Game Porting Toolkit 1.0 Beta 4 README

Overview

The Game Porting Toolkit helps game developers create an environment for evaluating their existing Windows games right on Apple Silicon Macs running macOS 14. Using the included binaries and installation and configuration instructions, you can run your game to get a sense of how it can feel and play right away, even before you begin your porting journey.

Requirements

- The Game Porting Toolkit currently only runs on Apple Silicon Macs running macOS 14 Sonoma Beta.
- Translated games require more resources, so developer-focused Macs with 16GB of RAM or more are recommended.
- The provided macOS graphics bridge libraries must be configured with a custom version of the Wine translation environment in order to create your game evaluation environment.
- Instructions and scripts for building and configuring the custom version of Wine are included here.

Installation and Setup

1. Setup your development and Homebrew environment

- Ensure the latest Command Line Tools for Xcode 15 beta are installed. Visit https://developer.apple.com/downloads to download these tools.

- Open Terminal.

- The Game Porting Toolkit runs under Rosetta 2. Ensure that Rosetta 2 is installed.

```
softwareupdate --install-rosetta
```

- Enter an x86_64 shell to continue the following steps in a Rosetta environment. All subsequent commands should be run within this shell.

```
arch -x86_64 zsh
```

- Install the x86_64 version of Homebrew if you don't already have it.

```
/bin/bash -c "$(curl -fsSL https://
raw.githubusercontent.com/Homebrew/install/HEAD/
install.sh)"
```

- Make sure the brew command is on your path:

```
which brew
```

If this command does not print `/usr/local/bin/brew`, you must either modify your PATH to put `/usr/local/bin` first, or fully specify the path to `brew` in the subsequent commands.

- Tap the Apple Homebrew tap, which can be found at `https://github.com/apple`:

```
brew tap apple/apple http://github.com/apple/homebrew-apple
```

- Install the `game-porting-toolkit` formula. This formula downloads and compiles several large software projects. How long this takes will depend on the speed of your computer.

```
brew -v install apple/apple/game-porting-toolkit
```

- If during installation you see an error such as "`Error: game-porting-toolkit: unknown or unsupported macOS version: :dunno`", your version of Homebrew doesn't have macOS Sonoma support. Update to the latest version of Homebrew and try again.

```
brew update
brew -v install apple/apple/game-porting-toolkit
```

2. Create a new Wine prefix for your Game Porting Toolkit environment

A Wine prefix contains a virtual C: drive. You will install the toolkit and your game into this virtual C: drive.

- Run the following command to create a new Wine prefix named **my-game-prefix** in your home directory.

```
WINEPREFIX=~/my-game-prefix `brew --prefix game-porting-toolkit`/bin/wine64 winecfg
```

A "Wine configuration" window should appear on your screen.

- Change the version of Windows to Windows 10.
- Choose Apply and then OK to exit winecfg.

If the "Wine configuration" window does not appear, and no new icon appears in the Dock, verify that you have correctly installed the x86_64 version of Homebrew as well as the `game-porting-toolkit` formula.

3. Install the redistributables from the toolkit into the Wine prefix

The graphics bridge libraries need to be placed inside your Wine prefix in order to

finalize your game evaluation environment. These instructions assume you have mounted the Game Porting Toolkit at `/Volumes/Game Porting Toolkit-1.0`.

- Copy the Game Porting Toolkit library directory into Wine's library directory.

```
ditto /Volumes/Game\ Porting\ Toolkit-1.0/redist/lib/
`brew --prefix game-porting-toolkit`/lib/
```

Launch your game

Open your Wine prefix's virtual C: drive in Finder (open `~/my-game-prefix/drive_c`) and copy your game into an appropriate subdirectory.

The provided `bin/gameportingtoolkit*` scripts can be copied onto your path to facilitate different forms of logging and launching. You can run these scripts from any shell; you don't need to switch to the Rosetta environment first.

A. Standard launching:

```
gameportingtoolkit ~/my-game-prefix 'C:\Program
Files\MyGame\MyGame.exe'
```

This launches the given Windows game binary with a visible extended Metal Performance HUD and filters logging to output from the Game Porting Toolkit.

B. Launching without a HUD

```
gameportingtoolkit-no-hud ~/my-game-prefix 'C:\Program
Files\MyGame\MyGame.exe'
```

Launches your game without the extended Metal Performance HUD visible.

C. Launching with Wine ESYNC disabled

```
gameportingtoolkit-no-esync ~/my-game-prefix 'C:\Program
Files\MyGame\MyGame.exe'
```

Finally, the no-esync versions of the script disable Wine's ESYNC option, a common compatibility flag. If your game experiences issues with multithreading, or you get an error message about running out of files, you can try launching your game without this Wine environment variable enabled to see if disabling esync clears the problem.

Logging

Logging output will appear in the Terminal window in which you launch your game as well as the system log, which can be viewed with the Console app found in Applications ▸ Utilities. Log messages from the Game Porting Toolkit are prefixed with **D3DM**. By

default the `gameportingtoolkit*` scripts will filter to just the **D3DM**-prefixed messages.

Troubleshooting

My game won't run and crashes with an invalid instruction

Invalid instruction crashes are often (but not always) caused when Rosetta 2 is unable to translate AVX/AVX2 instructions. You may be able to recompile a version of your game without AVX/AVX2 instructions in order to evaluate its potential on Apple Silicon with the Game Porting Toolkit when you hit this error. When porting your code natively to Apple Silicon, NEON instructions are a high-performance replacement for AVX/AVX2.

My game won't run because its anti-cheat or DRM software is incompatible with Wine translation.

You may be able to rebuild a custom version of your game in your Windows development environment with anti-cheat or DRM disabled for your own evaluation purposes. When porting your code natively to Apple Silicon and macOS, contact your anti-cheat or DRM provider—most have native Apple Silicon solutions for your native build.

My game won’t run because it thinks the version of Windows is too old.

First, make sure you have selected an appropriate Windows version in winecfg. This affects the major and minor Windows versions that are reported to your game.

If your game checks for a specific build version, you can alter this value by changing the `CurrentBuild` and `CurrentBuildNumber` values of the `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT` registry key. You must perform this step *after* selecting a Windows version in winecfg. Run the following commands, replacing «BUILD_NUMBER» with the specific build number your game checks for:

```
WINEPREFIX=~/my-game-prefix `brew --prefix game-porting-
toolkit`/bin/wine64 reg add
'HKEY_LOCAL_MACHINE\Software\Microsoft\Windows
NT\CurrentVersion' /v CurrentBuild /t REG_SZ /d
«BUILD_NUMBER» /f
WINEPREFIX=~/my-game-prefix `brew --prefix game-porting-
toolkit`/bin/wine64 reg add
'HKEY_LOCAL_MACHINE\Software\Microsoft\Windows
NT\CurrentVersion' /v CurrentBuildNumber /t REG_SZ /d
«BUILD_NUMBER» /f
WINEPREFIX=~/my-game-prefix `brew --prefix game-porting-
```

```
toolkit`/bin/wineserver -k
```

The last command will shut down the virtual Windows environment to ensure that all components agree on the Windows version the next time you launch your game.

My game won't run because it requires Mono, .NET, or the MSVCRT runtime.

The game porting toolkit's evaluation environment does not pre-install these runtime support packages. If your game makes use of one of these packages, consider searching for and downloading appropriate installers (.exe or .msi) and installing them to your evaluation environment. Additional runtime installers can be run on your environment by just launching the installer and following its installation instructions:

```
WINEPREFIX=~/my-game-prefix `brew --prefix game-porting-
toolkit`/bin/wine64 <some-installer.exe>
```

And .MSI packages can be installed by launching the Windows **uninstaller** application and choosing to install a downloaded .msi package:

```
WINEPREFIX=~/my-game-prefix `brew --prefix game-porting-
toolkit`/bin/wine64 uninstaller
```

My game won't boot anymore even though I made no changes.

If the game stopped booting without being updated, you can try clearing the shader cache.
Run the following commands:

```
cd $(getconf DARWIN_USER_CACHE_DIR)/d3dm
cd «GAME_NAME»
rm -r shaders.cache
```

Do you have a different problem or other feedback?

Please let us know through https://feedbackassistant.apple.com.