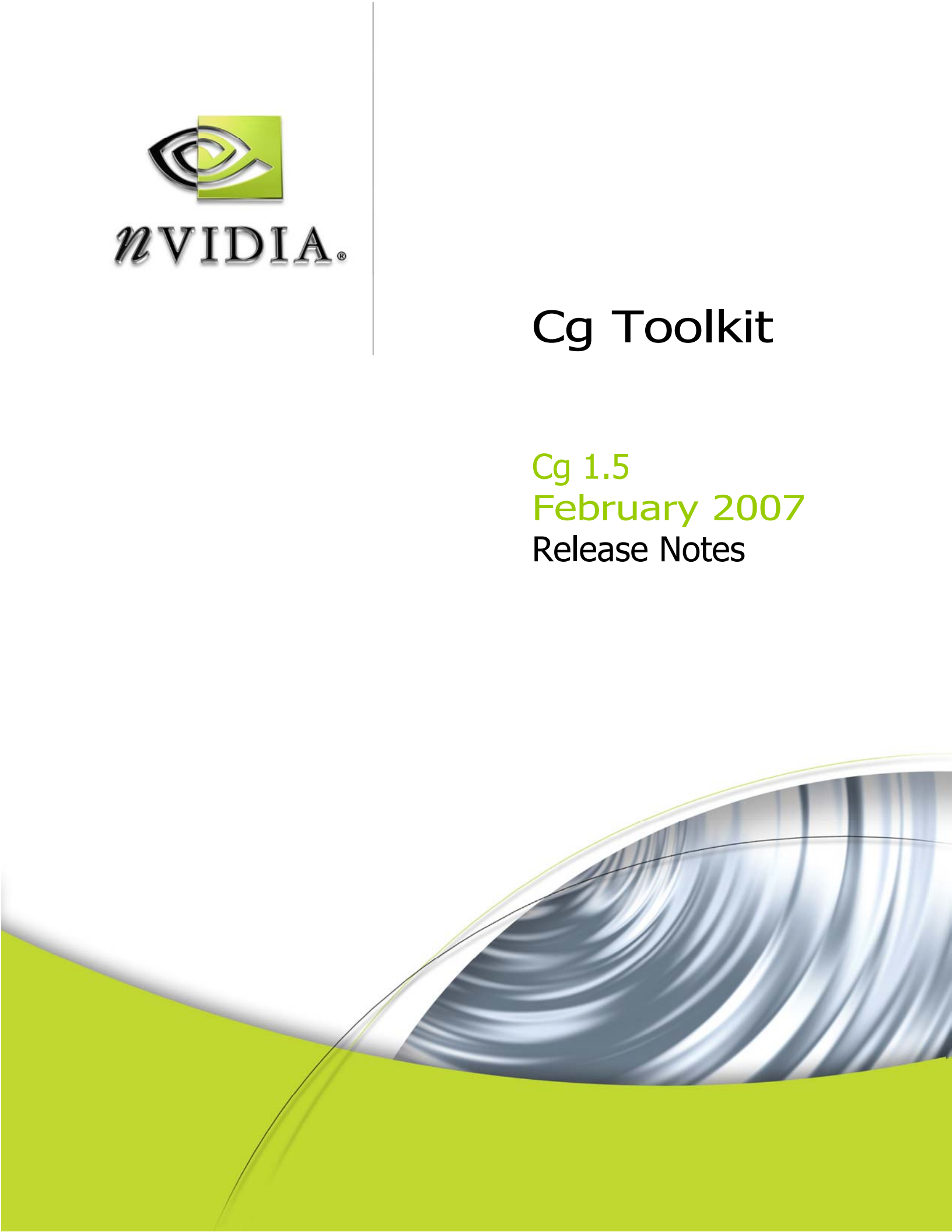

Cg Toolkit

Cg 1.5
February 2007
Release Notes

Cg Toolkit Release Notes

The Cg Toolkit allows developers to write and run Cg programs using a wide variety of hardware platforms and graphics APIs.

Originally released in December 2002, the Toolkit now supports over 20 different DirectX and OpenGL profile targets. It provides a compiler for the Cg language, runtime libraries for use with the OpenGL and DirectX graphics APIs, support for the CgFX effect files, example applications, and extensive documentation.

The Cg 1.5 February 2007 release contains these changes to the previous SDK:

- Library improvements:
  - CgFX support works for GLSL profiles now
  - Fixed a semantic aliasing problem on OSX
  - Reduced per program memory usage
  - Performance improvements
- Updated examples:
  - OpenGL/advanced/cgfx_interfaces
  - OpenGL/advanced/cgfx_bumpdemo
    - Now includes GLSL techniques
  - OpenGL/basic/15_particle_system
  - OpenGL/basic/16_keyframe_interpolation
- Updated documentation:
  - Whitepaper on bumpdemo example
    - docs/Cg_bumpdemo_Tutorial.pdf
  - Whitepaper on CgFX version of bumpdemo example
    - docs/CgFX_bumpdemo_Tutorial.pdf
- Supported platforms:
  - OSX Panther is no longer supported

The Cg 1.5 September 2006 release incorporated these updates:

- New OpenGL profiles for the OpenGL Shading Language (GLSL)
- New Direct3D profiles for Shader Model 3.0
- New API for programmatic effect generation
- New API for combining programs from multiple domains
- The Runtime is now multithread safe

- The CgFX API works for Direct3D
- Universal binaries for Mac OS X 10.4 (Tiger) with support for both PPC and x86
- Support for Solaris10 on x86
- New OpenGL and Direct3D9 examples including:
  - examples/OpenGL_Basic: 15 complete ANSI C OpenGL-based Cg examples from *The Cg Tutorial*.
  - examples/OpenGL_Advanced: 2 complete OpenGL-based Cg examples.
    - cgfx_bumpdemo: Demonstrates using CgFX files for loading and applying effects.
    - cgfx_procfx: Demonstrates procedurally building the same effect in the cgfx_bumpdemo example.
  - examples/Direct3D9_Basic: 7 complete Direct3D9-based Cg examples from *The Cg Tutorial* (matching the first 7 examples from OpenGL_Basic) written in C++. Requires the Microsoft DirectX SDK to build these.
  - The older Cg examples were moved to examples/old.
- Examples include Visual C 6, Visual Studio .NET 2003, and Visual Studio 2005 projects for Windows users. Makefiles are provided for non-Windows platforms
- Expanded documentation
  - Cg Standard Library documentation added
  - Cg Runtime documentation (Core, GL, and D3D9) has been improved
  - CgFX State documentation has been added
- Many bugs have been fixed

With Cg 1.5, the Direct3D-specific Cg runtime libraries support shared parameters as well as a native D3D state manager for CgFX. This release also provides Direct3D profiles for Shader Model 3.0.

With minor exceptions, Cg 1.4, Cg 1.3 and Cg 1.2 applications should work with Cg 1.5 without the need to recompile the program. See "Compatibility Notes," below, for more information.

Applications and effects that used previous versions of CgFX will have to be modified to use the new CgFX API. If the supplied OpenGL state manager for CgFX is used, the effect files themselves must be modified. The effects must be changed to use OpenGL state assignments, rather than the D3D state assignments supported in prior versions of CgFX.

Cg is available for a wide variety of hardware and OS platforms (as mentioned above). Please visit the NVIDIA Cg website at developer.nvidia.com/Cg for complete availability and compatibility information.

Please report any bugs, issues, and feedback to NVIDIA by email at cgsupport@nvidia.com. We will expeditiously address any reported problems.

Compatibility Notes

Although the 1.5 release of Cg is generally compatible with previous releases, several improvements and other changes may affect existing applications. This section details these potential compatibility issues.

In the 1.2 and 1.3 releases of Cg, the Cg compiler would always generate generic attribute names (e.g., "ATTR0") for varying input parameters, even if conventional semantic names (e.g., "POSITION") were specified under OpenGL profiles. Since release 1.4.1, the type of attribute now matches the type of semantic. As a result, applications that explicitly assumed and checked for generic attribute names may need to be modified to handle both generic and conventional attributes.

In this release, the Cg compiler uses an improved, more efficient constant register allocation scheme. As a result, uniform parameters in compiled programs may no longer reside in the same constant registers as they had in previous releases of Cg. Applications that (incorrectly) assume a particular constant register numbering or order should be modified to either remove this assumption, or to use register allocation semantics (e.g., "register(C0)") to implement the layout assumed by the application.

In Cg 1.4, CgFX was redesigned and re-implemented. Existing applications that used the previous CgFX API must be modified to use the new API. In addition, if you wish to use the supplied "native" OpenGL state manager, which assumes that OpenGL-style state names and values are used in effects, existing effects must be modified to use OpenGL-style states. See the "Introduction to CgFX" chapter in the Cg Users Manual for more details. If you have questions or problems related to porting existing CgFX applications, please contact NVIDIA developer relations.

Supported Profiles

The Cg compiler currently supports the following hardware profiles:

OpenGL

- **glslv** OpenGL Shading Language (GLSL) for OpenGL 2.0 vertex shader
- **glslf** OpenGL Shading Language (GLSL) for OpenGL 2.0 fragment shader
- **arbvp1** ARB_vertex_program 1.0
- **arbfp1** ARB_fragment_program 1.0
- **vp40** ARB_vertex_program + NV_vertex_program2 option
- **fp40** ARB_fragment_program + NV_fragment_program2 option
- **vp30** NV_vertex_program 2.0
- **fp30** NV_fragment_program 1.0
- **vp20** NV_vertex_program 1.0

- **fp20** NV_register_combiners and NV_texture_shader

DirectX 8 & 9

- **vs_1_1** Vertex Shader 1.1
- **ps_1_1** Pixel Shader 1.1
- **ps_1_2** Pixel Shader 1.2
- **ps_1_3** Pixel Shader 1.3

DirectX 9

- **vs_2_0** Vertex Shader 2.0
- **vs_2_x** Extended VS 2.0
- **ps_2_0** Pixel Shader PS 2.0
- **ps_2_x** Extended PS 2.0

DirectX 9.0c

- **vs_3_0** Vertex Shader Model 3.0
- **ps_3_0** Pixel Shader Model 3.0

Supported OS/Hardware Platforms

Cg is available for these platforms:

- Windows 32
- Windows 64
- Linux x86
- Linux x86-64
- MacOS 10.4 (Tiger) [universal binary with PPC and i386]
- Solaris 10 x86

The Cg Runtime libraries include:

- The Cg core runtime library for managing parameters and loading programs
- The CgGL runtime library for OpenGL based applications
- The CgD3D8 runtime library for DirectX 8 based applications
- The CgD3D9 runtime library for DirectX 9 based applications
- Since Cg 1.4, CgFX is incorporated directly into the Cg Runtime libraries. With Cg 1.5, CgFX supports Direct3D as well as OpenGL.

Improvements & Bug Fixes

Improvements

- Many changes to the infrastructure in both the compiler and runtime have been made in order to improve both performance and reliability
- The Runtime is now multithread safe
- Cg header files explicitly declare their use of the cdecl calling convention
- Semantics were added to map uniforms to program.env

Improvement: CgFX

- The new implementation of CgFX (from Cg 1.4) now supports D3D
- CgFX now supports D3D-style state names

Improvement: Cg Runtime efficiency

The Cg Runtime has been tuned to remove overhead from the runtime calls and avoid redundant 3D API calls when possible.

Improvement: Cg compile times

Many Cg programs now compile faster than in previous releases. This is especially true of long shaders, and shaders that make heavy use of Cg's "interfaces" feature.

Improvement: Windows Installer

- The Cg 1.5 installer prompts to install Microsoft DirectX 9 end-user runtime if it's not found installed on the system (based on registry queries). Cg 1.5 installer also prompts to install Microsoft DirectX SDK if DXSDK_DIR environment variable (typically set by the DirectX SDK installer) is not set.
- Cg 1.5 for Windows installs a more complete set of example code.

Improvement: Mac OSX

- Cg is now a universal binary which supports PowerPC and i386
- Cg installs in /Library/Frameworks instead of /System/Library/Frameworks
- The Cg Framework uses @executable_path/../Frameworks as it's installation path

Improvement: Documentation

- Note: The Cg Users Manual has not yet been updated for Cg 1.5.
- Cg 1.5 for Windows fully indexes the Cg Microsoft Compiled Help File (CgReferenceManual.chm).
- Cg 1.5 adds Cg Standard Library documentation (incomplete).
- Cg 1.5 begins to add standard CgFX state documentation (incomplete).

- Cg 1.5 adds D3D8 stub documentation.
- The documentation for the Core, GL, and D3D9 Runtimes has been improved
- The Cg Users Manual in PDF format is included in the install at `docs/CgUsersManual.pdf` or from the Windows Start menu (*Start->All Programs->NVIDIA Corporation->Cg Toolkit->User's Manual*).
- Reference manual pages for the Cg runtime API, profiles, and Cg standard library are now included in HTML and PDF form. The HTML pages are in the `docs/html` directory. `CgReferenceManual.pdf` is in the docs directory or you can select the PDF icon from the Windows Start menu (*Start->All Programs->NVIDIA Corporation->Cg Toolkit->Reference Manual).*
- The Microsoft HTML Help file is updated to include profile and standard library documentation. From the Windows Start menu (*Start->All Programs->NVIDIA Corporation->Cg Toolkit->Reference Manual*) choose the HTML Help icon.

Removed features

- We have dropped support for the previous CgFX API
- We no longer provide the `cgD3D8d.dll` and `cgD3D9d.dll` debug libraries

Bug Fixes

- All known memory leaks in CgFX and the Cg Runtime have been fixed
- The CG_OBJECT option to cgCreateProgram and cgCreateProgramFromFile has been fixed to allowing pre-compiled programs for less compiler overhead

Known issues

Known runtime issues

- cgCopyProgram does not work.
- The DirectX 8 runtime has not been updated to support the Cg 1.2 'interfaces' feature.
- The Cg runtime does not currently support created shared parameters containing varying members.
- Unsized arrays and interface parameters cannot currently be used on the right-hand side of state assignments. Doing so will trigger an error.
- Values set by cgGLSetOptimalOptions(...) can be un-set after a call to cgDestroyContext( ). As a work around, call cgGLSetOptimalOptions( ) after each call to cgDestroyContext( ) when more Cg contexts are going to be created.
- Error reporting should be improved in many cases.
- More OpenGL state needs to be exposed through CgFX state assignments. If you have specific feedback about any missing state, please speak to your Developer Support contact.

Known compiler issues

- Long shader programs that make heavy use of interfaces may still see very long compiler times.
- Very little error checking is performed on the OpenGL state semantics string (`state.*`); it is just copied to the output assembly. As a result, a typo in the string may compile correctly, and no error will be apparent until the application attempts to load the assembly shader.
- Error reporting: Some error and warning messages are not as clear as they could be. Some of the issues to be aware of are:
  - Reported line numbers do not match source code lines when standard library functions are being used
  - In some cases, errors are not reported in the order they appear in the program
  - Errors are not reported when constants are out of range for untyped constants.
- Side-effects in conditional expressions ('?:') and logical expressions ('&&' and '||') are always evaluated, regardless of the condition, and currently warnings are not always issued. Hence developers need to watch out for this case.
- Only one return statement is allowed per function. There is an error issued if there is any unreachable code. Return statements in if/for blocks are not supported.
- At most one binding semantic per uniform variable is supported by the compiler. Multiple profile-specific binding semantics per uniform variable are not supported.
- Only loops with single induction variables are unrolled. Loops that require more than 1 induction variable will fail to compile.
- Local variable arrays which are written to in one block of code, and then read via a non-constant index in a different block will cause the compiler to crash. Current hardware does not support this feature, but the compiler should not crash.
- Invalid Cg programs can, at times, generate invalid code, instead of a compiler error.

Known profile-specific issues

- The ps2* profiles do not yet support MRTs
- Because the underlying hardware support for the fp20 and ps_1_* profiles is quite limited and inflexible, it isn't always possible to compile even seemingly simple Cg programs under these profiles. For more details on these limitations, please see the NV_register_combiners and NV_texture_shader OpenGL extension specifications, or the DirectX PixelShader 1.* specifications.
- The FOG varying input semantic is not yet supported under the fp20 profile.

New API

Cg 1.5 includes new API for programmatic effect generation and for combining programs from multiple domains. The complete list of new routines appears below.

cgCombinePrograms
cgCombinePrograms2
cgCombinePrograms3
cgCreateEffectAnnotation
cgCreateEffectParameter
cgCreateEffectParameterArray
cgCreateEffectParameterMultiDimArray
cgCreateParameterAnnotation
cgCreatePass
cgCreatePassAnnotation
cgCreateProgramAnnotation
cgCreateSamplerStateAssignment
cgCreateStateAssignment
cgCreateStateAssignmentIndex
cgCreateTechnique
cgCreateTechniqueAnnotation
cgGetBoolAnnotationValues
cgGetEffectName
cgGetFirstEffectAnnotation
cgGetMatrixSize
cgGetNamedEffect
cgGetNamedEffectAnnotation
cgGetNamedSubParameter
cgGetNumProgramDomains
cgGetParameterEffect
cgGetProfileDomain
cgGetProgramDomainProfile
cgGetStateContext
cgGetStateEnumerantName
cgGetStateEnumerantValue
cgGetTypeBase
cgGetTypeClass
cgGetTypeSizes
cgSetBoolAnnotation
cgSetBoolArrayStateAssignment
cgSetBoolStateAssignment
cgSetEffectName
cgSetFloatAnnotation
cgSetFloatArrayStateAssignment
cgSetFloatStateAssignment
cgSetIntAnnotation
cgSetIntArrayStateAssignment
cgSetIntStateAssignment
cgSetProgramStateAssignment
cgSetSamplerStateAssignment
cgSetStringAnnotation
cgSetStringStateAssignment
cgSetTextureStateAssignment
cgGLDisableProgramProfiles
cgGLEnableProgramProfiles
cgGLSetDebugMode
cgD3D9GetManageTextureParameters
cgD3D9GetTextureParameter
cgD3D9IsProfileSupported
cgD3D9RegisterStates
cgD3D9SetManageTextureParameter
cgD3D9SetTextureParameter
cgD3D9UnloadAllPrograms

Release Types

Cg 1.5 is released in two forms:

1. The Cg Toolkit provides a complete Cg Software Development Kit (SDK) including documentation, examples, standalone compiler, headers and libraries.
2. Cg Binary Distributions provide updated redistributable libraries that Cg-based applications can ship with.

Beta versions were released as platform specific installers containing the full toolkit (libraries, documentation, examples, etc.) They can be downloaded from

http://developer.nvidia.com/object/cg_toolkit.html

Binary distributions contain only the libraries, and have all supported platforms bundled in a single file. The libraries supplied in a binary distribution should be feature-for-feature and bug-for-bug compatible across all the platforms supported by a given distribution (meaning are all compiled from the same source code). Cross-platform software vendors are encouraged to redistribute Cg libraries from a single binary distribution to minimize platform variances in Cg.

Cg binary distributions can be found at

http://developer.nvidia.com/object/cg-redistributable-binaries.html

Distribution License

The docs directory contains a file Cg_Redist_License.pdf providing a non-exclusive, world-wide, royalty free licensee for redistributing Cg with your applications. See this license for details.

Release History

The following table summarizes release dates and library versions for Cg 1.5 releases:

Cg Release Name	Release Date	Library Version
binary6	02/02/07	1.5.0018
binary5	12/01/06	1.5.0015
SDK1	09/19/06	1.5.0014
binary3	08/22/06	1.5.0012
binary2	07/17/06	1.5.0011
binary1	07/07/06	1.5.0010
beta2	05/23/06	1.5.0008
beta1	04/03/06	1.5.0006

The Cg library version is returned by cgGetString(CG_VERSION)

Change History

1.5.0018

- Fixed CgFX support for GLSL profiles.
- Fixed a semantic aliasing problem on OSX.

1.5.0015

- Performance improvements.

NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050
www.nvidia.com