Platforms to Run Harmony Development Kit on

The current page is created to define "sure-to-work" configurations. This information is essential for figuring out which platforms are reliable and easy to use. Please, taking into consideration your experience, feel free to add comments whether this or that platform works or not and how: stable, or unstable, or buggy etc.

HDK is known to run on the following platforms:

Windows

PLATFORM	COMMENTS
Windows XP x86	
Windows Server 2003 on x86 and x86_64 platforms	
Windows Vista on x86 and x86_64 platforms	
Windows XP with VS.NET 2005 Community Edition	

Linux

PLATFORM	COMMENTS
Ubuntu 6 x86	
Ubuntu 5 x86	
Ubuntu on IA32 and IA64 platforms	
RHEL version 2.6.9.	
RHEL on IA32 and IA64 platforms	
Fedora Core 5 x86	
FC4/5	
FC on IA32 and IA64 platforms	
SUSE version 9, 10, 11 on x86, x86_64 and IPF (aka ia64)	Building comments using SLE v.11
Debian 3.1 x86	
Gentoo with gcc 4.1.1 on x86 and x86_64	

Discussion Issues

Issue 1

Should we start working on porting harmony over to macosx at least on intel chips? The possibility is that pretty much nobody will care about powerpc anymore in the nearest future.

Hi guys,

I hope you don't take this amiss but after seeing this remark I have to ask - are you all stupid ?

There is no need for an aftermarket Java on Windows XP. There is no need for an aftermarket Java on Mac OS X on Intel.

There *is* a crying need for an aftermarket Java with current features on HP-UX, on AIX, on OS/2, on BeOS, on Irix, on Mac OS 9, on RISCOS and so on. Maybe even on DOS.

Why the hell do you insist on reinventing the wheel when there is a crying need elsewhere ? Is it ego or stupidity or what ?

I'm just curious because I see this behavior in so many "open source" projects.

Issue 2

The work with 100% free toolchain on Windows (Windows XP with VS.NET 2005 Community Edition) requires a lot of effort to make even classlib work with IBM VME, not to mention compiling DRLVM on it. The problem is in the Microsoft secure API initiative, DLL manifests and stuff like that in VS.NET 2005.

Issue 3

DRLVM has a limitation today: we can run only on PC with at least SSE support, moreover JIT works much better (more stable, faster) with SSE2. This can be an advanced task for JIT gurus to complete available x87 support, but before that we can't claim that we officially support hardware without SSE2.

Issue 4

Before we decide whether to support this or that platform or not, we should define a set of tests that must pass on that platform after each commit; we do roll back if they fail.

Proposal 1: To define support as passing >90% of classlib unit and smoke/c-unit/kernel in DRLVM.

Lowlights: Logically there could be a platform that we don't know, but that platform could pass 99% of the tests. In this case theoretically we support a platform we don't have any idea about. Such a situation is possible with some Linux clones.

Proposal 2: Fix "Harmony-broken" platforms, if somebody really needs them. We can create a "critical platforms list" containing "Harmony-broken" platforms, which should be *small enough* (<7).

Proposal 3: If we want a *large enough* list with "all good platforms we want to work on", let's make the list just before each release. It would be much easier and would not require us to maintain lists of platforms here and there.

Solution: We can easily have three categories:

a) platforms that we certify as being compatible, and support

Policy: We do our best not to break it with commits. "Do our best" to be defined later. If a commit breaks that platform, we stop further commits and either fix or roll it back ASAP.

b) platforms in-progress that we certify as being compatible, but don't make any support promises (Alexei Zakharov has generously volunteered to run tests on and report issues for Windows 2000 periodically).

c) platforms that we do not support