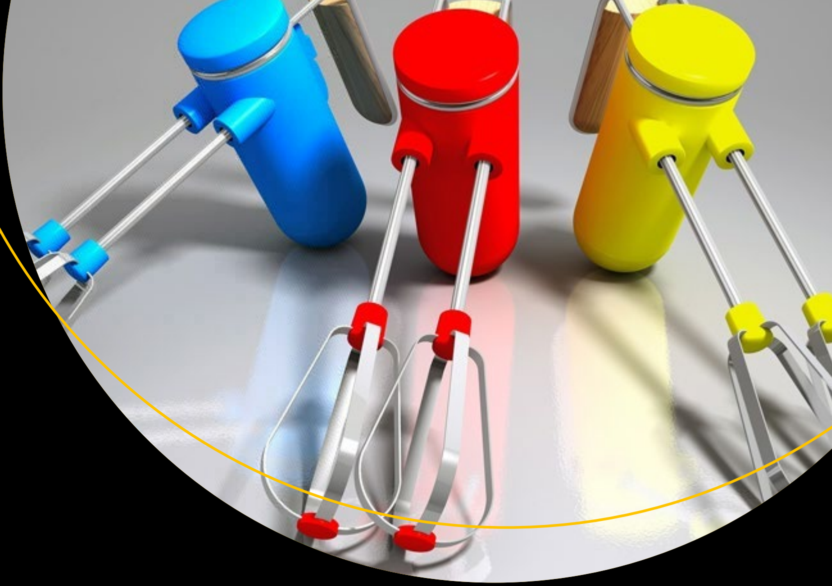

Creating Game Environments in Blender 3D

Learn to Create Low Poly Game Environments

—

Ezra Thess Mendoza Guevarra

Apress®

Creating Game Environments in Blender 3D

Learn to Create Low Poly Game Environments

Ezra Thess Mendoza Guevarra

Apress®

Creating Game Environments in Blender 3D: Learn to Create Low Poly Game Environments

Ezra Thess Mendoza Guevarra
Laguna, Philippines

ISBN-13 (pbk): 978-1-4842-6173-6 ISBN-13 (electronic): 978-1-4842-6174-3
https://doi.org/10.1007/978-1-4842-6174-3

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Spandana Chatterjee
Development Editor: Matthew Moodie
Coordinating Editor: Divya Modi

Cover designed by eStudioCalamar

Cover image designed by Pixabay

Distributed to the book trade worldwide by Springer Science+Business Media New York, 1 New York Plaza, New York, NY 10004. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book’s product page, located at www.apress.com/978-1-4842-6173-6. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

To Father God.

To my mother and father.

To my fellow Filipinos.

To my relatives and friends.

To all PWDs (especially Epileptic).

Table of Contents

About the Author xi

About the Technical Reviewer xiii

Acknowledgments xv

Introduction xvii

Chapter 1: Getting Started 1

- Game Environment 1
- Game Environment Artist 2
- Game Environment Artists’ Skills 2
- Game Assets 3
- Essential Technical Skills 3
- Types of Games 4
- Low Poly vs. High Poly? 7
- Blender’s Installation Process 10

Chapter 2: Let’s Create! 11

- Blender Preferences 11
- Workspaces 12
- The Objects 14
  - Mesh Objects 15
  - Curve Objects 17
  - Surface Objects 20
  - Metaballs Objects 23
  - Text Objects 27

- Grease Pencil Objects 27
- Armature Object 29
- Lattice Object 31
- Empty Objects 31
- Light and Light Probe Objects 34
- Camera and Speaker Objects 38

Modeling Specific Tools 39

- Extrude Region 40
- Extrude Along Normals 42
- Extrude Individual 43
- Extrude to Cursor 43
- Inset Faces 44
- Bevel 45
- Loop Cut 45
- Offset Edge Loop Cut 46
- Knife Tool 47
- Bisect Tool 47
- Poly Build Tool 48
- Spin Tool 49
- Spin Duplicates 50
- Smooth Tool and Randomize Tool 50
- Edge Slide Tool and Vertex Slide Tool 51
- Shrink/Flatten Tool 51
- Push/Pull Tool 52
- Shear Tool 52
- To Sphere Tool 53
- Rip Region Tool and Rip Edge Tool 54

Sample Project - Pawn Chess Piece 55

Sample Project - Rook Chess Piece 64

Sample Project - Knight Chess Piece 67

Sample Project - Bishop, Queen, and King 72

Chapter 3: Color It! 79
Color Fundamentals 79
Color Theory 79
Color Meaning 81
Color Harmony 84
Color Scheme 84
Hue, Saturation, Values (HSV) and Others 89
Color Systems 91
UV Workspace 96
3D Viewport in UV Workspace 97
The UV Editor of UV Workspace 114
UDIM in Blender 124
Chapter 4: Texture It! 129
Texture Paint Workspace 129
3D Viewport in Texture Paint Workspace 130
Blending Mode 132
Mix Blend Mode 134
Darken Blend Mode 135
Multiply Blend Mode 136
Color Burn Blend Mode 136
Linear Burn Blend Mode 137
Lighten Blend Mode 138
Screen Blend Mode 139
Color Dodge Blend Mode 140
Add Blend Mode 141
Overlay Blend Mode 142
Soft Light Blend Mode 143
Hard Light Blend Mode 144
Vivid Light Blend Mode 145
Linear Light Blend Mode 146
Pin Light Blend Mode 147

Difference Blend Mode 148
Exclusion Blend Mode 149
Subtract Blend Mode 150
Hue Blend Mode 151
Saturation Blend Mode 152
Color Blend Mode 153
Value Blend Mode 153
Erase Alpha and Add Alpha Blend Modes 154
Tool Settings 155
Brush 155
Texture 156
Texture Mask 157
Stroke 158
Falloff 160
Cursor 161
Symmetry 161
Texture Slot 162
Mask 163
Options 164
Annotation Stroke Placement 165
Properties 165
Color Palette 166
Workspace 166
Image Editor in Texture Paint Workspace 167
Tool Settings 169
Advanced Settings 169
Brush Tip 171
Tiling 171
Sample Project 172
UV Mapping 173
Texturing 180

Chapter 5: The Finale 191

Importing Game Assets in Blender 191

Importing Using .DAE Format 193

Importing Using .ABC Format 197

Importing Using .OBJ Format 202

Exporting Game Assets in Blender 208

Open Source Game Engine 216

Godot Engine (godotengine.org) 216

Stride (stride3d.net) 222

Armory3D (armory3D.org) 223

Appendix A: HotKeys 225

Appendix B: Blender’s History 235

Index 239

About the Author

Ezra Thess Mendoza Guevarra graduated with a bachelor of science degree in information technology from STI College, the Philippines. Though she's become a web developer, her passion for the arts, which started in her childhood, never went away. In 2016, she became interested in 3D modeling. Ezra is the administrator of the website `www.didyouknow45.wordpress.com` through which she shares her research.

Despite being epileptic, she continues to pursue her dreams and "break the walls." A researcher and passionate artist, Ezra is currently using her skills as a freelancer.

About the Technical Reviewer

Mieren Taylor is a 3D generalist who works with independents and small businesses to create product visuals and promotional material, as well as patching pipelines in various productions. Some of the projects she has worked on include making product assets for JOV Photo Lab, 3D Sarees for Yes!Poho; creating 3D visuals for Kairma Air filters, and many more. She has been working with Blender as a freelancer for 5+ years and has every intention to carry on using this amazing software and exploring the new features of its latest version for all new projects.

You can see Mieren's latest work on her website `www.Tomations.com`.

Acknowledgments

First of all, I'd like to thank my dear readers for purchasing my second book. Of course, without you, this book's purpose will just be in vain.

I'd also like to thank the efforts of all the staff who worked with this book. Despite being in a pandemic, you guys were able to work hard on this book. I don't know all of you, but of course, you know who you are.

Of course, thank you Ms. Spandana Chaterjee, Ms. Divya Modi, Mr. Matthew Moodie, and Ms. Mieren Taylor for your assistance, effort, and patience. Thanks for everything.

Thank you to my dearest mother, Cecilia Guevarra; and dearest father, Alexander Guevarra, for your support on everything.

Thank you to my HIJCC family who always pray for every member. Thank you also to my friends and previous employers who have given me memories and necessary experiences.

Of course, I will always thank my number one partner here, Father God, who always gives me strength and wisdom. Thanks for everything.

Introduction

Good day my dear reader! Wherever you are coming from, whatever time you are reading this book of mine, I'd like to greet you and say, "Welcome to the world of 3D game!"

Game is something that is been enjoyed by anyone. Whatever gender you are, whatever age you are, wherever country you came from, there is always a chance you have already played a game. It's just that now, many games are already digital.

From physical games to digital, from pixel or 2D to 3D, games already have a long history. They have even developed lots of genres and types along the way.

We are now in a time that most of the games developed are in 3D. Even the past popular games like Mario Brothers have been adopted to 3D for the new era.

But creating a game is divided into two parts: designing and development. In this book, we will discuss the designing part.

In this book, we will talk about how to create a basic 3D game environment using Blender 3D – from modeling, to texturing, and adding lighting.

So, let's start learning end I hope you enjoy this book!

CHAPTER 1

Getting Started

Good day, dear readers! Welcome to Chapter 1 of my book, which is about getting started creating game environments in Blender 3D.

Here we're going to talk about what exactly a game environment is, the role of a game environment, and the skills needed to make it in this field. We're going to talk about some tools that will be able to speed up the process, too, in creating your project with Blender.

So, that's enough for our introduction – let's get started.

Game Environment

Igi-global.com defines a game environment as being a dimension that brings game rules, objectives, subjects, and theoretical aspects together as a whole to provide an interactive flow of activity.

This means that a game environment isn't just the appearance or the graphics that you can see in the game. When you say game environment, it involves the rules, purpose, and target – meaning the whole mood of the game.

You might also encounter the word **level design** and be confused between the two, as I used to be. **Level design** is about the challenges that the player will encounter in the game while **game environment** is about the aesthetics, architectural design, and mood you will create in the game.

In game design, you should be able to convey your feelings to your target audience or player.

Note "Environmental design establishes the world." – Josh Byser, Level Design vs. Environmental Design at Medium.com

---

Game Environment Artist

Let's now proceed to discuss the role of an environment artist, as well as the skills and tools needed for this position.

A game environment artist is the one who creates the world of the game. For example, if you create a game with a cooking simulation theme, he or she is the one who will create the restaurant feel with vibrant color. Ah! Don't be confused when you see the term **prop artist**. In the given example theme, restaurant game, a prop artist is the one in charge of creating the *game assets* like dishes, chairs, etc., but the position of prop artist depends on the studio. Their jobs can also be done by a game environment artist. In other words, it means creating game assets that pertain to every object that can be seen in the games; creating game rules; creating a game system that is the flow of the game; and creating a game concept, which is the idea of the game or the story of the game and all fall under the game environment artist role. But depending on the studio you are working with, you can divide it into different to increase efficiency in each area. For example, if you hire someone for concept art only, you can create a concept for a game that can attract many target users. This is usually done by big studios to create an outstanding game while some indie game studios only assign that listed role to one or two artists because of a lack of budget; and in some indie studios, they will hire artists and randomly assign tasks based on what the artist can do.

Game Environment Artists' Skills

So now, let's talk about the skills necessary for a game environment artist. According to Robert Hodri, a senior 3D artist and environment artist at Id Software in his article at Arstation.com, having a passion for video games and art in general would be a plus for a game artist.[1] I'd like to add, if you have the passion, then working with games will not be work for you, but rather play since that's what you love, though this isn't required by companies.

Since those who create a game environment or game assets are also called artists, **creativity skills** are obviously present. Don't worry, everyone is born with creative skills. It just so happens that not everyone has been able to harness this skill of theirs as they grow. Just like communication skills, creative skills must be trained as you grow up so they will not be lost. If you ever feel that you have already lost them, don't worry. You can bring them back by practicing again with them.

[1]https://magazine.artstation.com/2017/03/game-environment-artist/

Next to creative skills are **communication skills**. There will be a time that you will communicate with different types of people. Without this type of skill, you might fail to deliver your idea to your team. You also need to have the ability to work independently because there are times that you will work alone. So, in other words, you should be **flexible**. This means that you can easily adapt to what the projects need.

Communication skills are one big asset. The flexibility to easily adapt to sudden changes in plans and time management are also essential, especially if you have other things that you're doing aside from creating games. Also important is decision-making. You need to learn to decide when to say **yes** and when to say **no**. I guess you don't want to die before your game get sick in the market, right? It's just a joke but to remind you to think of your health, too, and not just sit in front of a computer 24 hours a day and drink coffee. You need to learn patience also. Yeah, patience is a skill now!

As you probably notice, I discuss more about soft skills rather than technical skills. Remember, technical skills can easily be learned, but soft skills need to become habit before they become part of you.

Game Assets

Game Assets. I already gave a sneak peek of game assets, but what exactly are game assets? Game assets are anything that is part of the game: objects, scenes, materials or textures, lighting – everything is called game assets.

Essential Technical Skills

Let's talk about the technical skills that could help you increase your productivity in creating a game environment.

Since we're creating a game, number one on the list is the game engines. Learning the game engines will help you know how your props or assets will be implemented. We have tons of game engines around here nowadays. Popular ones are Unity3D, Unreal Engine, and CryEngine. We also have open source game engines like UPBGE and Armory3D. They are both a base in Blender, so making a game asset from Blender and importing it to these two will not be hard.

If you already want to create more outstanding textures and materials, Photoshop and Substance Painter are there to assist you. These two software are capable of helping you to create more creative texture since they have features that are specifically for that part. Blender has its own features that can be found in texture workspaces, but I will also introduce these to you so you can have options.

Although Blender also has its features for sculpting that can be found in the Sculpting workspace, I'd like to introduce to you these two: Zbrush and 3D Coat. If you want to create more stunning assets where you need to sculpt, like in characters, you can use them.

Just take note that Photoshop, Substance Painter, Zbrush, and 3D Coat are all paid software. If you want a free one aside from Blender that you can try on, here are some:

- Gimp is a free open source software that is known to be an alternative to Photoshop but I can say that in order to create texture and materials here, it's not an easy task unless you have already mastered this software.
- SculptGL is a free web application that is an alternative to Zbrush and 3D coat. You can visit it here: `https://stephaneginier.com/sculptgl/` and since it was a web application, you can't use it without internet but I like it as an alternative one. It's easy to use.

Well, the above list contains options for you, but this book will focus on creating game environments with Blender3D. So, in the end, we will be with Blender 3D.

Types of Games

We also need to know the different types of games before we start creating a game so we will know what type of game we will create, who will be our target, etc.

In general, we have three main types of games: Party Games, Tabletop Games, and Video Games.

When it comes to video games, we also have different genres. These include:

- Action-Adventure games: A combination of core elements of action and adventure.
- Adventure games: Where its main elements are focused on an adventure theme.

- Escape games: They are a kind of game with settings of a locked room and you will find a clue on how to escape from it. It's a sub-genre of adventure games.
- First Person Shooter (FPS): This is a game that is centered in a weapon-based combat in a first-person perspective. The player sees the game through the eyes of the main character of the game.
- Third Person Shooter: Here, the player can see the game in a third-person perspective. This is a game where you can see the main character fight and move rather than see what's the perspective of the main character.
- Multiplayer online battle arena: A sub-genre of strategy games. For the best description, think *Dota 2* and *League of Legends.* Yeah. Those kinds of games where there are players teaming up, creating two teams to fight against each other online.
- Platform games: This is a sub-genre of action games. This type of genre focuses on games where the player must jump or climb to avoid obstacles in the suspended platforms. Its features often involve uneven terrain that has different heights.
- Real-time strategy games: This is another sub-genre of strategy games. In this game, the two players will protect their assets by constructing new assets/units or destroying the other player's assets/units.
- Role-Playing video games (RPG): This is where the player controls the main character of the game toward a certain goal in its game world. *Legend of Mana* and *Pokémon* are some good examples.
- Simulation games: This type of game is created to simulate a real-life scene.
- Sports games: Games where its main elements are focused on sports themes.
- Casual games: Games with simpler rules, simple gameplay, and shorter sessions. It's usually targeted to the masses more than a specific audience like the other genres.

- Flash games: Games that you can play in the browser. They are usually made with HTML5, CSS, and JavaScript. But now, there are also some game engines that can help you easily create web-based games like Godot Engine.
- Mini games: These are the small games usually included inside a video game. You can call this a bonus game or side game.
- Alternate reality games: These are the kinds of games in which the main platform is the real world while the player gets instructions from different forms of media. For example, someone sends a fax of page 1 of a manuscript or novel that is soon to be published. Some may reject it because it was just a piece of paper, but some notice that there is an anomaly in the words and try to decode it, search it, and see that there are instructions in it indicating what to do next and so on. A real-life example of this was the game *Year Zero.*[2] There are no announcements about the game. For players to be able to participate, they must realize that on the back of the Nine Inch Nail 2007 European tour shirt, there is a code that will reveal the word **I am trying to believe** and after they search it, they get this website – IAmTryingToBelieve.com – about strange phenomenon and the game continues.
- Augmented reality games: Pokémon Go. Yes. If you played Pokémon Go, you know how these kinds of games work; but okay, I'll also explain. This kind of game is like a mix of the real world and virtual world.

There's a lot of types of games around here nowadays. As technology evolves, games evolve too. There might be some types that I'm not able to list, but at least we have some here and can consider them as we can think about when we start to create a game.

[2]Extra Credits: ARGs https://www.youtube.com/watch?v=tiU4AYPdIOw

Low Poly vs. High Poly?

Low Poly vs. High Poly? There's a lot of this question on the internet, but is there really such a thing as **low poly vs. high poly**? Okay. Let's start by defining each term.

The word poly comes from the word polygon, which is a two-dimensional shape made of straight lines and angles. In mathematics, a polygon is a closed shape where the sides are all in line segments, and each side must intersect with exactly two other sides but only at their endpoints. We also have different types of polygons, and they are named based on how many sides they have. In Blender 3D, polygons are shapes with three or more sides defined by three-dimensional points called vertices, connected by lines called edges; and the interior region of the polygon is called faces.

Polygons, in theory, can have any number of sides but are commonly broken down into triangles for display. In general, the more triangles in a mesh, the more detailed the object is and becomes high poly but becomes harder for the computer to display it. In order for you to optimize the whole scene, the number of triangles in the scene must be reduced, by using low poly meshes.

What is the basis if one object is a low poly or a high poly? Well, the answer is unknown. What? Unknown? Well, not exactly unknown – just like there is no explanation for it. What I mean with this is there is no definite basis if one object is low poly or high poly. An object will be called low poly or high poly depending on the poly count or number of polygons used in your game asset in relation to first, its target hardware; second, the kind of game you will create; third, the position of the game assets in your game environment; and fourth, the hardware you will use in creating your game. Let me explain it further.

The game Super Mario 64 is considered low poly at this time, but during its release, it was a major achievement. This is because as time passes by, technology advances and standards in the design also become higher than before. This is why what we called high poly before is low poly now.

Both high poly and low poly have their own parts. High poly is important to achieve a detailed surface while low poly is important for game engines, subdivision modeling, and low-polygon proxy geometries of high poly ones for rigging and animation. It's just that both of them have their own consequences when you use them.

When it comes to high poly, the problem lies in controlling an enormous amount of vertices, maybe from sculpting, subdivision surfaces made in low poly, curves or nurbs surfaces, or generated by 3D scanning etc., while in low poly, the polygons' problems lie in topology, which has to follow some rules so that the shading looks right.

In creating a game, you will not just create a game base on a one-sided computer specification, right? For example, I create an RPG game. It's not enough that I create an interesting design. It's not enough that I create interesting game play. It's not enough that I create an interesting game concept. I should think about how my target audience will be able to play my game. If every part of my game is very detailed like how you see in movies like *Jumanji*, will a computer with 4GB RAM be able to handle it? Will a cellphone with 2gb RAM be able to handle it? Oh, come on. Don't laugh. Those specs still exist. There are those who are still using Windows XP service pack 2 thinking it is the safest Windows version among all. So you need to know your target market and your target market's ability when you create your game so that your effort will not be wasted.

Another thing: when you compare the high poly assets created in a game in high poly models used in films, obviously the one used in a game will be called low poly compared to the other. This is why some said low poly and high poly are relative terms. This means that it really depends on the usage of the model and when the model is created.

What's the point of the previous examples? First, we don't have an exact poly count that we will say is the maximum number of low poly or a minimum number for high poly. No. We don't have that. Second, you need both low poly and high poly. When creating a game, you want some parts to be emphasized and look better, especially the part that is close to the screen. But since we're thinking of making the game lighter, you can use low poly in some parts of the game that are not that close to the screen. What do I mean by this? It's the level of detail. To look good close up, degrade those in the background.

There are many techniques created in order to use low poly and high poly in a game efficiently without making it harder for computers, and here are some of those:

- You can convert low poly to high poly by using MultiRes modifiers to sculpt high poly while retaining low poly geometry.
- You can convert high poly to low poly through Retoplogy or decimation, and the details from high poly can be baked into texture maps.
- For textures on low poly, you want them two times larger than a texture that has same size on a model as pixels on screen because of the so-called **texture filtering**.

- For high poly, you want polygons small enough so that the surface detail will be well defined.
- Use normal or bump mapping by making a low poly object appear to contain more detail than it does.

Okay, what is this texture filtering? Texture filtering deals with how a texture or the 2D image is displayed in 3D. It takes data of the texture – for example, its color – to improve the quality of the texture that will be displayed on the screen.

Let's discuss some kinds of texture filtering:

- Bilinear Filtering is the simplest method of texture filtering. When pixels fall between texels or textured, it samples the four nearest texels to find its color.
- Trilinear Filtering smooths the transition between **mipmaps** by taking samples from both.
- Anisotropic Filtering (AF) significantly improves texture quality at oblique angles. When you look at a snapshot of a 3D game with an AF off and a snapshot with AF on, you will see how the texture, especially in grounds, becomes more detailed and even improves its colors of some game assets. It has 2x, 4x, 8x, and 16x flavors. This refers to the steepness of the angle the filtering is applied to.

Let's have a little side note before I end up this section.

What are Mipmaps? Mipmaps are smaller, pre-filtered versions of a texture image, representing different levels of detail of the texture. They are often stored in sequences and progressively smaller textures called mipmap chains, with each level half smaller compared to the previous one. It is used for situations when the distance of the object and the camera can be changed. As the object become far away from the camera, the texture or texel become smaller and smaller. The textures will have to be scaled down in the process called minification filtering. The problem with this method is when you need to sample the entire texture at runtime for an object that can only be a single pixel wide.

This is the purpose of mipmaps. Instead of sampling just a single texture, the application can be set up to switch between any of the lower resolution mipmaps in the chain, depending on the distance from the camera.

The use of mipmaps can improve the image quality by eliminating aliasing effects caused by oversampling textures, and they can also improve performance because they increase cache efficiency and as full-size textures will not be needed as often, the lower resolution mipmaps will fit more easily in the texture cache.

Blender's Installation Process

First of all, of course you need to download it from its website www.blender.org. I'd like to make a note that if you want to download other versions, like the older one, rather than clicking the button in the header of the site that says Download Blender, you can go and click the Download tab ➤ go to the part of the page where it says "Go experimental" and click "Get Blender Experimental." Here, you can see all the Blender versions from the oldest.

You also have two options when downloading a Blender release. You can see it in the download page, under the MacOS, Linux, and other versions drop menu. Here you can see that Blender has an .exe file and a portable file for Windows.

For Windows: when you install the .exe file, you just only need to right-click and open or run it and follow the direction of the installation setup, unless you have modifications you wanted to do in its installation: for example, for the drive that will be installed, you can click the **advance** option and follow the steps. When installing the zip or portable file, you just need to extract it with the software you have: for example, WinZip or WinRAR.

For MacOs: when you install the dmg (disk-images) file, double-click it and then drag **Blender.app** into the application folders. When you install using the zip file, extract it and drag the **Blender.app** into the applications folder.

For Linux: when you install from Package Manager, I'd like to note that some Linux distribution may have a specific package for Blender in their repositories. Installing Blender via the distribution's native mechanism ensures consistency with other packages on the system. Be aware, too, that the package may be outdated compared to the latest official release. When you install from Blender.org, download the Linux version for your architecture and extract it to the desired location.

For more about the installation process, you can visit docs.blender.org/manual/en/latest/getting_started/installing/index.html.

So, this is the end of Chapter 1. I hope you learned something in this chapter. In the next chapter, we will discuss the first part of creating a game environment using Blender. I'll be using Blender 2.82 when demonstrating the project.

CHAPTER 2

Let's Create!

Good day, dear readers! Now we're here at Chapter 2 of our book. This chapter covers the topics related to modeling tools and features of Blender. I'll also be discussing a bit about some basic stuff for beginners of Blender so they can follow along without difficulty.

So, let's get started!

Blender Preferences

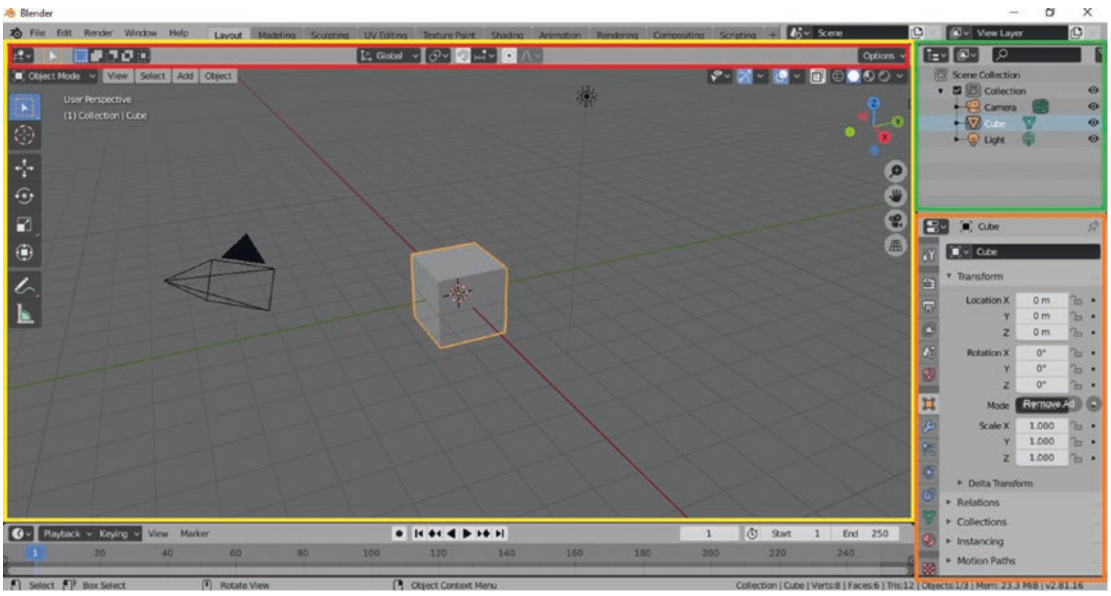

Figure 2-1. *Blender 2.82: Layout Workspace(Red:Tool Setting; Yellow:3D viewport, Orange:Properties;Green: Outliner)*

In Blender 2.80, the default color of the interface is dark with colored icons; and for Blender 2.82 (Figure 2-1), the color is light gray. But don't worry, guys! If you want to use the theme from 2.80 instead of 2.82, you can just go to edit ➤ preferences ➤ themes ➤ and go to the select menu, as shown in Figure 2-2, and choose Blender Dark.

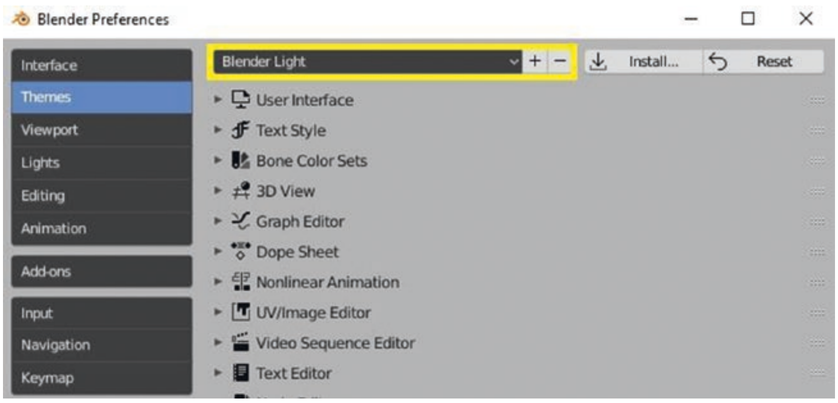

Figure 2-2. *Blender Preferences*

This Blender Preference appears as a new small window after you click **preference** in the **edit menu**. You can also add your own edited theme here by clicking the plus icon beside the select menu and deleting an existing theme by clicking the minus icon.

Now let's discuss one of the basic things that we need to review before digging into an advanced lesson: workspaces.

Workspaces

Workspaces are essentially predefined window layouts. You can also create your own workspaces by clicking the plus icon beside the sculpting workspace tab. Workspaces help you create your project with ease since they are already set up for different projects like modeling, animating, sculpting, texturing, shading, rendering, etc. For this chapter, since our focus will be on modeling, I will discuss the layout workspace and modeling workspace, which are both helpful in modeling aspects.

Take a look at Figures 2-3 and 2-4 to see the look of layout workspace and modeling workspace.

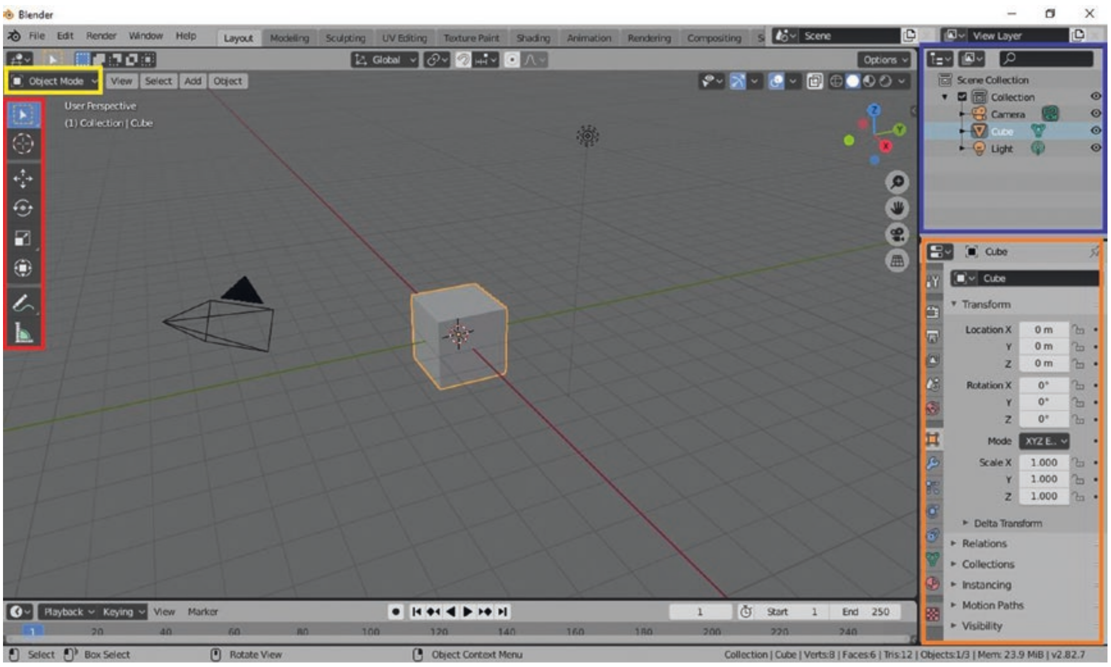

Figure 2-3. *Layout Workspace*

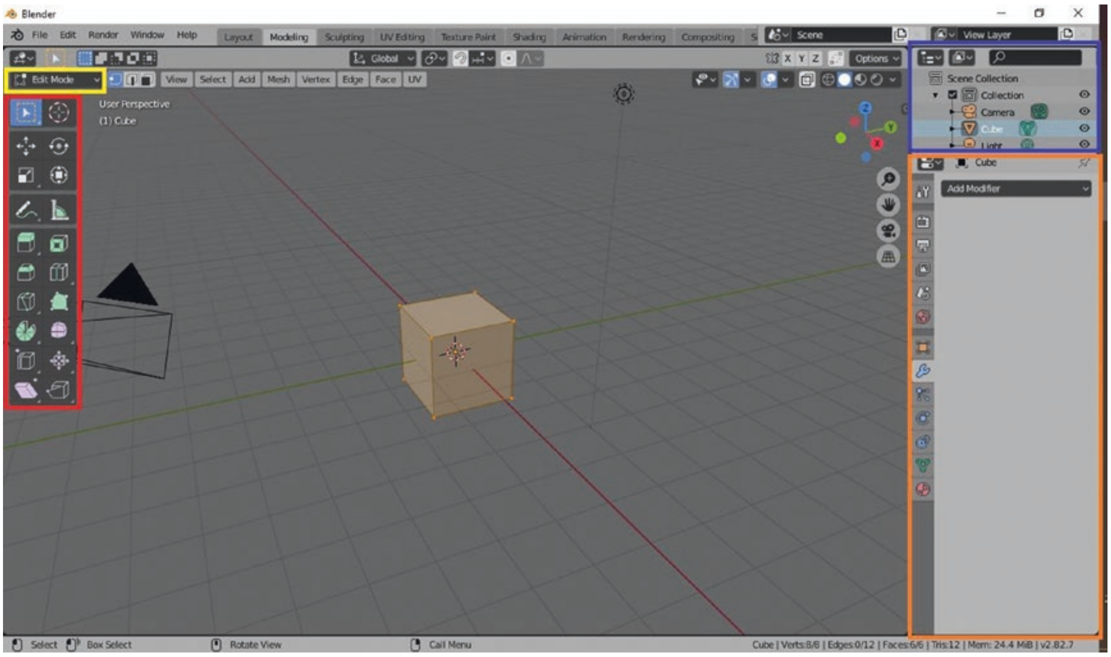
Blender
File Edit Render Window Help
Layout Modeling Sculpting UV Editing Texture Paint Shading Animation Rendering Compositing
Scene
View Layer
Edit Mode
View Select Add Mesh Vertex Edge Face UV
Global
X Y Z
Options
User Perspective
(1) Cube
Scene Collection
Collection
Camera
Cube
Light
Cube
Add Modifier
Select Box Select Rotate View Call Menu
Cube | Verts:8/8 | Edges:0/12 | Faces:6/6 | Tris:12 | Mem: 24.4 MiB | v2.82.7

Figure 2-4. *Modeling Workspace*