Risk
Computer Player Reference

This section of the Help discusses how to implement your own computer player strategies in Objective C.

It has undergone some changes since version 0.97. This should make it easier to make conformant computer players. See the Chaotic player's subproject for a good example of a computer player bundle.

All players are subclasses of RiskPlayer (or a subclass of RiskPlayer). Each computer player resides in its own directory inside the Risk app wrapper. The computer players that come with Risk are no different from your own. The directory's name must end in ".riskplayer". Inside this directory is at least a Mach-O file and

At launch, Risk searches it's app wrapper. It dynamically loads all object files it can find that are specified by the NSExecutable key in the Info-nextstep.plist of player bundles. Other files such as .nib files can be put in the directory as well if you need them. See the Chaotic.riskplayer directory in the Risk distribution for an example of how this works.

There is no longer a Diagnostic computer player. Instead, the RiskPlayer class provides a console window for writing messages, with a switch to force the player to pause until the continue button is pressed (much like the Dice Inspector panel.) The console panel for each player may be shown by selecting the Tools -> Player N -> Console... menu item. Messages don't appear in the console until the console is shown on screen for the first time.

Some auxiliary files are used by Risk if they are specified in the Info-nextstep.plist file. When someone chooses "About..." from the Setup Panel, an about panel is brought up. If an image is specified by the PlayerIcon key in the .plist file, that image is shown in the top left part of the panel. If an RTF file is specified by the AboutPlayerFile key in the .plist file, this is loaded into the ScrollView. The image file will be scaled to a 48 by 48 icon like an app icon.

For example, this is the CustomInfo.plist file for the Chaotic player:

```
{
        SuperClass = RiskPlayer;
        PlayerTypeName = Chaotic;
        AboutPlayerFile = Chaotic.rtf;
        PlayerIcon = Chaotic.tiff;
}
```

The RiskPlayer class implements several utility methods which you will use in your own code, as well as defining the methods that you must implement. See the source code for RiskPlayer for a description of each of the methods.

Supplied Utilities

The RiskPlayer class has several methods which are not meant to be overridden. These methods are utilities for interacting with Risk and finding out certain things about the game status. The RiskGameManager has methods that must be called during game play. The RiskWorld, Continent and Country classes also have some useful methods. RiskCard and CardSet should be the only other classes you need to know about.

Pitfalls and Gotchas

The current API does a bit more to restrict cheating or simple errors by computer players, but they can still directly manipulate the assorted objects to their advantage. A disadvantage to this is that the checks are simply assertions, which could halt the game if they fail, say for an incomplete computer player.

Another consideration is that not returning from any method you implement will cause Risk to hang.