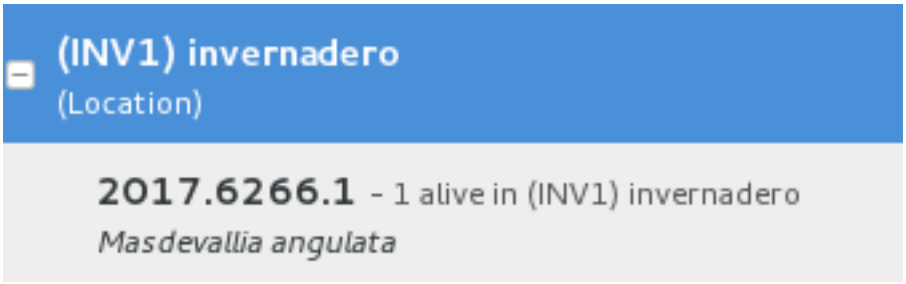
People using Ghini only sporadically may prefer ignoring the database structure and look at it as two nested sequences of objects, each element of the sequence being necessary to add

element at the next level.

In order to get down to an Accession, you will need four levels, as in this example:

A quite complete set of Families and Genera are inserted in your database at the moment Ghini initializes it. So all you need is adding Species and Accessions, in this order.

When placing a physical Plant (relative to an Accession) somewhere in the garden, you need to describe this «somewhere» digitally, as a Location in the garden.



1.1.4 Puntos destacados

no-tan-breve lista de destacados, para abrir tu apetito.

información taxonómica

Cuando primero arrancas Ghini y conectas a una base de datos, Ghini inicializa la base de datos no sólo con todas las tablas que necesita para funcionar, sino también importa en las tablas de taxonomía los datos de la «RBG Kew's Family and Genera list from Vascular Plant Families and Genera compiled by R. K. Brummitt and published by the Royal Botanic Gardens, Kew in 1992». En 2015 se han revisado los datos sobre Orchidaceae, utilizando «Tropicos, sistema de información botánica en el jardín botánico de Missouri - www.tropicos.org» como fuente.

importar data

Ghini le permitirá importar los datos que pones en formato json intermedio. Lo que importa completa lo que ya tienes en la base de datos. Si usted necesita ayuda, puede pedir algún profesional Ghini para ayudarle a transformar sus datos en formato de json intermedio de Ghini.

sinónimos

Ghini le permite definir sinónimos para especies, géneros, familias. También esta información puede ser representada en el formato json de intercambio y puede ser importada in una base de datos Ghini.

científicamente responsable

Ghini implementa el concepto de "accesión", conexión abstracta entre la planta física (o un grupo de plantas físicas) y la especie o taxón a que pertenecen. Cada accesión puede asociar todo el mismo grupo de plantas que le pertenece a uno o más taxones, en el caso no haya acuerdo entre los taxónomos a disposición del jardín: cada taxónomo puede insertar su opinión en la base de datos, sin necesitar borrar la información anteriormente presente. Todas las verificaciones se encuentran en la base de datos, con fecha y firma.

facilita la identificación a distancia

Ghini le permite asociar fotos a plantas físicas, esto puede ayudarle a reconocer la planta en caso de que se pierda la nota, o ayudar a la identificación taxonómica si no está disponible un taxónomo.

exportación e informes

Ghini le permite exportar los datos en cualquier formato textual le parezca útil. Ghini utiliza un motor de plantillas muy flexible, que se llama "mako", con el cual solo su imaginación es el límite a los formatos en que exportar los datos. Una vez instalado, hay varios ejemplos en la carpeta *mako*.

anotar la información

A practicamente todos objectos en la base de datos es posible asociar notas. Las notas se pueden categorizar y por supuesto utilizar en búsquedas.

jardines o herbarios

manejo de ubicaciones de plantas.

historial de la base de datos

Todos los cambios en la base de datos se almacenan en un log. Todos los cambios son "firmados" y sellados temporalmente. Ghini hace fácil la recuperación de todos los cambio en el último día o semana, o en un período especifico del pasado.

búsquedas sencillas o complejas

Ghini le permite consultar la base de datos empleando palabras clave sencillas, p.e.: el nombre del lugar o un nombre de género, o puede escribir consultas más complejas, que no llegarán a la complejidad de SQL paro que le permitirán localizar sus datos con un nivel de detalle aceptable.

en su propia base de datos

Ghini no es un sistema de gestión de base de datos, ni intenta serlo. Ghini guarda la colección en una base de datos SQL, y se puede conectar a virtualmente cualquier sistema SQL para que exista un conector SQLAlchemy. En práctica esto incluye todos los más modernos sistemas relacionales, como MySQL, PostgreSQL, Oracle. También puede trabajar con SQLite, que, para usuarios independientes es un sistema muy eficiente y por supuesto más que suficiente. En combinación con un sistema de gestión SQL, es posible construir un sistema LAMP (Linux-Apache-MySQL-Php) o similar, e incluir la colección actualizada en el sitio web de su institución.

en su propio idioma

The program was born in English and all its technical and user documentation is first written in that language. Both technical and user documentation use `gettext`, an advanced tool for semi-automatic translation.

The program has been translated and can be used in various other languages, including Spanish (97 %), French (82 %), Portuguese (71 %), to name some Southern American languages, as well as Ukrainian (100 %) and Czech (71 %).

Translation of documentation goes a bit slower, with only Ukrainian, Spanish and Italian at more than 50 %.

en su propia plataforma software

Instalación de Ghini en Windows es un proceso fácil y lineal, no tarda más de 10 minutos. Ghini nació en Linux y lo instale en ubuntu, fedora o debian es por lo tanto más fácil. MacOSX se basa en unix, es posible ejecutar correctamente el procedimiento de instalación de Linux en cualquier ordenador Apple reciente, después de unos pasos de la preparación.

facil puesta al día

El proceso de instalación produce una instalación que, si deseado, se puede poner al día en menos que un minuto. Dependiendo de la cantidad de respuesta recibida, quien trabaja en Ghini produce una nueva distribución cada par de días o de vez en cuando.

garantizado por pruebas unitarias

Ghini es continuamente y unidad probado extensivamente, lo que hace la regresión de la funcionalidad cerca de imposible. Cada actualización es automáticamente calidad comprobado, en el servicio de integración continua de Travis. Integración de TravisCI con la plataforma de github resultará difícil para nosotros liberar todo lo que tiene una sola falla prueba.

Virtualmente cada cambio y extención que se hace en Ghini, va acompañada por unas nuevas pruebas unitarias, que define el comportamiento y evidenciará en el futuro cualquier cambio no deseado.

adaptable/extensible

Ghini puede ser expandido a través de complementos (plugin), y puede ser adaptado según lo que necesite la institución que lo adopte.

1.2 Misión y Visión

Aquí afirmamos Quiénes somos, qué pensamos de nuestro trabajo, lo que puede esperar de este proyecto.

1.2.1 Quién está detrás de Ghini

Ghini is a small set of programs, meant to let collection managers manage their collection also digitally.

Ghini was born back in 2004 as Bauble, at the Belize Botanical Garden. It was later adapted to the needs of a few more gardens. Brett Adams, the original programmer, made this software a commons, by releasing it under a GPL license.

After years of stagnation Mario Frasca revived the project, and rebranded it as Ghini in honour of Luca Ghini, founder of the first European botanic garden and herbarium. Mario Frasca started advocating, travelling, distributing, developing, expanding, redefining, documenting it, and it is now Mario Frasca writing this, looking for users, requesting feedback.

Behind Ghini there's not only one developer, but a small but growing global users community.

Translations are provided by volunteers who mostly stay behind the scenes, translating missing terms or sentences, and disappearing again.

Para que sea más claro cuando hablamos de Ghini, deberíamos —y lo hacemos en este documento— indicar si estamos indicando a Ghini(el software), o a Ghini(la gente), a menos que obviamente nos referimos a ambas cosas.

1.2.2 Misión

Our goal as Ghini Software is to provide free software, of proven quality, and to let anybody install it if they feel like it. We also aim at facilitating access to functional knowledge, in the form of documentation or by laying the contact among users or between users and software professionals.

All our sources, software and documentation, are open and free, and we welcome and stimulate people to use and to contribute. To facilitate community forming, all our platforms can be consulted without registration. Registration is obviously required if you want to contribute.

Ghini welcomes the formation of groups of users, bundling forces to define and finance further development, and we welcome developers contributing software, from any corner in the world, and we stimulate and help them comply with the high quality requirements, before we accept the contributed code in the software sources.

1.2.3 Visión

La visión es una afirmación, sirve para indicar el camino a seguir y proyecta una imagen futura de lo que queremos con nuestra organización, en una forma realista y atractiva. Sirve como motivación porque visualiza el desafío y la dirección de los cambios necesarios para crecer y prosperar.

- en el año 2020
- punto de referencia
- comunidad
- desarrollo
- integración con portal web
- información geográfica

CAPÍTULO 2

Instalar Ghini

2.1 Instalación

ghini.desktop es un programa independiente de la plataforma software pues puede funcionar en sistemas unix como Linux y MacOSX así como en Windows.

one-liner for hurried users.

> Linux users just download and run the installation script. You may read the documentation later.
>
> Windows users in a real hurry don't the instructions and use a recent Windows installer. You do not miss any functional feature, but you have less chances to contribute to development.
>
> Mac users are never in a hurry, are they?

El grupo de desarrollo de Ghini es pequeño, y nos enfocamos en mejorar el código, o documentarlo, y no nos sobra tiempo para también mantener los paquetes de instalación. En lugar de tantos paquetes, uno por plataforma suportada, ofrecemos la misma procedura de instalación para todas plataformas. Esto no sólo nos ahorra tiempo sino tiene unas avantajas importantes, que se harán evidentes en el uso del programa.

La instalación está basada en ejecutar un script.

- El script para GNU/Linux se ocupa de todo, desde solucionar las dependencias hasta la instalación para los usuarios en el grupo `ghini`.
- El script de Windows le pide primero de instalar un par de cosas.

- En MacOSX vamos a utilizar el mismo script de GNU/Linux, pero como OSX no tiene un gestor de paquetes, antes de ejecutar el script vamos a instalar uno.

Si sigue los siguientes pasos, se encontrará con Ghini instalado en un entorno virtual, todas la dependencias Python estarán instalada localmente y no entrarán en conflicto con otros programas Python que puedan estar el la misma computadora.

Las dependencias que no caben en un entorno virtual Python son: Python, virtualenv, GTK+, and PyGTK. Su instalación depende por plataforma.

Si por cualquier razón quisiera usted desinstalar Ghini, lo único que tiene que hacer es eliminar el entorno virtual, que es una carpeta, con todo su contenido.

2.1.1 Instalación en GNU/Linux

Open a shell terminal window, and follow the following instructions.

1. Download the *devinstall.sh* script:

   devinstall.sh

2. Invoke the script from a terminal window, starting at the directory where you downloaded it, like this:

   ```
   bash ./devinstall.sh
   ```

   The script will produce quite some output, which you can safely ignore.

   ---

   global installation

   When almost ready, the installation script will ask you for your password. This lets it create a `ghini` user group, initialise it to just yourself, make the just created `ghini` script available to the whole `ghini` user group.

   If feeling paranoid, you can safely not give your password and interrupt the script there.

   Possibly the main advantage of a global installation is being able to find Ghini in the application menus of your graphic environment.

   ---

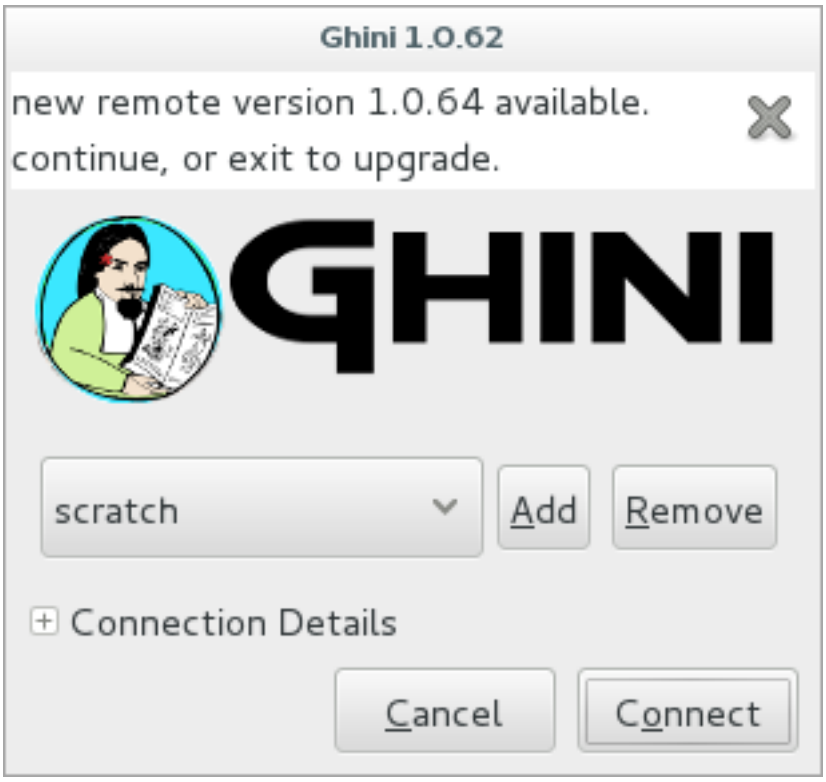
3. You can now start ghini by invoking the `ghini` script:

   ```
   ghini
   ```

   1. You use the same `ghini` script to update ghini to the latest released production patch:

      ```
      ~/bin/ghini -u
      ```

      This is what you would do when ghini shows you something like this:



2. Users of the global installation will also type `ghini` to invoke the program, but they will get to a different script, located in `/usr/local/bin`. This globally available `ghini` script cannot be used to update a ghini installation.

3. Again the same `ghini` script lets you install the optional database connectors: option `-p` is for PostgreSQL, option `-m` is for MySQL/MariaDB, but you can also install both at the same time:

```
~/bin/ghini -pm
```

   Please beware: you might need solve dependencies. How to do so, depends on which GNU/Linux flavour you are using. Check with your distribution documentation.

4. You can also use the `ghini` script to switch to a different production line. At the moment `1.0` is the stable one, but you can select `1.1` if you want to help us with its development:

```
~/bin/ghini -s 1.1
```

Nota para principiantes

Para ejecutar una secuencia de comandos, asegúrese de que anotar el nombre del directorio a los que han descargado el script, luego abres una ventana de terminal y en que ventana tipo "bash" seguido de un espacio y el nombre completo de la escritura incluido nombre y pulsar en la tecla enter.

nota técnica

You can study the script to see what steps if runs for you.

In short it will install dependencies which can't be satisfied in a virtual environment, then it will create a virtual environment named `ghide`, use git to download the sources to a directory named `~/Local/github/Ghini/ghini.desktop`, and connect this git checkout to the `ghini-1.0` branch (this you can consider a production line), it then builds ghini, downloading all remaining dependencies in the virtual environment, and finally it creates the `ghini` startup script.

If you have `sudo` permissions, it will be placed in `/usr/local/bin`, otherwise in your `~/bin` folder.

Siguiente...

Conectarse a una base de datos.

2.1.2 Instalar en MacOSX

Being macOS a unix environment, most things will work the same as on GNU/Linux (sort of).

Primer paso, construir en MacOSX un entorno reconocible como unix:

1. herramienta de desarrollo: xcode. por favor averigue en la wikipedia cual es la versión de xcode adapta a su mac.
2. gestor de paquetes: homebrew (o si utiliza un sistema más viejo, tigerbrew).

Installation on older macOS.

Every time we tested, we could only solve all dependencies on the two or three most recent macOS versions. In April 2015 this excluded macOS 10.6 and older. In September 2017 this excluded macOS 10.8 and older. We never had a problem with the lastest macOS.

The problem lies with homebrew and some of the packages we rely on. The message you have to fear looks like this:

```
Do not report this issue to Homebrew/brew or Homebrew/core!
```

The only solution I can offer is: please update your system.

On the bright side, if at any time in the past you did install ghini.desktop on your older and now unsupported macOS, you will always be able to update ghini.desktop to the latest version.

With the above installed, open a terminal window and run:

```
brew doctor
```

asegúrese de haber entendido los problemas que serás reportados, y corríjalos. pygtk necesita xquartz y brew no puede solucionar la dependencia en manera automatica. instale xquartz utilizando brew:

```
brew install Caskroom/cask/xquartz
```

y finalmente instalar las últimas dependencias:

```
brew install git
brew install pygtk  # takes time and installs all dependencies
```

siga las instrucciones relativas a como activar lo que acaba de instalar.

In particular, make sure you read and understand all reports starting with `If you need to have this software`.

You will need at least the following four lines in your `~/.bash_profile`:

```
export LC_ALL=en_US.UTF-8
export LANG=en_US.UTF-8
export PATH="/usr/local/opt/gettext/bin:$PATH"
export PATH="/usr/local/opt/python/libexec/bin:$PATH"
```

Activate the profile by sourcing it:

```
. ~/.bash_profile
```

Antes de poder ejecutar `devinstall.sh` como en GNU/Linux, todavía necesitamos instalar un par de paquetes de python, a nivel global. Lo hacemos así:

```
sudo -H pip install virtualenv lxml
```

ahora puede seguir como en una normal máquina unix, y para eso tenemos un archivo de instalación. Lea las instrucciones para GNU/Linux, las siga, y disfrute.

As an optional aesthetical step, consider packaging your `~/bin/ghini` script in a platypus application bundle. The `images` directory contains a 128×128 icon.

Siguiente...

Conectarse a una base de datos.

2.1.3 Instalar en Windows

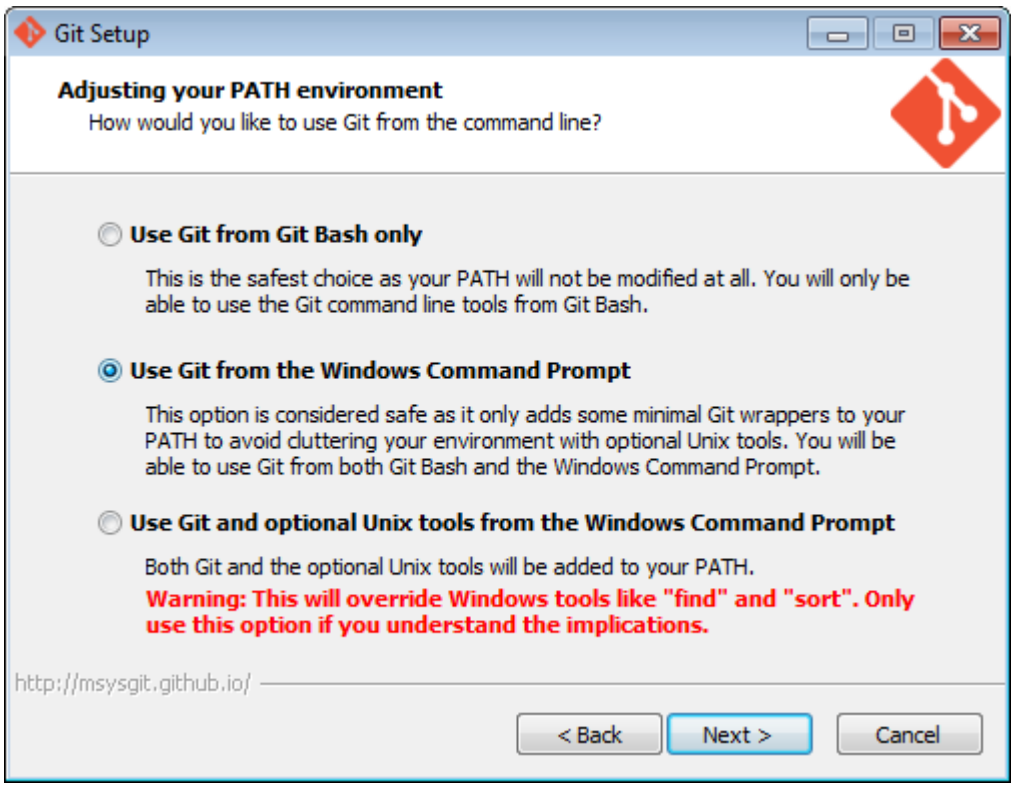
Los pasos descritos aquí les enseñan cómo instalar Git, Gtk, Python, y los conectores python para base de datos. Una vez correctamente configurato este entorno, el procedimiento de instalación de Ghini funciona como en GNU/Linux. Los pasos finales son otra vez específicos de Windows.

Nota: Ghini has been tested with and is known to work on W-XP, W-7 up to W-10. Although it should work fine on other versions Windows it has not been thoroughly tested.

Los pasos de instalación en Windows:

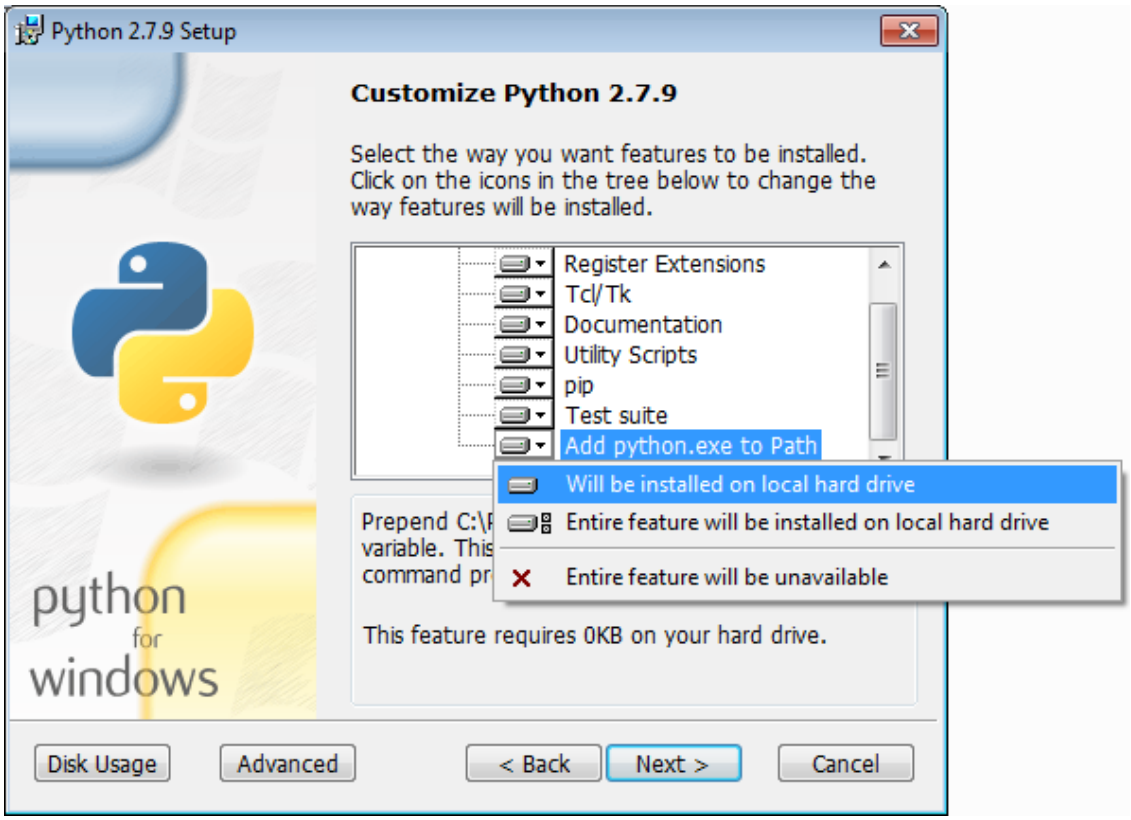
1. download and install `git` (comes with a unix-like `sh` and includes `vi`). Grab it from the Git download area.

   todas opciones por defecto están bien, excepto que queremos poder encontrar git desde la línea de mando Windows:



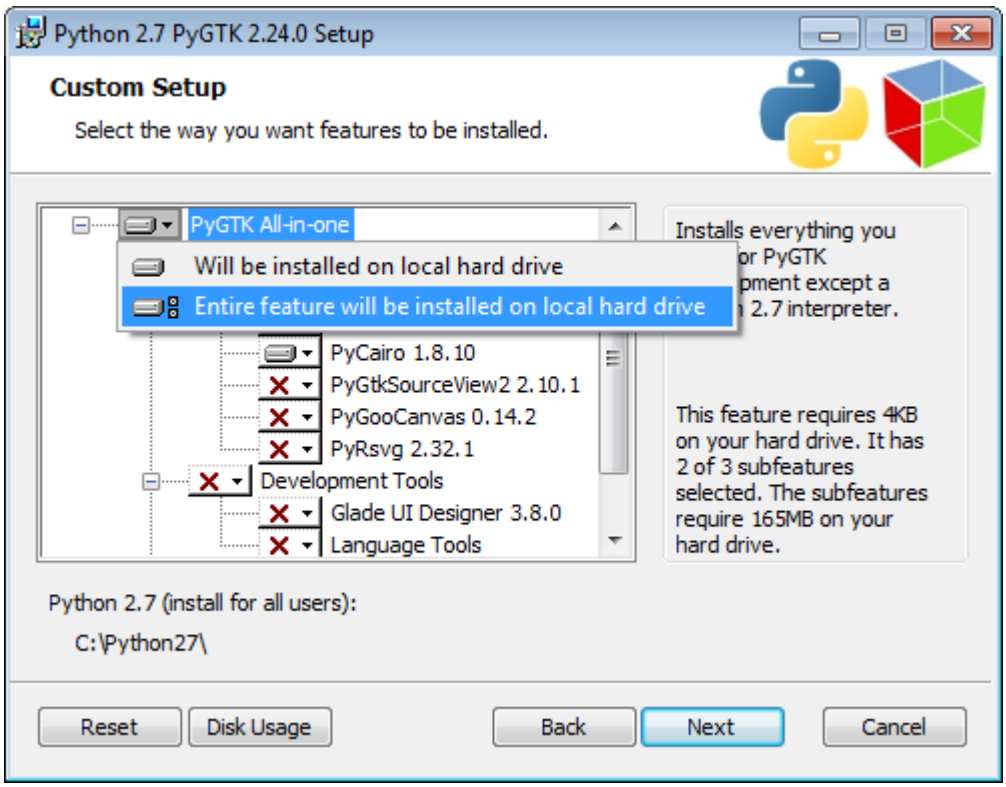
2. download and install Python 2.x (32bit). Grab it from the Python official site.

   When installing Python, do put Python in the PATH:



3. download `pygtk` from the official source. (this requires 32bit python). be sure you download the «all in one» version.

   Make a complete install, selecting everything:



4. (Possibly necessary, maybe superfluous) install lxml, you can grab this from the pypi

archives

Recuerde que usted necesita la versión de 32 bits, para Python 2.7.

5. (definitely optional) download and install a database connector other than `sqlite3`.

   If you plan using PostgreSQL, the best Windows binary library for Python is psycopg and is Made in Italy.

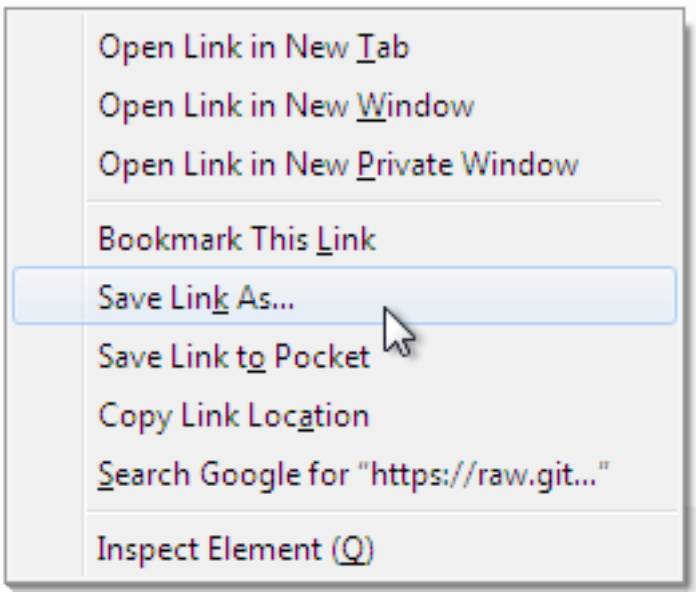
6. **REINICIALICE**

   así es Windows, cambió cosas en el sistema, tiene que reiniciar!

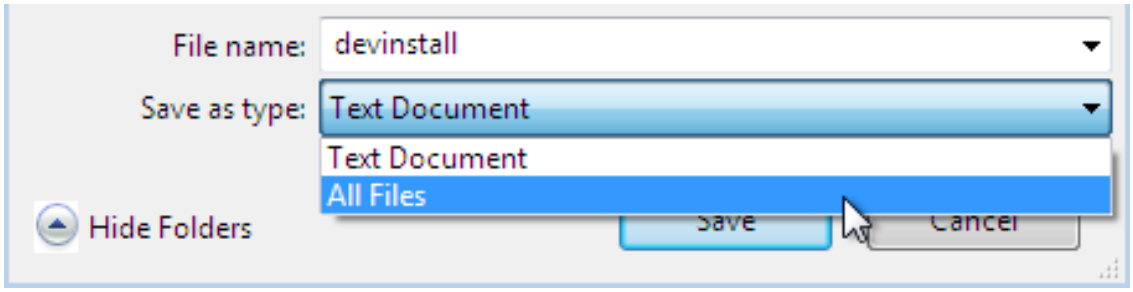
7. We're done with the dependecies, now we can download and run the batch file:

   devinstall.bat

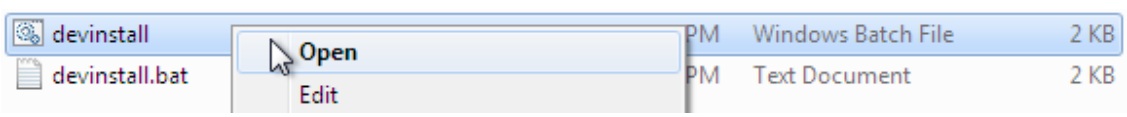
   Please don't just follow the above link. Instead: right click, save link as...



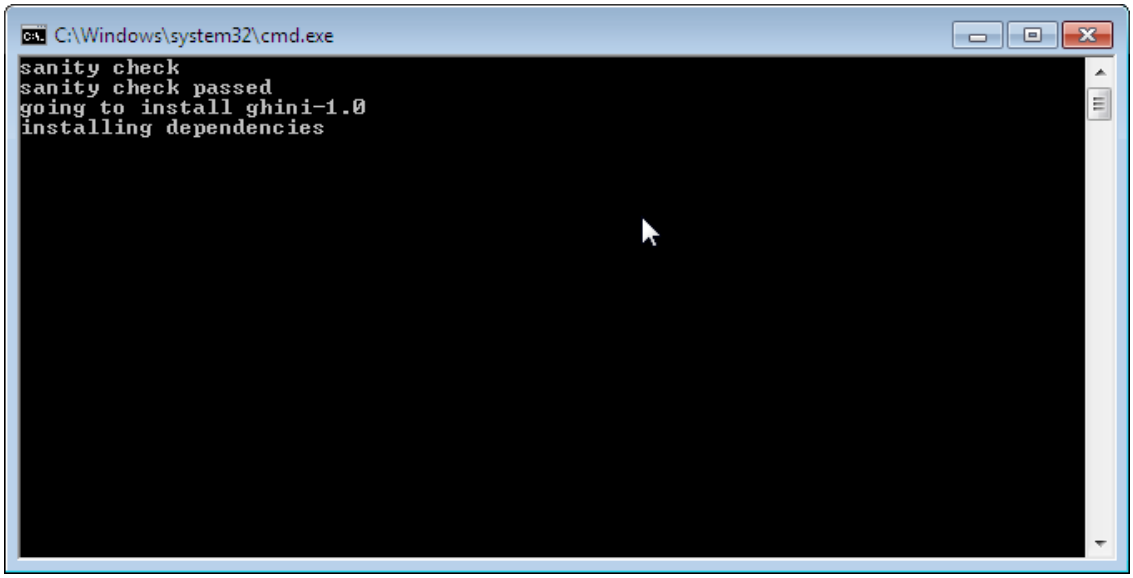
Also make sure you don't let Windows convert the script to a text document.



Now **Open** the script to run it. Please note: in the below image, we have saved the file twice, once letting Windows convert it to a text document, and again as a Windows Batch File. Opening the batch file will run the script. Opening the text document will show you the code of the batch file, which isn't going to lead us anywhere.



If you installed everything as described here, the first thing you should see when you start the installation script is a window like this, and your computer will be busy during a couple of minutes, showing you what it is doing.



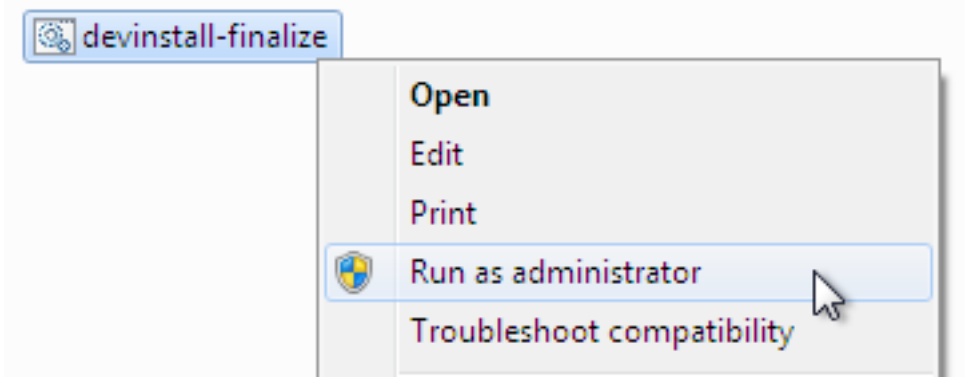
Running `devinstall.bat` will pull the `ghini.desktop` repository from github to your home directory, under `Local\github\Ghini`, checkout the `ghini-1.0` production line, create a virtual environment and install ghini into it.

You can also run `devinstall.bat` passing it as argument the numerical part of the production line you want to follow.

This is the last installation step that depends, heavily, on a working internet connection.

The operation can take several minutes to complete, depending on the speed of your internet connection.

8. la última etapa de instalación crea el grupo Ghini y accesos directos en el menú de inicio de Windows, para todos los usuarios. Para ello, necesita ejecutar una secuencia de comandos con derechos administrativos. El script se llama “” devinstall-finalize.bat””, es derecho en su carpeta de inicio y se ha creado en el paso anterior.



Right-click on it, select run as administrator, confirm you want it to make changes to your computer. These changes are in the Start Menu only: create the Ghini group, place the Ghini shortcut.

9. download the batch file, it will help you staying up-to-date:

   ghini-update.bat

   If you are on a recent Ghini installation, each time you start the program, Ghini will check on the development site and alert you of any newer ghini release within your chosen production line.

   Any time you want to update your installation, just run the `ghini-update.bat` script, it will hardly take one minute.

   How to save a batch file, and how to run it: check the the quite detailed instructions given for `devinstall.bat`.

If you need to generate PDF reports, you can use the XLS based report generator and you will need to download and install Apache FOP. After extracting the FOP archive you will need to include the directory you extracted to in your PATH.

If you choose for PostScript reports, you can use the Mako based report generator and there are no further dependencies.

Siguiente...

Conectarse a una base de datos.

2.1.4 Installing on Android

`ghini.desktop` is a desktop program, obviously you don't install it on a handheld device, but we do offer the option, for your Android phone or tablet, to install `ghini.pocket`.

`ghini.pocket` is a small data viewer, it comes handy if you want to have a quick idea of a plant species, its source, and date it entered the garden, just by scanning a plant label.

Installation is as easy as it can be: just look for it in Google Play, and install it.

Export the data from `ghini.desktop` to pocket format, copy it to your device, enjoy.

CAPÍTULO 3

Manual del usuario

3.1 Configuración Inicial

Después de una instalación exitosa, las organizaciones más complejas necesitarán configurar su base de datos y por consecuencia Ghini. Esta página se enfoca en esta tarea. Si no tiene ni idea de qué estamos hablando, por favor siga no más con la parte relativa a SQLite.

3.1.1 Que tal SQLite?

¿Esta es la primera vez que utiliza Ghini? ; ¿trabaja usted sólo en su ordenador personal? ; ¿no tiene ni la menor idea de cómo manejar un sistema de gestión de base de datos? ; Si respondiste sí a cualquiera de las anteriores, probablemente mejor quédese con SQLite, la base de datos fácil, rápida, cero administración basada en un sólo archivo.

Con SQLite, no hace falta preparar nada y puede seguir con *connecting*.

Por otro lado, si desea conectar más de una estación de trabajo a la misma base de datos, o si desea ponerlos a disposición de otros clientes, como podría ser un servidor web en ambiente LAMP, debe considerar mantener su base de datos en un sistema de gestión de base de datos como PostgreSQL o MySQL/MariaDB, ambos apoyados por Ghini.

When connecting to a database server as one of the above, you have to manually do the following: Create at least one user; Create your database; Give at least one user full permissions on your database; If you plan having more database users: Give one of your users the `CREATEROLE` privilege; Consider the user with the `CREATEROLE` privilege as a super-user, not meant to handle data directly; Keep your super-user credentials in a very safe place.

When this is done, Ghini will be able to proceed, creating the tables and importing the default data set. The process is database-dependent and it falls beyond the scope of this manual.

Si ya tienes escalofríos o dolores de estómago, no se preocupe más, simplemente adopte SQLite, sin perder características ni prestaciones.

Some more hints if you need PostgreSQL

Start simple, don't do all at the same time. Review the online manual, or download and study the offline version.

As said above, create a database, a user, make this user the owner of the database, decide whether you're going to need multiple users, and preferably reserve a user for database and normal user creation. This super-user should be your only user with `CREATEROLE` privilege.

All normal users will need all privileges on all tables and sequences, something you can do from the *Tools→Users* menu. If you have any difficulty, please open an issue about it.

Connect using the `psql` interactive terminal. Create a `~/.pgpass` file (read more about it in the manual), tweak your `pg_hba.conf` and `postgresql.conf` files, until you can connect using the command:

```
psql <mydb> --username <myuser> --no-password --host
↪<mydbhost>
```

With the above setup, connecting from ghini will be an obvious task.

3.1.2 Conectarse a una base de datos

Al arrancar Ghini, lo primero que aparece es una ventana en que seleccionar la conexión.

Pues si esta es la primera vez que arranca Ghini por supuesto no hay conexiones. Ghini se lo deja notar.

Esta alerta se mostrará en primera activación y también en el futuro si su lista de conexiones se vuelva vacía. Como dice: haga clic en ** Agregar ** para crear tu primera conexión.

No más inserte el nombre de la conexión, algo sencillo y relevante, que se asocie a la colección que va a poner en la base de datos (por ejemplo: «parques municipales»), y pise **OK**. Ghini vuelve a la ventana anterior, donde el nombre de la conexión está seleccionada y los Detalles de conexión se han expandido.

especificar la información de conexión

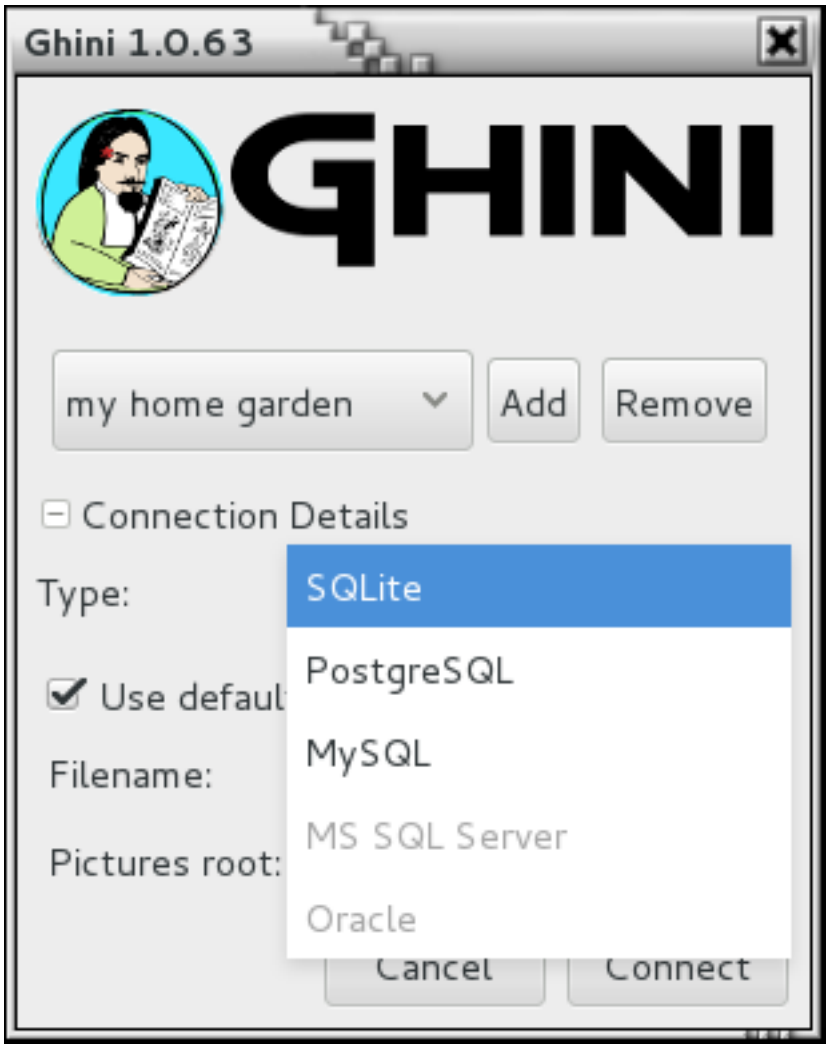
Si no sabría que hacer acá, Ghini le da una mano. Active el **Utilice los valores por defecto** y finalice el crear la conexión pisando **Conectar**.

El resto de esta sección da más detalles de natura técnica. Si no quiere adentrarse, puede seguir con la sección siguiente.

detalles de conexión: sintonización fina

Ghini utiliza por defecto el archivo base de datos de SQLite. Durante el proceso de instalación tenía la opción (y aún la tiene después de la instalación), para agregar conectores de base de datos distintos del predeterminado SQLite.

En este ejemplo, Ghini puede conectarse a SQLite, PostgreSQL y MySQL, pero no hay conectores para Oracle o MS SQL Server.



Si utiliza SQLite, lo único que realmente necesita especificar es el nombre de la conexión. Si permite a Ghini utilizar el nombre predeterminado, Ghini crea el archivo de base de datos con el mismo nombre que la conexión y con extensión `.db` y una carpeta de fotos con el mismo nombre y sin extensión, que se encontrarán en `~/.bauble` en Linux/MacOSX o en `AppData\Roaming\Bauble` en Windows.

Siempre con SQLite, usted quizás recibió o descargó un archivo de base de datos, y quiere utilizarlo. En este caso no deje a Ghini utilizar el nombre predeterminado, sino busque en su computadora la ubicación donde guardó ese archivo SQLite.

Si utiliza otro conector de base de datos, el cuadro de diálogo se verá diferente y le ofrecerá la opción de afinar todos los parámetros necesarios para conectarse a la base de datos de su elección.

If you are connecting to an existing database you can continue to *Editar e insertar información* and subsequently searching-in-ghini, otherwise read on to the following section on initializing a database for Ghini.

Si va a asociar fotos a las plantas, también indique la *carpeta raíz* para las imágenes. El significado de esto se explica con mayor detalle en *Fotos* en edición-y-insertar-data.

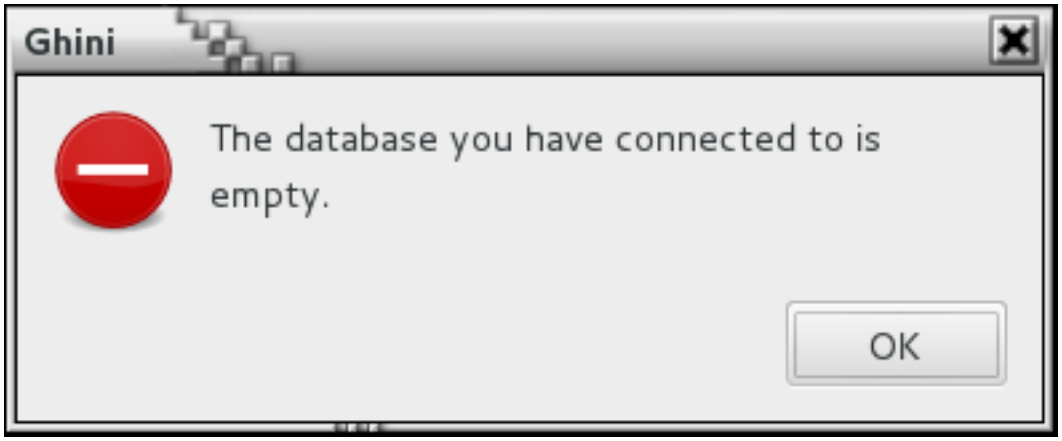
A sample SQLite database

Indeed we have a sample database, from our pilot garden «El Cuchubo», in Mompox, Colombia. We have a zipped sample database for ghini-1.0.

Download and unzip it to the location of your choice, then start Ghini, create a connection named possibly `cuchubo`, or `sample`, and edit the Connection Details. Keep the connection type at the default SQLite, but instead of using the default locations, make sure that Filename points to your unpacked `cuchubo.db` file.

3.1.3 Inicializar una base de datos

La primera vez que se abre una conexión a una base de datos, nunca antes vista con Ghini, el programa presenta el aviso:



e inmediatamente después la pregunta:

Tenga cuidado al especificar manualmente los parámetros de conexión: los valores que ha introducido pueden referirse a una base de datos existente, no debe ser usado con Ghini. Dejando Ghini inicializar una base de datos, la base de datos estará vacía y todo su contenido se pierde.

Si usted está seguro que desea crear una base de datos a este respecto a continuación, seleccione «Yes». Ghini entonces comenzará a crear las tablas de la base de datos e importar los datos por

defecto. Esto puede tardar un minuto o dos así que mientras que todos los datos por defecto es importado a la base de datos asi que tengan paciencia.

Cuando se haya creado, configurado, inicialisado la base de datos, estamos listos para *Editar e insertar información* y lo más interesante *Buscar en Ghini*.

3.2 Buscar en Ghini

Buscar le permite ver, navegar y crear informes a partir de los datos. Puede realizar búsquedas introduciendo las consultas en el formato principal de búsqueda o utilizando el generador de consultas para que cree las consultas para usted. Los resultados de las búsquedas de Ghini se enumeran en la ventana principal.

3.2.1 Estrategias de Búsqueda

Ghini ofrece cuatro estrategias de búsqueda distinctas:

- por valor — implícitamente en todos dominios;
- por expresión — en unos campos implícitos de un dominio explícito;
- por query — en un sólo dominio;
- por nombre binomial — busca solo el dominio Especies.

Todas estrategias de búsqueda, con la notable excepción de la búsqueda del nombre binomial: son minúsculas.

Búsqueda por valor

Búsqueda por valor es la forma más sencilla de búsqueda. Ingrese una o más cadenas y ver qué partidos. El resultado incluye objetos de cualquier tipo (dominio) donde una o más de sus campos contienen uno o más de las cadenas de búsqueda.

No se especifica el dominio de búsqueda, todos están incluidos, ni indica los campos que desea, esto está implícito en el dominio de búsqueda.

La siguiente tabla ayuda a entender los resultados y te guía en la formulación de sus búsquedas.

Buscar dominio Resumen		
nombre y abreviaturas	campo	Tipo de resultado
family, fam	epithet (family)	`Family`
genus, gen	epithet (genus)	`Genus`
species, sp	epithet (sp) ×	`Species`
vernacular, common, vern	nombre	`Species`
geography, geo	nombre	`Geography`
accession, acc	code	`Accession`
planting, plant	code ×	`Plant`
location, loc	code, name	`Location`
contact, person, org, source	nombre	`Contact`
collection, col, coll	locale	`Collection`
tag, tags	nombre	`Tag`

Ejemplos de búsqueda por valor serían: Maxillaria, Acanth, 2008.1234, 2003.2.1, indica.

A menos que explícitamente citado, espacios separan las cadenas de búsqueda. Por ejemplo, si usted busca “” bloque 10”” luego Ghini buscará las cadenas bloque y 10 y volver todos los resultados que coincidan con cualquiera de estas cadenas. Si desea buscar para bloque de 10 como una cadena entera entonces debe citar a la cadena como «»Block 10»””.

× Claves primarias compuestas

Un epíteto de **especie** significa poco sin el género correspondiente, asimismo un código de **planta** es único solamente dentro de la accesión a la que pertenece. En la terminología de la teoría de la base de datos, epíteto y código no son suficientes para formar una **clave primaria** respectivamente para especies y planta. Estos dominios necesitan una **clave primaria compuesta**.

Búsqueda por valor le permite buscar **plantas** por su código de planta completo, que incluye el código de la accesión. Tomados en conjunto, código de accesión y código de planta forman una **clave primaria compuesta** para plantas. Para **especies**, hemos introducido la búsqueda binomial, que se describe a continuación.

Búsqueda por la expresión

Búsqueda con expresión le da un poco más control sobre lo que está buscando. Usted restringe la búsqueda a un dominio específico, el software define los campos de búsqueda dentro del dominio especificado.

Una expresión se construye como `<domain> <operator> <value>`. Por ejemplo la búsqueda: `gen = Maxillaria` devuelve todos los géneros que coinciden con el nombre Maxillaria. En este caso el dominio es `gen`, el operador es = y el valor es `Maxillaria`.

La tabla resumen búsqueda dominio te dice los nombres de los dominios de búsqueda, y por el dominio de búsqueda, los campos que se buscan.

La cadena de búsqueda `loc like block %` devolverá todas las ubicaciones cuyo nombre o código comience en «block». En este caso el dominio es `loc` (una abreviatura para `location`), el operador es `like` (esto viene del SQL y permite buscar «fuzzy»), el valor es `block %`, los campos implícitamente examinados son `name` y `code`. El signo de porcentaje es utilizado como comodín por lo que si se busca `block %` se encontrarán todos los valores que comienzan con `block`. Si usted busca `%10`, busca todos los valores que terminan en `10`. La cadena `%ck %10` busca todos los valores que contienen `ck` y acaban en `10`.

Cuando una consulta toma edades para completar

Dar una consulta, se necesita tiempo para calcular, el resultado contiene injustificadamente muchas entradas. Esto sucede cuando desea utilizar una estrategia, pero las cadenas no forman una expresión válida. En este caso Ghini cae de nuevo a * búsqueda por valor *. Por ejemplo la cadena de búsqueda “” gen lik maxillaria”” buscará la cadenas “” gen””, “” como “” y “” maxillaria””, devolver todo lo que coincida con al menos uno de los tres criterios.

Búsqueda de binomio

También puede realizar una búsqueda en la base de datos si sabes la especie, simplemente colocando unas letras iniciales del género y especie epítetos en el motor de búsqueda correctamente capitalizado, es decir: ** género epíteto ** con una letra capital principal, ** especie epíteto ** todo en minusculas.

De esta manera usted puede realizar la búsqueda “” así ha””.

Estos serían las iniciales de Solanum hayesii, o Solanum havanense.

La búsqueda de binomio viene a compensar la limitada utilidad de la búsqueda anterior por la expresión cuando se trata de buscar una especie.

Es la capitalización correcta ** Xxxx xxxx ** que informa el software de su intención de realizar una búsqueda de binomio. Guess segundo del software será una búsqueda por valor, que posiblemente resultará en más partidos que había esperado.

La solicitud similar “” por lo que ha”” volverá de nuevo, en una instalación nueva, más de 3000 objetos, empezando por la familia «Acalyp(**ha**)ceae», y terminando en geografía «Western (** para **) uth America».

Buscar por consulta

Las consultas «Query» permiten el máximo control de búsqueda. Con consultas usted puede buscar a través de las relaciones, columnas específicas, combinar criterios de búsqueda utilizando operadores booleanos como `and`, `or`, `not`, y sus abreviaciones `&&`, `||`, `!`, encerrar en paréntesis, y más.

Póngase en contacto con los autores si desea más información, o si usted voluntario para documentar esto más a fondo. Mientras tanto pueden empezar a familiarizarse con la estructura de base de base de datos de Ghini.

Figura 1: **la estructura esencial de la base de datos Ghini**

Para nombrar algunos ejemplos:

- plantaciones de la familia Fabaceae en ubicación bloque 10:

```
plant WHERE accession.species.genus.family.epithet=Fabaceae AND␣
↪location.description="Block 10"
```

- lugares que no contienen plantas:

```
location WHERE plants = Empty
```

- accesiones, asociados a una especie de conocido nombre binomial (p. ej.: Mangifera indica):

```
accession WHERE species.genus.epithet=Mangifera AND species.
↪epithet=indica
```

- accesiones propagó en el año 2016:

```
accession WHERE plants.propagations._created BETWEEN␣
↪|datetime|2016,1,1| AND |datetime|2017,1,1|
```

- accesiones que modificó en los últimos tres días:

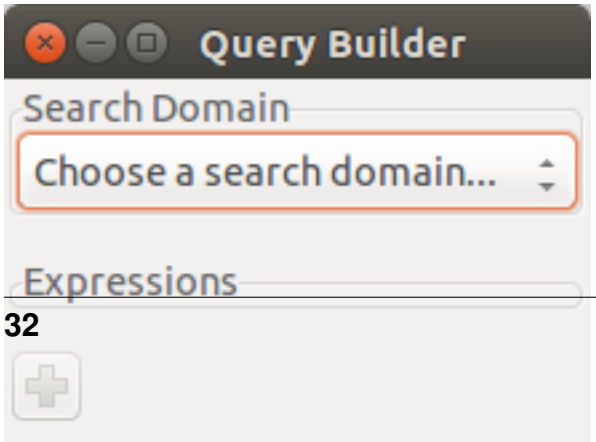
```
accession WHERE _last_updated>|datetime|-3|
```

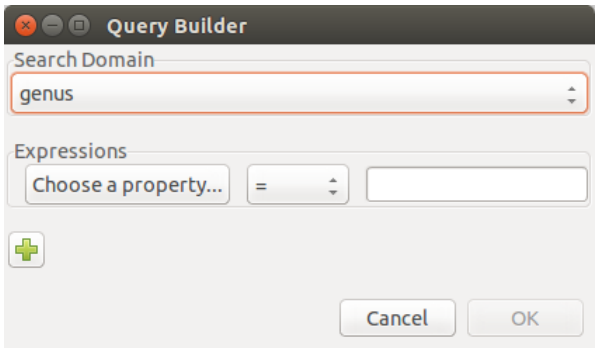
Búsqueda con las consultas requiere algunos conocimientos de la sintaxis un poco y una idea de la extensa estructura de tabla de base de datos de Ghini. Ambos adquieren con la práctica y con la ayuda del generador de consultas.

3.2.2 El generador de consultas

Ghini ofrece un generador de consultas, que le ayuda a construir consultas complejas de búsqueda a través de un punto y haga clic en interfaz. Para abrir el clic Query Builder el | query-builder | icono a la izquierda de la entrada de búsqueda o seleccione: menuselection: "Herramientas–> generador de consultas" en el menú.

Una ventana aparecerá, que le llevará a través de todos los pasos necesarios para construir una consulta correcta que se entiende por estrategia de búsqueda de consulta de Ghini.





En primer lugar indica el dominio de búsqueda, esto permitirá que el generador de consultas completo su interfaz gráfica de usuario, luego añades tantas cláusulas lógicas como usted necesita, conectándolas con una “” y “” o “” o “” operador binario.

Cada apartado está formado por tres partes: una propiedad que se puede llegar desde el inicial dominio de búsqueda, un operador de comparación que seleccione de la lista desplegable, un valor que puede escribir o seleccionar de la lista de valores válidos para el campo.

Agregar muchos buscar propiedades como usted necesita, haciendo clic en el signo más. Seleccione o próximo al nombre de propiedad para elegir cómo se combinarán las cláusulas en la consulta de búsqueda.

Cuando haya terminado de construir la consulta haga clic en Aceptar para realizar la búsqueda.

En este momento el generador de consultas escribe la consulta en el campo de búsqueda y la ejecuta. Ahora puedes editar la secuencia como si hubiera escrito usted mismo. Observe cómo los valores de lado de mano derecha son interpretados por el generador de consultas y encerrados en comillas simples si los reconoce como cadenas, dejados como están si se ven como números ó son uno de los dos textos reservados `None` o `Empty`. Usted puede editar la consulta e insertar comillas si necesita, por ejemplo, si usted necesita buscar literalmente la cadena `Empty`.

`None` es el valor de un campo vacío. No es lo mismo que la cadena larga cero `''` ni el numérico `0` ni el boolean `False` ni el set `Empty`: `None` indica que el campo no tiene valor en absoluto.

`Empty` es el conjunto vacío. Siendo un conjunto, puede ser comparado con otros conjuntos (por ejemplo: todas plantas de una accesión o todas accesiones de una especie), no con elementos (por ejemplo: cantidad de una planta, que es un número, o una descripción de un lugar, que es un texto). Sin embargo, el generador de consultas no le permite elegir un valor de lado de mano izquierda en un conjunto, sino que espera la selección de un campo. Escoja cualquier campo: en el momento de producción de la consulta, cuando el generador de consultas se encuentra con una cláusula con valor de lado de mano derecha el literal `Empty`, el nombre del campo es ignorado y le permite comparar el conjunto de la izquierda con el `Empty` de la derecha.

No tenemos literales `False` y `True`. Estos son valores con tipo, y el generador de consultas

no sabe cómo producirlos. En lugar de `False` escriba `0` y en lugar de `True` escriba `1`.

3.2.3 Gramática de consulta

Para los que no temen un poco de precisión formal, el código siguiente BNF te da una idea bastante precisa de la gramática aplicada la estrategia de búsqueda de la consulta. Informalmente se definen algunas categorías gramaticales; cualquier falta que quedan a su fértil imaginación; los literales se incluyen comillas simples; la gramática es sobre todo mayúsculas y minúsculas, a menos que se indique lo contrario:

```
query ::= domain 'WHERE' expression

domain ::= #( one of our search domains )
expression ::= signed_clause
             | signed_clause 'AND' expression
             | signed_clause 'OR' expression
             ;
signed_clause ::= clause
                | 'NOT' clause  #( not available in Query Builder)
                ;
clause ::= field_name binop value  #( available in Query Builder)
         | field_name set_binop value_list
         | aggregated binop value
         | field_name 'BETWEEN' value 'AND' value
         | '(' expression ')'
         ;
field_name ::= #( path to reach a database field or connected table␣
↪)
aggregated ::= aggregating_func '(' field_name ')'
aggregating_func ::= 'SUM'
                   | 'MIN'
                   | 'MAX'
                   | 'COUNT'
                   ;
value ::= typed_value
        | numeric_value
        | none_token
        | empty_token
        | string_value
        ;
typed_value ::= '|' type_name '|' value_list '|'
numeric_value ::== #( just a number )
none_token ::= 'None'    #( case sensitive )
empty_token ::= 'Empty'  #( case sensitive )
string_value = quoted_string | unquoted_string

type_name ::= 'datetime' | 'bool' ;  #( only ones for the time␣
↪being )
quoted_string ::= '"' unquoted_string '"'
```

(continué en la próxima página)

(proviene de la página anterior)

```
unquoted_string ::=  #( alphanumeric and more )

value_list ::= value ',' value_list
             | value
             ;
binop ::= '='
        | '=='
        | '!='
        | '<>'
        | '<'
        | '<='
        | '>'
        | '>='
        | 'LIKE'
        | 'CONTAINS'
        ;
set_binop ::= 'IN'
```

Tenga en cuenta que el lenguaje de consulta de Ghini es bastante un poco más complejo que lo que puede producir el generador de consultas: las consultas se pueden construir con el generador de consultas forman un subconjunto apropiado de consulta reconocido por el software:

```
query ::= domain 'WHERE' expression

domain ::= #( one of our search domains )
expression ::= clause
             | clause 'AND' expression
             | clause 'OR' expression
             ;
clause ::= field_name binop value
         ;
field_name ::= #( path to reach a database field or connected table␣
↪)
value ::= numeric_value
        | string_value
        ;
numeric_value ::== #( just a number )
string_value = quoted_string | unquoted_string ;

quoted_string ::= '"' unquoted_string '"'
unquoted_string ::=  #( alphanumeric and more )

binop ::= '='
        | '=='
        | '!='
        | '<>'
        | '<'
        | '<='
        | '>'
```

(continué en la próxima página)

(proviene de la página anterior)

```
        | '>='
        | 'LIKE'
        | 'CONTAINS'
        ;
```

3.3 Editar e insertar información

La principal forma que añadir o cambiar información en Ghini es mediante el uso de los editores. Cada tipo de base de datos tiene su propio editor. Por ejemplo hay un editor de familia, un editor de género, un editor de accesión, etcetera.

Para crear un nuevo registro clic en la: menuselection: “Insertar” en la barra de menu menú y seleccione el tipo de registro su desea crear. Esto abre un nuevo editor en blanco para el tipo.

Para editar un registro existente en la base de datos haga clic derecho sobre un elemento en los resultados de la búsqueda y selección: menuselection: “Editar” en el menú emergente. Se abrirá un editor que le permite cambiar los valores en el registro seleccionado.

La mayoría también tienen hijos que pueden agregar haciendo clic derecho sobre el padre y seleccionando «Add??? … «en el menú contextual. Por ejemplo, una familia tiene niños de género: puede Agregar un género a una familia haciendo clic derecho sobre una familia y selección de «Add genus».

3.3.1 Notas

Casi todos los editores en Ghini tienen una pestaña *Notas*, que funciona del mismo modo, independientemente del editor.

Si introduce una dirección web en una nota y el enlace aparece en el enlaces de la caja cuando el elemento está edición se selecciona en resultados de la búsqueda.

Puedes buscar las notas para un artículo en la base de datos mediante el cuadro de notas en la parte inferior de la pantalla. Si el elemento seleccionado no tiene notas desensibilizar el cuadro notas.

3.3.2 Familia

El familia editor le permite agregar o cambiar una familia botánica.

La * familia * campo en el editor le permite cambiar el epíteto de la familia. El campo de la familia es necesario.

La * calificador * campo le permite cambiar la clasificación de la familia. El valor puede ser * sensu lato *, * sensu stricto *, o nada.

Synonyms allow you to add other families that are synonyms with the family you are currently editing. To add a new synonyms type in a family name in the entry. You must select a family

name from the list of completions. Once you have selected a family name that you want to add as a synonym click on the Add button next to the synonym list and the software adds the selected synonym to the list. To remove a synonym, select the synonym from the list and click on the Remove button.

Para cancelar los cambios sin guardar y haga clic en el botón *cancelar*.

Para salvar a la familia con que está trabajando haga clic en *Aceptar*.

Para salvar a la familia con que está trabajando y agregar un género, haga clic en el botón *añadir géneros*.

Para agregar otra familia cuando termine de editar la actual, haga clic en *siguiente* en la parte inferior. Esto guarda la familia y abre un nuevo editor de familia en blanco.

3.3.3 Género

El editor de géneros le permite añadir o cambiar un género botánico.

El campo *familia* en el editor de género le permite elegir la familia para el género. Cuando usted comience tipeando un nombre de familia se le mostrará una lista de familias para elegir. El nombre de familia ya debe existir en la base de datos antes de establecer como la familia para el género.

El campo *género* le permite definir el género de esta entrada.

El campo *autor* le permite definir el nombre o la abreviatura de los autores del género.

Synonyms allow you to add other genera that are synonyms with the genus you are currently editing. To add a new synonyms type in a genus name in the entry. You must select a genus name from the list of completions. Once you have selected a genus name that you want to add as a synonym click on the Add button next to the synonym list and it will add the selected synonym to the list. To remove a synonym select the synonym from the list and click on the Remove button.

Para cancelar los cambios sin guardar y haga clic en el botón *cancelar*.

Para salvar el género que está trabajando en y haga clic en * Aceptar *.

Para salvar el género que está trabajando en y añade una especie que haga clic en la * especie * añadir botón.

Para añadir otro género cuando haya terminado de editar el actual clic en la * siguiente * botón en la parte inferior. Esto ahorrará el género actual y abrir un nuevo editor de género blanco.

3.3.4 Especie O Taxón

Por razones históricas llamados una “especie”, pero por esto nos referimos a una “taxonomía” fila “especies” o inferior. Representa un nombre único en la base de datos. El editor de especies le permite construir el nombre, así como asociar metadatos a la taxonomía como su distribución, de sinónimos y otra información.

La * partes * en el editor de especies infraespecíficas permite especificar "taxonomía" más en el rango de "especies".

Para cancelar los cambios sin guardar y haga clic en el botón *cancelar*.

Para salvar a las especies que está trabajando en y haga clic en * Aceptar *.

Para salvar a las especies que está trabajando en añade una accesión a él y haga clic en la * accesión * añadir botón.

Para agregar otra especie cuando haya terminado de editar el actual clic en la * siguiente * botón en la parte inferior. Esto salvar la especie actual y abrir un nuevo editor de especies en blanco.

3.3.5 Accesiones

El editor de accesión nos permite añadir accesiones a una especie, y la puesta al día de la información correspondiente. En Ghini una accesión representa un grupo de plantas o clones que son todas del mismo taxón, fueron propagadas en la misma manera y el mismo tratamiento, fueron recibidas de la misma fuente, y al mismo tiempo.

Elegir el nombre taxonómico, agregar uno si usted se olvidó de hacerlo por adelantado.

Puede exprimir incertidumbre en la identificación mediante la adición de un calificador de identificación, al rango adecuado, para que puedas tener por ejemplo una planta inicialmente identificada como *Iris* cf. *florentina* eligiendo *Iris florentina* en el nombre del taxón, calificador de la identificación "cf.", aplicándose al rango especie.

Escriba la ID de la accesión, preferiblemente también la cantidad recibida.

Fuente de la accesión

La fuente de las accesiones te permite añadir más información sobre de donde vino esta accesión. Seleccione un contacto de la lista desplegable y elija «Propagación de jardín», que se coloca como primer elemento predeterminado en la lista de contactos.

Una propagación de jardín es el resultado de un intento de propagación exitoso.

Al acceder a material de una propagación de jardín, inicialmente se deja la primera pestaña (General) y se empieza de la segunda pestaña (Fuente). Seleccione como contacto «Propagación de jardín», indique cual planta es la planta madre y escoja entre las propagaciones todavía no completamente accesadas, la que usted desea agregar como una accesión en la base de datos.

Cuando seleccione una propagación, el software llena varios campos en la ficha General, que ahora puede revisar. La taxonomía (tal vez se pudo obtener algo ligeramente distinto a la planta madre). La cantidad inicial (en caso que no todas las plantas van en la misma accesión). El Tipo de Material, deducido del tipo de propagación.

3.3.6 Planta

El la base de datos Ghini una `Plant` describe una planta individual en la colección. Una planta pertenece a una accesión, y se encuentra en una ubicación específica.

Creación de múltiples plantas

Puede crear múltiples plantas mediante el uso de rangos en la entrada de código. Esto sólo es permitido al crear nuevas plantas y no es posible editar las plantas existentes en la base de datos.

Por ejemplo la gama, 3-5 se formará planta código 3,4,5. La gama 1.4-7.25 creará plantas con códigos 1,4,5,6,7,25.

Al entrar en la gama de la entrada de código de planta la entrada dará vuelta al azul para indicar que ahora está creando múltiples plantas. Los campos que se establecen en este modo se copiará en todas las plantas que se crean.

Fotos

Como a casi todos los objetos en la basedatos Ghini pueden ser asociadas *notas*,a plantas y alas especies se pueden asociar *fotos*: al lado de la ficha de notas, el editor de plantas contiene una ficha adicional para las fotos. Se pueden asociar tantas fotos quantas usted quiera.

Al momento de asociar una imagen a una planta, el software copia el archivo en la carpeta de fotos, mientras una miniatura (500x500) es generada y copiada en la carpeta "miniaturas" dentro de la carpeta de fotos.

A partir de Ghini-1.0.62, fotos no se mantienen en la base de datos. Para asegurar imágenes están disponibles en todas las terminales donde se han instalado y configurado Ghini, puede utilizar una unidad de red o un servicio como Tresorit o Dropbox para compartir archivos.

Recuerde que ha configurado la carpeta de raíz de fotografías cuando usted especifica los detalles de su conexión a la base de datos. Una vez más, debe asegurarse de que la carpeta raíz de cuadros es compartida con el archivo de servicio de la opción de uso compartido.

When a Plant or a Species in the current selection is highlighted, its pictures are displayed in the pictures pane, the pane left of the information pane. When an Accession in the selection is highlighted, any picture associated to the plants in the highlighted accession are displayed in the pictures pane.

In Ghini-1.0, pictures are special notes, with category «<picture>», and text the path to the file, relative to the pictures root folder. In the Notes tab, Picture notes will show as normal notes, and you can edit them without limitations.

A Plant is a physical object, so you associate to it pictures taken of that individual plant, taken at any relevant development stage of the plant, possibly helping its identification.

Species are abstract objects, so you would associate to it pictures showing the characteristic elements of the species, so it makes sense to associate a flora illustration to it. You can also do

that by reference: go to the Notes tab, add a note and specify as category «<picture>», then in the text field you type the URL for the illustration of your choice.

3.3.7 Ubicaciones

Editor de Ubicaciones

zona de peligro

El editor de ubicación contiene una sección inicialmente oculta llamada * zona de peligro *. Los widgets incluidos en esta sección permiten al usuario combinar la ubicación actual en una ubicación diferente, dejando al usuario corregir errores de ortografía o implementar cambios en la política.

3.4 Tratar con propagaciones

Ghini ofrece la posibilidad de asociar ensayos de propagación a las plantas y para documentar sus tratamientos y resultados. Los tratamientos son parte integrante de la descripción de una prueba de propagación. Si un ensayo de propagación tiene éxito, Ghini le permite asociarla a una nueva accesión. Sólo puede asociar una accesión a un ensayo de propagación.

Aquí describimos cómo usar esta parte de la interfaz.

3.4.1 Creación de una propagación

Una propagación (ensayo) se obtiene de una planta. Ghini lo refleja en su interfaz: Seleccione una planta, abra el Editor de planta en él, activar la pestaña de propagación, haga clic en Agregar.

Cuando haces lo anterior, se obtiene una ventana de Editor de propagación. Ghini no considera ensayos de propagación como entidades independientes. Como resultado, Ghini trata al Editor de propagación como una ventana de editor especial, que sólo se puede llegar desde el Editor de planta.

Para una nueva propagación, se seleccione el tipo de propagación (esto se convierte en una propiedad inmutable de la propagación) luego insertar los datos que lo describen.

Usted podrá editar los datos de propagación a través de la misma ruta: Seleccione una planta, abra el Editor de la planta, identificar la propagación que desea editar, haga clic en el botón Editar. Podrás editar todas las propiedades de una juicio, excepto el tipo de propagación existente.

En el caso de una propagación de semilla prueba, usted tiene un padre de polen y un padre de la semilla. Siempre se debe asociar el ensayo de propagación a los padres de semilla.

Nota: En Ghini 1.0 especifica la planta polen del padre en el «Notes» del campo, mientras Ghini 1.1 tiene un campo (relación) para él. Según ITF2, puede haber casos en los ensayos de propagación de semilla donde no se sabe que planta que papel. Una vez más, en Ghini 1.0 debe usar una nota para indicar que si este es el caso, Ghini 1.1 tiene un campo (booleano) que indica si este es el caso.

3.4.2 Usando una propagación

Un ensayo de propagación puede ser exitoso y producir una nueva accesión.

Ghini le ayuda a reflejar esto en la base de datos: crear una nueva accesión, inmediatamente cambie a la pestaña fuente y seleccione «Garden Propagation» en el campo (aunque algo engañoso).

Comience a escribir el número de plantas y aparecerá una lista de plantas que empareja con ensayos de propagación para que usted seleccione.

Seleccione la planta, y en la mitad inferior de la ventana aparecerá la lista de los ensayos de propagación a e unaccessed.

Seleccione un ensayo de propagación todavía unaccessed de la lista y haga clic en Aceptar para completar la operación.

Usando los datos del ensayo de propagación, Ghini completa algunos de los campos en la ficha General: nombre de la taxonomía, tipo de material y posiblemente de procedencia. Usted podrá editar los campos, pero tenga en cuenta que el software no impedirá introducir inconsistencias conceptuales en su base de datos.

Puede asociar un ensayo de propagación a la sola accesión.

3.5 Tagging

Tagging is an easy way to give context to an object or create a collection of object that you want to recall later.

The power in this tagging action is that you can share this selection with colleagues, who can act on it, without the need to redo all your collecting work.

For example if you need to print accession labels of otherwise unrelated plants, you can group the objects by tagging them with the string «relabel». You or one of your colleagues can then select «relabel» from the tags menu, the search view will show all the objects you tagged, and performing a report will act on the tagged objects.

Tagging acts on the active selection, that is the items in the search results which you have selected.

Please remember: you can select all result rows by pressing `Ctrl-A`, you can deselect everything by pressing `Ctrl-Shift-A`, you can toggle tagging of a single row by `Ctrl-Mouse click` on it.

Once you have an active selection, tagging can be done in two ways:

3.5.1 dialog box tagging

Press `Ctrl-T` or select *Tag→Tag Selection* from the menu, this activates a window where you can create new tags and apply any existing tag to the selection.

The tag window is composed of three parts:

1. The upper part mentions the list of objects in your active selection. This is the list of object of which you are editing the tags;
2. The middle part has a list of all available tags, with a checkbox that you can activate for applying the tag to or removing the tag from the selection;
3. The lower part only holds a link to new tag creation, and the Ok button for closing the dialog box.

If, when opening the tag dialog box, the active selection holds multiple items, then only the tags that are common to all the selected items will have a check next to it. Tags that only apply to a proper subset of the active selection will show with an "undecided" status. Tags that don't apply to any object in the active selection will show blank.

The most recently created tag, or the most recently selected tag becomes the active tag, and it shows with a check next to it in the tags menu.

3.5.2 windowless tagging

Once you have an active tag, pressing `Ctrl-Y` applies the active tag to all objects in the active selection. `Ctrl-Shift-Y` removes the active tag from all objects in the active selection.

3.6 Producir informes

A database without exporting facilities is of little use. Ghini lets you export your data in table format (open them in your spreadsheet editor of choice), as labels (to be printed or engraved), as html pages or pdf or postscript documents.

3.6.1 The Report Tool

You activate the Report Tool from the main menu: *Tools→Report*. The Report Tools acts on a selection, so first select something, then start the Report Tool.

Report on the whole collection.

To produce a report on your whole plant collection, a shortcut would be from the home screen, to click on the `Families: in use` cell.

> If your focus is more on the garden location than on taxonomy and accessions, you would click on the `Locations: total` cell.

Reports are produced by a report engine, making use of a report template. Ghini relies upon two different report engines (Mako & XSL), and offers several report templates, meant as usable examples.

Choose the report you need, specify parameters if required, and produce the report. Ghini will open the report in the associated application.

Configuring report templates, that's a task for who installs and configures ghini at your institution. Basically, you create a template name, indicating the report engine and specifying the template. Configured templates are static, once configured you are not expected to alter them. Only the special `**scratch**` template can be modified on the fly.

The remainder of this page provides technical information and links regarding the formatter engines, and gives hints on writing report templates. Writing templates comes very close to writing a computer program, and that's beyond the scope of this manual, but we have hints that will definitely be useful to the interested reader.

3.6.2 Uso del formateador de informes Mako

El formateador de informes Mako utiliza su propio idioma de plantilla para generar informes. Puede encontrar más información sobre el Mako y su lenguaje en makotemplates.org

El sistema de plantillas de Mako ya debería estar instalado en su equipo si está instalado Ghini.

Creación de informes con Mako es similar en lo que sería crear una página web desde una plantilla. Es mucho más simple que la Formatter(see below) XSL y debería ser relativamente fácil de crear plantilla para cualquier persona con un poco pero de experiencia de programación.

El generador de la plantilla produce archivos con la misma extensión que la plantilla. Por ejemplo, para generar una página HTML, el nombre de su plantilla debe ser algo como "report.html". Si la plantilla generará un archivo de valor separado por comas, nombre la plantilla "report.csv".

La plantilla va a recibir una variable llamada "valores" que contendrán la lista de valores en la búsqueda actual.

El tipo de cada valor en los "valores" será el mismo que el dominio de búsqueda utilizado en la consulta de búsqueda. Para más información sobre dominios de búsqueda ver: ref: "dominios de búsqueda".

Si la consulta no tiene un dominio de búsqueda, los valores pueden ser de diferente tipo y la plantilla de Mako debe preparan para manejarlas.

3.6.3 Utilizar al formateador de informe XSL

El formateador XSL Informe requiere una XSL para el representador PDF para convertir los datos en un archivo PDF. Apache FOP es un XSL-> representador de PDF gratuito y de código

abierto y es recomendado.

Si utiliza Linux, Apache FOP debe ser usando su gestor de paquetes instalable. En Debian/Ubuntu es instalable como “” fop”” en Synaptic o usando el siguiente comando:

```
apt-get install fop
```

Instalación de Apache FOP en Windows

Tienes dos opciones para la instalación de FOP en Windows. La forma más sencilla es descargar su instalador **‘ApacheFOP-0.95-1-setup.exe <http://code.google.com/p/apache-fop-installer/downloads/detail?name=ApacheFOP-0.95-1-setup.exe&can=2&q=#makechanges >‘_**.

En alternativa, puede descargar el fichero. Tras descomprimir el fichero debe añadir el directorio extraído a su variable de entorno PATH.

3.7 Importación y exportación

Aunque Ghini puede extenderse a través de complementos para soportar formatos alternativos de importación y exportación, por defecto sólo puede importar y exportar archivos de valores separados por comas o CSV.

Hay algún apoyo para exportarlo al acceso de datos de colecciones biológicas es limitada.

También hay soporte limitado para exportar a un formato XML que refleja más o menos exactamente las tablas y una fila de la base de datos.

Exportación de ABCD y XML será no cubierto aquí.

> **Advertencia:** Importación de archivos probablemente destruirá los datos que tienes en la base de datos, así que asegúrese de que usted tiene copia de sus datos.

3.7.1 Importar desde CSV

En general es mejor sólo importar archivos CSV a Ghini previamente exportados de Ghini. Es posible importar cualquier archivo CSV pero que está más avanzada que cubrirá este doc.

Para importar archivos CSV en Ghini seleccione :seleccione menú:‘Herramientas - >Exportar–>Valores Separados por Comas » en el menú.

Tras pulsar OK en el diálogo que solicita confirmación de la acción, se abrirá un selector de fichero. Seleccione los ficheros que desee importar en el selector de ficheros.

3.7.2 Exportar a CSV

Para exportar los datos de Ghini en formato CSV, seleccionar en el menu *Tools→Export→Comma Separated Values*

Esta herramienta le pedirá que seleccione un directorio para exportar los datos CSV. Todas las tablas en Ghini se exportarán a los archivos en el formato tablename.txt donde NombreTabla es el nombre de la tabla donde los datos se exportan desde.

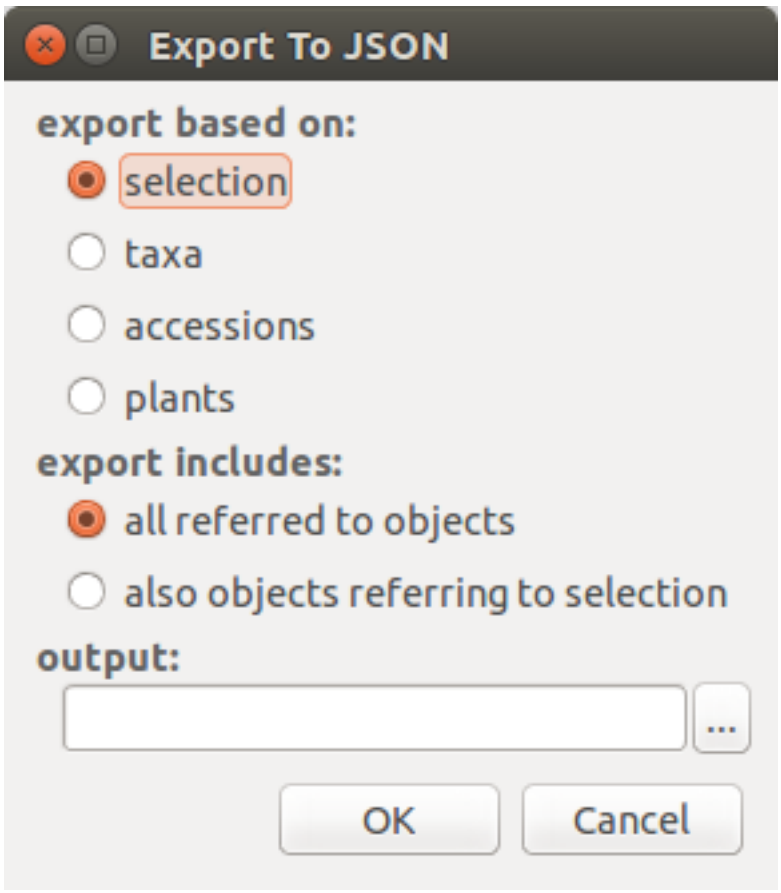
3.7.3 Importación de JSON

Es * la * manera de importar datos en una base de datos existente, sin destruir el contenido anterior. Un ejemplo típico de esta funcionalidad sería importar su colección digital en una base de datos de Ghini fresco, sólo inicializado. Convertir una base de datos en formato de intercambio de json de adorno es más allá del alcance de este manual, por favor contacto uno de los autores si usted necesita cualquier ayuda adicional.

Utilizando el formato de intercambio de json Ghini, puede importar datos que han exportado desde una instalación diferente de Ghini.

3.7.4 Exportación a JSON

Esta función está todavía en desarrollo.



al activar esta herramienta de exportación, le dan la opción para especificar qué exportar. Puede usar la selección actual a limitar la duración de la exportación, o puede empezar en el contenido completo de un dominio a elegir entre especies, accesión, planta.

Exportar *Especies* exportará su base de datos solo pero toda información taxonómica. Seleccionando *Accesión* se exportarán sus accesiones y toda la información taxonómica utilizada: taxones sin utilizar no se exportarán. Seleccionando *Planta* se exportarán todas las plantas vivientes (algunos accesiones pueden no ser incluidas), todos lo referido a ubicaciones y taxa.

3.7.5 Importing from a Generic Database

Esta funcionalidad es objeto de la issue #127, para que todavía no tenemos solución genérica.

If you're interested in importing data from some flat file (e.g.: Excel spreadsheet) or from any database, contact the developers.

3.7.6 Importar una colección de imágenes

Podemos mirara a una colección de fotografías de plantas, en que cada una es claramente asociada a una sola planta, como una forma particular de base de datos botánicos.

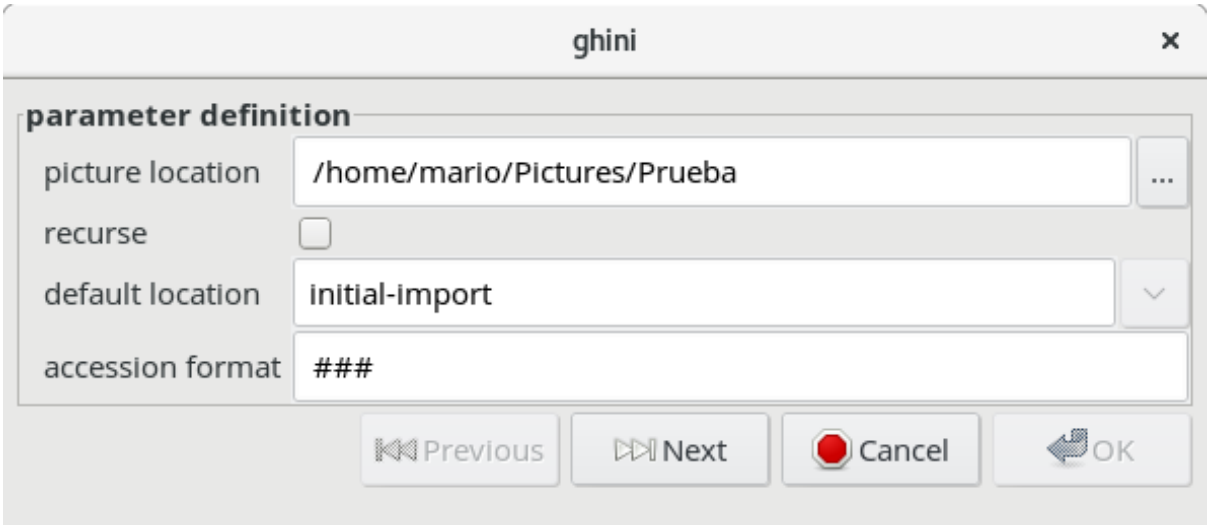
Aún sin utilizar un software específico para colecciones de fotos, es posible asociar fotos a accesiones, respetando un mismo estilo en dar nombre a los archivos.

Por ejemplo, `2018.0020.1 (4) Epidendrum.jpg` podría ser el nombre de la cuarta foto asociada a la planta número 1 de la accesión veinte del año 2018, y está identificada a rango género como Epidendrum.

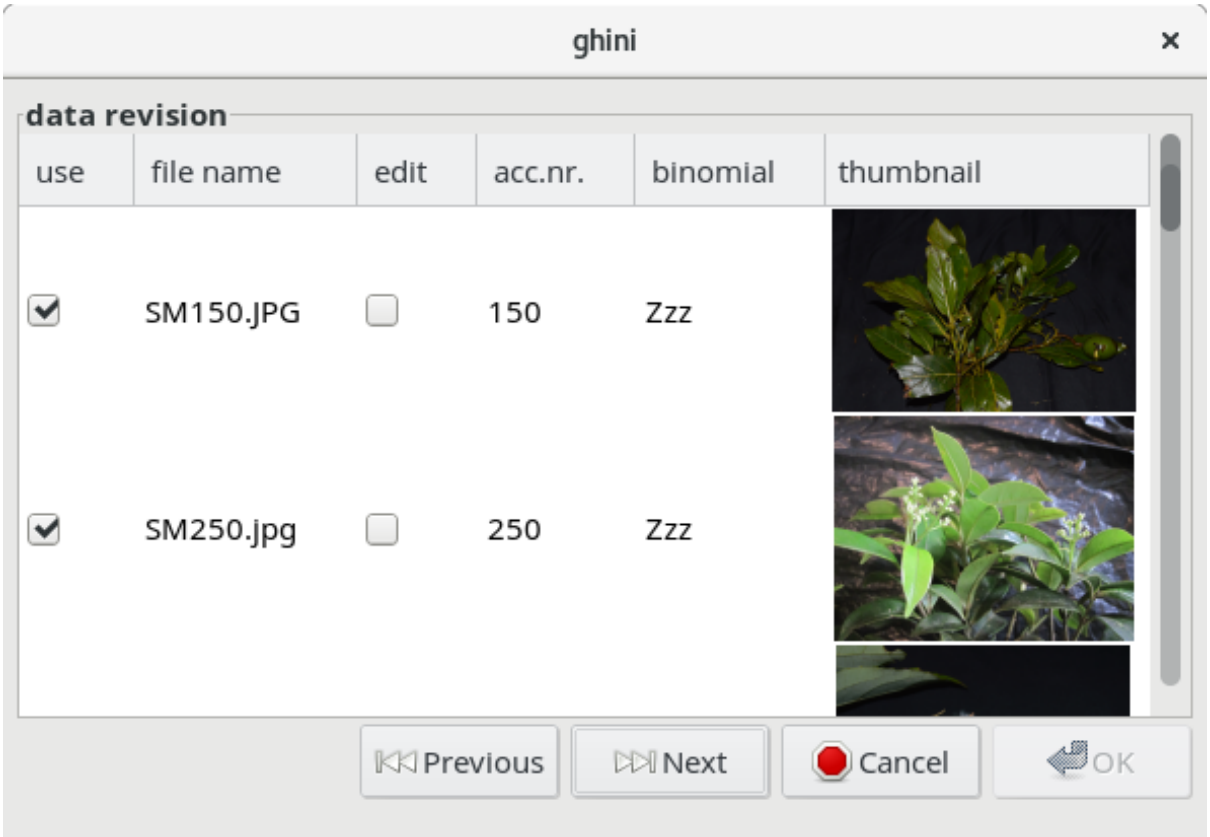
La función *Herramienta→Importar→Fotos* aquí descrita permite de utilizar una colección de fotos para inicializar una base de datos ghini, o para añadir información de forma periódica.

Use *Tools→Import→Pictures* to activate this import tool. Import goes in several steps: parameter definition; data revision and confirmation; the import step proper; finally review the import log. At the first two steps you can confirm the data and go to the next step by clicking on the `next` button, or you can go back to the previous step by clicking on the `prev` button. Once the import is done and you're reviewing the log, you can only either confirm —or abort— the whole transaction.

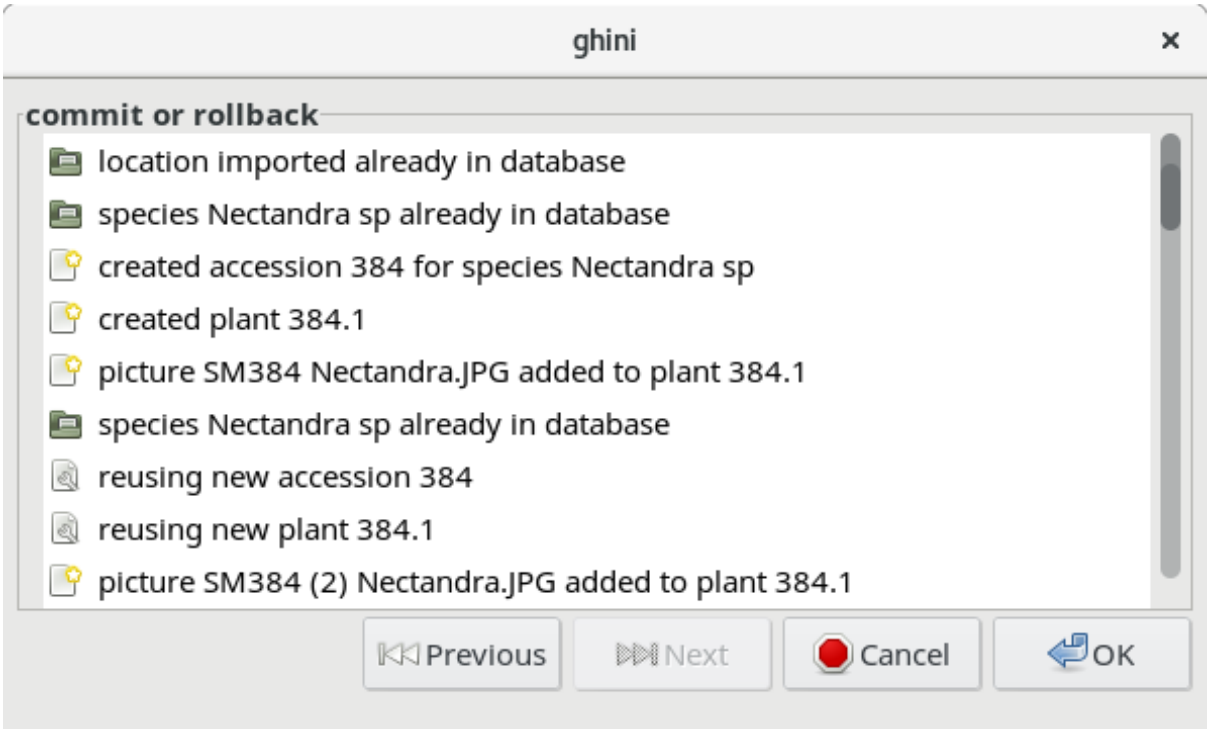
En el panel «definición parámetros» usted puede: seleccionar el directorio de que desea importar fotografías; indicar si desea importar fotos de forma recursiva; seleccionar o crear una ubicación que se utilizará como ubicación predeterminada para nuevas plantas; describir el estilo seguido para dar nombre a los archivos.



In the «data revision» pane you are shown a table with as many rows as the pictures you are importing. Each row holds as much information as the tool managed to extract from the picture name. You can review the information, correct or confirm, and indicate whether or not the row should be imported.



In the final «commit or rollback» pane you read the logs relative to your data import, and decide whether to keep them (commit them to the database), or undo them (rollback the transaction).



When the Picture Collection importer creates or updates objects, it also sets a Note that you can use for selecting the objects involved in the import, and for reviewing if needed.

3.8 Cómo gestionar usuarios

Nota: Sólo está disponible en bases de datos PostgreSQL basado en el plugin de los usuarios de Ghini.

The Ghini Users Plugin will allow you to create and manage the permissions of users for your Ghini database.

You must log in to your database as a user with `CREATEROLE` privilege in order to manage other users.

3.8.1 Creación de usuarios

Para crear un usuario nuevo...

3.8.2 Permisos

Ghini permite la lectura, escritura y ejecución.

CAPÍTULO 4

Recetario

4.1 Colección de recetas aportadas

Esta página presenta listas de casos de uso. Si busca información práctica y directa, estás en el lugar correcto. Si prefiere una presentación exhaustiva de la estructura de la base de datos y led software, consulte la sección software para jardines botánicos

Todo el material ha sido aportado por jardines usuarios del software y compartiendo sus experiencias con la comunidad de usuarios.

Los autores del software le agradecen de corazón.

4.1.1 Jardín Botánico de Quito

En el JBQ, Jardín Botánico de Quito, hemos adoptado el software de Ghini en abril de 2015. Desde entonces, hemos ido acumulando experiencia con el programa, y somos nosotros mismos en la necesitad de documentarla, para asegurar el conocimiento a la institución. Estamos encantados de compartirlo.

Detalles técnicos

- Trabajamos en GNU/Linux, una plataforma que no muchos usuarios dominan, y nuestra base de datos está dentro de un sistema de gestión de base de datos remota. Esto implica medidas que no son evidentes para la mayoría.

Como arrancar un programa

Para iniciar un programa dado su nombre, active la tecla  , al lado de Alt, o haga clic en  , a continuación, comience a escribir el nombre del programa, en nuestro caso «Ghini», o simplemente haga clic en el símbolo del programa  , que aparece cerca de la margen izquierda de la pantalla.

Servidor de bases de datos

Elegimos para un servidor de bases de datos centralizado PostgreSQL. Así estamos protegidos de conflictos entre cambios concurrentes, y todos los cambios están disponibles simultáneamente en todos los clientes de ghini. Tenemos, porsupuesto, que externalizar la gestión del servidore de bases de datos.

Agregar un nuevo usuario

En el registro de los cambios a la base de datos, Ghini también graba el nombre de usuario. En nuestro jardín, aparte de los usuarios institucionales, tenemos usuarios temporales habilitados a la escritura a la base de datos. Esto nos hizo decidir que mejor permitimos a Ghini tener un registro completo de los eventos de base de datos.

Dado que trabajamos usando PostgreSQL, los usuarios grabados por Ghini en el historial son los usuarios de la base de datos, no los del sistema.

Cada usuario conoce su propia contraseña, y sólo conoce la suya. El responsable del contenido de la base datos también tiene la contraseña del usuario `bauble`, que sólo lo usamos para crear usuarios.

No utilizamos nombres de cuenta como `voluntario`, porque tales cuentas no nos ayudan asociar el nombre a la persona.

— Agregar un nuevo usuario de sistema (linux/osx)

Agregar un usuario del sistema es no es estrictamente necesario, pues ghini no los usa en los registros, sin embargo, hacerlo permite la separación de las preferencias, configuración de conexiones, historial de búsqueda. En algunos de nuestros sistemas contamos con una sola cuenta compartida, con varias conexiones configuradas, en otros sistemas tenemos una cuenta por usuario.

En sistemas con una cuenta por cada usuario, nuestros usuarios tienen una sola conexión configurada, y tenemos la contraseña de la base de datos en el archivo `home/<account>/.pgpass`. Este archivo sólo es legible para el usuario `<account>`.

En sistemas con una cuenta compartida, el usuario tiene que seleccionar su

conexión propia y escribir la contraseña correspondiente.

Estos son los pasos para agregar usuarios del sistema:

```
sudo -k; sudo adduser test
sudo adduser test adm; sudo adduser test sudo
sudo adduser test sambashare; sudo adduser test ghini
```

— Agregar un nuevo usuario de base de datos

Ghini tiene una interfaz muy simple para la gestión de usuarios, que solo funciona con postgresql y mucho carece de mantenimiento. Hemos abierto *issues* esperando poderla utilizar, por el momento usamos el script `create-role.sh`:

```
#!/bin/bash
USER=$1
PASSWD=$2
shift 2
cat <<EOF | psql bauble -U bauble $@
create role $USER with login password '$PASSWD';
alter role $USER with login password '$PASSWD';
grant all privileges on all tables in schema public to
 ↪$USER;
grant all privileges on all sequences in schema public␣
 ↪to $USER;
grant all privileges on all functions in schema public␣
 ↪to $USER;
EOF
```

El `alter role` siguiente el `create role` es redundante, pero nos permite aplicar la misma secuencia de comandos para la corrección de cuentas existentes.

Nuestra base de datos de ghini se llama `bauble`, y `bauble` es también el nombre de nuestro super usuario base de datos, el único usuario con privilegio `CREATEROLE`.

Por ejemplo, la invocación siguiente crearía el usuario `willem` con contraseña `orange`, en la base de datos `bauble`, en el anfitrión 192.168.5.6:

```
./create-role.sh willem orange -h 192.168.5.6
```

- Entender cuándo se debe actualizar

Actualizar el sistema

Poner al día Ubuntu es una tarea mucho más liviana que poner al día Windows. Si el sistema sugiere una puesta al día, se puede dejárselo hacer sin

problemas: mientras tanto el sistema estará a disposición y nunca te obligará a reinicializar.

Actualizar Ghini

La primera ventana presentada por Ghini se ve así. Normalmente, usted no necesita hacer nada en esta ventana, sólo tiene que pulsar Conectar y seguir a la pantalla principal del programa.

Ocasionalmente, en la parte superior de la pantalla aparece un texto de información, diciéndole que una versión más reciente está disponible en línea.

El procedimiento de actualización es sencillo, y depende del sistema operativo que utilice, no estamos explicando aquí de nuevo.

Es generalmente una buena idea actualizar el software. En caso de duda, póngase en contacto con el autor, o escribir al grupo.

- Entender la pantalla inicial de ghini

Pantalla completa

En el momento de la escritura, nuestra pantalla inicial se veía así:

Aparte del menú principal de la aplicación, el interfaz de Ghini ofrece tres secciones especiales, con información y herramientas para explorar la base de datos.

Resumen numérico

La tabla en la mitad derecha de la pantalla muestra un resumen de todas las plantas registradas. Cada entrada en negrita es un enlace a la consulta que selecciona los objetos correspondientes.

	total	in use	unused
Families:	511	7	504
Genera:	25394	158	25236
Species:	637	623	14
Accessions:	7722	7675	47
Plants:	7676	7676	0
Locations:	170	163	7

Consultas almacenadas

La mitad inferior del lado derecho contiene un conjunto de consultas almacenadas. Usted las puede editar a su gusto, sin embargo nuestras sugerencias incluyen seleccionar aquellas accesiones que no han sido identificados en rango especie. Y una para la historia de la base de datos.

Botones de acción y consulta

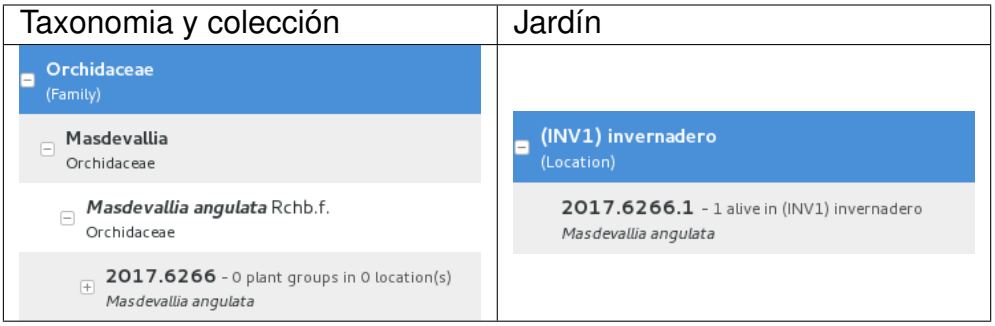
En la parte superior de esta pantalla usted puede encontrar el campo en el que introducir sus búsquedas.

- Con el  botón, en la forma de una casa, usted puede volver de sus búsquedas en la pantalla principal.
- Con el  botón en forma de flecha, puede volver a su última búsqueda.
- Con el botón  en forma de engranaje, usted puede iniciar la «Query Builder», que le ayuda a componer complejas búsquedas en una simple forma gráfica.

- A menudo tenemos voluntarios que trabajan en el jardín sólo por un tiempo muy corto. Es con ellos en la mente que hemos desarrollado una vista hipersimplificada en la estructura de la base de datos de ghini.

Detalles

Las dos figuras aquí muestran todo lo que necesitan saber nuestros colaboradores temporales.



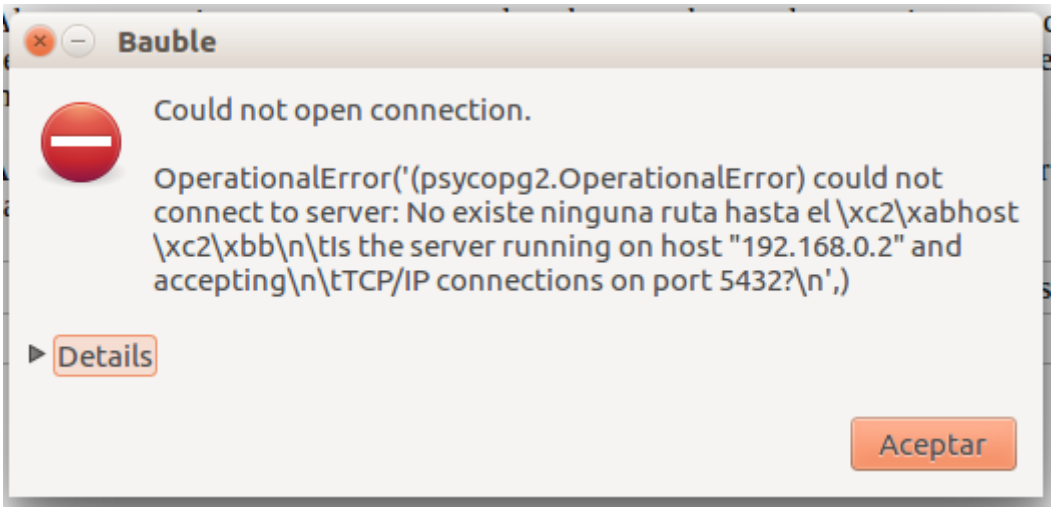
- A veces, el programa da mensajes de error. 

, reintente, o infor-

me a los desarrolladores.

Problemas de red

Para trabajar, el programa necesita una conexión de red estable con el servidor de base de datos. Puede suceder que: se inicia el programa, y no se puede conectar a nuestro servidor de base de datos. El mensaje de error obtenido es bastante explícito pero también muy mal compuesto.

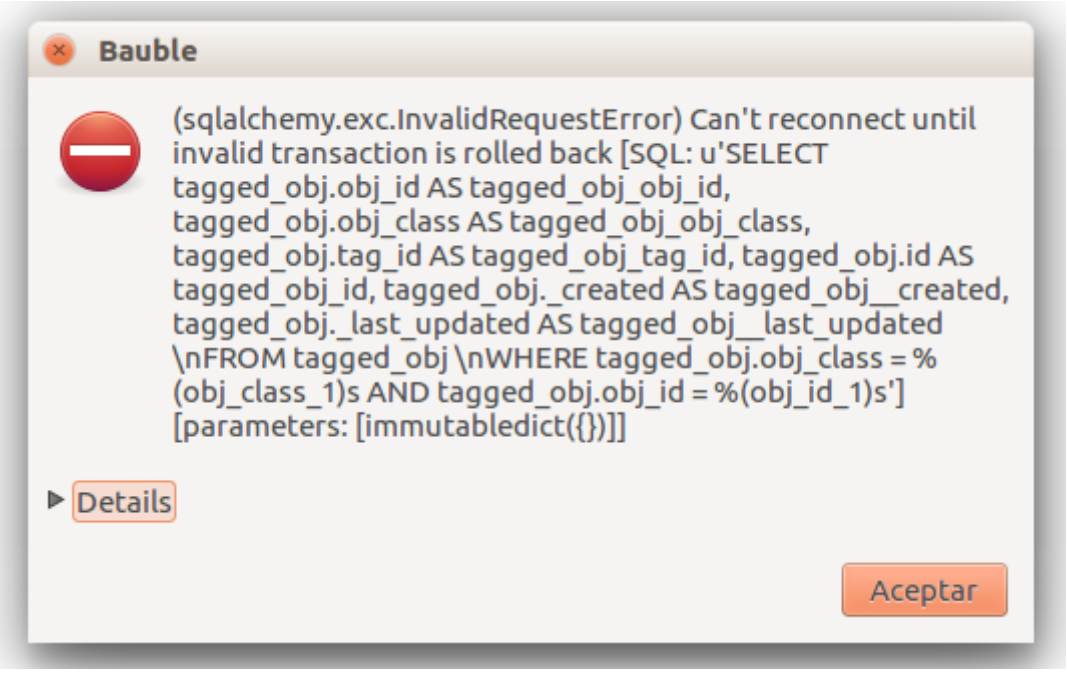


Sólo ignorarlo y vuelva a intentarlo.

La búsqueda falla con error

A veces sin causa aparente, una búsqueda no se ejecutará correctamente y aparecerá una ventana con un mensaje de error. En este caso sólo tienes que intentar realizar la misma búsqueda otra vez.

Un ejemplo de un mensaje de error:



La búsqueda no devuelve algo que acabo de insertar

Códigos de accesión que empiezan con cero y compuesto de sólo cifras, como por ejemplo `016489` son considerados por el software como números, así que si usted no incluye la cadena de búsqueda entre comillas, cualquier 0 inicial será eliminado y el valor no se encuentra.

Vuelva a intentarlo, pero encierran su cadena en comillas simples o dobles.

Número en la etiqueta	búsqueda correspondiente
16489	“016489”

Nota: cuando usted busque un código de planta, no una accesión, los ceros iniciales se vuelven opcionales, por lo que en el ejemplo anterior tal vez es más fácil escribir `16489.1`.

- Una situación grave sucedió una vez, y absolutamente queremos evitar que vuelva a suceder: un usuario elimina un género, con todo lo que estaba abajo, especies y accesiones y sinonimias.

Solucionandolo con permisos de usuario

Una forma sería a través de diferentes perfiles, asociado a usuarios de base de datos, cada usuario con permisos diferentes.

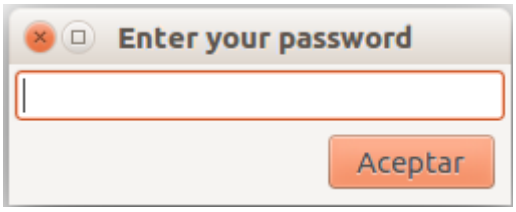
Permiso completo (BD-JBQ) Solamente el personal calificado obtiene este tipo de acceso.

Insert y update (BD-JBQ-limitado) Usamos éste para aquellos usuarios

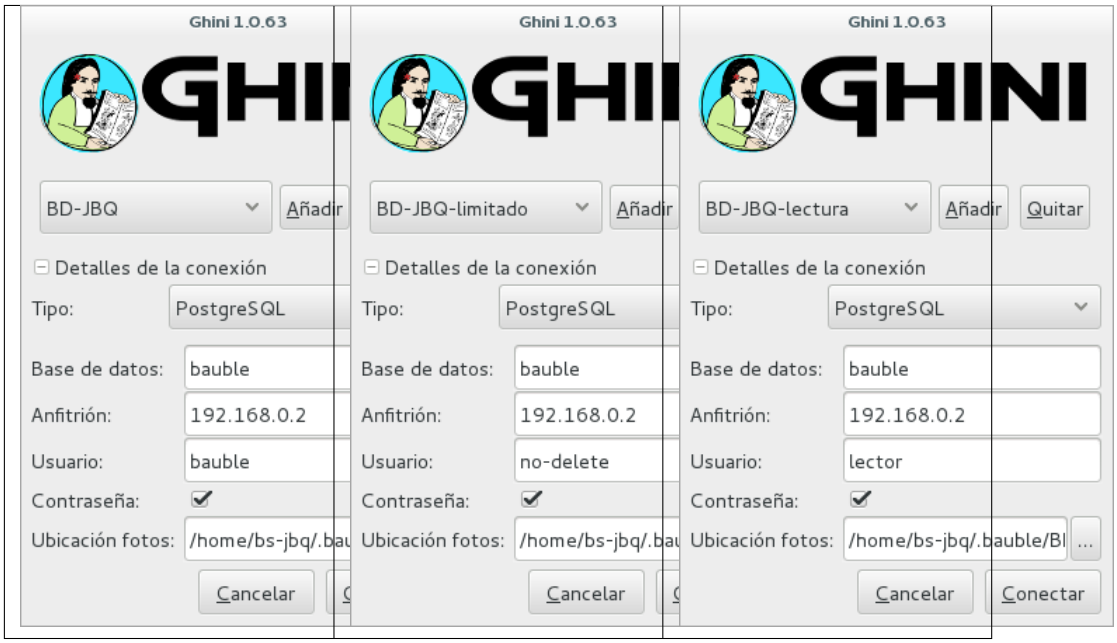
que vienen a ayudarnos por un tiempo limitado, y que no recibieron una completa introducción a los conceptos de base de datos. Está destinado a evitar errores costosos.

Sólo lectura (BD-JBQ-lectura) puede ser compartido con cualquier persona que visita el jardín

Seleccione la conexión en el inicio, y el software le pide la contraseña correspondiente a la conexión seleccionada.



Si usted quiere revisar los detalles de la conexión, haga clic en el junto a "Detalles de la conexión", cambiará a y la ventana de conexión aparecerá como una de las siguientes:



Como se puede ver, estamos conectando al mismo servidor de base de datos, cada conexión utiliza la misma base de datos en el servidor, pero con diferente usuario.

Pensándolo más

Por otro lado, se estamos cuestionando si es del todo apropiado, dejar que cualquier usuario borre algo a un nivel tan alto como una familia, o un género, o, para lo que importa, de cualquier cosa que implique la eliminación inmediata de accesiones de la colección.

En ghini la forma de cuestionar las características de software es abriendo un asunto correspondiente.

- Al contactar los desarrolladores, sin duda pedirán información técnica o por lo menos para ver una captura de pantalla. Ayúdales a ayudarte.

Tomar una captura de pantalla

En Linux hay tres formas de crear una captura de pantalla, todas implican golpear la tecla "PrtSc". Posiblemente la más práctica es pulsando la tecla "PrtSc" en combinación con Ctrl y Shift. Esto iniciará una herramienta de copia de la pantalla interactiva. Selecciona un rectángulo y este es copiado en el portapapeles. Pegarlo en el correo que estás escribiendo, o en la línea de chat donde los desarrolladores están tratando de ayudarle.

Donde están los log

Ghini escribe continuamente en un archivo de registro muy informativo, `~/.bauble/bauble.log`. No se moleste en abrirlo, envíelo no más. Contiene mucha información técnica.

Alerta continua no tripulada

Otra opción es activar el controlador Sentry. Notificará a nuestro servidor Sentry de cualquier situación grave en el software. Si te has registrado, los desarrolladores sabrá cómo comunicarse con usted si es necesario.

Para el paranoico de buena salud: no estamos supervisando lo que estás haciendo, estamos supervisando el funcionamiento de nuestro software. Siempre puede optar hacia fuera.

Activar el controlador Sentry en la página `:prefs`: buscar la fila con el nombre "" bauble.use_sentry_handler"", si el valor es no lo que desea, haga doble clic en la línea y cambiará a otro valor.

Taxonomía

- Introducción

Complejidad taxonómica de Orchidaceae

En el JBQ, trabajamos sobre todo con orquídeas, familia Orchidaceae, una de las más grandes familias de plantas, con no menos de 850 géneros, organizados – según Dressler, en aproximadamente 70 subtribus, 22 tribus, 5

subfamilias. Cómo representamos esta información no es obvio y necesita ser explicado.

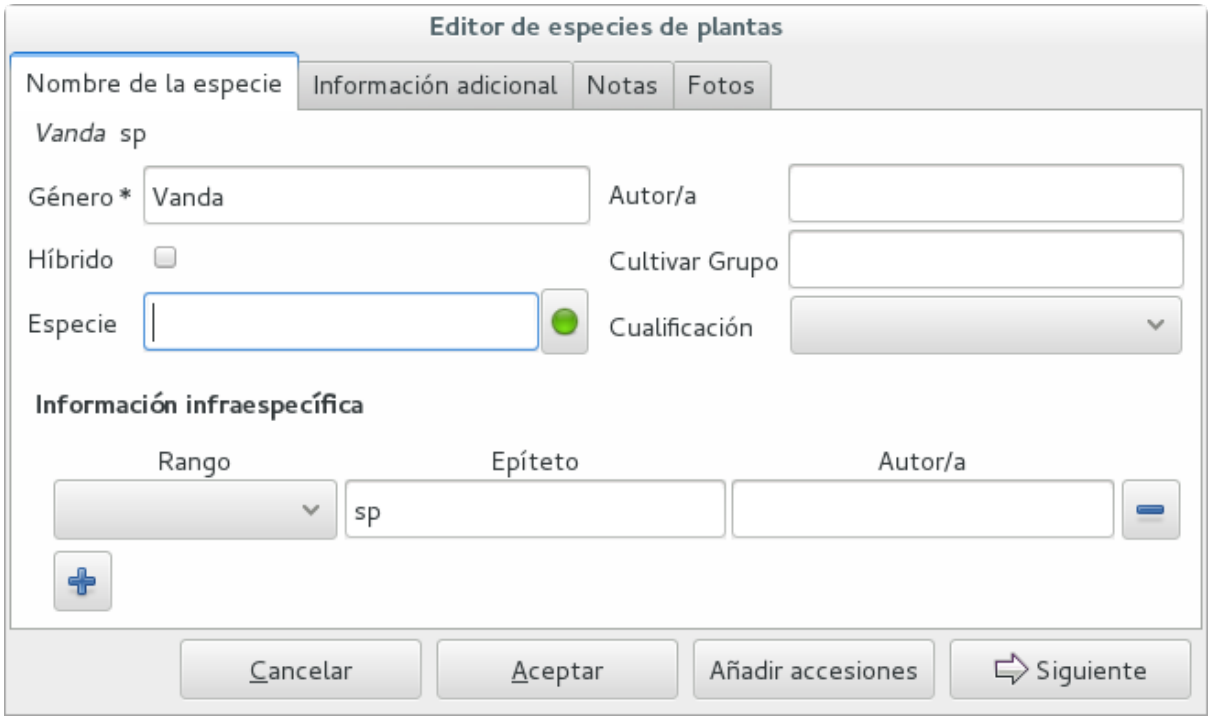
La taxonomía de la familia Orchidaceae es continuamente revisada. Géneros se agregan, se rechazan, se reorganizan, son reconocido como sinónimos, algunos taxónomos prefieren otra agrupación de especies o géneros, otros vuelven a dividir y de manera otra vez diferente, los botánicos de diferentes nacionalidades pueden tener diferentes opiniones sobre el asunto. Todo esto suena muy complejo y especializado, pero es parte de nuestra rutina diaria, y que puede ser almacenado en nuestra base de datos Ghini.

- Identificar a rango género o familia

Al rango género

Ghini 1.0 prescribe que una accesión se identifique al rango especie, en todos los casos. El responsable actual reconoce que esto es un error, procedente de los primeros días de Bauble y que Ghini 1.0 tiene en común con otros software botánicos. Hasta que esto se corrija, contamos con las prácticas establecidas.

Si se identifica una accesión al rango género, agregamos una especie ficticia en ese género, sin especificar su epíteto de especie (pues lo desconocemos) y añadimos un epíteto sin clasificar en la sección de información infraespecífica, como este:



Cuando aparece en un resultado de búsqueda, se muestra como este:

Al rango familia

Si una accesión es identificada sólo al rango de la familia, también necesitamos un género ficticio, a que podemos añadir la especie ficticia. Puesto que nuestro jardín se concentra principalmente en Orchidaceae, utilizamos el nombre corto **Zzz** para el género ficticio dentro de la familia, como este:

El responsable actual sugiere utilizar el prefijo **Zzz-** y detrás del prefijo escribir el nombre de la familia, posiblemente eliminando la **e** final. La eliminación de la **e** final es útil para no obtener resultados que incluyan nombres de género cuando busca cosas que terminen en **aceae**.

Con la excepción del ya mencionado género **Zzz**, de la familia Orchidaceae, sí seguimos esta práctica aconsejada, así que por ejemplo nuestra colección incluye *Zzz-cactacea* y *Zzz-bromeliacea*.

Recuerde: nuestro **Zzz** representa un género desconocido en la familia **Orchidaceae**, no lo utilice como género desconocido en otras familias.

- Identificar en un rango que no está permitido por el software (por ejemplo: Subtribu o subfamilia)

Al rango subtribu

A veces no podemos identificar un taxón en el género de rango, pero logramos ser más precisas que tan sólo «es una orquídea». Muy a menudo somos capaces de indicar la subtribu, esto es útil cuando se desee producir híbridos.

El software no nos permite almacenar rangos intermedios entre familia y género, así que tuvimos que inventar algo, y esto es lo que hacemos:

Insertamos un género ficticio, nombrando como la Subtribu, prefijando con "Zzx-", como en este ejemplo:

Este Zzx-Laeliinae es algún género de la subtribu Laeliinae.

Con el fin de poder seleccionar géneros por subtribu, también vamos a añadir una nota al género ficticio Zzx-Laeliinae, así como para todos los géneros en la misma subtribu: categoría subtribus, valor el nombre de la subtribu.

Esto permite consultas como:

```
genus where notes.note=Laeliinae
```

También esperamos con impaciencia que el problema-9 sea solucionado!

Al rango subfamilia o tribu

Como hemos reservado el prefijo Zzx - para la subtribu, nos reservamos los prefijos Zzy - para la tribu, Zzw - para la subfamilia.

En particular, la información de la subfamilia es relevante, porque hay subfamilias dentro de la familia Orchidaceae, que no están más separados.

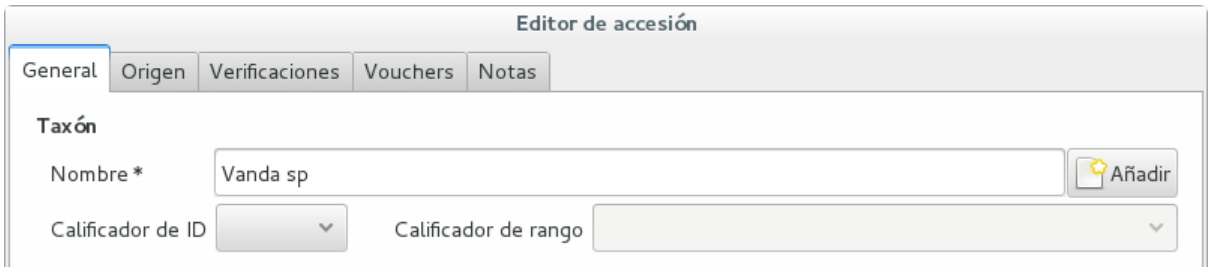
- Edición de la identificación de accesión - los detalles de las especies

---

Especie comodín para accesiones individuales

Escenario uno describe la identificación de una accesión individual, que había sido asociada a una especie comodín genérica, algo así como «*Zzz* sp» o «*Vanda* sp»;

En este caso, cuando se da a conocer las especies de la planta, cambiamos la asociación en la accesión, seleccionando una especie diferente.



No editamos la especie comodín, pues esta puede estar conectada a otras accesiones totalmente sin relación con la que identificamos.

Especie desconocida para accesiones múltiples

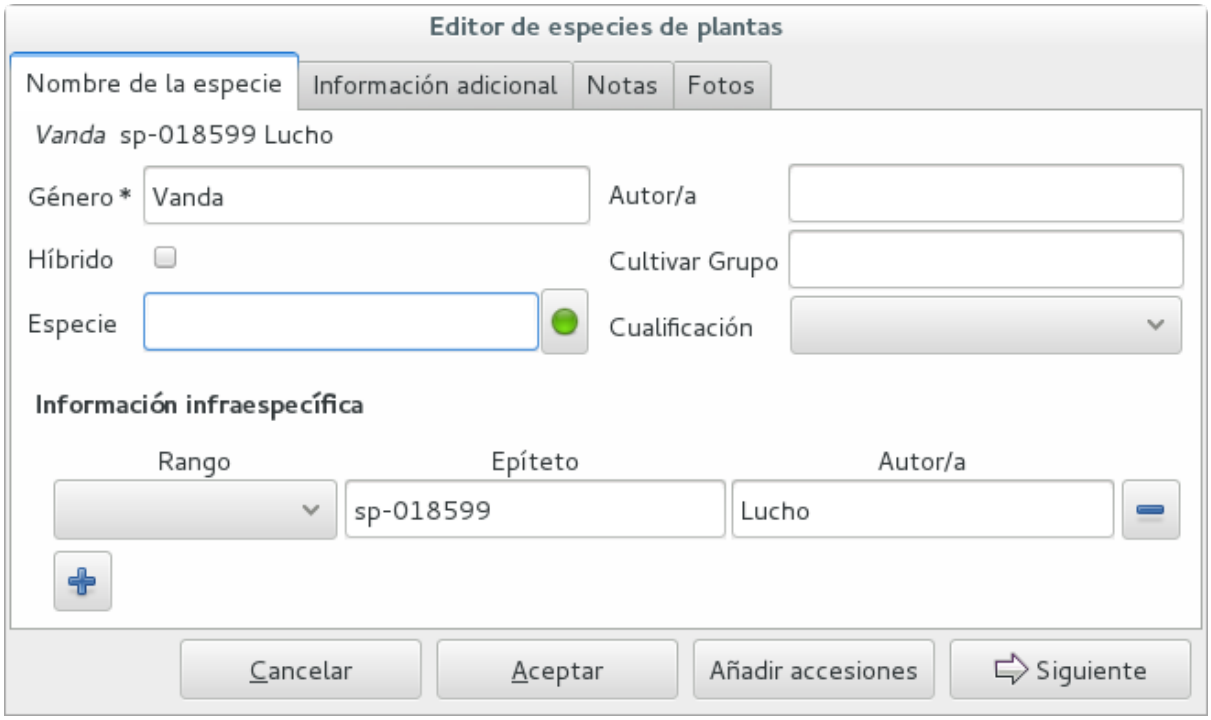
Un caso diferente es cuando tenemos un lote entero de accesiones, todas obviamente la misma especie, pero no hemos sido capaces de identificarla. En este caso, asociamos las accesiones con una especies incompletamente especificada, algo así como «*Zzz* sp-59», preferiblemente añadiendo el nombre del taxónomo, que hizo la asociación.

Una especie como «*Vanda* sp-018599» no es una especie comodín, es una especie muy concreta, que aún no hemos identificado.

En este caso, cuando la especie es identificada (y podría incluso ser una nueva especie), editamos directamente la especie, así que todas sus accesiones recibirán el cambio.

- Una nueva planta es relativa a una especie que todavía no está en nuestra colección.

Especie de último minuto



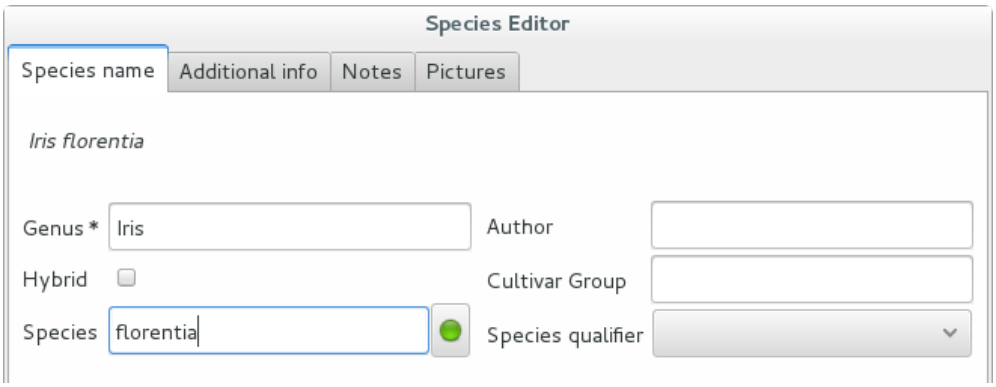
Comenzamos este de la Ventana de Accesiones y es muy sencillo, simplemente haga clic en el **+** al lado del nombre de la especie, nos metemos en la Ventana de Especie.

- Añadir una especie y utilizar servicios taxonómicos en línea

---

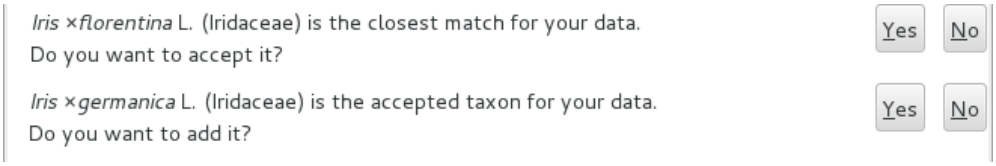
Añadir una nueva especie — the plant list.

Empezamos la manera obvia: insertar el epíteto de género, posiblemente lo seleccione de la lista de finalización, luego el epíteto de la especie, o al menos su mejor aproximación.

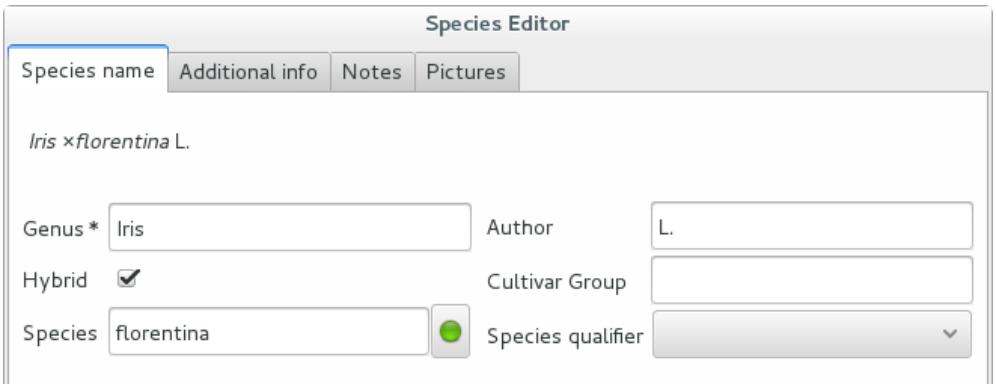


Al lado del campo del epíteto de especie hay un pequeño botón,  , que nos une a theplantlist. Haga clic en él, un área de mensajes aparece en la parte superior de la ventana.

Dependiendo de la velocidad de su conexión a internet, pero también de qué tan cerca su conjetura de un nombre correcto publicado, la zona superior cambiará a algo como esto:



Aceptar la sugerencia y será como si usted hubiera escrito los datos usted mismo.



Revisar una selección entera — TNRS.

Esto se describe en el manual, es muy útil, no se olvide de él.

Que la base de datos refleje lo que es el jardín

- Una tarea interminable es revisar lo que tenemos en el jardín para que coincida con lo que tenemos en la base de datos.

Estado inicial y recursos variables

Cuando adoptamos ghini, importamos en él todo lo que teníamos en una base de datos filemaker. Esa base de datos se centraba únicamente en orquídeas y aún así estaba lejos de ser completa. En la práctica, nos encontramos todavía con plantas etiquetadas en el jardín que nunca se han insertado en la base de datos.

De vez en cuando, logramos obtener recursos para revisar el jardín, comparándolo con la colección en la base de datos, y la principal actividad es insertar códigos de accesión a la base de datos, tomar fotos de la planta en cuestión y anotar su ubicación, todas tareas que se describen en el resto de esta sección.

La pequeña aplicación android ghini pocket fue añadida a la familia Ghini en ocasión de una visita de un programador Ghini al Jardín Botánico de Quito. ghini.pocket nos permite llevarnos en el bolsillo una copia reducida del contenido de la base datos, pero también nos facilita la rápida realización de un inventario

---

Procedimiento de Inventario

Ingresamos a ghini.pocket, anotamos el nombre de la locación donde se estará realizando el inventario. Ejem (INV 1) para invernadero 1. Anotamos o escaneamos la accesión (si la planta cuenta con código de barra o código QR) y se realiza la búsqueda.

Ghini.pocket marca la fecha con horario, la ubicación, el código buscado en un archivo textual que luego podrá ser importado a la base de datos.

Para un invernadero con alrededor de 1000 plantas el tiempo estimado a realizarse es de dos días, en un horario de 8:00 am a 5:00 pm, trabajando sin apuro.

Luego de haber importado el archivo generado por ghini.pocket, es fácil evidenciar cuales plantas faltan. Por ejemplo: si hicimos el inventario del INV3 del 4 al 5 de septiembre, la búsqueda correspondiente es:

```
plant where location.code = 'INV3' and not notes.note␣
↪like '2017090%'
```

Todas estas plantas se pueden marcar como pérdidas o muertas, según decida el Jardín.

Evidenciar la necesidad de identificación taxonómica.

Nuestro protocolo incluye otro detalle pensado para evidenciar las plantas que necesitan la atención de un taxónomo.

Las plantas que solo aparezcan en nuestra base de datos identificadas a nivel familia o que no estaban en la base de datos reciben una señal visual (por ejemplo: un palito plástico o de madera para paletas o papas fritas), para resaltar que no está identificada y de esta forma el taxónomo o encargado cuando realice un recorrido por el invernadero pueda identificarlas y proceder a agregar la identificación en la base de datos.

- Convención de nombres en lugares del jardín

Detalles

code	descripción
CAC-B *x*	Solo las cactáceas afuera de los orquidearios en el jardín
CRV:	Exposición de Nepenthaceae
IC-*xx*:	orquidearios de calor en el jardín (1A a 9C son lugares especificos entre del orquideario)
IF-xx:	orquidearios de frío en el jardín (1A a 5I son lugares específicos dentro del orquideario)
INV1:	invernadero 1 (calor)
INV2:	invernadero 2 (frío)
INV3:	invernadero 3 (calor)

- Añadir una accesión para una planta

Obviamente seguimos incrementando nuestra colección, con plantas provenientes de fuentes comerciales, o recolectados de la naturaleza, más raramente procedentes de expediciones a zonas remotas de nuestro país, o recibimos plantas que fueron recolectadas ilegalmente.

A veces tenemos que añadir plantas a la colección digital, sólo porque las tenemos físicamente, se encuentran en el jardín, con o sin su etiqueta, pero sin sus contrapartes digitales.

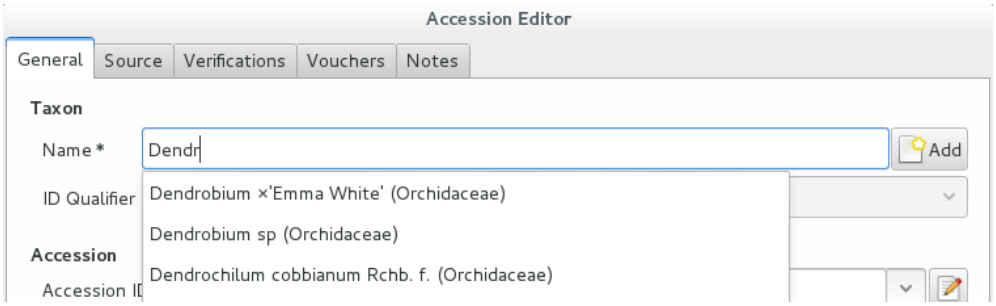
Planta existente, se encuentra en el jardín con su propia etiqueta

Esta actividad comienza con una planta, que se encuentra en una ubicación específica del jardín, con una etiqueta accesión y con el conocimiento de que el código de accesión no está en la base de datos.

Para este ejemplo, supongamos que se va a insertar la siguiente información en la base de datos.

Accesión	Especie	Ubicación
008440	*Dendrobium* ×"Emma White"	Invernadero 1 (calor)

Vamos directamente en el Editor de Accesión, comienzo escribiendo el nombre de la especie en el campo correspondiente. Por suerte, la especie estaba ya en la base de datos, de lo contrario utilizaría el botón **agregar** junto al campo de entrada.



Seleccionamos las especies correctas y llenamos unos campos más, dejando el resto a los valores por defecto:

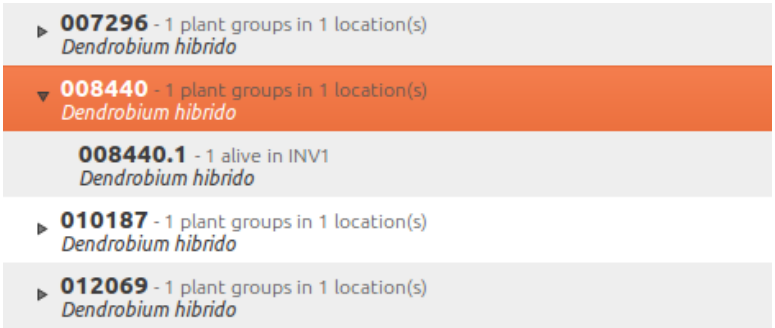
ID de la accesión	Tipo de Material	Cantidad	Procedencia
008440	Planta	1	Desconocido

Después de esto, seguimos en el editor de planta, haciendo clic en **Añadir plantas**.

No llenamos en la accesión los «**lugares previsto**», porque no sabemos cuál fue la intención original cuando la planta fue adquirida en primer lugar.

En la Planta Editor, podemos insertar la Cantidad y la Ubicación. Y hemos terminado.

La planta es ahora parte de la base de datos:



Planta formando un nuevo ingreso en la colección

Esta actividad comienza con una nueva planta, que se acaba de adquirir de una fuente conocida, una etiqueta de la planta y una ubicación deseada en el jardín.

Aquí hacemos más o menos lo mismo que para el caso de una planta que ya se encuentra en el jardín, hay dos diferencias: (1) conocemos la fuente de la planta; (2) adquirir esta planta fue una acción planificada, y tenemos la intención de colocarlo en un lugar específico en el jardín.

De nuevo, vamos directamente en al Editor de Accesión, comenzamos escribiendo el binomial de la especie y seleccionamos de la lista de las correspondencias, o añadir sobre la marcha.

ID de la accesión	Tipo de Material	Cantidad	Fuente
033724	Planta	1	especificado

Después de esto, seguimos en el editor de planta, haciendo clic en **Añadir plantas**.

En el Editor de Planta, insertamos la Cantidad y la Ubicación.

Tenga en cuenta que la planta puede ser colocada inicialmente en un invernadero, antes de que llegue su lugar previsto en el jardín.

Planta existente, en el jardín sin su etiqueta

Cuando esto sucede, no podemos estar seguros de que la planta nunca haya estado en la colección. Actuaremos como si re-etiquetando la planta. Esto se describe en la siguiente sección.

- Cuando asociamos físicamente una etiqueta a una planta, siempre hay la posibilidad de que algo ocurra a la planta (puede morirse) o la etiqueta (puede llegar a ser ilegible) o a la asociación entre las dos (pueden quedar separadas). Tenemos protocolos asistidos por software para estos eventos.

Encontramos una planta muerta

Cuando una planta es encontrada muerta, recogemos su etiqueta y la ponemos en una caja junto al terminal de inserción de datos, la caja está marcada «plantas muertas».

Definitivamente por lo menos una vez por semana, la caja se vacía y se actualiza la base de datos con esta información.

Las plantas muertas no son *eliminadas* de la base de datos, se quedan allí, pero consiguen una **cantidad** cero. Si se conoce la causa de la muerte, esto también se escribe en la base de datos.

Una vez más recuerde que una **planta** no es una **accesión** y por favor, recuerde que no quitar los objetos de la base de datos, sólo tiene que añadir a su historia.

Introduzca el código de la planta completo (algo así como `012345.1`, o `2017.0001.3` y no necesita ceros iniciales ni comillas), haga clic en la fila correspondiente y haga clic en **edit**. cambie la cantidad a 0, especifique la razón y de preferencia también la fecha del cambio.

Si necesita añadir más detalles acerca de la muerte de la planta, por favor, ponga una **nota**, y utilice la categoría «death_cause».

Plantas con **cantidad** cero se muestran con un color diferente en la vista de resultados. Esto ayuda a distinguirlos de las plantas vivas.

Encontramos una planta sin su etiqueta

No podemos saber si la planta haya anteriormente estado en la colección o no. Asumimos que sí, y que se perdió su etiqueta.

Perder una etiqueta de planta es lamentable, pero es algo que sucede. Lo que hacemos es poner una nueva etiqueta a la planta e indicamos claramente que la etiqueta es un reemplazo de un original.

Luego manejamos el caso como por una nueva accesión, además ponemos en la accesión una nota con categoría «etiqueta», texto «etiquetar».

- Hacer el seguimiento de diferentes fuentes de material vegetal

---

Qué diferentes fuentes podemos tener

En este jardín botánico, recibimos plantas de diferentes tipos de origen. Pueden ser de expediciones (plantas provenientes de la naturaleza, recogidas con permiso legal de MAE - Ministerio de Ambiente Ecuatoriano), plantas donadas sobre todo como regalos de coleccionistas o de empresas de comercialización de orquídeas, compradas, o plantas confiscadas (generalmente proveniente de incursiones del MAE a lo largo del país).

Si la planta proviene una fuente salvaje

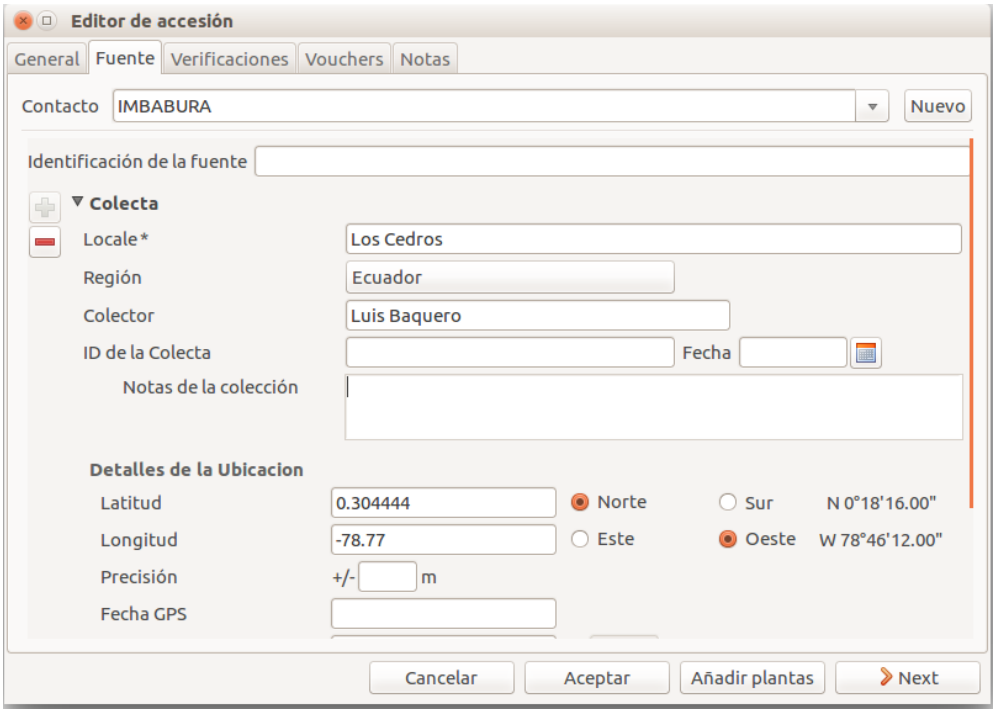
En la pantalla de accesión contamos con la opción «origen». Cuando una planta viene de una fuente salvaje, podemos especificar su origen. Dado que queremos cumplir con ITF2 y que ghini 1.0 respeta sólo en parte esa norma, hay que tener un poco de cuidado. Las opciones que cumplen con el ITF2 son:

- Salvaje: Accesión de fuente salvaje.
- Cultivado: Propagación de una planta de fuente salvaje.
- No salvaje: Accesión no rastreable a una fuente salvaje.
- Datos insuficientes

En el caso de una planta donada, es mejor poner información detallada así como una nota en la accesión de la planta; en el caso de una planta con un origen desconocido, seleccionamos la opción datos suficientes.

Como usar la ficha de la fuente en el editor de accesión

En esta sección podemos crear o utilizar un contacto, nuestra fuente de material vegetal. Podría ser de una expedición a un sitio de colecta, y en este caso especificar la región y el nombre de la expedición, o podría ser el nombre de la persona o empresa, que donó un lote específico de plantas.



Una vez que usted seleccione o cree la información de contacto, en esta sección se despliegan más opciones, aquí puede especificar la región, en que se puede elegir el país de origen, y una ubicación específica dentro de la región, la georreferenciación de la información (incluidos los datos de GPS), la descripción del hábitat, el nombre del colector. Por último, recomiendo también escribir al lado del nombre del colector, la fecha de colecta (ej. Luis Baquero 11/10/2016).

Plantas donadas, compradas o confiscadas

Aun así útil para expediciones o para donantes donde la información principal es geográfica, este tabulador de fuente no es muy práctico en nuestros casos restantes: manejamos tres más categorías: confiscado, adquirido y dado, para estas categorías las opciones disponibles en el tabulador de fuente no aplica: demasiada información y no al punto.

En estos casos, se añade un conjunto de notas, según el caso.

— Plantas donadas

Si la planta fue donada por un individuo, agregamos al individuo entre nuestros contactos y lo indicamos como fuente, a continuación añadimos las notas:

categoría	texto
source-type	donación
source-detail	Razón: Contribución científica JBQ

— Plantas compradas

Si la planta fue comprada, añadimos el dueño anterior entre nuestros contactos y lo indicamos como fuente, luego añadimos las notas:

categoría	texto
source-type	compra
source-detail	opcional, texto libre
factura	el número de la factura

— Plantas decomisadas

Si la planta fue confiscada, agregamos el dueño anterior entre nuestros contactos y lo indicamos como fuente, a continuación añadimos las notas:

categoría	texto
source-type	decomisada
source-detail	Posiblemente, detalles legales, número de ley ...

- Producción o reproducción de etiquetas

Renovar etiquetas de plantas

A veces hay que renovar las etiquetas, por ejemplo todo lo que está en un invernadero, o tal vez sólo un conjunto de plantas debido al riesgo que sus etiquetas se vuelvan ilegibles.

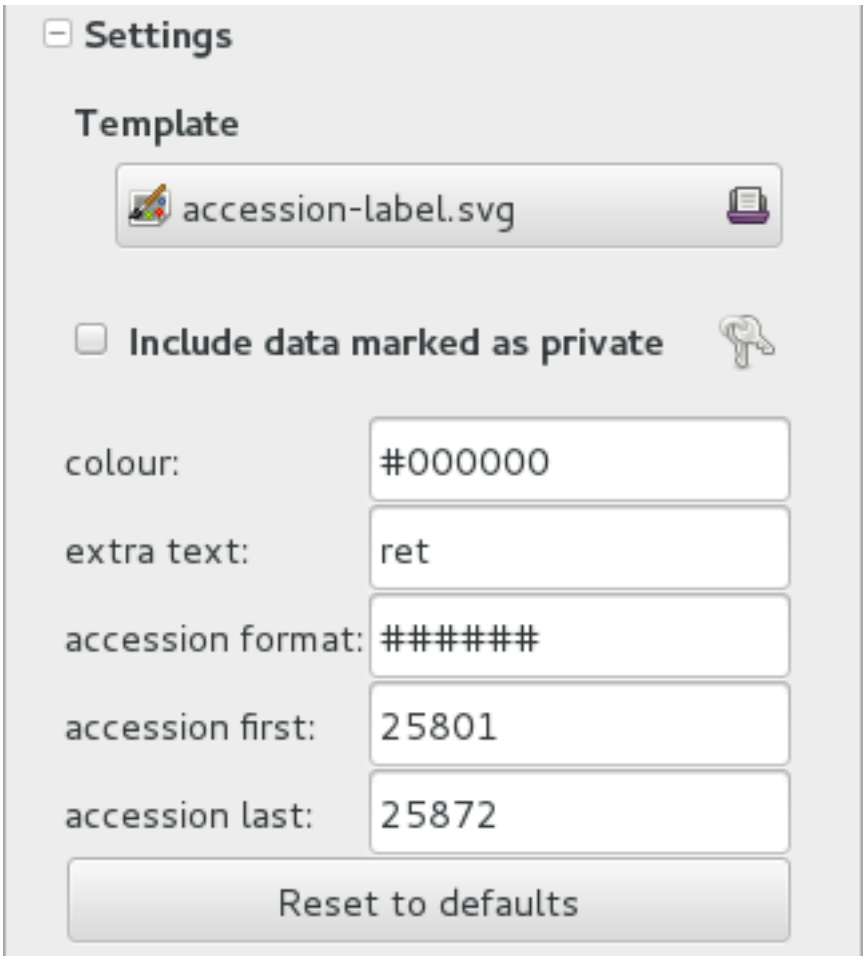
En el primer caso es fácil seleccionar todas las plantas en el lugar, simplemente escriba el nombre de ubicación, o utilice la búsqueda `location like <location name>`.

El segundo caso es un poco más delicado. Lo que hacemos es crear un **Tag** temporal que usamos para juntar todas las plantas que se encuentren en necesidad de una nueva etiqueta.

Dada la selección, inicie la herramienta de informe, utilizando la plantilla mako "accession-label.svg"". Restaure los valores por defecto, y ya que las etiquetas son para imprimir y no grabar, utilizamos el color `black` en lugar que `blue`.

Preparar las etiquetas de plantas que no están en la base de datos

Para preparar un lote de 72 etiquetas, utilizamos una plantilla de mako, denominada `accession-label.svg`. Esta plantilla acepta parámetros, este es un ejemplo que produce las etiquetas de 025801 hasta 025872.



Consideramos las etiquetas de dos tipos: (1) plantas que el jardín acaba de adquirir; (2) o plantas en el jardín, sin una etiqueta. Distinguimos los dos casos añadiendo el texto extra "ret" para las plantas re-etiquetadas.

Tenemos dos cajas con etiquetas de los dos tipos, listas para ser utilizadas.

- Nuestro jardín cuenta con dos invernaderos de exposición y varios invernaderos de frío y de calor donde guardamos la mayor parte de nuestra colección. Movemos las plantas

al área de exposición al florecimiento y al almacén cuando se vuelven menos llamativas para la exposición. Para cada planta en nuestra colección tenemos que saber su ubicación actual y la historia de los movimientos.

Acción planificada

Esta acción empieza al mover físicamente las plantas en el jardín, anotando el código de la planta o en papel, o en nuestra aplicación móvil, si tuviéramos una.

A continuación, vamos a la terminal de escritorio y revisamos todas las plantas una por una, poniendo al día su ubicación en la base de datos. Es importante que la fecha del cambio de ubicación esté memorizada correctamente, porque esto nos dice cuánto tiempo una planta se queda en la exposición.

Teniendo esa aplicación móvil, sólo sería cargar la información en el servidor y ya estaríamos.

Corrección ex-post

Al revisar el jardín, encontramos una planta en un lugar que es no lo que dice la base de datos. Actualizamos la información de la base de datos.

Por ejemplo, la planta perteneciente a la accesión «012142», especies «*Acineta* sp», fue encontrado en «Invernadero 1», mientras que la base de datos dice que está en «ICAlm3».

Todo lo que hacemos es encontrar la planta en la base de datos y actualizar su información. No cambiamos nada en la accesión, el cambio sólo es relativo a la planta.

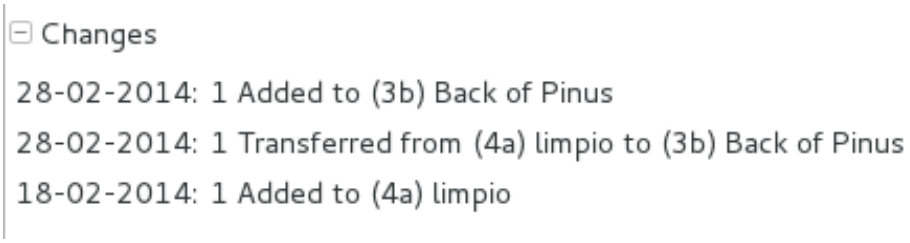
Escriba el código de accesión en el campo de búsqueda, entre comillas, y presione enter. Los resultados de la búsqueda ahora muestran la accesión, que nos informa cuantas plantas le pertenecen. Haga clic en el **+**, así que aparezcan en los resultados también las plantas pertenecentes a la accesión.

Clic derecho sobre la fila de la planta, aparecen las tres opciones: «Editar, dividir, borrar», seleccione Editar, se abre el Editor de planta.

Sólo corregir el campo Ubicación y haga clic en Aceptar.

El InfoBox contiene información sobre el último cambio en el objeto:

Para las plantas, aún más interesante, se construye una historia de los cambios, una lista que incluye cambios de ubicación, o cambios de cantidad.



- Como las plantas entrarán a la etapa de floración, podemos revisar su identificación directamente, o podemos tomar fotografías de los detalles de la flor, con la esperanza que más luego un especialista visitante pueda completar la identificación.

Fotos

Estamos practicando con ODK Collect, un pequeño programa que se ejecuta en los dispositivos android. Todavía no identificamos una mejor práctica en el uso de ODK Collect. Eche un vistazo a la issue correspondiente en github.

- Regularmente, es necesario elaborar informes acerca de nuestra colección, informes que el Ministerio ecuatoriano de medio ambiente (MAE) exigen y que justifican la existencia misma del jardín.

Producir informes

Cada año el jardín botánico tiene que presentar un informe (informe anual de gestión) y el mantenimiento de la colección de orquídeas que cumpla con los requisitos del Ministerio ecuatoriano de medio ambiente.

Para ello, partimos de seleccionar las plantas que tenemos que incluir en el informe. Podría ser toda accuisición en el año pasado:

```
accession where _created between |datetime|2017,1,1|␣
↪and |datetime|2018,1,1|
```

O todas las plantas dentro de una ubicación, o todas planta que pertenecen a una especie, o simplemente todo (pero esto tomará tiempo):

```
plant where location = 'abc'
plant where accession.species.epithet='muricata' and
 ↪accession.species.genus.epithet='Annona'
plant like %
```

Habiendo seleccionado los objetos de base de datos que queremos en el informe, inicie la herramienta de informe, que actúa en la selección.

Buscar en la base de datos

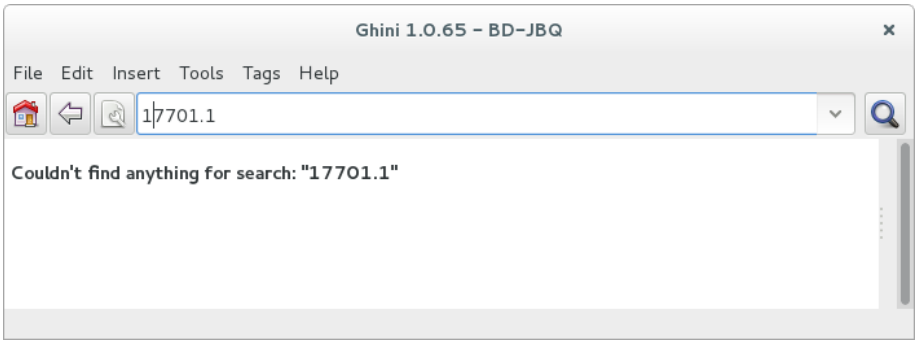
Puedes buscar en la base de datos con el fin de editar los datos, o porque se quiere producir un informe. De todos modos se empieza escribiendo algo en el campo de búsqueda

y ojalá sus resultados aparezcan en el resultado de búsqueda.

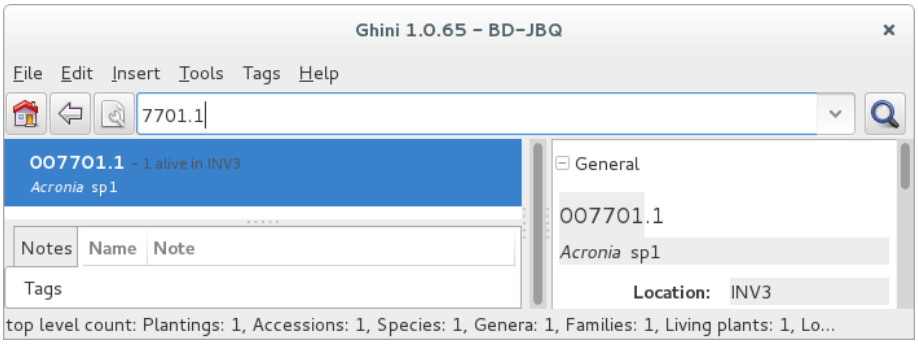
Búsqueda para editar (planta o accesión)

Al buscar con el fin de editar, usted quiere ser muy específico, y seleccionar el número menor posible de objetos. La búsqueda más afinada es la basada en el número de la planta: se conoce el código, se obtiene un objeto.

Si la planta no está allí, la pantalla tendrá este aspecto:

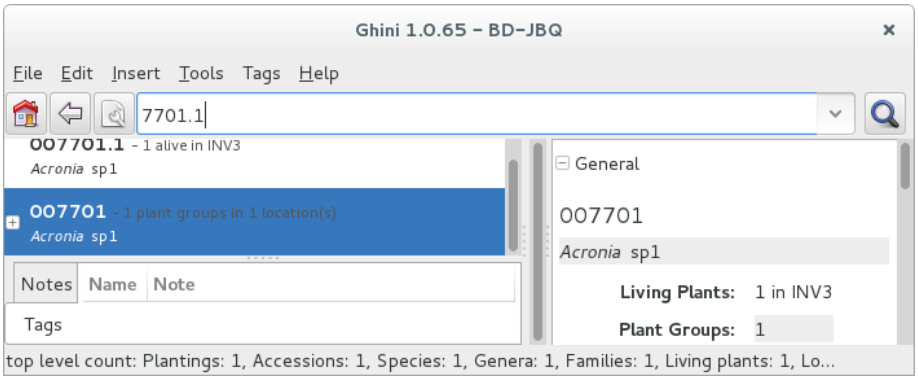


Otro ejemplo, la planta `007701.1` está en la base de datos:



Todos los campos con un fondo más oscuro en el infobox en la derecha son hipervínculos a otros objetos en la base de datos. Haciendo clic sobre uno de ellos tiene como efecto, ó el reemplazar el texto en el campo de búsqueda y ejecutar la nueva consulta ó simplemente agregar el objeto a los resultados.

Hacer clic en la accesión hace el último.



Disponemos ahora de planta y accesión en la vista del resultado de búsqueda y podremos editar ambas.

Búsqueda para informes

Al buscar con el fin de crear un informe, se desea que la búsqueda sea específica (no quiere Informar sobre objetos irrelevantes) y amplia (que no quere informar sobre un solo objeto).

A veces el informe mismo sugiere la consulta correspondiente, como por ejemplo: todas las plantas en invernadero 3. o: todas las plantas pertenecientes a especies en peligro (almacenamos esta información en una nota asociada a la especie); o: todas las plantas que agregamos a la colección durante este año;

```
plant where location.code = INV3
plant where accession.species.notes.note="endangered
↪"
plant where accession._created > |datetime|2017,1,1|
```

De lo contrario una manera flexible para lograr esto es trabajar con **etiquetas**.

El uso de Tags como búsqueda mejorada

A veces queremos ejecutar la misma acción sobre objetos del mismo tipo, pero no podemos pensar rápidamente en una consulta que agrupe todo lo que necesitamos y deje por fuera lo que no necesitamos.

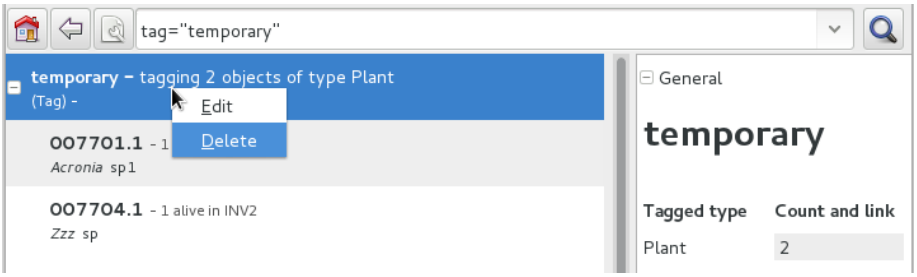
Se trata de un posible uso de los **Tag**. Comenzamos con una selección,

y ponemos en un Tag temporal todos los objetos en la selección. Supongamos llamarlo «temporal».

Seguimos buscando y añadiendo objetos a la etiqueta temporal hasta que la etiqueta identifique todo lo que necesitamos.

Finalmente en el menú de etiquetas seleccionamos el que acabamos de crear (en nuestro ejemplo corresponde a la búsqueda `tag = "temporal"`) y podemos invocar el informe.

Terminado el trabajo con una etiqueta temporal, no hace falta quedarse con ella, así que simplemente la borramos.



Conozca las estrategias de búsqueda

Esto está muy bien documentado, «più non dimandare» y lea la documentación.

4.1.2 using ghini for a seed database

We keep getting involved in groups focusing on endagered plant seeds. They want to note down when seeds come in, but also when they go out to people that order the seed.

In ghini, we keep speaking of ›Plants‹, ›Locations‹, while such user groups focus on ›Seeds‹ and ›Jars‹ and ›Drawers‹ and ›Boxes‹ and ›Envelopes‹. So people wonder whether ghini could be adapted to their use case, or for directions on how to develop their own database.

Does ghini need being adapted for such a seed database?

no it doesn't need any adaptation, it's just that you need to read some of its terms differently.

the taxonomy part is just taxonomy, plant species information, no need to explain that, no way to interpret it otherwise.

›Accessions‹ and ›Plants‹, you know what an ›Accession‹ is, but since you're consistently handling ›Plants‹ still only in seed form, the Wikipedia explanation of an accession sounds like this: it is a seed or group of seeds that are of the same taxon, are of the same propagule type (or treatment), were received from the same source, were received at the same time.

If you hold seeds in jars, or in other sort of containers that is able to hold hundreds of seeds, please make sure that a jar contains seeds of just one accession, as above described: same taxon, same treatment, same source, same time.

Each one of your ›Jars‹ of seeds is in ghini speak a ›Plant‹, and the amount of seeds in the ›Jar‹ is the ›Plant‹ ›quantity‹. An ›Envelope‹ is just the same as a ›Jar‹: a container of seeds from the same ›Accession‹, just presumably smaller.

A ›Box‹ (where you keep several ›Envelopes‹) or a ›Drawer‹ (where you keep several ›Jars‹) are in ghini speak a ›Location‹.

Since a ›Jar‹ or an ›Envelope‹ contains seeds from an ›Accession‹, you will clearly label it with its ›Accession‹ code (and trailing ›Plant‹ number). You might write the amount of seeds, too, but this would be repeating information from the database, and repeating information introduces an inconsistency risk factor.

How do I handle receiving a batch of seeds?

Nota: When we receive seeds, we either collect them ourselves, or we receive it from an other seed collector. We handle receiving them possibly on the spot, or with a small delay. Even when handled together with several other batches of seeds we received, each batch keeps its individuality.

We want to be later able to find back, for example, how many seeds we still have from a specific batch, or when we last received seeds from a specific source.

As long as you put this information in the database, as long as you follow the same convention when doing so, you will be able to write and execute such queries using ghini.

One possibility, the one described here, is based on ›Notes‹. (Ghini does not, as yet, implement the concept «Acquisition». There is an issue related to the Acquisition and Donation objects, but we haven't quite formalized things yet.)

You surely already use codes to identify a batch of seeds entering the seed bank. Just copy this code in a ›Note‹, category "received", to each ›Accession‹ in the received batch. This will let you select the ›Accessions‹ by the query:

```
accession where notes[category='received'].note='<your code>
↪'
```

Use the "Source" tab if you think so, it offers space for indicating an external source, or an expedition. When receiving from an external source, you can specify the code internal to their organization. This will be useful when requesting an extra batch.

How do I handle sending seeds?

what you physically do is to grab the desired amount of seeds of the indicated species from a jar, put it in an envelope and send it. what you do from a point of view of the database is exactly the same, but precisely described in a protocol:

- Use the database to identify the ›Jar‹ containing the desired amount of the right seeds.
- remove that amount of seeds from the ›Jar‹ (decrement the quantity),
- put the seeds in an ›Envelope‹ (yes, that's a database object).
- send the envelope (but keep it in the database).

this in short.

When I send seeds, it's not just one bag, how does ghini help me keeping things together?

There's two levels of keeping things together: one is while you're preparing the sending, and then for later reference.

While preparing the sending, we advise you use a temporary ›Tag‹ on the objects being edited.

For later reference, you will have common ›Note‹ texts, to identify received and sent batches.

Can you give a complete example?

Right. Quite fair. Let's see...

Say you were requested to deliver 50 seeds of Parnassia palustris, 30 of Gentiana pneumonanthe, 80 of Fritillaria meleagris, and 30 of Hypericum pulchrum.

step 1

The first step is to check the quantities you have in house, and if you do have enough, where you have them. You do this per requested species:

```
accession where species.genus.epithet=Parnassia and species.
↪epithet=palustris and sum(plants.quantity)>0
```

Expand in the results pane the ›Accession‹ from which you want to grab the seeds, so you see the corresponding ›Jars‹, highlight one, and tag it with a new ›Tag‹. To do this the first time, go through the steps, just once, of creating a new ›Tag‹. The new tag becomes the active tag, and subsequent tagging will be speedier. I would call the tag ›sending‹, but that's only for ease of exposition and further completely irrelevant.

Repeat the task for Gentiana pneumonanthe, Fritillaria meleagris, Hypericum pulchrum:

```
accession where species.genus.epithet=Gentiana and species.
↪epithet=pneumonanthe and sum(plants.quantity)>0
accession where species.genus.epithet=Fritillaria and␣
↪species.epithet=meleagris and sum(plants.quantity)>0
accession where species.genus.epithet=Hypericum and species.
↪epithet=pulchrum and sum(plants.quantity)>0
```

Again highilight the accession from which you can grab seeds, and hit Ctrl-Y (this tags the highighted row with the active tag). Don't worry if nothing seems to happen when you hit Ctrl-Y, this is a silent operation.

step 2

Now we prepare to go to the seeds bank, with the envelopes we want to fill.

Select the ›sending‹ ›Tag‹ from the tags menu, this will bring back in the results pane all the tagged ›Plants‹ (›Jars‹ or ›Envelopes‹), and will tell you in which ›Location‹ (›Drawer‹ or ›Box‹) they are to be found. Write this information on each of your physical envelopes. Write also the ›Species‹ name, and the quantity you can provide.

Walk now to your seeds bank and, for each of the envelopes you just prepared, open the ›Location‹, grab the ›Plant‹, extract the correct amount of seeds, put them in your physical envelope.

And back to the database!

step 3

If nobody used your workstation, you still have the Tag in the results pane, and it's expanded so you see all the individual plants you tagged.

One by one, you have to ›split‹ the plant. This is a standard operation that you activate by right-clicking on the plant.

A plant editor window comes in view, in "split mode".

Splitting a plant lets you create a database image of the plant group you just physically created, eg: it lets you subtract 30 items from the Gentiana pneumonanthe plant (group number one, that is the one in the jar), and create a new plant group for the same accession. A good practice would be to specify as ›Location‹ for this new plant the "out box", that is, the envelope is on its way to leave the garden.

Don't forget to delete the temporary "sending" ›Tag‹.

step 4

Final step, it represents the physical step of sending the envelope, possibly together with several other envelopes, in a single sending, which should have a code.

Just as you did when you received a batch of plants, you work with notes, this time the category is "sent", and the note text is whatever you normally do to identify a sending. So suppose you're doing a second sending to Pino in 2018, you add the note to each of the newly created envelopes: category "sent", text: "2018-pino-002".

When you finally do send the envelopes, these stop being part of your collection. You still want to know that they have existed, but you do not want to count them among the seeds that are available to you.

Bring back all the plants in the sending "2018-pino-002":

```
plant where notes[category='sent'].note = '2018-pino-002'
```

You now need to edit them one by one, mark the ›quantity‹ to zero, and optionally specify the reason of the change, which would be ›given away‹, and the recipient is already specified in the "sent" ›Note‹.

This last operation could be automated, we're thinking of it, it would become a script, acting on a selection. Stay tuned.

CAPÍTULO 5

Administración

5.1 Administración de bases de datos

Si utiliza un verdadero DBMS para mantener sus datos botánicos, necesitará hacer algo de administración de base de datos. Aunque la administración de base de datos está fuera del alcance de este documento, por lo menos alertamos a nuestros usuarios de esta necesidad.

5.1.1 SQLite

SQLite no es lo que se podría considerar un DBMS real: cada base de datos SQLite está en un archivo. Haga copias de seguridad y todo estará bien. Si no sabe dónde buscar los archivos de base de datos, considere que, por defecto, bauble coloca sus datos en el directorio `~/.bauble/`.

En Windows está en algún lugar en el directorio `AppData`, probablemente en `AppData\Roaming\Bauble`. Tenga en cuenta que Windows hace todo lo posible para ocultar la estructura de directorios `AppData` a los usuarios normales.

La forma más rápida de abrir es con el explorador de archivos: escriba `%APPDATA%` y pulse entrar.

5.1.2 MySQL

Haga referencia a la documentación oficial.

Encontrará documentación detallada sobre cómo realizar copias de seguridad y cómo restaurar bases de datos a partir de ‘esta página <https://mariadb.com/kb/en/the-mariadb-library/backing-up-and-restoring -base de datos /> ‘_.

5.1.3 PostgreSQL

Por favor, consulte la documentación oficial. Una discusión muy completa de las opciones de copia de seguridad inicia al capítulo 24.

5.2 Configuración de Ghini

Ghini utiliza un archivo de configuración para almacenar valores entre invocaciones. Este archivo está asociado a una cuenta de usuario y cada usuario tendrá su propio archivo de configuración.

Para revisar el contenido del archivo de configuración de Ghini, escriba `:prefs` en el área de entrada de texto donde normalmente se escribe las búsquedas, luego presione entrar.

Normalmente no es necesario ajustar el archivo de configuración, pero puede hacerlo con un programa de editor de texto normal. El Archivo de configuración Ghini está en la ubicación predeterminada para bases de datos SQLite.

5.3 Reportar errores

Si nota algo inesperado en el comportamiento de Ghini, que no le parezca adecuado, por favor abra un *issue* en el sitio de desarrollo de Ghini.

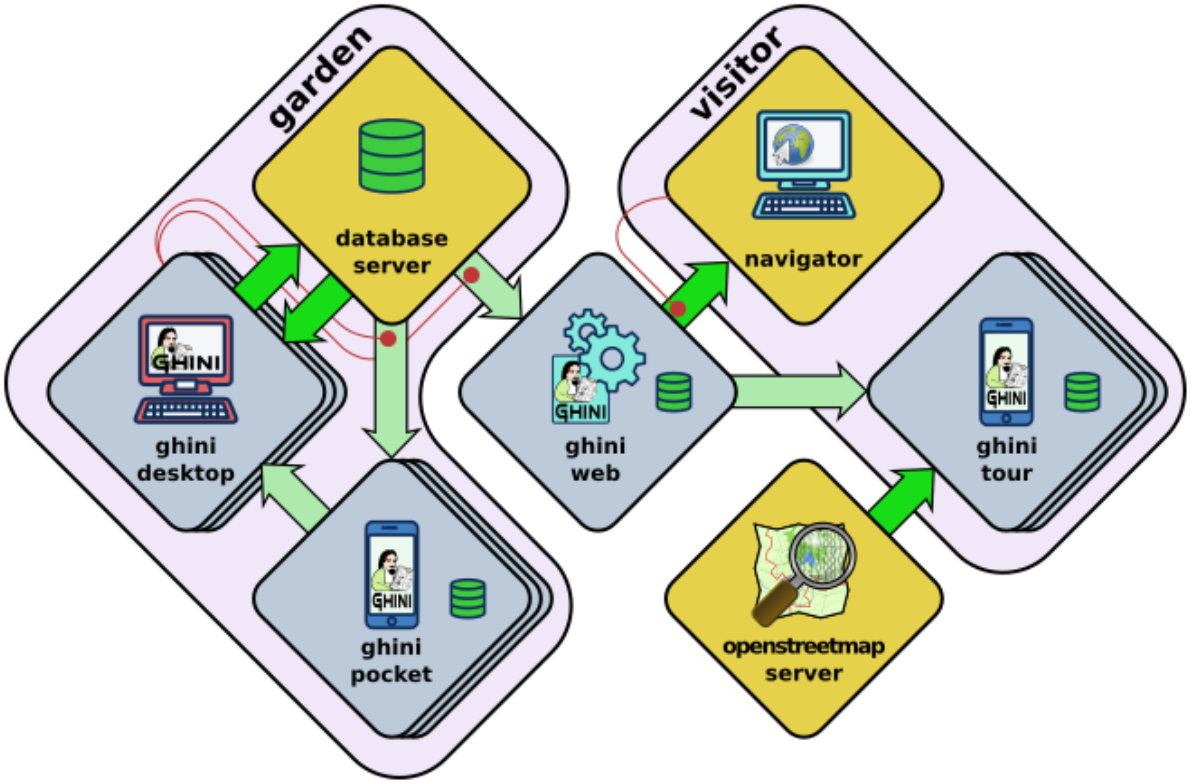
El sitio de desarrollo de Ghini es accesible desde el menú de Ayuda.

CAPÍTULO 6

La familia Ghini

6.1 the Ghini family

Let's start by recalling the composition of the Ghini family, as shown in the diagram:



You have learned how to use ghini.desktop, here we introduce the other members of the family, and their interaction.

6.1.1 ghini.pocket

ghini.pocket is an Android app which you can install from the play store. ghini.pocket is definitely the tool you will use most, next to ghini.desktop.

With ghini.pocket you always have the latest snapshot of your database with you.

Type an accession number, or scan its barcode or QR label, and you know:

- the identification of the plant,
- whether it already has pictures,
- when it entered the garden and
- from which source.

Apart as a quick data viewer, you can use ghini.pocket for...

data correction

If by your judgement, some of the information is incorrect, or if the plant is flowering and you want to immediately take a picture and store it in the database, you do not need take notes on paper, nor follow convolute procedures: ghini.pocket lets you write your corrections in a log file, take pictures associated to the plant, and you will import this information straight into the database, with further minimal user intervention.

inventory review

The initial idea on which we based ghini.pocket is still one of its functionalities: inventory review.

Using ghini.pocket, reviewing the inventory of a greenhouse, in particular if you have QR codes on plant labels, goes as fast as you can walk: simply enter the location code of your greenhouse, reset the log, then one by one scan the plant codes of the plants in the greenhouse. No further data collection action is required.

When you're done, import the log in ghini.desktop. The procedure available in ghini.desktop includes adding unknown but labelled plants in the database, marking as lost/dead all plants that the database reports as alive and present in the inventoried location, but were not found during the inventory.

taxonomic support

> As a bonus, ghini.pocket contains a phonetic genus search, and a quite complete database of botanic taxa with rank between order and genus, including tribes, and synonymies.

check further *data streams between software components*.

6.1.2 ghini.web

ghini.web is a web server, written in nodejs.

Its most visible part runs at http://gardens.ghini.me and shows as a map of the world, where you browse gardens and search their published collection.

It also serves configuration data to ghini.tour instances.

check further *data streams between software components*.

6.1.3 ghini.tour

ghini.tour is an Android app which you can install from the play store.

People visiting your garden will install ghini.tour on their phone or tablet, enjoy having a map of the garden, knowing where they are, and will be able to listen to audio files that you have placed as virtual information panels in strategic spots in your garden.

world view

> at startup, you see the world and gardens. select a garden, and enter.

garden view

> when viewing at garden level, you see panels. select a panel, and listen.

check further *data streams between software components*.

6.1.4 data streams between software components

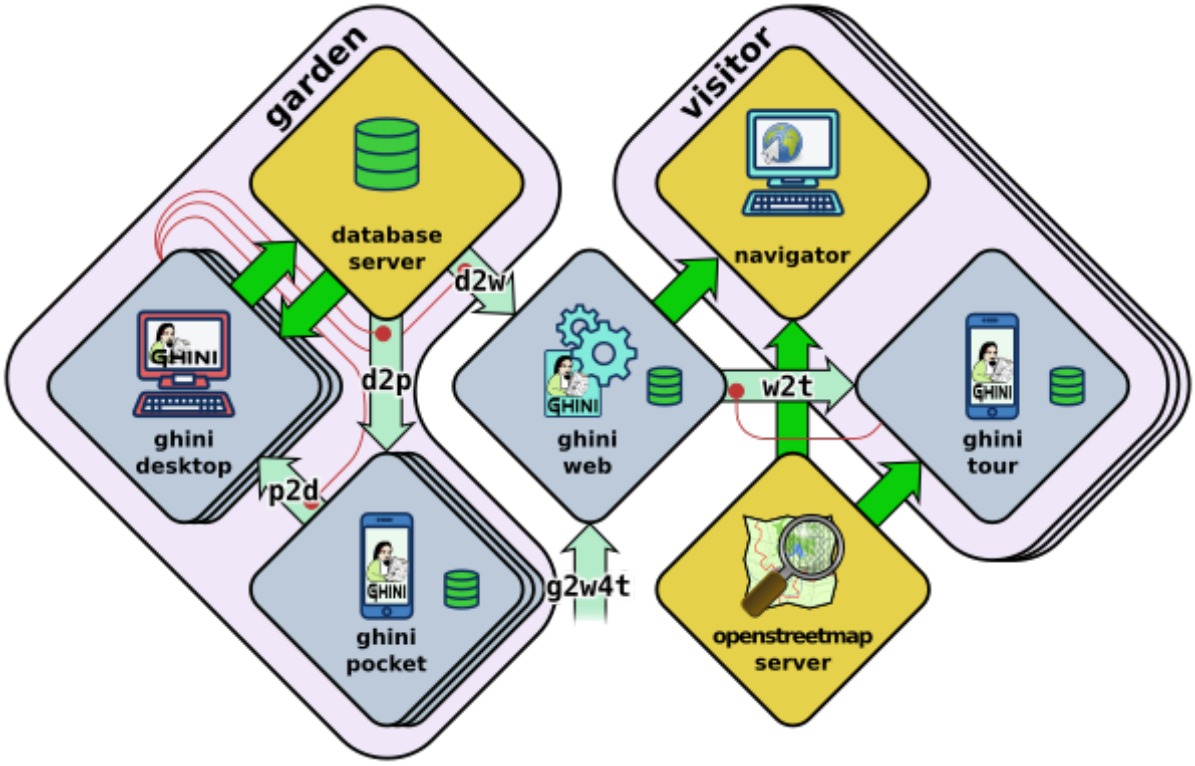
Nota: This section contains technical information for database managers and software developers.

In the diagram showing the composition of the Ghini family, the alert reader noticed how different arrows representing different data flows, had different colours: some are deep green, some have a lighter tint.

Deeper green streams are constant flows of data, representing the core activity of a component, eg: the interaction between ghini.desktop and its database server, or your internet browser and ghini.web.

Lighter green streams are import/export actions, initiated by the user at the command panel of ghini.desktop, or in the ghini.tour settings page.

This is the same graph, in which all import data streams have been given an identifier.



d2p: copy a snapshot of the desktop database to ghini.pocket

- export the desktop database to a pocket snapshot
- copy the snapshot to the handheld device

ghini.pocket integrates closely with ghini.desktop, and it's not a tool for the casual nor the external user. One task of your garden database manager is to regularly copy an updated database snapshot to your Android device.

We advise enabling USB debugging on the device. In perspective, this will allow ghini.desktop writing directily into the ghini.pocket device.

Export the file from ghini.desktop, call the file pocket.db, copy it to the phone:

```
adb -d push /tmp/pocket.db /sdcard/Android/data/me.ghini.
↪pocket/files/
```

The above location is valid even if your phone does not have a memory card.

Other options include bluetooth, or whatever other way you normally use to copy regular files into your Android device.

---

p2d: import from the ghini.pocket log file and pictures into the central database

even if we're still calling it "inventory log", ghini.pocket's log contains more than just inventory corrections.

- produce a log on the handheld device
- import the log in the desktop database

first of all, copy the collected information from ghini.pocket into your computer:

```
export DIR=/some/directory/on/your/computer
adb -d pull /sdcard/Android/data/me.ghini.pocket/files/
↪searches.txt $DIR
adb -d pull -a /sdcard/Android/data/me.ghini.pocket/files/
↪Pictures $DIR
```

then use ghini.desktop to import this information into your database.

---

d2w: send a selection of your garden data to ghini.web

Offer a selection of your garden data to a central ghini.web site, so online virtual visitors can browse it. This includes plant identification and their geographic location.

content of this flow:

- garden: coords, name, zoom level (for initial view)
- plants: coords, identification, zoom level (for visibility)
- species: binomial, phonetic approximation

---

g2w: add geographic non-botanic data to ghini.web

- Write geographic information about non-botanic data (ie: point of interest within the garden, required by ghini.tour) in the central ghini.web site.

  content of this flow:

  - virtual panels: coords, title, audio file
  - photos: coords, title, picture

  virtual panels don't necessarily have an associated photo, photos don't necessarily have an associated audio file.

---

w2t: importing locations and POIs from ghini.web to tour

content of this flow:

- Garden (coords, name, zoom level)
- Points of Interest (coords, title, audio file, photo)

CAPÍTULO 7

Desarrollo de Ghini

7.1 Manual del desarrollador

Si usó las instrucciones de instalación de `devinstall`, ha descargado las fuentes conectadas al repositorio de github. Se encuentra en la situación ideal para comenzar a investigar el software, comprender cómo funciona y contribuir al desarrollo de ghini.desktop.

7.1.1 Ayuda al desarrollo de Ghini

Si desea contribuir al Ghini, puede hacerlo de varias diferentes maneras:

- Utilice el software, anote lo que menos le gusta, abra una issue para cada punto. Un desarrollador se activará antes que usted pueda imaginarse.
- Si opina que algo le hace falta al software pero no cree poderlo formalizar en cuestiones separadas, podría considerar contratar a un profesional. Se trata de la mejor manera de asegurarse de que algo ocurra rápido en Ghini. Asegúrese de que el desarrollador abra issues y publique sus contribuciones en github.
- ¡Traducir! Cada ayuda con las traducciones será bienvenida, así que por favor ¡traduzca! puede hacer esto sin necesidad de instalar nada en tu ordenador, simplemente usando el servicio de traducción en línea ofrecido por weblate
- haga un fork del repositorio, eliga un problema, lo resuelva, abra una pull request. Ver el *bug solving workflow* a continuación.

Si todavía no has instalado Ghini y quiere echar un vistazo a la historia de su código y documentación, puede abrir nuestra página de github y ver todo lo que ha estado sucediendo con Ghini, desde sus inicios en el año 2004 bajo el nombre de Bauble.

Si instaló el software según las instrucciones `devinstall`, ya tiene todo el historial completo en el clone git local.

7.1.2 Fuente del software, versiones, ramas

Si desea una versión particular de Ghini, nosotros las ponemos a disposición como ramas (branch). Usted debe `git checkout` de la rama correspondiente a la versión que desea.

línea de producción

Nombres de rama para las versione esbables (producción) de Ghini son de la forma `ghini-x.y` (por ejemplo: ghini-1.0); nombres de rama para el desarrollo de Ghini son de la forma `ghini-x.y-dev` (por ejemplo: ghini-1.0-dev).

7.1.3 Flujo de trabajo de desarrollo

Nuestro flujo de trabajo incluye la entrega rápida y continua a la rama en prueba, empujando frecuentemente a github, para que travis-ci y coveralls.io comprueben la calidad de las ramas, por último, de vez en cuando, fusionando la rama en prueba en la versión correspondiente.

Cuando trabajo en cuestiones mayores, que pienso van a durar más que un par de días, podría preferir abrir una rama asociada a la cuestión. No hago esto muy a menudo.

cuestiones mayores

Frente a un tema mayor bien identificado, crear una rama etiquetada en el final de una línea de desarrollo principal (por ejemplo: `ghini-1.0-dev`) y seguir el flujo de trabajo descrito en

https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging

en breve:

```
git up
git checkout -b issue-xxxx
git push origin issue-xxxx
```

Trabajar en la nueva rama temporal. Cuando esté listo, vaya a github, fusione la rama con la línea principal de desarrollo de que ramificó, resuelva los conflictos si necesario, elimine la rama temporal.

Cuando esté listo para su publicación, combinar la línea de desarrollo en la línea de producción correspondiente.

7.1.4 Actualizar el conjunto de cadenas traducibles

De vez en cuando, durante el proceso de actualización del software, el programador añade o modifica cadenas de texto en las fuentes python, o en la documentación, o en las fuentes glade.

La mayoría de nuestras cadenas son traducibles y son automáticamente ofrecidas a weblate para la contribución de traducciones por voluntarios, en forma de archivos `po`.

Un archivo `po` está asociado a un idioma y contiene texto en pares: original y traducción. Cuando un traductor añade una traducción en weblate, eso es copiado a nuestro repositorio en github. Cuando un programador añade un texto en el software, esto es copiado a weblate para ser traducido.

Weblate alberga el proyecto Ghini. El proyecto está organizado en componentes, cada cual corresponde a una rama de uno de nuestros repositorios en github. Cada componente acepta traducciones a varios idiomas.

componente	repositorio	rama
Desktop 1.0	ghini.desktop	ghini-1.0-dev
Desktop 3.1	ghini.desktop	ghini-3.1-dev
Documentation 1.0	ghini.desktop-docs.i18n	ghini-1.0-dev
Documentación de Ghini 1.0	ghini.desktop-docs.i18n	ghini-3.1-dev
Web 1.2	ghini.web	master
Pocket	ghini.pocket	master
Tour	ghini.tour	master

Para poner al día los archivos `po` relativos al *software*, active una linea de comando en la carpeta base del checkout de *ghini.desktop* y ejecute el script:

```
./scripts/i18n.sh
```

Para poner al día los archivos `po` relativos a la *documentación*, active una linea de comando en la carpeta base del checkout de *ghini.desktop-docs.i18n* y ejecute el script:

```
./doc/runme.sh
```

Cuando se ejecuta uno de los scripts mencionados, lo más probable es que usted necesite alterar *todos* archivos `po` del proyecto. Puedes querer revisar los cambios antes de enviarlos el repositorio. Esto es muy importante cuando se realiza una corrección marginal en una cadena, como quitar un error de tipeo.

Algo que pasa: acabar en un conflicto. Resolución de conflictos no es difícil una vez que uno sepa cómo se hace. En primer lugar, añadir weblate como repositorio remoto:

```
git remote add weblate-doc10 https://hosted.weblate.org/git/ghini/
↪documentation-10/
```

A continuación, asegúrese de estar en el repositorio correcto, en la rama correcta, actualizar la fuente remota, fusionar con ella:

```
git checkout ghini-1.0-dev
git remote update
git merge weblate-doc10/ghini-1.0-dev
```

“Nuestra documentación <https://readthedocs.org/projects/ghini/ >‘_ en readthedocs tiene una versión original en inglés y varias traducciones. Seguimos la **‘descripción de localización**

<http://docs.readthedocs.io/en/latest/localization.html >‘_, no hay nada que nosotros mismos inventamos aquí.

Readthedocs comprueba la configuración de idioma del proyecto e invoca `sphinx-intl` para formatear la documentación en el idioma de objetivo. Con la configuración predeterminada, que no alteramos, `sphinx-intl` espera un archivo `po` por cada documento fuente, llamado como el documento de origen, y que resida en el directorio `local/$(LANG)/LC_MESSAGES/`.

Por otro lado, Weblate (y nosotros) preferimos un único fichero `po` por lenguaje, y los mantenemos todos en el mismo fichero `/po`, tal como lo hacemos para el proyecto de software: `/po/$(LANG).po`.

Con el fin de no repetir la información, y para permitir que ambos sistemas trabajen en su forma natural, tenemos dos conjuntos de enlaces simbólicos (git los honra).

Para resumir: cuando se actualiza un archivo de la documentación, el script `runme.sh` se ocupa de:

1. copia los archivos `rst` del software a la documentación;
2. crea un nuevo archivo `pot` para cada documento;
3. junta los archivos `pot` en un único `doc.pot`;
4. pone al día todos `doc.po` (uno por idioma) en base al nuevo `doc.pot`;
5. crea todos enlaces simbólicos:
    a. para `sphinx-int` en `/local/$(LANG)/LC_MESSAGES/`
    b. para weblate en `/po/$(LANG).po`

Definitivamente podríamos escribir lo anterior en un Makefile, o incluso mejor incluirlo en `doc/Makefile`. Quién sabe, tal vez lo haremos.

7.1.5 Producing the docs locally

The above description is about how we help external sites produce our documentation so that it is online for all to see. But what if you want to have the documentation locally, for example if you want to edit and review before pushing your commits to the cloud?

In order to run sphinx locally, you need to install it **within** the same virtual environment as ghini, and to install it there, you need to have a sphinx version whose dependencies don not conflict with ghini.desktop's dependecies.

What we do to keep this in order?

We state this extra dependency in the `setup.py` file, as an `extras_require` entry. Create and activate the virtual environment, then run `easy_install ghini.desktop[docs]`. This gets you the sphinx version as declared in the `setup.py` file.

If all you want is the html documentation built locally, run `./setup.py install docs`. For more options, enter the `doc` directory and run `make`.

7.1.6 Como hacen nuestras cadenas de texto para llegar a los usuarios?

Hace poco un traductor me puso esta pregunta, añadiendo: »se trata de un proceso automático de Weblate a Git al Ghini desktop ya instalado, o requiere intervención manual?

Es que la interacción es algo más compleja, y depende de cuál componente estamos considerando.

When you install `ghini.desktop` or one of the Android apps, the installation doesn't assume a specific run-time language: a user can change their language configuration any time. So what we do is to install the software in English together with a translation table from English to whatever else.

At run-time the GUI libraries (Android or GTK) know where to look for the translation strings. These translation tables are generated during the installation or upgrade process, based on the strings you see on Weblate.

The path followed by translations is: You edit strings on Weblate, Weblate keeps accumulating them until you are done, or you don't interact with Weblate for a longer while; Weblate pushes the strings to github, directly into the development line `ghini-1.0-dev`; I see them and I might blindly trust or prefer to review them, maybe I look them up in wikipedia or get them translated back to Italian, Spanish or English by some automatic translation service; sometimes I need to solve conflicts arising because of changed context, not too often fortunately. As said, this lands in the development line `ghini-1.0-dev`, which I regularly publish to the production line `ghini-1.0`, and this is the moment when the new translations finally make it to the distributed software.

Users will notice a *new version available* warning and can decide to ignore it, or to update.

For `ghini.pocket`, it is similar, but the notification is handled by the Android system. We publish on the Play Store, and depending on your settings, your phone will update the software automatically, or only notify you, or do nothing. It depends on how you configured automatic updates.

For `ghini.web`, we haven't yet defined how to distribute it.

Para la documentación de ghini, el proceso es totalmente automático, manejado por readthedocs.org.

7.1.7 Añadir pruebas unitarias

Si te interesa contribuir al desarrollo de Ghini, una buena manera de hacerlo sería por ayudarnos a encontrar y escribir las pruebas unitarias que falta.

Una función de bien probada es uno cuyo comportamiento se puede cambiar sin romper al menos una prueba de unidad.

Todos estamos de acuerdo que en teoría de la teoría y la práctica coinciden perfectamente y que uno escribe las pruebas primero implementa la función. En la práctica, sin embargo, práctica no coincide con la teoría y nos hemos estado escribiendo pruebas después de escribir y publicar incluso las funciones.

Esta sección describe el proceso de agregar pruebas unitarias para `bauble.plugins.plants.family.remove_callback`.

Para probar

En primer lugar, abra el índice del informe de cobertura y eliga un archivo con baja cobertura.

Para este ejemplo, en octubre de 2015, aterrizamos en `bauble.plugins.plants.family`, en el 33 %.

https://coveralls.io/builds/3741152/source?filename=bauble%2Fplugins%2Fplants%2Ffamily.py

Las dos primeras funciones que necesitan pruebas, `edit_callback` y `add_genera_callback`, incluyen la creación y activación de un objeto depende de un cuadro de diálogo personalizado. Debe realmente primeras pruebas de unidad de escritura para esa clase y luego volver aquí.

La siguiente función, `remove_callback`, también activa un par de cuadros de diálogo y mensajes, pero en la forma de invocar una función de solicitud de entrada del usuario mediante cajas de sí-no-ok. Estas funciones podemos reemplazar fácilmente con una función de burlarse de la conducta.

Cómo se hace

Así, después de haber decidido qué describir en prueba de la unidad, nos fijamos en el código y vemos que es necesario discriminar un par de casos:

**** parámetro corrección ****

- la lista de familias no tiene elementos.
- la lista de familias tiene más de un elemento.
- la lista de las familias tiene exactamente un elemento.

cascade

- la familia no tiene géneros
- la familia tiene uno o más géneros

confirmar

- el usuario confirma la eliminación
- el usuario no confirma la eliminación

borrar

- todo va bien al eliminar la familia
- hay un error al eliminar la familia

Decidir sólo se centrará en la ** cascada ** y el ** confirmar ** aspectos. Dos preguntas binarias: 4 casos.

donde poner las pruebas

Localizar el script de prueba y elija la clase donde poner las pruebas de unidad extra.

https://coveralls.io/builds/3741152/source?filename=bauble%2Fplugins%2Fplants%2Ftest.py#L273

---

que se hace con las pruebas ignoradas

La clase de `FamilyTests` contiene una prueba omitida, implementarlo a ser un poco de trabajo porque necesitamos reescribir la FamilyEditorPresenter, la separan de la FamilyEditorView y reconsiderar qué hacer con la clase FamilyEditor, que creo que debe eliminado y reemplazado con una sola función.

---

las pruebas de la escritura

Después de la última prueba en la clase de FamilyTests, añadir los cuatro casos que quiero describir, y aseguro que no, y como soy perezosa, escribo el código más compacto que conozco para generar un error:

```
def test_remove_callback_no_genera_no_confirm(self):
    1/0

def test_remove_callback_no_genera_confirm(self):
    1/0

def test_remove_callback_with_genera_no_confirm(self):
    1/0

def test_remove_callback_with_genera_confirm(self):
    1/0
```

Una prueba, paso a paso

Vamos a empezar con el primer caso de prueba.

Al escribir ensayos, generalmente siguen el patrón:

- T_0 (condición inicial),
- acción,
- T_1 (prueba el resultado de la acción dada las condiciones iniciales)

¿qué hay en un nombre — las pruebas unitarias

Hay una razón por la cual las pruebas unitarias se llaman pruebas unitarias. Por favor, nunca pruebe dos acciones en una prueba.

Así que vamos a describir T_0 de la primera prueba, una base de datos manteniendo una familia sin géneros:

```
def test_remove_callback_no_genera_no_confirm(self):
    f5 = Family(family=u'Arecaceae')
    self.session.add(f5)
    self.session.flush()
```

No queremos que la función está probadaspara para invocar la función interactiva `utils.yes_no_dialog`, queremos `remove_callback` para invocar una función de reemplazo no interactivo. Ello simplemente haciendo punto de `utils.yes_no_dialog` a una expresión de `lambda` que, al igual que la original función interactiva, acepta un parámetro y devuelve un valor booleano. En este caso: `False`:

```
def test_remove_callback_no_genera_no_confirm(self):
    # T_0
    f5 = Family(family=u'Arecaceae')
    self.session.add(f5)
    self.session.flush()

    # action
    utils.yes_no_dialog = lambda x: False
    from bauble.plugins.plants.family import remove_callback
    remove_callback(f5)
```

A continuación probamos el resultado.

Bueno, no sólo queremos prueba si o no el objeto Arecaceae fue suprimida, nosotros también debemos el valor devuelto por `remove_callback`, y si `yes_no_dialog` y `message_details_dialog` se invocaron o no.

Una expresión de `lambda` no es suficiente para esto. Hacemos algo aparentemente más complejo, que hará la vida mucho más fácil.

Primero definamos una función más bien genérica:

```
def mockfunc(msg=None, name=None, caller=None, result=None):
    caller.invoked.append((name, msg))
    return result
```

y `parcial` del `functools` módulo estándar, parcialmente aplicar arriba `mockfunc`, dejando sólo `mensaje` no se especifica, y utilizar esta aplicación parcial, que es una función que acepta un parámetro y devuelve un valor, para reemplazar las dos funciones en `utils`. La función de prueba ahora se ve así:

```
def test_remove_callback_no_genera_no_confirm(self):
    # T_0
    f5 = Family(family=u'Arecaceae')
    self.session.add(f5)
    self.session.flush()
```

(continué en la próxima página)

(proviene de la página anterior)

```
    self.invoked = []

    # action
    utils.yes_no_dialog = partial(
        mockfunc, name='yes_no_dialog', caller=self, result=False)
    utils.message_details_dialog = partial(
        mockfunc, name='message_details_dialog', caller=self)
    from bauble.plugins.plants.family import remove_callback
    result = remove_callback([f5])
    self.session.flush()
```

La sección de prueba comprueba que no se invocó el `message_details_dialog`, que `yes_no_dialog` fue invocado, con el parámetro de mensaje correcto, que existe todavía la Arecaceae:

```
# effect
self.assertFalse('message_details_dialog' in
                 [f for (f, m) in self.invoked])
self.assertTrue(('yes_no_dialog', u'Are you sure you want to '
                 'remove the family <i>Arecaceae</i>?')
                in self.invoked)
self.assertEquals(result, None)
q = self.session.query(Family).filter_by(family=u"Arecaceae")
matching = q.all()
self.assertEquals(matching, [f5])
```

Etcétera

> «hay dos clases de personas, aquellos que terminan lo que empiezan y así sucesivamente»

Próxima prueba es casi lo mismo, con la diferencia que el `utils.yes_no_dialog` debe volver `True` (esto logramos especificando `resultado = True` en la aplicación parcial de la genérico `mockfunc`).

Con esta acción, el valor devuelto por `remove_callback` debe ser `True`, y no debe haber ninguna familia Arecaceae en la base de datos más:

```
def test_remove_callback_no_genera_confirm(self):
    # T_0
    f5 = Family(family=u'Arecaceae')
    self.session.add(f5)
    self.session.flush()
    self.invoked = []

    # action
    utils.yes_no_dialog = partial(
        mockfunc, name='yes_no_dialog', caller=self, result=True)
```

(continué en la próxima página)

(proviene de la página anterior)

```
    utils.message_details_dialog = partial(
        mockfunc, name='message_details_dialog', caller=self)
    from bauble.plugins.plants.family import remove_callback
    result = remove_callback([f5])
    self.session.flush()

    # effect
    self.assertFalse('message_details_dialog' in
                     [f for (f, m) in self.invoked])
    self.assertTrue(('yes_no_dialog', u'Are you sure you want to '
                     'remove the family <i>Arecaceae</i>?')
                    in self.invoked)
    self.assertEquals(result, True)
    q = self.session.query(Family).filter_by(family=u"Arecaceae")
    matching = q.all()
    self.assertEquals(matching, [])
```

Echa un vistazo en 734f5bb9feffc2f4bd22578fcee1802c8682ca83 commit para las otras dos funciones de la prueba.

Pruebas y Registro

Nuestros objetos de `bauble.test.BaubleTestCase` utilizan handler de la clase `bauble.test.MockLoggingHandler`. Cada vez que se inicia una prueba unitaria, el método `setUp` crea un nuevo `handler` y lo asocia al logger root. El método de `teardDown` se encarga de eliminarlo.

Se puede comprobar la presencia de mensajes de registro específicos en `self.handler.messages`. `mensajes` es un diccionario, inicialmente vacío, con dos niveles de indexación. Primero el nombre del registrador añadiendo al registro, luego el nombre del nivel del mensaje de registro. Las claves son producidas al necesitarse. Los valores en el diccionario son listas de mensajes, con formato según el formateador asociado al controlador, por defecto `logging.Formatter("%(message)s")`.

Al necesitarlo, puede borrar los mensajes colectados mediante la invocación de `self.handler.clear()`.

Al ponerlo todo junto

De vez en cuando usted quiere activar la clase de prueba trabaja en:

```
nosetests bauble/plugins/plants/test.py:FamilyTests
```

Y al final del proceso que desea actualizar las estadísticas de:

```
./scripts/update-coverage.sh
```

7.1.8 Estructura de interfaz de usuario

La interfaz de usuario se construye según el ** modelo ** — ** vista - presentador ** patrón arquitectónico. Gran parte de la interfaz, ** modelo ** es un objeto de base de datos de SQLAlchemy, pero también tenemos elementos de la interfaz donde existe un modelo de base de datos correspondiente. En general:

- El ** ver ** se describe como parte de un ** claro ** archivo. Esto debe incluir la señal de llamada y ListStore TreeView asociaciones. A reutilizar la clase base que define “” GenericEditorView”” en “” bauble.editor””. Cuando se crea una instancia de esta clase genérica, pasarlo el ** claro ** nombre del archivo y el nombre del widget de raíz, luego entregar esta instancia a la ** presentador ** constructor.

  En el archivo glade, en la sección de `action-widgets` que cierra la descripción del objeto GtkDialog, asegúrese de que cada elemento `action-widget` tenga un valor `response` válido Use los valores válidos de GtkResponseType, por ejemplo:

  - GTK_RESPONSE_OK, -5
  - GTK_RESPONSE_CANCEL, -6
  - GTK_RESPONSE_YES, -8
  - GTK_RESPONSE_NO, -9

  No hay manera fácil de escribir pruebas unitarias para una subclase de una vista, así que por favor no subclasen vistas, no hay realmente ninguna necesidad.

  En el archivo de glade, cada widget de entrada debe definir qué controlador se activa en que señal. La clase genérica de presentador ofrece llamadas genéricas que cubren los casos más comunes.

  - GtkEntry (entrada de texto de línea singola) encargará de la señal de `changed`, con `on_text_entry_changed` o `on_unique_text_entry_changed`.
  - GtkTextView: asociarlo a un GtkTextBuffer. Para manejar la señal `changed` en la GtkTextBuffer, tenemos que definir un manejador que invoca el genérico `on_textbuffer_changed`, el único papel para esta función consiste en pasar a nuestro controlador genérico el nombre del atributo modelo que recibe el cambio. Este es un workaroud para un bug sin resolver en GTK.
  - GtkComboBox con textos traducidos no se pueden gestionar fácilmente desde el archivo de glade, por lo que aún no tratamos. Utilice el método de “” init_translatable_combo”” de la clase genérica de “” GenericEditorView””, pero por favor invocarlo desde el ** presentador **.

- El **modelo** es sólo un objeto con atributos. En esta interacción, el **modelo** es un contenedor de datos pasivo, no hace nada más sino dejar que el **presentador** los modifique.
- Lo que fue obtenido como subclase de **presentador** define e implementa:

  - `widget_to_field_map`, un diccionario que asocia nombres de widget a nombre de los atributos del modelo,

- `view_accept_buttons`, la lista de nombres de widget que, si activados por el usuario, significa que la vista debe ser cerrada,
- todos los callbacks necesarios,
- Opcionalmente, también juega el papel de **modelo**.

El **presentador** actualiza continuamente el **modelo** según los cambios en la **vista**. Si el **modelo** corresponde a un objeto de base de datos, el **presentador** entrega todas actualizaciones en el **modelo** a la base de datos cuando la **vista** se cierra con éxito, o los deshace si la **vista** se cancela. (este comportamiento es influenciado por el parámetro `do_commit`)

Si el ** modelo ** es otra cosa, entonces el ** presentador ** hará algo más.

Nota: Un buen comportamiento ** presentador ** utiliza la ** vista ** api para consultar los valores insertados por el usuario o por la fuerza a establecer Estados de widget. Por favor, no se aprende de la práctica de nuestros presentadores portando mal, algunos que manejar directamente campos de `view.widgets`. Al hacerlo, estos presentadores nos impide escribir pruebas unitarias.

La clase base para el presentador, `GenericEditorPresenter` en `bauble.editor`, implementa muchas devoluciones de llamada genéricos útiles. Hay una clase de `MockView`, que se puede utilizar al escribir pruebas para sus presentadores.

Ejemplos

`Contact` y `ContactPresenter` se implementan siguiendo las líneas de arriba. La vista se define en el archivo `contact.glade`.

Un buen ejemplo de patrón de vista/presentador (no modelo) está dada por el administrador de conexiones.

Se utiliza el mismo patrón arquitectónico de interacción no-base de datos, estableciendo el presentador también como modelo. Para ello, por ejemplo, el cuadro de diálogo de exportación JSON. El siguiente comando le dará una lista de instancias de `GenericEditorView`:

```
grep -nHr -e GenericEditorView\( bauble
```

7.1.9 Ghini extensible con Plugins

Casi todo en Ghini es extensible mediante plugins. Un Plugin puede crear tablas, definir búsquedas personalizadas, agregar elementos de menú, crear comandos personalizados y mucho más.

Para crear un nuevo plugin debe extender la clase de `bauble.pluginmgr.Plugin`.

La `Tag` es un ejemplo mínimo, incluso si el `TagItemGUI` cae fuera el patrón de arquitectura Model-View-Presenter.

7.1.10 Estructura de plugins

Ghini es un marco para el manejo de colecciones y se distribuye junto con un conjunto de plugins haciendo Ghini un gerente de colección botánica. Pero Ghini mantiene un marco y podría en teoría quitar todos los plugins que distribuir y escribir su propio, o escribir tus propios plugins que ampliar o completar el actual comportamiento de Ghini.

Una vez que ha seleccionado y abre una conexión de base de datos, aterrizas en la ventana de búsqueda. La ventana de búsqueda es una interacción entre dos objetos: SearchPresenter (SP) y vista búsqueda (SV).

SV es lo que ves, SP contiene el estado del programa y encarga de las peticiones que expresan a través de SV. Manejo de estas solicitudes afectan el contenido del SV y la situación del programa en SP.

Resultados de la búsqueda se muestra en la parte más grande de SV son filas, objetos que son instancias de clases registradas en un plugin.

Cada una de estas clases debe implementar una cantidad de funciones para comportarse adecuadamente en el marco de Ghini. El marco de Ghini Reserva espacio para clases de acoplamiento.

SP sabe de todas las clases (enchufadas) registradas, que son almacenados en un diccionario, asociar una clase a su implementación del plugin. SV tiene una ranura (un gtk. De la caja) donde usted puede agregar elementos. En cualquier momento, a lo sumo sólo un elemento en la ranura es visible.

Un plugin define una o más clases de plugin. Una clase de plugin desempeña el papel de presentador parcial (pP - plugin presentador) implementar las devoluciones de llamadas necesitadas el vista parcial asociada en la ranura (pV - plugin view), y el patrón MVP es completado por el presentador de padre (SP), otra vez actuando como modelo. Resumir y completar:

- SP actúa como modelo,
- la vista parcial de pV se define en un archivo de glade.
- las devoluciones de llamadas implementadas por pP se hace referencia en el archivo de glade.
- un menú contextual para la fila de SP,
- una característica de los niños.

Cuando se registra una clase de plugin, el SP:

- agrega el pV en la ranura y hace no visible.
- agrega una instancia del pP en las clases de plugin registrados.
- dice el pP que el SP es el modelo.
- conecta todas las devoluciones de llamada de pV pP.

Cuando un elemento en pV desencadena una acción en el pP, el pP podrá enviar la acción a SP y puede solicitar SP que actualiza el modelo y actualiza la vista.

Cuando el usuario selecciona una fila en SP, SP oculta todo en la ranura de acoplamiento y muestra sólo el pV solo en relación con el tipo de la fila seleccionada y pide al pP para refrescar el pV con lo que es relativo a la fila seleccionada.

Aparte de establecer la visibilidad de la pV varios, nada necesita discapacitado ni quitado: un pV invisible no puede desencadenar eventos!

7.1.11 error solución de flujo de trabajo

flujo de trabajo de desarrollo normal

- si al usar el software, usted nota un problema, o se le occurre de como mejorar algo, pienselo lo suficiente para perfeccionar su idea de lo que realmente es, lo que ha notado. Abra un tema y nos comparta su idea. Alguien podría reaccionar con notas.
- puede visitar el sitio de temas y eligir uno que desea abordar.
- asigne el tema a si mismo, de esta manera informa al mundo que tiene la intención de trabajar en ello. alguien podría reaccionar con notas.
- Opcionalmente el repositorio en su cuenta de la bifurcación y preferiblemente crear una rama, claramente asociada a la cuestión.
- escribir pruebas unitarias y consignar a su rama (preferiblemente, no empuje a github pruebas unitarias fallando, ejecute `nosetests` localmente primero).
- escribir mas pruebas unitarias (idealmente, las pruebas forman la descripción completa de la función que está agregando o corrigiendo).
- Asegúrese de que la función que se agrega o corrige es realmente completamente descrita por las pruebas unitarias que usted escribió.
- Asegúrese de que las pruebas de unidad son atómicas, es decir, que cada una controle efectos de cambios relativos a una sola variable. no le dé a pruebas unitarias valores de entrada complejos o pruebas que no caben en una sola pantalla (25 líneas de código).
- Escriba el código que hace que las pruebas tengan éxito.
- actualizar los archivos de i18n (ejecutar “”./scripts/i18n.sh””).
- siempre que sea posible, traducir las cadenas nuevo que pones en archivos de código o glade.
- cuando modifique cadenas, por favor, asegúrese de que las traducciones antiguas siguen aplicandose.
- cometer los cambios.
- empuje a github.
- abrir una solicitud de extracción.

publicar en la linea de producción

please use the `publish.sh` script, in the `scritps` directory. This one takes care of every single step, and produces recognizable commit comments, it publishes the release on pypi, and in perspective it will contain all steps for producing a `deb` file, and a windows executable.

you can also do this by hand:

- abrir el tirón solicitud página usando como base una cadena de producción “” ghini-x.y””, en comparación con “” ghini-x.y-dev””.
- Asegúrese de que confirmar “” bache “” está incluido en las diferencias.
- debe ser posible fusionar automáticamente las ramas.
- crear la nueva solicitud de extracción, lo llaman como «publicar en la línea de producción».
- posiblemente necesita esperar para que travis-ci realizar los controles.
- combinar los cambios.

don’t forget to tell the world about the new release: on facebook, the google group, in any relevant linkedin group, and on our web page.

su propio “fork”

Si desea mantener su propio fork del proyecto, tenga en cuenta que esto está en continuo progreso, por lo que mantenerse actualizado requerirá algún esfuerzo de su parte.

La mejor manera de mantener el propio “fork” consiste en enfocarse en algún tema específico, trabajar relativamente rápido, a menudo abrir solicitudes de “pull” para su trabajo, y asegurarse que sea aceptado. Sólo siga el estilo de programación de Ghini, añada pruebas unitarias, concisas y abundantes, y no habrá ningún problema en ver su trabajo incluido en Ghini.

Si su *fork* se puso fuera de sincronización con Ghini de upstream: leer, entender, seguir el guía de github sobre configurar los remotos y como sincronizar.

paso final

- revisar el flujo de trabajo. considerar esto como una guía, a ti y a tus compañeros. por favor, ayudar a hacer mejor y coincidencia de la práctica.

7.1.12 Distributing ghini.desktop

Python Package Index - PyPI

This is not much mentioned, but we keep ghini.desktop on the Python Package Index, so you could install it by no more than:

```
pip install ghini.desktop
```

There are a couple packages that can't be installed with `pip`, but otherwise that's really all you need to type, and it's platform independent.

Publishing on PyPI is a standard `setup` command:

```
python setup.py sdist --formats zip upload -r pypi
```

Windows

For building a Windows installer or executable you need a running Windows system. The methods described here has been used successfully on Windows 7, 8 and 10. Windows Vista should also work but has not been tested.

If you are on GNU/Linux, or on OSX, you are not interested in the remainder of this section. None of Ghini's contributors knows how to produce a Windows installer without having a Windows system.

The goal of the present instructions is to help you produce a Windows installer, that is a single executable that you can run on any Windows workstation and that will install a specific version of ghini.desktop. This is achieved with the NSIS script-driven installer authoring tool.

As a side product of the installer production, you will have a massive but relocatable directory, which you can copy to a USB drive and which will let you use the software without needing an installation.

The files and directories relevant to this section:

- `scripts/build-win.bat` — the single batch script to run.
- `setup.py` — implements the NSIS and py2exe commands.
- `scripts/build-multiuser.nsi` — the nsis script, used by the above.
- `nsis/` — contains redistributable NSIS files, put here for conveniency.
- `ghini-runtime/` — built by `py2exe`, used by `nsis`.
- `dist/` — receives the executable installation file.

Most steps are automated in the `build-win.bat` script. Installation of a few tools needs to be done manually:

1. Download and install Git, Python 2.7 and PyGTK.

   This is outlined in the `devinstall`-based installation instructions.

2. Download and install NSIS v3.

3. A **reboot** is recommended.

4. Clone the ghini.desktop repository.

Use your own fork if you plan contributing patches, or the organization's repository `https://github.com/Ghini/ghini.desktop.git` if you only wish to follow development.

Clone the repository from GitHub to wherever you want to keep it, and checkout a branch. Replace `<path-to-keep-ghini>` with the path of your choice, e.g. `Local\github\Ghini\`. Production branch `ghini-1.0` is recommended as used in the example.

To do this, open a command prompt and type these commands:

```
cd <path-to-keep-ghini>
git clone <ghini.desktop repository URL>
cd ghini.desktop
git checkout ghini-1.0
```

The result of the above is a complete development environment, on Windows, with NSIS. Use it to follow development, or to propose your pull requests, and to build Windows installers.

All subsequent steps are automated in the `scripts\build_win.bat` script. Run it, and after a couple of minutes you should have a new `dist\ghini.desktop-<version>-setup.exe` file, and a working, complete relocatable directory named `ghini-runtime`.

Read the rest if you need details about the way the script works.

The `build_win.bat` script

A batch file is available that can complete the last few steps. To use it use this command:

```
scripts\build_win.bat
```

`build_win.bat` accepts 2 arguments:

1. `/e` — executable only.

   Produce an executable only, skipping the extra step of building an installer, and will copy `win_gtk.bat` into place.

2. `venv_path` — A path to the location for the virtual environment to use.

   Defaults to `"%HOMEDRIVE%%HOMEPATH%"\.virtualenvs\%CHECKOUT%-exe`, where `CHECKOUT` corresponds to the name of the branch you checked out.

If you want to produce an executable only and use a virtual environment in a folder beside where you have ghini.desktop, you could execute `scripts\build_win.bat /e ..\ghi2exe`

---

py2exe no funciona con eggs

Para producir un ejecutable Windows no debes instalar los paquetes como eggs. Hay varios métodos para alcanzarlo, por ejemplo:

- Install using `pip`. The easiest method is to install into a virtual environment that doesn't currently have any modules installed as eggs using `pip install .` as described below. If you do wish to install over the top of an install with eggs (e.g. the environment created by `devinstall.bat`) you can try `pip install -I .` but your mileage may vary.
- By adding:

```
[easy_install]
zip_ok = False
```

to setup.cfg (or similarly `zip_safe = False` to `setuptools.setup()` in `setup.py`) you can use `python setup.py install` but you will need to download and install Microsoft Visual C++ Compiler for Python 2.7 to get any of the C extensions and will need a fresh virtual environment with no dependent packages installed as eggs.

The included `build-win` script uses the `pip` method.

---

installing virtualenv and working with environments

Install virtualenv, create a virtual environment and activate it.

With only Python 2.7 on your system (where `<path-to-venv>` is the path to where you wish to keep the virtual environment) use:

```
pip install virtualenv
virtualenv --system-site-packages <path-to-venv>
call <path-to-venv>\Scripts\activate.bat
```

On systems where Python 3 is also installed you may need to either call pip and virtualenv with absolute paths, e.g. `C:\Python27\Scripts\pip` or use the Python launcher e.g. `py -2.7 -m pip` (run `python --version` first to check. If you get anything other than version 2.7 you'll need to use one of these methods.)

Populate the virtual environment

Install dependencies and ghini.desktop into the virtual environment:

```
pip install psycopg2 Pygments py2exe_py2
pip install .
```

Compile for Windows

Build the executable:

```
python setup.py py2exe
```

The `ghini-runtime` folder will now contain a full working copy of the software in a frozen, self contained state.

This folder is what is packaged by NSIS.

This same folder can also be transferred however you like and will work in place. (e.g. placed on a USB flash drive for demonstration purposes or copied manually to `C:\Program Files` with a shortcut created on the desktop). To start ghini.desktop double click `ghini.exe` in explorer (or create a shortcut to it).

Fixing paths to GTK components.

If you run the relocatable compiled program, unpackaged, you might occasionally have trouble with the GUI not displaying correctly.

Should this happen, you need to set up paths to the GTK components correctly. You can do this by running the `win_gtk.bat`, from the `ghini-runtime` folder.

You will only need to run this once each time the location of the folder changes. Thereafter `ghini.exe` will run as expected.

Finally, invoke NSIS

Build the installer:

```
python setup.py nsis
```

This should leave a file named `ghini.desktop-<version>-setup.exe` in the `dist` folder. This is your Windows installer.

---

about the installer

- Capable of single user or global installs.
- At this point in time ghini.desktop installed this way will not check or or notify you of any updated version. You will need to check yourself.
- Capable of downloading and installing optional extra components:
  - Apache FOP - If you want to use xslt report templates install FOP. FOP requires Java Runtime. If you do not currently have it installed the installer will let you know and offer to open the Oracle web site for you to download and install it from.
  - MS Visual C runtime - You most likely don't need this but if you have any trouble getting ghini.desktop to run try installing the MS Visual C runtime (e.g. rerun the installer and select this component only).

- Can be run silently from the commandline (e.g. for remote deployment) with the following arguments:
  - `/S` for silent;
  - `/AllUser` (when run as administrator) or `/CurrentUser`
  - `/C=[gFC]` to specify components where:

    `g` = Deselect the main ghini.desktop component (useful for adding optional component after an initial install)

    `F` = select Apache FOP

    `C` = select MS Visual C runtime

Debian

Between 2009 and 2010 someone packaged the then already obsolete Bauble 0.9.7 for Debian, and the package was included in Ubuntu. That version is still being distributed, regardless being it impossible to install.

Only recently has Mario Frasca produced a new bauble debian package, for the latest bauble.classic version 1.0.56, and proposed for inclusion in Debian. View it on mentors. This version depends on `fibra`, a package that was never added to Debian and which Mario also has packaged and proposed for inclusion in Debian. Mario has been trying to activate some Debian Developer, to take action. There's not much more we can do, other than wait for a sponsor, and hoping the package will eventually get all the way to Ubuntu.

Once we get in contact with a Debian Sponsor who will review what we publish on mentors, then we will be definitely expected to keep updating the debian package for `ghini.desktop` and `fibra`.

I am not going to explain in a few words the content of several books on Debian packaging. Please choose your sources. For a very compact idea of what you're expected to do, have a look at `scripts/pubish.sh`.

7.2 Template Letters

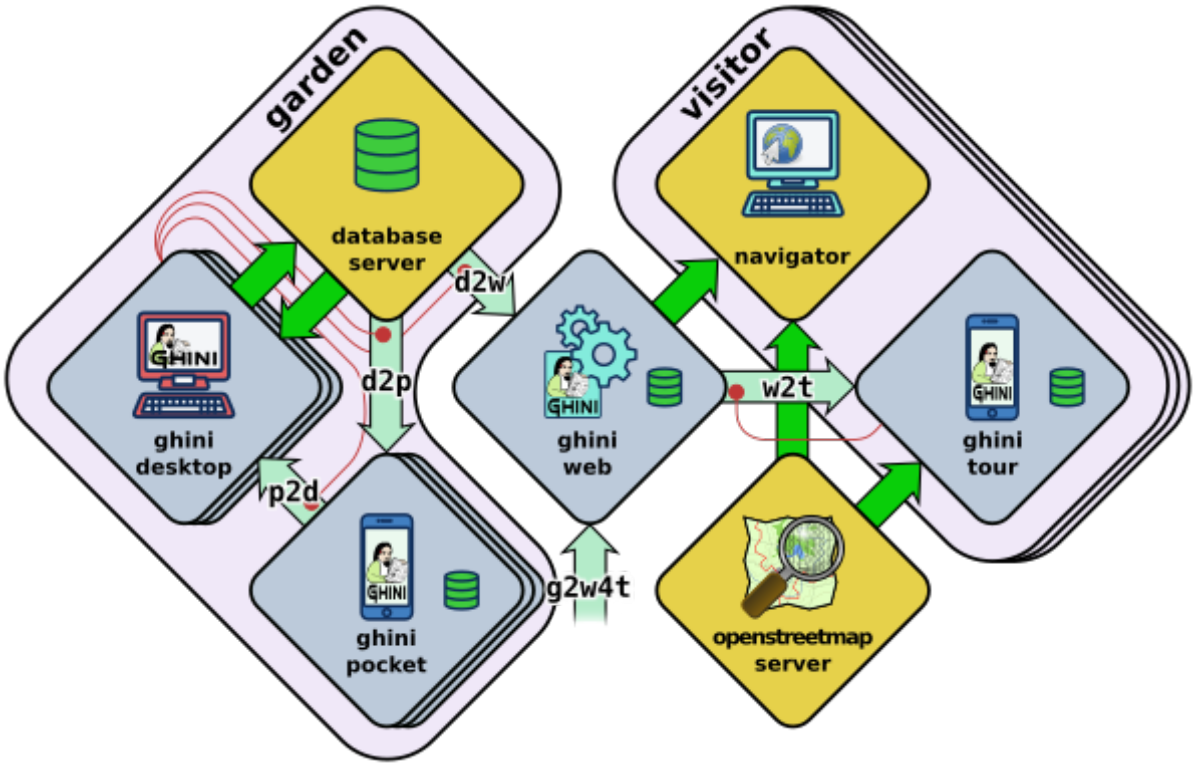
The reader getting to this point in the documentation probably understood that this Ghini project is above all a very open and collaborative project.

Here in this page you find some template letters, used to welcome new users, or that you can correct, print, and go with it to a garden, and propose them to adopt Ghini, or share with a group of your local friends, so you can make Ghini become a (voluntary, or paid) part-time job for you.

7.2.1 Dear conservator or scientist,

You are reading Ghini's presentation letter. Ghini is a libre software project on GitHub, focusing on botany. Brought to you by a small community of coders, botanists, translators, and supported by a few institutions around the world, among which, gardens that have adopted it for all their collection management needs.

The Ghini family is a software suite composed of standalone programs, data servers and hand-held clients, for data management, and publication:



- Ghini's core, `ghini.desktop`, lets you
  - enter and correct your data
  - navigate its links,
  - produce reports
  - import and or export using several standard or ad-hoc formats
  - review your taxonomy using online sources

  all according best practices suggested by top gardens, formalized in standard formats like ABCD, ITF2, but also as elaborated by our developers, based on the feedback of Ghini users.

  `ghini.desktop` is developed and continously tested on GNU/Linux, but runs equally well on Windows, or OSX. [1]

- `ghini.pocket` is your full time garden companion, an Android app installed from the Play Store,
  - assisting you in collecting or correcting data while in the field,

    - associate pictures to your plants, and verify taxonomic information.
    - Import your collected data into the desktop client when back in the office,

  `ghini.pocket` reduces the time spent in front of your desktop PC to a true minimum.

- `ghini.web` is a web server and a courtesy data hub service, offering you world wide visibility: Export a selection of your data from your desktop database, and handle it for publication to the Ghini project, and we will include it at http://gardens.ghini.me/, at no cost while we're able to do that, or for a guaranteed minimal amount of time if you are able to support our hosting costs. `ghini.web` serves a world map to help locate participating gardens, and within each garden, its contributed georeferenced plants.
- `ghini.tour`, a geographic tour Android app aimed at visitors, using OpenStreetMap as a base map, retrieving its data, gardens and virtual panels, from the web data aggregator `ghini.web`.

All software within the Ghini family is either licensed GNU Public License v2+ or v3+. It is a strong copyleft license. In short, the GPL translates the ethical scientific need to share knowledge, into legal terms. If you want to read more about it, please refer to https://www.gnu.org/licenses/copyleft.html

Ghini's idea about knowledge and software ownership is that software is procedural knowledge and as such, should be made a «commons»: With software as a commons, «libre software» and more specifically «Copylefted software», you not only get the source code, you receive the right to adapt it, and the invitation to study and learn from it, and to share it, both share forward to colleagues, and share back to the source. With proprietary software, you are buying your own ignorance, and with that, your dependency.

This fancy term «copyleft» instead of just «libre software», means the software you received is libre software with one extra freedom, guaranteeing every right you were granted upon receiving the software is never lost.

With copylefted software you are free —actually welcome— to employ local software developers in your neighbourhood to alter the software according to your needs, please do this on GitHub, fork the code, develop just as openly as the common practice within Ghini, and whenever you want, open a pull request so your edits can be considered for inclusion in the main branch. Ghini is mostly continuously unit tested, so before your code is added to the main branch, it should follow our quality guidelines for contributions. With libre software you acquire freedom and contribute to it, something that earns you visibility: Your additions stays yours, you share them back to the community, and will see them completed and made better by others. Having your code added to the main branch simplifies your upgrade procedure.

You can also contribute to the software by helping translate it into your native language. [5]

Some videos are published on YouTube, highlighting some of the software capabilities. [6]

Share back with the community. Several developers have spent cumulatively many thousand hours developing this software, and we're sharing with the community. We hope by this to stimulate a community sentiment in whoever starts using what we have produced.

Thanks for your consideration; please let me know if you have any questions,

In case you're interested in publishing your tree collection on the net, I would be happy to

include your plants, species, coordinates to http://gardens.ghini.me. Georeferenced textual information panels are also very welcome, all offered as a courtesy: We're still defining the offer. The idea behind this is allowing visitors to explore aggregated garden collections, and the current focus is on trees.

A small example: http://gardens.ghini.me/#garden=Jardín %20el %20Cuchubo

Mario Frasca MSc

[1] http://ghini.readthedocs.io/ - http://ghini.github.io/
[2] https://play.google.com/store/apps/details?id=me.ghini.pocket
[3] http://gardens.ghini.me/
[4] https://play.google.com/store/apps/details?id=me.ghini.tour
[5] https://hosted.weblate.org/projects/ghini/#languages
[6] https://www.youtube.com/playlist?list=PLtYRCnAxpinU_8WEDuRlgsYnNVe4J_4kv

7.2.2 free botanic data management systems

Many institutions still consider software an investment, an asset that is not to be shared with others, as if it was some economic good that can't be duplicated, like gold.

As of right now, very few copylefted programs exist for botanic data management:

- `ghini.desktop`, born as `bauble.classic` and made a commons by the Belize Botanical Garden. `ghini.desktop` has three more components, a pocket data collecting Android app, a Node.js web server, aggregating data from different gardens and presenting it geographically, again a geographic tour app aimed at visitors using the web data aggregator as its data source. You can find every Ghini component on GitHub: http://github.com/Ghini
- Specify 6 and 7, made a Commons by the Kansas University. A bit complex to set up, very difficult to configure and tricky to update. The institutions I've met who tried it, only the bigger ones, with in-house software management capabilities manage to successfully use it. They use it for very large collections. Specify is extremely generic, it adapts to herbaria, seed collections, but also to collections of eggs, organic material, fossils, preserved dead animals, possibly even viruses, I'm not sure. It is this extreme flexibility that makes its configuration such a complex task. Specify is also on GitHub: https://github.com/specify and is licensed as GPLv2+.
- Botalista, a French/Swiss cooperation, is GPL as far as rumours go. Its development has yet to go public.
- `bauble.web` is an experimental web server by the author of `bauble.classic`. `bauble.classic` has been included into Ghini, to become `ghini.desktop`. Bauble uses a very permissive license, making it libre, but not copylefted. As much as 50 % of bauble.web and possibly 30 % of ghini.desktop is shared between the two projects. Bauble seems to be stagnating, and has not yet reached a production-ready stage.

- `Taxasoft-BG`, by Eric Gouda, a Dutch botanist, specialist in Bromeliaceae, collection manager at the Utrecht botanical garden. It was Mario Frasca who convinced Eric to publish what he was doing, licensing it under the GPL, but the repository was not updated after 2016, April 13th and Eric forgot to explicitly specify the license. You find it on github: https://github.com/Ejgouda/Taxasoft-BG
- `BG-Recorder`, by the BGCI, runs on Windows, and requires Access. Developed mostly between 1997 and 2003, it has not been maintained ever since and isn't actively distributed by the BGCI. I've not managed to find a download link nor its license statement. It is still mentioned as *the free option* for botanic database management.

Of the above, only `ghini.desktop` satisfies these conditions: Copylefted, available, documented, maintained, easy to install and configure. Moreover: Cross platform and internationalized.

7.2.3 Welcome to Ghini/Bauble

Dear new user,

Welcome to Ghini/Bauble.

As the maintainer, I have received your registration for bauble.classic/ghini.desktop, many thanks for taking your time to fill in the form.

I see you are using bauble.classic-1.0.55, whereas 1.0.55 is the last released version of bauble.classic, however, bauble.classic is now unmaintained and superseded by the fully compatible, but slightly aesthetically different ghini.desktop. Install it following the instructions found at http://ghini.rtfd.io

The registration service says you're not yet using the newest Python2 version available. As of 2018-05-01, that is 2.7.15. Using any older version does not necessitate problems, but in case anything strange happens, please update your Python (and PyGTK) before reporting any errors.

Also thank you for enabling the «sentry» errors and warnings handler. With that enabled, Ghini/Bauble will send any error or warning you might encounter to a central server, where a developer will be able to examine it. If the warning was caused by an error in the software, its solution will be present in a subsequent release of the software

If you haven't already, to enable the sentry and warnings handler, open the «:config» page in Ghini and double click on the row «bauble.use_sentry_client».

I hope Ghini already matches your expectations, if this is not the case, the whole Ghini community would be very thankful if you took the time to report your experience with it.

The above is one way to contribute to Ghini's development. Others are: - contribute ideas, writing on the bauble google forum (https://groups.google.com/forum/#!forum/bauble), - contribute documentation, or translations (https://hosted.weblate.org/projects/ghini/), - give private feedback, writing to ghini@anche.no, - rate and discuss Ghini openly, and promote its adoption by other institutions, - open an issue on GitHub (https://github.com/Ghini/ghini.desktop/issues/), - contribute code on GitHub (fork the project on (https://github.com/Ghini/ghini.desktop/), - hire a developer and have a set of GitHub issues solved, per-haps your own - let me include your garden on the still experimental worldmap (http://gardens.ghini.me)

I sincerely hope you will enjoy using this copylefted, libre software

Best regards, Mario Frasca

https://ghini.github.io https://github.com/Ghini/ghini.desktop/issues/

7.2.4 Do you want to join Ghini?

Nota: I generally send a note similar to the following, to GitHub members who «star» the project, or to WebLate contributors doing more than one line, and at different occasions. If it's from GitHub, and if they stated their geographic location in their profile, I alter the letter by first looking on institutos botánicos if there's any relevant garden in their neighbourhood.

Dear GitHub member, student, colleague, translator, botanist,

Thank you warmly for your interest in the Ghini project!

From your on-line profile on github, I see you're located in Xxxx, is that correct?

If you are indeed in Xxxx, you live very close to gardens Yyyy and Zzzz. Maybe you would consider the following proposition? All would start by contacting the botanical garden there, and get to know what software they use (what it offers, and at which price) and if they're interested in switching to ghini.desktop+pocket+tour+web.

The business model within Ghini is that the software is free and you get it for free, but time is precious and if a garden needs help, they should be ready to contribute. Maybe you already have a full-time job and don't need more things to do, but in case you're interested, or you have friends who would be, I'm sure we can work something out.

Let me know where you stand.

best regards, and again thanks for all your contributed translations.

Mario Frasca

CAPÍTULO 8

Apoyar Ghini

Si usted utiliza Ghini, o si le parece poder apoyar su desarrollo, por favor considere una donación.