



NVIDIA®

State of the Art
Cross Platform Shader
Development

Agenda



- **FX Composer 2.0 Overview**
- **Cross-Platform Shader Authoring**
- **Production Pipeline Integration**
- **Conclusion**
- **Q&A**

FX Composer 2.0

What and Who?

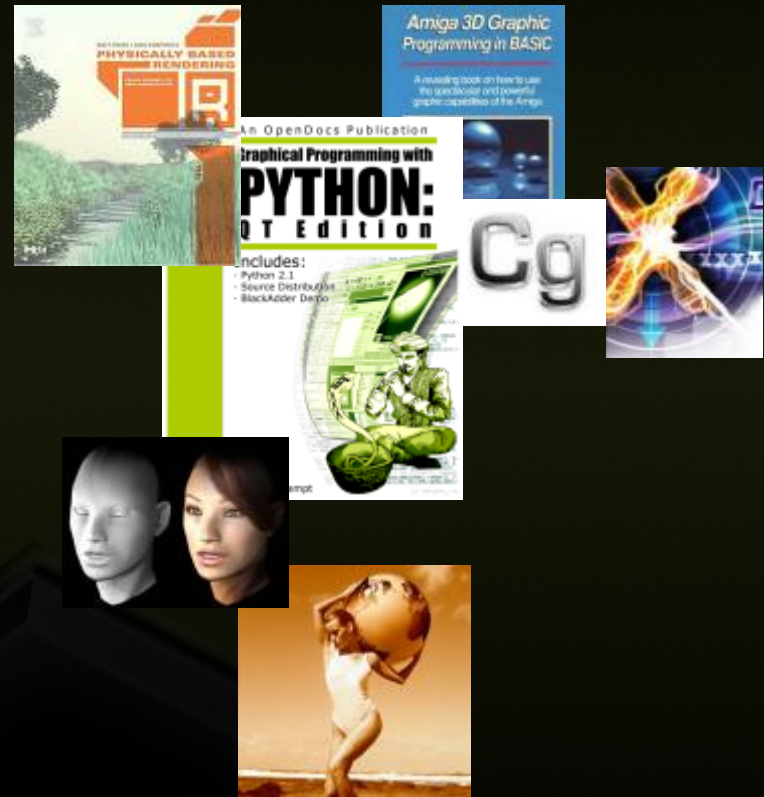


● What is it for?

- Shader Authoring IDE
- Debugging and Profiling
- Scene Integration
- Asset Management

● Who is it for?

- Graphics Programmers
- Technical Directors
- Technical Artists
- Artists



Your Requirements

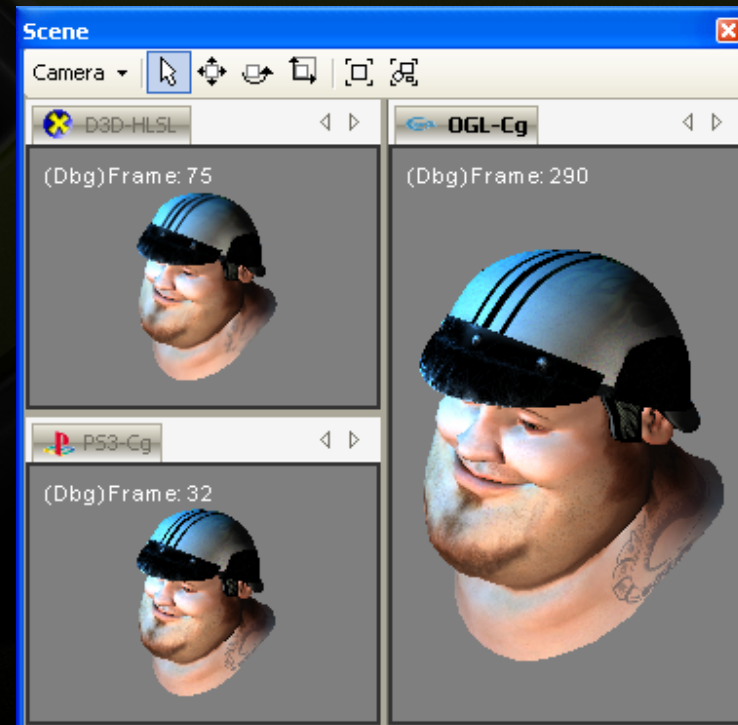


- **Handling of complex rendering**
- **Highly customizable layout management**
- **Powerful user interface**
- **Shader performance profiling**
- **Plug-in based architecture**
- **Scriptable**

Flexible Render



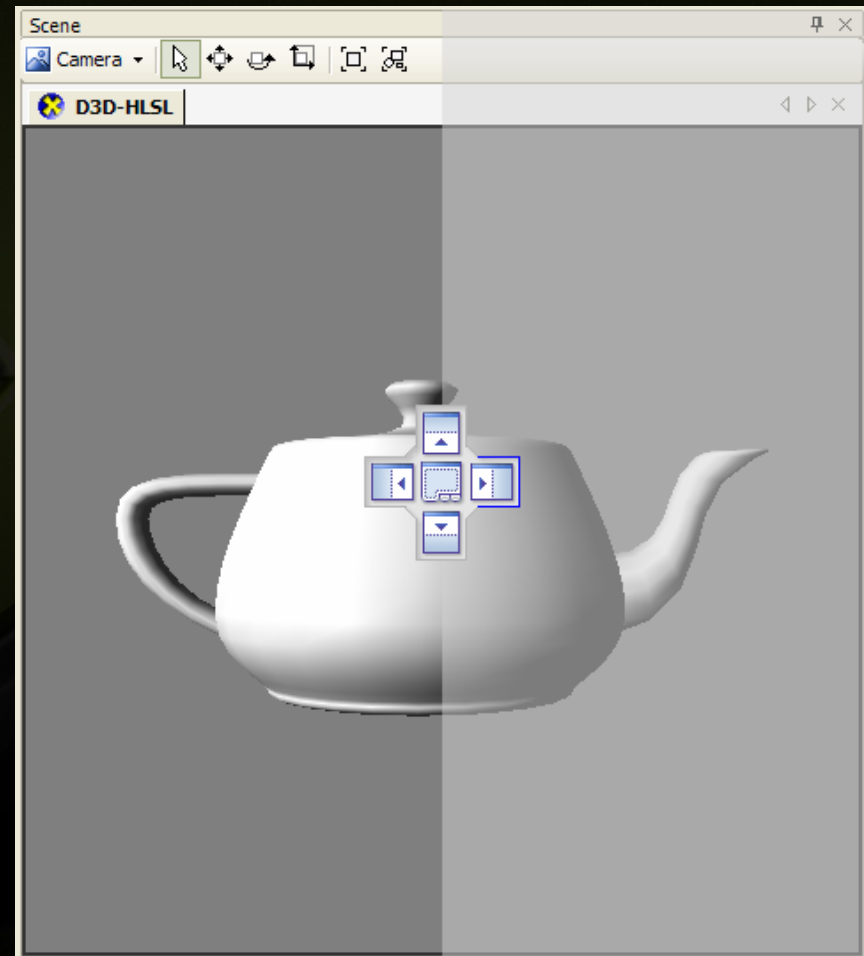
- Many API & shading language combinations
- Surface and fullscreen shaders



Improved User Interface



- Customized user layouts
 - Show or hide panels
 - VC2005 docking style
 - Save/Load layouts



Improved User Interface



The screenshot displays a complex software interface for a game engine. At the top, there are tabs for 'Code Editor' and 'Scene'. The 'Code Editor' shows a C++ file named 'CookTorrance...Fresnel.cgfx' with the following code:

```
246  
247 float4 fresPS_t(vertexOutput IN) : COLOR  
248 {  
249     UBE (emapSampler, IN.Tex  
250     1 * tex2D(emapSampler  
251  
252  
253  
254  
255  
256  
257  
258  
259     pass envpass  
260     DepthTest  
261     DepthMask  
262
```

The 'Scene' view shows a 3D environment with several objects and a camera. A 'Material Scene Bindings' panel is open, showing a 'PointLight' material with a '<Unbound>' status. Below it, a 'Node Properties: No' panel lists various properties like Name, Uri, Visible, Frozen, Object Scale, Object Shear, and Object Rotation. A 'Bright Surface Color' panel shows a color picker and a value of 0.8 0.5 0.1. A 'Dark Surface Color' panel shows a value of 0.5 0.4 0.05. A 'Gooch cool tone' panel shows a value of 0.05 0.05 0.6. A 'Minimum for Gloss Drp' panel shows a value of 0.7. A 'Strength of Glossy' panel shows a value of 0.200. A 'ColorMap' panel shows a value of 0.7. A 'Gooch warm tone' color picker is also visible, showing a color wheel and RGB values (R: 0.500, G: 0.375, B: 0.234, A: 1.000). The interface is cluttered with various panels and controls, illustrating the 'Improved User Interface' mentioned in the title.

Typical FX Composer Layout



The screenshot displays the NVIDIA FX Composer 2 interface, organized into several key panels:

- Management:** The top-left panel shows the Library/Viewer and Project Explorer. The Library/Viewer contains a tree view of assets including Cameras, Effects (anisotropic1-fx, post_sepia), and scene26_ms_eye_left-fx. The Project Explorer shows the current project structure.
- Coding:** The central panel is the Material Editor, displaying the shader code for EYE.cgfx. The code includes preprocessor directives for debugging, material properties, and default settings. A Python console at the bottom shows a warning: "profile already exists".
- Properties:** The top-right panel shows the Properties window for the selected material, scene26_ms_helmet. It lists various parameters such as Position, Look Direction, and World Bounds ABB.
- Textures:** The bottom-left panel is the Texture Explorer, showing a grid of texture thumbnails for the current scene.
- Info, Scripting, Errors:** The bottom-middle panel displays the Output, Task List, Python console, and Shader Performance monitors.
- Preview:** The bottom-right panel is the Scene preview window, showing a 3D view of the scene with a camera and a helmet model. It includes a frame counter and a playback control bar.



DEMO: Shader Authoring

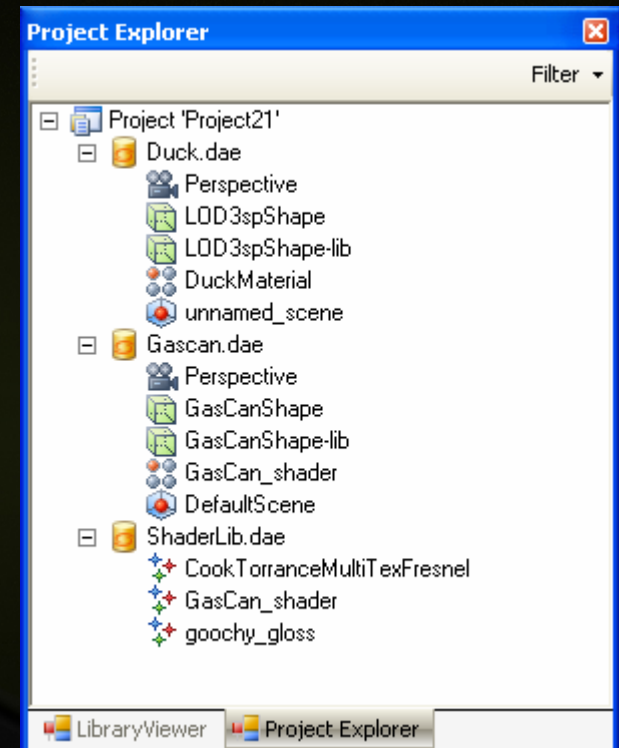
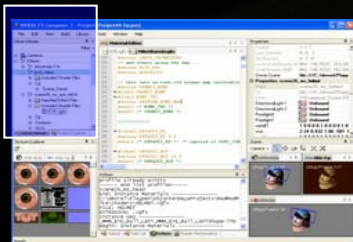
- Loading Project
- Compiling
- Errors and Tasks

Project Explorer



- **Manage multiple documents**
- **Documents contain one or more assets**
- **Assets are effects, materials, meshes, and other scene elements**
- **Organize you assets**
 - Move, copy, delete, rename
 - Drag and drop
 - One or many documents
 - Effect Libraries

● COLLADA





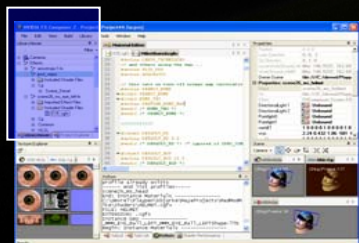
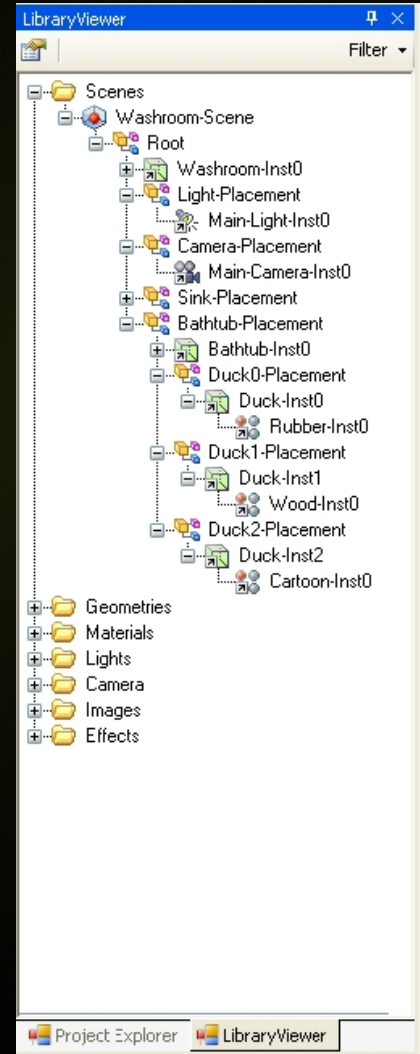
- **Open asset exchange format (.dae)**
- **Governed by the Khronos Group**
 - Includes numerous ISVs and IHVs
 - Mature DCC plugins for extensive support
- **Supported by FX Composer 2**
 - Import & Export
 - Supports *effects and materials*
 - Facilitates asset exchange with DCC apps
 - Other file formats supported



Library View



- Organize across documents
- Sort assets by type
- Visualize Assets
 - Scenes
 - Effect
- Authoring



Effect Authoring

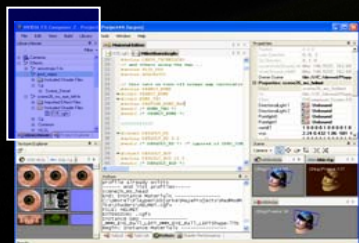
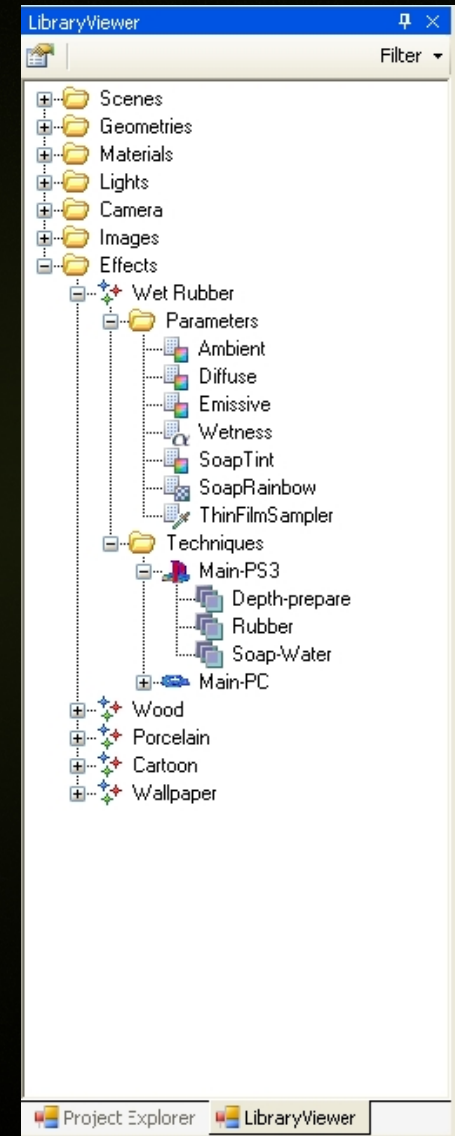


View effect structure

- Techniques
- Passes
- Parameters

Authoring using toolbars and context menus

- Add children
- Remove children
- Advanced options



CgFX & COLLADA FX Cg

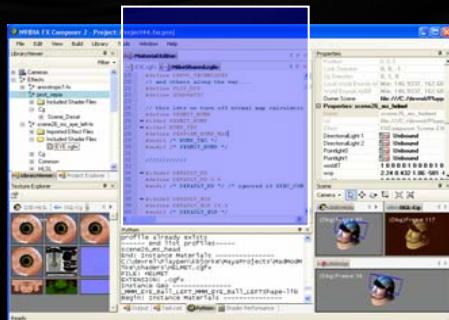


● CgFX

- Hand coded
- Less user interface assistance

● COLLADA FX for Cg authoring

- Most user friendly experience
- Fully editable via user interface
- Can migrate your CgFX assets
- Less hand coding
 - Zero XML
 - Cg, GLSL only when writing the GPU shader code





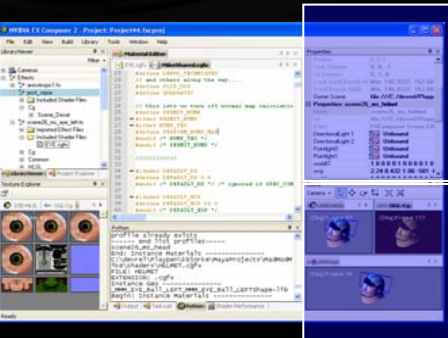
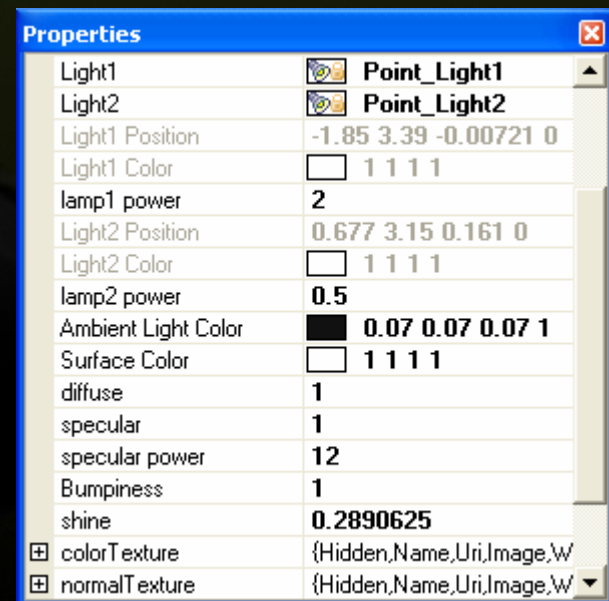
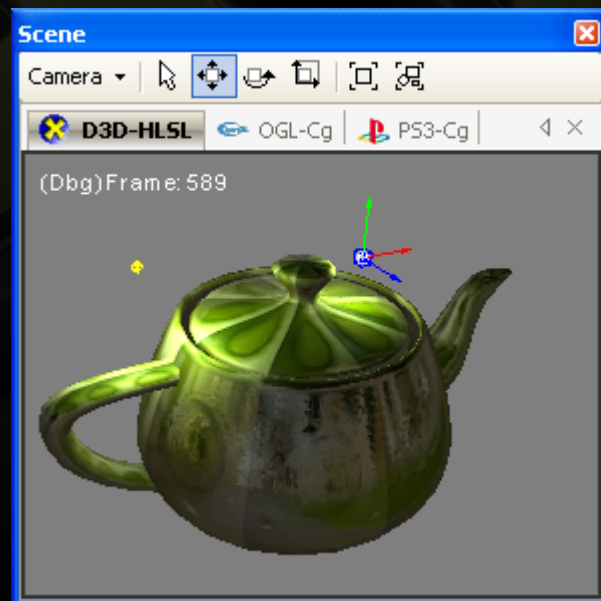
DEMO: Material Authoring

- Creating materials
- Assigning effects
- Tweaking material parameters



DEMO: Scene Integration

- Light creation
- Scene traversal
- Bind light to material
- Realtime manipulation



Shader Performance Simulation



eye-left.cgfx (PC-OGL) | eye-left.cgfx (PS3) | eye-left.cgfx (PS3)

Analyse a Pass
 Compare Passes

Techniques:

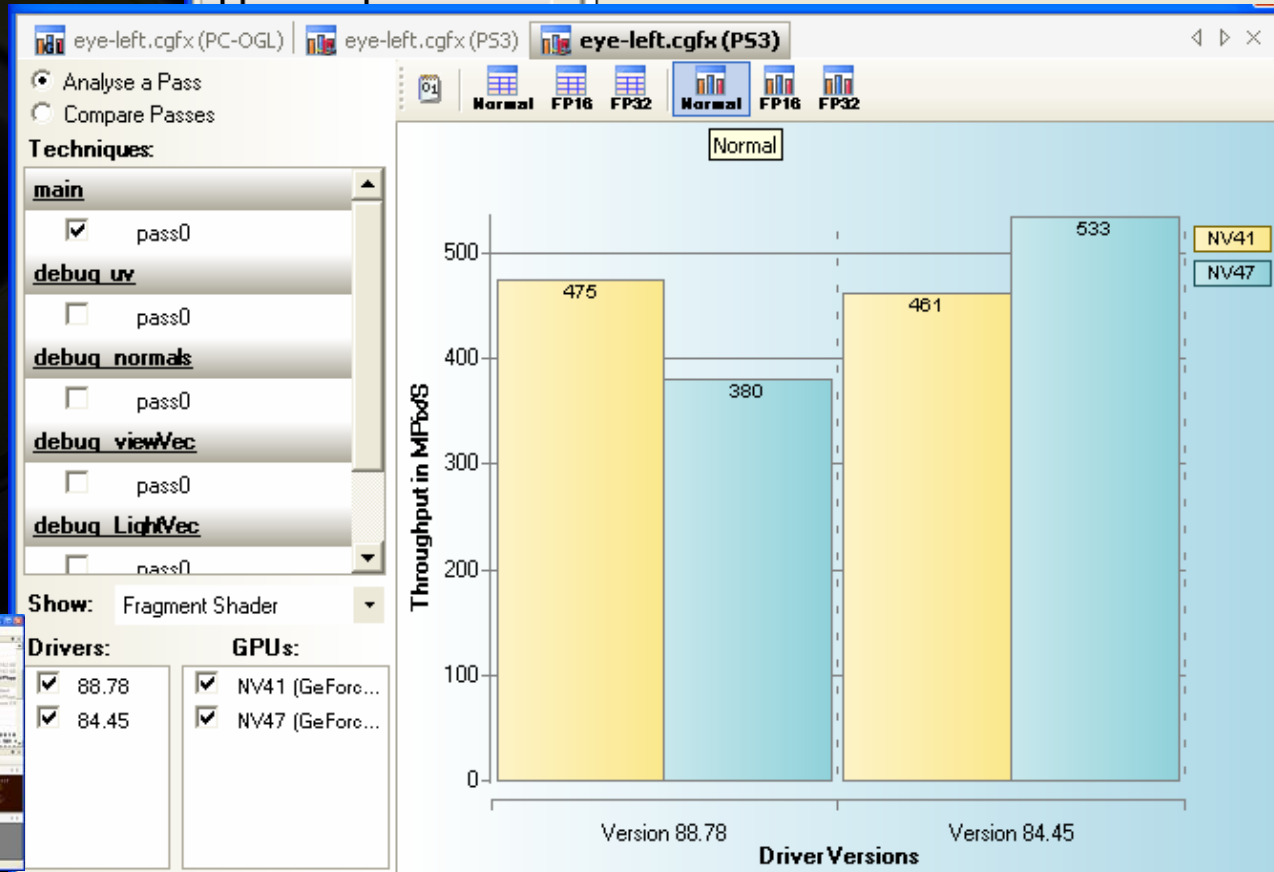
main

- pass0

debug_uv

- pass0

	Version 88.78			Version 84.45		
	Regs.	Cycles	MPix/s	Regs.	Cycles	MPix/s
pass0						
NV41 (GeForce 6600)	3	7	475	3	16	461
NV47 (GeForce 7800)	3	9	380	3	2	533



Production Pipeline Integration



- **Scene binding expressions**
- **Automation via scripting**
- **Plug-in based architecture**
- **Source control integration**
- **Production Pipeline Samples**

Scene binding expressions



Semantic



```
float4 LightPos : Position
```

```
<
```

```
string Object = "PointLight";  
string Space = "Object";
```

Annotations

```
> = {-10.0f, 10.0f, -10.0f, 0.0f};
```

Custom Semantics and Annotations



- Hook parameter to scene and system data
- Expressions via xml configuration file
- Extensive Library of Operators
 - dot & cross products, mux, demux, matrix ops, ...
- Custom operators
 - Built from XML using operators
 - Via plug-in

```
1 :<RemappedSemantic name="myWorldView">
2 :   <MatrixMultiply description="World * View">
3 :     <input type="internalsemantic" value="world"/>
4 :     <input type="internalsemantic" value="view"/>
5 :   </MatrixMultiply>
6 :</RemappedSemantic>
```

Automation Via Scripting



- **Automatic assignment of**
 - **Materials to geometry**
 - **Shader parameters to scene objects**
(nearest lights, cameras, etc...)
 - **A model's accessories to attachment points**

- **Common-tasks toolbar (ala Maya/MEL)**

Automation Via Scripting



```
#Python scripting

# Convert any Possible Profile to COLLADA FX
def ConvertToCOLLADA():
    effects = FXRuntime.Instance.Library.FindLibraryItems(FXEffect)
    for effect in effects:
        for profile in effect.Profiles:
            if profile.CanConvertToColladaFX() == True:
                profile.ConvertToColladaFX()

# Create an effect
def bindMMM():
    CmdGroupBegin.Do("script: assign cgfx files to MMM ")

    SelectRenderPort("OpenGL")
    ForceRedraw()

    CmdGroupEnd.Do()

# get the cgfx files to assign to MMM

files = FXEffectUtils.GetEffectFiles()
for pathname in files:
```

...



DEMO: Scripting

- Automatic assignment
- Etc.

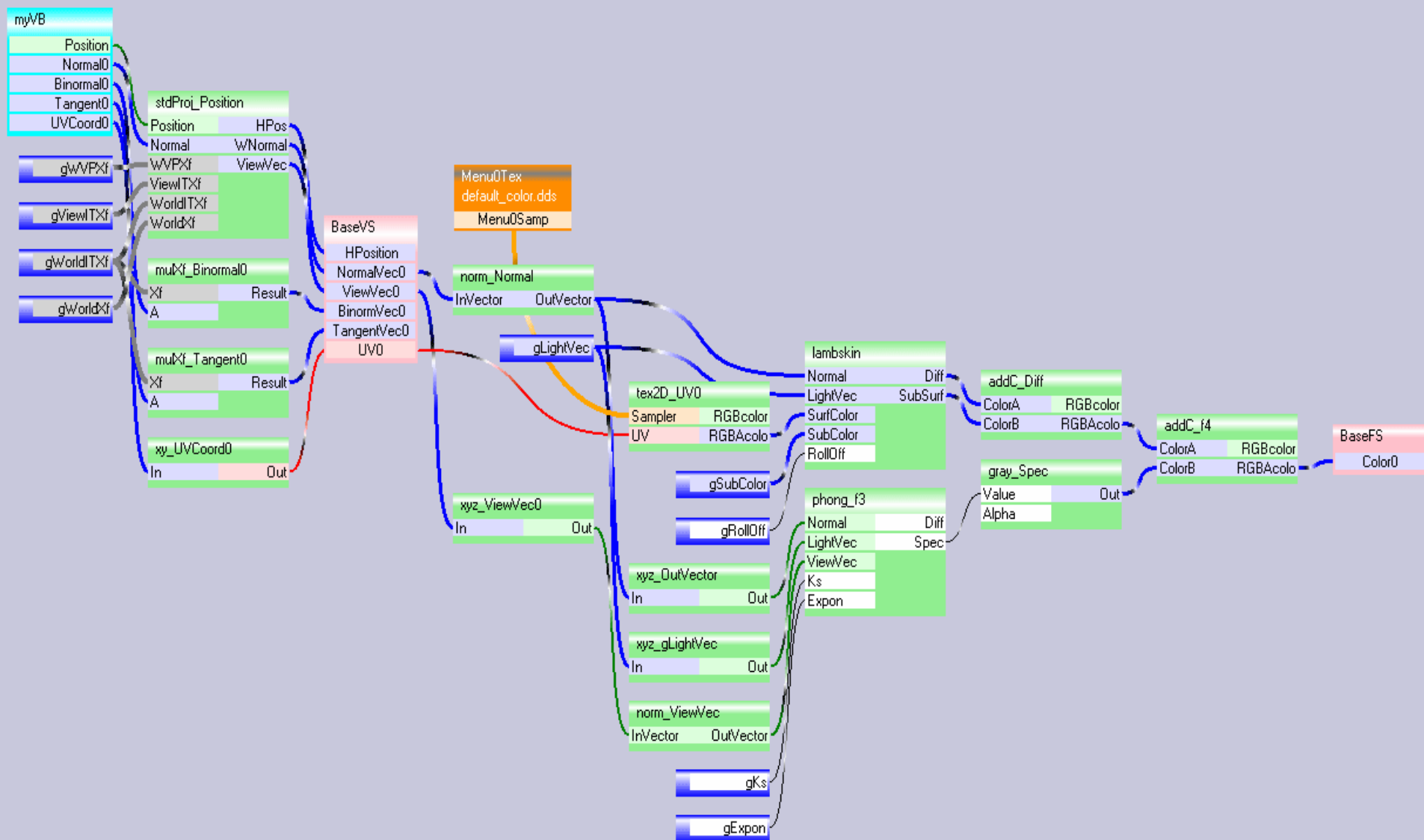
Custom Plug-ins



- Importers
- Exporters
- Semantic expression operators
- Rendering devices
- Procedural geometry generation (fins, hair, etc...)
- Custom authoring environment

...etc...

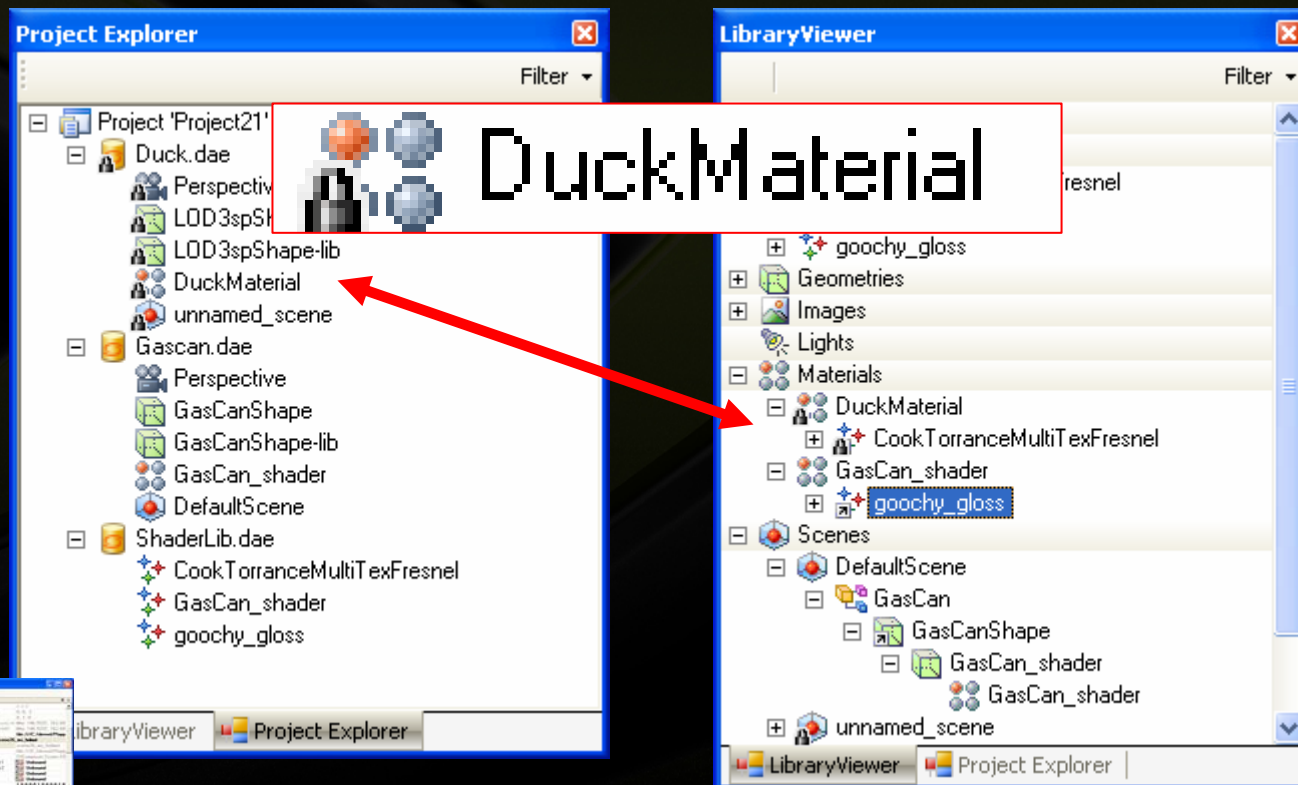
Custom Plug-ins: Sky is the limit



Source Control Integration



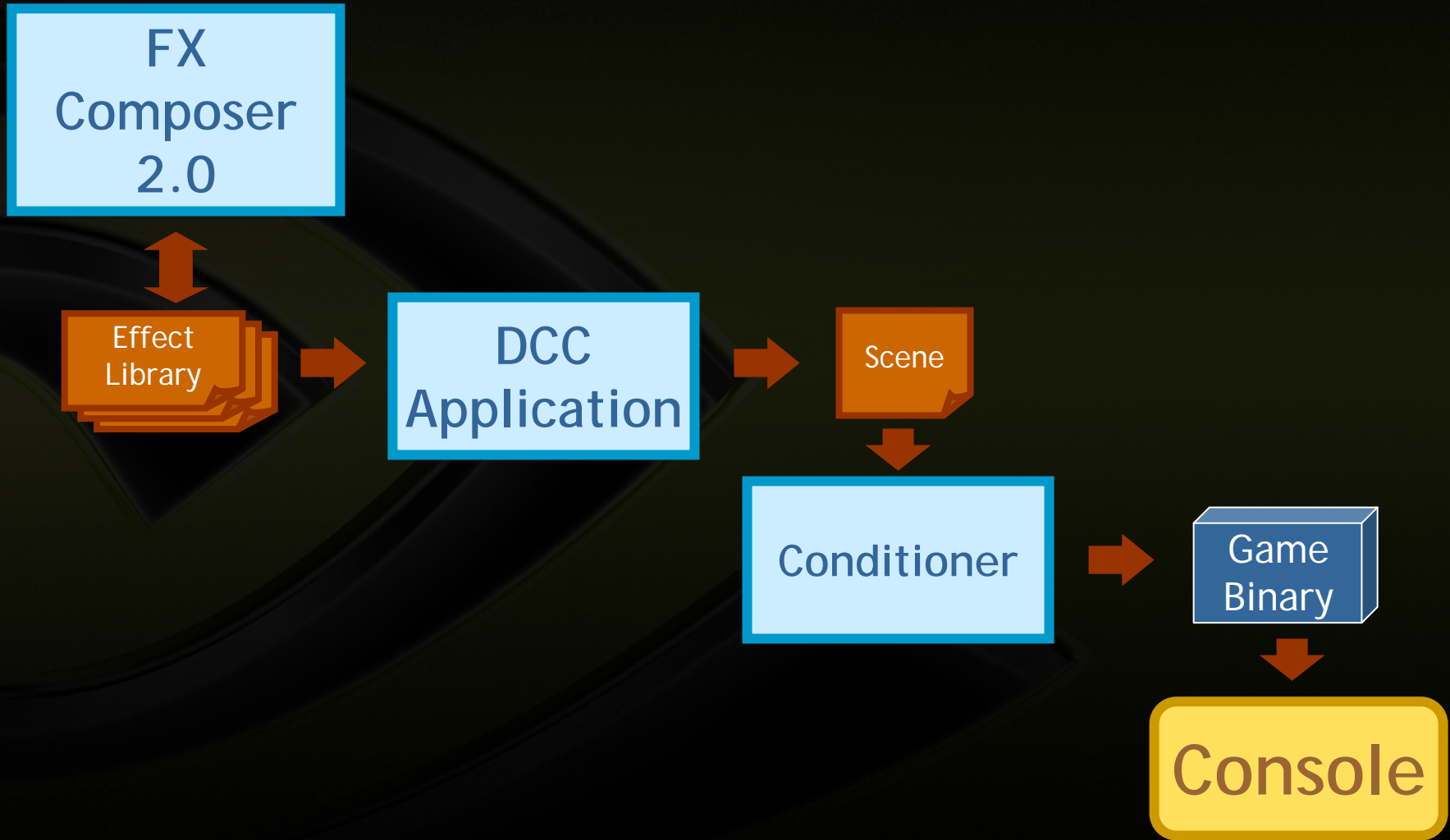
- Seamless integration into source control software
- Documents and assets reflect file-based state



Pipelines: FXC2 centric



Pipelines: DCC centric



Pipelines: Handheld



Host PC

Target Handheld

FX Composer 2.0 ES

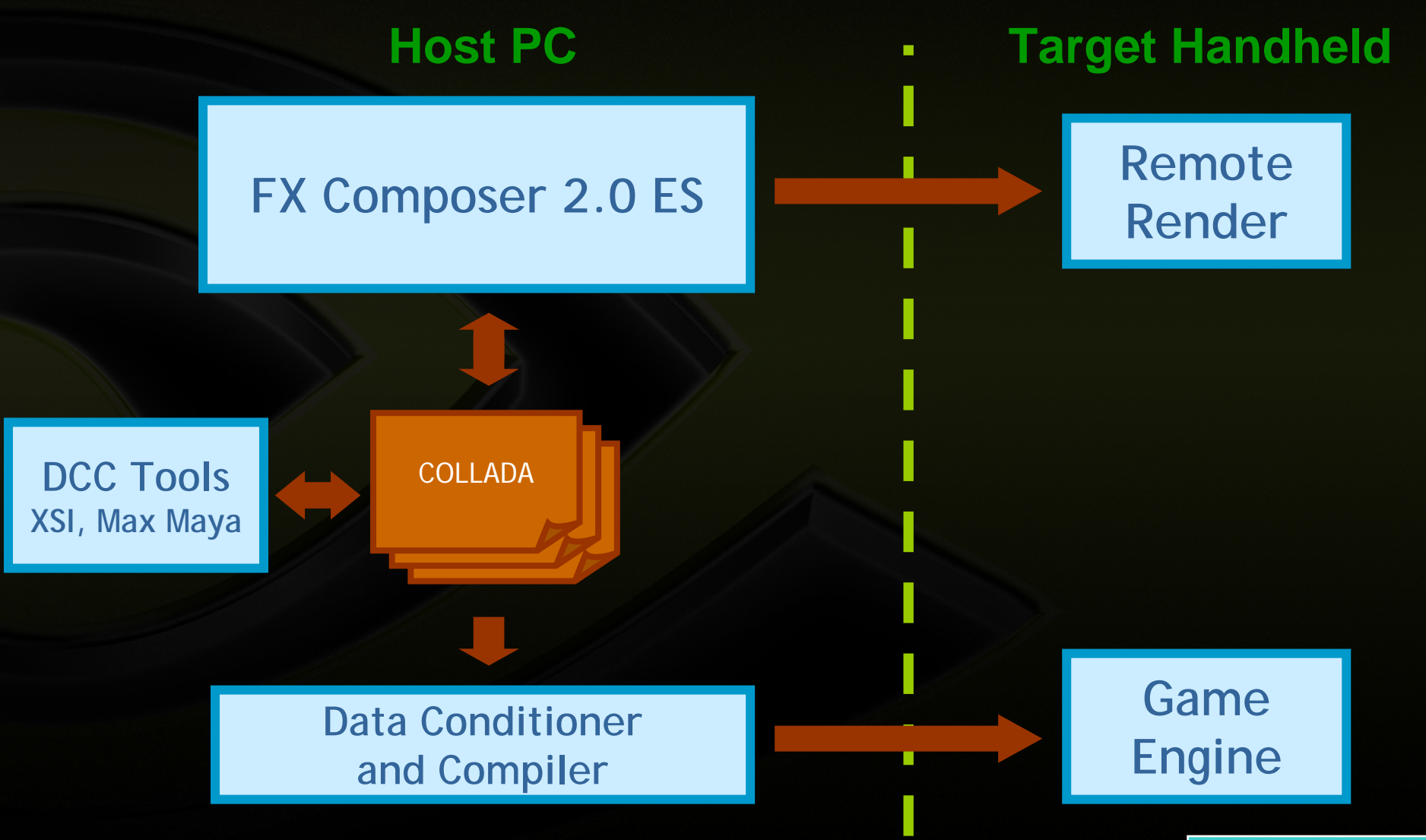
Remote
Render

DCC Tools
XSI, Max Maya

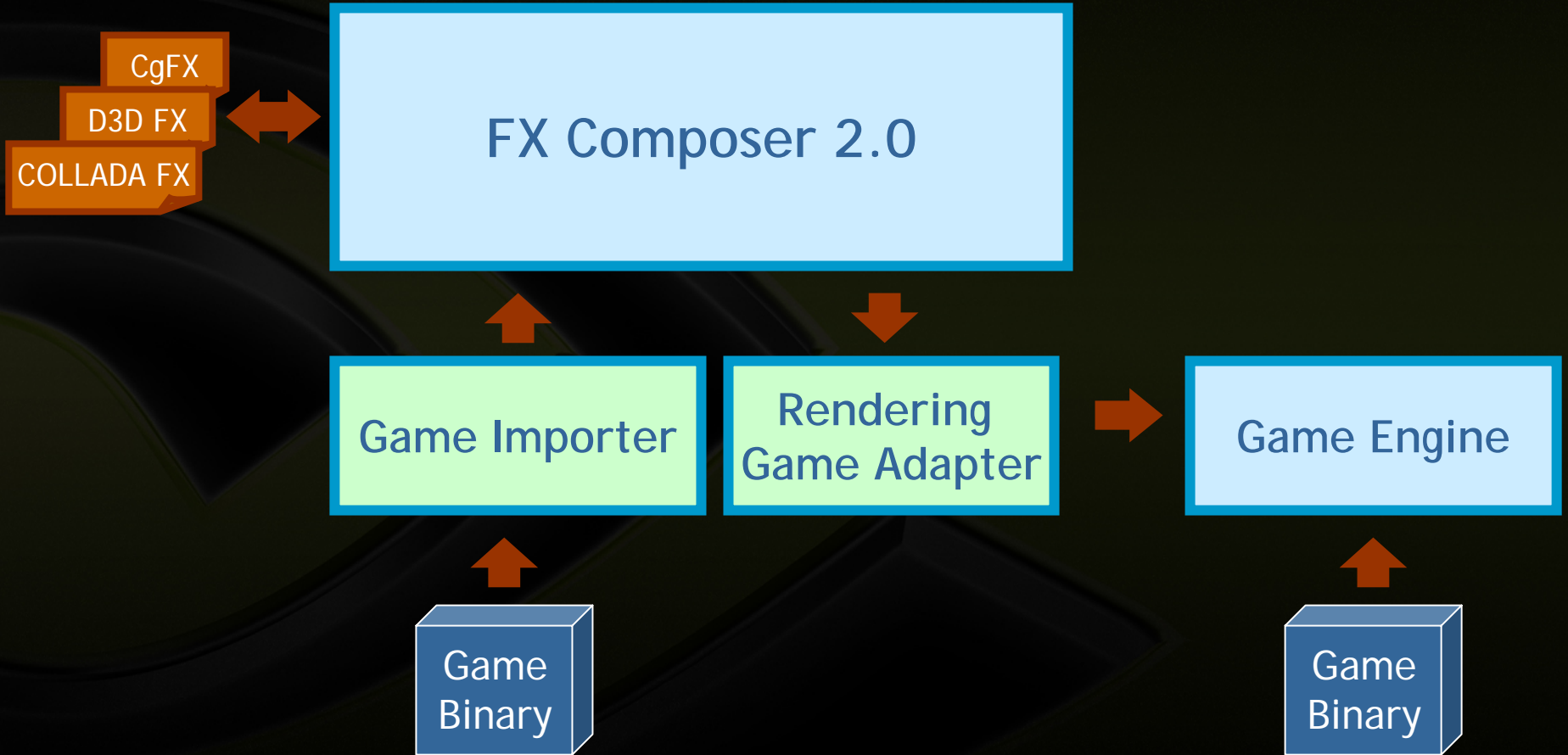
COLLADA

Data Conditioner
and Compiler

Game
Engine



Pipelines: Engine



FX Composer 2.0 Alpha 5



- **Alpha5 release ETA end of summer '06**
 - Document and asset management
 - COLLADA FX authoring
 - Shader parameter scene binding
 - Custom semantic and annotation support
 - Python scripting
 - Shader performance
 - Available to Sony PS3 developers and limited partners

- **Beta release ETA end of fall '06**
 - Open to public

Conclusion



- **Next-generation of shader IDE is on its way**
- **Production-ready with powerful features**
- **NVIDIA is closely working with Khronos and others to deliver a professional-grade authoring tool**

Q&A



- Send us emails for early alpha and beta releases

fxcomposer@nvidia.com

- Thanks

Philippe Rollin (prollin@nvidia.com)

The Source for GPU Programming

developer.nvidia.com

- Latest News
- Developer Events Calendar
- Technical Documentation
- Conference Presentations
- GPU Programming Guide
- Powerful Tools, SDKs and more ...



Join our FREE registered developer program for early access to NVIDIA drivers, cutting edge tools, online support forums, and more.

NVIDIA

developer.nvidia.com

©2004 NVIDIA Corporation. NVIDIA, and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation. Nalu is ©2004 NVIDIA Corporation. All rights reserved.