Quick Guide on Objective C and GNUStep for FreeBSD

Sunday, 13 May 2012
Last Updated Monday, 09 July 2012

In order to program in Objectiv C (which is supported by FreeBSD using GCC compiler) it will be usefull to instal GNUStep Library:

In order to install GNUStep first edit your /etc/make.conf file and add the following lines:

```
CC?=clang
CXX?=clang++
WITH_GNUSTEP_DEVEL=yes
GNUSTEP_WITH_LIBOBJC2=yes
```

Then install GNUStep library:

```
cd /usr/ports/devel/gnustep
make install clean
```

You can also install gnustep-gui (GUI Library) and gnustep-back (GUI Backend):

```
cd /usr/ports/x11-toolkits/gnustep-gui
make install clean

cd /usr/ports/x11-toolkits/gnustep-back
make install clean
```

Now we must add GNUSTEP_SYSTEM_TOOLS variable to our shell environment. If you are using .csh (usually used by root, if you did not change to other shell) then edit ~/.cshrc file and add the following line:

```
setenv  GNUSTEP_SYSTEM_TOOLS    /usr/local/GNUstep/System/Tools
```

If you are using sh shell, then edit ~/.profile file and add the following lines:

```
GNUSTEP_SYSTEM_TOOLS=/usr/local/GNUstep/System/Tools

export GNUSTEP_SYSTEM_TOOLS
```

Now let's make a simple application (example.m), which we will save in our homedir:

example.m#import <Foundation/Foundation.h>

```
int main( int argc, const char *argv[] ) {

   NSLog (@"First ObjC Program");

   return 0;
}
```

In order to compile it we will create a makefile which must be called GNUmakefile (case sensitive) when using GNUstep. So let's create GNUmakefile and add the following lines to it (asuming our source code is example.m and our binary file will be called exampleapp):

GNUmakefileGNUSTEP_MAKEFILES=/usr/local/GNUstep/System/Library/Makefiles

include $(GNUSTEP_MAKEFILES)/common.make

TOOL_NAME = exampleapp

exampleapp_OBJC_FILES = example.m

include $(GNUSTEP_MAKEFILES)/tool.make

Now we will compile our example.m file by running:

```
gmake
```

Then we will run our newly created binary:

```
# obj/exampleapp
2012-05-13 15:46:17.099 exampleapp[87927] First ObjC Program
```

Notice that binary files are created in obj subdirectory.

We've successfully compiled our first command line Objective C app. Now let's make an GUI app.

For that we will write the following code (example2.m):
example2.m#import <Foundation/Foundation.h>

#import <AppKit/AppKit.h>

int main (void)

{

NSAutoreleasePool *pool;

pool = [[NSAutoreleasePool alloc] init];

[NSApplication sharedApplication];

```
  NSRunAlertPanel (@"First App", @"Our First GUI App using GNUStep",
                   @"exit", nil, nil);

  [pool drain];

  return 0;
}
```

Here is the GNUmakefile for our example:
GNUmakefileGNUSTEP_MAKEFILES=/usr/local/GNUstep/System/Library/Makefiles

```
include $(GNUSTEP_MAKEFILES)/common.make

APP_NAME = example2app

PACKAGE_NAME = example2app

VERSION = 1.0

example2app_OBJC_FILES = example2.m

include $(GNUSTEP_MAKEFILES)/tool.make

include $(GNUSTEP_MAKEFILES)/application.make
```

To launch our app we will run:

```
  openapp ./example2app.app
```

We will get something like this:

Notes:
- to compile our app with debug information in order to debug the app we will run:
gmake debug=yes

- to compile our app faster using multiple threads we will run:
gmake -j 4 # asuming we have a CPU with multiple cores