Sams Teach Yourself Mod Development for Minecraft® in 24 Hours
Jimmy Koene

Sams Teach Yourself
Mod Development for Minecraft®
in 24 Hours

SAMS
800 East 96th Street, Indianapolis, Indiana, 46240 USA
# Contents at a Glance

Introduction .............................................................................................................. 1

## Part I: Introduction

**HOUR 1** Setting Up the Minecraft Development Environment .................. 7

1. Creating the Basics for Forge ................................................................. 21
2. Working with Recipes and Other Small Modifications .............................. 39

## Part II: Items

**HOUR 4** Making Your First Item ................................................................. 53

4. Creating Multiple Items in a Smart Way .............................................. 67
5. Cooking Up a Food Item ........................................................................ 83
6. Making Your Own Tools .......................................................................... 95
7. Creating Armor ........................................................................................ 111

## Part III: Blocks

**HOUR 9** Making Your First Block ............................................................... 123

9. Creating Multiple Blocks in a Smart Way ............................................. 135
10. Making Your Blocks Unique ................................................................. 145
11. Creating a Tile Entity ............................................................................... 175

## Part IV: World Generation

**HOUR 13** Generating Ores ........................................................................... 195

13. Generating Plants .................................................................................. 207
14. Using MCEdit ........................................................................................ 223
15. Generating Your Structure ..................................................................... 231

## Part V: Entities

**HOUR 17** Learning About Entities ............................................................... 245

17. Creating an Entity Model Using Techne ............................................. 253
19  Coding a Mob ................................................................. 265
20  Creating a Throwable .................................................. 283

Part VI: Final Pointers

HOUR 21  Editing Vanilla Minecraft .................................. 297
22  Structuring Your Mod .................................................... 311
23  Releasing Your Mod ....................................................... 325
24  What’s Next ................................................................. 333

APPENDIX A  Additional Code Fragments ......................... 341
INDEX ................................................................................. 395
# Table of Contents

## Introduction

- Who Should Read This Book ......................................................... 1
- How This Book Is Organized ....................................................... 2
- Conventions Used ........................................................................ 4
- Downloading the Code and Resources ........................................ 5

## Part I: Introduction

### HOUR 1: Setting Up the Minecraft Development Environment 7

- Understanding How Minecraft Is Written and What You Will Do with It ........ 7
- Learning About Forge ..................................................................... 8
- Setting Up the JDK ........................................................................ 9
- Setting Up Eclipse ......................................................................... 16
- Setting Up Forge ........................................................................... 16
- Summary ...................................................................................... 19
- Q&A ............................................................................................ 19
- Workshop .................................................................................... 20
- Exercises ..................................................................................... 20

### HOUR 2: Creating the Basics for Forge 21

- Understanding the Java in the ExampleMod ................................... 21
- Creating Your Own Package ......................................................... 27
- Creating Your Own Class File ...................................................... 31
- Creating the Mod File ................................................................... 34
- Summary .................................................................................... 35
- Q&A ............................................................................................ 35
- Workshop .................................................................................... 36
- Exercises ..................................................................................... 36
HOUR 3: Working with Recipes and Other Small Modifications 39
Learning About Recipes in Minecraft ................................. 39
Crafting a Recipe ................................................................ 40
Creating a Shapeless Recipe ............................................... 44
Creating a Smelting Recipe ............................................... 45
Using Special ItemStacks .................................................... 46
Changing the Mob Spawn in a Dungeon ................................. 48
Changing Chest Items ......................................................... 49
Examining the Results in the Mod File ................................. 49
Summary ........................................................................... 51
Q&A ................................................................................... 51
Workshop ......................................................................... 51
Exercises ......................................................................... 52

Part II: Items

HOUR 4: Making Your First Item 53
Understanding What an Item Is ........................................... 53
Creating a Basic Item .......................................................... 54
Summary ........................................................................... 65
Q&A ................................................................................... 65
Workshop ......................................................................... 65
Exercises ......................................................................... 66

HOUR 5: Creating Multiple Items in a Smart Way 67
Adding Multiple Items Using the Same File ......................... 67
Adding Metadata to Items .................................................. 72
Summary ........................................................................... 81
Q&A ................................................................................... 81
Workshop ......................................................................... 81
Exercises ......................................................................... 82

HOUR 6: Cooking Up a Food Item 83
Understanding Food in Minecraft ....................................... 83
Creating a Food Item .......................................................... 84
Adding a Potion Effect .............................................................................. 89
Summary .................................................................................................. 91
Q&A......................................................................................................... 91
Workshop ............................................................................................... 92
Exercises ............................................................................................... 92

**HOUR 7: Making Your Own Tools** .................................................. 95
Creating a ToolMaterial ........................................................................ 95
Creating a Pickaxe .................................................................................. 96
Creating Special Tools for Harvesting Different Blocks ...................... 103
Summary ............................................................................................... 108
Q&A......................................................................................................... 108
Workshop ............................................................................................... 108
Exercises ............................................................................................... 109

**HOUR 8: Creating Armor** ................................................................ 111
Creating Your ArmorMaterial .......................................................... 111
Creating the Armor ............................................................................... 112
Customizing Armor ............................................................................... 119
Summary ............................................................................................... 119
Q&A......................................................................................................... 120
Workshop ............................................................................................... 120
Exercises ............................................................................................... 121

**Part III: Blocks**

**HOUR 9: Making Your First Block** ................................................. 123
Understanding What a Block Is .......................................................... 123
Creating a Basic Block .......................................................................... 124
Adding a Block Texture ......................................................................... 129
Summary ............................................................................................... 132
Q&A......................................................................................................... 132
Workshop ............................................................................................... 133
Exercises ............................................................................................... 134
HOUR 10: Creating Multiple Blocks in a Smart Way

Adding Metadata to Blocks .............................................. 135
Adding Block Textures ......................................................... 139
Summary ................................................................................. 143
Q&A ....................................................................................... 144
Workshop ............................................................................ 144
Exercises .............................................................................. 144

HOUR 11: Making Your Blocks Unique

Adding Sided Textures ......................................................... 145
Making Half Blocks ............................................................... 148
Creating a Plant ..................................................................... 152
Summary ................................................................................. 171
Q&A ....................................................................................... 171
Workshop ............................................................................ 172
Exercises .............................................................................. 172

HOUR 12: Creating a Tile Entity

Understanding What a Tile Entity Is .................................. 175
Creating a Tile Entity Block ................................................. 176
Creating a Tile Entity ........................................................... 180
Server Synchronization ....................................................... 183
Summary ................................................................................. 193
Q&A ....................................................................................... 193
Workshop ............................................................................ 193
Exercises .............................................................................. 194

Part IV: World Generation

HOUR 13: Generating Ores

Using Forge and an EventHandler .................................... 195
Creating a SamEventHandler .............................................. 195
Using OreDictionary ............................................................ 202
Summary ................................................................................. 204
Q&A ....................................................................................... 204
<table>
<thead>
<tr>
<th>Workshop</th>
<th>Exercises</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>205</td>
</tr>
<tr>
<td><strong>HOUR 14: Generating Plants</strong></td>
<td>207</td>
</tr>
<tr>
<td>Creating a Custom WorldGen Class</td>
<td>207</td>
</tr>
<tr>
<td>Adding Code to SamEventHandler</td>
<td>215</td>
</tr>
<tr>
<td>Summary</td>
<td>220</td>
</tr>
<tr>
<td>Q&amp;A</td>
<td>220</td>
</tr>
<tr>
<td>Workshop</td>
<td>220</td>
</tr>
<tr>
<td>Exercises</td>
<td>221</td>
</tr>
<tr>
<td><strong>HOUR 15: Using MCEdit</strong></td>
<td>223</td>
</tr>
<tr>
<td>Why Use MCEdit</td>
<td>223</td>
</tr>
<tr>
<td>Getting MCEdit</td>
<td>223</td>
</tr>
<tr>
<td>Using MCEdit</td>
<td>225</td>
</tr>
<tr>
<td>Summary</td>
<td>230</td>
</tr>
<tr>
<td>Q&amp;A</td>
<td>230</td>
</tr>
<tr>
<td>Workshop</td>
<td>230</td>
</tr>
<tr>
<td>Exercises</td>
<td>230</td>
</tr>
<tr>
<td><strong>HOUR 16: Generating Your Structure</strong></td>
<td>231</td>
</tr>
<tr>
<td>Installing the Schematic Converter</td>
<td>231</td>
</tr>
<tr>
<td>Using the Schematic Converter</td>
<td>232</td>
</tr>
<tr>
<td>Adding Mobs to Your Structure</td>
<td>240</td>
</tr>
<tr>
<td>Summary</td>
<td>243</td>
</tr>
<tr>
<td>Q&amp;A</td>
<td>243</td>
</tr>
<tr>
<td>Workshop</td>
<td>244</td>
</tr>
<tr>
<td>Exercises</td>
<td>244</td>
</tr>
<tr>
<td><strong>Part V: Entities</strong></td>
<td></td>
</tr>
<tr>
<td><strong>HOUR 17: Learning About Entities</strong></td>
<td>245</td>
</tr>
<tr>
<td>Understanding What an Entity Is</td>
<td>245</td>
</tr>
<tr>
<td>Creating Proxies</td>
<td>246</td>
</tr>
<tr>
<td>Creating a Basic Entity</td>
<td>248</td>
</tr>
<tr>
<td>Summary</td>
<td>251</td>
</tr>
</tbody>
</table>
HOUR 18: Creating an Entity Model Using Techne

What Are Models? ........................................................................................................ 253
Downloading and Installing Techne ........................................................................ 254
Using Techne .............................................................................................................. 255
Summary ...................................................................................................................... 263
Q&A ............................................................................................................................. 263
Workshop .................................................................................................................... 263
Exercises ..................................................................................................................... 264

HOUR 19: Coding a Mob

Rendering the Mob ...................................................................................................... 265
Making the Mob Spawn .............................................................................................. 272
Making EntitySamMob a Mob .................................................................................... 275
Summary ...................................................................................................................... 281
Q&A ............................................................................................................................. 281
Workshop .................................................................................................................... 282
Exercises ..................................................................................................................... 282

HOUR 20: Creating a Throwable

Registering a Throwable Entity .................................................................................. 283
Creating the Entity Class ......................................................................................... 284
Using the Entity .......................................................................................................... 287
Rendering the Entity ................................................................................................. 290
Summary ...................................................................................................................... 293
Q&A ............................................................................................................................. 293
Workshop .................................................................................................................... 294
Exercises ..................................................................................................................... 294
### Part VI: Final Pointers

#### HOUR 21: Editing Vanilla Minecraft  
297
- How to Change Minecraft Indirectly ........................................ 297
- Other Events ........................................................................... 303
- Giving Your Armor Egg-Throwing Abilities .............................. 303
- Summary .................................................................................. 308
- Q&A ......................................................................................... 308
- Workshop .................................................................................. 309
- Exercises ................................................................................... 309

#### HOUR 22: Structuring Your Mod  
311
- Why It Is Important to Structure Your Mod .............................. 311
- How to Structure Your Mod ........................................................ 311
- Summary ................................................................................... 322
- Q&A ......................................................................................... 322
- Workshop ................................................................................... 322
- Exercises ................................................................................... 323

#### HOUR 23: Releasing Your Mod  
325
- Exporting Your Mod with Forge .................................................. 325
- Making Your Mod Public ............................................................... 328
- Summary ................................................................................... 329
- Q&A ......................................................................................... 330
- Workshop ................................................................................... 331
- Exercises ................................................................................... 331

#### HOUR 24: What’s Next  
333
- Using IRC .................................................................................. 333
- Using Open Source Mods ............................................................ 336
- Learning Java ............................................................................ 338
- Summary ................................................................................... 339
- Q&A ......................................................................................... 339
- Workshop ................................................................................... 339
- Exercises ................................................................................... 339
## APPENDIX A: Additional Code Fragments

<table>
<thead>
<tr>
<th>Code Fragment</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configs</td>
<td>341</td>
</tr>
<tr>
<td>Rotating Blocks</td>
<td>342</td>
</tr>
<tr>
<td>Custom Block Rendering</td>
<td>343</td>
</tr>
<tr>
<td>Custom Tree Generation</td>
<td>350</td>
</tr>
<tr>
<td>Automatic Update Checking</td>
<td>353</td>
</tr>
<tr>
<td>Custom Creative Tabs</td>
<td>356</td>
</tr>
<tr>
<td>Custom TNT</td>
<td>357</td>
</tr>
<tr>
<td>Changing Mob Behavior</td>
<td>363</td>
</tr>
<tr>
<td>Clothes</td>
<td>364</td>
</tr>
<tr>
<td>Armor Effects</td>
<td>365</td>
</tr>
<tr>
<td>Creating a Custom Dimension</td>
<td>367</td>
</tr>
<tr>
<td>Custom Biomes</td>
<td>383</td>
</tr>
<tr>
<td>Custom AI</td>
<td>384</td>
</tr>
<tr>
<td>Spawning Particles</td>
<td>390</td>
</tr>
<tr>
<td>Playing Sounds</td>
<td>391</td>
</tr>
<tr>
<td>Using a Model for an Item</td>
<td>391</td>
</tr>
</tbody>
</table>

## Index

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configs</td>
<td>341</td>
</tr>
<tr>
<td>Rotating Blocks</td>
<td>342</td>
</tr>
<tr>
<td>Custom Block Rendering</td>
<td>343</td>
</tr>
<tr>
<td>Custom Tree Generation</td>
<td>350</td>
</tr>
<tr>
<td>Automatic Update Checking</td>
<td>353</td>
</tr>
<tr>
<td>Custom Creative Tabs</td>
<td>356</td>
</tr>
<tr>
<td>Custom TNT</td>
<td>357</td>
</tr>
<tr>
<td>Changing Mob Behavior</td>
<td>363</td>
</tr>
<tr>
<td>Clothes</td>
<td>364</td>
</tr>
<tr>
<td>Armor Effects</td>
<td>365</td>
</tr>
<tr>
<td>Creating a Custom Dimension</td>
<td>367</td>
</tr>
<tr>
<td>Custom Biomes</td>
<td>383</td>
</tr>
<tr>
<td>Custom AI</td>
<td>384</td>
</tr>
<tr>
<td>Spawning Particles</td>
<td>390</td>
</tr>
<tr>
<td>Playing Sounds</td>
<td>391</td>
</tr>
<tr>
<td>Using a Model for an Item</td>
<td>391</td>
</tr>
<tr>
<td>Index</td>
<td>395</td>
</tr>
</tbody>
</table>
About the Author

Jimmy Koene maintains one of the world’s most popular Minecraft mod websites (www.wuppy29.com/minecraft/modding-tutorials/forge-modding/). He is one of the only Minecraft modders who consistently maintains and updates his online tutorials. Also an expert Java and C++ programmer, he has written an Android Minecraft app that has been downloaded more than 30,000 times along with various other smaller apps. He also maintains several Minecraft mods that cover many different aspects of Minecraft, ranging from slight changes to how food works to mods that completely change the way you play.
Dedication

To the Minecraft mod community, who make Minecraft one of the best-selling and amazing games there is. Thanks to the community I started modding, which led to the writing of this book.

Acknowledgments

First of all, I want to thank Rick Kughen, who was the one who approached me about writing a book about Minecraft mod development. Before he mailed me with his ideas, I never thought this would be possible. One of the biggest problems with developing mods for a game such as Minecraft is that the code you are working with will constantly change. Every update of Minecraft or Forge, which is the application programming interface (API) used in this book, may change quite a lot of the code. Thankfully, eBooks can be easily updated and Rick told me about the possibility of having an online database where I could add updates and changes to the book frequently. This made it possible to write this book.

Some other people I should thank are Markus “Notch” Persson, who created Minecraft. Without his idea, I would never have started programming and this book would never have been written. There are many others who helped create Minecraft and the modding community. Some of those people are Jeb, Dinnerbone, Searge, ProfMobius, R4wk, CodeWarrior, ZeuX, Risugami, cpw, AbrarSyed, and LexManos. These are the creators of MCP, MCEdit, Techne, ModLoader, and Forge. All of these things are (indirectly) used in this book. Finally, I would like to thank everybody who worked on the book. This includes the editors, Mark Renfrow and Rick Kughen, the reviewers, and everybody else who did anything related to the book. Without all of these people, this book would never have been possible. I especially want to thank LexManos, the creator of Forge. His expertise in this field helped a lot to improve this book.
We Want to Hear from You!

As the reader of this book, you are our most important critic and commentator. We value your opinion and want to know what we’re doing right, what we could do better, what areas you’d like to see us publish in, and any other words of wisdom you’re willing to pass our way.

We welcome your comments. You can email or write to let us know what you did or didn’t like about this book—as well as what we can do to make our books better.

Please note that Pearson cannot help you with technical problems related to the topic of this book. However, the author has a Google User Group for this. You can access this group using the following link: http://bit.ly/1wpMwef.

When you write, please be sure to include this book’s title and author as well as your name and email address. We will carefully review your comments and share them with the author and editors who worked on the book.

Email:   consumer@samspublishing.com

Mail:   Sams Publishing
        ATTN: Reader Feedback
        800 East 96th Street
        Indianapolis, IN 46240 USA

Reader Services

Visit our website and register this book at informit.com/register for convenient access to any updates, downloads, or errata that might be available for this book.
Introduction

Minecraft is a great game. It’s one of the best-selling PC games, and it’s exceedingly popular. One of the many reasons why it is so popular is because there are mods. If you picked up this book, you likely already have an interest in Minecraft and you might have already played with some mods. This book teaches you how to create these mods yourself.

Creating mods yourself has two big advantages. First, you can improve the game in the way you want, shaping the mod to exactly fit your needs and making the game more fun for you to play. Second, modding is an easy way to get into programming, which is a booming industry, and, according to many people (including me), modding is a really interesting thing to do.

By the end of this 24-hour book, you will be able to create your own mods and be well on your way to programming with Java. Each hour covers an important aspect of Minecraft mod development, including various items, blocks, and entities. You will learn a lot about each of these subjects by following along with each hour, actively participating in the hands-on Try It Yourself activities, and completing the Workshops and Exercises at the end of each hour. You will also gain more information on how to use your development environment, learn things by yourself, and many other topics provided in various Notes and Tips sprinkled throughout the book.

Who Should Read This Book

This book is written for anyone who likes Minecraft and isn’t afraid to learn a thing or two about programming and Minecraft.

If you expect a full guide on how to start programming, this book isn’t for you. However, if you want to get into programming or already know how to program, this book will certainly help. Creating mods for Minecraft is how I got started with programming and the same goes for many other mod creators.

It might be helpful to have some familiarity with Java before beginning this book; however, the Java basics are explained as they are used throughout the book. Therefore, it isn’t necessary to have prerequisite programming knowledge. It is helpful if you have experience playing Minecraft, though, because many terms used in the game are also used in the book.
Additionally, playing Minecraft gives you an insight into how certain aspects of the game work, which makes it easier to understand what you are creating. When you get to the last hour of this book, it is suggested that you learn Java, but by that time you should already have a reasonable understanding of it, which makes it easier to learn. After the last hour is an appendix that contains even more code, which you will likely need in your own mod.

How This Book Is Organized

Learning everything Minecraft has to offer in just 24 hours might seem like a lot to learn—and it is. This book doesn’t cover the creation of every single thing in Minecraft. Instead, it covers creating the most commonly used parts, which gives you the knowledge to make the more specific things. The following breakdown contains the details of what you will learn in each of the hours:

- In Hour 1, “Setting Up the Minecraft Development Environment,” you set up the environment in which you will start developing your Minecraft mods. In this hour, you learn about most of the programs you will be using throughout the book.
- In Hour 2, “Creating the Basics for Forge,” you start coding things. In this hour, you create the framework of your mod, which you will build on in all the other hours.
- Hour 3, “Working with Recipes and Other Small Modifications,” is the first hour where you add custom content to Minecraft in the form of crafting and smelting recipes and more.
- In Hour 4, “Making Your First Item,” you create your first item. Starting in this hour, you add the first truly unique and interesting content to your mod. Four hours until you start adding unique content is a long time, but it is required, because you can’t mod without the code or a mod class.
- In Hour 5, “Creating Multiple Items in a Smart Way,” you expand the code you made in Hour 4 to make sure your mod looks clean and works well with other mods.
- In Hour 6, “Cooking Up a Food Item,” your goal is to create an item that the player can eat and that heals the player. This is one of the many custom item types you can create.
- Hour 7, “Making Your Own Tools,” is another hour containing a custom item type. This time, you learn how to make custom pickaxes, shovels, and completely new tools.
- In Hour 8, “Creating Armor,” you learn about another custom item type—armor. You learn how to create armor and then you learn how to customize it.
- In Hour 9, “Making Your First Block,” similar to Hour 4, you learn about another important aspect of Minecraft. In this hour, you learn about the things the world is made of—blocks.
- In Hour 10, “Creating Multiple Blocks in a Smart Way,” you learn the same thing you learned in Hour 5 about items, but for blocks.
In Hour 11, “Making Your Blocks Unique,” you learn various ways to make your blocks interesting. You learn how to make half blocks, plants, and several other items.

In Hour 12, “Creating a Tile Entity,” you learn about one of the most complicated parts of Minecraft—a tile entity.

Hour 13, “Generating Ores,” is the first hour in which you generate something. In this hour, you learn how to create a class that will do all the world generation for you. It uses premade code to generate your ores.

In Hour 14, “Generating Plants,” you take what you learned in Hour 13 one step further. You now learn how to create custom code to generate things in the world.

In Hour 15, “Using MCEdit,” you learn how to use a program called MCEdit to create a structure that will be generated in your mod in Hour 16.

In Hour 16, “Generating Your Structure,” you make the structure created in the previous hour generate in the world of Minecraft.

In Hour 17, “Learning About Entities,” you learn what entities are, which kind of entities there are, and how to create the fundamentals for an entity.

In Hour 18, “Creating an Entity Model Using Techne,” you learn how to create a model for the entity you will create in Hour 19. This uses a tool called Techne, which is a piece of modeling software created specifically for Minecraft mods.

In Hour 19, “Coding a Mob,” you use the model you created in Hour 18, combine it with code from Hour 17, and then add some more code to create something similar to a zombie, but slightly different.

In Hour 20, “Creating a Throwable,” you learn about a different kind of entity that can be thrown by the player.

In Hour 21, “Editing Vanilla Minecraft,” you learn how to edit Minecraft in such a way that multiple mods can edit the same thing without it breaking the game.

In Hour 22, “Structuring Your Mod,” you learn how to clean up your mod a little. At this point, several files will look quite crowded, which can be harmful to your experience. After this hour, you should be able to continue coding in an environment that is much easier to use.

Hour 23, “Releasing Your Mod,” is a very important hour. In this hour, you learn how to use your mod outside of your development environment and how to distribute it to users.

In Hour 24, “What’s Next,” you learn how to continue working with Minecraft mod development after completing this book. The intent of this book is not just to be a guide for copying what it has to say. The intent is that you learn how to create things yourself.
Conventions Used
This book uses several conventions to make it easier for you to understand what is being explained.

Try It Yourself
The Try It Yourself activities are one of the most important conventions in this book. The Try It Yourself activities provide hands-on opportunities to actively experience and engage in the topic covered in that hour. These hands-on activities provide an opportunity to “learn by doing.” This is the best way to learn Minecraft modding.

Notes
The Note elements provide extra tidbits of information. These interesting tidbits provide supplemental content or expand the information given in the nearby text.

Tips
A Tip element provides extra information that can be useful when you are modding. This information can range from tips on how to better use a program to certain things you might want to change in the code. Tips identify handy tricks or expert advice that will help you along the way.

Cautions
The Caution element is used to warn you when an action you might take could have dire consequences later. This cautionary text warns you of potential hazards and provides advance warning of outcomes you want to avoid.

Code Listings
Code listings are another helpful element you’ll find in this book. Code listings provide sample snippets of code that relate to the topic at hand. You’ll be able to examine the sample code to get a feel for how your code should be written.

In this book, formal code listings are always surrounded by two lines to show where the code starts and where it ends. The text surrounding the listing explains where this code should be placed, if this isn’t clear from the code itself.

When a line of code is too long to fit on one line of text, it is wrapped to the next line. In that case, the continuation is preceded with a code-continuation arrow (➥).
Downloading the Code and Resources

Sometimes your mod will just not work, and no matter how much you look at it, you just can’t find the error. In this case, it might be helpful to copy the code used in this book directly into your workspace. Examining the sample code and comparing it with the code you have written might help you locate and correct the error in your code. Additionally, not everyone is good at making textures, so you might want to obtain the textures used in this book. Both of these things can be accessed by registering your book:

2. Click Register Your Product and log in or create a new account.
3. Enter this ISBN: 9780672337192. This is the ISBN of the print book and must be used to register every edition.
4. Click the Access Bonus Content link in the Registered Products section of your account page, which will take you to the page where your downloadable content is available.

Furthermore, as mentioned in the Acknowledgments, the code used to make mods for Minecraft can change quite often. To make sure this book will remain useful in the future, there are online updates and errata, which can be found at www.informit.com/title/9780672337192 on the Updates tab.
What You’ll Learn in This Hour:

- Understanding what an item is
- Creating a basic item

This hour contains the basics about items. This hour begins with a short explanation of what items in Minecraft are and uses examples from the game. In this hour, you learn how to write the most basic version of a Minecraft item. Then, you learn how to add a texture to your item. In the other hours of Part II, “Items,” you learn more about customizing items.

Understanding What an Item Is

Before creating an item in Minecraft, it’s important to know exactly what an item is. It’s also useful to know the limitations, requirements, and comparisons of items.

Items are important in Minecraft, but a basic item can’t do that much. The player can hold it, it can be dropped on the ground and used in recipes, but, except for that, the most basic kind of item doesn’t have much functionality. However, you can create incredibly interesting and useful things with a basic item. To do that, you first need to know how to make a basic item, which this hour covers. The next three hours cover some special items. Hour 6, “Cooking Up a Food Item,” explains how to create food items. Hour 7, “Making Your Own Tools,” covers exactly how to make a pickaxe and various other tools, and Hour 8, “Creating Armor,” details how to create armor. Although this book focuses on those items, the different kinds of items you can make and what you can do with them are endless.

Now that you know what an item is, you should become familiar with some special cases. For example, mushrooms might seem like items, but they are actually blocks because they can be placed.

A block is a 1 × 1 × 1 meter cube that cannot move. If you want a more detailed explanation of a block, read the start of Hour 9, “Making Your First Block.”
NOTE

**Blocks and Items**

A fun fact about Minecraft is that all blocks are actually items when you have them in your inventory.

An arrow might not seem like an item, but it is. When it's in the inventory, an arrow is an item. However, when the arrow is fired with a bow or a dispenser, it becomes an entity. As mentioned, items can be tools, food, and many other things. Items can also be used to, for example, spawn entities, which is what an entity egg does in Minecraft.

In short, an entity is something that moves around in the Minecraft world. If you want to know more about entities, read the start of Hour 17, “Learning About Entities.”

It is important to know what you can and can't do with items. Imagine that you have written a big and complicated item. Then you find out that what you want to do with it is not possible with an item and what you actually require is a block. If you are familiar with what you can and can't do with items, you can plan ahead to create what you require—and avoid frustration later!

### Creating a Basic Item

It's time to start creating your own item. To get started, you start in your mod file. Just below the two String variables and just above the init method, create a new method called preInit. The parameter for this method must be FMLPreInitializationEvent. Don’t forget to add the @EventHandler annotation above the method, or your item won’t work without showing any errors. Once you have done that, the file should look like Listing 4.1.

#### Listing 4.1  Mod File with preInit Method

```java
package com.wuppy.samsmod;

import net.minecraft.enchantment.Enchantment;
import net.minecraft.init.Blocks;
import net.minecraft.init.Items;
import net.minecraft.item.ItemStack;
import net.minecraft.util.WeightedRandomChestContent;
import net.minecraftforge.common.ChestGenHooks;
import net.minecraftforge.common.DungeonHooks;
import cpw.mods.fml.common.Mod;
import cpw.mods.fml.common.Mod.EventHandler;
import cpw.mods.fml.common.event.FMLInitializationEvent;
import cpw.mods.fml.common.event.FMLPreInitializationEvent;
import cpw.mods.fml.common.registry.GameRegistry;

@Mod(modid = SamsMod.MODID, version = SamsMod.VERSION)
public class SamsMod
```
```java
{
    public static final String MODID = "wuppy29_samsmod";
    public static final String VERSION = "1.0";

    @EventHandler
    public void preInit(FMLPreInitializationEvent event)
    {
    }

    @EventHandler
    public void init(FMLInitializationEvent event)
    {
        //Recipes
        GameRegistry.addRecipe(new ItemStack(Items.apple),
            "XXX",
            "XXX",
            "XXX",
            'X', Blocks.leaves);
        GameRegistry.addRecipe(new ItemStack(Items.arrow),
            "YZ",
            "X",
            'X', Items.flint, 'Y', Items.stick, 'Z', Blocks.leaves);
        GameRegistry.addShapelessRecipe(new ItemStack(Items.dye, 2, 1),
            Items.redstone, new ItemStack(Items.dye, 1, 1));
        GameRegistry.addSmelting(Blocks.stone, new ItemStack(
            Blocks.stonebrick), 0.1F);

        ItemStack enchantedSwordItemStack = new ItemStack(
            Items.stone_sword);
        enchantedSwordItemStack.addEnchantment(Enchantment.sharpness, 1);

        GameRegistry.addShapelessRecipe(enchantedSwordItemStack,
            Items.flint, Items.stone_sword);

        //Dungeon changes
        DungeonHooks.removeDungeonMob("Spider");
        DungeonHooks.addDungeonMob("Creeper", 100);
        ChestGenHooks.removeItem(ChestGenHooks.DUNGEON_CHEST, new
            ItemStack(Items.saddle));
        ChestGenHooks.addItem(ChestGenHooks.DUNGEON_CHEST, new
            WeightedRandomChestContent(new ItemStack(Blocks.cobblestone), 25, 50, 10));
    }
}
```
There are three initialization phases with Forge: PreInit, Init, and PostInit. PreInit is where you register all the basics for your mod. Init is where you place mod-specific code that is not directly related to registering. Recipes should also go in Init. PostInit is where you add code that interacts with other mods.

This forces modders to do all those things at the same time, making it easier for other modders who create mod bridges or other application programming interfaces (APIs) to use a later initialization phase to find out which blocks are added or which mods are installed.

Because of these rules, you need to create the preInit method for the registry of items.

To create an item, you first must create a new variable. This is a variable that will likely be used in several files. Therefore, it must be public and static. It must also be placed outside of any methods in the class, just below the two String variables used in the @Mod line. Because you are going to create an item, the variable should be an item. The name for the variable can be anything you want, but best practice is to give it the name of the item.

In this book, the basic item you create is a key for the dungeon created over the course of this book. The variable line should look like this:

```java
public static Item key;
```

If you are getting an error under item, import the Item class from the Minecraft package. Be certain to get the one from the Minecraft package, or you will get a lot of errors in the following code.

In the preInit method, two lines of code are required to make most items. One line is to let Java know what the key object actually is, and the other line is to tell Forge to register it as an item. The following code snippet shows the code for the key. Note that the code contains an error, which you fix shortly.

```java
key = new ItemKey();
GameRegistry.registerItem(key, "Key");
```

The top line sets key to a class, which you create in a few minutes. It can also be called anything you want, but the general name for it is Item followed by the item name. Having an error under the ItemKey, or whatever your name for it is, is normal. It’s because ItemKey is a class that isn’t created yet. From now on, this line will be called the item init line.

The second line of code registers an item to Forge using the GameRegistry file, which makes sure Minecraft knows the item exists and can be used. The method used this time is registerItem, which takes two parameters. The first parameter is an Item object and the second parameter is a name for it. The name can also be anything you want, but the best way to name it is the name of the item. It’s important to make sure that there aren’t any duplicate names in the item registry. If there is another item called Key, it might not work. However, Forge
automatically adds the modid in front of the name you use to register here. This ensures that multiple mods can have an item called Key without problems. If item registry line is mentioned in this book, it references this line.

The following code shows the preInit method and the new variable after adding the item. All the other code in the file is the same, except for a few new imports:

```java
public static Item key;

@EventHandler
public void preInit(FMLPreInitializationEvent event)
{
    key = new ItemKey();

    GameRegistry.registerItem(key, "Key");
}
```

Creating an Item File

To fix the error under the class name in the item init line, you create an item file with the name used in the init line. To create a file like that, hover your mouse over the error and click Create Class ‘...’. This causes the New Java Class window to open. In this window, click Finish. You will now have a file that should look like Listing 4.2.

**LISTING 4.2 Empty Item File**

```java
package com.wuppy.samsmod;

import net.minecraft.item.Item;

public class ItemKey extends Item
{

}
```

To make a fully functional item, you must add a constructor to this file. A constructor is a method without a return type and the method name has to be exactly the same as the class file. A constructor is a special method in Java that is used to create an object for a class. This method will run whenever the Item object is created. An empty constructor for ItemKey will look as follows:

```java
public ItemKey()
{
}
```
There must be two lines of code in this constructor for Minecraft to recognize it as an item and actually use it. The first line makes sure the item has some sort of name:

```java
setUnlocalizedName(SamsMod.MODID + "_" + "key");
```

This sets the unlocalized name of the item to something. An item requires an unlocalized name, or it will not show up anywhere. Without a name, it isn’t an item. It’s important to make sure that the unlocalized name is unique for every item in every mod. To make sure this happens, use the modid of your mod followed by an underscore and then finally the name of the item variable in your mod file. This ensures that there can’t possibly be duplicates, because a modid has to be different for every mod installed, and you can’t have two variables with the same name.

When you have an item, it doesn’t automatically show up in the Creative menu. This is quite useful because not everything should be on one of the Creative tabs. However, in the case of most items, you want it to be visible somewhere. To do that, you will need the following code:

```java
setCreativeTab(CreativeTabs.tabMisc);
```

This method requires a single parameter, which is a CreativeTab. To get the correct CreativeTab, you might want to look through the file in which all of the tabs are listed. ItemKey will be added to the Miscellaneous tab.

If you start the game right now, you will see there is an item in the Creative menu with a weird black and purple texture. The name is also strange.

### Fixing the Name

To fix the names of an item, you have to add a localization for it. By default, items in Minecraft have the name `item.unlocalizedname.name`. This is not a desirable name and should almost always be changed. You do this by adding localizations. Minecraft has different localizations for every language. If you play Minecraft with a localization different than en_US, which is the default American English, it first tries to look for localizations under that language. If it doesn’t find them, it tries to get them from the en_US ones. If it can’t find them there either, the default name is used.

Fixing the name of the item is going to be quite a bit of work. First, navigate to the forge folder created in Hour 1, “Setting Up the Minecraft Development Environment.” Then enter `src/main/resources`. Figure 4.1 shows the correct folder structure.
In this folder, you need to create a few custom folders. The first folder is called assets. Within the assets folder, you need to make another folder with the name of your modid, which has to be the same as the MODID variable in your mod file. The final folder you need to create is called lang. If you now go into your Eclipse, you should see a white package with the names you just used under the resources folder. In the case of the mod written in this book, the folder structure is assets/wuppy29_samsmod/lang. It should look similar to Figure 4.2.
HOUR 4: Making Your First Item

FIGURE 4.2
The folder structure for lang.

CAUTION
The Folder Structure
Be certain that the folder structure follows the exact same structure and has the modid listed correctly. If there is even a minor mistake in one of the folder names, the names and textures will not work.

Once you are certain the folder structure is correct, you need to add a new file. The file should be called en_US.lang.

CAUTION
File Extensions
After creating it, double-check the extension of the file to make sure it’s .lang and not something like .lang.txt. If it’s .lang.txt, your name localizations will not work.
Open the file using a text editor of your choice. If you look at the name of the item in the game, it’s `item.UnlocalizedName.name`. The `.lang` file created is used to add a localized name for the item. To create a localized name for the key, you need the following line in the `.lang` file:

```
item.wuppy29_samsmod_key.name=Key
```

The first part is the name of the item when displayed without localizations. After that, add an equal sign and then add the name on the other side. Ensure there are no spaces between the unlocalized name, the equal sign, and the localized name. If there are, it will either not work or look strange. You can have spaces in the unlocalized and localized names themselves in this file.

**NOTE**

**Translations**

If you want your mod to have multiple localizations, you must add an extra `.lang` file. The name of this file is the code of the language, for example `en` or `nl`, followed by the code for the country, for example `US` or `NL`. All the language filenames are available on the Minecraft wiki at [http://minecraft.gamepedia.com/Language](http://minecraft.gamepedia.com/Language).

Now when you start the game and take a look at the item you created, the name will be fixed.

**Adding a Texture**

To fix the strange purple texture on the item, you must do a few things. First, you must actually create the new item texture. It is outside the scope of this book to cover exactly how to create the best textures, but this section does cover the basics. You need the image editor software of your choice, such as GIMP, Photoshop, or Paint. In one of these programs, create a 16×16 pixel image of the item you want. It doesn’t really matter what you put in there. However, what does matter is the file format and filename when you save it. The file format has to be `.png`. The filename has to be something you can easily use in code. In this book, it will always be the same name as the variable unless specified differently. The item texture used for the key is called `key.png` and looks like Figure 4.3.

**FIGURE 4.3**
The key item texture.
Now that you have a texture for the item, you need to place it somewhere where Forge can find it for you. This place is quite similar to the location of the .lang files. In the forge folder, open the following folders: src\main\resources\assets\wuppy29_samsmod, replacing the wuppy29_samsmod part with the modid of your mod. The path should look similar to Figure 4.4.

![Figure 4.4](image)

Modid folder structure.

In this folder should be the lang folder you created earlier. This time, create a folder called textures. In that folder, create a final folder called items. Place the texture file you created earlier in there. In the case of the ItemKey, the texture is Figure 4.3. The folder structure for it should look similar to Figure 4.5.
Now the texture is ready to be used. Now, you just need to tell Forge to get that texture for the item. To do this, you add a new line of code to the constructor. It should look like the following:

```java
setTextureName(SamsMod.MODID + "":"key");
```

The texture name has to be first the modid, which should be stored in a variable in your mod file. Then you need a colon and the unlocalized name of the item.

Right now, the constructor looks like this:

```java
public ItemKey()
{
    setUnlocalizedName(SamsMod.MODID + "":"key");
    setTextureName(SamsMod.MODID + "":"key");
    setCreativeTab(CreativeTabs.tabMisc);
}
```

Once you have added this line of code, your item will work. Figure 4.6 shows the key in the game.
FIGURE 4.6
The ItemKey in Minecraft.

This file isn’t coded very well because the same String is used twice. If you want to change the name of the item, you must change the name in both locations. To make this a little bit easier, create a private String that contains key. Then put the String variables in these methods.

## TRY IT YOURSELF

### Rename Your Item

Change the name of your item and change everything accordingly.

1. Change the name String.
2. Change the localization and textures.

After completing the Try It Yourself activity and all other tasks in this hour, the whole item file should look like Listing 4.3.

### Listing 4.3  Finished ItemKey File

```java
package com.wuppy.samsmod;

import net.minecraft.creativetab.CreativeTabs;
import net.minecraft.item.Item;

public class ItemKey extends Item {
    private String name = "key";

    public ItemKey() {
```
Now when you launch the game, you should see the item with the correct name and the correct texture.

**Summary**

In this hour, you learned how to create an item. At this point, you should understand what items in Minecraft are and what you can do with them. You should also be able to add a basic item and give it a texture.

**Q&A**

**Q.** Is there a maximum number of items you can add to Minecraft?

**A.** Yes, but it’s close to impossible to reach it. However, you should always try to use as few different items as possible. Hour 5, “Creating Multiple Items in a Smart Way,” covers how to do that.

**Q.** How do I add a second item to my mod file?

**A.** Simply rewrite the code for the first item, but make sure the variables and unlocalized names are all different.

**Q.** Can you have multiple items using the same file?

**A.** Yes you can. Just make sure the unlocalized name is different for every item.

**Q.** Can you add the same item to multiple creative tabs?

**A.** No, this is not possible.

**Workshop**

You now know the basics of items. In the next few hours, you will learn more about them. Therefore, it’s important to fully understand the basics. To do that, you might want to work through this workshop.
Quiz

1. What would the best unlocalized name be for an item variable called `alarmClock`?

2. What is wrong with the following code?

```java
public static Item key;

@EventHandler
public void preInit(FMLPreInitializationEvent event)
{
    GameRegistry.registerItem(key, "Key");
    key = new ItemKey();
}
```

Answers

1. The best unlocalized name is `alarmClock`, because it’s generally smartest to use the variable name.

2. The item is registered before initializing it to something. This will crash your mod. Always make sure the `registerItem` method is below the Item init line.

Exercises

Improve the item even more by passing the unlocalized name of the item through the constructor. Some things you will have to do for that:

1. Add a `String` variable to the constructor.

2. Add the unlocalized name in the mod file where you use the constructor, which should be the item init line.

3. Remove the `String` variable in the `ItemKey` class because you don’t need it anymore.

The code in the mod file:

```java
key = new ItemKey("key");
```

The constructor of `ItemKey`:

```java
public ItemKey(String name)
{
    setUnlocalizedName(SamsMod.MODID + "_" + name);
    setTextureName(SamsMod.MODID + ":" + name);
    setCreativeTab(CreativeTabs.tabMisc);
}
```
This page intentionally left blank
Symbols

{} (braces), 27
// comment notation, 27
/* */ comment notation, 27
2x2 grid space, 42-43
3x3 grid grid space, 42

A

addArmorMaterial method, 111-112
addBlockDrops method, 302
addDungeonMob method, 48

adding. See also creating
blocks to Creative menu, 126
items, 54-57
to Creative menu, 58
finished ItemKey file, 64-65
item files, 57-58
multiple items, 67-70
names, 58-61
preinit method, 54-57
textures, 61-64
variables, 56
localizations to item names, 58-61
metadata to items, 72
constructors, 72-73
getIconFromDamage method, 76-77
image registration, 75-76
item names, 73-74
ItemKey with metadata, 77-78
mod file with metadata keys, 79-80
textures, 75
mods to websites, 328-329
textures to blocks, 139-143
addItem method, 49
addRecipe method, 41-42, 203
addShapelessRecipe method, 40-44
addSmelting method, 45
addSpawn method, 274-275
addSpawnBiome method, 383
addToolMaterial method, 95-96
Adfly, 16
AdvancedModelLoader class, 263
AI (artificial intelligence)
  adding to mobs, 276-277
custom AI, 384
  adding, 390
  attacking by night, 386-390
  fleeing when hurt, 384-386
alternative shapes, 42-44
annotations, 23-24
  @Override, 138
  @SidedProxy, 246, 313
  @SideOnly, 139-140
  @SubscribeEvent, 298
applyEntityAttributes method, 278
armor, 111
  ArmorMaterial, 111-112
clothes, 364-365
creating
  ItemSamArmor class, 113-118
  mod file code, 112-113
customizing, 119
effects, 365-367
egg-throwing abilities, 303-307
localizations, 113
textures
  armor piece textures, 114, 116-118
  in game, 114
ArmorMaterial, 111-112
arrays, IIcon, 139-140
artificial intelligence. See AI (artificial intelligence)
assets folder, 58-59
attack villagers, 277
attacking by night, 386-390
audio
  playSound method, 391
  step sounds, 126-127
autocomplete in Eclipse, 184-185
automatic update checking, 353-356
axe
  code listing, 99
  in game, 101
textures, 101
B
base values, 278
behavior, changing
  entity behavior, 298-302
  mob behavior, 363-364
berry item
  ItemBerry class, 84-86
  mod file code, 84
BiomeGenBase method, 383
BiomeGenExample class, 383-384
biomes, custom, 383-384
blast resistance, 126
Blender, 263
Block class, 129
BlockCustomRenderWoodExample class, 343-346
BlockRotateExample class, 342-343
blocks
  adding to Creative menu, 126
  BlockSamStone class, 124-126
creating, 124-126
customizing, 126-129
in game, 131
half blocks, 148-151
textures, 129-131, 141-142
compared to items, 53-54
custom block rendering
  BlockCustomRenderWoodExample class, 343-346
  RenderCustomWoodExample class, 346-350
customizing, 126-129
blast resistance, 126
drops
  editing, 302-303
  randomizing, 299
explained, 123-124
half blocks, 148-151
initializing, 124
metadata, 135
  block bounds based on metadata, 152
  block registry, 136
  in Creative menu, 135-136
  ItemBlock class, 136-138
  ItemSamStone class, 137-139
textures, 139-143
mod file code, 124
names, 126
plants, 152
creating, 152-155
crops, 155-157
customizing, 165-171
in game, 170-171
natural growing, 158-159
random plants, 157-158
seeds, 159-165
registering, 124, 136
rotating, 342-343
stone slabs, 123
textures, 129-131
adding, 139-143
sided textures, 145-147
tile entity blocks, 176-180
TNT block, 357-359
unbreakable blocks, 127

Blocks class, 129

BlockSamPlant class, 167-170

BlockSamPlant constructor, 153

BlockSamStone class

creating, 124-126
customizing, 126-129
in game, 131
half blocks, 148-151
textures, 129-131
Sam’s Wall texture, 141-142
sided textures, 145-147

BlockSamTE class

creating, 176-180
right-clicking in, 185-188

BlockSamTE constructor, 177
BlockTNTExample class, 357-359
body parts, rendering, 268-270
boots. See armor
bounds (block), 148-151
braces (), 27
Braces options (code formatting), 213

Cadenhead, Rogers, 338
canPlaceBlockOn method, 165
changing
  entity behavior, 298-302
  mob behavior, 363-364
  proxies, 313
channels (IRC), joining, 335
chat. See IRC (Internet Relay Chat)
checking for updates, 353-356
checkUpdate method, 355-356
chest items, 49
  adding, 49
  removing, 49

ChesGenHooks class

addItem method, 49
removeItem method, 49

ChunkProviderExample class, 369-382

ClientProxySam, 248, 272, 291
CommonProxySam, 247
Config, 341-342
creating, 31-33

DungeonHooks, 48
addDungeonMob method, 48
removeDungeonMob method, 48

Entity, 245, 250

How can we make this index more useful? Email us at indexes@samspublishing.com
EntityAI, 276
EntityAIAttackOnCollideNight, 386-390
EntityAFleeOnHurt, 384-386
EntityCaveSpiderCustom, 363-364
EntityExampleTNTPrimed, 359-361
EntityFX, 245
EntityMob, 275
EntitySamMob, 249-250
AI, 276-277
base values, 278
drops, 278-280
EntitySamMob extending EntityMob, 275
finished EntitySamMob, 279-280
in game, 280
EntitySamThrowable, 284-287
EntityThrowable, 284
EventHandler, 195-202
ExampleMod, 24-25
ExampleTab, 356
fields, 25-26
final fields, 25-26
public fields, 25
static fields, 25
ForgeDirection, 147
importing, 23
IISimpleBlockRenderingHandler, 149
ItemBerry, 84-86
constructor, 85
empty ItemBerry file, 84
potion effect, 89-91
textures, 85-86
ItemBlock, 136-138
ItemSamArmor
armor piece textures, 114, 116-118
creating, 113-118
in game, 114
localizations, 113
mod file code, 112-113
ItemSamAxe
code listing, 99
in game, 101
textures, 101
ItemSamHoe
code listing, 99
in game, 101
textures, 101
ItemSamPickaxe
code listing, 104-105
creating, 104-105
func_150893_a method, 105-106
func_150897_b method, 105-106
in game, 107
hitEntity method, 106-107
localization, 107
mod file code, 103
texture, 107
ItemSamPickaxe, 97-98, 101
ItemSamPlant, 160
ItemSamSeed, 159-165
ItemSamShovel
code listing, 100
in game, 101
textures, 101
ItemSamStone, 137-139
ItemSamSword
code listing, 100
in game, 101
textures, 101
ItemSamThrowable, 287-290
IWorldGenerator, 196-197
ModBlocks, 319-320
ModelSamMob
finished ModelSamMob, 266-268
setRotationAngles method, 268-270
OreDictionary, 202-204
recipes, 203-204
registerOre method, 202-203
public classes, 24-25
Render, 270-271
RenderCustomWoodExample, 346-350
RenderExampleItem, 391-393
RenderExampleTNTPrimed, 361-363
RenderingRegistry, 272
RenderLiving, 271
RenderSamMob, 270-271
SamDungeon, 235
SamEventHandler, 195-202
addOreSpawn method, 198-200
finished SamEventHandler class, 200-202, 237-239
generation methods, 197-198
initial SamEventHandler class, 196
IWorldGenerator class, 196-197
with WorldGenSamPlant code, 215-219
SamsMod
cleaning up, 320-322
code listing, 314-319
ModBlocks, 319-320
String, 24
TileEntityMobSpawner, 241-243
TileEntitySam, 180-183, 190-192
ToolMaterial, 95-96
UpdateChecker, 353-356
WorldGen, 207-215
   checking and placing, 209
finished
   WorldGenSamPlant, 210
initial WorldGenSamPlant class, 207
random start metadata, 210
randomizing location, 208-209
WorldGenSamPlant extending WorldGenerator, 208
WorldProviderExample, 367-369
cleaning
   mod classes, 314-322
   packages, 312-313
client proxy, 272
ClientProxySam class, 248, 272, 291
clothes, 364-365
code formatting
   Formatter window, 211-212
   formatting options, 212-215
   New Profile window, 211
   Preferences window, 211
commands
   gradlew eclipse, 17
   gradlew setupDecompWorkspace, 17
comments, 27, 213, 337
CommonProxySam class, 247
Computer Properties window, 10
Config class, 341-342
configs, 341-342
configuring. See setting up constructors, 57
   BlockSamPlant, 153
   BlockSamStone, 125
   BlockSamTE, 177
   ItemBerry, 85
   ItemSamArmor, 113
   ItemSamPickaxe, 97
   ItemSamPlant, 160
   metadata, 72-73
Control Statement options (code formatting), 213
copying code, 338
crafting grid
   2x2 grid space, 42-43
   3x3 space, 42
crafting recipes, 40-41
   explained, 39
   GameRegistry, 41-42
   shape of recipes
      alternative shapes, 42-44
      explained, 42
createNewTileEntity method, 178
creating. See also adding
   armor
      addArmorMaterial method, 111-112
      ArmorMaterial, 111-112
      ItemSamArmor class, 113-118
      mod file code, 112-113
      blocks, 124
         BlockSamStone class, 124-126
         custom blocks, 126-129
         half blocks, 148-151
         sided textures, 145-147
         textures, 129-131, 145-147
      chest items, 49
      classes, 31-33
      cubes, 257-258
      drops, 278-280
      entities. See entities
         entity models, 253
      adding/removing cubes, 257-258
creating
downloading and installing Techne, 253-254
editing cube properties, 256-257
exporting models, 259-263
model characteristics, 253-254
Techne model creation screen, 255
items, 54-57
finished ItemKey file, 64-65, 67
food items, 84-89
item files, 57-58
ItemSamGeneric example, 69-71
names, 58-61
preInit method, 54-57
textures, 61-64
variables, 56
mcmod.info files, 326-327
mob spawners, 241-243
mobs. See mobs
mod files, 34-35
packages, 27-30
plants, 152-155
crops, 155-157
customization, 165-171
natural growing, 158-159
random plants, 157-158
seeds, 159-165
proxies, 246-247
ClientProxySam class, 248
CommonProxySam class, 247
recipes
crafting recipes, 40-44
shapeless recipes, 44
smelting recipes, 45
SamEventHandler class, 195-202
addOreSpawn method, 198-200
finished SamEventHandler class, 200-202
generation methods, 197-198
initial SamEventHandler class, 196
IWorldGenerator class, 196-197
tile entities, 180-183
tile entity blocks, 176-180
ToolMaterial, 95-96
tools
axe, 99
hoe, 99
paxel, 103-107
pickaxe, 96-98
shovel, 100
swords, 100
variables, 56
WorldGen class, 207-215
checking and placing, 209
finished
WorldGenSamPlant, 210
initial WorldGenSamPlant class, 207
random start metadata, 210
randomizing location, 208-209
WorldGenSamPlant extending WorldGenerator, 208
Creative menu
adding blocks to, 126
adding items to, 58
block registry, 136
metadata in, 135-136
CreativeTabs, customizing, 356
crops, creating, 155-157
cubes
adding/removing, 257-258
editing cube properties, 256-257
customizing
AI, 384
adding custom AI, 390
attacking by night, 386-390
fleeing when hurt, 384-386
armor, 119
clothes, 364-365
effects, 365-367
biomes, 383-384
blocks, 126-129
blast resistance, 126
hardness, 126
harvest level, 127-128
localizations, 128-129
step sounds, 126-127
textures, 129-131
CreativeTabs, 356
dimensions, 367
ChunkProviderExample class, 369-382
WorldProviderExample class, 367-369
ItemStacks, 46
plants, 165-171
Render class, 391-393
entities  401

ToolMaterial, 95-96
tools (ItemSamPaxel class),
103-107
code listing, 104-105
creating, 104-105
func_150893_a method,
105-106
func_150897_b method,
105-106
in game, 107
hitEntity method, 106-107
localization, 107
mod file code, 103
texture, 107
tree generation, 350-352

displaying line numbers, 86-89

downloading
  MCEdit, 223-224
  Schematic Converter, 231
  Techne, 253-254

dropBlockAsItemWithChance
method, 166

drops
  adding, 278-280
  editing, 302-303
  randomizing, 299

DungeonHooks class, 48
  addDungeonMob method, 48
  removeDungeonMob method, 48

dungeons
  mobs. See mobs
  SamDungeon class, 235
  WorldGenSamDungeon class,
  236

dust item, 70-72

edalegories, OreDictionary class,
202-204
recipes, 203-204
registerOre method, 202-203
dimensions, custom, 367
  ChunkProviderExample class,
  369-382
  WorldProviderExample class,
  367-369
eclipse.org, 16
editCounter method, 188
editing
  block drops, 302-303
  cube properties, 256-257
  effects (armor), 365-367
egg-throwing abilities, 303-307
empty item file, 57
en_US.lang file, 60
enchantedSwordItemStack, 41-42
etnies
  AI
    adding to mobs, 276-277
    attacking by night,
    386-390
    fleeing when hurt, 384-386
    changing behavior of, 298-302
    compared to blocks, 123-124
    creating basic entities,
    248-250
    EntitySamMob class,
    249-250
    registerModEntity method,
    248-249
Entity class, 245, 250
entity models, creating with
Techne, 253
  adding/removing cubes,
  257-258
  downloading and installing
  Techne, 253-254
  editing cube properties,
  256-257
  exporting models, 259-263
  model characteristics,
  253-254
  Techne model creation
  screen, 255

How can we make this index more useful? Email us at indexes@samspublishing.com
entities

EntitySamMob class, 249-250
EntitySamThrowable class, 284-287
explained, 54, 245-246
mobs. See mobs
proxies
ClientProxySam class, 248
CommonProxySam class, 247
creating, 246-247
explained, 246
removing, 297-298
throwable entities, 283-293
in game, 291-293
ItemSamThrow class, 287-290
registering, 283
rendering, 290-291
tile entities
adding to blocks, 176-180
creating, 180-183
explained, 175-176
server synchronization, 183-192
TileEntitySam class, 180-183, 190-192
TNT entity, 359-361
Entity class, 245, 250
entity models, creating with
Techne, 253
adding/removing cubes, 257-258
downloading and installing
Techne, 253-254
editing cube properties, 256-257
exporting models, 259-263
model characteristics, 253-254
Techne model creation
screen, 255
EntityAI classes, 276
EntityAIAttackOnCollideNight
class, 386-390
EntityAFleeOnHurt class, 384-386
EntityCaveSpiderCustom class,
363-364
EntityExampleTNTPrimed class,
359-361
EntityFX class, 245
EntityList file, 48
EntityMob class, 275
EntitySamMob class, 249-250
AI, 276-277
base values, 278
drops, 278-280
EntitySamMob extending
EntityMob, 275
finished EntitySamMob,
279-280
in game, 280
EntitySamThrowable class,
284-287
EntityThrowable class, 284
Environment Variables window, 13
@EventHandler, 26
EventHandler class
creating, 195-202
addOreSpawn method, 198-200
finished SamEventHandler
class, 200-202
generation methods, 197-198
initial SamEventHandler
class, 196
IWorldGenerator class, 196-197
explained, 195
events
egg-throwing example,
303-307
how they work, 303
initialization events, 26
LivingDropEvents, 298-302
PlayerInteractEvent, 303-307
registering, 303
examining recipes in mod files,
49-51
ExampleMod
code listing, 18-19, 21-22
ExampleMod class, 24-25
fields, 25-26
methods, 26-27
imports, 22-23
@Mod line, 23-24
packages, 22-23
ExampleTab class, 356
explosions, 286
Export button (MCEdit), 229
exporting
models from Techne, 259-263
mods with Forge, 325-327
structures from MCEdit,
228-229
extending
Entity class, 250
EntityMob class, 275
EntityThrowable class, 284
RenderLiving class, 271
extensions (file), 60
F

fields, 25-26
  final fields, 25-26
  public fields, 25
  static fields, 25
files
  eclipse.exe, 16
  en_US.lang, 60
  EntityList, 48
  file extensions, 60
  GameRegistry, 41-42
  ItemKey, 64-65, 67-68
  ItemSamGeneric, 69-71
  level.dat, 227
  mcmod.info, creating, 326-327
  mod files
    clean mod file (SamsMod), 40
    creating, 34-35
    examining recipe results in, 49-51
    mod file with metadata keys, 79-80
    mod file with preInit method, 54-57
  .obj files, 263
final fields, 25-26
final keyword, 25-26
finding open source mods, 336-337
fleeing when hurt, 384-386
folders
  assets, 58-59
  folder structure, 60

G

GameRegistry, 41-42
  generate method, 197-198
  generateEnd method, 197-198
  generateNether method, 197-198, 200
  generateSurface method, 197-198
  generating
    ores
      EventHandler class, 195
      nether ore generation, 200
      OreDictionary class, 202-204
      results in game, 204
      SamEventHandler class, 195-202
    plants, 207
      results in game, 219
      SamEventHandler class, 215-219
    structures, 231-239
    trees, 350-352
getArmorModel method, 364-365
getArmorTexture method, 117-118
getBlockData method, 182
getCurrentArmor method, 305
descriptionPacket method, 188-190
getDropItem method, 278
getDrops method, 166-167
getIcon method, 140, 146-147
generateFromDamage method, 76-77

How can we make this index more useful? Email us at indexes@samspublishing.com
getItemDropped method, 165-166
getItemMetadata method, 137
getPlant method, 161-162
getPlantMetadata method, 161-162
getPlantType method, 161-162
getQuantity method, 165-166
getRenderType method, 155
getSubBlocks method, 135-136
getSubItems method, 76-77
getUnlocalizedName method, 74, 137-138
GitHub repository, 328-329
GlScale, 257
gradlew eclipse command, 17
gradlew setupDecompWorkspace command, 17
greyKey item, 68

IDEA (IntelliJ), 8
IDEs (Integrated Development Environments), 8
IIcon, 75, 139-140, 153-154
images, registering, 75-76
import keyword, 23
importing
  classes, 23
  explained, 22-23
  worlds, 227
ingot item, 70-72
Init event, 26
init method, 26
initialization events, 26
initializing blocks, 124
installing
  IRC (Internet Relay Chat), 333-335
  MCEdit, 225
  Schematic Converter, 231-232
  Techne, 253-254
Integrated Development Environments (IDEs), 8
IntelliJ IDEA, 8
Internet Relay Chat. See IRC (Internet Relay Chat)
IRC (Internet Relay Chat), 333
  installing, 333-335
  joining channels, 335
  settings, 334-335
ISimpleBlockRenderingHandler, 149
item files, creating, 57-58

ItemBerry class, 84-86
  constructor, 85
  empty ItemBerry file, 84
  potion effect, 89-91
  textures, 85-86
ItemBlock class, 136-138

ItemKey
  adding to Creative menu, 58
  creating, 54-57
    finished ItemKey file, 64-65, 67
    item files, 57-58
    names, 58-61
    preInit method, 54-57
    textures, 61-64
    variables, 56
  ItemKey file for multiple items, 68
metadata
  constructors, 72-73
  item name, 73-74
  ItemKey with metadata, 77-78
  mod file with metadata keys, 79-80
  textures, 75
  multiple items, adding to same file, 67-70
  name
    localizations, 58-61
    unlocalized name, 58
items
  adding to Creative menu, 58
  armor, 111
    armor piece textures, 114, 116-118
    ArmorMaterial, 111-112

half blocks, 148-151
hardness of blocks, 126
harvest level, 127-128
helmet. See armor
HexChat
  Edit EsperNet window, 334
    installing, 333
    settings, 334-335
hitEntity method, 106-107
hoe
  code listing, 99
    in game, 101
    textures, 101
hurt, fleeing when, 384-386
clothes, 364-365
creating, 112-118
customizing, 119
effects, 365-367
in game, 114
localizations, 113
cmpared to blocks, 53-54
compared to entities, 54
creating, 54-57
finished ItemKey file, 64-65, 67
item files, 57-58
ItemSamGeneric example, 69-71
names, 58-61
preInit method, 54-57
textures, 61-64
variables, 56
explained, 53-54
food items, 84-89
creating, 84-89
explained, 83-84
foodLevel, 83-84
foodSaturationLevel, 83-84
ItemBerry class, 84-86
mod file code, 84
potion effect, 89-91
ItemKey, 54-57
adding to Creative menu, 58
finished ItemKey file, 64-65, 67
item files, 57-58
ItemKey file for multiple items, 68
localizations, 58-61
metadata. See metadata
multiple items, adding to same file, 67-70
names, 58-61
preInit method, 54-57
textures, 61-64
unlocalized name, 58
variables, 56
metadata, adding, 72
constructors, 72-73
getIconFromDamage
method, 76-77
image registration, 75-76
item names, 73-74
ItemKey with metadata, 77-78
mod file with metadata
keys, 79-80
textures, 75
models, 391-393
multiple items, adding to same file, 67-70
names
localizations, 58-61
metadata, 73-74
unlocalized names, 58
samdust, 70-72
samingot, 70-72
tools. See tools
ItemSamArmor class
armor piece textures, 114,
116-118
creating, 113-118
in game, 114
localizations, 113
mod file code, 112-113
ItemSamAxe class
code listing, 99
in game, 101
textures, 101
ItemSamGeneric class, 69-71
ItemSamHoe class
code listing, 99
in game, 101
textures, 101
ItemSamPaxel class
code listing, 105
creating, 104-105
func_150893_a method,
105-106
func_150897_b method,
105-106
in game, 107
hitEntity method, 106-107
localization, 107
mod file code, 103
texture, 107
ItemSamPickaxe class, 97-98,
101
ItemSamPlant class, 160
ItemSamPlant constructor, 160
ItemSamSeed class, 158-165
ItemSamShovel class
code listing, 100
in game, 101
textures, 101
ItemSamStone class, 137-139
ItemSamSword class
code listing, 100
in game, 101
textures, 101
ItemSamThrow class, 287-290
ItemStack parameter
  custom ItemStacks, 46
  explained, 41-42
IWorldGenerator class, 196-197

J
.jar files, renaming, 328
Java, 7, 338
Java Development Kit. See JDK
  (Java Development Kit)
Java Runtime Environment (JRE), 7-8
JDK (Java Development Kit)
  overview, 7-8
  setting up, 9-15
    Linux, 15
    Mac, 15
    Windows, 9-15
joining IRC channels, 335
JRE (Java Runtime Environment), 7-8

K
key items. See ItemKey
keywords
  final, 25-26
  import, 23
  public, 24-25
  static, 25
KiwiIRC, 334
knockback sword, creating, 47
knockbackItemStack, 47

L
lang folder, 58-59
leggings. See armor
level.dat file, 227
libraries, forgeSrc, 28-29
line numbers, displaying, 86-89
Line Wrapping options (code formatting), 213
Linux, JDK (Java Development Kit)
  setup, 15
LivingDropEvents, 298-302
loadBlocks method, 320
loadConfig method, 342
localizations, 107
  adding to item names, 58-61
  armor, 113
  blocks, 128-129
  ItemSamPickaxe class, 125
  ItemSamStone class, 139
  randomizing location, 208-209
locations, spawning, 241

M
Mac computers, JDK (Java Development Kit) setup, 15
making mods public, 328-329
manual mob spawning, 240-241
material
  ArmorMaterial, 111-112
  ToolMaterial, 95-96
Maya, 263
MCEdit
  advantages of, 223
  creating structures, 225-227
downloading, 223-224
exporting structures, 225-226, 228-229
importing worlds, 227
installing, 225
mcmMod.info files, creating, 326-327
MCP (Mod Coder Pack), 8
metadata
  adding to blocks, 135
  block bounds based on metadata, 152
  block registry, 136
  in Creative menu, 135-136
  ItemBlock class, 136-138
  ItemSamStone class, 137-139
  textures, 139-143
  adding to items, 72
  constructors, 72-73
  getIconFromDamage method, 76-77
  image registration, 75-76
  item names, 73-74
  ItemKey with metadata, 77-78
  mod file with metadata keys, 79-80
  textures, 75
methods. See also specific methods (for example addBlockDrops method)
  constructors, 57
  explained, 26-27
  names, 26-27
  return types, 26
Mibbit, 334
mIRC, 334
mirror texture, 257
mob spawners, creating, 241-243
mobs, 48, 257-258
adding, 48, 240
  manual spawning of entities, 240-241
mob spawners, creating, 241-243
AI, 276-277
changing behavior of, 363-364
custom AI, 384
  attacking by night, 386-390
  fleeing when hurt, 384-386
entity models, creating with
  Techne, 253
  adding/removing cubes, 257-258
downloading and installing
  Techne, 253-254
  editing cube properties, 256-257
exporting models, 259-263
  model characteristics, 253-254
  Techne model creation screen, 255
models (item), 391-393
ModelSamMob class
  finished ModelSamMob, 266-268
  setRotationAngles method, 268-270
modid variable, 24
modlist.mcf.li, 329, 337
moving body parts, 268-270
multiple items, adding to same file, 67-70
multiple localizations, adding to
  item names, 61

N

Named Binary Tag (NBT), 189
names
  block names, 126
  item names, 58-61
  localizations, 58-61
  metadata, 73-74
  unlocalized names, 58
  method names, 26-27
  texture names, 63
naming packages, 30
natural growing (plants), 158-159
natural spawns, 274-275
NBT (Named Binary Tag), 189
NetBeans, 8
nether ore generation, 200
Network List window, 334
New Java Class window, 31
New Java Package window, 29
New Profile window, 211
New User Variable window, 14
newEx, 285
night, attacking by, 386-390
numbers (line), displaying, 86-89

@Override annotation, 138

P
packages, 21
cleaning, 312-313
creating, 27-30
explained, 22-23
naming, 30
renaming, 22
particles, spawning, 390-391
PATH variable, 14

pixe, 96
in game, 101
ItemSamPickaxe class, 97-98
mod file code, 96
planetminecraft.com, 329
plants, generating, 152, 207
creating, 152-155
crops, 155-157
customizing, 165-171
in game, 170-171
natural growing, 158-159
random plants, 157-158
results in game, 219
SamEventHandler class, 215-219
seeds, 159-165

WorldGen class, 207-215
checking and placing, 209
finished
WorldGenSamPlant, 210
initial WorldGenSamPlant class, 207
random start metadata, 210
randomizing location, 208-209
WorldGenSamPlant extending WorldGenerator, 208

PlayerInteractEvent, 303-307
playing sounds
playSound method, 391
step sounds, 391
playSound method, 391
PostInit event, 26
potion effect, 89-91

Preferences window, 211
PreInit event, 26

Oracle NetBeans, 8
OreDictionary class, 202-204
recipes, 203-204
registerOre method, 202-203
ores, generating
EventHandler class, 195
nether ore generation, 200
OreDictionary class, 202-204
results in game, 204

SamEventHandler class, 195-202
addOreSpawn method, 198-200
finished SamEventHandler class, 200-202
generation methods, 197-198
initial SamEventHandler class, 196
IWorldGenerator class, 196-197

@Override annotation, 138

O
.obj files, 263
onArmorTick method, 119, 365-367
onDataPacket method, 188-190
onImpact method, 285
onItemRightClick method, 289-290
onItemUse method, 162-163
Open a Level option (MCEdit), 227
open source mods
copying code, 338
finding, 336-337
how to use, 337-338
understanding code, 337

.obj files, 263
.onArmorTick method, 119, 365-367
.onDataPacket method, 188-190
.onImpact method, 285
.onItemRightClick method, 289-290
.onItemUse method, 162-163
Open a Level option (MCEdit), 227
open source mods
copying code, 338
finding, 336-337
how to use, 337-338
understanding code, 337

Oracle NetBeans, 8
OreDictionary class, 202-204
recipes, 203-204
registerOre method, 202-203
ores, generating
EventHandler class, 195
nether ore generation, 200
OreDictionary class, 202-204
results in game, 204

SamEventHandler class, 195-202
addOreSpawn method, 198-200
finished SamEventHandler class, 200-202
generation methods, 197-198
initial SamEventHandler class, 196
IWorldGenerator class, 196-197

@Override annotation, 138

P
packages, 21
cleaning, 312-313
creating, 27-30
explained, 22-23
naming, 30
renaming, 22
particles, spawning, 390-391
PATH variable, 14

pixe, 96
in game, 101
ItemSamPickaxe class, 97-98
mod file code, 96
planetminecraft.com, 329
plants, generating, 152, 207
creating, 152-155
crops, 155-157
customizing, 165-171
in game, 170-171
natural growing, 158-159
random plants, 157-158
results in game, 219
SamEventHandler class, 215-219
seeds, 159-165

WorldGen class, 207-215
checking and placing, 209
finished
WorldGenSamPlant, 210
initial WorldGenSamPlant class, 207
random start metadata, 210
randomizing location, 208-209
WorldGenSamPlant extending WorldGenerator, 208

PlayerInteractEvent, 303-307
playing sounds
playSound method, 391
step sounds, 391
playSound method, 391
PostInit event, 26
potion effect, 89-91

Preferences window, 211
PreInit event, 26
preInit method, 54-57
- armor code, 113
- tool code, 98-99
proxies
- changing, 313
- ClientProxySam class, 248, 272
- CommonProxySam class, 247
- creating, 246-247
- explained, 246
public classes, 24-25
public fields, 25
public keyword, 24-25

Q-R

random plants, creating, 157-158
randomDisplayTick method, 390-391
randomizing drops, 299
randomizing location, 208-209
readFromNBT method, 188-190
recipes
- chest items, 49
  - adding, 49
  - removing, 49
- crafting recipes, 40-41
  - explained, 39
- GameRegistry, 41-42
- shape of recipes, 42-44
- custom ItemStacks, 46
- examining in mod files, 49-51
- explained, 39-40
- ItemStacks, 41-42
- mob spawn in dungeons, 48
  - adding mobs, 48
  - changing behavior of, 363-364
- custom AI, 384-390
- removing mobs, 48
- OreDictionary class, 203-204
- shapeless recipes
  - creating, 44
  - explained, 39
- smelting recipes
  - creating, 45
  - explained, 40
- redKey item, 69
- registerBlock method, 124, 136
- registerBlockHandler method, 350
- registerBlockIcons method, 177
- registerDimension method, 367
- registerEntityEgg method, 273
- registerEntityRenderingHandler method, 272, 363
- registerIcons method, 75-76, 146
- registering
  - blocks, 124, 136
  - events, 303
  - images, 75-76
  - ores, 202-203
  - Render class, 393
  - throwable entities, 283
- registerItem method, 56-57
- registerItemRenderer method, 393
- registerItemRenderers method, 290
- registerModEntity method, 248-249, 283, 363
- registerOre method, 202-203
- registerRendering method, 247-248
- registerTileEntity method, 180
- releasing mods, 325
  - exporting mods with Forge, 325-327
  - uploading mods to websites, 328-329
- removeDungeonMob method, 48
- removeltmob method, 49
- removeSpawn method, 297-298
- removing
  - chest items, 49
  - cubes, 257-258
  - entities, 297-298
  - mobs, 48
- renaming
  - .jar files, 328
  - packages, 22
- Render class, 270-271
- RenderCustomWoodExample class, 346-350
- RenderExampleItem class, 391-393
- RenderExampleTNTPrimed class, 361-363
- rendering
  - blocks
    - BlockCustomRenderWood
      - Example class, 343-346
    - RenderCustomWood
      - Example class, 346-350
    - custom Render class, 391-393
    - getRenderType method, 155
  - mobs
    - client proxy, 272
    - model class, 265-268

How can we make this index more useful? Email us at indexes@samspublishing.com
moving body parts, 268-270
registerEntityRendering-Handler method, 272
Render class, 270-271
RenderingRegistry class, 272
throwable entities, 290-291
TNT, 361-363
RenderingRegistry class, 272
RenderLiving class, extending, 271
RenderSamMob class, 270-271
repositories (GitHub), 328-329
return types, 26
right-clicking in BlockSamTE, 185-188
rotating
blocks, 342-343
entity models, 257

S

SamDungeon class, 235
samdust item, 70-72
SamEventHandler class, 195-202
addOreSpawn method, 198-200
finished SamEventHandler class, 200-202, 237-239
generation methods, 197-198
initial SamEventHandler class, 196
IWorldGenerator class, 196-197
with WorldGenSamPlant code, 215-219
samingot item, 70-72
samium ToolMaterial, 95-96
Sams Teach Yourself Java in 24 Hours (Cadenhead), 338
Sam's Wall texture, 142-143
SamsMod class
cleaning up, 320-322
code listing, 314-319
creating, 31-33
ModBlocks, 319-320
Schematic Converter
downloading, 231
generating structures with, 231-239
generation file after export, 234-237
SamEventHandler class, 237-239
installing, 231-232
seeds, creating, 159-165
Select Workspace Directory window, 18
Selection tool (MCEdit), 228
sending data with server synchronization, 188-192
server synchronization, tile entities, 183-185
right-clicking in BlockSamTE, 185-188
sending data, 188-192
setAlwaysEdible method, 90
setBlock method, 242
setBlockBoundsBasedOnState method, 152
setBlockBounds method, 148
setBlockMetadataWithNotify method, 182
setBlockName method, 126
setBlockTextureName method, 131, 139-140
setCreativeTab method, 58
setHardness method, 126
setHarvestLevel method, 127-128
setHasSubTypes method, 73, 137
setPosition method, 241
setPotionEffect method, 89-90
setResistance method, 126
setRotationAngles method, 268-270
setStepSound method, 126-127
setTextureName method, 63
setting up
Eclipse, 16
Forge, 16-19
GitHub repository, 328-329
IRC (Internet Relay Chat), 334-335
JDK (Java Development Kit), 9-15
Linux, 15
Mac, 15
Windows, 9-15
setUnlocalizedName method, 58, 73
shape of recipes
alternative shapes, 42-44
explained, 42
ShapedOreRecipe class, 203
shapeless recipes, 39
shapes, creating, 148-151
shovel
code listing, 100
in game, 101
textures, 101
Show Line Numbers check box, 88
sided textures, adding to blocks, 145-147
@SidedProxy annotation, 246, 313
@SideOnly annotation, 139-140
smelting recipes
  creating, 45
  explained, 40
sounds
  playSound method, 391
  step sounds, 126-127
soundTypeStone, 126-127
source folders, 22
spawn eggs, 273-274
spawnEntityInWorld method, 241
spawning
  mobs
    natural spawns, 274-275
    spawn eggs, 273-274
  particles, 390-391
static fields, 25
static keyword, 25
step sounds, 126-127
stone slabs, 123
String, declaring, 24
structures
  adding mobs to, 240
    manual spawning of entities, 240-241
    mob spawners, creating, 241-243
  creating in MCEdit, 225-227
  exporting from MCEdit, 228-229
generating, 231-239
Schematic Converter
  downloading, 231
  generating structures with, 231-239
  installing, 231-232
structuring mods
  cleaning mod class, 314-322
  cleaning packages, 312-313
  importance of, 311
@SubscribeEvent, 298
swords
  enchanted sword, 41-42
  ItemSamSword
    code listing, 100
    in game, 101
    textures, 101
  knockback sword, 47
synchronization (server), 183-192
syso, 27
System Properties window, 11
System.out.println (syso), 27

T

tabs, CreativeTabs, 356
Technne
  creating entity models with, 253
    adding/removing cubes, 257-258
    editing cube properties, 256-257
  exporting models, 259-263
  model characteristics, 253-254
  model creation screen, 255
downloading, 253-254
installing, 253-254
textures
  adding to blocks, 129-131
  Sam’s Wall texture, 139-143
  sided textures, 145-147
  adding to entity models, 256-257
  adding to items, 61-64
  armor
    armor piece textures, 114
    in game, 114
    worn armor texture, 116-118
  ItemBerry class, 85-86
  ItemSamAxe, 101
  ItemSamPaxel class, 107
  ItemSamPickaxe class, 97
  ItemSamSeed class, 161
  metadata, 75
  mirror texture, 257
  zombie texture, 257-258
throwable entities, 283-293
  EntitySamThrowable class, 284-287
  in game, 291-293
  ItemSamThrow class, 287-290
  registering, 283
  rendering, 290-291
throwEggs method, 303-307
tile entities
  adding to blocks, 176-180
  compared to blocks, 123-124
  creating, 180-183

How can we make this index more useful? Email us at indexes@samspublishing.com
explained, 175-176
server synchronization, 183-185
right-clicking in BlockSamTE, 185-188
sending data, 188-192
TileEntitySam class, 180-183, 190-192
TileEntityMobSpawner class, 241-243
TileEntitySam class, 180-183, 190-192
TNT
custom TNT block, 357-359
RenderExampleTNTPrimed class, 361-363
TNT entity, 359-361
ToolMaterial, creating, 95-96
tools
custom tools (paxel example), 103
ItemSamPaxel class, 104-107
mod file code, 103
explained, 95
in game, 101
ItemSamAxe
code listing, 99
in game, 101
textures, 101
ItemSamHoe
code listing, 99
in game, 101
textures, 101
ItemSamShovel
code listing, 100
in game, 101
textures, 101
ItemSamSword
code listing, 100
in game, 101
textures, 101
MCEdit
adding, 223
advantages of, 223
creating structures, 225-227
downloading, 223-224
exporting structures, 228-229
importing worlds, 227
installing, 225
online resources, 225-226
mod file with tool code in preInit, 98-99
mod file with tool code outside of preInit method, 98
pickaxe, 96
in game, 101
ItemSamPickaxe class, 97-98
mod file code, 96
textures, 101
ToolMaterial, 95-96
tree generation, 350-352

U-V
unbreakable blocks, 127
unlocalized item names, 58
UpdateChecker class, 353-356
updateEntity method, 180-184, 245-246
updates, checking for, 353-356
updateTick method, 158-159
uploading mods to websites, 328-329
variables
creating, 56
modid, 24
PATH, 14

W-X-Y-Z
websites
eclipse.org, 16
modlist.mcf.li, 329
planetminecraft.com, 329
uploading mods to, 328-329
WeightedRandomChestContent parameter (addItem method), 49
windows
Computer Properties, 10
Environment Variables, 13
Formatter, 211-212
HexChat: Edit EsperNet, 334
Network List, 334
New Java Class, 31
New Java Package, 29
New Profile, 211
New User Variable, 14
Preferences, 211
Select Workspace Directory, 18
System Properties, 11
Windows computers, JDK (Java Development Kit) setup, 9-15
world generation

MCEdit
- advantages of, 223
- creating structures, 225-227
- downloading, 223-224
- exporting structures, 228-229
- importing worlds, 227
- installing, 225
- online resources, 225-226

ores
- EventHandler class, 195
- nether ore generation, 200
- OreDictionary class, 202-204
- results in game, 204
- SamEventHandler class, 195-202
- plants, 207
- results in game, 219
- SamEventHandler class, 215-219
- WorldGen class, 207-215
- structures. See structures

WorldGen class, 207-215
- checking and placing, 209
- finished WorldGenSamPlant, 210
- initial WorldGenSamPlant class, 207
- random start metadata, 210
- randomizing location, 208-209
- WorldGenSamPlant extending WorldGenerator, 208

WorldProviderExample class, 367-369

worn armor texture, 116-118
writeToNBT method, 188-190
zombie texture, 257-258