Quick answers to common problems

CryENGINE 3 Cookbook

Over 90 recipes written by Crytek developers for creating third-generation real-time games

CRYENGINE 3

Dan Tracy Sean Tracy

[PACKT] PUBLISHING

CryENGINE 3 Cookbook

Over 90 recipes written by Crytek developers for creating third-generation real-time games

Dan Tracy

Sean Tracy

[PACKT] PUBLISHING

BIRMINGHAM - MUMBAI

CryENGINE 3 Cookbook

First published: June 2011

Production Reference: 1170611

Published by Packt Publishing Ltd.
32 Lincoln Road
Olton
Birmingham, B27 6PA, UK.

ISBN 978-1-849691-06-2

www.packtpub.com

Cover Image by Sean Tracy (`seanp@crytek.com`)

Credits

Authors
Sean Tracy
Dan Tracy

Acquisition Editor
Steven Wilding

Development Editor
Alina Lewis

Technical Editor
Aditi Suvarna

Copy Editor
Laxmi Subramanian

Indexer
Rekha Nair

Project Coordinator
Zainab Bagasrawala

Proofreader
Aaron Nash

Graphics
Geetanjali Sawant

Production Coordinator
Shantanu Zagade

Cover Work
Shantanu Zagade

About the Authors

Sean Tracy is Crytek's Senior Field Application Engineer for the award-wining CryENGINE. He is responsible for adapting the engine and its features to individual licensees, as well as developing full technical and vertical slice demos for prospective and existing clients. Describing himself as a generalizing specialist, he also gives support directly to CryENGINE licensees, while designing and maintaining their workflows, pipelines, and development techniques.

Sean was recruited by Crytek in 2008 after working as an electronics technician for the Canadian Military. He was recruited due to his role in founding and leading the development on the award winning total conversion project MechWarrior: Living Legends. Since then, he has been featured in numerous gaming magazines and has been invited to speak at many game related trade shows and seminars. He is an avid gamer with extensive modding experience on titles including Never Winter Nights, the Battlefield engine Frostbite, Doom, and Quake.

I would like to thank my wife for her understanding and support throughout the process of writing this book and for her ongoing support in allowing me to do what I truly enjoy for a living. I would also like to thank my brother for co-authoring the book with me as it's a pleasure to be able to work with someone with the same love for the technology as I have. Finally, I'd like to thank Crytek and Packt for their support in allowing me to write this book and for making one of the best game engines in the market.

Dan Tracy is Crytek's Technical Level Designer for the award-winning CryENGINE and Crysis 2. He is responsible for the creation and maintenance of numerous technical features and external applications used for telemetry and optimization. Viewed as more than a level designer, Dan prides himself on pushing the envelope when it comes to improving both technical and game related designs across multiple production disciplines.

Dan was recruited by Crytek in 2009 after previously working as a Quality Assurance Technician for BioWare. He was recruited due to his pivotal role in co-founding and leading development on the award winning total conversion project MechWarrior: Living Legends. Since then, he has been featured in numerous gaming magazines and has been interviewed by multiple media outlets. Dan is a passionate gamer, but an even more passionate modder and game designer, with vast knowledge and experience with multiple engines and titles including Never Winter Nights' Aurora, Battlefield's engine Frostbite, Unreal 3, and CryENGINE. This is Dan's first book.

I would like to thank my friends and family for giving me their support during the crunch time of the Crysis 2 production, which also paralleled the creation of this book. If it wasn't for them, this wouldn't have been possible. I would also like to thank my brother for co-authoring the book with me as well as Crytek for providing me with this amazing opportunity to share my knowledge of CryENGINE with the world. Finally, I'd like to thank Packt for their support and setting this whole project in motion and publishing my first ever book.

---

www.PacktPub.com

Support files, eBooks, discount offers and more

You might want to visit `www.PacktPub.com` for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and, as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on Packt books and eBooks.

http://PacktLib.PacktPub.com

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read, and search across Packt's entire library of books.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via web browser

Free access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface

With the overall complexity involved in creating games becoming exceedingly difficult and expensive with every successive console generation, many game developers have turned to middleware engines, such as the CryENGINE, that offer a complete pipeline for the game development process. CryENGINE is a perfect fit for most developers as it allows users to create their content quickly and easily and thus, allow games to meet and exceed current generation quality standards and still be created by fewer and fewer people.

As CryENGINE 3 is globally recognized as one of the world's most powerful real-time middleware development platforms, with this book we will deliver the best of what the engine has to offer. Through the use of CryENGINE's intuitive and powerful toolset, named Sandbox, designers, artists, animators, and even programmers will be treated to real-time creation and iteration tools for bringing their visions to life.

What this book covers

Chapter 1, *Getting Started*, helps you set up the entire CryENGINE 3 Software Development Kit, which can be a difficult task. This chapter will guide you through the stages in setting up the required folder structure and how to set up your layout for the Sandbox Editor.

Chapter 2, *Sandbox Basics*, helps you to learn the basic and most commonly used features provided by Sandbox.

Chapter 3, *Basic Level Layout*, helps you create your first Level Layout within the Sandbox Editor and learn some of the more advanced techniques used by designers for object placement and manipulation.

Chapter 4, *Environment Creation*, utilizes the CryENGINE 3 rendering tools to create photorealistic environments.

Chapter 5, *Basic Artificial Intelligence*, helps you learn the basics of how to the use AI to navigate in your levels.

Chapter 6, *Asset Creation*, helps you learn the pipeline of asset creation and export your 3D models to the CryENGINE format.

Chapter 7, *Characters and Animation*, describes how to create new characters to be used in the engine along with your own custom animations.

Chapter 8, *Creating Vehicles*, describes how to create a new vehicle from scratch and set up the entity code required so your players can drive.

Chapter 9, *Game Logic*, helps you to get started with the highly versatile Flow Graph Editor and create many useful scripts used in the level.

Chapter 10, *Track View and Cut-Scenes*, helps you to learn how to create your own cinematic cut-scenes within CryENGINE.

Chapter 11, *Fun Physics*, describes how to set up some enjoyable physics contraptions using CryENGINE 3's physics system.

Chapter 12, *Profiling and Improving Performance*, helps you to learn the tools behind profiling your levels and discover the best methods for improving performance.

What you need for this book

The Software Development Kit version of the CryENGINE is used for all examples in this book, thus, the reader should have a version of the development kit to be able to follow the recipes contained in this book.

Who this book is for

This book is written with the casual and professional developer in mind. With that said, it is important that the readers have some fundamental knowledge of some Digital Content Creation Tools, such as Photoshop and 3d Studio Max. Though not a fundamental requirement, having some basic knowledge of real-time graphics software and, consequently, the terminology used will make the goal of these recipes more clear.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: "The level must also be inside of your `Build` folder."

A block of code is set as follows:

```
<DamageMultipliers>
  <DamageMultiplier damageType="bullet" multiplier="0.125"/>
  <DamageMultiplier damageType="collision" multiplier="1"/>
</DamageMultipliers>
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
<DamageMultipliers>
  <DamageMultiplier damageType="bullet" multiplier="0.125"/>
  <DamageMultiplier damageType="collision" multiplier="1"/>
</DamageMultipliers>
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "To open an existing level, we need to access the **File** menu."

Warnings or important notes appear in a box like this.

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title via the subject of your message.

If there is a book that you need and would like to see us publish, please send us a note in the **SUGGEST A TITLE** form on `www.packtpub.com` or e-mail `suggest@packtpub.com`.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code and colored graphics for this book

You can download the example code files for all Packt books you have purchased from your account at `http://www.PacktPub.com`. If you purchased this book elsewhere, you can visit `http://www.PacktPub.com/support` and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/support`, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from `http://www.packtpub.com/support`.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
CryENGINE 3: Getting Started

In this chapter, we will cover:

- Opening a level using the CryENGINE 3 Sandbox
- Navigating a level with the Sandbox Camera
- Setting up a personalized toolset layout
- How to customize toolbars and menus
- Using the Rollup Bar
- Selecting and browsing level objects
- Restoring the CryENGINE 3 default settings

Introduction

The main focus of this particular chapter will be in getting the **CryENGINE 3 Software Development Kit** installed and having you up and editing a level in the Sandbox editor right away! One of the key things to keep in mind when learning a game compositing tool such as Sandbox is to remember to experiment and have fun! It is important not to forget that most of us (game developers) are trying to make things fun and not dreary and dull.

With such a powerful toolset waiting for you to dive in, let's get right to it!

Opening a level in the CryENGINE 3 Sandbox

As most people involved in the game's development process should be familiar with opening levels, this section will take you through the relatively straightforward task of opening a level within the CryENGINE 3 Sandbox editing tool.

Getting ready

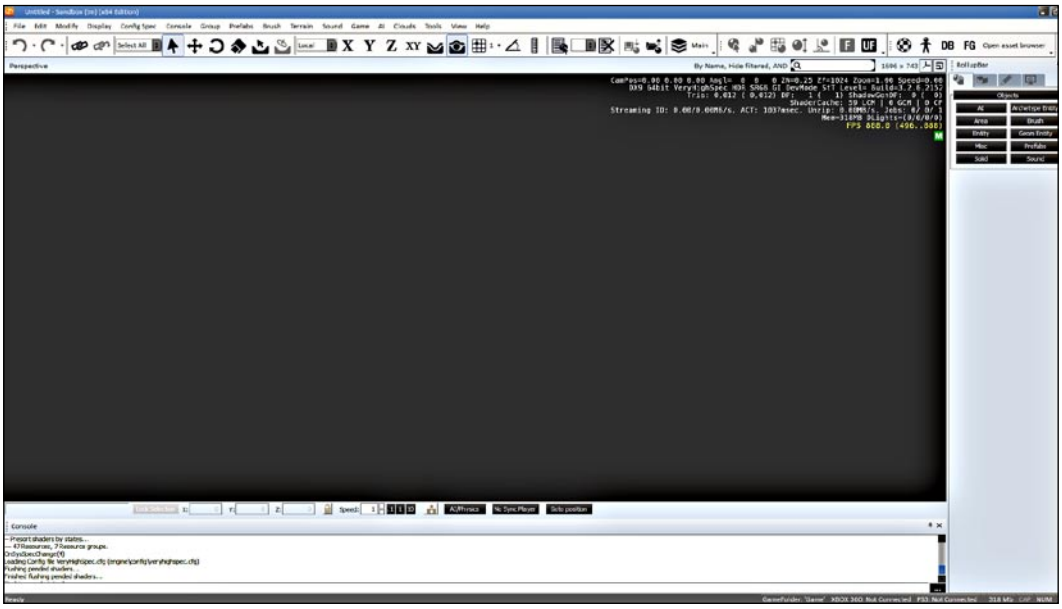
Having already located the `Editor.exe` in either your `bin32` or `bin64` folders, it will now be started in this section.

If a level is not already loaded, the editor's subsystems can still access assets and resources from your game. Keep this in mind as some tasks don't require the loading of a level.

How to do it...

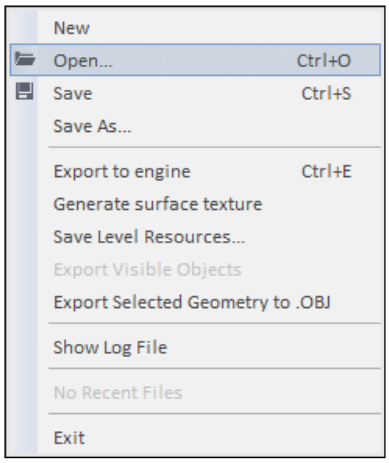
Let's get to opening a level:

1. Launch the `Editor.exe` located in either the `Bin32` or `Bin64` folder. You will be presented with an interface, as shown in the following screenshot:

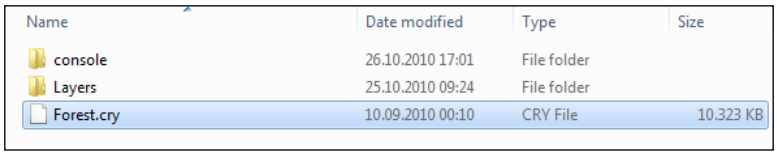


2. As there are already example levels installed with the SDK, we can open them.

3. To open an existing level, we need to access the **File** menu.



4. The **File** menu includes commands related to the handling of level files such as opening, saving, showing log files, and a list of recently-loaded levels.
5. As we want to open a pre-existing level, choose the **Open** option.
6. You will then be presented with a browser window defaulting to the `CryENGINE3/game/levels` folder.
7. Browse to `Forest` and open the folder.
8. Within the folder there is a `Forest.cry` file that contains raw level data for editing.
9. Open the `Forest.cry` file.



The editor will now start to load this level for you to start exploring!

How it works...

The editor reads the `.cry` files and can also access the subfolder layers within the level folder.

As the level loads, it reads the information present in the `.cry` file.

There's more...

You may want to know what the `.cry` file is composed of or even how to apply console command changes to individual levels.

What is a .cry file?

A `.cry` file is the principle level editing format for all levels built in the CryENGINE. It is actually an archive comprised of binary and XML data that is used only by the editor. You can open `.cry` files in the editor, or you can open them with an appropriate archiving program such as WinRAR.

Using a level.cfg

Similar to the `system.cfg`, the `level.cfg` is a file that is executed upon the loading of a level. The `level.cfg` can simply be stored in the level's directory. You may add console variables or level-specific configurations to this file.

See also

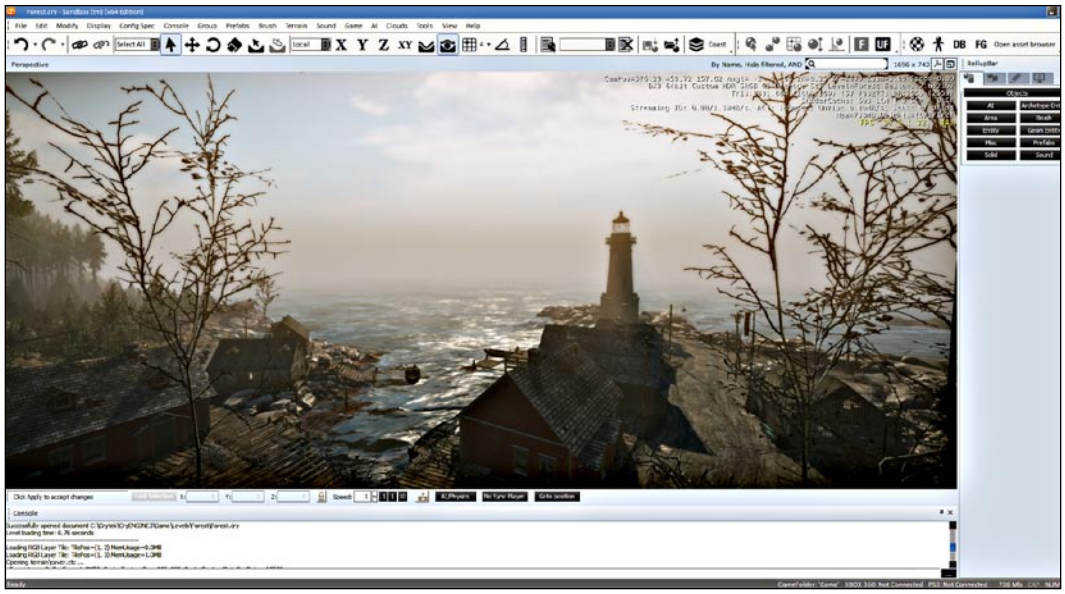
- Having launched the Sandbox, you can continue to the *Navigating a level with the Sandbox Camera* recipe in this chapter
- To get right to modifying a level, go to the *Selecting and Browsing level objects* recipe later in this chapter

Navigating a level with the Sandbox Camera

The ability to intuitively navigate levels is a basic skill that all developers should be familiar with. Thankfully, this interface is quite intuitive to anyone who is already familiar with the WASD control scheme popular in most *First Person Shooters Games* developed on the PC.

Getting ready

You should have already opened a level from the CryENGINE 3 Software Development Kit content and seen a perspective viewport displaying the level.

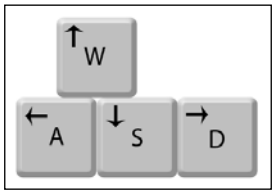


The window where you can see the level is called the **Perspective Viewport window**. It is used as the main window to view and navigate your level. This is where a large majority of your level will be created and common tasks such as object placement, terrain editing, and in-editor play testing will be performed.

How to do it...

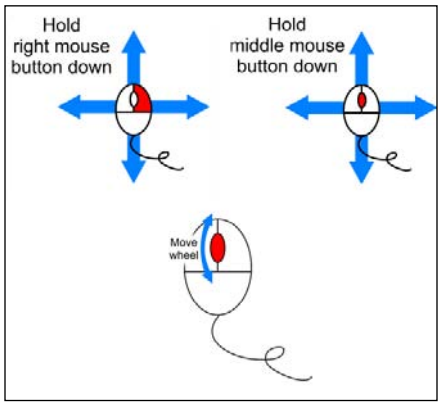
The first step to interacting with the loaded level is to practice moving in the Perspective Viewport window.

Sandbox is designed to be ergonomic for both left and right-handed users. In this example, we use the WASD control scheme, but the arrow keys are also supported for movement of the camera.



1. Press **W** to move forwards.
2. Then press **S** to move backwards.

3. **A** is pressed to move or strafe left.
4. Finally, **D** is pressed to move or strafe right.
5. Now you have learned to move the camera on its main axes, it's time to adjust the rotation of the camera.
6. When the viewport is the active window, hold down the right mouse button on your mouse and move the mouse pointer to turn the view.
7. You can also hold down the middle mouse button and move the mouse pointer to pan the view.
8. Roll the middle mouse button wheel to move the view forward or backward.
9. Finally, you can hold down *Shift* to double the speed of the viewport movements.



How it works...

The Viewport allows for a huge diversity of views and layouts for you to view your level; the perspective view is just one of many. The perspective view is commonly used as it displays the output of the render engine. It also presents you a view of your level using the standard camera perspective, showing all level geometry, lighting, and effects.

To experiment further with the viewport, note that it can also render subsystems and their toolsets such as flow graph, or character editor.

There's more...

You will likely want to adjust the movement speed and how to customize the viewport to your individual use. You can also split the viewport in multiple different views, which is discussed further.

Viewport movement speed control

The **Speed** input is used to increase or decrease the movement speed of all the movements you make in the main Perspective Viewport.

The three buttons to the right of the **Speed:** inputs are quick links to the **.1**, **1**, and **10** speeds.

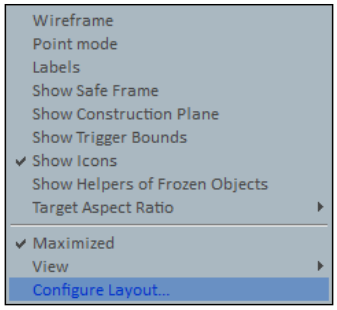
Under Views you can adjust the viewport to view different aspects of your level

Top View, Front, and Left views will show their respective aspects of your level, consisting of bounding boxes and line-based helpers. It should be noted that geometry is not drawn.

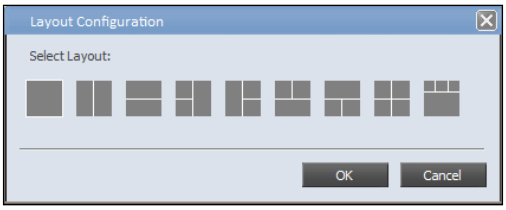
Map view shows an overhead map of your level with helper, terrain, and texture information pertaining to your level.

Splitting the main viewport to several subviewports

Individual users can customize the layout and set viewing options specific to their needs using the viewport menu accessed by right-clicking on the viewports header.



The Layout Configuration window can be opened from the viewport header under **Configure Layout**. Once selected, you will be able to select one of the preset configurations to arrange the windows of the Sandbox editor into multiple viewport configurations. It should be recognized that in multiple viewport configurations some rendering effects may be disabled or performance may be reduced.



See also

- To start building your own objects immediately, go to the *Making basic shapes with the Solids Tool* recipe in *Chapter 3*, *Basic Level Layout*
- To modify the terrain of the current level, go to the *Terrain Sculpting* recipe in *Chapter 2*, *Sandbox Basics*

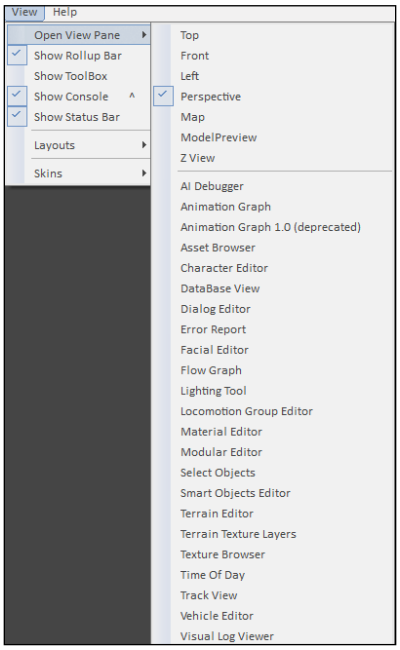
Setting up a personalized toolset layout

It would suffice to say that every user of Sandbox will have different preferences to how different views and toolsets should be distributed on screen. The CryENGINE 3 Sandbox allows for this kind of user-based customization and this recipe will take you through the use of some of the built-in tools for customizing your interface.

Getting ready

Before starting, it's important to introduce the view menu. The view menus allow you to turn various windows, toolbars, and subsystems on or off as well as open the various Sandbox extended editors and tool dialogs.

While experimenting with views, be aware that if you close a window and want to open it again, this can be done easily using the **View | Open View Pane** menu.



Another important toolset that you will likely want on your layout is the **Rollup Bar**.

The Rollup Bar is similar to the 3ds Command Panels for those already familiar with 3ds. It is a quick menu bar for the majority of the functions available to the editor exposed to the developer in an easily accessible format.

The final important tool you will likely want is the **Console**.

The Console is a direct command-line editor to the CryENGINE 3. This essentially allows access to various advanced functions within the Sandbox editor, including various debug and test profiles.

To start this tutorial, you should have Sandbox `Editor.exe` started.

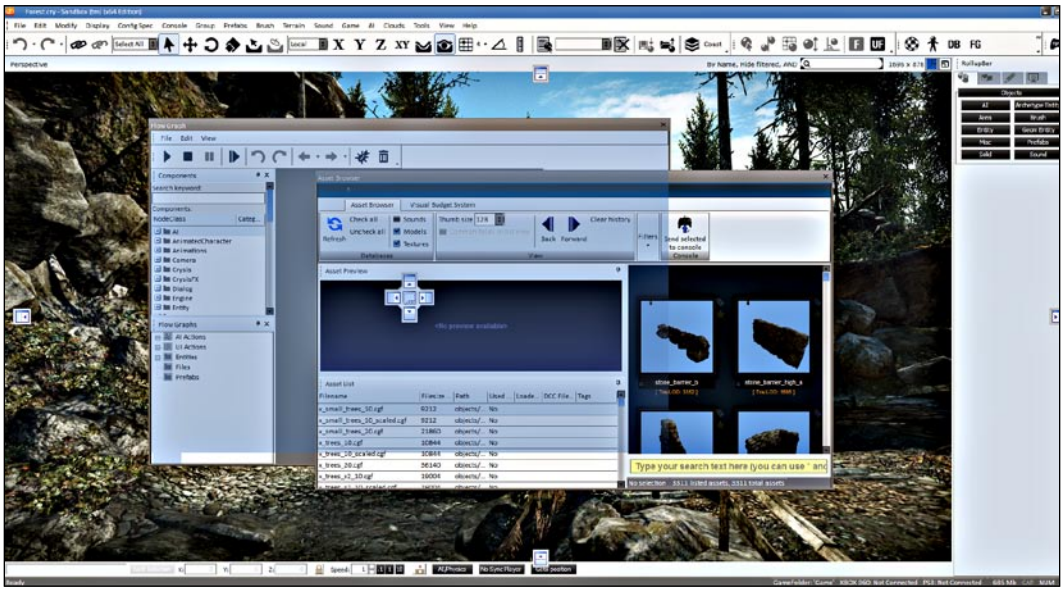
How to do it...

The first step of customization will be to learn how to scale and move the various windows in Sandbox around:

1. Let's first open a new window.
2. Click **VIEW** | **Open Viewpane** | **Asset Browser**.
3. This will open up a window containing the asset browser in the centre of your screen.
4. Ignore the contents for now as the asset browser will be explained later on in this recipe; let's resize the window.
5. Move the mouse pointer to the edge of the window, so that it turns into a double-ended black arrow. Click and drag the mouse pointer to scale the window.
6. Now that we have resized the window to our liking, let's use the docking toolbars to anchor the asset browser into the layout.
7. You can see the docking helpers whenever you drag a window over another window, or the Sandbox editor itself.
8. Click and drag the window from the title bar and move it over various docking helpers shown around the main view window.
9. Notice that once you release the mouse button, the window will dock itself into that location.

Now that the window is docked, we should learn how to undock it:

1. Similar to when we docked the window, drag the title bar again and move the selected window away.
2. Notice that the window maintains its original size and shape. You may thus want to resize the window once you have undocked it.
3. Another important interface to master is the ability to dock a window within other windows.
4. Go back to the **View** menu and open another window.
5. For this example, open the **Flow Graph** window.
6. Now, drag the **Asset Browser** window to the **Flow Graph** window.
7. You will observe the docking buttons being displayed again.
8. Use the lower, central button to dock the selected window in the lower half of the **Flow Graph** window.



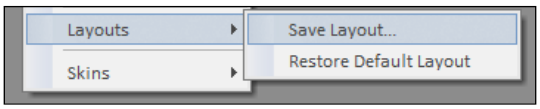
You can also dock windows at the top and sides of other windows using the other docking buttons:

1. To do this drag the title bar of the **Asset Browser** out of the **Flow Graph** window and away to another docking helper within the flow graph window to move it.
2. The final tool that is available to you in customization is **Docking a Window as a Tab in Another Window**.
3. For people using only one monitor, this is almost essential!

4. You may have noticed previously that when you move a window to another window, a new central, circular button is displayed.

5. This anchor will allow you to place the currently selected window as a tab within another window.
6. Drag the **Asset Browser** to the **Flow Graph** window, and on to the central dock as the tab button.
7. You will notice that there are now two tabs at the bottom of the window, **Flow Graph** and the **Asset Browser**.
8. You can now select each window by clicking on its corresponding tab.

9. Keep in mind that you can undock a window by dragging its tab to another part of the screen.
10. Once you are happy with your layout you can save this layout for easy loading later.
11. To do this, we will access the **Layout Configuration** window on the **Display** menu under **Configure Layout**.
12. Select **Save Layout** from the **Configure Layout** dialog.



13. You will then be presented with an opportunity to name this layout. Type the name of the configuration in this window and click **OK**.

How it works...

The docking helpers work very similar to windows office applications, so any users of those applications may be familiar with this system.

The save layout process creates a folder under the `CryENGINE3/editor` directory called `layouts`.

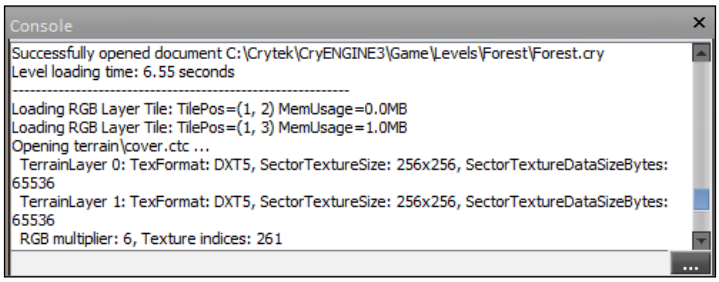
In this folder, it then saves a `.layout` file that is essentially an `.xml` file. This means that it could be edited by hand if required but can also be version controlled, which means multiple presets can be shared across large teams.

The Status Bar

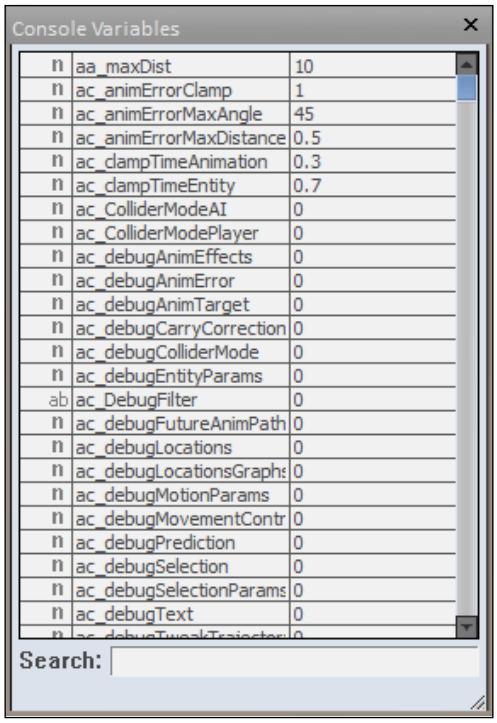
The Status Bar contains translation/rotation/scaling information for selected objects, editor interaction shortcuts, and camera controls.

The Console

The Console in the Sandbox editor is used to input variables. It can be visually toggled on or off by going to the **View Menu** and selecting **View Console** or by pressing the caret key (^) while the **Perspective Viewport** is selected.



In the editor, a full list of console variables can be accessed by double-clicking the input field on the **Console** to open the **Console Variables** window.



Search for variables with partial or complete commands. Information on individual variables can be shown by hovering the mouse over a Console Variable for a couple of seconds in the **Console Variable** window; text will then be displayed as a tool-tip.

The Toolbox

The Toolbox is a set of user-defined tools that contains some example shortcuts to useful editor command lines and different functionality. It can also be added to with user-specific shortcuts and/or console variables.

See also

- To get right to using some of the interfaces, go to the *Using the Rollup Bar* recipe later in this chapter
- To learn how to customize toolbars and menus, go to the next recipe

How to customize toolbars and menus

This section will now introduce you to the various toolbars and menus available in the Sandbox. With these toolbars, users can very quickly access many of the features of the Sandbox editor by using simple icons and groups of icons at the top of the interface. These toolbars can be configured to fit the preferences and needs of individual users.

Getting ready

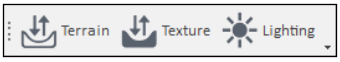
Before adjusting the toolbars, it is important that we explore a brief summary of the default toolbars that are available in Sandbox.

The **Standard ToolBar** contains open, save, hold, and fetch options.

The **EditMode ToolBar** contains various tools for level editing. These tools include undo and redo, link and unlink, select all, object movement/scaling, axes and terrain options, as well as object selection, saving, and loading.

The **Object ToolBar** contains tools for object alignment. The icons are go to selected object, align selection, align object to grid, set object(s) height, align object to the surface normal, and fix and unfix selected objects.

With the **Mission ToolBar**, you can select the current mission, duplicate a mission, delete a mission, and reload and edit mission scripts.



The **Terrain ToolBar** contains shortcuts to tools within the **Terrain Editor**, the **Terrain Texture Layer** editor, and **Terrain Lighting** dialog.

The **Dialogs ToolBar** contains icons used to access extender editor such as the **Materials Editor**, the **Character Editor**, the **DataBase View**, and the **Flow Graph Editor**.

The **Console ToolBar** has options specific to console game development. The buttons include sync data to console, automatically sync data to console, sync camera, options, load current level on console, and launch current level on console.

How to do it...

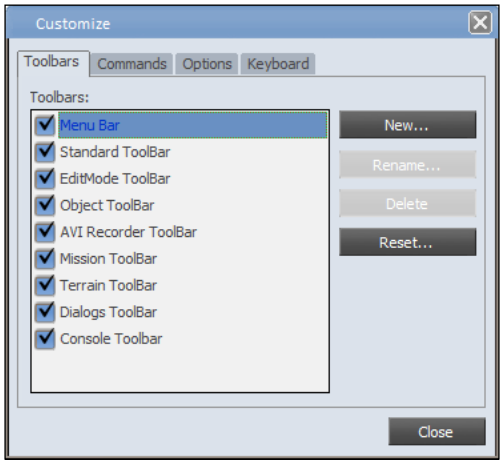
Now that we know about the default toolbars, let's go ahead and set up our layout:

1. To do this, we will need to access the **ToolBar** settings menu.
2. To access it, right-click anywhere on the **Icon Bar**.

3. This will display the **ToolBar** settings menu.
4. Selecting a toolbar from the list will display it or hide it on the main header.

Toolbars can be arranged horizontally at the top of the editor, vertically on the edges, or completely undocked from the editor.

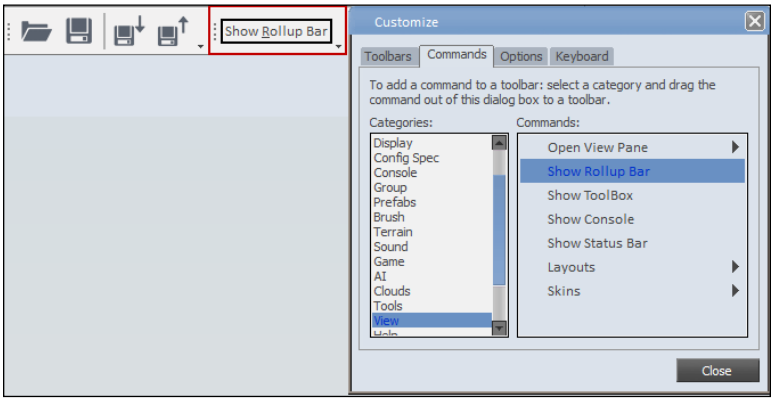
5. To customize these toolbars and to create new ones simply click on the **Customize** option at the bottom of the **ToolBar** settings menu.
6. The **Customize** dialog box allows users to customize preset toolbars, as well as create custom user toolbars.



7. The **New** button allows you to create a new custom toolbar. On pressing the button, a prompt will appear requesting a name for the new toolbar.
8. The **Rename** button allows you to rename any of the custom toolbars you have created.
9. The **Delete** button allows you to delete any of the custom toolbars you have created.
10. The **Reset** button returns any changes made to the toolbars back to default.
11. Click on **New** and name it `my_toolbar`.
12. You will then see the toolbar added as an empty container on the interface.

13. We can now add commands to this toolbar for easy and customized access!

14. The **Commands** tab allows you to drag-and-drop any of the icons in the **Commands** box into any toolbar, even custom ones.
15. Go to **View** in the categories options.
16. Click and drag the **Show Rollup Bar** command into your new toolbar.



17. Click **Close** to accept the changes to your custom toolbar.
18. To test its function, click the newly created button and notice that it now toggles the rollup bar on and off.

How it works...

Custom toolbars and overall toolbar configuration is saved in the registry information written by Sandbox to the PC. The toolbars are quite organic in the fact that you can undock and move them anywhere in the Sandbox interface, which, of course, makes it far easier to interface with some of them.

There's more...

You may want to explore some of the other tabs available within the **Customize** interface.

The Options tab

In the **Options** tab, there are some options that allow you to edit the way the interface reacts.

The first option is **Always show full menus**, which as the name suggests always shows the full menu of the currently selected menu.

The next is the **Show full menus after a short delay**, which will, after a short delay, display the full menu even if a skin collapses the menu to only show frequently used items.

The **Reset** menu and toolbars usage data deletes the record of the commands that you've used in the editor, restoring the defaults.

Under the **Other Header**, we have the **Large icons** checkbox. This displays large icons when the editor skin has the choice of using large or small icons.

We also have the **Show screen tips on toolbars** checkbox, which displays screen tips when the mouse is held over toolbar buttons.

There is a subcheckbox to the screen tips, which is to show shortcut keys in screen tips. This shows keyboard shortcuts along with the screen tips.

Finally, we have **Menu animations**. This changes how menus are displayed. The options include: (System default), Random, Unfold, Slide, Fade, and None.

Personalized menus and toolbars

The **Keyboard** tab allows the user to assign different shortcuts to certain functions within the editor.

You can browse through different categories using the **Category** drop-down list. Depending on the category selected, different commands will be listed in the commands frame. If the function is already assigned to a key, it will be shown in the **Key Assignments** frame.

To assign a shortcut key, you must have a command selected and then click within the **Press new shortcut key** textbox and that key will then be ready to be assigned to the selected command. To accept the shortcut assignment click the **Assign** button. You can also remove this later by using the **Remove** button.

Using the Rollup Bar

The **Rollup Bar** is one of the most commonly used tools within Sandbox. By default, it is located to the right of the viewport typically along the right edge of the interface. This is where entity parameters, settings, and controls are listed and accessed.

The **Rollup Bar** is split into four very different panels, which are accessed from their corresponding tabs.

The first tab contains the object and entity creation tools for the editor, as well as being the tab that will display all entity-specific information and dialogs.

The second tab has the overall environmental, vegetation, and terrain editing tools. The tools in this tab are used to modify the specific level you currently have loaded in Sandbox.

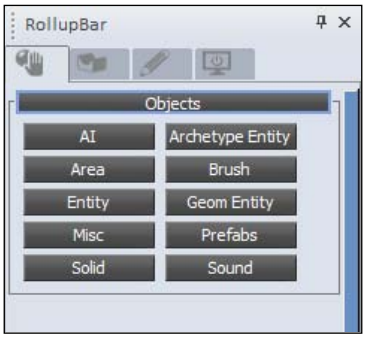
The third tab contains the display options.

The last one is the layer organizational tool.

Getting ready

In this task, we will only use the **Objects** and **Entities** tab, which is the first and default tab within the **Rollup Bar**. To access the majority of scene elements throughout this book, we will use this tab.

It holds interfaces to the various Database libraries and the brush database on your hard drive.



You must have a pre-existing level opened in Sandbox to complete this recipe.

How to do it...

Let's start using the Rollup Bar.

1. In the Rollup Bar, click on the **Brush** button.
2. This will display a browser linked to the `CryENGINE3/Game/Objects` directory of your build.
3. Brushes are compiled geometry containing no extra data other than collision. Typically, most levels are created with brushes as they are simple geometry.

At the bottom of the browser window you will see a dialog box that says **filter:**

1. This is a very useful feature when browsing for specific objects.
2. Type in `village` in the **filter** dialog.
3. Once you hit *enter* the browser will then simplify the contents of the browser to the folder that contained any objects with the name **village** in them.