# microSICol Install Guide

last update: Wed Apr 23 16:22:00 2014

## Introduction

microSICol supports multiple operating systems as host, however, this document will only guide you in a Unix-like system, e.g. Linux. It shouldn't be hard to adapt the instructions here to other systems.

This document *assumes* you have some familiarity with system administration techniques. microSICol is **not** a general package. It's a niche software a regular user wouldn't try to install. There is no way to blindly follow steps to get microSICol running. The discernment of the sysadmin is always required, since there is no system that's equal to another. Some of them change the default path of configuration files (e.g. some systems put `httpd.conf` in `/etc/httpd/conf/` instead of `/var/www/conf/`), some change the contents of the configuration files of packages, and so on.

## Requirements

- Apache 2.2.x
- Python 2.6.x (it will not run with Python 3)
- SQLite 3.6.x
- MySQL Server 5.1.x
- MySQL-python 1.2.x
- python-sqlite2 2.3.x

The detailed instructions on how to add these packages are very system-specific and will depend on what OS and version you're trying to install microSICol. In Unix-likes, this means using the *package manager* available (e.g. *apt* on Debian-based systems like Ubuntu or *yum* on Red Hat-based systems) or manually installing each package. Every system administrator has its own techniques, however, the latter is not recommended. Try to use what's available on the OS you're installing microSICol on.

## Getting microSICol

A compressed zip file containing microSICol can be obtained by the following link:

https://github.com/cria/microSICol/archive/vx.xx.zip

Extracting it will yield a directory called `microSICol-vx.xx/`.

# microSICol Install Guide

last update: Wed Apr 23 16:22:00 2014

## Extracting

Here is assumed that you downloaded the compressed version of microSICol. The first thing to do is to extract the package in a appropriate directory. microSICol is a web application, so this usually means the `/var/www/` or `/var/www/htdocs/` directory in Unix-like system. However, this can greatly vary between different OS. It's also better to create a separate directory to install microSICol, to keep things clearer and not affect what may be eventually using the default directory:

```
# mkdir /var/www/htdocs/microsicol
```

Copy the microSICol package to this directory and extract it:

```
# cp microSICol-xxxxxx.zip /var/www/htdocs/microsicol
```

Enter the directory and extract the file:

```
# cd /var/www/htdocs/microsicol
```

```
# unzip microSICol-latest.zip
```

Now you're ready to configure Apache.

## Configuring Apache

Configuring Apache means you'll have to deal with the `httpd.conf` file. This file is usually located at `/var/www/conf/` directory, but as already noted in the *Introduction* section, this can vary.

Below is a `VirtualHost` container included in a `httpd.conf` for a common configuration of microSICol:

```
  <VirtualHost *:80>
          ServerName microsicol.xxx.xxx.br
          ServerAdmin suporte@xxx.br
          DocumentRoot /var/www/sicol/sistema/
          LogFormat combined
          ErrorLog  /var/log/httpd/microsicol.xxx.br.error.log
          CustomLog /var/log/httpd/microsicol.xxx.br.access.log
combined
          DirectoryIndex index index.html index.php index.py
          ScriptAlias /py/ /var/www/sicol/sistema/py/
          <Directory "/var/www/sicol/sistema/">
                  AddHandler cgi-script .cgi .py
                  Options ExecCGI
                  AllowOverride All
                  Order allow,deny
                  Allow from all
          </Directory>
  </VirtualHost>
```

There are some important directives you must be aware when configuring Apache to run microSICol. They are:

`VirtualHost` is a container which isolates the configuration if you want to maintain multiple domains/hostnames on your machine. Even if you'll use just one domain, it's still recommended to use it;

`Directory` is the one where `microsicol-latest.zip` was extracted;

`ServerName` is a valid DNS name for your host;

`ServerAdmin` is the e-mail of the administrator and where problems with the server is sent;

`ErrorLog` errors on the context of Apache, and it's not generated by microSICol;

`CustomLog` same as above;

`DirectoryIndex` what kind of files Apache looks when accessed;

`ScriptAlias` this is important, and in this case is the same as the `DocumentRoot`. The user Apache is running must have proper permissions to this directory;

This snippet is necessary for the catalog:

```
ScriptAlias /pycat/ /var/wwwsicol/catalogo/pycat/
<Directory /var/www/sicol/catalogo/pycat/>
        AddHandler cgi-script .cgi .py
        AllowOverride All
        Options ExecCGI
        Order allow,deny
        Allow from all
</Directory>
```

## Setting Up Databases

The first thing to do is to copy two files from the microSICol distribution to the directory where microSICol will be installed. These files are `config_default.xml` and `sqlite_default.db`. They need to be renamed respectively to `config.xml` and `sqlite.db`. More specifically, assuming that you're installing microSICol in `/var/www/sicol/sistema` directory as before, they'll become `/var/www/sicol/sistema/config.xml` and `/var/www/sicol/sistema/db/sqlite.db`.

In `config.xml`, you'll need to change the following line to reflect your current hostname:

```
<config name="index_url">http://microsicol.xxx.br/sicol/</config>
```

And the following line to reflect the current directory of your current install directory:

```
<config name="root_dir">/var/www/… </config>
```

You'll also need to setup the databases too. microSICol uses SQLite to keep track of it's users, but for its collection data, can use a variety of RDBMS, such as MySQL, PostgreSQL, among others. At this moment, only MySQL is activated in the code base, and it's what we'll use here.

To set the root password of MySQL:

```
# mysqladmin -u root password "mypass"
```

Of course you can choose whatever password you like. And it's better to choose a password with a strong strength, since it lowers overall risk of a security breach.

Enter the directory where the init scripts are stored:

```
cd /var/www/htdocs/microsicol/db/scripts/v110/
```

We'll need to initialize the databases.

The first of them is the log database, which is where microSICol keeps track of all the changes made in the system:

```
# mysql -u root -pmypass < mysql_log_script_empty.sql
```

Run the following script:

```
# mysql -u root -pmypass < mysql_script_empty.sql
```

You'll need to customize the script `mysql_script_start_data.sql`. The standard version is well-suited for most uses, but the needs of different biological collections may greatly vary.

To finish, run the following 3 scripts:

```
# mysql -u root -pmypass < mysql_script_start_data.sql
# mysql -u root -pmypass < mysql_taxonomy.sql
# mysql -u root -pmypass < mysql_countries.sql
```

If all steps went well, you should be able to start microSICol through the configured server URL using the username and password *"sicol"* at first access.

If you have any problems with the first access to the system, check the permissions of the folder `/system/py`.

If you have any other problems, the file named `/system/py/sicol.log` can be of help