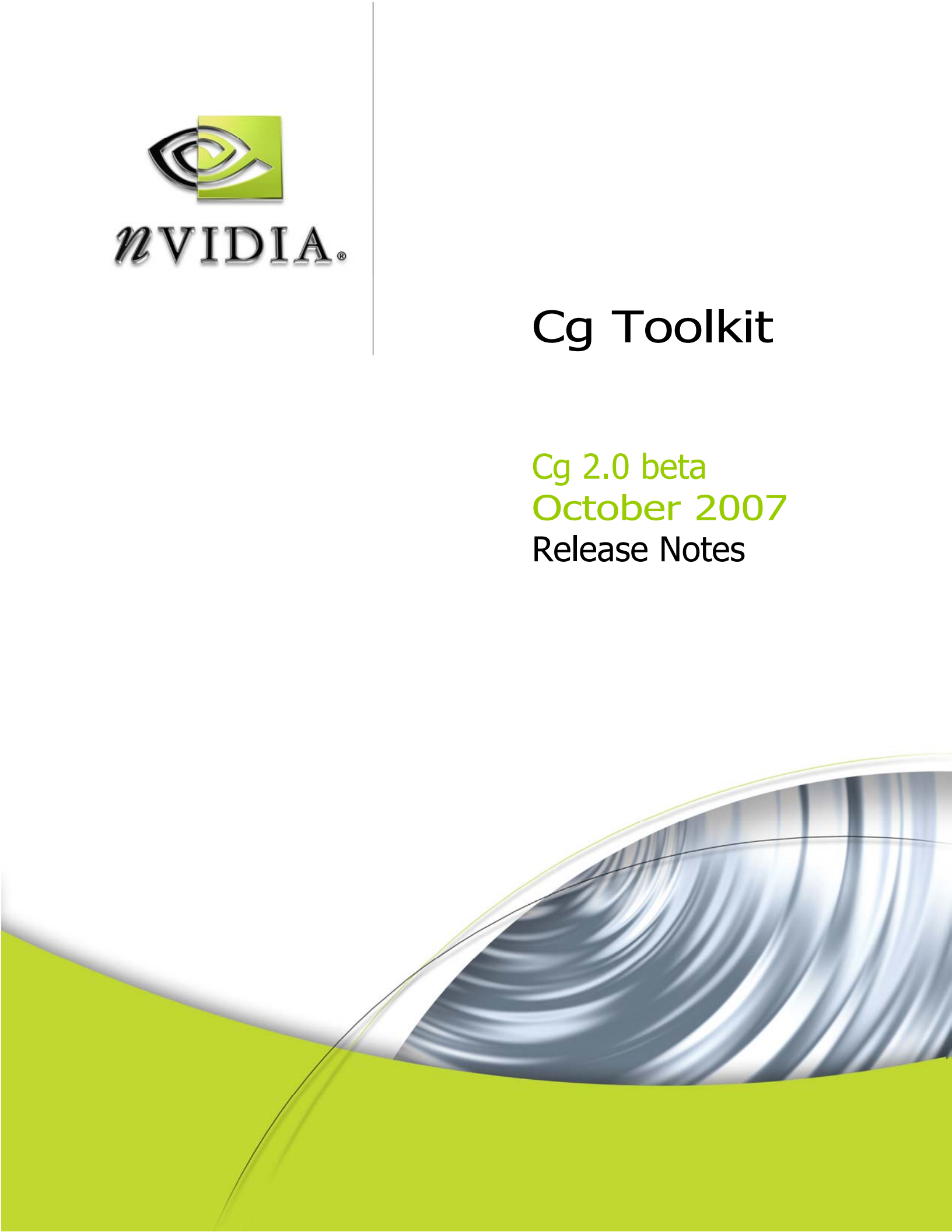

Cg Toolkit

Cg 2.0 beta
October 2007
Release Notes

Cg Toolkit Release Notes

The Cg Toolkit allows developers to write and run Cg programs using a wide variety of hardware and OS 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 2.0 beta release of Cg incorporates the following updates:

- New OpenGL GPU Program4 profiles
- New Direct3D9 HLSL translation profiles
- Support for geometry programs
- Support for constant buffers
- Support for texture arrays
- Support for true integer data types
- Performance improvements and bug fixes
- New OpenGL examples including:
  - examples/OpenGL/basic (from *The Cg Tutorial*)
    - 21_bump_map_wall:
  - examples/OpenGL/advanced
    - buffer_lighting:
    - cgfx_boxfilter:
    - cgfx_texture_array:
    - gs_interp_quad:
    - gs_md2render:
    - gs_quadnormal:
    - gs_shrinky:
    - gs_simple:
    - interpolation_modifier:
    - texture_array:
    - vertex_texture:
- Expanded documentation including a new version of the Language Specification

Cg 1.5 applications should work with Cg 2.0 without the need to recompile the program. See "Compatibility Notes," below, for more information.

Please visit the NVIDIA Cg website at developer.nvidia.com/page/cg_main.html for complete availability and compatibility information.

Please send bug reports, issues, and feedback to cgsupport@nvidia.com.

Compatibility Notes

There aren't any known compatibility issues with programs written against Cg 1.5. For programs written against Cg 1.4 or earlier, refer to the Compatibility Notes section of the release notes for Cg 1.5.

Supported OS/Hardware Platforms

Cg is available for these platforms:

- Windows 32
- Windows 64
- Linux x86
- Linux x86-64
- MacOS 10.4 (Tiger)
- MacOS 10.5 (Leopard)
- 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

Supported Profiles

The Cg compiler currently supports the following hardware profiles:

OpenGL

- **gpu_gp** NV_geomemtry_program4
- **gpu_vp** NV_vertex_program4
- **gpu_fp** NV_fragment_program4
- **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

- **hlslv** HLSL9 Vertex Shader
- **hlslf** HLSL9 Fragment Shader
- **vs_3_0** Vertex Shader Model 3.0
- **ps_3_0** Pixel Shader Model 3.0

Improvements & Bug Fixes

Improvements

- Single threaded programs can now disable the thread locking introduced in Cg 1.5.
- The new examples are now packaged in the non-Windows installers.

Improvement: CgFX

- Increased compatibility with HLSL FX files.

Improvement: Documentation

- **Note**: The Cg Users Manual has **not** been updated for Cg 2.0.
- New version of the Cg Language Specification (this is a work in progress)
- New reference pages for the new Runtime features
- Updated Standard Library documentation and Profile reference pages

Bug Fixes

- Numerous bugs have been fixed.

Known issues

Known runtime issues

- **cgCopyProgram** and **cgCopyEffect** do not work.
- Loading precompiled code via **CG_OBJECT** in **cgCreateProgramFromFile** doesn't work for shaders which use semantic type modifiers.
- The DirectX 8 runtime does not support Cg interfaces.
- The Cg runtime does not support creating 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.

-

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, as specified in the Cg language specification. Hence developers need to watch out for this case.
- 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 a single induction variable are unrolled. Loops that require more than 1 induction variable will fail to compile on older profiles that do not support loops.
- 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 fail to compile on older hardware that does not support this feature. Current hardware supports this feature.
- Invalid Cg programs can, at times, generate invalid code, instead of a compiler error.

Known profile-specific issues

- The ps2* profiles do not 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 2.0 adds new API for features like constant buffers and geometry programs. The complete list of new routines appears below.

cgCopyEffect

cgCreateBuffer

cgCreateObj

cgCreateObjFromFile

cgDestroyBuffer

cgDestroyObj

cgGLCreateBuffer

cgGLGetBufferObject

cgGetBufferSize

cgGetConnectedStateAssignmentParameter

cgGetLockingPolicy

cgGetParameterBufferIndex

cgGetParameterBufferOffset

cgGetParameterResourceSize

cgGetParameterResourceType

cgGetParameterSettingMode

cgGetProgramBuffer

cgGetProgramInput

cgGetProgramOutput

cgGetSemanticCasePolicy

cgGetStringAnnotationValues

cgMapBuffer

cgSetBufferData

cgSetBufferSubData

cgSetLockingPolicy

cgSetParameterSettingMode

cgSetProgramBuffer

cgSetSemanticCasePolicy

cgUnmapBuffer

cgUpdateProgramParameters

Release Types

Cg 2.0 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 all supported platforms are 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 releases:

Cg Release Name	Release Date	Library Version
SDK3	09/26/07	1.5.0023
SDK2	08/16/07	1.5.0022
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

1.5.0023

- Bug fixes.

1.5.0022

- Performance improvements, especially for CgFX.
- Fixed CgFX support for DepthBounds and DepthBoundsEnable.

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