

This is the source for the GameKit.

I have already built it (under NS 3.1, for Motorola and Fat) so you should only need to rebuild it under special circumstances. You will find them in the folders `^built-m68k^` and `^built-fat^`. Although built under 3.1, the Motorola libraries should work on 3.0 systems. The prebuilt libs are as follows:

gamekit	-	gamekit, but without MusicKit support
gamekit_mk	-	gamekit with MusicKit support (see Makefile to turn this on)
gamekit_g	-	gamekit debug library without MusicKit support
gamekit_mk_g	-	gamekit debug library with MusicKit support

Note that the versions which support the MusicKit are only available in the thin (Motorola) versions. This is because the MusicKit does not currently exist for Intel hardware. When it eventually gets ported, I will do what I can to support it. Until then, you'll have to live without.

In order to use the GameKit, you need to move the appropriate libs to `/usr/local/lib` and the "gamekit" folder in `../Headers` should be placed in `/LocalDeveloper/Headers`.

This release of the gamekit relies upon another library I'm slowly adding object to: the daymiskit. These are objects which are used by the gamekit - and other projects I work on - but which are useful in more than just games. In order to compile an app that uses the gamekit, you'll need to also include the daymiskit in the list of libraries to be linked. If you're using the MusicKit version, then you'll also need to link in the following libraries (before the GameKit and in this order): unitgenerators, synthpatches, musickit, and dsp. All four libraries are found in the latest CCRMA MusicKit release (3.1).

## Things to note:

- This version is the most stable so far. I've fixed zillions of bugs and added all kinds of neat things. Note that if you had a High Score server installed, you need to re-install with the new one!
- The high scores and sound systems are completely different. I think it's now a lot better, and the rest of the kit will probably see similar enhancements as I get the time. Feel free to send suggestions and/or comments about what you'd like to see.
- The GameInfo object will be placed on an IB palette so that you can do most of the GameKit configs in IB without subclassing. (Right now, do **not** use the palette! I am considering breaking up the GameInfo object a bit, and the changes will cause you headaches if you attempt to use the palette.)

- I plan to eventually change all the object names, constants, and types to have GK as a prefix so that I can be sure that they'll not conflict with NeXT kits. Let's hope they don't start using <sup>a</sup>GK°! You'll notice that some objects are starting to do this already...
- To make it easier to make the .nibs, if you don't make all the needed connections between the various objects, they will connect themselves up. For this to work, you must connect the GameBrain up to *all* it's sub objects; the other objects set up their connections via the GameBrain. This should make things a lot simpler. If you do make the connections in IB, then the self-connecting behaviour won't happen; only if you leave the outlets unconnected will they try to self-connect. In some cases, the GameBrain will create an object if it is missing, but for most sub-systems, it will not.
- The new docs and sample .nibs for the high scores are wrong. They're close, but not quite right. Let me migrate my recent changes into them before you trust them. The PacMan app has most correct HighScore.nib of any of them.
- Well, that's the main stuff. There's tons yet left to do, but hopefully things are starting to look more like they will in the 1.0 version. Until then, I'm not going to be very concerned about compatibility between releases, as you might have already noticed.

As always, talk to me if you have questions, problems, comments, or requests!

Don\_Yacktman@byu.edu