

LevelEditor User's Guide

The *LevelEditor User's Guide* describes the basics of the LevelEditor and explains how to use its key features. It describes major components of the LevelEditor's graphical user interface (GUI) and how to construct a game level using the LevelEditor.

This document is intended for game designers who are building levels with the standard LevelEditor GUI. It does not discuss programming issues or specific plugins other than default plugins.

This document assumes that you have a solid, working knowledge of game development. It does not require you to be familiar with programming.

Version	Revision Date	Author(s)	Comments
1.0	02-Nov-2007	Joseph Schmitz	Initial version
1.1	25-Jan-2008	Joseph Schmitz	Updated to match version 2.3 of the software
2.4	16-May-2008	Joseph Schmitz	Updated to match version 2.4 of the software
2.5	15-Aug-2008	Joseph Schmitz	Updated to include Align Objects, Duplicating Objects, and Rotate on Snap features.
2.5	22-Aug-2008	Ron Little	Updated to include pick cycling feature, adding an existing level document as a subdocument, Move Manipulator, and add DAE (Collada) to the list of asset formats.
2.6	23-Jan-2009	Ron Little Susan Gallagher	Updated to match version 2.6 of the software
2.9	26-Apr-2010	Ron Little	Documented snap-along-axis feature. Changed document name
3.2	4-Aug-2011	Ron Little Martin Froehlich John Perry	Updated for ATF 3.2 Level Editor
3.5	2-July-2012	Alan Beckus MC Morrison	Updated for standalone LevelEditor 3.5 component; reorganized some of the information
3.6	7-Jan-2013	MC Morrison	Added new features for LevelEditor 3.6



Confidential. World Wide Studios use only.

© Copyright 2008-2013, Sony Computer Entertainment America, Inc.

Material contained in this document may not be copied, reproduced, reduced to any electronic medium or machine readable form or otherwise duplicated and the information herein may not be used, disseminated or otherwise disclosed, except with the prior written consent of an authorized representative of Sony Computer Entertainment America, Inc.

Sony is a registered trademark of Sony Corporation. XMB is a trademark of Sony Corporation and Sony Computer Entertainment Inc. PlayStation is a registered trademark of Sony Computer Entertainment Inc. PSP and UMD are trademarks of Sony Computer Entertainment Inc.

All other trademarks are the properties of their respective owners.

Filename: LevelEditorUserGuide.pdf

Table of Contents

Introduction.....	7
What the LevelEditor Does	7
Who Should Use This Guide.....	8
Additional Resources for LevelEditor Users	8
What this Guide Contains	8
LevelEditor System Requirements.....	9
Installing the LevelEditor	9
What's New for LevelEditor 3.5.....	9
What's New for LevelEditor 3.6.....	10
Conventions Used in this Guide.....	10
1 Getting Started	11
Installing the LevelEditor	11
Starting the LevelEditor.....	12
Opening a Sample Level.....	12
Getting Around	13
Learning about Game World Objects.....	17
Manipulating Objects.....	19
Placing Objects in the Level.....	21
2 Basic Concepts	23
Resources	23
Game Objects	24
XML Schema Basics	27
Documents	27
Locators	28
Prototype Assets	29
ControlPoints and Curves	29
Scripts	30
Integration with StateMachine.....	30
3 The LevelEditor User Interface	31
The LevelEditor Workspace	31
LevelEditor Menus	31
LevelEditor Toolbars.....	31
Design View.....	36
Palette.....	37
Output Window	39
IronPython Window	39
Property Editor.....	39
Grid Property Editor.....	40

Project Lister.....	40
Layers List	41
Resources List.....	42
Resource Preview	43
Resource Metadata	44
Resource Root Folder	45
Bookmarks.....	46
Render Settings.....	46
Customizing the Default Workspace.....	48
Keyboard Shortcuts	49
Load or Save Settings	50
Preferences	51
Using Basic Features	52
Managing and Editing Files	52
Setting User Preferences	53
Adding and Organizing Assets	54
Displaying Different Views of the World	54
Specifying How to Navigate the World	55
Specifying How to Render Game Objects.....	55
Moving, Rotating, and Scaling Objects	56
Displaying and Hiding Game Objects.....	57
Creating Linears	57
Grouping and Ungrouping Objects	58
Working with Docking Controls.....	58
4 Typical Work Flow.....	61
Creating Resources and Project Infrastructure.....	61
Constructing and Inspecting Game Levels	61
Creating the Final Output Format	62
5 Working with Resources	63
Using Supported Resource File Formats.....	63
Managing Resources on Multiple Computers	63
Adding Resources to a Level	64
Selecting an Individual Resource.....	64
6 Working with Game Objects	67
Viewing Game Objects in the Project Lister	67
Creating Game Objects	68
Using the Palette as a Source of Game Objects.....	68
Using Prototypes to Create Similar Game Objects	69
Using Locators to Create Game Objects.....	70
Associating Resources with Game Objects	70
Working with Game Object Properties	71
Using the Property Editor to View and Edit Properties.....	71
Using the Grid Property Editor to View and Edit Properties	72

Locking Game Objects.....	72
Preparing StateMachine Assets.....	73
7 Working in Design View	75
Displaying Different Views of the World.....	75
Navigating in the Design View	77
Zooming the World Using the Mouse	77
Rotating the World.....	78
Panning the World	79
Using Bookmarks to Save Frequently Used Views	80
Placing Game Objects in the World	81
Placing an Object at any Point in the World	81
Snapping Game Objects	82
Snapping a Game Object to the Surface of Another.....	83
Selecting Game Objects	84
Controlling Which Objects Are Selected	84
Selecting Obscured Objects.....	84
Moving, Rotating, and Scaling Objects	85
Using Pivots to Control Rotation and Scaling.....	85
Moving Objects	86
Scaling Objects.....	87
Extending Objects	89
Rotating Objects	90
Duplicating Game Objects	92
Rotating Game Objects to Face a Nearby Surface	93
Displaying and Hiding Game Objects	94
Zooming Specific Game Objects	95
Rendering Game Objects	95
Using Layers to Unclutter the World	101
Using Sublevels to Manage Large Game Worlds.....	102
Using Linears to Lay Out Lines and Boundaries	105
Adding Light Objects	108
Directional Light.....	108
Box Light.....	109
Point Light.....	109
Working with StateMachine Assets.....	109

Introduction

The LevelEditor is a powerful tool for constructing and assembling game levels. It provides a WYSIWYG interface (as shown in *Figure 1*) and tools for creating robust game levels. You can also add plugins that customize and extend the LevelEditor.

This guide describes LevelEditor components and features, tells you how to use the LevelEditor to build levels, and explains where to find additional information.

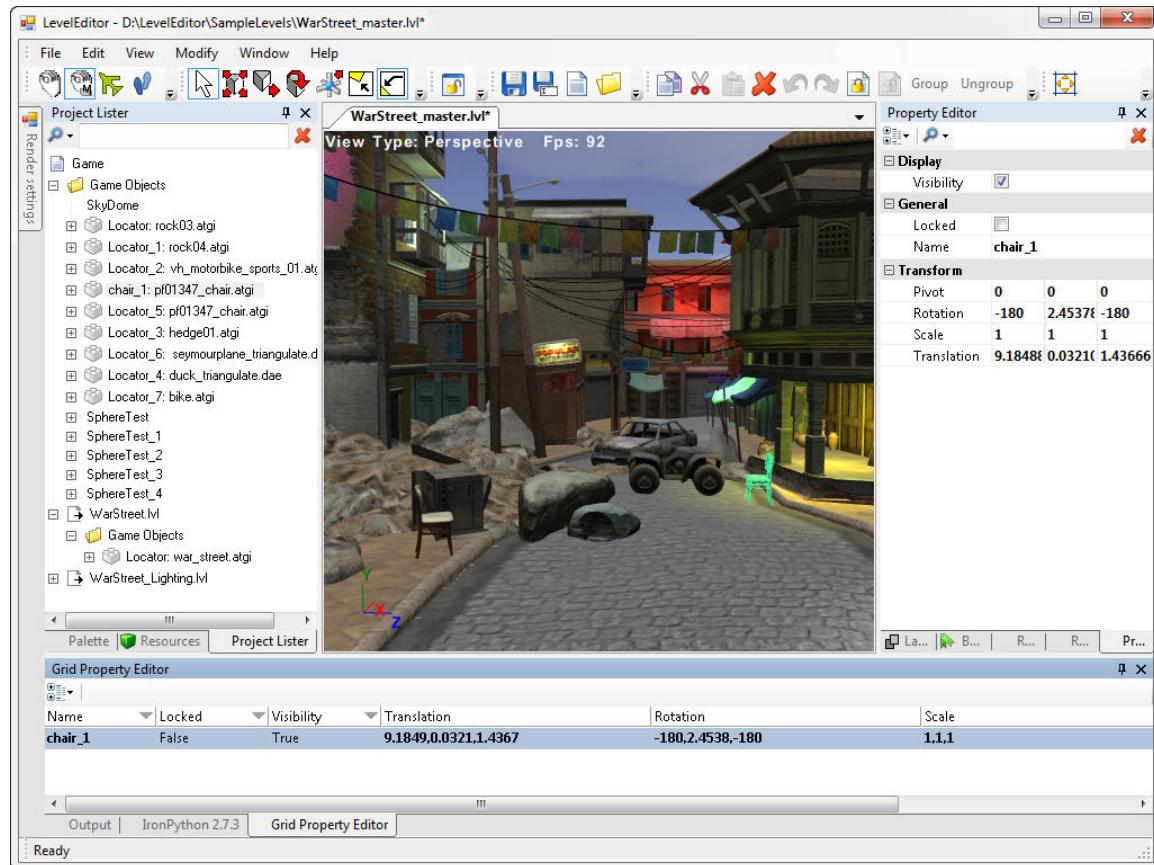


Figure 1 LevelEditor WYSIWYG interface

What the LevelEditor Does

The following LevelEditor features help you construct game levels efficiently and collaboratively:

- Work with a variety of file formats
- Associate assets with game objects
- Position, rotate, scale, and snap game objects precisely



- Edit game object properties
- Show or hide groups of game objects to unclutter the view as you work
- Construct Linears (lines and curves)
- Add defined behaviors to game objects with a StateMachine plugin

Who Should Use This Guide

The *LevelEditor User's Guide* is for designers who want to build game levels using the LevelEditor interface.

This guide assumes that you are familiar with game development concepts and terminology, game-construction practices, game components, and working with a three-dimensional workspace.

You do not need to understand programming to use this guide. This guide describes how the LevelEditor uses plugins and XML schemas, but it does not expect you to program them yourself.

For information about creating tools for games using the Authoring Tools Framework (ATF) API, writing plugins, or annotating XML to extend the LevelEditor, see the ATF programmer documentation in the **wws_atf\Docs** folder.

Additional Resources for LevelEditor Users

For more information, visit the following SHIP pages:

- [LevelEditor](#) page

This page is a great place to get started with the LevelEditor: it describes features and benefits, and provides links to downloads, documentation, and video tutorials. This guide refers to specific videos that were available when this guide was released, but new videos are added all the time. Check in regularly to see what's new.

- [Authoring Tools Framework](#) page

Find out more about the framework used to build the LevelEditor and other game-development tools, such as the State Machine Editor.

What this Guide Contains

This guide provides the following chapters:

- [Getting Started](#): a brief introduction to some of the major features of the LevelEditor and how to use them
- [Basic Concepts](#): an overview of the basic concepts used by the LevelEditor
- [The LevelEditor User Interface](#): a description of the LevelEditor graphical user interface, including the toolbars, the windows, and how to use them
- [Typical Work Flow](#): a high-level description of the major tasks associated with game design
- [Working with Resources](#): how to work with resources (game assets)



- [Working with Game Objects](#): how to work with game objects, including how to create them and how to modify their properties
- [Working in Design View](#): how to use the LevelEditor Design View to create and modify a game world

LevelEditor System Requirements

You can use the LevelEditor on a computer that provides the following required functionality:

- Microsoft® Windows Vista® or Microsoft Windows® 7 operating system (32-bit or 64-bit)
- Microsoft DirectX® 10 (or later) compatible graphics card

Installing the LevelEditor

The LevelEditor is installed using Package Manager; see "[Installing the LevelEditor](#)" in the "Getting Started" chapter.

If you are a programmer, you can build your own LevelEditor executable from the source code by performing the following tasks with Microsoft Visual Studio® 2010 (or later):

- Build the native rendering engine using the following solution:
`.\components\wws_leveleditor\build\LvEdRenderingEngine.sln`
- Build the LevelEditor.exe using the following solution:
`.\components\wws_leveleditor\build\LevelEditor.vs2010.sln`

What's New for LevelEditor 3.5

LevelEditor 3.5 is a standalone tool, independent of the ATF. This version of the LevelEditor provides the following advantages and new features, compared with the ATF Level Editor 3.2:

- Support for 64-bit Windows operating systems.
- Native rendering (the rendering is done in C++ with DirectX11).
- Integration with GameEngine through a bridge API.
- Level resources are loaded using a background thread.
- Support for different types of light sources that can be placed anywhere in the level.
- Dynamic shadows to help object placement.
- Can fully or partially use game engine code for rendering.
- Can easily add new shaders.
- Faster level loading.
- Can easily add new file format using native API (Collada DOM, Atgi lib, in-house binary format).
- Resource Preview window for ATGI resources.

What's New for LevelEditor 3.6

LevelEditor 3.6 adds the following features:

- Package Manager. You now use the Package Manager to install LevelEditor.
- Integration of StateMachine with the LevelEditor. You can use StateMachine assets within the LevelEditor to define specific behaviors for game objects.
- Prototype assets. You can save one or more game objects as a prototype asset that you can then reuse in any game level. This feature replaces the previous version's Prototypes feature.
- Rendering normals. You can render game object normal vectors to help debug an object's geometry.
- Pick-cycling for overlapped objects. You can easily select objects that overlap or are visually obscured by other objects.
- Extension manipulator. You can extend or stretch an object along a single axis.
- Move pivot point command. You can quickly move an object's pivot point to one of several predefined locations.
- Pick filter. You can apply a simple filter to control which objects can be selected in the Design View.
- Sublevel object placement. You can easily drag objects to the Design View and have them appear within a particular sublevel, rather than at the top main level hierarchy.
- Bezier spline linear. You can define a linear as a Polyline, a CatmullRom spline, or a Bezier spline.

Conventions Used in this Guide

This guide uses the following conventions:

- The term “LevelEditor” (with initial capital letters) refers to the LevelEditor component.
- The term “level editor” (all lower-case) refers to a general level editor.
- Text in `monospaced` font is used for code examples.
- Sections marked as “Advanced Topics” contain advanced information that you do not need to understand to use the LevelEditor. However, if you have a background in programming or modeling, the information in these topics can be useful.



1 Getting Started

This chapter describes some of the basic features of the LevelEditor and how to use them to perform basic tasks associated with designing a game level. It uses one of the sample levels to show how you can work with existing content, as well as how to add your own content based on existing resources (game assets). This chapter describes how to start the LevelEditor, open a sample level, get around within the level, learn about and manipulate game objects, and how to place objects in the level.

The LevelEditor provides additional features beyond those described in this chapter. See the later chapters in this book for a description of what the LevelEditor can do and how you can use it to design sophisticated game levels.

Installing the LevelEditor

Use Package Manager to install the LevelEditor distribution. You can download it from the [Package Manager download](#) page on SHIP. For more information about Package Manager, see [WWS SDK Package Manager](#).

To install the full release, perform the following tasks:

1. Create a folder in which to install LevelEditor, such as C:\LevelEditor36.
2. Open a Windows Command Prompt window, and enter the following command:
wwspm install wws_leveleditor -path C:\LevelEditor36
3. If the following prompt appears, enter y (for “yes”) and press Return:

```
No WWS SDK installation found. Would you like to create a new  
one at "C:\LevelEditor36" ?
```

```
[yn] :
```

4. Package Manager finds the components needed for LevelEditor and displays the following messages:

```
WWS SDK installation found at "C:\LevelEditor36".
```

```
Analyzing components...
```

```
Components to install:
```

```
[wws_atf,3.6]
```

```
[wws_leveleditor,3.6]
```

```
Would you like to proceed? [yn] :
```

5. Enter y and press Return. Package Manager installs the components needed for LevelEditor and lists the installed components:

```
Retrieving components...
```

```
Deploying component [wws_atf,3.6] to "C:\LevelEditor36"...
```

```
Deploying component [wws_leveleditor,3.6] to
```



"C:\LevelEditor36"...

You can run the LevelEditor as soon as the installation is complete.

Starting the LevelEditor

To start the LevelEditor, navigate to the folder into which you installed LevelEditor and double-click the **LevelEditor.exe** file.

The LevelEditor opens the default workspace for an untitled level, as shown in *Figure 2*.

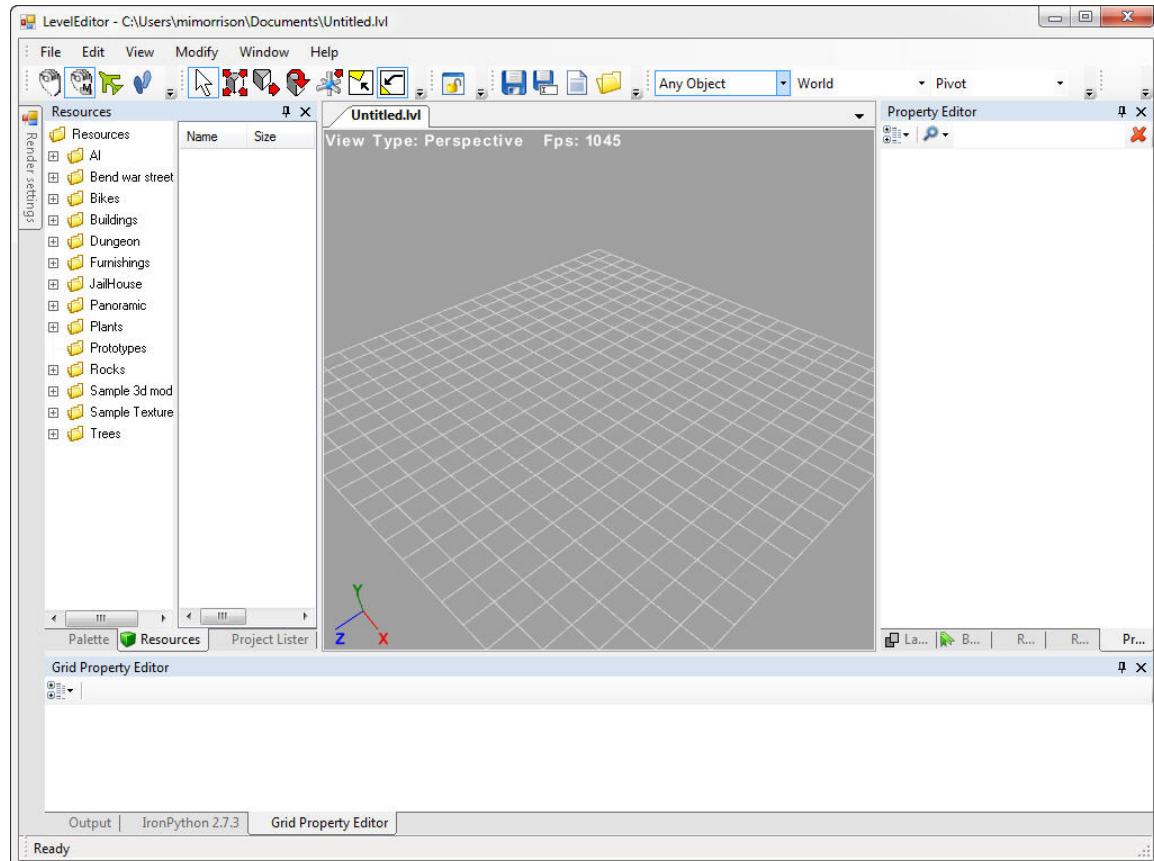


Figure 2 Untitled level

Opening a Sample Level

To open a sample level, select **File > Open Game**, navigate to the folder that contains the Basic sample level (for example, **\SampleLevels\Basic**), select the **_Main.lvl** file, and click **Open**. The LevelEditor opens the main level of the Basic sample, as shown in *Figure 3*.

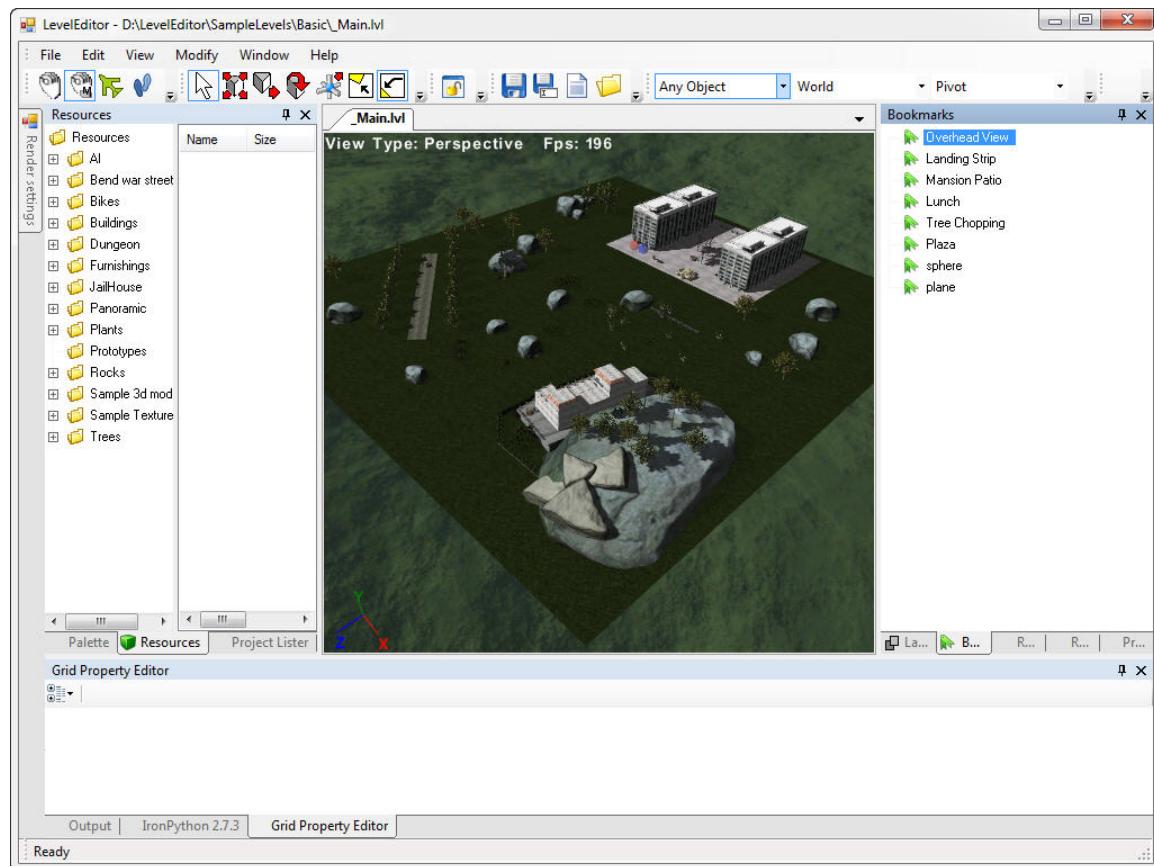


Figure 3 Overhead view of the Basic sample level

The Basic sample level includes a number of visual elements, including an office complex, a mansion, and an airplane landing strip.

Getting Around

The central pane of the user interface that shows the game objects and scenery of a level is called the Design View. By default, the Basic sample level is displayed using a three-dimensional perspective view. You can specify the following views (called projections) for the game world within the Design View: perspective, top, bottom, right, left, front, and back.

To specify the projection for a view, right-click anywhere within the Design View, select **Projection**, and then select the view that you want to use. For example, *Figure 4* shows four views of the Basic sample level: perspective, front, top, and right.

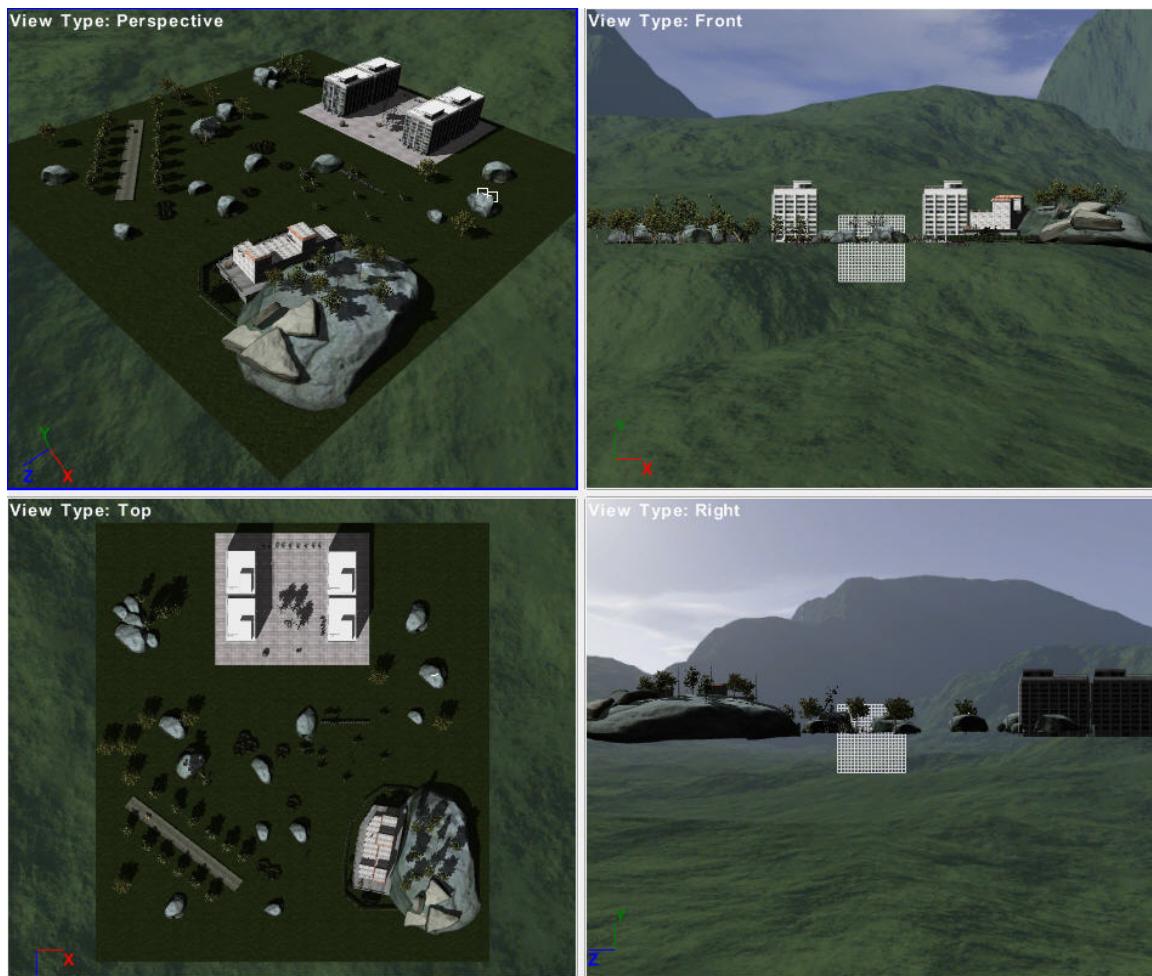


Figure 4 Four views of the Basic sample level

For more information about displaying different views, see the ["Working in the Design View"](#) chapter.

The Basic sample level also includes a number of predefined bookmarks, which allow you to switch to one of several saved views quickly; for example, overhead view, the landing strip, the mansion patio, a lunch scene, a tree chopping scene, and the plaza. To change the view shown in the Design View to one of the bookmark views, click one of the bookmarks in the Bookmarks pane to the right of the Design View. *Figure 5* shows four of the bookmarked views for the Basic sample level.



Figure 5 Four of the bookmarked views within the Basic sample level

You can also create your own bookmarks for your own views; see [Using Bookmarks to Save Frequently Used Views](#) in the "Working in the Design View" chapter.

Within a particular view, you can zoom in or change the camera angle, so that you can see any part of the game world, from up close or from far away. The LevelEditor has several modes that let you use the mouse (or other pointing device) to zoom, rotate, and pan your view of the world: Arcball, Maya® Style Trackball, Fly, and Walk. Each of these modes varies in what it can do and how you use it; see the ["Working in the Design View"](#) chapter.

Note:

The following examples assume that your Design View is set to the perspective projection view and that you use the Maya style Trackball mode. To specify this mode, click the Maya icon () in the Camera toolbar at the top of the LevelEditor window.

While in perspective view using the Maya camera, you can zoom in or out of the game world using either of the following techniques:

- Hold down the ALT key down, then drag while pressing and holding the right mouse button (you can drag up or left to zoom out, and you can drag down or right to zoom in)
- Scroll the mouse wheel

Figure 6 shows four different zoom levels of the Lunch bookmark view within the Basic sample level. The first zoom level is the default bookmark view; each of the other zoom levels shows increasingly more of the world as the camera zooms out.



Figure 6 Four zoom-level views of the Lunch bookmark view within the Basic sample level

While in perspective view using the Maya camera, you can rotate the view of the game world and pan the view of the game world. To rotate the world view around the camera, press and hold the ALT key and drag while pressing and holding the left mouse button:

- Dragging left and right rotates the world around a vertical axis.
- Dragging up and down rotates the world up and down around a horizontal axis.

To pan the world view around the camera, press and hold the middle mouse button while dragging up and down (for vertical movement). For horizontal movement, drag left and right.

Figure 7 shows four different rotation and pan views (with some zoom changes) of the Airfield bookmark view within the Basic sample level. The upper-left image shows the default bookmark view of the airfield.

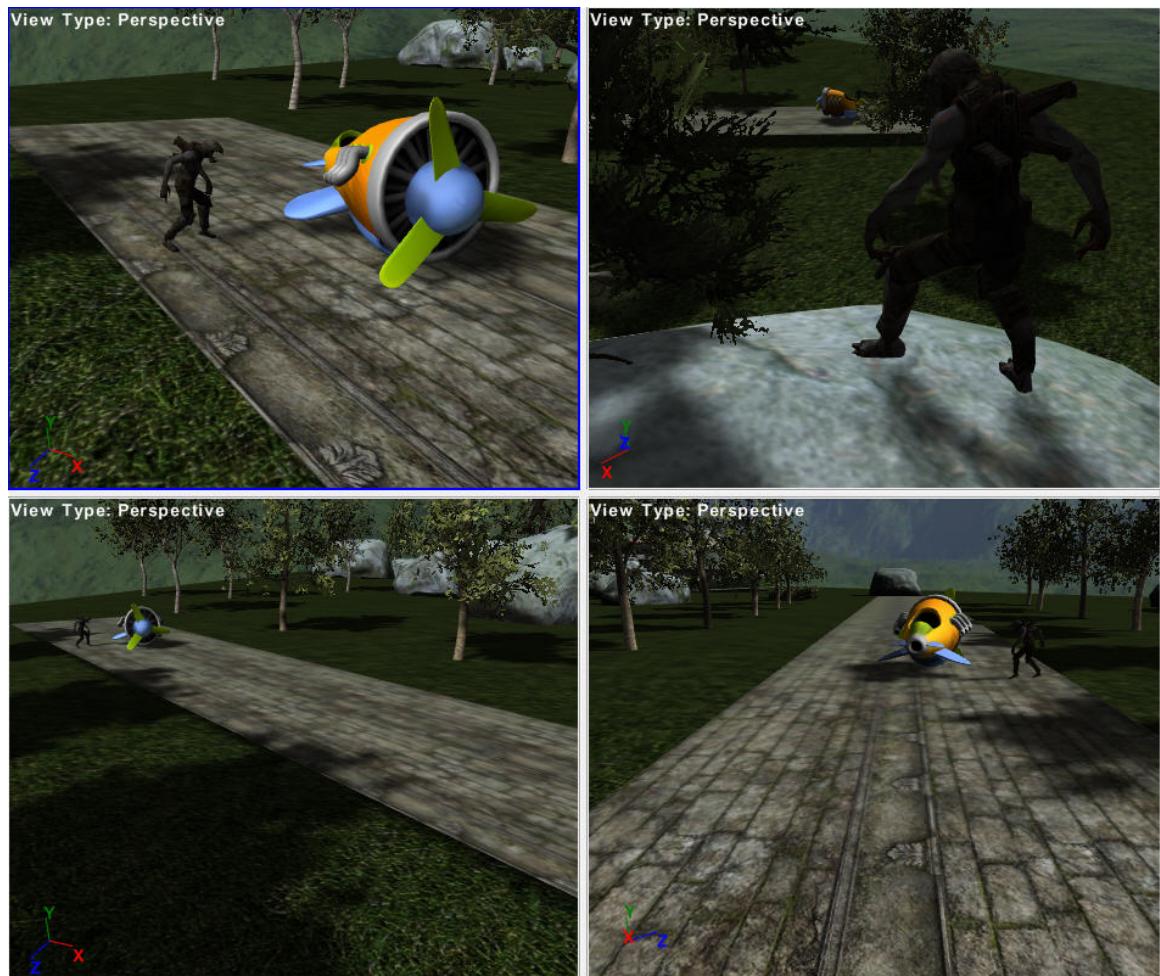


Figure 7 Four different rotation and pan views of the Airfield bookmark view within the Basic sample level

See the "[Working in the Design View](#)" chapter for more information about rotating and panning the world view.

Learning about Game World Objects

For each object within the game world, you can view and modify the object's properties. To view an object's properties, click on the object in the Design View, then select the Properties tab (to the right of the Design View) to display the Property Editor. You can also use the CTRL button to select multiple game world objects, and use the Grid Property Editor to view the properties of all selected objects. Click the Grid Property Editor tab (below the Design View) to display the Grid Property Editor.

Figure 8 shows both the Property Editor and the Grid Property Editor for selected objects in the Basic sample level. The arrows indicate the locations of the Property Editor and the

Grid Property Editor. The Property Editor shows the properties of the most recently selected object (the front-most chair in the figure below). The Grid Property Editor shows the properties of all four selected chairs.

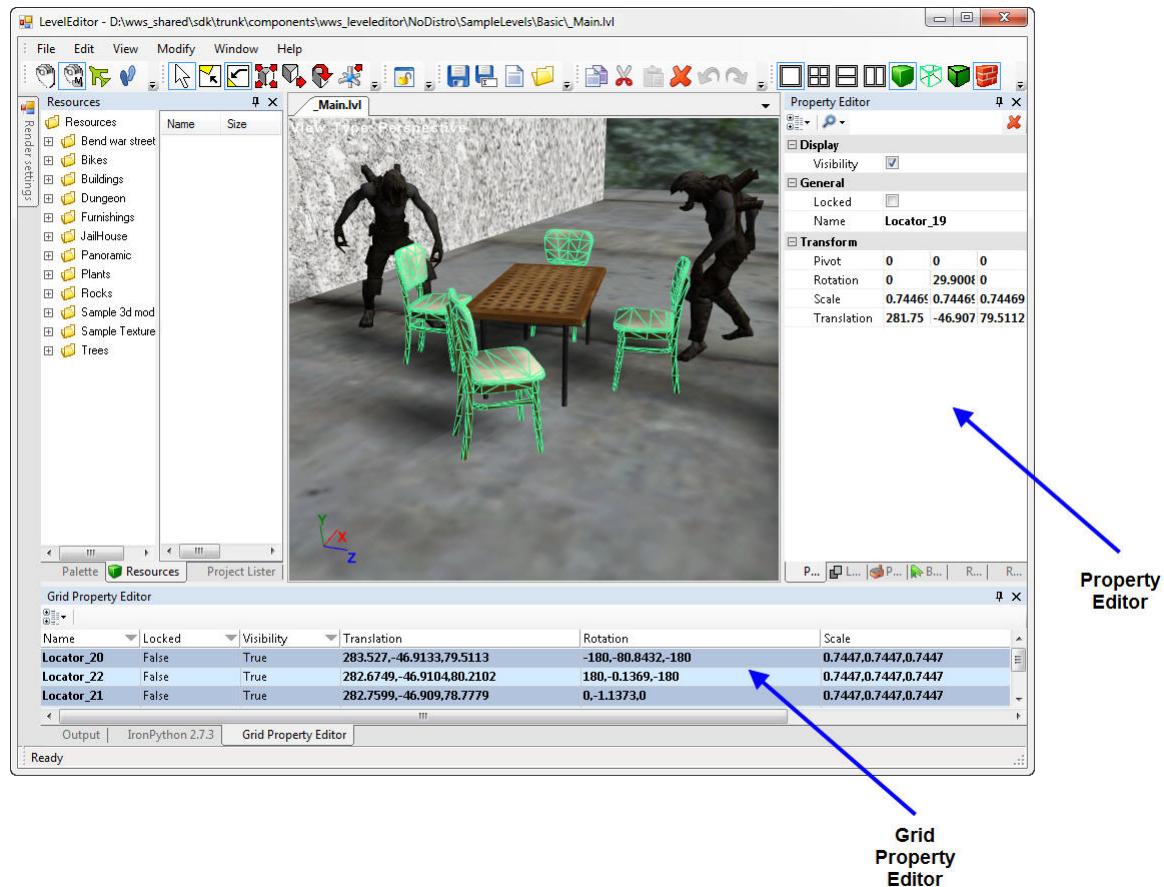


Figure 8 The Property Editor and Grid Property Editor for objects within the Basic sample level

Within either the Property Editor or the Grid Property Editor, you can click on any field to modify the selected property of the specific object. For example, for the chair object shown in *Figure 9*, modifying its **Rotation** properties (angular position within the world) and its **Translation** properties (position within the world) causes the chair to appear to have been knocked over. To achieve this effect, the **Rotation** property was changed from "0, 29.9009, 0" to "-90, 29.9009, 0", and the Translation property was changed (slightly) from "281.75, -46.9076, 79.5112" to "281.75, -46.67, 79.5112". Note that neither the **Pivot** property nor the **Scale** property was modified for this example.

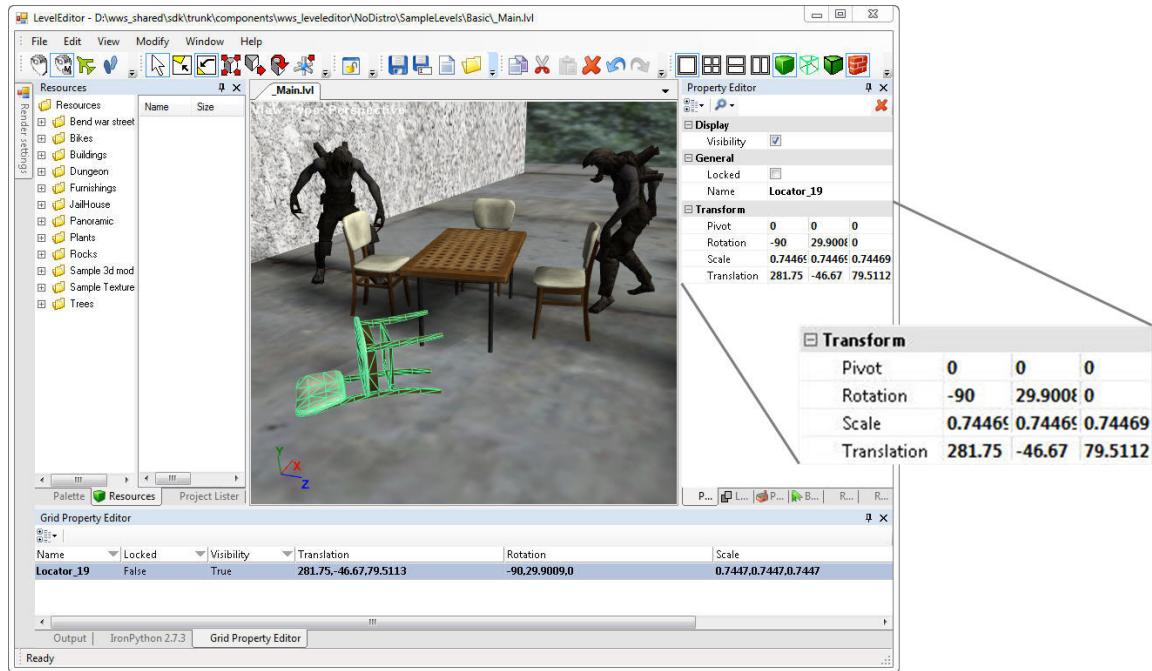


Figure 9 Modifying the properties of an object using the Property Editor

You can also use the mouse to manipulate certain properties of objects, such as their position and rotation, as described in the next section.

See "[The LevelEditor Workspace](#)", in the "The LevelEditor User Interface" chapter, for more information about the Property Editor or the Grid Property Editor.

Manipulating Objects

You can use the mouse to manipulate the position, rotation, and scale of an object. To reposition an object in the X, Y, or Z directions, click the Move icon () in the Select toolbar at the top of the LevelEditor window. When you select an object in the game world, its Move manipulator is displayed, as shown in *Figure 10*.



Figure 10 An object's Move manipulator

Using the three arrows of the Move manipulator, you can move an object up or down, left or right, and front or back.

To modify an object's roll, pitch, and yaw, click the Rotate icon () in the Select toolbar at the top of the LevelEditor window. When you select an object in the game world, its Move manipulator is displayed, as shown in *Figure 11*.

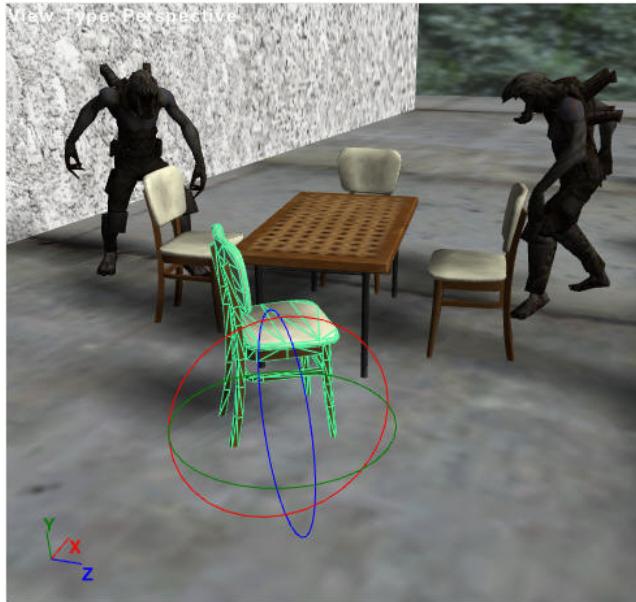


Figure 11 An object's Rotation manipulator

Using the three circles of the Rotation manipulator, you can move an object side to side (roll), forward or back (pitch), and left or right (yaw). Each of these rotations is relative to

a fixed point (the object's pivot point).

Alternately using the Move manipulator with the Rotation manipulator, you can move an object to any position within the game world. For example, you can use the Move manipulator and the Rotation manipulator to place a chair on the table in the Lunch scene of the Basic sample level, as shown in *Figure 12*.

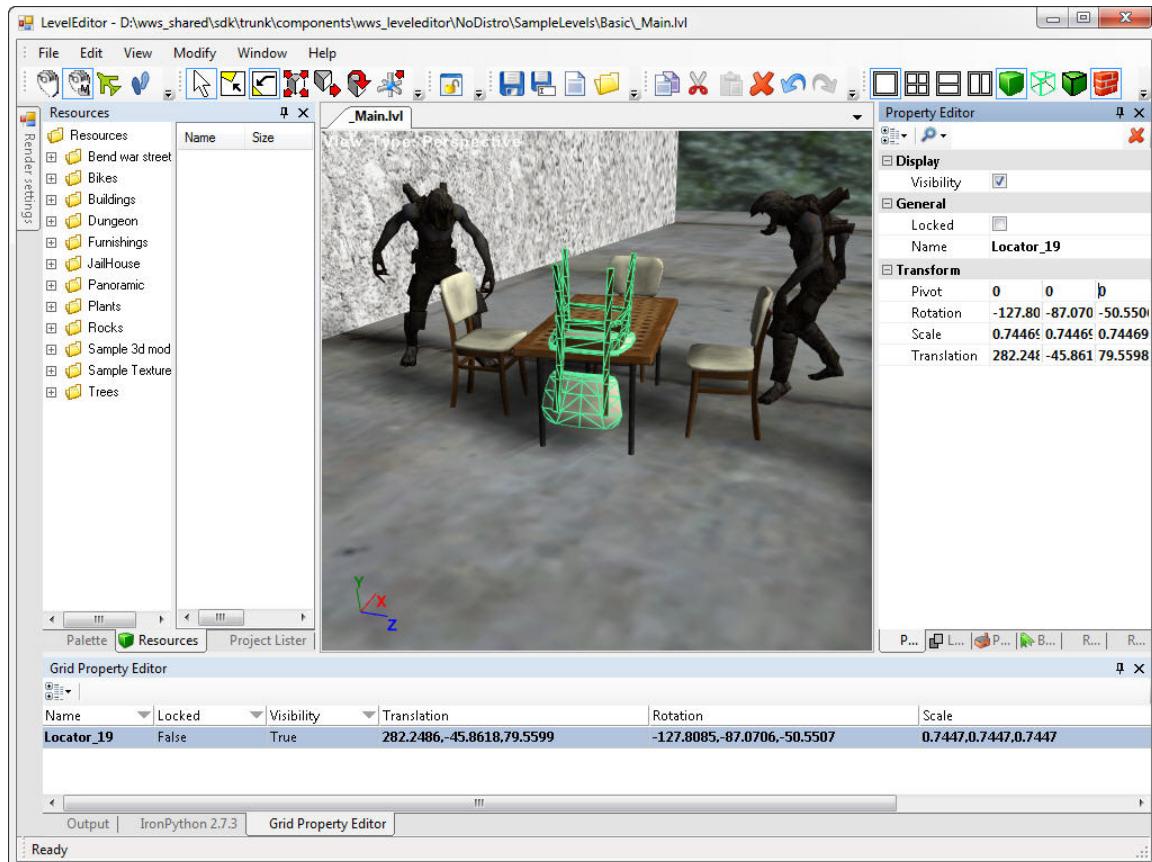


Figure 12 Moving and rotating an object

For more information about moving or rotating an object, see the ["Working in the Design View"](#) chapter.

Placing Objects in the Level

You can place objects into an empty level or into an existing level (such as the Basic sample level). You can use either the Palette or the Resources list to add objects to the level. The Palette contains basic game object types that you use to create game objects; to view the Palette, select the Palette tab to the left of the Design View. The Resources list displays all of the assets in the **ResourceRoot** folder; to view the Palette, select the Resources tab to the left of the Design View. See ["The LevelEditor Workspace"](#), in the "The LevelEditor User Interface" chapter, for more information about the Palette or the Resources list.

To add an object to the level, select an object from either the Palette or the Resources list, and drag it to the Design View. Then use the Move manipulator (and optionally, the

Rotation manipulator) to place the object precisely where you want it to appear. You might also need to rotate or pan the world view to verify the object's location. Use the Property Editor for fine-tuned modifications to the object's placement.

Figure 13 shows the Lunch scene of the Basic sample level, with three objects added from the Palette (the SphereTest, the ConeTest, and the CylinderTest), along with an object added from the Resources list (PF01352_stool from the **Furnishings** folder). Each of these objects has been moved, resized, or rotated. Use the Resource Preview pane to view a Resources object before you place it in the level. Use the Property Editor to change the colors of the three Palette objects (by default, they appear white). The view is slightly zoomed out from the Lunch bookmark view.

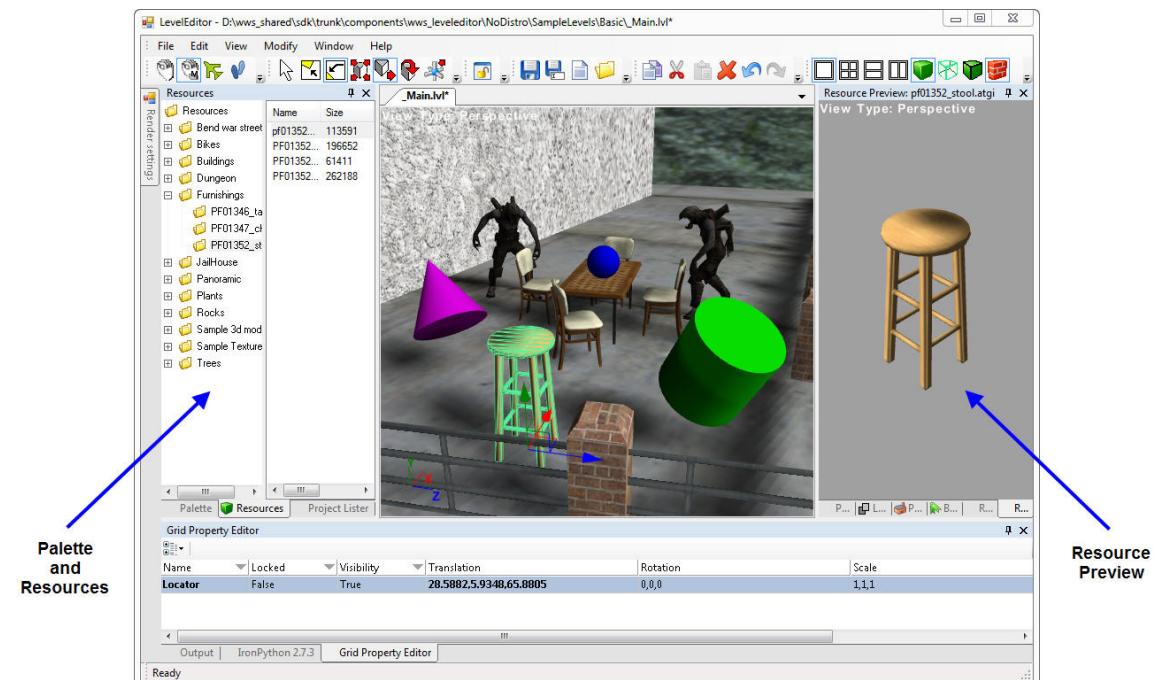


Figure 13 Adding objects to the level using the Palette and Resources

See "[Placing Game Objects in the World](#)", in the "Working in Design View" chapter, for more information about placing objects, including how to snap objects to other objects.



2 Basic Concepts

In general, the LevelEditor uses concepts and terminology that should be familiar to game designers, but certain concepts and terms might have slightly different meanings for the LevelEditor. Therefore, this chapter describes relevant game design concepts and terminology as used in the LevelEditor.

Resources

A *resource* (also called an *asset*) is content from an external file that was created in an application other than the LevelEditor. Although resources can be of any file type, they are typically files that you use in a level, such as models made with the Autodesk® Maya® animation software, behaviors written in a text editor, and textures created with Adobe® Photoshop® software. The LevelEditor provides built-in support for the resource file types described in ["Using Supported Resource File Formats"](#) in the "Working with Resources" chapter.

When you drag and drop a resource from the Resources list to your LevelEditor document, the LevelEditor creates a reference to that resource. You can also drag-and-drop files from the Windows Explorer with the same result.

The Resources list displays Resources that are in a folder known as the *Resource Root*. The default Resource Root folder is the **\Release.vs2010\Resources** folder. To specify a different folder, select **Edit > Preferences > Resources > ResourceRoot**. For more information, see ["Resource Root Folder"](#) in the "The LevelEditor Workspace" chapter.

LevelEditor documents (.lvl files) use a relative path from the Resource Root folder to save the reference to the Resource. Thus, an *.lvl file can always find a resource, regardless of where you save the .lvl file.

For more information about resources, see the ["Working with Resources"](#) chapter.

Try It!

To see how to add assets to the LevelEditor:

1. Open the LevelEditor.
2. If the Resources list is not open, select the **Window > Resources** menu item.
3. Specify the thumbnail view for the Resources list: Right-click in the Resources list. If the **Thumbnail View** menu option appears, select it. If, when you right-click the Resources list, the **Details View** menu option appears, the thumbnail view is already active and you can click anywhere in the LevelEditor to dismiss the menu option.
4. Navigate to the **Resources\Plants\Old_Whitebox_Foliage** folder, as shown in [Figure 14](#).

The left pane shows the directory tree for the assets. Thumbnail images of available resources appear in the right pane. Depending on the software installed on your computer, some asset types might not show a thumbnail view.

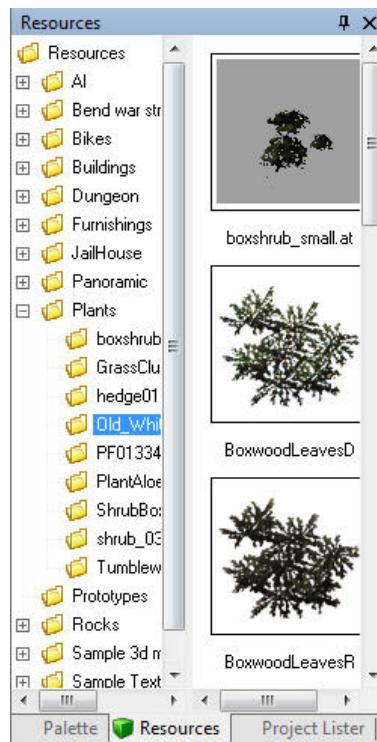


Figure 14 Assets in the Resources list

5. Optionally, you can open the Resource Preview pane, select an ATGI file from the Resources list, and view a preview of the object in the Resource Preview pane. In addition, the Resource Metadata pane displays metadata about the selected resource.

Game Objects

A game object is any object that has all of these characteristics:

- A location in the world.
- A listing in the Project Lister.

Game objects always appear in the **Games Objects** folder in the Project Lister.

- A set of values for game object properties.

Available game object properties vary according to the type of game object.

Typical properties include position, scale, rotation, and whether the game object is visible.

A game object can have one or more resource slots, which associate assets with the game object. A slot appears in the Project Lister tree as a child of a Game Object; you might have to expand a Game Object node to see its available slots.

To associate an asset with a game object, drag the asset to the appropriate resource slot. The XML schema that defines a particular type of game object specifies its available properties and the kinds of assets that can be associated with it. Common assets include geometry, texture, and behaviors. For example, the Orc example in the Palette's

Examples pane is defined as having four types of assets that can be associated with it and specific properties that can be set.

A programmer can create other game object types by creating an XML schema to define that type. For more information about XML schemas, see the *ATF 3.0 Programmers Guide: Document Object Model (DOM)*.

Try it!

To see how a game object appears in the Project Lister and how its properties appear in the Property Editor, perform the following tasks:

1. Open the Project Lister, Property Editor, and Palette windows.
2. Open the **Examples** section of the Palette window.
3. Drag the **Orc** example to **Design View**. Because it has no geometry associated with it, the Orc appears as a cube.
4. In the Project Lister, expand the tree for the **Orc** example game object, as shown in *Figure 15*.



Figure 15 Project Lister with one Orc example game object

When you dragged the Orc example to Design View, the LevelEditor created an Orc game object. The Ogre game object has four resource slots (geometry, animation, target, and friends) because its XML schema defines these slots. You can assign resources to each of these slots. For example, to make the orc look less like a cube, and more like a creature, you can drag the Hybrid model from the **Resources > Sample 3d models > Atgi assets** folder of the Resources window to the **geometry** resource slot of the Orc in the Project Lister. *Figure 16* shows the Orc game object before applying the Hybrid resource to the Orc's geometry, and after applying the resource to the geometry. If you want a larger orc, you can use the Scale manipulator to resize the orc.

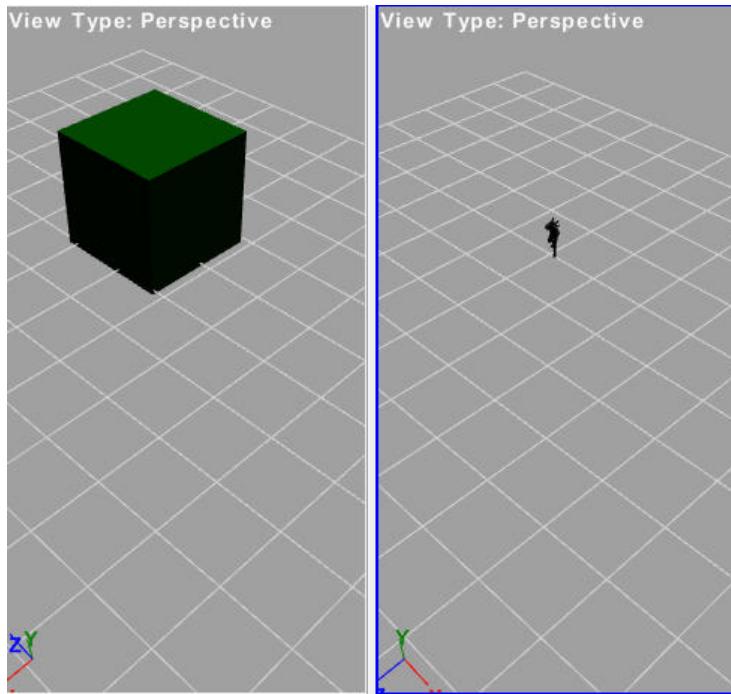


Figure 16 The Orc game object – before and after applying the Hybrid to its geometry

Figure 17 shows the Project Lister for the Orc game object after adding the Hybrid model to the Orc's **geometry** resource slot.



Figure 17 Project Lister with the Orc example game object after applying the Hybrid geometry

Figure 18 shows the Property Editor window for the Orc game object.



Figure 18 Orc properties in the Property Editor

XML Schema Basics

A game designer does not need to write or interpret XML schemas to work with the LevelEditor, but it is useful to understand their relationships to assets, game objects, and levels.

In general, a schema is an outline or diagram that describes an object. An XML schema uses XML elements and attributes to describe an object's data structure. The LevelEditor uses this description to interpret objects presented as assets.

The LevelEditor uses a variety of XML schemas to interpret objects. For example, the LevelEditor can work with ATGI files because an XML schema is available that describes the data in an ATGI file. To interpret a particular object, the LevelEditor requires an XML schema that describes all objects of that type. For example, the ATGI schema describes the structure of any ATGI object; the LevelEditor refers to this schema whenever it manipulates any ATGI object.

By writing or annotating XML schemas, an XML programmer can add features such as new game object types, new attributes, new resource slots for existing game objects, and support for additional types of assets.

For information about writing XML schemas to define additional game object types, see the *ATF 3.0 Programmers Guide: Document Object Model (DOM)*.

Documents

The LevelEditor saves a level as an XML document that contains descriptions of everything about the level. A level file has the **.LVL** file extension.

These are some items commonly found in an LVL file:

- References to files that the level uses as assets
- Game objects in the level, their associated assets, and property settings



Try It!

To create an LVL file:

1. With the document that you worked with in the ["Game Objects"](#) section still open, select **File > Save As** from the menu bar.
2. Use the Save As dialog box to name and save the file.

You can use any text editor to view the XML source within an LVL file.

Important:

Do not modify the content of the LVL file outside the LevelEditor.
Changing the XML within this file can corrupt the saved level.

The XML content of the document that you worked with in the ["Game Objects"](#) section should look similar to the XML shown below:

```
<?xml version="1.0" encoding="utf-8"?>
<game xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns:xs="http://www.w3.org/2001/XMLSchema" name="Game" xmlns="gap">
    <gameObjectFolder name="Game Objects" visible="true" locked="false" displayFlags="0">
        <gameObject xsi:type="orcType" transform="1 0 0 0 0 1 0 0 0 0 1 0 8.148263
          8.635567 8.18291 1" translate="8.148263 8.635567 8.18291" rotate="0 0 0" scale="1 1 1" pivot="0 0
          0" name="Orc" visible="true" locked="false" weight="0" emotion="0" goals="0" color="0"
          toeColor="0">
            <geometry uri="Sample 3d models/Atgi assets/hybrid.atgi" />
        </gameObject>
    </gameObjectFolder>
    <prototypeFolder name="RootProtoFolder" />
    <layers />
    <bookmarks />
    <grid size="20" subdivisions="20" height="0" snap="false" visible="true" />
</game>
```

This LVL file is fairly short because it contains only a single game object. LVL files with additional resources and objects are more complex.

Locators

A *locator* is an object that can have a specific position in space and one associated resource. The resource can be any type, but typically the resource is a geometry that you want to place in the world. Like any game object, a locator has additional properties that are available in the Property Editor.

A locator provides a way to position geometries that do not reference additional assets, such as behaviors or textures. For example, you could use locator game objects to position rocks on a hillside or streetlights along a street.

To create a locator, perform either of the following actions:

- Drag a resource directly to the Design View or to the **Game Objects** folder in the Properties Lister. The resource that you drag becomes associated with its own locator's single slot.
- Drag the predefined Locator object from the Palette to the Design View or to the **Game Objects** folder in the Properties Lister. When you create a Locator in this



way, no resource is associated with it until you assign one to the single slot that the locator provides.

When you drag to the Design View, the initial position of the object is wherever you drop it. When you drag to the **Game Objects** folder, the locator game object's origin is at 0, 0, 0. Either way, you can subsequently drag the object to a new position in the Design View or edit its position coordinates in the **Translation** field of the Property Editor.

For more information about locators, see the ["Working with Resources"](#) chapter.

Try It!

To create a locator game object directly from a resource:

- Using the level that you created in the ["Game Objects"](#) section, drag a resource from the Resources list to the Design View.

The new game object appears in the Project Lister and the Property Editor displays its position as the Translation properties. The resource is associated with the game object's resource slot.

To create a locator game object from the Locator object in the Palette:

- Drag the Locator object from the Palette to the Design View.

The new game object appears in the Project Lister and the Property Editor displays its position as the Translation properties. The locator game object has one resource slot, but no resource is associated with it.

Prototype Assets

Prototype assets facilitate the creation of large numbers of identical objects, such as a set of cars, a battalion of troops, or trees in a forest. You can create a prototype asset from one or more game objects, and use the prototype asset as a template for creating identical game objects. For example, you could create a set of tree game objects, save the set as a prototype asset, and then use the prototype to populate a forest. Each new tree game object that you create has the same properties as the prototype; that is, all the trees would have the same geometry, size, orientation, and relative position with other trees in the prototype. You can then modify each new tree's properties to create a realistic forest.

You can edit a prototype asset without affecting existing game objects made from the prototype. Likewise, you can edit game objects created from a prototype asset without affecting the prototype asset itself.

For more information about prototype assets, see the ["Working with Resources"](#) chapter.

ControlPoints and Curves

ControlPoint and Curve objects are not intended to be visible to the player of a game. Instead, you use them to create game play elements that do not have a visible manifestation, such as trigger volumes, and game AI path-finding.

To create an instance of any of these objects, drag it from the Palette and drop it into the design view or the Project Lister.

For more information, see the ["Working in Design View"](#) chapter.

Scripts

Advanced topic

The LevelEditor includes the IronPython programming language and an IronPython console. IronPython is an open-source implementation of the Python™ programming language designed primarily for the Microsoft .NET Framework. See ironpython.net or ironpython.codeplex.com for more information about IronPython. See www.python.org for more information about the Python language.

To open the IronPython command console, select **Window > IronPython**. You can use this window to write Python scripts to implement new commands or macros within the LevelEditor.

Integration with StateMachine

Advanced topic

StateMachine is an end-to-end solution for authoring, executing, and visually debugging hierarchical and concurrent state machines. The LevelEditor can use StateMachine projects to attach state machine assets to game objects, thus giving designers the ability to change the properties of a game object using GUI controls.

For more information about using StateMachine assets with the LevelEditor, see the ["Preparing StateMachine Assets"](#) section of the "Working with Game Objects" chapter and the ["Working with StateMachine Assets"](#) section of the "Working in Design View" chapter.

3 The LevelEditor User Interface

This chapter introduces the LevelEditor graphical user interface and describes many of the basic features of the LevelEditor.

The LevelEditor Workspace

The LevelEditor provides a workspace and tools for everything that you need to construct a game level, including a design view, resources list, project list, palette, property editor and grid property editor, output window, script window, layers list, resource metadata lister, resource preview, render settings, and bookmarks.

LevelEditor Menus

Table 1 lists the LevelEditor menus, and describes the functionality that is provided by each menu.

Table 1. LevelEditor Menus

Menu	Description
File	Use the File menu to open new or saved game levels, and to save current game levels.
Edit	Use the Edit menu to undo or redo recent actions, copy and paste game objects in the Design View, delete selected objects, select or deselect all objects in the level, group or ungroup selected game objects, lock or unlock selected game objects, change keyboard shortcuts, load or save settings, and set LevelEditor preferences.
View	Use the View menu to frame selected game objects or all game objects, hide game objects or show hidden game objects, isolate selected objects, specify the camera view, specify the layout, specify the projection, and specify the render mode.
Modify	Use the Modify menu to specify an object's pivot point, specify the currently active manipulation tool, and specify the snap behavior of an object.
Window	Use the Window menu to specify how windows should display, specify which windows are currently visible, and lock the user interface layout.
Help	Use the Help menu to display the current version of the LevelEditor.

LevelEditor Toolbars

The LevelEditor toolbars provides quick access to commonly used menu options. The toolbars are shown in *Figure 19*. These toolbars appear at the top of the LevelEditor main window; each is shown separately in the figure.

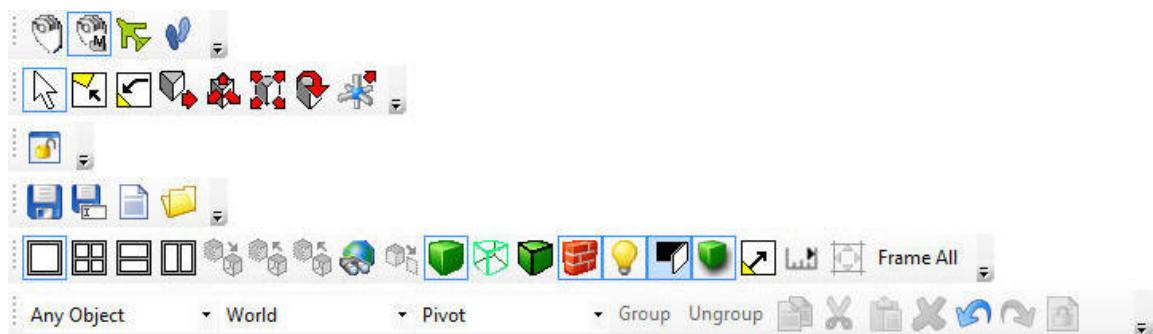


Figure 19. LevelEditor Toolbars

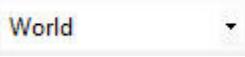
You can click a button on a toolbar to use the feature provided by that button. *Table 2* describes the buttons.

Table 2. LevelEditor Toolbar Buttons

Toolbar Button	Description
	Arcball Use this button to select the arcball navigation mode. In this mode, you use the ALT key and left-mouse button to rotate the world camera view, and you use the ALT key and right-mouse button to zoom the camera in or out. You can also scroll the mouse wheel to zoom the camera view. For arcball mode, the world rotates around the camera look-at point.
	Maya style trackball Use this button to select the Maya style trackball navigation mode. In this mode, you use the ALT key and left-mouse button to rotate the world camera view, and you use the ALT key and right-mouse button to zoom the camera in or out. You can also scroll the mouse wheel to zoom the camera view. For Maya mode, the view does not roll when you rotate the world.
	Fly Use this button to select the Fly navigation mode. In this mode, use the W, A, S, and D keys (or the arrow keys) to move the world camera view parallel to the grid and to zoom. Use the mouse wheel to adjust the fly speed. Fly mode is available only in Perspective view.
	Walk Use this button to select the Fly navigation mode. In this mode, use the W, A, S, and D keys (or the arrow keys) to move the world camera view parallel to the grid and to zoom. Use the mouse wheel to adjust the walk speed. Walk mode is available only in Perspective view.
	Select Use this button to allow mouse clicks to select game objects in the Design View. This button also disables any of the currently selected manipulators (scale, move, rotation, or move-pivot).
	Snap to vertex Use this button to snap one object to another object's vertex. Hold the SHIFT key and drag the object to the vertex.

Toolbar Button	Description
	Rotate on snap Use this button to snap one object to another object's surface. Hold the SHIFT key and drag the object to the surface.
	Move manipulator Use this button to activate the Move manipulator. The Move manipulator allows you to move game objects within the level.
	Extension manipulator Use this button to activate the Extension manipulator. The Extension manipulator allows you to extend or stretch objects in a single dimension at a time.
	Scale manipulator Use this button to activate the Scale manipulator. The Scale manipulator allows you to resize and deform game objects relative to their pivot point.
	Rotation manipulator Use this button to activate the Rotation manipulator. The Rotation manipulator allows you to rotate game objects relative to their pivot point.
	Move-Pivot manipulator Use this button to activate the Move-Pivot manipulator. The Move-Pivot manipulator allows you to move a game object's pivot point.
	Lock UI layout Use this button to lock the layout of the LevelEditor user interface so that it cannot be changed.
	Unlock UI layout Use this button to unlock the layout of the LevelEditor user interface so that it can be changed.
	Save Use this button to save the current level and all sublevels (if any).
	Save As Use this button to save the current level and all sublevels (if any) with a new file name.
	New document Use this button to open a new, empty, level.
	Open document Use this button to open an existing level.
	Frame selected objects Use this button to display the currently selected game objects in the center of the Design View. Framing an object changes the camera view, but does not change the world rotation or move the selected objects.
	Single view Use this button to show the current game level in the entire Design View.

Toolbar Button	Description
	<p>Quad view</p> <p>Use this button to show four separate views of the current game level within the Design View. You can change the projection for each of the views independently.</p>
	<p>Dual horizontal view</p> <p>Use this button to show two separate horizontal views of the current game level within the Design View. You can change the projection for each of the views independently.</p>
	<p>Dual vertical view</p> <p>Use this button to show two separate vertical views of the current game level within the Design View. You can change the projection for each of the views independently.</p>
	<p>Hide selected objects</p> <p>Use this button to hide selected objects from view.</p>
	<p>Show selected objects</p> <p>Use this button to make currently hidden selected objects visible.</p>
	<p>Show last hidden object</p> <p>Use this button to make the most recently hidden object visible.</p>
	<p>Show all hidden objects</p> <p>Use this button to make all objects visible.</p>
	<p>Show selected objects and hide other objects</p> <p>Use this button to make selected objects visible and hide all nonselected objects.</p>
	<p>Solid rendering</p> <p>Use this button to render game objects in the Design View as solid objects.</p>
	<p>Wireframe rendering</p> <p>Use this button to render game objects in the Design View as wireframe objects.</p>
	<p>Solid over wireframe rendering</p> <p>Use this button to render game objects in the Design View as solid objects with a wireframe overlay.</p>
	<p>Textured rendering</p> <p>Use this button to render game objects with their associated textures, if any.</p>
	<p>Lighting</p> <p>Use this button to use a global non-directional light source in the world. Local light sources (and shadows) are overwritten by this global light source.</p>
	<p>Render back faces</p> <p>Use this button to render the back faces of each game object.</p>

Toolbar Button	Description
	Render shadow Use this button to render shadows from local light sources in the world. The light itself is not affected by turning on or off shadows.
	Render normals Use this button to render normal vectors for each vertex of each visible object in the world.
	Realtime rendering Use this button to render objects shown in the Design View in real time
	Frame all objects Use this button to display all game objects in the center of the Design View.
	Pick-filter selector Use this button to which objects are selected when you click in the Design View. You can select Any Object , Locators , Basic Shapes , or No Cubes .
	Move selector Use this button to specify how objects move within the world using the Move manipulator. When you select World , you move objects relative to the world. When you select Local , you move objects relative to a local X, Y, or Z plane.
	Snap-mode selector Use this button to quickly set the snap point of an object to one of the following predefined settings: Pivot (the object's current pivot point), Origin , TopCenter , BottomCenter , FrontCenter , BackCenter , LeftCenter , or RightCenter . See " Snapping Game Objects " in the "Working in Design View" chapter.
	Group selected objects Use this button to group selected objects together so that you can perform actions (such as a rotation) to multiple objects at once.
	Ungroup selected objects Use this button to ungroup selected objects.
	Copy to clipboard Use this button to copy currently selected objects to the clipboard.
	Cut to clipboard Use this button to copy currently selected objects to the clipboard and remove them from the game world.
	Paste from clipboard Use this button to paste objects from the clipboard to the game world or to one of the LevelEditor windows (such as the Layers window). Pasting objects also selects the newly pasted objects.
	Delete selected object Use this button to delete the selected objects.
	Undo last change Use this button to undo the most recent change to the game world.

Toolbar Button	Description
	Redo last edit Use this button to redo the most recent change to the game world.
	Lock selection Use this button to lock the current selection so that it cannot be changed.
	Unlock selection Use this button to unlock the current selection so that it can be changed.

You can customize which buttons are displayed for each toolbar. Click the down arrow at the lower right corner of the toolbar to display the **Customize** dropdown list, as shown in *Figure 20*. From that list, select which buttons you want to display on the toolbar.

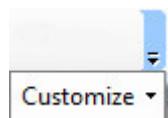


Figure 20. Toolbar customize option

The next several sections describe important panes and palettes in the LevelEditor workspace.

Design View

The Design View displays a graphical representation of the level. It is the main view that you use to lay out and construct a game level. The Design View enables you to add game objects to the level, arrange game objects within the level, and examine the level from multiple perspectives simultaneously. For example, *Figure 21* shows two perspectives of a sample level, with several game objects visible in each perspective.

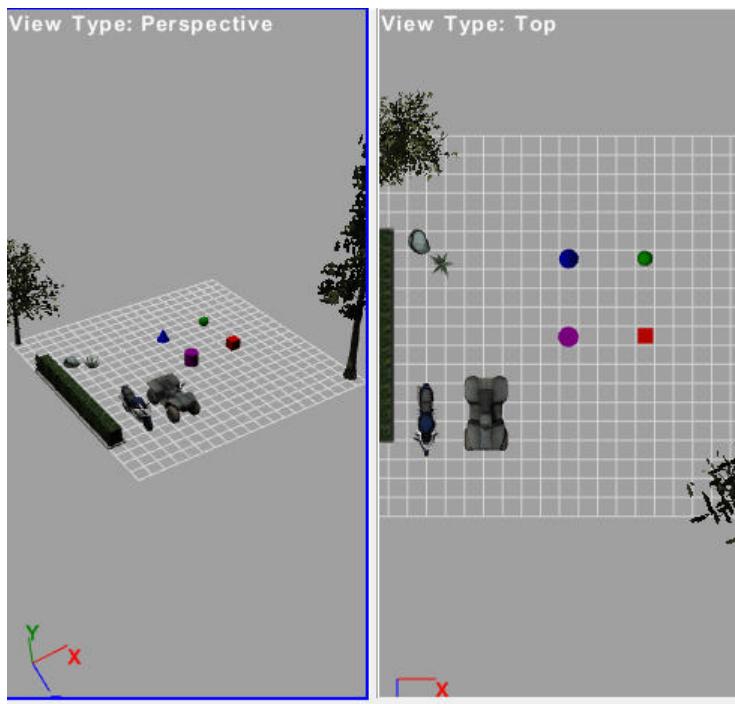


Figure 21 Design View example

The Design View can display **Perspective**, **Top**, **Bottom**, **Left**, **Right**, **Front**, or **Back** view. The grid helps you visualize the world space and dimensions. To show or hide the grid, click **Game** in Project Lister, open the Property Editor, and select the **Visible** property for the Grid. Use the Property Editor to change the size and other properties of the grid.

You can also control the way that the LevelEditor renders objects. For example you can render in wireframe, wireframe over solid, textured or untextured, and so on. For more information, see ["Specifying How to Render Game Objects"](#) later in this chapter.

Palette

The Palette contains game object types that you use to create game objects, as shown in Figure 22. The Palette contains four categories of objects:

- *Examples* contains sample game objects
- *Level Tools* contains objects for creating folders, object groups, and locators.
- *Lights* contains light source game objects.
- *Linears* contains objects for creating polylines and curves.

**Figure 22** Palette

Dragging an item from the Palette to either the Project Lister or the world in the Design View creates a new game object. If you drag an object directly to the Project Lister, the new game object is automatically placed in the world with its pivot point at 0, 0, 0. For more information about pivot points, see ["Moving, Rotating, and Scaling Objects"](#) later in this chapter.

- **To open the Palette**, select the **Window > Palette** menu item.
A checkmark appears next to the **Palette** item in the **Window** menu and the Palette becomes frontmost.
When the Palette is open, but not frontmost, you can click its tab to make it visible.
- **To close the Palette**, select the **Window > Palette** menu item again.
No checkmark appears next to the **Palette** item in the **Window** menu and the Palette is hidden.

For more information about using the Palette, see the ["Working with Game Objects"](#) chapter.

Note:

XML schemas define the objects that appear in the Palette. Creating XML schemas is an advanced topic intended for programmers, but it can be helpful to understand how the LevelEditor uses XML schemas. For more information about uses for XML schemas, see the ATF Annotation Guide in the [ATF Reference](#).



Output Window

The Output window provides diagnostic messages and other information as you work in the LevelEditor.

IronPython Window

The IronPython window allows you to write Python scripts to implement new commands or macros within the LevelEditor.

Property Editor

The Property Editor displays and allows you to edit the properties of a selected object; for example, you can specify an object's name, its rotation and translation, its visibility, and whether it is locked.

- **To open the Property Editor**, select the **Window > Property Editor** menu item. A checkmark appears next to the **Property Editor** item in the **Window** menu and the Property Editor becomes frontmost.
When the Property Editor is open, but not frontmost, you can click its tab to make it visible.
- **To close the Property Editor**, select the **Window > Property Editor** menu item again.
No checkmark appears next to the **Property Editor** item in the **Window** menu and the Property Editor is hidden.

Figure 23 shows the Property Editor display of properties for a sphere game object.



Figure 23 Property Editor example

For more information about using the Property Editor, see the ["Working with Game Objects"](#) chapter.

Grid Property Editor

The Grid Property Editor displays the properties for all currently selected objects as a grid, with each row displaying the properties of a different object.

Figure 24 shows the Grid Property Editor display for properties for several game objects.

Grid Property Editor					
Name	Locked	Visibility	Translation	Rotation	Scale
Locator	False	True	-3.8694,-0.0852,4.2664	0,0,0	5.8017,5.8017,5.8017
CylinderTest	False	True	0.4647,0.5266,0.5572	0,0,0	0.5368,0.5368,0.5368
Locator_4	False	True	-7.009,-0.5568,5.0636	0,0,0	1.8704,1.8704,1.8704

Figure 24 Grid Property example

For more information about using the Grid Property Editor to compare and set properties for multiple objects, see the ["Working with Game Objects"](#) chapter.

Project Lister

The Project Lister displays all the game objects in the level and sublevels. The directory tree shows the game object's grouping and hierarchy, as shown in Figure 25.

When you open a new level, the Project Lister contains a single folder. The **Game Objects** folder contains all game objects in the level. Sublevel folders are shown in folders at the same hierarchy level as the **Game Objects** folder.

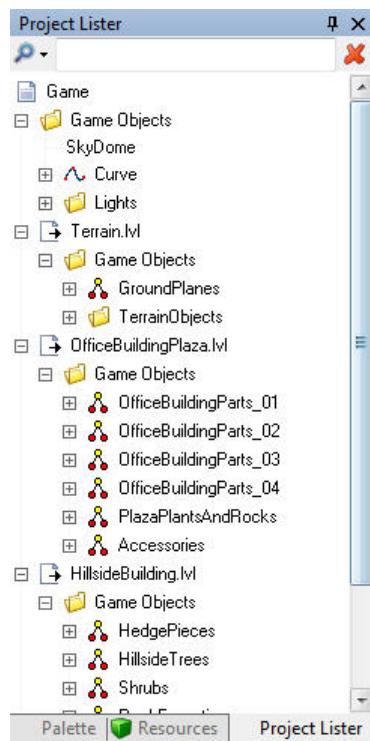


Figure 25 Project Lister example

- **To open the Project Lister**, select the **Window > Project Lister** menu item.
A checkmark appears next to the **Project Lister** item in the **Window** menu and the Project Lister becomes frontmost.
When the Project Lister is open, but not frontmost, you can click its tab to make it visible.
- **To close the Project Lister**, select the **Window > Project Lister** menu item again.
No checkmark appears next to the **Project Lister** item in the **Window** menu and the Project Lister is hidden.

For more information about game objects, see the ["Working with Game Objects"](#) chapter.

Layers List

The Layers list lets you assign objects to layers that group objects persistently. You can make objects in a layer visible or invisible by selecting or clearing the layer's checkbox in the Layers list. Thus, layers provide a handy way to reduce clutter while working with a particular set of objects. You can also use sublevels to manage large groups of objects.

Figure 26 shows an example of the Layers list.



Figure 26 Layers list example

For more information about working with layers or sublevels, see the ["Working in Design View"](#) chapter.

Resources List

The Resources list displays all of the assets in the **ResourceRoot** folder. The Resources list is visible by default, but you can choose its item in the Window menu to close and reopen it at will.

- **To open the Resources list**, select the **Window > Resources** menu item.
A checkmark appears next to the **Resources** item in the Window menu and the Resources list appears at the top of the LevelEditor workspace.
When the Resources window is open, but not frontmost, you can click its tab to make it visible.
- **To close the Resources list**, select the **Window > Resources** menu item again.
No checkmark appears next to the **Resources** item in the Window menu and the Resources list is hidden.

When the Resources list is open, its left pane displays a tree view of the contents of the **ResourceRoot** folder. The right pane displays information about the current selection in the tree view. By default, the right pane shows the Details View, which provides Name,

Size, Type and Date information. To view thumbnails instead, right-click in the Resources list, and choose **Thumbnail View** item from the contextual menu; to go back to the Details View, right click in the thumbnail view and choose **Details View** from the contextual menu.

The Resources list example in *Figure 27* shows a hierarchical view of folders and subfolders that contain resources. The **Textures** folder is selected, so the right pane displays information about its contents; in this example, the right pane displays thumbnails of the texture files in the folder.

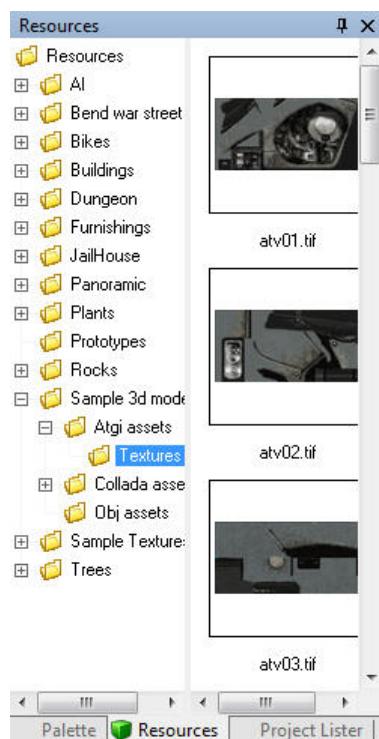


Figure 27 Resources list example

For more information about adding and managing assets, see the ["Working with Resources"](#) chapter.

Resource Preview

The Resource Preview pane allows you to view a fully rendered Resources object before you place it in the world. You can move the object within the preview using the same Arcball or Maya navigation that you use within the Design View. In addition, you can change the object's render mode and perspective.

To preview an object:

1. Open the Resource Preview pane.
2. Open the Resources list pane.
3. Within the Resources list, navigate to the object that you want to preview. You can use the Details or Thumbnail view within the Resources list to determine

which object you want to select. For example, select the **rock03.atgi** resource within the **Rocks\Rock3** folder.

4. Within the Resources list, select an ATGI object. The selected object appears fully rendered in the Resource Preview pane. For example, for the **rock03.atgi** resource, the Resource Preview displays the rendered rock, as shown in *Figure 28*.



Figure 28 Resource Preview example

You can right-click within the Resource Preview pane to alter the render mode (solid, wire frame, textured, rendered with back faces, or lit) and the view (front, back, top, bottom, left, right, or perspective) for the previewed object. These selections apply to the Resource Preview pane only, and do not affect the render mode or view within the Design View.

Resource Metadata

Many resources include metadata that you can use when selecting the object to place in the world or apply to another object. For example, artwork files typically have metadata about their compression type and memory layout.

Open the Resource Metadata pane, as shown in *Figure 29*, to view metadata for any object that you select within the Resources list. The figure shows the metadata for the **Rock03_diff.png** file in **Rocks\Rock3** folder.

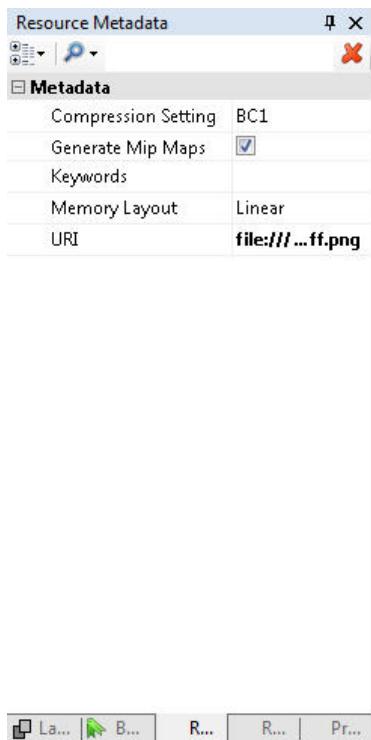


Figure 29 Resource Metadata example

Resource Root Folder

To make an asset available to the LevelEditor, place it in the **ResourceRoot** folder. The default location of the Resource Root is the folder where the **LevelEditor.exe** runs, but you can change this location.

To change the ResourceRoot, perform the following tasks:

1. Select the **Edit > Preferences** menu item.
The Preferences window appears.
2. Click the **Resources** item in the left column.
The **ResourceRoot** preference appears in the right column of the Preferences window.
3. Specify an absolute or relative path to the new **ResourceRoot** folder:
 - **To type the pathname yourself**, double-click the pathname to edit it.
 - **To browse to the new location**, select the field and click the browse button, navigate to the new location, select it, and click the **OK** button.
4. Click the **OK** button in the **Preferences** window.
The LevelEditor saves your changes and dismisses the Preferences window.
The Resources list displays the resource content of the new Resource Root.

Each level file saves the relative path to the **ResourceRoot** folder, so if you open a level



file on a different computer, be sure its path to Resource Root has the same relative structure as the original computer.

For more information about adding, managing, and updating assets, see the ["Working with Resources"](#) chapter.

Bookmarks

A *bookmark* records a specific view direction and distance for quick access in the Design View.

To list bookmarks, select the **View > Bookmarks** menu item.

To save the current view as a bookmark, right-click in the Bookmarks window and select the **Add > Bookmark** menu item.

Figure 30 shows bookmarks for several locations within the Basic sample level.



Figure 30 Bookmarks example

For more information, see ["Using Bookmarks to Save Frequently Used Views"](#) in the "Working in Design View" chapter.

Render Settings

The Render Settings window allows you to specify global preferences for how objects are rendered in the world. These preferences apply to any level file that you open in the LevelEditor.



The Render Settings window list is autohidden by default, but you can choose its item in the **Window** menu to close and reopen it at will.

- **To open the Render Settings window**, select the **Window > Render Settings** menu item.

A checkmark appears next to the **Render Settings** item in the Window menu and the Render Settings appears at the left of the LevelEditor workspace.

When the Render Settings window is open, but not frontmost, you can click its tab to make it visible.

- **To close the Render Settings window**, select the **Window > Render Settings** menu item again.

No checkmark appears next to the **Render Settings** item in the Window menu and the Render Settings window is hidden.

You can specify the following global rendering preferences:

- DisplayBound: Allows you to specify whether objects in the Design View display their bounding boxes. You can specify **None**, **Always**, or **Selection**.
- DisplayCaption: Allows you to specify whether objects in the Design View display a caption. You can specify **None**, **Always**, or **Selection**.
- DisplayPivot: Allows you to specify whether objects in the Design View display their pivot point. You can specify **None**, **Always**, or **Selection**.
- RenderShadow: Allows you to specify whether the Design View renders shadows for local lights sources by default. You can specify **True** or **False**. You can override this setting at any time by selecting the **Render shadow** option from the design toolbar.
- SelectionColor: Allows you to specify the color of the wireframe of a selected object.
- WireFrameColor: Allows you to specify the wireframe color for objects rendered as wireframes or solid with wireframes.

Figure 31 shows the Render Settings window.

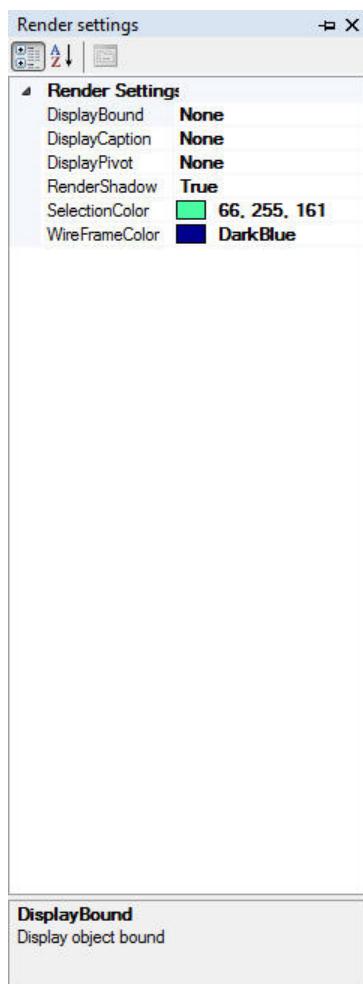


Figure 31 Render Settings window

Customizing the Default Workspace

This document shows the default workspace, but you can customize the workspace in a variety of ways:

- Drag tabs to reorder them.
- Drag a tab out into the workspace to create a floating window.
- Drag palettes to any location you prefer; to dock a palette, drag it to an edge of the workspace and drop it on the button that appears as you near the edge.
- To enable automatic hide/show of a docked window, click the push-pin icon that appears in its upper-right corner. In this mode, the window disappears when the mouse pointer is not within its boundaries. For example, to show the Render Settings window, move your mouse pointer to the location at which the Render Settings window is pinned.
- To resize a pane or a docked window, drag its edge.

- To close a window permanently, click the X that appears in its upper right corner.
- To open a window or tab that has been closed permanently, choose its name from the Window menu. You can also use the Window menu to activate (make frontmost) a tab or window that is not closed.

Figure 32 shows the LevelEditor with many of its main panes open and docked to the main window.

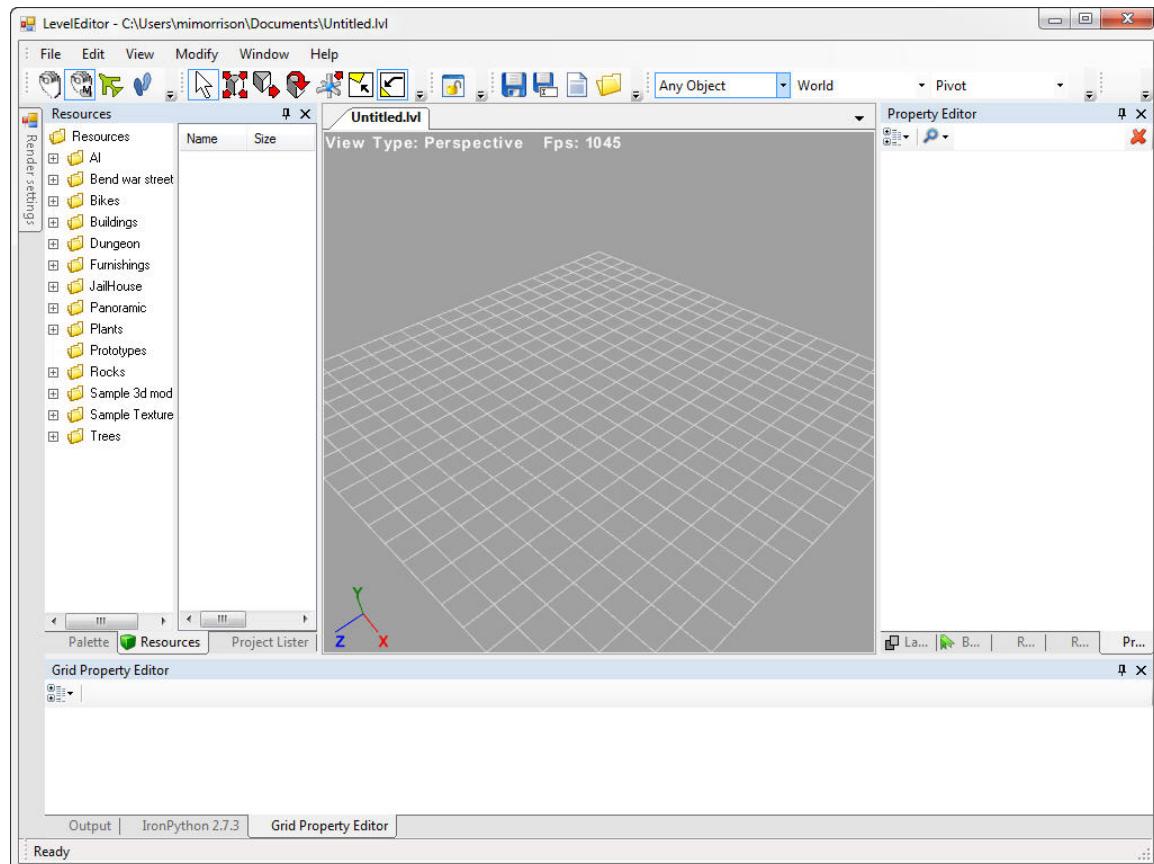


Figure 32 LevelEditor workspace

Any changes that you make to the LevelEditor workspace (such as moving and redocking windows) are saved only when you exit the LevelEditor. Workspace changes are not saved with level files, and are not saved when you save a level file.

Keyboard Shortcuts

You can customize the keyboard shortcuts for commands and actions within the LevelEditor. By default, many commands have a keyboard shortcut that allows you to perform the command without using the mouse. Each Menu item lists the keyboard shortcut defined for a command, if any.

To set or modify keyboard shortcuts, select the **Edit > Customize Keyboard Shortcuts** menu item. The Customize Keyboard Shortcuts dialog opens, as shown in *Figure 33*.

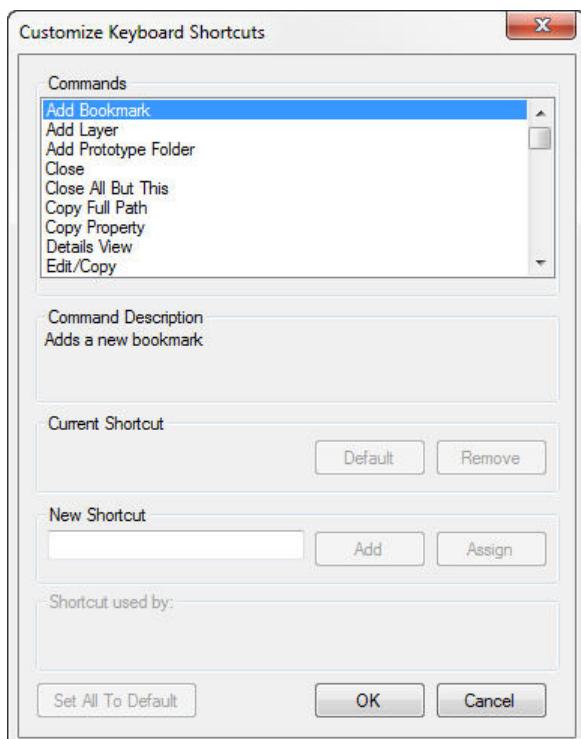


Figure 33 Customize Keyboard Shortcuts dialog

Within the Customize Keyboard Shortcuts dialog, select a command to display its currently defined keyboard shortcut. For each command, you can reset its shortcut to a default setting, remove the shortcut, or add a new shortcut.

Load or Save Settings

You can save your LevelEditor workspace settings so that you can either use multiple settings for different projects or share the settings with coworkers.

To load or save workspace settings, select the **Edit > Load or Save Settings** menu item. The Load and Save Settings dialog opens, as shown in *Figure 34*.

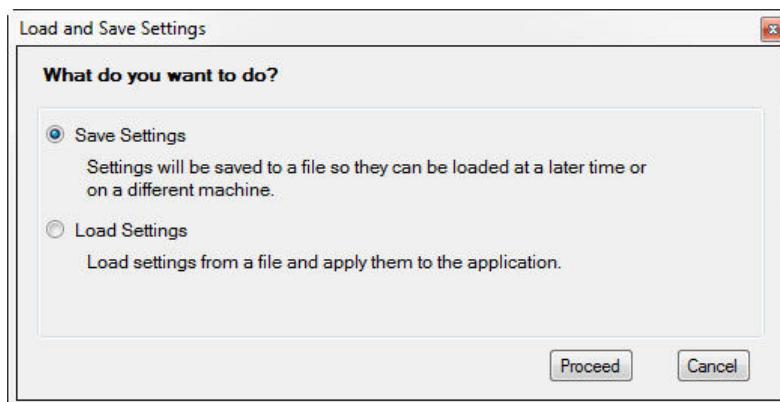


Figure 34 Load and Save Settings dialog

The workspace settings are saved to an XML file with a name that you specify.

Preferences

You can specify global settings for the LevelEditor, including:

- Application settings:
 - The size of the icons shown in the toolbars
- Document settings:
 - Whether to create a new document when you open the LevelEditor
 - Whether to autoload recently opened documents
 - The number of recent documents to display in the **File** menu
 - Whether to resolve subdocuments (sublevels) that are referenced in a document when you load it
- Resources settings:
 - The **ResourceRoot** folder; see "[Resource Root Folder](#)" earlier in this chapter.
- Editor settings:
 - The background color for the Design View
 - The default camera navigation mode
 - The camera far plane for the Design View
 - The default setting for Rotate On Snap
 - The default setting for Snap to Vertex

To specify preferences, select the **Edit > Preferences** menu item. The Preferences dialog opens, as shown in *Figure 35*.

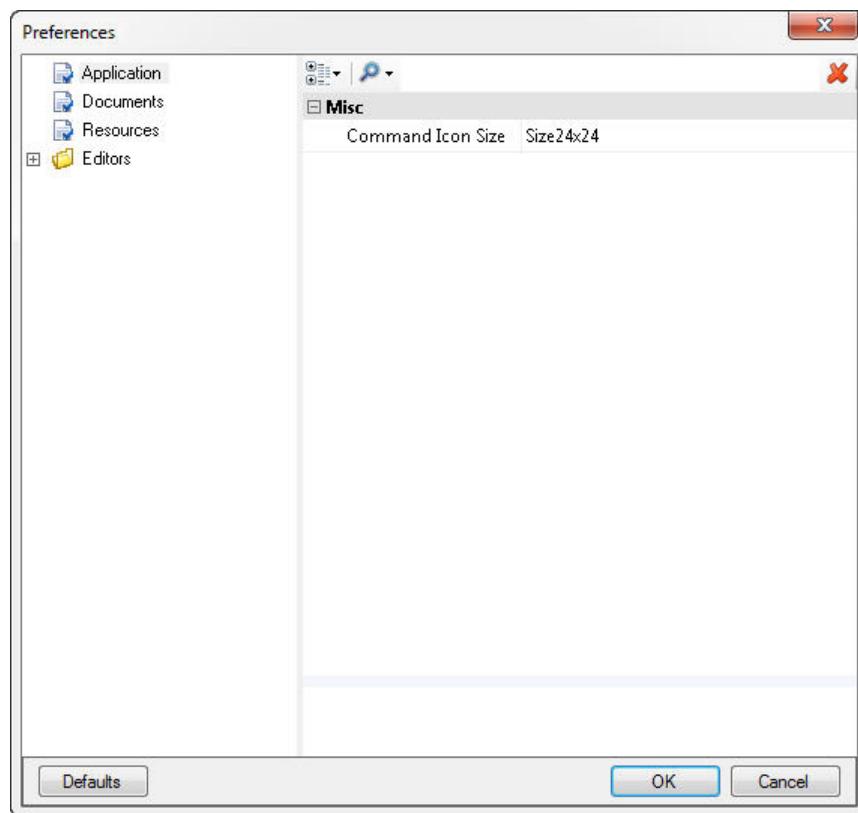


Figure 35 Preferences dialog

These preferences are saved in a default preferences file when you exit the LevelEditor. You can also save them to your own preferences file; see ["Load or Save Settings"](#) earlier in this chapter.

Using Basic Features

This section describes some of the basic features, commands, and toolbar icons of the LevelEditor to perform basic tasks.

Managing and Editing Files

The LevelEditor can save one or all documents, create subdocuments, create a new game, or open an existing game. You can also use the LevelEditor to perform basic editing tasks such as cut, paste, undo, and redo, as listed in *Table 3*.

Table 3 Managing and editing files

To...	Click this icon	Or use this command
Save the current LVL file		File > Save
Save the active file under a new name		File > Save As

To...	Click this icon	Or use this command
Create a new document (level file)		File > New Game
Open an existing document (level file)		File > Open Game
Undo the last change		Edit > Undo
Redo the last change		Edit > Redo

Setting User Preferences

Using the Preferences dialog box, you can set many aspects of the LevelEditor interface to your personal preference; see ["Preferences"](#) earlier in this chapter.

The left pane in the Preferences dialog box has a directory tree from which you can choose settings, as listed in *Table 4*.

To open the Preferences dialog box:

- Select the **Edit > Preferences** menu item.

Table 4 Setting user preferences

To set...	Choose...
Toolbar icon image size	Application > Command Icon Size
Whether the LevelEditor creates an empty document on startup	Documents > Auto New Document
Whether the LevelEditor automatically loads previously open documents on startup	Documents > Auto-load Documents
The number of recent files to display in the File menu	Documents > Recent Files Count
Whether to resolve references to subdocuments (sublevels) on document load	Documents > Resolve on Load
The ResourceRoot folder	Resources > ResourceRoot
Background color for the Design View	Editors > Design View > Background Color
The distance away from the eye point at which objects disappear from view	Editors > Design View > Camera Far Plane
The default setting for the Rotate On Snap command	Editors > Design View > RotateOnSnap
The default setting for the Snap to Vertex command	Editors > Design View > SnapVertex

Adding and Organizing Assets

The LevelEditor refers to assets by paths that are relative to the resource root. To place an asset (such as an **.atgi** file) in a level, perform the following tasks:

1. Create a new level or open an existing **.lvl** file.
The 3-D Design View displays the level. If you prefer to work in the Project Lister, click its tab.
2. Drag an asset from the right side of the Resources window into the 3D Design View or the Project Lister.

Displaying Different Views of the World

The Design View can project the world from the top, bottom, left or right side, front, back, or in three-dimensional perspective. You can also display one, two, or four of these views simultaneously, as listed in *Table 5*.

Table 5 Multiple viewing panes

To have...	Click this icon	Or choose this command
One viewing pane		View > Layouts > Single View
Two views side-by-side		View > Layouts > Dual Vertical View
Two views, one on top of the other		View > Layouts > Dual Horizontal View
Four views		View > Layouts > Quad View

To specify the projection for a view:

1. Click in the Design View.
2. Choose one of the projections listed in *Table 6*.

Table 6 Projections

To project the world...	Choose this command
In three-dimensional perspective	View > Projection > Perspective
From the front	View > Projection > Front
From the back	View > Projection > Back
From the top	View > Projection > Top
From the bottom	View > Projection > Bottom
From the left side	View > Projection > Left
From the right side	View > Projection > Right

For more information about displaying different views, see the ["Working in Design View"](#) chapter.



Specifying How to Navigate the World

The LevelEditor has several modes that let you use the mouse to zoom, rotate, and pan the world, as listed in *Table 7*. These modes—Arcball, Maya Style Trackball, Fly, and Walk—vary in what they do and how you use them.

Table 7 Navigation modes

To specify this navigation mode	Click this icon	Or choose this command
Arcball		View > Camera > Arcball
Maya Style Trackball		View > Camera > Maya
Fly		View > Camera > Fly
Walk		View > Camera > Walk

For more information about navigation mode features and uses, see the "[Working in Design View](#)" chapter.

Specifying How to Render Game Objects

You can render the surfaces of a game object as smooth, wireframe, or outlined (smooth with shading and wireframe). You can also enable or disable rendering features such as textures, lighting, whether the LevelEditor renders objects' back faces, and object normals.

Use the commands listed in *Table 8* to specify wireframes and surfaces.

Table 8 Rendering with wireframe or fill

To render game objects with	Click this icon	Or choose this command
Wireframes only		View > Wireframe
Smooth surfaces		View > Solid
Smooth surfaces with wireframes		View > SolidOverWire

Use the commands listed in *Table 9* to enable or disable rendering of textures, lighting, back faces, shadows, and normals.

Table 9 Rendering textures, lighting, back faces, shadows, and normals

To render game objects with	Click this icon	Or choose this command
Textured surfaces		View > Textured

To render game objects with	Click this icon	Or choose this command
Lighting		View > Lighting
Back sides visible		View > BackFace
Shadows		View > Shadow
Normals		View > Normals

For more information, see "[Rendering Game Objects](#)" in the "Working in Design View" chapter, or view the "Rendering Objects" video on the [LevelEditor videos page](#) on SHIP.

Moving, Rotating, and Scaling Objects

To move, rotate, or scale objects, you can use in-world manipulators, the Property Editor, or the Grid Property Editor. You can also use these controls and editors to move an object's pivot point, which is the point around which an object rotates and scales.

A manipulator is an icon that you drag to move or modify an object or a group of objects. When positioning objects, you can snap them to the grid, to a vertex on the grid, or to a point on another game object. You specify snapping points on the moved item and the target item independently.

To move or modify game objects, perform the following tasks:

1. Select game objects to modify.

To select a single object, click it in the world or in the Project Lister.

To select multiple objects, take any of the following actions:

- Hold the SHIFT or CTRL key while clicking individual objects to add them to the selection.
- Drag a selection rectangle around multiple objects.

2. Use the appropriate manipulator or edit properties, as listed in *Table 10*.

Table 10 Object manipulators

To...	Click this icon to activate the manipulator	Or edit these properties
Move an object		Translation
Rotate an object		Rotation
Scale an object		Scale

To...	Click this icon to activate the manipulator	Or edit these properties
Move an object's pivot point		Pivot
Extend or stretch an object in a single direction		Scale and Translation

For more information about positioning, rotating, and scaling objects, see the ["Working in Design View"](#) chapter, or view the "Using Manipulators" video on the [LevelEditor videos page](#) on SHIP.

Displaying and Hiding Game Objects

To make it easier to see your work in the Design View, you can hide, display, and zoom game objects. You can also assign game objects to layers or save them as sublevels to display and hide sets of game objects easily. *Table 11* lists the commands for hiding and showing objects.

Table 11 Hiding and showing objects

To...	Click this icon	Or type this command
Zoom in on selected game objects in the active viewing pane		(Keystroke command equivalent not available)
Zoom in on selected game objects in all viewing panes	(No icon available)	(Keystroke command equivalent not available)
Hide selected game objects		H
Show selected game objects		SHIFT + H
Show the last hidden object		CTRL + H
Display only selected game objects		I
Display all game objects		SHIFT + R

For more information about displaying and hiding game objects and working with layers or sublevels, see the ["Working in Design View"](#) chapter.

Creating Linears

A *linear* is a special type of game object; it defines a geometric shape that provides a set of control points. You can click and drag the control points to modify the linear.

The LevelEditor provides these tools for creating linears:

- The Palette has Polyline, CatmullRom spline, and Bezier spline objects that you

can use to lay out linear.

- Drag control points to move or bend curves and linear.
- Click on the linear to add more control points.

For more information about creating linear, see the "[Working in Design View](#)" chapter, or view the "Creating Linear" video on the [LevelEditor videos page](#) on SHIP.

Try it!

To see how to create Linear in the LevelEditor, perform the following tasks:

1. Drag a linear object type from the Palette onto the Design View.
2. Click on the linear to add control points.
3. Drag control points to manipulate them; notice how the line to either side of the control point changes in response.
4. (Optional) Open the Property Editor for the linear, and select the **IsClosed** checkbox to add a line between the end control points to close the linear.

Grouping and Ungrouping Objects

The LevelEditor can create enduring groups of objects that you can edit simultaneously. The group itself is a game object and appears as a group game object in the Project Lister. Alternatively, you can group objects temporarily by drag-selecting adjacent objects, which remain grouped until you click off the group.

- For more information about creating enduring groups of game objects, see the "[Working with Game Objects](#)" chapter.
- For more information about temporarily grouping neighboring game objects, see the "[Working in Design View](#)" chapter.

Working with Docking Controls

You can drag window tabs (or windows) to the docking controls to reposition the tab within the LevelEditor interface. The docking controls, as shown in *Figure 36*, are visible only while you are dragging a tab or window. The figure shows the main docking controls; additional docking controls appear at the top, bottom, or sides of the window as you drag the tab or window to the top, bottom, or sides of the window.

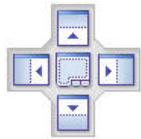


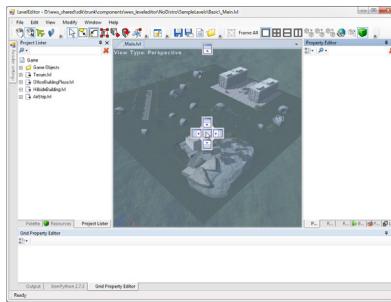
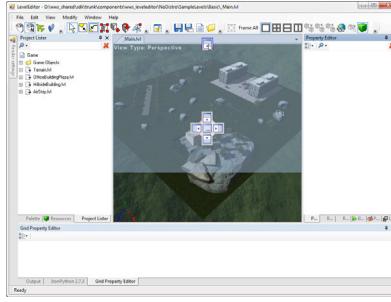
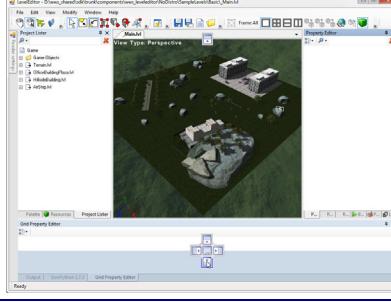
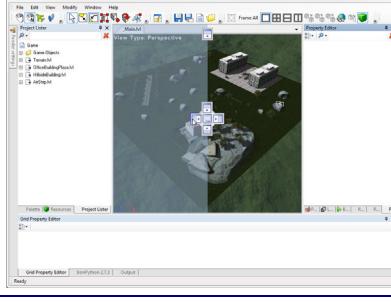
Figure 36 LevelEditor docking controls

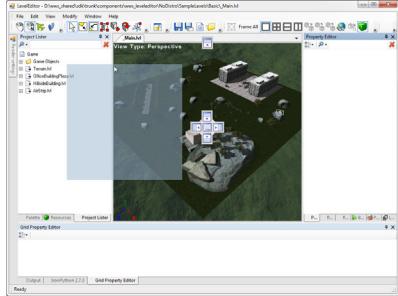
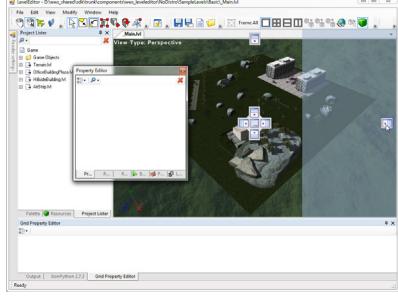
As you drag the tab, parts of the LevelEditor window are shaded light blue to represent the final location of the tab or window when you drop it.

You can drag a tab or window to many different locations within the LevelEditor work space, as described in *Table 12*. The table assumes that you are moving the various editor tabs (Property Editor, Bookmarks, Layers, Prototypes, Resource Preview and

Resource Metadata), but the table also applies to other tabs groups, docked windows, and floating windows. You can drag into any of the existing window areas, such as the Project Lister or the Output window to create a customized layout.

Table 12 Working with docking controls

To ...	Do this ...	While the docking controls look like this ...
Move editor tabs to the Design View work area	Drag one editor tab onto the Design View tab	
Move editor tabs to the top (or bottom) of the Design View work area	Drag the tab to the upper docking control within the Design View You can also drag the tab to the bottom docking control to place the window at the bottom of the Design View	
Move a tab into another window area	Drag the tab to one of the window areas, using the docking control to specify whether the tab fills the area or splits the area (vertically or horizontally) You can drag to any of the window areas	
Split the Design View area vertically	Drag the tab to the left of the central docking control You can also drag to the right of the central docking control	

To ...	Do this ...	While the docking controls look like this ...
Convert a tab to a floating window	Drag the tab to any location other than a docking control	
Convert a floating window to a tab	Drag the floating window to one of the docking controls, as described in this table	

4 Typical Work Flow

Typically, the major tasks involved in building a game level include creating resources, managing project infrastructure, constructing game levels and inspecting them, and creating the final output format of the game level.

Creating Resources and Project Infrastructure

Most complex game projects require the use of external files, such as those used as resources (also known as assets). Some games also require the creation of XML schemas that describe custom game object types to the LevelEditor. (XML schemas would be written by a programmer. For more information about writing and annotating XML schemas, see the *ATF Developers Guide*.)

Although you can add items to the Resource Root on demand, it is common for a commercial game project to be planned thoroughly before commencing the design work for which the LevelEditor is intended. Thus, the early stages of your game creation process might already have produced numerous assets that you'll use in the creation of levels for the game.

In a team environment, you might place these assets on a networked file share or use a source-control system to manage them.

Thus, before commencing work in the LevelEditor, you'll typically install resources in some appropriate location and set the location of the Resource Root on your own machine to make those resources available in the LevelEditor; see "[Resource Root Folder](#)" in the "The LevelEditor User Interface" chapter.

Constructing and Inspecting Game Levels

Most features of the LevelEditor help you create game levels and populate them with resources that implement the gaming world that a particular level provides.

Populating a game level with game objects consists of the following general tasks:

1. Add new resources to the LevelEditor as necessary. To add new resources, add files to the **ResourceRoot** folder or drag files to the Resources list. For more information, see the "[Working with Resources](#)" chapter.
2. Create game objects, associate assets with them, and set their properties. To create sets of identical game objects that have the same properties, use prototype assets. For more information, see the "[Working with Game Objects](#)" chapter.
3. Populate the level with game objects and inspect what you have constructed.

This task involves a variety of subtasks, such as placing and modifying game objects, viewing the world from different directions, and hiding and displaying game objects. For information about the ways that you work in Design View to populate the world, see the "[Working in Design View](#)" chapter.



Creating the Final Output Format

When you finish editing a level, you need to convert it to a format that is usable by your game platform, so that you can test its behaviors or distribute a final product. Typically, the game programming team writes a plugin that converts the level to a platform-specific game format. Alternatively, they can create a build process that converts the saved level file to an intermediate format or to the format used by the game.

For more information about writing plugins that convert a level to a format that a gaming platform can consume, see the *ATF Developers Guide*.

5 Working with Resources

The LevelEditor lets you bring in and manage the resources (also called assets) that you use to create a level. This chapter explains the basics about working with resources.

Using Supported Resource File Formats

A resource is an external file created in an application other than the LevelEditor. Resources can provide geometries, behaviors, textures, and similar items.

You can use almost any type of asset to create a game object: If you drag an asset to the Design View, the LevelEditor creates a Locator game object that is associated with the asset. Whether the asset is visible depends on how the LevelEditor supports the asset file type.

The LevelEditor includes plugins that support common geometry, image, and texture file formats. The only exception is for specific unsupported versions of model or texture files. For example, if the plugin for ATGI version 1_20_3 is not available to the LevelEditor, you cannot drop a version 1_20_3 ATGI file into the Design View from the Resources list.

The LevelEditor does not create thumbnails for every texture format that it supports. For example, the LevelEditor does not generate a thumbnail in the Resources list for a TIFF file, but you can still use a TIFF file as a game object texture.

Table 13 lists the types of objects that the Resources list supports. These types of objects have geometry and can be dropped into the Design View.

Table 13 Supported asset file formats

Model files	Image files	Texture files
ATGI	JPG, JPEG	JPG, JPEG
DAE (<i>Collada</i>)	BMP	BMP
	PNG	PNG
	TIF, TIFF	TIF, TIFF
	GIF	GIF
	PSD	DDS
	PCX	TGA
	PNM	
	SGI	
	PPM	

Managing Resources on Multiple Computers

Development teams typically work on a variety of computers, so it is important that the LevelEditor can locate resources regardless of their location on individual computers.

The LevelEditor refers to resources by paths that are relative to the resource root. Thus, the LevelEditor can locate resources regardless of the resources' position on a specific computer. However, the folder and file structures within the **Resources** folder on each computer must be consistent.

The default Resource Root folder is the **\Release.vs2010\Resources** folder. To set a different folder, select **Edit > Preferences > Resources > ResourceRoot**. For more information, see "["Resource Root Folder"](#)" in the "The LevelEditor User Interface" chapter.

Adding Resources to a Level

The Resources list displays all resources (also called assets) available from the Resource Root. The right pane displays a list view of the selected folder's resources or thumbnail views of them, as shown in *Figure 37*. To change between these two views, right-click in the Resources list and choose **Details View** or **Thumbnail View** from the context menu that appears.

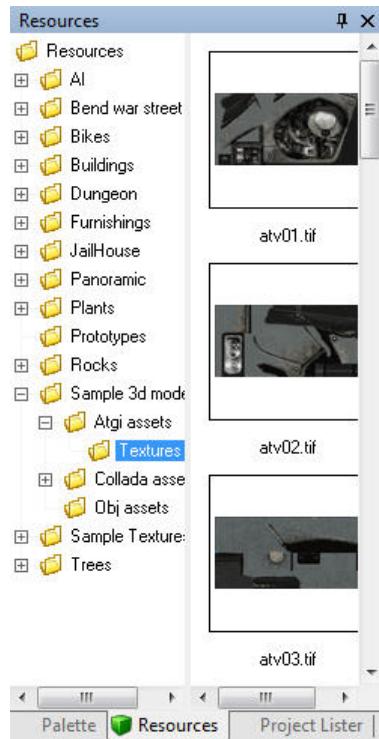


Figure 37 Resources list

You can add resource references to your level by dragging-and-dropping from the Resources list into the Design View or into the Project Lister. If a window is not visible, you can drag to the tab to bring the window to front so that you can drop into it. You can also drag-and-drop resources from Windows Explorer.

Selecting an Individual Resource

You often need to select an individual resource, such as when you associate the resource with a game object.



To select an individual resource:

1. Navigate in the directory tree to the folder that contains the resource.
The right pane displays thumbnails or a list view of the folder's resources.
2. Click an individual resource to select it.

6 Working with Game Objects

Game objects are a level's building blocks. There are many different types of game objects: game characters, static background objects, and so on. The LevelEditor provides tools for working with game objects to build a level.

This chapter describes the ways you can work with game objects using the LevelEditor.

Viewing Game Objects in the Project Lister

The Project Lister displays every game object in a level or sublevel, as shown in *Figure 38*. Expanding a game object in the Project Lister shows the hierarchy of resources associated with the game object. Some game objects, such as a GameObjectGroup, might have other game objects as children. These, too, can be viewed by expanding the parent game object.

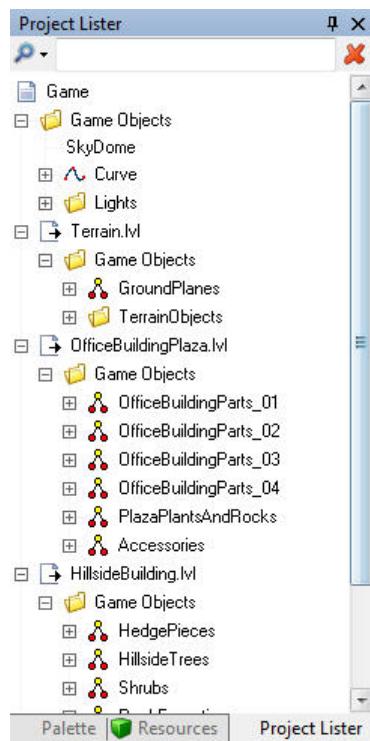


Figure 38 Project Lister

XML schemas define the available game object types, including the game object's properties and the resources with which a game object can be associated. These XML schemas are written by programmers and should be available before you start building a level.

Creating Game Objects

To create a game object manually, drag it from the Palette or the Resources list into the Design View or the Project Lister.

- Dragging an object directly to Design View creates a game object at the point where you drop it.
- Dragging an object directly to the Project Lister creates a game object centered at 0, 0, 0 on the grid.

The object that you drag creates a game object that has the resource slots already defined for that type of game object. For example, the Orc object in the Palette creates a game object that has resource slots for associating a geometry resource, an animation, an additional game object as a target, and additional game objects as friends.

If you drag a resource from the Resources list, the LevelEditor creates a Locator game object that has that resource associated with its sole resource slot, as described in *Table 14*.

Table 14 Creating game objects

To create a game object	Do this...
Positioned at a specific point in the world	Drag an object to the desired location in the Design View.
Positioned at 0,0,0	Drag an object to the Games Objects Lister.

Using the Palette as a Source of Game Objects

The Palette contains example objects from which you can create game objects:

- SkyDome – a 360° background image for the world
- CubeTest – a simple cube
- SphereTest – a simple sphere
- ConeTest – a simple cone
- CylinderTest – a simple cylinder
- PlaneTest – a simple plane
- BillboardTest – a simple billboard, onto which you can place an image that always faces the viewer, regardless of camera view
- Orc – a simple object with properties that are representative of game creatures
- TerrainTest – a simple terrain object

Each object creates a game object of a specific type with predefined properties and slots for resources. The types of objects available in the Palette are defined in XML schemas.

Typically, you use Palette objects as the starting material for building game objects. After you create a game object from a Palette object, you associate resources with it and set its properties, for example:



- For the cube, sphere, cone, and cylinder, you can set the object's properties (such as its color), as well as scale the object, to create interesting and useful game objects.
- For the plane, you might want to turn on **Render back faces** to be able to see both sides of the plane as you change the camera view. The plane can be useful as a wall, floor, or ceiling surface onto which you add a texture.
- For the billboard, you can add a texture (as its **Diffuse** property) and its brightness (its **Intensity** property).
- For the orc, you can associate a geometry object, an animation, a target, and friends, as well as set its various properties.
- For the terrain, you associate a height map to give the object its base geometry, then associate a base layer (and up to 4 additional layers) of texture to define a terrain object in the world.

The Palette also provides tools for organizing your project or levels within the project:

- **GameObjectFolder** – a folder for grouping objects in the Project Lister; objects within the folder are not grouped in the world, only in the Project Lister (and the level file XML)
- **GameObjectGroup** – a meta-object that allows you to work with several objects as a group
- **Locator** – a meta-object that acts as a container for a resource (see ["Using Locators to Create Game Objects"](#) later in this chapter)

Finally, the Palette provides Light objects (see ["Adding Light Objects"](#) in the "Working in Design View" chapter) and Linear objects (see ["Using Linears to Lay Out Lines and Boundaries"](#) in the "Working in Design View" chapter).

Using Prototypes to Create Similar Game Objects

Often, you want to create a large number of identical objects: a set of cars, a battalion, or trees in a forest. Prototype assets provide a convenient way to make multiple objects that are identical.

To create a prototype asset, you must first create one or more game objects, set properties for each object, and associate resources with them. You then create a prototype asset from the game objects, and use the prototype asset as a template.

To create a prototype asset:

1. Select one or more existing game objects in the Design View.
2. Select **File > Create Prototype** to save the selected objects as a prototype asset. The Save As dialog opens so that you can define the prototype asset's file name and folder location. Save the prototype asset within your Resource Root folder.

Recommendation: Define a folder named "Prototypes" within your Resource Root folder to store your prototypes. Define subfolders within this new "Prototypes" folder as needed for each type of prototype asset.

To use the prototype asset to create additional instances of a game object or set of

game objects:

1. Open the Resources list.
2. If you store your prototype assets within a **Prototypes** folder, select that folder in the Resources list. If you use a different folder for prototype assets, select that folder.
3. From the right-hand pane of the Resources list, select the prototype asset. You can use the Resource Preview window to display the fully rendered view of the prototype asset.
4. Drag the prototype asset to the world. The copy of the prototype asset appears as a game object in the world and is added to the Project Lister.

Game objects created from prototype assets appear in the Project Lister as `GameObjectGroup` objects, with a `Locator` child object for each unique object within the prototype asset. You can work with each object individually by selecting the object; to work with the objects as a group, select the `GameObjectGroup` (its name is the same as the prototype asset) in the Project Lister.

After you create the prototype asset, the LevelEditor treats it as any other game resource or asset. You can place as many of the prototype assets in the world as you need, and you can edit the properties of any these newly placed objects from the prototype asset, without affecting the base prototype asset itself. Likewise, if you modify the prototype asset after you use it within a level, the changes to the prototype asset do not affect any objects in a level that are based on that prototype asset.

Using Locators to Create Game Objects

A Locator is useful for placing an instance of a geometry resource into the level. Like all game objects, a Locator has a position. However, the only additional property it has is a resource slot.

To create a locator, perform any of the following tasks:

- Drag a resource to Design View.
The LevelEditor creates a locator game object. The point at which you drop the locator becomes the locator game object's position. Pressing the SHIFT key while you drag snaps the locator game object to another object in the level. Pressing the CTRL key while you drag snaps the locator game object to the grid.
- Drag a locator object from the Palette to Design View.
- Drag a locator game object from the Palette to the Project Lister.

Associating Resources with Game Objects

Many game objects have slots for associating assets with the game object. You associate an asset with a game object by dragging the asset to the appropriate slot for the game object in the Project Lister.

The types of assets that a slot can accommodate depend on how the slot was set up; you can associate only assets of a type that the slot can accept. For example, the geometry slot for the Orc example can accept only geometry assets. If you drag a behavior or texture asset to the geometry slot, the LevelEditor does not accept the asset.



- **To associate an asset with a slot**, drag the asset to the game object's slot in the Project Lister. You can overwrite an existing asset by dragging a new asset to the slot.
- **To remove an asset from a slot**, select the asset in its assigned slot and select the **Edit > Delete** menu item.

Working with Game Object Properties

You can view and edit game object properties in either the Property Editor or the Grid Property Editor.

Different types of game objects can have different properties, thus the properties that appear for a specific type of game object can vary. However, the following properties are typical for game objects:

- *Name* specifies the object name.
- *Visibility* specifies whether the LevelEditor displays the object in the Design View.
- *Locked* specifies whether the object properties are locked and unavailable for editing.
- *Display Properties* specifies additional information that appears when the object is selected. When selected, the options have the following effect:
 - *Box* displays the bounding box.
 - *Pivot* displays the pivot point.
 - *Caption* displays the name.
- *Translation* specifies the object's X, Y, and Z coordinates in the world. You can reposition an object by editing the X, Y, and Z values.
- *Rotation* specifies how much the object is rotated around its pivot point.
- *Scale* specifies how much the object size is scaled in X, Y, and Z dimensions. The Scale value is 1, 1, 1 when the object is its original size.
- *Pivot* specifies the offset, in the object's local coordinate system, from the object's origin to the pivot point. The Rotate Manipulator rotates the object around this pivot point. The Scale Manipulator scales the object around this pivot point.

Using the Property Editor to View and Edit Properties

The Property Editor displays properties of the selected game object. If multiple game objects are selected, the Property Editor displays only properties that are common to all selected game objects. The values shown are those of the most-recently selected game object. Editing a value modifies all selected game objects.

The Property Editor example in *Figure 39* shows available properties for a sphere game object.

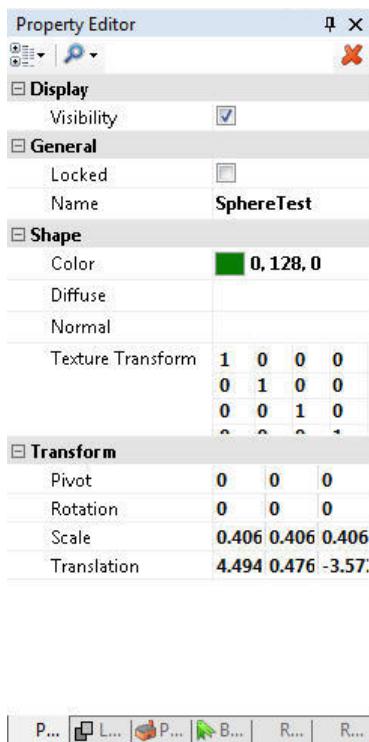


Figure 39 Property Editor

Using the Grid Property Editor to View and Edit Properties

The Grid Property Editor displays the common properties that are available for a set of game objects. The properties are displayed in a grid that makes it easier to compare different property values for a selection of game objects.

Figure 40 shows a Grid Property Editor that displays the properties for several selected game objects.

Name	Locked	Visibility	Translation	Rotation	Scale
Locator	False	True	-3.8694,-0.0952,4.2664	0,0,0	5.8017,5.8017,5.8017
CylinderTest	False	True	0.4647,0.5266,0.5572	0,0,0	0.5368,0.5368,0.5368
Locator_4	False	True	-7.009,-0.5568,5.0636	0,0,0	1.8704,1.8704,1.8704

Figure 40 Grid Property Editor

Locking Game Objects

Locking a game object ensures that other developers cannot accidentally edit its properties. The ability to restrict unintentional edits can be crucial for distributed work flows. To lock a game object's properties, set its **Locked** property to **True**.

To lock a game object's properties:

1. Select the object.

2. Click the Lock icon ().

—Or—

Set the object's **Locked** property to **True**.

To unlock a game object's properties:

1. Select the object.

2. Click the Unlock icon ().

—Or—

Set the object's **Locked** property to **False**.

Preparing StateMachine Assets

States, transitions, and state machines can have associated properties that you set within the StateMachine Editor. Objects can have additional properties defined that are accessible to other tools, such as the LevelEditor. The LevelEditor can read StateMachine Editor project files (**.stmpobj**) directly.

Within the StateMachine Editor, each game object has a category in the **Properties** window that lists its properties in the state machine project. Each property has a control for changing its value to refine the object's behavior. For example, the aggression level of a soldier could be adjusted using a slider bar. This aggression level becomes the default state for game objects in the LevelEditor that use the soldier state machine. Within the LevelEditor, you can then modify the default state properties for each instance of the game object.

Using StateMachine and LevelEditor together in this way, designers can quickly test and iterate their designs without needing programmer intervention. For more information about setting properties, see the *StateMachine Editor User's Guide*.

To run state machines in the LevelEditor, perform the following tasks:

1. In your state machine designs, create instance properties using the StateMachine Editor **Properties** window or **Property Table** window. These properties will be visible and modifiable in the LevelEditor.
2. Copy StateMachine Editor project files (**.stmpobj**), state files (**.stm**), and Lua scripts (**.lua**) to the LevelEditor Resource Root folder.

The designer can then work with the LevelEditor, as described in the ["Working with StateMachine Assets"](#) section in the "Working in Design View" chapter.

7 Working in Design View

The Design View is your workspace for constructing and inspecting a complete game level. You can use it to view the scene from different directions, to position and modify game objects, and to inspect the assembled scene.

The LevelEditor also allows you to display and hide game objects, control how game objects are rendered, and build linears quickly.

This chapter describes tasks you can perform in the Design View to build a scene.

Displaying Different Views of the World

Design View can have one, two, or four views open simultaneously:

- *Single View*  displays one view in one pane.
- *Quad View*  displays four views arranged in quadrants.
- *Dual Horizontal*  displays two wide views, one on top of the other.
- *Dual Vertical*  displays two tall views, side by side.

When multiple views are open, the LevelEditor divides the Design View into separate viewing panes. The pane that was clicked last is the active one and has a blue border. Each viewing pane can use one of the following projections to display the world:

- *Perspective* displays the world as a three-dimensional space.
- *Front* and *Back* display the world in the X-Y plane.
- *Top* and *Bottom* display the world in the X-Z plane.
- *Right* and *Left* display the world in the Y-Z plane (side view).

Note: The Design View shows a set of axes in the lower left corner of each view to represent the current orientation of the X, Y, and Z axes.

Figure 41 shows the Design View in Dual Vertical display. The left pane's projection is in perspective; the right pane's projection is from the top, looking down at the X-Z plane along the Y-axis. The left pane has a blue border because it is the active pane.

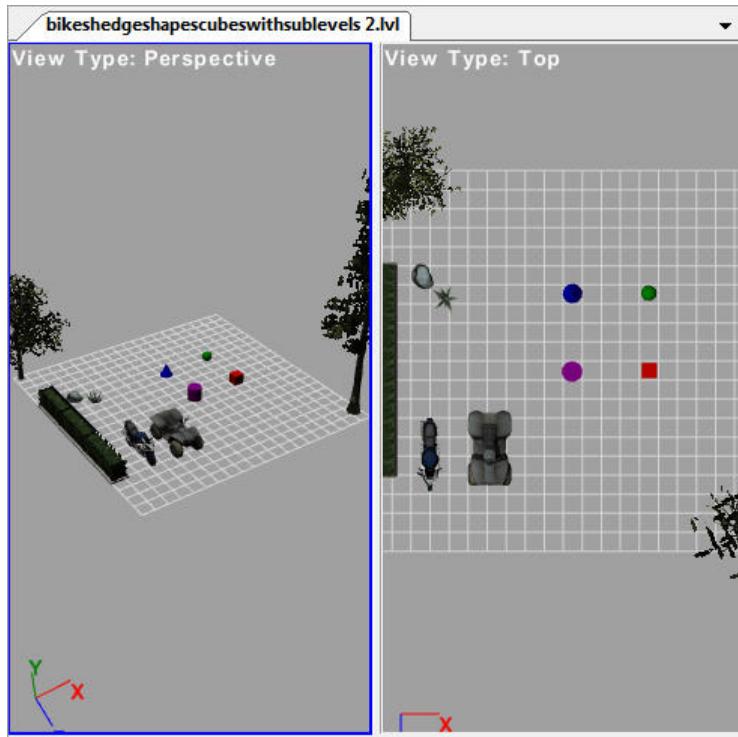


Figure 41 Design View example

To specify how many viewing panes the Design View displays:

- Select the **View > Layouts** menu item, then choose a layout from the submenu.

To specify the view projection within an individual pane:

1. Click in the pane to make it active.
2. Select the **View > Projection** menu item, then choose a projection from the submenu.

—Or—

Right-click in the pane, select **Projection** from the pop-up menu, and then choose a projection from the submenu.

Advanced Topic

If you think of the Design View world in terms of observation points, the following factors determine the Design View display:

- The *eye point* is the point from which you are viewing the world when you look at Design View.
- The *camera look-at point* is the point in the center of the viewing pane and is a specific distance from the eye point.
- The *look-at distance* is the distance from the camera look-at point to the eye point. Changing the look-at distance makes the world zoom in and out.
- The *look-at direction* is the direction from the eye point to the camera look-at

point.

Moving the eye point and camera pivot-point while maintaining their relative positions pans the world.

Navigating in the Design View

You can adjust Design View to display the world from different directions and distances. The ways that you can navigate in Design View depend on the navigation mode:

- *Arcball* mode  lets you drag freely to rotate the world, zoom in and out, and pan. In Arcball mode, the world rotates around the camera look-at point.
- *Maya Style Trackball* mode  behaves the same as Arcball mode, except that the view does not roll when you rotate the world.
- *Fly* mode  lets you pan left and right, and zoom in on and out of the world. You can pan past any game objects that appear between the eye point and the camera look-at point. Fly mode is available only in Perspective view.
- *Walk* mode  lets you pan left, right, forward, and back in the world while staying in a plane parallel to the grid. Walk mode is available only in Perspective view.

To specify a navigation mode:

1. Click in the pane to make it active.
2. Select the **View > Projection** menu item, and choose the projection that suits your navigation needs.
3. Click one of the four navigation mode buttons in the navigation toolbar.

—Or—

Select the **View > Camera** menu item, and select one of the four navigation modes.

Zooming the World Using the Mouse

You can use the mouse to zoom in on and zoom out of the world. The way that you use the mouse to zoom in or out depends on the navigation mode. The zoom is toward or away from the center of the active viewing pane, and is possible from any viewing angle. For a demonstration of how to zoom the world, view the "Zooming the World" video on the [LevelEditor videos page](#) on SHIP.

To zoom in on specific game objects, use the **Frame** and **Frame All** commands. For information about zooming specific game objects, see ["Zooming Specific Game Objects"](#) later in this chapter.

To zoom in or out of the world:

1. Click in the viewing pane to make it active.



2. Take one of the actions listed in *Table 15*, depending on the navigation mode.

Table 15 Zooming the World

In this mode	Zoom in this direction	By doing this...
Arcball	In	Pressing ALT and the right mouse button while dragging down or right. —Or— Rolling the middle mouse button down.
	Out	Pressing ALT and the right mouse button while dragging up or left. —Or— Rolling the middle mouse button up.
Maya	In	Pressing ALT and the right mouse button while dragging down or right. —Or— Rolling the middle mouse button down.
	Out	Pressing ALT and the right mouse button while dragging up or left. —Or— Rolling the middle mouse button up.
Fly	In	Pressing the W key or Up arrow key.
	Out	Pressing the S key or Down arrow key.
Walk	In	Pressing the W key or Up arrow key.
	Out	Pressing the S key or Down arrow key.

In both Fly and Walk mode, use the mouse wheel to adjust the zoom speed.

Rotating the World

You can drag to rotate the world around the camera look-at point by pressing ALT and the left mouse button while dragging:

- Dragging left and right rotates the world around a vertical axis through the camera look-at point.
- Dragging up and down rotates the world up and down over a horizontal axis through the camera look-at point.

You can also rotate the world around the eye point in Fly or Walk navigation mode by pressing the left mouse button while dragging. This gives the illusion that the world is moving around you as you rotate your head.

How the world rotates depends on the distance to the camera look-at point:

- At a shorter distance, the world rotates around a point that is closer to you. This is useful for inspecting nearby objects from various directions.
- At a longer distance, the world rotates around a point that is farther away. This is



useful for rotating an entire scene to see large sections of it from different directions.

Clicking a different point in the world resets the look-at distance. The distance from the eye point to the point you click becomes the new look-at distance.

Note:

The point you click does not become the new camera look-at point. The camera look-at point is always in the center of the screen. However, clicking points that are nearby or far away changes the look-at distance.

When you click to reset the look-at distance, you can click without selecting the object at the cursor.

To rotate the world around the camera look-at point:

1. Choose Arcball or Maya Trackball navigation mode.
2. Press ALT and the left mouse button while dragging the world.

To rotate the world around the eye point:

1. Choose either Fly or Walk navigation mode.
2. Press the middle mouse button while dragging the world.

To change the look-at distance:

- Click a point that is at a distance similar to what you want for the look-at distance.

To change the look-at distance without selecting an object when you click:

- Press ALT while clicking the point.

Panning the World

When the Design View is in Perspective view, you can pan the world. The way that you pan depends on which navigation mode is selected. *Table 16* summarizes the ways that you can pan in different navigation modes.

Table 16 Panning in different navigation modes

In this mode	Pan in this direction	By doing this...
Arcball	Vertically	Pressing the middle mouse button while dragging up and down.
	Horizontally	Pressing the middle mouse button while dragging left and right.
Maya	Vertically	Pressing the middle mouse button while dragging up and down.
	Horizontally	Pressing the middle mouse button while dragging left and right.
Fly	Toward or away from the center point	Pressing the W and S keys or the Up and Down arrow keys.
	Horizontally	Pressing the A and D keys or the Left and Right arrow keys.

In this mode	Pan in this direction	By doing this...
	Up and down	Pressing ALT and the middle mouse button while dragging up and down.
Walk	Forward and back in a plane parallel to the grid	Pressing the W and S keys or the Up and Down arrow keys.
	Horizontally	Pressing the A and D keys or the Left and Right arrow keys.
	Up and down	Pressing ALT and the middle mouse button while dragging up and down.

Using Bookmarks to Save Frequently Used Views

Bookmarks save your favorite views of the world so that you can return to them quickly. When you click a bookmark in the Bookmark window, the Design View (or the currently selected pane of the Design View) displays the world from the direction and distance saved in the bookmark.

Figure 42 shows bookmarks for the Basic sample level.



Figure 42 Bookmarks window

For the Basic sample level, the Overhead view shows the entire level. The other bookmarks are views of specific locations in the level.

You create bookmarks by navigating to the view you want, opening the Bookmarks window, right-clicking in the window and selecting **Add Bookmark**. The default bookmark name is "New bookmark," which you can rename. To delete a bookmark, right-click the



bookmark and select **Delete**.

Existing bookmarks cannot be edited: You create a modified version of a bookmark by modifying the view, then creating a new bookmark for that view.

To save the current view as a bookmark, right-click in the Bookmarks window and select **Add Bookmark**.

To rename a bookmark, click the bookmark's name in the Bookmarks window, and then edit the name.

To go to the view saved in a bookmark:

1. Select the **Window > Bookmarks** menu item to open the Bookmarks window.
2. Click the bookmark to view.

To delete a bookmark:

1. Select the bookmark.
2. Select the **Edit > Delete** menu item.

Alternatively, you can right-click the bookmark and choose **Delete** from the contextual menu. Yet another approach is to select the bookmark and press the **DELETE** key.

Placing Game Objects in the World

You can create game objects from any object in the Palette or Resources list by dragging the object to the Design View.

The LevelEditor lets you place the object at any point in space or at a specific point on the grid or on another game object. For example, you can stack a group of boxes on top of each other, lay out a road over a curving hillside, or position one object within another.

When you drag an object to the Design View and release the mouse to drop the object, the LevelEditor places the object close to the top of any existing object nearest to the mouse pointer. If there is no existing object at the mouse pointer when you drop the object, the LevelEditor places the object in space at a distance of approximately $6*R$ from the mouse pointer location, where R is the radius of the object.

The game object's Translation properties listed in the Property Editor are the game object's X, Y, and Z coordinates.

To move a game object that is already in the world, drag its Move manipulator or edit its Translation properties in the Property Editor. For more information about moving game objects, see ["Moving Objects"](#) later in this chapter.

Placing an Object at any Point in the World

To place a new object manually at any point in the world:

- Drag an object from the Palette or Resources list and release it at the desired location. The object is added to the top-most **Game Objects** folder.

To add a new game object to a specific sublevel (rather than to the top of the game object hierarchy), you can perform either of the following tasks:

- Drag the object to the Project Lister and drop it within the **Game Objects** folder of the specific sublevel. The object is placed at location 0,0,0 of the world, but is included within the sublevel.
- Select an object that is already within the sublevel, then drag and drop a new object anywhere within the world. The object has the location you specify, and is also included within the sublevel.

See "[Using Sublevels to Manage Large Game Worlds](#)" later in this chapter for more information about working with sublevels.

Snapping Game Objects

If you are snapping objects, the **Snap Mode** combo box () specifies which part of the moving object snaps to the target object surface.

You can specify one of the following snap points:

- *Pivot* snaps the pivot point.
- *Origin* snaps the origin.
- *TopCenter* snap the center point of the top.
- *BottomCenter* snaps the center point of the bottom.
- *FrontCenter* snaps the center point of the front.
- *BackCenter* snaps the center point of the back.
- *LeftCenter* snaps the center point of the left.
- *RightCenter* snaps the center point of the right.

Pressing CTRL while dragging constrains the Snap Mode point to the grid.

Pressing SHIFT while dragging constrains the object to the surface of the target object. If Rotate on Snap is turned on, the moved object also rotates so that its Up axis is parallel to the target object's normal axis. This feature is unavailable for some proxy objects because they do not have surface normals.

Figure 43 shows how Rotate on Snap rotates objects when they are snapped to another object. In both cases, the vehicle object was snapped to a vertex on the model. Because Rotate on Snap was turned on, the LevelEditor rotated the object so that its Up axis was kept normal to the target object's surface.

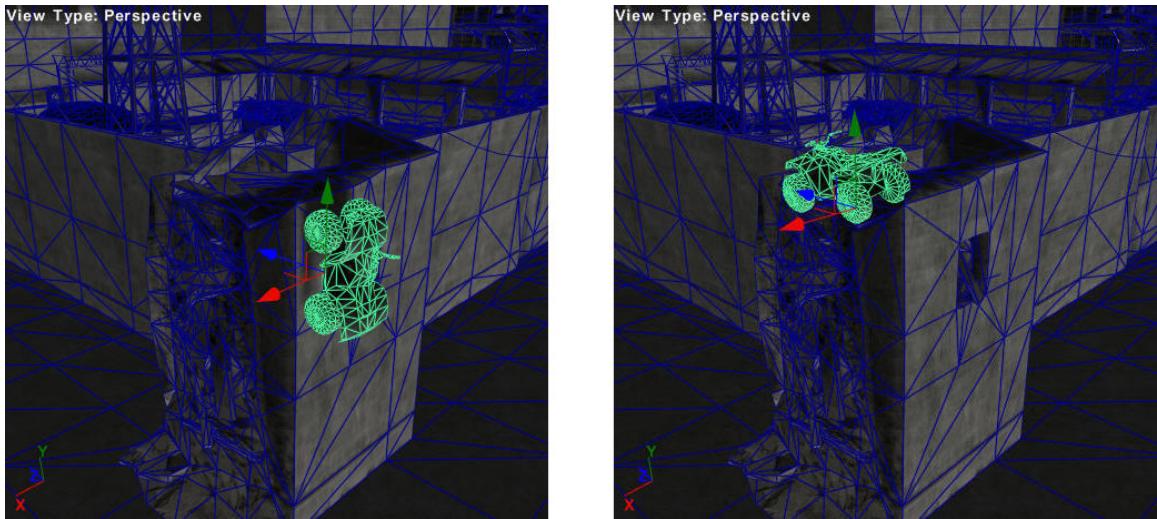


Figure 43 Rotate on Snap example

You can also specify where on the target object that snapping occurs:

- Turning on Snap to Vertex () aligns the snap point to a vertex on the target object.
- Turning on Rotate On Snap () aligns the snap point to the surface of the target object.

For example, if you specify **Bottom Center** for Snap Mode and turn on Rotate On Snap, you can precisely stack an object on top of another by pressing SHIFT while dragging the object on the surface of the target object.

Snapping a Game Object to the Surface of Another

For more information about snapping existing game objects, see ["Moving Objects"](#) later in this chapter.

To snap a game object to the surface of another:

1. In the **Snap Mode** combo box, specify which point of the moving object you want to snap to the target.
2. Press SHIFT while dragging the object to the target object.

To snap a game object to a vertex of another object:

1. In the **Snap Mode** combo box, specify the point that you want to snap to the target.
2. Turn on Snap to Vertex.
3. Press SHIFT while dragging the object to the target object.

To rotate the moving object so that its Up axis is parallel to the normal axis of the target object when it snaps to a vertex:



- Before snapping the object to the vertex of another object, select the **Modify > Rotate on Snap** menu item to turn Rotate on Snap on.

Selecting Game Objects

Using the mouse, you can select a group of game objects, a single game object, or a component within a single game object.

Select a group of game objects by drawing a selection rectangle. All objects whose bounding boxes fall within the rectangle are selected.

Select a single game object or a component within a game object by clicking it.

To select a set of neighboring objects, draw a rectangle in the area that includes the objects.

Controlling Which Objects Are Selected

You can control which types of objects are selected when you click in the Design View or use the mouse to select multiple objects in the Design View. The **Pick Filter** combo box

(**Any Object**) allows you to set a filter for what the LevelEditor selects when you click within the Design View:

- *Any Object* allows you to select any game object. This selection is the default and is equivalent to setting no filter.
- *Locator* allows you to select only locator objects.
- *Basic Shapes* allows you to select only basic shapes.
- *No Cubes* allows you to select any type of game object except those based on the CubeTest example object from the Palette.

With a pick filter active, if you click on an object that is not included within the filter, your click is passed to objects behind or within that object. For example, if you specify the “Basic Shapes” pick filter, and click on a linear that is above a cube, the cube is selected rather than the linear; the linear is not selectable. Using the filter allows you to select one or more basic shapes to work with them (for example, to rotate them) without affecting the linear (or any other object that is not a “basic shape”) that is near these objects.

These pick filters are provided as examples; a programmer can add additional pick filters that are specific to your game development project. For example, you might want to add a filter for “No Terrain Objects” so that you can easily click objects on a terrain without ever accidentally selecting the terrain itself.

Selecting Obscured Objects

The pick-cycling feature enables you to select an object that is obscured by another object. It is intended to help you select an object that intersects another object, or that is hidden inside, behind, or beneath another object.

To use the pick-cycling feature, click repeatedly on the obscured object in the Design View without moving the mouse. Each click selects a different object beneath the cursor in turn, one at a time.

Recommendation: Turn off any currently active manipulator (for example, the Move manipulator) for objects near the cursor to ensure that each click selects an object rather than a manipulator.

Moving, Rotating, and Scaling Objects

You can use the LevelEditor to move, rotate, and scale objects in the following ways:

- By dragging their Move, Scale, Rotation, and Extension manipulators
- By setting their Translation, Scale, and Rotation properties in the Grid Property Editor

For a demonstration of how to move, rotate, and scale objects, view the “Using Manipulators” video on the [LevelEditor videos page](#) on SHIP.

Using Pivots to Control Rotation and Scaling

The LevelEditor scales and rotates an object around its pivot point. The default pivot point is at the object’s origin, but you can reposition the pivot point by dragging its Move-Pivot manipulator, using the Move Pivot command, or editing the **Pivot** property in the Property Editor.

The cube in *Figure 44* has its pivot point positioned away from the object’s origin and placed outside the cube.

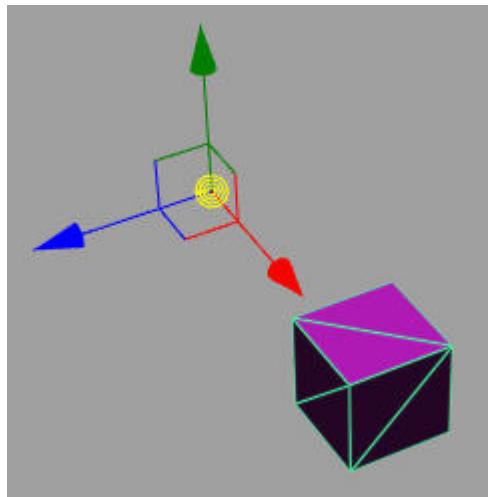


Figure 44 Cube with repositioned pivot point and Move-Pivot manipulator active

For grouped game objects, the group has its own pivot point, which is independent of the pivots for the individual game objects that belong to the group.

For a demonstration of how you can use pivot points to control scaling and rotation, view the “Moving Pivot Points” video on the [LevelEditor videos page](#) on SHIP.

To position an object’s pivot point using the mouse:

1. Select the **Modify > MovePivot** menu item to turn on the Move-Pivot manipulator.

2. Select the object whose pivot point you want to move.
3. Drag a Move-Pivot manipulator axis to move the pivot point in that direction. Drag a shaded (yellow) square to move the pivot point freely in two dimensions.

To position an object's pivot point using the Move Pivot command:

- Use the **Modify > Move Pivot > X** menu item to set the object's pivot point along the X axis:
 - **Min** sets the pivot point to the object's minimum along the specified axis (for example, at “-0.5” for an object whose axis is 1 unit in length).
 - **Center** sets the pivot point along the center of the specified axis for the object (for example, at “0” for an object whose axis is 1 unit in length).
 - **Max** sets the pivot point to the object's maximum along the specified axis (for example, at “+0.5” for an object whose axis is 1 unit in length).
- Use the **Modify > Move Pivot > Y** menu item to set the object's pivot point along the Y axis. Select **Min**, **Center**, or **Max**.
- Use the **Modify > Move Pivot > Z** menu item to set the object's pivot point along the Z axis. Select **Min**, **Center**, or **Max**.
- Select **Modify > Move Pivot > All > Center** to set the object's pivot point to the object's center (typically, the point “0, 0, 0”).

You can also use the keyboard shortcuts (**1, 2, 3** to move the pivot along the X axis; **4, 5, 6** to move the pivot along the Y axis, **7, 8, 9** to move the pivot along the Z axis, and **0** to move the pivot to the object's center.

To position an object's pivot point using the Property Editor:

1. Select the object.
2. Open the Property Editor, and edit the values for the **Pivot** property. The value **0, 0, 0** represents the object's origin.

Moving Objects

You can move an object, or group of objects, by dragging its Move manipulator or editing its **Translation** properties in the Property Editor.

The Move manipulator is a three-arrowed icon with shaded squares at the center. Each arrow controls movement along one axis. Each square controls movement in a plane. The cube game object in *Figure 45* has its Move manipulator active.

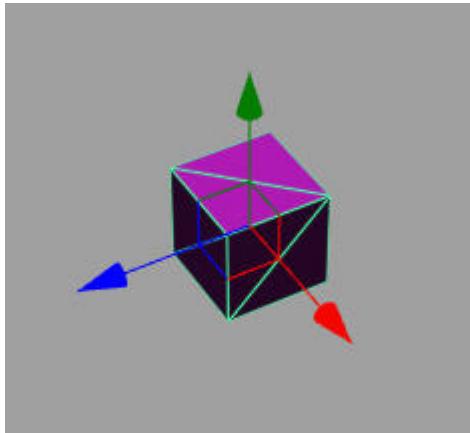


Figure 45 Cube with Move manipulator active

When you move a group of objects, there is only one Move manipulator. The objects in the group maintain their positions relative to each other.

The Translation property's parameters in the Property Editor specify the X, Y, and Z position of the object's origin.

To move objects by dragging their Move manipulator:

1. To use the Move manipulator, select the **Modify > Move** menu item.
2. Select the objects that you want to move.
3. The Move Manipulator appears in the selected object.
4. Drag a Move Manipulator axis to move the object in that direction. Drag one of the shaded (yellow) squares to move the object freely in two dimensions.
5. Holding SHIFT down while clicking on the shaded squares snaps the object (or objects) to the underlying object.
6. Holding SHIFT down while clicking and dragging on one of the axis arrows will snap the object (or objects) along the axis in the direction of the drag.

To move an object by editing its Translation properties:

1. Choose the **View > Properties** menu item to open the Property Editor.
2. In the **Translation** fields, edit the X, Y, and Z coordinates to specify new values.

Scaling Objects

You can scale a game object, or group of game objects, by dragging its Scale manipulator or setting its **Scale** properties in the Property Editor.

The Scale manipulator is a three-arrowed icon with a yellow square. Each arrow scales the object along one axis. The yellow square controls scaling in all three dimensions. The cube game object in *Figure 46* has its Scale manipulator active.

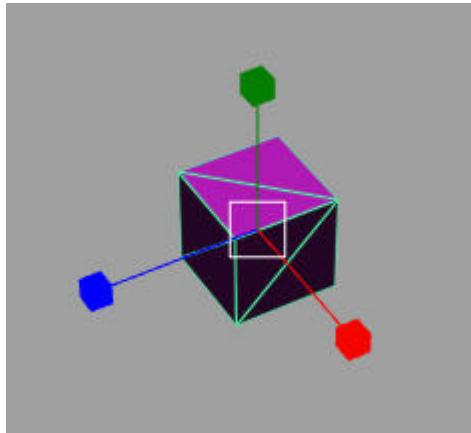


Figure 46 Cube with Scale manipulator active

The Scale property parameters specify the amount of scaling in the X, Y, and Z directions. The Scale property values are measured according to the original dimensions of the object, with 1, 1, 1 representing its original size.

Scaling equally in all three dimensions maintains the object's overall proportions. Scaling along an individual axis deforms the object. For example, making one axis of a sphere shorter would create an ellipsoid (or flying saucer).

An object scales away from its pivot point. Thus, when the pivot point is at the origin, scaling expands or contracts the object in or out from its origin.

Placing the pivot point away from the origin lets you set a boundary for the object's scaling. For example, if you want to scale a monster standing on the ground, position its pivot on the ground. Scaling will occur away from the pivot point, and the monster will still stand on the ground (rather than scaling into and through the ground).

If you scale a group of game objects, each game object scales by the same amount around its own pivot point; the objects' pivot points do not move farther apart or closer together.

If you scale a game object through zero to negative values, the game object inverts.

To scale game objects and maintain their proportions:

1. Select the **Modify > Scale** menu item.
2. Select the game objects.
3. Drag the yellow box. Drag to the left to make the object smaller; drag to the right to enlarge it.

To deform a game object:

1. Select the **Modify > Scale** menu item.
2. Select the game objects.
3. Drag one axis of the Scale manipulator.

The game object shrinks or enlarges along that axis, but the other dimensions remain constant.

Extending Objects

You can use the Scale manipulator to scale an object from its pivot point. However, if you want to extend or scale an object in a single direction, and not from its pivot point, you can use the Extension manipulator. For example, you can easily extend a cube so that it has a rectangular shape, without having to reposition its pivot point.

The Extension manipulator is a six-arrowed icon. Each arrow extends the object along one side of an axis. The controls at the end of the axes are colored to represent the positive and negative sides of each axis, where a darker color represents positive, and a lighter (pastel) color represents negative. The cube game object in *Figure 47* has its Extension manipulator active.

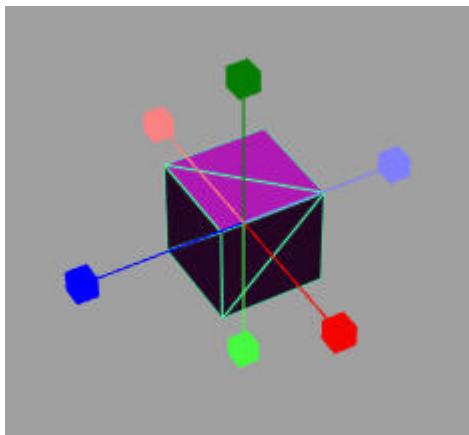


Figure 47 Cube with Extension manipulator active

When you extend an object, you deform it along one axis, making it longer or shorter in that direction. Only the end of the object that you drag extends, however; the other end does not move. You can achieve similar effect by resetting the pivot point and using the Scale manipulator.

In general, the Extension manipulator is most useful for simple objects, such as cubes, cylinders, spheres, and so on. More complex objects are not typically designed to be extended along a single axis and retain visual integrity.

Figure 48 shows a comparison of using the Scale manipulator (along one axis) and using the Extension manipulator. The pivot point remains at each object's origin for all three panels. The first panel of the figure shows three cubes. The second panel shows how the Scale manipulator deforms the object along the Y axis, both upward and downward. The third panel shows how the Extension manipulator extends the object in only one direction along the Y axis, upward.

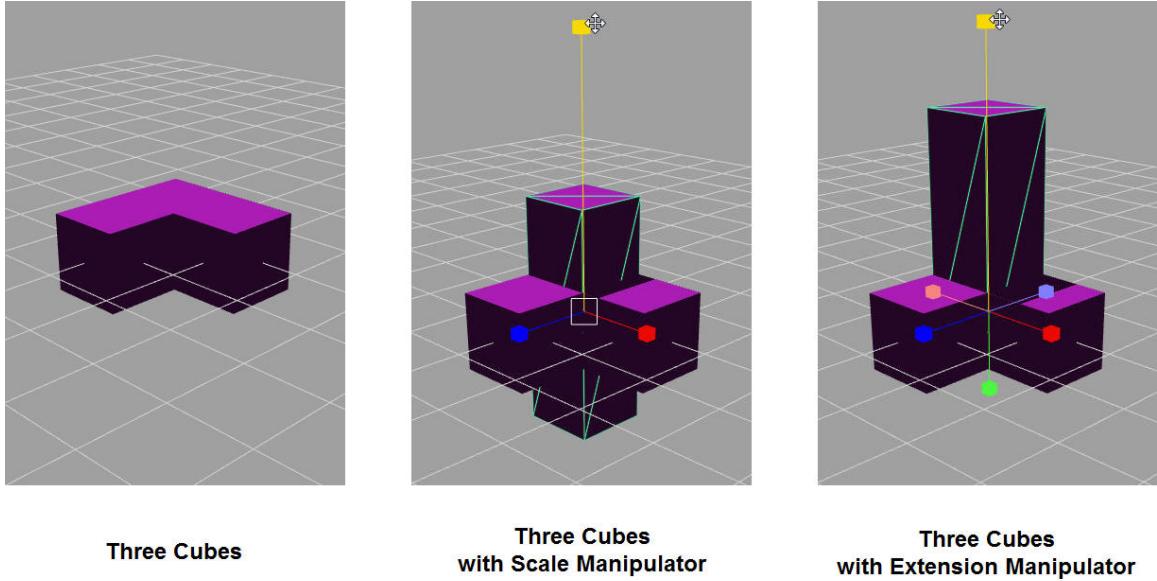


Figure 48 Comparing the Scale manipulator and the Extension manipulator

You can also adjust an object's Scale property and Translation property parameters to specify the amount of extension for an object. You must adjust both parameters because the Scale parameter acts like the Scale manipulator, and scales the object from its pivot point. Thus, after you adjust an object's Scale parameter, you must adjust its Translation parameter for the same axis by an appropriate amount. For example, to achieve the same result as shown in the third panel of *Figure 50*, you could adjust the cube's Scale parameter for the Y axis to from 1 to 3 and then adjust its Translation parameter for the Y axis from 0 to 1.

If you extend a group of game objects, each game object extends by the same amount along the selected axis.

If you extend a game object through zero to negative values, the game object inverts.

To extend game objects:

1. Select the **Modify > Extension** menu item.
2. Select the game objects.
3. Drag one of the six axes to extend the object.

Rotating Objects

The Rotation property parameters specify the amount of an object's rotation in the X, Y, and Z directions. Rotation property values are measured in degrees.

You can rotate an object by dragging its Rotation manipulator or setting its Rotation properties in the Property Editor. The Rotation manipulator has three concentric circles. The red, blue, and green circles control rotation around individual axes.

Figure 49 shows a cube object with its Rotation manipulator active. Rotation occurs around the object's pivot point.

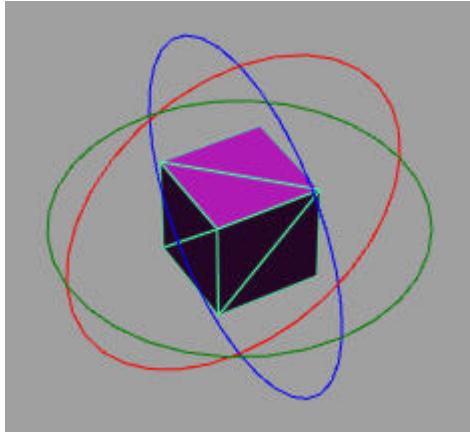


Figure 49 Cube with Rotate manipulator active

You can rotate multiple game objects at the same time. Which pivot point the objects rotate around depends whether the objects belong to the same group object or are selected by clicking and dragging.

Grouped game objects rotate around the group's pivot point. Only one rotation manipulator appears for the grouped game object.

Figure 50 shows three cubes selected for rotation:

- The cubes on the left were selected by dragging a selection rectangle over all three objects; the Rotation manipulator is displayed for only one cube, and each cube will rotate about its own pivot point. None of the cubes will change their translation values (that is, they will not move in the X, Y, or Z directions).
- The cubes on the right are a group; the Rotation manipulator is positioned at the group's pivot point, and all three cubes will rotate together about the common group pivot point. Depending on how the group is rotated, at least one cube will change its translation values (that is, move in the X, Y, or Z directions). Each cube will maintain its orientation relative to the other two cubes.

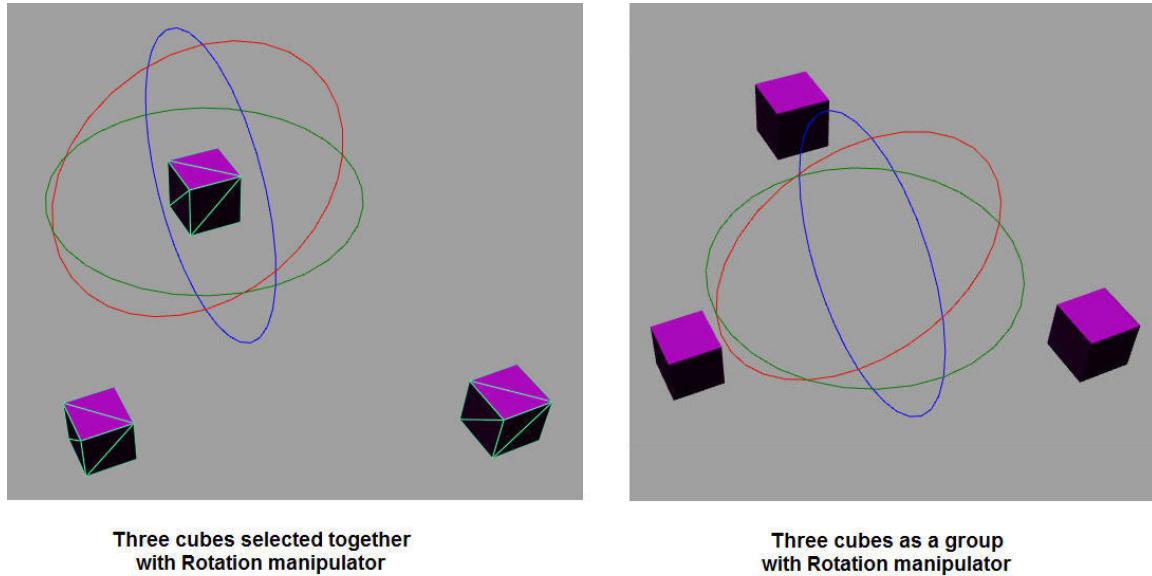


Figure 50 Cubes selected together and as a group for rotation

To rotate a game object:

1. Select the **Modify > Rotate** menu item.
2. Select the game object.
3. Drag the circle that controls rotation around a particular axis.

Duplicating Game Objects

The LevelEditor duplicates game objects if you press the SHIFT and CTRL keys while you click the Move manipulator. You can then drag the Move manipulator to position the new game objects in the Design View.

If you duplicate a set of game objects, the objects maintain the same position relative to each other as the originals. For example, in *Figure 51*, the selected game objects are duplicates of the others. The duplicate game objects have the same relative positions as the original game objects.

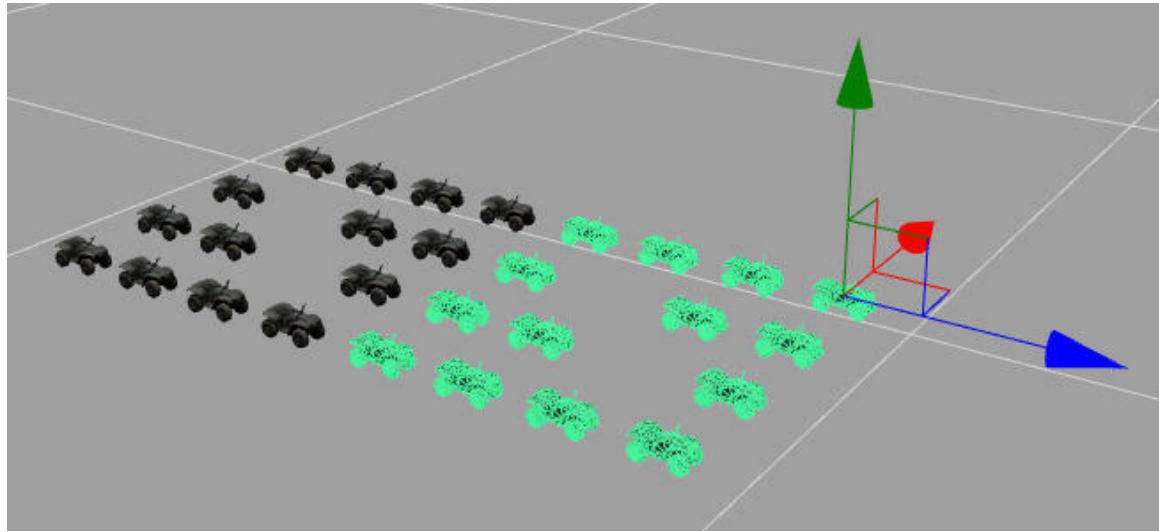


Figure 51 Duplicate set of game objects

To duplicate game objects:

1. Click on the Move manipulator.
2. Select the game objects in the Design View.
3. Press the SHIFT and CTRL keys while you click the Move manipulator.
4. Drag the Move manipulator to reposition the duplicate game object (or objects).

For a demonstration of how to duplicate game objects using the LevelEditor, view the “Duplicating Objects” video on the [LevelEditor videos page](#) on SHIP.

Rotating Game Objects to Face a Nearby Surface

Using the Rotate on Snap feature, you can have a game object automatically orient itself to the surface it is snapping to if you hold down SHIFT while you click inside the Move manipulator's shaded squares. If you orient a set of game objects, each game object rotates individually to face the surface polygon that is nearest to it.

For example, *Figure 52* and *Figure 53* show the same game objects in different locations on the bus. The game objects always rotate to face the surface of the bus that is nearest to them.

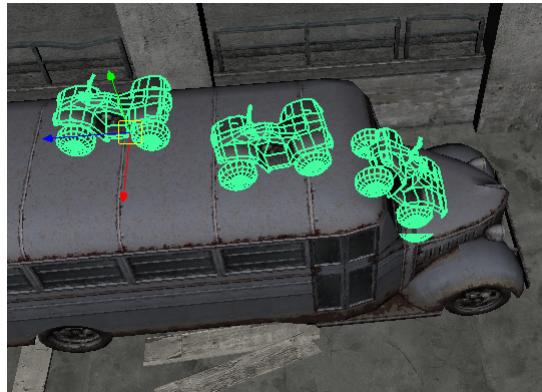


Figure 52 ATVs facing their nearest surface polygon

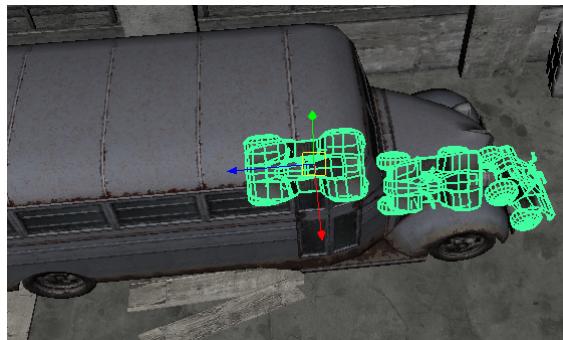


Figure 53 ATVs rotated to face a new surface polygon

To rotate an object to the nearest surface:

1. Select the **Modify > Rotate on Snap** menu item.
2. Turn on the Move manipulator.
3. Select the game objects to rotate.
4. Press SHIFT while you drag the Move manipulator's shaded squares to re-position the game objects.

For a demonstration of how to use **Rotate on Snap** in the LevelEditor, or view the "Rotating Objects to Face a Nearby Surface" video on the [LevelEditor videos page](#) on SHIP.

Displaying and Hiding Game Objects

The LevelEditor can display and hide game objects in the following ways so that you can focus on different game objects as you work:

- Hide selected game objects
- Display selected game objects only
- Display all game objects
- Display previously hidden game objects

When a game object is visible, its **Visibility** property is set to **True**. When a game object



is invisible, its **Visibility** property is set to **False**.

To hide selected game objects:

1. Select the game objects to hide.
2. Select the **View > Hide** menu item.

To display selected game objects and hide all the others:

1. Select the game objects.
2. Select the **View > Isolate** menu item.

To make all objects visible:

- Select the **View > Show All** menu item.
—Or—
- Select the **View > Show Last Hidden** menu item.
—Or—
- Set the objects' **Visibility** property to **True**.

You can also assign objects to layers or sublevels, which both provide an efficient way to hide and display sets of game objects. For information about layers, see "[Using Layers to Unclutter the World](#)" and "[Using Sublevels to Manage Large Game Worlds](#)" later in this chapter.

Zooming Specific Game Objects

In addition to zooming in and out on the world, the LevelEditor can also zoom in closely on selected game objects by fitting them to fill an entire viewing pane. You can frame game objects in all open viewing panes or in only the active pane, leaving other viewing panes unchanged.

To zoom in and frame game objects in the active viewing pane only:

1. Select the game object or group.
2. Select the **View > Frame Selection** menu item.

Rendering Game Objects

The LevelEditor lets you render game objects in different styles to visually simplify the Design View display. None of these render options has a significant impact to rendering times.

For a demonstration of the different rendering modes that are available in the LevelEditor, view the “Rendering Objects” video on the [LevelEditor videos page](#) on SHIP.

You can render game objects with the following features:

- **Wireframe** displays all game objects as wireframes without fills, as shown in *Figure 54*. This option can be useful for seeing game objects that are hidden behind or within other game objects.

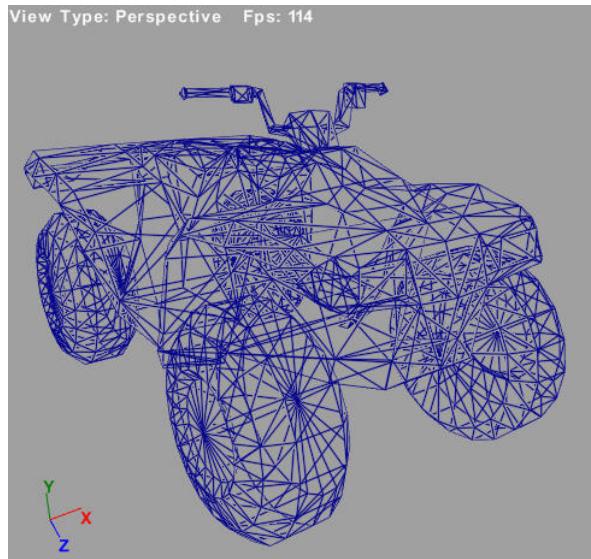


Figure 54 Game object with wireframes only

- *Smooth* renders the surface, as shown in *Figure 55*. If the game object has no texture associated with it, the surface is a smooth solid.

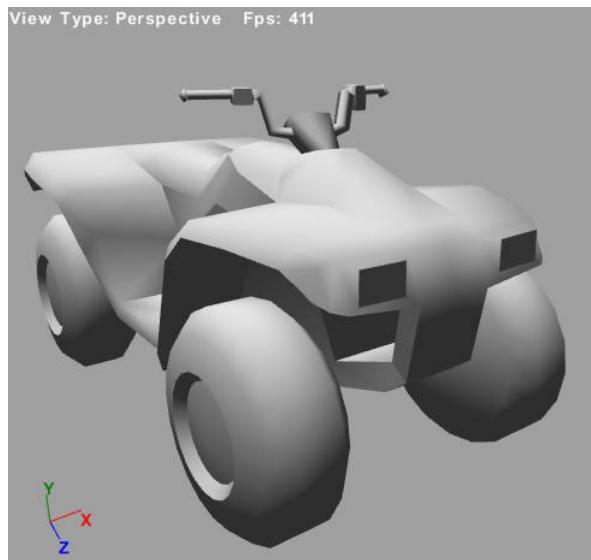


Figure 55 Game object with filled surface but no texture

- *Solid over wireframes* renders the outer surface with its wireframe outline, as shown in *Figure 56*. Selected game objects always display wireframes, but this feature displays wireframes for all game objects in the world. When all game objects display wireframes, the selected game object's wireframes are in green; the unselected game objects' wireframes are blue.

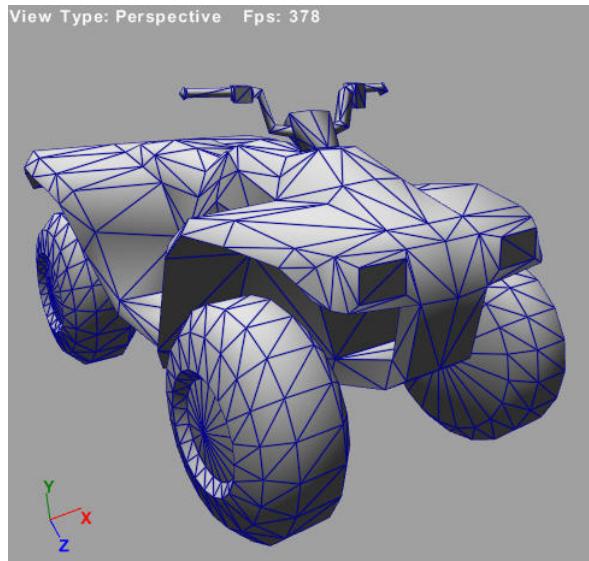


Figure 56 Game object with filled surfaces and wireframes

- *Textured* lets you see how the final scene looks, as shown in *Figure 57*. However, turning textures off can sometimes make it easier to see the spatial relationships among game objects.



Figure 57 Game object with texture

- *Textured* together with *Solid over wireframes* lets you see how the final scene looks and adds the wireframe outlines so you can see the object's structure, as shown in *Figure 58*. However, turning textures off can sometimes make it easier to see the spatial relationships among game objects. Note that selected objects show their wireframes (in green); this option shows all wireframes (in blue).

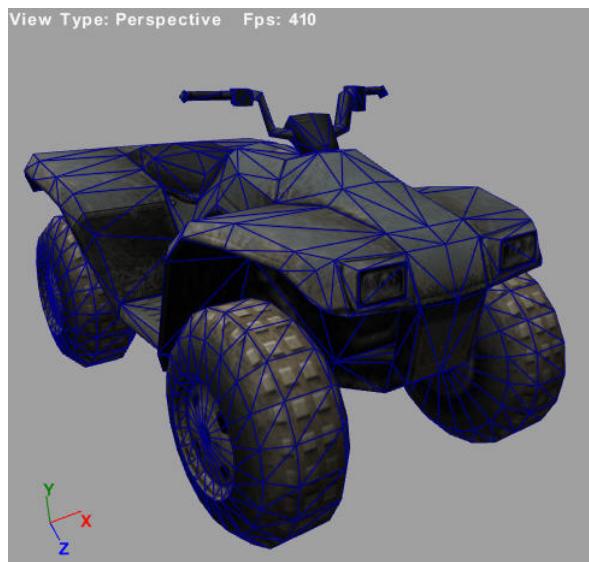


Figure 58 Game object with texture and wireframes

- *Lighting* makes contours and shapes more obvious by allowing local light sources to provide shadows, as shown in *Figure 59*. Without lighting, the object appears bright or dark according to its associated texture.



Bike with Lighting On



Bike with Lighting Off

Figure 59 Game object with lighting on and with lighting off

- *Render shadow* lets local light sources render shadows onto and from an object, as shown in *Figure 60*. The effect is more pronounced with a nearby Light object.



Figure 60 Game object with shadows rendered

- *Normals* renders the normal vector for each surface polygon, as shown in *Figure 61*. LevelEditor renders a normal vector at each vertex of a game object. The rendered normal vector represents an average normal vector for each of the vertex's adjacent polygon surfaces (faces). You can use rendered normals to verify that a game object's model is correctly rendered in the world.

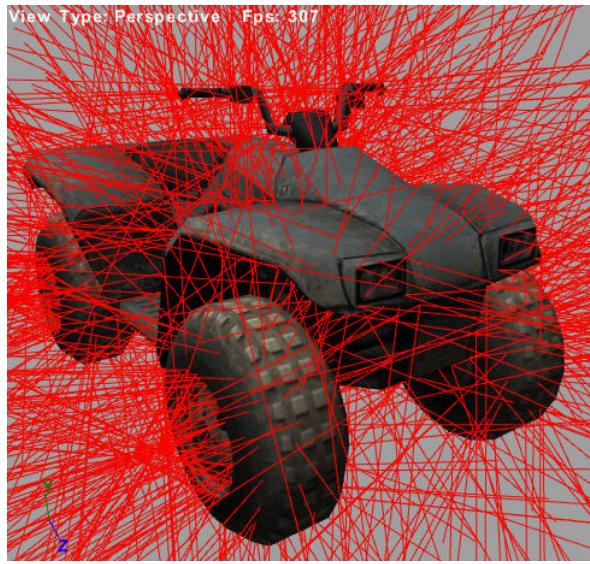


Figure 61 Game object with normals rendered

- *Back Face* renders the back-facing polygons of a model so that you can better see how the game object fills the space. *Figure 62* shows the same landscape scene rendered with and without back-facing polygons.

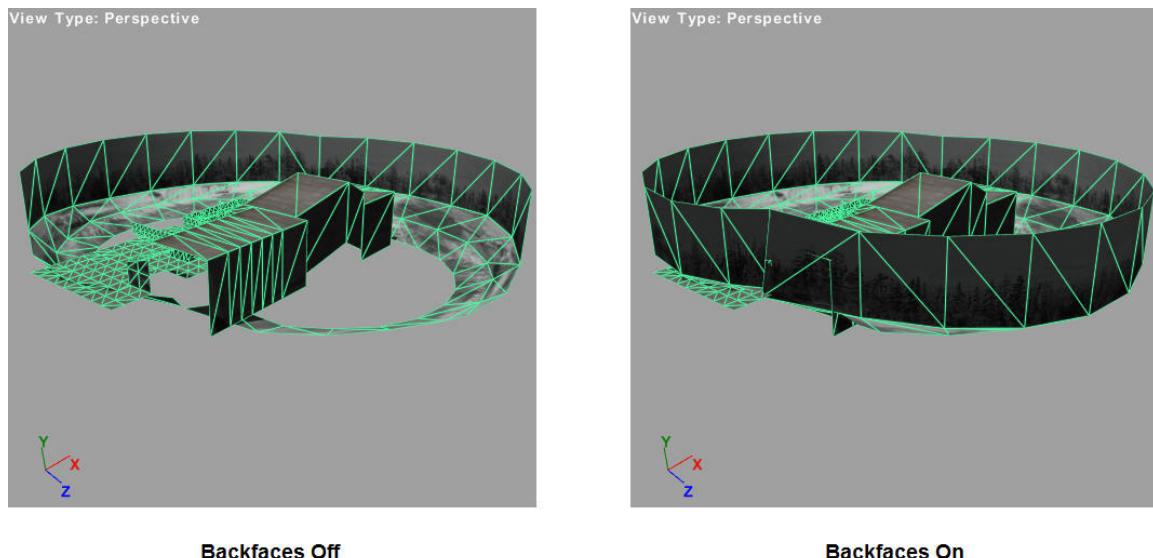


Figure 62 Model with and without back faces rendered

Use the **View** menu to choose how objects appear in the world. *Table 17* summarizes how to choose a render mode.

Table 17 Specifying rendering style

To render game objects with	Choose
Wireframes only	View > Wireframe —Or—  Click
Smooth surfaces but no wireframes	View > Solid —Or—  Click
Smooth surfaces with wireframes	View > SolidOverWire —Or—  Click
Textured surfaces	View > Textured (The Smooth or Outlined option must already be selected) —Or—  Click

To render game objects with	Choose
Lighting effect	View > Lighting (The Smooth or Outlined option must already be selected.) —Or—  Click
Shadow effect	View > Shadow —Or—  Click
Normals	View > Normals —Or—  Click
Back-faces rendered	View > BackFace (The Smooth or Outlined option must already be selected.) —Or—  Click

You can also cycle through each of the rendering options to quickly compare how the world looks in each mode:

- Select the **View > CycleRenderModes** menu item.
- Or—
- Press the space bar.

Using Layers to Unclutter the World

Unlike a group of objects selected by clicking or dragging, a layer embodies a persistent grouping of objects. As you work in the Design View, you might find it convenient to show or hide layers as one way to display or hide groups of objects quickly.

Layers are not game objects, and they do not appear in the Project Lister. However, they are distinct objects that appear in the Layers list, as shown in *Figure 63*.



Figure 63 Layers list example

The checkbox next to the name of a layer controls the display of objects in that layer. When a layer's checkbox is selected, the LevelEditor displays objects in that layer. Clearing the checkbox hides objects in that layer.

- **To show the Layers list**, select the **Window > Layers** menu item.
- **To create a new layer**, right click in the **Layers** window and select **Add Layer** from the contextual menu.
- The LevelEditor creates a new layer object in the Layers window.
- **To add a sub-layer** under the current level, select an existing layer, right-click, and select **Add Layer** from the contextual menu.
- **To assign Game Objects to a layer**, perform any of the following tasks:
 - Drag and drop objects from the Project Lister into the Layers list
 - Select one or more objects in the Project Lister or Design View, and copy/paste them into a layer
- **To display objects in a layer**, select the checkbox next to the layer name to show its objects
- **To hide objects in a layer**, clear the checkbox next to the layer name to hide its objects.

Using Sublevels to Manage Large Game Worlds

Like a layer, a sublevel (or subgame) is a persistent grouping of game objects. As you

work in the Design View, you might find it convenient to show or hide sublevels as one way to display or hide groups of objects quickly.

Unlike layers, however, sublevels are game objects, and they do appear in the Project Lister, as shown in *Figure 64*. In the figure, the main game level is at the top of the project hierarchy (the **Game** node), and four sublevels are shown, Terrain, OfficeBuildingPlaza, and HillsideBuilding (the Airstrip sublevel is not shown). A single game world can be divided into many sublevels.

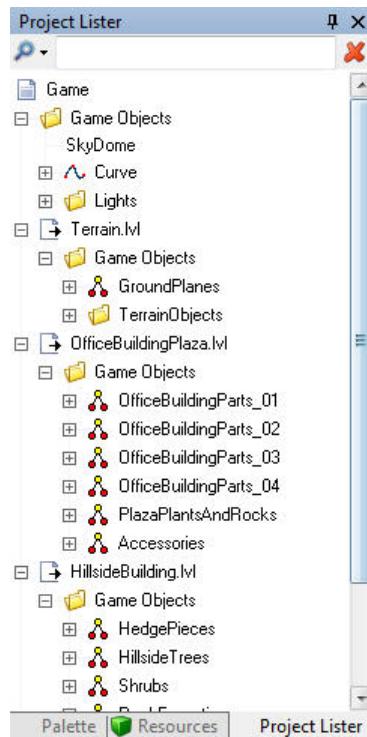


Figure 64 Sublevels shown in the Project Lister

Each sublevel is a separate LVL file; your main game LVL file includes a reference to each of the sublevel LVL files. Thus, you can work on sublevels independently of the main game level, as needed, or share sublevel files with other designers to create a large game world in parallel. Or, you can use the sublevels to partition the game world into more easily managed sections that you can show or hide as needed.

Because the sublevels are separate LVL files, you can dynamically resolve (render in the Design View) them in the main game level, include or exclude them in the main game level, or simply load them outside of the main game level and work on them separately. When a sublevel is included and resolved in the main game level, you work with it in the same way as the main game level, adding or modifying game object as desired.

You can add new game objects to a sublevel by performing either of the following tasks:

- Drag the object to the Project Lister and drop it within the **Game Objects** folder of a sublevel. The object is placed at location 0,0,0 of the world, but is included within the sublevel.
- Select an object that is already within the sublevel, then drag and drop a new object anywhere within the world. The object has the location you specify, and is



also included within the sublevel.

When you save a game level that includes sublevels, any changes to the sublevels are saved at the same time as the main level. If you do not want to accidentally modify objects in a sublevel, you can lock the objects, as described in "[Locking Game Objects](#)" in the "Working with Game Objects" chapter.

To add a new sub-level:

1. Right-click the **Game** node (or any existing sublevel node) in the Project Lister.
2. Select **Add new SubGame** from the context menu.
3. Specify a file name for the new sublevel in the Save As dialog.

The sublevel appears in the Project Lister below the **Game** node (or existing sublevel node) with the name that you specified.

To add an existing level as a sublevel:

1. Right-click the **Game** node (or any existing sublevel node) in the Project Lister.
2. Select **Add existing SubGame** from the context menu.
3. Select the sublevel file from the Open dialog.

The sublevel appears in the Project Lister below the **Game** node (or existing sublevel node).

To exclude a sublevel:

1. Right-click a sublevel in the Project Lister.
2. Select **Exclude SubGame** from the context menu.

Note:

When you exclude a sublevel, the reference to it is removed from the main game level. If you want to include the sublevel at a later time, you must add it as an existing sublevel.

To unload a sublevel:

1. Right-click a sublevel in the Project Lister.
2. Select **Unresolve Subgame** from the context menu.

The sublevel and all of its game resources are removed from the Design View. However, the main game level retains the reference to the sublevel. You can use the unresolved feature to quickly hide all objects of a sublevel. Another method of hiding the objects of a sublevel is to select the sublevel's Game Objects folder, open the Property Editor, and deselect its **Visible** property.

To reload an unresolved sublevel:

1. Right-click an unresolved sublevel in the Project Lister.
2. Select **Resolve Subgame** from the context menu.

The sublevel and all of its game resources are added to the Design View.

By default, all sublevels are automatically resolved when you open a main game level. You can use the Preferences dialog to specify that subdocuments are not automatically resolved: select the **Edit > Preferences** menu item to open the Preferences dialog.

Select **Documents**, and then select **Resolve on load**, as shown in *Figure 65*. If you specify **Resolve on load** as False, sublevels are not automatically resolved when you open a main game level. You can selectively resolve referenced sublevels as needed.

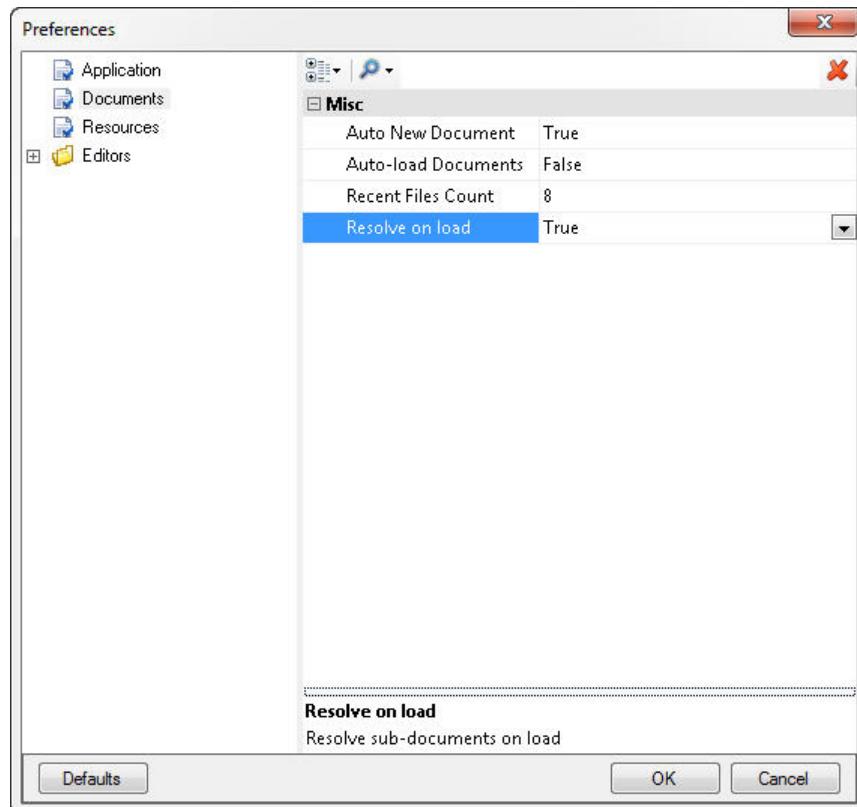


Figure 65 Preferences dialog for *Resolve on load* option

See "[Preferences](#)" in the "The LevelEditor User Interface" chapter for more information about specifying LevelEditor preferences.

For a demonstration of how to use sublevels, view the "Using Sublevels" video on the [LevelEditor videos page](#) on SHIP.

Using Linears to Lay Out Lines and Boundaries

A *linear* is a special type of game object; it defines a geometric shape that has a set of control points.

The Palette provides an object for creating lines and curves:

- A Polyline object that defines a set of connected lines.
- A CatmullRom spline object that defines a Bezier curve.
- A Bezier spline object that defines a Bezier curve.

Polylines and spline curves have endpoints that anchor the object's two ends in the world. Dragging a polyline's or curve's endpoint repositions it. *Figure 66* shows three linears in the Design View, each with five control points in addition to their endpoints. Each linear is defined with the same parameters; the blue one is a polyline, the green

one is a CatmullRom spline, and the orange one is a Bezier curve. The figure shows the linears from four different projection views: perspective, top, front, and left. These linears could represent a short, but complex, flight path of a game object.

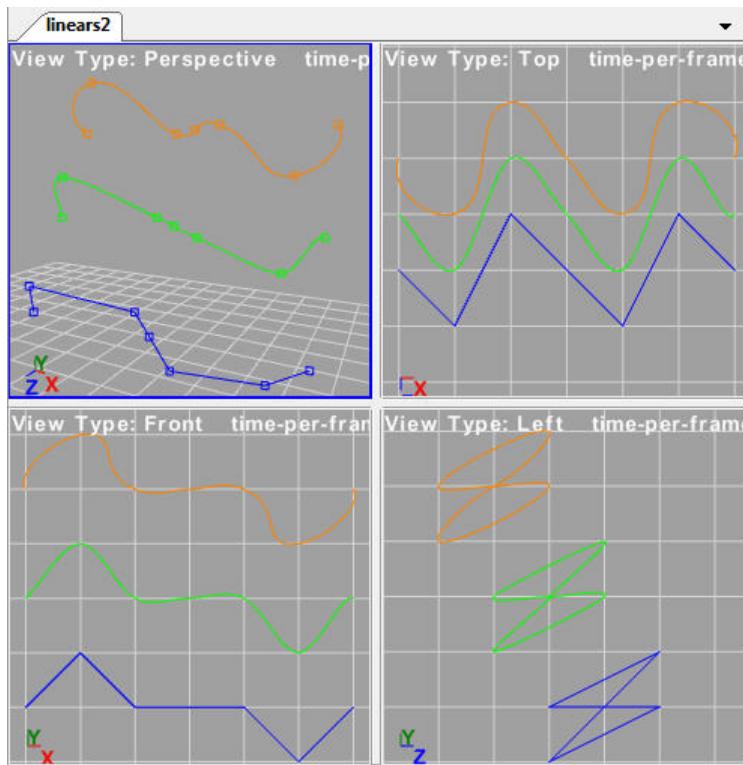


Figure 66 Two linears: a polyline and a curve

You can create closed polylines and curves by selecting the linear, opening the Property Editor, and setting the **Is Closed** property to **True**. Closing a linear connects the linear's two end control points to form a closed shape. *Figure 67* shows the same three linears as in *Figure 66*, but the linears are now closed.

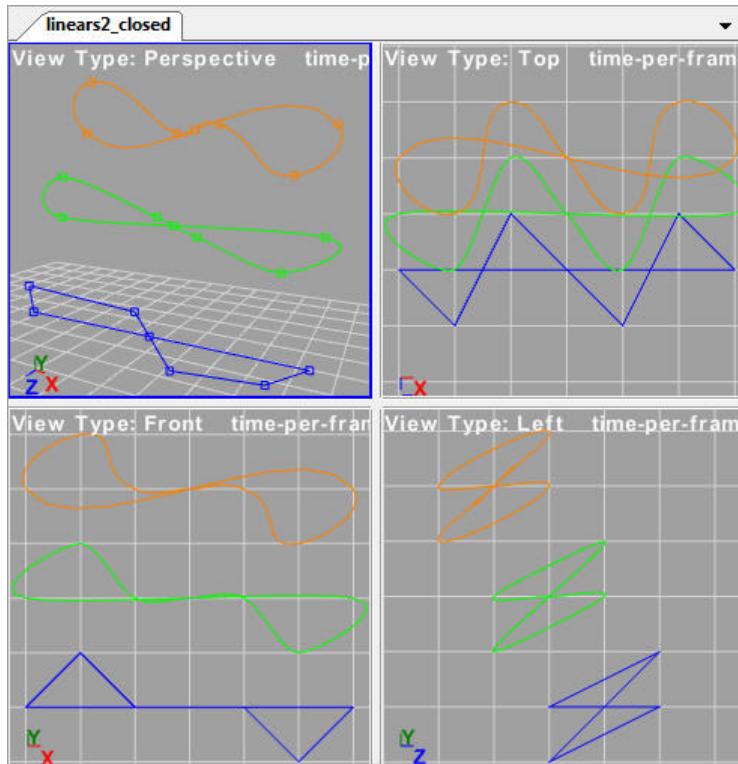


Figure 67 Two closed linears

When laying out a linear onto another surface, you can snap the control points to other surfaces the same way you snap to any other object.

For a demonstration of how to create linears, view the “Creating Linears” video on the [LevelEditor videos page](#) on SHIP.

To create a straight-sided linear:

1. Open the Palette, and drag a polyline object to the world.
2. Use the Move manipulator to drag the curve or its endpoints to the beginning and end locations you want.

To create a curved linear:

1. Open the Palette, and drag a CatmullRom spline object or Bezier spline object to the world.
2. Use the Move manipulator to drag the curve or its endpoints to the beginning and end locations that you want.

To adjust a curve or polyline:

1. Shift-click on the curve or polyline to create additional control points.
2. Use the Move manipulator for each control point to shape the linear.

Wherever you shift-click a linear, the LevelEditor creates a control point. Dragging a control point on a linear repositions that section of the linear. Moving the control point only affects the line segment or curve segment between the control point that you are moving and the nearest adjacent control point.



To close a curve or polyline:

1. Select the curve or polyline.
2. Open the Property Editor and set the object's **Is Closed** property to **True**.

Closing a linear adds an additional line segment or curve segment between the two end control points of the linear. You can add additional control points to reshape the newly closed linear, but these new control points are added to the original linear, not to the closing line (that is, the new control point moves the end control point and shortens the closing line).

Adding Light Objects

The Palette provides three sample Light objects:

- DirLight – a directional light source of unlimited range
- BoxLight – a directional light source that is limited in range to a box boundary
- PointLight – an omnidirectional light source that is limited in range to a spherical boundary

You can place each of these light sources anywhere in the world, as needed. After you specify the properties for the light source, you can typically deselect its **Visibility** property; the light remains in the world even if the Light object itself is not visible.

All of the Light objects have the following properties to specify the color of the light:

- Ambient – specifies the ambient color for the light from the source
- Diffuse – specifies the color of light that reflects off other game objects; diffuse reflections are multidirectional
- Specular – specifies the color of light that reflects off other game objects; specular reflections are monodirectional, directly in line with the viewer

Recommendations:

- For natural light effects, specify a grey color, such as RGB 75,75,75 for the **Ambient** property.
- To use a Light object as a source of darkness, specify a dark color, such as RBG 0,0,0 for the **Diffuse** property.
- When you place a Light object in the world, consider turning on the **Render shadow** option.

For a demonstration of how to use Light objects, view the “Using Lights in a Game Level” video on the [LevelEditor videos page](#) on SHIP.

Directional Light

Because the Directional Light has unlimited range, you can place it anywhere in the world; you set its properties to specify the direction of the light.

Set the Directional Light object's **Direction** property to specify the direction of the light source in the X, Y, and Z directions of the world. You can specify positive or negative values.

Box Light

The Box Light has a specified range, so you should place the object in the world at the location where you want the light to appear.

Use the Scale manipulator to specify the size of the Box light object's boundary. The boundary is a cube, which makes this object ideal for uniformly lighting a room or similar area.

Set the Box Light object's **Direction** property to specify the direction of the light source. You can specify positive or negative values.

Set the Box Light object's **Attenuation** property to specify the brightness of the light source in the X, Y, and Z directions of the world. You can specify positive or negative values. If the **Ambient** and **Diffuse** property values specify different colors, the **Attenuation** property can alter the mix of the two different colors.

Point Light

The Point Light has a specified range, so you should place the object in the world at the location where you want the light to appear.

Set the Point Light object's **Range** property to specify the range of the light source. You can specify positive or negative values. The object's boundary is a sphere, which makes this object ideal for lighting outdoor areas.

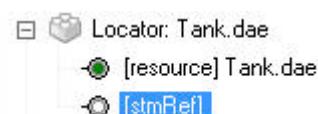
Set the Point Light object's **Attenuation** property to specify the brightness of the light source in the X, Y, and Z directions of the world. You can specify positive or negative values. If the **Ambient** and **Diffuse** property values specify different colors, the **Attenuation** property can alter the mix of the two different colors.

Working with StateMachine Assets

For game objects that include a state machine reference, you can drag and drop state machine assets onto game objects in the LevelEditor Project Lister. The game object's behavior is then determined by the state machine to which it is bound. For each individual game object that includes a state machine reference, you can modify its state machine properties to specify the object's desired behavior in the game environment.

To add state machines to a game object, perform the following tasks:

1. Add the game object to the project, and select it in the Project Lister. Ensure that the object has a state machine reference (**stmRef** property).
2. Drag a StateMachine Editor project file (.stmpobj) from the "AI" folder of the Resources list to the game object's **stmRef** property in the Project Lister, as shown in *Figure 68*.



*Figure 68 A game object's **stmRef** property in the Project Lister*

3. Select the game object's **stmRef** property and open the Property Editor (or Grid Property Editor) to modify the object's state machine properties, as shown in *Figure 69*. The figure shows a state machine for a tank object.

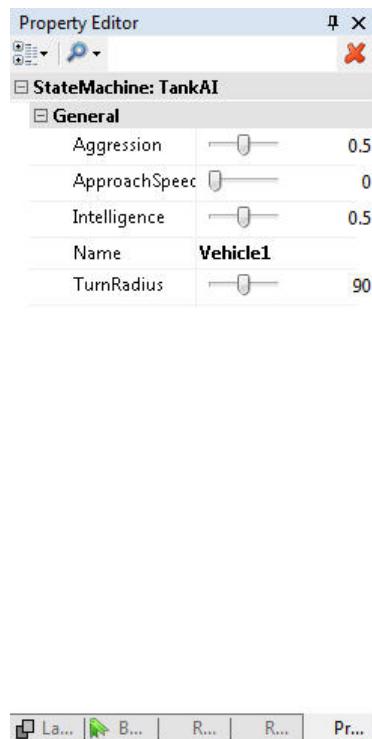


Figure 69 A game object's state machine properties

Figure 70 shows a tank object with an associated state machine that specifies the object's name, aggression, approach speed, intelligence, and turn radius. You can change the default state values using the slider bars in the Property Editor, or you can edit them directly in either the Property Editor or Grid Property Editor. You can select a state property to view a description of that property; these descriptions are defined as annotations in the StateMachine Editor.

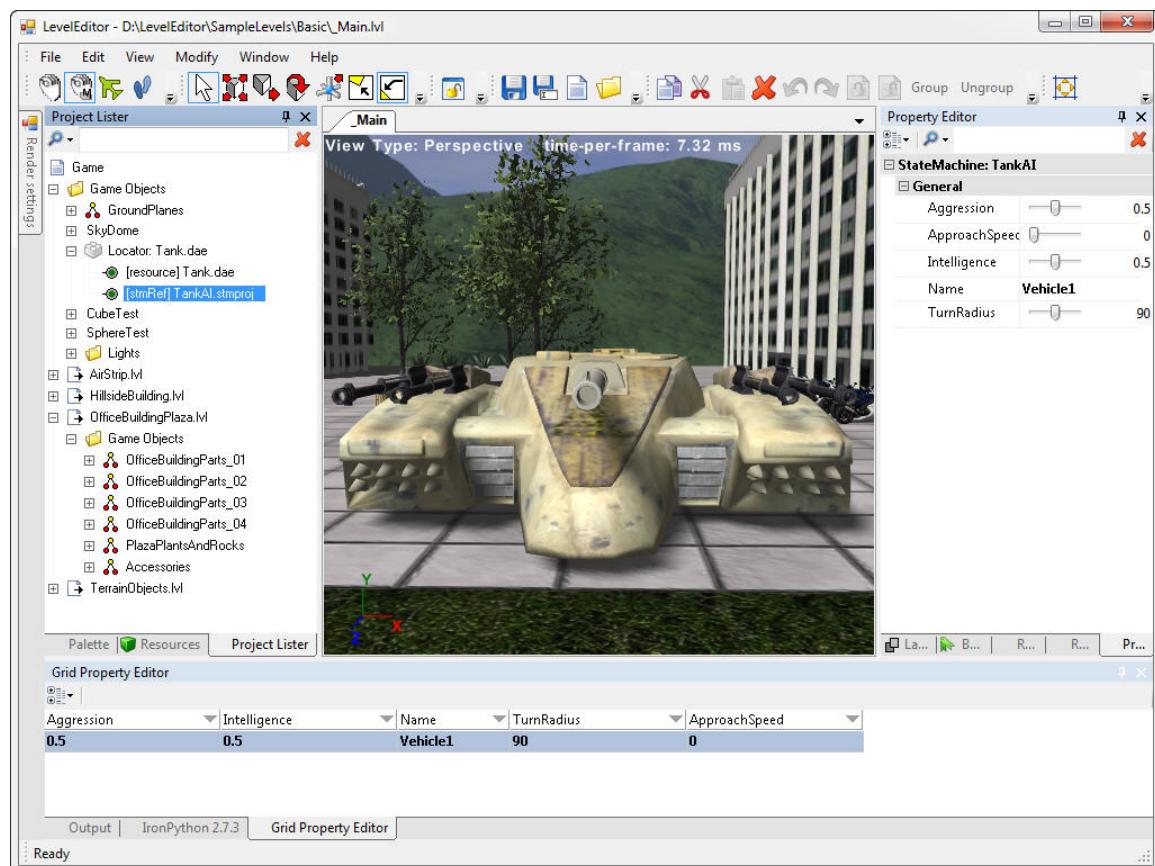


Figure 70 A tank object with an associated state machine