Quick answers to common problems

Unity 4.x Cookbook

Over 100 recipes to spice up your Unity skills

Matt Smith Chico Queiroz

[PACKT] PUBLISHING

Unity 4.x Cookbook

Over 100 recipes to spice up your Unity skills

Matt Smith

Chico Queiroz

PACKT PUBLISHING

BIRMINGHAM - MUMBAI

Unity 4.x Cookbook

First published: June 2013

Production Reference: 1070613

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-84969-042-3

www.packtpub.com

Cover Image by J. Blaminsky (milak6@wp.pl)

Credits

Authors

Matt Smith

Chico Queiroz

Reviewers

Peter Bruun

Jate Wittayabundit

Acquisition Editor

Mary Nadar

Lead Technical Editor

Dayan Hyames

Technical Editors

Dennis John

Dominic Pereira

Project Coordinator

Amey Sawant

Proofreaders

Lindsey Thomas

Joel Johnson

Indexer

Hemangini Bari

Graphics

Abhinash Sahu

Production Coordinator

Prachali Bhiwandkar

Cover Work

Prachali Bhiwandkar

About the Authors

Matt Smith is senior lecturer in computing at the Institute of Technology Blanchardstown, Dublin, Ireland (`www.itb.ie`). In 1980 (you do the math) Matt started computer programming (on a ZX80) and has been programming ever since. In 1985, Matt wrote the lyrics, and was a member of the band that played (and sang, sorry about that by the way) the music on the B-side of the audio cassette carrying the computer game Confuzion (`wikipedia.org/wiki/Confuzion`).

Matt holds a bachelor's degree in Business Computing (Huddersfield University, UK), and as that was a bit boring, he went on to get a masters in Artificial Intelligence (Aberdeen University, Scotland), and a PhD in Computational Musicology (Open University, UK). Having run out of money after 10 years as a full-time student, he began his career as a lecturer and academic. He has been lecturing and researching on programming, artificial intelligence, web development, and interactive multimedia for almost 20 years, holding full-time positions at Winchester University and London's Middlesex University, before moving to his present post in Ireland in 2002. In recent years, Matt has replaced Flash-based 2D multimedia with Unity-based 3D game development and interactive virtual environments subjects for his computing and digital media undergraduates.

To keep himself fit, Matt took up the Korean martial art of Taekwon-Do (he developed and runs his club's website at `www.maynoothtkd.com`), and a group of his BSc students are now developing a Unity-based Taekwon-Do interactive "tutor" with Microsoft Kinect cameras. Some of his previous Irish-French student team games can be found and played at `www.saintgermes.com` (thanks for continuing to host these, Guillem!). Matt was one of the two technical experts for a recent multimedia European project for language and cultural student work mobility (`vocalproject.eu`).

Matt is currently struggling to learn Korean (for his Taekwon-Do), and Irish (since his daughter Charlotte attends an Irish-speaking school and he doesn't believe her translations of her teacher's report cards ...). In 2012, he started taking classical piano lessons again (after a 20-year gap), with a view to sitting exams starting May, 2013.

Matt's previous authoring includes contributions to *Serious Games and Edutainment Applications*, *Springer* (2011), *Musical Imagery*, *Routledge* (2001). He was also lead editor for *Music Education: An Artificial Intelligence Approach*, *Springer* (1994), and a technical reviewer for *Internet and World Wide Web: How to Program (3rd Edition) by Deitel, Deitel & Goldberg*, *Prentice Hall* (2003).

Thanks to my family for all their support. Thanks also to my students, who continue to challenge and surprise me with their enthusiasm for multimedia and game development.

I would like to dedicate this book to my wife Sinead and children Charlotte and Luke.

Chico Queiroz is a multimedia designer from Rio de Janeiro, Brazil. Chico initiated his career back in 2000, soon after graduating in Communications/Advertising (PUC-Rio), working with advergames and webgames using Flash and Director at LocZ Multimedia. Here he contributed to the design and development of games for clients, such as Volkswagen and Parmalat, along with some independent titles.

Chico has a Master's Degree in Digital Game Design (University for the Creative Arts, UK). His final project was exhibited at events and festivals, such as London Serious Games Showcase and FILE. Chico has also published articles for academic conferences and websites, such as `gameology.org`, `gamasutra.com`, and `gamecareerguide.com`.

He curated and organized an exhibition, held at SBGames 2009, which explored connections between video games and art. SBGames is the annual symposium of the Special Commission of Games and Digital Entertainment of the Computing Brazilian Society.

Chico currently works as a Digital Designer at the Computer Graphics Technology Group (TecGraf), within the Pontifical Catholic University of Rio de Janeiro (PUC-Rio), where he, among other responsibilities, uses Unity to develop interactive presentations and concept prototypes for interactive visualization software. He also works as a lecturer at PUC-Rio, teaching undergraduate Design students 3D modeling and Technology/CG for Games, in which Unity is used as the engine for the students' projects.

I would like to thank my friends, family, co-workers, and all who have made this book possible and have helped me along the way. Special thanks to Stefano Corazza, Anaïs Gragueb, and Oliver Barraza for their fantastic work at Mixamo; Eduardo Thadeu Corseuil, my manager at TecGraf, for giving me the opportunity of using Unity in our interactive projects. Peter Dam and Peter Hohl from TecGraf, and Paul Bourke from the University of Western Australia, for their help and advice on stereo 3D visualization; Aldo Naletto for sharing his knowledge on sound engineering; my students and colleagues at PUC-Rio Art and Design department.

I would like to dedicate this book to my wife Ana and my daughters Alice and Olivia. Thank you for all your love and support.

About the Reviewers

Peter Bruun is an independent game developer based in Copenhagen, Denmark. He loves beautiful games and old sci-fi B-movies from the 1950s. For many years Peter has been jumping from project to project as a freelance programmer in the games industry. More recently, he was the lead game programmer on the hit mobile game Subway Surfers, which has been played by millions of people worldwide.

Jate Wittayabundit was an interior architect for several companies in Bangkok, Thailand. Then, he moved to Toronto, Canada and is now a Game Developer and Technical Artist at `Splashworks.com` Inc. and hopes to build his own company in the near future.

He is passionate about gaming and new technology, especially Unity. He is also the author of *Unity 3 Game Development Hotshot*, *Packt Publishing*.

In his spare time, he loves to work on 3D software, such as Zbrush or 3D Studio Max. He also loves painting and drawing. Currently, he's trying to merge his architectural and 3D skills with his game development skills to create the next innovation game.

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

Game development is a broad and complex task. An interdisciplinary field covering subjects as diverse as Artificial Intelligence, character animation, digital painting, and sound editing. All those areas of knowledge can materialize as the production of hundreds (or thousands!) of multimedia and data assets. A special software application—the game engine—is required to consolidate all of those assets into a single product.

Game engines are specialized pieces of software, which used to belong to an esoteric domain. They were expensive, inflexible, and extremely complicated to use. They were for big studios or hardcore programmers only. Then along came Unity.

Unity represents true democratization of game development. An engine and multimedia editing environment that is user-friendly and versatile. It has free and indie versions and a Pro version that includes even more features. As we write this preface, Unity offers modules capable of publishing games to Windows, Mac, Linux, iOS, Android, XBox 360, Wii U, and PS3; as well as web-based games using the Unity plugins.

Today, Unity is used by a diverse community of developers all around the world. Some are students and hobbyists, but many are commercial organizations ranging from garage developers to international studios, using Unity to make a huge number of games—some you might have already played in one platform or another.

This book provides over 100 Unity game development recipes. Some recipes demonstrate Unity application techniques for multimedia features, including working with animations and using preinstalled package systems. Other recipes develop game components with C# scripts, ranging from working with data structures and data file manipulation, to artificial intelligence algorithms for computer controlled characters.

If you want to develop quality games in an organized and straightforward way, and want to learn how to create useful game components and solve common problems, then both Unity and this book are for you.

What this book covers

Chapter 1, *Getting Started with Unity 4.x*, is written for those who have just started, or are about to start, using Unity 4.x. It covers software installation, interface concepts, user preferences, and some workflow tips.

Chapter 2, *Using Cameras*, will explain recipes covering techniques for controlling and enhancing your game's camera. This chapter will present interesting solutions to work with both single and multiple cameras.

Chapter 3, *Creating Maps and Materials*, contains recipes that will give you—whether you are a game artist or not—a better understanding on how to use maps and materials in Unity 4.x. It should be a great resource for exercising your image editing skills.

Chapter 4, *Creating GUIs*, is filled with GUI (Graphical User Interface) recipes to help you increase the entertainment and enjoyment of your games through the quality of the interactive visual elements. You'll learn a wide range of GUI techniques, including working with scroll wheels for input, and displaying directional compasses, radars, and graphical inventory icons.

Chapter 5, *Controlling Animations*, demonstrates focusing on character animation, how to take advantage of Unity's new animation system—Mecanim. It covers everything from basic character setup to procedural animation and ragdoll physics.

Chapter 6, *Playing and Manipulating Sounds*, is dedicated to making sound effects and soundtrack music in your game more interesting. It also touches on playback and volume control techniques.

Chapter 7, *Working with External Resource Files and Devices*, throws light on how external data can enhance your game in ways, such as adding renewable content and communicating with websites. External devices, such as the Microsoft Kinect, can totally change the game's interactions. Learn about communicating with external resources and devices in this chapter.

Chapter 8, *Working with External Text Files and XML Data*, provides recipes for different methods to work with text files in general, and with XML text data specifically. This chapter is included because XML and other text-based data is common and very useful, both being computer and human readable.

Chapter 9, *Managing Object States and Controlling Their Movements*, relates to the many games that involve moving computer-controlled objects and characters. For many games animation components can be sufficient. However, other games use artificial intelligence for directional logic. This chapter presents a range of such directional recipes, which can lead to games with a richer and more exciting user experience.

Chapter 10, *Improving Games with Extra Features and Optimization*, provides several recipes providing some ideas for adding some extra features to your game (pausing, slow motion, 3D stereography, and securing online games). The rest of the recipes in this chapter provide examples of how to investigate and improve the efficiency and performance of your game's code.

Chapter 11, *Taking Advantage of Unity Pro*, is a concise chapter with interesting uses for some Unity Pro capabilities. It includes recipes for sound, render texture, video texture, and image effects.

What you need for this book

You will need a copy of Unity 4.x, which can be downloaded for free from `http://www.unity3d.com`. If you wish to create your own image files for the recipes in *Chapter 3*, *Creating Maps and Materials*, you will also need an image editor such as Adobe Photoshop (which can be found at `http://www.photoshop.com`) or GIMP, which is free and can be found at `http://www.gimp.org`.

Who this book is for

This book is for anyone who wants to explore a wide range of Unity scripting and multimedia features and find ready-to-use solutions to many game features. Programmers can explore multimedia features, and multimedia developers can try their hand at scripting.

From beginners to advanced users, from artists to coders, this book is for you and everyone in your team!

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, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "Unity's menu actually reads the `Standard Packages` folder content when starting up, instead of getting that information from somewhere else."

A block of code is set as follows:

```
private void ChangeMaterial()  {
  materialIndex++;
  materialIndex = (materialIndex % materialArray.Length);
  Material nextMaterial = materialArray[ materialIndex ];
  renderer.sharedMaterial = nextMaterial;
}
```

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: "Let's create a new material. Access the **Project** view, click on the **Create** drop-down menu and choose **Material**. Rename it to `Grid`."

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 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

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/submit-errata`, 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
Getting Started with Unity 4.x

In this chapter, we will cover:

- Installing Unity 4.x
- Setting your preferences
- Understanding and optimizing the User Interface
- Saving assets created in Unity as Prefabs
- Discovering Unity's content
- Importing your own content
- Importing Unity packages into your project
- Importing custom packages into your project
- Exporting custom packages from your project
- Adding custom packages to Unity's quick list
- Using the Project browser

Introduction

This chapter is tailored for those who are about to start using Unity or have just arrived to it. In this chapter, you will find some introductory steps into making this engine more comfortable and familiar.

Installing Unity 4.x

Unity is a very powerful and versatile game engine. It is available in both Indie (free) and Pro (paid) versions. In case you haven't installed Unity yet, this recipe will show you how to do it.

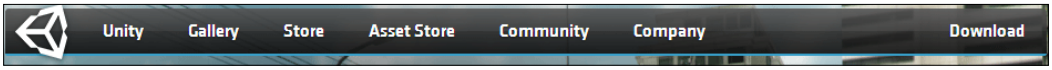
Getting ready...

You will need Internet access to follow this recipe.

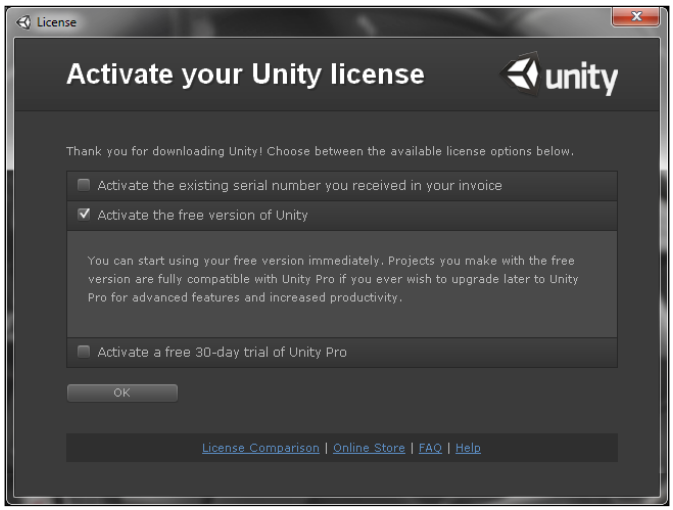
How to do it...

To Install Unity, please follow these steps:

1. Access the Unity website at `www.unity3d.com`.
2. Locate and click the **Download b**utton, placed in the top-right corner.



3. Now, on the **Download** page, click the button to get the latest version of Unity. Wait for the download to complete.
4. Run the Installer. This is a very straightforward process that will install everything you need in a couple of minutes.
5. Once the software is installed, run Unity. That should take you to the activation dialog where you can choose between activating Unity Pro (provided you have a valid serial number), Unity Free, or a 30-day trial of Unity Pro:



6. Select your choice and click **OK**. You should be prompted to log in or create an account.
7. Register (if necessary) and log in to activate your copy of Unity and start using it right away.

There's more...

You can expand Unity's capabilities and reach new audiences by adding more platforms to your editor.

Acquiring new licenses

iOS and Android exporters are now included free, and others can be bought from the Unity Store at `https://store.unity3d.com`.

Setting your preferences

Setting the editor to your preferences might sound superfluous to some. However, it could accelerate your development process and make Unity even more comfortable to use. In this recipe, we will learn how to adjust some of those settings to your taste.

How to do it...

To adjust Unity's preferences, follow these steps:

1. Inside the Unity editor, navigate to **Edit** | **Preferences...** (or, if you are using Mac OS, **Unity** | **Preferences...**).
2. As the **Preferences** window shows up, notice that it is divided into these sections: **General**, **External Tools**, **Colors, Keys**, and **Cache Server**.
3. Select the **General** tab. If you're working with multiple projects, you might want to leave the **Always Show Project Wizard** option checked.
4. Also, if you use OS X and are used to its native color picker, leave the **OSX Color Picker** option checked.
5. Now select the **External Tools** tab. In case you want to use a different script editor than Unity's built-in MonoDevelop, you can use the drop-down menu in **External Script Editor** to browse to your favorite application.
6. If **Image Application** is set as **Open by File Extension**, you might end up working with several image editors simultaneously. To avoid that, use the drop-down menu to browse to your favorite software.
7. Also, if you happen to develop to Android, make sure to browse to the SDK in **Android SDK Location**.

8. Let's move on to **Colors** tab. The default settings are fine, but feel free to change colors that make you most comfortable.
9. Now select the **Keys** tab. You might select any action to change its shortcut. Again, the default settings are perfectly fine. Use this opportunity to learn more about them.

There's more...

As you probably noticed, Unity's **Preferences** window has more options than was covered here. If you want a full explanation for each setting, please check the online documentation at `http://docs.unity3d.com/Documentation/Manual/Preferences.html`.

Changing the editor's player quality settings

Depending on your target platform, you might want to adjust the level of graphical quality of your game. This can be done through **Quality Settings**, which controls, for instance, the resolution of real-time shadows, or how much anti aliasing will be applied. Those options (and much more) are organized in levels that range from **Fastest** to **Fantastic**. If you want to experience a particular quality setting when running your game from the Editor, navigate to **Edit** | **Project Settings** | **Quality** and select it from the table in the **Inspector** view.

See also

- The *Understanding and optimizing the User Interface* recipe.

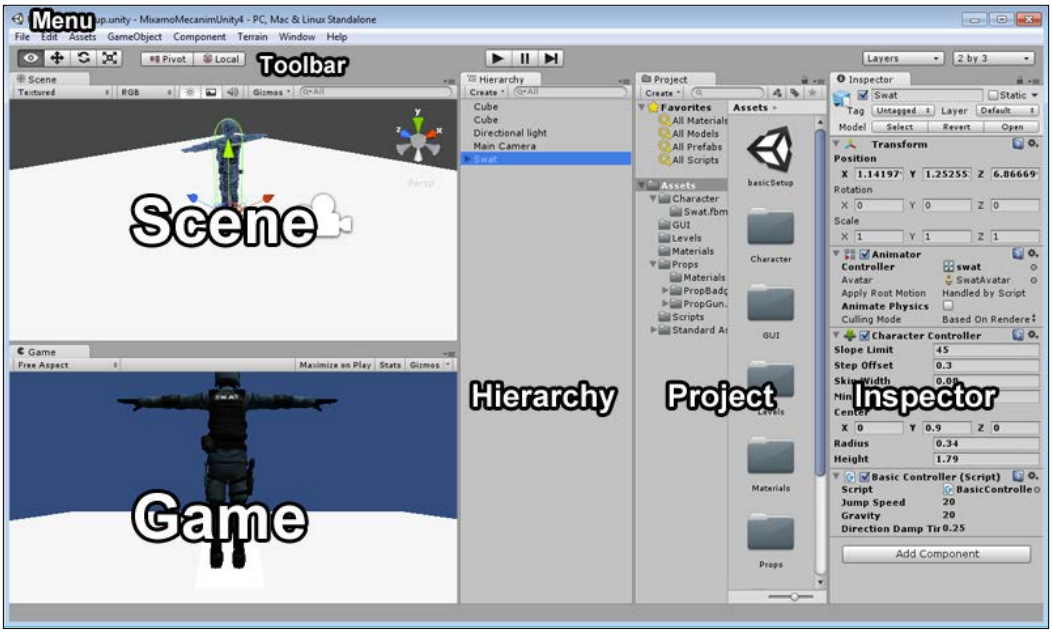
Understanding and optimizing the User Interface

Game engines, especially 3D-capable ones, can be a bit intimidating the first time you open them. Although Unity is particularly intuitive, user-friendly, and well documented, we have provided this recipe to show you how to operate inside its User Interface (UI).

How to do it...

Let's take a look at Unity's user interface:

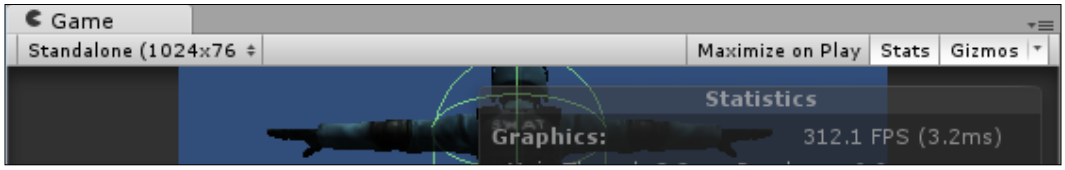
1. Run Unity. Unless you have previously changed it, its layout should initiate in Wide mode. Access **Window** | **Layouts** and choose another option, such as **4 Split** or **2 by 3**, and notice how the interface is organized into **Views**:



Let's take a look at those views:

- **Scene:** This view is used to position, rotate, scale, and select game objects, and also navigate your level.
- **Game:** This is the place to play and test your game. It will reproduce the player's experience as accurately as possible.
- **Hierarchy**: Game objects (as diverse as characters, cameras, level geometry, lights, and even GUI textures) placed in our scene will be listed here.
- **Project**: This is where you create, organize, and access your game assets. From 3D models and 2D textures to C# scripts and Prefabs, every re-usable element will be listed here.
- **Inspector**: From the Inspector, you can configure any game objects (selected from the Hierarchy view) or assets (selected from the Project view). That includes changing its **Transform** settings, configuring existing components and attaching new ones. Also, you can adjust other preferences for your game, once you have accessed them from the menu, in the Inspector view.
- **Toolbar**: Includes transform tools (used for manipulating game objects and navigating the scene), control tools (used for playing / pausing and stopping the level), and drop-down tools (used for managing layers and layouts).
- **Menu**: Gives access to a diverse list of commands covering asset import/export, preferences setting, game object creation, components, terrain, layout, and documentation.

2. If you want to customize the layout any further, drag and drop the views to relocate and/or dock them.
3. If you like your custom layout, save it through the **Window** | **Layouts** | **Save Layout...** menu.
4. When testing your game, it might be a good idea to check the **Maximize on Play** button, in the **Game** view. Also, if you work with more than one display monitor, you could drag the **Game** view into the second display, leaving a display exclusively for the Editor.
5. You can also adjust the **Game** view resolution. It's a good idea to test your game running on its standard standalone resolution and every supported aspect ratio.
6. In case you want to check the graphics performance of your game during testing, you should turn on the **Stats** button (you can also turn it off during testing, if so you wish).
7. Finally, activate **Gizmos** if you want them to be drawn at runtime, making it easier to spot rays, colliders, lights, cameras, and so on in your scene, as shown here:



8. There is another view you should pay attention to: the **Console** view. Access it by navigating to **Window** | **Console**. This is a very important view when it comes to debugging your game, as it displays errors, warnings, and other debug output during testing.
9. Another interesting view (for those with Unity Pro) is the **Profiler** (**Window** | **Profiler**), where you can check out detailed statistics of your game performance in real time.

There's more...

To get an extensive explanation on each UI feature, please check out Unity's documentation at `http://docs.unity3d.com/Documentation/Manual/LearningtheInterface.html`.

See also

- The *Setting your preferences* recipe.
- The *Searching assets with the Project browser* recipe.

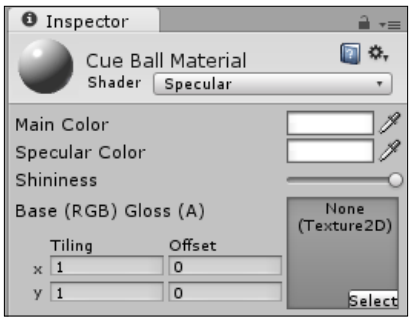
Saving assets created in Unity as Prefabs

You can easily create primitive geometry with Unity. In this recipe, we will create a game object from Unity's resources and keep it in our project as a **Prefab**.

How to do it...

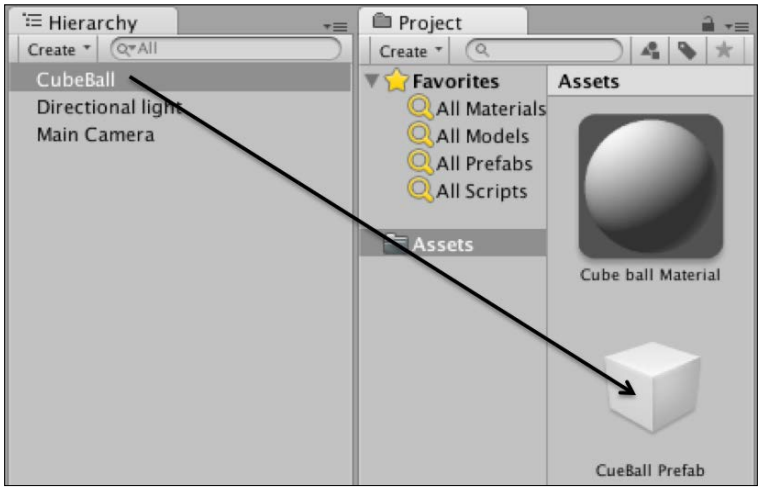
To create a Prefab, follow these steps:

1. Inside the Unity editor, navigate to **GameObject** | **Create Other** | **Sphere**.
2. In the **Hierarchy** view, right-click **Sphere** and choose the appropriate option from the context menu to rename it to **Cue Ball**.
3. Now, in the **Project** view, click on the **Create** button and choose the **Material** option. Then, rename the new material to **Cue Ball Material**.
4. In the **Project** view, select **Cue Ball Material**. Then, in the **Inspector** view, change its **Shader** value to **Specular**.
5. Also, set **Specular Color** to white and set its **Shininess** to the maximum, as shown in the following screenshot:



6. From the **Project** view, drag **Cue Ball Material** into the **Cue Ball** game object, in the **Hierarchy** view.
7. Select **Cue Ball**. Then, access **Component** | **Physics** | **Rigidbody**. That should attach a **Rigidbody** component to that game object.
8. Now that your game object is complete, click on the **Create** button in the **Project** view and choose the **Prefab** option. Then, rename it to **Cue Ball Prefab**.

9. Drag the **Cue Ball** game object from the **Hierarchy** view into the Prefab in the **Project** view. Your game object is ready to be re-used in this project.



How it works...

In Unity, game objects can be stored as Prefabs. This is very useful in case you want to re-use a game object in several levels or instantiate it through scripting. Adobe Flash users can think of it as the Unity equivalent of MovieClips.

There's more...

There are many other ways to use Unity's built-in resources. Here are some ideas:

Adding external files

In this recipe, we haven't used any external asset. However, there's no reason you could not have imported a texture and used it as the `Cue Ball Material` base map, for instance.

Taking your Prefabs to another project

Also, if you plan of re-using your Prefab in other projects, you can do it by exporting it as a custom package.

Creating other kinds of game objects

As you have probably noticed, spheres are not the only entities you can create directly with Unity. Other primitives are also available, as well as many other types of entities: lights, camera, GUI textures, and so on. Navigate to **GameObject** | **Create Other** and experiment with the options available.

See also

- The *Exporting Custom packages from your project* recipe.

Discovering Unity's content

As you enter Unity for the first time, you might think you'll need to build and code everything from scratch. However, Unity comes with several collections of content called **Packages**, designed to save you time when implementing commonly required features.

How to do it...

Let's find out what's inside Unity's standard packages:

1. Inside the Unity editor, access the **Assets** menu.
2. Expand the **Import Package** submenu.
3. You will see a list of available packages from Unity. These are filled with ready-to-use content.

How it works...

Unity makes implementing commonly requested features easy by making them available as packages ready to be imported and used in your project. These packages include First-Person and Third-Person Character Controllers, Image Effects (Pro Only), Terrain textures and assets, Skyboxes, Water, Tree Creator, and more.

There's more...

There are other ways to learn from ready-made material. Here are some:

Studying the sample project

Unity also comes with a sample project ready to be dissected by you. It automatically opens the first time you start the software.

Downloading more resources

You can find and download even more resources, such as packages, projects, tutorials and assets, from Unity's resources page at `http://unity3d.com/support/resources/`.

See also

- The *Importing Custom packages into your project* recipe.

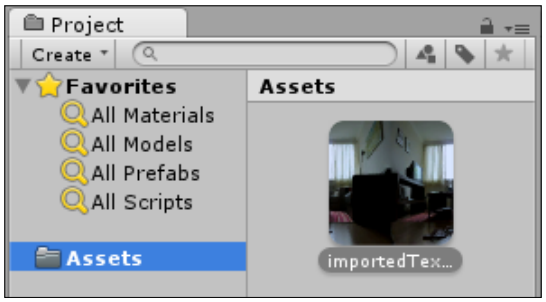
Importing your own content

After you have created a 3D model, audio clip, movie clip, or texture, you can import it into your project. In this recipe, we will learn how it can be done.

How to do it...

Follow these steps to import an asset:

1. Inside the Unity editor, access the **Assets** menu.
2. Select the **Import New Asset...** option.
3. Browse to your file and click **Import**.
4. Your file should be now listed in the **Project** browser, as shown here:



How it works...

Unity makes a copy of your file, converts it to an appropriate format (if necessary) and saves it into the **Project** browser's **Assets** folder.

There's more...

Here's a couple of helpful pieces of information on the subject:

Organizing it with the Project view

Unity updates its **Project** view whenever a new file is added to the **Assets** folder. You could then save or export your work directly into that folder. You could also paste or move multiple files into there. However, you should not reorganize or rename your imported files via your OS file management system (Window's Explorer or Mac OS Finder), as this could damage important information kept by Unity about those files.

Exporting your assets to Unity 4.x

If you are not sure about how to prepare and export your work to Unity, or which file format you should use, please check out Unity's documentation at `http://docs.unity3d.com/Documentation/Manual/AssetImportandCreation.html` for a very comprehensive guide on the subject. Some other useful pages regarding the subject are:

- Importing objects from 3D Studio Max: `http://docs.unity3d.com/Documentation/Manual/HOWTO-ImportObjectMax.html`
- Importing objects from Maya: `http://docs.unity3d.com/Documentation/Manual/HOWTO-ImportObjectMaya.html`
- Export FBX – How-to: `http://docs.unity3d.com/Documentation/Manual/HOWTO-exportFBX.html`

See also

- The *Importing Custom packages into your project* recipe.

Importing Unity packages into your project

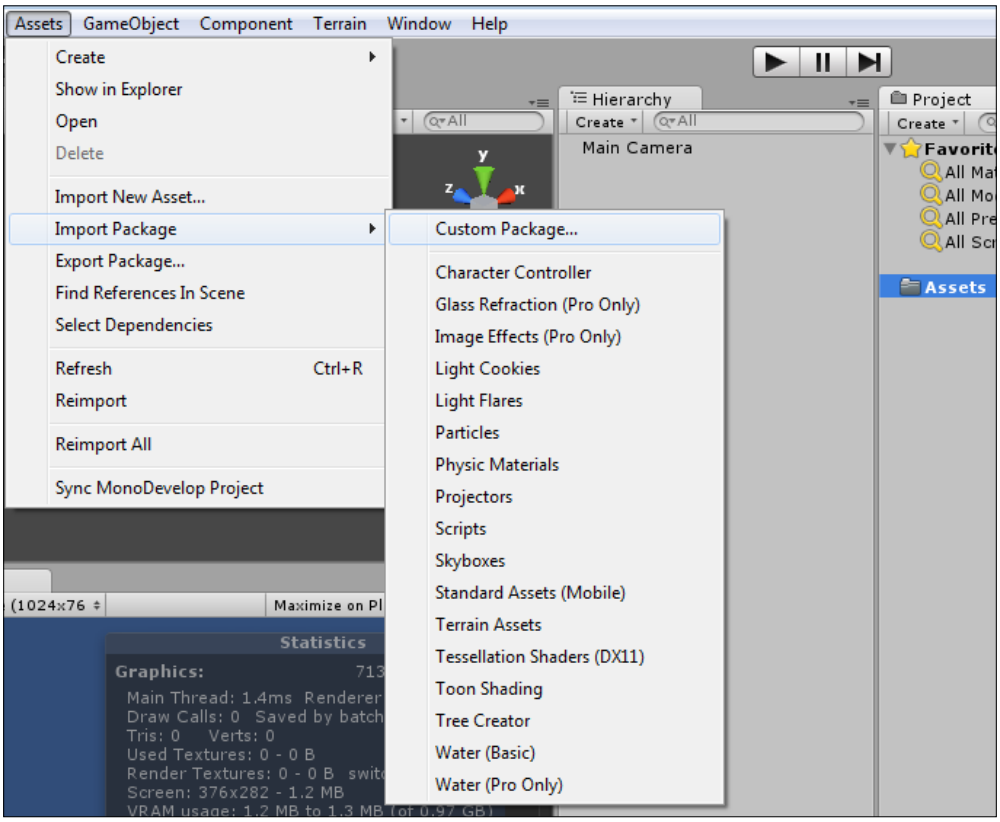
The packages provided by Unity can save you a lot of development time. They usually contain resources (such as texture maps, materials, and so on) and fully implemented features ready to go into your project. When creating a new project, Unity offers to install those packages into the **Assets** folder. However, if you've missed it at first, you can still import them into your project at any time.

How to do it...

To import a Unity package, follow these steps:

1. Inside the Unity editor, access the **Assets** menu.
2. Enter the **Import Package** sub-menu and choose a package from the list.
3. Make sure every needed component is selected and click **Import**.

4. Package contents should be ready and listed in the **Project** view.



How it works...

Unity installation files include a number of packages that can be imported into your project as ready-to-use resources. Inside those packages are all the assets needed to implement a specific feature or functionality. Once imported, new features can be accessed through the **Project** view (and dragged and dropped into your level) or through newly added menu items (the **Tree Creator** package, for instance, adds the **Tree** option into the **Create Other** sub-menu of the **GameObject** menu).

There's more...

Importing Unity packages can also be done through the Project Wizard. When starting a new project, check the boxes of the packages you want to import.

See also

- The *Importing custom packages into your project* recipe.
- The *Exporting custom packages from your project* recipe.
- The *Adding custom packages to Unity's quick list* recipe.

Importing custom packages into your project

Custom Unity packages are available from a variety of sources, and they can be very helpful when developing a project.

Getting ready

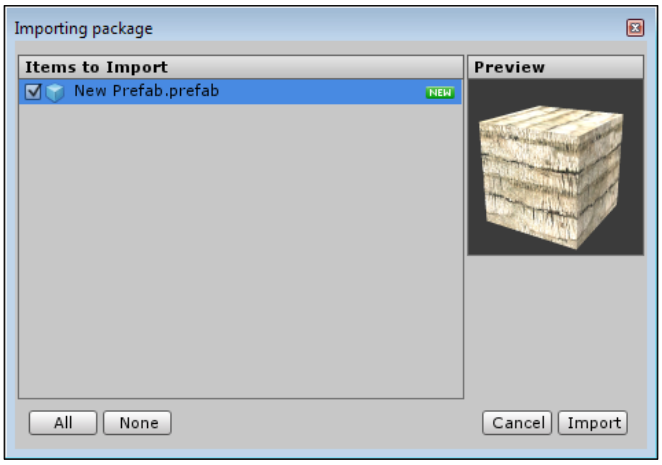
As its title implies, this recipe requires a custom package to be imported. If you need one for testing purposes, please use the one inside the folder named `0423_01_09-11`.

How to do it...

To import a custom package, follow these steps:

1. Inside the Unity editor, access the **Assets** menu.
2. Enter the **Import Package** sub-menu and choose the **Custom Package...** option.
3. Browse to the package file you have saved on your disk and click **Open**.
4. Preview package contents in the top-right **Preview** window, if you like.
5. Make sure every needed component is selected and click **Import**.

6. Package contents should be ready and listed in the **Project** view.



Downloading the example code

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.

How it works...

Custom packages are commonly used to distribute a number of assets inside a single compressed file. As they are made by third parties, the content inside those packages may vary, as they can include scripts, 3D models, texture maps, materials and any other file handled by Unity. Once imported, the package content is uncompressed into your project's **Assets** folder, and can be accessed through the **Project** window.

There's more...

Third-party made content can also be found, downloaded, and bought from the Unity Asset Store. For more information, access `http://unity3d.com/unity/asset-store/`.

See also

- The *Importing Unity packages into your project* recipe.
- The *Exporting custom packages from your project* recipe.
- The *Adding custom packages to Unity's quick list* recipe.

Exporting custom packages from your project

Creating packages can be a very useful and practical way of storing your game objects and assets for future use and reference. If you want to save a feature, a group of assets, or even a prefab from the project you are currently working on, it's a good idea to export them as a package, so you can easily import them into your future projects.

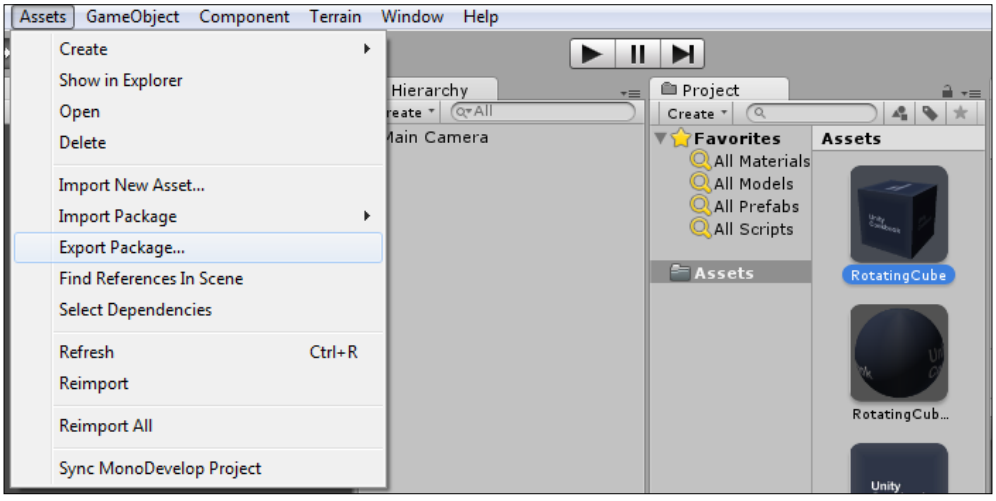
Getting ready

In order to export a package, you will need a project containing some assets. If you need one for testing purposes, please use the one inside the downloadable content for this book, which can be found inside the `0423_01_10` folder.

How to do it...

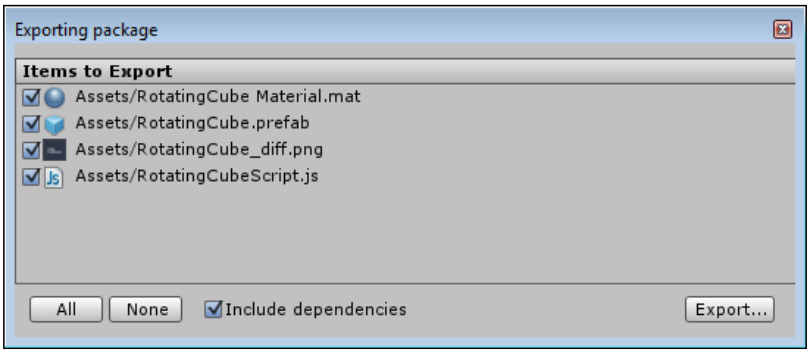
To export content as a custom package, follow these steps:

1. Select the **RotatingCube** prefab in the **Project** view.
2. Go to the **Assets** menu and choose the **Select Decencies** option. This will highlight, inside the **Project** tab, all assets that are linked to the **RotatingCube** prefab.
3. Once again, select only the **RotatingCube** prefab.
4. Go to the **Assets** menu and choose **Export Package...**. A new window will now pop up:



5. Inside the **Exporting Package** window, make sure the **Include dependencies** checkbox is selected. It is important that checkboxes for all listed objects are also selected.

6. Click **Export** and save your package into your disk. You can give it any name you want (although a name similar to **RotatingCube** will make things easier later, when you want to use it).
7. Your custom package is ready to be imported.



How it works...

By exporting a package, you have stored your selected objects and dependencies into a single compressed file. Importing them to your project will uncompress them into its **Assets** folder.

See also

- The *Importing Unity packages into your project* recipe.
- The *Importing custom packages into your project* recipe.
- The *Adding custom packages to Unity's quick list* recipe.

Adding custom packages to Unity's quick list

If you have one or more packages you'd like to include frequently in your projects, it might be a good idea to add them to Unity's package quick list.

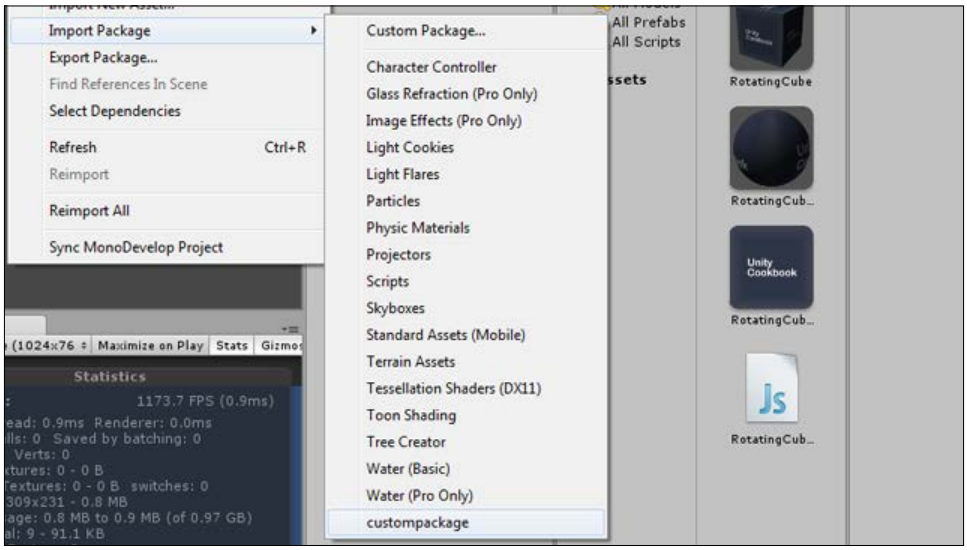
Getting ready

In order to complete this recipe, you will need a custom package (any package will do). If you need one for testing purposes, please use the one inside the `0423_01_09-11` folder.

How to do it...

To add a custom package to the quick list, follow these steps:

1. Using your file manager (Windows Explorer on Windows, Finder on Mac OS), browse to the package file and copy it by pressing *Ctrl* + *C* or *Command* + *C*.
2. Go to Unity's `Editor` folder. On Windows, that would typically be `C:/Program Files (x86)/Unity/Editor` or `C:\Program Files\Unity\Editor`. On Mac OS, it should be `Applications/Unity`.
3. Access the `Standard Packages` folder.
4. Paste the previously copied package into this folder.
5. Restart Unity. It should now be listed on the **Assets** menu, under the **Import Package** sub-menu, as shown in the following screenshot:



How it works...

Unity's menu actually reads the `Standard Packages` folder content when starting up, instead of getting that information from somewhere else. This is very practical, as it always reflects the actual content of that folder and also allows the user to quickly retrieve his favorite packages.

There's more...

Custom packages stored in the `Standard Packages` folder will also appear in the **Create New Project Wizard** window, making it simple to add them to new projects.

See also

- The *Importing Unity packages into your project* recipe.
- The *Importing custom packages into your project* recipe.
- The *Exporting custom packages from your project* recipe.

Using the Project browser

It doesn't matter how organized you keep your project folders, there will be times when you will need to search for one or more specific assets. To make things easier, Unity 4 includes the **Project** browser. In this recipe, we will learn how to save time by using it.

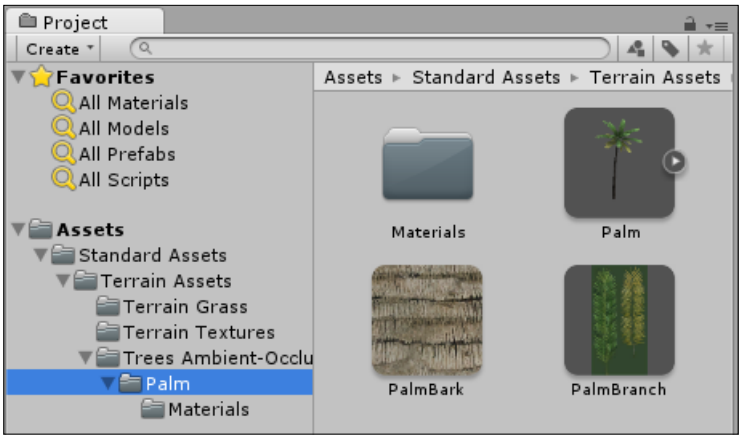
Getting ready

All we need to follow this recipe is a collection of assets. We will use Unity's **Terrain Assets** to populate our **Project** view.

How to do it...

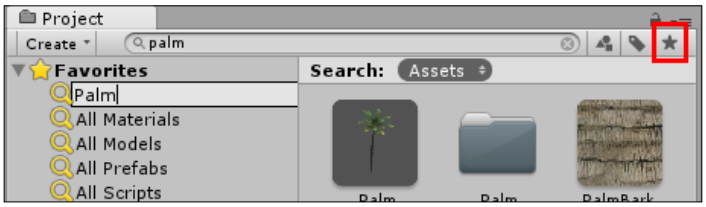
Let's take a look at the **Project** browser:

1. Import the **Terrain Assets** package (**Assets** | **Import Package** | **Terrain Assets**).
2. Browse through the **Assets** subfolders to see the files that have been imported and how they are organized, as shown in the following screenshot:



3. Now let's search for all the assets containing the word "Palm" in their names. On the search field located above, type in `Palm`. Observe how every file and folder is listed on the search results window.

4. Click on the **Save Search** button (the one with the star icon) and name your search as `Palm`.



5. We could filter our search results by tag or file type. Click the **Search by Type** button (the one with a circle, triangle, and square) and select only the **Models** and **Textures** categories. You can select multiple options by holding *Ctrl* (Windows) or *Command* (Mac).
6. Click on the **Save Search** button and save it as a new filter (name it `Palm Models and Textures`).
7. Browse through the other filters saved in **Favorites**—they are shortcuts to browse through specific file types (**All Materials**, **All Models**, and so on).

How it works...

Unity's new browsing system scans through the assets and lets you save and organize your search results.

See also

- The *Understanding and optimizing the User Interface* recipe.