Getting started

Before you start coding, look at the sources and determine area in which you are interested to develop. We have wide range of tasks and, most probably, one of them will be interested for you. Only remember, our current main goal is to provide less or more full set of OS/2 API on top of L4 microkernel.

Development Tools

Language	Compiler
C	Open Watcom
C++	Open Watcom
FORTRAN	Open Watcom
Pascal	FreePascal
REXX	ReginaREXX

Other languages may be used to develop parts of the OS. However, they must be open source and use the OS/2 API. Usage of free (read: can be downloaded from lot of places) not premited, but not recommended. Also, use the Watcom tools to build this source.

When conflicting tools exists (ex. NMAKE and WMAKE) always use the Open Watcom tool.

Downloading and Compiling

First of all you need to download all above tools for your platform from corresponding sites:

- http://www.openwatcom.org/ for OpenWatcom
- http://www.freepascal.org/ for FreePascal
- http://regina-rexx.sf.net/ for ReginaREXX

You can download non-regular source snapshot from the site or latest sources from SVN. osFree sources hosted at Sourceforge SVN. Sourceforge also provide possibility to download any revision as .tar.gz archive.

Before compiling check files setvars-somename.cmd and somename.conf file, and change settings according your system. After changes just execute setvars-somename.cmd and enter

```
wmake
```

build process will be started. For clean up source tree say

```
wmake clean
```

For more information about build system read Build system document.

The Directory Tree

Please look at the SVN code tree to understand how files are to be placed. Please understand that the osFree SVN tree containg the code for an operating system and some toolkit tools. Do NOT place unrelated tools or applications in this tree.

Global/Shared/Private Headers

Each level of the SVN tree contains two standard directories:

```
FIX THIS! Now it's slightly different!
```

shared	Contains code shared among all source at this level and deeper levels
include	Contains header files for the above

Each levels/part of the OS should have a specific prefix that allows a developer to easily find what part of the OS a header/library file belongs to. For example code shared by the whole tree should be included with:

```
#include <all_shared.h>
```

and code shared by all commandline tools should include:

```
#include <cmd_shared.h>
```

Try to create as few shared code headers as possible. Each “shared” directory should contain one (1) library (.lib) file (xxx_shared.lib) with all shared code and each “include” directory should contain one main header file including all other (xxx_shared.h).

Example of use of common files:

```
// Use the normal OS/2 INCL_ since our toolkit is the OS/2 toolkit
#define INCL_DOSERROR

// Do NOT include os2.h, use osfree.h instead
#include <osfree.h>

// Include any needed normal C library
#include <malloc.h>
#include <string.h>

// Include all shared code and shared code for command line tools
#include <all_shared.h>
#include <cmd_shared.h>
```

Documentation

- Private code (not shared with other code) should be documented only in the source.

- Shared code (shared with code at the same level or at all levels) should be documented in source and in a “building and developing” document.
- The API of the OS and the tools documentation should NOT be documented in the source tree but in the toolkit and release tree.
- Source code should be documented in the source file (not the header files).
- Each function should be prefixed with a description of what it does, what parameters it uses (in and out) and any external references it uses.
- Place comments in the source that helps the reader to understand the logic and don't overdo it.

When Developing

- Use static linking, do not use dynamic libraries (LIBC style) or dynamic runtime.
- Use the makefiles provided with the source tree, don't “do your own”.
- Currently osFree development is done on OS/2 (minimum Warp 4) but in the future development will be hosted on osFree.
- We use SVN to share code among developers.
- We use Doxygen and Wiki to document our work.

Submitting a Patch (FIX THIS!!!)

- Make sure your changes follow the coding guidelines above.
- Make sure you are using the current versions of the sources so that the resulting diffs are comparing your changes with the head of the source tree.
- Create your patch either by using cvs diff -u (if you are using CVS) or diff -u original-file changed-file (if you are using a source archive - you can also create differences for the whole directory contents using diff -r) In the latter case include the old code first, the new code last – in the patch anything you added will be prefixed with a +.
- Remove all/any lines that reference files without changes.
- Send the patch file as an attachment in your email. Do not paste the patch directly into the email body.
- Maintainers will often reply in response to your patch, pointing out things to fix up, etc. before a patch can be checked in. Please always follow the maintainer suggestions closely and respond by sending a new corrected patch. Please do not expect the maintainers to rework your changes, you want to be able to claim all the credit for your great patches!

From:
https://ftp.osfree.org/doku/ - **osFree wiki**

Permanent link:
https://ftp.osfree.org/doku/doku.php?id=en:develop:guidelines&rev=1402587481

Last update: **2014/06/12 15:38**