How to set up the toolchain for coding MIOS32 apps with OS X

You have probably already installed the developper tools provided by apple. You need them before going further. (if you have installed XCode, that's fine).

steps:

0- about the terminal
1- get the mios32 files from the repository server
2- install the STM32 toolchain
3- configure the paths

0- about the terminal

Most of the things now will be done with the Terminal, if you don't know anything about it here are the basic commands you need:
Launch it from your “utilities” folder.
After each command, press “enter” to validate it.

Type “pwd” to show the actual path of where you are.
Type “ls” to list all the files and directories in your current directory
Type “cd aDirectoryName” to go inside that directory
Type “cd aPath” to go to a particular path
Type “cd ..” to go to the parent directory
Type “cd ~” to go to your home directory

A nice shortcut if you want to go to a particular folder without having to type the full path is to type “cd ” in the terminal, then open the parent directory of the one you want to go in with the finder, and drag and drop the folder icon inside the terminal. It's path will appear by magic, press “enter”.

1- get the files from the repository server

The simplest way to do this is to get the “svnX” software, launch it, in the “repositories” window click on the “+” sign, give a name to the repository, put the repository path in the path field: svn://svnmios.midibox.org/mios32 and double clic on the repository name to get a connexion.
Then you are able to drag and drop the “trunk” folder from the server to wherever you want on your computer. I suggest that you create an “snv” folder in your home directory, then a “mios32” folder and to put the dragged “trunk” folder inside that folder. Now the Unix path of your mios32 files is “~/svn/mios32/trunk” (“~” stands for “home directory”).

You can also do this from Xcode itself:
In the “SCM” menu choose “configure SCM repositories”.
Under the repositories list clic on the “+” button to add a repository. Give it a name (mios32 for example), and change the “SCM system” to “Subversion”.
Then in the “path” field put the repository server path: svn://svnmios.midibox.org/mios32 and clic “OK”.
In the “SCM” menu choose “repositories”, you now have the repository window where you can access

to the mios32 files.

2- install the STM32 toolchain

A nice pre-built binaries archive can be found at this page:
“www.paintyourdragon.com/uc/osxstm32/index.html”.
You don't need to do everything written on this page as we won't use Circle OS and FatFryer.
Just download the pre-built binaries in the section “acquiring the software”. These will work on OS X 10.5, for earlier OS version you'll probably have to download the source files and compile them, I haven't checked how you do that.
Expand the archive, you will then have a “stm32-bin” folder containing another archive file and a “README” file. Follow the instructions given in the “README” file to set up the toolchain.
Check that all the files expanded properly in your /usr folder.
The trick is that the /usr folder is hidden by the finder so if you want to open it, in the finder's menu click on “go” → “go to folder” and just type “/usr” in the pop up window, your “/usr” folder will open and you'll be able to check the files. You should have a folder called “stm32” inside the “local” folder.

Personally I tried to expand the files with the terminal as said in the “README” but it didn't work, don't know why. I just double clicked the archive and it all expanded in a new “usr” directory made in the same folder as the archive. I then opened this folder and its “local” subfolder, copied the “stm32” subfolder inside the real “/usr/local” folder of my computer and it worked fine. You now need to set the path variable for the toolchain, see the next section about this

3- set up the paths variables

For those who don't know anything about Unix like me a few days ago, here's a quick explanation:
In Unix we call a shell the piece of software that provides an interface for users. For example, when you use the terminal in OS X, you are talking to the computer through the “bash” shell.
Your compiler also uses the shell to access to the different files involved. Thus, it has to know where to find these files. For this, we set up variables in the shell. For example, we will put the path of your mios32 folder in the variable “MIOS32_PATH” so that when the compiler will want to check inside your “mios32” folder, it will just call this variable.

Setting up path variables in the shell is quite easy: in a terminal window just type
`export VARIABLE_NAME=something`
and if you want to check to what the variable has been assigned, type:
`echo $VARIABLE_NAME`
this will print what's inside the variable.
For example, type
`export MIOS32_PATH=~/svn/mios32/trunk`
to set up the mios32 folder path, then type
`echo $MIOS32_PATH`
to check that the path variable has been assigned.

In our case, we need to set up the path variable for the STM32 toolchain: type in the terminal
`export PATH=$PATH:/usr/local/stm32/bin`
and then we need to set up variables for mios32 itself:
`export MIOS32_PATH=~/svn/mios32/trunk`

```
export MIOS32_BIN_PATH=$MIOS32_PATH/bin
export MIOS32_GCC_PREFIX=arm-none-eabi
export MIOS32_FAMILY=STM32F10x
export MIOS32_PROCESSOR=STM32F103RE
export MIOS32_BOARD=MBHP_CORE_STM32
export MIOS32_LCD=clcd
```

Here I considered that you have put all the files downloaded from the svn server in the directory "~/svn/mios32/trunk" but up to you to put them elsewhere if you want and change the "MIOS32_PATH" variable accordingly.

Now everything is set up to compile properly your projects. Test if everything is OK, open a terminal window, type

```
cd $MIOS32_PATH/apps/tutorials/001_forwarding_midi
```

to go to the "forwarding_midi" tutorial folder and type

```
make -s
```

you should have a return looking like this:

```
————————————————————————-
Application successfully built for:
Processor: STM32F103RE
Family: STM32F10x
Board: MBHP_CORE_STM32
LCD: clcd
————————————————————————-
```

and with the finder, go to this tutorial folder, you should have now a .hex file uploadable to your core32 with "MIOSStudio beta 9".

The annoying point there is that the variables you set up in the shell just disappear when you shut down or log off your computer. That means you have to set them up again when you power on your computer again. There's a way to avoid this annoying stuff:
The shell will always read particular files before starting up. If you put these variables there they will always be set up. I didn't really understood if the proper file was "~/.bash_profile" "~/.bash_login" "~/.profile" or "/etc/profile" to do that, it looks like they all do the job.
Moreover, all these files are hidden from the finder, so the best way to edit them is to open a terminal window and edit them with the "pico" editor.
If you don't understand what's going on, here's what I did:

in a terminal window, type:

```
pico ~/.profile
```

then copy and paste these lines inside the pico editor:

```
export PATH=$PATH:/usr/local/stm32/bin
export MIOS32_PATH=~/svn/mios32/trunk
export MIOS32_BIN_PATH=$MIOS32_PATH/bin
export MIOS32_GCC_PREFIX=arm-none-eabi
export MIOS32_FAMILY=STM32F10x
export MIOS32_PROCESSOR=STM32F103RE
export MIOS32_BOARD=MBHP_CORE_STM32
export MIOS32_LCD=clcd
```

do "ctrl-O" to save the file
do "ctrl-X" to exit

log off or shut down your computer
login of start up the computer
open a terminal window and check if the variables are all set up:
`echo $MIOS32_PATH` for example
if they are not, that's probably another file you need to edit.
Useful info about bash and initialisation files here:
http://johnnywey.wordpress.com/2008/04/17/fixing-bash-profile-in-os-x/
and here:
http://macdevcenter.com/pub/a/mac/2004/02/24/bash.html