



THIRD EDITION

LEARNING Blender

A Hands-On Guide to Creating 3D Animated Characters



OLIVER VILLAR

Praise for *Learning Blender*

“Oliver Villar’s book will give you a solid foundation in Blender and computer graphics in general. Filled with well-crafted examples and lessons, this book will give you the tools you need to succeed as an artist.”

—David Andrade, Producer, Theory Studios

“The days are now over when beginners found learning Blender 3D difficult. Oliver Villar introduces to beginners the best of Blender’s 3D features and 3D fundamentals in fun and exciting ways. His approach of completing a character from scratch, touching every aspect of 3D from Blender’s point of view, is truly filled with explanations of techniques and important tools that will help readers to bring their ideas to life creatively while following professional workflows in 3D.

Starting with the fundamentals of 3D, this is a great resource for every beginner artist who is looking to learn Blender 3D. It’s truly a book written with great dedication!”

—Waqas Abdul Majeed, CG Generalist, www.waqasmajeed.com

“I found Oliver Villar’s book *Learning Blender* to be an essential tool for not only getting users acquainted with Blender, but also preparing them by explaining the history and the magic that has made Blender what it is now. His book also prepares users to be productive and informed by explaining the community and its various portals. His book is complete in explaining all the aspects of the UI and acquainting users with the classic G, S, and R. The exercises are perfect for getting users on the level to begin making their own worlds. I was even pleased to see him discussing F2, ripping with V, and even Knife Project, which are classics I usually consider to be more advanced. This book is a no-holds-barred approach to getting the most out of this capable little program. I must also add that the character created is attractive and well created, and is a fine example of using the program for character modeling. Oliver is truly a skilled artist and that shines through in his use of this program.”

—Jerry Perkins, 3D Conceptor, Fenix Fire

This page intentionally left blank

Learning Blender

Third Edition

This page intentionally left blank

Learning Blender

A Hands-On Guide to Creating 3D
Animated Characters

Third Edition

Oliver Villar

 Addison-Wesley

Boston • Columbus • New York • San Francisco • Amsterdam • Cape Town
Dubai • London • Madrid • Milan • Munich • Paris • Montreal • Toronto • Delhi • Mexico City
São Paulo • Sydney • Hong Kong • Seoul • Singapore • Taipei • Tokyo

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Visit us on the Web: informat.com/aw

Library of Congress Control Number: 2020952176

Copyright © 2021 Pearson Education, Inc.

Cover illustration by Oliver Villar

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, request forms and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearson.com/permissions/.

The Blender® brand name and logo are a copyrighted property of NaN Holding B.V., and has been licensed in 2002 to the Blender Foundation.

Maya® is a registered trademark or trademark of Autodesk, Inc., in the USA and other countries. This book is independent of Autodesk, Inc., and is not authorized by, endorsed by, sponsored by, affiliated with, or otherwise approved by Autodesk, Inc.

3ds Max® is a registered trademark or trademark of Autodesk, Inc., in the USA and other countries. This book is independent of Autodesk, Inc., and is not authorized by, endorsed by, sponsored by, affiliated with, or otherwise approved by Autodesk, Inc.

Photoshop® is a registered trademark of Adobe Systems incorporated in the USA and/or other countries.

Krita® is a trademark owned by Stichting Krita Foundation.

ISBN-13: 978-0-13-641175-8

ISBN-10: 0-13-641175-4

ScoutAutomatedPrintCode

Vice President and Publisher

Mark L. Taub

Editor-in-Chief

Brett Bartow

Sponsoring Editor

Malobika Chakraborty

Development Editor

Sheri Replin

Managing Producer

Sandra Schroeder

Sr. Content Producer

Julie B. Nahil

Project Editor

Rachel Paul

Copy Editor

Keir Simpson

Indexer

Jack Lewis

Proofreader

Rachel Paul

Cover Designer

Chuti Prasertsith

Compositor

The CIP Group



To Grandma. I'll keep working to make you proud.



This page intentionally left blank

Contents at a Glance

Preface	xxv
Acknowledgments	xxxii
About the Author	xxxiii

Part I The Basics of Blender **1**

1	What You Need to Know About Blender	3
2	Blender Basics: The User Interface	13
3	Your First Scene in Blender	41

Part II Beginning a Project **67**

4	Project Overview	69
5	Character Design	75

Part III Modeling in Blender **91**

6	Blender Modeling Tools	93
7	Character Modeling	127

Part IV Unwrapping, Painting, and Shading **181**

8	Unwrapping and UVs in Blender	183
9	Painting Textures	205
10	Materials and Shaders	225

Part V Bringing Your Character to Life **253**

11	Character Rigging	255
12	Animating Your Character	311

Part VI Getting the Final Result **327**

13	Camera Tracking in Blender	329
14	Lighting, Compositing, and Rendering	345

Part VII Keep Learning **369**

15	Other Blender Features	371
-----------	------------------------	------------

Index	379
-------	------------

This page intentionally left blank

Contents

Preface	xxv
Acknowledgments	xxxii
About the Author	xxxiii

Part I The Basics of Blender **1**

1	What You Need to Know About Blender	3
	What Is Blender?	3
	Commercial Software Versus Open-Source Software	4
	Commercial Software	4
	Open-Source Software	5
	But Can I Sell the Works I Create with Blender?	5
	History of Blender	6
	Blender Foundation and Blender Development	8
	Who Pays for Blender's Development?	10
	The Blender Community	10
	Summary	11
	Exercises	12
2	Blender Basics: The User Interface	13
	Downloading and Installing Blender	13
	Using Blender with Recommended Hardware	13
	Using Blender's User Interface	15
	Splash Screen	15
	Top Bar and Status Bar	16
	Default Editors	16
	Understanding Areas and Editors	16
	Resizing Areas	16
	Splitting and Joining Areas	17
	Swapping and Duplicating Areas	17
	Understanding the Types of Editors	18
	Using Workspaces	21
	Getting to Know Blender's Interface Elements	23
	Getting to Know Menus and Popovers	23

Getting to Know Panels	24
Getting to Know Pie Menus	24
Understanding the 3D Viewport	26
Understanding Regions	27
Understanding the 3D Viewport's Header	29
Navigating the 3D Scene	31
Navigating the 3D Scene by Using the Mouse, Keyboard, and NumPad	31
Navigating from the View Menu	33
Navigating with the 3D Viewport's Navigation Gizmos	33
Selecting Objects	33
Selecting All and Deselecting All	35
Using Active Tools to Perform Selections	35
Understanding the 3D Cursor	35
Placing the 3D Cursor	37
Understanding Blender's User Preferences	37
Saving User Preferences	39
Resetting User Preferences	39
Creating Your Own Startup File	39
Summary	40
Exercises	40
3 Your First Scene in Blender	41
Creating Objects	41
Moving, Rotating, and Scaling	42
Using Active Tools	42
Using Manipulators	44
Using Keyboard Shortcuts (Advanced)	46
Using Menus	47
Arranging Objects in Your Scene	48
Naming Objects and Using Datablocks	49
Renaming Objects	49
Managing Datablocks	49
Naming Your Scene's Objects	51
Using Interaction Modes	51
Applying Flat or Smooth Surfaces	53

Working with Modifiers	54
Adding Modifiers	54
Adding a Subdivision Surface Modifier to Your Object	55
Using Workbench, EEVEE, and Cycles	57
Understanding Viewport Shading	58
Switching Viewport Shading Modes	60
Managing Materials	60
Adding and Adjusting Materials	60
Turning On the Lights	62
Light Options	62
Adding Lights to Your Scene	62
Moving the Camera in Your Scene	62
Rendering	63
Saving and Loading Your .blend File	64
Launching and Saving the Render	65
Summary	66
Exercises	66

Part II Beginning a Project **67**

4 Project Overview	69
Three Stages of a Project	69
Preproduction	69
Production	70
Postproduction	70
Defining the Stages	71
A Film Without Visual Effects	71
A Visual-Effects Film	72
An Animated Film	72
A Photograph	73
Making a Character-Creation Plan	73
Character Preproduction	73
Character Production	73
Project Postproduction	74
Summary	74
Exercises	74

5 Character Design	75
Character Description	75
Personality	76
Context	76
Style	77
Appearance	77
Designing the Character	78
Silhouettes	78
Base Design	79
Head	81
Details	82
Refined Design	83
Adding Color	84
Finalizing the Design	85
Making Character Reference Images	86
Using Other Design Methods	88
Summary	89
Exercises	89

Part III Modeling in Blender **91**

6 Blender Modeling Tools	93
Working with Vertices, Edges, and Faces	93
Selecting Vertices, Edges, and Faces	94
Accessing Modeling Tools	94
Making Selections	95
Shortest Path	95
Proportional Editing	96
Linked Selection	97
Loops and Rings	97
Border Selection	98
Grow and Shrink Selection	98
Select Similar	99
Linked Flat Faces	99
Select Boundary Loop and Loop Inner-Region	99
Checker Deselect	100
Other Selection Methods	100

Using Mesh Modeling Tools	100
Bevel	100
Bisect	102
Boolean Operations	102
Bridge Edge Loops	104
Connect	104
Delete and Dissolve	105
Duplicate	106
Extrude	106
Fill and Grid Fill	108
Inset	108
Join	109
Knife	110
Knife Project	111
Loop Cut and Slide	112
Make Edge/Face	113
Merge	113
Offset Edge Loop	114
Poke	115
Rip and Rip Fill	115
Separate	116
Shrink/Fatten	116
Slide	117
Smooth Vertex	118
Solidify	118
Spin	118
Split	119
Subdivide	119
Using Modeling Add-Ons	120
Working with LoopTools	120
Working with F2	122
Using Other Useful Blender Options and Tools	123
Auto Merge	123
Global and Local View	124
Hide and Reveal	124
Snapping	124
X-Ray	125

Summary	125
Exercises	125
7 Character Modeling 127	
What Is Mesh Topology?	127
Choosing Modeling Methods	129
Box Modeling	129
Poly to Poly	129
Sculpt and Retopology	129
Modifiers	130
The Best Method	130
Setting up the Reference Images	131
Modeling the Eyes	135
Creating an Eyeball	135
Using Lattices to Deform the Eyeballs	137
Mirroring and Adjusting the Eyes	138
Modeling the Face	139
Studying the Face's Topology	139
Blocking the Face's Basic Shape	140
Defining the Face's Shapes	143
Modeling the Torso and Arms	150
Modeling the Basic Shapes for the Torso and Arms	152
Defining the Arms and Torso	154
Detailing the Backpack and Jacket	156
Finishing the Belt and Adding a Neck to the Jacket	158
Modeling the Legs	159
Modeling the Boots	161
Modeling the Hands	164
Building the Basic Hand Shape	164
Adding the Fingers and Wrist	166
Modeling the Cap	168
Creating the Base of the Cap	168
Adding Details to the Cap	170
Modeling the Hair	172
Shaping Locks of Hair	172
Adding Natural Details to the Hair	174

Modeling the Final Details	176
Eyebrows	176
Communicator	177
Badges	177
Teeth and Tongue	178
Other Clothing Details	179
Summary	180
Exercises	180

Part IV Unwrapping, Painting, and Shading **181**

8 Unwrapping and UVs in Blender	183
Seeing How Unwrapping and UVs Work	183
Unwrapping in Blender	184
Using the UV Editor	185
Navigating the UV Editor	187
Accessing the Unwrapping Menus	188
Working with UV Mapping Tools	188
Defining Seams	190
Considering Before Unwrapping	191
Working with UVs in Blender	193
Marking the Seams	193
Creating and Displaying a UV Test Grid	194
Unwrapping Jim's Face	196
Using Live Unwrap	197
Adjusting UVs	198
Separating and Connecting UVs	199
Reviewing the Finished Face's UVs	200
Unwrapping the Rest of the Character	200
Packing UVs	202
Summary	203
Exercises	204
9 Painting Textures	205
Defining the Main Workflow	205
Texture Painting in Blender	206
Texture Paint Workspace	206

Texture Paint Interaction Mode	207
Before You Start Painting	209
Conditions for Painting	210
Texture Slots	212
Limitations of Blender's Texture Paint Mode	213
Creating the Base Texture	214
Placing Texture Elements	214
Saving Your Image	215
Packing Your Images	215
Understanding the Elements of a Texture	215
Introduction to PBR Materials	215
Understanding Material Channels	216
Texturing in Other Software	216
Pros and Cons of Texturing in Blender and Other Software	217
Texturing in 2D Image-Editing Software	217
3D Texturing Software	221
Seeing the Painted Character in Blender	223
Summary	223
Exercises	224
10 Materials and Shaders	225
Understanding Materials	225
Applying Materials	225
How Materials Work	226
PBR Materials	226
Shaders and Mix Shaders	229
Masks and Layers	230
Channels	231
Procedural Textures	233
Differences and Compatibility Between EEVEE and Cycles	234
Nodes	235
Shading Your Character	236
Adding Several Materials to a Single Object	236
Understanding the Material Properties Tab	238
Using Shaders	240

Mixing and Adding Shaders	241
Loading Textures	242
Shading Jim	243
Shading the Eyeballs in EEVEE	244
Shading the Eyeballs in Cycles	245
Running Render Tests	246
Adding Lights and Environment	246
Rendering in EEVEE	248
Rendering in Cycles	250
Summary	252
Exercises	252

Part V Bringing Your Character to Life 253

11 Character Rigging 255

Understanding the Rigging Process	255
What's a Rig?	255
Rigging Process	256
Working with Armatures	257
Manipulating Bones	258
Working in Object, Edit, and Pose Modes	261
Bone Hierarchies	261
Adding Constraints	262
Forward and Inverse Kinematics	264
Practice with Bones and IK Constraints	264
Rigging Your Character	267
A Few Tips Before You Start Rigging	267
Using Rigify to Generate Jim's Rig	268
Organizing Bones	273
Bone Groups	273
Armature Layers	274
Understanding the Rigify Rig	275
Performing Adjustments to the Rigify Rig	276
Skinning	278
Understanding Vertex Weights	278
Vertex Groups	278

Setting Up the Model for Skinning	280
Enabling Deforming Bones Only	281
Knowing What Objects Don't Need Weights	281
Adding an Armature Modifier	283
Defining Weights	284
Creating the Facial Rig	288
Rigging the Eyes	288
Mirroring the Eye Rig	290
Naming Bones Automatically	290
Mirroring Bones	291
Possible Side Effects of Mirroring Bones	291
Rigging the Jaw	291
Skinning the Eyes and the Jaw	292
Chest Badge Deformation	294
Modeling Shape Keys	295
Mirroring Shapes	298
Creating the Face Controls	298
Using Drivers to Control the Face Shapes	299
Organizing the Facial Rig	303
Creating Custom Shapes	305
Making Final Retouches	306
Reusing Your Character in Different Scenes	307
Library Linking	307
Linking	308
Appending	308
Working with Collections	308
Protecting Layers	309
Using Proxies to Animate a Linked Character	310
Summary	310
Exercises	310
12 Animating Your Character	311
Using the Character's Rig	311
Posing the Character	312
Inserting Keyframes	312
Adding Keyframes Manually	312
Adding Keyframes Automatically	313

Adding Keyframes Using Keying Sets	314
Adding Keyframes to Properties in Menus	315
Working with Animation Editors	315
Timeline	316
Dope Sheet	317
Graph Editor	317
Non-Linear Animation	319
Common Controls and Tips	319
Animating a Walk Cycle	321
Creating an Action	321
Creating the Poses for the Walk Cycle	322
Repeating the Animation	324
Walking Along a Path	325
Summary	326
Exercises	326
Part VI Getting the Final Result	327
13 Camera Tracking in Blender	329
Understanding Camera Tracking	329
Shooting Video for Easy Tracking	330
Using the Movie Clip Editor	332
Tracking the Camera Motion	333
Loading Your Footage	333
Studying the Anatomy of a Marker	335
Tracking Features in the Footage	336
Configuring Camera Settings	339
Solving Camera Motion	339
Applying Tracked Motion to the Camera	340
Adjusting Camera Motion	341
Testing Camera Tracking	343
Summary	343
Exercises	343
14 Lighting, Compositing, and Rendering	345
Lighting Your Scene	345
Analyzing the Real Footage	345

Creating and Testing Lights	346
Showing/Hiding Objects in Render	347
Testing EEVEE and Cycles	348
Using the Node Editor	349
Compositing	349
Understanding Nodes	349
Studying the Anatomy of a Node	351
Using the Node Editor	352
Getting Started with Nodes	352
Previewing the Result	355
Rendering and Compositing Your Scene in Cycles	356
Creating a Shadow Catcher	357
Rendering in Cycles	357
Node Compositing in Cycles	359
Rendering and Compositing Your Scene with EEVEE	361
Creating a Shadow Catcher in EEVEE	361
Rendering in EEVEE	364
Compositing in EEVEE	364
Exporting the Final Render	365
Setting the Animation Output	365
Launching the Final Render	366
Summary	367
Exercises	367
Part VII Keep Learning	369
15 Other Blender Features	371
Simulations	371
Particles	371
Hair Simulation	371
Cloth Simulation	372
Rigid and Soft Bodies	372
Fluids Simulation	372
2D Animation	373
Grease Pencil	373

Cartoon Shaders with EEVEE	373
Freestyle	373
VFX: Masking, Object Tracking, and Video Stabilization	373
Video Editing	374
Sculpting	374
Retopology	375
Maps Baking	375
Add-Ons	375
Included Add-Ons	376
More Add-Ons	376
Python Scripting	376
Summary	376
Index	379

This page intentionally left blank

Preface

Character creation is a big undertaking. It involves several very different skills, and that's what you're going to learn soon enough. In this preface, I quickly show you what this book is about and what you can expect from it. If you already have some experience with other 3D software, you've come to the right place, as you'll find some instructions on how to handle switching between different programs, which can be frustrating—and sometimes even more difficult than learning a program for the first time.

Welcome to Learning Blender!

Welcome to the third edition of *Learning Blender: A Hands-On Guide to Creating 3D Animated Characters*. In this book, you'll learn how to use Blender in a complete and complex project. You'll see every part of the process so that you can understand what is involved in the creation of a 3D character and decide which part you like the most afterward. In other words, this book is not a specialized book that will make you a modeling genius or an expert animator; instead, it helps you understand the basic concepts behind every part of the process. The idea is that when you finish reading this book, you'll have the knowledge you need to start any other project, from preproduction to the final result.

If you're a freelancer (or want to be), this book is tailored to you, as freelancers often get small but very different and varied jobs, and having basic or medium skills in different tasks can be more useful than being very good at a single specific thing.

If you want to work for a big company and prefer to specialize, it helps to understand the full process. If you're a modeler, for example, but you also understand how rigging works, when you create your models, you'll be able to recognize the possible issues that your rigger mates will encounter, which will make their work easier. When you work on a team, you'll work on only part of the project, but if you have at least a little understanding of what the rest of the team's job is, your work will be more valuable to them, and everyone will be happier!

Maybe you're already familiar with Blender and want to learn about 3D character creation. Very good. You can skip the first two or three chapters and go straight to the main part of the book—but do this only if you're sure that you understand the basics of Blender.

Finally, if you just want to get started in this amazing world of 3D and dive into the sea of vertices, this book will give you a good insight into how 3D projects are handled. If you have never used 3D software before, don't worry if it looks a bit

overwhelming in the beginning. That's normal. The software has lots of options and crazy stuff that will be unknown to you, and we all tend to be afraid of what we don't know. If you keep going, however, when you start using and understanding Blender, you'll start enjoying the learning process, and your results will get better with time and practice. Good luck!

Do You Come from Other 3D Software?

I took this path myself years ago, so I understand what you will go through. That's why throughout the chapters, I share tips, keeping in mind the differences between Blender and other 3D software. I came to Blender after using commercial software such as 3ds Max, Maya, and XSI for years. Back then (version 2.47), Blender was less user-friendly, but it's been greatly improved since then. It's still a little alien compared with other software, though, and it may feel intimidating to you at first. Don't worry; that reaction is completely understandable. Just don't give up!

Learning Blender may not be easy at first. It took me three or four times checking different versions of Blender until I finally decided to start learning it for good. You'll see weird features, such as the omnipresent 3D cursor, which you always see in the scene but apparently has no function. (I've heard someone say that it looks like a sniper's visor for shooting at your models.)

Also, you'll be "forced" to learn a lot of shortcuts. This requirement makes the learning curve for Blender difficult in the beginning, but when you get used to Blender, you'll love it, as shortcuts help you work a lot faster in the long run!

Before I used Blender, it was difficult for me to work with fewer than three 3D views on the screen at the same time, for example. Now I work in full-screen mode with only one view in a much more comfortable way; it's like using the expert mode in other software all the time! I even feel weird sometimes when I need two 3D views for some special reason.

I've taught a lot of people and talked with many others who came to Blender from other software, and usually, they kind of hate it at first. (That's why most people give up and stick with commercial software.) After a short time using it, though, they start loving Blender and get addicted to it. They find that a lot of tasks are easier or faster to do in Blender than in other software. It's so common to love Blender after that first rejection stage, however, that there's a name for this feeling in the community: Blenderitis.

Blender has its limitations, of course, but for the general needs of most users, it's more than enough.

I really encourage you to keep exploring Blender and find out what it has to offer. I've learned to use a lot of different software and tools, and after repeating the learning process and switching software several times, I've found that Blender works best for me.

I'll share the method I've used with you. Maybe it'll help you too. The key to making a successful change (not only in software, but also in life, work, or whatever you want) is to *learn how to adapt and be flexible*. You have to free your mind to some extent

to leave space for the new situation, software, or anything else to get in. In these situations, a lot of people can only complain (“This software doesn’t have that tool,” “That was easier on the old one,” and so on). Avoid this behavior at all cost, and *try to understand the new software*, as each program has different philosophies behind its development and workflow. Complaining is a waste of energy and time that you could be spending on something much more useful, such as learning how to use the new software.

What is the best way to adapt? Force yourself!

Set a deadline (that way, you’ll have a good or bad result, but at least you’ll finish something), and decide what you’re going to do. Think of an easy project, and go for it. Having a deadline keeps you from drifting around for days, going crazy over small details that make the process too long.

Usually, people start playing around with no purpose. They don’t get a specific result, but something random. This result doesn’t motivate them and gives them the impression that they can’t use the software.

Instead, if you propose a little project, you’ll have a goal to work toward, which allows you to find the tools you need to achieve that goal. When you finish, even if the project is not perfect, you’ll have learned some tools and achieved a result, which will motivate you to do better next time or to start a different project so you can learn about other tools.

Keep in mind that you probably don’t want to start with a very big or difficult project. The key is to start learning little by little, taking small steps to keep yourself motivated. If you start with something big that involves a lot of steps, you may get stuck at some point, which will frustrate you. When you work on something small, even if it goes wrong, you won’t have spent too much time after all, so getting attached to the project won’t be a real issue.

Over time, after you make a few small projects, you’ll have a knowledge base, and you’ll understand how the new software works. At that point, you can judge whether you’re interested in learning more or whether you’re more comfortable with the previous software.

A lot of software is out there, and each program is different, so depending on your work, style, taste, and personality, you may prefer one or another. What is intuitive and comfortable for some people isn’t for others. Nonetheless, if you give the new software a good test drive, even if some things that you’re used to are missing, you’ll learn about others that are really cool that you didn’t see before!

In my case, I was very comfortable with 3ds Max, but after using Blender extensively for a few days (yes, only a few days; they were very intense days, though!), I honestly couldn’t go back. I missed some tools, of course, but I found that the advantages clearly surpassed the disadvantages for me, so I’ve used Blender ever since.

I hope that this book motivates you to try Blender and give it a chance instead of deciding that you don’t like it because you can’t master it in five minutes. (I’ll bet you didn’t understand any other software in five minutes the first time you used it!)

The essence of practicing to learn is to set a feasible goal, set a deadline (due date), and try your best to reach that goal. No excuses; no complaints! Discipline and not giving up are the keys.

My method is just a guideline. It may not be useful for you, or you may find a better approach. But if you don't know where to start and feel discouraged, just try it!

How to Use This Book

This book is divided into parts to help you to keep track of your progress:

- **Part I, “The Basics of Blender” (Chapters 1, 2, and 3):** Understanding Blender and learning the basics
- **Part II, “Beginning a Project” (Chapters 4 and 5):** Preproduction, project preparation, and character design
- **Part III, “Modeling in Blender” (Chapters 6 and 7):** Starting production, focusing on character modeling
- **Part IV, “Unwrapping, Painting, and Shading” (Chapters 8, 9, and 10):** Unwrapping, texturing, and applying materials
- **Part V, “Bringing Your Character to Life” (Chapters 11 and 12):** Rigging and animation
- **Part VI, “Getting the Final Result” (Chapters 13 and 14):** Postproduction, camera tracking, rendering, and compositing
- **Part VII, “Keep Learning” (Chapter 15):** Other Blender features

You can start with the part you’re most interested in, of course, but if you’re new to Blender, I recommended that you start from the beginning so that you understand the software before you jump into something as complex as the creation of a 3D character.

In each chapter, if some basic knowledge is required, I explain it before you dive into the real thing. You’ll also find tips and useful shortcuts along the way to help you work faster and more efficiently.

If you’re already familiar with Blender, you can skip the first three chapters and start reading about character creation.

Chapter 1, “What You Need to Know About Blender,” talks about Blender, open-source software, how the development process works, its history, and what Blender is all about. You don’t really need to know these things to use Blender, but it’s interesting and gives you an overview of some of the strong points of Blender.

Chapter 2, “Blender Basics: The User Interface,” takes you through the user interface, basic navigation, selections, and Blender’s innovative nonoverlapping window system for dividing and merging the interface as you see fit.

In Chapter 3, “Your First Scene in Blender,” you learn how to create your first scene with Blender. This very basic scene lets you play with the main tools, as well as work

with simple modeling, materials, and lighting, and it helps you understand the differences between rendering with Blender Render and rendering with Cycles.

After this introduction, you start on the main project: creating a 3D character. The reason you create a character as a project for this book is that it involves almost every part of the software: modeling, texturing, rigging, animation, and so on.

This part of the book explains everything you'll go through, talking about preproduction and how to get ready for any project. You'll learn that preparation is essential!

In the final chapters, you see how to track the camera of a real video and composite your character into that scene so that you end up with something cool you can show your friends, not just a character inside Blender.

I discuss some other features of Blender in Chapter 15, “Other Blender Features,” so that you get a glance at them, including dynamic simulations, particles, smoke and fire, the Grease Pencil, and add-ons.

I encourage you to create your own stuff from scratch and use your own video to track the camera, but if you prefer to follow the book in detail (with the same material used in it) or want to skip some parts, you'll find all the material you need to start from any point of the book in the production files (at www.blendtuts.com/learning-blender-files), which include

- .blend files with different levels of progress so you can start with whatever part of the book interests you. You don't have to start from scratch.
- Texture images for the character.
- Real video for camera tracking.
- Final results.
- Video tutorials of some parts of the book.

What's New in This Edition

What you have in your hands is the third edition of *Learning Blender*. The whole book has been updated to be compliant with Blender version 2.83 and beyond. Blender 2.83 is the first LTS (long-term support) version of Blender, which means that it will be usable and errors will be fixed for at least two years after its release, making it a good version to start with. (General Blender versions have very short life spans, with new releases every three to four months.) Most figures have been updated or reworked to improve readability and to reflect the changes in the new Blender versions. The whole character-creation process has been redone to make sure that the instructions are compatible with Blender 2.83. New tools are discussed throughout the book, especially (but not limited to) selection and modeling tools. In Version 2.80, a new real-time render engine was added to Blender (EEVEE), and the book shows you how to use it as well. A lot of new tips and tricks have been added, and some chapters have been extended or changed in approach, based on feedback I got from previous editions. That said,

I hope that you find these new additions interesting and that they improve your experience with the book and with Blender.

Without any more hesitation, get ready to start learning. You have a long way to go!

Register your copy of *Learning Blender, Third Edition*, on the InformIT site for convenient access to updates and/or corrections as they become available. To start the registration process, go to informit.com/register and log in or create an account. Enter the product ISBN (9780136411758) and click Submit. Look on the Registered Products tab for an Access Bonus Content link next to this product, and follow that link to access any available bonus materials. If you would like to be notified of exclusive offers on new editions and updates, please check the box to receive email from us.

Acknowledgments

Although the author of a book usually takes most of the credit, a lot of people are needed to make it, improve it, and get it to your hands. Thanks to Laura Lewin and Olivia Basegio for giving me the opportunity to participate in this project since the first edition in 2015; they helped me with everything I needed during the entire process. The same goes for Malobika Chakraborty, who supervised this third edition, and who was very patient and understanding (because this edition was a very challenging one, but more on that later). Thanks to Michael Thurston, Daniel Kreuter, Mike Pan, and Tim Harrington, who did an amazing job helping me write the first edition of this book. Thanks to Andrea Coppola, David Andrade, and Aditia A. Pratama, who worked on the revisions for the second edition of *Learning Blender*. Thanks to Abraham Castilla and Aidy Burrows, who helped make sure that everything was up to date with the latest Blender versions after the huge changes introduced in Blender 2.80. Thanks to Sheri Replin, who made sure that everything in this edition was in place. Thanks also to Rachel Paul, Julie Nahil, and Keir Simpson. Thanks to everyone else who was involved in any way in the creation of this book.

One reason why this edition was so challenging (and why it was so long in the making) is that Blender took a quantum leap between versions 2.79 and 2.80, requiring a huge amount of changes in the book (to the point that some chapters were almost completely rewritten), and the versions 2.81 to 2.83 introduced subtle but rather important changes that polished all that was introduced in 2.80. All those changes made me go back and forth, making sure that everything was as current as possible for Blender 2.83. The fact that it's difficult to keep up with development speaks volumes about the great work that the team behind Blender is doing and how quickly the software is getting improvements and new features, so I'd also like to appreciate the incredible work of the Blender Foundation, Ton Roosendaal, all the Blender developers, Pablo Vázquez for his effort in communicating all the changes introduced in the software to the public, and the amazing Blender community. Thanks, everyone.

Last but not least, this book wouldn't exist without the invaluable help and support of my girlfriend, who never stopped encouraging me. Thank you.

Special thanks to César Domínguez Castro, who filmed the footage used in the camera tracking and compositing chapters that you'll find in the bonus files (www.blendtuts.com/learning-blender-files), which will allow you to experiment with Blender tools.

This page intentionally left blank

About the Author

Oliver Villar, born in Galicia (Spain) in 1987, has been drawing since he was a kid. His interest in art brought him to 3D, which he's been studying since 2004. He used different commercial 3D software before stumbling onto Blender in 2008. Since then, he has used Blender professionally as a freelance 3D designer and tutor.

In 2010, he founded blendtuts.com, a website that offers quality Blender training videos to the community. After a few years, he decided to dedicate more effort to the Spanish community, in which Blender learning material is lacking, and started blendtuts.es.

For years, he's been one of the organizers of the main Spanish Blender event: Blendiberia.

Currently, he teaches Blender to his own students, for online schools, and for the University of Murcia.

This page intentionally left blank

The Basics of Blender

- 1** What You Need to Know About Blender
- 2** Blender Basics: The User Interface
- 3** Your First Scene in Blender

This page intentionally left blank

What You Need to Know About Blender

Blender has quite a remarkable story, as open-source software (OSS) works in a very different way from typical commercial software. It is helpful to know this if you intend to use Blender professionally, as it may give you insight into how powerful its concept is. In this chapter, you'll learn about how Blender was created, how the development process works, how it is funded, and what type of community surrounds the Blender world.

What Is Blender?

Blender is OSS that provides one of the most complete 3D-graphics creation suites. It includes tools for modeling, texturing, shading, rigging, animation, compositing, rendering, video editing, 2D animation, and more. Since the development of version 2.50, which marked a big milestone, Blender's user base has grown significantly, and more professionals have started using it. It has reached animation studios and has been used for some purposes in top movie productions such as *Life of Pi*, *Spider-Man 2*, and *Red Riding Hood*; it was used to animate and compose a creature in *Warcraft*; and in 2018, it was used to create an entire animation film for Netflix (*Next Gen*, by Tangent Animation). More recently, Blender version 2.80 has brought a lot of professional attention to the software because of things like the new 2D animation tools (Grease Pencil), the real-time render engine (EEVEE), and a complete redesign of the user interface.

Its principal target audience is professional freelance 3D artists and small studios, and Blender works very well for their needs. It still isn't widely used by big studios for several reasons: Large studios typically have long-established software, and the commercial software they use often has impressive third-party plug-ins that have been developed over years for specific uses in production. Blender is still growing and lacks a lot of third-party support, but despite being relatively new to the professional landscape, it's quickly becoming a viable option for freelancers and studios alike.

Blender is known for being very different from other software, and that's why some people are hesitant to use it (although, as mentioned before, things have been vastly improved since version 2.80, which introduced a lot of changes to make it more

user-friendly). It doesn't follow a lot of the same standards that other 3D software has been using for decades, and this is sometimes an issue for new users. That's also the charm of Blender; once you experience it, it is very possible that you'll love it because it is so different!

Blender, because it is open-source, doesn't need to sell licenses, so it can bypass the way other software uses "standards" and instead go for something new and unique. In the words of Ton Roosendaal (Blender Foundation's chairman and the creator of Blender), "I would never look up to average; I want to lift up the average. It's not following conventions; it's following a vision."

Blender's development is funded primarily through voluntary donations from users. This should give you an idea of how useful a lot of people find it: They donate to its continued development even when they can use it for free. This can be difficult to understand for people who use only commercial software, but it's something you often find with OSS: People are more willing to contribute voluntarily and help the cause because it's free.

Popular OSS such as Blender can have lots of contributors and grow quite fast. This is very good for users because they get new features and tools periodically. It has a downside, though: It's difficult to stay abreast of everything new and be aware of updates to the latest versions. Also, instructional material can have a short lifespan. Although this material can be used for years because general features and workflow remain generally the same, some options, buttons, icons, and tools will be replaced, improved, changed, or removed.

Commercial Software Versus Open-Source Software

You can't understand OSS from the point of view of the "usual" copyright and privacy system, in which you can't use something if you didn't pay for it. The business model is completely different as well.

Commercial Software

Usually, the business model for companies that develop commercial software is to sell the software license itself. If you want to use commercial software, you have to pay for a license, but you don't really own the software. Some software companies may not allow you to use the software for particular purposes (such as getting into its code to study or change it), and you may be able to use it for only a fixed amount of time before having to pay for an upgrade or a new license. Lately, in fact, it's become a new commercial practice that you pay for a subscription every certain amount of time, and you can use the software only as long as you keep paying—essentially renting the software. In certain cases, you can use the software for free, but only for learning purposes; you need to purchase a license if you want to use it professionally to generate income. In other cases, what you get for free is a limited version of the software, and you need

to purchase the full version to access all its features. Here's where piracy comes in: Some people can't afford the software, and others just don't want to pay for it, so they use illegal copies, which results in a negative economic effect on the commercial software developers.

You can't develop new features in commercial software if you're not employed by the company that owns the software, and even if you are, you have to follow that company's guidelines (and you're not allowed to copy your code or show it to the general public). Anyone may develop plug-ins, but you are not allowed to change the software core or its basic features.

Open-Source Software

Open-source software is usually misunderstood as being free-of-charge software. The word *free* has a double meaning here, however: Not only is the software free to use, but also, its source code is freely available to everyone. Some software can be free of charge (freeware) but not free in terms of liberty of use—that is, you can't access the core (the source code) and modify it to fit your needs.

What *open-source* means is that the user has the power to access the source code of the software and modify it at will. Developers also encourage you to check the code, use the software for commercial purposes, and even redistribute it. In this sense, OSS is the exact opposite of commercial software. You can download this software and immediately use it commercially. The business model of a company that creates OSS is not to sell the software itself, but to sell related services such as instructional material, training, technical support, and merchandising. This type of company often relies on donations from the public as well.

The good thing about open-source is that anyone in the world can download the source code and develop a feature they like, and other people can use that new feature later. You are free to modify the source code, copy it as many times as you want, learn from it, and give it to your friends or classmates. Sometimes, OSS is developed by an individual or a small team. Some OSS is, of course, quite complex and highly organized, and it may even have external companies using it and contributing to its development.

Another fact worth noting is that there are several types of open-source licenses, such as General Public License (GPL), Eclipse Public License (EPL), and Massachusetts Institute of Technology (MIT) license. Before using OSS, you should get some information about the specific terms of those licenses to make sure you understand what you are allowed to do with that software.

Blender is released under the GNU General Public License (GPL, or free software). You can find more information at <https://www.blender.org/about/license>.

But Can I Sell the Works I Create with Blender?

This question is a common one among people who are new to OSS, and the answer is a crystal-clear yes. You can use the software to create work that you charge for

professionally. If you create a design for a client, for example, the work belongs to you, and as such, you can do what you want with it.

History of Blender

Lots of people think that Blender is new, but that's not accurate. Blender was born in the early 1990s, making it close to 30 years old. Blender Foundation Chairman Ton Roosendaal could find an “ancient” file—Blender’s first bit of code—that dated back to December 1992. It is true, however, that the software didn’t become popular until much later.

In 1988, Roosendaal founded a new Dutch animation studio, NeoGeo. Not long after, the new studio decided that it had to write new software to create its animations; as a result, it officially started to build Blender in 1995. In 1998, Roosendaal founded a new company called Not a Number (NaN) to further develop and market Blender. Due to difficult economic conditions at the time, NaN wasn’t a success, and investors stopped funding the company, shutting down Blender’s development in 2002.

Later in 2002, Roosendaal managed to build the not-for-profit Blender Foundation. Users community donated 100,000 euro (an amazing sum, raised in only seven weeks) to get the previous development investors to allow for the open-sourcing of the software and make it available for free. Finally, Blender was released under the terms of the GNU General Public License on October 13, 2002. Since that day, Roosendaal has led a team of enthusiastic developers who want to contribute to the project.

The first Open Movie project (*Elephant’s Dream*) was born in 2005, with the goal of gathering a team of artists who could use Blender in a real production and also give developers feedback that would ultimately improve the software significantly. The goal was not only to create the movie with open-source tools, but also to release the end result and production files to the public under a Creative Commons open license.

The project ended up being a great success, and Roosendaal created the Blender Institute, located in Amsterdam, the Netherlands, in the summer of 2007. The institute is now the core of Blender’s development, and many more Open Movies have been made there since, including *Big Buck Bunny* (2008), the video game *Yo Frankie!* (2008), *Sintel* (2010), *Tears of Steel* (2012), *Cosmos Laundromat* (2015), *Agent 327: Operation Barbershop* (2017), and *Spring* (2019).

The development of Blender version 2.50 began in 2008. It offered a major improvement of the software’s core, which was already becoming outdated. The final release of this version came in 2011. *Sintel* was made to put this new version to the test. It also helped improve the tools and brought back previous functionalities that were lost in the recent update. Since then, Blender has experimented with some other significant new features, such as Cycles, a new render engine that supports GPU real-time, path-tracing-based rendering, and more complex and realistic light and materials calculations.

Tears of Steel, for example, was made to implement and improve visual-effects tools such as camera tracking, compositing nodes improvements, and masks (to name a few), making Blender one of the most flexible tools in the 3D software panorama.

One of the latest Open Movies, released in 2015, was *Cosmos Laundromat* (see Figure 1.1), born to be the first full-feature film made by the Blender Institute. The crowdfunding campaign didn't reach the final goal, however, so the project turned into a short film that has been awarded and praised in many festivals, including the Jury's Award at Siggraph 2016. During its creation, hair simulation, Cycles rendering capabilities, video editing options, and many more features were heavily improved.



Figure 1.1 *Cosmos Laundromat* (2015) was a completely crowdfunded Open Movie project. The movie was made with these goals in mind: improving hair simulations, rendering capabilities, and other Blender features.

(CC) Blender Foundation | <https://gooseberry.blender.org>

The latest major release of Blender was in the works for three years. Version 2.80, released in November 2019, included a completely redesigned user interface, a new keymap, a real-time render engine called EEVEE, a 2D animation and drawing tool named Grease Pencil 2.0, and many other improvements. Especially because of how impressive and useful EEVEE looked, Version 2.80 caught many artists' attention during its development.

The most recent Open Movie to date is *Spring* (see Figure 1.2), which the Blender Animation Studio used to make sure that Blender 2.80 worked well enough in a production environment.

Since then, Blender 2.81, 2.82, and 2.83 have been released. These versions were not as ambitious as 2.80, but they were also very important: They not only included many new features and improvements, but also added a lot of polish to big features and changes introduced in Blender 2.80.

Although Blender 2.80 was an impressive milestone, Blender 2.83 was more polished and included so many important improvements that it became the first LTS (long-term support) Blender version. Typically, a new version of Blender is released

every three to four months, and previous versions don't receive bug fixes, even when those fixes are critical. This is not great for studios or professionals working on big projects that can last for months or even years, which is why Blender 2.83 LTS was born: to give those artists a version that they could use for a long time, knowing that it would be supported for at least two years after the release date, and that although new features would be present in future versions, important errors would be fixed.

The rest of this book is made with version 2.83 so that you can follow the instructions with a version that will be supported for as long as possible. If you use versions after 2.83, you may find some small changes or new features, but most of the time, there shouldn't be major changes that will make the process too difficult to follow.

If you're new to Blender, you may want to consider downloading version 2.83 to learn with this book (even though more recent versions may be available by the time you read the book) and then upgrade to the current version. When you have a general knowledge of how Blender works, it will be easier to make the transition to newer versions.



Figure 1.2 Still frame from the *Spring* Open Movie (2019), created to test Blender 2.80 (in production during its development). © Blender Foundation | <https://cloud.blender.org/films/spring>

Blender Foundation and Blender Development

The Blender Foundation is housed at the Blender Institute, located in Amsterdam, and it's an independent non-profit public benefit corporation that makes Blender possible. Open Movies are created by the Blender Animation Studio, also at the Blender Institute. The purpose of the Blender Foundation is to establish services for Blender users and developers, maintain and improve the Blender product through the GNU General Public License, and create ways to fund the foundation's goals and expenses (including Blender development and Open Movies creation).

The chairman of the Blender Foundation, Ton Roosendaal, organizes and sets the goals for the software and anything else related to Blender. Everyone can make proposals about features they'd like to see added to Blender, and after the main development team analyzes those proposals to see which of them are feasible, the team begins development. This system is very different from the development of commercial software, in which the company decides what needs to be done and developers have no say in what is added to the software.

In fact, Blender users don't even need to request new features; they can just develop a feature themselves and then send it to the foundation through the established channels. If the feature is found to be useful and interesting, and if it fits the Blender guidelines (it has to be consistent with the rest of the software), the main development team will work on adding it to the official Blender version.

The Blender Foundation hires a group of full-time developers to complete specific tasks and reach the most important goals, but most developers are volunteers who lend their time to learn and practice using the software, or just to participate in its ongoing development or add features they are interested in themselves. Some developers even raise their own funds to create and perfect the features they want to see in Blender.

Because it is OSS, Blender has public master versions and branches. The *master version* is the official version released at <https://www.blender.org>; it contains the stable features of Blender. *Branches* are development versions for testing new features or alternative features that may make it into the official master version at some point. (Commercial software also uses this method, but all the work happens behind the curtains. You can't create your own branch or test development versions unless the company that owns the software releases a beta version to generate feedback before the software's actual release.)

Chaos would result if everyone could just step in and add ideas to the software, of course, so one of the Blender Foundation's main tasks is organizing all the developers, defining priorities, and deciding what features should make it to the official versions. The foundation determines which features need branches and which branches should be removed. It also provides and maintains the platform for Blender and operates the bug-tracker system, in which users can report bugs that are then assigned to specific developers for correction (usually, really quickly).

Note

Interested in testing development versions of Blender? Visit graphicall.org and <https://developer.blender.org>, where you can find the version for your system. Testing is not recommended if you aren't an experienced user, as these versions are experimental and unstable, so you must use them at your own discretion. The Blender Foundation also offers automatic daily builds. Go to the Download tab of blender.org, scroll to the Bleeding Edge option at the bottom, and download the latest additions to Blender's master version.

Who Pays for Blender's Development?

Although Blender is free, its development costs money, and as mentioned, people work full-time to make it possible. But where does the money come from?

The Blender Foundation has established several income channels for being able to continue maintaining Blender development and the services surrounding it:

- **Blender Cloud:** People can pay a monthly subscription to get access to a library of training videos, 3D assets from Open Movies, making of documentaries, and exclusive services.
- **Development Fund:** Users who want to help Blender development can subscribe to donate monthly amounts.
- **One-time donations:** Anyone can donate any amount of money to help Blender, without any recurrence.
- **Private investment:** Studios and companies that use Blender sometimes pay for developers to support certain features they need, or they develop those features in-house and then give the code to the public as a contribution.
- **Crowdfunding:** For special occasions, the Blender Foundation organizes campaigns to crowdfund specific purposes. As an example, during 2018, Code Quest was launched to gather funds from Blender users to finance the main development of the very ambitious 2.80 version, with the goal of uniting the group of core Blender developers in Amsterdam.
- **Blender Store:** The store at blender.org sells merchandising and books. Profits from those sales help finance Blender activities.

Note

If you want more information about the Blender Foundation, Blender, development, or official documentation, you can find it on the official website: <https://www.blender.org>.

The Blender Community

For every software type, it's important to have a community surrounding it to provide feedback and engage other users. But a community is even more important for OSS. The community not only provides feedback, but also proposes new features, discusses development, creates new features, organizes events, supports projects, and donates money.

OSS communities are open-minded and enthusiastic, supporting a collaborative mindset, and Blender communities are some of the friendliest and most helpful within the 3D landscape. It's not uncommon to find people who previously belonged to other 3D software communities comment on how positive the Blender community is when they arrive.

The Blender community includes everyone who uses Blender and shares their experience in forums, websites, blogs, podcasts, and videos. The community helps new users, provides tutorials, writes articles, and raises and donates money to the Blender Foundation. Although learning Blender is not an easy endeavor, its great community makes the process considerably less difficult, thanks to the huge amounts of free instructional content that users produce. You'll also find people who are willing to address your doubts and answer your inquiries in forums and on social networks.

In the past few years, a new feature has appeared in the community: content and add-ons stores. Some stores make it easy for content creators and developers to sell their creations (models, textures, materials, animations, and so on) and add-ons (external tools that add specific features to Blender). This development creates a new path for the community, as professionals can now pay developers directly for tools that make their jobs easier, and small studios can buy material made by others to advance their projects quickly and meet their deadlines. (These possibilities were already widely available in other software.)

Following is a partial list of community forums and reference websites, in no particular order:

- **<https://blenderartists.org>:** A forum where you can show your work and get feedback, critique other artists' works, ask questions, or discuss any subject related to Blender and 3D
- **<https://blender.community>:** A collection of Blender communities in different languages on the same platform
- **<https://blender.chat/home>:** A global chat room for Blender users, including developers and professionals
- **<https://blendermarket.com>:** A marketplace for Blender add-ons, training material, and 3D assets such as models and shaders
- **www.blendernation.com:** The main Blender news website, which you can visit daily for updates, new add-ons, interesting works, tutorials, and more

Many other communities (including local user groups), websites, and resources are available online. I encourage you to do some research and find the ones that fit you best.

Summary

Blender has been around for many years. It's free to download and use, even for commercial purposes. The Blender Foundation organizes the software's development from the Blender Institute, and anyone can contribute to it by programming, reporting bugs, donating money, subscribing to the Blender Cloud, and purchasing the Blender Store's products. Blender Animation Studio makes Open Movies that help test Blender in a production environment.

Two of the most attractive features of an OSS like Blender are the ability to play with the core code of the software to make it fit your needs (or your studio's) and the opportunity to interact with its developers and its diverse, open-minded communities.

Exercises

1. What is open-source software?
2. When did Blender's development start?
3. Do you need to buy a license to use Blender commercially?
4. What are the main functions of the Blender Foundation?
5. Can you sell the content you create with Blender?

Blender Basics: The User Interface

This chapter helps you understand how Blender’s user interface (UI) and main navigation features work. Blender has a distinct way of using windows and menus, so you might have to rethink how you work with interfaces while you catch up, but don’t worry; it’ll be fun!

Downloading and Installing Blender

Before you start using Blender, you need to install it, of course! This is really easy: Go to <https://www.blender.org> (Blender’s official website). Once you’re there, click the Download link on the home page or go to the Download tab. There, you will find a panel with the current official version. You have to select your operating system and specify whether you want an installer (for Windows only) or a portable version. (Yes, you can copy a portable Blender to your pen drive and use it everywhere you go.) You must also select whether your OS is 32-bit or 64-bit. (If you don’t know, check your OS version in Control Panel.)

Caution

Before you download and start using Blender, check the minimum hardware requirements and other information at <https://blender.org/download/requirements>. You should also install the latest drivers for your hardware to make sure everything is up to date.

Using Blender with Recommended Hardware

3D has some hardware requirements that other, more basic software (such as software for text editing or office work) doesn’t. In this section, I recommend some hardware that you should have to get the most out of Blender, even though you can use it without this equipment:

- **A mouse with three buttons:** 3D software requires you to navigate a tridimensional world and usually takes advantage of the three buttons of a mouse for this purpose. Having a scroll wheel or a middle mouse button is very convenient. A scroll wheel is optional in Blender, but if you have it, you can use it for

zooming in or out and for things such as scrolling long menus. It will be difficult to work without a middle mouse button, however, so Blender offers you the option to hold down **Alt** and left-click to emulate a middle mouse button (you can find this option in User Preferences, as mentioned later in this chapter), but this process is more uncomfortable and renders other tools that require the use of **Alt+left-click** unusable. Many people also use a pen tablet to control Blender, although it's less common and may require some setup to make it comfortable. On a personal note, I use a pen tablet for painting and sculpting in Blender, and I simply set the pen buttons to mimic those of a mouse. (The pen tip is a left click, and the two side buttons of the pen act as right and middle clicks.)

- **Numerical keyboard (NumPad):** If you don't know what it is, it's the part to the right of a full keyboard that resembles a calculator. You can definitely work without a numerical keyboard (especially in newer versions, in which visual controls help you navigate the scene), but Blender makes good use of it if you have one. Also, this type of keyboard makes working in a 3D world a lot easier, giving you access to camera controls and allowing you to jump to predefined points of view quickly by pressing its keys. Blender offers you the option of using the alphanumerical keys (the numbers above the letters on your keyboard) instead, but this option limits functionality with other tools that use those keys. I have a portable numerical keyboard to plug into my small laptop so I can use it with Blender when I don't have access to a full keyboard.
- **CPU:** All computers and laptops have a CPU. Blender is quite lightweight, so any current computer should be able to run the software. Keep in mind, however, that Blender uses a CPU for most of its computing, and the more complex your scene is, the more powerful your CPU needs to be; otherwise, your computer will start slowing down as you work and decrease performance. I won't recommend a specific CPU here. Just keep in mind that the more powerful your CPU is, the better Blender performs.
- **8GB RAM:** It's important to have a decent amount of RAM installed, as Blender uses it for several purposes, such as storing the scene before rendering. (If your scene takes more memory than you have, you won't be able to render it.) Although you can use the software with a minimum of 4GB of RAM, I recommend having at least 8GB.
- **Graphics card with CUDA or OpenCL support:** If you're planning on rendering with Cycles (Blender's path-tracing render engine), a good graphics card can help, as a GPU (graphics processing unit) is much faster than the CPU. To use the GPU, your graphics card must be compatible with CUDA (Nvidia) or OpenCL (AMD) technology.

Hardware is not within the scope of this book, so if you're looking to get a computer to work with Blender, ask an expert for recommendations that fit your budget.

Using Blender's User Interface

In this section, you'll learn about the main parts of Blender's UI (see Figure 2.1) and see how to adapt it to your needs.



Figure 2.1 This window is what you see the first time you open Blender.

Splash Screen

The Splash Screen (A) is what you see in the center of the interface as you open Blender. Inside the Splash Screen, you'll see a list of presets to create a new file, open or recover a previous session, a list of recent files, and so on.

The first time you open Blender, it also shows some controls and lets you select a different setting. If you had a previous version of Blender installed, you would also see an option to load your settings from the previous version into the new one. I don't always recommend doing that, however, as changes between versions can create conflicts with newer options. I recommend loading old settings only if you made extensive changes in User Preferences and don't want to repeat all that work in the new version, but I'd encourage you to go through the options first to see whether some of them would need to be updated.

Warning

In the rest of the book, I'll use Blender's default settings, and I'll specify when I change something. If you change the settings on your own, keep in mind that you may need to adapt some of the things you see in the book to work with your settings (different keyboard shortcuts, for example).

I generally recommend working with the defaults so that you get used to them, as any software is usually designed to work the way it comes set up. By changing the settings, you may end up losing or missing on some functionality.

How do you close the Splash Screen and start working? Just click anywhere on the interface (outside the Splash Screen), or click the Splash Screen’s image. You can open the Splash Screen again by clicking the Blender icon in the left corner of the Top Bar, next to the main menu, and choosing Splash Screen from the menu that pops up.

Tip

In case you don’t like seeing the Splash Screen every time you use Blender, you can disable it in User Preferences, within the Display options on the Interface tab.

Top Bar and Status Bar

Let’s start with these two parts of Blender’s UI, which are fixed in place and surround the rest of the interface (refer to Figure 2.1):

- **Top Bar (B):** The horizontal bar at the top of the interface, where the main menu resides. It also hosts the Workspace tabs and the controls for scenes and view layers.
- **Status Bar (C):** The horizontal bar at the bottom of the interface. It shows information about the current action or tool, serves as a progress bar when Blender is computing something like a render or physics simulation, and also shows scene statistics. It’s context-sensitive, so it displays different information depending on where your cursor is and what action you’re performing.

Default Editors

As you open Blender, the main interface within the Top Bar and the Status Bar is divided into four parts: 3D Viewport (D), Outliner (E), Properties Editor (F), and Timeline (G). I’ll explain what these editors are for in the “Understanding the Types of Editors” section later in this chapter.

Understanding Areas and Editors

Blender’s innovative interface is defined by the concepts of areas and editors. Essentially, the interface can be divided into nonoverlapping areas, and the user can choose which editor to display in each of those areas. Each division in the interface in Figure 2.1 is an area.

Resizing Areas

The most basic thing you can do with areas is to resize them. If you hover the mouse over the junction between two areas, the cursor will turn into a line with two arrows at its ends, and in that mode, you can click and drag to resize an area.

Splitting and Joining Areas

Areas can be split and joined to adapt the interface to your liking so that it's comfortable to execute the task at hand. Maybe you want a 3D Viewport and a big outliner to organize a complex scene, or maybe you need several 3D Viewports to see your scene from different perspectives. You can do all that.

Every area is rectangular, and the corners are rounded. If you hover the mouse over those corners, the cursor will turn into a crosshair; this means you're ready to perform actions with areas!

Click and drag toward the adjacent area and you'll be able to join both. Click and drag toward the center of the current area and you'll divide it. The direction in which you drag is also meaningful: it will divide horizontally if you drag vertically and vice versa. (See Figure 2.2.)

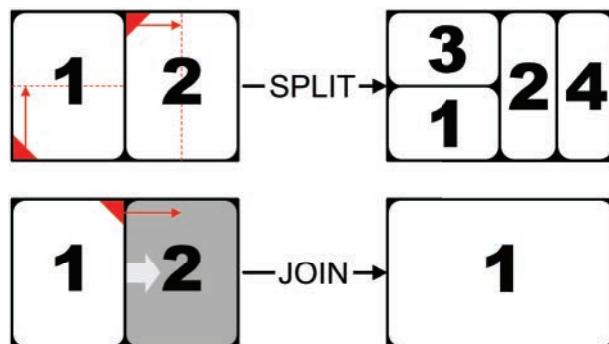


Figure 2.2 The effect of clicking and dragging to split and join areas. Red triangles indicate the position of the click, and red arrows symbolize the direction of the drag action.

During joining, the area that gets darker is the one that will disappear. In Figure 2.2, area number 2 disappears and area number 1 takes the whole space.

An alternative option for splitting or joining areas is to right-click the junction between areas. You'll see a menu with options to split and join. Choose the one you want, drag, and click when you're happy with the result. Press **Tab** while dragging to split switches between vertical and horizontal divisions.

Caution

Keep in mind that to join two areas, you must make sure that both of them have the same size on the shared side that you're trying to eliminate. If clicking and dragging to join doesn't work, check that areas on both sides have the same height or width (depending on how you're joining them).

Swapping and Duplicating Areas

You can perform other actions with areas, such as swapping and duplicating. Hover the cursor over the corner of an area, as you'd do for splitting or joining, and when the cursor turns into a crosshair, do either of the following:

- Hold **Ctrl** while you click, and drag the cursor toward a different area. This action swaps the editors that are being displayed in those areas.
- Hold **Shift** while you click, and drag in any direction. The area will be duplicated in a new window. That new window can be split into more areas as the main interface. Now you have a secondary Blender window that you can use if you have multiple monitors, for example.

Maximizing an Area and Making It Full-Screen

No matter how your interface is divided, you can focus in the current area at any time and turn it full-screen so that the rest of areas disappear temporarily. Just follow these steps:

1. Place your mouse cursor on top of the area you want to turn full-screen.
2. Press **Ctrl+Space** to maximize that area.
3. Press **Ctrl+Space** again to see the entire interface again.

Press **Ctrl+Alt+Space** instead of **Ctrl+Space** for a “true” full screen in which even the Top Bar and the Status Bar disappear.

Understanding the Types of Editors

Inside each area, you can choose to display an editor. *Editors* in Blender are contents that you add to an area, and each of the many available editors contains different tools for different purposes.

Most editors have a *header*: a bar at the top that hosts a menu with options for that particular editor and the tasks you are able to perform in it. You’ll generally find a button in the top-left corner of each editor; this button displays a drop-down list that allows you to choose a different editor to be displayed in that area. (See Figure 2.3.)

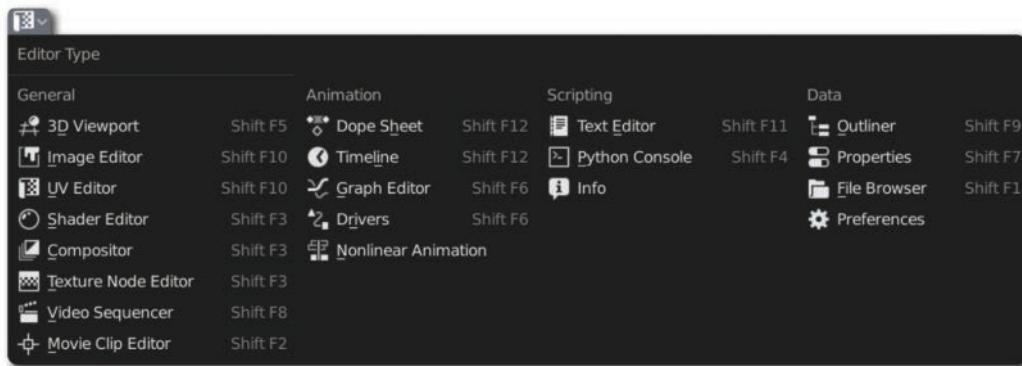


Figure 2.3 When you click the editor selector in the top-left corner of an area, you see this menu, which lists the available editors.

The following are the available editors and what they are used for:

- **3D Viewport:** This editor is where the magic happens, where you create and place objects, model and build your 3D scene, and so on.
- **Image Editor:** This editor allows you to load reference images, view images you're using in the scene, and even paint on images.
- **UV Editor:** When you have to unwrap your model to define how you want the textures to be projected on the 3D surface, this editor is the one you'd use for that purpose.
- **Shader Editor:** A node editor for creating materials.
- **Compositor:** A node editor for compositing the final render, passes, and layers, and for adding effects, color corrections, and so on.
- **Texture Node Editor:** This editor is not useful at the moment, but it's expected to become a node editor for creating textures in future Blender versions.
- **Video Sequencer:** Did you know that Blender includes a video editor? Video Sequencer is it.
- **Movie Clip Editor:** In this editor, you can load videos, create and animate masks on them for later use in compositing, and analyze the footage for camera and motion tracking.
- **Dope Sheet:** This editor helps you control the timing of your animations.
- **Timeline:** The Timeline editor allows you to see the length (in time) of your project, change the current frame, establish keyframes, and perform basic keyframe and timing editions to an animation.
- **Graph Editor:** In this editor, you can fine-tune the curves of your animation to control how Blender interpolates the keyframes of your animation.
- **Drivers:** You use this editor to set up conditions that make one object's property control (*drive*) other properties.
- **Non-Linear Animation:** You can save different actions (animations) in an object and then combine them as strips in this editor. The process is like video editing, but for animations on your characters and rigs.
- **Text Editor:** You use this editor for scripting (you can even run Python scripts from it) or adding text notes to the scene. Text Editor is especially useful if you work with a team and want to add information or instructions about how to use the scene.
- **Python Console:** This built-in console allows you to interact with Blender by using its Python application programming interface (API). This editor is mainly for developers.
- **Info:** The Info editor's console-like interface shows the log of actions that Blender performs, warnings, and so on. Info is also useful for developers.

- **Outliner:** The Outliner shows a tree graph of the elements in a scene—which is very useful when you’re looking for objects or navigating all the elements of a scene. You can select specific objects or groups in complex scenes or even search for them by name.
- **Properties Editor:** The Properties Editor is one of the most important editors in Blender, with different tabs for different groups of options. (Depending on your selection, the tabs will change, as they are context-sensitive.) This editor is where you set up your render size and performance, add modifiers, set the object parameters, add materials, control particle systems, and indicate which measuring units you want for your scene. These tabs are organized from most general to most specific. (See Figure 2.4.)

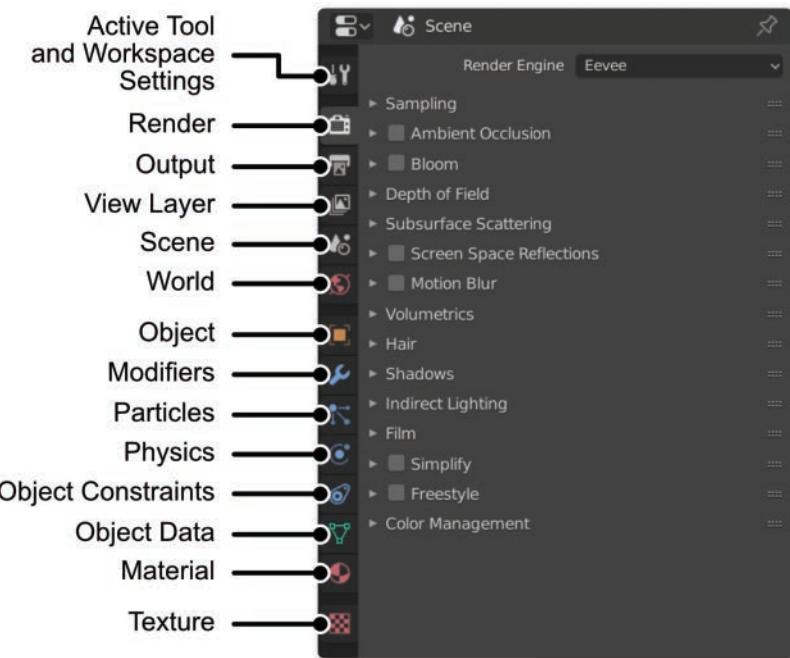


Figure 2.4 The tabs of the Properties Editor. Tabs may vary depending on the current selection, as the editor is context-sensitive.

- **File Browser:** File Browser allows you to navigate your system’s folders to look for images, for example. Keep in mind that you can drag images to other editors, such as dragging an image to the 3D Viewport to use it as a background reference. File Browser is also useful when you’re editing video and frequently need to access the video files and load them into the Timeline.

- **User Preferences:** The User Preferences window has tabs that allow you to customize Blender's keyboard shortcuts and interaction options, change the interface's colors and theme, adjust performance settings, and manage add-ons.

Tip

Any header or menu (including the Top Bar and Status Bar) will be partially hidden if the interface is not wide enough. You can click **MMB** over a menu and then drag to "slide" the contents of the menu and see the parts that are hidden. Alternatively, you can use the scroll wheel to the same effect.

Using Workspaces

Now that you know what areas and editors are, and you've seen how to work with them, Workspaces are the next stop: they allow you to save configurations of areas and editors so that you can access them with a single click.

Blender comes with a series of predefined Workspaces for the most common tasks, but you can customize them to your liking, delete them, and add new ones that adapt to your needs.

You can find Workspaces on the Top Bar, and you can switch among them by clicking the tab with the name of the Workspace you want to go to. (See Figure 2.5.)

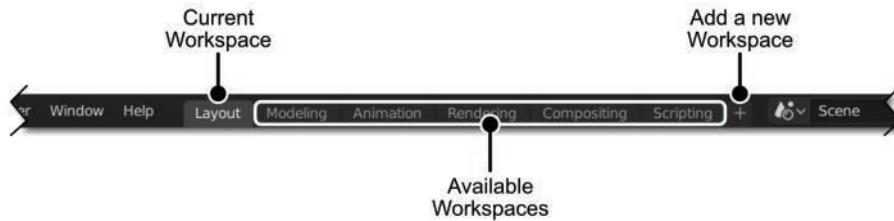


Figure 2.5 Workspaces are on the Top Bar (at the top of the interface).

Workspaces remember the last state you left them in, so if you modify the areas and editors of a Workspace and leave it, the Workspace will be the way you left it when you come back to it.

Tip

You can click and drag through any row or column of tabs in Blender to cycle through all of them and explore them with a single click.

When you right-click a Workspace's tab, you'll see more options, such as reordering the tabs, deleting that Workspace, and renaming it.

You can create a new Workspace based on a default Workspace and adapt it to your needs by clicking the last tab on the right (which has a + sign on it).

Note

Workspaces are saved within the .blend file. If you configure your Workspaces and save the .blend file, when you open it (even on a different computer), Blender will remember the Workspaces you set up. When you open a file, you see a Load UI option, which you can enable or disable. If you disable Load UI, the file will be loaded in your default interface setup; if it's enabled (the default), Blender loads the UI saved within the file itself.

Workspaces also have options that you can configure. (See Figure 2.6.)

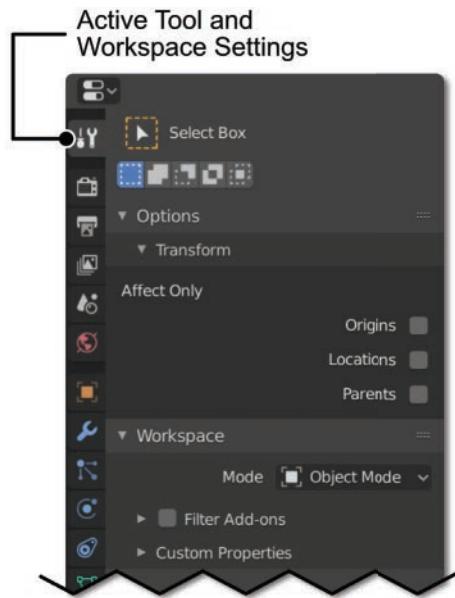


Figure 2.6 In the Properties Editor, you can access the options for a Workspace.

You can find those options in the Properties Editor. On the Active Tool and Workspace Settings tab, you'll find the following options:

- **Mode:** You can select the interaction mode in which the current Workspace will be opened. (For more information about interaction modes, see Chapter 3.)
- **Filter Add-Ons:** You can enable this option and then choose the add-ons you want to filter, customizing which add-ons are shown or hidden in given Workspaces.

Tip

If you're working full-screen in the 3D Viewport, you can also access these Workspace options from the Sidebar (open it by pressing N while your mouse is on the 3D Viewport). This technique allows you to access these options without having to get out of full-screen mode to see the Properties Editor. You'll learn about the Sidebar later in this chapter.

Getting to Know Blender's Interface Elements

This section explores some of the most relevant elements that you'll find in Blender's UI, so you can familiarize yourself with them.

Getting to Know Menus and Popovers

Menus and popovers are very common; you'll generally see them coming out of other menus, editor headers, and buttons. They reveal (among other things) options, tools that you can launch, and check boxes that enable or disable parameters. (See Figure 2.7.)

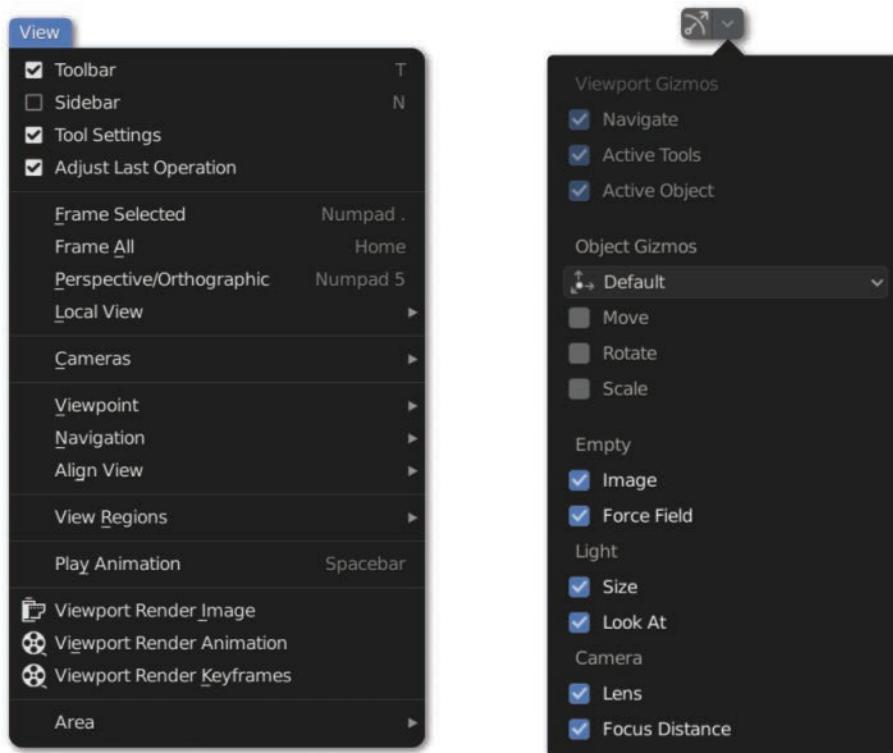


Figure 2.7 Menus (left) and popovers (right). Both types of elements show up when you click a button or a menu and then hide after you choose an option or move the mouse cursor away from them.

Getting to Know Panels

You'll see panels in many parts of the interface, especially in editor Sidebars and the Properties Editor. Panels organize the information and options in blocks with titles. (See Figure 2.8.)

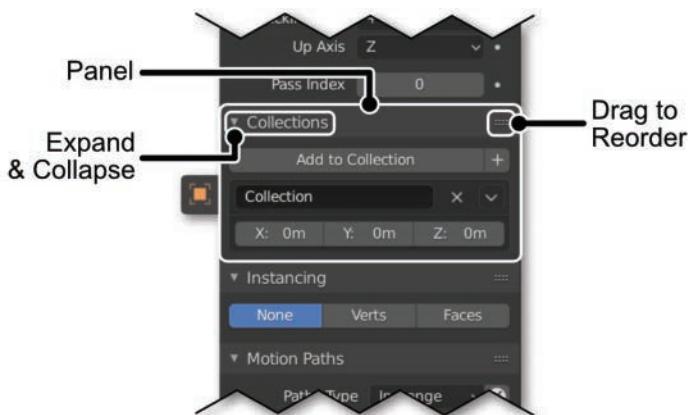


Figure 2.8 Panels and their main controls

Near the title of a panel, you'll always see a triangle. That triangle points to a side if the panel is collapsed, and it points downward if the panel is expanded. You can expand and collapse a panel by clicking that triangle or title. Alternatively, you can press **A** while you hover the mouse cursor anywhere over the panel to collapse and expand.

You can click and drag over the dotted area in the top-right corner of a panel to reorder it (which is easiest to do while the panels are collapsed).

Tip

If you click the title of a panel and drag toward other panels, you can collapse or expand all the panels within that menu—which is very useful when you want to clean up or organize your interface!

Getting to Know Pie Menus

Pie menus are radial menus that show up in certain cases, such as when you press **Z** to select a viewport shading mode. (See Figure 2.9.) When you press a keyboard shortcut that has a pie menu assigned, the options within that menu appear and expand around the mouse cursor's position (represented in Figure 2.9 by the circle in the center).

Now you can rotate around that circle in the center with your mouse cursor, and the option in that direction will be highlighted. (In Figure 2.9, the Solid option is highlighted.) You can click the button with the option that you desire.

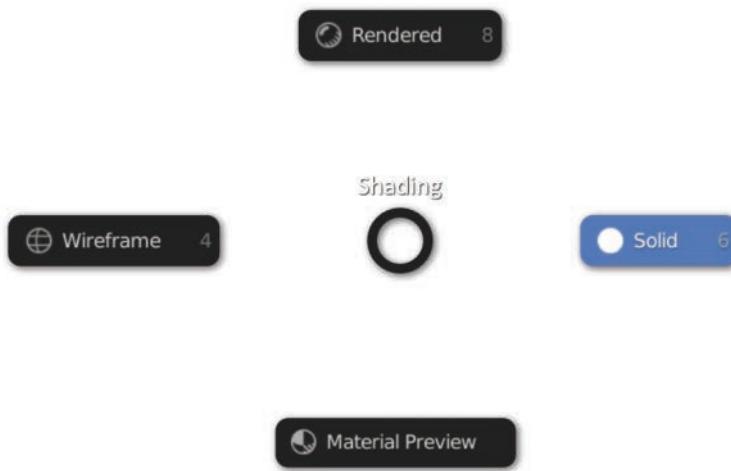


Figure 2.9 A pie menu lets you select an option based on the direction in which you're moving the mouse when the menu pops up.

Pie menus rely on muscle memory; after using them a few times, you will remember the positions of the options, so having to click them is not the most efficient way to use them. Here's how you can make the most of pie menus:

1. Press and hold the key that has a pie menu assigned.
2. Move the mouse in the general direction of the option you want to pick.
3. Release the key that has the pie menu assigned.

That's it! No clicks are involved; just hold and release a key while you move the mouse. When you get used to this process, you'll be doing it faster and faster, and it won't take more time than pressing a simple keyboard shortcut. This technique has an advantage: It allows you to remember only one key for different related actions (in Figure 2.9, viewport shading options) and the direction in which you need to move the mouse.

In the case shown in Figure 2.9, you'd press and hold **Z** on your keyboard while you move the mouse toward the right, and when the Solid option is highlighted, you'd release the **Z** key.

Another thing to keep in mind is that instead of clicking the option you want, you can press the number in the NumPad shown next to each option. You can think of the NumPad as being a wheel that orbits the number 5, so each of the other numbers represents a direction of the pie menu.

Tip

It doesn't matter if you start moving the mouse *before* you press and hold the key that has the pie menu assigned to it, so you don't have to wait to move the mouse until after you press the key. This allows you to use pie menus even faster.

Understanding the 3D Viewport

In this section, I examine the elements of the 3D Viewport editor, which is the main editor in Blender. (See Figure 2.10.)

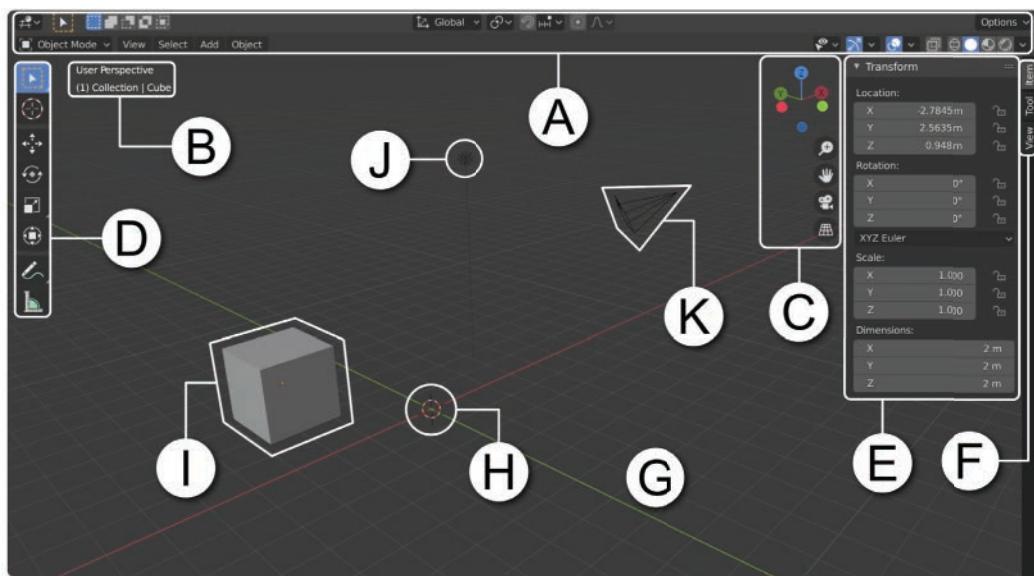


Figure 2.10 3D Viewport, where you'll be working most of the time

Here are the elements marked in Figure 2.10:

- **Header and tool settings (A):** Every editor has a *header*, a horizontal bar at the top or bottom of the current view with menus and options for that particular editor. (See “Understanding the 3D Viewport’s Header” later in this chapter for more information.) The header is divided into two rows if the Tools Settings bar is enabled, in which case the active tool settings are shown on the top row. You can enable or disable the tool settings and header by right-clicking them and selecting the desired check boxes.
- **Viewport text info (B):** In the top-left corner (by default), you see the name of the current view (such as User Perspective, Front Ortho, or Right Ortho). If

you're not sure what camera or view you're using to look at your scene, a quick look at the view name will give you a clue. You'll also find the current frame (the number in parentheses), the current collection (you can organize your scene by using collections, such as folders; more on that later), and the active object (the last one you selected).

- **Navigation gizmos (C):** The top-left corner of the 3D Viewport lets you navigate your scene with easy-to-access shortcuts. (See “Navigating the 3D Scene” later in this chapter for more information.)
- **Toolbar (D):** The toolbar shows a list of tools you can use in the current situation.
- **Sidebar (E):** The Sidebar shows some menus that contain options for the current selection, the views, and the add-ons you have installed.
- **Sidebar tabs (F):** Tabs on the Sidebar let you change which options you see. Generally, add-ons have tabs themselves.
- **Floor grid (G):** The grid represents the floor of your scene, with the X (red) and Y (green) axes being references to the scene's orientation and size. By default, each square in the grid represents 1 meter, but you can use the options in the 3D Viewport's Overlays popover (see “Understanding the 3D Viewport’s Header” later in this chapter) to customize the grid scale and number of divisions.
- **3D cursor (H):** The 3D cursor defines where objects will be created and acts as an alignment tool and pivot point. See “Understanding the 3D Cursor” later in this chapter for more information.
- **Default cube (I):** The first time you start Blender, a cube appears in the center of the scene, so you already have a geometric shape to start working with. You can delete it by pressing **X** or **Del** on your keyboard and confirming whether you prefer to start with a different object or a blank scene.
- **Lamp (J):** If you want your render to look nice, you need lights that illuminate your scene and generate shadows. By default, Blender has a point lamp in the scene to provide basic illumination.
- **Camera (K):** You can't take a *render* (the resulting 2D image or animation generated from your 3D scene) without a camera in the scene. The camera defines the point of view, the field of view, zoom, and depth of field, as well as other options that help you see in the 3D Viewport what will be rendered in the final image.

You'll become more used to all these elements as you spend some time working with them in the 3D Viewport.

Understanding Regions

Any editor is made of *regions*: independent pieces that make up the entire editor. The header, toolbar, and Sidebar of an editor are regions, for example, and they're not fixed

parts of the editor: they can be resized and even hidden when they’re not needed. These are the main regions in an editor:

- **Toolbar:** The toolbar shows Active Tools, which define the action that is performed when you click them. The toolbar can be resized or hidden:
 - Hover the mouse button on the right border of the toolbar until the cursor turns into a double-arrowed line. Left-click and drag to resize the toolbar, which turns first into two columns and then into one column with the name of the tool next to it. Resize it all the way down toward the left border to hide it.
 - You can make the icons bigger or smaller by holding **Ctrl**, middle-clicking, and dragging up and down while hovering the mouse cursor over the toolbar. (This technique also works in many parts of the interface to zoom in or out of menus.)
 - You can hide and show the toolbar by pressing **T** while your mouse cursor is on the editor.
 - When the toolbar is hidden, you can unhide it by pressing **T**, but you also see a little button with an arrow icon on the left border of the editor. Click that button, and the toolbar appears again.
 - Sometimes, you may have many tools on the toolbar, and if the interface is not big enough, not all of them may fit onscreen. When that happens, you’ll see a scroll bar next to the toolbar’s right border; you can also use the scroll wheel on your mouse or middle-click the toolbar and drag up and down to slide the tools and see the ones that are hidden.
- **Sidebar:** You can cycle through all the tabs quickly by holding **Alt** and moving your scroll wheel up or down while your mouse cursor is over any of the tabs.
 - You can show and hide the Sidebar by pressing the **N** key on your keyboard.
 - You can resize it by hovering the mouse cursor over the left border of the Sidebar; the mouse cursor turns into a double-arrowed line. If you resize the Sidebar all the way to the right, it will be hidden.
 - While the Sidebar is hidden, you’ll see a small button with a left arrow on the right border of the editor. Click that button to reopen the Sidebar.
- **Header:** The header is the host of many of the options and tools you can use in any editor. You can hide it, move it from top to bottom, and so on. Right-click any button on the header and navigate to the Header submenu, and you’ll find several options (see the next section):
 - **Show Header:** This option is enabled by default, but you can hide the header if you disable it.
 - **Tool Settings:** If you enable this option, a bar appears over the header, showing the settings for the active tool (the one you’ve chosen on the toolbar),

which you usually see on the Active Tool panel of the Sidebar or the Work-space tab of the Properties Editor.

- **Show Menus:** If you disable this option, the editor menus are hidden in a hamburger icon. If you’re working on a small screen, you may find this option to be useful.
- **Flip to Bottom/Flip to Top:** Choose this option to move the header from the top of the editor to the bottom and vice versa.

Understanding the 3D Viewport’s Header

The header of the 3D Viewport is probably one of the most complex headers in Blender, given that 3D Viewport is also the editor in which you’ll spend the most time and perform the most actions. (See Figure 2.11.)

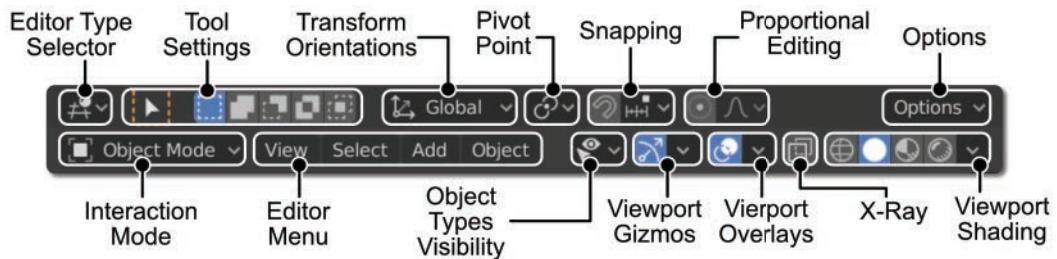


Figure 2.11 3D Viewport’s header and tool settings, involving two rows of buttons that adapt to the width of the interface

You’ll find the 3D Viewport’s header at the top of the 3D Viewport. (Most editors’ headers are at the top of the editor.) Here is a brief explanation of what you can do from the header (from left to right):

- **Editor Type Selector:** Selects the type of editor that is shown in the current area.
- **Interaction Mode:** Selects the mode in which you’re working with your scene, depending on the task you want to do (Edit Mode for modeling, Sculpt Mode for sculpting, and so on).
- **Tool Settings:** Provides a series of buttons and options to control the behavior of the active tool.
- **Editor Menu:** Provides options and tools you can use within a specific editor. In this case, for 3D Viewport, you have the View, Select, Add, and Object menus available.
- **Transform Orientations:** Lets you choose which orientation to choose for performing transforms (move, rotate, and scale), such as the local axis of an object or the global axis of the scene.

What Are Transforms and Edits?

In 3D, you can generally perform two types of actions: transforms and edits. Moving, rotating, and scaling are referred to as *transforms*. *Edits* are changes you make that create or modify the shape (geometry) of objects.

- **Pivot Point:** Provides a reference point in space for transforming an object.
- **Object Types Visibility:** Allows you to show/hide objects and make them selectable or not, based on the type of object (meshes, cameras, lights, curves, and so on).
- **Snapping:** Offers several options to snap the selection to other elements when you perform a transform operation. Click the button with the down arrow to choose the snapping options, and click the switch with the magnet icon to enable or disable snaps.
- **Viewport Gizmos:** Lets you choose which gizmos (manipulators) are shown or hidden in the 3D Viewport's interface. If you click the button with a down arrow, you can choose which gizmos to show or hide. If you disable the switch with the icon, you can hide all the gizmos.
- **Proportional Editing:** Lets you select different falloff methods that affect the objects surrounding your selection when you transform it. A circle around the selection indicates the range of the effect, for example; you can modify its size with the scroll wheel of your mouse. The farther an object is from the selection, the less affected by the transforms it will be. Click the button with the down arrow to set the falloff curve, and click the switch with the icon to enable or disable proportional editing.
- **Viewport Overlays:** Provides a plethora of options to show or hide information that Blender can display over the 3D scene while working on a scene: outlines of selected objects, their origin's position (pivot point), and so on. If you click the down arrow, you get a popover with a lot of options for what to display. If you disable the switch with the icon, all overlays are hidden.
- **X-Ray:** Lets you see through objects. This option works only in Wireframe and Solid viewport shading modes. You can configure the intensity of this option from the Viewport Shading controls.
- **Options:** Gives you some additional options that are contextual to the interaction mode you're working in.
- **Viewport Shading:** Lets you choose how you want to see your scene. Generally, you choose one of the four options: Wireframe, Solid, Material Preview, or Rendered. The arrow to the right of each option lets you choose options to customize that specific shading mode.

Navigating the 3D Scene

Now that you know what's going on in the 3D Viewport, you're ready to see how to navigate it to check your scene and get in touch with the world you're creating. When your cursor is hovering in the 3D Viewport, you can perform a variety of actions to change your point of view. You also have different ways to navigate through your scene.

Navigating the 3D Scene by Using the Mouse, Keyboard, and NumPad

The main way to navigate the 3D scene is to use the mouse and keyboard. In Figure 2.12, you see the main controls for navigation.

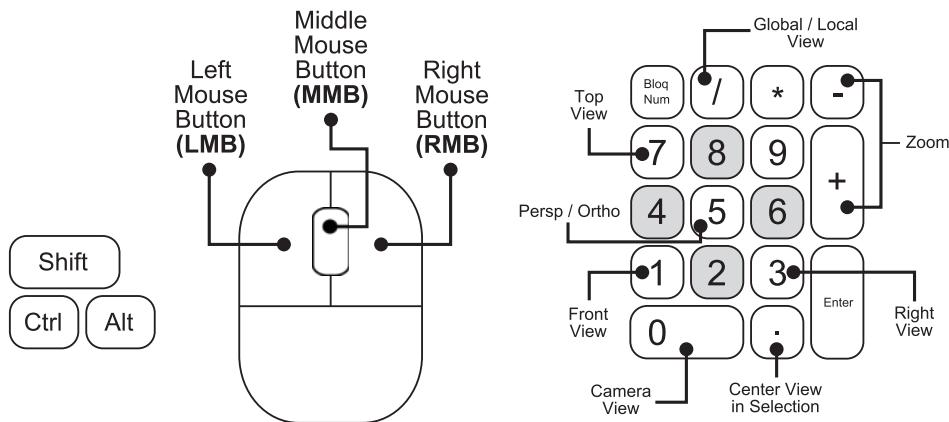


Figure 2.12 Most of the buttons you can use to navigate in the 3D scene are on the mouse and NumPad. The buttons highlighted in gray let you orbit the camera in increments.

Following is a list of the navigation actions you can perform with the mouse and keyboard:

- **Pan (Shift+MMB):** This action moves the camera parallel to the current view.
- **Orbit (MMB or NumPad 4, 8, 6, and 2):** This action rotates the camera around the scene. Hold **Alt** while orbiting with **MMB** for the point of view to snap to predefined views with increments of 45 degrees between them.
- **Zoom (scroll wheel or Ctrl+MMB drag or NumPad + [plus sign] and NumPad - [minus sign]):** This action moves closer to or farther from a point.
- **View Selected (NumPad .):** This action zooms and centers the camera in the selection.
- **Predefined (Front, Right, and Top) Views (NumPad 1, 3, 7):** This action changes the point of view aligned to one axis. Press **Ctrl** at the same time to get

the opposite view (Back, Left, and Bottom). Combine the previous keys with **Shift** to align the view to the selection’s orientation.

- **Perspective/Orthographic Switch (NumPad 5):** This action switches between Perspective and Orthographic view modes.

Auto Perspective

By default, Blender automatically switches between Perspective and Orthographic modes when you start orbiting or when you jump to a predefined view (such as front, side, or top). If you don’t like this behavior and prefer to switch between Perspective and Orthographic modes only when you decide to do so, you can switch the Auto Perspective option off in User Preferences, using the Orbit and Pan panel of the Navigation tab.

- **Camera View (NumPad 0):** This action jumps to the point of view of the active camera. Select a camera and press **Ctrl+NumPad 0** to make that camera the active one. Press **Ctrl+Alt+NumPad 0** to place the active camera in the current view. Keep in mind that with **Ctrl+NumPad 0**, you can turn any object into a camera, so don’t worry if you accidentally had an object selected when you pressed that shortcut and are suddenly seeing the scene from a strange point of view; this option is meant to help you orient objects in a different manner. You can use this feature to see the scene from the point of view of a directional light, for example, which will give you a better sense of what the light will illuminate.
- **Global View/Local View (NumPad /):** Local View hides everything except the selection so that you don’t have other objects blocking your view while you work. Press **NumPad /** again to switch back to Global View. Select an object and press **M** to move it out of Local View if you don’t need it anymore.

Caution

While you’re in Local View, you will lose access to some options that require you to be in Global View to function properly. If you miss some option while working, make sure that you’re not in Local View. You can find out whether you’re in Local View by checking the 3D Viewport’s text info in the top-left corner. If you’re in Local View, you’ll see (Local) next to the current view’s name.

- **Walk Mode (Shift+):** This action moves the point of view slowly around the screen. Use the arrow keys (**Up**, **Down**, **Left**, and **Right**) or **W**, **A**, **S**, and **D** to navigate through the scene as though you were moving in a video game. Use **Q** and **E** to move up and down. Use the mouse to rotate the camera. You can increase or decrease the moving speed with the scroll wheel. Press **G** to enable a gravity effect. The experience in Walk Mode with gravity enabled is similar to playing in a first-person videogame, as the camera falls down to stay on top of the scene’s geometry, allowing you to “walk” on the surfaces. You can press **Shift** to move faster and **V** to jump. Left-click to accept the movement and right-click to cancel it.

An important thing to mention is that this keyboard shortcut may be different depending on your keyboard layout. On a Spanish keyboard, for example, the shortcut is **Shift+Ñ**.

Fly Mode

Fly Mode is an alternative to Walk Mode. You can switch between both in User Preferences (discussed later in this chapter), using the Fly and Walk panel of the Navigation tab. You'll find a switch that lets you choose which of the two you prefer to use when pressing **Shift+F**.

In Fly Mode, you can fly through the scene instead of walking. You can rotate the camera with the mouse, pan by pressing **MMB** and dragging, and fly forward and backward with the scroll wheel. (You also control the speed of flight with the scroll wheel.) Press **LMB** to accept the movement and **RMB** to cancel it.

Both Fly Mode and Walk Mode can be used to look at the scene from a camera. Accepting the movement will change the position and orientation of the camera, so these navigation modes are interesting to use to control the camera's point of view.

Navigating from the View Menu

Sometimes, you don't have a NumPad (if you're working on a laptop, for example). You can find most of the navigation controls on the View menu of the 3D Viewport's header.

Navigating with the 3D Viewport's Navigation Gizmos

As shown in Figure 2.10, the top-right corner of the 3D Viewport has a few navigation controls, which are extremely useful if you're working with a computer without a NumPad or if you don't have a three-button mouse (when working with a trackpad or a pen tablet, for example).

You'll find four buttons and an axis. The first button on the left can be clicked to switch between Perspective and Orthographic views; the other three let you click and drag to pan, orbit, and zoom, respectively. The axis represents the current orientation of the view; it can be clicked and dragged to orbit, and if you click any of the circles at the extremes of the axis, the view will jump to the point of view they represent (top, bottom, right, left, front, and back).

Selecting Objects

To do anything in Blender, you'll need to use selections.

By default, you can select objects by left-clicking them. (You can switch to RMB selection in User Preferences.) Selected objects are highlighted with an outline.

Hold **Shift** while you click to add objects to the selection.

Active Selection and Subtracting Objects from the Selection

Blender uses the concept of active selection: the last object that you selected. When you add several objects to a selection, the last one has the brightest outline: that object is the active object. When you change parameters in the Properties Editor, switch interaction modes, add a material, and so on, the changes affect only the active object. You can also use the active selection as a pivot point.

To change properties in the active selection and apply them (where applicable) to the rest of the selection, hold **Alt** while you change that property.

If you've selected several objects, you can **Shift+left-click** any of them to turn it into the new active object.

For subtracting objects from the selection, you have to **Shift+left-click** them. **Shift+clicking**, however, turns a selected object into the active object, so keep that fact in mind. **Shift+left-click** to subtract the active object from the selection, and if the object is not the active object, **Shift+double left-click** it (first to turn it into the active object and then to subtract it from the selection).

There are other methods for performing selections: box, circle, and lasso (see Figure 2.13).

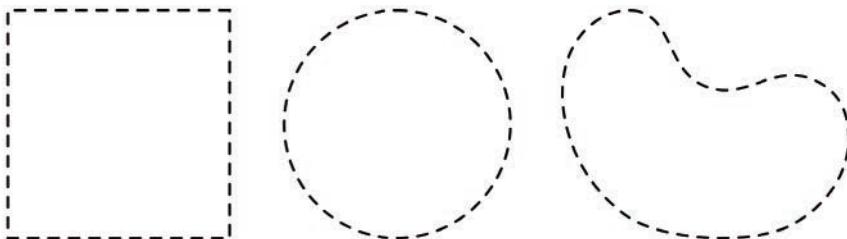


Figure 2.13 Box selection (left), circle selection (center), and lasso selection (right)

Here is how to perform these selections:

- **Box selection:** Press **B** and **LMB** drag to determine the size and position of the box. By default, any objects inside the box will be added to the current selection, but you can subtract them by dragging the box with **MMB** instead of **LMB**.
- **Circle selection (**C**):** Press **C**, and a circle appears around the cursor. Move the scroll wheel to modify the size of the brush; then click and drag with **LMB** to add to the selection or drag with **MMB** to subtract it. Press **Esc** or **RMB** to exit Circle selection mode.
- **Lasso selection:** Press **Ctrl+RMB** and drag to draw the lasso shape over the objects you want to select. The objects inside the lasso will be added to the current selection. Press **Shift+Ctrl+RMB** to deselect by using the lasso.

Selecting All and Deselecting All

In Blender, you can select all the objects in the scene by pressing **A** on your keyboard. To deselect everything, press **Alt+A**. Remember that in Blender, the **Alt** key is usually used for subtracting or making the opposite action of the key you mix it with. **I** is used to add a keyframe, for example, and **Alt+I** is used to remove it; **G** is used to move, and **Alt+G** resets the position of an object.

Using Active Tools to Perform Selections

So far, I have explained how to use keyboard shortcuts to perform selections. Since Blender 2.80, however, a new method is available: Active Tools.

On the toolbar, the first button from the top is for selections. If you pay attention, some tools have a little triangle in the bottom-right corner; this triangle means that the tool has options. Click and hold that button to choose those options.

You can also press the keyboard shortcut assigned to that Active Tool to cycle through the different options. In the case of selections, the keyboard shortcut is **W**. Press **W** repeatedly, and you'll cycle through the selection Active Tools.

Choosing an Active Tool means that whenever you left-click the editor (in this case, the 3D Viewport), that action will be performed, so if your Active Tool is one of the selection modes, selection will happen when you left-click and drag. (Left-clicking alone still selects as usual.)

As you'll recognize, the Active Tools for selection are Box Select, Circle Select, and Lasso Select. The first one, called Select, has an icon of the mouse cursor and a crosshair: this option lets you select and drag (move) an object when you left-click and drag. The others perform box, circle, and lasso selections, respectively, when you left-click and drag.

Keep in mind that Active Tools work a bit differently from the keyboard-shortcut selections, as their purpose is to work while you're left-clicking.

All the selection Active Tools add to the selection, and you can hold **Ctrl** while left-clicking and dragging to subtract from the selection.

You'll find options for selection action (subtract, intersect, add, and so on) and radius (in the case of circle selection) on the Active Tool and Workspace tab of the Properties Editor, the Tool Settings bar on the header, and the Active Tool panel on the Tool tab of the Sidebar.

Understanding the 3D Cursor

Something you may not understand when you first open Blender is why a crosshair circle is always present in the middle of the 3D Viewport. What is its function? This circle is the *3D cursor*—a special feature in Blender. Although this cursor can be a little disturbing initially, it becomes really useful when you get used to working with it.

Here are the main functions of the 3D cursor:

- Sets the location where new objects will be created
- Serves to align objects
- Serves as a pivot point to rotate or scale objects

Press **Shift+S** to reveal the Snap menu (see Figure 2.14), which includes various options, some of which require the use of the 3D cursor.

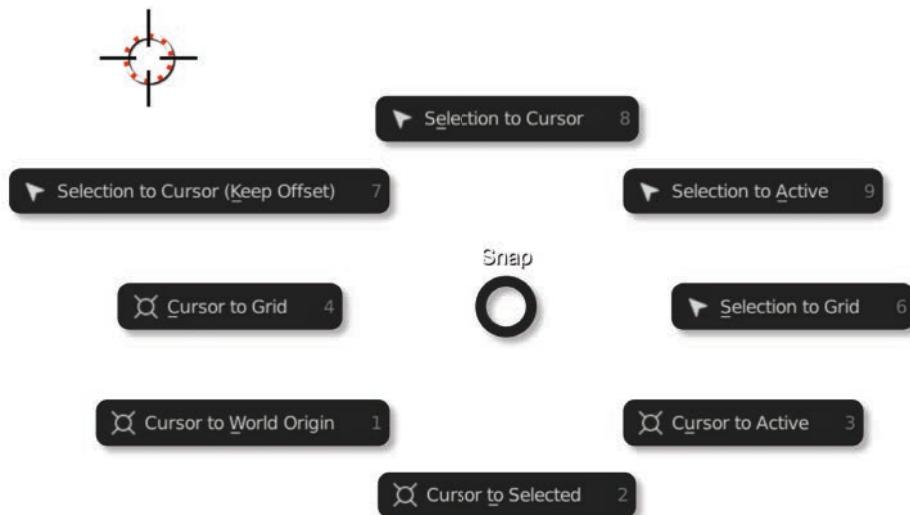


Figure 2.14 The Snap pie menu (press **Shift+S**) and 3D cursor (top left)

If you want to align an object to a specific place on the surface of another object, for example, you can select a vertex of that object, press **Shift+S**, and select Cursor to Selected. Then select the other object, press **Shift+S**, and click Selection to Cursor.

Sometimes, placing the 3D cursor and using it as a pivot to rotate or scale objects comes in very handy! Suppose that you want to pose a character. Thanks to the 3D cursor, you don't need a skeleton for simple posing; you can select the vertices of the leg, place the 3D cursor in the articulation, and rotate those vertices by using the cursor as a pivot point.

You can use the 3D cursor as a pivot point in two ways:

- Select the 3D cursor as a pivot point from the Pivot Point popover on the 3D Viewport's header.
- Pressing . (period) on the keyboard gives you a pie menu with the same options for selecting a pivot point. Choose 3D Cursor, and you're all set.

At this point, you may not be familiar with Blender terms such as *transforms*, *vertices*, and *skeletons* or know how to access them. Don't worry; you'll learn about them in the following chapters.

Note

If you change the 3D cursor to a pivot point and want to go back, remember that Blender's default pivot point is Median Point, which takes the average position of the pivot points in the selection as a pivot point. I recommend that you try the other pivot points to see how they work.

Placing the 3D Cursor

There are three main ways in which you can place the cursor in the spot where you want to use it:

- Press **Shift+RMB** where you want to move it. If you do this over a 3D object, it will snap to its surface under your mouse cursor's position. It will always move from your point of view's plane, so keep that in mind: you can make sure it stays on the floor, for example, if it's already on the floor and you place it while using the top view.
- Enable the 3D cursor Active Tool and left-click the 3D Viewport. The effect is the same as using the previous option.
- Press **Shift+S** to move the 3D cursor to the selection's position, for example.
- Position the 3D cursor with precision by inserting a numerical 3D location into the 3D Cursor panel, on the View tab of the 3D Viewport's Sidebar.

Understanding Blender's User Preferences

From the Edit menu, choose User Preferences. Blender's User Preferences appear in a new window, which you can close when you're done selecting your preferences (see Figure 2.15). You can also open User Preferences by choosing it as the editor to display in any area.

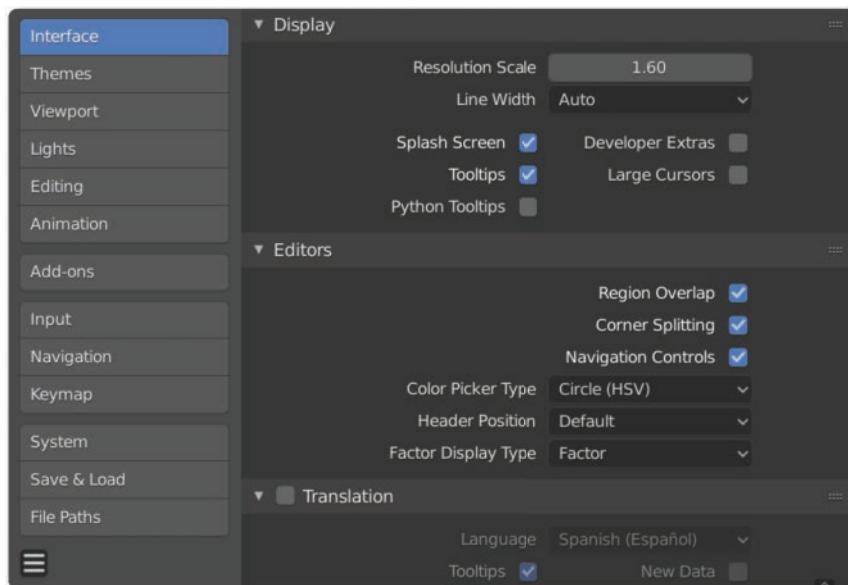


Figure 2.15 Blender's User Preferences editor. You can find it on the Edit menu of the Top Bar.

On the left side of the editor is a series of tabs, each with a related set of options:

- **Interface:** On this tab, you can control the scale of the interface (useful if you have a high-resolution screen, for example) and the way that the interface works.
- **Themes:** Here, you can create your own color schemes and some other look-and-feel settings to make Blender more appealing or to fit your corporate color scheme.
- **Viewport:** On this tab, you can control options for the 3D Viewport, such as the quality of the rendering (increase it for better quality, or reduce it to improve performance).
- **Lights:** If you use Studio Lights, MatCaps, or HDRIs for Material Preview and Rendered viewport shading modes, you can install and set your own images and presets on this tab.
- **Editing:** These options define how Blender works when you're editing objects.
- **Animation:** These options control animation, keyframes, and animation curves (F-Curves).
- **Add-ons:** Within this tab, you can manage extensions that come with Blender or install others that add new functionalities to Blender. Most of these extensions are disabled by default, but you can look for the ones you're interested in and turn them on to use them while you work.
- **Input:** If you want to emulate a NumPad or three-button mouse, or configure the double-click speed or drag thresholds, this tab is the place to go.
- **Navigation:** You can find some options for 3D navigation on this tab.
- **Keymap:** You can edit keyboard shortcuts on this tab, as well as set some other options that determine how you work with Blender by using your mouse and keyboard. This tab also lets you create, edit, export, and import custom keymaps. In fact, Blender comes with a predefined keymap that aims to help you make the transition to Blender from other 3D software (called the Industry Standard keymap).
- **System:** On this tab, you can tell Blender which hardware to use for rendering (you have to set it up if you want Blender to use your graphics card for rendering) and memory management, and how to deal with sound.
- **Save & Load:** These options control autosaves and backups, presets for saving and loading files, and related actions.
- **File Paths:** This tab defines general file paths, what external software to link with Blender (such as a default player for rendered animations or an image editor), and how Blender saves files.

Saving User Preferences

User Preferences are saved automatically by default when you close Blender. In the User Preferences editor, you'll find a button with three horizontal lines in the bottom-left corner. If you click that button, you'll see several options:

- **Load Factory Preferences:** This option resets User Preferences to the default status (more in the next section).
- **Revert to Saved Preferences:** If you change something in the preferences but don't save them, you can revert them to the latest saved status. This option is handy if you want to play with different settings to see what happens without fear of breaking something. Just remember to revert them to their last saved status before closing Blender; otherwise, they will be saved automatically!
- **Save Preferences:** This option lets you save the current preferences manually. It creates a new configuration to revert to when you use the Revert to Saved Preferences option. You'll have to use this option to save the changes in User Preferences if you disable Auto-Save Preferences.
- **Auto-Save Preferences:** This option is enabled by default, but you can disable it if you prefer to save preferences manually.

Resetting User Preferences

If you've changed something and don't know how to go back, or if you want to restart Blender's User Preferences, you can use either of two methods:

- Choose File > Defaults > Load Factory Settings. This command restarts everything in Blender in factory configuration. If you want to save these settings, you have to save User Preferences manually (as explained in the previous section) and also save the startup file (as explained in the next section).
- In the User Preferences editor, choose Save & Load > Load Factory Preferences. (You can find the Save & Load button in the bottom-left corner of the User Preferences editor.) This option restarts preferences but not files, Workspaces, areas, editor arrangements, and so on. If you want to save the new settings after using this option, you must save the preferences manually (as explained in the previous section).

Creating Your Own Startup File

So far, you've seen how customizable the interface is. What if I told you that you can create a file that will be opened every time you start Blender, including Workspaces, panel positions within menus, objects in the scene, cameras, and visibility settings? Well, you can! This file is called a *startup file*.

First, arrange everything as you want it in your interface. Then, when you’re done, choose File > Defaults > Save Startup File. The next time you open Blender, the file is what you’ll see.

This technique is interesting because it allows you to set up everything according to your needs; you don’t have to repeat the process every time you start Blender and work on a new file.

Summary

At this point, you understand how Blender’s interface works. You know how areas divide the interface and what types of editors you can display in them, and you know the shortcuts for navigating the 3D scene. You also know that you can perform easy, quick customization if you don’t like the default interface. In Chapter 3, “Your First Scene in Blender,” you learn how to manipulate objects and get the job done.

Note

If you want to learn more about every detail of the interface and navigation (or any other part of Blender), visit Blender’s official user manual at <https://docs.blender.org/manual/en/>.

Exercises

1. Create a new Workspace, splitting and joining areas to get a single area, and then delete the Workspace.
2. What is the NumPad used for in Blender?
3. Select all objects in the scene, and deselect them again.
4. What are the main functions of the 3D cursor, and how is it used?
5. Is it possible to change keyboard shortcuts in Blender? If so, how?
6. What format does Blender use for saving files?

3

Your First Scene in Blender

You've been introduced to the basics of Blender, and with practice, you'll have the interface under control. It's time to create objects; interact with them; add modifiers, materials, and lights; and then render your creation. This chapter presents a very simple exercise to help you better understand how to create your first scene. You also learn about Blender Render and Cycles, the two render engines included by default in Blender. If you're using Blender for the first time, you'll find this chapter to be especially useful. The idea is that after reading this chapter, you have a basic understanding of the workflow to create a scene in 3D and export it as an image.

Creating Objects

When you open Blender, you'll find the familiar default cube sitting in the middle of the scene. You can use that cube to build your model, or you can delete it. To delete objects in Blender, just select them, press **X** or **Del**, and click Delete in the dialog box that appears to confirm the deletion. (If you press **Del** instead of **X**, you won't be asked to confirm.)

To start, you want to create an object. There are different ways to do it:

- Choosing an option from the Add menu in the 3D Viewport's header.
- Pressing **Shift+A** in the 3D Viewport. (The Add menu from the previous option will appear at your mouse cursor position.)
- Pressing **F3** to display the Search menu, and typing the name of the object you want to create. The menu will filter the options/tools that include what you've entered. If you type **cube**, for example, the menu will show the option Add Cube; click that option, and the cube will be created.

When you use any of these options, the object is created in the position of the 3D cursor inside the 3D scene.

After you create an object, the Adjust Last Operation menu in the bottom-left corner of the 3D Viewport will show the options available to control that object. If you create a cylinder, for example, you'll be able to control its parameters later, such as size and number of sides.

Adjust Last Operation Menu

After you perform any action that can be adjusted afterward, the Adjust Last Operation menu will show up in the bottom-left corner of the 3D Viewport. The menu can be collapsed or expanded by clicking its title (which will show the name of the last action).

Inside the menu, you'll find all options available to tweak the last operation. For example, if you move an object, you will be able to modify the final position in X, Y, and Z; adjust the orientation; and enable or disable proportional editing. Make sure to look at these options after using tools and options, as sometimes, you may discover interesting possibilities you didn't know about before.

If you don't like having this menu enabled all the time, you can hide it from the View menu on the 3D Viewport's header by clicking the option Adjust Last Operation.

Whether the menu is enabled or disabled, you can always call it on a pop-up that will appear at your mouse cursor's position when you press **F9**. Some people prefer hiding the menu and use it (by pressing **F9**) only when they need it.

Make sure that you have adjusted anything before your next action. For example, if you move an object after creating it, the Adjust Last Operation menu will present the options for the Move tool instead of the options for object creation. You can't go back to a previous operation to recover this menu; you'd have to undo (**Ctrl+Z**) and perform that action again.

Animation software often has a test object. In Blender, that object is the *monkey head* (called *Suzanne*), and you'll use it for the test scene in this chapter. Create a monkey head mesh, using any of the methods described earlier in this section. Then create a plane, as this plane later will serve as the floor of your scene. Don't worry if the head and plane intersect in the middle of the world and are not correctly aligned; you'll adjust them in the next step.

Moving, Rotating, and Scaling

After you create objects in your 3D scene, you need to be able to control where they are located, how they are oriented, and what size they are. In this section, you see how to do just that. Moving, rotating, and scaling are the three different transform operations you can perform on any 3D object, and there are several ways to do it.

Using Active Tools

The most obvious way to transform an object is to use Active Tools: the buttons with icons that are shown on the 3D Viewport's toolbar (you can show or hide this bar pressing **T**), as shown in Figure 3.1.

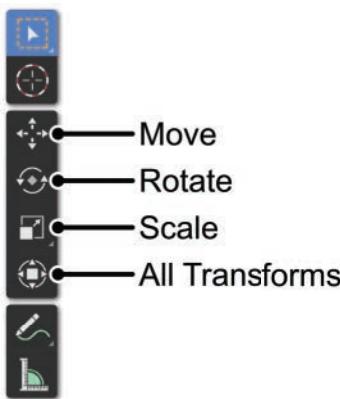


Figure 3.1 Active Tools to move, rotate, and scale objects, located on the 3D Viewport's toolbar

It's simple: you choose the Move, Rotate, or Scale tools, and they become the Active Tools. The manipulators for the selected type of transform will be always shown for the current selection, and you can click and drag parts of those manipulators to perform the transformation. (For more information about using manipulators, see the next section.)

There is a fourth Active Tool for transforms, conveniently named Transform, as it shows manipulators for moving, rotating, and scaling simultaneously.

Although this method for transforming objects can be obvious to new users, it's not always the most efficient method. Sometimes, you may prefer that your Active Tool be a different one, and you have to switch back and forth.

It's convenient when you have the objects in your scene and all you need to do is to place them. The purpose of Active Tools, after all, is to stay active so you can use them repeatedly, but if you're using different tools often, Active Tools may not be your best option.

Tip

You can access the toolbar while it's hidden in a pop-up menu by pressing **Shift+Space** and clicking the desired Active Tool within the menu. In this menu, you'll also see each tool's keyboard shortcut. If you press that shortcut when that menu (**Shift+Space**) is shown, you'll set that tool as the Active Tool. But if you press the same shortcut without the Active Tool menu showing, you'll launch the normal tool, which is not persistent and will stop working after you perform the action.

Essentially, if you want to move many objects in a row, you can use the Active Tool by pressing **Shift+Space** and then **G**; this action will enable the Move tool as the Active Tool (the same as clicking the Move Active Tool on the 3D Viewport's toolbar). If you only want to move the current selection and keep doing other actions, you can just press **G** to use the Move tool, which will be disabled when the action is accepted. Read the next sections for more information about transforming objects by using keyboard shortcuts.

Using Manipulators

There is an option to show manipulators for transforms while using other Active Tools. All you have to do is enable them on the Viewport Gizmos pop-up within the 3D Viewport's header. (See Figure 3.2.)

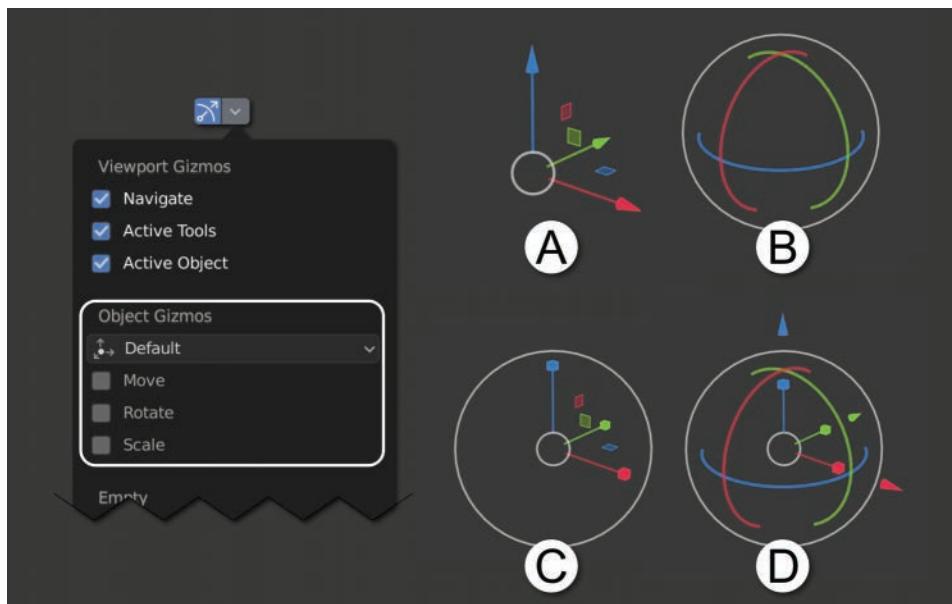


Figure 3.2 Manipulators in the Viewport Gizmos menu on the 3D Viewport's header and different manipulations

When you want to transform objects or elements in the 3D scene, Blender offers manipulators that help you control those transformations. The following are the manipulators:

- **Move (A):** Changes the position of an object in space
- **Rotate (B):** Controls the orientation of an object
- **Scale (C):** Manipulates the size of an object
- **All Transforms (D):** Allows you to use more than one transform manipulator at the same time

In the 3D Viewport's header, you can select the type of transform you want to perform. If you press **Shift** while clicking different transform icons, you can perform multiple transforms at the same time. (In Figure 3.2, example D shows all three transform manipulators being used at the same time.)

Using the manipulators, you can move, rotate, and scale objects. These manipulators appear at the pivot point of the object (marked as a little orange spot called the *origin* in Blender), and you perform an action with them by using the following controls:

- Left-click one of the axes to make the object move, rotate, or resize on that specific axis. (X is red, Y is green, and Z is blue.) Left-click again to confirm the transform. Or press **Enter** to confirm or **Esc** to cancel.
- To enable Precision Mode, press and hold **Shift** after you click to transform. This action makes the transform slower, allowing you to make precise adjustments.
- To lock one axis and manipulate the other two, press and hold **Shift** before you click the axis you want to lock. If you press **Shift** and then click the Z-axis to move it, for example, the object actually moves on the X- and Y-axes, as the Z-axis is locked. (This option works only for moving and scaling; it is not available for rotations.) On top of using keyboard shortcuts, you can use the little squares present in the Move and Scale manipulators. You'll see that, for example, there's a green square between the X- and Z-axes, and it's green because it locks the Y-axis (green).
- Move and Scale manipulators have a small white circle in their centers. Click and drag the circle of the Move manipulator to move the object, using the current point of view as a reference (dragging it parallel to the view). Click and drag the small white circle of the Scale manipulator to scale the object on every axis.

The Rotation manipulator also has an outer white circle but is slightly different; click and drag that circle to rotate the object, using the current point of view as the rotation axis. Instead of having a small white circle in its center, the manipulator for rotations has a spherical shape in transparent gray, and its axes are drawn on the surface of that sphere. Click and drag anywhere within the Rotation manipulator's sphere (without clicking any of its axes) to enter Orbit Mode, which allows you to rotate on all axes at the same time.

- Hold down **Ctrl** while using these manipulators to switch between normal transforms and Snap Mode. This feature allows you to snap to several elements while you perform transforms. If snapping is enabled, holding down **Ctrl** frees the object when transforming; if it's disabled, holding down **Ctrl** enables the snapping. This feature is very useful because you won't need to continuously turn the Snap tool on and off by clicking the Snap icon on the 3D Viewport's header. You'll learn more about snapping tools later in the book.
- In the 3D Viewport's header, you can select Pivot Point and Transform Orientation. Pivot Point defines the point around which objects rotate and scale. By default, Transform Orientation (access this menu by pressing **Alt+Space**) is global, which means that it's aligned to the 3D World axes (default scene axes: X is left/right, Y is front/back, and Z is top/bottom). You can switch Transform Orientation to the local axes of the selection to transform objects using their own orientation.

Tip

If you don't like the default behavior of transforms in Blender (click once to start transforming, and click a second time to confirm), you can activate the Release Confirms option on the Input tab of User Preferences. Release Confirms makes the transform behavior faster so that you can click and drag, and the transform is confirmed as you release the mouse button. This behavior is typical in other software.

Using Keyboard Shortcuts (Advanced)

Although you can use manipulators easily, the expert, really fast way to transform objects in Blender is to use keyboard shortcuts. Sometimes, the manipulators are useful, but most of the time and especially for simple transforms, using the keyboard is faster and more efficient (even though it requires a bit of getting used to and memorizing the keyboard shortcuts). Here are some of the most relevant keyboard shortcuts that make transforms easier and faster:

- Press **G** (Grab) to move, **R** to rotate, and **S** to scale. When you do these things to move and rotate the objects, they move and rotate according to the view. Left-click or press **Enter** to confirm, and right-click or press **Esc** to cancel.
- After pressing **G**, **R**, or **S**, if you press **X**, **Y**, or **Z**, the selection transforms only on that global axis. Press **X**, **Y**, or **Z** twice to align to the selection's local axis.
- Press **R** twice to enter Trackball Rotation Mode, which makes the object rotate in all axes simultaneously following your mouse movements.
- As an alternative to the previous option, when you're transforming with no attachment to a given axis, you can press **MMB**. Lines for the axes appear, and if you move the object close to one of those lines, it is automatically locked to that specific axis.
- The options for precise transforms, snapping, and axis locking using **Shift** and **Ctrl** while transforming with manipulators also apply when you use keyboard shortcuts. Press **G** and then **Shift+Z** to translate the object in the X- and Y-axes at the same time, for example.

Numerically Precise Transforms

When you're performing a transform, Blender allows you to input numerical values. If you look at the 3D Viewport's header when you are rotating an object, you'll find that the header buttons disappear and are replaced by a display of the values of the transform in action. At this point, you can enter values directly from your keyboard, and Blender will use them for the current transformation. Here are two examples:

- To move an object 35 units on the X-axis, use manipulators and write the desired numerical value while dragging. Press **G** to move; then press **X** to snap the object's movement to the X-axis. Now you can drag the object through the X-axis. Type **35** on your keyboard, and the object moves 35 units on the X-axis. Left-click or press **Enter** to confirm the operation.
- Press **R** to rotate, press **Y** to snap to the Y-axis, and enter **-90** on your keyboard to rotate an object -90 degrees on the Y-axis. (When you're entering a numerical

value for a transform, you can add the negative value by pressing the minus key at any time, before or after the number. If you press the minus key again, the value becomes positive.) Left-click or press **Enter** to confirm the operation.

Not only that, you can even enter mathematical expressions to save you time if you start by writing an equal sign (which makes Blender understand that you're writing an expression instead of just a number). For example, you can press **R**, **Z**, and then write $=360/12$ to rotate an object a fraction of a whole circle in the Z-axis without having to calculate it on your mind or spending time opening the calculator for complex operations. When you do this, the information in the header will not only display the expression you're writing, but also show the resulting transform. In the case of the previous example, the header would show this: Rot:[360/12] = 30° along global Z.

As you can see, using this method makes transformations really fast and easy to perform. The shortcuts are intuitive, and you can use them in most editors; **G**, **R**, and **S** always move, rotate, and scale, for example.

Using Menus

You can also use numerical fields in menus to transform objects. You'll find such fields in two places of the interface (see Figure 3.3):

- 3D Viewport's Sidebar (press **N** to show and hide). Within the Sidebar, pick the Item tab, and you'll find the Transform panel, where you can see numerical fields for every location, rotation, and scale axes.
- In the Object tab of the Properties Editor, you will also find the Transform panel.

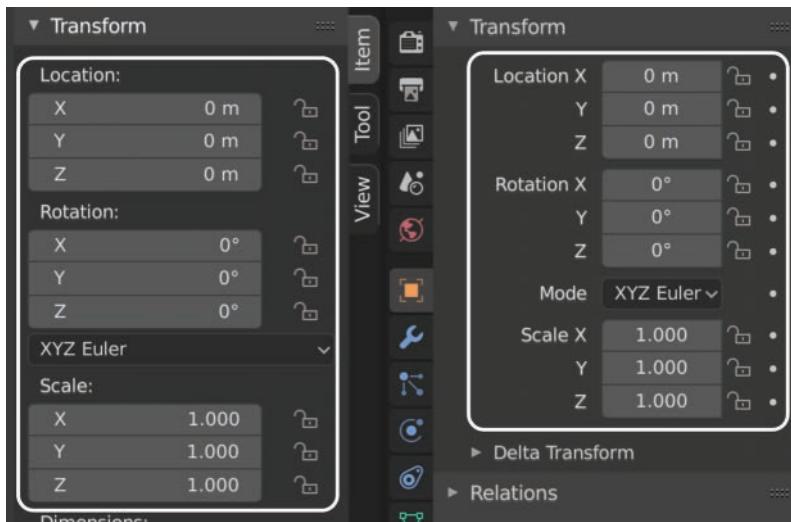


Figure 3.3 On the left side, you can see the Item tab of the 3D Viewport's Sidebar.

On the right side, you can see the Object tab of the Properties Editor.

Both Transform panels can be used to input values to transform objects.

In any of those panels, you can do either of the following things:

- Click and type a specific number in the input field.
- Click the arrows on the sides of the input field to increase or decrease the number.
- Click and slide left and right to increase or decrease the number. Hold **Shift** while sliding to change the number with more precision. Hold **Ctrl** while sliding to change in increments. Hold **Shift+Ctrl** while sliding to change in smaller increments.
- If you change a value in one of those parameters, the change will affect only the active selection. Hold **Alt** while you change a value to affect the entire selection; this command essentially expands the changes from the active selection to the rest of the selected objects where applicable. Click and drag up and down to select several adjacent fields (works only when those fields are grouped together) and then release to write a number that will be entered simultaneously in all of those fields, or drag left and right to use the sliding options in all of the selected fields at the same time. For example, if you wanted to scale an object in all axes, you could click and drag from the X scale field toward the Z scale field, release, type **2** on your keyboard, and press **Enter** to input the value of 2 in the X, Y, and Z scale axes in a single action.

Arranging Objects in Your Scene

Now that you know how to transform objects, you can make your floor bigger and sit the monkey head on it (see Figure 3.4), as follows:

1. Right-click to select the plane, press **S** to scale, enter **5** on your keyboard to make the plane 5 times bigger, and press **Enter** to confirm. (Or use the manipulators if you feel more comfortable with them.)
2. Select the monkey head, moving and rotating it until it looks as though it's sitting on the floor. As a recommendation, you can switch the 3D Viewport to a side view to see what's going on more clearly and transform the head there by pressing **G** and **R**. Keep in mind that if you're in a side view and rotate using **R**, the object will rotate on the X-axis.

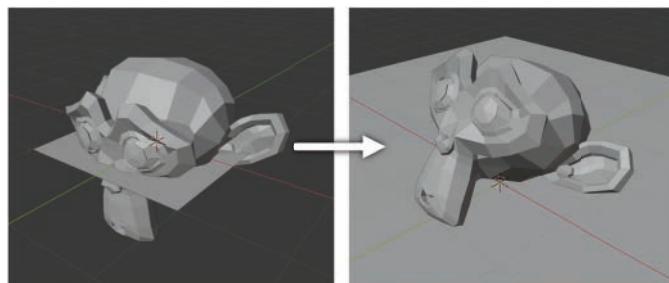


Figure 3.4 The scene before and after the transforms have been performed

Keep in mind that you can do the same thing using any of the methods for transforming objects shown in the previous section, even though throughout the book, I'll use keyboard shortcuts in the explanations to help you use and memorize them.

Naming Objects and Using Datablocks

Before proceeding, you need to learn how to rename objects. This skill will come in handy when you're working in complex scenes and want to recognize objects by their names. Otherwise, you'll find yourself lost in a sea of objects called Plane.001, Sphere.028, and similar generic names.

If a Blender scene were a wall made of bricks, each brick would be a datablock. Every object in Blender has a datablock inside that represents its contents: meshes, materials, textures, lights, curves, and so on. Datablocks can be named and used in the ways discussed in the following section.

Renaming Objects

You have several ways of renaming an object:

- Locate the object in the Outliner. Right-click its name and choose ID Data, Rename within the contextual menu. Alternatively, you can double-click the name, type the new name, and press **Enter** to confirm.
- Press **F2** anywhere in the interface, and a pop-up with the name text field will show up. Press **Ctrl+F2** to open the menu for bulk renaming when you have multiple objects selected.
- In the Properties Editor, go to the Object tab (the one with a yellow cube); type the new name in the text field in the top-left corner; and press **Enter** to confirm.

Managing Datablocks

Datablocks are the most basic Blender components. All the elements you can build—such as objects, meshes, lamps, textures, materials, and armatures (skeletons)—are made of datablocks. Everything in the 3D scene is contained in an object.

Whether you're creating a mesh, a lamp, or a curve, you're creating an object. In Blender, any object has object data inside it, so the object itself acts as a kind of container for the data and stores information about its location, rotation, scaling, modifiers, and so on. Object data defines what's inside an object. If the object data is a mesh, for example, you see a mesh with its vertices and faces inside the object. When you access the object data, you can adjust its parameters. If you click the drop-down list of the object data datablock, you can load a different object data into the object. You could load a different mesh into the object's position, for example. Several objects can use the same object data. (These objects are called *instances* or *linked duplicates*.) This means that even if the objects are in different positions in the scene, all of them synchronize their contents, so if you manipulate the mesh vertices in one of them, the others reflect those changes.

Figure 3.5 shows the difference between the Object and Object Data tabs and how to look for an object's name inside the Properties Editor. The image to the right shows that the mesh's name is inside the object's name. In the image, the object data is a mesh; if it were a lamp or a curve, the icon would change accordingly. The Properties Editor always shows information about the selected object, but if you click the Pin icon, the selected object's information is pinned, and even if you select a different object, the Properties Editor keeps displaying the pinned object's information.

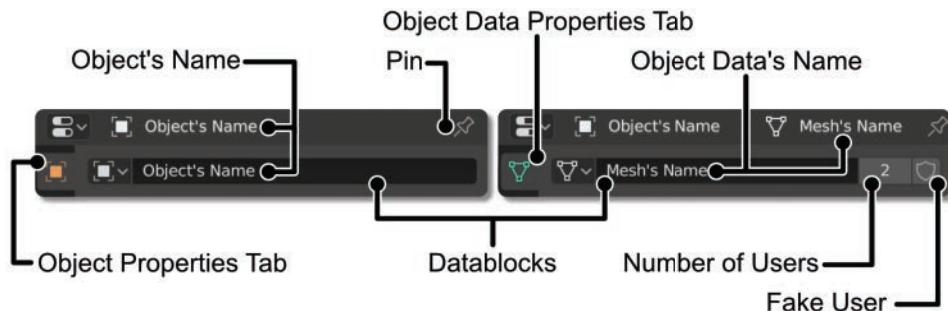


Figure 3.5 Left: Object Properties tab. Right: Object Data Properties tab. You'll find both tabs in the Properties Editor, and in the image, you can see where the names for objects and object data can be found. You can also see how the title of the Properties Editor shows a hierarchy: Object's Name > Object Data's Name, which also serves as an indication of how the object data is contained in the object. The Object and Object Data tabs have been isolated in the image for clarification; you will find those tabs within the rest of the tabs of the Properties Editor.

Duplicates and Instances (Linked Duplicates)

You need to understand the difference between a duplicate and an instance. A *duplicate* is a new object created from an existing one so that it looks the same as the original but is independent, and no link exists between the new one and the original. An *instance* (or as Blender calls it, *linked duplicate*) is also a new object; it can be in a different position, but its content (object data) is directly linked to the original, so if you change the object data in an object, the change also affects all its instances.

When you duplicate an object (**Shift+D**), some Object Data is duplicated with it, and other object data is instanced. You can define the default behavior on the Editing tab of User Preferences. If you duplicate an object, for example, by default Blender duplicates the mesh data contained in it, but it uses the same material data, so both objects use the same material datablock.

On the other hand, instancing (**Alt+D**) duplicates only the object; the rest of the object data it carries inside is linked and synchronized with the original object. An alternative way to instance a mesh (or any other datablock) is to go to the Properties Editor's object data tab and select a different mesh from the drop-down list in its datablock.

To the right of some datablock names, you find a button with a Shield icon as well as a number. The number indicates the number of users that the datablock has. In Figure 3.5, the mesh datablock has two users, which means that two different objects are using that mesh data (they're instances). If you want to turn an instance into an independent, unique datablock, just click the number. Blender creates a duplicate and indicates a single user for the new one.

Blender purges all datablocks with zero users when the file is closed to not accumulate unnecessary data, so if you're not careful, you can lose that great material you created but weren't using. That's why the Shield button next to datablocks exists; it creates a fake user of that datablock. Even if you're not using the datablock in the scene, that datablock will have a [fake] user, which prevents the datablock from being deleted when you quit Blender. Datablocks that have zero users are called *orphan data*.

Caution

If you want to make sure that you keep a datablock in the file when you quit Blender, even if it's not being used (such as a material), click the Shield button next to the datablock's name to make Blender know that you want to keep that datablock.

Keep in mind that you usually work with the names of objects. Most of the time, you don't need to access the names of object data like meshes inside objects, so if you are running low on time, you can generally skip object data naming.

Naming Your Scene's Objects

After you understand what datablocks are and how to rename objects, you can name the objects in your scene accordingly. (You might name the plane Floor, for example.) Sometimes, you have to select a datablock's name from a list, so naming objects and datablocks intuitively will help you find the one you're looking for.

Tip

When you have lots of objects in a scene, it can be difficult to select a specific one, as others may be in the way. If you click the objects in the 3D Viewport several times, the selection jumps between the objects behind the mouse cursor, and if you press **Alt+LMB**, Blender displays a list of objects behind the mouse cursor, so you can select the one you need. This feature is useful only when your objects are named intuitively, of course.

Using Interaction Modes

Blender provides different ways to modify objects in your scene (such as modeling, texturing, sculpting, and posing), called *interaction modes*. By default, when you work in Object Mode, you are able to move, rotate, and scale; Object Mode essentially allows you to place objects in a scene. Probably one of the most useful modes is Edit Mode, which you use to edit object data. For example, you would use Edit Mode to model a mesh; access its vertices, edges, and faces; and change its shape.

You can find the Interaction Mode menu on the 3D Viewport's header (see Figure 3.4); the options it displays depend on the type of object you have selected. For now, I focus on the Object and Edit modes. You'll learn about the other modes throughout the rest of the book.

You use Object Mode to create and place things in your scene (even animate them if you aren't using *armatures*, which are Blender skeletons used to animate characters and deform objects). In Edit Mode, you can perform modeling tasks on the mesh. You can quickly switch between these modes without having to access the selector by pressing the **Tab** key on your keyboard.

When you select an armature, you use Edit Mode to access the bones inside it and manipulate them. Pose Mode is available as well; it's the mode you'll use when animating a skeleton. (For more information, see Chapter 11, "Character Rigging," and Chapter 12, "Animating Your Character.") If you select a mesh, you have access to modes such as Sculpt, Texture Paint, and Vertex Paint, as shown in Figure 3.6.

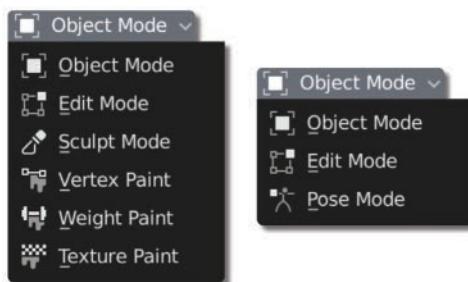


Figure 3.6 The Interaction Mode selector. On the left are the options available when a mesh object is selected; on the right are the options available when an armature is selected.

You can also press **Ctrl+Tab** to launch a pie menu with the available interaction modes for the selected object.

Warning

If you come from previous versions of Blender, or you experience some issues while selecting objects that are in different interaction modes (for example, a mesh object and an armature in Pose Mode), there is a new option that you can try to disable/enable to change the behavior of selections between different types of objects that may have different interaction modes. This option is called Lock Object Modes, which you will find under Edit in Blender's main menu.

As you can see, a lot of options are available, and depending on what you want to do at any point in time, you just have to select the right interaction mode for the actions you want to perform.

Applying Flat or Smooth Surfaces

The monkey head looks weird with the rough edges and polygons that currently compose its shape. This look is useful for some things, but for objects that should look more organic, you may prefer to have a smooth surface. This option changes the surface's appearance but doesn't add any geometry. You have several ways to make a surface look smooth in Blender:

- Select the object you want to smooth. Press **RMB** and choose the Shade Smooth option from the contextual menu (choose Shade Flat for the opposite result).
- Select the object. Click the Object menu of the 3D Viewport's header and select the Shade Smooth option.
- In Edit Mode, select the faces you want to shade with the smooth or flat method, press **RMB**, and select Shade Smooth or Shade Flat from the contextual menu. Alternatively, you'll also find those options within the Face menu in the 3D Viewport's header.

Figure 3.7 shows where these options are in Blender's interface.



Figure 3.7 A comparison of flat and smooth surfaces and the menus in which you can find these options. On the left, you can see the Object menu from the 3D Viewport's header. On the right, you can see the object's right-click contextual menu.

Working with Modifiers

Even though you used smooth shading in the mesh, the object still doesn't look just right, as it has very low polygonal resolution. You could use a Subdivision Surface modifier to add more detail to the surface and smooth it out (at the cost of adding more polygons to the object). A *modifier* is an element you can add to an object to alter it, such as a deformation, the generation of geometry, or the reduction of existing geometry. Modifiers won't affect the original mesh and adapt automatically to the changes you perform in the original mesh, which gives you a lot of flexibility, and you can turn modifiers on and off when you want. You should be careful, though, as adding too many modifiers may cause your Blender scene to operate slowly.

Adding Modifiers

Clicking the wrench icon in the Properties Editor opens the Modifiers tab, where you can add modifiers (see Figure 3.8). When you click the Add Modifier button, a pop-up menu displays every modifier you can add to the active object. (Not all the modifiers are available for every type of object.) The modifiers are listed in columns based on their functions: Modify, Generate, Deform, or Simulate. Left-click a modifier in the list to add it to the active object.

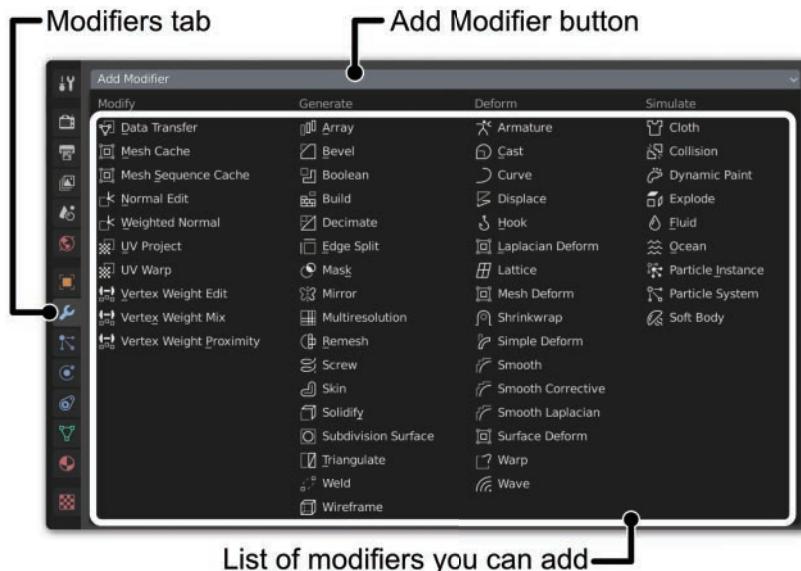


Figure 3.8 On the Properties Editor's Modifiers tab, you can add modifiers to the active object.

When you add a modifier, a block is added to the modifier stack, which works similarly to layers; if you keep adding modifiers, they add their effects to the previous modifiers. Keep in mind that the modifier stack works in the opposite order of layers in other software, such as Adobe Photoshop. In Blender, the last modifier you add is at the bottom of the stack, and its effect alters the effects of the modifiers above it in the list. The order of the modifiers is crucial in defining the resulting effects that the modifiers have on the object.

If you model one side of a mesh, for example, you can assign a Mirror modifier to generate the other half and then assign a Subdivision Surface modifier to smooth the result. The Subdivision Surface modifier should be at the bottom of the list; otherwise, the object is smoothed before being mirrored, and a seam may appear visible in the middle.

Copying Modifiers to Other Objects

When you assign a modifier, it affects only the active object, which is the last selected object (even if you have 20 selected objects). If you want that modifier to be applied to every object in the selection, you have two ways to do this:

- Press **Ctrl+L** to access a menu of linking options. In this menu, you'll find an option that lets you copy modifiers or materials from the active object to the rest of the selection.
- Activate the Copy Attributes add-on in User Preferences (this add-on comes bundled with Blender) and press **Ctrl+C** to access a special menu to copy attributes from the active object to the rest of the selected objects. You'll find the modifiers within those attributes as well.

It's important to know that both **Ctrl+L** and Copy Attributes add-on's Copy Modifiers option will overwrite the existing modifiers that objects in the selection have. If you want to keep those, using the Copy Selected Modifiers option within the Copy Attributes add-on's menu will add those modifiers from the active object to the existing modifiers in the rest of the selected objects.

Adding a Subdivision Surface Modifier to Your Object

The Subdivision Surface modifier is one of the most common modifiers used in models, because it allows you to increase the details and smoothness of a low-resolution model interactively. You can change the number of subdivisions at any time to display a smoother surface. The modifier basically divides each polygon and smooths the result. As a rule of thumb, when you apply this modifier, the number of faces in your model is multiplied by 4 for each subdivision you apply; therefore, be mindful of the polygon count when setting high subdivision values. You can use this modifier to smooth your monkey-head object, as shown in Figure 3.9.

When you add a modifier, you get a panel in the modifier stack with options that are specific to the modifier you picked. Here are the main options you'll find with a Subdivision Surface modifier:

- In the top row of the panel that encloses the modifier, you can expand/collapse the modifier (by clicking the little triangle to the left), rename it (give the modifier an intuitive name when you have a lot of modifiers added to an object), and define the contexts in which this modifier should be visible. Two buttons with arrows pointing up and down allow you to change the order of the modifiers when you have more than one modifier in the stack. Clicking the X button deletes the modifier.
- Next, you find two buttons: Apply and Copy. Apply transfers the effect of the modifier to the mesh itself. It deletes the modifier, but its effect on the mesh is permanent. Copy duplicates the modifier.
- In the Subdivisions section are two fields that let you define the number of subdivisions that the modifier will perform in the 3D Viewport and in the render. This option is very useful because when you're in the 3D Viewport, you usually want to save resources to ensure that this view is responsive, but in a render, you want a high-quality result. You can set a low number of subdivisions for the 3D Viewport and a higher number for the render.

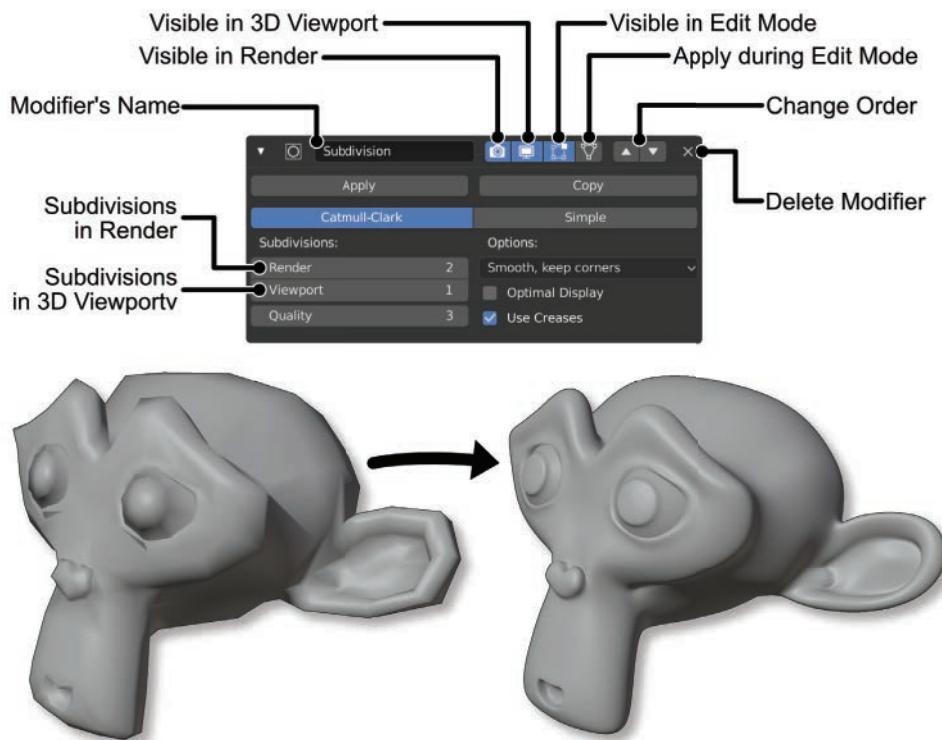


Figure 3.9 Subdivision Surface Modifier options and the monkey head before and after applying the Subdivision Surface modifier

Tip

Subdivision Surface is a widely used modifier, so Blender comes with a keyboard shortcut that lets you add and control it. Press **Ctrl+1** (you must have a mesh selected in Object Mode for this to work) to add a Subdivision Surface modifier with one subdivision. The number you press together with **Ctrl** defines the number of subdivisions shown in the 3D Viewport (doesn't change the subdivision level while rendering). If the object already has a Subdivision Surface modifier added, use this shortcut to change its number of subdivisions. Additionally, if the object has multiple Subdivision Surface modifiers, the shortcut will change the number of subdivisions of the first Subdivision Surface modifier in the stack.

Using Workbench, EEVEE, and Cycles

Blender provides different methods to display and render images, each of them with their uses, pros, and cons. Let's talk about them:

- **Workbench:** This engine runs Blender's 3D Viewport while you're working in Wireframe and Solid viewport shading modes. It's basic, but it has some level of control over how things look. It's lightweight and simple, perfect for general work like modeling, rigging, and animation.
- **EEVEE:** EEVEE has been one of the greatest additions to Blender lately, and it's a real-time render engine, using technologies similar to those used in videogame engines. It can get good-quality results very fast (as long as you have a computer that supports it and can run it with a good performance), although it's based on tricks and sacrifices many calculations to accelerate the render time. It's good for rendering animations that don't require high levels of realism and for previewing scenes and materials that would be rendered with Cycles later. EEVEE is used when you choose the Material Preview viewport shading mode, and it shows at its best in Rendered viewport shading mode (when EEVEE is selected as the active render engine).
- **Cycles:** This realistic renderer is included in Blender. It provides high quality and realism, but it's also much slower than EEVEE, as it doesn't use tricks or sacrifice complex calculations to be faster: it performs all the calculations necessary to achieve the best result. If EEVEE could be compared with what you see in videogames, Cycles would be a render engine used for movies or general video, where render speed is not as relevant as image quality.

You can change the render engine that you want to use in the Render tab within the Properties Editor (see Figure 3.10). When you render the final image, as explained at the end of this chapter, the active render engine is the one that will be used.

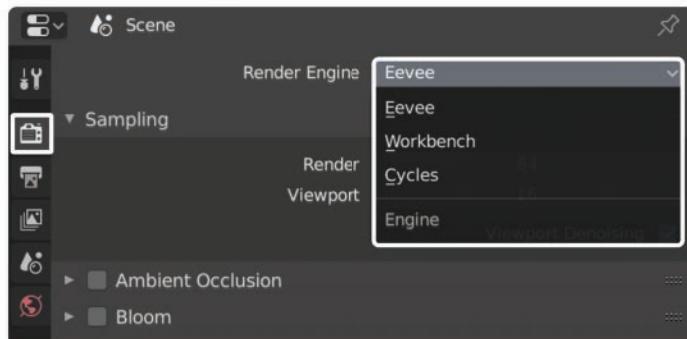


Figure 3.10 You can change the render engine between Workbench, EEVEE, and Cycles from the drop-down menu within the Render tab of the Properties Editor.

Materials Compatibility

Workbench doesn't use materials, but EEVEE and Cycles have been designed so the materials are as compatible as possible between them. Of course, there are certain things that may work only on one of the engines or look different, given that they use different technologies, but in general they are surprisingly compatible.

This makes it possible to create materials using EEVEE (allowing for fast previews) and then render them with Cycles with minimal or no adjustments.

Some advanced rendering effects, such as emissive materials (that emit light from their surface), refractions, and Subsurface Scattering, will work only in Cycles or with certain limitations in EEVEE.

Understanding Viewport Shading

Viewport shading defines how objects are visualized in the 3D Viewport, and it's important to understand how they work before you start adding materials to the scene.

While working, you can change the viewport shading mode in the 3D Viewport to show Wireframe, Solid, Material Preview, or Rendered mode (see Figure 3.11), and different engines will be activated for different modes, although Rendered viewport shading mode will always display a result similar to the final render, but interactively and in real time, using the selected active render engine. Depending on what render engine you use, options for viewport shading will change:

- **Workbench Engine:** Material Preview viewport shading will not be available, as Workbench doesn't use materials (although colors and other properties can be added to objects); it's meant only for general work and simple screenshots. Wireframe, Solid, and Rendered viewport shading modes all use Workbench Engine. A render taken from this engine would be like a screenshot, useful for quick playblasts to check your animations.

- **EEVEE Engine:** Wireframe and Solid viewport shading modes will use Workbench, but Material Preview and Rendered viewport shading modes will use EEVEE.
- **Cycles Engine:** Wireframe and Solid viewport shading modes will use Workbench, Material Preview will use EEVEE, and Rendered viewport shading mode will use Cycles.

Whichever render engine you're using, you'll see their options for rendering in the Render tab of the Properties Editor.

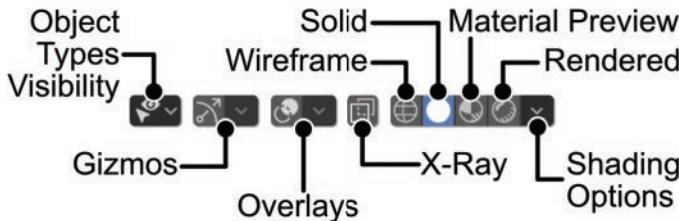


Figure 3.11 Viewport Shading and visibility options. You can find them in the right corner of the 3D Viewport's header.

You can also find interesting options for how the image is shown by clicking the Shading arrow next to the 3D Viewport's shading mode selector. For example, Material Preview Mode will let you change the environment to see how the materials behave under different lightings, and Solid Mode will let you choose different options for how objects are visualized.

Real-Time Preview Rendering

In Blender, you have options to see a rendered preview in the 3D Viewport while you work and adjust parameters using the Rendered viewport shading. It's very useful to see what's going on in the scene and how the shadows and materials behave as you arrange them.

Rendered viewport shading mode, in this case, is not actually real time; it just means that Blender is performing the render interactively, and you can change things in the scene as it's rendered. The speed of the render depends, of course, on your computer's processing speed. For Cycles, a powerful CPU or GPU is recommended for faster performance; EEVEE relies mainly on GPU.

While using Rendered viewport shading mode, you can choose to show or hide manipulators, object outlines, and so on from the Overlays menu on the 3D Viewport's header.

Switching Viewport Shading Modes

To switch viewport shading modes, you can simply click the button for that mode in the 3D Viewport's header (refer to Figure 3.11).

Alternatively, and to speed things up, you can use keyboard shortcuts:

- Press **Z** to launch the viewport shading pie menu, and choose one of the options.
- Press **Shift+Z** to switch between the current viewport shading mode and Wireframe viewport shading mode.

Managing Materials

Materials define how an object looks, such as what its color is, whether the object is dull or shiny, and whether it is reflective or transparent. With materials, you can make an object look like glass, metal, plastic, or wood. In the end, both materials and lighting define how your objects look. In this section, you see how to add materials to your objects by using both Blender Render and Cycles.

On the Materials tab (the red sphere with a checkered pattern icon) of the Properties Editor, you can add new materials or select existing ones from the drop-down list shown in Figure 3.12. A single object can have multiple materials, and these materials appear in the list at the top of the material properties. You can add and remove new slots for materials by clicking the + and - buttons on the right side of the list, and you can assign each of those materials to a selection of faces when you're in Edit Mode.

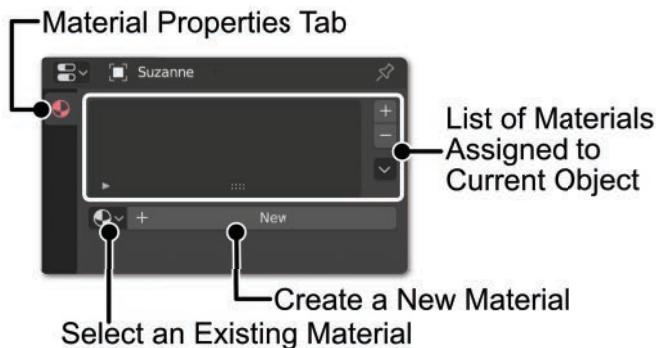


Figure 3.12 Use this menu on the Material Properties tab of the Properties Editor to add materials. Keep in mind that the Material tab has been isolated for clarification; in the Properties Editor, you'll find it within the rest of the tabs.

Adding and Adjusting Materials

Advanced materials require the use of node trees in the Shader Editor, but don't worry; for now, I'll keep things simple. Inside a material, you'll find the Surface panel, which includes various types of surface shaders, such as

- **Diffuse:** Creates a basic material with only color—no shine, reflection, or other special properties.
- **Glossy:** Makes the material reflective and shiny.
- **Emission:** Makes the material emit light into the scene.
- **Transparent:** Lets light pass through the material.
- **Glass:** Simulates a glass surface.
- **Principled BSDF:** Includes many of the properties provided by the others, so it's a very useful shader type, and conveniently, it's the one you get by default when you create a new material.
- **Mix:** Mixes two shaders to achieve a more elaborate effect.

Many surface shaders are available; these are just some of the main ones. Each of the shaders has different parameters to control how light affects that shader, such as color and roughness. Accessing nodes makes it easier to create complex and custom materials by combining the effects of some of the shaders and using textures. (See Chapter 10, “Materials and Shaders.”)

Remember that these materials are almost completely compatible between EEVEE and Cycles, so for simple cases, the same settings will work in both engines (even though there may be some differences in the result, given the different approaches that both engines used to calculate the final image).

Tip

When working with materials, it's recommended that you work with Material Preview or Rendered viewport shading in EEVEE, as they will let you see how the material looks in real time.

To add materials with different colors to your scene, you just need to select each object and create a material from the Material Properties tab in the Properties Editor. Follow this procedure:

1. Select the monkey head.
2. Go to the Material Properties tab of the Properties Editor.
3. Add a new material by pressing the New button and name it properly to make it recognizable.
4. You'll get the options for a Principled BSDF shader. Adjust the base color parameter to choose the color you'd like for the material. Play with other parameters of the material to see how it changes.
5. Repeat the process with a new material for the floor plane.

Turning On the Lights

Now that you have materials set up, it's time to make the scene look more realistic with some light and shadows. Lights are also compatible between EEVEE and Cycles, even though some of their options are different. For basic use, however, there shouldn't be any problem. (Chapter 14, "Lighting, Compositing, and Rendering," provides more information about lighting.)

Light Options

There are different types of lights with different properties, but there are two properties that all of them share and that are compatible in both EEVEE and Cycles: Color and Power. Color, as the name implies, will change the color of the light, and Power will increase or decrease its intensity (it's measured in Watts).

To access the light properties, select a light in your scene, and the Object Data tab of the Properties Editor should change into a teal light bulb. From that tab, you will find options to change the light type and its properties.

Remember that using EEVEE in Rendered viewport shading mode, you should be able to preview the lights' effect on your scene as you adjust them.

Adding Lights to Your Scene

Follow these steps to create a basic lighting scheme for your scene (and remember that you can access the menu for adding new objects to the scene by pressing **Shift+A**):

1. Select the light in your scene (or create a new one if you don't have a light yet).
2. Duplicate the light, and place it on the other side of the scene to fill the shadow areas.
3. Adjust the Color and Power of your lights so that the one on the right is brighter (this will be your main light), while the one on the left is dimmer and a different color (this will act as a fill light, to prevent the area in the shadow from being completely dark).

Moving the Camera in Your Scene

You need a camera in your scene, of course, so that Blender knows the point of view to look from when it takes the final render. Follow these steps to position the camera:

1. Select the camera in your scene or create a new one (**Shift+A**) if you deleted it previously.
2. Place the camera so that it focuses on the monkey head from a point of view that appeals to you. You can divide the interface into two 3D Viewports. In one of those views, you can look through the camera (**NumPad 0**), and in the other

view, you can adjust the placement of the camera. Alternatively, you can use Walk Mode or Fly mode (**Shift +**) to position and orient the camera while you're in Camera View.

Figure 3.13 shows what your scene should look like at this point of the process.

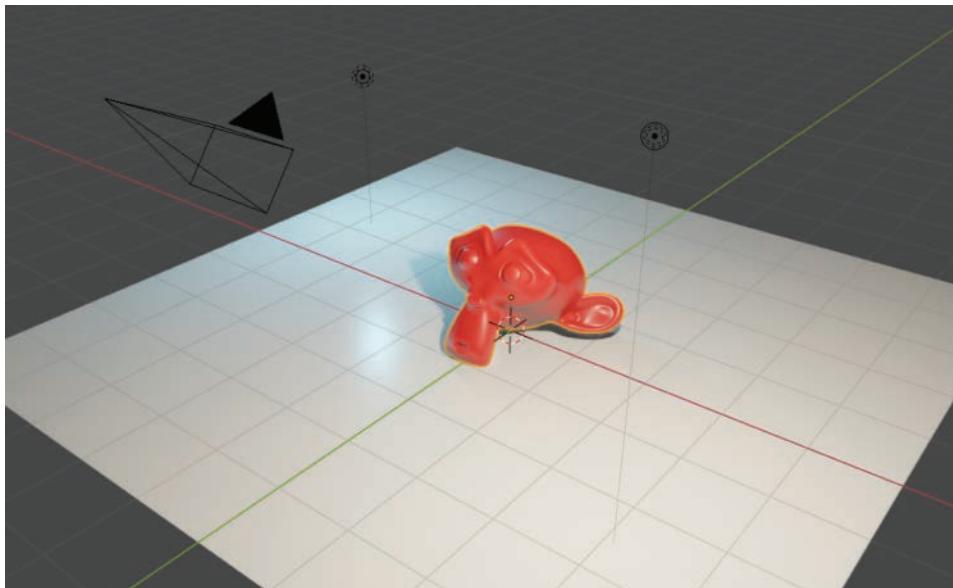


Figure 3.13 At this point, your scene should look something like this image. The monkey head is on the floor, the camera is pointing at it, and two lights illuminate everything in the scene.

Rendering

Rendering is the process that converts your 3D scene to a 2D image or animation. During this process, Blender calculates the properties of materials and lights in the scene to apply shadows, reflections, refractions, and so on—everything you need to build your cool final result and turn it into an image or a video.

Whether you use EEVEE or Cycles, you would access options for rendering within the Render tab in the Properties Editor.

Select your desired render engine. For such a simple scene, not many changes should be made, but here are a couple of things you can try:

- **For EEVEE:** If you want the surfaces to reflect other objects, you can enable Screen Space Reflections in the Render Properties tab.
- **For Cycles:** Cycles calculates light paths and bounces throughout the scene. This generally means that the more calculations (and more render time), the cleaner

the result. If you have a low samples count, you'll have noise in the resulting render, as the pixels still don't have enough information to display the complete result. You can increase the render samples amount in the Render Properties tab of the Properties Editor to get a cleaner image.

Enabling GPU Rendering with Cycles

GPUs can be much faster than CPUs for rendering with engines such as Cycles. If you want to use your GPU to render the scene, follow these steps:

1. Open User Preferences.
2. On the System tab, you'll find a panel called Cycles Render Devices. Depending on your graphics card, some of the options within that panel will be available. Make sure to select one of them and enable the GPU or GPUs that you want to use.
3. Go back to your scene, and within the Render Properties, you'll find the Device menu (right under the render engine selector). Select GPU.
4. Finally, go to the Performance panel in the Render Properties, and set the Tile Size to a number such as 64, 128, 256, 512... try rendering the scene with different values in the Tile Size, as depending on your GPU, you will get better results with different sizes.

Tile Size defines the size of the squared parts of the image that the CPU or the GPU can render at a time. Generally, CPUs work better with small values (16, 32, 64, and so on), while GPUs work better with bigger values (128, 256, 512, and so on). Setting the right value for your hardware can help you get faster results, but keep in mind that in general, rendering with Cycles requires high-performing equipment, so anything that isn't very powerful may be very slow, regardless of the settings.

In User Preferences, you can also choose if you want to use CPU and GPU to render together (you just have to enable both the CPU and GPU). If you choose this option, it's recommended to use little tile sizes so that the CPU doesn't lag behind GPU while rendering.

Now you're ready to launch the final render. But first, let's learn how to save the .blend file.

Saving and Loading Your .blend File

Now you're at a good point to save your file. Rendering can take some time, and something can go wrong in the meantime (such as power failures or software crashes) that could cause you to lose your work. That's why it's recommended that you save your file often.

You can save your file by pressing **Ctrl+S**. If you're saving a file for the first time, Blender displays a menu where you can select the location where you want to store your file and name the file. If you've saved the file previously, press **Ctrl+S** to overwrite the

previous version. If you press **Shift+Ctrl+S**, Blender displays the Save menu again so that it allows you to create a new version of the file with a different name.

To open a file, press **Ctrl+O**. Blender shows you the folder navigation menu, where you can look for the .blend file you want to open. On the File menu, you can also access the Open Recent option, which shows you a list of the latest files you've worked on so you can open them quickly.

You don't need to use those shortcuts, of course; you can always choose the Save, Save As, Save Copy, and Open options from the File menu.

Save Copy doesn't have an assigned keyboard shortcut and it's a bit unusual, so what does it do? Well, it's similar to Save As except that it saves the current status of the scene in a file, but then you keep working on the original instead of in the new file.

Tip

There's a little trick for saving different versions of a file really fast. Sometimes, you want to save your progress in a new file, so you'll have different files from different parts of the process and can go back to a previous version if necessary. Choose Save As from the File menu (or press **Shift+Ctrl+S**), and press the NumPad + key. Blender automatically adds a number to the filename. If the filename is already numbered, Blender adds 1 to it.

Launching and Saving the Render

Before launching the render, remember to select the desired Render Engine from the Render tab in the Properties Editor. Remember as well that the image format can be set up in the Output tab of the Properties Editor. Then, you can launch it in several ways:

- Press **F12** for a still render.
- Press **Ctrl+F12** to render an animation.
- Select the Render Image or Render Animation options from the Render menu on the main menu at the top left of Blender's interface.

By default, the render will appear in an Image Editor in a new window. You can change this behavior so the render shows up within the main interface: for example, turning the biggest area into an Image Editor and displaying the render in it.

To save the image once it's rendered, you can do it in different ways:

- In the Image Editor where the render is shown, go to the Image menu within the header, and select the option to save the image.
- In the same Image Editor, you can launch the Save menu by pressing **Alt+S** or **Shift+S**.

When you're rendering an animation, images are automatically saved after being rendered, using the format, name, and destination defined in the Output tab of the Properties Editor.

You can press **Esc** after the render to go back to switch to the main interface again. If you chose to display the render within the main interface, the Image Editor will turn into the previous editor type, while if the render is shown in a different window, it will remain open (you can close it).

Figure 3.14 shows the images that result from both engines' renders. Given that the scene is very basic, there are no many differences between EEVEE and Cycles renders, although some subtle differences can be spotted. Cycles, for example, has bounced light that spreads some of the monkey head's red color on the ground and makes the areas in the shadows a bit brighter, making for more realistic lighting. Still, EEVEE got very close with a fraction of the render time. In complex scenes with complex materials, you will find a more noticeable difference, but for now, I just wanted you to try both to see how easy it is to switch between them, given the high compatibility of materials and lights.



Figure 3.14 Resulting renders. Left: EEVEE. Right: Cycles.

Summary

In this chapter, you learned how to create and transform objects; add modifiers, lights and materials; and launch a render. This chapter gave you a lot to process, but I hope that you now know the basics of interacting with your scene. You're ready for the more extensive information in the chapters that follow.

Exercises

1. Create and manipulate a few objects.
2. Add some other modifiers and play with them to see their effects.
3. Try different things with EEVEE and Cycles to get acquainted with the differences between them.
4. Add more lights to the scene and play with materials to change the results.



Beginning a Project

4 Project Overview

5 Character Design

This page intentionally left blank

4

Project Overview

Every project has different steps you need to follow to be successful. The order in which you proceed through the steps to reach the final result can be called the *workflow* or *pipeline*. In this chapter, you learn about the process that you'll follow throughout the rest of this book to create a character from scratch. You gain basic knowledge of how to divide any project into stages and execute it.

Three Stages of a Project

Usually, any project in 3D, graphics, or video goes through three stages: preproduction, production, and postproduction.

Preproduction

Preproduction is everything that happens before the actual production of a project, such as preliminary sketches, ideas, designs, and planning. It's probably the most crucial stage of any project, and a lot of amateur projects fail because of the lack of good pre-production. (Sometimes, people even try to start a project with no preproduction at all, which is usually a bad idea; the project will probably be unsuccessful.)

When you plan and organize what you need to do to complete a project, chances are that you're going to be better prepared for what's to come. If you skip preproduction and jump right into production because you can't wait to see the results, you'll likely encounter problems and issues you didn't anticipate. You'll have to redo a lot of work and lose lots of time. In the worst-case scenario, you'll give up because you'll find too many complications during the process.

Good planning allows you to anticipate any possible problems so you can prevent them. If you run into something you don't know how to do, you can make some quick, basic tests to find a solution *before* you get far enough into the project to discover that it doesn't work.

As a result of this preparation, the actual work during production will be faster, easier, and more focused, as you'll already know how to proceed. Keep in mind that even with a good preproduction stage, you will likely still run into issues. This is the nature of projects (especially complex ones), but at least a lot of those issues will be handled before they become bigger problems. The more preparation, the better.

Preproduction has another important advantage: It can motivate you during the production stage. When you think about everything you'll have to do and then define the process step by step, the work suddenly gets easier because you don't have a *big* project before you; instead, you have a list of small, manageable tasks. Go through this list, keeping track of your progress, and you'll always know what you have done so far, what you still need to do, and what may be missing.

A popular phrase sums up preproduction pretty well: "Think twice, work half." A great result doesn't always come from working harder, but from working more efficiently. You need to think of smart ways to work. Usually, you discover efficient methods only after you've done something wrong, but that's when you learn and gain valuable experience!

Production

When you have everything planned for a project, it's time to start the actual job, which is *production*. When you're making a movie, for example, production is the stage of the project in which the sets are built and the scenes are filmed with the actors and props in place as planned during preproduction. Thorough preproduction helps you complete production more easily and in a more straightforward manner.

Production is probably the hardest stage of a project because it's the point of no return. When production is complete, it's difficult to change things. Suppose that you're building a house. During preproduction, it's easy to change the design and plans of a house by using a computer or an architectural drawing, but it's really difficult and time-consuming to make changes after the walls are in place!

Preproduction is crucial because it helps you make sure that you're not going to make mistakes while you're developing the final product. Production is difficult enough, involving a lot of challenges. It's impossible to predict every possible problem until you're actually making the product, so any preparations you can make to smooth the process help a lot.

Postproduction

Postproduction is everything that happens between production and the final result. It's like putting the finishing touches on a new house, such as painting the walls and adding interior decoration. In a movie project, postproduction is the stage at which you add final visual effects and retouch what was filmed during production.

Depending on the project, postproduction can be simple or complex, and it can involve minor details or really important things. Postproduction is when you decide the look of the finished project.

Suppose that you film two actors having a conversation. During postproduction, you can color-correct the scene, switch it from day to night, change what will be seen through a window, blur things, zoom in, or even add a new character. The possibilities are endless, and they define what people will see when you release your image, video, film, or whatever your project may be.

Defining the Stages

Now that you know the three main stages of a project, it's important to know where each stage ends and the next one starts, as each project is different. This section presents some examples that illustrate these differences.

A Film Without Visual Effects

Today, almost every film has some visual effects, but consider a film that doesn't have any visual effects, which will help you understand the basic process of film production.

Note

Remember that visual effects are not just explosions, spaceships, aliens, and monsters. Many visual effects (commonly referred to as *invisible effects*) are subtle, and you may not notice them while you're watching a film. Set extensions, background replacements, wire deletion, and set cleaning, for example, are present in almost every movie made today, and they are also visual effects.

During preproduction, the filmmakers create the film's script and decide what will be the climax moments (and maybe even film them on their phones to test whether the scenes really work). Every film goes through *storyboarding*: the process of making quick drawings to define where the cameras will be placed and what will happen during each shot. Storyboarding helps the production team plan each shot, see what they'll need on set, know what type of lenses to use in the camera, and identify where the actors will be located and how they'll move through the set. Then the filmmakers search for the locations where they're going to film the scenes. They also have to create the costumes the actors will wear and all the props the actors will have to interact with. Then the filmmakers cast the actors and all the extras who will appear in the film. Finally, the filmmakers assemble the team that will film the movie and manage all the equipment, build the sets, and so on. Usually, composers begin to develop the music at this stage so that a rough edit of the film can be made by following the storyboard and the timing of each shot can be defined.

Everything is ready now, so production begins. At this point, the actors are ready to do their jobs, and the team members know what they need to do on each shot and what has to appear on camera. Production usually is not a long process because all aspects of the project were organized during preproduction to make the production stage (the most expensive) as short as possible. When production is complete, the movie has been filmed according to the decisions made during preproduction, more often than not including changes that had to be added during production because, as I established earlier, nothing ever goes completely according to plan.

When filming is complete, postproduction can begin. The film must be edited at this point to put all the shots together in the order they should be to tell the story, perhaps by using some color correction to make a scene look more vivid, warm, or cold,

depending on the feeling the director wants each scene to convey. Perhaps the director decides that a shot would work better if the main actor's face were closer, so the video editor zooms in a little. Postproduction is the stage in which the last retouches are added to the film, the complete soundtrack and all the sound effects are included, and the final result is achieved.

A Visual-Effects Film

In this section, I analyze the differences between the preceding example and a film with complex visual effects.

During preproduction, the production team needs to think about what visual effects to use, how they're going to be filmed, and what is required to create them. Generally, the visual-effects team works closely with the filmmakers during preproduction to see what's possible, what's not possible, and how the effects will be achieved. (Usually, almost anything is possible with visual effects, but the effects may be too expensive for a particular film's budget.)

During production, the visual-effects team may need to film some shots in special ways, using green screens or markers or puppets that the actors can interact with so that later, the team can add an animated character to the scene. Lighting in the sets has to be measured and recorded so that the team can simulate it later in the 3D world to make it match the lighting on the real set. Effects such as explosions may need to be filmed separately so that they can be integrated with the footage of the actors.

After the movie has been filmed, it's time to begin the postproduction stage, but because this film involves visual effects, the line between production and postproduction tends to blur, and sometimes, there is an overlap between these stages. The visual-effects artists probably work on some shots before production begins so that during filming, all the elements that comprise a scene fit together seamlessly.

The visual-effects team has its own preproduction, production, and postproduction stages. Team members plan the specific effects and determine how a shot will be accomplished. Then they proceed to production and work to create the elements of the visual effects. Finally, they combine those elements, adjusting colors, shapes, textures, and so on.

An Animated Film

The stages of an animated film are even more difficult to distinguish, as the entire film is computer-generated. The line between production and postproduction is not so clear.

During preproduction, all aspects of the film are planned and designed as usual, but production and postproduction tend to overlap because every aspect of these stages happens in 3D software. Usually, it's easier to divide the stages, with production creating the action (developing characters, sets, and animation), and postproduction creating the effects (water, splashes, particles, cloth, dust, smoke, fire, explosions, and other simulations). Then final compositing brings these diverse elements together.

A Photograph

Yes, even something as simple as a photograph people take on their smartphones can be divided into three production stages. Even if the person taking the photo is not conscious of the fact, he is performing the three stages of production.

First, the photographer completes the preproduction stage by thinking about what they're going to shoot and where. During production, they must choose the framing, pose the subject (if there is one), and take the photo. Then they can do some post-production work, such as adding filters as an aging effect to the photo, increasing its contrast, or even changing it to black and white.

Making a Character-Creation Plan

This section introduces the process you'll follow throughout the remainder of this book to create a complete 3D animated character.

Character Preproduction

Character creation starts, of course, with character design:

- **Character idea:** Design starts in your mind. Before designing a character, you must imagine it, think about its story and personality, the world it lives in, and so on.
- **Designing:** Make some drawings to define what the character will look like from different points of view, what clothes it will wear, and what features suggest its personality.

Character Production

This stage can be rather complex and extensive because it's the main part of the process that takes you from the design to the completed character.

- **Modeling:** Model the 3D character in Blender, following the design you created in the preproduction stage.
- **Unwrapping:** Unfold the 3D model into a 2D mesh internally (UVs) so that you can project a 2D image texture onto it.
- **Texturing:** Paint the textures for the 3D model's surface, such as clothing textures, skin, and hair color.
- **Shading:** Take texturing a step further by creating materials that define the surface properties of your character, such as how reflective or shiny they are and whether they are rough or smooth.
- **Rigging:** Add a skeleton to your character to define how it will deform and control the character.

- **Animating:** Pose the character, using keyframes at different times of an animation to make the character perform an action such as walking or running.
- **Video recording:** Record a video in which you'll place your character later.

Project Postproduction

When the character is finished, you still have some work to do to make it look pleasing or to put it in a scene.

- **Camera tracking:** Analyze the real video and mimic the camera movement with a 3D camera so that when you add 3D objects to the live video, the movements fit together.
- **Lighting:** Add lighting to your scene to make the lights and shadows fit the video you recorded during the production phase. The addition of lighting is usually part of the production stage, but because this project's main goal is the creation of a character, lighting is considered to be part of postproduction.
- **Rendering:** Convert the 3D scene in an image sequence, calculating lights, shadows, reflections, and so on.
- **Compositing:** Combine the video and the 3D objects, and make any necessary adjustments so that everything fits together and the final result looks good.

Summary

After reading this chapter, you have an idea of the process you're going to go through to create your own animated character, and you understand the three main stages of any project. Preproduction is especially important—a fact that you have to keep in mind for future projects. A lot of people fail even after significant planning and preparation, so imagine the probability of failing if you don't make thoughtful preparations and design choices in preproduction. Almost every professional 3D artist has gone through failure because of bad preparation and understands the importance of preparing and organizing projects. Learn from their experiences.

In the next chapter, you start working on the actual project.

Exercises

1. Think of a movie you recently watched, and imagine how it was divided into pre-production, production, and postproduction stages.
2. Have you ever had a project fail? Analyze where you failed and think about how you'd do the project differently, using the three stages discussed in this chapter.

5

Character Design

The first stage of your character-creation project is, of course, preproduction (discussed in Chapter 4, “Project Overview”). There are a lot of ways to design a character, and each artist uses a different method. In this chapter, you explore a common approach that you can later adapt according to your skills. I also note some other methods so that you can try them if they sound interesting to you.

You can use any medium to design your character, such as paper or software. In this chapter, the whole process is carried out with digital painting software and a pen tablet, but you can use any other painting medium.

Character Description

Before you start drawing or imagining what clothes a character will wear, how big its eyes will be, or what color to make its hair, you need to have at least a basic idea of who your character is. The design ultimately represents the character’s personality, so understanding how that character thinks and acts will help you represent it. It’s easier to know the type of clothes the character will wear if you know its profession. A knight or a soldier will wear armor or battle gear, for example, but it would make no sense at all for an accountant to wear armor or carry a weapon, no matter how cool it looks!

Also, the attitude of a character can define its looks. A dynamic character would be fast-moving, whereas a sad character would slouch and move slowly. A happy character would have a big smile and large, expressive eyes, whereas a sad character would have small, tearful eyes and a frown.

Developing a good description of your character helps you understand how it will behave. As a result, you can get into its head, imagining what it will wear and how it will walk, talk, smile, laugh, and cry. Different people can react in different ways to the same situation, and this also applies to the characters you create. Establishing personality traits helps you understand your character and define its lifestyle, for example.

You don’t need to develop the character’s complete history or personality if you don’t want to go that far. A simple description of its personality and appearance may be enough.

In the following sections, you look at a brief description of the character you’ll build in this book. The character is called Jim. From now on, you need to have a personal relationship with Jim so you can understand how he thinks and acts. Consider Jim to be a living being, not a thing.

Note

Learning about body language helps you determine how a character with a specific attitude would look, including the clothes it would wear. And the way the character looks will make the viewer recognize what its attitude is like. Reading a specialized book on the subject is highly recommended if you want to design great characters. I encourage you to take a look at *What Every BODY Is Saying: An Ex-FBI Agent's Guide to Speed-Reading People*, by Joe Navarro and Marlin Karlins (William Morrow Paperbacks, 2008).

Personality

Following is a description of Jim. The various aspects of his personality are related, and some of them are influenced by the others. A lazy person would never want to be an adventurer, for example, and if he isn't happy and doesn't love challenges, there's no way he would go into the unknown to discover new things. He needs lots of motivation to do that. The lesson here is that the aspects of a character's personality must be coherent (unless for some reason, the story requires it to be otherwise).

Jim is a 15-year-old kid. He is very active and participates in a lot of sports with his friends. He's always happy and positive, he likes challenges, and his dream is to become an adventurer and discover new things. His ambitions are motivated by his never-ending curiosity, and as a result of his curiosity, he pays a lot of attention to details. He also likes to be different from the rest of the kids his age. And...he manages to get himself into trouble quite often.

Context

You have a basic understanding of Jim's personality, but another very important thing defines a character: the story's context, or the world in which the character dwells. Check it out:

It's 2512, and humans have populated quite a few planets. Space exploration is a major topic in the news, and astronauts are treated as heroes. Cars fly and don't produce pollution. Robots are everywhere, making human life easier, and some even develop relationships with humans. A downside of this setting is that it's difficult for a person to stand out from the rest of the population. Everyone wears the same kinds of clothes, everyone owns similar cars, and everyone lives in buildings that look the same.

Can you see how this context could affect what Jim would look like? Also, humans living on several planets and space exploration being the focus of the news every day are good reasons to make a young boy dream about being an astronaut, right? If the context was prehistory, a kid's dreams would be very different; maybe he'd want to be a mighty hunter or a fearsome shaman.

Context (where the character lives, his culture, and his relationships) clearly defines your character's personality and his ways of thinking and acting. Jim's context encourages him to be an adventurer, explore space, and find new planets...or even aliens!

Style

Before you begin to imagine how your character would look, define the style you want to aim for. For this example, I'd encourage you to opt for a cartoony look. The reason is that although you'll explore the full animation process, you don't want Jim to be so complex that the character creation is hard to achieve. For learning purposes, Jim will have smooth shapes without too many details.

Although you'll keep Jim simple, you still want him to look cool. You can go for something more realistic, something dark, or even kind of abstract. Decide ahead of time and create some drawings, or find some photos or other reference materials to help you determine his style, which will be essential to how the character ultimately looks.

Something else you want to keep in mind while you define the style (and every other thing that has to do with the appearance of a character) is technical limitations. You may not want to have really long hair on a character, for example, because long hair would make the animation or simulations much more complex.

Also, style depends on the medium in which the character will be used. In movies, you can use more detail and complexity, as you have enough time and sufficiently powerful machines (even render farms) to render each frame. But if you're working on a videogame character, the limitations are greater, as the character needs to work in real time. You need to use fewer polygons, less resolution in textures, or less complicated effects to increase the performance and allow a computer (or a console) to render the images on the fly.

Apearance

Now that you have Jim's personality, the context of his world, and his style, you can start thinking about his appearance. He lives in a futuristic environment, and when you think of futuristic worlds, you might picture plain-colored clothes with clean lines. White/gray and blue clothes could look right.

Here, it's OK to use clichés because certain characteristics are embedded in people's minds and give them an idea about a character or theme. That's precisely why clichés work, so don't fear or avoid them.

Suppose that clothes in the future are tight and fitted to the body. Furthermore, Jim is healthy and active, and he plays sports, so he's in good shape. It wouldn't be a problem for him to wear close-fitting clothes, as he wouldn't be ashamed of his body. Also, he's an adventurer, so maybe he worries less about appearance and more about clothes that are functional as well as comfortable.

The character's description (refer to "Personality" earlier in this chapter) mentions that he likes to be different from everyone else, so if everyone wears similar clothes, he would surely add some unique details to his attire to represent that attitude. He might wear a pin, a jaunty hat (which would also help represent him as adventurous), or some kind of reference to space travel and exploration: the dream he's fighting for.

Designing the Character

In this section, you follow a typical process you can use to create a character. The process progresses from general forms to details. You start by creating the basic shapes; then, little by little, you refine those shapes and add color to get to the final result.

Silhouettes

It's a good idea to start with quick sketches and basic shapes that help you explore and find the right proportions for your character (see Figure 5.1). Then you can pick the ones you like and keep developing and adding details to them. Many artists use this technique in designing a character. Silhouettes are very important; a great character is recognizable by its silhouette. You can recognize Super Mario, Mickey Mouse, Sonic, and Pikachu just by their silhouettes because they have original, unique designs.

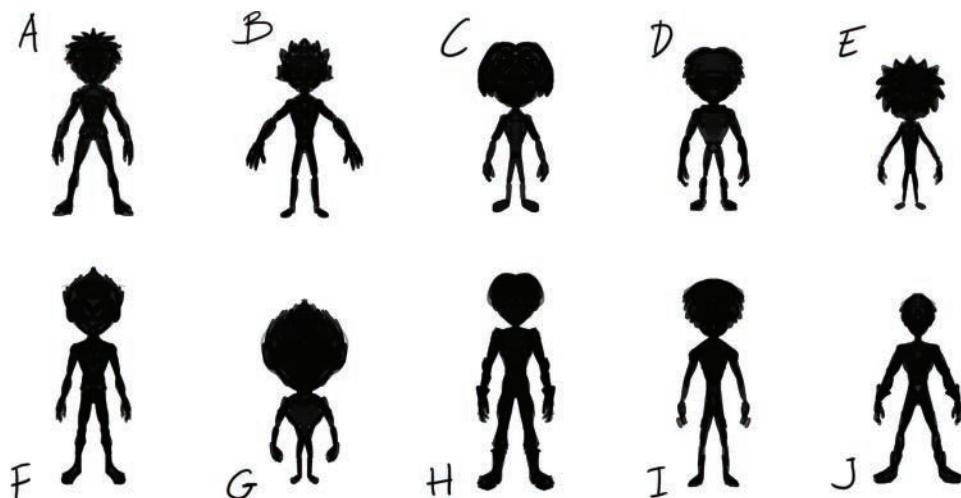


Figure 5.1 Silhouettes of Jim drawn for the purpose of studying the desired proportions and shapes

In this case, you're learning about 3D character creation, not designing the ultimate marketable character, so you don't need the whole world to recognize Jim by his shape. The goal is to design a character that looks cool and has personality.

Looking at the silhouettes in Figure 5.1, you can see how, from the character description, you can imagine completely different shapes for a character. Now you have to decide which one you like most or which one best fits the style you're looking for. Suppose that the silhouettes you like most for their shapes and proportions are A and F. They're kind of realistic but have big hands, feet, and heads. A big head (in comparison to the body size) helps identify Jim as a kid. J looks more like an adult, as its head is smaller, and E, which has a really big head compared with the body, resembles a small child.

Figure 5.2 shows the final silhouette, which is a mix of the versions liked most, with a little more detail added. The first silhouettes are little thumbnails that gave you an idea for Jim's shape; the final one is bigger and has more detail, so it can be used as a reference in the next stage. The silhouette doesn't have clothing details or a hat; those items will be added over the base design. For now, all you need is the character's main shape.

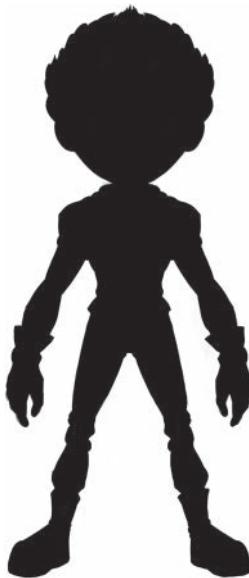


Figure 5.2 The final silhouette is a mix of versions A and F.

Note

The silhouettes in Figures 5.1 and 5.2 were done in Krita, a free open-source painting software that you should check out. You can find more information at <https://krita.org>. You have many alternatives for painting, of course, and you can use any software you feel comfortable with, such as MyPaint, Painter, GIMP, or Adobe Photoshop. You could even use the new Grease Pencil, the tool for 2D drawing and animation in Blender (not covered in this book). A feature that is very useful for creating silhouettes is mirror painting (not every drawing software has that feature, but Krita does), which allows you to mirror in real time what you're painting on the opposite side of the canvas. This feature really speeds the process, because you can see the full shape coming together while you're painting only one side of it.

Base Design

Next, you create your base design by taking your final thumbnail and turning it into a drawing. Just sketch some strokes around the borders and define some of the figure's interior shapes. At this stage, you need only a basic version of the character, so don't worry too much about details; you'll add those in the next pass.

You also add clothing at this stage. You can place as many variations as you want over the silhouette. It's OK to explore and create different versions of this base design so that later, you can choose the designs you like, mix them, and keep modifying them until you achieve the design you like best. Keep in mind, however, that you don't want to add things that will make the 3D design too complex later, so you should avoid complex design elements and aim for something that is easy to achieve.

You don't need a really refined drawing at this point; a quick rough sketch will do, such as the one in Figure 5.3. This sketch helps you understand how the character may eventually look so that later, you can dive into the specific design of each of its parts and finally combine everything for a clean, finished version. If something doesn't look great, don't worry; you still have a lot of time to change things.



Figure 5.3 Comparison between the final silhouette and the base design

As you can see in Figure 5.3, the hairstyle is not well defined. That part of the character will require some technical thinking, as hair is always a challenge. If you want to go for realistic hair, you can later use 3D particles to grow hair on a surface, comb it, cut it, and add the effects of gravity or wind to it. If you want to go that route, you need to have a detailed understanding of how hair particles function to make sure that the hair will work properly when designed using this method. (Otherwise, it's easy to screw things up.) If you prefer to go with a hand-modeled mesh, you have more options, but keep in mind the limitations of the mesh method.

For this character, you'll use a mesh for the hair, as it's the simplest option, and for this type of character design works just fine. If you don't have much drawing experience, keeping things simple by starting with generic geometric spheres helps you

understand the volumes. The head can be a distorted sphere, for example, and arms and legs are in essence similar to cylinders. If you have difficulty coming up with the character's shapes, try starting with this geometric base and then draw the details on top.

Head

Now it's time to design the head, face, and hairstyle. Maybe now is also a good time to work on hats, as they affect the hairstyle. Figure 5.4 shows several sketches for Jim's face. Keep in mind that you probably won't come up with anything great during your first tries. (For this figure, I'm showing you only my best attempts.) You should keep drawing and designing until you're happy with the result. The process may take hundreds of drawings, or you may be lucky and find something you like on the first try, but don't count on that!



Figure 5.4 Different sketches for Jim's head and some studies with a cap

After you look at all the designs, you may opt for the one featuring a cap, as a cap gives Jim some personality and looks cool. It also covers a big part of the head, making hair creation a lot easier!

A typical baseball cap may not be common in 2512, because people in that future society may wear crazy stuff on their heads, but remember that Jim wants to be different. In this chapter, you're creating only Jim, but if you also had to depict the city where he lives and more of its citizens, he would surely look different just by wearing a baseball cap in 2512.

Details

You now have a base design for Jim's body and his head, so it's time to add the details. Maybe you don't know how to draw or paint, but don't worry! The goal of making these designs is not to make a perfect drawing. Designs are just sketches—quick drawings that help you understand the shapes of your character. Understanding the character and how its details are constructed allows you to translate everything into the 3D model.

Suppose that you are modeling a watch. If you start modeling right away, maybe you'll end up encountering some issues and failing. Maybe the watch won't look realistic or the gears won't fit, and the reason may be that you didn't study the shapes. It's always good to look for references and use them when designing. You can go ahead with only an idea in your head, but I recommend that you put the idea on paper (or a screen) so you can see and explore it before diving into the 3D modeling process.

Figure 5.5 shows some of the sketches I made to define Jim's clothing and the details selected for the definitive design, which were drawn from several points of view. The jacket, for example, has designs on its front and back. These features are important because you probably wouldn't need the back for a 2D drawing, but in a 3D model, every side is equally important.

The suit has rectilinear shapes that maintain the style and make the material look more elaborate than plain fabric. The shoulders, elbows, and knees have padding, which gives the clothes the look of a uniform and is just what you want for Jim, because everyone in the future Jim lives in is wearing the same kind of suit.

Jim is wearing an earpiece to listen to music or receive calls. The cap is a personal detail that distinguishes him from others, and he's going to wear it turned around, as a young rebel would. His personality can be reflected in the colors of his clothes as well. Maybe some parts of the suit will be different colors to make them stand out from the clothes worn by the rest of the people in his world; you explore colors in the coloring stage (see "Adding Color" later in this chapter).

The back of the jacket has a small backpack-shaped bulge where the suit's electronic systems may be stored. The cuts in the arms, above the elbows, seems to resemble the design of today's astronaut suits.

One other detail has been added to the suit. Jim put space-exploration symbols on the chest and on the front of the cap; these symbols, along with the uniformlike style

of the clothes, make him look like an actual space explorer. For those symbols, I used Saturn, which is both a recognizable planet and a known icon of space exploration.

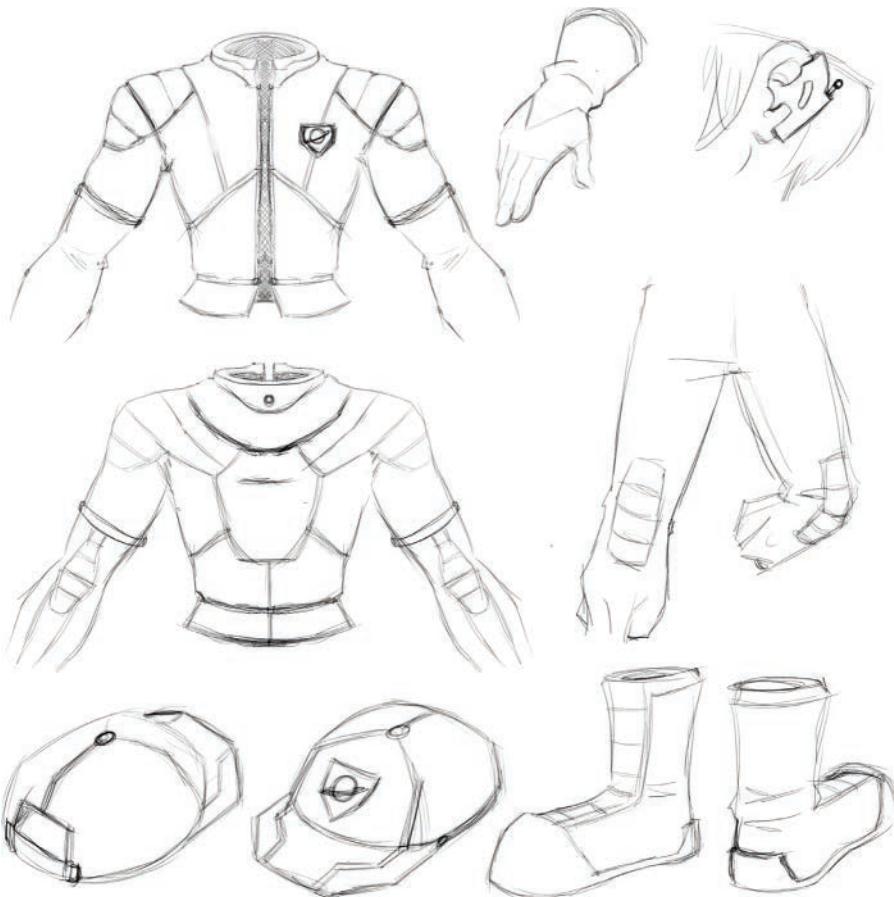


Figure 5.5 Sketches of character-design details, including clothes, earpiece, boots, gloves, and cap

Refined Design

At this point, you have a clear understanding of how everything looks: the face, hair, and clothes, as well as other details. Before you create the final artwork, get back to the base design and add more detail. Now is also a good time to sketch the character from the back (see Figure 5.6).

Everything is in place and looking good! In the next section, you do some color testing.



Figure 5.6 Refined design over the base design and a design for the back of the character

Adding Color

Now that the base design is complete, it's time to add some colors to Jim and see how he looks with different color schemes. (If you have been working on paper up to this point, now is a good time to scan your design and start using a digital 2D image editor, as it allows you to test more than one color scheme for the same design and to make easy, fast retouches.) You need a version of Jim's front view with clean lines that allows you to use the color bucket tool in your editing program to quickly fill in areas with color (see Figure 5.7). Store each part of the character in a different layer, which lets you play with the skin-color values, for example, without affecting the rest of the colors. With this method, you can try several options.

When your colors are well organized by layers, testing a new color scheme may take only a couple of minutes. Figure 5.7 shows different hair colors so you can see how the process works. Suppose, though, that you already know that the hair is going to be blue, just like the eyes, as blue complements the bluish grays of the suit. You also need to pick a definitive color scheme for Jim's suit. Continue with the version in the middle, which already has the blue hair you chose, as the light tones of the suit have less contrast than the tones in the other two versions.

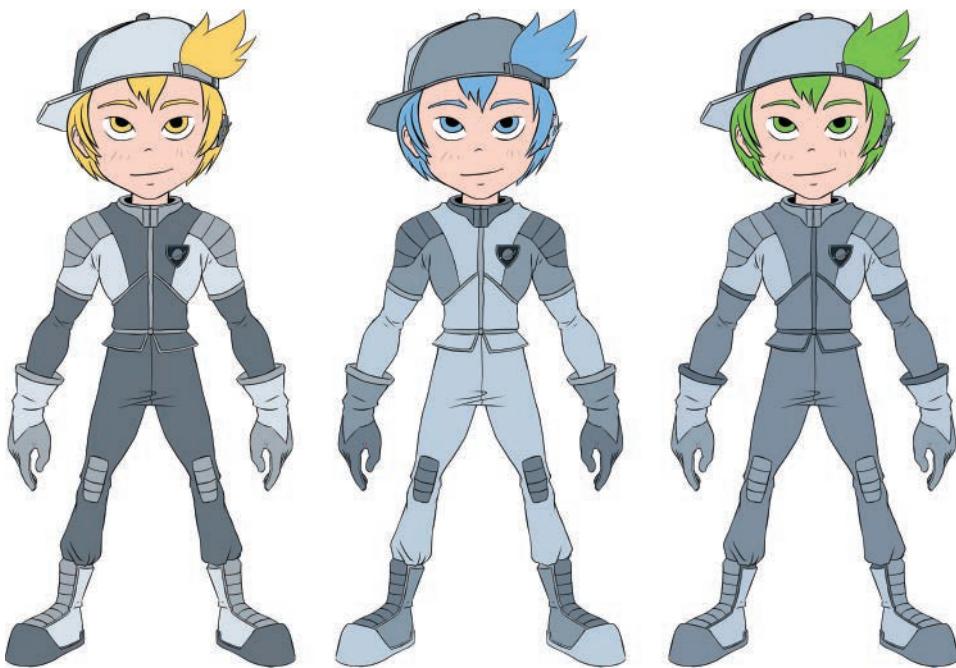


Figure 5.7 Testing different color schemes over the design

Finalizing the Design

At this point, making a final illustration of the character is possible, as you know how it's built, how it looks, and what its design details are. Figure 5.8 shows the final version of Jim. This level of quality is unnecessary for a preliminary design, but it helps you become familiar with the character so you can learn about him and understand his proportions and features. Also, the process of creating some final artwork sometimes even identifies potential problems in the character's design when you try to pose it and watch it from a particular perspective. Some pieces of the suit may not work properly in certain positions, for example. Usually, when they create a complex character, concept artists do a lot of illustrations like these to make sure that the character not only looks cool but also has all its pieces working well together.

Note

In the associated files for this book, you can find a video showing the painting process for this illustration. I hope that this video will help you understand the steps I followed and motivate you to paint your own character.



Figure 5.8 Using the final design to create an illustration of Jim

Making Character Reference Images

You've designed your character. If you're an experienced designer, you can start modeling right away. But regardless of your expertise, you probably want to create some references to use in the 3D modeling program so that you have at least a general idea of the character's size and basic shapes, which will make the modeling process much faster. These images stay in the background of the 3D Viewport while you model, so you can model on top of them. The reference images should represent the character in a neutral pose, as that pose is what you'll need to model the character. Cool poses will come later.

For this example, you need six different reference images to place in 3D views. You'll put these images in the background and place your model on top of them:

- Head, front view
- Head, back view
- Head, side view

- Body, front view
- Body, back view
- Body, side view

These reference images help you make your 3D model fit your original design. Design elements tend to change during 2D-to-3D conversion, as 2D and 3D are different worlds. But by using references, you get something in 3D that's close and proportionate to the 2D design you created.

For the head views, you don't need hair for now. First, you need the head's shape; you'll add the hair on top of that shape later (see Figure 5.9).

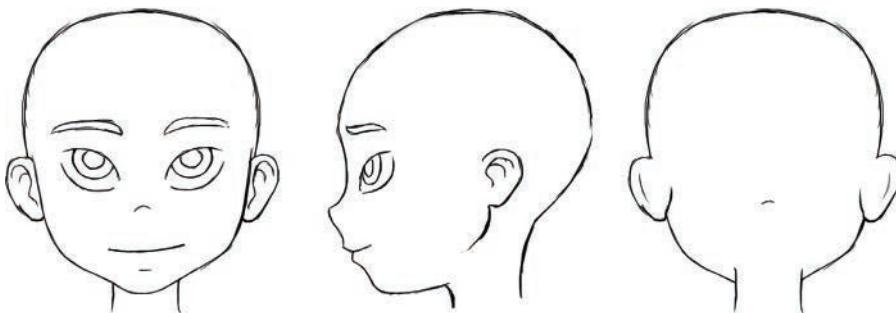


Figure 5.9 Head with front, side, and back views

For the body views shown in Figure 5.10, you can see that the side view doesn't have arms. This omission is intentional, as you don't want the arms to be in the way as you model the rest of the body. Later, you can model the arms from the front and back images. The side view would not have much relevant information about the arms, and they would cover the body of the character, which needs to be visible in the side view.

Notice the horizontal lines in the sketch. These lines serve as references so the character's features are in the same position in all the views so that later, you won't have trouble placing the characters in 3D. It's to be expected that your images won't be perfect; after all, they're hand drawings, and there will always be some errors, but the better aligned your images are, the easier the modeling will be later on. If the reference images are not aligned, you'll have to do a lot of guesswork when modeling your character; you may even have to reinvent some areas while you model.

Note

Feel free to play with these designs and change them to something that looks better to you or has a style that you like more. The intention is to give you a good starting point if you've never done character design and need some initial guidance. This example should give you a base to start with and a method to follow, but you don't have to stick to it. Character design is a creative process, so don't be afraid to try new and crazy things!

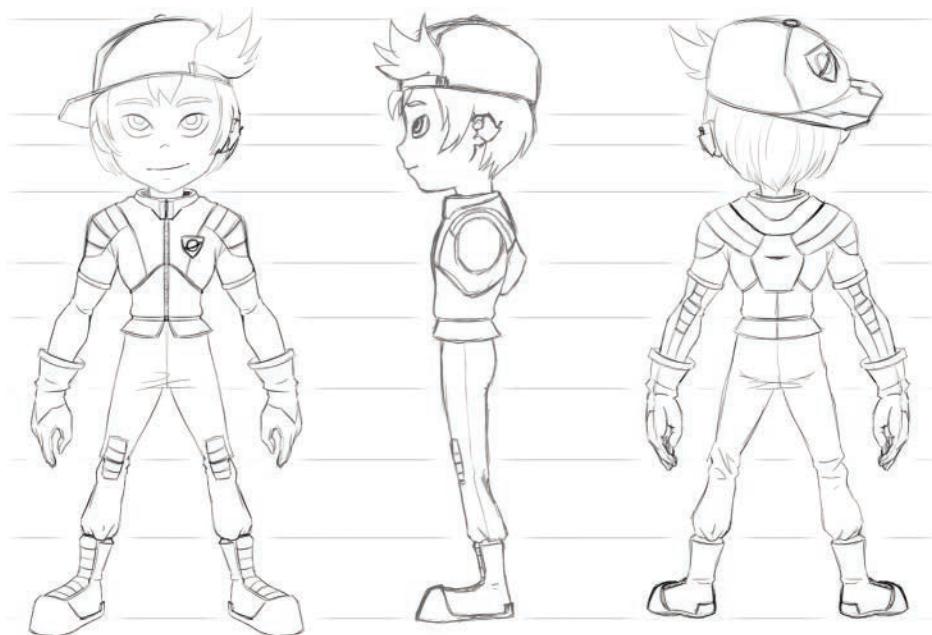


Figure 5.10 Body with front, side, and back views

Using Other Design Methods

The method I just described isn't the only one available for designing a character. A lot of artists create their own methods and techniques over time. The following list provides some other options you might try:

- Use a simple 3D model made of spheres, cubes, cylinders, or metaballs to explore the silhouette's basic shape and proportions. This method allows you to see how the character will look in 3D.
- Use random brushes in a painting program when exploring shapes. This method is likely to give you some unexpected results, and you may discover cool things that you would have missed by working with pencil and paper.
- Use vector imaging software such as Adobe Illustrator or Inkscape to test silhouettes. You could also use 3D vector curves in Blender as well. This method is similar to the simple 3D-model method, except that it's in 2D. This is cool because it lets you scale and rotate parts of the body easily to try different proportions in your design.
- Use the Skin modifier in Blender for character prototyping. Basically, you draw a character's skeleton with vertices and edges and then use the Skin modifier to give it thickness and a solid mesh in which you can also control the thickness of

each part. This modifier is intended to be used to create base meshes for sculpting, but you can also use it to test shapes for character creation.

- Use image compositing to pick parts of various photos or drawings and combine them to build your character's silhouette.

Summary

Character design can be quite complex, and you have to think about a lot of elements. You can dive into the modeling with just an idea in mind, of course, but that's probably going to be difficult, as you'll have to invent stuff on the fly. This design stage is crucial because it lets you define everything about your character: personality, attitude, looks, clothing, details, and so on. When you've done all that, you'll know your character very well, and you'll know for sure that it will look good when it's modeled in 3D. Otherwise, you may find that after all your time and effort, the idea you had in mind was not clear enough, and some things may not work as expected.

Design is a process of iteration, so it's important to keep in mind that not everything will work on the first try. You should be prepared to try, fail, and repeat parts of this process until you reach the point at which you're satisfied with your resulting design.

Remember: Preproduction is your friend!

Exercises

1. Starting with Jim's designs, add or replace elements to make the character look different.
2. If you're up for the challenge, design your own character!

This page intentionally left blank



Modeling in Blender

6 Blender Modeling Tools

7 Character Modeling

This page intentionally left blank

6

Blender Modeling Tools

Modeling is probably the most important stage in the character-creation process because it's when you generate the polygons that become the principal forms of your finished character. In this chapter, you learn the basics of modeling in Blender and see how to use some of the main tools you'll have at your disposal. As a result, when one of these tools is mentioned in the following chapters, you'll be familiar with it. You need to address three primary technical considerations before you can model properly in 3D: identify the elements that comprise a mesh, learn how to select them, and know what tools you can use to work with them.

Working with Vertices, Edges, and Faces

Every 3D model is composed of the same three elements: vertices, edges, and faces. A *vertex* is a point in space. When you connect two vertices, you've created an *edge*, and if you connect three or more vertices in a closed loop and fill in the gap between the edges, you've created a *face*. A *face* is basically a polygon. You can see how these elements look in Figure 6.1.

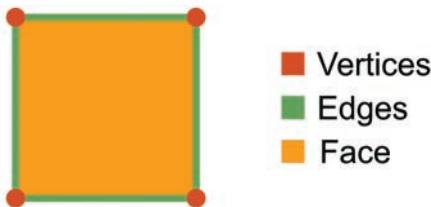


Figure 6.1 Vertices, edges, and faces: the elements that make up every 3D polygonal mesh

The three types of faces are *triangles*, *quads* (four-sided faces), and *n-gons* (faces with more than four sides). In 3D modeling, there is a rule: If you possibly can, use only quads when creating a model. There are different reasons for this rule, but mainly, if you're using subdivisions (Subdivision Surface modifier), the model deforms and can be smoothed better if it's made out of quads. Triangles and n-gons can be problematic at times because they can cause pinches in the mesh, especially if they are used in curved shapes or if the model is deforming because of animation.

In some situations, though, using a triangle or an n-gon is more beneficial for the mesh. In some complex formations, n-gons create better deformations and subdivisions than four-sided polygons do. It takes experience in modeling to discern those situations, and I encourage you to look into articles on the subject by experienced modelers (search online for topics such as 3D modeling topology), as this topic is beyond the scope of this book. The way vertices, edges, and faces are distributed to shape the surface of your model is called *topology*.

Selecting Vertices, Edges, and Faces

To access these mesh elements, first enter Edit Mode through the Interaction Mode selector on the 3D Viewport's header, or press **Tab** on your keyboard. When you're in Edit Mode, you'll be able to select vertices, edges, and faces. In Figure 6.2, you can see those elements' icons on the header.

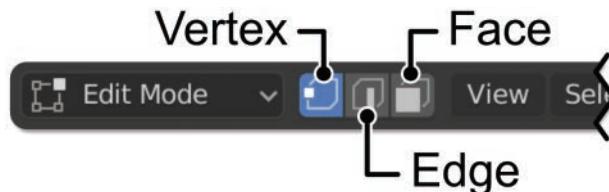


Figure 6.2 The 3D Viewport's header and its icons for selecting vertices, edges, and faces

Tip

If you press **Shift** while you click the element icons on the 3D Viewport's header, you'll be able to select two types of elements at the same time. In Vertex Selection mode, for example, hold down **Shift** while you left-click the Edge icon, and you'll be able to select vertices and edges simultaneously.

A faster way to switch among vertex, edge, and face selection modes is to press **Ctrl+Tab** while you're in Edit Mode. A little pop-up menu appears at your cursor's position, listing options that let you select a different element.

You can also switch among vertices, edges, and faces by pressing **1** (vertices), **2** (edges), and **3** (faces) on your keyboard.

Accessing Modeling Tools

You have different ways to access the modeling tools in Blender. You can access nearly all of the tools from menus, but the most common tools also have their own keyboard shortcuts. You can access the modeling tools as follows:

- **3D Viewport header:** In the 3D Viewport's header, you'll find several menus (such as Mesh, Vertex, Edge, and Face) where most tools can be found.

- **Toolbar (T):** You can use the active tools in the toolbar for modeling. Not all the tools are available from there, but the main ones are.
- **Contextual menu:** Right-click in the 3D Viewport while in Edit Mode to see a contextual menu with related tools.
- **Search:** In Blender, when you press **F3**, a Search menu appears. You can type the name of the tool you want to use and then choose it from the menu to apply it.

Here are the keyboard shortcuts you can use to access these tools while in 3D View:

- Vertices: **Ctrl+V**
- Edges: **Ctrl+E**
- Faces: **Ctrl+F**

Making Selections

In this section, you discover some tips for making selections when you’re in Edit Mode. For the most part, selections work exactly the same way as they do in Object Mode (selection is covered in Chapter 2, “Blender Basics: The User Interface”), but in Edit Mode, they are applied to vertices, edges, or faces. You can add to a selection by pressing **Shift**, for example, or you can perform a box selection by pressing **B**. The following sections cover a few selection techniques that are available only in Edit Mode.

Shortest Path

If you left-click to select a vertex and then hold down **Ctrl** while you left-click a second vertex, Blender automatically selects the shortest path between the two vertices (see Figure 6.3). This selection method also works with edges and faces.

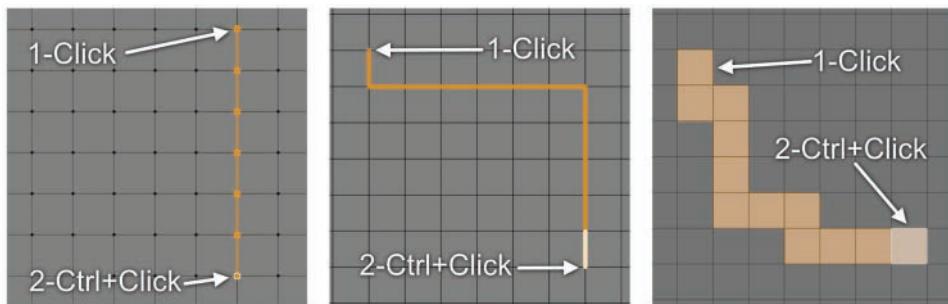


Figure 6.3 Some examples of how Shortest Path Selection works with vertices, edges, and faces

If you keep pressing **Ctrl** and left-clicking, the new paths are added to the selection, making this method a very fast way to select a series of vertices that follow a path. (This technique proves to be very useful in Chapter 8, “Unwrapping and UVs in Blender,” where it’s used to mark seams in a model.)

The interesting features of Shortest Path Selection include the nth selection, skip, and offset options, which you can access from the Operator panel (or by pressing **F9**). These options let you change the values to deselect vertices in the path alternatively. Selecting one vertex every three vertices, for example, is possible thanks to this option, which in big models with many vertices can save lots of time. Shortest Path Selection’s options include other interesting features, and you should take a look at those as well.

Tip

Another great use of this tool is to hold down **Shift** while using Shortest Path Selection to select a rectangular grid of faces that uses your first and last selected faces as opposite corners.

Proportional Editing

Proportional editing is a very useful feature, especially when you’re working with organic modeling. Basically, you select an element (vertices, edges, faces, or objects), and when you transform it, the surrounding elements follow the selection, depending on the falloff type and radius of influence you define (see Figure 6.4).

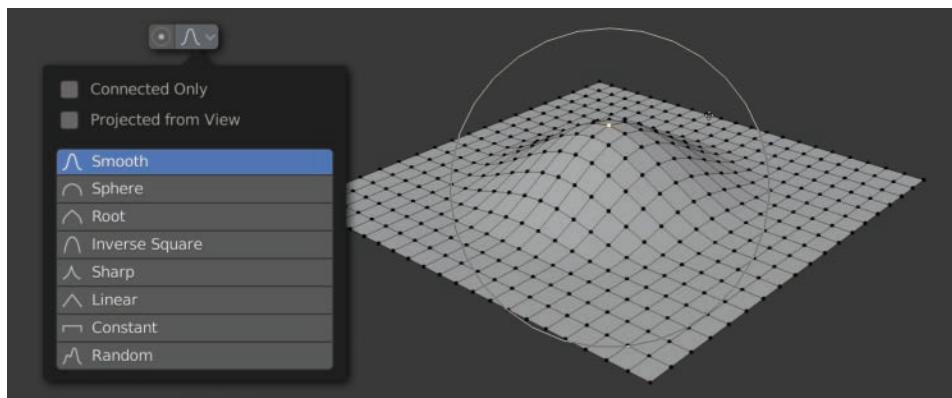


Figure 6.4 The proportional editing menu on the 3D Viewport’s header with options and falloff types, and the effect of the proportional editing tool on a mesh when moving a single vertex

Using proportional editing is very easy: Just find the icon on the 3D Viewport's header and select one of the methods to enable it. Alternatively, you can press **O** on your keyboard to turn it on and off. If proportional editing is enabled, when you perform a transform, a circle appears around the selection that indicates the radius of influence. You can roll the scroll wheel of your mouse up and down to increase or decrease the size of the circle.

Next to the button to enable/disable proportional editing, you can find a button that holds a pop-up menu, allowing you to select different falloff types. Here are two other options you can enable if needed within that menu (see Figure 6.4):

- **Connected Only:** This option affects only the vertices, edges, or faces that are directly connected to the selection. It won't affect parts of the same mesh that are separated.
- **Projected from View:** This option's effect depends not on the mesh or three-dimensional space's distance, but on the point of view from which you're looking at the mesh.

Tip

In other software, this feature is called falloff selection, soft selection, smooth selection, or other names with a similar meaning. These features may work in slightly different ways, but they're basically the same as Blender's proportional editing.

Linked Selection

A mesh can be made of different parts that are not connected by edges, and you may want to select some parts without selecting the others. These isolated parts of a mesh are also called *islands*. You have two quick ways to select those linked parts:

- Select one of the mesh's vertices, edges, or faces, and press **Ctrl+L**. All the elements of that island are selected.
- With nothing selected, place the cursor on top of an island and press **L** to select it. Press **L** on top of another element to add it to the selection, and press **Shift+L** to subtract from the selection.

Loops and Rings

The shape that edges follow along a surface is usually called *edge flow* or *mesh flow*, and it's very important in modeling (as you learn in Chapter 7, "Character Modeling"). In any mesh, you can find loops and rings. A *loop* is a series of connected edges that follow a path. A *ring* is a series of parallel edges along the surface of a mesh (see Figure 6.5).

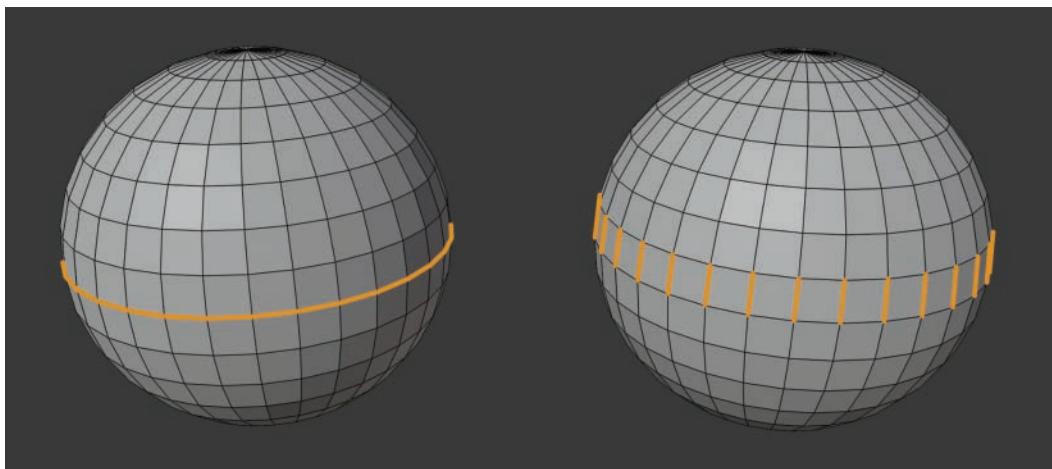


Figure 6.5 An edge loop (left) and an edge ring (right)

You can quickly select loops or rings with two keyboard shortcuts:

- **Selecting loops:** Place your cursor on top of an edge, press **Alt**, and right-click to select the whole loop.
- **Selecting rings:** Place your cursor on top of an edge, press **Ctrl+Alt**, and right-click to select the whole ring.

Hold down **Shift** combined with either of these shortcuts to add to the selection.

This technique works with vertices, edges, and faces, but in the case of faces, selecting a loop or a ring returns the same results.

A secondary option to select loops and rings is to do it from the Select menu in the 3D Viewport's header. You have to select at least one edge in the model. From the Select menu, choose Select Loops, and then choose Edge Loops or Edge Rings: Blender will select the loops or rings to which the selected elements belong.

Border Selection

A *border* is the series of edges that define the limits of a mesh that is not closed. Take a plane, for example. A plane's four edges are open, and that's a border. A cube, on the other hand, is closed.

To select a border quickly, hold down **Alt** while you left-click the outer edges of the mesh twice.

Grow and Shrink Selection

When you have a selection of vertices, edges, or faces, you can press **Ctrl+NumPad +** (plus sign) or **Ctrl+NumPad -** (minus sign) to grow or shrink the scope of the selection through the connected elements.

Select Similar

After making a selection, pressing **Shift+G** (or using the Select Similar options within the Select menu from the 3D Viewport's header) shows different options depending on the type of element that you just selected. When you select an element, such as an edge, and use the Select Similar tool, for example, you can select all the similar edges in that mesh automatically. You can select by length, face angles, direction, and several other parameters.

Pay attention to the Adjust Last Operation menu (remember that you can press **F9** to see it), as it gives you options for modifying the selection. An especially useful option is the threshold, which lets you define the amount of similarity with the original selection and requires a certain value for the other elements to be selected.

Linked Flat Faces

You can find the Linked Flat Faces option in the Select menu of the 3D Viewport, within the Select Linked submenu. This option selects all the faces around the selection as long as they are on a flat surface and the selection reaches an edge that has an angle. The sharpness value in the Operator panel lets you tell Blender how big the angle between faces has to be to limit the selection.

Select Boundary Loop and Loop Inner-Region

You can find options for selecting the boundary loop and the loop inner region in the Select menu of the 3D Viewport's header and in the Select Loops submenu.

When you have a selection of several faces on a surface, you can use the Select Boundary Loop tool. This tool leaves only the boundary loop selected on the borders of the previously selected faces (see Figure 6.6).

The Select Loop Inner-Region tool is exactly the opposite of Select Boundary Loop. This tool allows you to select a closed loop on a surface and to select all elements inside that loop (see Figure 6.6).

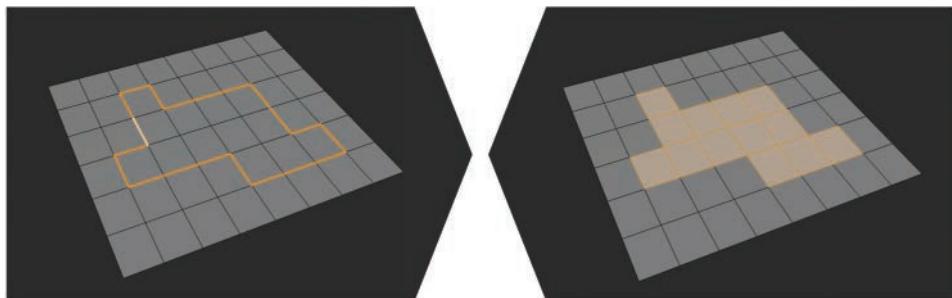


Figure 6.6 The effect of the Select Boundary Loop tool (left) and Select Loop Inner-Region selection (right). Each image shows the effect of its selection tool applied to the selection in the other image. The image on the left shows the result of the Select Boundary Loop tool applied to the selection in the right image.

Checker Deselect

The Checker Deselect tool is actually a reverse take on selections. First, you select an area; then you use Checker Deselect to deselect given elements, thus ending up with the desired selection. You can access this tool from the Select menu on the 3D Viewport's header.

Basically, this tool generates a pattern based on three values that you set up in the Adjust Last Operation panel: nth selection, skip, and offset. Then you use this pattern to deselect some of the elements in the selection you made.

Other Selection Methods

In the 3D Viewport header's Select menu, you'll find all the selection methods discussed previously, as well as several others. The methods mentioned in this chapter generally are the ones that are used most often, but I encourage you to check out the rest of them, as you may find some that are useful. Also, keep in mind that you can always go to the Select menu in case you don't remember the keyboard shortcut for any of the methods. (Note that the Proportional Editing option cannot be accessed from the Select menu; it's accessed from the 3D Viewport's header.)

Using Mesh Modeling Tools

This section provides a reference for the main modeling tools (alphabetically ordered) that are available in Blender. You'll learn how to use them, see what options they have (view the Adjust Last Operation panel in the bottom-left corner of the 3D Viewport, or press **F9**), and know what their effects are. Test them and learn them, as they will be used a lot in the following chapters. Don't worry if you don't remember all their features, however; you can come back to this chapter whenever necessary.

Note

All the tools discussed here can be found in the menus explained in the previous section, but in this section, I specify only the keyboard shortcuts for them. Also, with many of these tools, you can drag things with your mouse to move them. Remember that you can use the keyboard shortcuts to aid the transform: Press **Shift** to move things with more precision; press **Ctrl** to snap; or enter numerical values and then use the **X**, **Y**, and **Z** keys to constrain the movements to their respective axes.

Bevel

Bevel is a very useful tool, especially for technical and inorganic models; it is used to create bevels and chamfers. It can be used with vertices, edges, and faces. (The Bevel tool works with vertices only when you enable the Only Vertex option in the Adjust Last Operation panel after invoking the tool.) You can see how it's used in Figure 6.7.

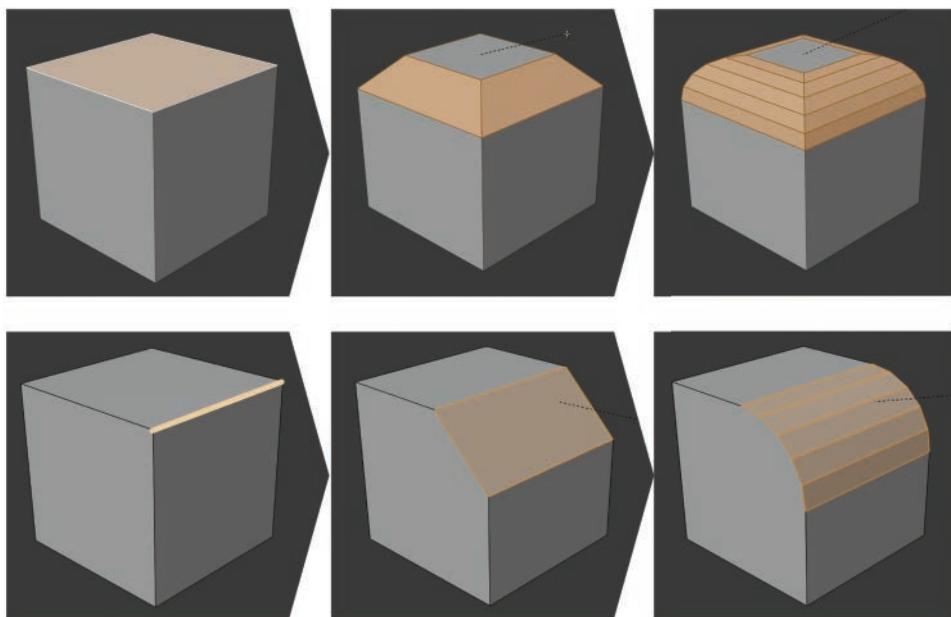


Figure 6.7 Using the Bevel tool in a face (top) and in an edge (bottom)

To use the Bevel tool:

1. Select the element you want to bevel.
2. Press **Ctrl+B** and drag your mouse to increase or decrease the bevel size.
3. Roll the scroll wheel to increase or decrease the bevel divisions (segments). Alternatively, press **S** and move the mouse or enter a number to define the segments amount.
4. Press **P** and move the mouse to change the profile (shape) of the bevel. (Numerical input of a specific profile value is allowed.)
5. Left-click to apply or right-click to cancel.

In the Bevel tool options of the Adjust Last Operation panel, you'll find the size calculation method, the size of the bevel, the amount of segments, and the bevel's profile (in or out), as well as an option that lets you apply the bevel only to vertices. (You can launch the Bevel tool in vertices-only mode directly by pressing **Ctrl+Shift+B**.)

Tip

Blender's Bevel tool is similar to the Chamfer tool in 3ds Max.

Bisect

The Bisect tool lets you create a line across a selection and project it to generate an edge loop that divides the mesh. After that, you’re able to leave only one side of the mesh from that division visible (from the Adjust Last Operation menu), which is useful for creating cross sections of objects (see Figure 6.8).

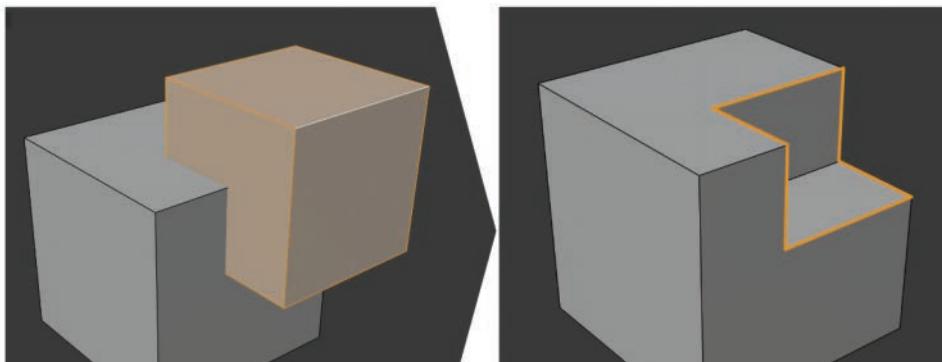


Figure 6.8 Using the Bisect tool on the default cube

To use the Bisect tool:

1. Select the part of the mesh you want to divide. (Sometimes, you want to divide the entire mesh, which you can select by pressing **A**.)
2. Select the Bisect tool from the menus or Search (**F3**). (This tool has no keyboard shortcut.)
3. Left-click the first point of the line you want to draw, and drag to indicate the line’s direction.
4. Release the mouse button to apply. A manipulator will show up for adjusting the cut.
5. From the Adjust Last Operation menu, choose the options that direct how the Bisect tool will perform.

Boolean Operations

The Intersect (Boolean) and Intersect (Knife) tools let you use two parts of a mesh and cut new edges in their intersections (see Figure 6.9). There are some differences between the tools, both of which you can find in the Face menu (**Ctrl+F**). Keep in mind that these tools work only when elements of a mesh (inside the same object) intersect.

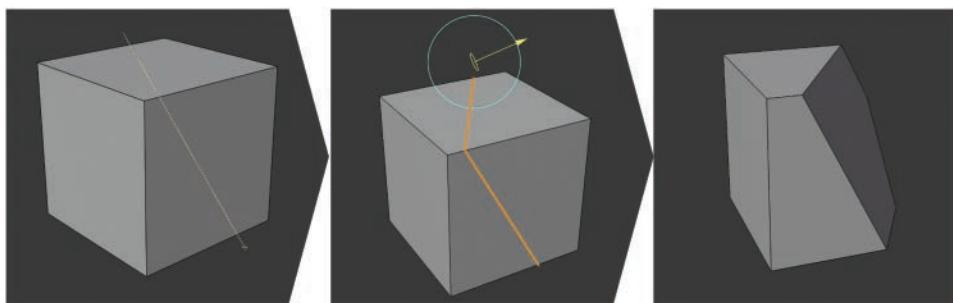


Figure 6.9 The selection of one cube inside another (left), the result of the Intersect (Boolean) tool in Difference mode (middle), and the effects of the Intersect (Knife) tool (right)

Intersect (Boolean)

The Intersect (Boolean) tool acts similarly to a Boolean modifier operation. Booleans let you subtract volumes from and add volumes to a mesh by intersecting it with another mesh. In Blender, you do this by using the Boolean modifier, but you can also do it in Edit Mode with the Intersect (Boolean) tool.

To use the Intersect (Boolean) tool:

1. Select the part of the mesh that you want to use as a cutter.
2. Press **Ctrl+F** and select Intersect (Boolean).
3. Select the type of Boolean operation that you want to apply from the Adjust Last Operation panel.

Intersect (Knife)

Intersect (Knife) works similarly to the Intersect (Boolean) tool, but instead of subtracting or adding volume, it cuts the mesh and generates new edges on the surface. It also separates the different intersecting meshes through those edges. This tool is very useful when you need to perform a cut with a given shape on a mesh. Just model a cutter mesh and use this tool to generate such a cut.

To use the Intersect (Knife) tool:

1. Select the part of the mesh that you want to use as a cutter.
2. Press **Ctrl+F** and select Intersect (Knife). The meshes are cut as though you had used the Knife tool, with the shape of the cutter mesh, but kept them in their positions.
3. Remove any parts that you don't need (such as the cutter element).

Note

Boolean operations are usually performed with the use of the Boolean modifier on an object level, but these modeling tools can speed the process when there's no need for having a nondestructive and interactive Boolean operation.

Bridge Edge Loops

The Bridge Edge Loops tool works well to join a series of adjacent edge loops. It's like an advanced Face tool (covered later in this chapter), but instead of creating one face at a time, it creates a group of them, joining two selected edge loops (see Figure 6.10).

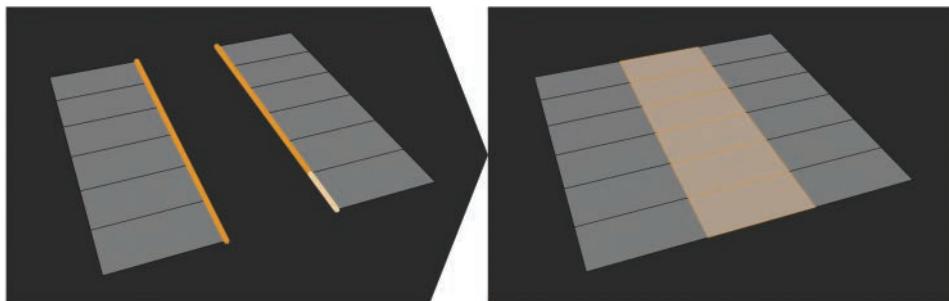


Figure 6.10 Using the Bridge Edge Loops to connect two separated edge loops

To use the Bridge Edge Loops tool:

1. Select a string of edges (edge loop).
2. Select another edge loop in a separate part of the model. (For optimal results, both edge loops should have the same number of edges.)
3. Press **Ctrl+E** to access the Edge menu, and select the Bridge Edge Loops tool. (You can also find that tool in the contextual menu if you're in Edge selection mode.)

The options for this tool let you control the type of connection that is made between the loops, twist the resulting geometry, and apply some merging options (but only if the edge counts are the same in both edge loops that you're trying to connect). These options also include several features that allow you to control the number of segments (cuts) in the new geometry, as well as its shape.

Connect

The Connect tool joins two vertices with a new edge across a face or a series of faces. For this tool to work, a face must exist between the selected vertices (see Figure 6.11).

To use the Connect tool:

1. Select two vertices.
2. Press **J** to connect them.

Tip

If you select a series of vertices (assuming that there is a face between one and the next), when you press **J**, Blender connects them in the same order in which you selected them, so you don't need to connect them one at a time.

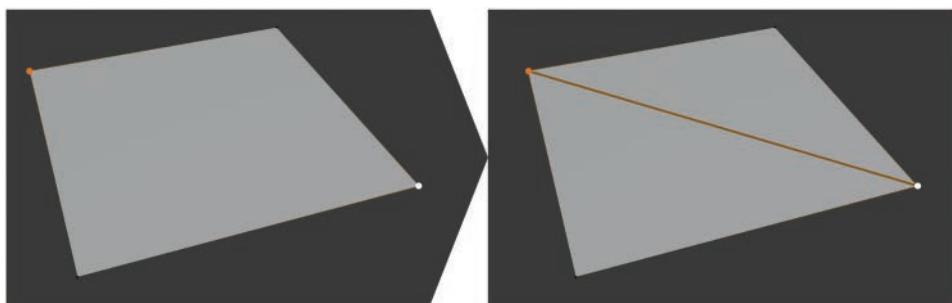


Figure 6.11 Using the Connect tool to join two vertices sharing a face with a new edge

Delete and Dissolve

In Blender, when you have some of a mesh's elements selected and you press **X**, a menu appears that gives you several options. You can use Delete to remove vertices, edges, or faces. If you select one of the other options, such as Only Faces, only the faces are deleted; the edges and vertices remain.

You also see the Dissolve tool, which is similar to Delete except that instead of making elements disappear, it replaces them with a single n-gon. This tool comes in handy when you're working on complex surfaces and need to adjust the way in which the edges are placed. You can dissolve the faces and then reconnect the remaining vertices manually (see Figure 6.12).

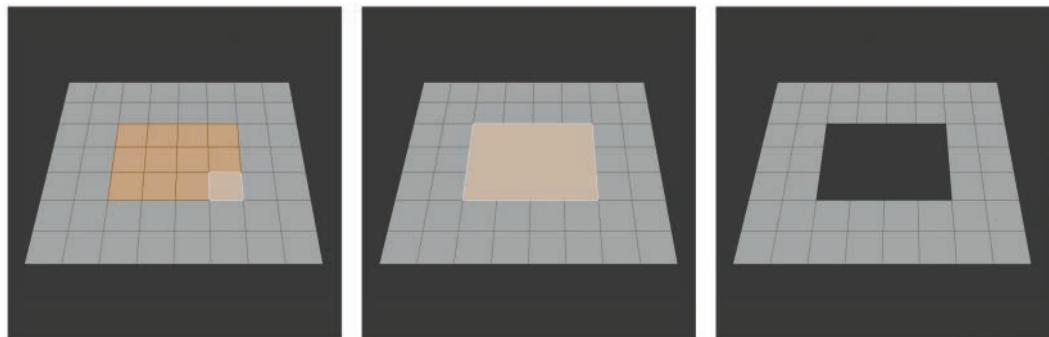


Figure 6.12 The selected faces (left), the effect of deleting those faces (center), and the result of dissolving those faces (right)

To use Delete or Dissolve:

1. Select a set of adjacent vertices, edges, or faces.
2. Press **X** or **Del** and then choose Delete or Dissolve from the pop-up menu to delete or dissolve the selection.

When you use the Delete tool, depending on the option you select, Blender presents another set of options that let you control the degree of deletion for that particular tool. With the Limited Dissolve option, you have control of the angle at which the dissolved faces are joined.

Tip

To save some time when dissolving faces, you can select the faces you want to dissolve and press **F**. The **F** key usually generates faces between vertices and edges, but when it's used on faces, it replaces all the selected faces with a single face (which is the same effect as Dissolve).

Duplicate

Duplicate is as simple as it seems. This tool lets you duplicate a piece of the mesh and place it somewhere else very quickly.

To use the Duplicate tool:

1. Select one or more vertices, edges, or faces.
2. Press **Shift+D**.
3. Drag your mouse to move the selection. You can use the **X**, **Y**, and **Z** constraints while moving, just as you would in a normal transform.
4. Left-click to confirm.

Duplicate is quite a simple tool, but it gives you a lot of options. From the Adjust Last Operation menu, for example, you can control the offset of the duplication, constrain it, and even access the proportional editing features.

Extrude

Another useful modeling tool is Extrude. To understand how it works, imagine the floor of a house. You select the floor, and when you extrude it, you move it up as though it were a duplicate to create a ceiling; then Blender generates the walls to join the floor and ceiling (see Figure 6.13). You can use Extrude for vertices, edges, and faces. Blender provides different methods for extruding and a few Extrude options, such as Extrude Faces, Extrude Faces Along Normals, and Extrude Individual Faces. The first option extrudes the entire selection at the same time in an average or manually defined direction; the second does the same but extrudes in the direction that the faces are facing; and the third extrudes each face within the selection individually.

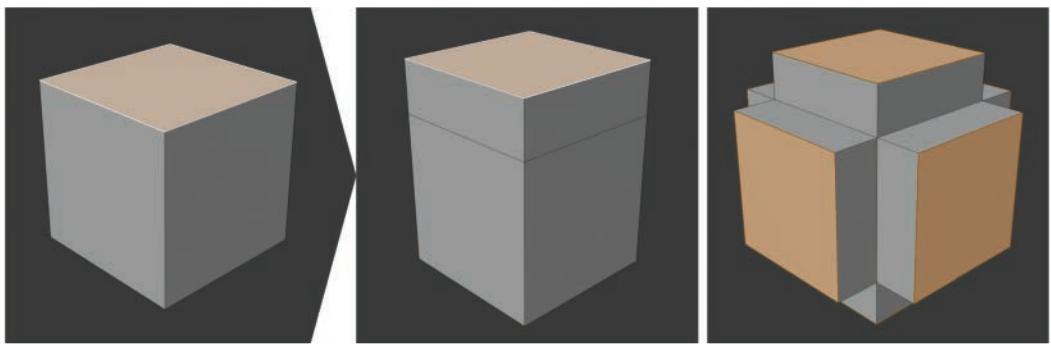


Figure 6.13 Extrude tool in action. In the image on the right, you see the result of extruding with individual faces enabled (**Alt+E**) when all faces of the cube are selected.

You have multiple methods for using the Extrude tool. Here is the first method:

1. Select one or more vertices, edges, or faces.
2. Press **E** to extrude.
3. Drag your mouse to move the new geometry. You can press **X**, **Y**, and **Z** to constrain it to a given axis. (If you extrude a face, the extrusion is constrained according to the orientation of the face by default.)
4. Left-click to confirm the extrusion.

This is the second method:

1. Select one or more vertices, edges, or faces.
2. Press **Ctrl** and right-click where you want the extrusion to go. Blender extrudes automatically.

And here is the third method:

1. Select one or more vertices, edges, or faces.
2. Press **Alt+E** to bring up a menu of extrude options, and select one.
3. Drag your mouse to adjust the height of the extrusion.
4. Left-click to confirm.

Within the Adjust Last Operation menu for an extrusion, you'll find options to change the extrusion's direction and size or constrain it to an axis, as well as some proportional editing features.

Fill and Grid Fill

With the Fill and Grid Fill tools, you can select a part of the mesh where you have a hole, and Blender fills it. Generally, Grid Fill provides better results and cleaner geometry (see Chapter 7, “Character Modeling,” for more information about clean geometry and topology) than Fill does (see Figure 6.14).

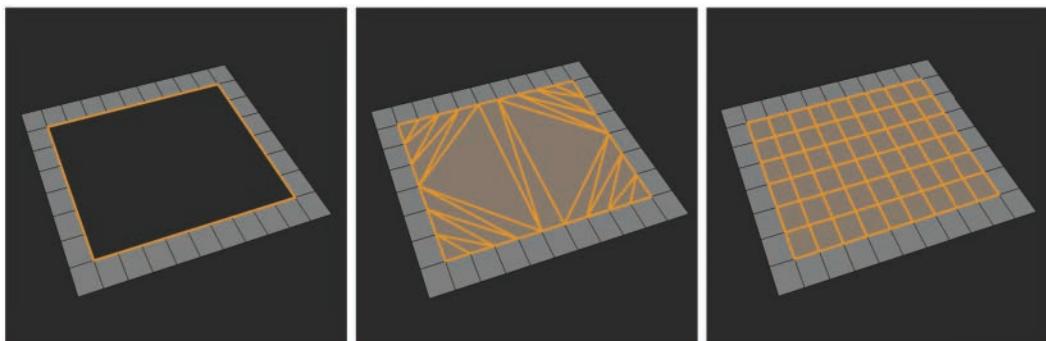


Figure 6.14 Mesh selection (left), Fill (middle), and Grid Fill (right)

To use the Fill tool:

1. Select the borders of the hole. (Sometimes, you can select them as a loop by pressing **Alt** and right-clicking.)
2. Press **Alt+F** to fill the hole with new geometry.

The Fill tool has a Beauty option within the Adjust Last Operation menu, which tries to give you a better result when generating the new geometry.

To use the Grid Fill tool:

1. Select the borders of the hole (number of edges must be even).
2. Press **Ctrl+F** to access the Face menu (Grid Fill doesn’t have a keyboard shortcut by default), and select Grid Fill to fill the hole with new geometry.

Grid Fill tries to create a new geometry made of four-sided faces (a grid). It gives you a couple of options in the Adjust Last Operation menu to rotate the pattern and get a cleaner geometry, depending on the shape of the hole that you’re filling. It also has a Simple Blending option that relaxes some of the grid’s surface tension.

Inset

The Inset tool is similar to the Extrude tool, but the new faces it creates are on the surface of the selection and do not change the surface’s shape initially. This tool generates a copy of the geometry that is an inset of the original selection; you can also use it to add height to the new geometry. This tool works only with faces (see Figure 6.15).

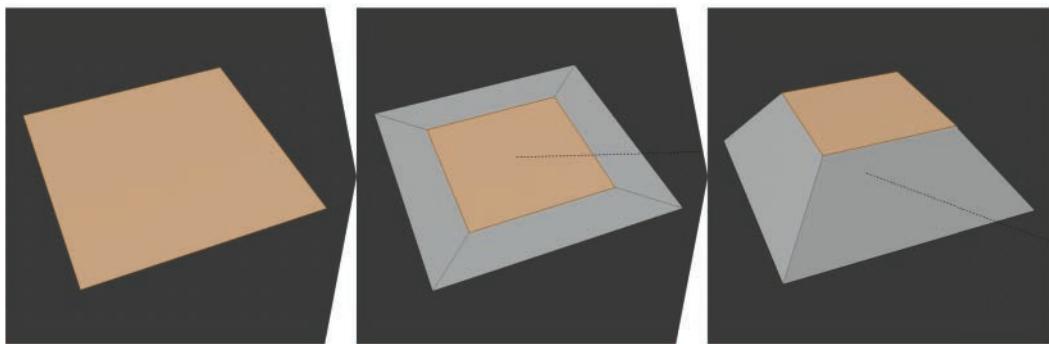


Figure 6.15 The Inset tool used in a face, with the inset amount defined first and then its height

To use the Inset tool:

1. Select a face or group of faces.
2. Press **I** to create an inset.
3. Drag the mouse to increase or decrease the inset's thickness. Press **Ctrl** while dragging to change the height of the inset.
4. Left-click to confirm the operation.

You'll find several interesting options for this tool. Boundary (press **B** while using the tool to enable or disable it), for example, takes into account the borders of the mesh when applying the inset. This option is very useful when you're working on a mirrored mesh and don't want the faces in the mirror's plane to be affected by the Inset operation.

Other options allow you to change how the thickness is calculated and define both the amount of the inset's thickness and its height (press **M** while using the tool). You have some options to outset instead of inset (press **O** while using the tool) or to apply the inset to each face of the selection individually (press **I** while using the tool). You can also select the outer or inner part of the inset after applying this tool, depending on which part you prefer to work with.

Tip

It may be confusing, but Inset in Blender is similar to Bevel in 3ds Max, whereas Bevel in Blender is similar to Chamfer in 3ds Max.

Join

The Join tool is used in Object Mode, not Edit Mode. With this tool, you can select two objects and join them so that they become a single object. This tool is the opposite of the Separate tool (discussed later in this chapter).

Keep in mind that Blender allows you to select and edit several objects simultaneously without having to join them in the same mesh, although some options are not available in such a case, like tools that create mesh between the two existing meshes.

To use the Join tool:

1. In Object Mode, select two or more objects. Determine the object you want to keep as the primary one and select it last to keep it as the active object. Attributes such as the center of the resulting object or its modifiers will be taken from the active object before you use the Join tool.
2. Press **Ctrl+J** to join the objects into a unique mesh.

Tip

Join and Separate (explained later) are similar to the Attach and Detach tools in 3ds Max.

Knife

Knife is a useful tool that allows you to cut through the surface of a mesh to divide it into faces and edges and to generate new geometry (see Figure 6.16).

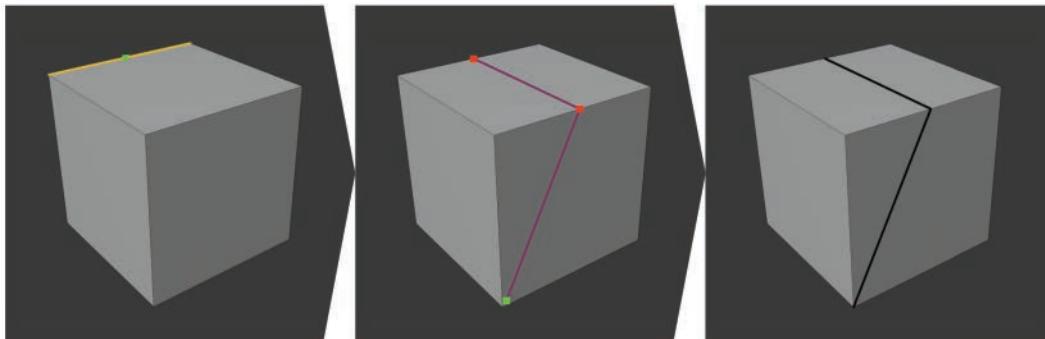


Figure 6.16 Using the Knife tool to cut through faces

To use the Knife tool:

1. Press **K** to start the Knife tool.
2. Left-click, move your mouse to define the cut line, and left-click again.
3. Repeat Step 2 until you're happy with the cut. The knife will show the points onscreen where it will add a vertex to the cut and, by default, the Knife tool will snap to vertices and edges. Press **Shift** to avoid snapping to vertices and edges and to do a free cut. Press **Ctrl** to snap to the center of the edges. Press **E** to start new cuts before accepting. Press **Z** to cut through the model on both sides.
4. When you're done with the cut, press **Enter** to confirm and apply the cut.

Knife Project

Knife Project is similar to the Knife tool, except that it uses another mesh to project the shape of the cut on a surface. That cutter shape is projected from your point of view onto the mesh and creates new edges (see Figure 6.17).

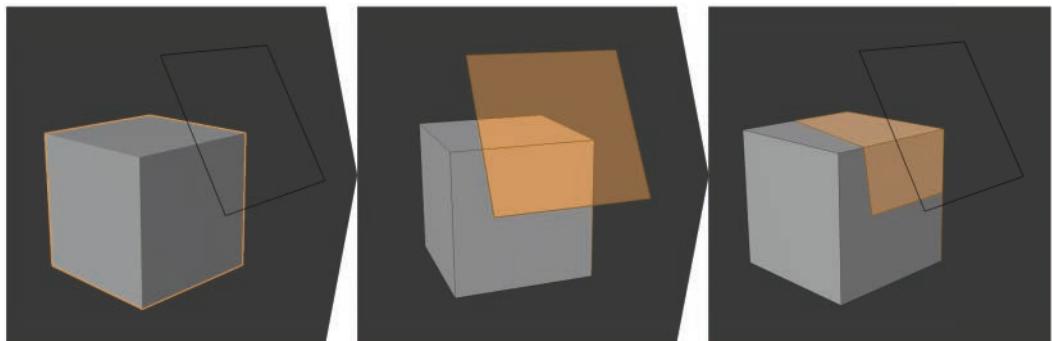


Figure 6.17 Model and cutter mesh (left), Knife Project operation from a given point of view (middle), and the result from a perspective where you can see the edges created on the surface (right)

To use the Knife Project tool:

1. Create a mesh with the shape you want to cut.
2. Hold down **Shift** and select (to add to the selection) the mesh in which you want to use the tool.
3. Change the point of view so that you can see the cutter mesh where the final cut should be.
4. Choose Knife Project from the Mesh menu on the 3D Viewport's header. (This tool has no keyboard shortcut assigned by default.)

The options for this tool let you make the cut go through the mesh, not just the side that is facing you.

Tip

You can select another object when in Edit Mode. Hold down **Ctrl** while left-clicking the other object, and that object is added to the selection as though you selected it before the current object. You can use this trick to select the cutter mesh even though you are in Edit Mode.

Loop Cut and Slide

The Loop Cut and Slide tool creates a cut through the entire ring you select, generating one or more new loops; then you can slide the new loop between the two adjacent loops (see Figure 6.18).

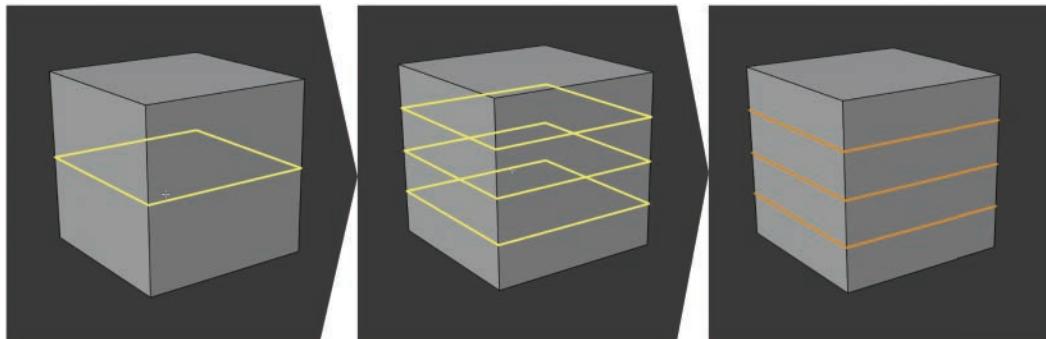


Figure 6.18 The Loop Cut and Slide tool used on the default cube

To use the Loop Cut and Slide tool:

1. Press **Ctrl+R**.
2. Move your mouse cursor around the model to define where you want to add the new loop. A preview appears in pink.
3. Roll your scroll wheel up and down to increase and decrease the number of loops to add.
5. Left-click to accept where you want the new loop.
6. Drag your mouse to slide the new loop or loops between the edges surrounding it.
7. Left-click again to confirm and apply the new loops. If you right-click instead, you cancel the sliding, and the new edge loop is perfectly centered when it is applied. (If you want to remove the new loop, you'd have to Undo with **Ctrl+Z**.)

The Adjust Last Operation menu provides some cool options for this tool. After you apply it, you can change the number of cuts and even their smoothness, as well as the falloff type of the smooth feature, to create curved shapes with the new geometry. Also, you can control the Edge Slide factor.

Tip

While you use the Loop Cut and Slide tool, the new edge loop average sits positioned between the previous and next loops. If you don't want the new loop to be averaged, but to adapt its shape to one or the other, press **E** (even) while sliding. A yellow line appears

perpendicular to the edge that you're sliding to show you the direction and limits of the slide, and a red dot marks the side to which the new loop's shape is adapting. Press **F** (flip) if you want it to adapt to the opposite side.

Make Edge/Face

The Make Edge/Face tool is very valuable because it lets you select two elements (only vertices or edges) and create an edge or face between them (see Figure 6.19). This tool has different effects, depending on the elements you select. If you select two vertices, the tool generates an edge between them. If you select three or more vertices (or two or more edges), the tool creates a triangle, a face, or an n-gon, according to your selection.

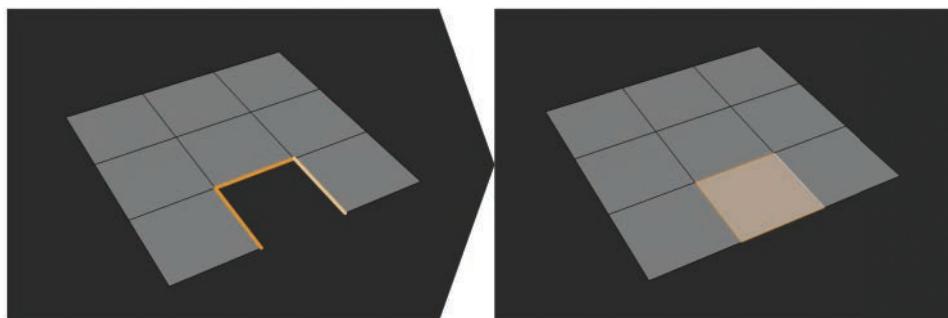


Figure 6.19 Using the Face tool to fill the hole between the four selected vertices

To use the Face tool:

1. Select two or more vertices or two or more edges. The vertices or edges need to be on the borders of the geometry; on the side where the face will be generated, no geometry should be connected to those elements.
2. Press **F** to create the edge or face.

Merge

With the Merge tool, you can select two or more elements and merge them into a single element. This tool is similar to the welding tools in other software. Merge can be used with vertices, edges, and faces.

To use the Merge tool:

1. Select two or more vertices, edges, or faces.
2. Press **M**, and select one of the options.

Blender offers several Merge options that have different results. With vertices, for example, you can usually decide where to merge: in the position of the first selected one or the last selected one, at the center of the selection, or at the point of the 3D cursor.

Collapse merges each of the groups of connected elements individually, so if they’re not connected, they won’t be merged. This option is useful if you want to get rid of a loop; you can select an edge ring and collapse it so that each edge is turned into a vertex. If you select two faces in different parts of a mesh and collapse them, for example, each of the faces is converted to a single vertex at its center.

Merge by Distance

One of the options that you can find in the **M** menu is Merge by Distance, which will merge vertices that are selected and are closer than a distance threshold that you can define in the Adjust Last Action menu. This used to be called Remove Doubles in old Blender versions.

After you merge elements, you can still change the merge type and where the elements are merged from the Operator panel. This feature is useful for experimenting, as it allows you to see the effects of the merge types. It’s also useful if you selected the wrong type accidentally, as it allows you to fix the mistake easily.

Offset Edge Loop

The Offset Edge Loop tool is especially useful for defining corners while using a Subdivision Surface modifier. It creates two parallel edges, one on each side of the originally selected edge (see Figure 6.20).

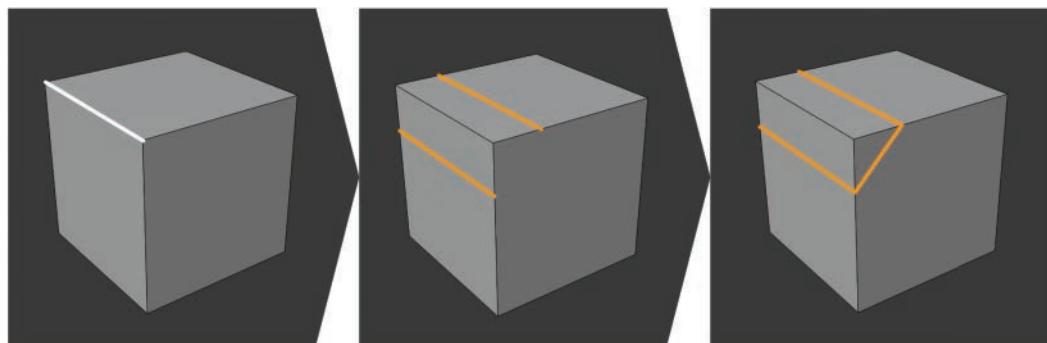


Figure 6.20 The original selection (left), two new parallel edges (middle), and the effect of the Cap Endpoint option (right)

To use the Edge Offset tool:

1. Select one or more edges.
2. Press **Ctrl+Shift+R**.

3. Slide the generated edges. You can press **E** to make the separation from the original even instead of calculating that separation automatically by averaging the distance from the next edge.

In the Operator panel, you can define the slide amount, as well as cap the endpoints (Blender connects the ends of both new edges if possible), even the distances, and set other parameters.

Poke

The Poke tool is simple but can be valuable. It creates a vertex in the center of each selected face and then creates an edge between that central vertex and all the vertices of the face (see Figure 6.21).

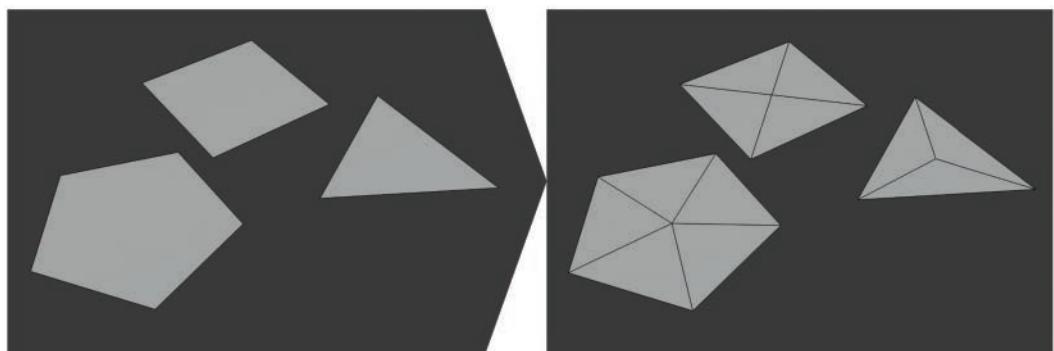


Figure 6.21 Using the Poke tool on different types of faces

To use the Poke tool:

1. Select one or more faces.
2. Choose the Poke Faces tool from the menus, or select it after pressing **Ctrl+F** for the Face menu. (The tool doesn't have an assigned keyboard shortcut.)

In this tool's options, you can define the height of the generated central vertex and define how that center is calculated.

Rip and Rip Fill

The Rip tool works only with vertices; it lets you rip apart the selected vertex or vertices and create a hole in the mesh. Rip Fill automatically fills the hole that you created (see Figure 6.22).

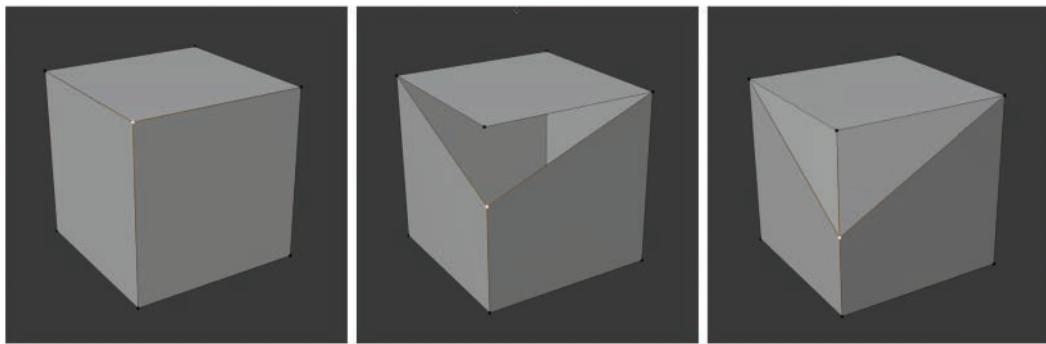


Figure 6.22 Selection (left), Rip (middle), and Rip Fill (right)

To use the Rip or Rip Fill tool:

1. Select one or more vertices.
2. Place your mouse cursor on the side of the vertex you want to take apart. That action defines the resulting vertex you'll displace after the rip.
3. Press **V** to use Rip or press **Alt+V** to use Rip Fill.
4. Drag your mouse to move the ripped vertex or vertices around.
5. Left-click to confirm the operation.

Separate

You can select a part of the mesh and separate it into a different object with the Separate tool.

To use the Separate tool:

1. Select the parts of the mesh you want to separate.
2. Press **P** to display a pop-up menu.
3. Choose the Selected option if you want to separate the selection, the By Material option if you want to separate the mesh into parts that use different materials (only if you've applied different materials to parts of the mesh), or the Loose Parts option if you want to separate the disconnected parts of the mesh. (You don't need to make a selection to use the Loose Parts option.)

Shrink/Fatten

Very simple but frequently very useful, the Shrink/Flatten tool scales the selected vertices, edges, or faces, depending on the direction of their normals. (A *normal* is what the orientation vector of a face is called in 3D.)

To use the Shrink/Flatten tool:

1. Select the parts of the mesh you want to modify.
2. Press **Alt+S** (similar to scaling) to launch the tool.
3. Drag your mouse to adjust the Shrinking or Fattening value.
4. Left-click to confirm the operation.

This tool also has some very simple options that allow you to adjust the shrinking or fattening value in the Adjust Last Action menu; it has some proportional editing options as well.

Slide

With the Slide tool, when you select a vertex, an edge, or a loop, you can slide the selection along the adjacent edges. Although you can use this tool with faces, it's usually more intuitive to use it with vertices and edges.

To use the Slide tool with vertices:

1. Select one or more vertices. (As a rule of thumb, use this tool with only one vertex at a time, as having more vertices selected makes the sliding unpredictable.)
2. Place your mouse near the edge where you want to slide the vertex.
3. Press **Shift+V** to launch the tool.
4. Drag your mouse to slide the vertex. Blender displays a yellow line to let you see where you can move the vertex.
5. Left-click to confirm the new vertex position.

To use the Slide tool with edges:

1. Select the edge or edge loop you want to slide.
2. Press **Ctrl+E** to access the Edge menu, and choose the Edge Slide option.
3. Drag your mouse to slide the edge or edge loop.
4. Left-click to confirm the operation.

The options in the Operator panel for the Slide tool are very simple. They allow you to adjust the distance and direction in which the selected elements slide along their edges.

Tip

A quicker way to slide, whether you're working with vertices, edges, or faces, is to press **G** twice. When you're using the Slide tool, the same options are available as when you slide the Loop Cut and Slide tool, such as pressing **E** and **F** to adapt the shape to the adjacent edge loops.

Smooth Vertex

Smooth Vertex does just what its name tells you: smooths the selected vertices' shape. This tool is useful when you have some unwanted bulges or just want to distribute in a more uniform way the vertices that you created in a mesh.

To use the Smooth Vertex tool:

1. Select a group of vertices, edges, or faces.
2. Select the Smooth Vertex tool from the Vertex Menu. (The tool has no keyboard shortcut.)

The options for this tool let you control different parameters to control how the smoothing is performed.

Solidify

The Solidify tool adds thickness to the selection. It works only with faces.

To use the Solidify tool:

1. Select the faces to which you want to add thickness.
2. Press **Ctrl+F** to access the Face menu, and choose the Solidify option.

When you apply Solidify to a face or group of faces, you can use its options in the Operator panel to adjust their thickness.

Spin

The Spin tool lets you pick a vertex, edge, or face (or a group of these elements) and extrude them around the 3D cursor from the current point of view (see Figure 6.23).



Figure 6.23 Using the Spin tool with a vertex around the 3D cursor

To use the Spin tool:

1. Select one or more vertices, edges, or faces.

2. Place the 3D cursor where you want the center of the resulting circle to be.
3. Position the point of view to define the orientation of the spin.
4. Press **Alt+R** to use the Spin tool.

Spin gives you options for defining the angle of the spin and the number of steps (extrusions during the spin) and for adjusting the center and axis from which the spin takes place. Also, the Dupli option makes duplicates of the resulting geometry instead of extrusions.

Split

The Split tool disconnects a selected part of the mesh from the rest of the mesh (see Figure 6.24). It works best with faces. (With vertices and edges, the effect is similar to duplicating them if they are parts of a face.) When the selected part is disconnected, you can move it freely to any other location. Split is different from the Separate tool in that Split keeps the mesh in the same object rather than creating a new object from it.

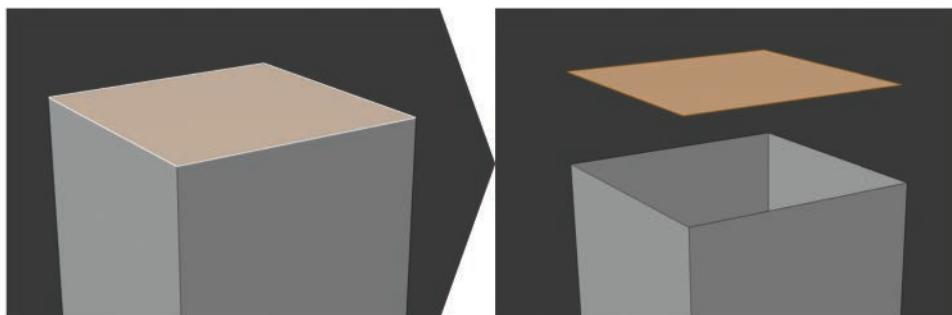


Figure 6.24 Splitting a face from the default cube

To use the Split tool:

1. Select the faces you want to disconnect.
2. Press **Y** to use the Split tool. Alternatively, press **Alt+M** (opposite to Merge tool) to see the advanced options for splitting.

Subdivide

As its name implies, the Subdivide tool subdivides geometry. It works with edges and faces. With this tool, an edge is divided in half, generating a new vertex right in the middle. A face is divided into four new faces, and if you select an edge ring, Subdivide generates a loop that divides all the edges of the ring right through the center of each edge. You can also increase the number of divisions (cuts) when using this tool (see Figure 6.25).



Figure 6.25 Subdividing a face with three cuts

To use the Subdivide tool:

1. Select the geometry you want to subdivide. (The minimum is two connected vertices that are equal to one edge.)
2. Right-click and choose the Subdivide option from the contextual menu.

You can define the number of divisions and their smoothness, depending on the surrounding geometry. You can also generate triangles around the subdivisions to prevent the creation of n-gons and apply fractal noise patterns to the resulting geometry.

Using Modeling Add-Ons

After exploring the main modeling tools, in this section you'll see a couple of add-ons that you can enable (they come bundled with Blender) from User Preferences. These add-ons will help in different situations.

Working with LoopTools

When the LoopTools add-on is active, LoopTools is accessible from the Sidebar in the 3D Viewport and at the top of the contextual menu in Edit Mode (see Figure 6.26).

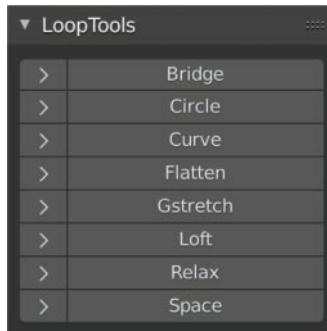


Figure 6.26 The LoopTools add-on tools in the Sidebar of the 3D Viewport

This add-on provides some interesting and useful modeling tools that I encourage you to check out. These tools speed the modeling process. Here's what they do:

- **Bridge:** This tool was present in the LoopTools add-on before it was implemented in Blender. The version included with Blender (called Bridge Edge Loops) is not exactly the same; its options are slightly different, though they provide similar results. The Bridge tool lets you select edges or faces and create new geometry that forms a bridge between them.
- **Circle:** This tool places your selected elements (vertices, for example) in a perfectly circular disposition.
- **Curve:** Select several vertices along a loop, and this tool will move the surrounding vertices in the loop to create a curve accommodating the shape between the selected vertices.
- **Flatten:** Select faces, edges, or vertices (a minimum of four), and this tool moves them to the same plane. Suppose that you created a surface mesh that's not planar. With this tool, you could select the entire surface and use the Flatten tool to make the surface flat.
- **Gstretch:** Gstretch lets you use Grease Pencil or Annotation strokes to change the shape of a loop.

Annotations Basics

To use the Annotations tool (which is used mainly for annotations in the 3D Viewport), press and hold **D** while you left-click; then drag to draw. Press and hold **D** while you right-click; then drag to erase the strokes. In the 3D Viewport's Sidebar, you'll find options for deleting or changing the parameters of the Annotations layers.

- **Loft:** This tool is very similar to Bridge Edge Loops but goes a little farther: It allows you to bridge more than two edge loops. Suppose that you have three circles. With Loft, you can select all of them and create a bridge from the first to the last, passing through the middle circle. The tool gives you the chance to control the shape in the middle of a bridge operation.
- **Relax:** This tool smooths the selection and prevents sharp shapes. It's similar to the Smooth Vertex tool, but it respects the position of the border elements and relaxes only what's inside.
- **Space:** This tool spaces your selection so that the distances between all the selected elements are the same. Space is useful for technical modeling when you have several divisions made by hand and need them to be equally spaced. Space works best when used in edge loops.

Working with F2

The F2 add-on adds some functionalities to the **F** key to increase the speed at which you can create new faces.

One of those functionalities allows you to select a corner vertex and press **F** to generate a new face from that corner. A different use is tricky to explain, but you can see it in Figure 6.27: Select two vertices and press **F** repeatedly to create new faces very quickly and fill the holes.

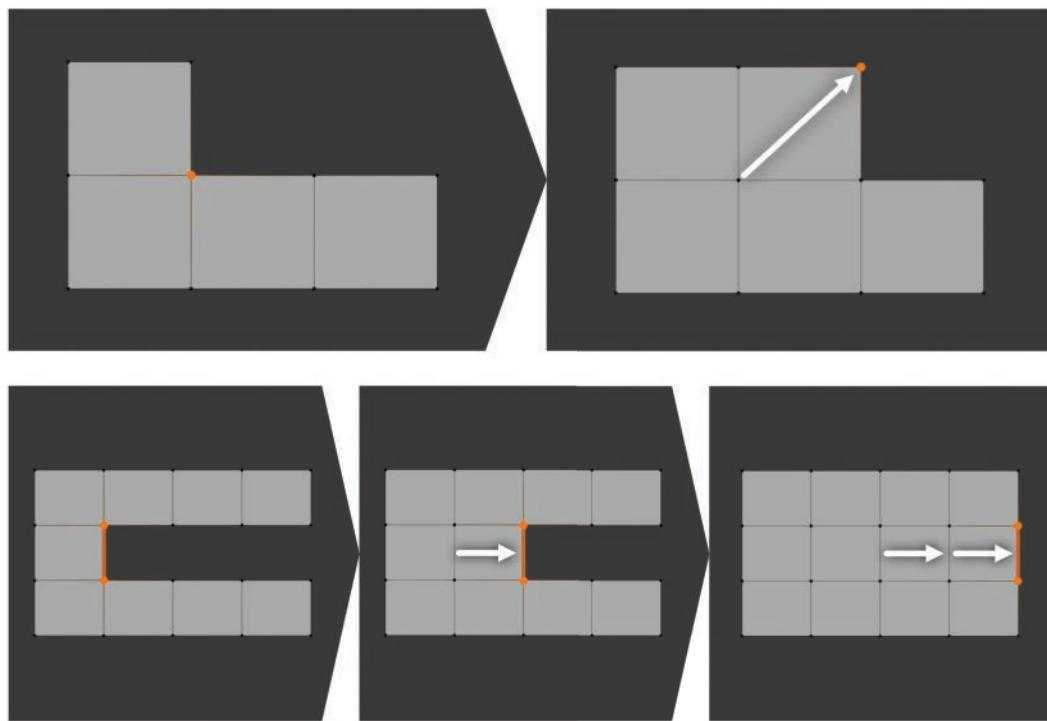


Figure 6.27 Each arrow in the image represents a press of the **F** key, and each row of images indicates the effect of the F2 add-on depending on the selection and surrounding elements to which you apply it.

Tip

When you enable the F2 add-on in User Preferences, expand its menu and you'll be able to activate the Auto Grab option. When Auto Grab is enabled, creating a face from a corner vertex automatically selects and grabs the generated vertex, allowing you to relocate it quickly. This option is especially handy when you're retopologizing a model.

Using Other Useful Blender Options and Tools

Some other tools and options are not modeling tools as such but can help with certain tasks and visualization while modeling.

Auto Merge

The Auto Merge feature is very helpful in modeling. When you activate it, if you place a vertex in another vertex's location, Auto Merge automatically merges those vertices.

If you activate the Snapping tool as well (or just set it to Vertex Snap and press **Ctrl** when you want to merge), when you move one vertex near another one, the vertex you're moving snaps to the second vertex's location, making merging very fast and straightforward.

This technique is similar to using welding tools in other 3D packages. You can activate Auto Merge from the right corner of the 3D Viewport's header (see Figure 6.28).

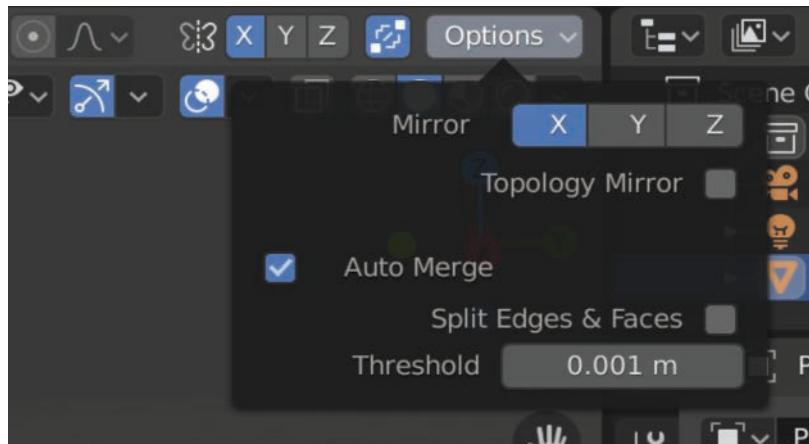


Figure 6.28 Auto Merge and its options

Tip

While in Edit Mode, you can find the option for Threshold inside the Options tab of the 3D Viewport's Sidebar or the Options button next to the Auto Merge button. This distance is the minimum distance at which you need to drop a vertex from another to make the vertices merge when Auto Merge is enabled. Increase that value if you want to Auto Merge vertices without using the snapping tools; the vertices will merge when you drop them at a distance smaller than the threshold.

Global and Local View

This tool is a very interesting one that helps you focus on the objects you're working on, especially in big scenes with many objects. Essentially, you can select an object or group of objects and isolate them, hiding all the rest of them and focusing on the selection.

You can switch between Global and Local View by pressing **/** in the NumPad, or choosing the option from the View menu on the 3D Viewport's header. It's similar to Hide and Reveal but quicker, and both tools can be combined.

Caution

Keep in mind that while you are in Local View mode, you won't be able to use certain options, and some things may not behave as usual. If you change the visibility of an object from the Outliner but that object is not a part of the current Local View, for example, the visibility options won't have any effect. So remember to go back to the Global View when you're done with the Local View.

If you notice that something's off, or you can't find some options (which may be available only in Global View), you can check whether you're in Local View. When you're in Local View, Blender says so in the top-left corner of the 3D Viewport; you'll see (Local View) next to the View's name.

Hide and Reveal

Hide and Reveal is very useful. You can select a part of your mesh and press **H** to hide it so that it's not in the way while you work on other parts of the model that may be hidden. When you're done, press **Alt+H** to reveal all the parts of the model that you've hidden. Also, you can hide what is unselected by pressing **Shift+H**.

This feature is not only helpful for hiding certain parts or making them visible, but also allows for selective adjustments. Suppose that you want to create a loop cut. Normally, doing so affects the entire edge ring. But if you want the loop cut to affect only part of the edge ring, you can hide some of the ring's parts, and the tool affects only the parts that remain visible. Hide and Reveal also works this way with most of the modeling tools, so take advantage of it!

Snapping

Just as in Object Mode, you can activate the snapping tools on the 3D Viewport's header and select the type of snapping you want: Vertex, Face, Increments, and so on. If the Snap tool is active, when you move things around, they snap to close elements, which is really useful when making alignments. In addition, while pressing **Ctrl**, you can move the selected elements freely. Alternatively, if Snap is disabled when you press **Ctrl**, things snap while you move them.

X-Ray

This option may also help you while visualizing your models, especially in complex ones that can have pieces within other pieces. You can find the X-Ray options in a button right next to the Viewport Shading buttons in the 3D Viewport's header. A slider to control the effect amount can be found in the Viewport Shading options.

While you're in Solid viewport shading mode, X-Ray will allow you to turn objects into semitransparent surfaces, and you will be able to see through them.

If you use the X-Ray slider in the Viewport Shading options, it will increase or decrease the effect's transparency.

In Wireframe viewport shading mode, X-Ray option is enabled by default, with its slider at 0. If you increase the slider value, you'll end with opaque surfaces and see only the wireframe of the surfaces facing the camera, which can be useful at times.

Summary

In this chapter, you learned about the main Blender modeling tools: what their effects are, how to use them, and how to adjust their options when they are applied to your mesh. This information should give you a head start on modeling, and you should be able to create simple models and use these tools to modify and shape your 3D meshes. You also learned that you can perform most of these actions by using keyboard shortcuts. (You can also access these tools through the menus and reference their keyboard shortcuts from those menus.) Remembering all the shortcuts may be difficult at first, but in the long run, you'll be thankful for knowing them, because they allow you to work a lot faster!

Exercises

1. Try using every modeling tool in this chapter on some simple objects.
2. Model a very simple object that Jim can use in his adventures (such as a flashlight). Figure out which tools to use, how to use them, and in which order to reach the final result.

This page intentionally left blank

Character Modeling

It's finally time to start modeling Jim! In this chapter, you learn about topology and some of the most popular modeling techniques; then you'll set up the reference images you created in Chapter 5 so that you can model over them. Finally, step by step, you model every part of Jim's body. This phase is one of the most crucial phases of the project because it defines the shape and look of the character with you're going to work with in the chapters to come.

What Is Mesh Topology?

Mesh topology is the way in which edges are distributed along the surface of a model. Two surfaces can have identical shapes but different topologies. Topology is important because it affects how the mesh subdivides (using a Subdivision Surface modifier) and deforms. It is especially critical for animated characters. When a character moves, the model is deformed. A good topology ensures that deformations look natural; otherwise, the mesh will pinch, stretch, or just deform incorrectly and look weird.

In Figure 7.1, you see two topology examples—one good and one bad. (These examples are exaggerated and are meant just to illustrate how a shape can be created with very different topologies.) In the example on the left, the topology is poor: Most of the edge loops run only vertically and horizontally, and they don't really adapt to the face's shape, which will certainly cause problems when, for example, you want to make the character open its mouth. In the example on the right, the topology is much better: Edge loops flow with the face's shape and define it correctly.

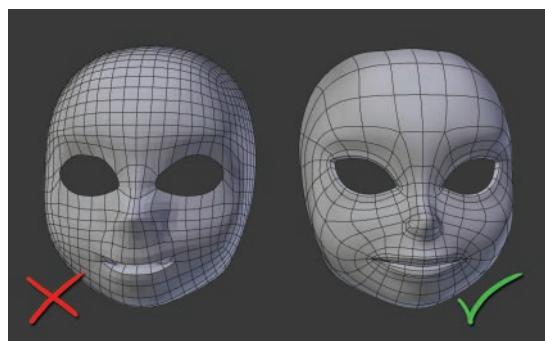


Figure 7.1 Examples of poor topology (left) and good topology (right)

Think of *topology* as being the skin and the muscles of a face or other body part. Depending on how they will deform, they need to follow the shapes of the model; otherwise, the skin will have serious problems.

Here are some things you should keep in mind to make sure that you have a good topology:

- **Use four-sided faces (quads) as often as you can.** Avoid the use of triangles or n-gons unless you really have a good reason to use them. Triangles and n-gons, when used carelessly, can generate pinches on the surface when the model is subdivided and deformed. This doesn't mean that you can't use triangles and n-gons—only that you need to know where they can be used without causing problems. These elements tend to be problematic on curved surfaces, but on a flat surface, you generally can use them without issues.
- **Use squares rather than rectangles.** Overall, if you're working with organic shapes, four-sided faces won't have the same length on all four sides, so try to avoid using really long rectangles, as they'll be difficult to manage during deformations and other stages of the animation process that follow modeling. Try to keep your topology as uniform as possible.
- **Keep an eye on areas that require complex deformations.** Some parts of a model are more complex than others, not just because of their shape, but also because when they're animated, you'll probably need a wide range of movement for them. The edge flow should be especially good in those areas to make sure the model will deform correctly later. The eyelids, shoulders, knees, elbows, hips, and mouth are some areas to which you should pay close attention to a good topology and perhaps even include more faces to provide additional geometry for more-defined deformations. Generally, for simple characters, using three edge loops in the articulations works quite well: a loop for the articulation itself and one on each side of it.
- **Keep a low poly count.** *Poly count* refers to the number of polygons (usually measured in triangles, as any polygon can be divided into triangles) that form a model. More polygons mean more work in all the stages of modeling, which could result in difficulties if you have to change something. As a rule of thumb, keep the number of polygons to the minimum needed to achieve the shape and the level of detail you want for your model.
- **Pay attention to density and tension.** When you're using a Subdivision Surface modifier, it's very important to keep an eye on density and tension. Subdivisions tend to smooth the surface, but sometimes, you want to create a corner. In that case, you need to add more density (more geometry). If you have low density, you have more tension; the shape becomes stretched when subdivided because there's not enough geometry to define the shape, so the edge loops move further from their original positions. Use density and tension to your benefit. Remember that more density defines the shape better because it reduces tension (but also requires more geometry).

- **Follow the shapes.** Your edges should flow with the shapes. Around the mouth, for example, you should have circular loops that create a smooth deformation when the character opens its mouth or talks. If those loops are vertical and horizontal lines, chances are that your character will look cubical and quite strange when it opens its mouth—a usual pitfall when you start modeling and don't yet have a good grasp of your edge flow.

Choosing Modeling Methods

Modeling can seem rather technical (and it really is), but it offers a lot of freedom and creativity. Quite a few methods and techniques are available to explore and use as you need them. Some of them may be more comfortable for you to work with than others, or you may want to pick a method based on the type of model you're working on. This section presents some of the most popular methods you can use.

Box Modeling

Box modeling is based on the premise that you can model anything from a simple primitive form, such as a cube, sphere, or cylinder. Don't be fooled by its name: The *box* in *box modeling* means only that the most essential shape can be the base for any kind of object, not necessarily a box. The idea behind this method is that if you start from a primitive, you can divide it, extrude it, and otherwise modify it to reach the shape you want.

With box modeling, you start from something very basic and, little by little, add details to it—first creating the biggest, most important shapes and then adding the smaller shapes on top of that base. From the beginning, your model has the basic shape it needs, so you have to add only as many details as you want. You can compare box modeling with sculpting in mud; you start with a raw general shape and then gradually add details. You move vertices around, add modifiers to achieve various effects, and smooth the surface with a Subdivision Surface modifier if needed.

Poly to Poly

Poly to poly (also called *poly2poly*) is about drawing a shape one polygon at a time. You create vertices and edges, extrude them, and join them to make faces, constructing the model as though you were building a brick wall. Again, you add a Subdivision Surface modifier to smooth the geometry you created.

Sculpt and Retopology

Although box modeling and poly to poly are probably the most traditional modeling methods, sculpting came to the 3D world just over 10 years ago and is now widely used, especially for organic models. With sculpting, you create a basic shape; topology doesn't matter much. Then you sculpt it, adjust the shape, and add a lot of detail to it. After that, you can use the *retopo* (short for *retopology* or *retopologize*) process, which involves creating the final topology for the model with a poly-to-poly method, but the new geometry snaps to the shape you sculpted initially.

This method is by far the most creative modeling method, and artists love it. It allows you to focus on the shapes of your model and not have to think about a lot of technical stuff. Only when you're happy with your shapes do you worry about topology, which you create very easily because you don't have to think about whether the shapes will look right; you already have them covered in your sculpture!

Automatic Retopology and Remesh

Recently, automatic generation of topology is getting more and more popular, as it can save a lot of tedious work and time. Software and add-ons allow the use of this technique, which essentially analyzes the shapes and generates a decent topology.

Depending on the case, however, you may have to do a lot of manual retouching to create production-ready geometry. Some of these tools allow you to draw lines over the model so that you can guide the topology that will be generated.

Automatic retopology is not to be confused with remeshing techniques, which usually don't analyze the shape—only the volume. Although remeshing can generate a mesh that is a good starting point for completing with manual retopology, it is more useful to re-create a mesh that has been highly distorted while sculpting; then you can continue the sculpting process with comfort.

You can think of remeshing as being a tool to re-create shapes with an uniform geometry while sculpting to avoid working on distorted and stretched geometry, which can be tricky and cause problems. You can use automatic retopology later, when the sculpting is finished and you want to convert the model to its final mesh.

In recent versions of Blender, you can find remeshing tools, both as modifiers and as options in the Mesh tab of the Properties Editor.

Modifiers

Using modifiers is not a method in itself, but modifiers play a big role in modeling in a lot of cases. Suppose that you're modeling a character. You can model one side of it and use a Mirror modifier so that the opposite side simultaneously builds itself, mirroring the one on which you're working. You can speed your work by using modifiers. If you have to work on a complex curved model, for example, make it in a plane and then deform it along a curve with a modifier. Modifiers help a lot, and in some cases, they're essential to the construction of your model, so they deserve a mention here.

The Best Method

If you thought that this section would tell you the best method to use in modeling, I'm sorry: There is no *best* method. Each person is more comfortable with certain methods than with others, depending on his or her skills and spatial vision, the particular project, and so on. Some people switch among methods as needed. Are you modeling a car? Use box modeling. Creating a monster? Try Blender's Sculpt and Retopo features.

The most powerful thing to keep in mind is that you can mix all possible methods (only some of the most popular ones for character creation are mentioned here) in the same model! You can model one part with box modeling, another with poly to poly, and the most organic parts with sculpting. You can even adjust the shapes of your character by using Sculpt Mode when you have the basic topology in place and then go back to working with box modeling.

The possibilities are endless, and that's why 3D modeling—even though it requires some technical knowledge—can be a very enjoyable and creative process.

Setting up the Reference Images

Before you start modeling, you need to load into your scene the character designs you created in Chapter 5, “Character Design,” as reference images to use as guides while modeling. Using references definitely helps you define the right proportions of your character.

There are different ways of loading reference images in Blender:

- **Loading images for side-by-side referencing:** You can use the Image Editor to load reference images and keep it open on the side, so you can keep an eye on it while modeling. This technique is especially useful when you don't have really accurate references and have to rely on your instinct, eyeballing the proportions and shapes.
- **Using reference images within the scene:** You can create reference images as planes within the scene and place them where they're supposed to be seen from (such as placing a side View aligned with the Y-axis so you can see it when you look at it from the side). Many settings can be controlled in this type of reference images, and I'll get to them next, as this method is the one you'll be using with Jim.
- **Using background images on the camera:** This property of cameras allows you to do something similar to reference images within the scene, but these background images are seen only from the camera view. This technique is useful when you need to match an object with a real photo that has a specific perspective.

To load the reference images in the 3D scene, you only need to drag from the folder and drop them. They will be placed exactly where you drop them as *empties*—reference objects that never show up in the render. You can use empties as images or points in space to mark positions or serve as part of a hierarchy where you don't want to have an actual mesh.

I recommend following this procedure:

1. Press **NumPad 1** or use any other method to place the 3D Viewport in a front orthographic view.

2. Drag and drop the head's front reference image to the 3D Viewport.
3. With the image selected, press **Shift+S**, and move the selection to the 3D cursor to ensure that the image is precisely centered. (Obviously, this move works only if the 3D cursor was at the scene's origin, so make sure that it's placed at 0, 0, 0 in the world before this step.)
4. Now switch to a side orthographic view, and repeat Step 3 for the side reference image. You should have something similar to what's shown in Figure 7.2.

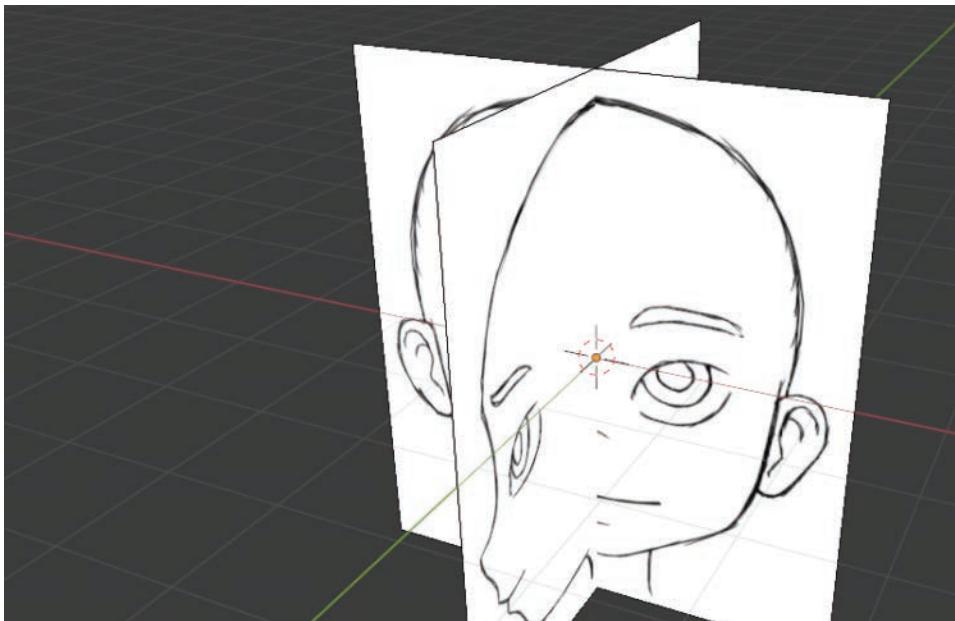


Figure 7.2 The reference images are loaded in their correct views.

Now you can set up the reference images so that they're more comfortable to use:

1. Select one of the reference images.
2. Go to the Object Data tab of the Properties Editor. You'll find several parameters that let you control how this image will be shown (see Figure 7.3).
3. Start by enabling Transparency and reducing Opacity to 0.2, which will make the image easier on the eyes.
4. In the Depth parameter, select Back. When you select this setting, the image will always be seen behind the objects in the scene, regardless of their positions, setting the image in the 3D Viewport's background.

5. Next, for the Side parameter, select Front. This option defines whether the image is one- or two-sided and, if it is one-sided, which side you can see the image from. This setting prevents the image for the front of the head to be also seen from the back.
6. (Optional) Enable Display Orthographic and disable Display Perspective to set the image so it's visible only from orthographic views and invisible from perspective views. Sometimes, you prefer a different setup, so you know what to do to enable one option or the other, or both!
7. (Optional) Another interesting option to enable if you want to make sure that you see the image only from a given point of view is Display Only Axis Aligned. This option makes the image visible only when the viewpoint is exactly in front of the image. In this case, the side reference will be visible only when you switch to the side orthographic view; if you rotate the view just a little, the image will disappear.

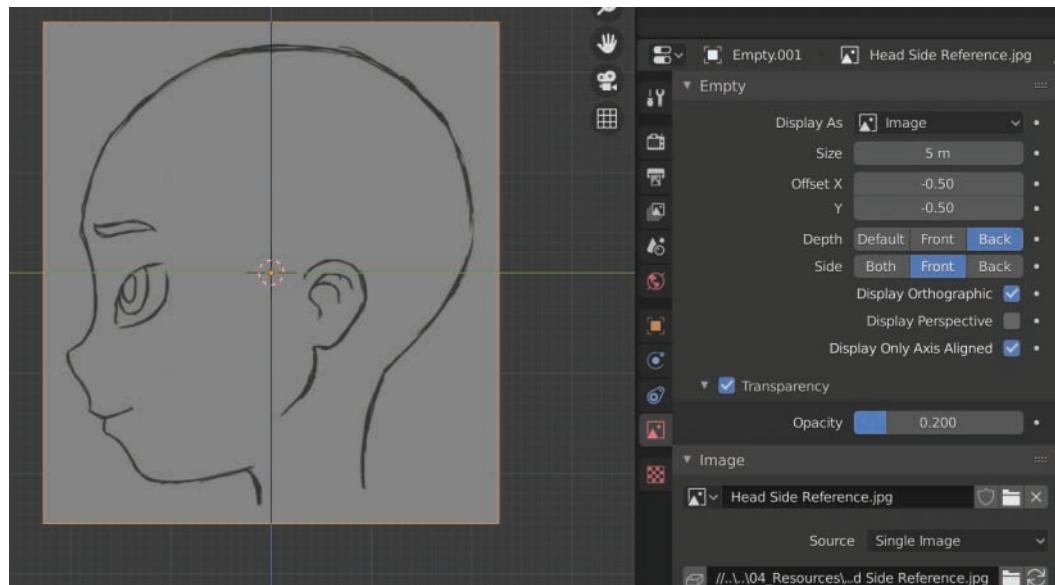


Figure 7.3 The reference image's parameters on the Object Data tab of the Properties Editor

Caution

It's very important for the reference images to be aligned properly. In this case, all the images have the same height, and the front and back images are centered. In some cases, however, the images can have different sizes or margins, so you may need to adjust the scale and position of the reference images to align them accurately. To help with those adjustments, you can use a very simple mesh that allows you to compare the size and position of the images with those of the 3D model. Just placing a simple cube or sphere that shows how the size and volume fits all the images' points of view would be very

helpful. Remember that the drawings are not physically accurate, so leave some space for modifications to the images, as it's practically impossible to make the 3D model fit the 2D images perfectly.

3D vs. 2D

Keep in mind that 2D drawings have, by nature, incongruities in volume and shape, so it's perfectly normal for the 3D model not to totally fit the image references, or at least not to fit all views. 2D references are just that: references. Don't stress if your 3D model doesn't fit the references in all the views with precision. The most important thing is to focus on the shape of the 3D model and whether it looks good, regardless of its accuracy with the reference images.

One more thing you can do is organize the reference images into a collection:

1. Select both images, press **M**, and create a new collection named References. The images may be difficult to select, as now they're set up in such a way that they're visible only from very specific points of view. You can still select them easily from the Outliner.
2. It's also a good time to rename the images with useful names, such as Front_Head_Reference and Side_Head_Reference. You can complete this step from the Outliner or by double-clicking the object's name, selecting the Object tab of the Properties Editor, or even pressing **F2** in the 3D Viewport.
3. Click the funnel button on the Outliner's header to display the Filters menu. If you click that menu, at the top you'll find the Restriction Toggles, which enable or disable icons on the right side of the Outliner, in line with each object. Enable the arrow icon. (If you hover your mouse cursor over the icon, you'll see a tooltip that reads Selectable.)
4. On the right side of each object, you see that arrow icon next to the eye icon. The arrow icon lets you make objects or collections selectable or unselectable. Click the arrow icon for the References collection. Now this collection appears to be empty, which means that you can't select objects within it. To make those objects selectable, click that icon again.

This technique is very useful. You can not only hide or show the references (individually if you click the eye icon next to each reference image and for all references simultaneously if you click the eye icon next to the collection that contains all the reference images), but also make them unselectable, which prevents you from moving them accidentally or getting in the way of selecting the meshes that will be on top of them.

Now the reference images have been set up and aligned properly. You can start working on the model!

Modeling the Eyes

Each person is most comfortable beginning to model in a different way. Some people prefer to start with the face; others, with the body. For Jim, I recommend that you start with the eyes, because that way, you'll be able to use them as references when modeling the rest of the face—especially the eyelids, as you'll be able to align those features with the eyes.

Creating an Eyeball

Jim's eyes are drawn in an animation/manga style (not completely round). The eyes are basically spherical, but to make them a little more realistic, you can create the cornea with the pupil beneath it. Figure 7.4 shows the process step by step. Here's what happens in each step:

1. In Object Mode, create a UV Sphere, and in the Operator panel, set it to have 16 segments and 16 rings.
2. Rotate the sphere 90 degrees on the X-axis so that the poles are positioned at the front and back. This way, you'll be able to use the circular edge loops on the front pole to build the pupil.
3. Enter Edit Mode by pressing **Tab**; then select the two edge loops of the front pole and the pole vertex. A quick way is to select the vertex in the pole and then press **Ctrl+NumPad +** to grow the selection twice.
4. Press **Shift+D** to duplicate the geometry you selected in Step 3, and move the new geometry out a little (or, for now, hide it by pressing **H**). Later, this geometry will become the eye's cornea.
5. Select the same geometry you selected in Step 3 from the sphere, and extrude it into the eye by pressing **E**.
6. Scale the selected geometry to invert its curvature by pressing **S**, pressing **Y** to scale on the Y-axis, and then typing **-1**. Press **Enter** to confirm. Then adjust the position of this geometry on the Y-axis so that it fits into the eye in case the geometry moved out of alignment when you inverted it.
7. Select the outer edge loop of that inverted circular area, and bevel it (**Ctrl+B**) to add some density and create a defined corner when you add the Subdivision Surface modifier later.
8. Press **Alt+H** to unhide everything and see the cornea you previously detached. Move it back to its place, and scale it up or move it a bit if necessary to cover the gaps that you made when beveling the borders of the pupil.
9. Add a Subdivision Surface modifier with two divisions; then right-click in the 3D Viewport and choose Shade Smooth from the contextual menu so the faces don't look flat.

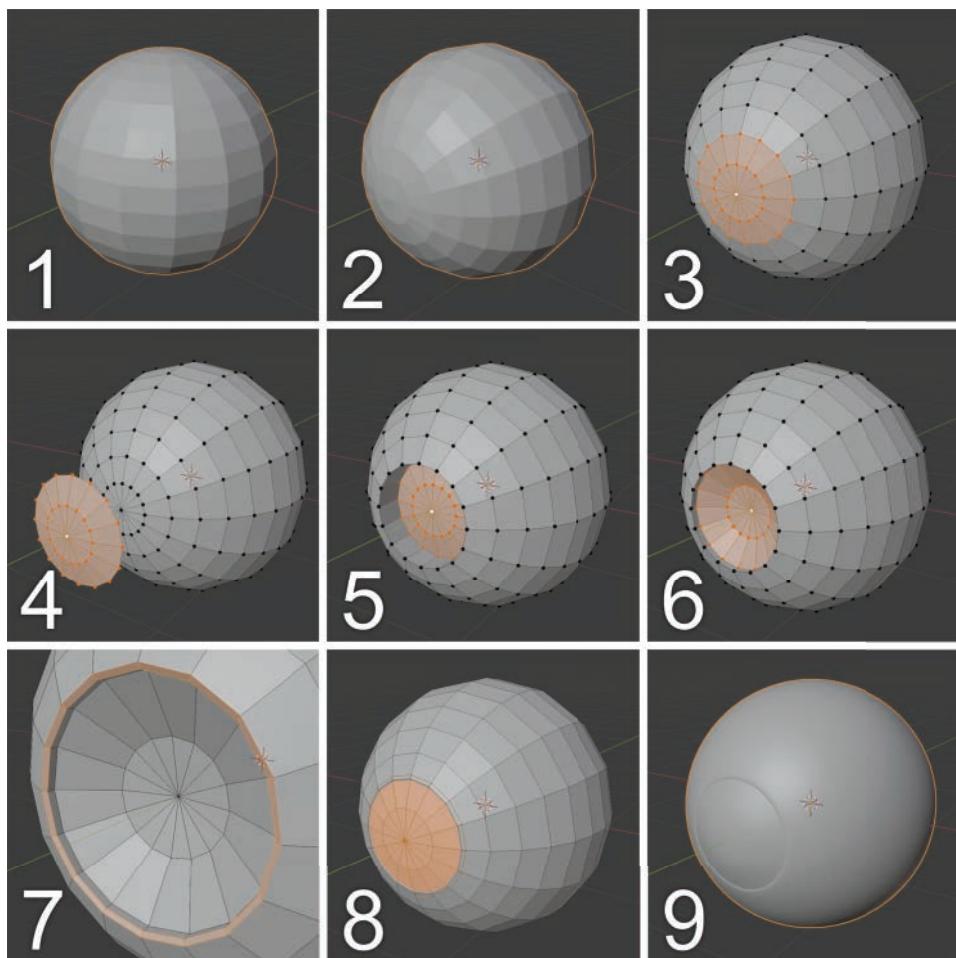


Figure 7.4 Steps to follow to model the eye

Note

For this exercise, the lamp and the camera are in different collections. Press **M**; select the option to create a new collection, and name it Hidden Stuff, for example. After that, you can go to the Outliner and hide that collection by clicking the eye icon to its right. (You'll learn more about collections in future chapters.) Alternatively, you can delete the lamp and the camera to get them out of the way. If you do, though, you'll need to re-create them when you need them again in future chapters.

Using Lattices to Deform the Eyeballs

Now you have one eyeball, but it's completely round; in the reference designs for Jim, the eyes are more oval-shaped. Fortunately, Blender has a tool called Lattice that lets you deform geometry and then maintain that deformation when you rotate the geometry, which is exactly what you need for these eyeballs. You could go ahead and scale the eye on the Y-axis to make it flatter, but when the eye moves during animation, it won't fit the eye socket. Figure 7.5 shows the effects of the Lattice modifier.

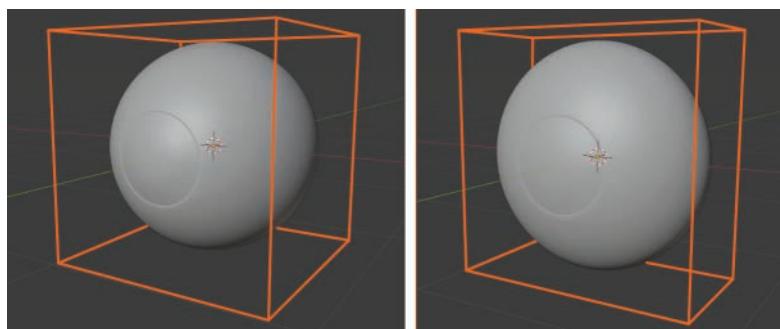


Figure 7.5 A lattice (left), which squeezes the eyeball when its points are moved in Edit Mode (right)

Follow these steps to apply a Lattice modifier to the eyeball:

1. Press **Shift+A**, and create a lattice.
2. Scale up and place the lattice so that it covers the entire eyeball (in Object Mode).
3. Select the eyeball, and add a Lattice modifier to it. It's best to add it on top of the Subdivision Surface modifier; that way, the lattice will deform the low-resolution mesh, and the deformed mesh will be subdivided afterward so that it works more smoothly. Remember that you can reorder a modifier by clicking the arrows in the top-right corner of the modifier.
4. From the Lattice modifier options, in the Object field, select the name of the lattice you created in Step 1. Alternatively, you can click the eyedropper icon in the Object field and then click the lattice in the 3D Viewport.

Caution

Be careful when using a Lattice modifier. When you're adapting it to the mesh you want to distort, you must scale it in Object Mode. Edit Mode is meant to create the distortion, but it will use its Object Mode scale as a reference for the nondistorted shape.

5. Select the lattice; press **Tab** to enter Edit Mode; and see how, as you move its vertices, the eyeball deforms accordingly.
6. Select all the vertices (press **A**), and scale them down on the Y-axis to make the sphere flatter.
7. Pick the outer side's edges to better align the eye with the side-view reference image.
8. Exit Edit Mode, and rotate the eyeball object to test how it deforms. It should rotate while keeping the lattice deformation in place, which is exactly what you need. To test this, it's useful to press **R** twice to start orbiting, which allows you to rotate an object in all axes at the same time by moving the mouse. When you're done testing, press **Esc** or right-click to cancel the rotation.

Mirroring and Adjusting the Eyes

You've made one eyeball, but Jim needs two! First, you need to align the existing eyeball with one of the eyes in the background image. Keep in mind that because the lattice is now deforming the eyeball, you need to select both the eyeball and the lattice to move them together. To create the second eyeball, duplicate and mirror the first one (see Figure 7.6).

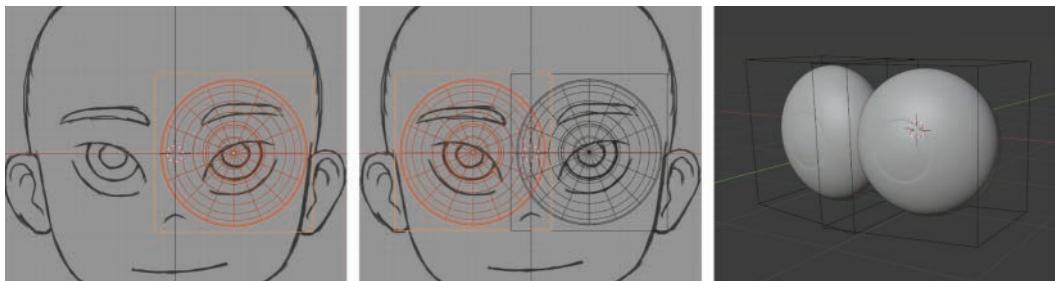


Figure 7.6 Aligning the eyeball (left), which squeezes the eyeball when its points are moved (right)

Here are the steps to adjust and mirror the eyeball:

1. Select the eyeball and the lattice.
2. Move and scale the eyeball and lattice to adjust them to the shape and size of the drawn eye in the front view. Adjust their positions in the side view as well. Don't worry if they don't fit perfectly. Keep in mind that when you have an object deforming another object (such as a lattice), you must transform them together to not break the deformation.
3. When the first eyeball is aligned, make sure to place the 3D cursor in the center of the scene. You can press **Shift+S** and select the Cursor to Center option, or you can press **Shift+C**.

4. Press **Shift+D** to duplicate the eyeball and its lattice. Right-click to cancel the movement and leave the new eyeball and lattice in the same locations as the originals.
5. Press **.** (period) on your keyboard to set the pivot point in the 3D cursor.
6. Press **Ctrl+M** to enter Mirror Mode. This mode makes the current pivot point the mirror plane (which is why you should use the 3D cursor for mirroring; otherwise, you'd be mirroring from the selection's pivot point).
7. Press **X**, **Y**, or **Z** to select the axis of the mirror. In this case, press **X**; the new eyeball and its lattice should move right into place (refer to Figure 7.6). Press **Enter** to confirm this action. Remember to set the pivot point to Median Point (**Ctrl+,**) or Bounding Box Center **(,**) before you continue working.

Caution

When using this mirroring method (**Ctrl+M**), you may find that sometimes it gives you unwanted, weird results, such as not mirroring in the expected way. Usually, this happens because you've rotated or negatively scaled that object in Object Mode before and its axes are not aligned correctly with the world-space axes. If you find yourself in this situation, select the object before mirroring, press **Ctrl+A** to apply Rotation and Scale, and try again. This technique should solve the problem.

An extra step you can take after you finish each part of the character is naming objects properly and organizing them in collections in the Outliner, making them easy to find and hide as necessary.

You can create a new collection by selecting objects in the scene and pressing **M**. In the Outliner, you can drag and drop objects and collections to organize them as you desire.

Modeling the Face

Now that Jim has a good pair of eyes, it's time to start modeling a cool face to support those eyes! Throughout this stage, you use box modeling to create the face so that you can get a good idea of how this method works.

Studying the Face's Topology

You may remember from Chapter 4, "Project Overview," how important the pre-production phase is for a project. It's also important for any modeling task, and the face is one of the trickiest parts of the body to model, so it needs really good preparation. It's useful to look at the reference drawings you created and study the topology that could work for the face so that you'll have an idea of how to model before you begin, which is a much better approach than modeling on the fly! Figure 7.7 shows a topology study for Jim's face, with quick sketches over the reference drawings.

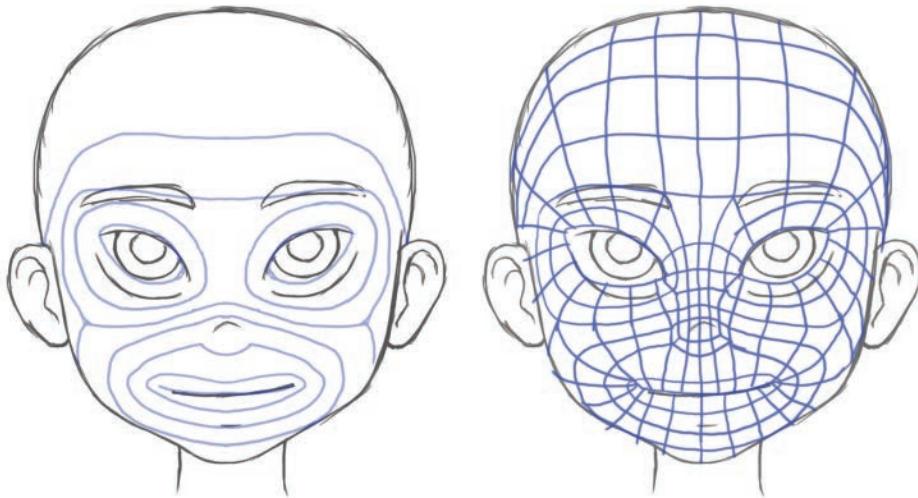


Figure 7.7 A representation of the edge flow around the main areas of the face, such as the eyes, nose, and mouth (left), and a drawing of a possible topology (right)

Note

Modeling a face is one of the most difficult topics discussed in this book and one of the most difficult parts of the character-creation process. Don't worry if you don't get it right on the first try; modeling takes practice, patience, determination, and skill. Also, remember that the following steps represent the key moments of the modeling process. Between steps, when you're modeling something organic (such as Jim's face), you need to move vertices around, using a bit of trial and error, to make the new geometry define the correct shape. Organic modeling requires constantly adjusting the positions of vertices. Vertices won't magically snap to where they should be; it's your task to tell Blender where you want them.

Don't get discouraged if you don't get great results on your first try modeling a face. Just try again, and you'll get better.

Blocking the Face's Basic Shape

In this section, you start by blocking the face. *Blocking* is the first stage of creating a model, animation, painting, or any other artistic endeavor. It's the stage in which you define, quickly and roughly, how something will look. You're not paying a lot of attention to detail—just defining the base. In this case, for example, blocking consists of creating the basic shape and geometry of the face to which you'll add definition in later steps.

Blocking is useful because it makes substantial changes easier and faster, so during this stage, you can experiment with different modeling ideas.

Tip

Keeping your scene organized is important. Now that you'll start creating a lot of new objects, it's a good idea to name them intuitively so that the names represent the objects that they belong to; generally avoid using default names such as Plane.001 and Sphere.013. Naming objects properly also helps you find what you're looking for in the Outliner when you're working on a complex scene.

In Figure 7.8, you see the first steps for modeling the face, which you use to create the basic shape:

1. Create a cube, go to Edit Mode, and use Loop Cut and Slide (**Ctrl+R**) to divide the cube as shown in the first image: three vertical divisions for the front of the face, one horizontal division through the front and sides of the face, and one vertical division for the sides of the face. These edge loops will help you set the basic shape of the face in the first stages. Here, we use three vertical lines in the front to allow for additional details for the mouth and the eyes.
2. Select all vertices by pressing **A**; then right-click and choose the Smooth Vertices tool from the contextual menu, or choose it from the Vertex menu (**Ctrl+V**). When smoothing is applied, increase its iterations in the Adjust Last Operation panel. The idea is to get a more spherical shape. Now scale the entire shape to make it approximately fit the size of the head in the reference images.
3. From the front view, select the vertices of the left section of the mesh (negative X) and delete them so that you're left with only half of the face model. Next, add a Mirror modifier; the default settings should be enough to make the mirroring work. Activate the Clipping option in the modifier so that the vertices in the middle are prevented from jumping to the opposite half of the mirror center. At this point, you can work on half of the face, and those changes will automatically be reflected on the other side.
4. Use proportional editing (press **O** to enable/disable it) to adjust the shape of the geometry to fit the reference images of the head. The eyes should be placed on the horizontal line in the front. At the bottom of the head, the faces in the back will be the base for the neck.
5. Select the faces in that back area, and extrude them to create the neck. To form the neck, adjust the vertices so that they look round. At this stage, you'll be defining the basic shape, so avoid cubic shapes as much as possible. Otherwise, as you start adding details, those cubic shapes will be more noticeable, and it will be more difficult to arrange them properly at that later stage. Don't worry if the neck doesn't fit the references perfectly; you can improve the shapes later, when you have more geometry to work with.
6. With the Knife tool (**K**), make a couple of cuts in the front, as highlighted in the sixth image in Figure 7.8. You're left with three horizontal edge loops in the center of the face.

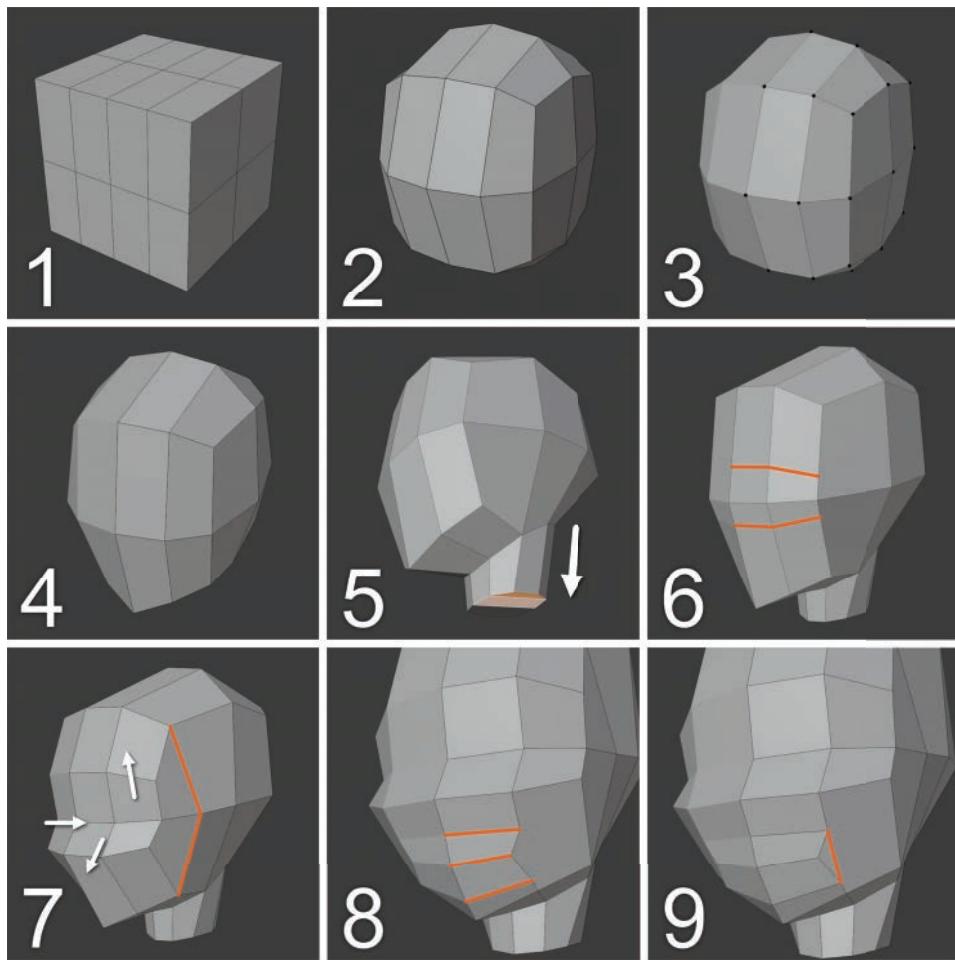


Figure 7.8 The first steps of modeling Jim's face

7. Move the three edge loops to fit Jim's face. The top one will define the eyebrows, the middle one will determine the center of the eyes, and the bottom one will establish the nose and cheeks. After that, using the Knife tool again, make the cuts highlighted in the seventh image in Figure 7.8 to end up with a round face loop that surrounds the face.
8. Perform three cuts in the mouth area, and move them as needed. The middle one will establish the mouth, the bottom one can help define the chin, and the top one will mark the area of the mouth near the nose.
9. Join the side vertices of the edges above and below the mouth to form a triangle in the corner of the mouth. You can do this by selecting those two vertices and pressing **J** or using the Knife tool.

Defining the Face's Shapes

After you complete the blocking stage, in which you create the face's basic shape, you add some definition to the geometry.

Figure 7.9 shows the next steps of the face's modeling process:

10. Using the Knife tool, cut the triangle shown in the mouth's corner in Step 9, and create two new edge loops to connect the mouth with the cheek and the jaw.
Now the loop around the mouth is composed of quads (four-sided faces).
11. Select the mouth's edge, bevel it just a little (**Ctrl+B**), and delete the new geometry (marked in red) so that you can make the mouth's opening and the area surrounding it with circular loops formed of quads.

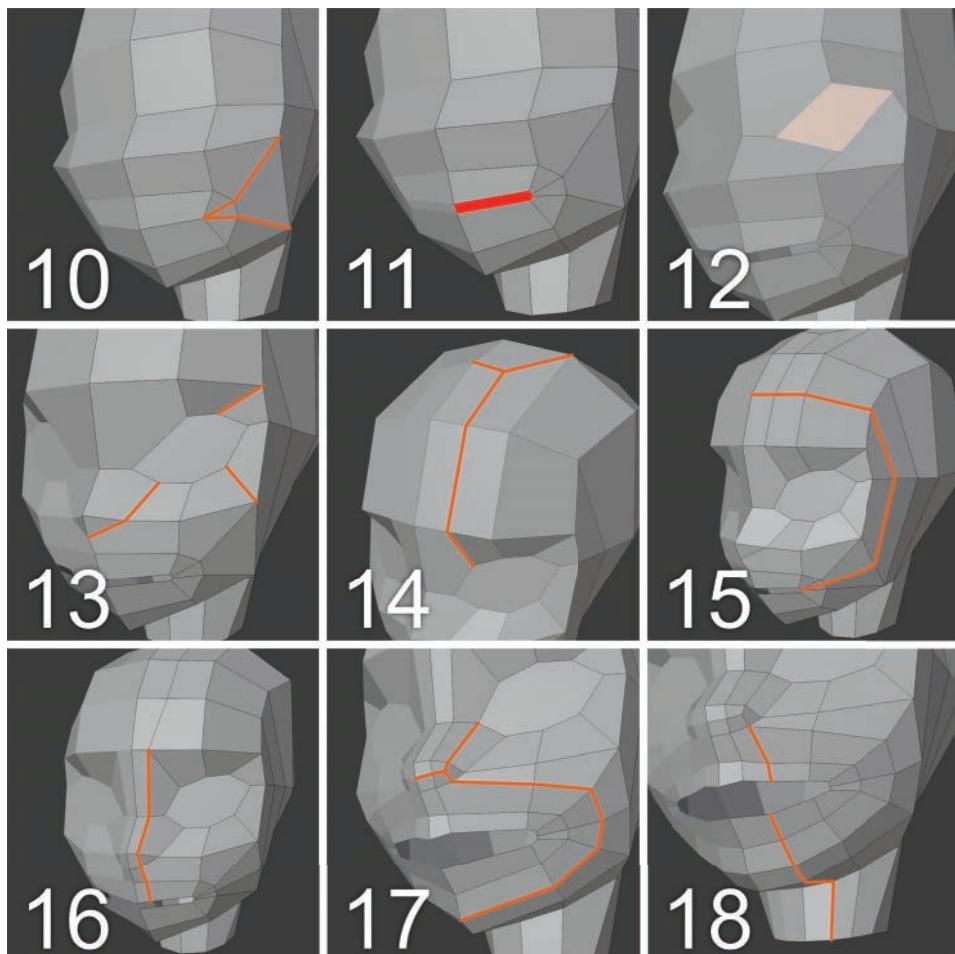


Figure 7.9 Continuing the face modeling

12. Select the vertex in the middle of the eye, and bevel it with the Bevel Vertices tool (**Shift+Ctrl+B**). Then move the resulting vertices to make a shape similar to the eye in your reference images.

Tip

When you're adjusting your mesh to the reference images, it's best to overestimate the shapes' sizes (make them a little bigger). The reason is that as you add more details, some of those vertices will be divided, causing the shapes to shrink. The same thing can happen with a Subdivision Surface modifier; when you smooth shapes with a Subdivision Surface modifier, they shrink, especially where you have less geometry density.

13. Perform several cuts around the eye with the Knife tool, as highlighted in the 13th image. This step gives you more vertices to define the eye's shape, as well as some additional cuts to start defining the nose's geometry.
14. One of the sides of the eye was not cut in Step 13, so cut it now. Keep the cut going to the top of the head, and end it with a corner pointing toward the mirror center. You need the loop only in the front of the head; this way, you can end the loop where it's no longer needed without affecting the back of the head. Such a geometry could create some bump in the surface when it's subdivided, but this area of the head will be covered with the hair.
15. Using the Loop Cut and Slide tool (**Ctrl+R**), add a new loop from the mouth corner all around the face, and adjust its shape and the surrounding vertices.

Tip

When you add a new loop to an organic shape with the Loop Cut and Slide tool (**Ctrl+R**), you can take advantage of the Smooth option in the Adjust Last Operation panel to make the new loop less rigid and flat between the two existing loops. Also, when you have an area with quite a few vertices, you can select all of them and use the Smooth Vertices tool to smooth all of them at the same time.

16. Cut a line from the eyebrow to the mouth, and define the nose a little more.
17. Make a new cut from the eye down to the nose, and join it with a new loop around the mouth area. Adjust the shapes, and you have the geometry necessary to create the nostrils.
18. Create some new edges from the bottom of the neck to the mouth and then from the nose to the mouth. (This cut turns the n-gon between the nose and the mouth into two quads.) Move things around a little to adapt the new vertices to the reference images. Then you're ready to add some more details and create the lips and other details.

Defining the Eyes, Mouth, and Nose

Little by little, Jim's face is starting to take shape! The next steps, shown in Figure 7.10, add definition to the eyes, mouth, and nose:

19. Select the mouth's loop, extrude it, and adjust the vertices to get the lips' shapes, according to the reference images. Don't worry about little intersections between the top and bottom lips.
20. Press **Ctrl+R**, and add a new loop to the lips to add more detail to that area. You can probably inflate the lips by adjusting the Loop Cut and Slide Smooth options in the Operator panel. At this point, you can add a Subdivision Surface modifier to the mesh and enable it from time to time to make sure that the geometry is behaving correctly when it's smoothed.

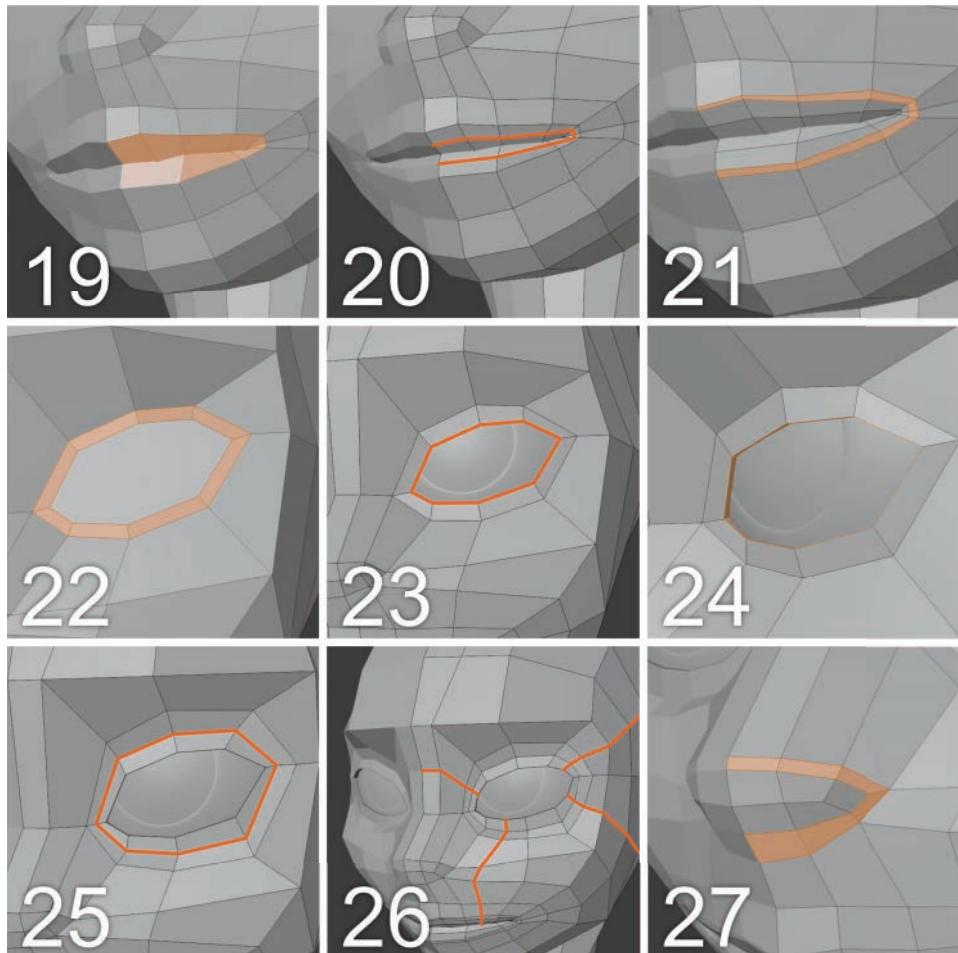


Figure 7.10 Adding more definition to the eyes and the mouth

21. Select the outer loop of the lips, and press **Ctrl+B** to add a bevel to it. Then slide the vertices near the mouth's corners to separate them a little more than the other vertices in those loops. This step adds density and defines a corner at the border of the lips when you add a Subdivision Surface modifier. Separating the vertices near the mouth corners makes those areas smoother, and the central area of the lips has more definition, as its loops are closer together. (Remember that if the vertices are close together, the subdivisions look more like corners, whereas if the vertices have more space between them, subdivisions are smoother.)
22. Select the n-gon of the eye, and press **I** to inset and create the base of the eyelids.
23. Unhide the eyes, and adjust the eyelids' geometry to the eyeball's surface. Proportional editing can help with this task. Leave a space between the eyelids and the eyeball. To do this, move the camera around and keep adjusting the vertices from different points of view until they look good.
24. Select the inner loop of the eyelids, and extrude it to fill in the space between the eyelid and the eyeball.
25. Select the outer loop of the eyeball, and slide the vertex out to make some space for a new loop to help define the eyelids a little more. You can slide vertices and edges or edge loops pressing **G** twice.
26. In the eye area, add some more loops by pressing **Ctrl+R** to define the section between the nose and the forehead, the corner of the eye to the mouth, and from the eye toward the side of the head. Then adjust the vertices to the reference images, and make sure that the shape blends smoothly with the rest of the head (with no unwanted bulges or holes).
27. Select the bottom of the nose and the nostril faces. Inset them, and turn the Boundary option off (you'll find it in the Operator panel) so that the nose's front faces don't inset in the center.

Tip

While you model, try to plan how you'll perform the next steps. If you have in mind what the final topology will look like, you can add loops and vertices to achieve that specific goal. Modeling blindly is possible, but you'll probably lose some time figuring things out and fixing issues that you could have seen coming. Sometimes, you'll have to delete certain parts and rebuild them to create a better topology.

Modeling is all about gaining experience, and you likely won't have enough practice during the creation of your early models to know what may happen next. Keep practicing!

Adding the Ears

The face is almost done! Figure 7.11 shows steps that add more details to Jim's face. At this stage, you add the ears and define the neck and head a little more:

28. Move the nose vertices to define the shape of the nose. Turn on the Subdivision Surface modifier to see how the model looks when subdivisions are enabled.
You may want to play with the nose geometry; in this case, as the design presents a pretty stylized character, you're not going to model the nostrils with all the detail.

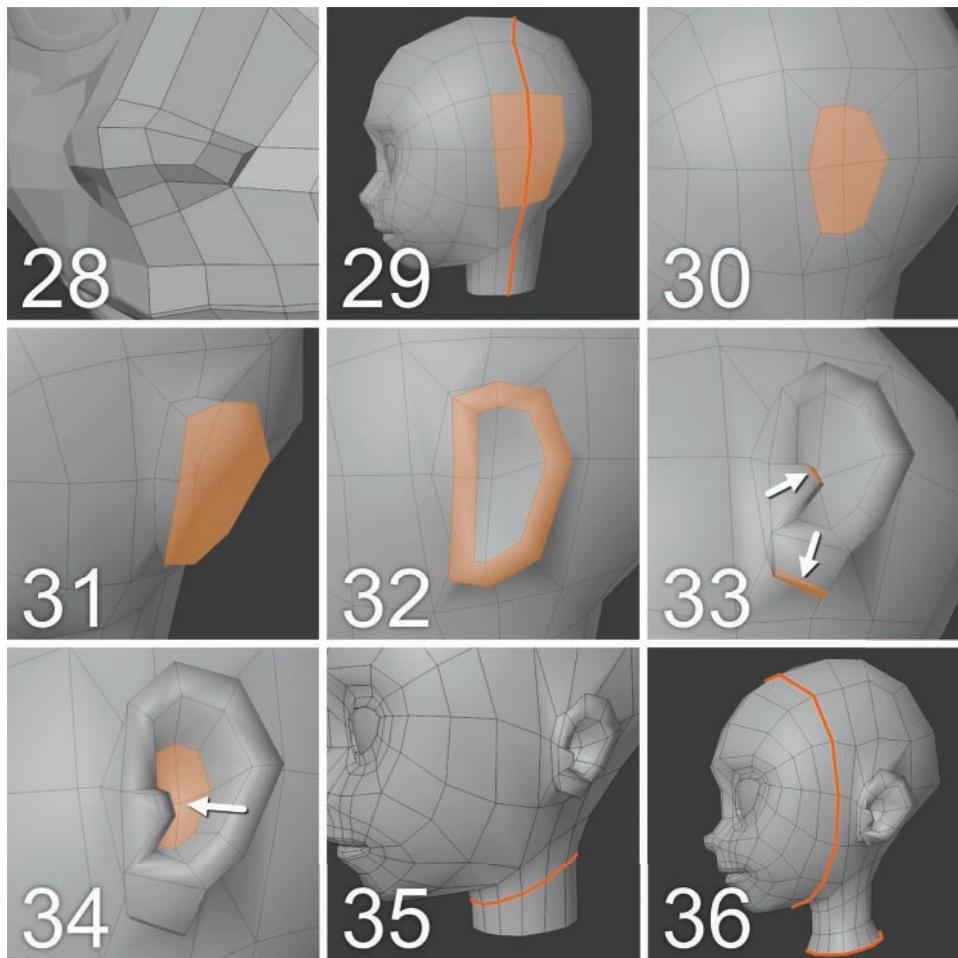


Figure 7.11 More adjustments and details to create the ears

Note

While you are working on the model, it's useful to enable the Subdivision Surface modifier from time to time to see whether the geometry behaves correctly when subdivided. Also, this modifier provides several display modes (the four buttons located at the top of the modifier panel), and the last two are really helpful for this stage. One of these modes allows you to see the subdivided model in Edit Mode while you keep working on the original mesh as though it were a cage for the subdivided model. The last mode lets you work on the subdivided model without displaying the original mesh cage (an option that can be useful and intuitive in some situations).

You can also switch among the subdivision levels of the selected objects very quickly by pressing **Ctrl+1** while in Object Mode, replacing the number 1 with any other number to define the subdivision level. If you press **Ctrl+0**, you get 0 subdivisions, so this keyboard shortcut is the equivalent of disabling the modifier if it gets in the way while you're modeling. Press **Ctrl+2** to display two subdivisions.

29. Go to the side of the head, and create a new loop from the neck to the top of the head. The highlighted faces are going to serve as a base from which to extrude the ear. Ears are quite tricky, but in this case, you're making an anime stylized design, so create a simple ear—one that is not very realistic but fits the overall look of the character.
30. Inset the selection to create the base for the ear.
31. Extrude the shape, and adjust it to resemble an ear.
32. Make an inset within the ear.
33. Extrude and shape the highlighted areas to define those parts of the ear.
34. Add a new extrusion to create the ear canal, and arrange those vertices a little. Don't worry if the geometry looks weird; just keep an eye on the subdivided mesh, because when it's smoothed, it will look quite different. Also, now is a good time to use proportional editing and shape the ear from the side view to make sure that it fits the ear reference.
35. Cut a new loop in the neck to create a bit more geometry between the torso and the base of the head.

Tip

In the images that accompany the modeling steps, you're seeing the low-resolution mesh so that you can get a clear idea of how the polygons and vertices are being added and modified. At this point in the process, however, you should be working with the subdivided mesh to get a peek at how it looks (even if you enable it only from time to time).

36. Keep adding more loops in the areas where you'd like to have more detail. In the image for this step, the two loops that were added to the mesh are highlighted.

One loop is at the bottom of the neck, and it's extruded so that later, you won't see an empty space under Jim's jacket. Also, a clean loop surrounds the entire face, which you can think of as being a division to extract a mask from the face.

Building the Inside of the Mouth

In this section, you add the final details to Jim's head. The face is looking good, but you need to create the interior of the mouth so that when Jim opens his mouth, you won't see an empty space or the back of his head! Look at Figure 7.12, and follow these final steps:

37. Select the inner lips' loop, and extrude it into the head. Scale the new loop up in the Z-axis and move the vertices as needed to get a more rounded shape instead of the flat one at the lips' intersection. You also need to make sure that the vertices on the top at the end of the new geometry belong to the top lip and the bottom vertices belong to the bottom lip. Given the lips' geometry and possible intersection, it's possible that you'll get the opposite result.

Tip

Remember that you can select what you want to keep in the view and press **Shift+H** to hide the unselected geometry. In these images, the rest of the head has been hidden so that you can clearly see what's going on inside the mouth.

38. Add some loops to better define a rounded area inside the mouth. It's very important to add a loop near the inner lips; otherwise, they'll lose some shape when they're subdivided due to high tension in the geometry. Don't worry if the geometry overlaps a bit in the inner-lips area.
39. Close the back of the hole, and refine the shapes a little. You can add another loop near the inner lips so that the inside of the mouth in that area is more vertical; this addition creates some space for the teeth later.

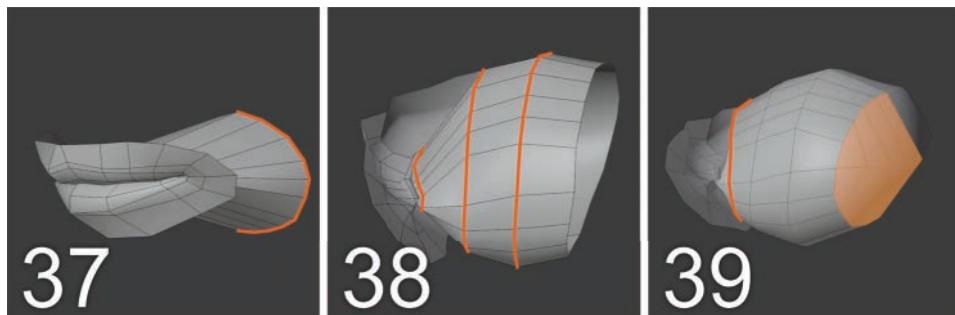


Figure 7.12 Creating the inside of the mouth so that it isn't empty space and Jim can open it

You’re done modeling Jim’s face! In Figure 7.13, you see the result. The face is often one of the trickiest parts of modeling a character. We’re so used to seeing faces that a character’s face will immediately look wrong if something is out of place, so achieving a pleasing result can be difficult.

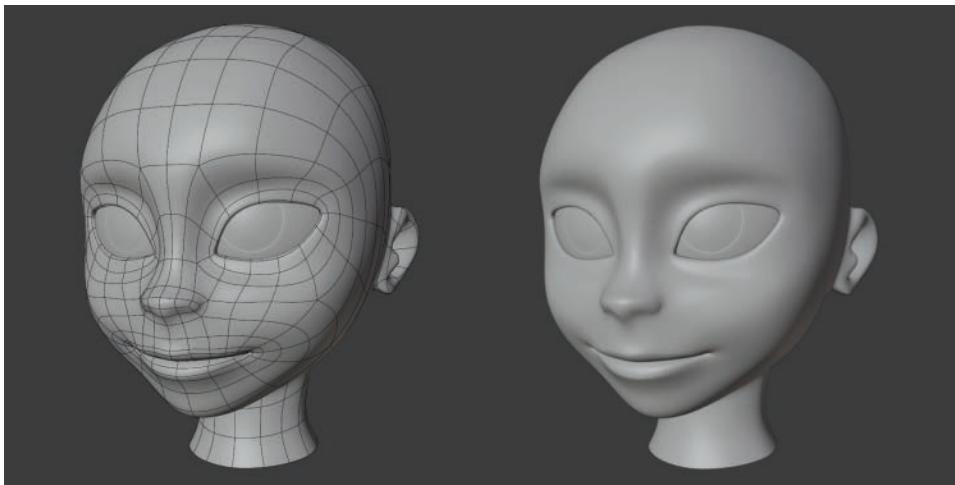


Figure 7.13 The topology for Jim’s face (left) and the final subdivided result (right)

Modeling the Torso and Arms

Up to this point, you’ve been working on the face, but now you switch to the body, which means that having the face’s reference images in the background are not useful anymore. You can disable the visibility of the head reference images from the Outliner. You can even create a collection for all the head references so that you can have a main collection of references and subcollections for different parts of the body. This practice will make it much easier to enable and disable different things in your scene.

Now you can add the body reference images, following the same procedure you used for the head references. This time, however, you’ll need to move the references up, because if you center them as you did with the face, half the body will be under the floor. This result, while undesirable, wouldn’t pose a major problem, as you saw while working with the face; you surely could move the body’s 3D models after the fact. But in this case, you can move the references up until the feet are touching the floor of the scene. (If you look at the model from the side view, the floor will be marked by the green horizontal line on the Y-axis, as shown on the right side of Figure 7.14.) You may also move slightly to the left (on the X-axis) the front reference, and also adjust its Z position so the heels are touching the floor. You must be ready to adjust your reference’s positions as necessary because more often than not, they won’t fit perfectly.

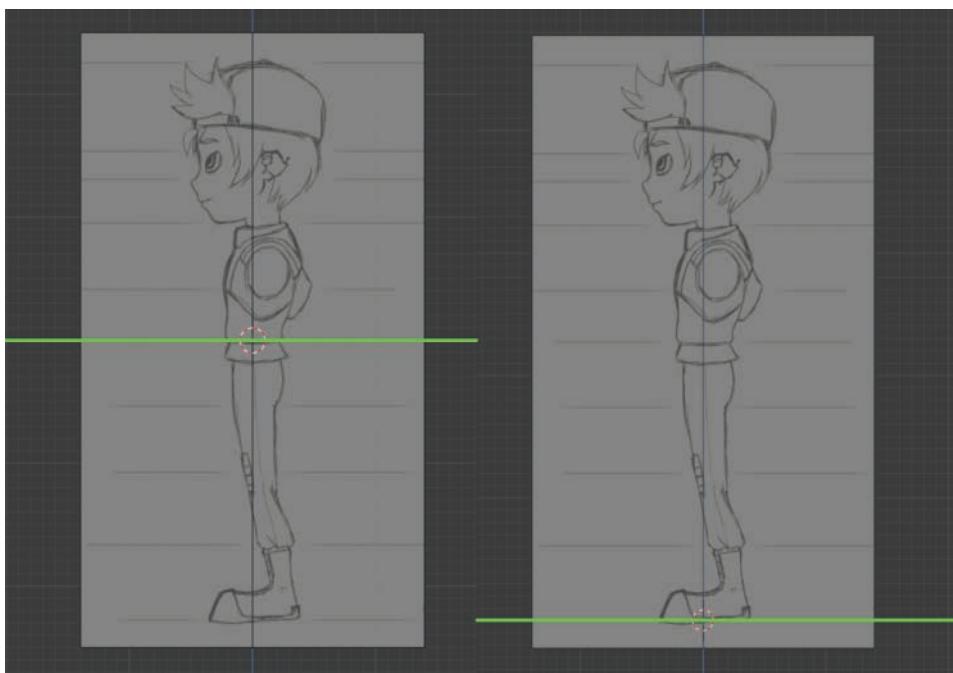


Figure 7.14 Jim's side-reference floor position when you import it (left) and after you slide it up until the feet are touching the floor (right)

The first thing you notice is that the face is really large and out of proportion, so select everything you've created so far (face, eyes, and lattices), move those elements, and scale them to make them fit the new references. Now the head should be in place, and you should have space to start working on Jim's body.

There is another problem you'll have to work out, which I created on purpose so that you'd have an opportunity to play with your imagination a little and confront a situation you may encounter at some point in your own projects. Jim's designs show the character's arms in a pose slightly different from what you're actually going to model. Sometimes, arms are modeled in what is often called a *T pose*: completely extended and parallel to the floor. Although the *T pose* is appropriate for modeling purposes (if everything is aligned to one of the 3D world's axes, it's easier to manipulate), it may not be the best thing for the model in future stages. If you create the arms completely extended and parallel to the floor, when they're deformed, you could have issues with the shoulders due to their large rotation range.

In Jim's case, the shoulder area has some details in the jacket, so you should create the arms slightly flexed (at 45 degrees or so). Whether you rotate the arms up or down, the deformation won't be really big, and Blender will keep those details, so in this situation, the reference images show something a little different. As a result, you'll have to create your model in a slightly different pose, but don't worry: During the process, you'll learn a couple of useful tricks that will help.

Modeling the Basic Shapes for the Torso and Arms

To start, block the shapes for Jim’s torso and arms. In Figure 7.15, you see the first steps of the process. Because the face was modeled with box modeling, you have a chance to see how poly to poly works at this stage, even though it will be combined with box modeling at some points:

1. Create a circle with 12 vertices, delete the left half, add a Mirror modifier to it (similar to what you did when you started working on the face), and place it in the base of the neck area. Enabling the Clipping option of the Mirror modifier helps keep the central vertices of the torso in the mirror plane and prevents them from crossing to the reflected side.
2. Perform three extrusions (**E**), each one using two edges of the circle (two edges at the front, two at the back, and two at the side). The two frontal edges define the front of the torso, the two side edges draw the trapezoids that go to the shoulders, and the two edges in the back extend to the hips and define the back.
3. Use the Loop Cut and Slide (**Ctrl+R**) tool to divide the trapezoid muscle’s faces where shown in image 3; then select the front and back outer edges, and extrude them down. Next, select the four vertices that have an empty space between them, and press **F** to create a face in that space. Do the same with the empty spaces in the front and the back.
4. Select the side middle vertex of the shoulder, extrude it down to the hips, select the vertices, and press **F** to fill the vertices with faces. At this point, you have the envelope of the torso.
5. Make three horizontal loop cuts through the torso, and adjust the shapes to the reference images. Keep an eye on the highlighted faces; you should be giving them the shape from which the arm will be extruded.
6. Select the faces highlighted in Step 5, and extrude them horizontally to create the entire arm down to the wrist.

Tip

Even if you extrude the arm completely horizontally (on the X-axis), use the arm in the reference as a guide. Imagine where the wrist would be if the arm were extended in a T pose. Later, you’ll pose the arm better and adjust it to the reference, but now it’s easier to work with it aligned to one axis (X, in this case) to make modeling easier. Also, the shape of the wrist will be a little irregular, so after the extrusion, you can scale its vertices to 0 on the X-axis; from the front view, it now looks flat.

7. Cut a new loop in the middle of the arm, and adjust it a little to define the elbow. From the top view, move the loop back just a bit, as though the arm were slightly flexed. This step helps you achieve the arm’s shape. Keep in mind that you should rarely have the arms extended completely straight at 90 degrees, as that would look unnatural (and would also cause issues while rigging the character, as you’ll learn in Chapter 11, “Character Rigging”).

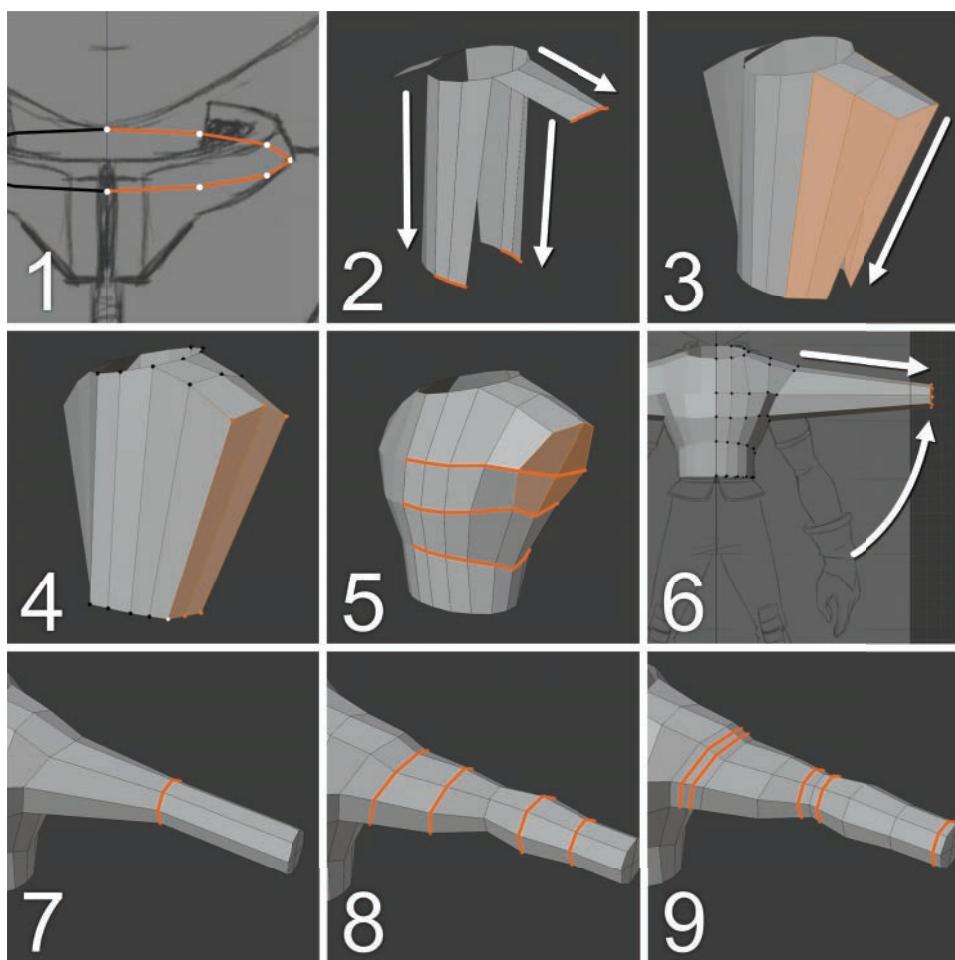


Figure 7.15 The first steps of modeling Jim's torso and arms

8. To continue defining the arm's shape, add some new loops (**Ctrl+R**) in the biceps and forearm areas. Each time you add a loop, move the vertices around a little; if you add a lot of loops and try to modify the shape later, the process will be more difficult.

Tip

You have a lot of ways to model shapes such as arms and legs. One good way is to extrude (in the case of an arm) from the shoulder to the wrist and then cut the elbow, divide the biceps and forearm, and keep dividing until you're happy with the result.

This way, first you'll have the general shape of the arm, and then you'll define the main articulation, which will have two divisions, then four, and then eight (or more).

This method is easiest for me to use in modeling, and it's better for developing details little by little than extruding the shape piece by piece. It also gives you a better grasp of the general shape before you add cuts, shapes, and details between the two extremes.

9. Add more loops in the articulation areas, such as the shoulders and the elbow. It's important to have enough geometry in those areas that they can deform correctly later. Remember that three loops in each articulation are often enough (at least for simple characters). You should also add another loop near the wrist to define the shape better when it's subdivided. As with the face, now is a good time to add a Subdivision Surface modifier and start checking on how the subdivided model will look.

Defining the Arms and Torso

In this section, you add more definition to the arms and torso, and you begin to add Jim's backpack. Continue with the next steps, shown in Figure 7.16:

10. Add some more loops and definition to the arms.
11. Select the faces surrounding the elbow area. You want to add some definition to the elbow to make sure that when Jim flexes his arm, the elbow responds accordingly.
12. Make an inset (**I**) in those faces, and use the sliding tools (select a vertex or edge and press **G** twice) to make the loops around the elbow rounded. Also, select the face in the middle of the elbow loops (the one highlighted in image 12) and move it out a little bit to give the outside of the elbow a little bulge.
13. Go back to the torso, and add some loops to define the waist area.
14. Using the Knife tool (**K**), make a few cuts similar to those shown in image 14, and adjust the new geometry to define the pectoral area.

Tip

If you really want a character's model to be awesome, work on the topology for the muscles in the arms and torso as you did for the face. In this specific model, the muscles are not very defined, and the jacket deformations will be pretty simple, so you can get away with topologies that are less complex.

15. With the Knife tool, cut some new edges around the neck and near the shoulder to add a new loop in that area. (In image 15, you can see only the front part of the torso, but the cut is exactly the same as the one you use to model the back of the torso.)
16. Make two cuts vertically near the frontal mirror plane to create the zipper for the jacket. Pull the inner cut and the mirror vertices in to create that indentation.

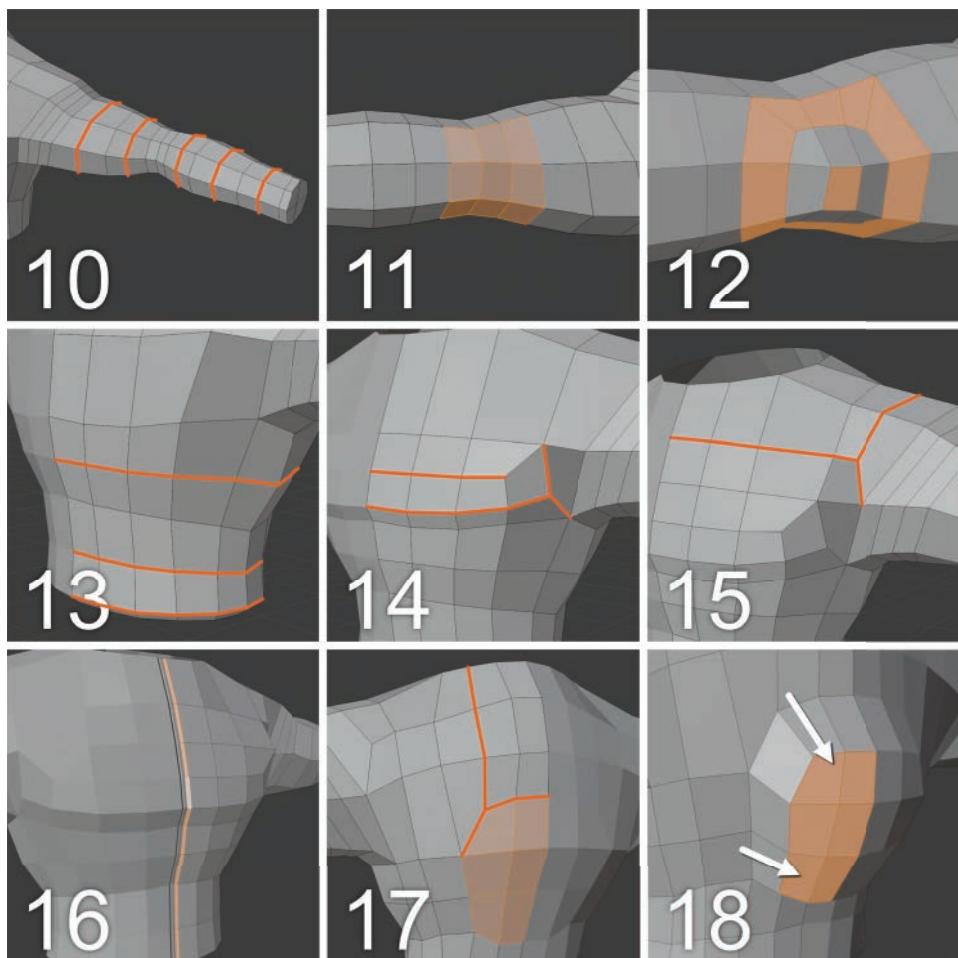


Figure 7.16 Continuing the torso-modeling process

17. Jump to the back side of the model. If you haven't done anything to it yet, adjust the shape of the vertices there according to the reference images. Create the cut indicated in image 17, and adjust the highlighted faces, which will serve as a base for the backpack.
18. Extrude the selected faces for the backpack, and adjust the shapes.

Detailing the Backpack and Jacket

Next, you add detail to Jim's jacket and backpack. Continue with the steps, shown in Figure 7.17:

19. Select the edges that comprise the corners of the backpack, and bevel them (**Ctrl+B**). A triangle will be generated by this operation (marked with a red circle), which you'll fix in the next step.

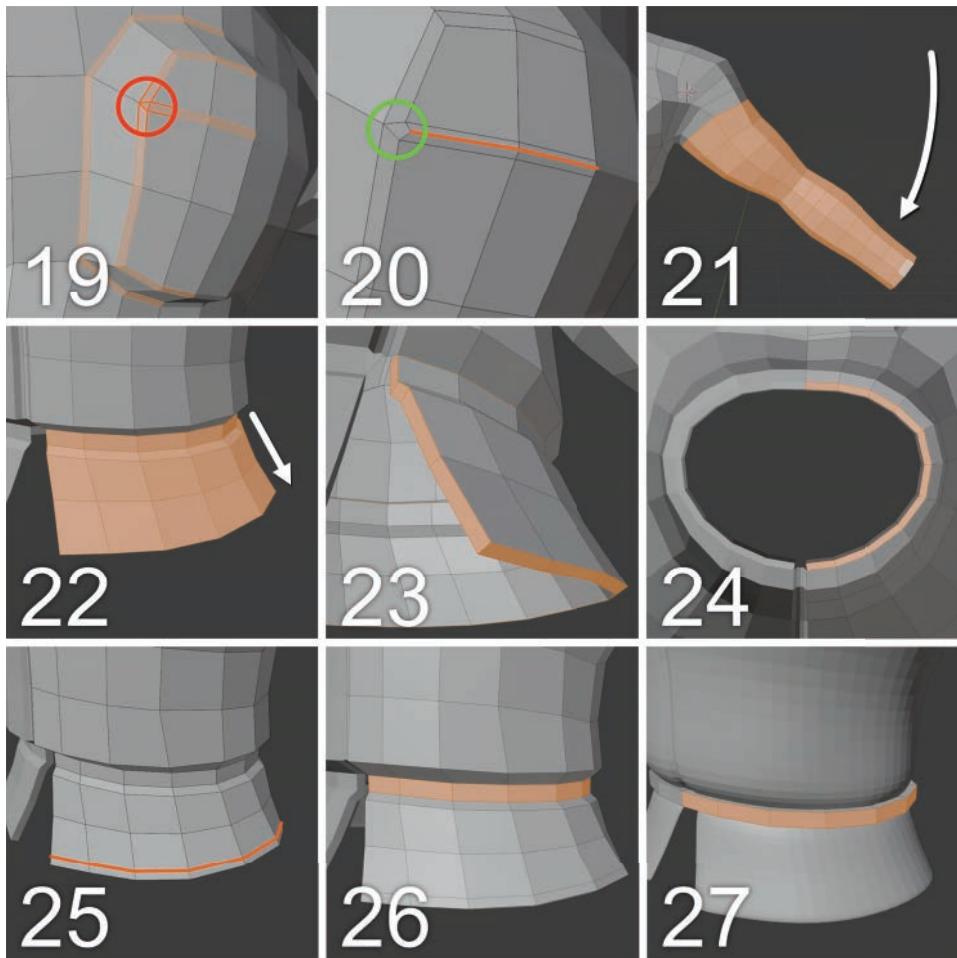


Figure 7.17 Adding some details to the back and the bottom of the jacket

20. Sometimes, you can weld vertices to remove triangles, but in this case, that method is trickier because if you take out a triangle, you also take out some of the detail added by the bevel. In this case, add a new loop to the side of the triangle that is

facing the beveled faces, and slightly move the fourth vertex created with the new loop. This way, if you add a few more faces to the model, you end up with a quad instead of a triangle, and with the added density, you define that corner a bit better. (If that result isn't what you're looking for, you can always separate those three loops to open some space between them and create a smoother surface.)

21. Place the 3D cursor in the shoulder, select all the arm's vertices except those near the shoulder, and use the Proportional Editing tool to rotate the arm to relax it a bit. (Make sure to switch to the 3D cursor pivot point first so that the rotation happens from the shoulder.) Also, select the four faces at the end of the arm (the wrist), and delete them. These faces won't be visible and won't be useful later, so it's a good idea to get them out of the way now.

Note

This step uses a very nice feature of the 3D cursor that allows you to pose characters that don't yet have a skeleton. Just place the cursor in the articulation, and select the part of the body you want to move; then, with the help of the Proportional Editing tool, you'll be able to pose your character pretty well. Keep in mind, though, that you may need to adjust the vertices around the articulations after these operations; they can get a bit distorted or overlap with other parts of the surrounding geometries.

22. Create a few extrusions to make up the shape of the flap at the bottom of the jacket.
23. With all the faces of the flap selected, press **Ctrl+F** to access the Face menu, and use the Solidify tool to add some thickness to the flap. In image 23, the thickness is highlighted.
24. The thickness you added by using Solidify in Step 23 also created faces at the top of the flap (in the interior) that are not very useful and can even cause some problems when the mesh is subdivided, as they're welded onto the backs of other polygons. Delete those faces now. In the image, the faces are highlighted in a view from the jacket's interior, looking down. Also, Solidify doesn't recognize the mirror's clipping on the back part of the jacket's flap, and it generated a few faces that crossed the mirror plane and created some intersections. Get rid of those faces as well, and make sure that all the vertices around them are at the center of the mirror. It's easier to select those inner faces when you're in Wireframe viewport shading mode.
25. Add a new loop at the bottom of the flap so that when the model is subdivided, it has more density, and that corner is more defined. You can merge (**M**) the bottom vertex of the zipper with the top vertex in the same area that the Solidify tool created so that you don't have an empty area.
26. Select all the faces around the waist to create a new object for the belt. Press **Shift+D** to duplicate, and right-click to leave the duplicated faces in their original positions. Next, press **P**, and choose the Selected option; this option sends those duplicated faces to a new object that you'll be able to modify.

27. Select the new object. In Edit Mode (**Tab**), select the edge in the middle at the front, and move it to the left until it attaches to the other side of the belt thanks to the Mirror modifier. Select all the object's faces, press **Alt+E** to extrude, and select the Extrude Faces Along Normals option.

Note

When you duplicate an object or use the Separate tool to move a set of polygons from one object to another, the new object keeps the original object's modifiers. Use this trick to save time setting up modifiers in objects that will have to use a setup similar to that of previous models.

Finishing the Belt and Adding a Neck to the Jacket

The jacket is almost finished, but you have to add a little more detail to the belt, and the jacket needs a neck. Follow these steps to model these features, as shown in Figure 7.18:

28. Select the top and bottom edges of the belt, and bevel them so that they'll look sharper when you subdivide them.

Tip

If you want to focus only on these details, you can hide the jacket itself so that it's not in the way while you work. Just select it and press **H** to hide it. Press **Alt+H** to unhide everything when you've finished.

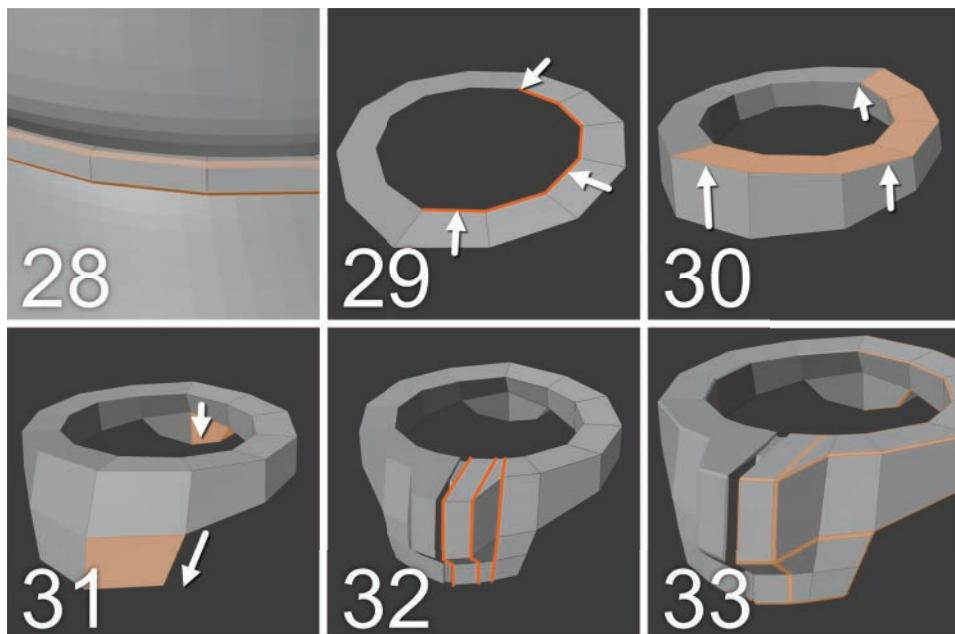


Figure 7.18 Modeling the jacket's neck details

29. Place the 3D cursor in the neck area, and create a circle. (Even though you could create the circle inside the jacket itself, for now, it may be better to create a new object so that you can treat the jacket and its neck separately.) Similar to what you did when you started modeling the jacket, make it with 12 vertices, delete the left-side vertices, and add a Mirror modifier to the object with the Clipping option enabled. Then move the vertices to form the shape of the jacket's neck. Select all the edges of that circle, and extrude them in.
30. Select all the faces, and extrude them up. You have the base shape of the jacket's neck.
31. Temporarily unhide the jacket and the head to adjust the shape of the jacket's neck to them, and perform the two extrusions shown in image 31 to add details to the neck.
32. Add a few loops to create the neck's front shape and the zipper.
33. When you add the Subdivision Surface modifier (which you can always add and enable/disable at any time to start checking the shapes), bevel the borders of the shape so that they are more defined. Also, watch the subdivided model, and adjust the shapes to make them fit Jim's head and body.

Modeling the Legs

Modeling the legs is pretty easy compared with everything you've done so far. The process is similar to the one you used for the arms, but you need to create the hips first, forming a base from which you can extrude the legs. Figure 7.19 shows the steps you need to follow to model Jim's legs:

1. Create another circle with 12 sides, delete the left half of the circle, and add a Mirror modifier with the Clipping option enabled. You can also duplicate the vertices from the bottom of the jacket, duplicate them and separate them into a new object, and then delete some of them to end up with just 12. This way, you keep the shape and the modifiers, saving you some work; it's up to you! Make two extrusions, and shape the model to look like hips.
2. Select the middle edges in the bottom of the model at the front and back sides of the hips, and create a face between them by pressing **F**. From this face, you'll create the crotch in Step 3.
3. Divide the face you created in Step 2 by using the Loop Cut and Slide tool (**Ctrl+R**), and add three divisions. Move the divisions down, and adjust them to shape the crotch. At this point, the model should resemble some sort of underwear.
4. Select the loop around the hole that should be the top of the thigh, and extrude it down to the tops of the ankles, where the boots will start. The same adjustments you made in the arms apply here. Instead of following the legs' current orientation exactly, make them a little more perpendicular to the floor (more vertically aligned).

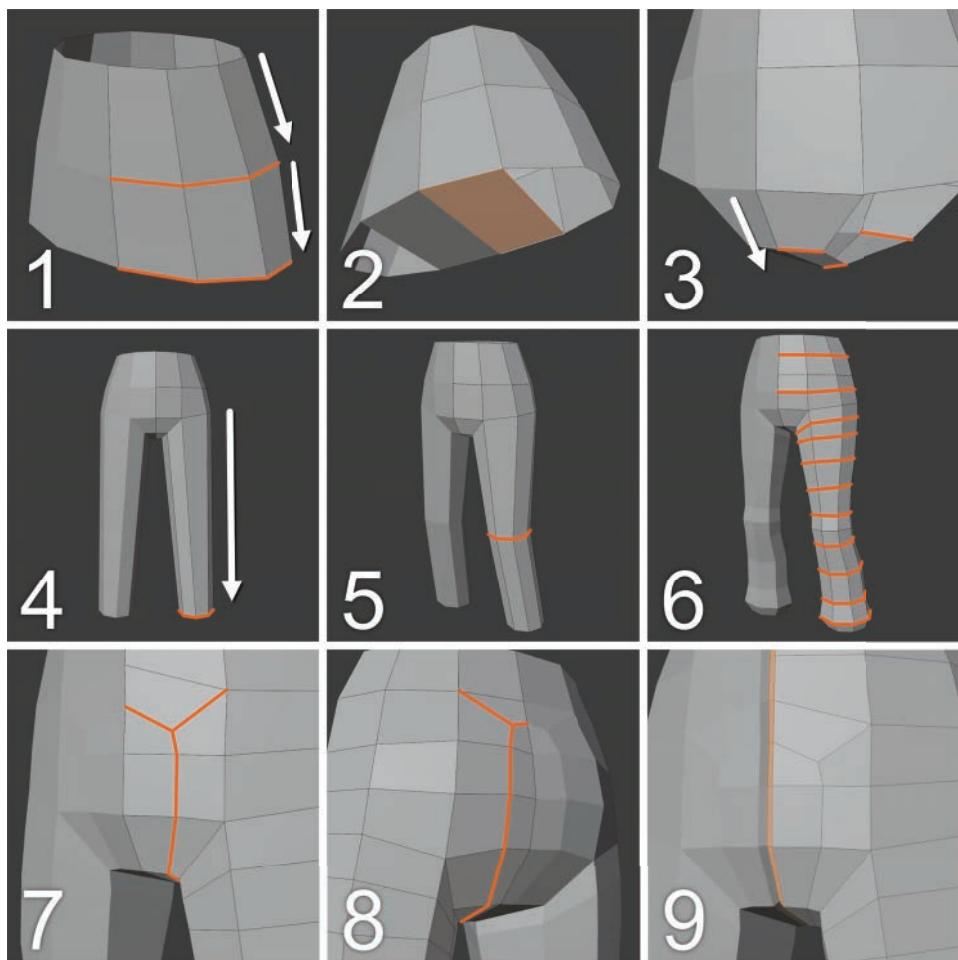


Figure 7.19 Steps for modeling Jim's hips and legs

5. As you did to define the elbow, cut a loop for the knee, and adjust its vertices according to the reference images.
6. Add some more loops to the leg, and keep adjusting their shapes. Remember to add at least three loops for the hip and knee articulations so that you can deform them properly later, when flexing the legs.
7. With the Knife tool (**K**), cut a shape similar to the one displayed in the image, both to add some definition to the crotch area and to add some geometry there to make sure that it will deform properly, as that area is very close to the articulations of the hips and legs.

8. Do something similar for the back of the trousers, but give this area a slightly different shape to resemble the buttocks.
9. Create a new loop in the center of the pants to make up the seam in the cloth. This step can be a little tricky, and you can use either of two methods:
The first method is to press **Ctrl+R** to cut a new loop and then slide the vertices near the mirror plane. At the bottom of the crotch near the legs, the new loop is very close to the others surrounding it; at the front, the loop is farther from the others, as there is more space. You could always press **E** when using the Loop Cut and Slide tool to even the distances to the mirror plane edges.
The second method is to disable the clipping option for the Mirror modifier, move the central vertices apart, activate the mirror clipping again, and extrude those vertices to the middle. This way, both loops are completely vertical and become welded into a single loop at the center.
When you're done with any of those methods to create that geometry, pick the central loop and scale it (**Alt+S**) slightly to the inside, but very subtly.
At this point, unhide the rest of the objects, and make sure that the trousers fit well inside the jacket's bottom flaps.

Tip

Sometimes, when you model using the poly-to-poly method, you may see some faces that are darkened or have edges with strange dark and light colors. When that happens, the normals of those faces usually are looking in opposite directions. Remember that a face's normal determines the direction in which the face is oriented, and it's important for all connected faces to be facing in the same direction; otherwise, shading problems may arise in the geometry. You have two options for fixing this issue:

- From the Face menu (**Alt+N**), choose Flip Normals to invert the normal of the selected faces.
- If you select several faces, you can press **Shift+N** to let Blender automatically calculate the direction in which all the normals should be facing; it conforms all of them to look toward their outer direction. (This direction may be difficult for Blender to figure out for complex and open-ended shapes in which it's not clear what is outside and inside.)

Modeling the Boots

Jim needs some feet to stand on, of course. In this section, you model the boots. Figure 7.20 shows the steps for doing so. At this point, you should be getting up to speed, and you're probably becoming more proficient with modeling and using Blender's tools, so the boots should be an easy task!

1. Create an eight-sided circle at the top of the boots, as shown in the reference images. Keep in mind that you need two boots. You can add a Mirror modifier

to get the second boot (place the origin of the boot in the center so both boots will be aligned), or you can manually duplicate and mirror the first boot. Alternatively, you could create an empty object (press **Shift+A** and select one of the options from the Empty category of objects), and use it as the center in the Mirror modifier to use that object as the reflection plane, regardless of the boot origin's location.

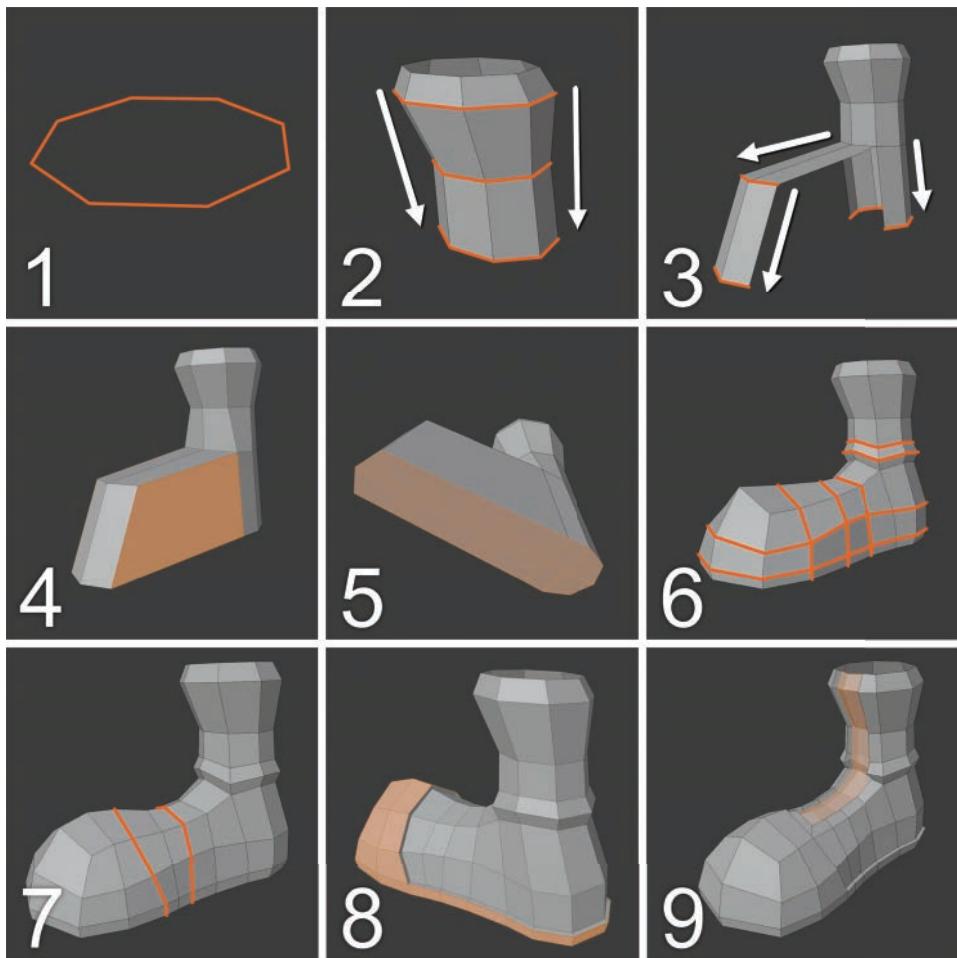


Figure 7.20 Steps for modeling Jim's boots

Tip

You can use a little trick to save some time: Select the bottom loop (**Alt+RMB**) of the trousers where they meet the boots, duplicate it (**Shift+D**), and separate it (**P**) into a different object that will become the boots. This way, you'll already have the

Subdivided Surface and Mirror modifiers added to the new object, and you also have a shape to start modeling from that will fit the shape it will be touching from the previous object.

2. Extrude (**E**) all the edges down to the ankle, and perform a couple of loop cuts to define the shape.
3. Select all the edges from the circle at the bottom except the two in front, and extrude them down to the heel. Then select the two in front, and make two extrusions to create the toe shape.
4. Create a face in the empty space at both sides of the boot.
5. Fill the sole of the boot with quads.

Tip

One reason why you should work with even numbers (such as eight sides for the initial object) is that you can usually fill empty spaces with quads quite easily. On the contrary, if you start with uneven numbers, you'll end up being forced to use triangles to fill the holes.

6. Add some loops to start defining the boot's shape.
7. Perform a couple of loop cuts (**Ctrl+R**) to add definition in the articulation of the foot (recall that you added several loops at the knee and ankle articulations) so that deformations will work well later. One of these loops also helps you create the shapes of some of the details in Step 8.
8. Select the area highlighted in image 8 (leave the sole of the boot unselected), extrude it (**E**), right-click to cancel the movement, and scale in the normal with **Alt+S**. You can also extrude by pressing **Alt+E** and selecting the Extrude Faces Along Normals option. This step adds some details to the boot.
9. Finish the details with a few minor operations. Select the two central face loops that go from the top almost to the toe, and create an inset (**I**) there (press **B** while doing it so the inset operation doesn't affect the borders at the top of the boot); then extrude the inset in to define the area where the shoelaces will be when the textures are in place. (See Chapter 9, "Painting Textures," to learn how to paint textures.) Also, when you extrude, two faces will be generated at the top of the boot with the same thickness as the extrusion, and you can delete them. As a finishing touch, add some loops to the thickness of the seams and small features to make them much more defined when subdivided. Last but not least, adjust the shapes of the boots where they meet the trousers. Some deforming may be needed; don't hesitate to use the Proportional Editing tool for that purpose. It's key that there are no gaps between objects.

Modeling the Hands

Hands are quite difficult to model, but I kept this example simple so that you can learn an easy method. If something goes wrong, don't worry; just start again, or save often after every step so you can go back and retry. Sooner or later, you'll get it right!

Building the Basic Hand Shape

Figure 7.21 shows the modeling process for a hand. You can model it wherever you want and move and scale it later to fit the rest of the model:

1. Start by creating a cube.
2. Make the cube narrower, and move one of the edges to the middle of what will be the palm of the hand. The diagonal face left there will serve as the base for the thumb.
3. Add two loop cuts, one near the wrist and another one near the fingers. The fingers will be in the top part of the shape.

Tip

Remember to keep adjusting the vertices as you add them. The sooner you make adjustments, the easier the modeling process will be when you add vertices. You should be trying to adjust the vertices positions as soon as you add them, to avoid ending up with a boxy character, which typically happens if you add a lot of geometry while keeping the original flat shape in the geometry where you added new vertices.

4. Select the base's diagonal face, and extrude it to create the preliminary shape for the thumb.
5. Select everything, right-click, and select the Subdivide tool. After that, jump to vertex selection mode, right-click again, and select the Smooth Vertices tool. This step gives you the geometry that you need to build the base of the hand model, but you'll have to move vertices around and adjust them so they have the desired shape.

Caution

People tend to fall into two major pitfalls when modeling hands: They model so that the thumb grows from the side of the hand instead of partially from the front, and they make the other four fingers start at the same height instead of having an arch in the knuckles area. Both of these errors can make the hand look very unnatural.

6. Delete the top faces (the base for the fingers), and on the back of the hand, add two cuts so that you end up with the same four bases for the fingers.

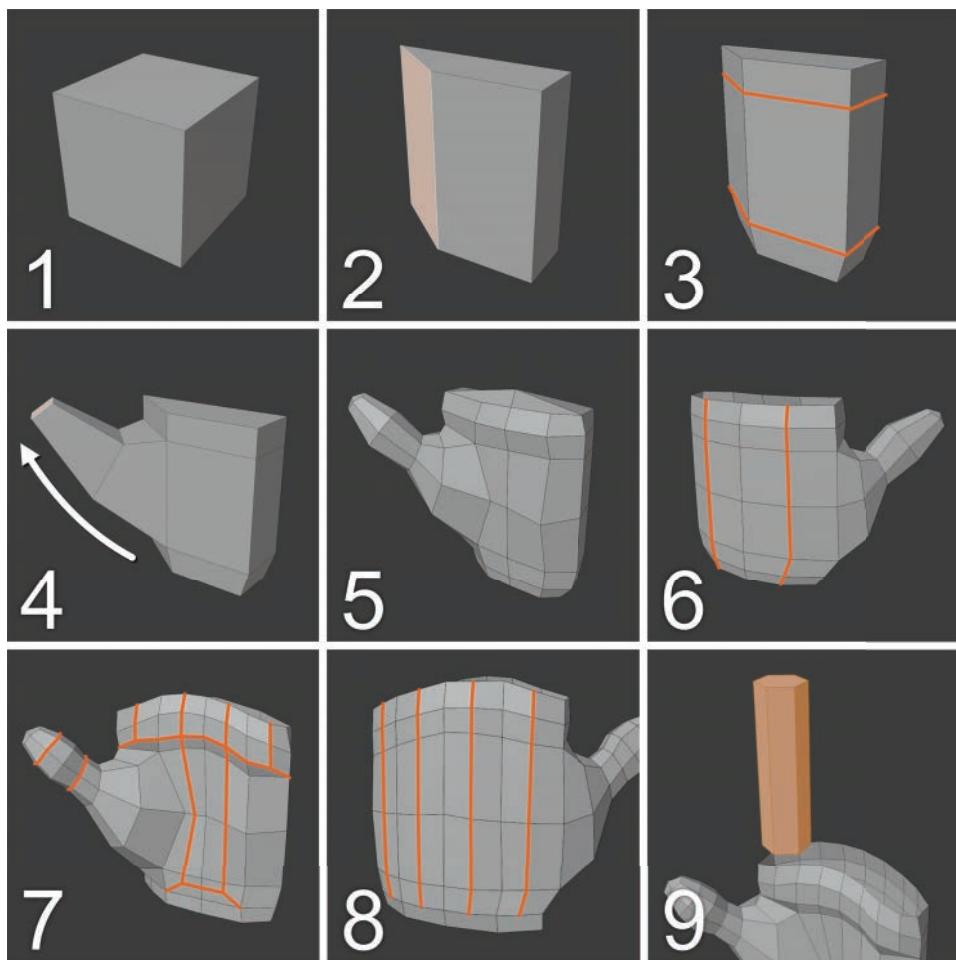


Figure 7.21 Steps for modeling Jim's hand

7. Make some cuts, as shown in step 7. The purpose of these cuts is to create two edges on each side of the hand to prepare the geometry for connecting the fingers later. The outer cuts are made in such a way that they stay at the top of the hand; the central cuts go almost to the wrist to help define the shapes of the palm. Also, add a couple of loops to the thumb to give it some detail.
8. Make some similar cuts on the back of the hand to help define the tendons and knuckles. Look at how the cuts end before reaching the wrist (image 8), which allows you to extrude the flap of the glove with fewer edges later and then arrange the topology with the Knife tool in a later step.

9. On the top of the hand, create a cylinder with six vertices, and fill the covers with an n-gon. Select the bottom face of the cylinder—the one close to the hand—and delete it to leave a gap between the vertices at the base of the cylinder and the hand. This cylinder will serve as a preliminary shape for the fingers. You'll model one of the fingers and then duplicate and modify those duplicates to create the other fingers.

Adding the Fingers and Wrist

As you've realized by now, modeling hands is quite challenging, and it's difficult to get them right. Follow the steps shown in Figure 7.22 to add the fingers and wrist and to finish the hand modeling. From time to time, it's a good idea to add a Subdivision Surface modifier to take a look at the smooth shape that results.

10. Make a few cuts in the finger to define the articulations. Join the two vertices at the top to convert the n-gon to two quads. You can delete the n-gon at the bottom, as you won't need it.
11. Add some more loops to the finger and define its shape.
12. Select the entire finger (you can select groups of connected faces by pressing **L** or select the vertices connected to a selection by pressing **Ctrl+L**), and duplicate it three times. Move, scale, and adjust the duplicates to fit on the hand and create the rest of the fingers.
13. Activate Auto Merge (on the 3D Viewport's header, next to the Options button), and use the vertex snapping tools to drag the vertices at the bottom of the fingers to the corresponding vertices at the top of the hand. Then adjust the vertices to make the fingers look as natural as possible. Also, add another loop at the base of the thumb.
14. Extrude the base of the hand down to create the shape for the glove's wrist flap.

Tip

The extruded edges may not be as circular as they should be. If you activate the LoopTools add-on (included in Blender), you can use the Circle tool to select that loop and convert it to a perfect circle. Then you can scale it and adjust it to become an oval that is more typical of a wrist.

15. Add two loop cuts to the flap to define the shapes a little more.
16. Remember those unfinished cuts from Step 8? Use the Knife tool (**K**) to arrange them, making some cuts like the ones shown in Step 16's image. Then, if the shape gets a little messy, you can select the entire area and use the Smooth Vertices option.

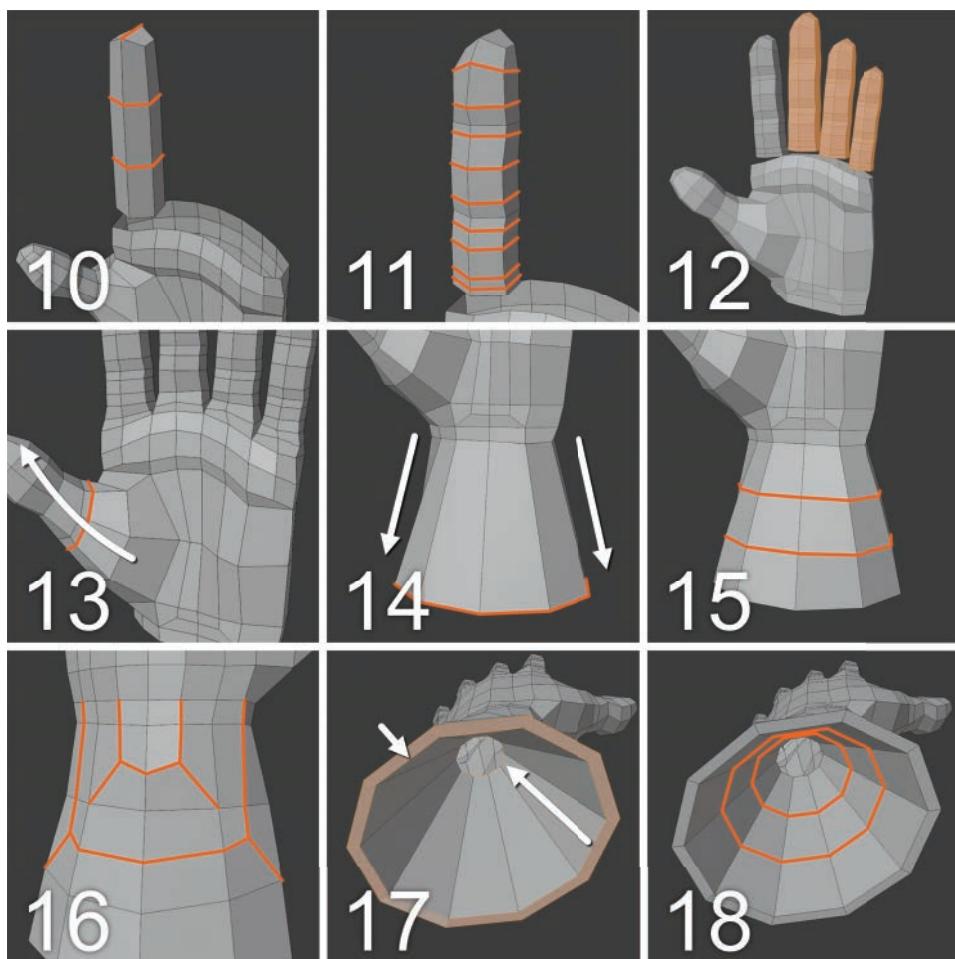


Figure 7.22 Final steps in modeling Jim's hand

17. Select the bottom loop of the flap, extrude it slightly to add some thickness to the glove, and extrude it up again to the interior of the wrist. This step prevents an empty space between the arm mesh and the glove.
18. Add a couple of divisions to the interior of the flap so that if you need to deform it, you'll have enough geometry.

At this point, you may want to adjust the overall shape of the hand. Unhide the rest of the models (if you hid them previously); scale, rotate, and move the hand to place it where it needs to be according to the arm; and make sure that it has the right scale. When the hand is in place, you can mirror it on the other side. You can use the same procedure you used to mirror the eyes (using the mirror tool with **Ctrl+M** and the

3D cursor), or you can use the same method you used for the boots (adding a Mirror modifier, using an empty to define the center of the reflection; you could even use the same empty object you created to mirror the boots). In Object Mode, you can press **Ctrl+A** to apply the rotation and scale to ensure that the hand mirrors as expected. Figure 7.23 shows the results up to this point.

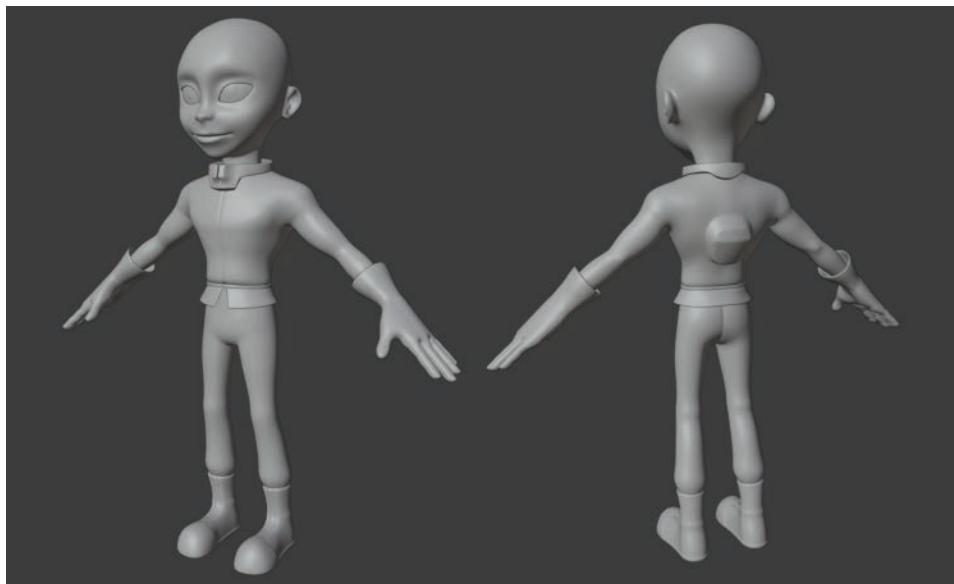


Figure 7.23 This is what Jim looks like so far. Only a few more details are needed.

Modeling the Cap

Next, you create Jim's cap. Modeling the cap should be easy, but it involves a few tricky steps.

Creating the Base of the Cap

Here are the first steps in building Jim's cap (shown in Figure 7.24):

1. Create a UV Sphere with eight segments and six rings. Delete the bottom and left sections of the sphere, and add a Mirror modifier. Scale the object on the Z-axis so that it's no longer perfectly spherical.
2. Convert the selected faces from triangles to quads by using the Dissolve Faces tool (**X**). Repeatedly select two adjacent triangles and invoke the Dissolve Faces tool until the four triangles are converted to quads. (You can also press **F** with two triangles selected for a similar effect.) Or you can add a Subdivision Surface modifier.

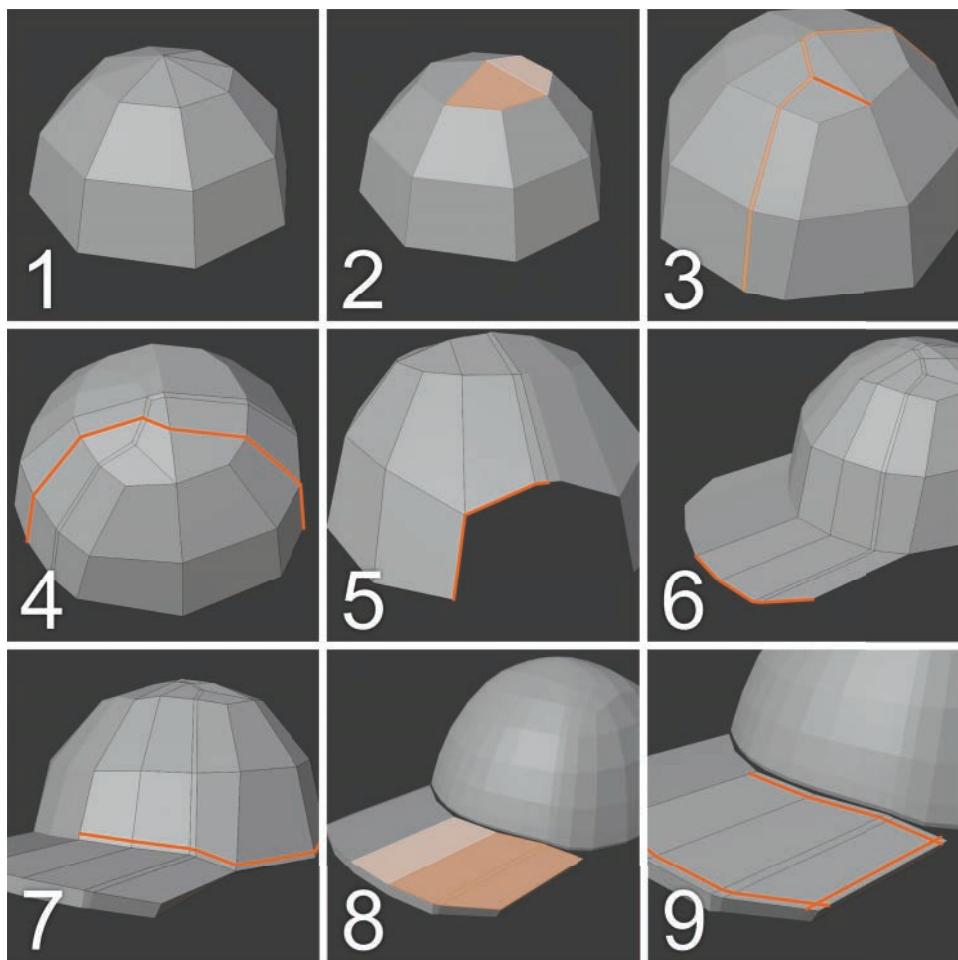


Figure 7.24 The first steps in modeling Jim's cap

3. With the Knife tool (**K**), make two cuts through the cap, similar to those in image 3. The two cuts should be very close to each other so that when the model is subdivided, the seam will be sharp. At the top of the cap, the inner line touches the center of the mirror, and the other line continues through the cap's center to its other side.
4. Perform a loop cut, and adjust the resulting vertices to smooth the cap's shape.
5. On the back side of the cap, delete the two central faces at the bottom.
6. Go back to the front side, and extrude the front edges at the bottom to create the cap's visor.

7. Move the central faces of the cap's front up a little to make the visor bend up in the middle. Also, add a new loop near the bottom all around the cap's body.
Add a Solidify modifier before the Subdivision Surface modifier in the stack (before the Subdivision Surface modifier), and adjust its thickness until it looks right to you.
8. Select the visor's faces, and press **P** to separate them from the cap's body. Remember that doing this after adding the modifiers to the original model gives you the modifiers in this new model as well.
9. Add some new loops near all the visor's borders to help define its shapes.

Adding Details to the Cap

Now that you have the cap's base and visor, keep going and add some details to them. Follow these steps (shown in Figure 7.25):

10. Move the sides of the cap's visor back until there is no empty space between them and the cap's body. For this step, enable the Subdivision Surface modifiers in both models to get a more accurate visual of the end result.
11. Create a little sphere, scale it on the Z-axis, and place it at the top of the cap.
12. Go back to the cap's body, and use the Knife tool (**K**) to create a new loop around the hole at the back of the cap. If you have to create some triangles or n-gons to perform that cut, do so, and merge the vertices afterward to leave only quads.
Also, create a new loop at the bottom of the cap.
13. Make an extrusion to create the strap on the back of the hat, and divide it once to give it more definition.
14. Select the two strap faces, and separate them to a different object by pressing **P** and choosing the Selection option. Make one extrusion to the left, adjust it to the cap, and cut a loop very close to the outer edge so that the shape will become more defined when you subdivide it. You can also adjust the Solidify modifier's thickness and offset to make the extrusion go outward instead of inward.
15. Select one of those outer faces of the strap, duplicate it (**Shift+D**), and separate it to a different object (**P**). You'll use this face as the support for the strap.
16. Make some extrusions and cuts to create something similar to what you see in image 16. You may also need to adjust the Solidify modifier's thickness.

Tip

Keep in mind that in the modifiers' options, you can activate the Subdivision Surface and the Solidify modifiers to see their effects while you're in Edit Mode, which gives you instant feedback while you're modeling.

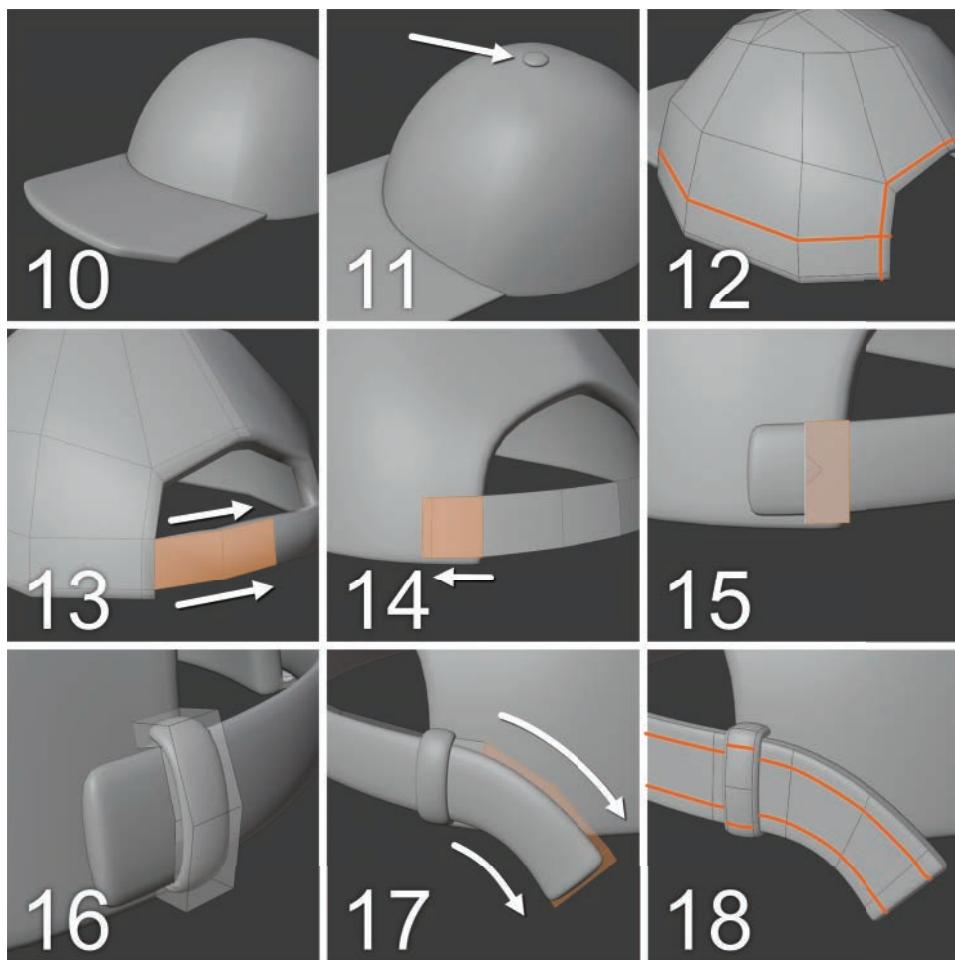


Figure 7.25 Adding some cool details to Jim's cap

17. Make one side of the strap a little loose. If you're in Edit Mode, switch to Object Mode by pressing **Tab**, and apply the Mirror modifier. (You can apply modifiers only when you're in Object Mode. After a modifier is applied, its effect is part of the object's original mesh.) After you apply the Mirror modifier, return to Edit Mode, and perform some extrusions on one side of the strap. Move and rotate the extrusions so that they hang down a bit.
18. At the top and bottom of the strap and support shapes, add new loop cuts (**Ctrl+R**) so that the shape is much more defined when subdivided. Another thing you can do at this point is select every object that will be part of the cap and then select

the cap's body. Press **Ctrl+P**, and choose Object from the pop-up menu. This step connects all the objects to the cap's body in a parenting operation. Now you need only select and move the cap's body; the rest of the pieces will follow along. In image 18, you can see the effect of all of the modifiers (Mirror, Solidify, and Subdivision Surface) enabled in Edit Mode.

The cap is finished; you just have to move it, scale it, and place it on top of Jim's head. Follow the reference images to do that, or at this advanced point of the modeling process, ignore the references and place the cap where you think it fits the 3D model best!

Modeling the Hair

Modeling hair is very complex. There are a lot of ways to create hair, each producing a different result. You can model hair with flat surfaces, each one of them representing a lock of hair, perhaps with a texture applied to them. Another option, which is the most realistic, is to use hair particles. Select the areas of the head where you want to have hair and add a particle system; Blender generates hair that you'll be able to comb, cut, and style. After that, you can even simulate the effects of gravity or wind on hair, which is a pretty complex operation (and one that requires a powerful computer to create, simulate, and display it).

In this project, for Jim, you'll be creating mesh hair—basically, modeling the hair shapes manually with polygons and adapting the meshes to the character's hairstyle.

Tip

In 3D, there is a general rule of thumb: If it won't be seen, don't create it! If Jim is always going to wear a cap, why should you create the hair under the cap? In this section, you create Jim's hair—but only the parts that will be visible.

Shaping Locks of Hair

In Figure 7.26, you see the basic steps to follow to start creating locks of hair:

1. Select the top faces of the head; the hair will be created from them. Duplicate them (**Shift+D**) and separate them into a new object (**P**). When the object is separated, apply the Mirror modifier (click the Apply button in the modifier's options), as from this point on, you'll be doing different things on each side of the head to make the hair look more realistic.
2. Delete some of the faces from the sides of the head to prevent every lock of hair from extruding from the same height on the scalp. (In image 2, the Subdivide Surface modifier has been temporarily disabled so that you can better see what's going on, but it is enabled again in the following steps.)

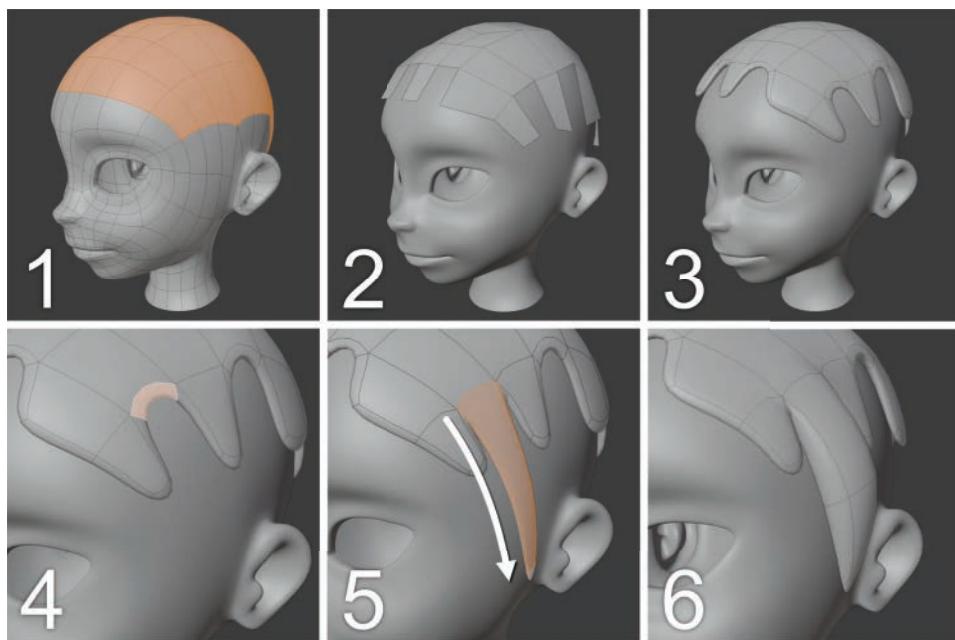


Figure 7.26 The first steps in creating Jim's hair

Tip

At this point, to make things a little easier, you can send all the objects you've created in the scene to one collection (press **M** and select one of the existing collections, or create a new one if needed), the cap to a second collection, and the hair to a third one. This technique allows you to work on the hair and show or hide the components of the cap very quickly from the Outliner.

3. Select all the faces of the scalp, go to the Faces menu (**Ctrl+F**), and choose Solidify. Adjust the Solidify options. You may want to select everything again (**A**) to scale the new model up or down as needed to give the hair just enough thickness to stay on the head's surface. Use **Alt+S** to scale the faces on their normal directions (an easy way of making an object “inflate” itself).
4. Select some of the faces created by the thickness of the Solidify tool. You'll be extruding the locks of hair from those faces.
5. Make a couple of extrusions, and scale the final face down a lot so that it will look like a spike when subdivided. Adjust the vertices to reflect how you want the lock of hair to look. The lock in image 5 is adjusted to the face's surface.
6. Add some thickness to the lock by moving the edges that are on the lock's outer side.

Continue to do the same thing all around the head. Turn on the visibility of the cap layer now and then to see how the hair fits with it.

Adding Natural Details to the Hair

To make the hair look even more natural, take the locks of hair you created in the preceding section, and adjust their vertices and edges to make them overlap other locks of hair (see Figure 7.27).



Figure 7.27 Overlapping locks make hair look a lot more natural.

When you've created and overlapped locks of hair all around the head (which may take a while), you'll probably have some empty areas. To fill those areas, select a lock of hair, duplicate it several times, and adjust its vertices to layer it over other locks of hair, placing it above or below them (see Figure 7.28). Figure 7.29 shows you how the geometry of the hair looks so far.

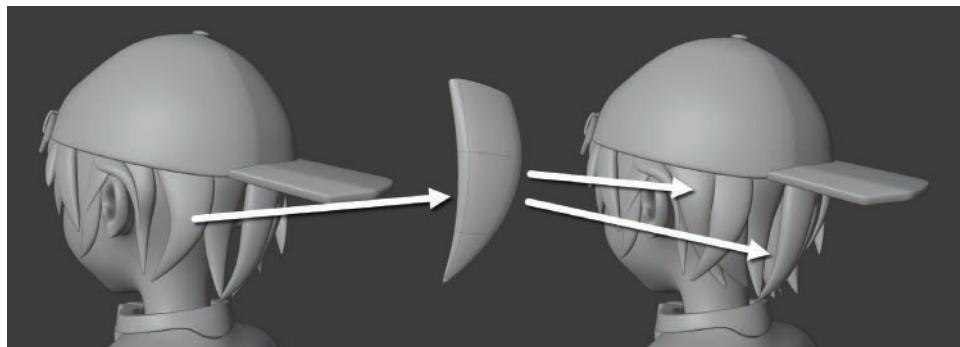


Figure 7.28 Duplicating locks of hair and placing them to cover empty spaces

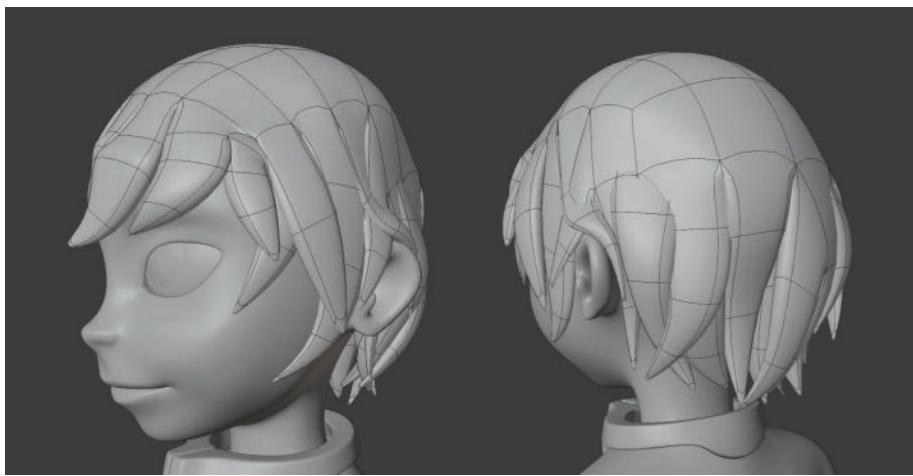


Figure 7.29 The result so far gives you an idea of how the vertices look.

Recall from the reference images that the cap is on backward, and the hair at the front of Jim's head that protrudes from the cap's hole will be a little tricky to model. You can select a lock of hair, similar to the one you duplicated before, and place it on the cap's hole. Add some loops to it, adjust it to fit the shape you want it to have, and make the lock large enough for its base to cover a good portion of the hole. After that, you can duplicate, scale, and rotate to adjust the new locks of hair and cover some of the remaining parts of the hole, this time making the locks much smaller. Try to cover the hole with the big lock of hair and two or three of the smaller locks. In Figure 7.30, you see the result of adding that aspect of the hair.



Figure 7.30 Adding the hair that will protrude from the hole in the cap

As you can see, modeling hair is not an easy task, especially because hair needs a lot of adjustments to fit to the head and the cap. The objective is to make it look as natural as possible until you're ready to move on and add the final details.

Modeling the Final Details

After all you've done throughout this chapter, you should know how to use the modeling tools and should be able to interact pretty well with meshes. In this last section, I briefly explain how I modeled some of these final details and added them to the previous models, but I don't provide any step-by-step instructions. The images show the results so that you can check them out and get ideas about other ways to work. Consider this section to be an exercise: Study the results, try to use your acquired knowledge about Blender modeling tools, and figure out how to model these final details on your own.

The details that are modeled in this section (and you can create more if you want to, of course) are the eyebrows, the communicator in Jim's ear, the badges for the chest and the cap, Jim's teeth and tongue, and a couple of details in the clothing.

Eye brows

Creating the eyebrows was pretty simple. I selected three edges on top of the eye that had the eyebrow shape. I duplicated and separated them in a new mesh. I extruded the edges up to have some shape and thickness, and I moved the vertices around a little to create the eyebrow's shape. Then I applied a Solidify modifier with a little thickness before adding the Subdivide Surface modifier. It's important to keep an eye on the modifiers' order. If you place Solidify after the Subdivide Surface modifier, the result will be different. In Figure 7.31, you can see what the eyebrows look like.



Figure 7.31 Jim's eyebrows added

Tip

For this detail and some of the details that follow, mixing several modifiers is quite helpful for achieving the results you need easily and quickly. There is no point in creating vertices manually to add thickness to a flat model when you can do it with a modifier that allows you to turn its effect on or off and control the thickness at any moment.

Communicator

The communicator earpiece was modeled starting from part of the ear. When you need to make one object fit another one, as in this case, it's a good idea to create the second object from a part of the first one, which ensures that the geometries fit together. Just select the faces of the ear that can be useful for the communicator and then duplicate and separate them, as you've already done several times. From the new object, model and give shape to the communicator, extruding, beveling, and using the modeling tools that you see fit. The antenna (see Figure 7.32) was created with two cylinders.

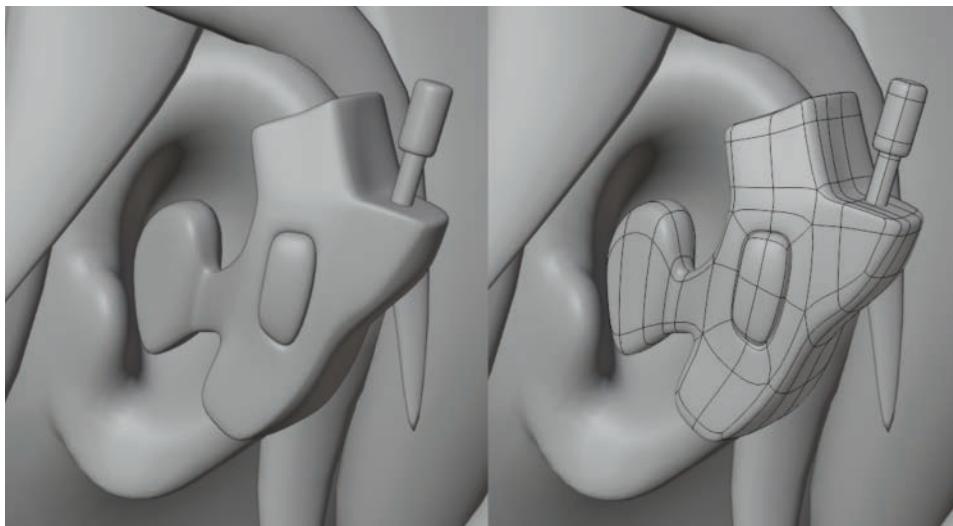


Figure 7.32 Communicator added to the ear

Badges

The badges take advantage of another modifier, Shrinkwrap, which lets you project an object over the surface of another object. Just place the object close to another; add the modifier; and in the object's name field in the modifier options, select the one over which you want to project the original object. (Alternatively, click the eyedropper next to the text field and then click the object to select it.)

Create and place the badge on the chest, which is a simple flat object (and can also be mirrored so that you can work on only one of its halves), to the surface on which

you want to project it, such as over the chest. Play with the options for the Shrinkwrap modifier until you’re happy with the projection; then apply a Solidify modifier. What’s cool about this feature is that if you applied thickness to the model, it would be lost in the projection with Shrinkwrap, but because the Solidify modifier adds thickness after the projection, it works perfectly!

Finally, add a Subdivision Surface modifier. Mixing modifiers to achieve a particular effect is a very powerful technique, as you see in Figure 7.33.

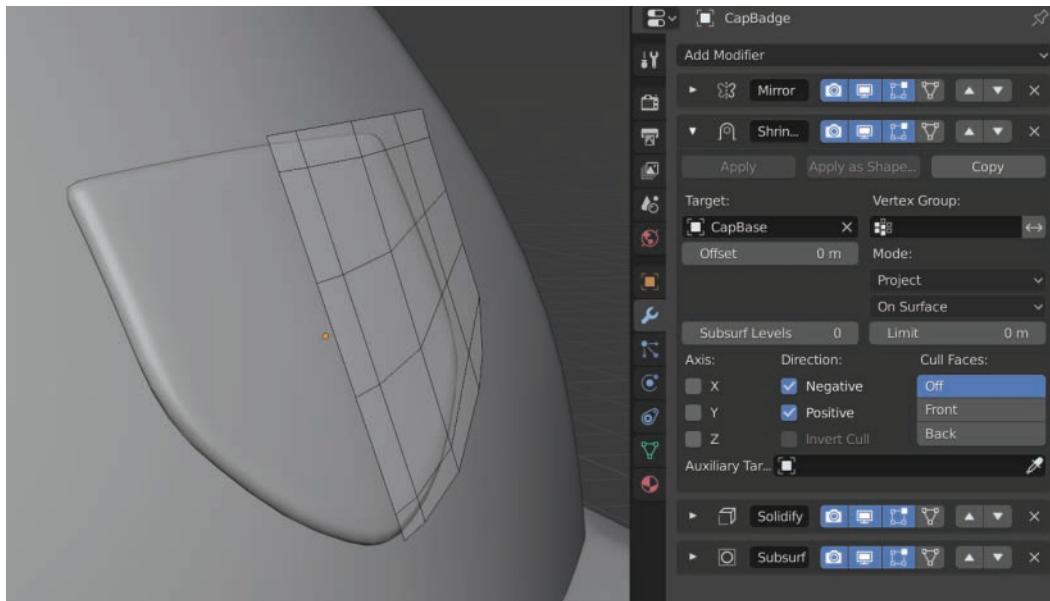


Figure 7.33 I used several modifiers to simplify the process of modeling the badge.

You can duplicate the badge, place it next to the front side of the cap this time, and choose the cap as the projection surface for the Shrinkwrap modifier.

Teeth and Tongue

The teeth and the tongue are simple objects to model. Two curved surfaces with thickness will do for the teeth, and the tongue is also a very basic shape. In Figure 7.34, you see these features and their basic topology. In the figure, the teeth are apart so that you can see the tongue, but they’re really touching inside Jim’s mouth.

You should be able to do something similar with the skills you’ve learned so far. These features are not very complex but are stylized to fit the rest of the character (I didn’t model every tooth, as such a level of realism wouldn’t fit the simplicity of the rest of the model) and are there so when Jim opens his mouth, you see something inside. When you’re working on these models, you may want to adjust the shape of the inner mouth so that it doesn’t cover the teeth or create other problems of intersecting geometry.

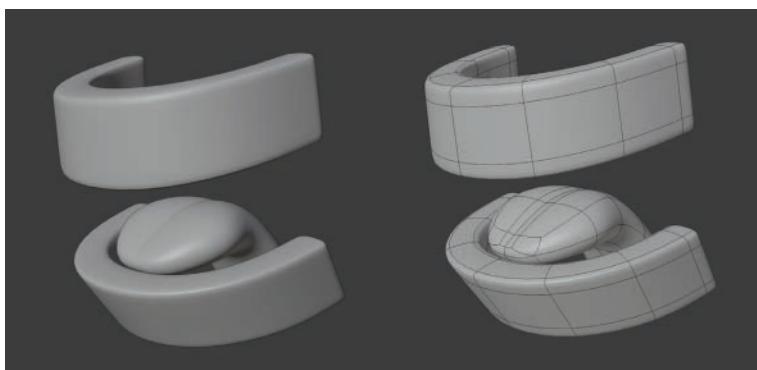


Figure 7.34 The teeth and the tongue, ready to go into Jim's mouth

Other Clothing Details

In the reference images, some details were added to the clothing. The technique I used was similar to the one I used for the eyebrows: duplicating and separating parts of the clothes, adjusting them, and applying modifiers such as Solidify to add some thickness.

In Figure 7.35, you see Jim up to this point. Looking good!



Figure 7.35 Jim is looking pretty cool!

Summary

Wow! This chapter was tough, but if you've come through it, you've learned a lot for sure, and you should have your finished model. You know how to model the different parts of a character step by step, and (I hope) you've developed some insights into how to use Blender's modeling tools. Also, if you followed the instructions and tried to use the keyboard shortcuts, you may already be more efficient and able to remember a lot of the keystrokes for the basic tools. Modeling with polygons may be a technical task when it comes to topology, but if you find it interesting, it can be a very enjoyable experience. When you're done and see the finished model, of course, modeling is quite rewarding.

One last thought: Don't get frustrated if the result is not as good as you expected. If this model is one of your first models, it's normal for it to not be really great. Greatness comes with practice, so now that you understand the tools and the technical part of the process, you can practice to improve the artistic part and get better results next time.

Exercises

1. Take the model further by adding some more details, such as lines on the cap or more details in the clothes.
2. Explain why having a good topology is essential for an animated model. List some rules to follow to ensure that you have a good topology.

IV

Unwrapping, Painting, and Shading

8 Unwrapping and UVs in Blender

9 Painting Textures

10 Materials and Shaders

This page intentionally left blank

8

Unwrapping and UVs in Blender

Unwrapping is a fundamental step in 3D design that comes before adding textures to a 3D model, and it creates UVs. Without UVs, the 3D software (Blender, in this case) wouldn't be able to determine where in the model's surface an image should be projected. *UVs* (the 2D counterparts of X, Y, and Z coordinates in 3D space) are the internal 2D positions of the vertices of a 3D mesh; they define how a 2D texture will be projected on the mesh's surface. One way to think of the unwrapping process is to visualize a globe. Imagine taking the Earth's surface and flattening it into a map. This process of converting a 3D shape to a 2D surface is referred to as *unwrapping*.

Unwrapping may look a little odd, and it's a task a lot of people dislike or fear, but that's usually due to their lack of understanding of how it works. Sometimes, unwrapping can indeed be a little tedious, but if you learn to like it, unwrapping can be a fun part of the process! Watching how everything starts to make sense and work properly can be a rewarding experience, but be aware: You'll need some patience.

Fortunately, Blender provides some helpful tools to unwrap your models, and people who come from other software usually love the way that the unwrapping process works in Blender. (There are even glowing reports from professionals working on Hollywood films who use Blender for UVs.) But as with almost everything else in Blender, unwrapping is done quite differently, so if you come from other software, forget the way you worked before, and open your mind for a little while!

Seeing How Unwrapping and UVs Work

Something tricky happens with textures, which define the colors of your models' surfaces: You have a 3D model, but the image texture is 2D. How do you "paint" a 3D model with a 2D texture? The answer is that you use UVs. The 3D model has its vertices located on the X-, Y-, and Z-axes, but for Blender internally, they're on the U- and V-axes as well, and the U- and V-axes are 2D positions, made to be used for texture projection. In the UV Editor, you can access those UVs and adjust them to define how a texture is projected onto a 3D model.

Unwrapping (also called *UV mapping*) is the process of adjusting an object's UVs so that the texture projection on the 3D model works properly. The process is probably easier to understand if you see how it works.

When you were a kid, did you take a piece of paper and, by cutting and folding it in a certain way, end up with a 3D shape, such as a cube? Unwrapping can be explained with that example, only in reverse order: You unfold a 3D model into a 2D shape by using the UV Editor and then convert it to a plane. (Unwrapping won't affect the shapes in your 3D model at all; the process happens under the hood.) Figure 8.1 illustrates this procedure.

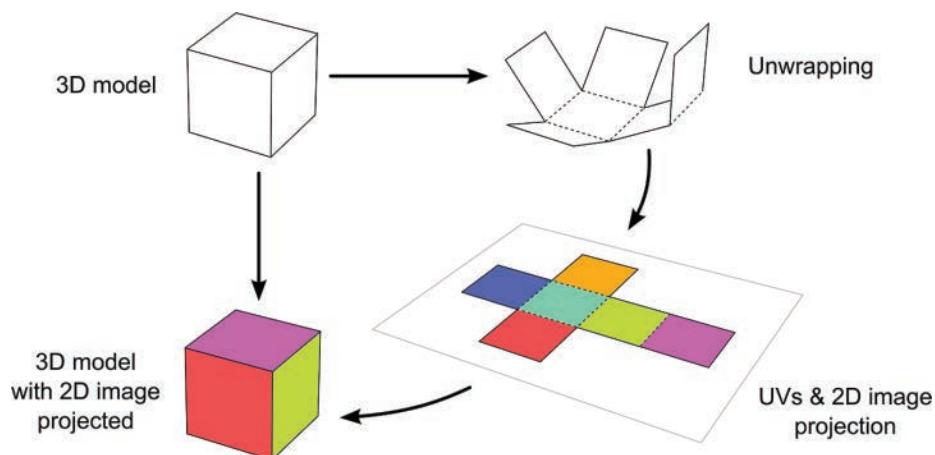


Figure 8.1 The unwrapping procedure, explained visually

As you can see in the figure, unwrapping is like unfolding the 3D model and converting it to a 2D shape. You can project an image onto that bidimensional shape, and the image that you place on that shape will be projected onto the 3D model itself.

Unwrapping in Blender

Now that you understand how unwrapping works, you can explore the tools that Blender provides to use unwrapping and the basic workflow of unwrapping.

To begin unwrapping in Blender, select the part of the model you want to unwrap, and access the unwrapping options (which I describe in the next section). When you unwrap the selection, it appears unfolded in the UV Editor, where you can adjust the UVs, weld them to other parts of the model you've unwrapped before, or place them where you want them to be in the image.

Keep in mind that after unwrapping a model, you usually create textures tailored to those specific UVs. Sometimes, however, you're in a hurry or following a different

workflow, and you need to display part of an image in a specific position of the 3D model. In that case, you can adapt the UVs to the desired image, not the other way around.

In the next section, you look at the different tools and find out how to use them. Then you see how to unwrap Jim's head step by step to understand the unwrapping procedures.

Using the UV Editor

The UV Editor is the part of the Blender interface where you can see and adjust the UVs. Figure 8.2 shows an overview of the UV Editor.

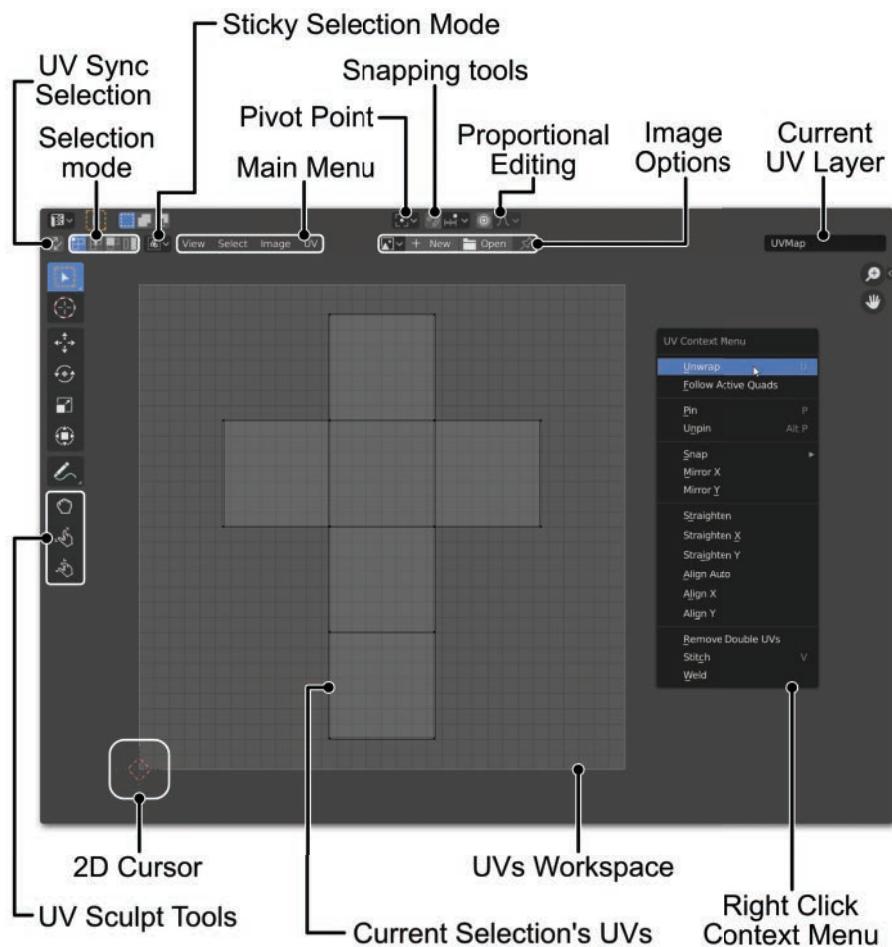


Figure 8.2 The UV Editor and its options

The list that follows explains some of the main options available in the UV Editor:

- **Interface:** This editor is a typical Blender editor. It has a toolbar, the Workspace in the center, and a header on which you can find buttons and menus for most of the options you need while arranging the UVs.
- **2D cursor:** The 2D Editor has a cursor that is similar to the 3D cursor; you can use it to align vertices or other UV elements. Place it with **Shift+RMB** and press **Shift+S** to see the snap options you can use with the 2D cursor.
- **Display panel:** In the Sidebar (press **N** to show/hide), on the View tab, you can find the Display panel, which lets you customize overlays and how you see the UVs. You can even find options to display face stretching, which shows the distortion using colors based on the angle or area of faces in the UV map compared with the 3D model—quite useful for identifying distortion in complicated areas. If the faces are very stretched (a blue face is good, and a turquoise face is not bad, but you should avoid green and yellow faces), the texture will probably look distorted or be of poor quality when you project it onto that part of the surface.

The header provides quite a few options:

- **UV Sync Selection:** This option synchronizes the UVs selection with the 3D model in the 3D Viewport and can be useful in complex models to find out where a specific vertex or face in the model is located on the UVs.
- **Selection Mode:** In the UV Editor, you can switch among Vertices, Edges, Faces, and Islands selection modes. (An *island* is a group of connected faces.) You can also select an island by pressing **L** while hovering the mouse cursor over it.
- **Sticky Selection Mode:** Sticky Selection Mode options are interesting. Suppose that on the UVs, Blender treats each of the faces separately but allows you to select vertices depending on certain conditions—such as the vertices sharing their X, Y, and Z positions—in the actual 3D model. When this option is disabled, and you select a vertex or a face, that vertex or face moves alone; the connected vertices or faces stay in their current places. If they are overlapped in the UV Editor, the Shared Location option treats them as though they were welded on the UVs, but only if they share a location in the 3D model. The Shared Vertex option selects the vertices that share the position in the 3D model, even if they’re separated on the UVs. The best way to understand this option is to try it and see what happens.

Note

To better understand the Sticky Selection Mode options, you have to know how Blender treats UVs. Vertices in a 3D mesh are positioned on the X-, Y-, and Z-axes, whereas vertices in a UV layout are positioned on the U- and V-axes. The difference is that whereas one face in a 3D mesh is typically connected to another, faces in the UV space can be freed from their neighboring faces, so you can apply a different texture to a different part of the model, even though the parts may be adjacent in the 3D model.

- **Main menu:** View, Select, Image, and UVs menus are available. The UVs menu is especially important, as it provides most of the unwrapping tools you'll need.
- **Pivot Point:** The Pivot Point option works exactly the same as in the 3D Viewport. You can select the pivot point from which to rotate or scale objects; that's how you can use the 2D cursor as a pivot point. As in the 3D Viewport, you can press . (period) on your keyboard to access the pie menu that lets you choose the pivot point you want to use.
- **Proportional Editing and Snapping:** You also have Proportional Editing and Snapping tools in the UV Editor to help you manipulate UVs. They work the same way as they do in the 3D Viewport while modeling.
- **Image Options:** You can load images, choose some that are already loaded in the .blend file from the drop-down list, or create new ones inside Blender (such as a solid-color image or a UV test grid, which I discuss later in this chapter). Loading an image through these options will display it as the background of the UVs Workspace, allowing you to adapt the UVs to that image.
- **Pin:** The last option within the image options is a pin icon. If you press this pin, the image displayed in the background of the UVs Workspace will be fixed. Generally, the background image is assigned to objects, which means that if you select a different object, that image will also change. By pinning it, you ensure that the pinned image is always shown, regardless of the selection. To stop that behavior, simply click the pin icon again.

Tip

Loading an image into the UV Editor can be as easy as dragging it from a folder on your hard drive. Drag the image file over the UV Editor and drop it. Blender should load it immediately.

- **Current UV Layer:** The current UV layer is pretty important because in Blender, a single object can have multiple UV layers that you can use independently when building complex materials, so you can use various textures that are distributed differently depending on the UV map they use. UVMAP is the default name of the first UV layer that is generated when you access the unwrapping tools within a model, and most of the time, you'll use only a single map. If at some point you want to create other UV maps, you can go to the Properties Editor and look for the UV Maps panel on the Mesh tab.
- **Contextual menu:** Right-clicking the UVs Editor will display the contextual menu, which provides access to some of the most-used unwrapping tools.

Navigating the UV Editor

Navigation in the UV Editor is pretty simple: Press **MMB** and drag to pan and use the scroll wheel, or drag with **Ctrl+MMB** to zoom in and out. A left click positions the

2D cursor. Apart from that, controls are exactly the same as in 3D Viewport: Click to select; click and drag to move; and press **G**, **R**, and **S** to move, rotate, and scale the selections, respectively.

Certain features from other editors work in the UV Editor as well, such as the Hide and Unhide feature (**H**, **Shift+H**, and **Alt+H**) or the ability to switch selection modes by pressing **1**, **2**, or **3**. (In the UV Editor, you can also press **4** to use the Islands selection mode.)

Accessing the Unwrapping Menus

Blender provides some unwrapping menus and tools that you can find within the interface:

- Select some faces in Edit Mode (usually, when unwrapping, you work with faces), and press **U** to display the UV Mapping pop-up menu. This technique also works in the UV Editor.
- Marking seams is one of the key unwrapping tasks. You can access the tools from the Edge menu (**Ctrl+E**) after selecting one or more edges. You can also access the options for marking seams within the contextual menu while you're in Edge selection mode. (I discuss seams later in this chapter.)
- On the 3D Viewport's header, you can also find the UV menu while you're in Edit Mode.

Working with UV Mapping Tools

In Figure 8.3, you see the UV Mapping menu (press **U** in the 3D Viewport) and the Edges menu (**Ctrl+E**), showing the Mark Seam and Clear Seam options that are essential for unwrapping in Blender.

In this section, I cover the UV Mapping tools briefly so that you have an idea of what they do before you start using them later in this chapter. The following list describes the main UV Mapping tools and how they work:

- **Mark Seam and Clear Seam:** From the Edges menu, select one or more edges; press **Ctrl+E** to access the Edge menu; and choose Mark Seam. This option shows those edges in the 3D Viewport with a red outline. To clear the seam mark, select the edges you want to clear, press **Ctrl+E**, and choose Clear Seam.
- **Unwrap:** This option is the main unwrapping tool in Blender. Press **U** to access the UV Mapping menu, and select Unwrap. This option unfolds the selected faces in the UV Editor, taking into account the mesh's borders and edges that you've marked as seams, and usually gives good results if the seams are placed correctly. Right below the Unwrap option in this menu, you'll see Live Unwrap; you'll learn about that tool later in this chapter.

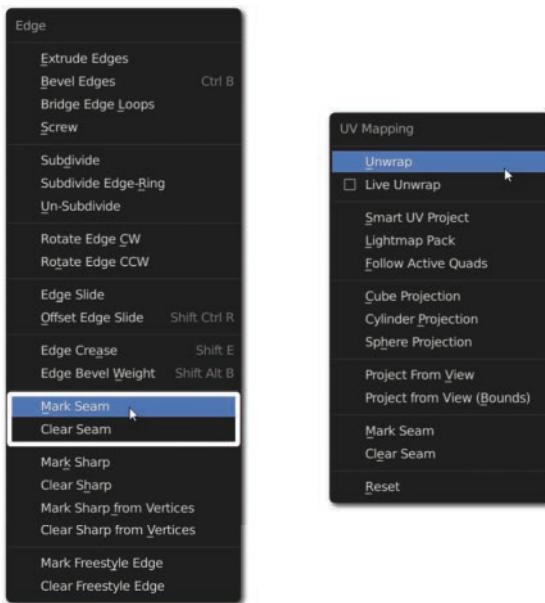


Figure 8.3 Edges menu (Ctrl+E) with Mark Seam and Clear Seam options, and UV Mapping menu (U)

Live Unwrap

Right below the Unwrap option in the UV Mapping menu is Live Unwrap. You can enable/disable two slightly different Live Unwrap options:

- The first is the one mentioned above. When active, it updates the unwrap in the UVs automatically when you mark/unmark edges as seams. If the option is disabled, you have to unwrap manually after changing what edges are seams.
- The second one lives in the UV menu within the UV Editor; it enables real-time unwrapping that adapts to the actions you perform in the UV Editor, such as moving vertices around. It's an interesting option to use to make adjustments and to see how what you do affects the automatic unwrapping.

When you use Live Unwrap in the UV Editor, keep a few things in mind:

- All the unwrapping and previous manual changes you made to the UVs are replaced by automatic unwrap when you use Live Unwrap.
- To use Live Unwrap to move vertices, you need to pin those vertices. Essentially, you pin a few vertices, and when you move one or more of them, all the ones that are not pinned adapt between them. You can pin vertices by pressing **P** when they're selected and unpin them with **Alt+P**.
- If you've already worked on parts of the UVs and are happy with, you can pin them; pinned vertices are not affected by Live Unwrap.

- **Smart UV Project:** With this option, you don't need to mark seams. Smart projection can work nicely for simple objects; it unwraps your object, separating the UV islands depending on parameters you define in a pop-up menu, such as the angle between faces.
- **Cube, Cylinder, and Sphere Projections:** These options are basic tools, but they can be useful at times. These projections use the pivot point of the object and the view angle to work. They unwrap the object by using a geometric shape (as their names imply). After using them, you can find some options in the Edit Last Action panel that may be useful for adjusting the results.
- **Project from View:** This option is quite interesting, as it takes the selection and unwraps it in the UV Editor projecting it as you see it in the 3D Viewport. Your point of view is key, of course, and perspective is maintained as well. If you select the Project from View (Bounds) option, Blender scales the resulting UVs to the borders of the UV Workspace.
- **Reset:** This option takes every selected face and returns it to its original state, occupying the entire UV map.

Tip

To better understand what these unwrapping tools do, try them, and see the results for yourself. Some of them may seem to be more efficient or easy to use than others, and it's very difficult to clearly understand their effects just by reading an explanation. As with every part of the learning process, trial and error are the best teachers!

Defining Seams

Seams are the borders of an unwrapping operation. When the cube in Figure 8.1 (earlier in this chapter) is unwrapped, the black continuous lines are the seams. You can also think of seams in clothing. Before a shirt is made, it's just a series of flat pieces that are later joined into a 3D garment by stitching them together at their seams. In 3D, the most popular unwrapping method involves seams. First, you define the seams in the 3D model; then you unfold the UVs according to those seams.

You can think of seams as the edges that you would cut with a scissor to unfold the 3D model.

Keep in mind that seams are not desirable in visible areas of a model, but they're needed to prevent distortion in the projection of textures. Seams are usually placed in areas where they're less visible. The reason is that when you apply a texture, in the seam area, you see a cut in the texture. That cut happens because where you have a seam, it appears as a border on the UVs. Even though you make the texture continuous, without any cuts, the sizes of the seams on the UVs may not be exactly the same, causing the resolution of the image to change and make the seam visible because of a resolution difference between its sides.

In UVs, the bigger a polygon is, the more resolution it requires in the image. Therefore, the texture displayed in the 3D model is sharper. This density of pixels per

polygon is referred to as *texel density*. It's important to know which parts of your model need more detail in the textures and therefore need more space on the UVs.

In Figure 8.4, you see the effects of seams and how the size of the UVs affects the projected texture. The sizes of the faces on the UVs (left side of the image) and in the actual 3D model (right side of the image) are independent, but the size of UVs defines the resolution of the texture in the 3D model.

Pay attention to how each island in the UVs translates into the 3D model. Both faces in the model have the same size, but the sizes of the UVs of each of those faces have different effects on the texture projection. The smaller UV island takes less resolution from the image, so the texel density is lower on the left side of the 3D model, making the texture appear to be bigger.

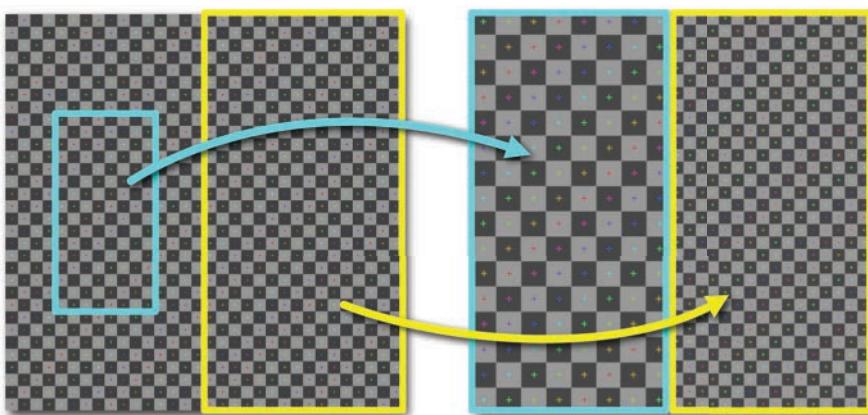


Figure 8.4 The UVs placed on the image (left) and how they affect the projection's resolution in the geometry (right)

Considering Before Unwrapping

Not all the objects you've created for Jim's 3D character have the same properties, so you need to make some decisions before jumping into UV mapping. The list that follows describes some things you need to keep in mind:

- **Meshes that don't require UVs:** If an object is supposed to look like a flat material, with no changes in its surface (which is not usual in realistic models but can happen in simple models for animations), you won't need to unwrap it. Unwrapping defines how an image is projected onto the surface of a model. But if you just need to add a flat color, a material with no texture will do, as is the case for Jim's hair. There are also materials that can use automatic coordinates instead of UV coordinates to project textures, so you wouldn't need to unwrap your model to use them either.

- **Meshes with modifiers:** Some of the meshes that comprise your character may have modifiers assigned. When a mesh uses modifiers that change its geometry, they affect the UVs as well. Take Jim's badges as examples. The badges are made with a Solidify modifier that adds thickness to the model; the UVs are available only for the original geometry, however, so the polygons generated by the modifier won't be able to display the texture properly. In the case of the Solidify modifier, the thickness polygons display the color of the texture's borders in the front, and the back side displays the same as the front side (which shouldn't matter in this case, because the back side is hidden). In such a situation, you must determine whether you'll get the correct unwrapping result without applying the modifiers or whether you need to apply the modifiers and then unwrap all the geometry. The answer depends on the amount of detail you're looking for and on where you need to display the textures accurately.
- **Mirrored meshes:** Mirror is a modifier, and it's especially important when it's used with UVs. If you do the UVs unwrapping, and you're using a Mirror modifier, the mirrored geometry uses the same UVs as the original geometry you unwrapped. Sometimes, you want to have asymmetric textures for an object, so you should apply the Mirror modifier before unwrapping. In other cases, a mirrored texture could work nicely, which means two things: You need to unwrap only half the object, and you need less space in the image to texture that object. In yet another case, you need an asymmetric shape, but the texture could be symmetric, so you could work with a symmetric shape, perform the unwrap operation, apply the Mirror modifier, and make any adjustments to the shape afterward. This way, you have mirrored UVs but an asymmetric shape. (The topology should be the same on both sides of the mesh. If you add or subtract geometry, UVs are broken.) The examples in the following sections, using Jim's face and jacket, will help you understand this option better.

Mirrored UVs

What does it mean to have mirrored UVs? Essentially, two things:

- The texture that you apply to the original side will be reflected on the other side. (Be careful if you have text on the textures, as it will appear inverted on the mirrored side.)
- The UVs of the mirrored side are overlapped with the originals. This saves space in the texture, so you can have more resolution. Instead of having both sides take space, you can unwrap only one side and make it bigger (have more texel density).

Keep one thing in mind: If you apply the Mirror modifier, the UVs will be overlapped, but overlap isn't allowed by unwrapping operations, so if you need to unwrap the model again, that overlapping will be lost.

Note

You need to determine whether it's more efficient to do the UVs and then keep adjusting your model, or whether it's better to apply modifiers and do the UVs afterward. The answer depends on your model, what you need to do to it, and what's most comfortable and efficient for you.

Working with UVs in Blender

In this section, you go through the process of unwrapping Jim's head step by step, and you learn how to use the basic unwrapping tools. Then, with some basic instructions, you'll be ready to unwrap the rest of the objects for your character. In this case, you won't use a mirrored texture for the face, so you'll need to unwrap the full face at the same time. To do this, select the face and apply the Mirror modifier in Object Mode.

Marking the Seams

The first step in unwrapping is marking the seams to tell Blender how you want your UVs to be unfolded. In Edit Mode (**Tab**), select the edges shown in Figure 8.5. You don't need to mark them all at the same time; you can mark some of them now and then mark seams in other areas as you proceed. To mark edges as seams, press **Ctrl+E** to access the Edge menu and select Mark Seam.

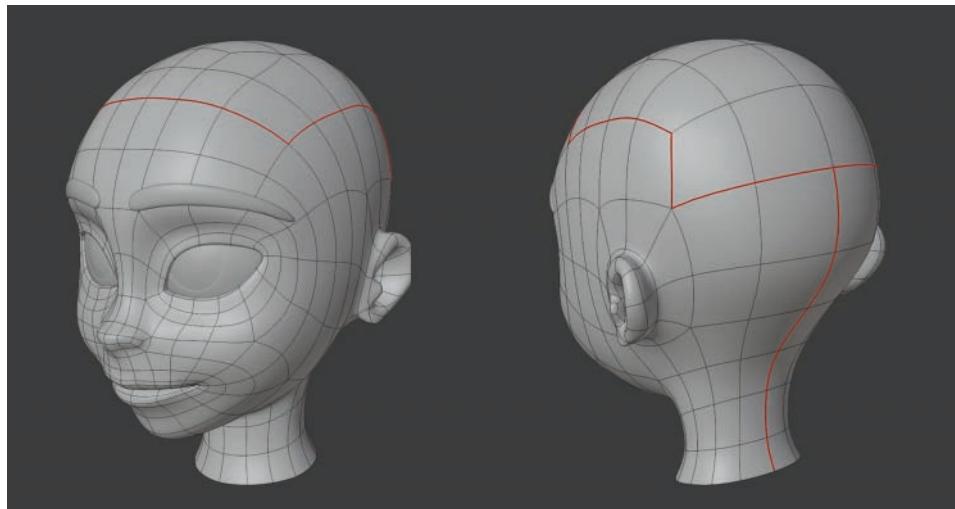


Figure 8.5 Seams marked to unwrap Jim's head (edges shown in red)

Pay attention to how the edges selected to be the seams of the UVs are placed in areas that won't be very visible. Those edges appear on the back and sides of the head

and over the forehead, where they'll be almost completely hidden by the hair. The closed seam selected at the top of the head is there to make that part an isolated UV island. After all, that part will always be hidden under Jim's cap, so it doesn't need a lot of definition. As a result, you can use more space in the UVs for the areas that matter and need more resolution in the texture.

Also, although you can't see it in Figure 8.5, a loop is marked as a seam in the inner part of the lips so that on the UVs, the geometry of the face's interior is separated from the face's exterior. Remember that you can press **Alt+RMB** to select edge loops.

Tip

For marking seams, the Shortest Path option is a useful selection method. When you need to select multiple edges on a single line, select one and press **Ctrl+RMB** on another edge. The edges between the two edges you've selected are also selected, allowing you to select long lines of edges very quickly. You can press **Ctrl+RMB** again and again until you have the entire desired edge loop selected. This method also works in vertex selection mode.

To speed things even more, after doing a Shortest Path selection, access the Edit Last Action panel or press F9, and choose Tag Seam in the Edge Tag option. Thereafter, when you use the Shortest Path selection, those edges are automatically marked as seams. Remember to set this option back to Select when you're done with the seams!

After setting up the seams, you may prefer to not see them marked in red. You can hide them from the Overlays menu in the 3D Viewport's header, under Edit Mesh.

Creating and Displaying a UV Test Grid

At this point, you could start unwrapping, but instead, you're going to create a UV test grid so that you can see how applying a texture to the face will look before unwrapping. Using the test grid helps you visualize how unwrapping affects the projection of the texture.

A *UV test grid* is a basic image made exclusively to test how the UVs of a mesh work. It's an image with a grid that, when it's projected onto the 3D model, provides a lot of information. The size of the grid shows you which parts of your object are using more of the texture; the smaller the grid, the more resolution (more pixels) the texture has in that area. Using the test grid, you can adjust the size of every part of the object to be more or less consistent, and you can set a smaller grid on those parts where you need more details. It's also useful to see the grid's distortion. If you notice that at some point, the grid is becoming heavily distorted, you can try to solve the problem by adjusting the UVs. Using a UV test grid, you can also see where the seams are and how well they work, and you can make sure that they're barely visible.

The UV test grid can display colors as well as letters or numbers. This feature tells you which part of the UVs is being shown on a specific part of the model by the color, number, or letter it displays on its surface.

Creating a New Image for a UV Test Grid

Blender has two types of UV test grids that you can generate and use in your models: a UV grid, which resembles a chessboard, and a color grid, which includes colors and letters to help you spot which part of the texture you’re using when you see it projected. To create these grids, click New Image on the UV Editor header, choose New Image from the Image menu, or press **Alt+N**.

In Figure 8.6, you see the New Image menu, which allows you to create images to paint on or UV test grids. You can set the image’s name (Untitled by default), its resolution (if you generate a UV test grid, the resolution will affect the number of squares in the grid—the more resolution, the more squares will be present), and its color. The color applies only when you select Blank as the Generated Type. If you select one of the UV test grids as the Generated Type, Blender ignores the color setting.

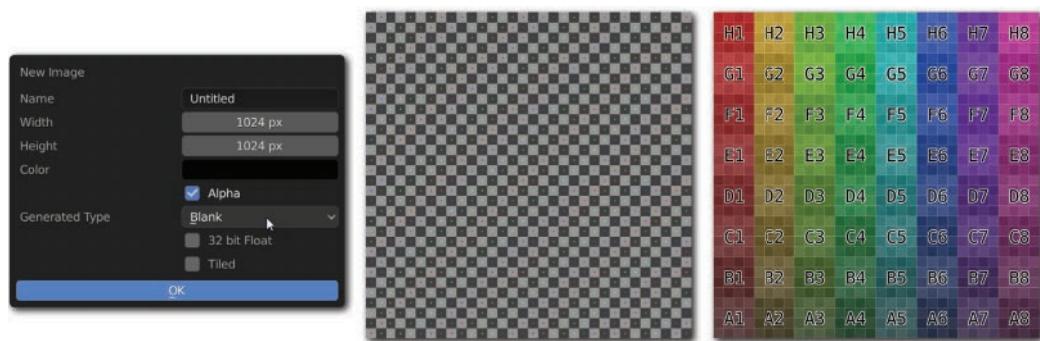


Figure 8.6 Generating a new image in Blender, with the New Image menu (left), a UV grid (center), and a Color grid (right)

When you’ve made your selections, click OK to generate the image. You can change the image’s name from the header and save it from the Image menu. If you open the Sidebar (**N**), in the Image panel, you can rename the image and access its parameters. Because the image is generated by Blender, you can change its type even after it is created and switch between a UV grid or a color grid (or a blank image, or course).

Displaying the UV Test Grid in Your Model

In this section, you’ll learn how to test whether the UVs are working properly by displaying a texture on the 3D model.

Create a new material from the Material tab in the Properties Editor, and call it something like `uv_test_mat`; this way, you’ll always have a UV test material in your scene, ready to be applied to the object with which you’re working. Inside the Material options, in the Color parameter, click the button that has a little dot at the right of the color selector, and select Image Texture from the list. Select the UV test grid you just created from the drop-down list. Now go to the 3D Viewport, and use Material

Preview Mode to see how the UV test grid looks. (You could also use Rendered Mode, but at this point the scene isn't set up with lights, so it would be too dark to work with.)

Unwrapping Jim's Face

Unwrapping Jim's face is simple. Select Jim's face, and in Edit Mode (**Tab**), select all the faces (**A**). Press **U**, and select the first option, Unwrap. What you see should be similar to Figure 8.7.

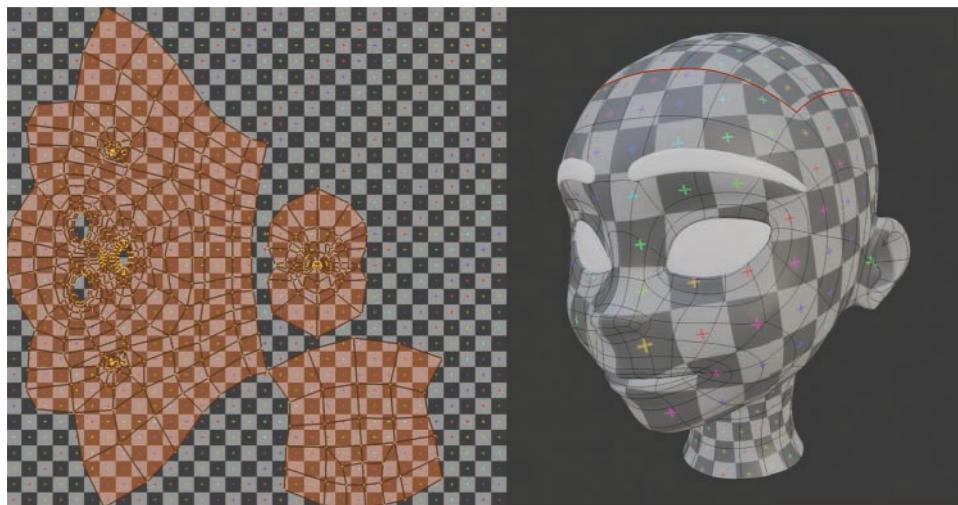


Figure 8.7 After unwrapping Jim's face, you see the UV map in the UV/Image Editor (left), and the UV test grid looks much more uniform (right).

After unwrapping the model, you see how the UVs have been unfolded. The face is flat, and you see the other two islands: the top of the head and the mouth's interior, as you defined them previously with the seams. Also, the face has been unfolded by using the seam from the back of the head.

When you have a test grid or an image in the background of the UV Editor, it may be difficult to see the UVs accurately. When an image is loaded in the UV Editor, on the right corner of the header, next to the UV Layer, is a button that appears only when you load an image for the UV Editor's background. You can see this button in Figure 8.8.

When you click that button, a few options appear that allow you to change how the image is displayed, including one that shows the current UV Texture Alpha channel (image transparency) and a white background when there is no Alpha to display, letting you see the UVs better. Another way to improve UVs visualization is through the

Sidebar (press **N** in the UV Editor to show/hide this part of the interface), where you can find some display options that may help.

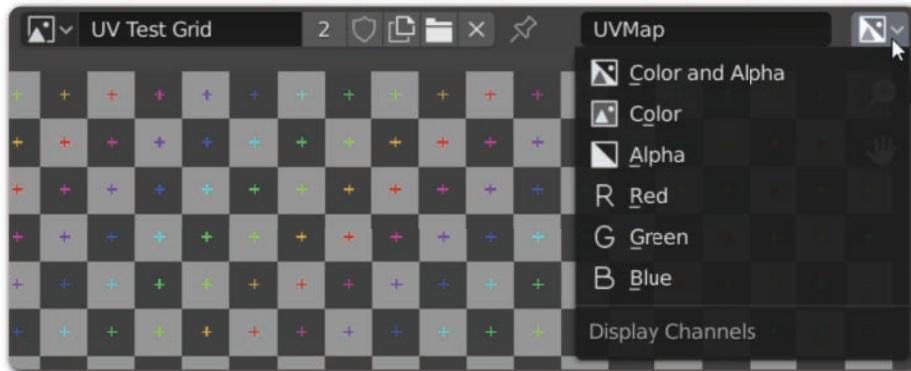


Figure 8.8 UV Editor's background image display options (channels)

Note

Remember that the UV Editor shows only the UVs of the parts of your model that are selected in 3D Viewport, and only while you're in Edit Mode. At first, this situation can be confusing for people who come from other software and are used to always seeing all the UVs. If you activate the sync between the 3D and UVs selection in the left corner of the UV Editor's header, you'll be able to see all the UVs, but if you want to see all the UVs without the sync option, go to the 3D Viewport and select all by pressing **A**.

Using Live Unwrap

Live Unwrap is a cool tool that lets you pin vertices in the UVs and move them to unwrap the rest of the mesh in real time as it adapts to your actions. Live Unwrap allows you to adjust the UVs very quickly without having to unwrap again and again or to move them vertex by vertex.

From the UV Editor's UV menu, activate the Live Unwrap option. To pin the vertices that you want to place in fixed positions, press **P**. Keep in mind that you must pin at least two vertices. After pinning those vertices, which can be the reference points or the corners of the mesh, move some of them (pinned vertices are marked in red) to see how all the UVs adjust to fit your moves. In this mode, move only the pinned vertices. If you move any other vertex, when you move a pinned one, the unpinned vertices are reset; only pinned vertices are fixed during the live unwrap.

When you're done, you can unpin the pinned vertices by selecting them and pressing **Alt+P**. Make sure that you deactivate Live Unwrap before further adjusting the UVs.

Two Live Unwraps

Blender has two Live Unwrap options: one in the 3D Viewport and one in the UV Editor. They have different, albeit similar, purposes and work well when enabled simultaneously:

- The first Live Unwrap is in the Unwrap tool within the UV Mapping menu (**U**) in the 3D Viewport. When active, it updates the unwrap in the UVs automatically when you mark/unmark edges as seams. If this option is disabled, you'd have to unwrap manually after changing what edges are seams.
- The second one lives in the UV menu within the UV Editor, and it enables real-time unwrapping that adapts to the actions you perform in the UV Editor, such as moving vertices around. It's an interesting option to use to make adjustments and to see how what you do affects the automatic unwrapping.

When you use Live Unwrap in the UV Editor, you should keep a few things in mind:

- All the unwrapping or previous manual changes to the UVs will be replaced by the automatic unwrap when you use Live Unwrap.
- Live Unwrap doesn't work when you're just moving vertices; you need to pin them. Essentially, you pin a few vertices, and when you move one or more of those pinned vertices, all the ones that aren't pinned will automatically adapt between them.

If there are parts of the UVs that you've already worked on and are happy with, you can pin them; pinned vertices aren't affected by Live Unwrap.

Adjusting UVs

You can adjust vertices on the UVs to make sure that they look right in the 3D model, of course, and you see real-time feedback in the UV test grid displayed on the 3D model's material as you make those adjustments.

Remember that you also have the Proportional Editing and UV Sculpt tools at your disposal to move groups of vertices subtly, and you can move, rotate, and scale them. Try to adjust the UVs while looking at the test grid on the face so that you have smaller squares in the face than on the back and top of the head. This practice helps you optimize the texture size and create more detail in the face, which is where you need it, instead of on the back of the head, where you don't.

You also have alignment tools for the UVs. Try the options on the UV menu on the header or the contextual menu. Maybe you'll find something interesting. Also, when you use a tool, you may need to make an adjustment to its effect before you confirm it. In such a case, the header shows information with instructions you need to follow to use that tool as well as its current parameters, so keep an eye on that header!

You can use the Snap tools, which are useful when, for example, you want to align one vertex on top of another. Snap tools work the same way as in the 3D Viewport.

Separating and Connecting UVs

Blender offers unusual ways to separate and connect parts of the UVs, but when you get used to them, they're quite useful.

Separating UVs

The quickest way to separate UVs is to use the Select Split tool. Select the faces you want to separate, and press **Y**; the faces are now separated, so you can move them independently. You can also access this option from the Select menu on the header.

On the UVs, Blender shows only the faces you've selected in 3D View. But here's an interesting fact to keep in mind: When you select only a face (or a group of faces) in the 3D Viewport and move it on the UVs, it moves independently and becomes separated from the rest of the UVs. You can manipulate only the visible UVs, and when you do, they're separated from the UVs that are currently hidden.

Another way to separate UVs is to unwrap again the faces you want to separate. Select them from the 3D model and unwrap them to separate them.

The last way to separate UVs is to use the Hide and Unhide features of Blender (**H**, **Shift+H**, and **Alt+H**), which are also available in the UV Editor. Select the faces you want to separate, press **Shift+H** to hide the unselected faces, and transform the selection. Press **Alt+H** to reveal all the UVs, and you see that those faces are separated.

Connecting UVs

Sometimes, it's easier to unwrap a complex mesh in small chunks and then join those as needed to create the least distortion possible.

Blender has a rule: Only vertices that are welded in the 3D mesh can be welded on the UVs. This means that you can snap or even weld (with the Welding tool from the contextual menu) two vertices, and they'll be in exactly the same place, but they won't be merged if they're not merged in the 3D mesh. Thus they will appear in the same spot in the UVs, but they won't really be merged and will still be independent.

That said, to merge vertices on the UVs that are also welded in the 3D model, you just have to put them in the same place by welding or snapping them.

One way to find out exactly which vertices are adjacent is to use the UV Sync Selection mode between the UVs and the 3D mesh. With this method, if you select a vertex in the mesh or on the UVs, Blender shows the shared vertices that should go with it in other UV islands as well. Another option is to temporarily activate the Shared Vertex selection (one of the options within the Sticky Selection Mode button on the header) to see which vertices are in the same place in the 3D model.

Stitch is a tool that can be very helpful for connecting UVs. You can find it on the UVs menu on the header; its keyboard shortcut is **V**. Select some vertices on a border of a UV island, and press **V**; a preview shows you where those vertices should go to be joined to other vertices on the UVs. If you like what you see, left-click or press **Enter** to apply; otherwise, right-click or press **Esc** to cancel the stitching.

Reviewing the Finished Face's UVs

In Figure 8.9, you see how the UVs for the head look after some adjustments. The UVs are nicely aligned, the top of the head and the mouth's interior now take less space in the texture, and the face portion has been enlarged so that it has more detail. The ears were separated to have some more space outside the head's UVs, but this separation is optional, as the ears in this particular model don't need that much detail in the texture.

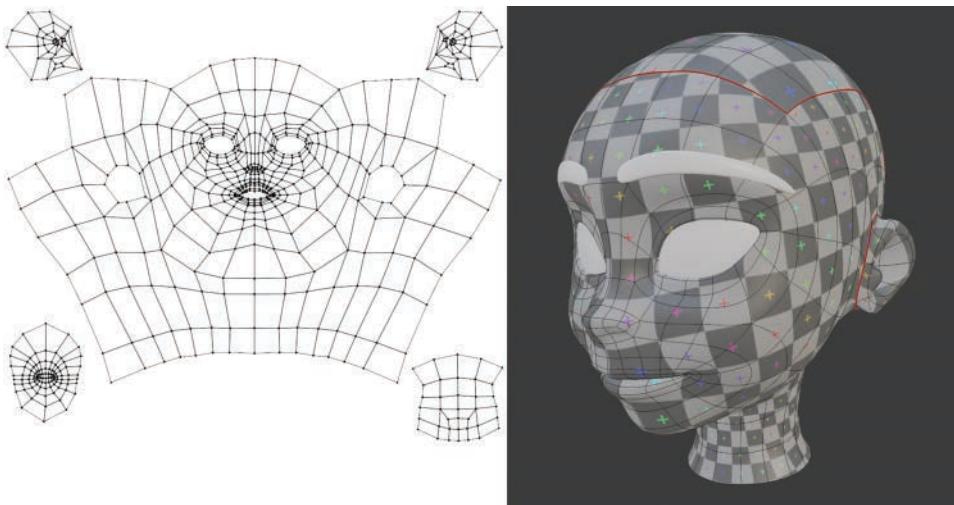


Figure 8.9 The UVs for Jim's face are finished!

Unwrapping the Rest of the Character

Unwrapping the rest of the character is pretty straightforward. First, I'll briefly explain the most important parts of the process so that you understand what to expect. Figure 8.10 gives you an idea of what you'll be doing in this section and shows the objects to be unwrapped: the glove, the boots, the pants, the jacket, the cap, and the neck detail.

Note

When you're unwrapping each of the objects, you can assign the UV test material you created previously so that the UV test grid shows on its surface (remember to set the 3D Viewport in Material Preview mode to be able to see it), allowing you to see whether the unwrapping you're doing is correct.

Unwrapping most of these objects should be quick and easy. The pants, for example, need only a simple seam along the inner leg, just like a pair of real trousers, and the unwrap should work nicely. Keep in mind that the Mirror modifier does its part as well.

For the hand, a seam is marked all along the palm, passing through the bottom part of the fingers. The resulting UVs are adjusted with the Live Unwrap tool, especially the island that represents the top of the hand; it has the sides of the fingers in it, so its shape is somewhat spherical.

The elements of the cap are basically unwrapped just as they were. No seams have to be used; just select all the cap's faces and unwrap them.

The piece at the neck should be simple too. It just needs a seam in the loop that goes all around the interior bottom section so that Blender can unfold it properly.

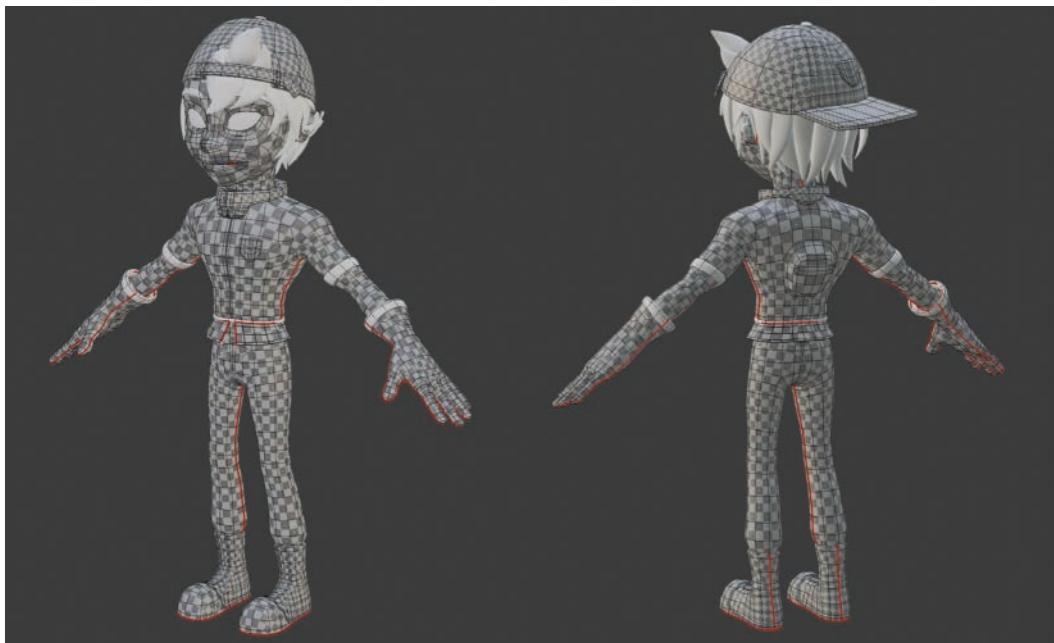


Figure 8.10 The meshes that needed to be unwrapped, with their seams marked in red

The jacket can be a little trickier. Unwrapping it in three pieces—the body, the arms, and the flaps—would be easiest. Later, after adjusting the UVs a little by using the Live Unwrap and Proportional Editing tools, you can join the arm to the side of the jacket, first using the Stitch tool and then snapping and moving some vertices around. The purpose of this junction is to have no seam in the shoulder, as the jacket will have the shoulder pads painted in the texture for that area, and a seam in the middle would potentially cause some issues.

You must also unwrap the badges. For those that need asymmetrical textures, you must apply the Mirror modifier, select all their faces, and then unwrap.

If you want to see what the UV maps look like, see the next section, “Packing UVs.” Right now, each UV island takes the entire space, and showing the islands together would be rather abstract. That’s why packing is important!

Packing UVs

After the objects are unwrapped, you must *pack* them—put all the UVs together in a single Workspace so that they don't overlap. That way, instead of having a different texture for each object, you can have the entire character textured with a single image so that each part of it occupies a part of the UV map.

Figure 8.11 shows what the finished UVs look like.

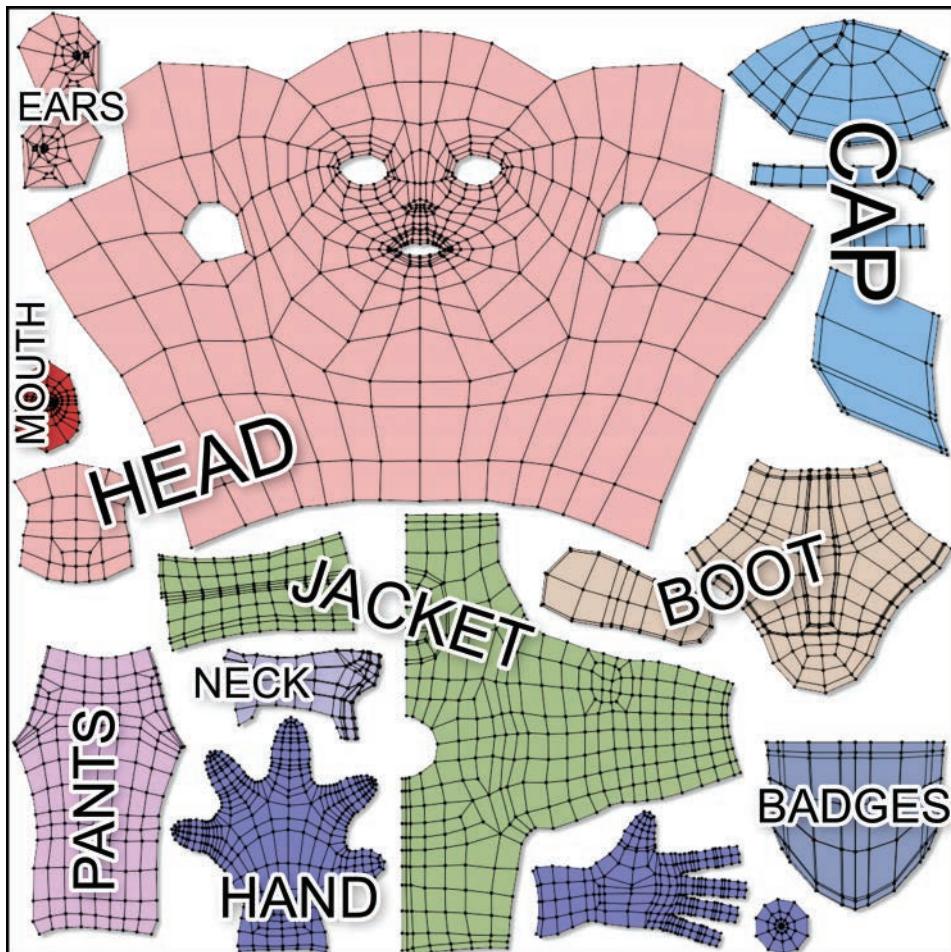


Figure 8.11 The packed UVs for Jim, shown with different colors for each object so you can see how the different objects are distributed

With all these objects in one place, a single texture is used for the entire character. As you can see in Figure 8.11, the face takes most of the texture space. There are also some spaces between objects. You can spend more time filling the entire texture space to have a more efficient texture, but you should always leave a little space between UV islands so that when you paint, they can bleed a little. Otherwise, you may see areas adjacent to the seams that weren't painted.

Packing UVs is very easy. First, you must select all the objects that you want to take part in the same UV space. Then, using Multi-Object editing, jump into Edit Mode (press **Tab**), and you'll have access to all the UVs of all the selected objects simultaneously.

Tip

Blender provides tools for packing UVs: Average Islands Scale and Pack Islands, both on the UV menu in the UV Editor's header. The Average Islands Scale tool scales the selected islands so their sizes are relative to the sizes of the faces in the 3D model. The Pack Islands tool scales and places the selected islands automatically so that they take the largest space possible inside the UV's space.

Here, a good way to start would be to scale the UVs by using the Average Islands Scale tool, using the Pack Islands tool, and then adjusting the sizes of the islands that require adjustment (making those that require more resolution bigger and those that doesn't smaller) and reorganizing them a bit. You may also rotate the UVs so they're oriented in a way that allows for easy painting of textures later. (If part of the UVs is sideways, for example, painting sideways if you need to use a 2D Image Editing tool for the textures wouldn't be comfortable.)

Summary

You have Jim unwrapped with proper UVs, and he's ready to be textured! As you may have noticed, unwrapping can be tricky, and you need patience for it. Unwrapping is a mandatory step in building quality characters, however, because you have to properly define how the textures will be projected onto the model in the most efficient way. Proper UVs are especially important if you're working on videogames, in which everything needs to be optimized (including textures) to work smoothly in real time. Some software packages and tools support UVs that are almost completely automated, which is cool for some specific situations. Usually, however, you want to have manual control of how UVs are unfolded so that it'll be easier to paint your character's textures later.

Exercises

1. Unwrap a cube, and add a texture to it. This exercise helps you understand how UVs work.
2. Add a photo to any model (even Jim's face), and try to unwrap it in such a way that the seams are hidden as much as possible and the texture isn't distorted.
3. Unwrap an object, and pack its UVs to use all the space you can from the texture.

Painting Textures

Textures are simply images that give color (or define other parameters, such as the amount of reflection or shininess an object has) to your models when projected onto their surfaces. In Chapter 8, “Unwrapping and UVs in Blender,” you unwrapped the parts of Jim that need textures. Now it’s time to use those UVs to paint a texture and give your character a layer of paint to move your project closer to the final result. Although Blender provides some texturing tools, textures are typically created with third-party 2D image-editing software such as Adobe Photoshop, Affinity Photo, GIMP, and Krita, or dedicated programs for 3D texturing such as Substance, Quixel, and Mari. These programs and the artistic skills needed to paint complex textures are outside the scope of this book, but this chapter briefly explains the general workflow with both 2D image editors and 3D texturing programs, as well as the basics of how you can paint textures in Blender itself.

Before we get started, I’d like to point out that textures and materials work together, so it may be difficult for you to understand one without the other. I recommend that you check out Chapter 10, “Materials and Shaders,” for a complete view of how they interact.

Defining the Main Workflow

There are two main workflows for using textures on 3D models:

- **Texturing before unwrapping:** Sometimes, depending on your needs, it’s easiest to create a texture and then adapt the UVs to fit that texture. In this case, of course, you need to create the texture before you start unwrapping. A good example of this method is a wooden floor. You may have a photo of wood, and to apply it to the surface, you load that photo and adapt the UVs so that the wood in the photo has the right size and position on your 3D model. This method works best in simple models; in complex models, the projected image needs to be adapted to the model.
- **Texturing after unwrapping:** This method is the usual texturing method for characters or complex objects because it specifically tailors textures to that model. First, you unwrap the model; then, if you’re using 2D image editors to create your textures, you export the UVs as an image to use for a reference when painting and adjusting your textures to make sure that they fit the faces in your

model. If you’re using Blender for texturing, you can just start painting on the 3D model. In the case of using another 3D texturing software, you can export your 3D model after unwrapping and paint textures on that model. As Jim is a complex model, we’ll be using this method of texturing after unwrapping.

The unwrapping part of workflows mentioned in the preceding list is discussed in Chapter 8, but it’s worth noting so that it will be fresh in your mind.

I’ll talk about painting textures within Blender first. Then I’ll explore other options (2D image editors and 3D texturing software) and talk about the pros and cons of each.

Texture Painting in Blender

Yes, you can paint textures right inside Blender, directly on the 3D model! In this section, you explore the Texture Paint Workspace and learn about the options that you have access to when using Texture Paint Mode on an object.

Texture Paint Workspace

The quickest way to start painting textures in Blender is to jump into the Texture Paint Workspace (in the list of predefined Workspaces at the top of the interface). You can see the Texture Paint Workspace in Figure 9.1.

In the Texture Paint Workspace, you can find all you need to start painting:

- **Image Editor**, where you can see the UVs of the selected object and paint the texture in 2D.
- **3D Viewport**, where you can see the 3D model and paint on it. What you paint also immediately shows up in the Image Editor (correctly projected onto the UVs), as long as you select the same image to be displayed in both editors (a topic that I’ll get to later).
- **Outliner**, where you can see the different elements of the scene if you need to select them to paint on other objects.
- **Properties Editor**, which is conveniently set up to show the current tool properties. In both the 3D Viewport and the Image Editor, you have access to the toolbar, the default tool being the brush. In the tool properties, you can find everything you need to configure your brush, such as color, tip shape, strength, the effect of pressure (if you use a pen tablet) on the stroke, the stroke type, and so on. You can also find the tool options in the Sidebar of the 3D Viewport and the Editors header.

When you’re in the Texture Paint Workspace, you’re ready to start painting, but first you need to understand a few things, such as how Texture Paint interaction mode works and what conditions the selected object must have for you to paint on it.

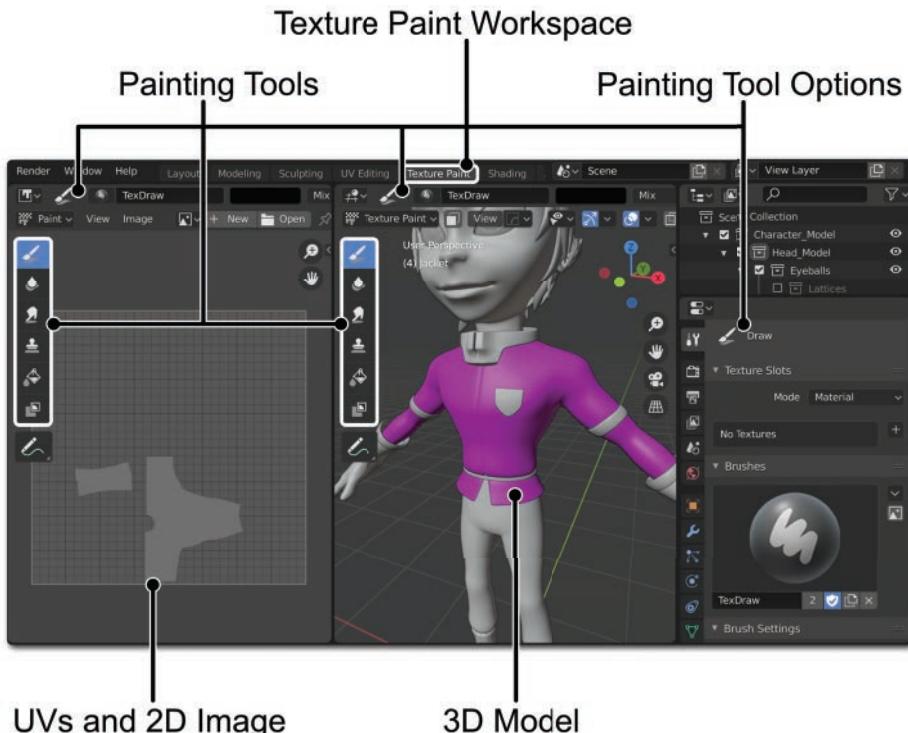


Figure 9.1 The Texture Paint Workspace shows the most important things you need to start painting: UVs, the 3D model, and the texture painting tools. The jacket appears to be magenta because it's the selected object, but it's not ready to be painted on just yet.

Texture Paint Interaction Mode

One of the object interaction modes (such as Object and Edit) is Texture Paint interaction mode, which you can access from the interaction mode drop-down list in the left corner of the 3D Viewport's header or in the pie menu that appears when you press **Ctrl+Tab** in the 3D Viewport. When you switch to Texture Paint Mode, the interface of the 3D Viewport changes to accommodate the painting tools and options (see Figure 9.2).

Even though this mode is meant to paint textures, the object needs a small setup process first, so I'll describe the tools here so that you understand what's going on. Later, you'll set up the object.

Here are the main things that change when you switch to Texture Paint interaction mode:

- The toolbar in the 3D Viewport shows tools and automatically select the brush as the current Active Tool.
- The 3D Viewport's header shows the options for configuring the brush (the Active Tool).

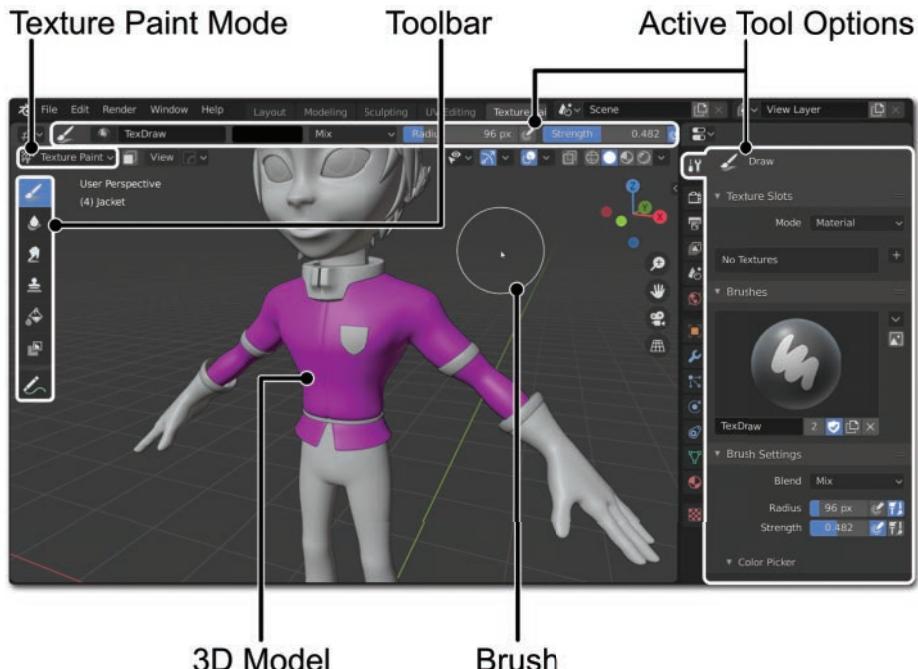


Figure 9.2 Texture Paint Mode and its options in the Sidebar.

- If you open the Tool tab in the Sidebar (press **N** in the 3D Viewport to show or hide this region), you see all the available options for configuring the brush (the Active Tool). You can also see these options in the Properties Editor if you click the Active Tool tab (the one with the wrench-and-screwdriver icon).

Tip

A benefit of using the Properties Editor to show the brush options is that you can place it anywhere on the screen—useful if you (like many other people) prefer to see these options on the left side of the screen. An extra tip is hiding the Properties Editor’s header and tabs. To hide the header, right-click it and choose the option to hide the header from the contextual menu. To hide the tabs, you can place your mouse cursor where they meet the menu; your mouse cursor will turn into the two-sided arrow icon, allowing you to drag the tabs toward the border and hide them. To display both the tabs and header again, click the little arrow button that appears at the top of the Properties Editor.

You can choose among a variety of tools (brushes) on the left side of the 3D Viewport, such as Fill, Clone, Smear, and Soften (similar to a blur). You can adjust the options and behavior of each tool in the aforementioned menus. You can adjust, for

example, radius and strength (opacity), texture, and profile curve; you can even use the stroke stabilization feature to stabilize the stroke while you move the brush. You can also create your own tool presets for fast access, as well as pick a color for your brush.

For the radius and strength of the brush, it's worth noting that you can quickly change those two values directly in the 3D Viewport. Press **F**, move the mouse to define the radius (you see a preview of the resulting brush radius as you move your mouse), and press **LMB** to change the radius. Press **Shift+F**, move the mouse, and press **LMB** to change the strength of your brush.

As you saw in the previous section, the Texture Paint Workspace also shows the image texture, and you can paint on it in the Image Editor. While you're in the Texture Paint Workspace, the Image Editor is set up for painting, but it's important to know how to set it up for painting manually. When you open the Image Editor, you see a drop-down menu in the left corner of its header. This menu is set to View by default, but you can click it and select Paint; when you do, you'll be able to access the same painting tools that you do in the 3D Viewport within the Image Editor. The tools will even be synced if you're also in Texture Paint Mode in the 3D Viewport, so adjusting the tool or other settings (such as selecting the texture you're painting on, discussed later in this chapter) in one of them affects the other. It's very convenient to be able to see and work on the 3D model and the 2D flat texture simultaneously.

Painting One Object at a Time

It's important to know that it's possible to paint on only one object at a time. If more than one object is selected, you can paint on only the active one (the last one you selected). To paint on a different object, switch back to Object Mode, select the desired object, and go back into Texture Paint Mode.

Before You Start Painting

Before you begin painting, you must make sure that

- The object to be painted on is unwrapped. Otherwise, you won't be able to paint on it.
- The object has at least one assigned texture in its material, or there is an image in the .blend file that you can use to paint on.

In the latest versions of Blender, the workflow to start painting has been heavily improved. If you don't fulfill these two conditions, Blender gives you options to create UVs and images or materials with just a few clicks. I go through this process in the next section.

If you've followed along with the previous chapters, you've unwrapped Jim's parts that need textures, and you have a material with an UV grid texture in the base color.

After unwrapping, you may want to remove the UV test material from those objects and add the definitive material with the final texture assigned to it. Alternatively, you could rename the UV test material (if you don't plan to use it anymore), load the UV test grid in the Image Editor, go to the image properties (press **N** to show the Sidebar), and switch the image type from UV Grid to Blank.

Tip

When you change the image from UV test grid to a blank image, you can define its color in the same menu. Usually, you'll want to set the color to white when you want to draw line details. Alternatively, you can use the Fill tool to fill the entire mesh with a single color by clicking the surface. The Fill tool paints only those parts that cover the image in the UVs, though, and leaves the rest in the original color. Using the Fill tool in the UV/Image Editor instead of the 3D model allows you to fill the areas between the existing UVs.

Generally, if you switch to Texture Paint interaction mode after selecting an object that has a material with an image texture assigned to its base color, you should be able to start painting. Just press **LMB** and drag on top of the 3D model to start painting!

Image Textures Resolutions

Textures should be square instead of rectangular, and the number of pixels should be a power-of-two number (8×8 , 16×16 , 32×32 , 64×64 , 128×128 , 256×256 , and so on). The reason is that these resolutions use the resources on your computer a lot better than rectangular images with random sizes. These numbers are called power-of-two numbers because to jump from one resolution to the previous or the next one, you must divide or multiply by 2. Usually, you should work in a pretty high resolution, because you can always scale it down if needed, but you'll lose detail and your model won't look right if you have to scale up. The size you finally use depends on the amount of detail you need for your model. If the model is high-resolution and will be seen from a great distance in certain shots, having textures with very high resolutions is unnecessary and would needlessly consume important resources. It's good practice to use high-resolution textures only while they're close to the camera and the details are visible. After you create a texture, it's best to make different sizes of it that you can use, depending on the object's distance from the camera.

In the next section, you'll see what to do if your model doesn't fulfill the main requirements to paint on it, along with how to set it up if it doesn't work initially.

Conditions for Painting

As mentioned earlier, Blender has improved the texturing workflow in recent versions. These improvements include options in Texture Paint Mode that can be used when the selected object isn't ready to be painted on (see Figure 9.3). To see these problems,

you'll have to pay attention to the Texture Slots panel, on the Tool tab of the Sidebar. (You can also find this panel in the 3D Viewport's header and on the Active Tool tab of the Properties Editor.) You'll learn how to use texture slots later in this chapter.

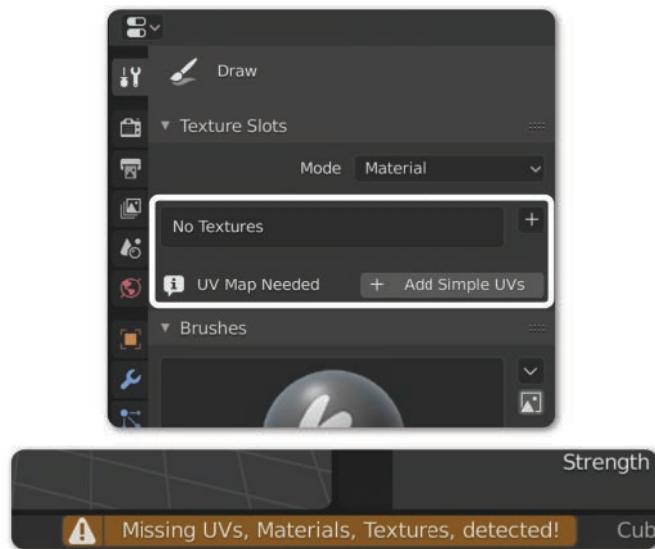


Figure 9.3 If the object you're trying to paint on doesn't meet the conditions for texture painting, Blender displays messages to let you know and options for fixing the problem.

These are the ways in which Blender helps you meet the required conditions for painting textures on your model:

- If you try to paint on an object that doesn't meet the requirements, you'll see an error in the Status Bar (at the bottom of the interface) to let you know what's wrong.
- Blender detects whether the object has UVs. If not, Blender displays the message 'UV Map Needed' in the Texture Slots panel and shows a button that automatically generates simple UVs for the selected object. These UVs won't be perfect, but most times, they're good enough for texturing simple objects.
- Blender also detects whether the object has an image or material to paint on. If not, the message 'No Textures' appears in the Texture Slots panel. You'll also see an error message in the Status Bar if you try to paint on an object that has UVs but not images assigned to paint on. If this happens, you can create a new image directly from this menu; you will learn how to do this in the next section.

Texture Slots

You'll often have different image textures in your object's materials (see Chapter 10, "Materials and Shaders"), each of which uses a slot. You can access all the image textures in the file or material from the Texture Slots panel mentioned in the previous section. This panel has options that let you specify which texture you want to paint on (see Figure 9.4). You can use two types of slots:

- **Material:** When you select Material on the Slots tab, you can paint on the available textures inside the material that you assigned to the active object. All the textures are listed inside the Available Paint Slots section of the panel, which allows you to switch among them quickly (see Figure 9.4).
- **Single Image:** If you select Single Image, you gain access to an image selector from which you can select any image that you created in the file or create a new one. This image is projected onto the active object, allowing you to paint on it. You can also select the UV map on which you want the image to be projected and then save all the images. (This option isn't available when you're working with Material Slots, as images inside materials already have their UVs defined.) Keep in mind that painting with this option can be very convenient, and you can switch the texture or create a new one at any time, but if you want it to be part of the object's material, you'll have to assign it manually later.

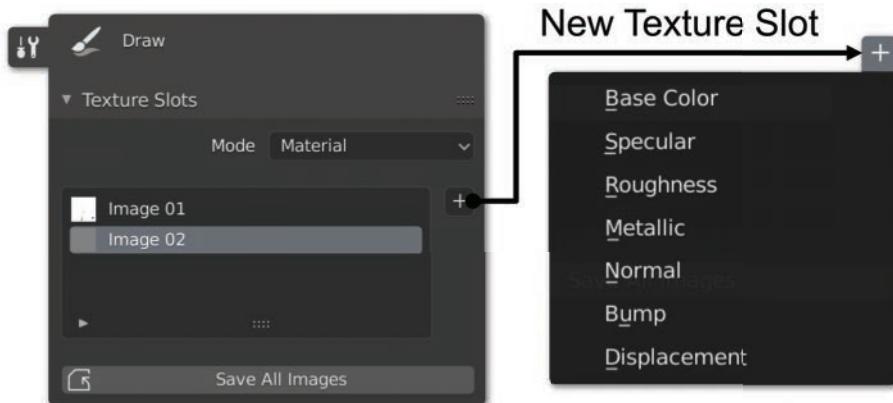


Figure 9.4 Texture Slots panel and the options to create new texture slots in a material

Below all these options, you'll find a button that allows you to save all the modified textures. If you close Blender without saving textures, a warning message asks whether you want to save them. If you don't want to save some changes in given textures, you can save them individually from the Image menu on the Image Editor's header. (An asterisk appears next to the Image menu if the current image has unsaved changes.)

Keep the following things in mind when you're working with texture slots:

- Single Image can be seen only while the 3D Viewport uses Solid shading mode. In Material Preview or Rendered mode, the object's material overrides the selected image.
- When you're using the Material Slots option, if the viewport shading is Solid, you'll see only the selected slot. If you're in Material Preview or Rendered viewport shading, you'll see the combined effects of all the textures, even though you'll still be painting only the selected slot.
- If you choose the Material Slots option and click the + button on the right side of the slots, you can create a new slot. If the selected object doesn't have a material assigned, creating a new slot generates a new material. When you generate a new slot, a menu allows you to choose which channel of the material you want the new image to affect; this menu is linked and set up with the corresponding material options. After choosing a channel (see Chapter 10, "Materials and Shaders"), a menu allows you to create a new image that you'll be painting on.

Limitations of Blender's Texture Paint Mode

Although Blender's Texture Paint Mode is powerful and has a lot of options (which you should explore further, as I discuss only the basics in this chapter), it does have limitations. The system has no layers, for example, and layers are useful for texture painting. Add-ons are available that provide Blender some layering capabilities, but they're not nearly as good as the layers you can work with in other specialized software.

Blender isn't a substitute for proper 2D image-editing or 3D texturing software, of course, but it does provide tools that allow you to do basic texture painting. Some people have taken Blender's painting options to the extreme and made impressive artworks with them!

You can use Texture Paint Mode to texture an entire character, but the decision depends on the character, of course, and on whether you feel comfortable texturing within Blender or need more powerful tools. Instead, you might try working with other software to use layers, make color corrections, add effects and filters, or apply masks—things you can't do in Blender's Texture Paint Mode.

That said, Texture Paint Mode proves to be really useful for a lot of things. It works well for creating a base for textures that you can complete in other software, for example. It may be difficult to see where a detail should be placed when editing a texture in 2D image-editing software, for example, so you have to find where it's supposed to go in the 3D model. But because you can paint in 3D in Blender, you have a good opportunity to start your texture by painting the details on the surface of the 3D model and using the resulting image as a reference for the final texture, which you'll make in different software.

Creating the Base Texture

It's time to paint the details over Jim's model. You won't do anything fancy for the base texture—just a quick black-and-white drawing that you can use later as a base in your 2D image editor of choice.

Placing Texture Elements

Using the character's reference images to see where to place the basic texture elements, start painting over the 3D character. This time, the reference images are not the ones in the 3D Viewport (by the way, at this point you can turn them off from the Outliner or delete them), but you can divide your interface to have an Image Editor display them if you prefer that to have them open in a different window. If you happen to have a second monitor, you can use it to see the references at all times without having to use space in your main Workspace. When you have all the basic texture elements in place (see Figure 9.5), save the image so that you can keep working on it in external software.



Figure 9.5 Jim's jacket with reference elements painted on to define their positions

Tip

Drawing smooth lines can be tricky. For some elements, you may prefer to have steadier lines. To draw steady lines, go to the Stroke options (3D Viewport header or Tool options), and activate the Stabilize Stroke option. Now when you paint, the brush follows your strokes at some distance and smooths them, providing very clean and steady lines, which are useful for drawing things like the seams in Jim's clothes.

Saving Your Image

Blender tells you when you have made changes to an image that you haven't yet saved. The Image menu on the Image Editor header shows an asterisk (*) that indicates unsaved changes. Open the Image menu, and you see the save options. Also, pressing **Alt+S** inside the UV/Image Editor saves the image. Pressing **Shift+Alt+S** performs a Save As operation for the image.

If you used the Texture Slots option to save the images up to this point, the textures are saved within the .blend file, but they don't really exist in your computer's folders. See the next section for more information.

Saving images through the Image menu allows you to save them as external files on your computer.

Packing Your Images

In this case, packing has nothing to do with the packing in UV maps. Blender has another packing feature that allows you to include external files (such as images) in the .blend file itself. This feature is very useful when you work on multiple computers or are collaborating with other people. Suppose that you loaded some textures from your hard drive into your model, and you want to send that model to a friend. If you send the model as it is, your friend won't see the textures because he doesn't have them on his computer, where Blender will look for them. In this case, you can pack the images. The images will be incorporated into the .blend file, and your friend will be amazed by your textured model!

To pack an image into the .blend file, go to the Image menu, and choose the Pack option. (This option won't appear if the image is already packed.) If you want to pack every single external file into the .blend file, go to the File menu, select External Data, and click Pack All Into .blend. In the same menu, you can find the Automatically Pack into .blend option, which, when enabled, automatically packs into the .blend file every image that you load from that moment on.

Keep in mind that saving all these files in the .blend file increases the file's size, which continues to multiply if you like to save your files in different versions as you progress.

Understanding the Elements of a Texture

Before I jump into the world of texturing, it's important to give you a basic understanding of how materials work and how textures affect them. You'll learn more about materials in Chapter 10, but textures and materials work together, so it's difficult to understand one without the other.

Introduction to PBR Materials

First of all, materials define how a surface looks and reacts to light and color when rendered. They need different properties, such as what color they are, how reflective they are, and whether they're metallic.

In the past few years, PBR (physically based rendering) materials have become commonplace and essentially standard in the 3D industry, so most 3D software and game engines need to support them.

PBR materials became very popular thanks to how consistent they are, and in recent years, advances in hardware and technology have made it possible to use them even in real-time environments.

Why are these materials consistent? As the name PBR implies, they follow rules based on how materials act in real life, which makes it easier to generate realistic materials that work well in any environment. Before PBR, materials had properties that could mimic real-life materials but were more rudimentary, making it more difficult to achieve realistic results.

Each property of a material can be controlled by a different texture, as some areas of the material can have different colors, be more or less reflective, and so on.

Understanding Material Channels

The fact that a material is made up of different properties that define how it looks, which we can call channels, means that we need multiple image textures for a single material. This requirement is one of the limiting factors of Blender: It's not possible to paint multiple textures at the same time.

2D image editors can paint in only one layer at a time, although it's possible to use filters and tricks to generate channels later.

3D texturing software such as Substance Painter, on the other hand, have been created in recent years with PBR materials in mind, so when we paint, the brush is simultaneously affecting multiple images, one for each channel. The brush can be set up to have different effects on each channel, and painting affects all the properties of the material on which you decide.

To sum up, you need to create textures, and you have many ways to create them:

- The channels can be independent, or you may not need some of them for a given material, so you can paint them as different images in Blender or other software.
- You can paint one of the images, such as the base color, and then use filters and other tricks in 2D image-editing software to generate the other images for the rest of the channels.
- You can use 3D texturing software and paint all the channels at the same time.

Many times, you can even mix some of these methods to achieve the results that you want.

Texturing in Other Software

As mentioned at the beginning of this chapter, you have multiple ways to generate textures for your objects. So far, I've explored the basics of painting textures within Blender. Now I'll discuss the pros and cons of different methods, and then share the main workflows of each.

Pros and Cons of Texturing in Blender and Other Software

Each method of painting textures has upsides and downsides:

- **Texturing with Blender:** Painting textures in Blender has a clear advantage, in that you don't have to go outside Blender at any time, so you avoid potential import/export issues. This approach is brilliant for basic texturing work because it's quick and efficient. It also makes it possible for you to adjust the textures at any point in time while working on the model. The downside is that the texturing tools in Blender aren't very powerful.
- **Texturing with 2D image-editing software:** 2D is in the title. You can't work with 3D models in this software (Photoshop, Affinity Photo, Gimp, Krita, Corel Painter, and so on), so you have the disadvantage of having to work in two dimensions without seeing the result of the work. On the other hand, when it comes to creating images, nothing is as powerful as these programs, which are designed to paint and edit photos. Chances are that you'll find many tools that can do anything you need, such as color correcting, applying all types of filters, adding and formatting texts, and even distorting images in these types of software—features you won't find (or that won't be very advanced) in 3D software.
- **Texturing with 3D texturing software:** This type of software has the most modern set of tools, designed specifically to create textures for 3D models. This software is very powerful for that purpose but requires all work on the model to be done somewhere else, so if you need to change something, you have to export and then reimport the files. It's very efficient to generate textures with these programs, even though the image-editing tools are not as powerful as the ones in 2D image editors. Software in this category includes Substance Painter, Mari, and Armor Paint.

Other factors in selecting a method are deciding which method is most effective for the style of texturing you want to achieve and what you feel most comfortable working with.

Texturing in 2D Image-Editing Software

You can take the base texture elements you created in Blender and keep working on them in your 2D image-editing software. I used Photoshop for the examples in this section, but you can use any software you want.

Exporting the UVs as an Image

It's important to see the UVs while you work on the textures; you need to make the textures fit those UVs so that they project correctly onto the model later. Here are the steps you need to follow to export Jim's UVs to an image file:

1. Select the object.
2. In Edit Mode, select everything (**A**).

Caution

When you want to export the UVs of multiple objects, as in Jim's case, make sure to select all the objects before exporting so that all the UVs appear in the exported image.

3. Open the UV Editor.
4. On the UV Editor's header, click the UV menu, and select Export UV Layout.
5. In the interface for saving the image, on the right side of the window, you need to adjust a few options. Select the Modified option to display the meshes with their modifiers applied, such as Subdivision Surface. (This setting is important because it allows you to see how the final mesh, on which the textures are going to be projected, will appear.) The All UVs option ensures that every UV is shown in the image, so enable it as well. For the resolution, 2048 × 2048 could work, but increase it to 4096 × 4096 so that you have enough room for details and so that it fits the final texture's size. As for the format, select .png. (You can also export vector images in .svg or .eps format.)
6. Select where you want to save the image, type its name, and click Export UV Layout. If you open the generated image, you should see something similar to Figure 9.6.

In the next section, I describe a typical method you can use to create your texture in 2D image-editing software.

Loading the UVs and Base Elements

First, load the reference images: the base elements you painted inside Blender. Then load the UV Layout image you exported from Blender. Place these images on the top layers, using Multiply blend mode (which makes only the dark areas visible and leaves the rest of the image transparent), so that you can turn the images on and off over your texture to check that everything is in place.

Adding Base Colors

The next steps are refining and cleaning the lines you painted in Blender. Then you can start filling the areas with colors. You can pick these colors from the color-scheme images you created during the character's design process.

Tip

At this stage, you can load the texture on the objects' material in Blender to check from time to time that everything looks right. If you're working with Photoshop, Blender also supports .psd files, so you can load them with no conversion. If you load the image in the Image Editor, you can update it rapidly by pressing **Alt+R**. Also, if you save a new version of the texture, you can choose Replace Image from the Image menu in the UV/Image Editor. This option replaces the old version of an image (everywhere you were using it in Blender) with the new version of that image. Sometimes, after painting the basic details, you see that the texture looks a bit off in some areas of the 3D model, so you can go to the UVs and adjust them slightly to get the desired result, or go back to the 2D image-editing software and paint those areas again.

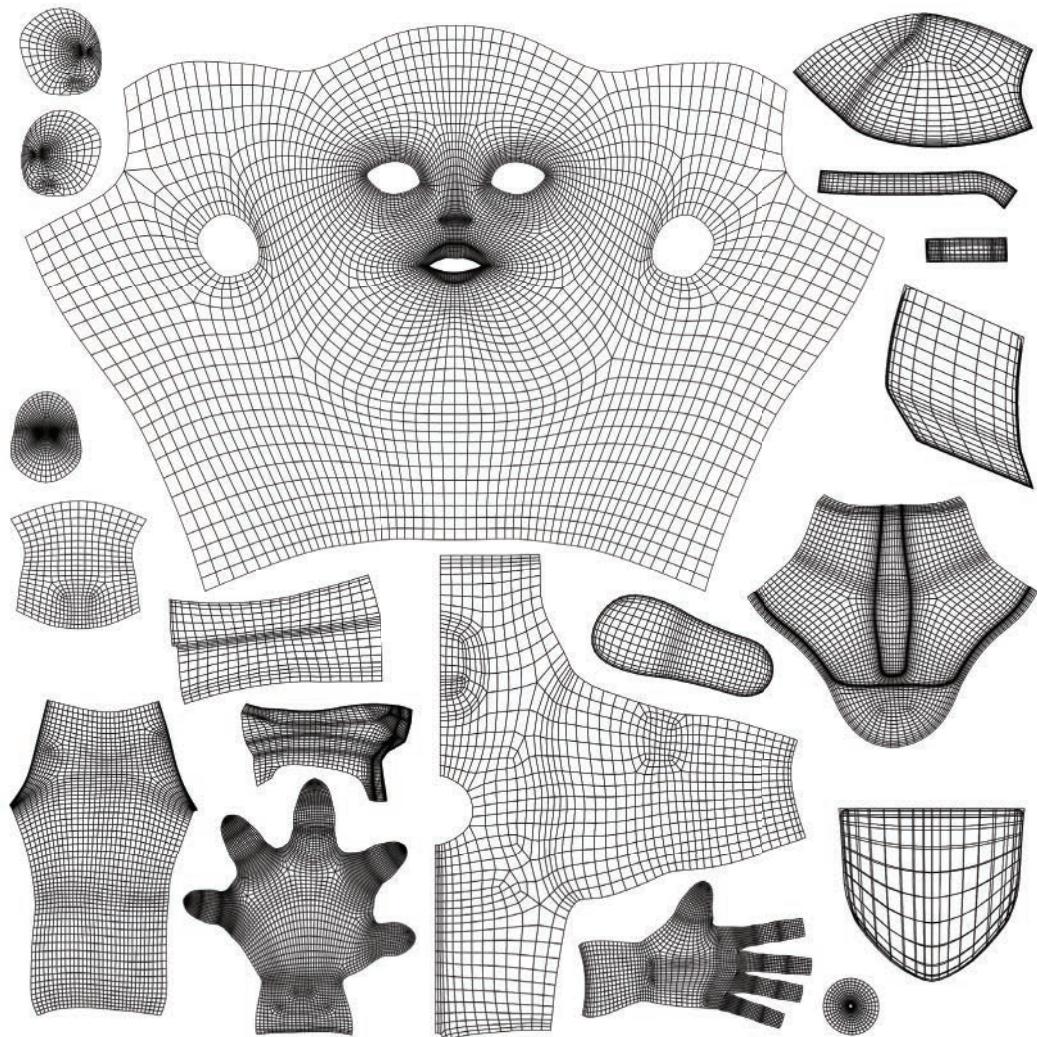


Figure 9.6 You can export the UVs in an image format and then adapt your images and painting to those UVs. This way, the images are projected correctly onto the surface of your 3D model.

Adding Details

When you're done with the texture's base colors, you can start painting the details. This example is pretty simple, with flat colors, but you can add as much detail as you want (depending on what you want to achieve in terms of the character's style).

You could add details such as thick gray lines in the clothing seams, a symbol on the badge, a darker lip color, and a little blush on the cheeks. You can also add soft shadows, wrinkles in the cloth, and other small details.

Applying the Final Touches

Finally, add a little more life to the texture by overlaying a photo of some noisy texture (such as leather or dirt) or perhaps painting with a brush that uses a texture. You can also define certain parts a little more. This stage is where you finish the texture. You can see the resulting texture in Figure 9.7.



Figure 9.7 Example of a texture created with this method, taken from an edition of this book published before 3D texturing software was commonplace

Generating Other Texture Channels

As mentioned earlier in this chapter, materials have different channels, and they need textures in each channel to control how they affect the result. If you’re using 2D image-editing software, you’d need to adjust the created image to generate those other channels.

Although this process is possible (and was common for many years), it’s cumbersome, and a lot of trial and error is involved in achieving the proper results.

You’ll see examples of other textures for different channels in the next section.

3D Texturing Software

This modern texturing process has become a standard, and the textures generated here are the ones that Jim will finally use. You can use other programs, but in this case, I use Substance Painter, one of the texturing programs from the Substance Suite by Adobe.

Exporting/Importing the 3D Model

The first step is exporting the parts of the model that need textures from Blender to Substance Painter. Generally, the most common format for such things nowadays is .fbx, supported by nearly every kind of 3D software.

To export the file, select the objects that need textures in Blender, choose File > Export option, and select .fbx format in the resulting list. The Export window pops up. Name the file, choose where it will be saved, and select the Selected Only option to ensure that only the objects that need textures are sent to Substance Painter.

When you create a new project in Substance Painter, you simply need to load the .fbx file you just exported.

Something else to remember is that there are two systems for calculating normal maps (textures that can give the illusion of bumps on the surface): OpenGL and DirectX. Blender uses OpenGL, so you should choose that option to avoid issues when you export the textures to Blender later.

Texturing Process

I won’t go into details here, as Substance Painter is outside the scope of this book, but I’d like to briefly explain how the process works so you understand it.

First, after loading the model, you generally bake maps. What does this mean? Substance Painter generates several images based on the model, such as proximity of objects (Ambient Occlusion), the directions in which different parts of the model look (World Normals), the angles and corners (Curvature), and the objects that make up the model (ID Colors). These images generally aren’t seen in the final textures, but they can be used to create effects, masks, and so on.

Jim is made up of many materials, but you don’t need a different material for each part of the object. The appearance of each part of the object is defined by the textures you paint, so you can paint Jim’s face to have different properties from those of his jacket or pants, such as different colors, roughness (how intense the reflections are), or metalness (whether a surface is metallic).

Each part of the character typically starts with a Fill layer that allows you to set the base parameters for that area; then that layer can be mixed with other layers to add details and effects.

In 2D image-editing software, a layers stack allows you to have different elements on different layers so that you can edit them separately and control how they mix with the layers below. In Substance Painter, you have several parallel layers, each for a different channel. Bit by bit, you keep adding layers, effects, and details until you're happy with the result.

You can also add automatic effects or use particle brushes, perhaps to spray dirt on the character and see how the falling particles of dirt paint on the surface of the material.

While you paint, you can preview the 3D model under different lighting situations by selecting an environment, and you can see the textures on the UVs in case some operation might be easier to do in two dimensions.

When you're done, export the created textures in an image format that can be loaded into the Blender materials.

Painting textures is a really fun experience, and I highly recommend trying these techniques. You can see the resulting images in Figure 9.8. You'll learn more about what these textures do to the material in Chapter 10, "Materials and Shaders."

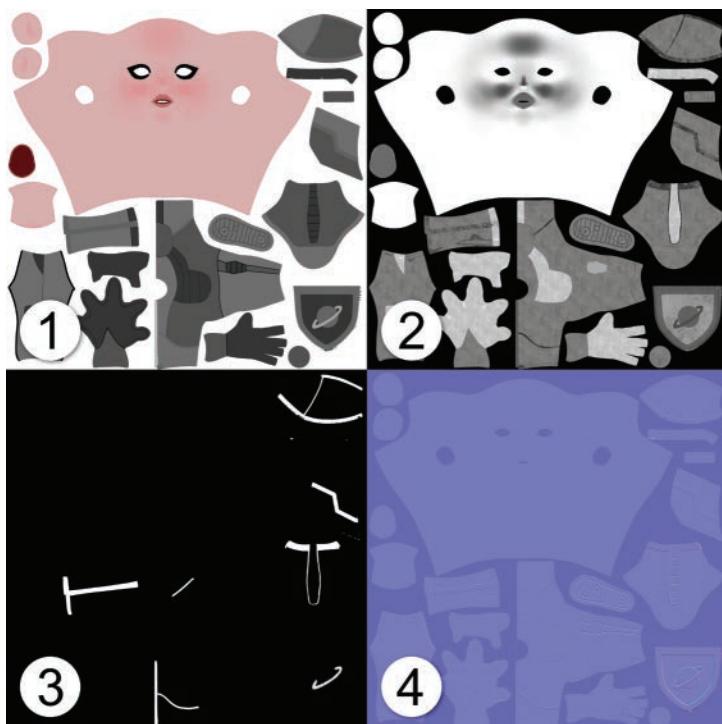


Figure 9.8 Textures generated with Substance Painter: base color (1), roughness (2), metalness (3), and normals (4)

Seeing the Painted Character in Blender

If you haven't done so already, it's time to load the texture into Blender to see how it looks on various parts of the character. Don't hesitate to go back and forth between your 2D editing software and Blender to refine the texture as needed and to test it on your model. Figure 9.9 shows Jim's model with the texture projected onto it. To do that, load the base color texture in the base-color parameter of the objects' material. In Chapter 10, you'll learn how to load the textures for other channels to complete the look of the materials.

Some parts still have the mesh color without texture or anything else on them. Those parts will get their color from the Blender material that you apply to them in Chapter 10, so they don't need textures.



Figure 9.9 Base color texture applied to Jim's material in the objects that will use textures instead of flat-colored materials

Summary

Texturing can be a lot of fun; it's a creative process, and you can take it as far as you want. Applying textures is the point in the process where you have the opportunity to make a character model look the way you want it. If you want realism, I recommend

that you use photos to create your textures (skin, wood, grass, sand, and virtually everything else) instead of painting them manually. In Jim’s case, the style isn’t photorealistic, so using basic textures to define the colors and adding some details is fine.

I mentioned it at the start of the chapter, but I’ll say it again: Textures and materials work together, so it may be difficult for you to understand one without the other. I recommend that you also check out Chapter 10 for a complete view of how they interact.

If you’re interested in creating really cool textures, you may want to learn how to use programs to edit images and paint textures and then combine them with Blender. These programs are only tools, of course, so it would also be helpful to learn the artistic part of painting (such as color theory) so that you can make the most of your textures and achieve great results. These programs are really powerful, allowing you to create awesome textures that bring your characters to life!

Exercises

1. Use the method of unwrapping after texturing by downloading the texture from this book’s companion website and making the UVs fit that texture.
2. Compose photos of skin or cloth in your image editor to make the texture look more realistic.

10

Materials and Shaders

You now have textures that define the color and properties of the surfaces, but something's still missing materials! *Materials* are sets of components and values that determine how the light is diffused, absorbed, reflected, or transmitted by a surface on your 3D model. A material can be reflective, may be transparent, and can even emit light. You have to set up these properties for materials so they look how you want them to look. Materials are made up by *shaders*, because a shader (internally, through programming) tells the software how to draw an object on your screen. (There are even shading languages to program how these shaders work.) Hence, the process of adding materials is called *shading*. First, you learn about the main workflow and how to use materials. The rest of the chapter explains how to create materials and load the textures you created for Jim in Chapter 9, “Painting Textures,” and you’ll see how they work in both EEVEE and Cycles.

Understanding Materials

Before you get into using materials, you should get an idea of how they work and understand the differences between EEVEE and Cycles, for even though they are mostly compatible, there are some limitations to keep in mind.

Applying Materials

The process for adding materials is as follows:

1. Select an object.
2. On the Materials tab of the Properties Editor, select a material or create a new one to be applied to the active object.
3. Tweak the material to achieve the result you want. It’s useful to add some basic lighting and take some test renders to see how the materials are behaving. Using the Material Preview or Rendered viewport shading mode in the 3D Viewport is a great help to see how materials behave.

The basics of creating and applying materials and controlling the 3D Viewport’s shading mode are explained in more detail in Chapter 3, “Your First Scene in Blender,” in case you need a refresher. You’ll learn about materials in depth in this chapter.

Tip

If you want to apply the material to more than one object at the same time, select all the objects, and apply a material normally. The material will be added to the active object (the one selected last). Then press **Ctrl+L**, and link the materials. The same material you added to the active object will be applied to the entire selection.

How Materials Work

In the real world, materials on the surfaces of objects have different properties that make the light that hits them react in specific ways. Glass, for example, lets light go through it; metal shines by reflecting light; and wood absorbs light (doesn't reflect it). Depending on the roughness of the surface, the light is reflected, and the reflection is more blurry on rougher surfaces. Something that is hot enough (such as tungsten or incandescent metal) can even emit light.

In 3D, you can control some parameters that make virtual light act in ways similar to how light behaves in the real world. You can define reflectivity, shininess, surface color, transparency, refraction coefficient, and more. With different configurations of those values, you can imitate real-world materials inside the 3D world.

PBR Materials

PBR (physically based rendering) materials were introduced in Chapter 9, “Painting Textures.” In this chapter, you’ll understand them a bit better.

PBR materials adhere to a set of rules that make them appear realistic. Here are few examples:

- Metallic materials are highly reflective, whereas nonmetallic ones have very low reflectivity.
- Metallic materials have a dark base color, and most of their color comes from their reflections, as they’re able to tint those reflections. Nonmetallic materials, on the other hand, always have white reflections (or the color of the light they receive), and more of their base color is visible.
- All materials have a Fresnel effect, through which parts of the material at a grazing angle are more reflective than the parts looking directly toward us.
- Materials have roughness, which is a microscopic property that doesn’t necessarily mean the surface has to be bumpy. Roughness makes the rays of light that bounce on the surface go in different directions, making the reflections blurry (see Figure 10.1).

In Figure 10.2, you can see examples of different PBR materials.

Blender has support for this type of materials, although it’s versatile enough to allow you to break this rule if you want to achieve other results for nonrealistic rendering.

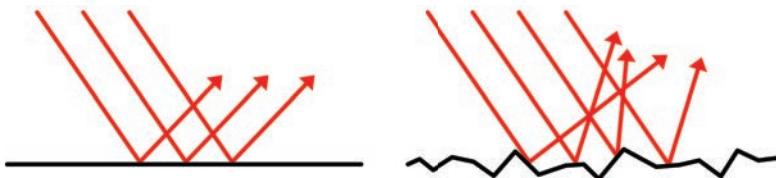


Figure 10.1 A material's roughness determines how blurry reflections are by imitating microscopic bumpiness on the surface, making rays of light that come from the same direction bounce in random directions. In the image, you can see how rays reflect on a smooth surface (left), providing clear reflections, and how they reflect on a rough surface (right), making reflections less clear. This property doesn't really affect the surface or geometry—only the material.

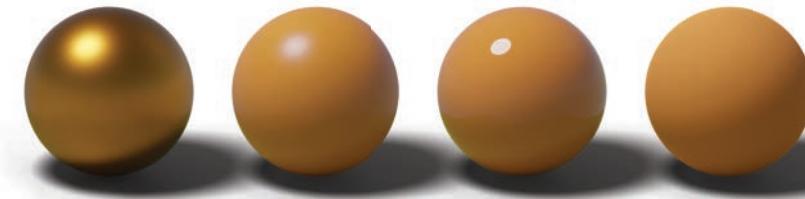


Figure 10.2 Different settings configurations for PBR materials, using the Principled BSDF shader: metallic material (left), nonmetallic material (center left), smooth material (center right), and rough material (right).

Typically, a good way to start with PBR materials is to use the Principled BSDF shader, which provides many of the options you need to create PBR materials and has the rules of PBR materials baked in so that they can't be broken. This shader is selected by default when you create a new material.

PBR Workflows

Two PBR workflows exist, which use different properties and slightly different rules: Metallic-Roughness and Specular-Glossiness. These workflows are very similar, and each of them has its own pros and cons. Blender is versatile enough to be able to work with both, although by default, it uses Metallic-Roughness, which is the workflow that the Principled BSDF shader is built on.

The main differences between these workflows are

- Metallic-Roughness has a metallic property that defines which parts of the material are metallic and which aren't, and the shader decides how metallic and non-metallic parts should behave, making it more difficult to “break” the material and making it look less realistic. Specular-Glossiness doesn't make this distinction, which means that on one hand, you have more control of how colors are reflected, but on the other hand, it makes it easier to get unrealistic results.

- Roughness and Glossiness are essentially the same things, just inverted. The rougher a surface is, the less glossy, and vice versa.

Essentially, you can get different levels of control and ease of use with each workflow. I encourage you to learn more about PBR and the different workflows. For the purpose of this book, we'll work with the Metallic-Roughness workflow.

Figure 10.3 shows the options of the Principled BSDF shader. I explore these options in the following sections.

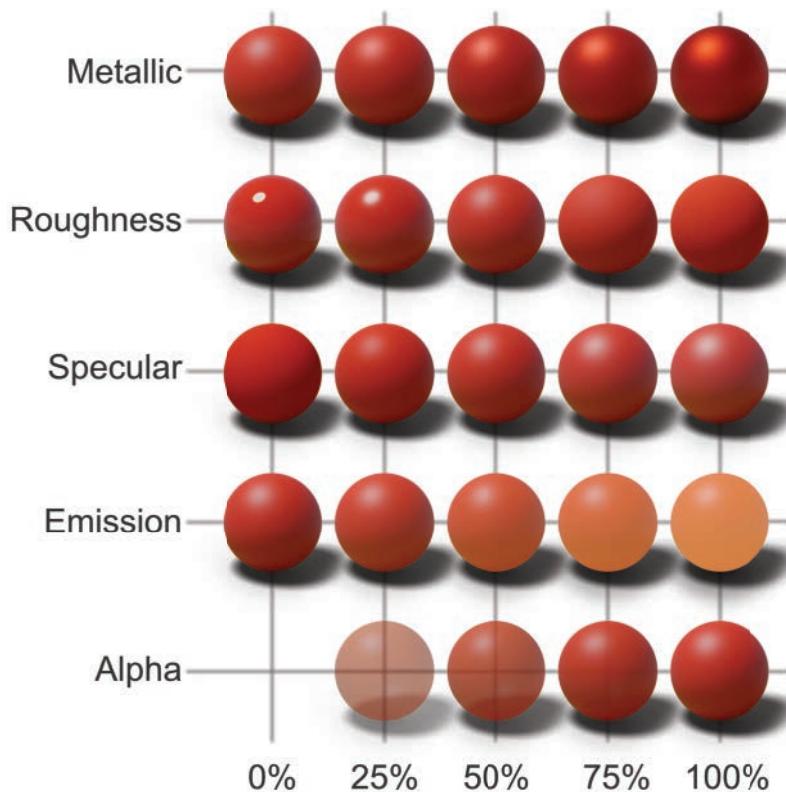


Figure 10.3 Different properties of a material and how the values of such properties affect the result

Shaders and Mix Shaders

Shaders are the elements that make materials. There are shaders for things like a material's color, glossiness, transparency, and refraction. Combining shaders lets you achieve materials that look any way you can imagine.

You can select the shader that you want for a material in the Properties Editor (see Figure 10.4).

Shaders can be combined through the use of two special shaders:

- **Mix:** Lets you connect two shaders and then define what percentage of each will be visible. If more of one material is visible, the other material becomes less visible.
- **Add:** Adds the results of the two shaders connected to it. This shader is useful for effects such as translucency, where the light should be added. While the Mix shader obscures one material to show more of the other one, the Add shader adds both materials at the same time, generally producing brighter results.

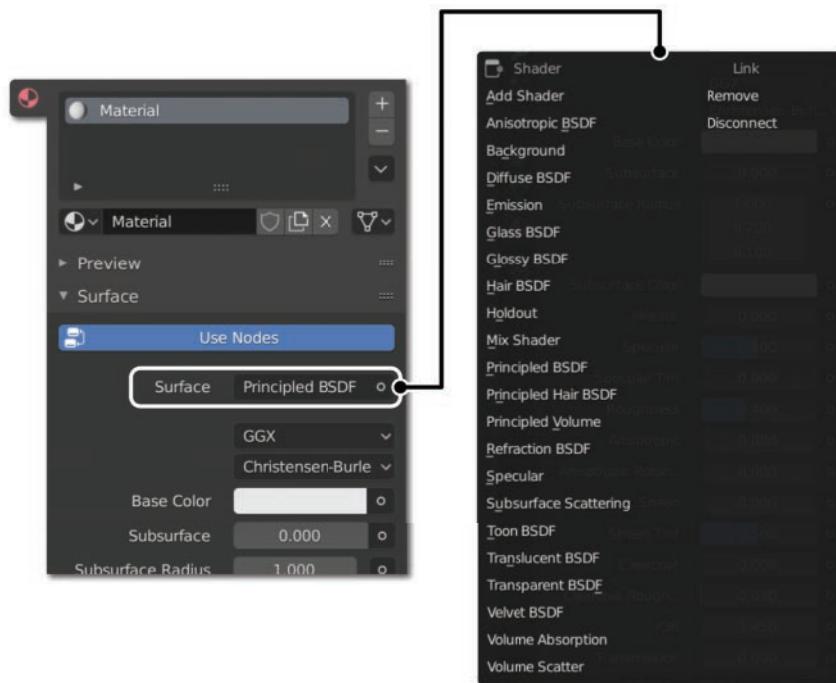


Figure 10.4 The shader selector in the material's properties. When clicked, this selector displays a list of all the available shaders. By default, when you create a material, a Principled BSDF shader is selected.

The Mix and Add shaders can be nested to mix multiple shaders and generate more complex materials.

The Principled BSDF is very convenient because it's like a premade set of shaders that allows you to control the most common properties of nearly any material, so you don't have to create and mix all the shaders one by one. On the other hand, creating the materials shader by shader gives you more control, letting you achieve any effect that the Principled BSDF may not allow.

Masks and Layers

Materials can be very simple . . . or very complex! You can have different properties of materials working together in different areas. Using masks (black-and-white images in which white means full effect and black means no effect), you can tell Blender where you want each parameter of the material to act. You can set specific parts of the material to be reflective and shiny, for example, by using masks. This technique is useful for, say, rusty metal; the rusty parts would have no reflectivity, whereas other parts of the metal would be shiny and reflective. You can control those properties by using textures to act as masks.

Materials can be made of several layers. You can stack textures and shaders or parameters to get complex layered materials. Suppose that you have your car paint material on your car model and you want to add some nice stickers to the model. You can use layers to put those sticker images on top of the base car paint material.

You create layers by using Mix shaders and then using a mask connected to the Mix shader's factor so that the mask controls which parts of shaders are shown after the Mix shader is applied.

Imagine rusty metal. You have a shader (or group of shaders) for the metal and another for the rust. Then you connect both of them to a Mix shader, and in its factor, you plug in a grayscale image in which the parts of the model that you want to show rust are white and the parts that you want to be metal are black (or vice versa). Figure 10.5 shows an example of how a mask affects the visibility of the first and second shaders inside a Mix shader.

Caution

If you plan to use a mask as displayed in Figure 10.5, through the Properties Editor's Material tab, the mask needs to be an image that has an alpha channel, and the mask needs to be stored in the alpha channel of the image, as that's what Blender will use automatically for the Mix shader's factor.

If you need to use the color of the image, not its alpha, you have to open a Shader Editor and change the node connections between the image texture and the Mix shader.

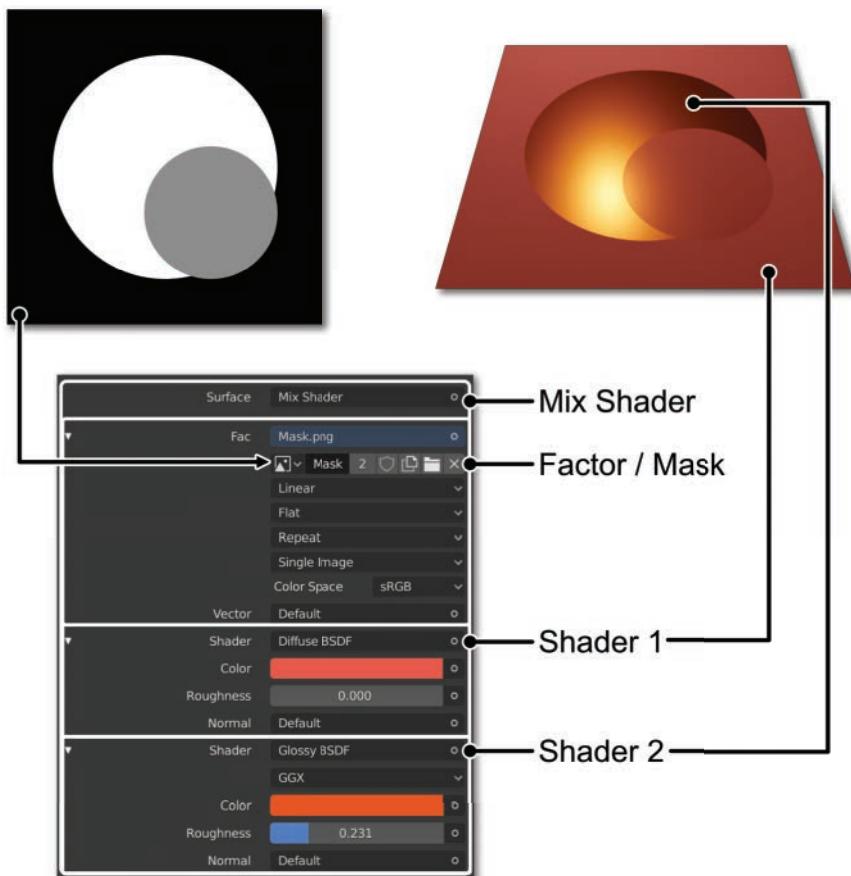


Figure 10.5 A mask image (top left), the material properties using a Mix shader with that mask (bottom), and the resulting material (top right). You can see how the mask brightness affects the visibility of each of the two shaders within the Mix shader. Essentially, the white parts act as a factor 1 (showing the second shader only), and the black parts act as a factor 0 (showing the first shader only). Pay attention to how the gray part shows a mix of both shaders. A neutral gray would be equivalent to a factor 0.5—a 50 percent effect from each of the shaders.

Channels

Materials can have different channels, or properties (usually, textures are loaded to affect each of those channels and control specific properties of a material), that affect different characteristics of the surface. Some PBR properties, such as roughness, are mentioned in previous sections; here, you'll learn a bit more.

In any property, you can have a value or connect a texture. If you connect a texture, the image will control how much or how little of that property will have an effect on a

given part of the surface. If you don't use textures and just input a value or a color, the entire surface will have the same amount of that property.

Some people find it hard to understand what these properties are or do, so here are brief descriptions of some of the most common ones. I'll use the names they go by in Blender, and in parentheses, I'll list typical names for those properties in other software so that you'll recognize them if you encounter them):

- **Base Color (also known as Diffuse Color):** This property defines the surface's base color.
- **Metallic (also known as Metalness):** This property defines whether the object is metallic. When you're using a black-and-white texture connected to it, the white parts make the surface metallic and the black parts nonmetallic. When the material is defined as metallic, reflectivity is higher, and the reflections are tinted with the color selected as the base color.
- **Specular:** This property controls the amount of reflectivity of a surface. In the Principled BSDF shader, it accentuates or decreases the reflectivity provided by the Metallic value.
- **Roughness (also known as Glossiness):** This black-and-white texture tells the software how rough the surface is, diffusing and blurring shines and reflections in areas in which the surface is rougher. Roughness and glossiness are opposites, and different software uses one property or the other. To convert roughness to glossiness, or vice versa, invert the map.
- **IOR:** *IOR* stands for *index of refraction*. If your material has refraction (the light can go through it as in a glass), this index determines how much the image you see through it is distorted.
- **Transmission (also known as Refraction):** This property specifies whether the material is refractive. If it is, light can pass through it and get distorted, similar to what would happen in a glass, and the amount of distortion is be defined by the IOR value.
- **Emission:** Usually, this channel is a black/transparent image with some colored parts. From those colored parts, the software takes the color as the light emission color and the alpha value as the emission intensity. (Usually, it's possible to define only the color and use a different texture or numerical values for the intensity.)
- **Alpha (also known as Transparency or Opacity):** This channel sets the parts of the surface that should be transparent or opaque. Usually, you use a grayscale image or an image in a format that supports alpha (RGBA). Depending on the render engine, black is transparent, and white is opaque, or vice versa. This property is different from Transmission, in that Alpha lets light pass through without any sort of distortion or refraction.
- **Normal (also called Bump, Normal Map, or Normal Bump):** This property is used to make bumps and reliefs in the surface by using textures instead of geometry. It requires some setup, but it can be used to create a bump

texture—a black-and-white image that adds some bumpiness to the surface. The software uses this feature to change how light reflects on the surface and makes the surface look more detailed. Bump is useful for small details that are not big enough to be modeled, such as scars and scratches. The Normal property can also be used to display normal maps, which are like advanced bumps—RGB (red, green, blue) textures in which each color tells the software the direction in which light should be reflected from the surface. This channel is widely used in video-games nowadays to make objects appear much more detailed than they really are. Normal maps (also called normal bumps) can be hand-painted with specific software or image-editing tools or created from two versions of a model: one with a high resolution and one with a low resolution. Normal maps are generated by baking the details of the high-resolution model into a texture for the low-resolution model.

These properties are not the only ones that you will find in shaders or in the Principled BSDF, but they are the most-used ones. I encourage you to try changing the values in each property and see how they affect the resulting materials.

When you’re thinking about adding textures to these properties, you can use this key for most of them: Black and white define values (0 to 1), and colors define the colors if the property allows for them. In a black-and-white texture, black is no effect whatsoever, and white is 100 percent effect; gray tones are all the values between no effect and full effect.

Procedural Textures

Procedural textures are widely used in computer graphics. Even procedural modeling exists. But what does *procedural* mean? It means anything that a computer program can generate mathematically through algorithms, using a set of parameters that the user can input to control the result to some extent.

Suppose that you want to build a city. You can create a few buildings, and then you want to populate your city with them. You can do the job manually, duplicating those buildings one by one and placing them where you want them to be. For a big city with thousands of buildings, of course, this method wouldn’t be very productive, which is why software offers you ways to do the job procedurally. Using different tools and giving you some level of control, the software can populate the city for you randomly.

A *procedural texture* is a texture that the software generates automatically to fit any surface. These textures are like patterns that repeat randomly, and you have some level of control of their features.

Blender provides procedural-textures options. For the texture type, instead of selecting Image Texture, you can select any other type to create procedural textures. (I discuss this option later in this chapter.) Noise is one of the types, for example; it generates a noisy pattern, which is useful for giving a surface variations in color. When you select a type, the Image panel is replaced by specific options for the type of texture you selected.

Blender has a lot of procedural texture types (Clouds, Blend, Noise, Wood, Checker, and so on), each with its own properties and uses. Make sure that you check them all out!

Differences and Compatibility Between EEVEE and Cycles

There are key differences between EEVEE and Cycles, the render engines that are included in Blender. You can review the main options for both in Chapter 3, “Your First Scene in Blender.”

EEVEE is meant for real-time rendering, using similar technologies to those found in videogame engines. It’s very fast, but the speed comes from limiting many things, including shadows and reflections.

Cycles is much slower and requires a powerful computer, but it provides more accurate and realistic results thanks to using more calculations through a method called *path tracing*, which measures the bounces of light, like photos in real life.

EEVEE doesn’t use path tracing, so although you can achieve impressive results, things like bounced light will have to be faked.

Essentially, you should use Cycles if you’re aiming for realism and have time to calculate the render, and you should use EEVEE if you don’t need much realism and want a faster result.

Here’s another analogy: EEVEE gives you results similar to videogames, and Cycles gives you results similar to movies.

I want to stress that even though EEVEE is not as realistic as Cycles, that doesn’t mean you can’t get realistic results with it for certain scenes (by taking advantage of some advanced techniques). You certainly can, and EEVEE has been even used for compositing with real-life video. Just keep in mind that there are limitations.

Despite their differences, EEVEE and Cycles are highly compatible, which means that even though they have different options that control how the render works (accessible from the Render tab of the Properties Editor), materials can be used interchangeably for the most part. There are some limitations, however, given that some specific properties of the materials may use calculations that are available in only one engine. Refractions work differently in EEVEE and Cycles, for example, and some effects require path-tracing to work, so they will work only with Cycles.

For basic materials, however, there should be little differences between rendering them in EEVEE and in Cycles, so you have a lot of leeway in creating materials, and especially providing a fast and efficient workflow: creating and visualizing materials in EEVEE in real time and then rendering them in Cycles (even though it may be necessary to adjust those materials to Cycles in some cases, as the results can vary slightly because both render engines use completely different rendering techniques).

Switching Between EEVEE and Cycles

For the most part, you can switch between EEVEE and Cycles without affecting the result in big ways unless if you use features that are exclusive to one engine or the other.

You can change the render engine at the top of the Render tab in the Properties Editor.

For Jim, you'll create materials by using the Principled BSDF shader, which is perfectly compatible between EEVEE and Cycles, even though some of its properties may need some additional setup or look slightly different in each engine.

Nodes

Up to this point, you've used the Properties Editor to access the options for the materials. Advanced materials are outside the scope of this book, but you can open a Shader Editor to access the nodes for a material. In this interface, shaders, textures, and other elements are blocks that can be connected in a schematic way to generate the final material.

It's essentially the same process you can perform from the Properties Editor, although it's more visual. You can do things such as connect the same node output to multiple nodes to reuse parts of a shader or textures.

Figure 10.6 shows how a mask and a Mix shader look in the Properties Editor and in the Shader Editor's nodes (same setup as Figure 10.5).

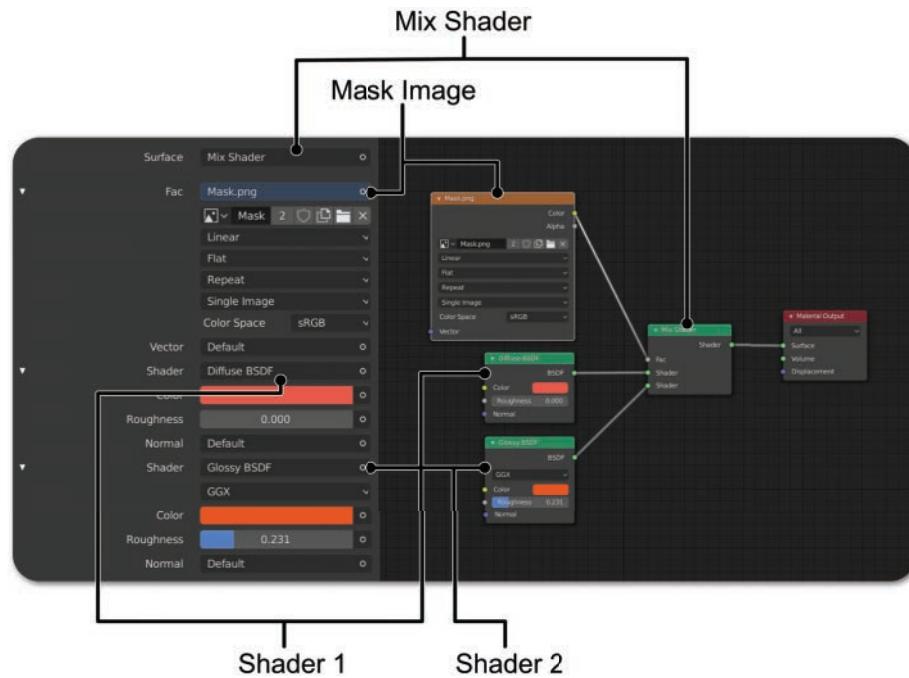


Figure 10.6 The basic material setup in the Properties Editor (left) and in the Shader Editor with nodes (right)

The Properties Editor gives you a quick way to access the main options for creating a material, but the menu can get complicated very quickly. Nodes, on the other hand, provide a more visual way of working with the complex elements that can make up a material.

Shading Your Character

In this section, you'll start creating the materials that will give Jim his final look. You'll learn how to add materials and how to assign textures to their properties using the textures you created in Chapter 9, "Painting Textures."

First, you'll focus on the eyes, learning how to use the Materials interface to add multiple materials to the same object.

Adding Several Materials to a Single Object

Jim's eyes are different from the other objects. Although they're a single object, they have different materials to define the pupil, iris, eyeball, and cornea. For now, all you need to know is how to create materials, as you saw in Chapter 3, "Your First Scene in Blender."

Before you start, you need to know how to use material slots (see Figure 10.7). At the top of the Material tab of the Properties Editor is a field where you can add or remove material slots. Each of those slots can hold a material that affects a given part of the object. When you select a slot, the material you choose in the datablock below it will be assigned to that slot.

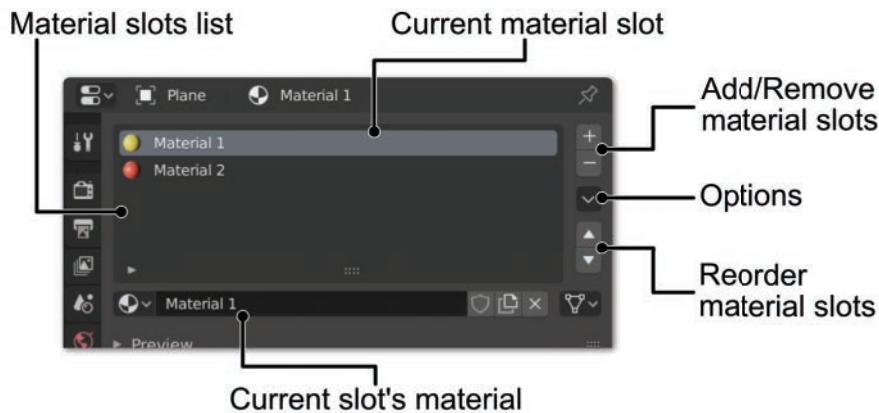


Figure 10.7 The material slots at the top of the Material tab of the Properties Editor

When you select an object and enter Edit Mode, three options appear below the material slots:

- **Assign:** Assigns the selected material slot to the selected faces in the model. Remember that you don't assign materials to those faces, but to the material slot. You can change the material that belongs to that material slot at any time, and those faces will show the newly selected material for that slot.

- **Select:** Selects the faces of the model that have the selected material slot assigned.
- **Deselect:** Deselects the faces of the model that have the current material slot assigned.

Now that you understand how material slots work, you're ready to add the materials to the eyeball. Figure 10.8 illustrates the materials you'll be using.

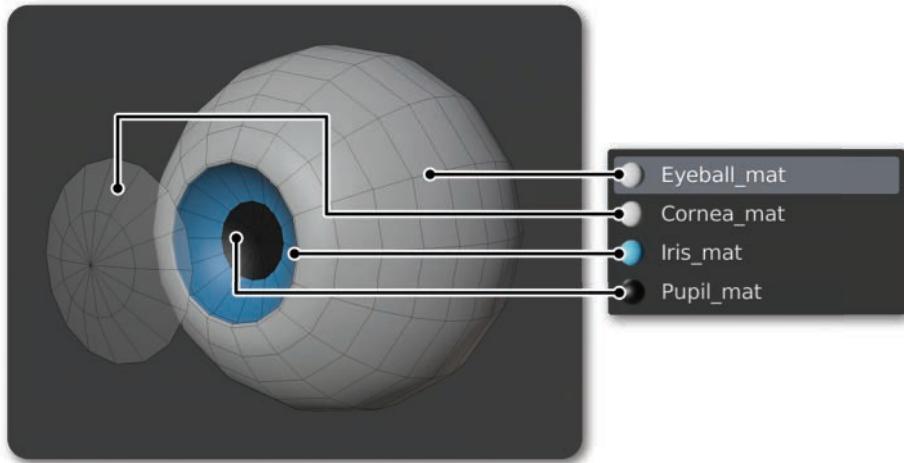


Figure 10.8 Jim's eyeball and the material slots assigned to each of the eyeball's parts

1. Select the eyeball, apply a material to it, and call the material Eyeball_mat.
2. Enter Edit Mode (**Tab**), add a new material slot to the list, and create a new material in the slot. Call that material Cornea_mat. Select the cornea (**L**). Make sure that you have Cornea_mat selected in the Materials list, and with the cornea selected in the model, click the Assign button to assign that material from the list to the selected faces. With the cornea selected, press **H** to hide it so that you can work on the faces inside it.
3. Using Figure 10.8 as a reference, repeat the process in Step 2 to create two new materials called Iris_mat and Pupil_mat, and assign them to the iris and pupil faces. Pupil_mat should be black, and Iris_mat should be the same blue color that was used for the hair. Then press **Alt+H** to unhide the cornea.
4. You have to repeat the entire process with the other eye or delete the second eye and duplicate the one you just worked on by mirroring it, as you did in Chapter 7, “Character Modeling.”

To see the eye materials in Solid viewport shading mode, you can tweak the Viewport Display properties for the materials. But you'll run into an issue: The cornea won't let

you see the pupil and iris, and there isn't any transparency or alpha property for the Viewport Display options. Here's a solution: When you select the color for the cornea in the Viewport Display options, you'll find that in the color selector, you have red, green, and blue colors, and also an alpha channel! If you reduce the value of the alpha channel within the color, the material will use the alpha value as transparency while the 3D Viewport is in Solid viewport shading mode.

For now, the materials are very basic, as you've only created them. In the next sections, you learn more about materials and see how to start adding textures.

Tip

You can quickly add different materials to a single mesh as you did for the eyeballs, using basic materials that are correctly named. After the materials are assigned to the faces of the mesh, you can select them from the list to tweak them and make them look cooler, even in Object Mode. You won't need to select the faces for any reason after the materials have been assigned.

Tip

If you hover the mouse over a property's value (even a color) on the menus and press **Ctrl+C**, you copy it to the clipboard, and by hovering the mouse over another property's value or color selector and pressing **Ctrl+V**, you paste that value or color. This technique is useful for things like copying the diffuse color from one material and pasting it in another one.

Understanding the Material Properties Tab

In the Material Properties tab of the Properties Editor (see Figure 10.9), you can add shaders, mix them, nest them, and so on, so the menu can become really complex, but the material will look exactly how you want it to look. When you create a material, you'll see the properties for the Principled BSDF shader. Here are some of the panels you'll find in this menu:

- **Current selection, Material Slots list, and current material:** The first three parts of the menu show the selected object and material; a list of material slots that allows you to use different materials inside the same object; and a drop-down list that allows you to select an existing material, rename it, or create a new one. (The menu shown in Figure 10.9 is the one that was made for the Iris_mat in the previous section, but using a Diffuse BSDF shader instead of Principled BSDF to keep the menu shorter and make its sections easier to see.)
- **Pin:** If you click the pin button in the top-right corner of the Material Properties, the current material will be fixed in this menu, regardless of your selection, until you unpin it (by clicking the pin button again). This feature is useful if you want to compare two materials, as you can divide the interface and create multiple Material Properties menus.
- **Preview:** The Preview panel shows a preview of the material, and you can choose different objects to preview: a sphere, cube, cloth, and so on.

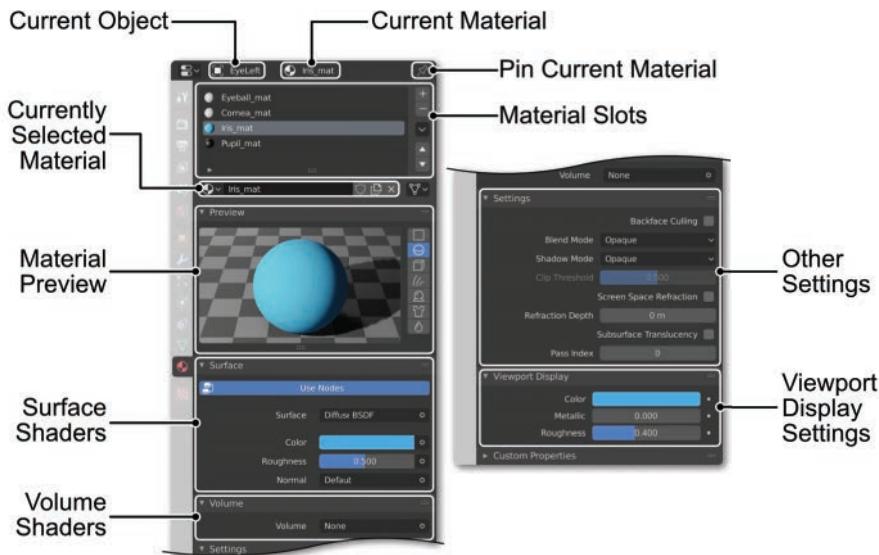


Figure 10.9 Material Properties tab of the Properties Editor and its options.

The figure shows the options for EEVEE; the ones for Cycles are slightly different, and there's an extra section of Displacement options.

- **Surface:** This option is probably the most important on the Material tab, allowing you to select the type of shaders you're going to use and set their properties for the surface of the 3D object. You learn about shaders in the next section, “Using Shaders.”
- **Volume:** This option allows you to use volumetric effects with the objects that use this material. *Volumetrics* is the technology that simulates fog, smoke, and gasses, and is very useful for adding ambience to a scene. Volumetrics let light pass through and create cool effects. Unfortunately, even though they've been improved in recent versions, volumetrics are still complex and slow to render in Cycles (but quite fast in EEVEE). If you want to use these effects, you usually want to add a shader (Volume Scatter or Volume Absorption) in this panel and have nothing on the surface (as Volume represents what's inside the object, and the object would be covered if something was on its surface). Imagine smoke, for example. Smoke has volume but no surface. You can have materials with both a surface and volume, but the volume inside would be visible only if the surface lets light pass through, via transparency or transmission (refraction) effects.
- **Displacement** (Cycles only): This technology allows you to use a grayscale (height map) texture to deform the surface of a mesh so that it has much more detail. Although it's possible to use this feature, Displacement is still under development in Cycles. Alternatively, you can apply a modifier to the mesh to use Displacement (the Displace modifier), but it's really heavy, as you have to subdivide

the mesh a lot to make it work properly. This option is an advanced option, and you won't use it for Jim.

- **Settings:** Here, you find some more options for tweaking the material. In EEVEE, some options are necessary to control how the material's transparency works. The rest of the options are more advanced and beyond the scope of this book.
- **Viewport Display:** In this menu, you can choose how you want the material to look like when you're in the "workbench" render engine (not using EEVEE or Cycles)—the engine that's used while you're in Solid viewport shading mode (not previewing materials).

Using Shaders

Cycles materials are made of shaders. How can you add shaders to a material? When you add a new material, inside the Surface panel, you see a Principled BSDF shader, and below that selector, you'll see the options for this shader. Clicking the shader's name displays a list of shaders. Following are some of the most-used shaders (but I recommend that you try the rest of them, as you may find them useful):

- **Diffuse BSDF:** This shader is the basic shader (just a colored surface).
- **Transparent BSDF:** This shader makes the surface transparent.
- **Glossy BSDF:** This shader is shiny and reflective, and by controlling its roughness, you get more blurred shines and reflections.
- **Hair BSDF:** This shader is specially tailored to be used with hair-strand particles.
- **Refraction:** This shader adds a refraction effect to the surface.
- **Glass:** This shader is a glass effect, including refraction, transparency, reflection, and glossiness.
- **Anisotropic BSDF:** This shader is very useful for simulating metallic pieces. On top of the effects of the Glossy shader, it adds anisotropic shines to the surface.
- **Volume Scatter and Volume Absorption:** These two shaders are used exclusively in the Volume part of a material. They represent a volumetric effect, such as fog, smoke, or some other phenomenon that happens inside transparent objects such as glass.
- **Emission:** Probably one of the coolest shaders, Emission allows you to convert any mesh to light. This shader emits light, and you can control its intensity and color. Although you could light a scene with objects that use this shader type, I recommend using lights when possible, as emissive materials can make your render slower and noisier. Keep in mind that it really emits light only in Cycles; in EEVEE, the Emission shader will be self-lit, but the light won't affect other objects (unless you use tricks like baking indirect lighting).

These shaders are basic and sometimes are all you need; each of them controls only a single property of a material. The Principled BSDF shader, on the other hand, is an advanced shader that is a mix of many basic shaders, giving you a fast way to create most materials.

There are more shaders than listed here, but you'll probably use these most often. Each has a couple of options that you can tweak to control its properties (color, roughness, intensity, and so on). In this book, you'll use only the Principled BSDF shader, but it's interesting to know how the other shaders work.

Mixing and Adding Shaders

Shaders can't do much on their own, so the Shaders list includes two special shaders that let you mix other shaders: Mix Shader and Add Shader. This book uses only the Principled BSDF shader to keep things simple, but it's important to know how the Mix shader works.

If you select a Mix shader in the Surface shader selector, two more shader selectors appear, with a factor value of 0.5 by default.

The factor defines how much of each of the shaders will be seen. A factor of 0.5 is 50 percent of each. The closer you get to 0, the more of the first shader will show, and the closer you get to 1, the more of the second shader will show.

Figure 10.10 shows a material that mixes two basic shaders to achieve a more complicated result. The first shader is a Diffuse BSDF with a color, and the second shader is a Glossy BSDF that adds some reflections.

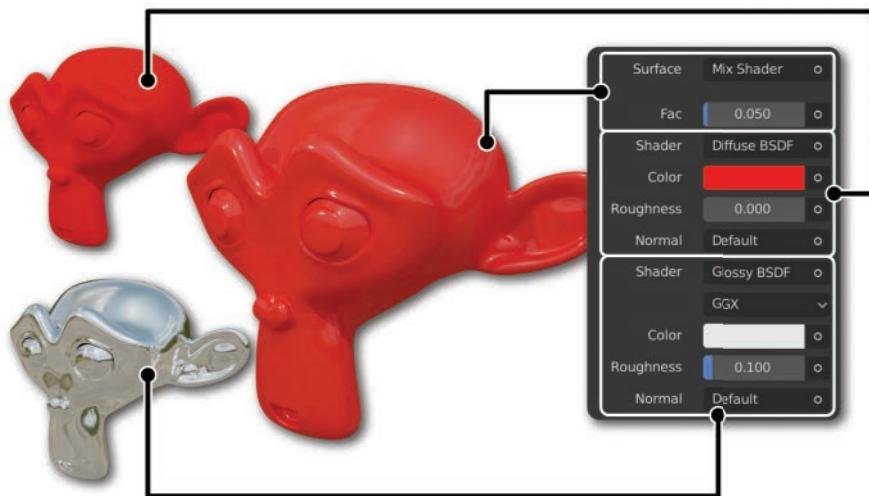


Figure 10.10 A Mix shader mixing two basic shaders to achieve a more complex result.

On the left is the monkey head with basic shaders (Diffuse BSDF and Glossy BSDF), and in the center is the resulting mix. The menu shows that the Mix shaders have a very low value for the factor property, making only a little of the Glossy BSDF shader come through, so a tiny amount of reflection is visible on top of the Diffuse BSDF shader.

Then you can control how much of each of the shaders you want to see by tweaking the value in the Mix shader's factor.

The Add shader doesn't have a factor parameter; like the Mix shader, it adds both shaders, generally making the result brighter. Still, you can control the intensity of the mix by making the added materials darker or brighter. If you're adding a Glossy shader on top of a Diffuse one, for example, you can control the amount of reflection by making the Glossy shader's color darker or brighter.

You can add Mix shaders inside Mix shaders (nesting) to get even more complex shaders. Do you see how complicated Cycles materials can get?

Loading Textures

Loading textures is very easy. To the right of each of a shader's properties is a button with a little circle. Clicking one of these buttons shows you a list of options. In that list, select Image Texture, and Blender displays below it the options for loading and controlling the image (see Figure 10.11). A new parameter called Vector shows up as well. Vector lets you define the texture's projection coordinates. If your object has UVs, the texture will use them by default. If you want to make sure that your texture is projected correctly, select the option Texture Coordinate | UV from the Vector list, as shown in Figure 10.11.

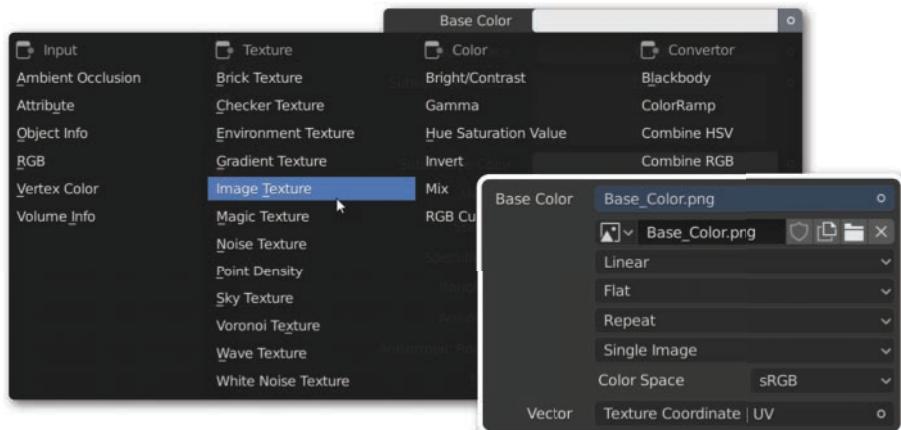


Figure 10.11 Next to each property that supports it, you'll find a button with a circle (in the image, it to the right of the Base Color's color selector). When you click that button, Blender displays the available options, one of which is an Image Texture. When you add an Image Texture, options for selecting the image and setting it up appear below the original property (Base Color, in this case), as shown on the bottom right.

Caution

When you click the circle button next to a shader's property to load a texture, you may not see the option you're looking for in the list. Pay attention to the top and bottom of the list that shows up: Sometimes, you have to hover your mouse at the top or bottom of that list so that the list scrolls, showing options that may not fit in the view.

Shading Jim

Now that you understand how materials work and how to add textures to them, you can jump into using shaders to make Jim look good.

Before you start, you may want to switch to Material Preview viewport shading mode and select a background. Doing this will show the result of the materials in EEVEE, in real time, but doesn't require you to add lights, for example.

Follow these steps to create Jim's basic shading in Cycles:

1. Select Jim's face, and add a new material called Jim_mat. A Principled BSDF shader will be added automatically. Load the image texture you painted in Chapter 9, "Painting Textures," in the Base Color property of the shader, and make sure that Vector is set to Texture Coordinates | UV.
2. Repeat Step 1 to load the textures in the Metallic and Roughness properties.
3. For the Normal Map, there are some extra steps to follow. First, in the Normal property of the Principled BSDF shader, instead of loading an Image Texture, load a Vector | Normal Map from the list that appears when you click the button for the Normal property. Within the options that appear, load the Normal Map image texture in the Color property. The last step is to choose Non-Color for the Color Space property of the Image Texture (sRGB by default) to make Blender understand that you want it to interpret this image as mathematical data instead of color information.
4. Add a material to the hair elements called Hair_mat, and set the Base Color blue, as shown in the character designs. You can adjust the Roughness value if you want the hair to look more or less greasy.
5. Add a material to the pieces of clothing on the arms, the gloves, and the belt. Call the material Metallic_details_mat and adjust the Metallic and Roughness properties to your liking to make those pieces shiny.
6. For the communicator in Jim's ear, you may need a few materials, depending on how you want it to look. You can add two plastic materials, one dark and one bright, for the different pieces. Also, create a material with an emission shader (instead of the Principled BSDF shader), and change its color to blue. This material can be added to the rounded bit in the center of the communicator to make it act as a light.

Only one part missing, and it's a tricky one: the eyeballs, specifically the cornea. The cornea will use a refractive material, which presents a little challenge, especially

because you'll need to set it up differently for EEVEE and Cycles. Before proceeding with the corneas, prepare the rest of the eyeballs:

1. Select the mesh of the cornea, and separate it into a different object from the rest of the eyeball by pressing **P** after selecting the mesh. If you already had the eyeball and cornea as separate objects in the modeling process, skip to Step 2.
2. Choose the colors for the pupil and iris materials. For both of those materials and the eyeball's material, adjust Roughness so that it's very low. This setting makes reflections less blurry, helping you sell the effect that the eyes are wet.

In the next section, you'll learn a few tricks for both EEVEE and Cycles.

Shading the Eyeballs in EEVEE

EEVEE can calculate refractions, but they're fake: The refractions are visible only if the refracted elements are on the screen (a trick used in videogames as well, called screen space reflections and refractions). This technique for rendering refractions isn't as accurate as actual refractions, such as the ones you can get in Cycles, but it can work quite well and is really fast to calculate. Here are the steps to follow to make the eyeball's corneas refractive in EEVEE:

1. On the Render Properties tab of the Properties Editor, enable Screen Space Reflections. In the Screen Space Reflections panel, active Refraction.
2. For the cornea material, increase the Transmission value of the Principled BSDF shader to 1 to make the material refractive.
3. In the Settings panel of Material Properties, enable the Screen Space Refraction option. At this point, you should see the refraction effect if you move the camera around Jim's eyes.
4. (Optional) When you move around the eyes, you may see some issues on grazing angles if you look at the eyes sideways. You can see how the border of the eyeball or the skin on the other side bleeds into the cornea—a limitation of real-time refractions. You can reduce the refraction effect by increasing Refraction Depth slightly (to 0.001 would be enough) in the Settings panel of Material Properties, right below the Screen Space Refraction option that you enabled in Step 3. Refraction Depth tries to imitate thickness in the refractive surface, sometimes alleviating such issues by sacrificing refraction effects. You can play with that value to see what it does.

With these adjustments, Jim should be ready to take a render test in EEVEE. You'll see how to do that shortly.

Shading the Eyeballs in Cycles

Cycles has no problems with refractions, although they can be a bit slow to calculate, especially if you have a complex scene. The issue you'll find is that when you have refractions in the cornea, the cornea will probably become very dark. Why? The cornea is producing shadows over the iris and pupil, and these shadows are a type that Cycles can't deal with easily: caustics.

Caustics are rays of light that go through or are reflected on a surface. You surely have seen them in real life. Those weirdly beautiful light effects that happen on the table under a glass with some liquid inside are caused by caustics. They happen when light bounces or goes through a curved surface, and the rays of light get distorted in the process, converging on the surface on which they end up.

Some render engines can render caustics, but Cycles uses an algorithm that is not friendly with such effects. In real life, light would go through the cornea and let you see the pupil and iris, but in Cycles, given the problem with light going through refractions, what's behind will appear dark. There are workarounds and tricks for mimicking caustics in Cycles, but they require some advanced knowledge of Cycles and nodes.

This is how you can get the right result in Cycles:

1. Create a sun light, and place it so that it illuminates Jim's face. To work on the Cycles shading, you need to add some lighting to the scene; otherwise, everything else will be dark. In EEVEE, this light is not strictly necessary because the Material Preview viewport shading mode is based on EEVEE and shows similar results. In Cycles, however, you need to work in Rendered viewport shading mode.
2. Select the cornea object, and go to the Object Properties tab of the Properties Editor (the tab with a square yellow icon).
3. In the Ray Visibility subpanel of the Visibility panel, disable the Shadow option. This step stops the cornea object from casting shadows while keeping all the rest of its properties, such as refraction effects and shiny surface.

Remember that you can change the IOR in the shader's settings to change the amount of refraction to your liking. This technique has different effects in Cycles and EEVEE, and I encourage you to try it in both to see how it behaves. You can see the result of Jim's shading in Figure 10.12.



Figure 10.12 Jim's materials are ready. In this image, you see the result of the shading in Material Preview viewport shading mode. In the viewport shading options, on top of changing the environment for the preview, you can increase the world's visibility and blurriness behind your 3D models.

Running Render Tests

Jim is already shaded, so it's time to do some test renders to see the results of your work. You'll see how to do those render tests in both EEVEE and Cycles, which require slightly different setups, but the basics are the same.

Adding Lights and Environment

Before anything else, it's important to add some lights so that the render isn't totally dark. You'll also add a background environment to the scene. These features are compatible between EEVEE and Cycles, so you have to set them up only once before you configure the render engines for the final render. Follow these steps:

1. Create a simple light setup so that the scene won't be totally dark when rendering. Create two suns: a main one that lights the character and a secondary one used as a rim light that comes from the side, with less intensity and a warmer color.
2. Create a camera (if you don't have one already) and place it so that you can see Jim clearly through it. Remember that you can press **NumPad0** to see from the camera view.

3. (Optional) On the Output Properties tab, change the resolution to give the image square dimensions.
4. Go to the World tab of the Properties Editor, and in the Surface panel, you see that World is similar to other materials. By default, World should have a background shader. Click Color to select a texture, choose Sky Texture, and arrange the settings until they look good to you. If you click and drag over the sphere, you change the sun's direction. For best results, check how this setting affects the scene in the Cycles engine, as the Sky Texture doesn't behave well in EEVEE (although it adds some fill light). Adjust the strength if you want the light from World to be more or less powerful. You see the resulting scene setup in Figure 10.13.

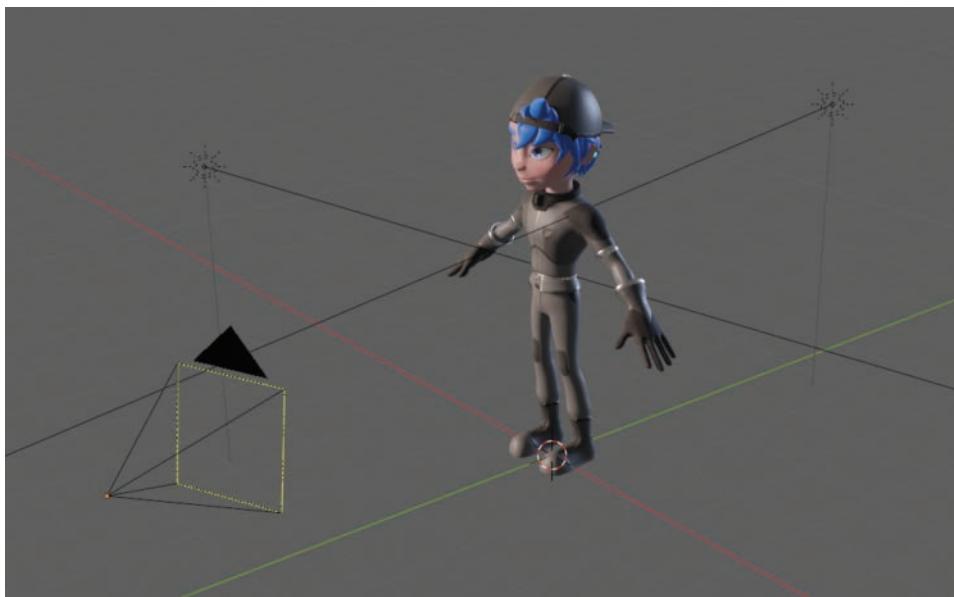


Figure 10.13 Render test scene setup with lights and a camera

If you want to get fancy with the background (which will also increase the detail of reflections in the materials and lighting), you could download a 360-degree image in .hdr format. Many free and paid resources online offer such images, but I recommend you try <https://www.hdrhaven.com>, a website that provides high-quality environment images for free (financed by voluntary donations). Choose and download one of the environments, and follow these steps:

1. Go to the World Properties tab of the Properties Editor. By default, the World's surface should have a Background shader.

2. In the shader options, you should see Color and Strength. Make sure that Strength has a value of 1.
3. In the Color property, load an Environment Texture (the same way you would add an Image Texture).
4. In the Environment Texture options, load the 360-degree image. Voilà!

Rendering in EEVEE

In this section, you'll see some of the options that you can enable to improve the render results for EEVEE. Follow these steps:

1. First, go to the Render Properties tab of the Properties Editor. You'll find most of the options to set up the render.
2. In the Sampling panel, you can increase the sampling. Generally, more samples improve quality (and take longer to compute). I encourage you to test results and render times with different settings for the samples to see what they do.
3. Enable Ambient Occlusion (AO). AO imitates soft shadows based on the proximity of objects. It may not seem to make much difference in the result, but if you pay attention to places where several objects are close, you'll see some improvements, such as between the face and the hair, and on the neck. Play with the settings for distance and factor. You can increase the factor over 1.00 if you click and type the number; this setting gives you a better view of what AO does to the render.
4. Enable Bloom. This setting gives a glare effect to shiny and bright parts of the model, adding some realism to the render, especially on metallic parts. For the light on the communicator in Jim's ear, you can increase the emission's strength to 30 or so to see how well that setting plays with the bloom effect. You can adjust the bloom settings to your liking. I recommend that you change values to see what they do.
5. You should already have enabled Screen Space Reflections and Refractions in previous steps. If you didn't, now is a good time to try enabling and disabling them to see the change in the results.
6. If you're aiming for better quality, try increasing the resolution for Cascade Size in the Shadows panel. Cascade Size affects mostly the shadows created by Sun lights, whereas Cube Size affects the shadows from Area lights.
7. (Optional) In the Film panel, you can enable the Transparent option to make the background behind Jim transparent.
8. In Step 6 you changed the size of the shadow's resolution. If you select each of the Sun lights, you'll find more options to improve the results. I strongly suggest enabling the Contact Shadows option. If you zoom in to Jim's face and try to enable and disable this option repeatedly, you'll find how much of a difference it makes.

Press **F12** to see the resulting render from EEVEE (see Figure 10.14). Many values can be tweaked to change the results and the quality of the render. Real-time render engines have many options for faking effects that you can enable or disable according to your needs. Decide whether you want to make sacrifices of render time to increase quality or prefer to lose some quality to have quicker renders.



Figure 10.14 Resulting render with EEVEE

Rendering in Cycles

In EEVEE, you need to make many adjustments to improve the results. Cycles is much slower than EEVEE, but it generally gives you better results with barely any setup. Here are some options that you could adjust to render Jim in Cycles:

1. If you've been working with EEVEE until now, go to the Render Properties tab of the Properties Editor, and at the top, switch Render Engine to Cycles. Next, I recommend switching the viewport shading mode to Rendered so that you can see what's going on in Cycles.
2. If you have set up your GPU in User Preferences (System tab), you should see a Device selector. If your graphics card is powerful, I suggest switching to GPU Compute instead of CPU, as you'll generally enjoy faster renders.
3. In the Sampling panel, you can increase the number of samples for the render to get better results (at the cost of more render time) and less noise. You can also enable Adaptive Sampling. This option renders the number of samples that you set up as a maximum, but if some areas don't need that many, Blender can render fewer samples in those areas to reduce render time. If Blender detects that some areas have no noise before reaching the number of samples that you defined, it goes to the next area after calculating less samples. Depending on the scene and your computer, the Adaptive Sampling option alone can easily reduce render times by half. In this render test, for example, a setting of 2,000 samples (a lot) takes 52 seconds on my computer with Adaptive Sampling off versus 24 seconds with Adaptive Sampling on. For this render test, I suggest using 300 samples. Keep in mind that you can set samples for both the 3D Viewport and for the render. The viewport samples will affect the 3D Viewport in Rendered shading mode, and the render samples will affect the final render (press **F12** or launch it from the Render menu).
4. (Optional) In the Film panel, you can enable the Transparent option. As in EEVEE, this setting makes the background behind Jim transparent.
5. If you launch a render (**F12**), you may still see some noise in areas that are poorly lit, such as the neck. Dark areas of the render don't receive enough light, which makes the calculations more difficult, so the noise takes many more samples to be cleared. You can enable Denoising to improve the result. But remember that denoising isn't black magic; if you don't render enough samples, denoising will make your render look patchy and blotchy. Denoising is meant to clear very fine noise in the render, even though in certain conditions, it can perform miracles. To enable denoising, go to the View Layer Properties tab of the Properties Editor; the Denoising option is at the bottom. You can play with the Denoising settings, but if the render has enough samples to get an almost-clean result, the default settings should work just fine. Denoising works only during rendering. You can see the resulting render in Figure 10.15.



Figure 10.15 Resulting render with Cycles

After this, you may not see much difference between the Cycles and EEVEE resulting renders. That result speaks volumes about the quality that you can get with EEVEE, but keep in mind that it's difficult to see the differences in simple scenes like this one. If you start paying attention to the details, you will find that in this basic example, shadows are much more accurate in the Cycles render than in EEVEE. (Look at the area where the hand meets the glove, for example.) You would see many differences if you had complex effects such as reflections and refractions.

EEVEE is very powerful and fast but has its limitations. Cycles, on the other hand, is much more accurate in complex cases at the expense of longer render times. My advice is to learn to use both and choose which to use on a project basis, depending on your specific needs. Blender makes it extremely easy to switch between Cycles and EEVEE,

so if you stumble upon some limitations, you shouldn't have many issues changing render engines at any point of your project (although not always and not for all purposes, of course).

Summary

At this point, you know how to create basic materials, use textures, and previewing them in EEVEE and Cycles. Materials let you define how the surface of a 3D model looks like when rendered and how it reacts to light or its environment. You can use textures to specify how different parts of a material affect the surface and to add variation to the material so it's not flat. By now, your character isn't a flat colored model anymore; it now has colors and looks more alive.

Exercises

1. What's the advantage of using textures that control material properties?
2. What are some of the differences between the EEVEE and Cycles render engines?
3. Can you make a mesh emit light in EEVEE? Can you make a mesh emit light in Cycles?



Bringing Your Character to Life

11 Character Rigging

12 Animating Your Character

This page intentionally left blank

11

Character Rigging

Rigging is probably the most technical and complex part of the character-creation process. Your character exists already and has a shape, but it's static: It needs a skeleton that moves and deforms the 3D mesh properly so that you can pose it, animate it, and bring it to life. In this chapter, you learn the basics of creating skeletons, rigging them (which means setting your skeleton up so that it works as expected and can be used intuitively and comfortably), and skinning a 3D mesh (which is the process that defines how the skeleton deform a mesh). After learning the basics, you'll use Rigify, an add-on bundled with Blender that allows you to automate the creation of a fully rigged skeleton for your characters. You also learn how to use Drivers to control the facial expressions of your character. When everything is ready, you set the character up to be reused in other scenes via linking or appending.

Understanding the Rigging Process

This section discusses the rigging process so that you have a better understanding of how everything works.

What's a Rig?

In Blender, rigs are called *armatures*. An *armature* is a container inside which you have the bones that conform to the rig. The purpose of a good rig is to make it easy and comfortable for an animator to control the character. Here is a list of the things that make up a rig:

- **Bones:** Everything inside a rig is made of bones, and the bones can have different uses, depending on how you set them up.
- **Hierarchy:** Bones have relationships that define how they behave. You can have parents and children: Children follow the movements of parents, but not the other way around. Imagine an arm. The forearm would follow the arm, the hand would follow the forearm, and the fingers would follow the hand; at the same time, the arm would be the child of the back bones, so if you move the character's back, the arms would follow. It's a top-down hierarchy that makes the bones stay connected as you move them. Without such a hierarchy, you'd have only independent bones that you'd have to move separately one by one, making things much more difficult.

- **Deform bones:** These bones deform the character model. Their purpose is to deform the mesh, so usually they’re hidden and moved by the control bones.
- **Control bones:** When you’re posing the character, it’s very helpful to have a set of bones made exclusively to control the entire rig. The leg is made of several bones, for example, but you can move them all with a single control bone. Later, you’ll animate control bones—the ones you can select and transform.
- **Helper bones:** These bones are very important, as they make the rig work. They exist only to help the rig behave as you’d expect, although they’re hidden, and you shouldn’t move them by any means. They’re also moved by control bones. You can consider them to be the engines and gears of the rig; they make it function, but they’re under the hood.
- **Constraints:** Constraints define how bones behave and react to other bones. You can tell a bone to follow the position of another bone, copy its rotation, limit its movements, or do other cool things such as looking at another bone (which is how eyes are rigged). You can think of constraints as modifiers that tell the bones what to do when you move the rig in certain ways.
- **Custom shapes:** Control bones themselves are not self-explanatory when you see them, as they’re just bones; that’s why you can assign custom shapes to them and give them a nicer, more intuitive look, which is how you customize the visible part of the rig. Thanks to custom shapes, you can make a bone look like an arrow, a circle, or any other shape that represents the function of a particular bone.

Note

In other software, each bone may be a different object, and dummies or helpers (called *empties* in Blender) can also be objects that are used by a rig. That can make it difficult to control the full rig at the same time and can mess everything up when you want to scale your character up or down, or duplicate it if need be. In Blender, on the other hand, a character’s rig is a single object, which makes it really easy to place it in the scene, scale it, or duplicate it. Inside that object are only bones, to which you can add custom shapes to make them look better, more intuitive, and easier to select.

Rigging Process

Here is the usual workflow that you need to follow to rig a character so you have an overview of what you’ll learn in the rest of this chapter:

1. Create an armature.
2. Enter the Edit Mode of that armature, and create the main bone structure and hierarchy.
3. In Pose Mode, add constraints to set up the rig, and jump to Edit Mode as needed to add helper bones.

4. When the rig is working, add custom shapes to it, organize the bones in layers and groups, hide the bones that aren't meant to be seen, and add anything else that will help you control the rig later.
5. Skin the meshes to the skeleton so that it deforms, and through weight painting, define the influence that each bone has over the vertices of the model. Your character is ready to animate!

Although these steps are the typical process if you start from scratch, you'll learn how to automate part of the process by using Rigify, an add-on that allows you to generate a fully functional rig adapted to your character.

When you use Rigify (or other add-ons that generate the rig automatically), the process is slightly different:

1. Create a Rigify skeleton (known as a *metarig*).
2. Adapt the metarig to the proportions of your character so the skeleton fits the model correctly.
3. Generate the final rig automatically, using the Rigify options. This step creates all the bones, hierarchies, constraints, and custom shapes necessary for adapting to the size and proportions you defined for the Rigify skeleton. After this step, you can delete or hide the original metarig.
4. Proceed with the skinning and weight-painting process so the rig deforms the mesh.

This sounds phenomenal! But if Rigify does everything for you, why learn to do it yourself? Well, it's important to know at least the basics, because depending on your character, you may need to tweak and adapt some parts of the rig that Rigify generates.

For Jim, you'll use Rigify to create the general rig in a few minutes instead of the hours or days that it would take to do so manually, especially with all the features that Rigify provides. But this rig won't have facial and eye controls, for example, so you'll have to create them yourself.

This mixed approach lets you learn the basic of both methods so you can quickly generate rigs with Rigify, but you'll also know the most important concepts of rigging so you can adapt automatic rigs or build basic ones on your own.

Additionally, automatic rigs work well for standard characters, such as bipeds or quadrupeds, but sometimes you have characters (or objects) that need rig shapes and structures that won't be achievable with autorigs.

Working with Armatures

In this section, you learn how to create and edit armatures. You also learn how to access the armature or bone's properties and add constraints.

Manipulating Bones

You create an armature by pressing **Shift+A** in Object Mode and clicking Armature/Single Bone. To create a skeleton and modify the shape of the bone structure, switch to Edit Mode (**Tab**). In Figure 11.1, you see the elements of a bone.

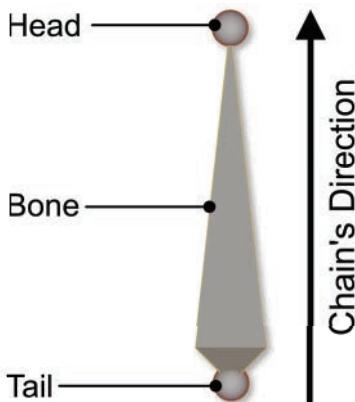


Figure 11.1 The parts of a bone

When you have a series of bones connected in a line, that line is called a *chain*. A bone's direction goes from its head to its tail, which is important because the bone's direction defines the bone chain hierarchy's direction as well. A bone connected to the tail of another bone follows the movements of the latter and is its child. You learn more about parenting and bone hierarchies later in this chapter.

Here are some of the things you can do with bones:

- Select them by clicking them and then move, rotate, and scale them the same way you do everything else in Blender (**G**, **R**, **S**). Bones are made of the bone itself and the spheres at the top and bottom, which are the bone's head and tail (also called *joints* when they're between two bones). You can transform the entire bone or just the head or tail to adapt the bone to the shape you need.
- You can access many properties of a bone from the Sidebar in the 3D Viewport (press **N** to show or hide it), inside the Transform panel of the Item tab.
- To create a chain of bones, you can select the tail of a bone and extrude it by pressing **E** or by pressing **Ctrl+RMB** in the position where you want the new bone to end, just as though you were extruding vertices. When you extrude a bone from the tail of another, the new one is a child of the originally selected bone. If you extrude from the head, Blender creates a new bone with no parenting applied to it.
- In the Tool Properties tab of the Properties Editor (or in the Sidebar of the 3D Viewport), on the Options panel, you can activate the X-Axis Mirror option. If

you do, mirrored parts of the skeleton mirror the transforms you do on one of the sides. To use this option properly, from a bone that is situated in the middle (the mirror plane), press **Shift+E** to extrude, which creates a first mirrored extrusion. Keep extruding and transforming the bones from that bone chain normally (by pressing **E**), and Blender mirrors them on the other side.

- You can duplicate bones by selecting them and pressing **Shift+D** and then moving them around (as you'd do when duplicating objects or meshes). Remember that with meshes, duplicating in Object Mode creates a different armature object, whereas duplicating a selection of bones in Edit Mode duplicates those bones within the same armature object.
- You can rename bones on the Properties Editor's Bone Properties tab. Another way to rename objects or bones is to press **F2**: a pop-up appears with a text field in which you can enter the new name. Keep in mind that armatures have three levels of names: the object's name (the *container*), the armature's name (as you can load an armature in a different object, as you would do with meshes), and the bone's name.

If you look at the Properties Editor, you'll notice that the tabs are slightly different from when you're working with meshes (see Figure 11.2). When you're naming the selected bone in the Properties Editor, be sure to do so on the Bone Properties tab; otherwise, you'll be renaming the armature or the object itself (the object that contains the armature that holds all the bones inside).

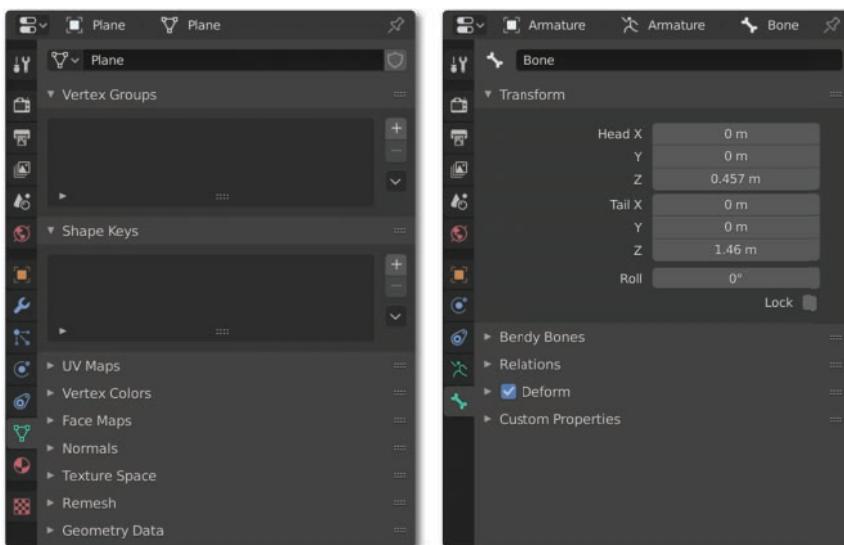


Figure 11.2 The Properties Editor's tabs change depending on the type of object that is the active selection. Left: Properties Editor when a mesh is selected.
Right: Properties Editor when an armature is selected.

- In Edit Mode, you can define the hierarchy of the bones. Select the ones you want (children) to follow a given bone (parent), then select the one you want to act as the parent, and press **Ctrl+P**. You see two options: Connected, which joins the tail of the parent with the head of the children, and Keep Offset, which performs the parenting operation while keeping the children disconnected.

To remove the connection between an object and its parent, select the object you want to set free and press **Alt+P**. You see two options: Clear Parent, which removes the relationship with the selected bone's parent, and Disconnect, which separates the head from the tail if they were connected but preserves the relationship.

Be careful with this option, since pressing **P** while having a bone selected (as with meshes) separates that bone into a new armature.

- If you select one or more bones, you can roll them by pressing **Ctrl+R** to control orientation. Pressing **Shift+N** gives you several automatic orientation options. Active Bone is a useful option, as it aligns the orientation of all selected bones to the active bone (the last one you selected); this option is handy for orienting the bones of chains, such as fingers, arms, and legs.
- If you select two joints and press **F**, Blender creates a new bone to fill the gap. This process is similar to how the Make Edge modeling tool creates an edge to connect two vertices in a mesh. Keep in mind that only the bone's head is connected to its parent; its tail will be free, and you have to parent the tail with the next bone to weld the bones.

If you select one or more bones, pressing **RMB** opens the contextual menu, where you can choose the Subdivide option to divide the selected bones into shorter ones. You can set the number of divisions from the Adjust Last Operation menu (see the bottom-left corner of the 3D Viewport or press **F9**).

In the same contextual menu, you can find many other options. Try them out!

- You can turn several bones in a chain into a single bone by dissolving bones or joints by pressing **Ctrl+X** (as you would do with vertices in a 3D mesh).
- If you want a chain to work in the opposite direction, you can switch the direction of a bone by pressing **Alt+F**. This keyboard shortcut switches the head and tail of a bone, changing the direction of the hierarchy.
- If you have undesired bones in your skeleton, delete them by pressing **X** or **Del**.
- As always, you can hide and unhide bones by pressing **H** and **Alt+H** to display only the ones you're working with or to hide some bones that are in the way. **Shift+H** hides everything except the selected bones.

Note

Remember that you can access all these options (or at least most of them) from the Armature menu of the 3D Viewport's header, but I show keyboard shortcuts so that you can start getting used to them.

Working in Object, Edit, and Pose Modes

The interaction modes for armatures are different from those for other objects. It's important to understand what you can do in each mode:

- **Object Mode:** The full rig is inside the armature object, so in Object Mode, you can move it around, rotate it, or scale it to change the size of your character. As the rig is inside this object, the scale of the object won't affect it or cause issues with its contents (bones).
- **Edit Mode:** In Edit Mode, you have access to the bones inside the armature. You can build your character's skeleton, define the structure's shape, and do parenting to define bone hierarchies. The position of bones in Edit Mode is the default pose of the bones in other modes.
- **Pose Mode:** When the hierarchy and the bones are in place in Edit Mode, you should go to Pose Mode to add constraints to the bones, pose your character, and set keyframes to create animations.

In Object Mode, you don't have access to the individual bones or controls; you can only transform the rig as a whole. In Edit Mode, you can modify the bones' position, size, and orientation to fit your character and also define the hierarchy. Finally, in Pose Mode, you can add constraints to the skeleton so that it works as you expect; then you can pose it. While you're setting up the rig, you'll be jumping a lot between Edit Mode and Pose Mode to create and adjust bones while you add and tweak the constraints.

When you're in Pose Mode, selected bones are shown in blue as a reminder that you're in that mode.

Tip

When switching between these modes, keep in mind that you can use keyboard shortcuts to accelerate the workflow. If you're in Edit Mode, **Ctrl+Tab** shows the pie menu that allows you to switch to Object or Pose mode; if you press **Tab**, you switch to the last mode you were in. In Object and Pose modes, pressing **Tab** always takes you to Edit Mode, but pressing **Ctrl+Tab** in Object or Pose mode switches between modes instead of showing the pie menu.

Bone Hierarchies

It's important that you understand very well how bone hierarchies work. You can change hierarchies from Edit Mode by defining parents and children. When you create bones by extruding other bones, hierarchies are generated automatically. Figure 11.3 shows examples of hierarchies, illustrating how they are represented by the bones' shapes. Here are some things to keep in mind:

- Bones can be independent of other bones, but they're generally part of a hierarchy.

- There's usually a bone that dominates the hierarchy, and all the other bones are its children. This bone is called the rig's *root*.
- Complex rigs can have forks in the hierarchy. In a character, the root can be at the hips; from the hips, the hierarchy forks to the back bones and each of the legs; from the back, it forks again toward each of the arms; and from the hands, it forks to each of the fingers.
- These forks happen only in one direction—from parents to children—never the other way around. One parent can have many children, but children can have only one parent. Simultaneously, of course, children can be the parents of other bones.
- You could imagine a treelike schematic to understand a bone hierarchy, in which you have a trunk, and then branches, subbranches, and so on.

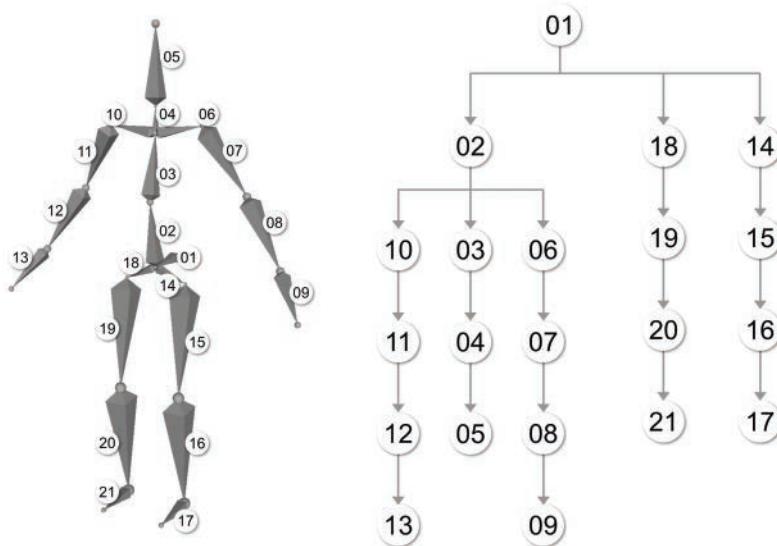


Figure 11.3 A simple diagram showing how hierarchies work. The skeleton bones are numbered, and in the schematics, you can see how the skeleton hierarchy flows.

Adding Constraints

Constraints make your rig work. You could say that constraints react to other bones. If you move a bone, you provoke a certain action in another part of the rig. Throughout the rest of this chapter, you use several constraints in Jim's rig, but for now, I describe how they work and how to add them to your bones.

First, you need to know that most constraints have a target, which means that they're assigned to a bone but target another bone to create the constraint between them.

If you use the Track To constraint for an eye, for example, you apply the constraint to the eye and select the bone that you want the eye to look at as the target.

Caution

You can apply constraints to any object in your scene, but understand the difference between object constraints and bone constraints. If you're in Object Mode and add a constraint, that constraint affects the entire rig. If you're in Pose Mode, Blender displays a different tab: Bone Constraints. The Object Constraints tab has a conveyor-belt mechanism as its icon, and the Bone Constraints tab shows the conveyor belt, with a bone as one of the gears of the mechanism. Check Figure 11.2 to see the differences in the icons.

You can add constraints in two ways when you're in Pose Mode:

- Select the bone to which you want to add the constraint. On the Bone Constraints tab of the Properties Editor, click the Add Bone Constraint button, and select the constraint type you want to add. The constraint is added to the list of constraints, affecting the selected bone in a way similar to what modifiers do. Inside the constraint panel is a Target field. In that field, enter the name of the armature. A new field for the bone's name appears. Enter the name of the bone you want to have as the target of the constraint.
- A faster method is to select the target bone first and then select the one to which you want to add the constraint while holding down **Shift**. Press **Shift+Ctrl+C** to open a menu and add constraints (or choose the constraint from the Armature menu on the header). This way, when you add a constraint, it automatically picks the first selected bone as a target, and you won't have to add it manually in the constraint panel.

Tip

When you have to insert the name of an object into a field, start typing the name, and a list appears, displaying the names of objects that start with the letters you typed. (Keeping your objects named correctly can help a lot!) If you don't type anything but click the text field, you see a list of all the objects that fall into the selectable object category for that specific field.

Also, to copy/paste a bone's name, place the mouse cursor over its name field in the Bone Properties, press **Ctrl+C** to copy it, select the object with the constraint, go to the desired field, place the mouse cursor on top of it, and press **Ctrl+V** to paste it.

There is yet another way to see the name of the bone you want to input in a constraint's field. In the Armature Properties tab, go to the Viewport Display panel, and enable the Names option. This option shows the name of each bone in the 3D Viewport. This process can get confusing if you have many bones, but it can help in certain situations.

Eyedropper

In image-editing software, the eyedropper is a common tool used for clicking a pixel and selecting that pixel's color. In Blender, the eyedropper is now used to select objects from different menus. When you add a modifier or constraint that requires a secondary object to work with, for example, you don't need to know the name, type it, or select it from the list; you click the eyedropper to the right of the text field and then click the object inside the 3D Viewport. Blender picks that object and introduces it into the text field for you.

Although the eyedropper tool is available for bone constraints, its behavior is a bit tricky. You can use it to pick the armature's name (Object), but the eyedropper won't work with bones; you have to select bones from a list or type them yourself.

Forward and Inverse Kinematics

Before proceeding, you should understand inverse kinematics (IK), which is one of the most-used constraints.

Usually, when you create a bone structure such as a leg, if you try to pose the structure, you have to rotate each bone, and its children follow it. For the leg, you should rotate the thigh first, followed by the ankle and then the foot.

This way of working is called forward kinematics (FK), but for legs or parts of a rig that need to touch surfaces (such as the feet), IK is a much more convenient option. IK works the opposite way from FK, so to pose the leg, you have only to move the foot; the knee flexes to fit the foot's position.

This feature is useful because when you move the character's torso, for example, the legs are automatically flexing and moving to make the feet stand on the floor.

Advanced rigs mix both methods so you can switch between them, depending on your needs. As an example, if your character is standing on the floor, using IK for the legs might be more comfortable, but if the character is jumping and the feet are in the air, you may prefer to use FK.

Rigify automatically generates this feature for Jim's rig. Generally, you have three skeletons for the leg: the FK, the IK, and the one that deforms the mesh, which adapts to FK or IK as you need it to.

Practice with Bones and IK Constraints

In this section, you'll complete a simple exercise that takes you through the process of creating a chain of bones, adding two control bones, and adding an IK constraint to make the skeleton work. You can see the steps of the process in Figure 11.4.

1. Create an armature by pressing **Shift+A**. From the Armature menu, choose Single Bone. Switch to Edit Mode so you can modify the bone you just created.
2. Move the bone up, and rotate it to place it horizontally. This bone will represent the hips.

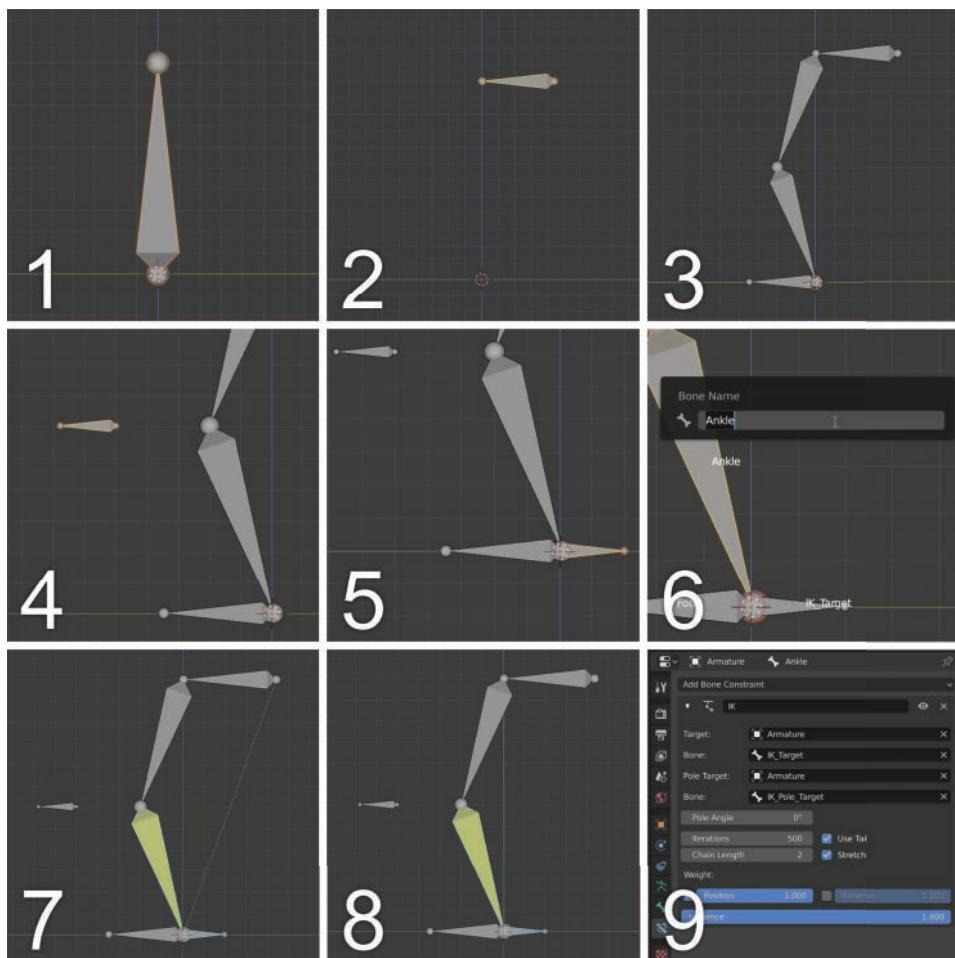


Figure 11.4 The main steps for creating an simple rig with an IK constraint

3. Select the tail of the bone and extrude it to create the leg, extrude it again to create the ankle, and extrude it one last time to create the foot. Remember that you can extrude bones by selecting their tail and pressing **E**.
4. Select the knee joint, and extrude it forward. Press **Alt+P**, and select Clear Parent. Then move the bone forward a bit to separate it from the leg. This bone will later control the knee orientation (using it as the pole target for the IK constraint).
5. Select and extrude the foot joint. Press **Alt+P**, and select Clear Parent. This bone will be the IK target, which will control the leg-flexing movement automatically. Essentially, the IK constraint makes the leg bend to adapt to the positions of the leg's parent bone (the hips, in this case) and the IK target.

6. Now that all the necessary bones are in place, name the bones properly so that it'll be easier to select them. You can easily name a bone by selecting it and pressing **F2**.
7. It's time to add the IK constraint. Switch to Pose Mode. Select the IK target bone, select the ankle bone while holding **Shift**, press **Shift+Alt+C**, and choose Inverse Kinematics from the third column. This process adds the IK constraint to the active bone (ankle) while automatically using the other selected bone as the IK target. When you do this, of course, you could also add the constraint from the Bone Constraints tab of the Properties Editor. The ankle should turn yellow, indicating that the bone has constraints affecting its behavior.
8. You need to set up the IK constraint's options. When the IK is added, you should see a dotted yellow line from the ankle's tail to the head of the hips. This line represents the chain of bones that will be affected by the constraint, which by default goes from the tail of the bone that has the constraint up to the beginning of the hierarchy (the hips, in this case). You need to limit the length of the chain so that it affects only the leg. Change the Chain length of the IK constraint to 2 so that it goes up the hierarchy by only two bones. (By default, the chain length is 0, which makes Blender go through the entire hierarchy.)
9. Another thing to add is the pole target. From the constraint panel's Pole Target field, choose the Armature. When the Bone field appears, select the pole target you created (the bone that you extruded from the knee joint).

Bone Orientation and Constraints

Due to the orientation of the bones, sometimes when you apply a pole to an IK chain (or other constraints), they get rotated. You can compensate with the Pole Angle value of the IK constraint. Usually, if bones are aligned correctly, round values such as 90, -90, and 180 are enough to correct orientation.

Alternatively, you could roll the leg's bones by pressing **Ctrl+R** so that they are aligned correctly. Enabling the display of axes for the armature enabled is helpful, as this display allows you to align the bone axes correctly. The X-axis of each bone in the chain (in this exercise, the leg and ankle) should be aligned with the pole target.

The orientation of the bones depends on many factors, including the point of view from which you created them and the rotation of the bones from which you extruded them. This trick lets you fix orientation issues that may pop up when you add constraints.

Now you can try the rig for yourself and see the effect of an IK constraint. You can move either of the chain's extremes: the hips or the IK target bone. When you do, you'll see how the leg bends to adapt to their position.

If you move the pole target (the bone in front of the knee joint), the leg changes its orientation to "look" at it.

Caution

It's important to pay attention to the IK chain's shape. In a leg, for example, the knee should be a bit bent, forming a triangle among the hips, knee, and ankle. This slight angle is important because it defines how the joint will bend when you add an IK constraint.

If you had the bones of the leg and the ankle positioned completely straight, without an angle between them, for example, the IK constraint wouldn't easily detect which direction it should bend.

Rigging Your Character

Now that you know the basics of manipulating bones, you're ready to create Jim's rig, starting with using Rigify to generate a functional automatic rig.

Enabling the Rigify Add-On

Rigify comes with Blender but is disabled by default. To enable it, choose Edit > Preferences, and click the Add-Ons section. In this section, you see a list of the add-ons that come with Blender (and the ones you've installed); you can search for Rigify manually or type **Rigify** in the search field in the top-right corner of the interface. When you find it, click the check box before the add-on's name to enable it (or disable it if it was enabled before).

Add-ons give new options to Blender, so you may see extra buttons, menus, or options when you enable add-ons. Many of these tools are disabled by default because not everyone needs them all the time. It's useful to enable or disable such options on demand to have a less cluttered interface.

A Few Tips Before You Start Rigging

Before you start, I'll share with you a few tips that can help when you're rigging a character and working with bones and armatures:

- On the Armature tab of the Properties Editor, find the Display panel, and activate the In Front option. This option always displays the bones on top, even if they're inside the mesh, which makes it easier to see what you're doing while keeping the mesh visible behind so that you can align the bones to it.
- Naming bones is essential when you're working on a rig so that when you add constraints later, you'll know which bones you're referring to. It's a lot easier to find a bone called D_hand than a bone called bone.064, for example.
- Did I say D_hand? Why the D? You can use prefixes to help yourself recognize the type of bone. D_name would be a bone that's part of the main structure that will deform the mesh, C_name would be a controller bone, and H_name would be a helper bone. Also, some of the bones may be used for deformation as well as

for control, such as the bones in the spine. Those bones use two prefixes: C_D_ name. Using this naming convention helps when you search for a bone in a list, as all bones will be organized by type in the list that shows up when you enter the prefix.

- When you’re testing a rig in Pose Mode, it’s very easy to reset the bone’s default pose (the one defined in Edit Mode). In the Armature menu of the 3D Viewport header, you find the Clear Transform option, which gives you different ways of resetting the pose. Otherwise, you can reset the movement by pressing **Alt+G**, reset the rotations by pressing **Alt+R**, and reset the scale by pressing **Alt+S**.

Using Rigify to Generate Jim’s Rig

Remember to enable the Rigify add-on (as explained in the preceding section) before you start this process. Otherwise, you won’t see the options used in this section.

Figure 11.5 summarizes the process: Create a base skeleton, adjust its shape (in the figure it’s exaggerated so that you can see the change, but you’d adapt it to your model), and generate the rig with Rigify. In the following sections, you’ll learn how to perform this process step by step.

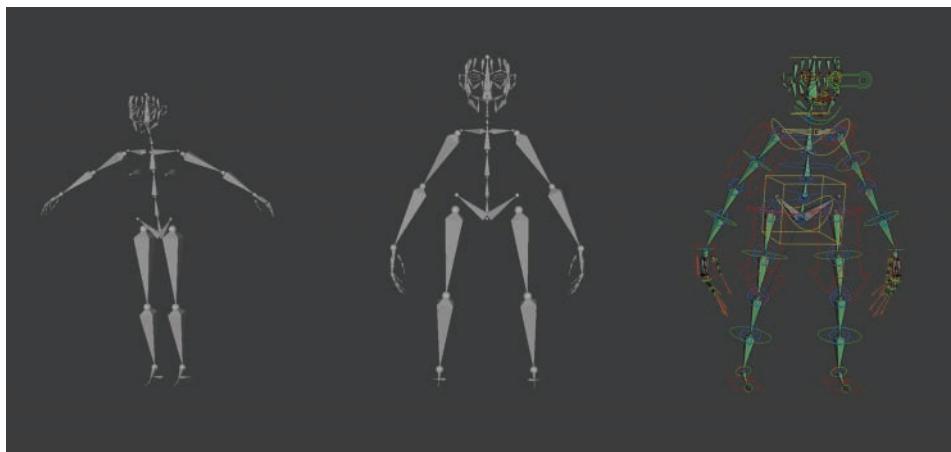


Figure 11.5 Creating a rig with Rigify. Left: Create the base skeleton.

Center: Adjust the skeleton to the desired form. Right: Generate the final rig with all the necessary bones and controllers, ready for animation.

Introducing Rigify

Rigify is an add-on that provides a few predefined skeletons for bipeds and quadrupeds; you can easily adapt these skeletons to the proportions of your character. When the skeleton fits your character, you’ll have options to generate a functioning rig automatically with all the necessary controls and bones, properly named and organized, and with the dimensions of the base skeleton that you adapted to your character.

Even though you may need to perform some adjustments in the resulting rig to adapt it to your needs, Rigify can save you many hours of work setting up bones so that they work as desired. All the constraints are already in place and set up, and the rig has advanced features such as IK/FK switches for arms and legs, stretching options for different parts of the body (especially useful for cartoon characters), and even scripts that provide an interface for the rig (which you'll learn about later).

Caution

Rigify uses some Python scripts for some of its features. For the rig to perform properly, without errors or missing features, you need to enable those scripts.

Generally, when you open a file that contains a Rigify rig, you'll be asked whether you want to execute the .py scripts within the file. Make sure to accept this option to avoid issues with the rig.

If you want, you can tell Blender to execute such scripts automatically: Enable the Auto Run Python Scripts option on the Save & Load tab of User Preferences.

Keep in mind, however, that you should use this option carefully and only if you trust the source of the .blend files that you're working with. Files from unknown sources may contain malicious Python scripts, so be careful about automatically running scripts contained in files whose creators you don't know.

Rigify automates the rigging process to eliminate repetitive work. It also lets you do things like duplicating the arms of the original skeleton and removing fingers, and then it generates controls accordingly.

Although I highly recommend that you learn basic and intermediate rigging skills so that you can make adjustments where necessary, add-ons like Rigify can certainly save you tons of time, especially when you'd have to repeat the same tasks for rigging characters that have very similar rigs except for their proportions (bipeds or quadrupeds, for example).

Creating and Adjusting the Skeleton

Let's get started! Up to this point, you haven't had to keep measurements and correct sizing in mind, so you'll have to go through the process of adjusting Jim's size.

Creating the Skeleton

Before adjusting the size, create the skeleton so that you can use it as a reference for the size:

1. Create a Human (Meta-Rig) skeleton. Press **Shift+A**, and inside the Armature category, you'll find Single Bone and other options provided by Rigify. Make sure that the skeleton is created in the center of the world: Press **Shift+S** and select Cursor to World Origin so that the 3D cursor is at the coordinates 0, 0, 0 before you create the skeleton.

2. Press **N** in the 3D Viewport to display the Sidebar, if it's not already displayed. Within the Dimensions section of the Item tab for that region, you see the actual size of the rig. By default, it's 1.90m on the Z-axis (height). Scale it down to about 1.70; that's how tall Jim will be.

The skeleton may be smaller or bigger than the 3D model. The height doesn't matter, though, because you'll fix it in the next section.

Adjusting the Sizes of the 3D Models

Using the skeleton as a reference, you need to scale the 3D meshes that make up Jim to fit the skeleton's height. Only the height matters at this point; you'll fix the rest later.

1. Make sure that the 3D cursor is in the scene's center, as it was when you created the skeleton, and select all of Jim's mesh objects. (Leave the skeleton unselected.)
2. Set the pivot point to 3D Cursor, and scale the selected objects up or down to fit the height of the skeleton. (Using the 3D cursor as the pivot point scales all of the model while keeping the feet on the floor.) It may be helpful to enable the In Front option for the skeleton's armature visibility so that you can see the skeleton on top of the models and adjust it easily. Keep in mind that the proportions of the skeleton are more realistic than those of Jim, so don't worry if the arms of the skeleton don't fit Jim's arms; just adjust the height so that the top of Jim's head fits the top of the skeleton's head.
3. When the scale of the models fits the skeleton's height, apply the scale to all the objects by pressing **Ctrl+A** and clicking Scale.

Caution

Remember that if you've used instances, when you apply transforms such as scale, Blender may display an error message instead of executing the action. When a mesh has multiple users, as in the case of instances, you can't apply modifiers or transforms, as those changes will also affect its other users.

In such a case, select those objects first and then choose Object > Relations > Make Single User > Object & Data to make the selected objects' meshes unique.

4. After applying the scale, you may see a side effect. Some modifiers, such as Solidify modifiers, may have their results change when the object has a different scale, for example. Analyze your model, find the modifiers that are providing wrong results, and adjust them. (In the case of Solidify, you may need to adjust their thickness.)

At this point, the base skeleton and the 3D models are the correct sizes and aligned.

Adjusting the Skeleton to the 3D Model

As explained earlier in this chapter, the Human (Metarig) is a base that you'll use to generate the final rig with Rigify. Before you generate that final rig with all the necessary bones and controls, you have to make that skeleton fit the shape and proportions of your character. You can see the steps in Figure 11.6.

1. Select the skeleton, and switch to Edit Mode.
2. Press **N** in the 3D Viewport to display the Sidebar, if it's hidden. On the 3D Viewport's Top Bar, click the Options button, and enable the X-Mirror option. This option allows you to position the skeleton on one of the sides, and the other side will replicate those movements, making the rig symmetrical and the process easier and less tedious.
3. Several bones are meant to be used for the facial rig. Select and delete them all. Do the same with the two bones for the breasts. You won't need the rig to generate those bones and controls, so if you delete them in the base skeleton, they won't be created when you generate the final rig. You'll create your own facial rig later in this chapter.
4. Select and place each of the skeleton's joints in the places that fit Jim's model.
5. When all the joints are in place, roll them to control their orientation. You can roll the selected bones by pressing **Ctrl+R**.

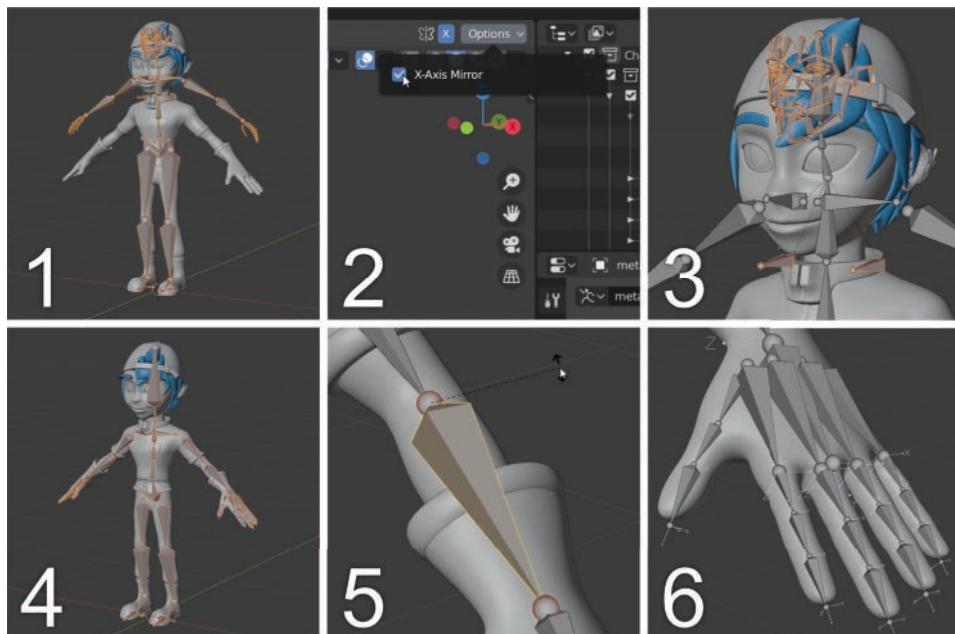


Figure 11.6 Steps for adjusting the base skeleton to Jim's model

6. There are many bones in the fingers, and it's very important to properly align them. Follow this procedure for each finger:
 - a. Roll one of the bones in the finger so that it fits the finger's orientation. It's useful to orbit around the bone with the camera to make sure that the bone is in the right position and oriented correctly.
 - b. Select all of the finger's bones, and select the bone you oriented last to make it the active selection.
 - c. Press **Shift+N**, and select the Active Bone option. This option automatically aligns all the selected bones with the active one.

You need to be careful with some things to make this process work correctly. If you don't, you may get error messages when you try to generate the final rig with Rigify:

- In the articulations, such as knees and elbows, you need to create an angle, as you did previously in this chapter to generate an IK chain. The reason is the same: That's how the rig will know the direction in which to bend those articulations.
- Some bones are children of others but different parts of the rig, so even though they have their heads and tails in the same places, they're not connected and can move independently. When you move those bones, always keep those joints together to avoid errors. The top spine bone and the first neck bone, for example, have an overlapping joint even though they're disconnected. Always move those bones together. If you get error messages about disjointed bones, correct the issue. You can align those bones by using the 3D Cursor, for example.
- When you place the bones, try not to think in terms of where the bone would be in reality, as this skeleton is not real. Place the joints in the center of the volumes of the character's body so that you have a better chance of proper deformation of the mesh later.

If it's not perfect, don't worry, you'll be able to make adjustments later.

Generating the Rig

Now that the metarig is in place, you can generate the final Rigify rig. First, it's important to apply the scale of the metarig skeleton if you adjusted its scale in Object Mode. If you don't, the generated rig will be created with the same proportions but a different size. You can apply the scale selecting the skeleton in Object Mode, pressing **Ctrl+A**, and selecting the Scale option.

When the scale is applied, generation should happen without hiccups if you followed the instructions provided in the previous sections.

In the Properties Editor, with the metarig skeleton selected in Object Mode, the Object Properties tab (in this case, Armature Properties) has three panels: Rigify Groups, Rigify Layer Names, and Rigify Buttons. The first two panels include options that let you control some aspects of the final rig, such as the colors of the parts and the names of the layers that will be created in the interface.

Although I encourage you to experiment with all those options, here I want to focus on the Rigify Buttons panel, which includes a button called Generate Rig. Press that button, and all the controls appear in the 3D Viewport. (Sometimes, it takes a few seconds for them to appear.)

Right below the Generate Rig button is Advanced Options, which you'll explore when you learn how to make adjustments to the rig.

Note

At this point, I recommend organizing the objects in the scene to make your work easier and more efficient. Right now, you have the metarig, the Rigify rig, and all the other objects that make up Jim.

You could separate the rigs in different collections, as you did with the reference images in previous chapters to hide and display them as necessary.

To do that, select both rigs, press **M** in the 3D Viewport, select New Collection, and enter a name. Now, in the Outliner, you'll have both rigs organized appropriately inside a collection. To find them and hide/show them, click the eye icon to the right of their names.

Organizing Bones

Your character's rig is already working, but you can organize it to increase usability. Two ways to do this are to use bone groups and armature layers. As you used Rigify, the rig that you created was already organized, helping you see how groups and layers can be used. Both the Bone Groups and Armature Layers panels are on the Armature Properties tab of the Properties Editor.

Bone Groups

Bone groups let you organize your bones comprehensively, change the display colors of the bone groups, and allow quick group selections. You can have different colors for different type of bones, such as Deform, Control, and Helpers groups. Figure 11.7 shows the Bone Groups panel on the Armature tab of the Properties Editor.

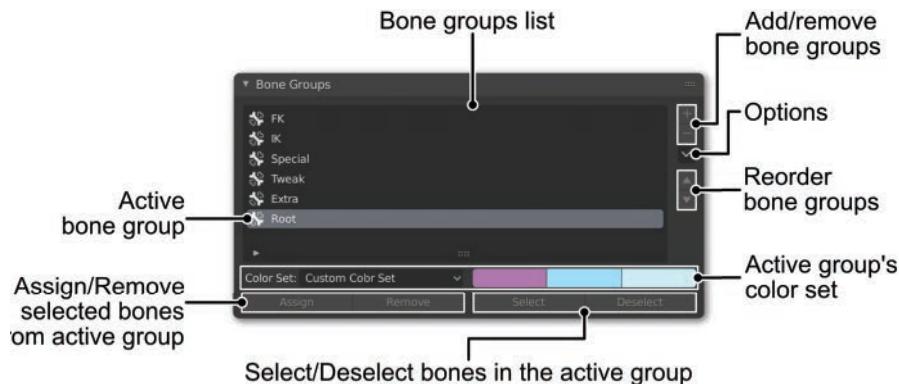


Figure 11.7 The Bone Groups panel

In the Bone Groups panel, you can add groups to or remove groups from the list, and when you click Assign, the selected bones become part of the active bone group (the one highlighted in the list). You can select a color set for that group or create your own color set by clicking the colors in that panel.

Rigify includes a setup of predetermined bone groups that you can explore and try. You can also add new bones (such as the eyes and facial controls you'll be adding later) to some of these groups or create your own.

Caution

You can see the colors of bone groups and add or remove bones only in Pose Mode.

Armature Layers

Armature layers allow you to show or hide sets of bones as you need them to make the rig easier to manipulate. On the Armature Properties tab, the Skeleton panel has four sets of little squares; each square represents a layer (see Figure 11.8). The first two sets, in the Layers section, are the armature layers themselves. The other two sets, in the Protected Layers section, allow you to mark specific layers as protected to prevent other users of the rig from manipulating them when linking the character. (I talk about linking near the end of this chapter.)

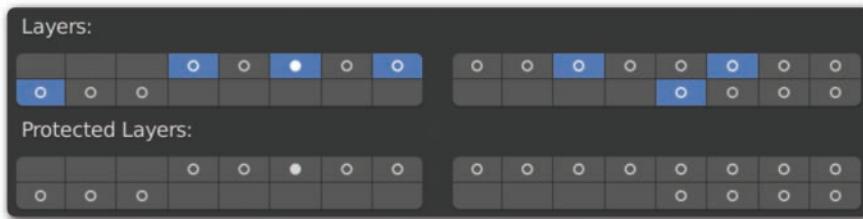


Figure 11.8 Armature Layers panel

When you're in Edit or Pose mode, you can select one or more bones and press **M** to access the Bone Layers menu. When you press **M**, a pop-up menu with little squares will appear, with each square representing a layer. This menu is the same as the ones in the Properties Editor. Select a square with a click, or select more than one bone by pressing **Shift+LMB** (the same bone can be part of more than one layer) to add the bones to the appropriate layer(s).

When you have bones added to layers, you can show or hide the layers from the Armature Properties tab of the Properties Editor. Left-click layers to show or hide them. Hold down the **Shift** key while you click to show or hide more than one layer.

As with bone groups, Rigify already uses layers, so you can enable or disable them to see what happens.

What you'll get from setting up bones in layers is that later, you can show or hide only the ones you need. When you're working on skinning, for example, you can show only the deformer bones so that the rest of the bones won't be in the way. When the rig is finished, you can hide the deformer and helper bones so that only the ones you should move to control the rig are visible. In other words, work is easier because you see only what you need to see.

While setting up Rigify, you deleted the face controls and bones, as you'll be creating your own later. You'll store that part of the rig on its own layer, which will let you work on the full character pose first; then you can show the facial rig and focus on the facial expressions. This way, the facial rig won't be displayed all the time; it could confuse you or cause you to move something accidentally when you're working on the body pose.

You can also have layers to store bones that help the rig function but that you don't need to see or manipulate.

Understanding the Rigify Rig

You have the rig in place, and you've learned about bone groups and armature layers. It's time to explore the main features of the Rigify rig, although you won't study them in depth; you'll learn how to use some of these features in Chapter 12, "Animating Your Character."

Rigify does a lot for you when creating the rig:

- Creates bones for deforming the mesh
- Creates bones that make the rig work with advanced features, such as IK/FK switches, squash, and stretch
- Creates the controller bones and their shapes (what you see instead of the bone itself to make it easier to select and understand what the bone does) so you can comfortably pose and animate the character
- Organizes all the bones in groups and layers
- Adds a script to the file that creates an interface to control many of the rig's features, visibility of layers, and specific bone properties

You can access Rigify's interface from the Sidebar in the 3D Viewport (press **N** if it's hidden). On the Item tab, you'll find a panel called Rigify Layers (if you have the rig selected in Object Mode). You can enable and disable layers to see what they show and hide. The buttons give you direct access to the armature layers, so clicking them has the same effect as turning the layers on and off in the Properties Editor. This menu just makes that process easier because that you have access directly from the 3D Viewport and because the buttons are named after the contents of the layers.

Tip

After you create the rig, you may see many dotted lines between bones. Those lines represent the parent/child relationships between objects and bones, but in complex rigs, many lines show up, potentially making work uncomfortable when they get in the way.

If that happens, when you're done setting up relationships and don't need to see the lines anymore, you can disable them from the 3D Viewport's Overlays menu (Objects section). If you need to see them again, you can turn them back on any time, of course.

If you jump into Pose Mode, you'll see much more! Depending on the bone you select, you'll see another panel on top of Rigify Layers: Rig Main Properties. Each control bone that you select displays different options in that panel. I encourage you to hover your mouse cursor over each option and read the tooltip description to learn what it does. You can also try the options and experience their effects firsthand.

As you can see, this rig is very advanced and provides a myriad of options that let you achieve complex poses and animations with the character. You'll learn how to use some of these options in Chapter 12, "Animating Your Character."

Performing Adjustments to the Rigify Rig

You can perform adjustments on the generated rigs in two ways: adjusting the base metarig and regenerating the rig, or adjusting the final rig directly.

Some parts of the rig may need to be adjusted because they may not be well aligned or don't work properly. Common problematic areas include the fingers, because depending on the model, their orientation, and other factors, they may not bend in the right direction when the rig is generated.

Fingers controls in the Rigify rig have long lines that end with a square. You can bend the fingers very easily by scaling those controllers up and down in Pose Mode. If you see that the fingers don't curl in the direction they should, you may need to adjust them.

In the following sections, you learn about two different ways to adjust the rig. These methods apply to any part of the rig, but for practical purposes, the following sections use the fingers as an example.

Adjusting the Metarig and Regenerating the Rigify Rig

After you generate the rig by clicking the Rigify button in the Properties Editor, the metarig will still exist in your scene. You can perform adjustments on that skeleton and update the final rig accordingly.

In the Rigify Buttons panel of the Properties Editor, right below the Generate Rig button, you'll find the Advanced Options button. If you click it, new options appear. I won't discuss every option, but the first one is the most important: a switch that lets you define whether you want to overwrite an existing rig or create a new one when you click the Generate Rig button. By default, the button should be set to Overwrite. This means that if you adjust your metarig skeleton and click Generate Rig again, the previously generated rig will be updated to fit the metarig's adjustments.

This option can help if you need to change the placement of some joints or roll some bones (the fingers, in this case) to adjust their orientation. The option is very useful, but sometimes, the rig generation process still has to guess many things based on the original metarig, which may result in inaccurate bone orientations. When that problem happens, and it can't be fixed by regenerating the rig, you need to adjust the final rig directly.

Adjusting the Rigify Rig Directly

After the rig is generated, you can still adjust its bones, but you need to be very careful, especially if you aren't experienced in rigging. Even the smallest changes can have many implications; many bones can affect others in many ways, given the large amount of constraints in the Rigify rig.

Caution

Before adjusting the final rig (the one generated by Rigify), make sure that the proportions and placement of the bones are correct and you won't need to re-generate the final rig from the metarig anymore. All adjustments made to the final rig would be lost when re-generating.

Let's see how you would adjust the fingers' orientation, for example:

1. Enable all the armature layers (holding down **Shift** while you click them to enable/disable more than one layer). This step is important because nearly every part of the rig has bones on several layers; you don't want to leave any bone behind when doing adjustments.
2. Select all the bones in each finger and adjust their orientation by pressing **Ctrl+R** to roll them. To see the orientation change, you can enable the axes' visibility on the Armature Properties tab of the Properties Editor.

You can jump back and forth between Edit and Pose modes to see how the adjustments you're making affect the rig controls, and you can even try to use the controllers to see whether the fingers are curling in the desired direction.

Caution

I recommend making such adjustments while in Wireframe viewport shading mode. Many bones have the same position, and there may be bones inside other bones, which will be easier to see and select when you're viewing the wireframe version of the bones.

You can adjust much more than just the orientation of bones with this method, of course. Just make sure that you don't leave any bone behind when you do so, because constraints between bones may make such adjustments go to waste and make things more complicated.

Skinning

Skinning is the process of telling Blender how bones have to deform the meshes. To do skinning, you use vertex weights. Each bone has an internal weight over the vertices to define how vertices follow the movement of bones. To see how weights work, see the following section.

Understanding Vertex Weights

In this section, you see a simple example of how weights work in a mesh. Take a look at Figure 11.9. The first image is a simple model: a cylinder with two bones inside. The second image shows the weight of the bone at the top. Red means 100 percent influence, and dark blue means 0 percent influence; all the colors in between (orange, yellow, and green) indicate amounts of effect between 0 and 100 percent. The third image shows what happens to the model when you rotate one of the bones. The parts in red follow the bone's movement completely, and the parts with less influence average their movement among the bones that affect them.

In the following sections, you'll learn how to set up vertex weights in a mesh so that bones deform it as you want them to.

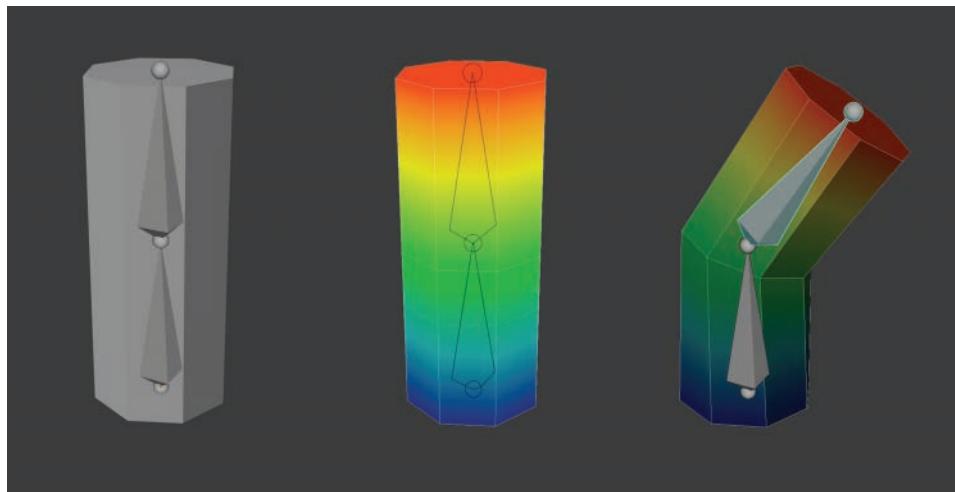


Figure 11.9 How weight affects meshes when you transform bones

Vertex Groups

Vertices can be added to a group and then define the amount of influence that the group has on each of the vertices. That's how vertex weights work for bones: Each bone generates a vertex group with the same name, and changing the influence of that group on a given vertex affects the deformation that the vertex will inherit from the bone.

Each vertex can belong to as many vertex groups as you want, so the same vertex can be influenced by multiple bones simultaneously.

You can find vertex groups in the Mesh Properties section of the Vertex Groups panel in the Properties Editor (see Figure 11.10). The menu will seem familiar to you at this point, as it's very similar to other menus, such as the Bone Groups panel.



Figure 11.10 Vertex Groups panel

This menu, however, has a slider with a value between 0 and 1. This slider defines the influence of the selected group on the vertices when you assign them to it.

Essentially, you follow this process to add vertices to a vertex group:

1. Select the vertex or vertices you want to add to the group.
2. In the Vertex Groups list, select the group to which you want to add those vertices.
3. Set the Weight slider to choose the influence amount you want that group to have on the selected vertices.
4. Click Assign to assign the weight from that vertex group to the selected vertices.

Vertex groups and weights are very useful for many purposes. In this case, they're used to determine the influence of bones on the mesh's deformation, but you'll find that many modifiers use a field to specify a vertex group, allowing you to make that modifier affect only the vertices in the selected group.

Caution

You can assign vertices to groups only in Edit Mode. But you can switch to Weight Paint interaction mode and "paint" those weights by selecting the vertex group you want to paint weights for and then paint on the model's surface to change the weights for that group.

Setting Up the Model for Skinning

You need to set up several things before you start defining vertex weights for the bones. In this section, you'll learn about deformer bones, as well as how to decide which objects need weights and which don't.

Deformer Bones

The rig has a lot of bones, and you need to tell Blender which ones will deform the meshes. By default, all the bones you create deform meshes. You should learn how this is done because you'll need the information later, but for now, the Rigify add-on should have taken care of the job. Only the bones that have to deform the mesh should do it; the other bones shouldn't.

On the Bone tab of the Properties Editor is the Deform panel, where you set up the envelopes and some features of the bone deformations. If you don't want a bone to deform the mesh, disable the Deform option. Disabling options for every bone would be very slow, however, so here are a few faster ways:

- In Edit or Pose mode, select all the bones you don't want to deform the mesh, press **Alt+W**, and click Deform to disable that option for the selected bones. Select the bones you want to deform the mesh, press **Shift+W**, and click Deform to enable that option for the selected bones.

Bone Settings with Shift+W and Alt+W

Pressing **Alt+W** is the same as pressing **Shift+W** to display the Bone Settings menu, so what's the difference? The **Alt** key is usually used to remove given effects in Blender. **Shift+W** switches the mode, so if you have the Deform option enabled, the option is disabled. Pressing **Alt+W**, on the other hand, always disables the option, so if you want to make sure that the option is disabled, press **Alt+W**.

- You can access the Deform option from the Bone Settings option on the 3D Viewport's header. You'll find that option in the Armature menu in Edit Mode and in the Pose menu in Pose Mode.
- Yet another way is to select all of the bones that you want to disable or enable the Deform option for. Make sure that you have an active bone by selecting it last. (If you select everything by pressing **A**, you may not have an active selection.) This way, the changes you make in the Properties Editor will affect only the active selection. But if you hold **Alt** while disabling or enabling the Deform option, all the selected bones that have that option in the same status will also change.

Bone groups and layers can help you with this process. If you have the bones organized, and all the deforming bones are in a separate layer or group, it will be easier to select all the deformers/nondeformers and change this option for all of them at the same time.

Although you shouldn't need to adjust the deformation status for bones after using Rigify, you'll need to know to do it when you create the facial rig.

Enabling Deforming Bones Only

At the moment, what you see on the Rigify rig are the controls, but those controls aren't the ones that deform the character's meshes. The bones you have to work with are the ones that deform the meshes, as explained in the previous section.

In a Rigify rig, a layer contains all the deformer bones. Go to the Armature Layers panel of the Properties Editor, and enable only the third layer from the end, as shown in Figure 11.11.

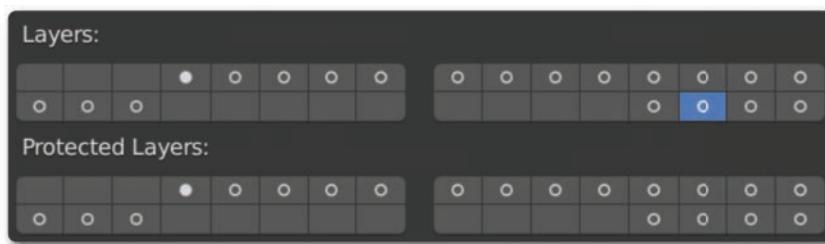


Figure 11.11 Armature Layers panel

The bones you see now are the ones that will deform your character. Each bone in the legs and arms is broken into two bones. This fact makes the skinning process a bit more difficult, but it will allow you to bend the rig in cartoonish ways later.

Knowing What Objects Don't Need Weights

That's right—the objects that aren't going to be deformed don't need to be weighted! You can parent those objects to a bone directly. As an example, the hair, cap, teeth, eyes, and tongue would need to be parented only to their respective bones, without an armature modifier or weights, as they don't need deformations. The tongue could be deformable, but in this exercise, to keep it simple, it stays put. The eyes keep their lattice modifiers, which are needed for distortion while they move, and they'll be children of their own bones, which you'll create when rigging the face.

To parent an object to a bone, make sure that you're in Pose Mode in the armature. Select the object, hold down **Shift**, and select the bone you want to serve as parent. Press **Ctrl+P** to do the parenting. Instead of selecting the Automatic Weights option, select the Bone option.

Caution

Parenting objects to bones directly requires selecting objects in multiple interaction modes (bones in Pose Mode and objects in Object Mode). To do that, make sure that the Lock

Object Modes option is disabled in the Edit menu. This option, if enabled, doesn't let you select objects that would require a change in the interaction mode. Meshes, for example, can't have a Pose interaction mode, so before selecting a mesh, you'd need to switch to Object Mode in the armature.

Start by parenting the objects that don't need weights to the deforming bones of the rig that are closer to them. Remember to follow this process:

1. Make sure that the armature is in Pose Mode. Otherwise, you won't be able to select a specific bone later, and if you do a parenting operation, the objects will follow the armature object itself, not a specific bone.
2. Select the object or objects that you want to make children of a bone.
3. Holding **Shift**, as you would to add elements to a selection, select the bone that you want to be the parent of the previously selected objects. This bone will become the active selection.
4. Press **Ctrl+P**, and select Bone in the menu that appears. This step makes the bone the parent of the previously selected objects.

You can use this method on the objects that make up the cap, hair, communicator, and cloth details on the arms, for example. The only objects that won't need deformations but that you can't set up for now are the eyes, corneas, teeth, and tongue. (Right now, the rig doesn't have the necessary bones to move those objects; you'll create them later.) You can hide those objects for now so that they don't get in the way.

Models

The following procedures may help you navigate your scene and make the skinning process easier:

- Apply Mirror, Solidify, and Shrinkwrap modifiers to objects that will need deformations so that those modifiers don't interfere when you weight the meshes. While I recommend it here, keep in mind that it's not always necessary to apply modifiers, and sometimes it's even recommended to keep them (when you think you may need to adjust the modifier's properties interactively after the skinning process is done).
- If you apply the Mirror modifier, some objects (such as the details of the arms) need to be separated so that each side of the object is a separate object, especially if these objects won't have deformations. In Edit Mode, select the faces on one of the sides and press **P** to separate the current selection into a different object. You can keep other objects, such as the gloves and boots, together. This way, whenever you add weights, they'll be mirrored, depending on their position relative to the bones on the other side of the rig. You can separate them later if you want to.
- Press **Ctrl+A** to apply the location, rotation, and scale of every mesh (except the lattices of the eyes, in case you scaled them when you made the deformations,

because they depend on that scale to cause the deformations). This action prevents bad things from happening when you link everything to the skeleton. If you scaled the objects, for example, you may have issues later when creating hierarchies between the bones and objects.

- When you apply the transforms, keep in mind that some things may go wrong. If an object gets darkened or looks weird, switch to Edit Mode, select everything, and press **Shift+N** to reset the normal. Also, the objects that use modifiers (such as Solidify) may have to be revised, as their thickness depends directly on their scale, so when you apply the scale, the thickness may be affected. This shouldn't happen if you've applied the modifiers before, but remember that this can happen if you prefer to keep those modifiers active.
- Name every object (if you haven't done so already) so that you can recognize objects by their names. This practice is useful at this stage and later in the process.

Adding an Armature Modifier

The Armature modifier tells the mesh to be deformed by the bones of an armature. As with constraints, you have two ways to add an Armature modifier to your mesh:

- Select the mesh you want to be deformed by the rig, go to the Modifiers tab of the Properties Editor, and add an Armature modifier. In the Armature field, type the name of the armature you want to use to deform the model. After that, the process of skinning is almost completely manual.
- Select the mesh, select a bone or armature while holding down **Shift**, and press **Ctrl+P** to parent. When you parent a mesh to an armature, Blender shows several options, some of which are in the Armature Deform group. Those options are Empty Groups (creates a vertex group for each bone, but you have to add weights manually), Envelope Weights (adds weights to the meshes, depending on the envelope settings for the bones), and Automatic Weights (creates vertex groups and adds weights based on the bones closest to the mesh). Automatic Weights usually is the best choice to start with, unless you've set up envelopes for the bones and such, but I don't cover that topic.

Caution

If you previously added a Subdivision Surface modifier to your models, when you add an Armature modifier, it works on top of the Subdivision Surface modifier, reducing performance and making weighting more difficult because it affects all the polygons generated by the Subdivision Surface modifier. Move the modifier up through the modifiers stack until it's before the Subdivision Surface modifier. (Subdivision Surface should be the last one at the bottom.) You should do this for every object.

A nice orderly way is to do this for one object, press **H** to hide it, and select the next one to repeat the process. As you hide the objects that you've already processed, only the unprocessed ones are visible. By the time all of them are hidden, you're done and can press **Alt+H** to unhide all the objects.

After adding the Armature Modifier with automatic weights and reordering the modifiers so that the subdivisions are at the bottom of the stack, Jim should deform when you try to move the rig, although it's possible that Blender won't do a very good job in some areas. You'll have to tweak weights manually to tell Blender exactly how it should deform the model.

Defining Weights

Before you start setting up the bone weights over the model, you may want to change the display of the armature to Stick (on the Viewport Display panel of the Armature tab) so that it doesn't block the view of the mesh and shows only the layer with the deform bones. It's unlikely that you disabled the In Front option to see the bones on top of the mesh, but if you did, enable it again to make the bone-selection process much easier.

Weight Painting

Weight painting is probably the fastest way to assign bone weights to your model. To enter Weight Paint Mode, go to the Interaction Mode selector on the 3D Viewport's header and select Weight Paint. You see the model in blue, yellow, green, and red; the colors represent the weights of the selected bone. You can also switch to this mode by pressing **Ctrl+Tab**.

Weight Paint Mode (see Figure 11.12) is similar to Texture Paint Mode. The painting tools are on the toolbar (left side of the 3D Viewport), and the options for the tools are in the usual menus (header, Sidebar, and Properties Editor).

When you add an Armature modifier to a mesh by using the Automatic Weights option, vertex groups are created in the mesh. Blender creates one group for each bone, and the groups have the same names as the bones so that you can recognize them. Each vertex group stores the vertex weights that define the influence of each bone of the armature.

Here's some information about working in Weight Paint Mode:

- On the left side of the 3D Viewport, you find the Active Tools that you can use while you work in Weight Paint Mode, such as the brush, blur, average, smear, and gradient.
- You can paint by left-clicking and dragging over the mesh's vertices. Using a pen tablet allows you to use pressure sensitivity while painting weights.
- You can set the size, strength, and weight of the brush. As in Texture Paint Mode, you can change the size and strength of the brush by pressing **F** and **Shift+F** in the 3D Viewport. You can also access such settings from the contextual menu when you right-click the 3D Viewport.
- On the Top Bar, you can access all the settings for the current tool.
- You can select which bone's influence you're painting by selecting the vertex group with the bone's name. You can find vertex groups on the Mesh tab of the Properties Editor.

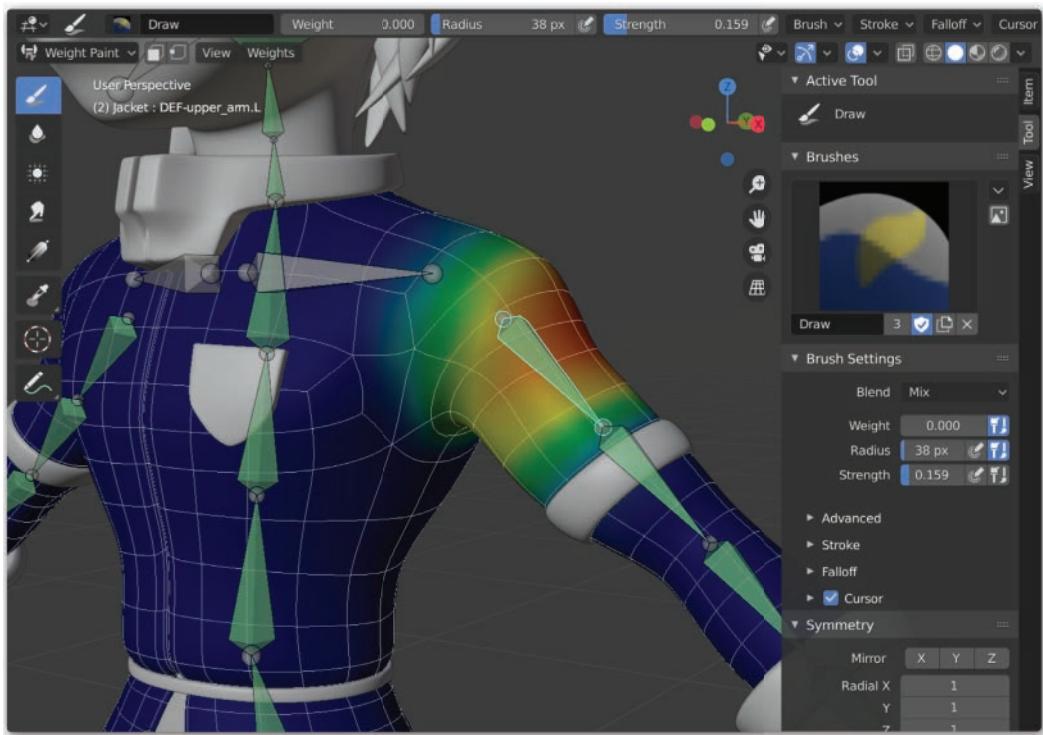


Figure 11.12 Weight Paint Mode's interface

Tip

Selecting the vertex groups from a list isn't the fastest method, though. Select the armature, and make sure that you're in Pose Mode. Now select the mesh. If you have the deformed bones visible in the armature, you can select the bones by pressing **Shift+LMB** while you paint weights. When you select a bone, you see its influence on the vertices, and its vertex group is selected.

Another alternative is to **Shift+RMB** on the mesh. A menu appears, displaying the vertex groups that affect that area.

- When you select bones to paint their weights with **Shift+LMB**, you have even more control. While weight painting, press **G**, **R**, or **S** to move, rotate, or scale the bone. This method allows you to move the character while you paint and test whether the weights behave as you expect. (From time to time, you may want to reset bones to their original positions. Press **A** to select all the bones in Pose Mode, or select the ones you want to reset; then press **Alt+G**, **Alt+R**, or **Alt+S** to reset their location, rotation, or scale.) Also, remember that pressing **Shift+LBM** will add bones to the selection, even in Weight Mode, so you may want to deselect all the selected bones from time to time by pressing **Alt+S**.

Keep in mind that this technique won't work with the deforming bones if you have a finished rig like the one generated by Rigify. The reason is that the movement of the deforming bones is linked to other bones that control the rig. In this case, you'd have to enable the visibility of those controls first.

You can even take this process further by creating animations (see Chapter 12, "Animating Your Character") that move the bones you're weighting so that you can see how they behave when you drag the cursor in the Timeline or press **Alt** while scrolling the mouse wheel.

- The Options button on the Tool Settings bar, on the 3D Viewport's header, enables some features that you may find useful. One of these features is the X-Mirror option. If your mesh is symmetrical on the X-axis, you're in luck! X-Mirror mirrors the weights you paint on one side to the other, on their correspondent bones. This option works properly only if your bones are correctly named with their left and right suffixes, of course (which is done by default in Rigify). Right before the Options button is an icon that resembles a butterfly with a dotted line in its center; this icon has three buttons next to it, one for each axis. These buttons do the same thing as the X-Mirror option on their respective axes.
- The Blur brush is handy. Paint basic weights first, and where you need them to be softer (as in articulation joints like the elbows and knees), use the Blur brush to paint and smooth the weight borders. Keep in mind that this brush blurs the weights of the vertices inside the brush size, so you may need to make it bigger to use it properly.

Tip

When painting weights, I recommend that you play with the options for the Weight Paints visualization in the Overlays menu of the 3D Viewport. You can also tweak the X-Ray option in the viewport shading options. Experiment with switching between Solid and Wireframe viewport shading modes; find the combination of options that gives you the best experience while weight painting.

I generally work with weight painting in Solid viewport shading mode, with the Show Wire option enabled and X-Ray turned on at a value of 1. I recommend that you play with the options until you find what works for you.

Tip

It's difficult to understand many of these options without using them. When you work on the character skinning to have the models be deformed by the rig, I recommend that you play with all the options and pose the rig to see how the options work and what they do.

Weight Values

Weight painting is generally easy but can be tricky. In complex areas of the model or in areas where you need specific and precise weights, it's better to add values numerically vertex by vertex.

In the Vertex Groups panel of the Mesh tab (the one with a little triangle and its vertices) of the Properties Editor are options that allow you to do this. Just follow these steps:

1. In Edit Mode, select the vertices that you want to weight with exact values.
2. Go to the Vertex Groups panel, find the group with the name of the bone to which you want to add the weight, and left-click to select it. If you click the little + button at the bottom of the list (you can find it on every list in Blender), you can filter and search groups by their name, so type the name and press **Enter** to display only the groups that have those letters in their name. (This is yet another case in which naming stuff is useful!)
3. Below the list are a few options, very similar to the ones in the Bone Groups panel. You can set the weight value and click Assign to set that weight to the active vertices for the selected bone/vertex group.

Another way to adjust vertex weights is to select one or more vertices in Edit Mode. If the vertices have weights assigned, a Vertex Weights panel appears in the Sidebar of the 3D Viewport (press **N** to display), on the Item tab. Tweak the weight values for each vertex group, copy weights from the active vertex, and paste those weights into the rest of the selected vertices (implying that whenever you make the vertex selection, you need to select one vertex last to turn it into the active selection).

Making Sure That Deformations Are Right

After you've weighted all the models that need deformations, you'll be able to pose Jim, check how the rig and model behave, and make adjustments if something needs to act differently. Here are some things to keep in mind as you're weighting your character (and afterward):

- The clothing details of the arms, for example, can be deformed by bones, but you can parent them to the arm bones. During design, the details were strategically placed because in those positions, only one bone would affect them at a time, so parenting them should be enough.
- You may need to experiment moving the controls of the Rigify rig to see what they do and how they deform the mesh objects.
- Don't pay too much attention to the head model, as you'll be rigging it later in this chapter.
- Try moving the rig controls in extreme ways to see whether the deformations work as expected, and if they don't, go back to the weight painting stage to adjust them.

Creating the Facial Rig

Only the facial rig is missing, which allows Jim to express himself and smile at you. For that job, you use shape keys. *Shape keys* are different statuses of the same model in which you can store different vertices' positions. One shape might be Jim's smile. Later, you'd be able to move the vertices from their neutral position to the smile position by sliding a bar from 0 to 100 percent, so you'd clearly see the transition. After you have those shapes modeled, you create a few new bones and learn how to use them to control those facial shape keys. In the end, you'll be able to control Jim's expressions with just a few bones.

But because you didn't use Rigify's facial rig, you'll have to create the rig for the eyes and jaw.

Rigging the Eyes

In this section, you create the rig for the eyes. You use a Track To constraint to control where the eyes are looking. Figure 11.13 shows the eyes rig. You create only the left side of the eyes rig, which you'll mirror later.

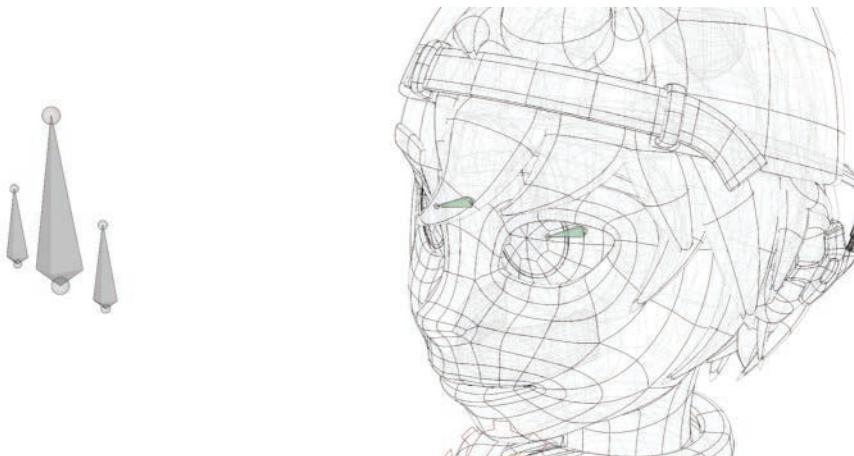


Figure 11.13 The eyes rig

You need to create the bone that moves the eye's model, which can be a little tricky in this case, as you have an eye that is being deformed by the Lattice modifier. In normal models, the eye would be a perfect sphere, so the bone would be in the center of the eye, but in this case, you have to align the bone to the original object, not the deformed one. Follow these steps:

1. Select the eye's object, and press **Shift+S** to move the 3D cursor to its center.
You see that the 3D cursor isn't centered in the eye. That doesn't matter, because

the eye will rotate from there before the deformation by the Lattice modifier takes place.

2. Go to Edit Mode in the armature, and create a bone in the 3D cursor's position. The bone will appear pointing upward. You can rotate it by using the 3D cursor as a pivot point so that it looks forward. You can call this bone D_eye.L. (The D stands for Deform, and the L at the end is for the left side of the rig.) Make this bone a child of the bone that deforms the head, which is in the armature layer you worked with during the skinning process. The bone should be called DEF-spine.006; that way, the eye bone will follow the head.
3. Select the tail of the bone you just created, and extrude it upward. With the new bone selected, press **Alt+P** and click Clear Parent to make this bone independent. You can move the bone forward so that it'll be the target that the eye bone will look at when it's set up. You can name it C_eye_target.L (C stands for Control, as this is a bone that you'll use to control where the eye is looking).
4. Duplicate the C_eye_target.L bone, and move it to 0 on the X-axis. You can also make it a bit bigger. The idea is that this bone will move both eye targets at the same time, but you can still move each eye target independently if you need to. This bone can be called C_look_target.

Tip

To make sure that a bone is in the center ($X=0$), you can insert the values manually into the Sidebar's Transform panel.

5. To add the Track To constraint and make the eye bone look at the target, select the target, add the eye bone to the selection by holding **Shift**, press **Ctrl+Shift+C**, and select the Track To constraint from the list. Now, if you move the eye target bone, you should see the eye bone adapt its direction to look at the target.
6. Make sure that the orientation of the eye bone is correct. On the Armature tab of the Properties Editor, enable the visualization of the bone axes in the viewport. Pay attention to the eye bone while you switch from Edit and Pose modes. If the bone changes its orientation while moving between modes, the constraint is rotating it, and you need it to stay in its original position. In Edit Mode, roll the bone by pressing **Ctrl+R** so that the orientation matches that of Pose Mode. Typically, you'll need to roll it by 90 or 180 degrees. It's important that the orientation is exact, so you should do it by typing the amount on your keyboard while you roll the bone.

When you're done, make sure that the eye bone has its Deform option enabled (it will be used to slightly deform the eyelids), but disable it from the target bones.

Mirroring the Eye Rig

You have one eye rigged, so now you must duplicate it on the other side.

Before mirroring, you need to know that later, you'll be able to copy poses and paste them mirrored. Blender knows how to do this by recognizing the names of the bones. During skinning, naming the bones also helps mirroring weights. Up to this point, Rigify did all this work, but you must know how mirroring works so that you can add parts of the rig on your own. Each bone has a name and a suffix that tell Blender whether it is on the left or right side. Here are some examples:

- **D_eye.R:** The .R suffix tells Blender that this bone is on the right side.
- **D_eye.L:** The .L suffix tells Blender that this bone is on the left side.
- **C_look_target:** When the name has no suffix, Blender knows that the bone is in the center.

This naming convention enables Blender to translate the pose from one bone to the same bone on the other side of the rig. When you're painting the weights for the skinning, you can mirror those weights to the other side as well.

Naming Bones Automatically

Blender has tools that add suffixes to the bones' names automatically. If you have many bones, you can name only the ones on one side and then automate the naming of the other side, or add suffixes in bulk. Select all the bones in Edit or Pose mode by pressing **A**, and from the Armature or Pose menu (depending on the interaction mode you're working in) on the 3D Viewport's header, choose the option AutoName Left/Right within the Names submenu. (You can also find this option in the contextual menu.) This option detects the bones that are on the positive and negative sides of the X-axis and names them accordingly, which is why it's important to center your character on the X-axis.

Then go to the bones that are in the center, and check their names. Sometimes, the bones are not exactly at $X=0$, so they get suffixes as well. For these bones, check their names, and delete the suffixes.

Tip

Remember that you can rename the active object very quickly by pressing **F2**. Another interesting tool renames objects in bulk, allowing you to replace parts of names and add prefixes and suffixes quickly. You can access the bulk-renaming tool by pressing **Ctrl+F2**. A menu appears with all the options available for renaming the selected objects.

Mirroring Bones

Now the bones have names, and if you worked on the left side of the eyes rig, all the bones (except C_look_target, which is in the center) have an .L suffix in their names. Follow these steps to mirror those bones:

1. In Edit Mode, select the bones for the eye and eye target.
2. Press **Shift+D** to duplicate them, and right-click to cancel the movement.
3. Place the 3D cursor on the center of the scene by pressing **Shift+C**, and switch the pivot point to the 3D cursor by pressing **.** (period).
4. Press **Ctrl+M** to mirror the selected bones, and press **X** to mirror them in the X-axis. Press **Enter** to accept.
5. The bones are mirrored, but they have names like D_eye.L.001. (Don't worry. This is what Blender does when duplicating an object; it automatically renames the object by adding a number to it so that you don't have two objects with the same name.) With the mirrored bones selected, use the Flip Names option as explained in the previous section. This option converts those duplicated names from the left side to appear on the right side, such as D_eye.L.001 to D_eye.R.001.
6. Right after using the Flip Names option (which by default switches the suffixes from one side to the other, but doesn't change anything else, such as the numbers after the suffix), you can go to the Adjust Last Action menu and enable the option for stripping numbers to turn D_eye.R.001 into D_eye.R, for example.

Possible Side Effects of Mirroring Bones

Mirroring bones can save you tons of work, adding constraints and repetitive tasks for both sides of the rig, but unfortunately, it may have some side effects.

When you mirror bones, some of them may get rotated in a weird way as an effect of inverting them in the mirrored axis, and you'll have to fix these rotations manually. Even though that's *not* very cool, it's usually less work than creating both sides manually!

You can adjust those rotations if necessary by enabling the visualization of their axes and rolling the bones by pressing **Ctrl+R** in Edit Mode. Sometimes, you also need to adjust some values in constraints to fix such issues.

Rigging the Jaw

Jim will need to open and close his jaw, so you must create a bone that goes from the head bone to the chin that will be able to rotate Jim's jaw. You can see the positioning of this bone in Figure 11.14.



Figure 11.14 Position of the jaw bone inside Jim's head

After you create this bone, make it a child of the bone that deforms the head, as you did with the eyes; that bone is named DEF-spine.006. Make sure that the jaw bone's Deform option is enabled.

Skinning the Eyes and the Jaw

Now that you've added bones for the eyes and the jaw, you should perform the skinning so that they affect the 3D mesh.

There is a difference between these new bones and the previous ones. Before, the vertex groups were added automatically when you created the Armature modifier, but now you can't do that. How do you add new bones to affect the deformation? The process is simple: Create new vertex groups named for the bones they represent.

When you have vertex groups named after the bones, you can paint weights on those groups, and they'll define the influence that the bone with the same name has over the vertices. Blender knows which vertex group should be affected by each bone because the vertex group's name and the bone's name are the same.

Note

Something interesting to know about how the vertex groups and bones work is that they're smart. When you create a vertex group with the same name as a specific bone so that you can add vertex weights, the vertex group and the bone will be linked, in a way. If you change the name of the bone later, the vertex group will reflect the name change so that the weights for that bone aren't lost and keep working.

The eyes aren't deformable, as mentioned earlier; they just follow the eye bones as their children. The eyelids, however, could use some deformation to make the eye movements more organic. (When the eyes move, the eyelids are slightly dragged with them.)

To achieve this effect, select the head mesh, enter Edit Mode, select the vertices around the eyes, and add a bit of weight for the eye bones vertex groups. You can use any of the techniques you've learned so far to add the weight, such as weight painting or vertex groups.

The jaw is a little trickier. The mesh in the jaw area in Jim's head is probably affected by other bones, so you'll have to add weight from the jaw's bone, and also remove weight from other bones that may counteract the jaw's deformation.

Tip

An important part of the skinning process is testing, to make sure that the weights you've assigned work well. While you paint weights, you should stop from time to time to move parts of the rig around the area that you're working on and check for unwanted deformations.

It's quite common for automatic weights to affect parts of the rig that they shouldn't affect, even if faintly. Sometimes, you won't even notice unless you test the rig with exaggerated movements.

In the jaw area, for example, if you turn the head up or to the side 180 degrees (a movement that you'll very rarely perform with the rig), you may be able to detect whether the jaw follows that rotation or gets distorted by the effects of other bones.

The jaw area is also tricky because it involves the mouth, which can be difficult to access for weight painting because the mouth in the model is closed. A technique I recommend for such occasions is the following (see Figure 11.15):

1. Go into Edit Mode, select the unwanted parts (the top of the mouth, for example), and hide them by pressing **H** so that they're not in the way.
2. Select the area that you want the jaw bone to affect, and add the weight on the Vertex Groups menu.
3. Exit Edit Mode, select the jaw bone in Pose Mode, and use it to open the mouth. The chin will probably move as expected, but the areas around the mouth and cheeks may not be deforming very well, as the weight is set up only on the parts that have full effect.
4. In Weight Paint mode, using the blur brush and a big size, perform strokes on the areas with bad deformations to make the transitions between the most affected areas and areas that are unaffected, easing the deformation in those areas. You'll also see the effect that the bone has when the mouth opens. Use the weight paint tools to make the needed adjustments.
5. Select the jaw bone and press **Alt+R** to reset its rotation and close the mouth.

Sometimes, it's easier to work with weights after moving or rotating some bones to take some areas of the model out of the way. In this case, you open the mouth to access the lips and inner-mouth parts that would otherwise be inaccessible.

Also, remember to use parenting to make the top teeth follow the head bone and the bottom teeth and tongue follow the jaw bone.

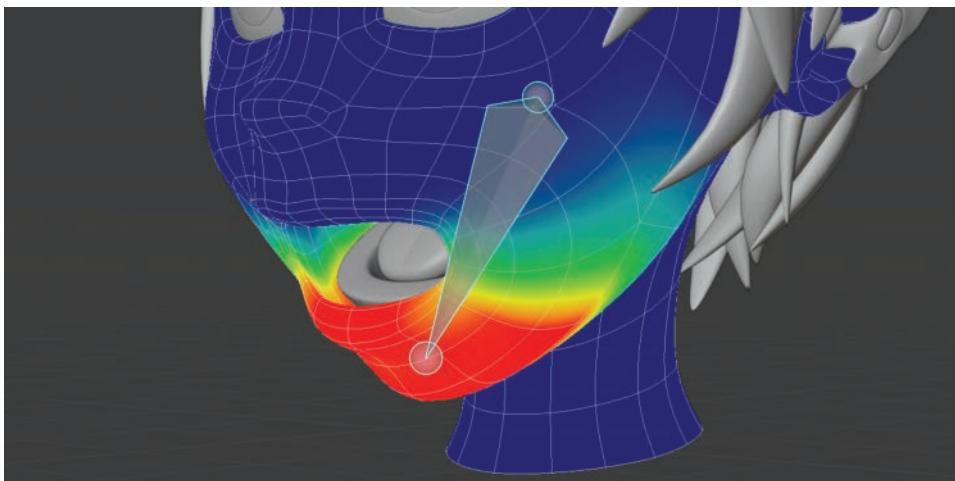


Figure 11.15 Working with the open mouth can help with weight-painting the jaw bone's vertex influences.

Chest Badge Deformation

There is something interesting about the chest badge: When deforming, it may intersect with the jacket model, because even if you do the skinning right, the badge and jacket are not in the same spots and will deform at different points.

In the case of the badge, you have a very easy solution: a flat object over the surface of the jacket. You can make an object deform following the deformation of another thanks to the Mesh Deform and Surface Deform modifiers. These modifiers link the geometry of a mesh to another, similar to the way in which automatic weights are added between a geometry and the bones that deform it, so that when the vertices of the deforming mesh are moved in space, they deform the object that has the modifier assigned.

Use Mesh Deform if you want a mesh with a closed volume to deform your object. Surface Deform, on the other hand, is suited for deforming objects on a surface, not necessarily closed. That option is what you need in this case. Follow these steps:

1. Select the badge. If you used an Armature modifier, disable it so that you don't have overlapping deformations from the Armature modifier and the Surface Deform.
2. Add a Surface Deform modifier to the badge, and place it on top of the Subdivision Surface modifier so that it has fewer vertices to deform.
3. In the Surface Deform modifier, in the field to select the deforming object, select the jacket's model. Then click Bind to link the vertices from the badge to the surface geometry of the jacket.

Now if you move the rig, you should see the badge follow the surface of the jacket.

Caution

After you bind the vertices of a mesh to another by using the Surface Deform or Mesh Deform modifier, remember that any change in the deforming and deformed meshes will break the binding. If you must perform adjustments in any of the geometries involved, you'll need to go back to the modifiers and bind them again.

Modeling Shape Keys

The first step in building Jim's facial rig is modeling the shape keys. You need to isolate different parts of the face and create different shape keys for each of their expressions. Here are a few examples of the shape keys you can create: smile, blink, frown, mouth open, eyebrows up, and eyebrows down. You must create those deformations for specific parts of the face separately so that you can control them individually.

For each of those expressions (except the ones that affect both parts of the model), you have to create two shape keys, one for each side of the face. You need the left and right eyelids to blink, for example. For Jim, things are pretty simple, as you don't need him to make any dramatic facial movements, only basic movements such as smiling and blinking. So can keep the shape keys to a minimum while learning how they work. If you want to create a really nice facial rig, you need a lot of shapes and even bones to aid the facial deformations and make them more accurate. Figure 11.16 shows the Shape Keys panel on the Mesh tab of the Properties Editor. You can create shape keys within the model itself, using the Shape Keys panel, or create them in separate models and merge them into the original model as shape keys.

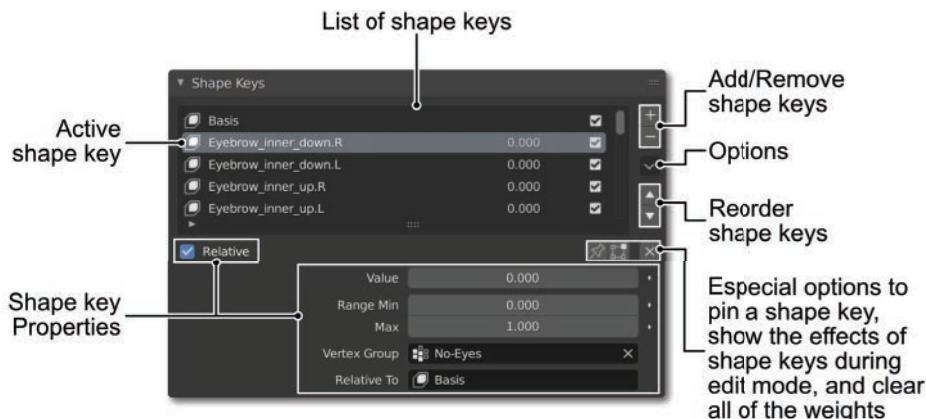


Figure 11.16 Shape Keys panel on the Mesh tab of the Properties Editor, where you can add, remove, and tweak shape keys

Creating Shape Keys in the Original Model

Here is how shape keys work. You start with the one called Basis. Basis is the base shape key, on which the rest build up the final deformation; Basis is the model itself.

Click the + button to add a shape key. Double-click a shape key to rename it. Suppose that you’re going to create the smile shape. Name it something similar to Mouth_smile.L (the left side of the mouth that will be smiling). With that shape selected, if you switch to Edit Mode, you can adjust the model to make it smile on the left side of the mouth. Proportional editing is useful at this stage. When you exit Edit Mode, the model jumps back to its original shape, and you don’t see the smile because the smile shape is at 0 percent effect at the moment. Near the shape’s name is a number that you can drag left and right to increase or decrease its value. Also, you can select the shape and drag the Value slider below the list.

The shape key’s value can extrapolate the position farther than the original shape key. If you set the value to 2, for example, the shape key can continue the movement of the vertices until they double the effect that you originally modeled.

Tip

While you’re in Edit Mode, selecting a shape key in the list shows you the deformations of that key. You can even press the up and down arrows on your keyboard to navigate through the shape keys list and see how they affect the model very quickly.

Creating Shape Keys from Separate Models

You can duplicate the original model as many times as the number of shape keys you need, give those objects the name of the shape keys, and deform the vertices as you desire for each shape key.

When you have all of the models ready, select all the models, add the original model to the selection, click the arrow below the buttons on the Shape Keys panel to reorder shape keys, and choose options from the Join Shapes menu. If you choose Join Shapes, all the selected models become shape keys of the original object as though you had modeled them there directly. The names of the generated shape keys will be the same as the object’s name.

What is the advantage of this technique? It allows you to see all the shape keys at the same time and compare them, which can be helpful. In Figure 11.17, you see the set of shape keys created for Jim and the names used for them. (Remember that only shape keys for the left side were created.)

Caution

When you create shape keys, you must be sure that the original mesh won’t have its geometry changed. If it does, there will be problems, as shape keys need the geometry to be the same. Shape keys store different positions for the vertices, so if you change the number of vertices, they will break.

The goal, regardless of the method you use, is to have different shape keys for each part of the face and mix them to create facial expressions. Happiness, for example, can be expressed by a smile, with the lips a little open, the eyes semiclosed, cheeks lifted, and eyebrows lifted. The mix of those shape keys helps you create the happiness

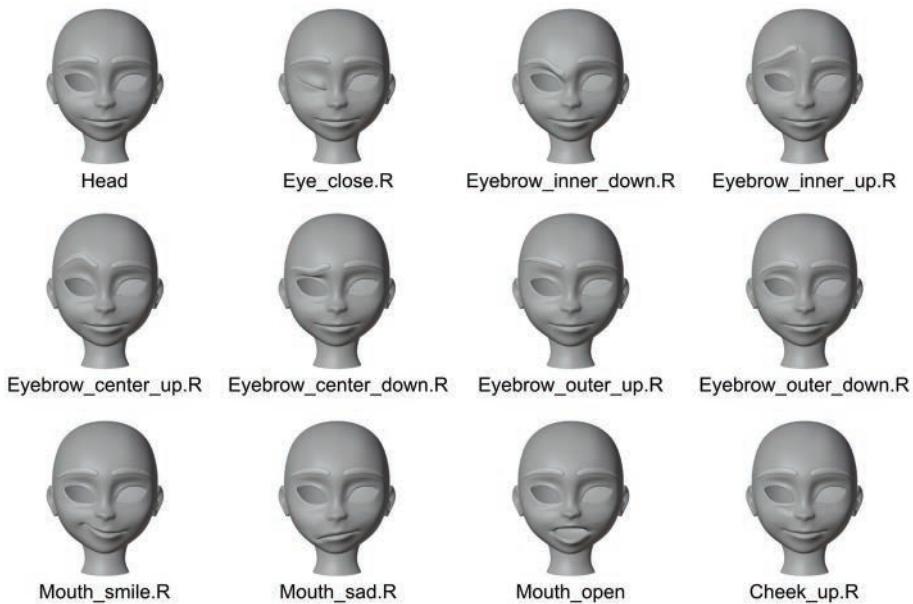


Figure 11.17 Facial shape keys that were modeled to create Jim's expressions

expression. Figure 11.18 shows different expressions on Jim's face created by mixing some of the basic shape keys. Shape keys exist on both sides of the face, as you'll see in the next section.



Figure 11.18 Jim's shape keys in action. Isolated parts of the face are moving together to create full facial expressions.

Mirroring Shapes

You need some shape keys to isolate one side of the face from the other, such as a right-eye blink and a left-eye blink. If your character's face is symmetrical, modeling the shapes for each side is tedious, and the shapes may be different. To mirror shapes, follow these steps:

1. Set the value of the shape key that you want to mirror to 1.0.
2. Click the button with a down triangle down near the + and - buttons to add shapes to and remove shapes from the list. Select the New Shape from Mix option to create a new shape with the vertex positions, including the values of current shapes. Rename the resulting shape for the other side.
3. Click the same button again, and select the option Mirror Shape to translate the vertex positions to the other side of the mesh.

Now you can turn the original shape key's value back to 0, and you'll have the new shape key that will have the same effect on the other side. Make sure to rename the newly created shape key to avoid automatic naming, which that can be unintuitive. Additionally, you can use the arrows at the right of the list to reorder the shape keys in a way that may be more comfortable for you.

Creating the Face Controls

You have many ways to create the facial rig: using bones to deform the face parts, using Lattice or Mesh Deform modifiers to deform the face, and so on. In this section, you create some bones to act as controllers that let you manipulate expressions from the 3D Viewport. This way, you won't have to tweak the shape keys in the Shapes panel one by one.

Depending on how you want to control your character expressions, you need different types of controls. In this section, you add a few bones in front of the face, each of which will control a few shape keys (facial expressions).

The bone for the mouth is set up so that when it's scaled up, the mouth opens. The bones for the eyebrows control the lifting of the eyebrows when you change their vertical positions. You have to think of these controls as intuitive (later, custom shapes make them even more intuitive), so they should make sense. If you move one of the controls up, something in the face should move up; having a different effect would be counterintuitive. In Figure 11.19, you see the bones that control Jim's basic expressions. A more complex model with more shapes (to make Jim talk, for example) would need more controls.

Parent those bones to the bone that deforms the head (the one called DEF-spine.006) and name them with names that define the parts that they will deform, such as C_eyebrow_outer.L, C_cheek.L, and C_mouth.

You can use the techniques you've explored up to this point to create the bones on one side and mirror them to the other side.



Figure 11.19 These bones are responsible for controlling Jim's facial expressions.

Make sure that you disable the Deform option for all these bones, as they shouldn't have any effect on the vertex weights. It shouldn't be important at this point, as the weight painting has already been done, so any new bones won't change the weights. (When you use automatic weights, they're created when you add the Armature modifier, but they don't update after that.) You may need to make some adjustments to the weights later, however, and it's better not to leave loose ends that could create errors.

Using Drivers to Control the Face Shapes

Now you need to tell Blender how the bones just created for the facial controls and their properties will control the shape keys' values. For that purpose, you can use drivers. This topic is quite technical and advanced, so you only get your feet wet in this section, but I encourage you to look into the topic and create more complex things with drivers.

Divide your interface, and enable one of the areas to have a 3D Viewport with your character and a Drivers editor in the other area. In the Drivers editor, you will customize the link and effect between two properties; in this case, you will make Blender change the value of shape keys when a bone is moved in a given axis.

Creating Drivers

At this point, the Drivers interface is empty, as you have no drivers in your scene. Creating drivers is easy: you just need to right-click on a menu property and click on Add Driver (you can also create it directly from the Drivers editor, but it requires more work). A menu will show, in which the property you clicked on will already be set up, and you can add another object's property to drive it.

Figure 11.20 shows the driver setup menu, using one of the shape keys and face control bones as an example (you can use the options shown as a reference).

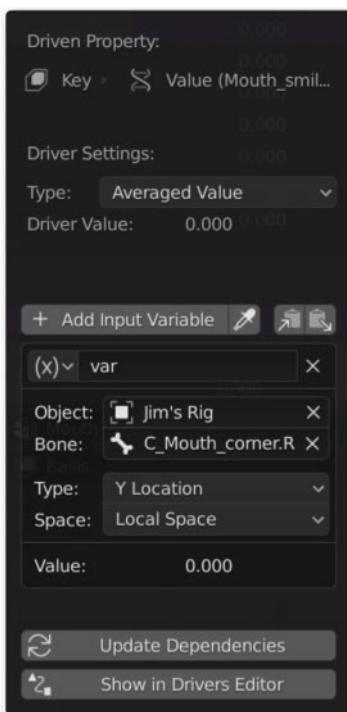


Figure 11.20 Driver setup menu, showing the options used to make a bone change the value of a shape key

Configuring Drivers

In this section, you learn how to set up the driver to make the left corner of the mouth smile. Then you use the same method for the rest of the shapes and control bones. You can do this setup in both the menu that pops up when you click Driver after right-clicking a property and the Drivers editor. (You'll find the menu for setting the driver up in the Sidebar by pressing **N** and clicking the Drivers tab, as shown in Figure 11.21.) Follow these steps:

1. First, you need to create the driver, so add one to the property you want to control. Choose the shape key that makes the left side of the mouth smile (in this case, Mouth_smile.L), and add a driver by right-clicking the value slider that controls how much of that shape key is being mixed with the original state of the mesh.
2. For the driver's type, choose Averaged Value.
3. Below the type, you'll see the variables. For a simple driver, you need only one, but you could create more variables for more advanced drivers. The first variable is created but empty: you'll have to choose Jim's rig as the object.

4. When you choose the rig as the object, a new field appears below it to allow you to select the bone. Having your bones properly named is a big help, as you'll have to select them from a list. In this case, the bone that controls the shape key is called C_mouth_corner.L.
5. After the bone is selected, define the transform property that will control the shape key's value. In this case, you want the shape key to increase its value when the bone moves up. In the case of a bone, you should check its axes to see where they're pointing to select the right one. You could also try different axes until it works, but that's not that efficient. In this case, the bone is pointing up, and the right axis is Y; because you want the bone to act as you move it, you should pick the Y location (instead of orientation or scale).
6. In the Space selector, pick Local Space. World Space has its origin in the center of the scene, but you want the point with zero effect in the original bone position. When the bone is at its rest position (the same position it has in Edit Mode), the value will be zero, but it will increase or decrease as you move it.
7. Click Update Dependencies to update the effect of the driver. You should see it appear in the Drivers editor. If you try to move the bone, and it doesn't move the value of the shape key, or moves too slowly or too fast, don't worry; you'll learn the last part of the driver's configuration next.

Before continuing, you must learn a couple of things about the Drivers editor (see Figure 11.21):

- On the left side is a list of the properties that have drivers assigned. By default, you should see only the properties with drivers within the current selection. You can see all the drivers in the scene or only the ones in the current selection by clicking the button with a cursor icon on the header.
- In the list of properties with drivers, you can select the one you want to set up and adjust. If you click the eye icon to the left of its name, you can show or hide that property's curve in the editor. When you have many drivers, it can be useful to hide the ones you're not working with so that you don't select undesired points and move them accidentally.
- In the center of the editor, you see a diagonal line (shown only after you create at least one driver). This diagonal line represents the effect the driver will have on the driven property, and in which amount. The X (horizontal) axis is the value of the driver property, and the Y (vertical) axis shows the value of the driven property.
- The default curve is a diagonal line that sets the driven property at 0 when the driver property's value is also 0, but when the driver is at 1, so is the driven property. This curve or line has two points that you can move around to change how the values of the driver and driven properties are related. If you make the line more vertical, for example, the effect will happen faster, and if you make it more horizontal, the effect will happen slower.

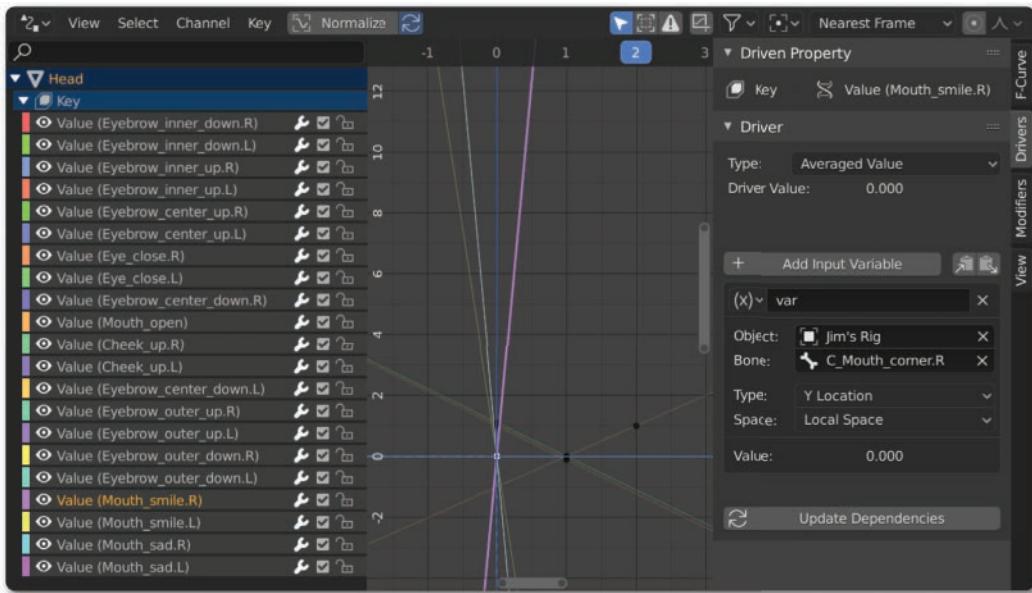


Figure 11.21 Drivers editor, where you can see and modify driver settings

- You can invert the line to point down instead of up if you want the effect to act in the opposite way.
- If you need to see a bigger range of values in the editor, you can use the general controls for navigation: **MMB** to pan and **Ctrl+MMB** and drag to zoom in or out (or use the scroll wheel).

With this basic knowledge, try to move the points in the curve so that the shape keys react as you expect them to when you move the control bones.

Tip

When you have some drivers set up, you'll probably find that you have to create new ones for other objects, bones, shape keys, and so on. Right-click a property with a driver, and choose **Copy Driver** from the contextual menu. Right-click the property to which you want to add the new driver, and choose **Paste Driver** from the contextual menu. Now you only need to go to the new driver and tweak its parameters to set it up for the new property.

Repeat the process for the rest of the drivers so that they act as you expect them to. With the eyes and mouth, remember to use scale instead of location to control how they open and close.

Also, keep in mind that the curve is very important. Depending on the effect that you want, you may have to invert it or move it forward. The position of the curve

determines how much effect on the driven property is taken from the specific value in the driver property, so if you want it to go backward, down, up, faster, or slower, you need to adjust the curve points.

Organizing the Facial Rig

The facial rig is ready and does what it's supposed to, but to make working with it more comfortable, you should organize it to hide it and show it as necessary. You could send it to a specific armature layer and show and hide it from there, or you can and use Rigify's interface to add buttons for the facial bones and the eye targets.

Sending Bones to the Proper Layers

Start by organizing the facial rig on the appropriate armature layers. Moving bones to armature layers is very easy: Select them in Edit or Pose mode, press **M**, and click the desired layers. (Each square in the menu that appears when you press **M** is an armature layer.)

In Rigify, the first three layers (the first three squares on the top row) are for the facial rig, but because you eliminated the face bones before generating the final rig, those layers are empty.

It would be useful to separate the facial controls within two layers: one for the jaw and face deformations control and another for the eye controls.

Select all the bones for controlling the face deformations and the jaw, press **M**, and click the first square of the top layer. Select the three controllers for the eyes (eye targets and look target), press **M**, and click the second square of the top layer.

One last thing: The eye bones could be sent to the same layer on which all the deformer bones for the rig live. Select both eye bones, press **M**, and click the third square from the end on the bottom row.

Now the bones are correctly organized!

Adjusting the Rigify Script to Add Custom Buttons

The facial bones are already in their respective armature layers, but it would be nice to have them accessible from the Rig Layers in the Sidebar of the 3D Viewport, wouldn't it? Figure 11.22 shows the resulting menu.

To do this, you'll have to adjust the rig's script, but don't be scared: You don't need to know programming. The process is easy, I promise:

1. Open an area in the interface with a Text Editor.
2. In the Text Editor's header, choose the Rigify script from the datablock selector, where you can create a new script or load an existing one. In this case, only the Rigify script (`rig_ui`) should exist, which is the one you should load. This script contains the instructions that make Blender generate the buttons and sliders for controlling the rig. The script is run when you open the file or generate the rig.

3. In the script, scroll up from the end until you find a set of lines similar to the following (also see Figure 11.22):

```
row = col.row()
row.prop(context.active_object.data, 'layers', index=3, toggle=True,
text='Torso')
```

These two lines define the buttons for the rig layers, the text in the buttons, and the armature layers that they show or hide when clicked.

4. Select and duplicate those two lines twice at the top of the set. For the first one, change the text in the second line to this:

```
row.prop(context.active_object.data, 'layers', index=0, toggle=True,
text='Face')
```

This new text generates a button named Face that will enable and disable the first layer in the armature layers. (The index is 0, not 1, because layer numbering starts at 0, so 0 is the first layer.)

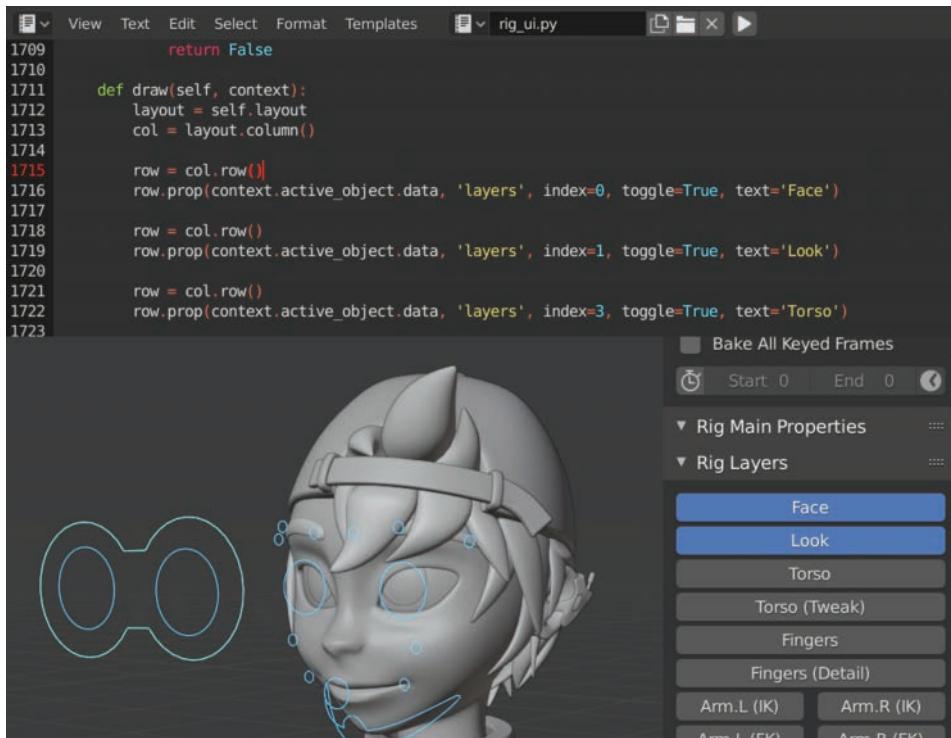


Figure 11.22 The script that you can modify to generate buttons that control the rig layers from the interface

5. For the other line, do the same, but this time change the text to Look and the index to 1. This step creates a button that enables and disables the second armature layer, for the eye controls.
6. You still don't see those buttons in the menu because you need to run the script for those new lines to be executed and generate the buttons. To run the script, press the Play button in the Text Editor's header, or press **Alt+P** while working in the Text Editor.

Now the buttons should appear, and if you click them, you should see their effects. Working is much more comfortable now, right?

Creating Custom Shapes

The rig is finished, but it's not looking very good. To make it better-looking, simpler, and easier to use, add custom shapes to the bones (see Figure 11.23). You can use any model or curve as a custom shape for the bones.

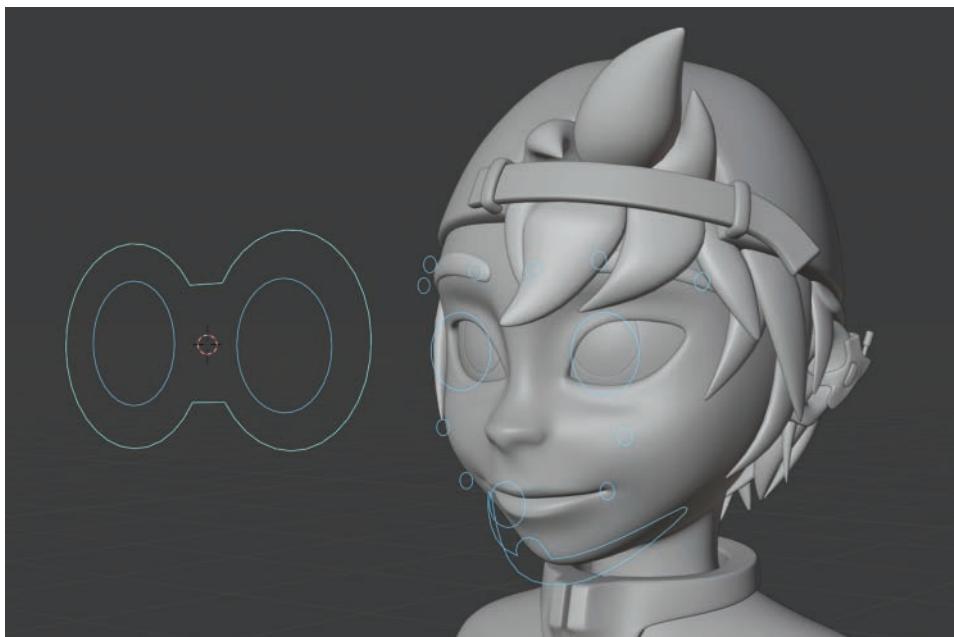


Figure 11.23 Using custom shapes for Jim's rig

Select a bone in Pose Mode. On the Bone tab of the Properties Editor, the Viewport Display panel has a Custom Shape subpanel. In that field, type the name of the shape (any object) you want to use instead of the bone. This shape is aesthetic; it has no influence on the functionality of the bone. In the custom shape's menu, you can find options to enable wireframes, showing only the edges of the selected as a custom shape, and a slider to control the displayed shape's scale.

Following is the usual procedure for using custom shapes:

1. Create a shape such as a plane or a circle, and name it. (An S_ prefix will help you recognize a shape when you search for it in a list.)
2. Select the bone, and choose the new object as a custom shape in the previous menu.
3. Go back to the object; enter Edit Mode; and tweak, scale, and rotate the shape until you're happy with how it shows the bone. You can also use the scale slider in the Custom Shapes menu to change the size of the displayed shape.
4. Repeat the process 1 for every controller bone.

You don't need to create a new object for each bone; you can use the same shape for many bones. For Jim's face, I created a circle that is used by most bones except the jaw and the look target. The jaw controller's shape slightly follows the jaw form to make it more intuitive and to differentiate the jaw bone from the other facial controls. The look target has a shape encapsulating both eyes' individual controls, to show the naked eye that the look target "contains" the individual targets.

Making Final Retouches

You can still do plenty of things to improve your rig, but this one is pretty complete. One thing you can do to make it much more comfortable to use is to lock the transforms that you shouldn't be using for bones. This action has two functions: preventing accidental transforms (such as scaling a bone that shouldn't be scaled or rotating a bone that is only meant to be translated) and helping you see what you can do with each bone.

Take the face controls as an example. They're supposed to be moved only on a single axis or scaled (but not moved or rotated). The 3D Viewport's Sidebar has a lock icon below the Transform panel, next to the axis. If you click that lock icon, you won't be able to transform the bone on that axis, which even disappears from the manipulators.

You see Jim posing for you in Figure 11.24.



Figure 11.24 Jim's final rig. Now you can pose him and make him do cool stuff!

Reusing Your Character in Different Scenes

In this last section, I talk about linking, appending, and other interesting ways to reuse your character in other scenes or files.

Library Linking

Library Linking is the process by which you link objects from one scene to another. Why would you want to do that? Suppose that you're working on a film and several animators are working on different shots at the same time. The same character needs to be in more than one scene, maybe interacting with other characters. You need to prepare your model so that it can be easily loaded into other scenes.

Linking and appending are the two ways in which you can bring objects from one .blend file into another.

Linking

Linking an object from one scene to another is easy. Choose File > Link, navigate to another .blend file on your hard drive, and access its contents. You can select anything from the file contents: objects, meshes, lamps, materials, node trees, groups, scenes, and so on. When you've selected one or multiple elements, click Link. Those elements are brought into your current scene.

Linking brings the contents from a different .blend file into your current .blend file, but it keeps a link to the original. You won't be able to modify those contents in the current file; you can only place them in the scene. To modify them, you have to go to the original .blend file and perform the desired changes. When you save the files and load the new one again, the changes are reflected in every scene in which that original object has been linked.

Suppose you have a character and are working on some shots that the character is involved in. You have some animations done, and you suddenly decide that you want to change the character's hair color. Making this change file by file would be tedious and problematic. (The change is just a color change, but imagine doing something more complex.)

This process is super-easy when you use links. Change the character in the original file, and when you save the file, the rest of the files that have this linked character will reflect those changes automatically.

Appending

Appending is similar to linking but doesn't keep the link with the original contents. It creates a new copy of the elements in the current scene. You have complete access to this copy and can modify it as you want.

Linking and appending have advantages and disadvantages. Depending on the project and the specific case, you may be more interested in using one technique or the other. Choose wisely.

You can also find the Append option on the File menu below Link.

Working with Collections

Whether you're working with links or appending, I encourage you to use collections instead of independent objects. Here are a couple of reasons why working with collections is better:

- When you work with a collection, all objects in the collection are linked or appended. If you use objects instead, you have to select all of them independently, which makes it tricky to work with complex models made up of many objects. With a group, you have to select just one!
- If you work with objects and add a new part to your character (a hat, a watch, new shoes, and so on), you have to go through all the scenes in which the

character is linked and link the new object. With collections, however, you just assign the object to the character's collection and save the file, and you're done. The contents of the character's collection will be updated in all the files in which that collection is linked.

Follow these steps if you want to turn your character into a collection:

1. Select all the objects, the armature, and anything else that's part of the character. (Remember to unhide hidden objects and layers.)
2. Press **M** to create a new collection from the selected objects.
3. Select Create a New Collection from the menu, and type a name for your new collection.

If you created a collection before, you can add the selected objects to an existing collection.

You can also create collections from the Outliner, clicking the button with a box and a + sign. In the Outliner, you can organize collections, double-click them to rename them, or even drag and drop objects between existing collections.

That's it! The next time you want to link or append a collection, when you navigate to the .blend file, go into the Collections folder inside the .blend file's contents. The character's collection will be there waiting for you.

What if you want to add objects to or subtract objects from your collection? Access the Object Properties tab of the Properties Editor. On the Collections panel of this tab, you can define the collections to which the active object belongs. You can remove the selected object from a given collection in the list by clicking the X button, or you can add the object to an existing collection.

Alternatively, to remove an object from a collection, select it, press **Ctrl+Alt+G**, and select the collection from which you want to remove it.

Caution

Every object in a .blend file has to belong to at least one collection. By default, a collection properly named Collection will be in a .blend file when you create it, and objects you create will belong to it.

If you remove an object from all collections, it will become an orphan and can't be represented in the scene.

Protecting Layers

In the Armature options of a skeleton in the Properties Editor, you see a Protected Layers section. In this section, you can mark some layers as selected to prevent modification of those layers when you link the character in another file. This option doesn't lock the bones so they can't be modified, but they're reset to their original states when the file reloads.

You can use this option to protect layers such as the helper bones and the deform bones, which you don't want the animator (or yourself) to mess up accidentally. This feature is a layer of security for your rig.

Using Proxies to Animate a Linked Character

When you link a character to a different scene, something interesting happens: You can't modify linked objects except in the original scene from which they're linked. Armatures are exceptions, precisely to enable you to keep your character linked while you pose the rig and create animations.

Armatures can be modified when they're linked in a scene thanks to proxies. A *proxy* is a copy of the rig that allows you to pose it but not modify it in any other way. You can't enter Edit Mode and modify the shape or hierarchy of the bones, for example.

To create a proxy, select the linked character, choose Object > Relations > Make Proxy, and select the armature from the list of objects that pops up. Then you'll be able to pose the copy of the rig, and the rest of the objects will follow.

This proxy will have only Pose Mode information. If you change bones or features for the rig in the original file, they update in the proxies as well.

Summary

Surely this chapter showed you that rigging is the most technical, complex part of the character-creation process. It takes time, and it can be frustrating, such as when the rig doesn't behave as you expect. It's like a puzzle!

You also learned some quick automatic ways to create rigs, such as the Rigify add-on, which generates a very cool rig automatically based on the proportions you set for your character and that may need only a few adjustments to be fully functional.

Still, you should be able to create at least basic rigs on your own. Sometimes, you need rigs for simple things, and it's not always possible to use an automatically generated rig. You may be working on something that's not a character, or maybe the character is very strange. In those cases, this knowledge is useful.

Furthermore, you learned how to create a collection with your character, link it to other scenes, and reuse it efficiently. Jim is much closer to starting walking around!

Exercises

1. Why is a rig so important for your character?
2. What is an armature?
3. What are vertex weights?
4. What is the difference between forward kinematics (FK) and inverse kinematics (IK)?
5. What are constraints used for?
6. Why are drivers useful?

12

Animating Your Character

Your character comes to life through animation. Animation is the process in which you make your character move along time. To animate characters convincingly, you have to study the principles of motion, such as action and reaction, and you need to understand how weight affects the way a character moves. Animation is a wide subject, with plenty of resources, books, and courses available. In this chapter, you learn the basics of the Blender animation system. You find out what tools are at your disposal, and you learn how to use keyframes and the animation editors that help you during this stage.

Using the Character's Rig

Before you start animating, it's important that you learn how to use the character's rig, which in this case was created with Rigify. In this section, I'll discuss some tips for using the rig while animating:

- When animating a rig, especially its rotations, I recommend setting the orientation for transforms in Local Mode instead of Global Mode to ensure that you're rotating bones with each bone's local orientation.
- In the Sidebar of the 3D Viewport, you'll find options for the Rigify rig, and these options will differ depending on which controllers you select. One such option is IK/FK (inverse kinematics/forward kinematics), available when you select controllers for the legs and arms (see Figure 12.1).
- For animating a walk cycle, I recommend using IK for the legs and FK for the arms. This technique allows you to control the feet's position on the ground. Legs adjust their angle automatically to fit the positions between feet and hips. But the arms may be easier to pose by rotating the bones because hands aren't touching any surface.
- To control the IK/FK switch, just move the slider to the direction of the option you desire (0 is IK; 1 is FK). This slider can be animated to change between using IK and FK during the animation. Also useful are the buttons that make the IK controls snap to the FK controls and vice versa.
- IK and FK controls live on different armature layers so that you can show and hide them as you need them, so if you want to switch between IK and FK controls, you should also show and hide the appropriate layers for the rig.

- Another interesting option for the legs and arms is IK Stretch (see Figure 12.1), which stretches the limbs if the leg isn't long enough to cover the distance between the foot and the hips. You may want to disable this option to avoid unnatural stretching of the legs when using IK controls.

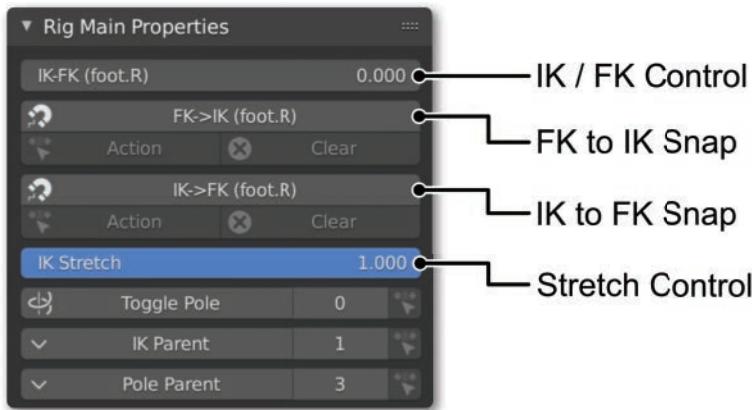


Figure 12.1 Rigify options menu for the leg, located in the Sidebar of the 3D Viewport

Posing the Character

An animation is nothing more than a set of poses that change over time. For this reason, I recommend that you try to create a few poses for your character to get familiar with the rig and its options.

Rigify rigs have many options in the Sidebar of the 3D Viewport. Try them all to learn what they do.

Inserting Keyframes

A *keyframe* is a specific stored value in different parameters of objects at a given point in time. If you want to make an object move from A to B, each position is a keyframe, and Blender automatically interpolates the movement between keyframes. In this section, I describe some ways to add keyframes in Blender.

Adding Keyframes Manually

Inserting keyframes manually is the most basic way to add keyframes to your objects. Set the frame in the Timeline to define the time at which you want to save that specific position, select your object or objects, and press **I** (see Figure 12.2). A menu appears, letting you choose among the channels and transform properties of that object so that you can record a keyframe in the desired channels. You can set a keyframe just for the location (Loc), for example, or for the rotation (Rot). To keep matters simple, choose LocRotScale to set a keyframe for the three transform types, which is what you usually want to do.

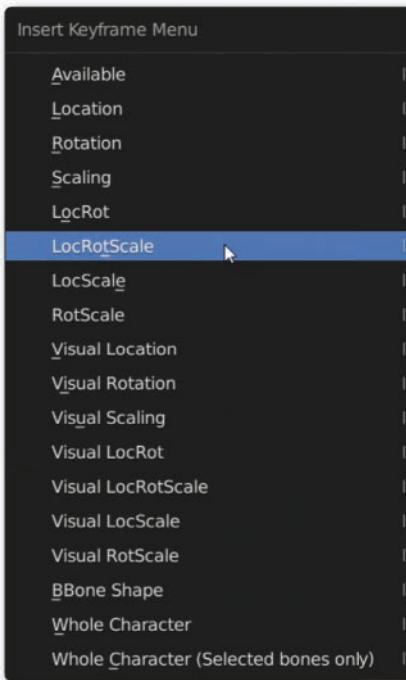


Figure 12.2 Inserting keyframes manually. This menu appears when you press I.

Adding Keyframes Automatically

In the Timeline, at the bottom of the default interface, you can define the duration of the animation, play it, scroll through it, and so on. A little button with a red circle, similar to the Record button on old cassette players, is displayed in the center of the Timeline's header, next to play and navigation controls. If you click this button, you enable the Auto Keyframing option. You learn more about how to use the Timeline later in this chapter.

When this option is enabled, every time you change something, a keyframe is saved for it automatically in the current frame. This option is really useful when you're animating, as it lets you insert keyframes by moving things and saves you the time it would take to insert keyframes manually. You have to be very careful with Auto Keyframing, though, because if you forget that it's turned on, you might save keyframes when you don't want to.

Automatically recording keyframes works with most parameters within Blender, from transforms (location, rotation, and scale) to modifier values. It may not be compatible with all parameters, however, and keyframes aren't saved for a parameter unless that parameter already has at least one keyframe. This restriction prevents the creation of unwanted keyframes, which are created only where animation tracks exist.

Adding Keyframes Using Keying Sets

Using keying sets may be the most comfortable way to add keyframes. Choose a keying set from the Keying menu on the Timeline's header, to display the Active Keying Set panel (see Figure 12.3). The panel has two buttons marked with a key—one for adding keyframes and the other for removing them. When you pick a keying set, you don't need to select the channels in which you want to set a keyframe each time you press **I**. Blender automatically adds a keyframe to all the properties that exist inside the active keying set.

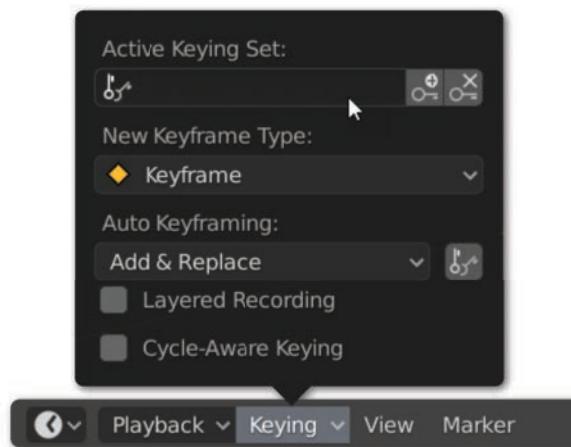


Figure 12.3 The Active Keying Set panel, where you can choose the keying set to which you want to add keyframes

If you select the LocRotScale keying set, for example, when you press **I**, Blender automatically adds a keyframe to the three channels; you don't need to confirm manually each time. Also, a keying set called Whole Character saves a keyframe for the full rig when you press **I**. You don't even need to select the other bones, which is very handy!

Creating Your Own Keying Sets

You can create your own keying sets to accelerate the animation process in specific situations. Imagine, for example, that you want to have a keying set for the facial expressions of your character so that when you press **I**, only the bones and properties that control facial expressions get new keyframes, not affecting any other parts of the body.

You can create new keying sets from the Scene Properties tab of the Properties Editor, within the Active Keying Sets panel. When you create your own keying sets, you can select one that will become the active one.

After you have an active keying set, you can find the properties you want to add to that keying set in the menus. Right-click those properties and choose Add to Keying Set from the contextual menu to add those properties to the active keying set.

Keep in mind that when you select an active keying set in the Active Keying Sets panel, that keying set becomes the one that gets keyframes. To change it, choose a different set from the Keying menu on the Timeline. Not all the keying sets on that menu are shown in the Active Keying Sets panel because those keying sets are built in—and hence not customizable.

Adding Keyframes to Properties in Menus

Animating a property is simple. Yes, you can animate properties within the menus; almost everything in Blender can be animated, such as the number of subdivisions in a modifier, the color of a material, or the influence of a constraint on an object. Place the mouse cursor over the property value field, and press **I**. Blender stores a keyframe for that property's field is yellow in the frames where Blender has stored keyframes and green in the rest of the frames, just to let you know that the property owns an animation (see Figure 12.4).

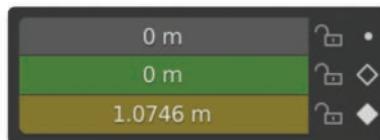


Figure 12.4 Animating properties from menus. Top: The property has no animation. Middle: The property has an animation. Bottom: The property has an animation and a keyframe in the current frame.

Many properties that can be animated have a little dot next to them. You can click that dot to create a keyframe (the same effect as pressing **I**), and the dot will turn into a diamond. If you click the diamond, the keyframe will be removed.

Right-clicking an animated property displays a contextual menu that allows you to insert a keyframe, delete the entire animation for that property, and clear the current keyframe, among other things.

Working with Animation Editors

Several editors help you manipulate an animation, from moving through the animation's time (as in the Timeline) to editing the animation curves to telling Blender how you want it to interpolate the movement between two keyframes. You can even mix animations as though you're editing a video! Figure 12.5 shows the editors.



Figure 12.5 The animation editors: Timeline (1), Graph Editor (2), Dope Sheet (3), and NLA Editor (4)

Timeline

The Timeline is the most basic animation editor, showing the time and displaying keyframes for the selected objects. You can change the current frame and scrub through the animation by holding **LMB** and positioning the green cursor along the Timeline.

In the Timeline, you can also set the start and end frames of the animation. Press **Ctrl+Home** to set the current frame as the start frame, and press **Ctrl+End** to set the current frame as the end. Alternatively, you can type frame numbers in the Start and End fields of the Timeline's header.

Also on the header, you'll find controls for playing the animation, jumping to the start or end, and jumping to the next or previous keyframe. Here are some shortcuts you can use in the 3D Viewport (or any other editor to control time):

- Press **Space** to play the animation. Press **Space** again to pause it. Press **Esc** to stop it and go to the frame from which you started playing.
- To quickly select the frame range you want to play, press **P** and then click and drag the box to cover the desired range. This technique doesn't affect the actual start and end frames; you can disable the frame range you established with **P** by pressing **Alt+P**.

- Press the **Left** or **Right** arrow key to move to the previous or next frame. Press the **Up** or **Down** arrow key to jump to the next or previous keyframe.

Many of these controls work the same way in other animation editors. Other controls that you'll see in the next section also apply to the Timeline.

Dope Sheet

Basic, but very useful, the Dope Sheet editor shows keyframes on objects (the left side lists the objects in the scene) as yellow diamonds on a timeline. This editor lets you move keyframes so that you can adjust the timing of your animation. It's essentially an advanced version of the Timeline editor for controlling the timing of all of keyframes in the scene.

You can use basic keyboard shortcuts such as **G** to move keys, **S** to scale them, and **Shift+RMB** to select several keys at the same time. Press **B** to box-select a group of keyframes (or click and drag to use a box selection). Press **Shift+D** to duplicate keyframes, and press **Ctrl+C** and **Ctrl+V** to copy and paste keyframes, respectively.

Summary, at the top of the list, displays all the keyframes in the same line, which allows you to control them together. Some options allow you to display only the selected objects' keyframes, if that's your preference.

The Dope Sheet has different modes, and the header features a mode selector. By default, the Dope Sheet is selected to allow you to access objects' keyframes. Other options are Action Editor, Shape Key Editor, and Grease Pencil. Each of those modes allows you to control the keyframes of given types of objects or animations. The Shape Key Editor, for example, shows keyframes stored in shape keys within the selected objects.

An object can store different animations (actions), and in the Action Editor, you can select which one to play. This editor also lets you name the actions. Later, you can mix these actions in the Non-Linear Animation Editor (explained later in this chapter). If you work on videogames, you probably know that a character can have different animations, such as walking, running, remaining idle, and picking up an object. You can use those actions in the same scene and switch among them, create new ones, and edit existing ones, thanks to the Action Editor.

You could say that the Dope Sheet is a general editor, displaying all the animations going on in the scene. The Action Editor works with a specific animation of a specific object and is especially useful with armatures, allowing you to store different animations for your characters. The Action Editor's header has a selector for the action you're working with.

Graph Editor

The Graph Editor is the scariest thing you'll see in animation. Seriously, it's not that complicated, but the first time you see the animation curves without properly understanding how they work, they can look complex and frightening.

In the Graph Editor, you can edit the animation curves, also called *F-Curves* in Blender. Animation curves are pretty simple.

When you create two keyframes, Blender automatically creates an interpolation between them. The curves define how the interpolation happens, and you can control the curves to change the interpolation in case you’re not happy with the default behavior.

Each transform or property has an F-Curve on each axis. Suppose that you want to make an object fly from point A to point B. Figure 12.6 shows what would happen with different curve configurations.

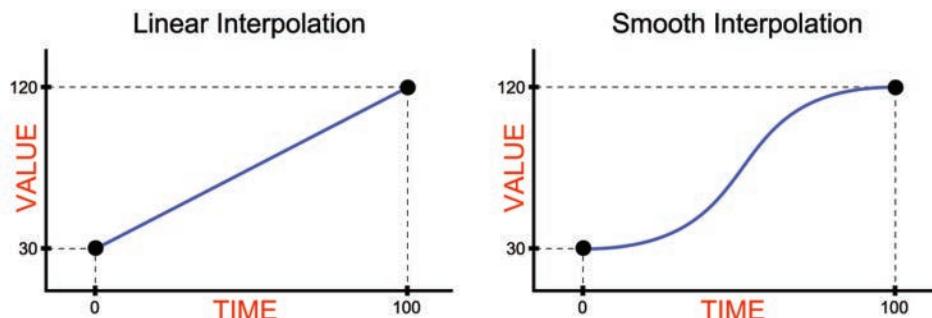


Figure 12.6 Different curves that interpolate the same two keyframes

Figure 12.6 shows two keyframes. The position of a keyframe horizontally defines its time, and its vertical position sets its value. In the figure, the first keyframe sets a value of 30 in frame 0. The second keyframe has a value of 120 in frame 100.

The image on the left represents a linear interpolation, which tells Blender to increase the value from 30 to 120 between frames 0 and 100 at a constant speed.

The image on the right shows a smooth interpolation. Blender starts increasing the value slowly and then accelerates; it starts decreasing slowly before reaching the value of 120 at frame 100.

As you can see in the figures, these two different movements between the same keyframes are defined by the animation curves.

In the Graph Editor, you can manipulate the curves. Move, rotate, or scale keyframes by pressing **G**, **R**, and **S**, respectively. You can duplicate keyframes by pressing **Shift+D**.

To change the interpolation type of a keyframe, select it, and press **V**. (You can do the same thing in the Dope Sheet, but the difference is more apparent in the Graph Editor.) A menu appears, displaying options that allow you to set the interpolation and the handle type of that keyframe.

Tip

The Editing tab of User Preferences has an option that allows you to set how your keyframe interpolations are adjusted by default when you create them. You can find the Interpolation option in the New F-Curve Defaults section.

You can tweak handles by clicking and dragging, or you can select the keyframe and press **R** to rotate the handles' positions. Modify the curvature of the interpolations to control how the animation accelerates or decelerates and at which rate. Another way to tweak handles is to select a keyframe and then rotate or scale it, which changes the behavior of the keyframe's handles.

Tip

Now that you've learned about both the Graph Editor and the Dope Sheet, you can switch between them quickly by pressing **Ctrl+Tab**.

Non-Linear Animation

The Non-Linear Animation (NLA) Editor is interesting. It works similarly to a video editor. You can use it to load strips, mix them, layer them, repeat them, make them longer or shorter, and so on, but instead of videos, you'll be using animations (actions)!

You can load actions that you saved in the Action Editor (a mode of the Dope Sheet, discussed earlier in this chapter) and mix them to build a bigger animation. Later in this chapter, in the “Repeating the Animation” section, you use this editor to repeat a walk cycle constantly. Instead of animating Jim taking ten steps, you animate a repeatable cycle of two steps (one for each leg), and with the NLA Editor, you can repeat that action for as long as necessary.

Suppose that you have another action of Jim running. You could add that action to the NLA Editor as another strip by pressing **Shift+A**; you could mix the action with the walk-cycle animation so that Jim goes progressively from walking to running.

Another option is to add tweaks to an animation as though they were layers. You could have an animation with Jim turning his head, looking to the side. You could add the secondary animation over the walking animation so that Jim could walk and move his head at the same time. Think of the possibilities!

Common Controls and Tips

All these editors have some controls in common. You can move or duplicate things in these editors the same way that you do in other Blender editors (by pressing **G** and dragging your mouse to move or pressing **Shift+D** to duplicate). You find other similarities for navigation:

- Press **MMB** and drag to pan so that you can see other parts of the charts.
- Press **Home** to adjust the zoom so that Blender shows the entire animation time range.
- Press **Ctrl+MMB** and drag your mouse to zoom. In some editors, you can zoom only horizontally, but in the Graph Editor, you can zoom horizontally and vertically. You can also zoom in and out by using the scroll wheel.
- Press **P** and drag the resulting box to define the start and end frames of the animation temporarily. You can disable the temporary frame range by pressing **Alt+P**.

- Press **Alt** and use the scroll wheel to go to the next or previous keyframe, which is a fast way to scrub through your animation (in small steps). You can also use the arrow keys to navigate through your animation. Press **Left** and **Right** to go to the previous and next frame or **Up** and **Down** to go to the next and previous keyframes you've saved for that object. Pressing **Shift** while tapping **Left** and **Right** jumps directly to the beginning and end of the Timeline.

Here are some options that can make it easier to work with these editors:

- On the header of most of the animation editors, you'll find a button that shows a mouse cursor. If you enable that option, you see the keyframes of only selected objects, which can prevent you from going crazy looking at dozens (or hundreds) of curves or keyframes at the same time. This option helps you recognize what you're seeing in those editors.
- Choose Only Selected Curves Keyframes from the View menu on the header in Graph Editor. This option displays only the keyframes of the curves from properties you select in the list, preventing you from selecting or accidentally moving keyframes of other curves that overlap the one you want to adjust. In the Graph Editor, for example, you see a list of objects and properties on the left side of the interface; you can use the buttons next to them to lock, hide or show, pin, and so on.
- The Graph Editor's header also has a Normalize option. When you're working with curves that have different value ranges, navigation can be hard, as you have to zoom in and out continually to see what you need to see. When Normalize is enabled, all the curves are proportionally adjusted to fit inside a 0-to-1 range for easier tweaking. This change is visual only; it doesn't affect the real values of the curves.

Frames per Second

I recommend that you set the frames per second (fps) at which you want your animation to play. Blender's default setting is 24, used in many formats, but depending on your country, this rate may vary. (In the United States, the typical fps rate for TV broadcasting is 29.97; in Europe, the typical rate is 25 fps.) Or you may want to select a different frame rate for artistic reasons; using 10 fps, for example, can help you mimic a stop-motion animation.

Whatever rate you choose, you should set it before you start working on your animation. Otherwise, if you change the fps rate later, playback speed will vary, and you'll probably have to adjust the timing of your animation.

You can change the frame rate in the Dimensions panel of the Properties Editor's Output Properties tab. I recommend you use 25 fps for the animations for the project in this book, as the videos in which Jim will be composed in Chapter 14, "Lighting, Compositing, and Rendering," have been recorded at 25 fps.

Animating a Walk Cycle

This section guides you through the process of creating a basic walking animation. Make Jim come alive!

Tip

When you animate, you should have a good performance in the 3D Viewport so that you can view your animation smoothly while you work. Blender has an interesting option called Simplify; look for it on the Render Properties tab of the Properties Editor. If you enable Simplify, you can select the maximum number of subdivisions for every object in the scene, both in the 3D Viewport and during render. In this exercise, the model probably uses two to three levels of subdivision, which is nice for a final render, but the effect on performance can be big, making the animation drop frames and play slower than it should. You can change the Simplify value to 1 or 0 while you work on the animation and leave it at 2 or 3 for the render. You can set these parameters in the Subdivision Surface modifier that you add to each of the objects. Simplify, however, acts on all the modifiers of the scene at the same time, making it very valuable when you have many objects, because tweaking those parameters one by one would be time-consuming.

Creating an Action

The walking animation is a cycle, meaning that you have to animate only one step that will be repeated. (The step occurs in a static place, so the character won't move across space; that movement will be animated later.) As you'll be repeating the step by using the NLA Editor, you need to create an action that you can load later.

Each object has different actions, so make sure that you have the rig selected in Pose Mode before you create an action. (If you're in Object Mode, Blender creates an action for the container object, not for the bones inside, which is what you need.)

Open the Dope Sheet, and switch to Action Editor mode. If you haven't animated anything yet, the header has a field in which you can create a new action. Click the New button, as shown in Figure 12.7, and this action is called (surprise!) Action. Rename the action something intuitive, such as Walk_Cycle. Now every keyframe you set for the bones in the rig is part of the Walk_Cycle action.



Figure 12.7 The Action Editor's header, where you can create new action for your character

Caution

If you create more than one action, remember that Blender deletes datablocks that are not in use when you close the program. Each object can use only a single action at a time. If you want to make sure that the rest of them are saved, click the shield button next to their names so that they have a fake user. This action prevents Blender from deleting them when you exit because they're technically in use.

Creating the Poses for the Walk Cycle

To make Jim walk, you must define the basic poses for the character that form the motion of walking. There are two contact poses: the moments at which both character's feet are touching the floor. There are two other poses when the character has one foot on the floor and the other one is in the air. Those four positions define the basic walk movement.

You must create a motion that can be repeated, which means that the animation needs to end in the same position as the one in which it starts. This type of animation is called a *loop*. Figure 12.8 shows the main poses and a couple of additional ones to refine the movement. Notice that after all the poses is an extra one that's equal to the first pose (pose 7). This pose is added so that the animation's end fits with its start so that the loop can be repeated.

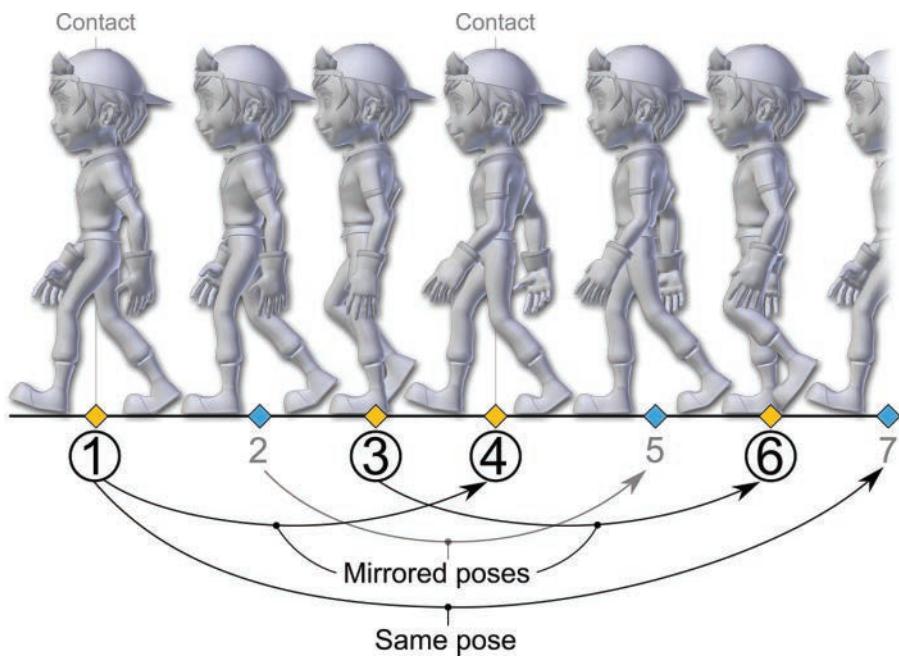


Figure 12.8 Here are the basic poses you need to create to make your character walk.

Figure 12.8 represents several things:

- Poses 1, 3, 4, and 6 (marked as yellow keys) are the main poses. Poses 2 and 5 (marked as blue keys) are extra poses to better define the movement, and pose 7 is the same as pose 1.
- Actually, you need to create only three poses: 1, 2, and 3! Poses 4, 5, and 6 are nothing more than mirrored versions of poses 1, 2, and 3. Pose 7 is a copy of

pose 1. Being able to copy and paste poses (even mirrored) makes the process a lot easier and faster.

- Poses 1, 4, and 7 are the contact poses, in which the foot that goes forward touches the ground. Poses 3 and 6 are the intermediate poses between two contacts to set the moments in which Jim is standing on a single leg.

Before diving into the animation process, take a look at the timing, which defines the speed of the animation (see Figure 12.9).

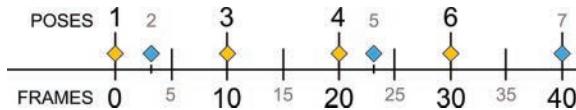


Figure 12.9 A representation of the Timeline, showing the poses on the top row and the frame numbers on the bottom row

Figure 12.9 shows the frame numbers at which you should create the poses for the animation. Keep in mind that regardless of the frame number in which you create a keyframe, you can always rework the timing later, using one of the animator editors (the Dope Sheet is especially useful for retiming keyframes), and make Jim walk faster or slower. What you can see is that a main pose happens every ten frames. (Keeping 10 frames between the poses isn't a rule—just a way to make work more convenient and organized at this early stage of the animation process.)

Follow this easy process to create the walking animation:

1. Set your animation to start at frame 0 and end at frame 40.
2. Pose your character in frame 0 with the first contact pose. Set a keyframe in all the bones of the rig. (You can select any bone, select Whole Character Keying Set, and press **I** in the 3D Viewport.)
3. Select all the bones (**A**), and copy this pose by clicking the Copy Pose button on the Pose menu or by pressing **Ctrl+C**. Go to frame 20, and paste the mirrored pose by pressing the Paste Inverted Pose button on the Pose menu or by pressing **Shift+Ctrl+V**. Set a keyframe. Move the Timeline to frame 40, and paste the pose normally (**Ctrl+V**). Set another keyframe. Now you have all your contact poses in place. You create those keyframes first because now if you go to frame 10, you have an intermediate pose almost ready, generated by the interpolation between the first two contact poses.
4. Pose the character so that he looks better in frame 10, and make sure that the pose of his feet is good to go (one foot on the floor and the other one passing through the air). Insert a keyframe, copy this pose, and paste it mirrored in frame 30. Set another keyframe.

5. A few frames after the contact poses (poses 1 and 4), adjust the poses to make the foot in front step on the ground completely, and keep elevating the foot in the back from the tip while the foot keeps sliding back. This simple pose gives the walk cycle a more natural look.
6. Play the animation to see how it looks, adjust the poses, and copy them to the other moments of the animation that are similar. If you want to make everything smoother, you can tweak the animation curves in the Graph Editor. If some curve looks rigid, adjust it while you play the animation to get instant feedback.

Repeating the Animation

Jim is walking in place but taking only one step. Before you make Jim move through space, you have to repeat the animation so that he performs more steps. You can accomplish this task in several ways. You could duplicate all the poses in the Dope Sheet to add another step after the first one. But in this section, you use the NLA Editor to make it easier. Follow these steps:

1. Open the NLA Editor, which looks similar to Figure 12.10. The editor displays a horizontal line with the name of your current action (inside the rig's name, which is the object that performs the action). The action's keyframes are displayed in the editor, along with a button with a down arrow pointing toward a couple of rectangles.

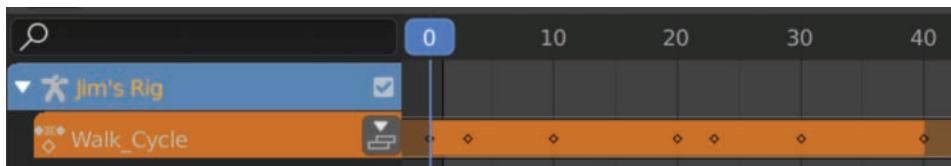


Figure 12.10 The NLA Editor when you open it

2. You have several ways to proceed.

The first option is to click the X button in the action's name inside the Action Editor (make sure that you click the F near it first) so that Jim stops performing that action and frees the NLA Editor to do its job. Go to the NLA Editor, press **Shift+A**, and select the Walk_Cycle action to add it as a strip.

The second option, which is much easier and faster, is to click the button next to the Walk_Cycle action in the NLA Editor. This option turns the action into an NLA strip and performs the actions from the first option automatically. Now you can move that strip to change the time at which the animation happens. You can even scale by pressing **S** to make the action faster or slower, or duplicate it to add more steps (see Figure 12.11). Alternatively, from the Action Editor, you could be working on the Walk_Cycle action and press the Push Down button in the header, which performs the same action as pressing it in the NLA Editor.

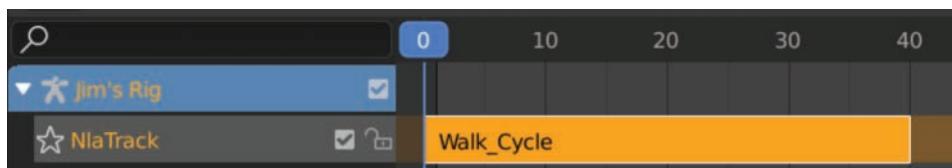


Figure 12.11 The NLA Editor with the Walk_Cycle action as a strip, ready to be edited

3. You could duplicate the Walk_Cycle strip to add more steps to your animation, but here's a more elegant method. Press **N** in the NLA Editor to open the Sidebar. In the Action Clip panel, within the Strip tab, cut frames at the start or end of the animation. Set the end as frame 39 instead of frame 40 so that pose 1 doesn't repeat in two frames; this setting makes the transition more natural when you repeat it. In the same panel, you can change the playback scale of the strip to adjust the animation speed. You can also set the repetitions in the Repeat slider to something like 5 or 6 so that you have enough steps in the animation.

As you can see, using the NLA Editor is a very easy way to control actions instead of dealing with a lot of keyframes to achieve the same result. This chapter is only an introduction, but you can do all sorts of cool things in this editor, such as overlapping actions and creating transitions between actions.

Walking Along a Path

Jim is taking a lot of steps now, but he's not moving in space. Follow these steps to use a constraint so that he follows a path while he walks, as shown in Figure 12.12:

1. Press **Shift+A**, and choose a Path object from the Curve category of the Add menu.
2. In Edit Mode, edit the points of the curve to describe the path you want Jim to walk. In this case, the path should be a straight line. Delete all the points of the curve except the start and end ones. Place the start vertex at the origin of the scene, where Jim is, and the end vertex aligned in the Y-axis in front of Jim.
3. Select Jim's rig in Object Mode to apply a constraint to it. On the Constraints tab of the Properties Editor, add a Follow Path constraint.
4. From the Follow Path constraint menu, choose the path you just created. If the path doesn't work automatically when you play the animation, click the Animate Path button available in the constraint. If you want a curved path, enable the Follow Curve option so that the character turns with the path.
5. Select the path, and on the Curve Properties tab of the Properties Editor, in the Path Animation panel, adjust the number of frames that Jim uses to go from the start to the end of the path. Also, adjust the length of the curve (by moving its vertices in 3D space) and the speed of the walking cycle until you get a nice result and Jim's feet slide over the floor as little as possible. (You can

activate the grid on the floor and add more lines to it from the Overlays popover in the 3D Viewport to get a reference for how Jim is moving over the floor.) You can also use the 3D cursor to mark where the heel steps in space, giving you a reference point for correcting the sliding of the feet.

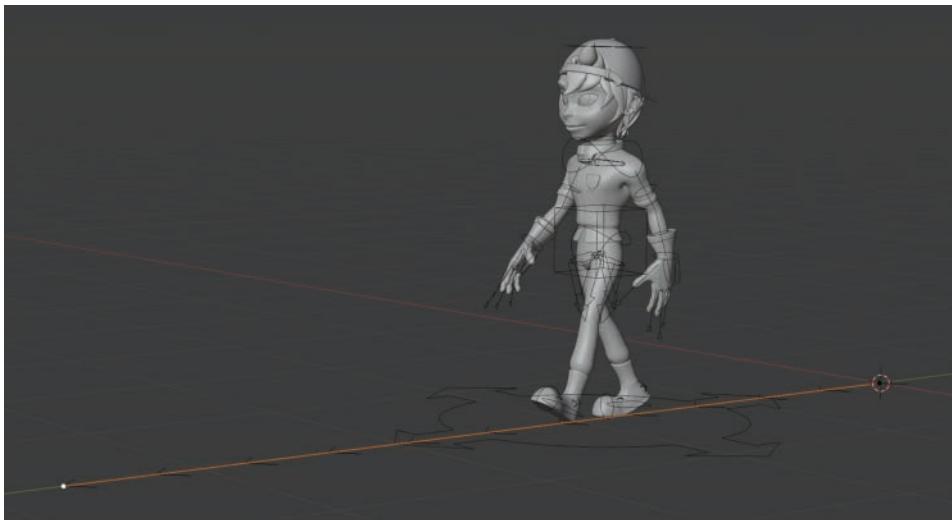


Figure 12.12 Jim walking along the path on the floor. He's alive!

Summary

Animation is a complex thing. Creating believable movements for your characters is tricky and requires a lot of knowledge, experience, and dedication. I hope that this chapter showed you the basics and encouraged you to learn more.

Now you have a walking, finished character. And you started this book with little or no idea of how Blender works! In Chapter 13, “Camera Tracking in Blender,” you will learn to move a 3D camera so that it matches the movement of a camera in a real video, which will allow you to add Jim (or any 3D object) to that real video.

Exercises

1. What is a keyframe?
2. What animation editors are available in Blender?
3. What are animation curves used for?
4. When would you use actions in Blender? What’s the advantage of using them?
5. Which constraint type would you use to move an object along a curve?

VI

Getting the Final Result

13 Camera Tracking in Blender

14 Lighting, Compositing, and Rendering

This page intentionally left blank

Camera Tracking in Blender

When you want to mix your 3D objects into a real-world video, you need a camera in the 3D world that moves exactly the same way as the camera that filmed the real-world footage so that the 3D objects fit perfectly into the scene. *Camera tracking* is the process that allows you to track features in the real video to give Blender information about the changes in perspective so that it can generate a 3D camera that mimics the movement of the real one. Until recent years, you needed expensive specialized software for this purpose. Now Blender provides an efficient alternative, and the best thing about it is that you don't need to import/export scenes or use any other software; everything you need is already inside Blender! In this chapter, you learn the basics of camera tracking in Blender.

Understanding Camera Tracking

Before you start the process of tracking, it's important that you understand how it works:

1. First, you load a video and track its features (patterns or distinguishable shapes and points) by using the tracking tools in Blender's Movie Clip Editor. Good features are typically details in the video that are highly visible and static, and have a high contrast or distinct shapes (such as the corner of a stone, a signpost, or a mark on the ground) so that Blender can recognize them easily from one frame to the next.

Tracking uses these features' motion to establish the image's perspective and generate a camera movement that fits those perspective changes. This process is generally done frame by frame, but sometimes it can be faster; Blender also provides some automatic tools that try to do most of the work, so you have to manually track only those areas and moments for which the algorithm can't figure out the movement.

2. Next, when you have enough tracked markers in the video, you input the camera settings to tell Blender the type of camera and lens you used to film the video. If you don't know exactly what lens you used (or if you didn't shoot the video yourself

and can't access that information), Blender can estimate the settings and give you camera parameters quite close to the ones used for filming the video.

3. You have to solve the camera's movement. At this stage, Blender analyzes the points you tracked through the video, and by comparing their movements in different frames, it reconstructs the camera's perspective and determines where the 3D camera should be on each frame to fit those tracker's positions. You end up with a 3D camera that moves exactly the same way as the real camera you used.
4. Finally, you align the 3D camera and adjust it to your scene so that its orientation is correct and your 3D objects fit within the real-world footage.

Shooting Video for Easy Tracking

Sometimes, when you have a video that you need to track, the level of difficulty directly depends on how the video was shot. If the video wasn't shot with some basic considerations in mind, you'll probably have a hard time tracking it. The video may not contain enough clearly visible features that you can use to track it, for example, or it may be very blurry or fast-moving. Keep in mind that a movie may have a lot of these tricky shots to track, but also realize that people who have significant experience and use expensive software put in a lot of effort (and sometimes even perform manual "magic") to make the 3D camera fit perfectly with the camera that took the real footage. If you want to create your own videos and prevent your shots from being too difficult to track, you just need to know how the process works and follow some guidelines.

Camera tracking uses what is called *perspective shift* or *parallax effect* to detect the perspective and depth in your footage. Suppose that you're inside a train, looking out at the window. The objects close to you appear to move really fast, and the objects farther away, such as clouds and mountains, are almost static. This situation is what perspective shift is all about: An object that is close to the camera shifts its perspective faster than one that is far away.

Knowing this, you can understand that a video in which you move the camera to show that perspective shift helps Blender determine where the markers are in 3D space. If there isn't enough perspective shift in the camera movement, you may have a hard time tracking the camera movement. If that camera motion doesn't fit the idea you have in mind for the shot, don't worry! You can shoot some video reference frames that capture perspective shift before shooting the actual video; then you can use the frames of the reference footage to show Blender the correct perspective, and the rest of the video will incorporate that perspective while tracking. This technique gives you good results when you're working with a video that has a minimal perspective shift. When you finally edit the video, you can cut out those reference frames at the beginning, of course.

You can also track camera movements that don't involve perspective shift, but in that case, the tracker will interpret only the rotation of the camera, but not change its location—as though the camera were on a tripod. When you solve the camera motion, you can choose which type of movement should be calculated.

Here are some other useful tips on shooting a video for tracking, especially in shots that involve camera movement (not just rotation):

- Because of perspective shift, it helps to have tracking features in both the foreground and the background to give Blender a good reference of the real scene's depth.
- Make sure that the video has enough recognizable features that you will be able to track throughout the video. (High-contrast elements with 90-degree corners offer the best results for tracking, as they're easy to even track manually if the automatic systems don't do a good job.) The more often a feature appears during the video, the more stable the camera motion's solution will be. If you feel that the video doesn't have a lot of trackable features, place something in the scene that can help you—a little stone here, a piece of paper there. Just add things that will help you later and won't distract the viewer. An alternative is to add physical markers (usually, small designs that contain contrasting patterns that you can print and place in your scene during filming). This technique can be tricky, however, because you may have to remove the markers later in postproduction, so it's best to place only small, unobtrusive objects in strategic positions.
- Avoid zooming if possible, because changes in the lens while shooting the footage can compromise the tracking and make it much trickier. You can always fake the zooming in postproduction (at the expense of some image quality) by making the image bigger.
- Try to prevent very fast movement that might blur the image. If you have blurry footage, chances are that you'll have to track parts of it manually, and the tracking probably won't be very precise because you won't be able to see the tracking features clearly. Use some kind of camera stabilization, if possible. Adding a simple weight under the camera can help a great deal!
- Shooting a good-quality video makes tracking a lot easier. If the video has compression artifacts or low resolution, small (and even big) features can change a lot from one frame to the next, making things very difficult for Blender's automatic tracker, and you'll have to do a lot of manual work.
- To track the camera movement, you can use only features that are static. Don't use things that move as features to track, because they can break Blender's perspective analysis. Keep this fact in mind when you plan how to shoot the scene to make sure that you have enough static features to track. Remember that the more features you track, the more stable and closer to the real footage the 3D camera movement will be.

Examples of things to track are stones on the floor, bricks in a wall, a manhole on the street, and a signpost. Anything that's fixed and not moving in 3D space is good.

Things to not track include leaves (they may be trackable, but there's a chance that they're moving with the wind, even if subtly); people (even if they're trying

to remain still, they'll be moving a bit); and corners generated between two objects that are not at the same depth, such as a post behind a wall (the point at which those two objects join in the image may change and slide if the camera perspective changes), reflections, and particles freely floating in water. All these things are not completely stationary, which will confuse Blender when it calculates the perspective.

- If possible, take note of the focal length and other camera parameters you used while shooting the video footage. That information will help Blender solve the 3D camera's motion.

Using the Movie Clip Editor

Camera tracking happens inside the Movie Clip Editor. Work on a Blender area in the Movie Clip Editor by selecting that editor type on the area's header. Figure 13.1 shows the editor.



Figure 13.1 The Movie Clip Editor is where you perform camera tracking.

You may not understand anything about the Movie Clip Editor right now, but don't worry. Here's a quick explanation:

- **Toolbar (A):** Here, you find options for creating new markers and for tracking and solving the camera's movement. Press **T** to show and hide this region. There are different tabs for tracking settings, camera movement solving, and annotations.

- **Movie Clip (B):** This area is where you'll see the real footage you've shot and is where you can track features by using markers. The integrated and tracking specialized timeline at the bottom lets you work with this editor full-screen.
- **Sidebar (C):** In this section of the editor are the settings of the selected marker, together with display and camera parameters on the Track tab, as well as footage settings on the Footage tab. You can also find another tab for Stabilization. Press **N** to show and hide this region.

You learn more about working with the Movie Clip Editor throughout this chapter.

Tracking the Camera Motion

In this section, you learn how to load video footage, track moving points in the image, and generate the 3D camera movement that simulates the movement of the real camera. For now, you need to see only the Movie Clip Editor, so you may want to make it full-screen by pressing **Ctrl+Space**.

Loading Your Footage

You can't track your footage if you have no footage, of course, so first, you must load it. You have different ways to load your footage, but the process is basically the same as loading images in the UV/Image Editor:

- You can press **Alt+O** and select your footage.
- You can click and drag the footage from your operating-system folder into the Movie Clip Editor.
- You can choose the Open Clip option from the Clip menu on the header.
- You can click the big Open button on the header.

After you load your footage, you can scrub through it by using the timeline at the bottom of the editor, as shown in Figure 13.2. When you're tracking, you don't need to use anything other than the Movie Clip Editor to scrub through the footage.

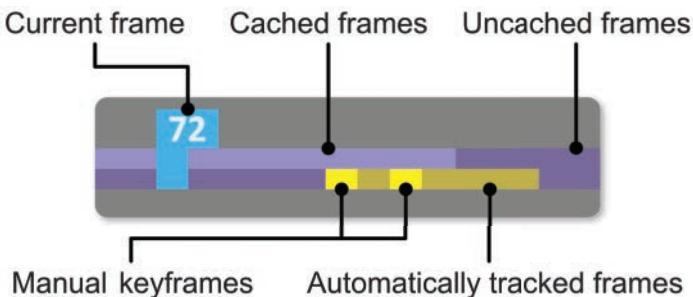


Figure 13.2 Movie Clip Editor's timeline

Pressing **LMB** and dragging the blue numbered cursor on the timeline changes the current frame and displays the frame number. The timeline also has two horizontal lines, one blue and one yellow; the blue line is for the footage. As you play the video (**Space**), Blender caches the frames, so the next time you play the video, the frames load a lot faster. I recommend that you play the entire video to cache it completely before you start tracking, which will make playback faster while working. Alternatively, if you click the Prefetch button at the top of the toolbar (or press **P**), Blender automatically caches the video without playing it. The blue line is lighter in the parts that are cached, so you'll know which parts are cached and which aren't.

Tip

By default, Blender uses only a small amount of RAM to cache your footage. If you're working with a big video or full HD, the default amount may not be enough to cache all of the frames in the footage. On the System tab of User Preferences, you find a Video Sequencer panel where you can increase the amount of memory to be used for caching videos. A value of 8000 (8GB) is enough for most videos (depending on their length, resolution, and format, of course). Also, your machine needs to have enough RAM for this purpose.

As an alternative, in the case that you still don't have enough memory to work on the full video, you can process the tracking portion by portion. You can track the first 100 frames and then work on the next 100, for example, by setting the timeline's start and end frames.

The yellow line is for the markers. When a tracker is at work, moving and following a feature, Blender displays a muted yellow line. In the frames where you added manual keyframes to the track, the line is brighter yellow, so you'll know which parts of the footage have keyframes that were added manually and which parts were tracked automatically.

Video Resolution and FPS

When you load a video, remember that resolution and the frames per second (fps) rate will be those that you set up in the Properties Editor. On the Output Properties tab (the tab with a camera icon), select the proper size and frame rate in the Dimensions panel. The downloadable videos provided with this book (inside the Resources folder) are 1920px × 1080px (Full HD) and 25 fps.

After you load a video in the Movie Clip Editor, you'll find the Footage tab on the Sidebar, including some information and settings for loading the video. Depending on the format you filmed in, you may want to change some of those settings.

Studying the Anatomy of a Marker

Markers (also called *trackers*) are the main tools you use to track features in your footage, so before you start tracking, you need to understand what a marker is and what its parts do (see Figure 13.3):

- **Pattern:** The pattern is the main part of the marker—the area of the image in one frame that Blender (or you, manually) will look for in the next frame to track it. Usually, you should use a feature of the footage that is easy to recognize as the pattern’s center: a high-contrast area or a specific shape that is unique in the image. You can move the marker, rotate it, and scale it as always by pressing **G**, **R**, and **S** (and combine them with **Shift** to enable precision transforms). Also, the Track panel of the Movie Clip Editor’s Sidebar displays an image that shows the selected marker’s pattern, so you can clearly see the pattern that the selected marker will be looking for in the next frame. You can also use general shortcuts such as **H** and **Alt+H** to hide or unhide markers and **Shift+H** to hide all markers except the selected ones.
- **Search area:** This area is where Blender looks for the pattern defined in the pattern area in the next frame. (This area isn’t visible by default; you need to enable it in the Marker Display panel of the Sidebar if you want to see it.) The faster the movement of the image, the bigger the search area needs to be, because if the pattern in the next frame falls outside the search area, Blender won’t find it. The bigger the search area, however, the slower automatic tracking becomes. You can change the size of this area or its position by pressing **LMB** and dragging its top-left and bottom-right corners.
- **Pattern orientation:** You can rotate the pattern (or distort it by dragging its corners) to make tracking it easier. If you do, you also have options to track the pattern’s rotation or perspective, which can be very useful at times, such as for tracking a feature that has a recognizable pattern but not a specific point that you could use to place the center manually and accurately. When you press **LMB** and drag the little square at the end of the short line, you can rotate or scale the pattern (or press **R** and **S** when you select the marker).

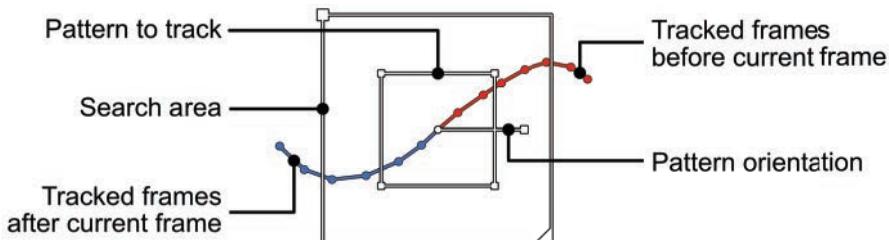


Figure 13.3 The main parts of a tracker

- **Tracked frames:** When you’re tracking, the marker shows a red line and a blue line, both with dots across them. The track is always in the current frame’s position; the blue line shows the trajectory and positions of the next frames (if tracked), and the red line displays the tracked positions in the previous frames. These lines can help you compare different markers’ movements to determine whether one of them is visibly off.

Tracking Features in the Footage

Now that you know the basics, you can explore the marker settings.

You can modify the tracking settings for markers in two places: the toolbar (left side) and the Sidebar (right side). The settings on the Sidebar belong to the selected marker, and the settings on the toolbar are the parameters that Blender applies when you create new markers.

Tracking settings include options and values that control things such as the color channel the settings will be tracking, the pattern and search size, and whether the marker will track location (you’ll generally use this option), as well as rotation, scale, and perspective of the features.

The Match Type option is very important. You can set it to match the keyframe or the previous frame. The Keyframe setting tells Blender to look in every frame for a feature similar to the one you set in the last manual keyframe for the marker. The Previous Frame setting looks for a feature similar to the one in the last tracked frame.

Usually, you want to use Previous Frame, as a tracking feature can change slightly throughout the video (because of perspective and distance from the camera), so it’s better for the tracker to adapt to that small change that happens frame by frame.

Otherwise, at some point, the feature will be very different from its appearance in the previous manual keyframe, and the tracker will stop working.

The amount of difference from the previous tracked frame or manual keyframe necessary for Blender to lose a tracking point and stop the automatic tracking can be defined in the Correlation setting, inside the Tracking Extra Settings subpanel in the marker’s Tracking Settings panel.

Now it’s time to start tracking. Here are the steps you need to follow to track features:

1. First, create a marker. On the Marker tab of the Toolbar, click the Add button; then left-click the footage where you want to create the marker.

Tip

A quicker way to create markers is to press **Ctrl+LMB** where you want to place the new marker.

2. Adjust the marker so that the pattern area fits the feature you want to follow (maybe a corner or a spot). Scale the marker up or down by pressing **S** to adjust

it to the feature's size. In the Track panel at the top of the Sidebar, look at the zoomed version of the pattern defined by the marker to make sure that it's placed correctly.

3. Track that feature along the footage by using the Track Forward and Track Backward features. In the Track panel of the toolbar, launch automatic tracking or press **Ctrl+T** to track forward or **Shift+Ctrl+T** to track backward. You can also find buttons for these options and other interesting ones (like clearing tracking data in case it's not right) in the Track panel of the toolbar's Track tab.
4. Track frame by frame by pressing **Alt+Right** and **Alt+Left**. Press **L** to center the view on the selected marker.
5. It's possible that a tracker will stop tracking for no apparent reason, as the pattern is still clearly visible. One possible cause is that the pattern is outside the marker's search area. The size of the search area defines the part of the footage in which Blender looks for the pattern in the next frame. The bigger the search area, the more pixels Blender has to analyze and compare, and the slower the tracking process can be.

However, if the search area is too small or if the pattern moves too fast from one frame to the next and falls outside the search area, Blender won't find it, and the marker will be disabled.

In the Movie Clip Editor's header, in the right corner, you'll find a pop-over menu called Clip Display with options to choose what elements to display in the Movie Clip Editor. The search area (called Search in this menu) may be disabled by default, so if you enable it, you'll be able to see the search area for the selected markers.

You can move and resize the search area of a marker by using the corners of the Search Area indicator around the marker, or you can use the settings inside the Marker panel of the Sidebar.

Tip

Sometimes, a feature won't be visible. Suppose that you're tracking a window in a building in the background, and it's obscured in some frames by a post in the foreground. You can stop tracking, skip some frames, and start tracking when the feature becomes visible again. Blender evaluates a marker only while it has tracking keyframes (manual or automatic), so if you skip some frames without tracking the marker, it will be treated as disabled for those frames. You can also manually force the marker to be disabled or enabled at a specific frame by clicking the button with an eye icon in the Track panel (on top of the pattern preview, next to the tracker's name) of the Sidebar.

6. Track the markers one at a time to make sure that tracking is progressing correctly. It's possible to track multiple markers at the same time. Keep in mind,

however, that you can't focus on more than one marker at a time to make sure that the track is stable, so I recommend that you track markers one by one, especially for tricky features that may cause problems.

- When a marker is correctly tracked in all parts of the footage where the feature that the marker is following appears, lock it to prevent accidental moves or tracks. The Track panel of the Sidebar has two icons: an eye and a lock. The eye icon enables and disables the tracker, and the lock icon makes it impossible to adjust the marker until you unlock it. You can also lock the selected trackers by pressing **Ctrl+L** and unlock them by pressing **Alt+L**.

Tip

When you press **Ctrl+T** or **Shift+Ctrl+T** to automatically track with a marker, tracking is very fast (depending on your computer and the complexity/size of the pattern Blender is searching for) and almost impossible to follow. Sometimes, that high-speed tracking feature is good because Blender completes the tracking quickly and stops only when tracking fails. In some situations, however, even though tracking doesn't fail, it's not correct because it slides little by little on the feature, meaning that the track won't be exact.

To better monitor tracking, go to the Tracking Settings panel of the Sidebar, and set the Speed option to Realtime or slower. This way, even though Blender can track the feature faster, it tracks it at normal speed so that you can see what's going on throughout the process. You can also set a frame limit so that the automatic tracking stops after a certain number of frames, giving you time to react if something goes wrong. A good value for this option is 20 to 30 frames.

You can also press **L** when you're tracking to enable the Locked to Selection option (next to the Clip Display button in the right corner of the Movie Clip Editor's header). When this option is turned on, the camera is centered on the marker, and only the background video moves, making it much easier to see whether the tracker is stable and avoid having to pan the view around all the time.

- Repeat the entire process from Step 1 for as many features as you can. Try to have a minimum of eight to ten tracked markers in every frame. Press **M** to mute the video (press it again to unmute), and play the animation to see only the markers moving against a black background. This technique will help you detect strange movements in some markers by comparing the motion with that of other markers.
- Make sure that no markers are going crazy, moving weirdly, and jittering compared with others. If a marker isn't right, don't worry; you can come back and adjust it at any time if the camera solution fails.

Configuring Camera Settings

Before you solve the camera motion, you need to tell Blender your camera parameters. Knowing the lens you used as well as other camera settings makes it easier for Blender to calculate perspective. In the Sidebar's Camera and Lens panels, you can input information about the focal length you used to film the footage, as well as the camera sensor and other parameters.

If you press the button with three bullet points on the Camera panel's header, you'll find some presets, so you could search to see whether your camera is between those presets and that way populate the general parameters. You can also save your camera as a preset so that you can reuse those settings next time you need to track a video filmed with the same camera.

If you don't know this information, no problem. Blender has a tool called Refine that estimates that information for those situations. You use this tool in the next section.

Solving Camera Motion

The toolbar's Solve tab has options for solving the camera motion that will ultimately be in your 3D scene. You might use the Tripod option, for example. If you filmed from a tripod, your footage won't have much perspective information, so Blender calculates the camera rotation only if you enable this option.

Keyframe selection is also important. Blender needs to select two frames of the footage that will serve as a base for calculating the perspective in the rest of the frames. Those two frames should include fairly different perspectives but have a significant number of markers in common. Blender compares the perspective shift of those markers between the two frames and uses that information as a guide. When you activate the Keyframe option, Blender selects those two frames for you, or you can enter them yourself as keyframes A and B.

The Refine option is useful when you don't have information about the camera's focal length or distortion values (the K1, K2, and K3 parameters). If you enable one of those options for Refine, Blender estimates those values for you.

When you've made the appropriate selections, click the Solve Camera Motion button, and look at the header, which displays the error margin on the right side (see Figure 13.4). Blender detects the difference between the 3D camera's and the real camera's perspective information, which it determined from the markers. A tracking solve error of 0 would be perfect tracking, but perfect tracking never happens; a small amount of error always occurs. Usually, tracking is acceptable if the tracking solve error value is less than 3. The camera can have a slide effect at times (noticeable when you place the 3D objects on the real footage), and a tracking solve error value of 1 is pretty good; a value of less than 0.4 or 0.3 is considered to be a very good tracking.

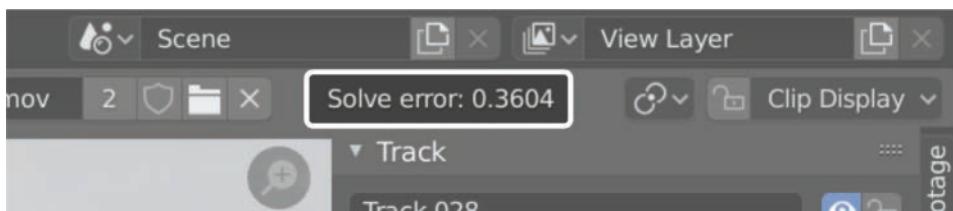


Figure 13.4 Solve Error indicator on the Movie Clip Editor's header

Although no tracking is perfect and there'll always be some amount of solve error, you can improve your results by adding more tracking points near the areas where you'll have 3D objects integrated into the real footage. This way, maybe some other areas won't perfectly match the movement, but the camera motion will better match where your 3D objects touch the real footage.

In this case, it's important to add a few markers on the floor, which Jim will be walking on, so that area fits the real camera motion as well as possible, even if the background match won't be too accurate.

There aren't a lot of easy points to track on the floor close to the camera, but you can use the perspective option for some markers and distort the patterns so that they take recognizable shapes on the floor, as shown in Figure 13.5. To avoid issues, make sure that the search area is bigger than the pattern.

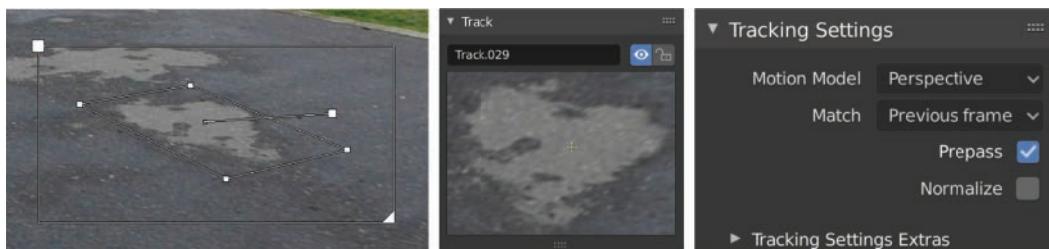


Figure 13.5 Using perspective tracking to track complex shapes in the footage. You can distort the pattern by moving its corners so that they fit some feature in the video and set it to track perspective from the previous frame. A pattern that distorts due to perspective in the video would be very difficult to track manually. Left: Marker in the editor. Middle: Sidebar's Track panel, where you can see the pattern (undistorted). Right: Tracking settings for perspective tracking on the Sidebar.

Applying Tracked Motion to the Camera

If you go to the 3D scene, you probably see nothing going on, because you still have to do one thing to make the scene work. Here are the steps to follow:

1. Select the camera, and on the Constraints tab of the Properties Editor, add a Camera Solver constraint.

2. Enable the active clip or select the clip's name from the list. Now you should see the camera and a set of little points in the scene. Each one of those points represents a marker in the Movie Clip Editor. If you can't see those points, make sure that the option for Motion Tracking is enabled in the Overlays pop-over in the 3D Viewport. (Other options change how the markers are displayed, their size, and so on.)
3. Scrub through the timeline. You should see the camera moving (see Figure 13.6).

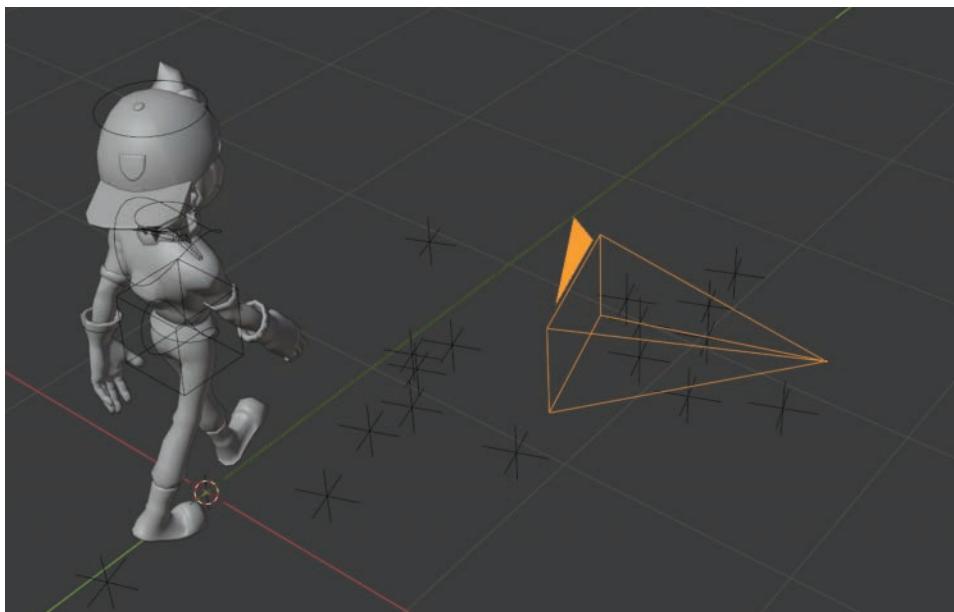


Figure 13.6 Camera motion is there, but you still need to align the camera so that the 3D scene is in place over the real footage.

Adjusting Camera Motion

Now you need to align the camera. The Movie Clip Editor offers a couple of tools for this purpose, but you can also align the camera manually. Follow these steps to use the Movie Clip Editor to align camera motion:

1. Select three markers in the footage that are placed on the floor. In the Orientation panel of the Solve tab, click the Floor button. Blender aligns the camera and all the markers in such a way that those three markers are on the floor, completely horizontal.
2. To define the scale of the scene, select two markers in the 3D scene. (It's best if you know the distance between them in the real scene, but at least make a guess.) In the Orientation panel of the Solve tab, in the Distance field, type the distance between

the two selected markers in the real scene. Click Set Scale. The camera with all the markers scales up or down to reflect that measurement in the 3D scene.

3. Click Set As Background in the Scene Setup panel, which is on the Solve tab on the toolbar. Blender applies the current footage to the Camera view as a background. Press **NumPad 0** to look at the scene from the active camera.
4. After this process, you should have the general alignment, but you may need or want to adjust it. You can move, rotate, and scale the camera manually until your 3D objects fit in the background footage where you want them. A good method is to place the 3D cursor in the origin of the scene (or where a 3D object such as the character is placed on the floor), and then rotate, move, and scale the camera and tracking points from the 3D cursor until the camera is aligned (see Figure 13.7).



Figure 13.7 The camera is aligned, and Jim is standing on the floor of the real footage.

Tip

At this point, you could click the Setup Tracking Scene button (below the Set As Background button) on the Solve tab of the Movie Clip Editor's toolbar. This option creates the nodes in the compositor, sets the clip as a background, creates a floor to make the character cast shadows on the floor, and so on. You learn how to do all these things manually in Chapter 14, “Lighting, Compositing, and Rendering.” This option works only for the Cycles render engine; however, it isn’t compatible with EEVEE.

Testing Camera Tracking

Press **Space** in the 3D Viewport to see whether the camera tracking works. The camera alignment may need some adjustments, or the camera may not be moving accurately. In that case, go back to the Movie Clip Editor and look for the frames in which the camera movement fails. Maybe a marker is moving strangely or is jumping from one point to another. Maybe those frames don't have enough markers, and you need to add more to make the frames more stable.

In any case, the process is as simple as retouching those markers or adding new ones (or even deleting a marker that you feel is moving inaccurately compared with the others or that's following a moving feature that confuses Blender). Solve the camera motion again, and realign the camera in the scene. Just keep trying until you get the camera motion to work, and don't give up!

Summary

Camera tracking can be fast and simple or tricky and frustrating. Every shot presents its own challenges, but this chapter gives you an idea of how the process works so that you can do basic camera tracking for your own projects. Keep in mind that your experience with tracking depends greatly on how you film the footage. Keep practicing, tracking as many videos as you can, and eventually, you'll understand what makes a video easy or difficult to track.

This chapter is just the tip of the iceberg in terms of what you can do with the Movie Clip Editor. You can include lens distortions, use tracking to stabilize footage, or even track objects in the video and translate their movements to objects in the 3D scene. (Some people even manage to use the tracking tools to capture facial expressions!) I hope that this chapter showed you basic tips that will encourage you to keep learning and to look for more information on the subject. In any case, you're close to finishing the project!

Exercises

1. Film a video, and track the camera motion using the techniques explained in this chapter.
2. Track the camera in a video shot with a tripod to understand how the footage looks when the camera motion is fixed.

This page intentionally left blank

Lighting, Compositing, and Rendering

Welcome to the final stage of the project! In this chapter, you light your scene to match the real footage, learn how to set everything up so that you can composite your scene with nodes, and launch the final render. Node compositing can be a little tricky to understand when you see it for the first time, but when you finish compositing a few scenes with some basic nodes, you'll like it a lot, and you'll start to understand the power that comes with using nodes. Compositing is an important, decisive part of the process, because during compositing, you tweak your scene and take it from a raw render to a nice-looking final render. You can retouch colors, add effects, mix elements, and do anything else you want.

In this chapter, you'll see how to mix the 3D character you created up to this point in the real video you used for tracking camera motion in Chapter 13, "Camera Tracking in Blender." You'll see how to do the compositing in both Cycles and EEVEE. Generally, integrating 3D models into live video is better achieved with Cycles, which is more realistic and accurate. Cycles also provides better tools for this purpose. But EEVEE can be enough in certain cases, and although it requires the use of some tricks, it can achieve a convincing result.

Lighting Your Scene

The first step, whether you're using EEVEE or Cycles, is adding lights to your scene so that the shadows the character projects on the ground match the lighting in the real footage. When you work on an animated 3D video, you can decide how you want your lights to light the scene, but when you're trying to mix a 3D object into real footage, you must make your 3D scene lighting fit with the real scene lighting as accurately as possible.

Analyzing the Real Footage

Before you add lights, carefully analyze the real footage into which you want to fit your 3D scene. Take a look at the shadows; they tell you where the light is coming from, as well as its intensity. Pay attention to how diffuse or sharp the shadows are. The color of the lights in the scene is also very important.

In the footage you'll be using in this chapter, the sky is cloudy, and the clouds act as a huge light diffuser, making shadows nearly nonexistent and very diffuse. Clouds let light pass through them, but water particles make light bounce inside them in all directions, causing shadows to not be projected from a particular direction. When you're outside on a cloudy day, you don't see many shadows—only subtle soft shadows where two objects touch (see Figure 14.1). This footage was chosen to make your first integration easier because you won't have to deal with strong shadows.



Figure 14.1 If the footage has shadows, they provide light angle, direction, and intensity. In this footage, though, the cloudy sky makes light bounce everywhere, and shadows nearly disappear.

Creating and Testing Lights

When you know about the lights and shadows in the real footage, you can start creating lights in your 3D scene to match that lighting. Again, the light settings may differ between EEVEE and Cycles, but for now, you'll just set up the common basics; later, you'll adjust the settings when you work in each render engine separately. Here, you'll find a set of tasks you perform to create the scene lighting:

- **Background's Alpha:** When you click the Set As Background button in the Movie Clip Editor, Blender automatically loads the clip as the background for the camera. By default, its opacity is 50 percent. To see the lighting integration better, you should set the opacity to 100 percent. Select the camera. On the Properties Editor's Camera Properties tab, you'll find the Background panel. Inside this panel, you can select the video/images to use as the background for the camera and also adjust some of its settings. Set Alpha to 100 percent to increase the background's opacity.

- **Floor:** You need to create a plane for the floor to receive shadows. For now, create a plane, and adjust its size to fit the area Jim is walking on and to be wide enough to receive his shadows. You can take the road in the video as a reference to create this plane.
- **Rendered viewport shading mode:** Whether you're working in EEVEE or Cycles, before you start adding lights, you should enable Rendered viewport shading mode so that you can preview the resulting lighting and shadows created by the lights in your scene.
- **Sun light:** If the real footage has defined shadows, you need a directional sun light that mimics the direction of the main light in the original scene. Just press **Shift+A** and create a sun lamp. Align the lamp, taking into account how the shadows are projected.

Next, adjust the softness of the shadows to fit the shadows in the footage. The footage you're using here has no defined shadows, so you can use a sun coming from the top and increase the value for its angle, which will make the light and shadows softer. In Cycles, you should see the results right away; in EEVEE, you probably won't, but don't worry because you'll adjust it later.

Tip

You can divide your screen to have a rendered preview of your scene on one side and the footage in the Movie Clip Editor on the other side.

- **World light:** If you have just the sun light, some parts of the character may be really dark because of the shadows. The world light can help fill those dark areas. By default, the world should have a neutral, subtle light.

You can find these settings on the World Properties tab of the Properties Editor. If you haven't changed anything before, you should see a Background shader with a gray color and a strength of 1.0. If you change the strength value to 0.0, you'll see how dark the scene is without the world lighting.

You can change the background color with a sky texture and play with it. You can also change its color to add a general light with a similar color to the real video's sky. Then increase the strength value so that the world's light fills the shadows on the character until the lighting matches the real video's lighting as much as possible.

You can keep adjusting the sun light and world settings to make the integration match the video as accurately as you can.

Showing/Hiding Objects in Render

You may have objects that you don't want to see in the scene (such as multiple heads with different facial expressions), and even though you clicked the eye icon in the Outliner to show or hide them, now that you can launch a final render, they show up in the rendered image.

The reason is that separate settings control the visibility of objects in the 3D Viewport and in the render. These separate settings come in very handy, given that you may have many objects that you don't need to see while working, but you still want them to appear in the final render.

In the Outliner's header is a funnel icon. Press **MMB** and drag the header left and right if the Outliner's width is too small. If you press the funnel icon, you'll see the Filters pop-over menu. At the top of that menu are the restriction toggles. When you enable those toggles, new icons appear next to the eye icon to the right of objects and collections in the Outliner.

You must enable the render visibility toggle, represented by a camera icon, which shows up next to the eye icon. Essentially, the eye icon controls the visibility in the 3D Viewport, and the camera icon controls the visibility in the final render.

Figure 14.2 shows the Filters menu and the icons next to datablocks in the Outliner.

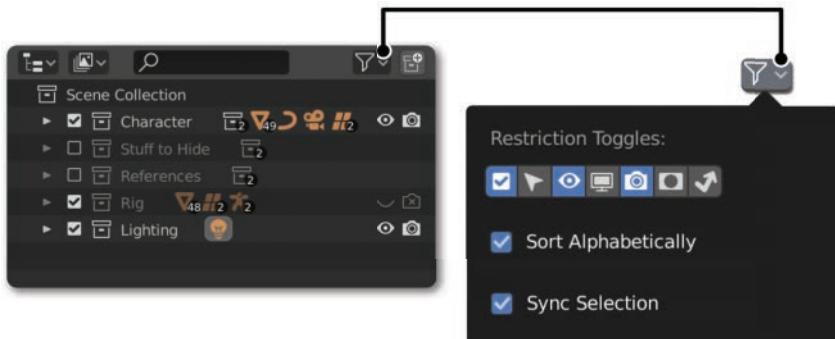


Figure 14.2 The Outliner and the Filter menu within the Outliner. The Restriction Toggles in the Filter menu show or hide buttons that show up next to the objects in the Outliner. Those buttons let you control the object's visibility in the 3D Viewport and in the render, for example.

Just find the objects you want to hide in the Outliner (or an entire collection, if you moved those objects into one), and disable the camera icon to hide those objects during the render.

Testing EEVEE and Cycles

Up until now, the settings are common to EEVEE and Cycles. To make sure everything is working, you can switch the render engine in the Render Properties tab of the Properties Editor.

My recommendation is that you work mainly with EEVEE to get fast feedback when you're setting things up and switch to Cycles (which is generally slower) to test how things work.

To see what's going on, remember to do this entire process in Rendered viewport shading mode in the 3D Viewport.

After learning the basics of working with nodes, you'll start finishing the scenes in each render engine separately to adjust the appropriate settings.

Note

At this point, everything is pretty much guesswork. But don't worry if you don't get everything perfect on the first try. Later, in the compositor, you'll see much more clearly whether your lighting fits the lighting in the real footage; if it doesn't, you can always adjust it and render again until the integration looks good. Remember that making something cool isn't a one-time process. At some point, you'll need to go back and forth, making retouches until you reach the result you like.

Using the Node Editor

Before you dive into setting the scenes in EEVEE and Cycles to integrate the 3D models into the real footage, it's important to learn the basics of nodes. In this section, I briefly show you how to use the Node Editor, and discuss what nodes are and how they work. After this introduction, you'll be ready to carry on with some basic compositing.

Compositing

Usually, a scene doesn't come out as you need it in the raw render, so you need to process it to make it look as good as possible. Sometimes, you need to render different elements on different layers and then compose them together in the compositing stage. Maybe you just want to place your 3D objects in a photo or real-world footage, so you need to mix those images with your render and adjust colors to make them match together. You can do such things in 2D image-editing software, such as Adobe Photoshop or GIMP, but you can use Blender compositing nodes for the same purpose.

You have two main ways to do compositing:

- **Do the compositing before the rendering.** You take a test render, composite it in the Node Editor, and then launch the final render (even an animation) with the effects of the compositor applied. For this purpose, you use the scene render as an input.
- **Do the raw render of elements and then load those image sequences or videos into the compositor to integrate them.** Suppose that you want to color-correct a video. Just load the video into the compositor, color-correct it, and render it. You don't need to touch the 3D Viewport and definitely don't need to render a scene again for such small retouches.

Understanding Nodes

When you take a simple render (a raw render, with no compositing involved), your scene is the input, and the output is the same as the input. When you enable the use of

nodes, the input and the output are connected, but you can add nodes between them that apply effects and changes over the input before it reaches the output, thereby modifying the final result. The modifications can be as simple as making color corrections or as complex as adding visual effects or mixing renders.

The structure created with nodes is referred to as a *node tree*, and it gets its name because it has one end but many branches can come out of it. (In fact, the different groups of nodes that are mixed to achieve the final result are called *branches*.) You can also think of nodes as rivers; many small rivers converge, getting bigger and bigger, until they create the final river that gets to the ocean. The small rivers coming from the mountains are the inputs, and the output is the big river arriving at the ocean.

Figure 14.3 shows how a basic node tree can evolve as you add nodes.

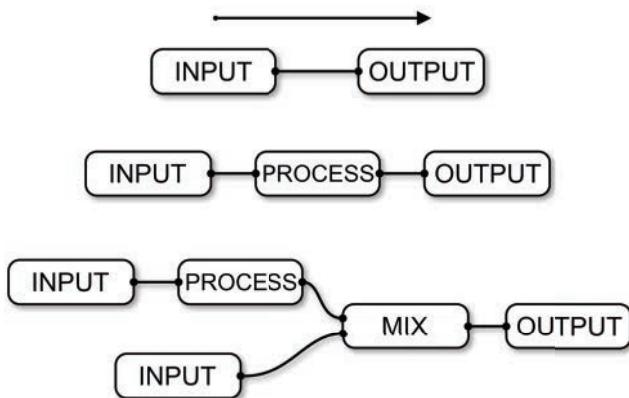


Figure 14.3 Three node trees, which could be the same in different stages of their evolution: a basic setup, which you get when you enable node editing (top); a modification added to the input before it reaches the output (middle); and a second input mixed into the result (bottom).

A node tree is always processed from left to right. Although you will find many node categories in the Node Editor, it may help to think about three main classes of nodes:

- **Input:** These nodes introduce information to the node tree. An input node can be an image, a video, or a render from the 3D scene.
- **Process:** These nodes take the information from the input nodes and modify that information. They can also mix information from different nodes to process them further in the node tree.
- **Output:** These nodes collect the previously processed information and produce the final result, which can be saving the resulting image in your hard drive.

You can always add nodes between other nodes. Consider the third example in Figure 14.3. Each of the inputs might be a different render of a different layer of the scene. The first input has a process going on before it's mixed with the second input; that process could be a color correction, for example. Suppose that you want to make your entire render look more reddish or have more contrast. You could add a color-correction node between the mix node and the output to affect all the previous nodes after the mix happens.

If you still don't get how nodes work, you need to get your hands on them to understand their inner workings. As you continue this chapter and create your first node tree, you see the result with your own eyes and understand how your changes affect the result.

Studying the Anatomy of a Node

Before you start using the Node Editor, you need to know the parts that make up a node. Figure 14.4 dissects an RGB Curves node, which you use to make color corrections in the nodes you connect to its input. (I changed the colors of the Node Editor in User Preferences to improve readability, so yours will look different.)

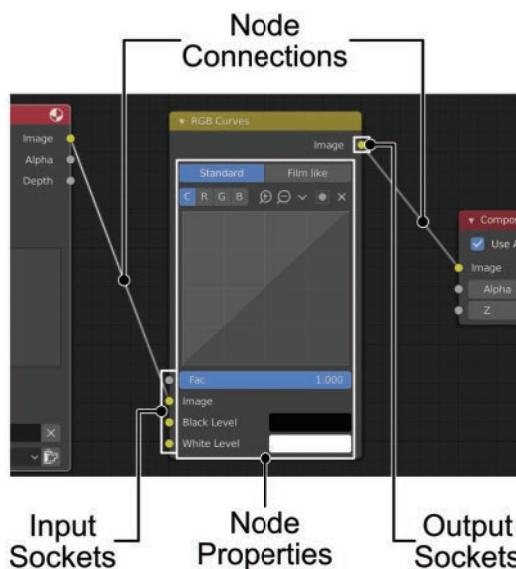


Figure 14.4 The main parts of a node: connections, input sockets, properties, and output sockets

- **Sockets:** *Sockets* are the little colored dots on the left and right sides of a node. They support the connections between nodes. The ones on the left are input sockets, and the ones on the right are output sockets. The color tells you what each connector is for. Gray is for values (or grayscale images); yellow is for RGB

(red, green, blue) images and colors; and blue is for vectors (three values: X, Y, and Z). A given type of output should always be connected to the same color input of the next node, and you can use some converter nodes to convert the output to a different type if you need to. Some types are converted automatically if you connect them. An RGB output converts to grayscale when it's connected to a gray input, for example, and RGB values turn into XYZ values when you connect a yellow output to a blue one (and vice versa). Next to those sockets is text that tells you what that socket should receive (if it's an input) or what it's sending (if it's an output).

- **Node properties:** Each node has different properties and is used for different purposes. You find all the properties inside the node itself. These properties define what operations are done to the information received from the input sockets. In the case of the RGB Curves node, the properties control how the colors will be processed before they're sent to the next node through the output socket.
- **Node connections:** A node does nothing by itself. Every node needs other nodes to work, which is why nodes are connected. The way and order in which you connect nodes define the result.

In the previous section, I mentioned three main classes of nodes: input, process, and output. You can identify which of these classes a node belongs to depending on the sockets they have. Input nodes have only output sockets, as they generate or load information. Process nodes have both input sockets and output sockets, because they need to be fed information that they will modify and then send to the next node. Output nodes are the end of the line, so they have only input sockets.

Using the Node Editor

In this section, you learn the basic controls of the Node Editor, as well as how to create and interact with nodes, connect them, and so on. Figure 14.5 shows the Node Editor.

Getting Started with Nodes

You can use nodes for different purposes in Blender. If you check the list of editors, you'll see several editors that use nodes, such as the Shader Editor, the Compositor, and the Texture Node Editor. Although the way they're used is similar, as they all work with nodes, the nodes themselves vary.

In the Shader Editor, for example, the nodes are meant to create materials and load textures, whereas in the Compositor, you use nodes to mix layers and add effects to the scene's final render.

When you enter the Compositor, you still don't see anything. Before you start using nodes, you need to enable them in your scene. Enable the Use Nodes option on the Compositor's header, and Blender displays basic node setup: the render (a Render Layers node) plugged into a Composite node (the render output). Nothing interesting is going on right now; the render goes out of Blender the same way it's generated.

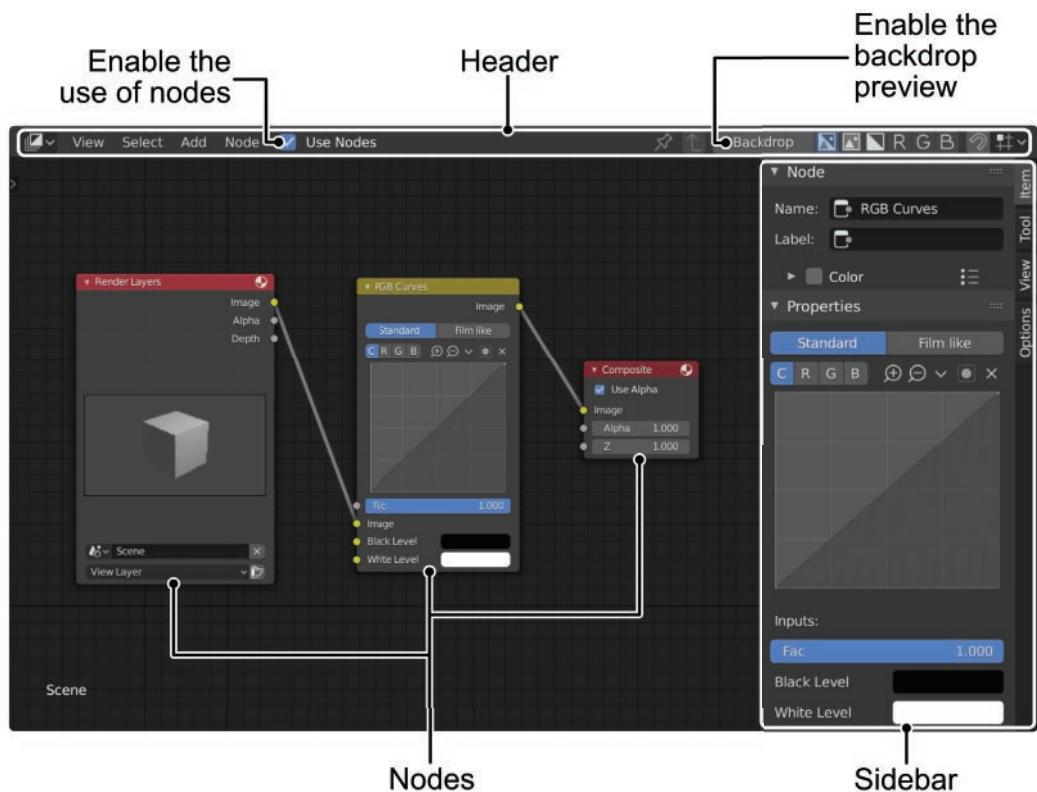


Figure 14.5 The Compositing Editor, one of the editors that use nodes

The Compositor is ready for you to start playing with nodes. If you launch a render now, it uses the node tree you have in the editor. If you need to launch a raw render without using the nodes, disable the Use Nodes option on the Compositor's header.

Navigation is pretty straightforward, as it uses Blender's standard controls. Press **MMB** to pan, and use the scroll wheel or press **Ctrl+MMB** to zoom in and out. General shortcuts for other editors also apply, such as pressing **NumPad .** to focus the view in the selected node or pressing **Home** to zoom out and show the entire node tree.

Creating Nodes

You have three ways to create nodes in the Node Editor:

- **Add menu:** On the Compositor's header, choose the type of node you want from the Add menu. Left-click the node to create it, move the mouse to the desired position where you want the node, and left-click again to place the node there.
- **Shift+A:** If you press **Shift+A** inside the Node Editor, the Add menu appears at the mouse cursor's position. Click the node type you want, drag it with your mouse, and left-click to drop the node in that place.

Using either of these methods, before you left-click to place the node in the editor, you can cancel by pressing **Escape** or **RMB**.

Tip

When you place a new node in the editor, if you drag it over a connection between two nodes (in the case that the node has available connections to do so), Blender highlights the connection, and when you drop the node, Blender automatically connects it between those nodes. This feature is a big time-saver!

Connecting and Manipulating Nodes

The core of working with nodes is connecting them to make them interact. It's also important to know how to move nodes so that your node tree is always organized. Otherwise, you can end up with a lot of overlapping nodes, which makes your work harder, as it becomes difficult to find specific nodes and understand their purposes. Here are some common controls that you use to manipulate nodes in any of the editors that use nodes:

- You can select nodes by pressing **LMB**. Drag nodes by left-clicking and dragging.
- Right-click to show the node's contextual menu.
- If you select several nodes by pressing **B** (box selection) or **Shift+LMB**, you can move them, rotate them, and scale them by pressing **G**, **R**, and **S**.
- Left-click and drag in an empty space to perform a box selection (also done by pressing **B**).
- To connect nodes, left-click and drag from an output socket of a node to the desired input noodle of another node.
- If a node has multiple input sockets, and you want to switch a connection among them, left-click and drag from one socket to the other.
- To remove a connection, left-click the input socket and drag the line connection to an empty space.
- To remove one or more connections quickly, press **Ctrl+RMB** and draw a line over the connections you want to cut. When you release **RMB**, Blender removes the connections under the cutting line.
- Select one or more nodes and press **M** to mute them. This method is a nice way to see in the preview how a node changes the image. Nodes become grayed out with a red line through them (if they have connections on both sides). Press **M** again to enable the muted nodes again.
- You can duplicate nodes or groups of connected nodes by pressing **Shift+D**. You can also copy and paste nodes with **Ctrl+C** and **Ctrl+V**.
- You can press **H** when you select a node to collapse it if you don't need to access its properties so that it takes less space. Press **H** again to expand the node. Press

Ctrl+H to hide unused sockets; this method is useful for nodes with many sockets to make the node tree less confusing. Press **Ctrl+H** to show those hidden sockets again.

- Detach a node and leave the connection between the previous and next nodes intact by holding down **Alt** while dragging the node.
- Select one or more nodes, and press **X** to remove them. Press **Ctrl+X** to delete them while keeping the connections between the previous and next nodes. As always, remember that **Ctrl+Z** lets you undo the latest actions if you make a mistake. Press **Shift+Ctrl+Z** to redo the last undo.

You can do many more things when working with nodes. I recommend that you take a look at the Node Editor's menu, where you'll learn about more options and keyboard shortcuts to execute them.

Previewing the Result

You don't need to work blindly in the Compositor. You can have an image showing real-time updates of what you're doing in the Compositor. To enable a preview, create a Viewer node from the Output nodes category. Add the node to your node tree, and connect the output of the node you want to preview to the input of the Viewer node.

An even faster method is to press **Shift+Ctrl** and left-click the node you want to preview. This method automatically creates a Viewer node and connects it to a specific node. Pressing **Shift+Ctrl+LMB** on any other node connects that node to the Viewer node so that you can switch the point of the node tree you want to preview really fast.

Tip

Most of the functions explained in this chapter apply to nodes everywhere. Pressing **Shift+Ctrl+LMB** to preview the output of a node, however, works only for the Compositor's nodes. An add-on that comes with Blender called Node Wrangler adds many new functions to work with nodes. One of those functions allows you to press **Shift+Ctrl+LMB** to preview the output of a node in the Shading Editor's nodes as well.

The Viewer node displays a preview of the node that is connected as an input.

When you have a Viewer node in your node tree, you have two options for previewing your work in the compositor:

- **Compositor Backdrop:** In the Compositor, enable the Backdrop option on the editor's header. Blender shows the resulting image behind your node tree, in the background of the Node Editor Workspace. Press **Alt+MMB** to pan, **V** to zoom out, and **Alt+V** to zoom in. Alternatively, you can drag the backdrop's gizmo controls in the corners and center to move and scale the image.
- **Image Editor:** Although the backdrop allows you to see everything in the same window, it can be very distracting at times, and the nodes on top of the image

don't always let you see what's going on (especially in a complex tree). In such a case, or if you want to see the result on a secondary monitor, you have a nice way to preview your work. Open Image Editor, and select the Viewer Node output from the images list on the header. You see the Viewer node preview as though it were an image in the Image Editor.

Now any changes that you perform in the node tree will change the result in that preview, whichever preview method you've chosen. (You can also use both simultaneously.)

Caution

Don't forget that to see what you're doing in the Compositor, you have to have rendered your scene (unless you're working with already-rendered images or a video instead of the current 3D scene). If you render your scene, close Blender, and open the scene again, the renders need to be made again, as they are stored in a temporary buffer. Keep that fact in mind, especially when you're working on big, complex scenes that take a long time to render. You can always save the render in an image file and work on the compositing with that file; then, when you're done, you can replace the image file with a Render Layers node.

Rendering and Compositing Your Scene in Cycles

Compositing in EEVEE and Cycles is quite different. The options for compositing in Cycles are better equipped for compositing than those in EEVEE, generally. An action such as creating an object that renders only the shadows it gets from other objects requires just one click in Cycles but a more complex setup in EEVEE. Each render engine has limitations.

Blender has many compositing options, such as View Layers, which allows you to separate the objects in the scene in layers, so that you'll have more control while compositing and adding effects. In this chapter, however, you'll perform very basic compositing to integrate your walking character into the real footage, but be aware that the possibilities go much further.

Before you jump in, make sure that you have Cycles selected as your render engine. You can do so on the Render Properties tab in the Properties Editor; choose the render engine you want to use from the Render Engine drop-down menu.

It's important to start with Cycles, as some of the parts are the same for EEVEE. After compositing with Cycles, you'll learn the differences that make the scene work in EEVEE, and end up with a scene that you can render in both render engines and get similar results.

In both render engines, you need to capture the shadows of the scene's floor so that Jim seems to be walking on the ground of the real video, starting by creating a shadow catcher in Cycles.

Creating a Shadow Catcher

A *shadow catcher* is an object that exists in the scene, but in the render, you see only the shadows that other objects cast on it. You can turn any object into a shadow catcher; then that object becomes transparent, receiving shadows only from other objects. You can create a plane for the floor and set it as a shadow catcher, for example. It will be transparent (letting you see the real video's floor behind) but receive shadows from other objects in the scene, making it look as though the shadows were projected onto the real floor in the video.

In Cycles, creating a shadow catcher is very easy. To turn the floor plane into a shadow catcher, follow these steps:

1. Select the floor plane.
2. On the Object Properties tab of the Properties Editor, look for the Visibility panel and enable the Shadow Catcher option. If you're in Rendered viewport shading mode in the 3D Viewport, you should see how the plane becomes transparent, but the shadows under Jim's feet are visible on top of the background. If you set up the camera background to show the real video in Chapter 13, "Camera Tracking in Blender," you have a pretty good view of the integration between the 3D character and the real video.
3. Make some more adjustments. You might change the color of the floor plane material so that it better resembles the color of the real video's floor, for example. In Cycles, light bounces, so the color of the floor will influence the light Jim receives from the sun light after it bounces on the floor.
4. Finally, now that you can see the integration, you can adjust the parameters of the sun light and the world lighting to make them match the real video's lighting as accurately as possible.

Now you have Jim's shadows in place. The next step is setting up the final render so you can do the compositing.

Rendering in Cycles

In this section, you'll see some of the settings that you may want to tweak so that Jim is rendered properly before compositing. Most of these options are on the Render Properties tab of the Properties Editor.

First, you need to adjust the samples. Remember that the more samples, the better quality and less noise in the render. In Render Properties, look for the Samples panel, and increase the samples number. There is no magical number; depending on your scene, the lighting, the complexity, materials used, and so on, more or less samples will be needed.

I recommend that you increase bit by bit and check the results to see when they're good enough. In my case, I used 300 samples for the render. You can change the

samples in the 3D Viewport while in Rendered viewport shading mode to see how many samples do a good job and then add that number to the render samples. In general, it's better to have a low sample number in the 3D Viewport to work faster.

The more samples, the better the quality, but also the longer the render time.

Note

Blender has several methods for noise removal in renders. I won't go into the details here, as using one or the other may depend on things like the hardware you're using. But the most direct method you can use is the integrated noise removal feature in Blender.

Go to the View Layer Properties tab of the Properties Editor, find the Denoising panel, and elect the check box next to the panel's name. That's it! This option should be enough to fix the smaller noise.

Keep in mind that denoising is not a magic solution; it needs enough samples to work well. But it will let you use fewer samples for lower render times and fix the noise. I recommend that you try different sample amounts while enabling and disabling the denoiser to see how the results vary and understand how this process works.

You can enable Motion Blur. On the Render Properties tab is a panel called Motion Blur that creates a blur in fast-moving parts of the 3D scene. You can adjust the amount of the effect with the shutter slider inside the Motion Blur panel. You can see the motion blur effect only when launching the render (which you can do by pressing **F12**); it won't be visible in the 3D Viewport.

The real video will most likely have some motion blur when the camera moves fast, so adding this effect to the 3D objects as well will improve the integration result.

You need to make the background of the world transparent so that while compositing, you can add the real video behind Jim and the floor shadows. If you launch a render right now, the background will have the world's colors.

Fixing this problem is very simple. On the Film panel in the Render Properties tab of the Properties Editor, enable the Transparent option. If you launch a render now, you should see Jim over a background full of dark squares; those squares represent that those parts of the render are transparent.

Why doesn't the background appear in the render too? Well, you can have different backgrounds for different viewports in the same scene; also, backgrounds in viewports are meant as references only. In the render, you have to add the background in other ways. One of those ways would be to load the real video in the world background that will be rendered. You'll use the compositing method, in which you load the background behind the character after you render.

Finally, on the Output Properties tab of the Properties Editor, make sure that the dimensions are set to the size of the full HD real video: 1920px in X and 1080px in Y.

You need a render to start compositing, so you can press **F12** in anticipation of that last stage of the process. The result should look similar to Figure 14.6.

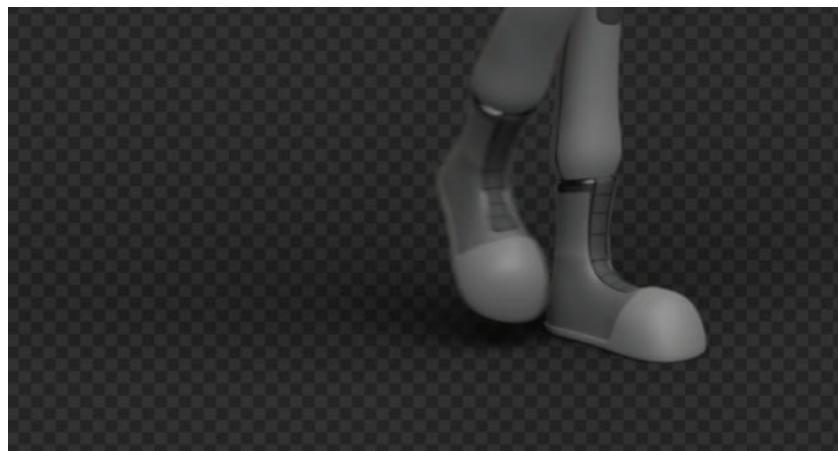


Figure 14.6 The render result so far. The background is filled with a pattern, indicating that the background is transparent, and the shadow is visible under Jim.

Node Compositing in Cycles

Let's start the compositing! After you launch the render (by pressing **F12** or from the Render menu), you're ready to start. You can open a Compositor editor in any area or switch to the Compositing Workspace (with the tabs at the very top of the interface). Here are the steps to follow to integrate the real video with the 3D render by using compositing nodes. You see the node tree in Figure 14.7.

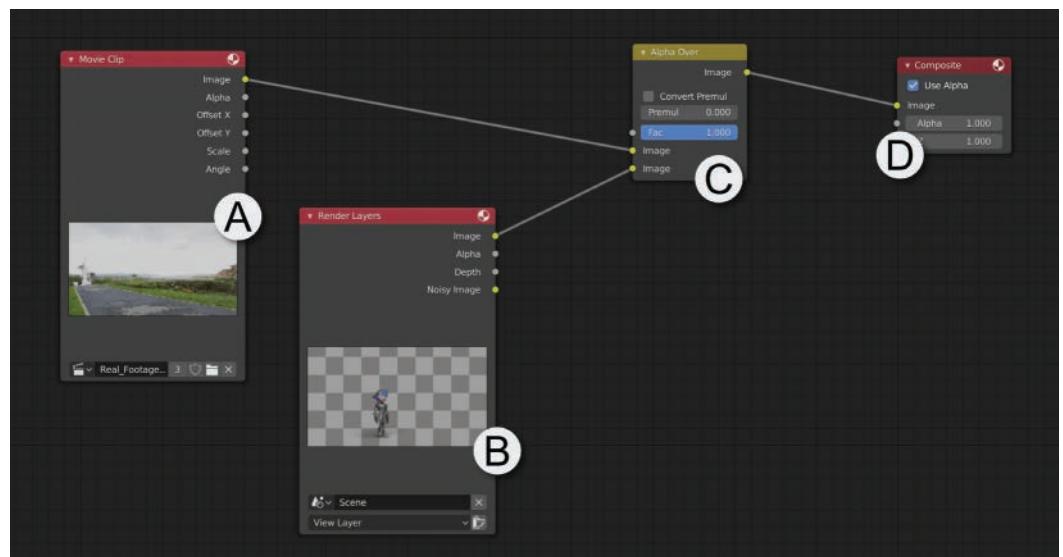


Figure 14.7 The compositing node tree for mixing Jim with the real video

1. Turn on the Use Nodes option in the Compositing node editor (on the Node Editor's header).
2. You should see two nodes: Render Layers (B) and Composite (D). Render Layers inputs the render in the compositor. Composite is the output of the compositing and always the last node in the tree. You can enable the backdrop from the header. Press **Ctrl+Shift+LMB** on the Render Layers node to show its result in the background of the Node Editor. After that, press **Ctrl+Shift+LMB** on any node that has output sockets to connect it to the viewer node (which controls what is displayed in the background preview).
3. You need to load the video clip for the background. Press **Shift+A**, and from the Input category, select a Movie Clip node (A). If the clip isn't selected already, choose it from the list at the bottom of the node. Keep in mind that this node loads a movie clip as long as it has been loaded into the Movie Clip Editor. In this case, you loaded the video in the Movie Clip Editor to perform the camera tracking, so it should be in the list of clips in the Movie Clip node.
4. Next, you need to mix the contents of the Movie Clip node and the Render Layers node. Typically, you could use a Mix node (Color nodes category) for this purpose, but because the Render Layers node is a render with alpha (transparency in the background), an Alpha Over node (C) will work better. Press **Shift+A**, and inside the Color nodes category, select Alpha Over.
5. The Alpha Over node essentially accepts two RGB (color or image) inputs and mixes them. You will find three input sockets: one for a factor (Alpha) and two for images. The first image input socket will be the background, and the second input socket will be shown on top of the first, respecting the alpha.
6. Connect the Image output from the Movie Clip node (A) to the first Image input in the Alpha Over node (C) and the Image output from the Render Layers node (B) to the second Image input of the Alpha Over node (C).
7. To preview what the Alpha Over node is doing, press **Ctrl+Shift+LMB** to connect it to the Viewer node, and display its contents in the Node Editor's backdrop. You should see Jim and his shadows on top of the real video.
8. To finish, connect the output of the Alpha Over node (C) to the Image input in the Composite node (D). This step makes the result produced by the Alpha Over node the final output of the composited render.

Very basic compositing is done. If you press **F12** to render now, Blender will know that you have compositing nodes in the scene, and it will do what the nodes order: first load the video and render the scene, then mix them together, and finally show the composited result (whatever comes into the Composite node). In Figure 14.8, you can see the resulting render.



Figure 14.8 Final Render in Cycles

You could complicate and improve the result by adding other nodes in between this basic structure. You could add RGB Curves nodes between the Movie Clip and Alpha Over nodes to tweak the colors of the background before it's mixed with Jim's render, for example. The possibilities are endless. You could end up with hundreds of nodes to add effects and achieve whatever result you desire.

Rendering and Compositing Your Scene with EEVEE

In EEVEE, the process differs a bit from that of Cycles, but some parts are the same. The tricky part is that there is no Shadow Catcher option in EEVEE.

Switch the render engine to EEVEE from the Render Properties tab of the Properties Editor to start the process, and launch a render. Even though you set the floor plane as a shadow catcher in Cycles, the plane is still visible in the EEVEE render.

Creating a Shadow Catcher in EEVEE

In Cycles, the Shadow Catcher option exists because Cycles is a path tracer, and it can differentiate between different types of light rays, which makes it possible to isolate rays that are projecting shadows. In EEVEE, however, that's not a possibility.

An alternative way to get a similar result to a shadow catcher relies on creating a material that turns a material and its shadows into a black-and-white image that you can use as an Alpha map for the floor. In Figure 14.9, you can see the node setup used for the floor material.

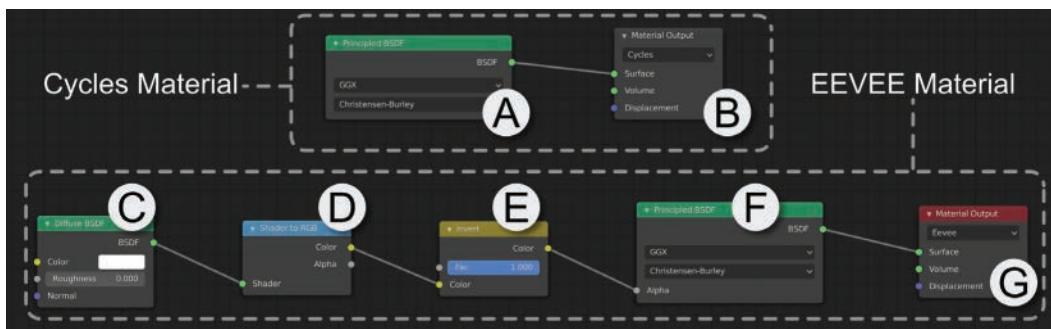


Figure 14.9 The node tree to create a shadow catcher material for the floor plane in EEVEE. There are two separate node trees in this material: one for the Cycles material (top) and a second one for the EEVEE material (bottom).

In Figure 14.9, the node tree for the floor material has two separate node trees. This is possible because you can have different outputs for Cycles and for EEVEE. Both render engines work differently, and some materials may produce different results, so Blender allows you to create two different node setups and show one or the other, depending on which render engine you’re using.

Follow these steps to create this node setup and make the floor work as a shadow catcher in EEVEE:

1. Select the floor, and open a Shader Editor.
2. If you created a material for the floor plane in Cycles to adjust its color, you should see a Principled BSDF node (A) connected to a Material Output node (B). If you didn’t, create a new material for the floor. Those same nodes should appear in the Node Editor.
3. This step is optional, but it may make working in the node editor more comfortable. Select the Principled shader (A) and press **Ctrl+H** to hide all the unused sockets so that the node takes less space. If you need to see all those sockets again, select the node and press **Ctrl+H** again.
4. In the Material Output node (B), choose All from the drop-down menu, and select Cycles. Now this material works only while you’re using Cycles as the render engine.
5. Now you’ll start building a separate node tree that works exclusively with EEVEE. Create a new Material Output node (G), press **Shift+A**, and select Material Output within the Output category. (You could also duplicate the previous Material Output node for the Cycles material by pressing **Shift+D**.) In this case, select EEVEE as the output render engine. The nodes that you’ll be creating from now on will be connected to this output.

6. Create a Diffuse node (C), pressing **Shift+A** and selecting it within the Shader category. Set its color to be white. This material is a white material that gets shadows from Jim, creating a grayscale result between the surface and the shadows.
7. Create a Shader to RGB node (D), pressing **Shift+A** and selecting it from the Converter category. Connect the Diffuse node (C) output with the Shader input in the Shader to RGB node (D). This node turns the shader connected to it into an RGB image, allowing you to use that result as though it were an image or texture instead of a shader.

Now you have a white material with shadows, and you're turning it into an image. This image is white, with dark shadows. You need the opposite to be able to use this image as an alpha for the new material.

8. When you're working with alphas, white represents opaque areas, and black represents transparent areas. That's why you need to create an Invert node (E) that will take the Color output generated with the Shader to RGB node (D) and invert the colors. You can create an Invert node by pressing **Shift+A** and selecting it within the Color node category.
9. Almost done! You have the Alpha texture that will make the shadows opaque and the rest of the surface transparent ready. You need a shader to use it with.
10. Create a new Principled BSDF (F), pressing **Shift+A** and selecting it from within the Shader category. Now connect the output from the Invert node (E) to the Alpha input socket in the Principled Shader (F). Choose a dark color for the Principled Shader (F) so that the opaque areas, which represent the shadows, are dark.
11. Connect the Principled BSDF (F) output to the Material Output node (G). You won't see it working yet because the material is not set up to allow the visualization of transparency effects. Go to the Material Properties tab of the Properties Editor, find the Blend Mode option in the Settings panel, and choose Alpha Blend to enable the material's transparency effects.

By now, if you have the 3D Viewport in Rendered viewport shading mode, you should see the result.

Here's a summary of what the nodes you've just created do:

- The Diffuse node (C) is a white shader that gets the shadows from the scene's objects.
- The Shader to RGB node (D) turns the Diffuse node's result into an image texture.
- The Invert node (E) takes the image created by the Shader to RGB node (D) and inverts its colors, creating an image that is all black, in which the shadows are white—just what's needed for a transparency map in which the shadows are opaque.

- The Principled BSDF node (F) takes the image created by the Invert node and uses it as its Alpha, making the parts with shadows opaque and the rest of the surface transparent.
- The Material Output node (G) gets the result from the Principled BSDF node (F) and turns it into the final material result. This result is shown only when you're using EEVEE.

As you can see, creating a shadow catcher in EEVEE can be more tricky than just enabling a check box in Cycles. You could potentially add more nodes in the mix to adjust the result, change the shadow colors, or change their intensity. That's the power of using nodes.

Rendering in EEVEE

You have to make some adjustments to ensure a good result with EEVEE. You may have made some of these changes during render tests of Jim's shading in previous chapters. If not specified otherwise, you'll find these settings on the Render Properties tab of the Properties Editor:

- In the Sampling settings, you can increase the samples in both the render and the 3D Viewport. More samples will give you better-quality results, especially in shadows, because they are soft and require more calculations to look soft. 120 samples should be fine.
- Enable Ambient Occlusion and adjust the settings to your liking. You can reduce the factor if the effect is too strong for the scene.
- Enable Screen Space Reflections so that the reflective objects get reflections from objects around them, not only from the lighting and environment.
- In the Shadows panel, increase the resolution, especially that of the Cascade Size, to improve how accurate the shadows are. Also, make sure that you enable the Soft Shadows option; otherwise, regardless of the light-angle settings, shadows will be hard.
- In the Film panel, enable the Transparent option. This option is equivalent to the Film option in Cycles, making the background of the scene transparent.

When all these settings are in place, you should be ready to jump to the compositing stage.

Compositing in EEVEE

This stage will be very quick because you've already done it! Generally, you'd have to follow the same compositing process for EEVEE as you did for Cycles, but given that both engines are very compatible, Cycles compositing should be working in EEVEE as well. If you press F12 now, the render should launch and the compositing you did for Cycles should give you similar results in EEVEE.

The Render Layers node renders the scene, and with the other nodes, the scene's render is overlaid on the real video clip. Render Layers renders whatever you have in the scene with the render engine that you have selected for that moment. Right now, the scene is set up in a way that it renders Jim with shadows on a transparent floor in both Cycles and EEVEE, so the compositing is the same as the one for Cycles.

In Figure 14.10, you can see the resulting render in EEVEE with the compositing already applied.



Figure 14.10 Final render in EEVEE

Exporting the Final Render

Whether you do it with Cycles or EEVEE, you need to learn how to export the final composited result with images and animations. For a single image, this step isn't too important, as you can save the image from the Image Editor when the render is done by pressing **Shift+Alt+S** or choosing the option from the Image menu on the editor's header.

In animations, though, when a frame finishes being rendered, Blender deletes it from temporary memory to start rendering the next one, so saving the rendered frames requires you to define beforehand where they should be saved.

Setting the Animation Output

In the Output panel on the Output tab of the Properties Editor, select the format you want to export the images in and the output folder where you want renders to be saved. For the format, I recommend PNG; the quality is better than JPG and not as good as TGA or TIFF, but the output takes up less space on your hard drive.

Tip

You can save the animation as a video by choosing a video format; the available formats may depend on the codecs installed on your system. But I recommend doing that only for quick render tests. If you expect the render to take a few hours or even days, it's better to export the animation in an image sequence. What you get, instead of a video, is a series of JPEG, PNG, or TGA images—one for each frame of the animation.

This method has several benefits. If the render fails at some point, you won't lose the rendered frames, whereas with a video, the entire video would be corrupted. Also, videos are usually highly compressed, so with images, you get full quality and then can convert the image sequence to a video quickly, as it's much easier and faster to render images than an entire 3D scene. (You can even do this in Blender, loading the sequence into the Video Sequence Editor, or as a clip in the compositor, rendering it in a video format.) This way, you have a sequence of uncompressed images and can compress it into a video later.

Launching the Final Render

You have only one thing left to do: Launch the render! Select Render Image or Render Animation from the Render button on the main menu at the top of the interface.

If you're becoming a pro with Blender, use the keyboard shortcuts: **F12** for a still-frame render and **Ctrl+F12** to render the entire animation. Figure 14.11 shows the results in Blender Render and Cycles. As you see, there are only subtle differences between Cycles and EEVEE in a basic scene that doesn't require advanced effects.



Figure 14.11 The final results with Jim integrated into the real footage in Cycles (left) and EEVEE (right)

Summary

Compositing is technical but also leaves room for you to be very creative. I hope that now you understand the basics, know how nodes work, and know why compositing has such a big effect on the final result.

You've come a really long way to get here, and you've completed this sample project. Congratulations! As you can see, integrating an animated character into a real video takes a lot of work and involves a lot of different skills: modeling, texturing, rigging, shading, animation, camera tracking, compositing, and so on. But all these skills can be really fun to use and learn, and they open up a lot of possibilities.

Now that you've gotten your feet wet, I encourage you to keep looking into the parts of the process you liked most. The goal of this book is to show you the entire process without digging too much into the details so that you can decide which areas to dedicate more time to. You may want to specialize in some areas, or you may like the entire process and want to become a generalist.

Exercises

1. Why is compositing so important?
2. What objects are meant to render only the shadows they receive?
3. What are View Layers for?
4. What node would you use to merge images?
5. Which node of your tree should you connect to the Composite node?

This page intentionally left blank

VII

Keep Learning

15 Other Blender Features

This page intentionally left blank

Other Blender Features

You've seen what Blender can do, but you've just scratched the surface! Blender has much more to offer than what you see in this book. I touched on the basics and covered only a few of the advanced tools and features. In this chapter, I discuss other features that Blender provides in case you're interested in learning more about them.

This chapter is not meant to be a manual or to show you how to use these features. It describes what tools are at your disposal so you can decide whether you want to learn more about them.

Simulations

You can perform several types of simulations in Blender to simulate real physics with different types of objects, such as particles, hair, fluids, or rigid bodies.

Particles

Particles are useful when you need to create and animate a lot of objects that behave similarly. Imagine snow, rain, or leaves falling. Instead of animating each snowflake or raindrop individually, create an emitter geometry, and on the Particles tab of the Properties panel, add a particle system to it.

From there, you can set the number of emitted particles, their time range, their behavior, their physics (such as gravity, weight, and friction), and so on. You can also select other objects and set them as obstacles so that particles collide with them. You can create forces such as wind, vortex, and turbulence to make the particles behave in certain ways. You can also use particles to simulate fluid substances.

Hair Simulation

Hair is a subset of particles, as you're actually creating hair particles. If you create a particle system and set it to be a hair emitter, instead of creating normal particles, the system creates a lot of hairs on the desired surface and grows strands from there.

Later, when you have a hair-particle system on the selected object, you're able to switch to Particle Mode in the 3D Viewport, which allows you to grow, cut, and comb the hair to create a hairstyle for your character.

When the hair is in place, you can run a simulation to make the hair follow the character's movements and automatically react to gravity and objects that touch it.

This hair-simulation feature isn't limited to characters. You can use it to spray a lot of objects over a surface, such as adding trees to a forest or simulating grass in a field.

Cloth Simulation

If you want to create anything from clothes for a character to banners or bedsheets, cloth simulation is the way to go. Don't worry about creating wrinkles or folds manually; on the Physics tab of the Properties Editor, turn the cloth mesh into a Cloth object, and click Play.

You can control the properties of the cloth to make it behave like a specific type of material. You also can set other objects as obstacles, and the cloth will react to them.

Cloth simulation even works in real time, so if you click Play and try to grab the cloth, it reacts to your touch and to the objects that collide with it.

Rigid and Soft Bodies

Similar to the cloth simulation, this feature is for rigid and soft objects. If you need to make a house fall down or destroy a wall, divide the object into pieces, and in the Physics panel, make those pieces rigid bodies. You can bind the objects to one another to define how they behave and limit the range of movements they can perform to simulate the force of gravity on them.

When you need to make an object behave as though it has weight, you can run a rigid-body simulation; Blender simulates gravity and object interactions in a realistic manner for you. Also, you can define other objects as obstacles so that the rigid or soft bodies collide with them.

Soft bodies are similar to rigid ones, except that they can be deformed. Using soft bodies, you can simulate an object behaving like jelly, for example.

Fluids Simulation

Blender includes MantaFlow since Blender 2.82, which is a system that allows you to simulate fluids, such as fire, smoke, and liquids.

You can let objects fall into a pool and create splashes, as well as many other simulations of a fluid's properties. Just play with the different types of fluid objects to add liquid to the simulation, subtract it, have it collide with other objects, or even use it to form shapes.

You can create these effects and preview them in real time in the 3D Viewport. (Remember that high-resolution simulations require powerful hardware and can be slow to process.) You can adjust parameters for these effects and control how the fluids, fire, and smoke behave, along with how they're rendered.

2D Animation

Blender is 3D software, but in the latest versions, some interesting 2D capabilities have been added, thanks to Grease Pencil.

Grease Pencil

An annotations system used to provide feedback on 3D scenes, Grease Pencil has been overhauled to allow for 2D drawing and animation. This tool gives users the advantage of using 2D drawing in a 3D environment, which opens the door for very interesting new workflows.

Many other tools are specific to Grease Pencil objects, such as modifiers and materials that are meant to work with 2D animations.

Cartoon Shaders with EEVEE

Thanks to some nodes that are specific to EEVEE, such as the Shader to RGB node, it's possible and easier than ever to create cartoon shaders to achieve a 2D animation look with 3D models and environments.

Freestyle

Freestyle is a set of options available in Blender that allows for nonphotorealistic rendering with cartoon- and anime-type results. This feature can draw the outlines of objects as though your 3D models were inked, for example. You can easily adjust those lines to achieve all kinds of interesting effects, such as a blueprint or sketch effect. Then you can mix those lines on top of a normal render and tweak the render to achieve flat colors if you so desire. I really recommended this feature if you're into motion graphics and 2D art.

VFX: Masking, Object Tracking, and Video Stabilization

In this book, you learned how to use the Movie Clip Editor to track the camera motion. Although that function is probably its best-known function, this editor can perform many more tasks. You can use the tracking data to stabilize your shaky footage and make the video look a lot smoother, for example.

You can also track the motion of objects in the video to apply that motion to 3D objects in your scene. You could shoot a video of an actor carrying paper markers on their hands, track those markers, and composite futuristic weapons and gadgets (for example) on top of the markers so they fit the movement and perspective.

The Movie Clip Editor also gives you the necessary tools to create masks over your footage that you can use later in the compositor. If you want something in the real video to stay in front of the 3D stuff, you can mask it and keep it in front by using the compositor.

Video Editing

Blender includes video editing tools which are more than enough for many use-cases and even allow for interesting workflows that integrate 3D scenes into the editing process. You can access these tools from the Video Sequence Editor and load images, videos, or 3D scenes as strips that you can move in the Timeline to use them as you would in video editing software.

Sculpting

Sculpting is one of the most creative ways of modeling. If you like organic modeling, sculpting is something that you really need to check out, especially for creating characters.

In the 3D Viewport, switch to Sculpt Mode and use it as you would with the texture brushes. This mode pushes and pulls geometry in different ways, however, depending on the brush you select. For sculpting, you usually want to use a Multi-Resolution modifier, which is similar to a Subdivision Surface modifier but also stores the details of each subdivision level, allowing you to detail geometry much more accurately. By combining a Multi-Resolution modifier with Sculpt Mode, you can create extremely detailed organic models in a very artistic way, almost as though you were sculpting with clay, which can be really entertaining.

Another tool to keep in mind when you want to sculpt is Dynamic Topology (or Dyntopo). It's not compatible with a Multi-Resolution modifier; instead, it dynamically divides the mesh under the brush to detail the model locally as needed. The resulting mesh needs retopology (see the next section) because it can get very dense and triangulated, but in exchange, it gives you a lot of freedom when sculpting, as you won't be limited by the original topology. Also, it divides the mesh locally, meaning that you don't have to subdivide the entire model to add a little localized detail.

Since Blender 2.81, Sculpting tools have been improved substantially, and there are also remeshers available natively in Blender: they allow you to combine different objects to build a single mesh based on those objects' volume, for example, or recreate the mesh's shape with better topology (especially useful after sculpting shapes with Dyntopo).

In this book, you learned how to create the character in a more traditional way, using polygonal modeling, but nowadays, many times, organic characters can be created through 3D sculpting.

Sculpting is a more artistic technique, so it will require lots of practice for you to get good results. If you have a traditional sculpting background, it will be much easier. That's why I stuck with a traditional workflow that can be explained with steps to follow, because sculpting relies heavily on your own artistic skills and experience.

Retopology

Retopology is usually the next step after sculpting. It's not a specific tool or set of tools—more like a technique of building new geometry with a good topology on top of other geometry that doesn't have a good topology. Retopology is all about re-creating a shape with a new topology. Some software is designed specifically for retopologizing, and sculpting software offers retopology tools.

You can easily use this technique in Blender, even though it's just normal modeling while snapping to other surfaces. A sculpted mesh usually is heavy, with lots of polygons, and you often start from a basic geometry that doesn't have an optimal topology when the details are in place. Retopology tools help you create the final mesh with a good topology, using the high-resolution mesh as a base for the shape.

Retopology is simple: Just enable the Snapping tool and set it to Snap to Faces. Thereafter, when you adjust geometry and create new vertices, for example, Blender snaps them to the surfaces of other objects, allowing you to re-create their shapes with the desired topology.

Maps Baking

Maps Baking is an interesting feature. If you have lights and shadows in your scene, you can bake that lighting into a texture for the selected object. When you load that texture into the object, it has the lights and shadows projected on it so you can see it in real time, avoiding heavy, long-rendering calculations.

Use this feature when you want to see an effect in real time while still getting the look of a final render (or just to render a lot faster). With this method, you can make a scene look like a final render by displaying a simple texture that already includes light and shadows.

You can bake details from other objects into the selected one to generate normal maps and displacement maps that you can use later to make the object look as though it has a lot more detail than it actually does.

You can find these tools on the Render tab of the Properties Editor (only available for Cycles at the moment of this writing). The setup for maps baking is different in Blender Render and in Cycles, but both engines can bake maps.

Add-Ons

Blender has a set of native tools, but it can be greatly expanded with new functionalities thanks to Add-Ons. Some are simple and perhaps add a tool or an automation for a process, but others are extremely complex and add a lot of value to what can be done in Blender.

Included Add-Ons

In User Preferences is a tab named Add-Ons, where you can define which add-ons should be enabled or disabled, and you can install external add-ons. Blender already includes some interesting, useful add-ons that you should check out. Some of these add-ons allow you to create special primitives (stars, pipes, and so on); others enable you to export and import files in different formats. Still other add-ons provide new features or modeling tools. Some of these add-ons may be interesting to you or useful for some specific task you need to do.

Blender includes dozens of add-ons. Check them out and enable the ones that you'd like to try.

More Add-Ons

If the add-ons included in Blender are not enough for you, or if you're looking for something more specific, you can perform an online search. Many Blender users have developed hundreds (if not thousands) of add-ons for their own needs and shared them with the public so that everyone can take advantage of them.

Since the first edition of this book, an online market has grown; now you can buy Blender resources (models, textures, and so on) online, as well as add-ons. This development opens the possibility for companies and individual developers to make a living by creating and maintaining great add-ons, thereby greatly increasing the quality of these tools.

Better selection tools and options, retopology tools, node improvements, material libraries, or even completely changing the interface with a set of tools for modeling using specific workflows—anything is possible with add-ons, and some add-ons will make your life a lot easier.

Python Scripting

If the tools that Blender provides are not enough for you, or if you need something specific, you can use Python scripts to develop your own tools. Blender lets you create and run scripts, change how the interface looks, and make your own add-ons to add new functionalities. Head to the Text Editor and start coding!

Python scripting makes Blender customizable and represents an attractive feature for developing tools that achieve specific results.

Summary

Blender is a lot more than meets the eye. This book describes some of the main features and tools available to you, but many more are available.

Blender is continually evolving software, and new features are added all the time. You can see this evolution as it happens because the development process is transparent; you don't need to wait until the next version is released to know what will be new.

You should be very proud that you've learned how to create characters in Blender, because that's no simple task. Character creation is challenging, but you're on your way to expressing your creativity by using the extensive features of this cutting-edge software!

I hope that you learned a lot from this book and that it helped you understand Blender's basic features so you can get started creating animations. Keep improving your skills and learning more about Blender's advanced features.

This page intentionally left blank

Index

Numbers

- 2D animation, 373
- 2D cursor, UV Editor, 186
- 2D image-editing software, 217–218
- 3D texturing software, 217, 221–222
- 3D camera, 330
- 3D cursor
 - 3D Viewport editor, 27
 - adding neck to jacket, 159
 - adjusting camera motion, 342
 - detailing backpack and jacket, 157
 - main functions of, 35–37
 - posing characters without skeletons, 157
 - rigging eyes, 288–289
- 3D models
 - character design using, 88
 - making character reference images for, 86–87
 - painting with 2D texture. *See* Unwrapping and UVs
 - vs. 2D references, 134
- 3D vector curves, character design, 88
- 3D Viewport
 - defined, 19
 - displaying UV text grid in your model, 195–196
 - elements, 26–27
 - Live Unwrap in, 198
 - manipulators, 43–44
 - Move, Rotate, Scale objects with Active Tools, 42–43
 - navigation gizmos, 33
 - Pivot Point and Transform Orientation, 45
 - regions, 27–29
 - Texture Paint interaction mode, 207–208
 - Texture Paint Workspace, 206–209
 - viewing animation smoothly, 321
 - Workspace options, 23
- 3D Viewport header
 - accessing modeling tools, 94
 - adding fingers to hand, 166–167

- animation controls in Timeline, 316
- creating objects, 41
- enabling manipulators, 44
- Interaction Mode menu, 52, 94
- navigating from View menu, 33
- numerically precise transforms, 46–47
- Pivot Point popover, 35
- proportional editing menu, 96–97
- Select menu, 98–100
- Shade Smooth and Shade Flat options, 53
- Snap icon, 45
- snapping tools, 124
- Texture Paint interaction mode, 207–208
- understanding, 29–30
- Viewport Shading modes, 59–60
- X-Ray options, 125

A

- Action Editor
 - creating walking animation, 321
 - NLA Editor loads actions saved in, 319
 - repeating walking animation with NLA Editor, 324–325
 - works with Dope Sheet editor, 317
- Active Keying Sets panel, keyframes, 314–315
- Active selection, objects, 33
- Active Tool and Workspace Settings tab, Properties Editor, 22–23
- Active Tools
 - enabling 3D cursor, 37
 - moving, rotating, and scaling objects, 42–43
 - performing selections with, 35
 - pros and cons of, 43
 - Texture Paint interaction mode using, 207–208
 - working in Weight Paint Mode, 284
- Adaptive Sampling, rendering in Cycles, 250
- Add Bone Constraint button, 263
- Add Driver, 299
- Add menu, Compositor, 353–354

- Add-Ons, 375–376
- Add-Ons tab, User Preferences, 38
- Add Shader, 229–230, 242
- Adjust Last Action menu, stripping numbers, 291
- Adjust Last Operation menu
 - creating new bones, 260
 - creating objects, 41–42
- Adjust Last Operation panel
 - blocking basic shape of face, 141
 - using Bevel tool, 101
 - using mesh modeling tools, 94
- Adjusting UVs, 198
- Adjustments, rigify rig, 276–277
- Advanced Options, Rigify Buttons panel, 276
- Agent 327: Operation Barbershop* (2017), 6
- Alignment tools, UVs, 198
- Alpha Over node, Cycles, 360–361
- Alpha (transparency or opacity) channel
 - masks and, 230–231
 - materials, 232
 - Viewport Display options for eye materials, 238
- Ambience, with volumetrics, 239
- Ambient Occlusion (AO), rendering in EEVEE, 248
- Anatomy
 - marker, 335–336
 - node, 351–352
- Animation
 - character production stage, 74
 - creating to move bones you are weighting, 286
 - of linked character, using proxies, 310
 - production stages of film, 72
- Animation, character
 - exercises, 326
 - overview of, 311
 - posing, 312–315
 - summary, 326
 - using character's rig, 311–312
 - walk cycle, 320–326
 - working with animation editors, 315–320
- Animation curves, in Graph Editor, 318
- Animation editors
 - common controls and tips for, 319–320
 - Dope Sheet editor, 317
 - Graph Editor, 317–319
- Non-Linear Animation Editor, 319
- Timeline, 316–317
- working with, 315
- Animation output, setting, 365–366
- Animation tab, User Preferences, 38
- Anisotropic BSDF shader, 240
- Annotations tool, 121
- AO (Ambient Occlusion), rendering in EEVEE, 248
- Appearance, character design, 77
- Appending
 - reusing character in different scenes, 308
 - working with collections, 308–309
- Areas, Blender UI
 - maximizing and making it full-screen, 18
 - resizing, 16
 - splitting and joining, 17
 - swapping and duplicating, 17–18
- Armature Deform group, 283
- Armature Layers panel, Properties Editor
 - armature layers, 274–275
 - enabling deformer bones only, 281
 - organizing bones, 273
- Armature menu, 3D Viewport header, 268
- Armature modifier, skinning, 283–284
- Armature Properties tab, Properties Editor, 274–275
- Armature tab, Properties Editor, 267, 289
- Armatures
 - adding constraints, 262–264
 - bone hierarchies, 261–262
 - character rigging workflow, 256
 - defined, 255
 - forward and inverse kinematics, 264
 - manipulating bones, 258–260
 - modifying when linked in scene with proxies, 310
 - moving bones to armature layers, 303
 - Object, Edit, and Pose Modes, 261
 - practicing with bones and IK constraints, 264–267
 - tips before you start rigging, 267–268
 - using in Object mode, 52
 - working with, 257
- Arms. *See also* Torso and arms, modeling
 - animating character's rig, 311–312
 - shading character, 243
- Auto Grab option, F2 add-on, 122

- Auto Keyframing option, 313
- Auto Merge, 3D Viewport header, 123, 166–167
- Auto Perspective option, 3D scene navigation, 32
- Auto-Save Preferences, 39
- Automatic effects, 3D texturing software, 222
forward and inverse kinematics, 264
retopology, 130
rig generation, 257
rigging process in Rigify, 269
- Automatic Weights
adding Armature modifier, 283–284
do not update after adding Armature modifier, 299
knowing which objects don't need weights, 281
testing skinning process, 293
- Automatically Pack into .blend, 215
- AutoName Left/Right, naming bones, 290
- Average Islands Scale tool, packing UVs, 203
- B**
- Background
applying current footage to Camera view as, 342
creating/testing scene lighting, 346–347
play animation to view markers in black, 338
rendering in Cycles, 358
shader, 247–248
- Background's Alpha, scene lighting, 346
- Backpack
defining arms and torso, 154–155
detailing, 156–158
- Badges
modeling, 177–178
packing UVs, 202–203
unwrapping, 201
- Base Color (Diffuse Color) property, materials, 232
- Base design, 79–81, 83–84
- Base skeleton, creating rig, 268
- Base texture, 214–215
- Basis, shape key, 295–296
- Beauty option, Fill tool, 108
- Belt
detailing, 157–158
finishing, 158
shading character, 243
- Bevel tool, 100–101, 109
- Bevel Vertices tool, eyes, 144
- Big Buck Bunny* (2008), 6
- Bisect tool, 102
- .blend file
every object belongs to collection in, 309
packing image into, 215
saving and loading, 64–65
saving Workspaces, 22
before you begin painting texture, 209
- Blender 2.83 LTS (long-term support), 7–8
- Blender Foundation, 6–10
- Blender Institute, 6
- Blender, introduction
Blender Foundation and development, 8–9
commercial software, 4–5
community, 10–11
development income channels, 10
exercises, 12
history of, 6–8
open-source software, 5
selling works created with, 5–6
summary, 11–12
understanding, 3–4
- Blender Store, 10
- Blocking
basic shapes for torso and arms, 152–154
face's basic shape, 139–140
- Bloom, rendering in EEVEE, 248
- Blur brush, 286, 293
- Blurry footage, when shooting video, 331
- Body language, designing great characters, 76
- Bone groups, disabling Deform option, 280
- Bone Groups panel, 273–274
- Bone hierarchies, 255, 260–262
- Bone Layers menu, 274
- Bone Properties tab, Properties Editor, 259
- Bone Settings menu, 280
- Bone tab, Properties Editor, 280
- Bones
adding custom shapes to, 305–306
adding to deformation, 292–293
character rigging workflow, 256

- configuring drivers, 301
- creating armature by manipulating, 258–260
- creating face controls, 298–299
- creating new, 260
- inside of rig is made of, 255–256
- mirroring, 291
- moving to armature layers, 303
- naming automatically, 290
- organizing, 273–277
- parenting objects to, 281–282
- resetting to original positions, 285–286
- tips before you rigging, 267–268
- working in Object, Edit, and Pose Modes, 261
- Boolean (Intersect) tool**, 102–103
- Boots**
 - adding details to, 83
 - modeling, 161–163
 - packing UVs, 202–203
 - unwrapping, 200–201
- Border selection**, 98
- Boundary option, Inset tool**, 109
- Box modeling**, 129
- Box selection**
 - connecting and manipulating nodes, 354
 - selecting objects, 33
 - using Active Tools, 35
- Branches**
 - Blender development versions, 9
 - of node tree, 350–351
- Bridge Edge Loops tool**, 104
- Bridge tool**, 121
- Brushes**
 - Texture Paint interaction mode, 207–209
 - Weight Paint Mode, 284
- Bump property, materials**, 232–233
- Buttons**, adding custom, 303–305

- C**
- Caching footage**, 334
- Camera**
 - in 3D Viewport editor, 27
 - moving in your scene, 62–63
 - render test scene setup with lights and, 246
 - taking reference images from background images on, 131
- Camera Properties tab, Properties Editor**, 346
- Camera Solver constraint**, 340–341
- Camera tracking**
 - adjusting camera motion, 341–342
 - anatomy of a marker, 335–336
 - applying tracked motion to camera, 340–341
 - character postproduction stage, 74
 - configuring camera settings, 339
 - exercises, 343
 - loading your footage, 333–334
 - overview, 329
 - shooting video for easy tracking, 330–331
 - solving camera motion, 339–340
 - summary, 343
 - testing, 343
 - tracking camera motion, 333
 - tracking features in footage, 336–338
 - understanding, 329–330
 - using Movie Clip Editor, 332–333
- Camera view, navigating 3D scene**, 32
- Cap**
 - adding details to, 82–83, 170–172
 - giving personality with, 81–82
 - modeling base of, 168–169
 - packing UVs, 202–203
 - shaping hair under, 172–175
 - unwrapping, 200–201
- Cartoon shaders with EEVEE, 2D animation**, 373
- Cascade Size, rendering in EEVEE**, 248
- Caustics, in Cycles**, 245
- Chain of bones**, 258, 260
- Chamfer tool**, 3ds Max, 101
- Channels**
 - masks and alpha, 230–231
 - of materials, 216, 231–233
 - other texture, 221
- Character**
 - creation plan, 73–74
 - shading. *See Shading your character*
 - texture painting. *See Texture painting*
 - unwrapping rest of, 200–201
- Character design**
 - adding color, 84–85
 - appearance, 77
 - base design, 79–81
 - context of story in, 76
 - description of character, 75–76

- details, 82–83
- exercises, 89
- finalizing, 85
- head, 81–82
- other design methods, 88–89
- overview of, 75
- personality, 76
- reference images, 86–88
- refined design, 83–84
- silhouettes, 78–79
- style, 77
- summary, 89
- Character modeling**
 - boots, 161–163
 - cap, 168–172
 - exercises, 180
 - eyes, 135–139
 - face. *See Face*
 - final details, 176–179
 - hair, 172–175
 - hands, 164–168
 - legs, 159–161
 - mesh topology and, 127–129
 - methods, 129–131
 - overview of, 127
 - setting up reference images, 131–135
 - summary, 180
 - torso and arms. *See Torso and arms, modeling*
- Character rigging**
 - adjusting Rigify rig, 276–277
 - animating using, 311–312
 - armature layers, 274–275
 - in character production stage, 73
 - creating custom shapes, 305–306
 - defining rigs, 255–256
 - defining workflow, 256–257
 - enabling Rigify add-on, 267
 - exercises, 310
 - final retouches, 306
 - organizing bones, 273
 - overview of, 255
 - reusing character in different scenes, 307–310
 - summary, 310
 - tips before you start, 267–268
 - understanding Rigify rig, 275–276
- using Rigify to generate Jim's rig, 268–273
- working with armatures. *See Armatures*
- working with skinning. *See Skinning*
- Checker Deselect tool**, selection, 100
- Cheeks**
 - adding blush to, 219
 - creating face controls, 298
 - creating facial expression with, 296
 - defining face's shape, 143
 - modeling basic face shape, 142
 - weight painting, 293
- Chest badge deformation**, 294–295
- Choose Only Selected Curves Keyframes, Graph Editor**, 320
- Circle selection**, 33, 35
- Circle tool, LoopTools**, 121, 166–167
- Classes, node**, 350–351
- Clear Parent**, bone hierarchies, 260
- Clear Seam tool, UV Mapping**, 188–189
- Clear Transform option, testing rig in Pose Mode**, 268
- Clichés**, designing character appearance, 77
- Clip Display, Movie Clip Editor**, 337
- Clipping option, Mirror modifier**, 141, 159
- Cloth simulation**, 372
- Clothing**, checking deformations, 287
- Cloud, Blender**, 10
- Collapse/expand nodes**, 354
- Collapse option, Merge tool**, 114
- Collections**, 134, 308–309
- Color**
 - adding base, 218–219
 - analyzing real footage before adding lights, 345–346
 - changing UV test grid to white for line details, 210
 - character design details, 82–83
 - configuring brush, 205
 - masks and image, 230
 - UV test grid in, 195–196
- Color property, lights**, 62
- Commercial software**, 4–5
- Communicator earpiece**
 - character design details, 82–83
 - modeling, 177
 - shading character, 243
- Community, Blender**, 10–11

- Compositing
 anatomy of nodes, 351–352
 in character postproduction stage, 74
 getting started with nodes via Node Editor, 352–355
 of nodes in Cycles, 359–361
 overview of, 349
 previewing result, 355–356
 summary, 366
 two main methods of, 349
 understanding nodes, 349–351
 your scene in Cycles, 356–357
 your scene in EEVEE, 364–365
- Compositing Workspace, 359–361
- Compositor Backdrop, 355
- Compositor Editor
 defined, 19
 getting started with nodes, 352–354
 previewing result, 355–356
- Conditions, for painting textures, 210–211
- Configuration, driver, 300–303
- Connect tool, 104–105
- Connected Only option, proportional editing, 97
- Connected option, defining bone hierarchies, 260
- Connecting UVs, 199
- Connections, node, 352, 354–355
- Constraints
 adding bone, 262–264
 Camera Solver, 340–341
 character rigging workflow, 256
 forward and inverse kinematic, 264
 inside rig, 256
 moving object along a curve, 325–326
 object vs. bone, 263
 practice with bones and IK, 264–267
 using eyedropper for bone, 264
- Contact Shadows, rendering in EEVEE, 248
- Context of story, in character design, 76
- Contextual menu
 accessing modeling tools, 95
 connecting and manipulating nodes, 354
 UV Editor, 187
- Control bones, 256
- Controls, creating face, 298–299
- Converter nodes, 352
- Copy Attributes add-on, User Preferences, 55
- Copy Pose animation, walking, 323
- Copying, modifiers to other objects, 55
- Correlation, marker settings, 336
- Cosmos Laundromat* (2015), 6–7
- Create New Collection, 309
- Crowdfunding, Blender development, 10
- Cube
 blocking basic shape of face, 141–142
 modeling basic hand shape, 164–166
- Cube Projection tool, UV Mapping, 190
- Current material, Material Properties tab, 238
- Current selection, Material Properties tab, 238
- Current UV layer, UV Editor, 187
- Cursor, 2D, 186
- Curve tool, LoopTools, 121
- Custom buttons, Rigify script adding, 303–305
- Custom shapes
 assigning to bones, 256
 character rigging workflow, 257
 creating, 305–306
- Cycle, walking animation is a, 321
- Cycles
 Color and Power light options, 62
 creating shadow catcher, 357
 Displacement, 239–240
 EEVEE versus, 233
 Emission shader, 240
 final render, 361
 GPU rendering, 64
 history of Blender, 6–7
 Maps Baking tool, 375
 rendering/compositing scene, 356–367
 rendering images, 57–58, 63–65, 250–252
 rendering in EEVEE versus, 66
 shading eyeballs, 245
 shadows more accurate in, 251
 switching between EEVEE and, 233–234
 testing scene lighting, 348–349
 viewport shading, 59
- Cylinder Projection tool, UV Mapping, 190

D

- Datablocks, 49–51, 321
- Default cube, 3D Viewport editor, 27

- Deform bones
 - character rigging with, 256–257
 - enabling only, 281
 - parenting objects not needing weights to, 282
 - protecting with layers, 310
 - selecting while you paint weights, 285
- Deform modifier, 54
- Deform panel, Bone tab of Properties Editor, 280
- Deformations
 - checking after weighting character, 287
 - checking for unwanted, 293
 - on chest badge, 294–295
 - disabling as option for bones, 280
 - lattices for eyeball, 137–138
 - mirroring and adjusting eyes, 138–139
 - requiring good mesh topology, 128
 - rigging eyes, 288–289
 - rigging jaw, 291–292
 - skinning and, 278
 - skinning eyelids and jaw, 292–293
 - vertex groups and, 278–279
- Delete and Dissolve tools, modeling, 105–106
- Delete tool, 260
- Denoising
 - enabling in Cycles, 250
 - rendering in Cycles, 358
- Density, good mesh topology, 128
- Depth parameter, reference image setup, 132
- Description, designing character, 75–76
- Deselect all, objects in scene, 35
- Deselect, material slots, 237
- Design, character preproduction stage, 73
- Details
 - adding character design, 82–83
 - painting texture, 2-D image-editing software, 219
- Development, Blender, 4, 10
- Development, commercial vs. open-source software, 4–5
- Diffuse BSDF shader, 240, 241–242
- Diffuse Color property, materials, 232
- DirectX, calculating normal maps, 221
- Disconnect, bone hierarchies, 260
- Displacement option (Cycles only), 239–240
- Display Only Axis Aligned, reference images, 133
- Display panel, 186, 267
- Display Perspective, reference images, 133
- Dissolve Faces tool, modeling base of cap, 168–169
- Donations, Blender development via one-time, 10
- Dope Sheet editor
 - animation with, 317
 - defined, 19
 - repeating walking animation, 324
 - retiming keyframes, 323
- Downloading Blender, 13
- Drivers
 - configuring, 300–303
 - controlling face shapes, 299
 - setup menu, 299–300
- Drivers editor, 19, 301–302
- Duplicate nodes, 354
- Duplicate tool
 - bones, 259
 - modeling with, 106
- Duplicating areas, 17–18
- Dynamic Topology (Dyntopo) sculpting, 374

E

- Earpiece. *See* Communicator earpiece
- Ears
 - adding to face, 147–149
 - modeling communicator earpiece. *See* Communicator earpiece
 - packing UVs, 202–203
- Eclipse Public License (EPL), open-source software, 5
- Edge flow, selecting loops and rings, 97–98
- Edge loops
 - blocking basic shape of face, 141–142
 - creating eyeball, 135–136
 - in good mesh topology, 127
 - joining using Bridge Edge Loops tool, 104
 - selecting, 98
- Edge Offset tool, 114–115
- Edge rings, 98
- Edges
 - defined, 93
 - marking as seams for UV unfolding, 193–194
 - selecting, 94
 - working with, 93–95

- Edit Mode
 - accessing unwrapping menus, 188
 - adding details to cap, 171
 - adjusting Rigify rig, 277
 - armature layers, 274
 - assigning vertices to groups only in, 279
 - changing bone hierarchies from, 261
 - character rigging workflow, 256
 - creating skeleton/modifying bone structure, 258
 - defining hierarchy of bones, 260
 - deforming eyeball using lattices, 137–138
 - detailing backpack and jacket, 158
 - disabling Deform option for bones, 280
 - interaction modes in, 51–52
 - marking seams for UV unfolding, 193–194
 - packing UVs, 203
 - practicing bones and IK constraints, 264–267
 - rigging eyes, 289
 - selecting vertices, edges, and faces, 94
 - selections in. *See* Selections, making
 - walking along path, 325–326
 - weight values, 287
 - working with armatures, 261
 - Editing tab, User Preferences, 38, 318
 - Editor Menu, 3D Viewport header, 29
 - Editor Type Selector, 3D Viewport header, 29
 - Editors
 - default, 16
 - that use nodes, 352
 - types of, 18–21
 - Edits, defined, 30
 - EEVEE (real-time render engine)
 - in Blender version 2.80, 3
 - cartoon shaders, 373
 - Color and Power light options, 62
 - differences/compatibility between Cycles and, 233
 - Emission shader in, 240
 - final render in, 365
 - history of, 7
 - Material Properties tab settings, 240
 - rendering in, 63–65, 248–249
 - rendering in Cycles versus, 57–58, 66
 - setting up scene lighting, 348–349
 - shading eyeballs, 244
 - switching between Cycles and, 233–234
 - viewport shading, 59
- Elbows
 - adding details to suit, 82
 - adjusting skeleton to 3D model, 272
 - defining arms and torso, 154–155
 - modeling basic shape for arms, 152–154
 - painting with blur brush, 286
 - requiring complex deformations, 128
- Elements, texture, 215–216, 218
- Elephants Dream* movie project, 6
- Emission property, materials, 232
- Emission shader, 240
- Emitter geometry, particles, 371
- Empties, defined, 131
- Empty Groups, adding Armature modifier to, 283
- Enable Display Orthographic, reference images, 133
- Envelope Weights, adding Armature modifier to, 283
- Environment, render test scene setup, 246–248
- EPL (Eclipse Public License), open-source software, 5
- Exercises
 - Blender UI, 40
 - camera tracking, 343
 - character animation, 326
 - character design, 89
 - character modeling, 180
 - character rigging, 310
 - introduction to Blender, 12
 - materials and shaders, 252
 - modeling tools, 125
 - painting textures, 224
 - project overview, 74
 - rendering, 366
 - texture painting, 224
 - user interface (UI), 40
 - your first scene in Blender, 66
- Export UV Layout, 2-D image-editing software, 218
- Exporting
 - 3D texturing software, 222
 - 3D model, 221
 - final render, 365–366
- Extrude Faces Along Normals, 158, 163

- Extrude tool, 106–109
- Eye icon, hiding/showing reference images, 134
- Eyeballs
 - assigning material slots to, 237
 - creating, 135–136
 - deforming using lattices, 137–138
 - shading character, 243–245
- Eyebrows, modeling, 176–177
- Eyedropper, select objects from different menus with, 264
- Eyes
 - blocking basic shape of face, 142
 - creating face controls using bones, 298
 - defining on face, 145–146
 - defining shape of, 144
 - deforming eyelids, 292–293
 - edge flow around, 140
 - mirroring and adjusting, 138–139
 - mirroring eye rig, 290
 - overview of, 135
 - rigging, 288–289
 - skinning jaw and, 292–293
- F**
 - F2 add-ons, modeling with, 122
 - .fbx format, exporting 3D model, 221
 - Face
 - adding ears, 147–149
 - blocking basic shape, 140–142
 - building inside of mouth, 149–150
 - controlling shapes using drivers, 299–303
 - creating controls for, 298–299
 - defining eyes, mouth, and nose, 145–146
 - defining shape of, 142–144
 - rigging jaw, 291–292
 - studying topology of, 139–140
 - Faces (polygons)
 - defined, 93
 - Knife tool cuts, 110
 - mesh topology uses quads for, 128
 - Poke tool for, 115
 - selecting, 94
 - separating UVs, 199
 - Solidify tool for, 118
 - Split tool for, 119
 - working with, 93–95
 - Facial rig
 - adjusting Rigify script to add custom buttons, 303–305
 - adjusting skeleton to 3D model, 271
 - creating, 288–289
 - creating face controls, 298–299
 - mirroring bones, 290
 - mirroring eye rig, 290
 - mirroring shapes, 298
 - modeling shape keys to build, 295–297
 - naming bones automatically, 290
 - organizing, 303–305
 - possible side effects of mirroring bones, 290
 - rigging jaw, 290–291
 - sending bones to proper layers, 303
 - skinning eyes and jaw, 291–292
 - using drivers to control face shapes, 299–303
 - working on full character pose first, 275
 - Falloff selection, proportional editing, 97
 - Features
 - developing in Blender, 9
 - shooting video for tracking using recognizable, 331
 - shooting video for tracking using static, 331–332
 - tracking in footage, 336–338
 - tracking when obscured by foreground frames, 337
 - Features, other Blender
 - 2D animation, 373
 - add-ons, 375–376
 - Maps Baking, 375
 - Python scripts, 376
 - retopology, 375
 - sculpting, 374
 - simulations, 371–372
 - summary, 376–377
 - VFX: masking, object tracking, and video stabilization, 373
 - video editing, 374
 - Feet
 - character design with silhouettes, 78
 - creating, 265
 - creating poses for walk cycle, 322–323
 - on floor, adjusting sizes of 3D models, 270
 - on floor, modeling torso/arms, 150–151

- modeling boots for. *See Boots*
- using IK/FK to control position on ground, 264, 311
- walking along a path, 325–326
- File Browser**, 20
- File Paths** tab, User Preferences, 38
- Fill and Grid** Fill tool, modeling with, 108
- Fill layer**, 3D texturing software, 222
- Film panel**
 - rendering in Cycles, 250
 - rendering in EEVEE, 248
- Film** without visual effects, production stages, 70–71
- Filter Add-Ons** option, Workspace, 22
- Filters** menu, Outliner, 134, 348
- Final design**, of character, 85
- Final render**
 - compositing before, 19, 349
 - in Cycles, 361
 - in EEVEE, 365
 - exporting, 365–366
 - launching, 65–66, 366
 - Maps Baking achieves look of, 375
 - Rendered viewport shading mode similar to, 58
 - showing/hiding objects in, 347–348
- Fingers**
 - adding to hand, 166–168
 - adjusting Rigify rig, 276, 277
 - modeling basic shape, 165–166
- First scene**
 - arranging objects, 48–49
 - creating objects, 41–42
 - exercises, 66
 - flat or smooth surfaces, 53
 - interaction modes, 51–52
 - lighting, 62
 - managing materials, 60–61
 - modifiers, 54–57
 - moving camera in, 62–63
 - naming objects/using datablocks, 49–51
 - overview of, 41
 - rendering, 63–66
 - transform objects with Active Tools, 42–43
 - transform objects with keyboard shortcuts, 46–47
 - transform objects with manipulators, 44–46
- transform objects with menus, 47–48
- Viewport shading**, 58–60
- Workbench**, EEVEE, and Cycles, 57–58
- FK** (forward kinematics)
 - animating your character, 311–312
 - character rigging, 264
- Flat surfaces**, applying, 53
- Flatten** tool, LoopTools add-on, 121
- Flip Names**, mirrored bones, 291
- Floor** button, adjusting camera motion, 341–342
- Floor**, creating shadow catcher in Cycles, 357
- Floor grid**, 27, 347
- Fluid simulation**, 372
- Fly Mode**
 - moving camera in your scene, 63
 - navigating 3D scene, 33
- Focal length**, shooting video for easy tracking, 332
- Follow Path** constraint, moving object along a curve, 325
- Foot**. *See Feet*
- Footage**
 - analyzing before adding lights, 345–346
 - tracking camera motion, 333–334
- Forks**, bone hierarchy and, 262
- Forward kinematics (FK)**
 - animating your character, 311–312
 - character rigging, 264
- Frame limit**, monitoring tracking, 338
- Frame range**, animation editor controls, 316, 319–320
- Frames per second (fps)**, 320, 334
- Frames**, tracking, 337
- Freestyle**, 2D animation, 373
- Fresnel effect**, materials, 226

G

- General Public License (GPL)**, GNU, 5, 8
- General Public License (GPL)**, open-source software, 5
- Generate modifier**, 54
- Generate Rig** button, 276–277
- Geometry**
 - adding ears, 147–149
 - blocking basic shape of face, 140–142
 - creating eyeball, 135–136

- defining eyes, mouth, and nose, 145–146
 defining face's shapes, 143
 deforming eyeball using lattices, 137–138
 modeling basic shapes for torso and arms, 152–154
 modeling legs, 159–161
Gizmos, 3D Viewport navigation, 33
Glass shader, 240
Global View
 modeling with, 124
 navigating 3D scene, 32
Glossiness (roughness) property, 232, 243
Glossy BSDF shader, 240, 241–242
Gloves
 character design details, 82–83
 shading character, 243
 unwrapping, 200–201
GPL (General Public License), GNU, 5, 8
GPL (General Public License), open-source software, 5
GPUs, enabling rendering with Cycles, 64
Graph Editor
 animation with, 317–319
 Choose Only Selected Curves Keyframes, 320
 defined, 19
 Normalize option, 320
 tweaking animation curves, 324
Grayscale texture, Displacement (in Cycles), 239
Grease Pencil, 317, 373
Groups, bone, 273–274, 280
Groups, duplicating connected node, 354
Groups, vertex, 278–279
Grow and shrink selection, 98
Grow selection, 98
GStretch tool, LoopTools, 121
- H**
- Hair**
 adding natural details to, 174–175
 base design for character, 80–81
 options for creating, 172
 shading character, 243
 shaping locks of, 172–174
 simulation, 371–372
Hair BSDF shader, 240
- Hands**
 adding fingers and wrist, 166–168
 common errors when modeling, 164
 modeling basic shape, 164–166
 packing UVs, 202–203
 unwrapping, 201
- Hardware**, recommended, 13–14
- Hat**. *See Cap*
- Head**. *See also Face; Facial rig*
 character design for, 81–82
 character reference images for 3D model, 86–87
 checking deformations, 287
 packing UVs, 202–203
- Header**
 3D Viewport editor, 26, 29–30
 as editor region, 28–29
 most editors have, 18
 UV Editor, 186–187
- Helper bones**, 256
- Hide and Unhide feature**
 bones, 260
 modeling with, 124
 navigating UV Editor, 188
 separating UVs, 199
- Hierarchies, bone**, 255, 260–262
- High-resolution simulations, hardware for**, 372
- High-speed tracking features**, 338
- History of Blender**, 6–8
- Horizontal loop cuts, torso/arms**, 152
- I**
- Idea, character preproduction stage**, 73
- IK (Inverse kinematic) constraint**
 animating your character, 311–312
 character rigging, 264–266
- Image Editor**
 defined, 19
 loading reference images side-by-side, 131
 placing texture elements using reference images, 214
 previewing work in Compositor, 355–356
 saving modified textures in, 212
 saving your image, 215
Texture Paint Workspace, 206, 209
 before you begin texture painting, 210

- Image menu, saving your image, 215
 Image output, compositing nodes in Cycles, 360
 Image Texture, 242, 243
 Images
 camera tracking and, 329–330
 compositing, in character design, 89
 creating new UV test grid, 195
 exporting animation as sequence, 366
 exporting final render, 365
 loading reference, 131
 packing your, 215
 saving, 215
 shooting video for easy tracking, 330–331
 texture painting and, 211
 texture resolutions, 210
 texturing with 2D image-editing software, 218
 UV Editor options, 187
 Import, 3D model, 221
 In Front option, Display Panel, 267
 Info editor, 19
 Input nodes, 350–352
 Input sockets, nodes, 351–352
 Input tab, User Preferences, 38
 Inset tool, modeling with, 108–109
 Installation, Blender, 13
 Instances, 49, 50
 Interaction Mode, 3D Viewport header, 29, 52
 Interaction modes
 modifying objects, 51–52
 Texture Paint, 207–209
 Interface tab, User Preferences, 38
 Interface, UV Editor, 186
 Interpolations, keyframe, 318–319
 Intersect (Boolean) tool, 102–103
 Intersect (Knife) tool, 102–103
 Inverse kinematic (IK) constraint
 animating your character, 311–312
 character rigging, 264–266
 IOR (index of refraction) property, materials, 232
 Islands, linked selection for, 97
- J**
- Jacket
 chest badge deformation for, 294–295
 creating base texture, 214–215
- creating zipper, 154
 detailing, 156–158
 modeling arms for details, 151
 packing UVs, 202–203
 unwrapping, 200–201
- Jaw
 rigging, 291–292
 skinning eyes and, 292–293
- Join Shapes menu, creating shape keys, 296
 Join tool, 109–110
 Joining areas, 17
 Joints, bone, 258
- K**
- Keep Offset, bone hierarchies, 260
 Keyboard shortcuts
 3D scene navigation, 31–33
 Active Tools, 43
 Active Tools versus, 35
 common animation editor controls, 319–320
 creating objects, 41
 Dope Sheet editor, 317
 manipulating bones, 260
 markers, 335
 performing selections, 33–35
 pie menus, 24–26
 saving and loading .blend file, 64–65
 selecting vertices, edges, and faces, 95
 Subdivision Surface modifier, 57
 Timeline editor, 316–317
 tracking features in footage, 336–338
 transforms using, 46–47
 using manipulators for transforms, 45–46
- Keyframes
 animating character via, 312–315
 choosing only selected curves, 320
 marker settings, 336
 solving camera motion, 339–340
- Keying sets, 314–315
 Keymap tab, User Preferences, 38
 Knees
 adding details to suit, 82
 adjusting skeleton to 3D model, 272
 creating, 265
 modeling boots, 163
 modeling legs, 160
 painting with blur brush, 286

- practice with bones and IK constraints, 265–267
- requiring complex deformations, 128
- Knife (Intersect) tool**, 102–103
- Knife Project tool**, 111
- Knife tool (K)**
 - adding details to cap, 170–171
 - adding wrist to hand, 166–167
 - blocking basic shape of face, 141–142
 - defining arms and torso, 154
 - defining face's shapes, 143–144
 - modeling base of cap, 169
 - modeling legs, 160
 - modeling with, 110
- Krita, creating silhouettes, 79

- L**
- Lamp, 3D Viewport editor, 27
- Lasso selection, 33, 35
- Lattice, 137–139
- Launching, final render, 366
- Layers
 - 3D texturing vs. 2D image-editing software, 222
 - armature, 274–275
 - limitations of Texture Paint Mode, 213
 - materials can be made of several, 230–231
 - moving bones to armature, 303
 - protected, 309–310
- Legs
 - animating character's rig, 311–312
 - animating walk cycle, 321–326
 - bone hierarchies and, 262
 - controlling using IK/FK, 264, 269, 311–312
 - creating, 265
 - enabling deforming bones only, 281
 - modeling, 153, 159–161
 - modeling boots, 161–163
 - moving with single control bone, 256
 - practicing with bones and IK constraints, 265–267
 - unwrapping pants, 200
 - using NLA Editor for animating, 319
- Library Linking, reusing character, 307
- Licenses
 - commercial software, 4–5
 - open-source software, 5

- Life of Pi*, 3
- Light**
 - adding to scene, 62
 - character postproduction stage, 74
 - Emission shader converts mesh to, 240
 - how materials work, 226–227
 - options, 62
 - render test scene setup with, 246–248
- Lighting your scene**
 - analyzing real footage, 345–346
 - creating and testing lights, 346–347
 - overview of, 345
 - showing/hiding objects in render, 347–348
 - testing EEVEE and Cycles, 348–349
 - using Node Editor, 348–349
- Lights tab, User Preferences**, 38
- Limited Dissolve option, Delete tool**, 106
- Linear interpolation, Graph Editor**, 318
- Lines, smooth versus steady**, 214
- Linked duplicates**, 49, 50
- Linked Flat Faces option, selection**, 99
- Linked selection, Edit Mode**, 97
- Linking**
 - animating linked character with proxies, 310
 - reusing character in different scenes, 308
 - working with collections, 308–309
- Live Unwrap tool**
 - adjusting UVs, 197
 - options for, 198
 - unwrapping hands, 201
 - in UV Editor, 189
 - in UV Mapping menu, 188–189
- Load Factory Preferences**, 39
- Load Factory Settings**, 39
- Load UI option, disabling/enabling**, 22
- Loading**
 - reference images in 2D image-editing software, 218
 - textures, 223, 242–243
- Local Space, configuring drivers**, 301
- Local View**
 - 3D scene navigation, 32
 - modeling with, 124
- Lock Object Modes**, 52
- Locked to Selection, Movie Clip Editor**, 338
- Locking, markers**, 338
- LocRotScale, adding keyframes**, 312, 314

- Loft tool, LoopTools, 121
- Loop animation, walk cycle, 322
- Loop Cut and Slide tool
 - blocking basic shape of face, 141
 - blocking basic shapes for torso and arms, 152
 - defining shape of mouth, 144
 - modeling boots, 163
 - modeling legs, 159, 161
 - modeling with, 112
- Loops, 97–98
- LoopTools add-ons, 120–121

- M**
- Main menu, UV Editor, 187
- Make Edge/Face tool, 113
- Manipulators, for transforms, 44–46
- MantaFlow, fluid simulation, 372
- Maps Baking, 375
- Mark Seam tool, UV Mapping, 188–189
- Markers
 - adjusting camera motion, 341–342
 - anatomy of, 335–336
 - creating, 336
 - locking, 338
 - settings for, 336
 - solving camera motion, 339–340
 - testing camera tracking, 343
 - tracking features in footage, 334, 336–338
- Masking, in Movie Clip Editor, 373
- Masks, and layers, 230–231
- Massachusetts Institute of Technology (MIT) license, 5
- Master versions, Blender, 9
- Match Type, Movie Clip Editor, 336
- Material
 - 3D texturing software, 221–222
 - adding and adjusting, 60–61
 - channels, 216
 - compatibility between EEVEE and Cycles, 58
 - conditions for texture painting, 211
 - definition of, 225
 - introduction to PBR, 215–216
 - managing, 60
- Material option, Texture Slots panel, 212–213
- Material Preview viewport shading mode
 - EEVEE used with, 57
 - how materials behave, 225
- shading character, 243–244
- understanding, 58–59
- viewing results of, 246
- Material Properties tab, Properties Editor, 60–61, 238–240
- Material slots, 236, 238
- Material tab, Properties Editor, 225–226, 236
- Materials and shaders
 - accessing nodes for material, 235
 - applying materials, 225–226
 - channels, 231–233
 - differences/compatibility of EEVEE and Cycles, 234–235
 - exercises, 252
 - how materials work, 226
 - masks and layers, 230–231
 - overview of, 225
 - PBR materials, 226–228
 - procedural textures, 233–234
 - running render tests, 246–252
 - shaders and mix shaders, 229–230
 - shading your character, 236–246
 - summary, 252
- Materials Properties tab, Properties Editor, 60–62
- Mathematical expression transforms, 47
- Menus
 - animating properties within, 315
 - Blender UI, 23
 - transforms using, 47–48
 - unwrapping, 188
- Merge by Distance option, 114
- Merge tool, modeling with, 113–114
- Mesh
 - adding Armature modifier to, 283
 - adding different materials to single, 238
 - all bones create deform, 280
 - converting to light with Emission shader, 240
 - deforming with deformer bones, 280
 - flow, 97–98
 - making skinning process easier, 282–283
 - modeling tools. *See* Modeling tools
 - remesher tools, 374
 - topology, 127–129
 - unwrapping, 191–192
- Mesh Deform modifier, chest badge, 294–295

- Metallic material setting, PBR materials, 227–228
- Metallic (Metalness) property, 232, 243
- Metallic-Roughness workflow, PBR materials, 227–228
- Metarig (Rigify skeleton), 257, 276–277
- Mirror Mode, eyes, 139
- Mirror modifier
 - adding details to cap, 170–172
 - adding fingers and wrist, 167–168
 - assigning, 55
 - blocking basic shape of face, 141
 - blocking basic shapes for torso and arms, 152
 - detailing backpack and jacket, 158
 - making skinning process easier, 282
 - modeling boots, 161–163
 - modeling legs, 159–161
 - shaping locks of hair, 172–173
 - unwrapping mirrored UVs, 192
- Mirror painting, creating silhouettes, 79
- Mirrored mesh, 109, 192
- Mirrored poses, walking animation, 322–323
- Mirrored UVs, unwrapping, 192
- Mirroring
 - bones, 291
 - eye rig, 290
 - shapes, 298
- Mix node, compositing nodes in Cycles, 360
- Mix Shader
 - creating layers, 230–231
 - defined, 229–230
 - mixing shaders, 241–242
- Mode option, Workspace, 22
- Modeling
 - in character production stage, 73
 - characters. *See Character modeling*
 - shape keys, 295–297
- Modeling tools
 - accessing, 94–95
 - Auto Merge, 123
 - Bevel tool, 100–101
 - Bisect tool, 102
 - Bridge Edge Loops tool, 104
 - Connect tool, 104–105
 - Delete and Dissolve tools, 105–106
 - Duplicate tool, 106
 - exercises, 125
- Extrude tool, 106–107
- F2 add-on, 122
- Fill and Grid Fill tool, 108
- Global and Local View, 124
- Hide and Reveal, 124
- Inset tool, 108–109
- Intersect (Boolean) tool, 102–103
- Intersect (Knife) tool, 102–103
- Join tool, 109–110
- Knife Project tool, 111
- Knife tool, 110
- Loop Cut and Slide tool, 112
- LoopTools add-ons, 120–121
- Make Edge/Face tool, 113
- Merge tool, 113–114
- Offset Edge Loop tool, 114–115
- overview of, 93
- Poke tool, 115
- Rip and Rip Fill tools, 115–116
- sculpting tools, 374
- selections. *See Selections, making*
- Separate tool, 116
- Shrink/Fatten tool, 116–117
- Slide tool, 117
- Smooth Vertex tool, 118
- Snapping, 124
- Solidify tool, 118
- Spin tool, 118–119
- Split tool, 119
- Subdivide tool, 119–120
- summary, 125
- working with vertices, edges, and faces, 93–94
- X-Ray, 125
- Models
 - making skinning process easier, 282–283
 - setting up for skinning, 280–281
- Modes, Dope Sheet, 317
- Modifier tab, Properties Editor, 54
- Modifiers
 - adding, 54–55
 - adding Armature, 283
 - adding subdivision surface, 55–57
 - deforming eyeball using lattice, 137–138
 - making skinning process easier, 282–283
 - modeling with, 130
 - overview of, 54
 - unwrapping mesh, 192

- Modify modifier, 54
- Monkey head (Suzanne), test object in Blender, 42
- Motion Blur, rendering in Cycles, 358
- Mouse, navigating 3D scene, 31–33
- Mouth
- blocking basic shape of face, 142
 - creating face controls using bones, 298
 - creating smooth deformation around, 128, 129
 - defining, 145–146
 - defining on face, 145–146
 - defining shape of, 143–144
 - edge flow around, 140
 - mesh topology for, 127
 - modeling inside of, 149–150
 - modeling teeth and tongue, 178–179
 - packing UVs, 202–203
 - rigging jaw, 291–292
 - skinning jaw, 293
- Move tool, 43, 44–46
- Movie Clip Editor
- adjusting camera motion, 341–342
 - applying tracked motion to camera, 340–341
 - camera tracking, 332–333
 - defined, 19
 - loading footage, 333–334
 - masking, object tracking, and video stabilization, 373
 - solving camera motion, 339–340
 - testing camera tracking, 343
 - tracking features in footage, 336–338
 - video tracking features i, 329
- Movie Clip node, compositing nodes in Cycles, 360–361
- Movie productions, Blender used in, 3
- Multi-Object editing, packing UVs, 203
- Multi-Resolution modifier, Sculpt Mode, 374
- N**
- N-gon
- defining eyes, 146
 - defining face's shapes, 144
 - Dissolve tool replaces elements with, 105
 - faces as, 93–94
 - modeling basic hand shape, 166
- Naming
- bones and IK constraints, 266
 - bones automatically, 290
 - bones before you start rigging, 267
 - bones for constraint, 263
 - bones using prefixes, 267–268
 - character parts when modeling, 139
 - how vertex groups and bones work, 292
 - makes skinning process easier, 283
 - mirrored bones, 291
 - newly created shape key, 298
 - objects intuitively, 141
 - objects using datablocks, 49–51
 - renaming bones, 259
 - renaming objects, 49
 - renaming objects in bulk, 290
 - renaming reference images in collection, 134
 - scene objects, 51
- NaN (Not a Number), history of Blender, 6
- Navigation
- 3D scenes in Blender UI, 31–33
 - common animation editor controls, 320
 - getting started with nodes, 353
 - gizmos in 3D Viewport editor, 27
 - UV Editor, 187–188
- Navigation tab, User Preferences, 38
- Neck
- adding ears, 148–149
 - adding to jacket, 158–159
 - modeling shape of face, 141, 143–144
 - modeling torso and arms, 152, 154
 - packing UVs, 202–203
 - unwrapping detail on, 200–201
- NeoGeo animation studio, 6
- New Image menu, UV test grid, 195
- Next Gen, 3
- No Textures message, texture painting, 211
- Node Editor
- basic controls, 352–356
 - compositing, 349
 - connecting and manipulating nodes, 354–355
 - creating nodes, 353–354
 - node anatomy, 351–352
 - overview of, 349
 - understanding nodes, 349–350

- Node tree, 350–351
- Nodes
 - accessing for material, 235
 - anatomy of, 351–352
 - compositing in Cycles, 359–361
 - connecting and manipulating, 354–355
 - creating, 353–354
 - getting started with, 352–353
 - understanding, 349–350
- Noise
 - procedural texture, 233
 - removing in renders, 358
 - rendering to reduce, 250
- Non-Linear Animation (NLA) Editor
 - creating walking animation, 321
 - defined, 19
 - overview of, 319
 - repeating walking animation, 324–325
- Nonmetallic material setting, PBR materials, 227–228
- Normal (Bump, Normal Map, Normal Bump) property, materials, 232–233
- Normal Map property
 - materials, 232–233
 - options for calculating, 221
 - shading character, 243
- Normalize option, Graph Editor, 320
- Nose
 - defining, 145–146
 - defining face's shape, 144
 - edge flow around, 140
 - first steps in modeling face, 142
 - playing with geometry of, 147
- Not a Number (NaN), history of Blender, 6
- Nth selection, Shortest Path Selection, 96
- Numerically precise transforms, 46–47
- NumPad
 - navigating 3D scene, 31–33
 - setting up reference images, 131
 - switching between Global/Local View, 124
 - working with pie menus, 25
- O**
- Object constraints, 263
- Object Data Properties tab, Properties Editor, 50
- Object Data tab, Properties Editor, 50, 132–133
- Object Mode
 - adding details to cap, 171
 - creating armature in, 258
 - creating eyeball, 135–136
 - deforming eyeball using lattices, 137
 - Join tool used in, 109–110
 - Lock Object Modes, 52
 - using interaction modes, 51–52
 - walking along a path, 325
 - working with armatures in, 261
- Object Properties tab, Properties Editor
 - adding or subtracting objects from collection, 309
 - creating shadow catcher in Cycles, 357
 - generating rig, 272
 - managing datablocks, 50
 - shading eyeballs in Cycles, 245
- Object tab, Properties Editor, 49
- Object Types Visibility, 3D Viewport header, 30
- Objects
 - adding keyframes to, 312–315
 - adding several materials to single, 236–237
 - applying material to multiple, 225
 - arranging in first scene, 48–49
 - creating, 41–42
 - knowing which ones don't need weights, 281–283
 - managing datablocks, 49–51
 - modifying with interaction modes, 51–52
 - naming scene, 51
 - renaming, 49
 - saving to prevent deletion by Blender, 321
 - selecting, 33–35, 264
 - showing/hiding in final render, 347–348
 - tracking in Movie Clip Editor, 373
 - transforming. *See* Transforms
 - viewing, 263
- Offset Edge Loop tool, 114–115
- Offset option, Shortest Path Selection, 96
- Opacity
 - scene lighting, 346–347
 - setting up reference images, 132
- Open Movies, 6–7, 8
- Open-source software (OSS), 4–6, 9
- OpenGL, calculating normal maps, 221
- Orbit, navigating 3D scene, 31

- Orientation
 - manipulating bones, 260
 - rigging eyes, 289
- Orientation panel, adjusting camera motion, 341–342
- OSS (Open-source software), 4–6, 9
- Outliner
 - creating collections from, 309
 - editor, 20
 - organizing reference images into collection, 134
 - renaming objects, 49
 - showing/hiding objects in render, 348
 - Texture Paint Workspace, 206
- Output
 - nodes, 350–352
 - setting animation, 365–366
 - sockets, 351–352
- Output Properties tab, Properties Editor, 358
- Overlays popover, 3D Viewport, 326

- P**
- Pack Islands tool, 203
- Packing images, 215
- Packing UVs, 202–203
- Painting
 - 3D models with 2D texture. *See* Unwrapping and UVs
 - with random brushes in character design, 88
 - software suggestions for, 79
 - texture. *See* Texture painting
 - weight, 284–286, 293
- Pan, navigating 3D scene, 31
- Panels, Blender UI, 24
- Panning, animation editor controls, 319
- Pants
 - modeling legs, 161
 - packing UVs, 202–203
 - unwrapping, 200–201
- Parallax effect, shooting video, 330
- Particle brushes, 3D texturing software, 222
- Particle Mode, hair simulation, 371–372
- Particles, creating simulations, 371
- Path object, 325
- Path tracing, Cycles, 233
- Pattern orientation, anatomy of markers, 335
- Patterns
 - anatomy of markers, 335
 - solving camera motion, 340
 - tracking features in footage, 337
- PBR (physically based rendering) materials, 215–216, 226–228
- Pen tablet, 284
- Personality
 - character design details, 82–83
 - designing character, 76
 - silhouettes to match character's, 78–79
- Perspective
 - camera tracking, 329–330
 - shift, 330–331
 - solving camera motion, 339–340
- Perspective/Orthographic switch, 3D scenes, 32
- Photographs, production stages, 73
- Physically based rendering (PBR) materials, 215–216, 226–228
- Pie menus, 24–26, 52
- Pin button, Material Properties tab, 238
- Pin icon, UV Editor, 187
- Pipeline, 69
- Piracy, commercial software, 5
- Pivot Point
 - 3D Viewport header, 30
 - rotating and scaling around, 45
 - using 3D cursor as, 36
 - UV Editor, 187
- Placing, 3D cursor, 37
- Planning, preproduction stage, 69–70
- Poke Faces tool, 115
- Poke tool, 115
- Pole target, 266
- Poly count, of good mesh topology, 128
- Poly to poly (poly2poly) modeling, 129
- Polygons. *See* Faces (polygons)
- Popovers, Blender UI, 23
- Pose Mode
 - adding constraints, 263
 - adding custom shapes to bones, 306
 - adjusting Rigify rig, 277
 - animating skeletons, 52
 - armature layers, 274
 - character rigging workflow, 256
 - disabling Deform option for bones, 280

- parenting objects to bone, 281
 - skinning jaw, 293
 - testing rig in, 268
 - working with armatures in, 261
 - Poses**
 - character, 312–315
 - walking animation, 322–324
 - Postproduction stage, 70–73, 74
 - Power property, lights, 62
 - Precision Mode, 45
 - Prefixes, naming bones using, 267–268
 - Preproduction stage, 69–73
 - Presets, Splash Screen, 15–16
 - Preview**
 - model, in 3D texturing software, 222
 - of results, in Compositor, 355–356
 - Preview panel, Material Properties tab, 238
 - Previous Frame, marker settings, 336
 - Principled BSDF shader
 - compatible with EEVEE and Cycles, 235
 - as mix of many basic shaders, 241
 - PBR materials, 227–228
 - selected when creating material, 229–230
 - shading character, 243–244
 - Private investment, Blender development, 10
 - Process nodes, 350–352
 - Production stage, 70–74
 - Project from View tool, UV Mapping, 190
 - Project overview
 - character-creation plan, 73–74
 - defining stages, 70–73
 - exercises, 74
 - summary, 74
 - three stages of project, 69–70
 - Projected from View option, proportional editing, 97
 - Properties**
 - animating within menus, 315
 - of material channels, 231–233
 - node, 351–352
 - Properties Editor
 - adding or subtracting objects from collection, 309
 - Armature Properties tab, 274
 - Armature tab, 267
 - basic material setup in, 235
 - Bone Properties tab, 259
 - defined, 20
 - managing datablocks, 50
 - Material tab, 225–226, 236
 - Modifier tab, 54
 - Object Data Properties tab, 50
 - Object Data tab, 132
 - Object Properties tab. *See Object Properties tab, Properties Editor*
 - Output Properties tab, 358
 - panels, 24
 - Protected Layers, 309–310
 - renaming objects, 49
 - Render Properties tab, 63–64, 248
 - Render tab, 57–58
 - shader selector for material in, 229
 - Texture Paint interaction mode, 208
 - Texture Paint Workspace, 206
 - Tool Properties tab, 258–259
 - Vertex Groups panel, 279
 - Workspace options, 22–23
 - World tab, 247–248
 - Proportional Editing and Snapping tools, UV Editor, 187
 - Proportional Editing tool
 - 3D Viewport header, 30
 - adjusting UVs, 198
 - blocking basic shape of face, 141
 - detailing backpack and jacket, 157
 - modeling boots, 163
 - selections in Edit Mode, 96–97
 - Protected Layers, 309–310
 - Proxies, animating linked character with, 310
 - Python Console editor, 19
 - Python scripting, 269, 376
- Q**
- Quads (four-sided faces)
 - defined, 93
 - good mesh topology using, 128
 - modeling boots, 163
- R**
- RAM, caching footage and, 334
 - Ray Visibility subpanel, Visibility panel, 245
 - Real-time cloth simulation, 372
 - Real-time preview rendering, 59

- Real-time render engine. *See* EEVEE (real-time render engine)
- Red Riding Hood*, 3
- Reference images
- fitting 3D models to character design, 86–88
 - loading for modeling, 131–134
 - placing texture elements with, 214
 - proper alignment of, 133
- Refine option
- configuring camera settings, 339
 - solving camera motion, 339–340
- Reflections, of materials, 226–227
- Refraction
- in EEVEE versus Cycles, 233
 - reducing effects of, 244
 - shading eyeballs in Cycles, 245
 - shading eyeballs in EEVEE, 244
- Refraction shader, 240
- Regenerating Rigify rig, 276–277
- Regions, 3D Viewport editor, 27–29
- Relax tool, LoopTools, 121
- Release Confirms option, Transforms, 46
- Remesh, 130
- Remesher tools, 374
- Renaming objects, 49
- Render
- character postproduction stage, 74
 - composite before, 349
 - creating Shadow Catcher in Cycles, 357
 - creating Shadow Catcher in EEVEE, 361–364
 - in Cycles, 250–252
 - defined, 63
 - in EEVEE, 248–249
 - enabling GPU in Cycles, 64
 - exercises, 366
 - exporting final, 365–366
 - final render in Cycles, 361
 - final render in EEVEE, 365
 - launching and saving, 65–66
 - launching final, 366
 - overview of, 63–64
 - real-time preview, 59
 - saving and loading .blend file, 64–65
 - scene in Cycles, 357–359
 - scene in EEVEE, 364
 - showing/hiding objects in final, 347–348
 - summary, 366
 - test scene setup, 246–248
- Render Animation, final render, 366
- Render Image, final render, 366
- Render Layers node, Cycles, 360
- Render Properties tab, Properties Editor
- rendering in Cycles, 250, 357–359
 - rendering in EEVEE, 248
 - rendering in EEVEE and Cycles, 63–64 with Simplify, 321
- Render tab, Properties Editor, 57–58, 375
- Render tests
- adding lights and environment, 246–248
 - in Cycles, 250–252
 - in EEVEE, 248–249
 - overview of, 246
- Rendered viewport shading mode
- displays result similar to final render, 58–59
 - enabling for scene lighting, 347
 - previewing light effects in EEVEE, 62
 - real-time preview rendering, 59
 - rendering in Cycles, 358
 - seeing how materials behave, 225
 - shading eyeballs in Cycles, 245
 - testing EEVEE and Cycles, 348–349
- Reset option, UV Mapping, 190
- Resetting, User Preferences, 39
- Resizing areas, 16
- Resolution
- image textures, 210
 - improving render results for EEVEE, 248
 - running render tests, 247
- Resolution, video
- FPS and, 334
 - in Graph Editor, 318–319
 - shooting for easy tracking, 331
 - in Timeline editor, 316
 - for walking animation, 323
- Resources
- Blender community forums/websites, 11
 - Blender Foundation and development, 10
 - testing development versions of Blender, 9
- Restriction toggles, 134, 348
- Retopology, 374, 375
- Retouches, final, 306–307
- Revert to Saved Preferences, 39
- RGB Curves node, 351

- Rig. *See* Rigify rig
- Rigging. *See* Character rigging
- Rigid body simulation, 372
- Rigify Buttons panel, Properties Editor, 276
- Rigify rig
- adding custom buttons, 303–305
 - adding details to, 179
 - adjusting 3D models, 270–272
 - adjustments to, 276–277
 - animating using character's, 311–312
 - armature layers, 274–275
 - armatures. *See* Armatures
 - bone groups, 273–274
 - Buttons panel, 273
 - character design, 80, 82–83
 - checking deformations, 287
 - creating skeleton, 269–270
 - enabling, 267
 - final retouches, 306–307
 - forward and inverse kinematics
 - automation, 264
 - generating rig, 257, 272–273
 - introducing, 268–269
 - modeling cap, 82–83, 168–172
 - modeling jacket, 154, 156–158
 - organizing bones, 273–274
 - posing character, 312–315
 - Python scripts for, 269
 - rigging process, 256–257
 - rigging your character. *See* Character rigging
 - shading character, 243
 - summary of process, 268
 - testing color schemes, 83–84
 - tips before you start, 267–268
 - understanding, 275–276
- Rigify skeleton (metarig), 257, 276–277
- Rings, selecting, 97–98
- Rip and Rip Fill tools, 115–116
- Roosendaal, Ton, 4, 6, 9
- Root, rig, 262
- Rotate tool, 43, 44–46
- Rotation
- animating character's rig, 311
 - side effects of mirroring bones, 291
- Rough material setting, PBR materials, 227–228
- Roughness (Glossiness) property, 232, 243
- Roughness, of materials, 226–227
- Rules, PBR materials, 226
- S**
- Sampling
- rendering in Cycles, 357–358
 - rendering in EEVEE, 248
- Sampling panel, Cycles, 250
- Save & Load tab, User Preferences, 38
- Saving
- .blend file, 64–65
 - image, 215
 - modified textures, 212
 - objects to prevent deletion by Blender, 321
 - rendered frames, 365
 - renders, 65–66
 - User Preferences, 39
 - UV exported as image, 218
 - Workspaces, 22
- Scale tool, 43, 44–46
- Screen Space Reflections, EEVEE, 63, 244, 248
- Script, adjusting Rigify, 303–305
- Scroll wheel, UI, 21
- Sculpt and Retopology modeling, 129–130
- Sculpt Mode, 3D Viewport, 374
- Sculpt tools, adjusting UVs, 198
- Sculpting, as creative way of modeling, 374
- Seams, 190–191, 193–194
- Search area, markers, 335
- Search menu, creating objects, 41
- Search, modeling tools, 95
- Select Boundary Loop tool, 99
- Select Loop Inner-Region tool, 99
- Select, material slots, 237
- Select menu, 3D Viewport header, 98–100
- Select Similar options, 99
- Select Split tool, 199
- Selection mode, UV Editor, 186
- Selection, object, 33–35
- Selections, making
- border selection, 98
 - Checker Deselect tool, 100
 - grow and shrink selection, 98
 - Linked Flat Faces option, 99
 - linked selection, 97
 - loops and rings, 97–98
 - other selection methods, 100

- proportional editing, 96–97
- Select Boundary Loop tool, 99
- Select Loop Inner-Region tool, 99
- Select Similar options, 99
- Shortest Path Selection, 95–96
- vertices, edges, and faces, 94
- Separate tool, 116, 158
- Separating UVs, 199
- Settings panel, Material Properties, 240, 244
- Setup Tracking Scene button, camera motion, 342
- Shade Smooth, 53, 135–136
- Shader Editor
 - accessing nodes, 235
 - defined, 19
 - using nodes, 352
- Shaders
 - materials made up of, 225
 - and Mix shaders, 229–230
 - mixing and adding, 241–242
 - most-used, 240–241
 - overview of, 240
 - Surface Shaders options, 31–33
- Shading
 - adding materials, 225
 - Viewport, 58–60
- Shading your character
 - adding several materials to object, 236–238
 - basic steps for, 243–244
 - loading textures, 242–243
 - Material Properties tab, 238–240
 - mixing and adding shaders, 241–242
 - most-used shaders, 240–241
 - overview of, 236
 - production stage, 73
 - shading character, 243–246
 - shading eyeballs in Cycles, 245
 - shading eyeballs in EEVEE, 244
- Shadow catcher
 - creating in Cycles, 357
 - creating in EEVEE, 361–364
- Shadows
 - analyzing real footage before adding lights, 345–346
 - creating and testing scene lighting, 346–347
 - more accurate in Cycles than EEVEE, 251
- Shadows panel, 248
- Shape Key Editor, Dope Sheet, 317
- Shape keys
 - configuring drivers, 300–301
 - creating facial rig, 288
 - mirroring shapes, 298
 - modeling, 295–297
- Shapes
 - blocking basic face, 140–142
 - creating custom, 305–306
 - creating rig with Rigify by adjusting skeleton, 268
 - defining face's, 142–144
 - good mesh topology follows the, 129
 - mirroring, 298
- Shield button, datablocks, 51
- Shoes. *See* Boots
- Shortest Path tool, 95–96, 194
- Shoulders
 - adding details to suit, 82
 - defining arms and torso, 154–155
 - detailing backpack and jacket, 157
 - modeling torso and arms, 151–154
 - requiring complex deformations, 128
 - unwrapping rest of character, 201
- Show/Hide
 - objects in render, 347–348
 - unused sockets, 354
- Show Wire option, weight painting, 286
- Shrink/Fatten tool, modeling with, 116–117
- Shrink selection, 98
- Shrinkwrap modifier, skinning process, 282
- Side parameter, setting up reference images, 133
- Sidebar, 3D Viewport
 - accessing bone properties, 258
 - animating character's rig, 311
 - as editor region, 28
 - LoopTools in, 120–121
 - posing character with, 312–315
 - tabs, 27
- Threshold option, Auto Merge, 123
- using menus for transforms in, 47–48
- X-Axis Mirror option, 258
- Sidebar, Movie Clip Editor
 - configuring camera settings, 339
 - loading footage, 332
 - marker settings, 336

- Sidebars, panels, 24
- Silhouettes, 78–79, 88
- Simplify option, animating with, 321
- Simulate modifier, 54
- Simulations, types of, 371–372
- Single bone, 264–267
- Single Image, Texture Slots panel, 212–213
- Sintel* (2010), 6
- Skeleton panel, 274–275
- Skeletons. *See also* Character rigging
 - 3D cursor poses characters without, 157
 - adding in character production, 73
 - adjusting to 3D model, 270, 271–272
 - animating in Pose Mode, 52
 - armatures in Blender as. *See* Armatures
 - creating, 269–270
 - generating rig, 272–273
 - made of datablocks, 49
 - not needed for posing with 3D cursor, 36
 - using Skin modifier for thickness, 88–89
- Skin modifier, character design, 88–89
- Skinning
 - adding armature modifier, 283–284
 - chest badge deformation, 294–295
 - control face shapes with drivers, 299–303
 - creating face controls, 298–299
 - creating facial rig, 288
 - defined, 278
 - defining weights, 284–287
 - enabling deformator bones only, 281
 - eyes and jaw, 292–294
 - knowing which objects don't need weights, 281–283
 - mirroring bones, 291
 - mirroring eye rig, 290
 - mirroring shapes, 298
 - modeling shape keys, 295–297
 - naming bones automatically, 290
 - organizing facial rig, 303–305
 - rigging eyes, 288–289
 - rigging jaw, 291–292
 - setting up model for, 280–281
 - vertex groups, 278–279
 - vertex weights and, 278
- Skip option, Shortest Path Selection, 96
- Sky Texture, render test scene, 247
- Slide tool, 117
- Smart UV Project tool, 190
- Smooth interpolation, Graph Editor, 318
- Smooth lines, placing texture elements, 214
- Smooth material setting, PBR materials, 227–228
- Smooth surfaces, 53
- Smooth Vertices tool
 - adding wrist to hand, 166–167
 - blocking basic shape of face, 141
 - modeling basic hand shape, 164
 - modeling with, 118
- Snap menu, 3D cursor, 36
- Snap Mode, 45
- Snap tools, adjusting UVs, 198
- Snapping, 30, 124
- Snapping tool, 123, 375
- Sockets, node, 351–352
- Soft body simulation, 372
- Software, texturing
 - 2D image-editing, 217–221
 - 3D, 221–222
 - overview of, 216
- Solid viewport shading mode, 58–59, 237–238, 286
- Solidify modifier, skinning process, 282
- Solidify tool
 - detailing backpack and jacket, 157
 - modeling cap, 170–172
 - modeling eyebrows, 176–177
 - shaping locks of hair, 173
 - working with, 118
- Solve Camera Motion button, 339–340
- Solve tab, camera motion, 339–340, 341–342
- Source code, 5
- Space selector, configuring drivers, 301
- Space tool, 121
- Specular-Glossiness workflow, PBR materials, 227–228
- Specular property, materials, 232
- Speed option, monitoring tracking, 338
- Sphere Projection tool, UV mapping, 190
- Spider-Man 2*, 3
- Spin tool, 118–119
- Splash Screen, Blender UI, 15–16
- Split tool, 119
- Splitting areas, 17
- Spring* (2019), 6–8

- Squares, in good mesh topology, 128
- Stabilization, camera, 331
- Stabilize Stroke option, drawing, 214
- Startup file, creating own, 39–40
- Static features, shooting video, 331–332
- Status Bar, 16, 211
- Steady lines, drawing, 214
- Sticky Selection mode, UVs, 186, 199
- Stitch tool, 199
- Stroke options, drawing, 214
- Style, character design, 77
- Subdivide tool, modeling, 119–120, 164
- Subdivision Surface modifier
- adding details to cap, 170–172
 - adding ears, 147–148
 - adding fingers to hand, 166–167
 - adding neck to jacket, 159
 - adding to your object, 55–56
- Armature modifier and performance of, 283
- creating eyeball, 135–136
- keyboard shortcut, 57
- modeling base of cap, 168–169
- modeling eyebrows, 176
- poly to poly modeling, 129
- rendering animation with Simplify, 321
- smoothing shapes with, 144
- texturing with 2D image-editing software, 218
- viewing geometry when modeling, 148
- Substance Painter (3D texturing software), 221
- Subtracting objects from selection, 33
- Sun light
- creating and testing scene lighting, 347
 - rendering in EEVEE, 248
- Surface Deform modifier, chest badge, 294–295
- Surface Shaders, 60–61, 239
- Surfaces, flat or smooth, 53
- Swapping areas, 17–18
- System tab, User Preferences, 38
- T**
- T pose, modeling arms in, 151
- Target bone, 262, 263
- Tears of Steel* (2012), 6–7
- Teeth
- building inside of mouth for, 149
 - modeling, 178
 - using parenting to make, 293
- Tension, good mesh topology and, 128
- Test grid, UVs, 194–196, 198
- Test object, 42
- Testing
- camera tracking, 343
 - EEVEE and Cycles for lighting scene, 348–349
 - lights for your scene, 346–347
 - rig, in Pose Mode, 268
 - skinning process, 293
- Texel density, 191
- Text Editor, 19, 303–305
- Text info, 3D Viewport editor, 26–27
- Texture
- character production stage, 73
 - definition of, 205
 - displaying UV text grid in model, 195–196
 - Maps Baking bakes lighting into, 375
 - modeling boots, 163
 - packing UVs, 202–203
 - painting boots, 163
 - procedural, 233–234
 - UV placement and seams, 190–191
- Texture Node Editor, 19, 352
- Texture Paint interaction mode, 207–209, 210
- Texture Paint Mode, 206–209, 210–213, 284
- Texture Paint Workspace, 206–207
- Texture painting
- with 3D texturing software, 221–222
 - conditions for, 210–211
 - creating base texture, 214–215
 - elements of texture, 215–216
 - exercises, 224
 - limitations of Texture Paint Mode, 213
 - main workflows for, 205–206
 - one object at a time, 209
 - with other software, 216–221
 - overview of, 205
 - seeing painted character, 223
 - summary, 223–224
- Texture Paint interaction mode, 207–209
- Texture Paint Workspace, 206–207

- texture slots, 212–213
 - before you begin, 209–210
 - Texture Slots panel, 211, 212–213, 215
 - Themes tab, User Preferences, 38
 - “Think twice, work half,” as preproduction stage, 70
 - Threshold option, Auto Merge, 123
 - Timeline editor
 - animation with, 316–317
 - creating walking animation, 323
 - defined, 19
 - Timeline, Movie Clip Editor, 333–334
 - Tongue, modeling, 178
 - Tool Properties tab, Properties Editor, 258–259
 - Tool Settings
 - 3D Viewport editor, 26
 - 3D Viewport header, 29
 - Weight Paint Mode, 286
 - Toolbar
 - 3D Viewport editor, 27
 - accessing modeling tools, 95
 - as editor region, 28
 - Movie Clip Editor, 332, 336
 - Tools
 - modeling. *See* Modeling tools
 - UV Mapping, 188–190
 - Top Bar
 - defined, 16
 - in Weight Paint Mode, 284
 - Workspaces, 21
 - Topology
 - defining muscles in arms and torso, 154
 - mesh, 127–129
 - retopology as recreating shape with new, 375
 - studying face, 139–140
 - Torso and arms, modeling
 - adding neck to jacket, 159
 - basic shapes, 152–154
 - blocking basic shapes, 152
 - defining, 154–155
 - detailing backpack and jacket, 156–158
 - finishing belt, 158–159
 - overview of, 150–151
 - Track Backward features, 337
 - Track Forward features, 337
 - Track To constraint, 263, 288–289
 - Tracked frames, anatomy of markers, 336
 - Tracked markers, camera, 329–330
 - Tracking camera motion, loading footage, 333–334
 - Transform Orientation, 3D Viewport header, 29, 45
 - Transform panel, 47–48
 - Transform property, 301
 - Transform tool, 43
 - Transforms
 - animating rig, 311
 - arranging objects in scene, 48–49
 - defined, 30
 - numerically precise, 46–47
 - Release Confirms option, 46
 - simplifies skinning process, 283
 - using Active Tools, 42–43
 - using keyboard shortcuts (advanced), 46–47
 - using manipulators, 44–45
 - using menus, 47–48
 - Transmission (refraction) property, materials, 232
 - Transparency
 - rendering background in Cycles, 358–359
 - rendering in Cycles, 250
 - rendering in EEVEE, 248
 - setting up reference images, 132
 - Transparent BSDF shader, 240
 - Triangles, faces as, 93–94
- U**
- UI. *See* User interface (UI)
 - Unwrap tool, 188
 - Unwrapping and UVs
 - exercises, 204
 - how they work, 183–184
 - introduction, 183
 - packing UVs after, 202–203
 - separating and connecting UVs, 199
 - summary, 203
 - texturing after unwrapping, 205–206
 - texturing before unwrapping, 205
 - unwrapping rest of character, 200–201
 - before you begin painting texture, 209–210
 - Unwrapping and UVs, unwrapping
 - accessing unwrapping menus, 188

- considerations before unwrapping, 191–193
- defining seams, 190–191
- navigating UV Editor, 187–188
- overview of, 184–185
- using UV Editor, 185–187
- UV mapping tools, 188–190
- Unwrapping and UVs, UVs**
 - adjusting UVs, 198–199
 - creating and displaying UV test grid, 194–196
 - marking seams, 193–194
 - overview of, 193
 - reviewing finished face's UVs, 200
 - separating and connecting UVs, 199–200
 - unwrapping character's face, 196–197
 - using Live Unwrap, 197–198
- Unwrapping, character production stage, 73
- Update Dependencies**, configuring drivers, 300
- Use Nodes option**, Compositor, 352–353
- User interface (UI)**
 - 3D cursor, 35–37
 - 3D viewport, 26–30
 - areas of, 16–18
 - creating own startup file, 39–40
 - default editors, 16
 - downloading and installing Blender, 13
 - editor types, 18–21
 - exercises, 40
 - menus and popovers, 23
 - navigating 3D scene, 31–33
 - panels, 24
 - pie menus, 24–26
 - recommended hardware, 13–14
 - selecting objects, 33–35
 - Splash Screen, 15–16
 - Status Bar, 16
 - summary, 40
 - Top Bar, 16
 - User Preferences, 37–39
 - viewing hidden menu or header in, 21
 - workspaces, 21–23
- User Preferences**
 - disabling Splash Screen in, 16
 - enabling F2 add-on, 122
 - overview of, 37
- resetting, 39
- saving, 39
- switching Walk Mode/Fly Mode in, 33
- tabs, 21, 38
- UV Editor**
 - defined, 19
 - Live Unwrap in, 197–198
 - overview of, 185–189
- UV menu**, 187, 188
- UV Sphere**, 135–136, 168–169
- UV Sync Selection mode**, 186, 199
- UV test grid**, 194–196, 210
- UVs**. *See also Unwrapping and UVs*
 - conditions for texture painting, 211
 - defined, 183
 - texturing in 2D image-editing software, 217–218

V

- Values**, weight, 287
- Variables**, drivers, 300
- Vector imaging software**, character design, 88
- Vector setting**
 - loading textures, 242
 - shading character, 243
- Vertex groups**
 - adding new bones to deformation, 292–293
 - skinning, 278–279
 - working in Weight Paint Mode, 284–285
- Vertex Groups menu**, 293
- Vertex Groups panel**, 279, 287
- Vertex menu**, 141
- Vertex weights**, 278–279
- Vertex Weights panel**, 287
- Vertices**
 - defined, 93
 - joining with Connect tool, 104–105
 - keep adjusting when adding, 164
 - selecting, 94
 - working with, 93–95
- Video**
 - editing, 374
 - editing tools, 374
 - export animation as, 366
 - recording, 74

- shooting for easy tracking, 330–331
- stabilization, in Movie Clip Editor, 374
- Video Sequence Editor, 19, 374
- View Layer Properties tab, Property Editor, 358
- View menu, navigating from, 33
- View Selected, 3D scenes, 31
- Viewer node, 355, 360
- Viewport Display, 237–238, 240
- Viewport Gizmos, 30, 44
- Viewport Overlays, 30
- Viewport shading, 30, 58–60
- Viewport tab, User Preferences, 38
- Visibility panel, 245
- Visual-effects film, 72
- Volume Absorption shader, 240
- Volume options, Material Properties tab, 239
- Volume Scatter shader, 240
- Volumetrics, 31–33

- W**
- Walk-Cycle action, NLA Editor, 324–325
- Walk cycle, animating
 - along a path, 325–326
 - creating action, 321
 - creating poses, 322–324
 - Non-Linear Animation Editor repeats, 319
 - repeating, 324–325
 - tips, 321
 - using character's rig, 311
- Walk Mode, 32–33, 63
- Warcraft, 3
- Weight
 - adding Armature modifier with automatic, 283–284
 - deformations and characters, 287–288
 - knowing what objects do not need, 281–283
 - making skinning process easier, 282–283
 - painting, 284–286
 - skinning eyes and jaw, 293
 - values, 287
- vertex, 278
- vertex groups and, 278–279
- Weight Paint interaction mode, 279
- Weight Paint Mode, 284–286, 293
- Weight Paints visualization, 286
- What Every BODY Is Saying: An Ex-FBI Agent's Guide to Speed-Reading People* (Navarro and Karlins), 76
- Whole Character Keying Set, 314, 323
- Wireframe viewport shading mode, 58–59, 277
- Workbench Engine, viewport shading, 58
- Workbench, rendering images, 57–58
- Workflows
 - character rigging, 256–257
 - defined, 69
 - PBR, 227–228
 - using textures on 3D models, 205–206
- Workspaces, 21–23
- World light, scene lighting, 347
- World Properties tab, Properties Editor, 347
- World tab, Properties Editor, 247–248
- Wrist, adding to hand, 166–168

- X**
- X-axis, 290
- X-Axis Mirror option, 258–259
- X-Mirror option, Tool Settings, 286
- X-Ray
 - 3D Viewport header, 30
 - modeling with, 125
 - weight painting with, 286

- Y**
- Yo Frankie! (2008)* video game, 6

- Z**
- Zoom
 - animation editor controls, 319
 - avoid when shooting video for easy tracking, 331
 - navigating 3D scene, 31–33

This page intentionally left blank

Game Development & Design Books, eBooks & Video



If you want to develop games, you need strong experience with modern best practices and professional tools. We are working with the best in the business to provide guidance on the programming languages, platforms, and graphics tools you need to know to build great games.

- General design & development
- Roblox
- Unity
- Unreal Engine
- AR & VR
- Blender
- Godot Engine
- Video Courses

Visit [informit.com/gamedev](https://www.informit.com/gamedev) to read sample chapters, shop, and watch video lessons from featured products.



Addison-Wesley · Adobe Press · Cisco Press · Microsoft Press · Pearson IT Certification · Que · Sams · Peachpit Press





Photo by izusek/gettyimages

Register Your Product at informit.com/register

Access additional benefits and **save 35%** on your next purchase

- Automatically receive a coupon for 35% off your next purchase, valid for 30 days. Look for your code in your InformIT cart or the Manage Codes section of your account page.
- Download available product updates.
- Access bonus material if available.*
- Check the box to hear from us and receive exclusive offers on new editions and related products.

*Registration benefits vary by product. Benefits will be listed on your account page under Registered Products.

InformIT.com—The Trusted Technology Learning Source

InformIT is the online home of information technology brands at Pearson, the world's foremost education company. At InformIT.com, you can:

- Shop our books, eBooks, software, and video training
- Take advantage of our special offers and promotions (informit.com/promotions)
- Sign up for special offers and content newsletter (informit.com/newsletters)
- Access thousands of free chapters and video lessons

Connect with InformIT—Visit informit.com/community



Addison-Wesley • Adobe Press • Cisco Press • Microsoft Press • Pearson IT Certification • Que • Sams • Peachpit Press

