

# Building on Mac OS X

- [1 Building the FireBreath Plugin](#)
- [2 Requirements](#)
- [3 Get the source](#)
- [4 Generate the example project files](#)
- [5 Generate your own project files](#)
- [6 Build the Plugin](#)
  - [6.1.1 The Xcode Projects](#)
  - [6.2 Using Xcode IDE](#)
- [7 Make the plugin accessible to browsers](#)
  - [7.1 Installing a plugin for the current user](#)
  - [7.2 Installing a plugin for all users](#)
- [8 Open in your browser and play with it](#)
- [9 A few JS commands to try:](#)

## Building the FireBreath Plugin

For more info on Mac specific details, see [Mac Plugins](#).

See Also: [FireBreath Tips: Working with Source Control](#)

## Requirements

- CMake version 2.8.8 or later, from <http://www.cmake.org/cmake/resources/software.html>.
  - **Make sure to grab the Binary distribution (cmake-*ver*-Darwin[64]-universal.dmg).**
- Apple's Xcode, available from your Mac App Store.
- Xcode's Command-Line Tools. To get these, first install Xcode. Then launch Xcode and click in the menu: Xcode Preferences Downloads. Select "Command Line Tools" and press 'Install'.
- Git (if you want to check out from source): <http://git-scm.com/>

## Get the source

First thing is first; get the source code.

To get a copy of the source, see the [download page](#).

## Generate the example project files

To generate the project files execute:

- `prepmac.sh examples`

The project build files will all be generated into the `buildex/` directory under the project root.

If an error is indicated or if cmake crashes, this may be due to [CMake Bug 13463](#). If the bug has not been fixed yet, use the workaround is given in *Step 5* of that bug's *Steps to Reproduce* which is better explained in [this post](#).

## Generate your own project files

If you have created your own project in the `projects/` directory, you can build it by running the same command as for example projects, but without "examples".

To generate the project build files:

- `prepmac.sh`

The project build files will all be generated into the `build/` directory under the project root - this only has to be done after either adding or generating another FireBreath project.

Don't forget to check the [Prep Scripts](#) page if you want to build for specific CPU architectures or against different SDKs.

## Build the Plugin

## The Xcode Projects

As alluded to in the [general overview](#), if you have  $N$  *plugin projects* in your `projects/` directory, running `prepmac.sh` will create  $N+1$  Xcode projects. That is, for each of your *plugin projects*, you'll get a *single-plugin* Xcode project, and you'll also get a *combined* Xcode project, in which each plugin project is a Bundle Target. The *single-plugin* projects are nicer to look at in the IDE, but they have only one real Target (the plugin bundle) and do not make a reference to nor have a dependency set upon the half-dozen FireBreath and Boost libraries, even though in fact they do depend on them. The Firebreath and Boost libraries are only built by the *combined* Xcode project, which has proper dependencies set within it. Therefore, you *must* build the *combined* project first, in all desired configurations, before building any of the *single-plugin* projects. And unless you're one of those people who can track project dependencies in their head for months and years, it's probably safer to use the *combined* project for all purposes.

### Using xcodemake

The example plugin can be built by changing to the directory `buildex/` and executing:

```
xcodebuild
```

To build your own plugin project (using the *combined* Xcode project), change to the directory `build/` and execute:

```
xcodebuild
```

## Using Xcode IDE

Open the *combined* project with Xcode:

- `buildex/FireBreath.xcodeproj` for the examples
- `build/FireBreath.xcodeproj` for your own projects

From there you can either build all projects (`ALL_BUILD`) or specific plugin targets.

## Make the plugin accessible to browsers

You have the option making the plugin available to all users or to simply install it for the local user only. Unless there is a good reason for it, install it for the current user only.

### Installing a plugin for the current user

To install a plugin for only the currently logged in user, symlink the <Plugin Bundle> to `~/Library/Internet Plug-Ins`

```
ln -s buildex/projects/FBTestPlugin/Debug/FBTestPlugin.plugin ~/Library/Internet Plug-Ins/
```

### Installing a plugin for all users

To make a plugin globally available to all users, symlink the <Plugin Bundle> to `/Library/Internet Plug-Ins`

```
ln -s buildex/projects/FBTestPlugin/Debug/FBTestPlugin.plugin /Library/Internet Plug-Ins/
```

## Open in your browser and play with it

Open the file `buildex/projects/FBTestPlugin/gen/FBControl.htm` in your preferred browser

Use Jash or firebug (or whatever) to make calls on the plugin. For supported calls, check out `projects/TemplatePlugin/MathAPI.cpp`.

## A few JS commands to try:

```
plugin().echo("echo this string!")
"Echoing: echo this string!"
```

```
plugin().valid  
true
```