



# BLADE GAMES WORLD

## BLADE 3D

REVIEWED BY GREG SNOOK

**BUILT UPON THE XNA FRAMEWORK,** Blade Games World's Blade3D eases the process of game development for independent and professional developers alike. While the XNA Game Studio products have been a boon to those who want to rapidly prototype ideas and craft full Xbox Live Community Games, these products are primarily a programmer's development environment. For the average developer, generating the tools and infrastructure needed to develop a complete game or prototype can still be a daunting task requiring significant programming ability. Blade3D seeks to alleviate this burden by housing a rich set of development, editing, and debugging tools in a well designed WYSIWYG scene editor.

### THE BLADE3D EDITOR

» Blade Games World's goal for Blade3D is to allow anyone to create a game with a reasonable investment of time and energy. As such, the interface of the 3D development environment is designed to contain all the features and amenities of the Blade3D engine without becoming an intimidating mess of dialogs and deeply-nested menus. The layout is clean, easily readable, and surprisingly intuitive to navigate. While there are an ample number of tutorial videos available to walk users through the most common tools, the design lends itself to exploration. Common elements are easy to locate, making the tool easy to use without needing to keep a crib sheet of keyboard commands on hand.

The layout is limited to a single viewport window, which can be distressing to content creators accustomed to using

multiple viewing angles at once. In practice, I didn't find this to be too much of an issue, but a second window would be useful when debugging AI behavior over a large environment or when trying to align objects along multiple axes. The single viewport does allow for the live frame rate monitor to give a fairly accurate depiction of your scene's performance, and swapping between active cameras is simple enough. Thankfully, all menus are user-positionable, and can be docked or left free-floating. This can help multi-monitor users capitalize on the single viewport by pushing all extraneous menus to another display.

Prominent in the layout is the unified object browser and property editor. All raw content items and game entities are shown here in a tree view hierarchy. This combines the properties of a scene graph with a complete content browser. Simple drag and drop controls allow users to place items from this browser into the scene or the visual scripting system. Materials can be dragged directly onto models, models can be dragged into visual scripts, and so on.

Also notable is that all entities in the scene browser can be edited at runtime using a uniform property editor. Any game entity or content asset can expose a set of tunable parameters to the property editor for live editing. This simple concept is in some ways the greatest part of the Blade3D package. While your game is running, you can inspect and tweak the properties of all game and content items. For iteration and tuning, this is a huge time saver.

To round out the tool, Blade Games World has also added

support for multiple levels of undo and redo, the building and deployment of binaries, integrating content from the Blade Games World community marketplace, taking screenshots, and navigating tutorials. There is even a menu button to submit bugs directly to the Blade3D development team. Clearly, Blade Games World has gone through a lot of effort to make this a comprehensive tool for game developers.

### ENGINE FEATURES

» The editing environment is clean and intuitive, but developers

need functionality above all else. Luckily, Blade3D covers all the basics admirably. Skeletal animation, kinematics, physics, particle systems, terrain and audio subsystems are all provided as well as more esoteric items like lens effects and character inventory management. On the graphics front, the engine provides a rendering system which supports HLSL shader authoring (with integrated editor) and a data-driven material system.

All subsystems are functional, but some are still under development. For example, while



the engine supports twenty different model formats, all skeletal animations for a given model must be laid out in a single FBX file and imported in bulk. Upon import, the user can specify the frame ranges of individual animations to carve them back up into individual assets. While the remainder of the runtime animation system is well-crafted, this odd quirk in the content pipeline can take some getting used to.

Similarly, the embedded physics system has been hand-crafted by Blade Games World and is "still a little rough" (to quote its web page) when compared to

traditional physics middleware packages like Havok or PhysX. For basic physical interactions between collision volumes, however, I found it to be more than adequate. Rag-doll support is still in the planning stages, but the current offering supports rigid body collisions between arbitrary meshes as well as spheres, boxes, capsules and height maps. A custom physics solution handling four-wheeled vehicles is also provided to ease the creation of driving games.

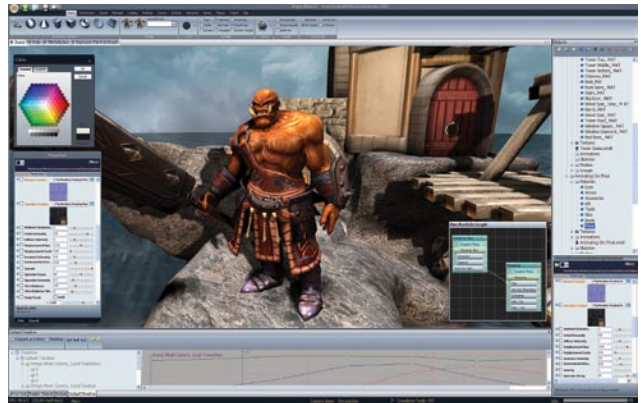
### VISUAL SCRIPTING AND EXTENSIBILITY

Blade3D provides a visual scripting system in the form of what it calls Visual Logic Diagrams. Using a flow chart-like representation, this system allows users to define logic by dragging objects and operators onto the design surface and hooking them together to achieve the desired result. These data flow diagrams allow authors to create animation blend trees, specify gameplay logic and define character behaviors. Like most other subsystems in Blade3D, these Visual Logic Diagrams can be edited in real time for easy debugging and tuning.

In addition to the Visual Logic Diagrams, Blade3D also provides an embedded script editor complete with syntax highlighting. Here users can craft C# game components to provide custom extensions to existing game objects. The entire engine and scene editor is also extendable using .NET languages and the provided Blade3D SDK.

### MARKETPLACE AND COMMUNITY

Blade Games World has created a development community of its own around Blade3D. The current user forum and wiki are sparsely populated, but growing at a healthy pace. Thankfully, this means new users have great access to the Blade Games World staff, who reply to most forum questions within 1–2 business days. The wiki contains many



empty entries, which Blade Games World will hopefully remedy in the near future. In the mean time, the Blade3D documentation, including both written and video tutorials, handles most common questions and gets users started quickly.

Blade Games World has also launched its own marketplace to allow users to buy, sell and distribute their games and game content. Marketplace browsing and downloads are handled directly through the scene editor, allowing content to be directly embedded into your game project. Like the forum and wiki, the marketplace is just getting off the ground. As a result, the content available is on the slim side.

### PRICING

Blade Games World has opted for a novel subscription-based model for Blade3D. Rather than purchasing or licensing the engine with a one time fee or royalty system, users pay a monthly fee to keep the editor operational. Several subscription tiers are available ranging from \$14.95 to \$99.95 a month (\$149.95 to \$999.50 if paid yearly).

The tiers all provide the same feature set and editor functionality, but differ in the level of support provided. Lower-priced tiers also require Blade3D watermarks or splash screens, while the higher tiers require no identifying marks and enjoy discounts to all Marketplace content.

The subscription model is innovative in that it allows

independent developers to get started for very little investment and step up their subscription level as needed over the course of development. The subscription is also tied to the ability to use the editor, not the game product created. This means that once your game is complete, you can freely distribute your product without maintaining a Blade3D subscription. Likewise, the monthly subscription model allows developers the flexibility to cancel and reinstate their subscriptions when the project needs to go on the back burner.

### CUT TO THE CHASE

Blade3D is a rich toolset for the indie game creator or the seasoned developer looking for a quick way to prototype ideas. The Blade Games World team has created a simple yet solid world editor, and packed into it a host of features to aid game developers of all skill levels. While some systems lack depth, all facets from animating user interface screens to managing game character inventories are provided in one form or another. When coupled with a responsive community and innovated pricing structure, the result is a welcome addition to any developer's toolbox.

**GREG SNOOK** is a development lead at Microsoft Game Studios, currently working on an unannounced project. He is also the author of Real-Time 3D Terrain Engines using C++ and DirectX 9 as well as a contributor to the Game Programming Gems book series. Email him at [gsnook@gdmag.com](mailto:gsnook@gdmag.com).

## BLADE GAMES WORLD BLADE3D

★★★★

### STATS

Blade Games World  
PO Box 1329  
Issaquah WA 98027-0053  
[www.bladegamesworld.com](http://www.bladegamesworld.com)

### PRICE

Tiered subscription model starting at \$14.95 per month

### SYSTEM REQUIREMENTS

Minimum:  
Windows XP with Service Pack 2, 2.0GHz CPU, 1GB RAM, SM 2.0b graphics card, 128MB, display resolution: 1280x1024 or higher, internet connectivity for activation and monthly verification

### Recommended:

Windows XP with Service Pack 3 or Windows Vista, 3.2GHz CPU, 2GB RAM, SM 3.0b Graphics Card, 256MB, display resolution: 1600x1200 or higher, 32bit color depth, full-time internet connection for community access

### PROS

- 1 Well designed, comprehensive scene editor
- 2 User extendable through visual script, custom components, and .NET development.
- 3 Innovative subscription model

### CONS

- 1 Relatively new community and marketplace needs time to mature
- 2 Some content pipeline quirks to overcome
- 3 Single-viewport interface limits scene visibility