Sams Teach Yourself

Scratch™ 2.0

in 24 Hours

Timothy L. Warner
## Contents at a Glance

<table>
<thead>
<tr>
<th>Introduction</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HOUR 1</strong> What Is Scratch?</td>
<td>7</td>
</tr>
<tr>
<td>2 Creating Your First Project</td>
<td>27</td>
</tr>
<tr>
<td>3 Working with Costumes and the Stage</td>
<td>45</td>
</tr>
<tr>
<td>4 Using Motion Blocks</td>
<td>63</td>
</tr>
<tr>
<td>5 Using Looks Blocks</td>
<td>81</td>
</tr>
<tr>
<td>6 Using Sound Blocks</td>
<td>99</td>
</tr>
<tr>
<td>7 Working with Pen Blocks</td>
<td>119</td>
</tr>
<tr>
<td>8 Using Events Blocks</td>
<td>137</td>
</tr>
<tr>
<td>9 Using Control Blocks</td>
<td>155</td>
</tr>
<tr>
<td>10 Using Operators Blocks</td>
<td>171</td>
</tr>
<tr>
<td>11 Using Sensing Blocks</td>
<td>187</td>
</tr>
<tr>
<td>12 Using Data Blocks</td>
<td>201</td>
</tr>
<tr>
<td>13 Using Cloud Data</td>
<td>217</td>
</tr>
<tr>
<td>14 Adding Multimedia to Your Project</td>
<td>233</td>
</tr>
<tr>
<td>15 Creating Your Own Blocks</td>
<td>249</td>
</tr>
<tr>
<td>16 Documenting Your Project</td>
<td>267</td>
</tr>
<tr>
<td>17 Publishing Your Project</td>
<td>283</td>
</tr>
<tr>
<td>18 Using the Scratch Offline Editor</td>
<td>299</td>
</tr>
<tr>
<td>19 Troubleshooting Your Project</td>
<td>317</td>
</tr>
<tr>
<td>20 Remixing a Project</td>
<td>331</td>
</tr>
<tr>
<td>21 Creating Your Own Sprites and Backdrops</td>
<td>347</td>
</tr>
<tr>
<td>22 Implementing Buttons and Multiple Screens</td>
<td>365</td>
</tr>
<tr>
<td>23 Connecting Scratch to the Physical World</td>
<td>381</td>
</tr>
<tr>
<td>24 Capstone Project: Arcade Game</td>
<td>397</td>
</tr>
<tr>
<td>Index</td>
<td>413</td>
</tr>
</tbody>
</table>
# Table of Contents

## Introduction
- Introduction

## Hour 1: What Is Scratch?
- What Is an Educational Programming Language?
- Understanding Scratch History
- Creating Your Scratch Profile
- Browsing the Scratch Websites
- Previewing Popular Scratch Projects
- Summary
- Workshop
- Challenge

## Hour 2: Creating Your First Project
- Navigating the Scratch Project Editor
- Formally Introducing Scratch Blocks
- Previewing Your Project
- Using My Stuff
- Summary
- Workshop
- Challenge

## Hour 3: Working with Costumes and the Stage
- Understanding Sprites
- Adding and Managing Costumes
- Understanding the Stage
- Adding and Managing Backdrops
- Let's Assemble the Pieces, Shall We?
- Summary
- Workshop
- Challenge
<table>
<thead>
<tr>
<th>Hour 4: Using Motion Blocks</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>Getting to Know the Stage More Intimately</td>
<td>65</td>
</tr>
<tr>
<td>Bouncing Around the Stage</td>
<td>69</td>
</tr>
<tr>
<td>Tracking a Sprite to the Mouse</td>
<td>72</td>
</tr>
<tr>
<td>The Hour 4 Project: Drawing in Scratch 2.0</td>
<td>75</td>
</tr>
<tr>
<td>Summary</td>
<td>77</td>
</tr>
<tr>
<td>Workshop</td>
<td>77</td>
</tr>
<tr>
<td>Challenge</td>
<td>78</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hour 5: Using Looks Blocks</th>
<th>81</th>
</tr>
</thead>
<tbody>
<tr>
<td>Getting to Know the Looks Blocks</td>
<td>81</td>
</tr>
<tr>
<td>Interacting with the Player</td>
<td>87</td>
</tr>
<tr>
<td>Getting Sprites to “Talk” to Each Other</td>
<td>90</td>
</tr>
<tr>
<td>Summary</td>
<td>94</td>
</tr>
<tr>
<td>Workshop</td>
<td>94</td>
</tr>
<tr>
<td>Challenge</td>
<td>95</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hour 6: Using Sound Blocks</th>
<th>99</th>
</tr>
</thead>
<tbody>
<tr>
<td>Understanding Notes and MIDI Instruments</td>
<td>100</td>
</tr>
<tr>
<td>Understanding the Backpack</td>
<td>104</td>
</tr>
<tr>
<td>Playing the Drums!</td>
<td>105</td>
</tr>
<tr>
<td>Using the Sound Library</td>
<td>108</td>
</tr>
<tr>
<td>Recording and Editing Your Own Audio</td>
<td>110</td>
</tr>
<tr>
<td>Summary</td>
<td>116</td>
</tr>
<tr>
<td>Workshop</td>
<td>116</td>
</tr>
<tr>
<td>Challenge</td>
<td>117</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hour 7: Working with Pen Blocks</th>
<th>119</th>
</tr>
</thead>
<tbody>
<tr>
<td>Getting to Know the Pen Blocks</td>
<td>120</td>
</tr>
<tr>
<td>Gaining Some Experience with the Pen Tools</td>
<td>123</td>
</tr>
<tr>
<td>Creating a Simple Drawing Program</td>
<td>126</td>
</tr>
<tr>
<td>Summary</td>
<td>134</td>
</tr>
<tr>
<td>Workshop</td>
<td>134</td>
</tr>
<tr>
<td>Challenge</td>
<td>135</td>
</tr>
</tbody>
</table>
# Hour 8: Using Events Blocks  
- Understanding Events Blocks ........................................ 137  
- Digital Versus Analog Events ........................................... 139  
- Understanding Broadcasts ............................................... 143  
- Watch Me Move! ................................................................. 149  
- Summary ........................................................................... 153  
- Workshop .......................................................................... 153  
- Challenge ......................................................................... 154

# Hour 9: Using Control Blocks  
- Introducing the Control Blocks ......................................... 155  
- Testing Out Control Structures ......................................... 159  
- Working with Clones .......................................................... 164  
- Summary ........................................................................... 167  
- Workshop .......................................................................... 168  
- Challenge ......................................................................... 169

# Hour 10: Using Operators Blocks  
- Becoming Familiar with the Operator Blocks ....................... 171  
- Performing More Complex Math ......................................... 174  
- Integrating Operators into Your Scratch Projects ................ 177  
- Summary ........................................................................... 184  
- Workshop .......................................................................... 184  
- Challenge ......................................................................... 185

# Hour 11: Using Sensing Blocks  
- Introducing Sensing Blocks ............................................... 187  
- Delving Deeper into Sensing .............................................. 192  
- Putting Together Everything You’ve Learned So Far ............... 195  
- Summary ........................................................................... 198  
- Workshop .......................................................................... 198  
- Challenge ......................................................................... 199
# Contents

<table>
<thead>
<tr>
<th>Hour 12: Using Data Blocks</th>
<th>201</th>
</tr>
</thead>
<tbody>
<tr>
<td>What Are Variables?</td>
<td>201</td>
</tr>
<tr>
<td>What Are Lists?</td>
<td>207</td>
</tr>
<tr>
<td>Combining Variables and Lists</td>
<td>213</td>
</tr>
<tr>
<td>Summary</td>
<td>214</td>
</tr>
<tr>
<td>Workshop</td>
<td>214</td>
</tr>
<tr>
<td>Challenge</td>
<td>216</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hour 13: Using Cloud Data</th>
<th>217</th>
</tr>
</thead>
<tbody>
<tr>
<td>What Is a New Scratcher?</td>
<td>218</td>
</tr>
<tr>
<td>Creating Cloud Variables</td>
<td>220</td>
</tr>
<tr>
<td>How to Post High Scores Using Cloud Data</td>
<td>226</td>
</tr>
<tr>
<td>More About Username</td>
<td>228</td>
</tr>
<tr>
<td>Summary</td>
<td>230</td>
</tr>
<tr>
<td>Workshop</td>
<td>230</td>
</tr>
<tr>
<td>Challenge</td>
<td>231</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hour 14: Adding Multimedia to Your Project</th>
<th>233</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adding Photos to Your Project</td>
<td>234</td>
</tr>
<tr>
<td>Adding Audio to Your Project</td>
<td>238</td>
</tr>
<tr>
<td>Adding Video to Your Project</td>
<td>243</td>
</tr>
<tr>
<td>What About Hyperlinks and Interactivity?</td>
<td>246</td>
</tr>
<tr>
<td>Summary</td>
<td>247</td>
</tr>
<tr>
<td>Workshop</td>
<td>247</td>
</tr>
<tr>
<td>Challenge</td>
<td>248</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hour 15: Creating Your Own Blocks</th>
<th>249</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spending Some Time with Layers</td>
<td>249</td>
</tr>
<tr>
<td>Understanding Custom Blocks</td>
<td>254</td>
</tr>
<tr>
<td>Creating Custom Blocks in Scratch 2.0</td>
<td>257</td>
</tr>
<tr>
<td>Building Your Own Blocks in Snap!</td>
<td>263</td>
</tr>
<tr>
<td>Summary</td>
<td>264</td>
</tr>
<tr>
<td>Workshop</td>
<td>265</td>
</tr>
<tr>
<td>Challenge</td>
<td>266</td>
</tr>
</tbody>
</table>
Hour 16: Documenting Your Project ............................................................ 267
  Commenting Your Code ......................................................................... 267
  Working with Pseudocode ..................................................................... 270
  Wireframing and Storyboarding ............................................................... 274
  Publicly Documenting Your Scratch Project ............................................ 277
  Summary ............................................................................................... 280
  Workshop .............................................................................................. 280
  Challenge .............................................................................................. 281

Hour 17: Publishing Your Project ............................................................ 283
  Sharing Your Project ............................................................................. 283
  Project Documentation, Revisited .......................................................... 285
  Interacting with Your Viewers Through Comments ............................... 286
  Interacting with Other Scratchers on the Forums .................................... 291
  Improving Your Project’s Visibility .......................................................... 294
  Summary ............................................................................................... 296
  Workshop .............................................................................................. 297
  Challenge .............................................................................................. 298

Hour 18: Using the Scratch Offline Editor ................................................. 299
  A Bit of Scratch Version History ............................................................ 300
  Introducing the Scratch 2.0 Offline Editor .............................................. 301
  Uploading and Downloading Assets ....................................................... 304
  Understanding the Scratch File Format ............................................... 307
  Integrating Scratch 1.4 with Scratch 2.0 ............................................... 310
  Converting Scratch Projects into Other Formats .................................... 311
  Summary ............................................................................................... 315
  Workshop .............................................................................................. 315
  Challenge .............................................................................................. 316

Hour 19: Troubleshooting Your Project .................................................... 317
  Learning the Basics of Debugging .......................................................... 317
  Resolving Common Scratch Script Errors ............................................. 322
  Accessing Code Block Help .................................................................. 326
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Workshop</td>
</tr>
<tr>
<td>Challenge</td>
</tr>
<tr>
<td><strong>Hour 20: Remixing a Project</strong></td>
</tr>
<tr>
<td>The Importance of Attribution</td>
</tr>
<tr>
<td>Understanding the Remix Tree</td>
</tr>
<tr>
<td>How to Remix a Scratch Project</td>
</tr>
<tr>
<td>Remixing Part of Another Scratcher’s Project</td>
</tr>
<tr>
<td>Improving the Visibility of Your Remix</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Workshop</td>
</tr>
<tr>
<td>Challenge</td>
</tr>
<tr>
<td><strong>Hour 21: Creating Your Own Sprites and Backdrops</strong></td>
</tr>
<tr>
<td>Introducing GIMP</td>
</tr>
<tr>
<td>Understanding Bitmap and Vector Graphics Modes</td>
</tr>
<tr>
<td>Creating a New Stage Backdrop</td>
</tr>
<tr>
<td>Creating a Custom Sprite—The Easy Way</td>
</tr>
<tr>
<td>Creating a Custom Sprite—The Difficult Way</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Workshop</td>
</tr>
<tr>
<td>Challenge</td>
</tr>
<tr>
<td><strong>Hour 22: Implementing Buttons and Multiple Screens</strong></td>
</tr>
<tr>
<td>Working with Multiple Screens in Scratch</td>
</tr>
<tr>
<td>Creating Multistate Buttons</td>
</tr>
<tr>
<td>Wiring Up Buttons to Your Screens</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Workshop</td>
</tr>
<tr>
<td>Challenge</td>
</tr>
<tr>
<td><strong>Hour 23: Connecting Scratch to the Physical World</strong></td>
</tr>
<tr>
<td>Setting Up Scratch 1.4</td>
</tr>
<tr>
<td>Introducing the PicoBoard</td>
</tr>
<tr>
<td>Introducing the MaKey MaKey</td>
</tr>
</tbody>
</table>
About the Author

Timothy Warner is an IT professional and technical trainer based in Nashville, Tennessee. Tim began his programming career in 1982 when his dad bought the family a Timex Sinclair 1000 home computer and he began teaching himself BASIC programming. Today Tim works as a technical trainer for Skillsoft, a premier provider of live instructor-led training. You can reach Tim directly via his LinkedIn profile at https://www.linkedin.com/in/timothywarner.
Dedication

To my beautiful, amazing daughter, Zoey Elizabeth, who loves technology and Scratch programming as much as her daddy does.

Acknowledgments

For a variety of reasons, this book was challenging to write. I extend my biggest debt of gratitude to my valiant and open-minded editors, Rick Kughen and Mark Renfrow. Thanks also to my publishers, Greg Wiegand and Paul Boger—you guys are great. Thank you to the always helpful and efficient Pearson production and administrative staff, including Kristen Watterson, Betsy Gratner, and Kristy Hart.

Technical books like this put a special burden on its content editors. Thanks so much to Patrick Mangan, my technical editor, and to Karen Annett, my copyeditor, for your thoroughness in making the manuscript as best as it can be.

Thanks to my family—Susan, Zoey, and our menagerie of pets—for putting up with my occasional grumpiness as I burned through the rough spots of this project.

Finally, thanks to you, my readers: Without you, I have no teacher-student circuit to complete, and I would write into the void. That’s no fun, so I want you to know how grateful I am that you’re reading this book and participating in this learning journey with me.
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 we cannot help you with technical problems related to the topic of this book.

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.
This page intentionally left blank
“My task, which I am trying to achieve is, by the power of the written word, to make you hear, to make you feel—it is, before all, to make you see.”

—Joseph Conrad, Lord Jim

So you want to learn programming? If you were here, physically right in front of me, I would ask you the following questions:

▶ What was it that got you interested in learning to do computer programming? An iOS or Android app? One of your teachers at school? A family member or friend?

▶ Where do you envision taking your programming skills? Are you considering programming as a career, a money-making venture, or simply a fun, satisfying hobby?

Regardless of your motivations, I’m happy to welcome you to the always challenging, sometimes fun, sometimes tedious world of learning to write computer programs. In choosing Scratch 2.0, you’ve made an excellent choice for your first programming language, if you are a beginner.

Why? Because Scratch is a visual, drag-and-drop programming environment that allows you to be creative without having to get bogged down in learning strange syntax rules like you do in more formal languages such as JavaScript or Python.

By the time you’ve finished this book and completed all of the Try It Yourself exercises, you’ll not only be an expert with Scratch programming, but you’ll also have a number of real-world programming best practices under your belt.

Please note that I don’t dismiss Scratch as a “toy” programming language. In this book, you’ll make various projects that other people can actually play and enjoy. We’re talking about programs like games, educational interactions, or multimedia storybooks—the proverbial sky is the limit.
Who Should Read This Book

Any author worth his or her salt always writes with the audience in mind. As far as I’m personally concerned, I envision my readers as coming from one (or more) of the following experience contexts:

- **Brand new to programming**: Welcome! You don’t need any prior experience with programming to gain value from this book. The only related experience I hope you have is a lot of time spent playing video games and using other multimedia apps so you have some ideas to pursue in Scratch.

- **Considering a career change**: Perhaps you are a K-12, junior college, or university student who has perhaps a bit of past programming experience, and you are pondering a full-time career as a software developer. Learning Scratch serves as an excellent diagnostic to gauge your aptitude and interest in the subject matter.

- **Just tinkering**: Maybe you are a technology buff who always wondered what work went into developing a software project. You have no real career aspirations in programming—you just enjoy tinkering and having fun. Well, welcome! You are bound to have a blast learning Scratch!

If you find that you don’t belong in any of the previous three classifications, then don’t worry about it. Set your sights on learning as much as you can and, above all else, having fun, and you’ll be fine!

How This Book Is Organized

Can you learn how to program with Scratch 2.0 in 24 one-hour sessions? Absolutely! The following chapter-by-chapter breakdown details how the material is structured:

- **Hour 1**, “What Is Scratch?” formally defines what Scratch is, how it came to be, and how you can use the platform to learn real, honest-to-goodness computer programming skills.

- In **Hour 2**, “Creating Your First Project,” you create your first Scratch project. If you are to become an expert Scratch programmer, then you need to get right into the mix.

- In **Hour 3**, “Working with Costumes and the Stage,” you turn your attention to sprites and their potentially many costumes. You also formally meet the Stage and its accompanying backdrops.

- In **Hour 4**, “Using Motion Blocks,” you begin a detailed consideration of every script block in Scratch 2.0. Here, you use Motion blocks to make stuff happen on the Stage.
In Hour 5, “Using Looks Blocks,” you learn how Looks blocks enable you to make sprites interact with your players.

In Hour 6, “Using Sound Blocks,” you add audio to your Scratch projects.

In Hour 7, “Working with Pen Blocks,” you learn to draw on the Stage by using the fun Pen blocks.

In Hour 8, “Using Events Blocks,” you get comfortable with event-driven programming and using Events blocks to orchestrate action in your Scratch projects.

In Hour 9, “Using Control Blocks,” you use the powerful Control blocks to program potentially complicated branching and looping logic.

In Hour 10, “Using Operators Blocks,” you apply (ugh!) mathematics and logical thinking by means of Scratch’s Operators blocks.

In Hour 11, “Using Sensing Blocks,” you start to operate with both analog and digital processes and interact with the player more intimately through the Sensing blocks.

In Hour 12, “Using Data Blocks,” you learn how to implement dynamic data (that is to say, variables and lists) into your Scratch 2.0 projects.

In Hour 13, “Using Cloud Data,” you take what you learned about local variables in Hour 12 and scale them out to the cloud by using cloud data—exciting, cutting-edge stuff here, people!

In Hour 14, “Adding Multimedia to Your Project,” you add multimedia (recorded audio, video) to your Scratch projects.

In Hour 15, “Creating Your Own Blocks,” you learn how to build your own custom blocks using the built-in tools in the Scratch 2.0 editor.

In Hour 16, “Documenting Your Project,” you pick up some real-world programming best practices as they relate to source code documentation and unit testing.

In Hour 17, “Publishing Your Project,” you take your work and publish your project on the Scratch website to enable other people from all over the world to play your game.

In Hour 18, “Using the Scratch Offline Editor,” you discover how you can work on your Scratch projects even if your computer is not connected to the Internet.

In Hour 19, “Troubleshooting Your Project,” you learn valuable tips and tricks for debugging your Scratch 2.0 projects, ensuring that they are free from errors and give players the best possible experience.

In Hour 20, “Remixing a Project,” you “stand on the shoulders of giants” by building new Scratch 2.0 projects based on the work of other Scratchers.
In Hour 21, “Creating Your Own Sprites and Backdrops,” you put on your artist’s beret and learn how to draw your own sprite costumes and Stage backdrops by using both the built-in Paint Editor as well as a third-party image-editing program.

In Hour 22, “Implementing Buttons and Multiple Screens,” you put your Scratch project on another level of quality by including multiple game screens and button controls.

In Hour 23, “Connecting Scratch to the Physical World,” you use third-party add-on products to link your Scratch programs with stuff happening in the real world. (Think temperature, touch, volume...this is some fun stuff, trust me!)

In Hour 24, “Capstone Project: Arcade Game,” you consolidate what you learned through the previous 23 hours by writing and publishing a fully functional game.

**Downloading the Sample Files**

As you’ll learn soon enough, the Scratch programming community is all about resource sharing. To that end, the solution files for every Try It Yourself exercise in the book are provided for you in the solution archive. To access the archive, go to [www.informit.com/title/9780672337093](http://www.informit.com/title/9780672337093) and click the Downloads tab.

**Conventions Used in This Book**

In my experience as an author and a teacher, I’ve found that many readers and students skip over this part of the book. Congratulations for reading it! Doing so will pay off in big dividends because you’ll understand how and why we formatted this book the way that we did.

**Try It Yourself**

Throughout the book, you’ll find Try It Yourself exercises, which are opportunities for you to apply what you’re learning right then and there in the book. I do believe in knowledge stacking, so you can expect that later Try It Yourself exercises assume that you know how to do stuff that you did in previous Try It Yourself exercises.

Therefore, your best bet is to read each chapter in sequence and work through every Try It Yourself exercise.

**About the is.gd Hyperlinks**

Whenever I want to point you to an Internet resource to broaden and deepen the content you’re learning, I provide a uniform resource locator (URL, also called an Internet address) in the form: [http://is.gd/uaKpYD](http://is.gd/uaKpYD)
You might wonder what the heck this is. The way I look at the situation, if I were reading this title as a print book and needed to type out a URL given to me by the author, I would rather type in a “shortie” URL than some long, crazy URL with all sorts of special characters, you know what I mean?

*The most important thing I have to tell you concerning the is.gd short URLs is that the ending part is case sensitive.* Therefore, typing the previous URL as `http://is.gd/UaKpyD` isn’t going to get you to the same page as what I intended.

---

**NOTE**

### URL Shortening Services

Is.gd is just one of many URL shortening services; others include bit.ly, goo.gl, and TinyURL.com. I like is.gd because the service is free and the owner operates with high integrity. For more information on is.gd, visit their FAQ page: [http://is.gd/faq.php#owner](http://is.gd/faq.php#owner).

---

**TIP**

### Notes, Tips, and Cautions

This book uses the Note formatting (see the previous URL Shortening Services note) to frame supplemental content that adds to the current topic of discussion. Or, perhaps the Note represents a clarification or an expansion of the information. This extra information could also be formatted as a Tip, which identifies tips, tricks, or other pieces of expert advice, or a Caution, which warns you of potential hazards. Call it potpourri!

---

### About the Code Images

For most Try It Yourself exercises, you’ll see one or more source code images that are annotated with alphabetical letters. The Try It Yourself steps are then cross-referenced with parts of each code image. Hopefully, you find this format convenient to your learning. Remember not to fall into the trap of blindly copying the provided code; instead, remember that learning to program requires (yes, requires) lots and lots of trial and error.

---

### System Requirements

You don’t need a heck of a lot, computer-wise, to perform all of the Try It Yourself exercises in this book. However, if you do not meet the necessary system requirements, then you are stuck. To that end, make sure that you have the following met prior to beginning your work:

- **A standard computer:** It doesn’t matter whether your computer runs Windows, OS X, or Linux. Likewise, you can use either a desktop or laptop computer. However, Scratch 2.0
won’t run on any device that does not fully support Adobe Flash. That rules out, at the least, iOS devices such as iPhones, iPads, and iPod touches.

- **An Internet connection:** Scratch 2.0 is a web application, so you need to be connected to the Internet to complete the exercises. Yes, in Hour 18, you’ll learn about the Scratch 2.0 Offline Editor. However, the Offline Editor does not support all Scratch 2.0 features, and in this book you learn to use all of the features in the product.

- **An Adobe Flash-enabled web browser:** Again, it doesn’t matter whether your web browser of choice is made by Microsoft, Apple, Google, or another vendor—what does count is whether the browser has the Adobe Flash plug-in installed. Point your browser to the Adobe website (http://is.gd/pCNcvd) to perform a check; if you don’t have the plug-in, you can install it at Adobe.com for free.

Okay—that’s enough of the preliminaries. It’s time to learn to program with Scratch 2.0!
What You’ll Learn in This Hour:

- Getting to know the Looks blocks
- Interacting with the player
- Getting sprites to “talk” to each other

The main theme for this hour is that of involving your user in your Scratch project. I’ve always enjoyed video games of the first-person shooter (FPS) variety; however, I have remarkably low patience for long, drawn-out cut scenes that have no player interaction. I just want to get to the good stuff and to start playing the darned game!

Likewise, you should always keep your player in mind as you develop your Scratch projects. Believe me, they don’t want to sit there twiddling their thumbs while you present all of your nifty animations. Instead, they want to control the behavior and perhaps the outcome of the project—and you can make that happen.

The Looks block palette in Scratch 2.0 includes lots of goodies that put the power and control into your players’ hands. In this hour, you learn how to communicate with your player both by using speech and thought bubbles. You also learn how to ask the player questions, get his or her feedback, and act accordingly on that feedback.

By the end of this hour, you’ll have the ability to grow, shrink, fade, or otherwise modify your sprites in novel and effective ways. Let’s get to work.

Getting to Know the Looks Blocks

You know the drill by now: Fire up a new, blank project, select the Cat sprite, and click the Looks tab from the Scripts palette. Next, study Table 5.1 to familiarize yourself with the purple Looks blocks.
### TABLE 5.1  Scratch Looks Blocks

<table>
<thead>
<tr>
<th>Block Image</th>
<th>Block Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>say Hello for 2 secs</strong></td>
<td>Stack</td>
<td>Makes a time-limited speech balloon appear above the selected sprite</td>
</tr>
<tr>
<td><strong>say Hello</strong></td>
<td>Stack</td>
<td>Makes a persistent speech balloon appear above the selected sprite</td>
</tr>
<tr>
<td><strong>think Hmm... for 2 secs</strong></td>
<td>Stack</td>
<td>Makes a time-limited thought balloon appear above the selected sprite</td>
</tr>
<tr>
<td><strong>think Hmm...</strong></td>
<td>Stack</td>
<td>Makes a persistent thought balloon appear above the selected sprite</td>
</tr>
<tr>
<td><strong>show</strong></td>
<td>Stack</td>
<td>Displays the sprite immediately</td>
</tr>
<tr>
<td><strong>hide</strong></td>
<td>Stack</td>
<td>Hides the sprite immediately</td>
</tr>
<tr>
<td><strong>switch costume to costume2</strong></td>
<td>Stack</td>
<td>Changes the sprite’s costume to another one</td>
</tr>
<tr>
<td><strong>next costume</strong></td>
<td>Stack</td>
<td>Transitions the sprite from the current costume to the next one in the sprite’s costume list</td>
</tr>
<tr>
<td><strong>switch backdrop to backdrop1</strong></td>
<td>Stack</td>
<td>Changes the Stage backdrop to a particular one</td>
</tr>
<tr>
<td><strong>change color\texttimes\texttimes effect by 25</strong></td>
<td>Stack</td>
<td>Changes the current value of a particular sprite’s graphical effect by a percentage; the effects are color, fisheye, whirl, pixelate, mosaic, brightness, and ghost</td>
</tr>
<tr>
<td><strong>set color\texttimes\texttimes effect to 0</strong></td>
<td>Stack</td>
<td>Sets the sprite’s effect to a percentage</td>
</tr>
<tr>
<td><strong>clear graphic effects</strong></td>
<td>Stack</td>
<td>Restores all of the sprite’s graphical effects to their default zero value</td>
</tr>
<tr>
<td><strong>change size by 10</strong></td>
<td>Stack</td>
<td>Changes the sprite’s default size (100) to a larger or smaller value</td>
</tr>
<tr>
<td><strong>set size to 100 %</strong></td>
<td>Stack</td>
<td>Sets a sprite’s size to a given magnification level</td>
</tr>
<tr>
<td><strong>go to front</strong></td>
<td>Stack</td>
<td>Brings the sprite to the top layer of the Stage “stack”</td>
</tr>
<tr>
<td><strong>go back 1 layers</strong></td>
<td>Stack</td>
<td>Moves the sprite back one layer on the Stage</td>
</tr>
</tbody>
</table>
Getting to Know the Looks Blocks

<table>
<thead>
<tr>
<th>Block Image</th>
<th>Block Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="costume #" /></td>
<td>Reporter</td>
<td>References the sprite’s current costume ID number</td>
</tr>
<tr>
<td><img src="image" alt="backdrop name" /></td>
<td>Reporter</td>
<td>References the current backdrop’s Name property</td>
</tr>
<tr>
<td><img src="image" alt="size" /></td>
<td>Reporter</td>
<td>References the sprite’s current size</td>
</tr>
</tbody>
</table>

Make sure that you have the Scratch Cat sprite selected on Stage and spend some time double-clicking each of the Looks blocks. You’ll observe that you can try out that block’s functionality without having to compose an actual script.

For instance, double-click the ![costume #](image) block. Now change the “Hello!” text to something else and test it out again.

This “click and try it” procedure is especially fun with the graphical effects stack blocks. Have fun, friend—that is largely what computer programming is all about!

By way of review, recall that stack blocks have a notch on top and a bump on bottom, just like a jigsaw puzzle piece. This is meant to indicate that you can easily stack these blocks together, one after the other, to chain actions and trigger events.

Reporter blocks are shaped like flattened-out ovals and hold values. You use reporter blocks by dropping them into a space within another block.

Boolean blocks (shaped like flattened-out hexagons) are a special type of reporter block. Whereas reporter blocks can hold alphanumeric data, Boolean blocks can contain only True or False values. You can insert Boolean blocks into the appropriately shaped holes in other Boolean blocks.

Hopefully, you are beginning to see the beauty and logic in how Scratch uses color-coded blocks to help you think like a programmer without getting bungled up in arcane syntax.

NOTE

**Imitation Is the Sincerest Form of Flattery**

You should know that Scratch is no longer the only game in town (pun most certainly intended) with regard to block-based, learning programming environments. Google released Blockly (http://cbt.gg/1969Kus) as a web-based, graphical programming editor whose color-coded blocks look and behave suspiciously like those in Scratch.

As you can imagine, some members of the Scratch community have mixed feelings about Google’s “appropriation” of Scratch’s main design and usage paradigms. On the other hand, competition
between software vendors can often lead the way to increased innovation. For a nice comparison among most of today’s graphical and/or block-based programming learning environments, check out Alfred Thompson’s Computer Science Teacher blog at http://cbt.gg/1969T10.

For the first project example in this hour, let’s play around with a sprite’s appearance. To set the foundation for this exercise, do the following in the Scratch Editor:

- Rename the Scratch Cat to Cat.
- Import the Stage backdrop named route66 and make it the active backdrop.
- Turn on the and reporter blocks to make them appear as Stage monitors. To do this, simply enable the check box to the immediate left of each reporter block in the Looks block palette. You can see what this looks like in Figure 5.1.

\[\text{FIGURE 5.1}\]
Stage monitors make Scratch reporter data visible on the Stage. This is extraordinarily useful when you debug your project code or when you want to display information (such as a game score) to the user.

\[\text{▼ TRY IT YOURSELF}\]

\[\text{Lookin’ Good!}\]

In this Try It Yourself exercise, you become familiar with how you can dynamically alter the appearance of a sprite by using Looks blocks. You start by having the Cat “say” something to the player, and then you make the sprite do a bunch of contortions, including growing, recoloring, warping, spinning, and then fading out.

The completed solution file is named Hour05a.sb2.

Complete the following steps using Figure 5.2 as a guide, which shows you the code in context:
1. Bring out a Green Flag block and ignore the code section labeled A in Figure 5.2 for now. Note the say and think blocks; the difference between these is the appearance of the bubble that appears by your sprite. For the purposes of this exercise, you want the Cat to say “Watch this!” for 2 seconds (see the code section labeled B in the figure).

2. After a 1-second pause, switch the Cat’s costume (see the code section labeled C in the figure). Remember that the wait blocks are found in the Control palette.

3. Sprites start out at a size value of 100. Thus, if you bring out a change size by block and use a value of 150 (see the code section labeled D in the figure), the sprite will grow by 150 percent. In other words, the sprite’s size will go from 100 to 250.
4. Change the sprite’s color effect by a factor of 50 (see the code section labeled E in the figure). A sprite’s default color effect is 0, so any value you add or subtract alters the sprite’s shade. A single sprite costume can take on 200 different color schemes by using the `set color effect to` block. Thus, if you set the change color effect block to 200, you’ll see no difference in the sprite’s color.

5. Take a moment to just play around with other effects (see the code section labeled F in the figure). The fisheye effect is pretty cool; the higher you set the value about the default value of 0, the more warping you see in the sprite.

6. Spin the sprite in a 360-degree (full) rotation (see the code section labeled G in the figure). There are probably several ways in which you could accomplish this goal. For my money, repeating a 15-degree turn 24 times (perform the arithmetic; you’ll find that 15 multiplied by 24 equals 360) gets the job done efficiently enough.

7. Instead of having the Cat simply disappear with a `clear` block, add some pizzazz to the project (see the code section labeled H in the figure). The ghost effect is excellent if you want to fade in or fade out a sprite. Here, increase the sprite’s ghost (transparency) effect by 10, 10 times. In Scratch 2.0, a costume can have 100 different transparency levels. If you run a +10 ghost level 10 times, then by the end of the loop the sprite is fully transparent.

8. You’ll get into adding and managing audio in your Scratch projects in the next hour. For now, simply play the default meow sound to signify the conclusion of the project (see the code section labeled I in the figure).

9. Add a hide block just for grins (see the code section labeled J in the figure). In programming, being explicit with your code is generally superior to being implicit.

10. Now return to the code section labeled A in Figure 5.2. This is discussed last so the blocks used make sense to you. If you try to rerun the project without this “cleanup” code, the project will look a mess and be pretty much unusable.

What do you think of the Stage monitors? Pretty cool, aren’t they? Here’s something else for you to try: Double-click each Stage monitor and notice what happens.

Sometimes the value you are reporting in a Stage monitor doesn’t need a label, or perhaps you added the label to the Stage itself. For instance, I like the larger, no-label view in some of my games.
Interacting with the Player

I don’t know about you, but when I’m playing a game or interactive presentation, I want to do something. The last thing you want to do as a Scratch developer is to bore your players.

Thus, the more options you give to your players, the more you ask (or require) them to take action on their part, the more involved they’ll be in your project.

One great way to interact with the player is to have a sprite “ask” the player for input. By prompting the user for input and then adding programming logic to react to that input, you accomplish many goals, including the following:

▶ The project becomes more dynamic instead of the same exact thing every time that it is run.
▶ The user feels that the project is personalized for him or her.
▶ The project has a longer “shelf life” because it has more than one outcome, and the outcome is at least partially dependent upon user input.

Here’s how it works: First, navigate to the Sensing palette and bring out an  block. You can add any text you want to the admittedly small text area.

During your program execution, a prompt box will appear at the bottom of the Stage, allowing the user to type some data and press Enter.

That answer from the user is captured and stored in the reporter block.

NOTE

Player, User, or Something Else?

This text has stressed to you many times the importance of coding your Scratch project with the user in mind. Just to be clear: When the text refers to “the user” or “the player,” it’s referring generically to the individuals who will access your project on the Scratch website.
Ask the User

In this Try It Yourself exercise, you have the Scratch Cat ask the player if he or she wants the Cat to grow or shrink in size. Depending upon the player’s answer, the Cat then obeys the player’s command.

You should create a new project that is set up the same way as what you had in the previous exercise. One important change: Add a second backdrop to the Stage, and name it end. You can use the Fill tool, which you’ll learn about in Hour 21, “Creating Your Own Sprites and Backdrops,” to create a black screen. You can then use the Text tool to add a simple “The End” banner, which you can see in the solution file.

The completed solution file is called Hour05b.sb2.

Complete the following steps, using Figure 5.3 as a guide, which shows you the code in context:

1. Make sure that the Cat sprite is selected, and head over to the Scripts area. In the code section labeled A in Figure 5.3, you have the “cleanup” code that ensures that the sprite shows up in the same spot and is the same size every time that the Green Flag is clicked.

2. You want to be as descriptive as possible when you bring out the ask block (see the code section labeled B in the figure). For instance, adding What do you want me to do? (Options are ‘shrink’ or ‘grow’) and wait tells the player exactly what is expected of him or her.
As you worked through the Ask the User Try It Yourself exercise, you probably had the thought, “What if the user were to type something other than *grow* or *shrink*?”

If you did have that question, then good for you! That’s what it takes to think like a computer programmer. This case of the user typing something you don’t anticipate is called an exception. Unhandled exceptions are one of the biggest reasons why programs crash.

Asking a user for input, as you can see in Figure 5.5, can be risky because that user can possibly submit invalid or unhandled input to your project.

Exception handling is covered in greater detail in Hour 19, “Troubleshooting Your Project”; for now, take a look at Figure 5.4 to see one way to handle this particular exception.

---

3. The main “engine” of this project occurs in the code section labeled C in the figure. You want to test for two conditions. The first condition will say “If the player types ‘grow,’ then the sprite should grow 250 percent. If the player types ‘shrink,’ then the sprite should shrink to 25 percent of its original, default size.”

You can turn that pseudocode into real code by using the if else Control block.

4. You can embed if or if else C blocks to test for more than one condition. In the code section labeled D in the figure, if the test for *grow* fails, then you proceed to the second, embedded if statement. Here, you catch the event of the player typing *shrink*.

5. Because the code in the code section labeled E in the figure needs to run regardless of whether the player grew or shrunk the sprite, you place these blocks outside of the C block structure. In this case, you switch to your second backdrop, hide the Cat sprite, and stop program execution.

---

![FIGURE 5.4](image)

Trapping exceptions with Scratch blocks might not always be pretty, but it is possible. This code picks up the case where the player enters anything other than the required keywords.
FIGURE 5.5
Asking the user for input is a great way to build buy-in and interest for your Scratch project.

You need to become comfortable with the notion of nesting reporter blocks into Boolean blocks, and Boolean blocks into other Booleans. Remember that reporter blocks store variable data; you can pop them into any block that has a white, rectangular cutout.

By contrast, Boolean blocks look like flattened hexagons and have hexagonal cutouts as well as reporter cutouts.

Getting Sprites to “Talk” to Each Other

A common question among Scratchers who have made some progress in their work is, “How can I get sprites to affect each other?” For instance, what if you want Sprite B to move when Sprite A touches it?

In Scratch, you use broadcasts to pass messages among sprites. You can even communicate between sprites and the Stage by sending and receiving broadcasts.

You will find the broadcast message and broadcast message and wait blocks in the Events palette.

You’ll also see a when I receive message Hat block that you use to catch outgoing broadcasts.

Trust me—before too long, you’ll appreciate how powerful broadcasts are; they will be an indispensable addition to your Scratch programming toolkit.
Here are a few key points to keep in mind regarding Scratch broadcast messages:

- A sprite can both send and receive the same broadcast message.
- Broadcast messages can be received by all sprites (and the Stage).
- Broadcasts are most commonly used to (1) connect different events, (2) run two scripts in the same frame, and/or (3) prepare a scene with multiple sprites.

**NOTE**

**Can You Broadcast to Specific Sprites?**

Unfortunately, Scratch 2.0 has no built-in method for using broadcasts to target specific sprites. As you’ve seen, the default behavior is to make broadcast messages available to all assets in a project.

However, the adventurous can hop on over to the Scratch Wiki (http://is.gd/wwryoS) to learn a workaround (less charitably called a hack) to this behavior.

Essentially, you can tag each sprite with a unique ID by using private variables and then define a global variable that determines who should receive a particular broadcast.

If this procedure sounds frighteningly complex, don’t worry about it for now. After all, you have yet to work with variables—you will, though, trust me!

To get set up for the Move from Room to Room Try It Yourself exercise, fire up a new, blank project that contains the following assets:

- **Sprite**: Default Scratch Cat; rename sprite to Cat.
- **Sprite**: Magic Carpet (look in the Transportation section of the Sprite Library); rename to Carpet.
- **Stage backdrops**: Add the room1 and room2 backdrops from the Indoors category of the Backdrop Library. You can delete the default backdrop for this exercise.

**Move from Room to Room**

For this Try It Yourself exercise, you try your hand at a very brief and very simple interactive story. You first have the Scratch Cat move from one room to another (a very cool trick that you’ll enjoy). Next, you have the Cat ask the user to press a key on the keyboard to move a second sprite (the Carpet) out of the way.

The Cat and Carpet scripts are illustrated in Figures 5.6 and 5.7, respectively.
The completed solution file is named `Hour05c.sb2`. Work through the following steps in order to complete this exercise. Let's get to work!

1. **Begin by coding the Cat's scripts.** Reset the story environment by ensuring that the room1 backdrop is active and the Cat is placed appropriately on the Stage, as shown in the code section labeled A in Figure 5.6.

2. **Have the Cat say something to the player to help him or her feel more engaged with the story** (see the code section labeled B in Figure 5.6).

3. **The repeat block shown in the code section labeled C in Figure 5.6** defines the Cat's animation across the first backdrop. How to do this was covered earlier in the book; simply move the sprite 10 steps, switch its costume, and insert a short pause.
4. Now you get to the heart of the matter. Bring out a broadcast block, open up the drop-down, and select New Message from the menu. You can name a broadcast anything; for this purpose, make it SwitchScreen. (See the code section labeled D in Figure 5.6.)

5. In the code section labeled E in Figure 5.6, you see something that might strike you as surprising; namely, that sprites can actually “listen for” and receive their own messages! In just a moment, you have the Carpet sprite listen for this broadcast as well.

For now, switch the Stage backdrop to the second room and place the sprite appropriately on the Stage. This action, combined with the previously defined sprite movement, gives the player the impression that the sprite traveled from one room to another.

6. Add in some animation blocks to get the Cat moved to the Carpet (see the code section labeled F in Figure 5.6). (We haven’t addressed the Carpet yet, I realize that.)

7. Insert an C block to test for the Sensing condition where the Cat sprite touches the Carpet sprite. If that statement evaluates to True, then bring the Cat sprite to the very top layer on the Stage and have it say “I’m touching the carpet.” (See the code section labeled G in Figure 5.6.)

8. After a 2-second pause, have the Cat sprite say “Press the space bar to get rid of the carpet, please.” You then need to create a second broadcast named EndGame that will be picked up by the Carpet. (See the code section labeled H in Figure 5.6.)

9. Now switch your focus to the Carpet sprite and cross-reference these steps with Figure 5.7. Place the Carpet sprite at (4, -114) on the Stage. Reminder: Coordinates are given as (x,y). Simply hide the Carpet until the Cat reaches the second room (see the code section labeled I in Figure 5.7).

10. Next, listen for the SwitchScreen broadcast. At that time, reset the style and position of the Carpet sprite, and explicitly show it on the Stage. (See the code section labeled J in Figure 5.7.)

11. Finally, listen for the EndGame broadcast, which is, you’ll remember, the broadcast that is triggered if the Cat sprite touches the Carpet sprite. (See the code section labeled K in Figure 5.7.)

Use the wait until block to test for the key space pressed Boolean condition (that block is found in the Sensing palette). If True, then code the carpet to glide horizontally across the screen to an X value that lies beyond the 480 pixels of the Stage. You then invoke the Stop all Control block to end the interaction.
Summary

By now, hopefully you are comfortable with the relationships between sprites, their costumes, the Stage, its backdrops, and how you can use broadcasts to connect scripts among sprites and the Stage.

Don’t feel discouraged if you feel that you’ve learned nothing more than “toy code” thus far. You have all the time in the world to build full projects from beginning to end in the final two hours of this book.

Also, there is no way around the sheer time, repetition, and practice that is required for you to know your way around the Blocks palette.

You know that you are well on your way to becoming an honest-to-goodness computer programmer when your mind starts thinking in terms of “Ohh, I just thought of how to have the program do such-and-such” as opposed to, “Now where is the wait for block again?”

The next hour continues the fun. There, you’ll learn how to integrate sound effects into your Scratch projects.

Workshop

Quiz

1. When you define a broadcast in Scratch, which assets can receive the broadcast message and therefore take action upon it?
   
   A. Only the sending sprite  
   B. Only the receiving sprite  
   C. Only the Stage and its backdrops  
   D. The Stage and all sprites  

2. Which of the following is a valid way to test for two or more conditions in a Scratch script?
   
   A. A Boolean block  
   B. A nested if C block  
   C. A repeat C block  
   D. A broadcast block
3. Which of the following Scratch graphical effects can be used to fade in or fade out a sprite on the Stage?
   A. Color
   B. Pixelate
   C. Mosaic
   D. Ghost

**Answers**

1. The correct answer is choice D. In Scratch 2.0, any broadcasts that are sent out from a sprite or the Stage are receivable by both the Stage as well as all sprites in the project. Recall that the sending sprite can also receive its own message. However, a sprite or the Stage is not obligated to receive a message.

   The Scratch Wiki (http://cbt.gg/1bop3og) includes a hack or workaround that does enable you to effectively target specific sprites with broadcast messages.

2. The correct answer is choice B. By default, an if C block tests for the truth or falsity of a single condition (although you can certainly embed multiple operator blocks—you’ll see how to do that in Hour 10, “Using Operators Blocks.” An easy approach to solving this problem that you discovered in this chapter is nesting one or more if C blocks inside of the outer, original C block.

3. The correct answer is choice D. The ghost graphic effect works well for fading in or fading out a sprite on the Stage. Changing the color effect can make a sprite look like it’s flashing. The pixelate effect makes a sprite look retro or old-fashioned. The mosaic graphic effect is useful for transitioning a sprite between costumes.

**Challenge**

Okay, here is a fun project for you to try out. This challenge gives you some more experience using graphical effects, implementing broadcasts, and responding to the player’s keyboard input.

Here are the design goals for this project:

- Have the Scratch Cat instruct the player to press the spacebar (or another key of your choosing) to transform the sprite into a butterfly.
- Use graphical effects to pixelate the Cat and have it “become” a butterfly.
- Have the butterfly “tell” the gamer that he or she can use the arrow keys to move the butterfly around the Stage.
- Code the butterfly for player control.

Figure 5.8 shows a screenshot of my version of this game, and you can also examine the solution file Hour05d.sb2.
Remember that you need to download and install the free Scratch Offline Editor (http://is.gd/sB2h1k) to view the solution files. You’ll learn everything there is to know about the Offline Editor in Hour 18, “Using the Scratch Offline Editor.”

Before finishing this hour, though, it’s only fair to discuss the pixelate graphic effect just a little because you didn’t use it at all earlier in this hour.

As you can see in Figure 5.9, all you have to do is to wrap a change pixelate effect by block inside of a repeat C block. Test out this sample code block on one of your sprites and experiment with different pixelate intensities.

Also, please understand that there is no shame at all in turning to the Scratch project gallery (http://is.gd/tsr9gM) for help when you get stuck. For example, you can search for projects that include the word pixelate in their title or description, or you can do a tag search for the same term.

Most Scratchers are actually complimented when other Scratchers remix their projects. That’s what the community effort of Scratch is all about, after all.
FIGURE 5.9
Pixelating a sprite has a cool retro effect that is useful for transitions.
Index

Numbers

\(10^\text{ function}, 175\)

A

\(\text{abs function}, 174\)

accounts

- contacting Scratchers, 332
- creating, 15-16
- new Scratcher accounts
  - limits on, 218-219
  - promoting to Scratcher accounts, 219-220
  - purpose of, 218
- Scratcher accounts
  - versus, 218
- profile page, updating, 17-18, 294

\(\text{acos function}, 175\)

activating multistate buttons, 376-378

Activity tab (studios), 344

adding

- audio, 238
- backdrops to Stage, 54-55, 367-368
- comments, 268-269
- costumes to sprites, 52
- custom sprites, 357
- projects to studios, 344-345
- sprites to projects, 47-48
- text to Stage, 370
- transparency to custom sprites, 359-362

Adobe AIR, 301
Adobe Creative Cloud, 348
Adobe Flash, 245
Adobe Photoshop Lightroom, 235
algorithms, 115, 260, 324
Alice programming language, 9, 249
analog events

- digital events versus, 139-140
- PicoBoard sensors, 140-141
- reacting to, 141-143
AND operator, 179, 194
Android player, 312
animating
flipbook animation, 163
GIF animation
converting video to, 243-244
creating, 358
importing, 357-358
Scratch Cat, 33, 56-58
sprites
bouncing, 69-72
collision detection, 73-75
with costumes, 50-51
in Dodgeball Challenge, 400-408
with keyboard, 66-67
with mouse, 72-74
by player control, 68-69
repeating actions, 70-71
stop motion animation
blocks for, 57-58
defined, 50
announcing projects, 286
appearance of sprites, changing, 84-86
arcade game project. See Dodgeball Challenge
Arduino, 383
arguments in Operators blocks, 173-174
arithmetic operators, 171
arrays, 207
asin function, 175
asking for user input, 87-89
aspect ratio, 356
assets
manually importing/exporting, 340
using with Scratch 2.0
Backpack, 338-342
atan function, 175
attaching comments to blocks, 269
attribute
adding manually, 340
obtaining permission, 332
providing, 153
Audacity, 239, 242-243
audio. See also assets; Sound blocks
adding, 238
Audacity, 242-243
ing, 110-112, 239-242
fair use, 238
file formats, 238-239
recording, 110-112, 241
Sound Library, 108-110
uploading, 239-242
Author link (project page), 36
.avi file format, 243
BK
backdrops
adding to Stage, 54-55, 367-368
bitmap versus vector graphics, 351
creating, 353-355
in Dodgeball Challenge, 400-401
multiple screens as, 367
multistate buttons as, 371-372
scaling images, 356
background area. See Stage area (project page)
Backpack (Scratch Project Editor), 28, 104
deleting assets, 105, 339-340
explained, 104-105
practicing with, 341-342
reusing assets, 338-340
Backpack button (Scripts area), 31
balloon popping project, 151-152
Balsamiq Mockups, 275-276, 366
bans, 296
BASIC programming language, 9
beta testing, 410-411
bitmap graphics
converting to vector graphics, 352
vector graphics versus, 351
Block Help button (Scratch Project Editor menu bar), 29
Block Plugin, 271-274
Blockly, 82-84
blocks
attaching comments to, 269
Boolean blocks, 35, 82
conditional logic, 159
with Control blocks, 157-158
in Data palette, 207
nesting, 90
in Operators palette, 171
in Sensing palette, 187-189
blocks

C blocks, 35
   in Control blocks palette, 155
   nesting, 156-157
   repeating actions, 70-71
   testing conditions, 72, 88-89
categories of, 35
   cleaning up in Scripts pane, 77
   clicking and dragging, 59
Control blocks
   Boolean logic, 157-158
   conditional logic, 159
   list of, 155
   looping, 158-159
   nesting C blocks, 156-157
   practicing with, 159-163
   repeating actions, 70-71
   testing conditions, 72
custom blocks
   advantages of, 255
   creating, 257-260
   explained, 255-256
   input parameters, 260-263
   limitations, 254
   in Snap!, 263
Data blocks
   combining variables and lists, 213
   list of list blocks, 207
   list of variable blocks, 202-203
   practicing with lists, 209-212
   practicing with variables, 204-206
   deleting from Scripts pane, 60, 74
detaching, 33
duplicating, 103
Events blocks
   broadcasts, 143-149
   digital versus analog events, 139-140
   list of, 137-138
   video sensors, 149-153
   for layers, list of, 251
Looks blocks
   list of, 81
   showing/hiding sprites, 70
More Blocks, 255-256
Motion blocks, list of, 65
   nesting, 90
   obsolete blocks, troubleshooting, 323
   online help, 326-328
Operators blocks
   arguments, 173-174
   comparison operators, 179
   exponentiation, 175-177
   list of, 171
   practicing with, 179-184
   strings in, 178-179
   trigonometric functions, 174-175
Pen blocks
   changing values, 122-123
   color, 121
   creating drawing program, 126-133
   drawing with, 75-76
   list of, 120-121
   practicing with, 123-126
   shade, 122
   size, 122
   repeating stacks, 260
Reporter blocks, 35, 82
   in custom blocks, dragging into targets, 263
   in Data palette, 203, 207
   in Looks palette, 81
   in Motion palette, 65
   nesting into Boolean blocks, 90
   in Sensing palette, 187-189
   in Sound palette, 99
   Stage monitors, turning on/off, 64-66, 84
   username block, 222-223, 228-229
   video sensors, 149
Sensing blocks
   collision detection, 194
   color-based collision detection, 189-192
   distance, detecting, 194
   list of, 187-189
   location, determining, 192-193
   with multimedia, 194-195
   for PicoBoard sensors, 387-388
   practicing with, 195-198
   shapes, significance of, 71
Sound blocks
   choir singing, 102-104
   drum types, 105-106
   list of, 99
   MIDI virtual instruments, 100-101
   multiple clips, 110
   note values, 101-102
   playing drums, 106-108

How can we make this index more useful? Email us at indexes@samspublishing.com
Stack blocks, 35, 82
    in Control blocks palette, 155
    in custom blocks, 256
    in Data palette, 203, 207
    in Events palette, 137
    for layers, 251
    in Looks palette, 81
    in Motion palette, 65
    in Pen palette, 120
    in Sensing palette, 187-189
    in Sound palette, 99
    video sensors, 149
    testing, 210
types of, 34-35
    for variables, 112-113
Blocks palette (Scratch Project Editor), 28
blogging platforms, 296
boilerplate text, 376
Boolean blocks, 35, 82
    conditional logic, 159
    with Control blocks, 157-158
    in Data palette, 207
    nesting, 90
    in Operators palette, 171
    in Sensing palette, 187-189
Boolean logic, explained, 179
bouncing
    realistic effects for, 409
    sprites, 69-72
breakpoints, 320
broadcasts, 114
    broadcast management program, 145-148
    creating, 145
    in Dodgeball Challenge, 401-403
explained, 143
naming, 93
removing, 145
sending and receiving, 90-92
synchronization with, 163
use cases, 144
waiting for completion, 145
browser, refreshing, 345
Bucket Fill tool (GIMP), 350
bugs. See also troubleshooting
    logical errors, 324
    origin of term, 317
    sending reports on, 411
types of, 318
Build Your Own Block (BYOB). See Snap!
built-in timer, 149, 324
buttons, multistate
    activating, 376-378
    creating, 373-374
    labeling, 374-375
    as Stage backdrops, 371-372
BYOB (Build Your Own Block). See Snap!
Cartesian coordinate system
    explained, 52-54
    xy-grid backdrop, 192
cat, 46
ceiling function, 174
centralizing project code, 321-322
change <variable> block, 113
changing
    pen values, 122-123
    project title, 36, 278
    sprite appearance, 84-86
    sprite color, 86
    volume, 103
chart project example, 222-223
Check for Updates command (Scratch 2.0 Offline Editor), 303
checkerboard pattern, 362
choir singing, 102-104
Choose Sound from Library (Sounds pane), 109
Choose Sprite from Library button (Sprites list), 46
cleaning up Scripts pane, 77
Clear button (Paint Editor), 51
clique and dragging blocks, 59
cloning. See also duplicating
    projects, 69
    sprites, 115, 164
    collision detection, 164
duplicating versus, 164
practicring with, 164-166
Close button (sprite information panel), 49
cloud, defined, 32
cloud services, explained, 217
cloud variables, 202
    creating, 220-221
data types allowed, 221-222
    practicing with high scores lists, 226-228
practicing with surveys, 223-226
username Reporter block, 222-223, 228-229
code blocks. See blocks
code refactoring, 326
code reuse, 255
collapsing comments, 269
collision detection, 73-75
with clones, 164
color-based, 189-192
with Sensing blocks, 194
with sprites, 190
color
collision detection, 189-192
drawing sprites, 128-130
of Pen tools, 121
of sprites, changing, 86
color picker control (GIMP), 350
color-coded script execution, 318-319
comments
adding, 268-269
popularity of projects, 270
on project page
advantages of, 286-287
deleting, 290
netiquette, 287-288
practicing with, 288-290
reporting, 288
in pseudocode, 272-274
purpose of, 267-268
Comments icon (project page), 285
Comments tab (studios), 344
communication between sprites, 90-92. See also broadcasts
comparison operators, 171, 179
compiling projects from Snap!, 312-315
compressing photos, 235
concatenating strings
with numbers, 176
with variables, 206
conditional logic, 159-163
Contact! game, 113-115
contacting Scratchers, 332
Control blocks, 35
Boolean logic, 157-158
conditional logic, 159
list of, 155
looping, 158-159
nesting C blocks, 156-157
practicing with, 159-163
repeating actions, 70-71
testing conditions, 72, 88-89
control structures. See Control blocks
converting
audio file formats, 239
graphics formats, 352
video to animated GIF, 243-244
coordinate system
explained, 52-54
xy-grid backdrop, 192
coordinates (Sprites list), 30
copying. See also duplicating
projects, 332-333
scripts, 104-105
cos function, 174
costume list (Paint Editor), 51
costumes, 50-51. See also assets
adding to sprites, 52
bitmap versus vector graphics, 351
Paint Editor, 50-51
renaming, 51
Create button (home page), 14
credit, providing, 153
cropping photos, 235
Curators tab (studios), 344
custom blocks. See also More Blocks
advantages of, 255
creating, 257-260
explained, 255-256
input parameters, 260-263
limitations, 254
in Snap!, 263
custom sprites
adding, 357
creating, 359-362

D

Data blocks, 35
combining variables and lists, 213
for layers, 251
list of list blocks, 207
list of variable blocks, 202-203
practicing with lists, 209-212
practicing with variables, 204-206
data types, 201-202, 221-222
date/time in mental math example, 179-184
Debug It Studio, 323

debugging. See also troubleshooting

bugs

logical errors, 324

origin of term, 317

types of, 318

with color-coded script execution, 318-319

defined, 318

with Stage monitors, 321-322

decimal fractions, 174

Define block in custom blocks, 256

Delete button

My Stuff page, 40

Scratch Project Editor menu bar, 29

deleting

Backpack items, 105, 339-340

blocks from Scripts pane, 60, 74

broadcasts, 145

comments, 269, 290

projects, 40-42

sprites, 48

dependencies in broadcasts, 145

detaching blocks, 33
deterministic, defined, 174
device drivers, installing, 384-385
digital events, analog events versus, 139-140

Direction control (sprite information panel), 49

Discuss button (home page), 14

discussion forums, 18-19. See also Scratch Forums

discussion threads, 292
displaying Stage monitor variables, 183-184, 203
distance, detecting with Sensing blocks, 194
documentation

comments

adding, 268-269

popularity of projects, 270

purpose of, 267-268

online help, 326-328

prototypes

building via reverse engineering, 276-277

creating with Balsamiq Mockups, 275-276

explained, 274-275

pseudocode, 55-56

comments in, 272-274

defined, 270

in Scratch Forums, 271-273

public documentation, 277-279, 285-286

Dodgeball Challenge

animating sprites in, 404-408

extra features, 399

game navigation code, 401-403

laying out screens, 400-401

main game logic, 408-409

purpose of, 397-398

remixing, 410

sprites in, 404-407

storyboarding, 398

testing, 409-411

timer in, 407

double-clicking

Stage monitors, 86
to test blocks, 210

double-slash comments, 272-274

downloading

assets

manually, 340

practicing with, 341-342

GIMP tool, 348

music, 109

projects, 306-307

Scratch 1.4, 382

Scratch 1.4 projects, 310, 388

sound clips, 110, 307

sprites, 306-307

Draft option (project page), 37, 278

drawing. See also GIMP tool; Pen blocks

with mouse, 193

with Pen tools, 123-133

with sprites, 75-76

sprites, 128-130

drawing tools (GIMP), 350
drum types, 105-108

Duplicate button (Scratch Project Editor menu bar), 29
duplicating. See also cloning:
copying

blocks, 103

sprites, 130, 372

cloning versus, 164

practicing with, 164-166
functions

E

e^\text{ function}, 175
Easy GIF Animator, 244
Edit menu
  Scratch Project Editor, 29
  Sounds pane, 110
editing
  audio, 110-112, 239-242
  photos for Scratch usage, 234-235
educational programming languages, 8-10
Effects menu (Sounds pane), 110, 112
embedded Flash content, 245
encapsulation, 255
environment, resetting, 86
equations, defined, 173
errors. See bugs
event receivers, defined, 138-139
event-driven programming, defined, 137
event-handling code
  broadcast management program, 145-148
  example of, 139
events
  analog events
    digital events versus, 139-140
    PicoBoard sensors, 140-141
    reacting to, 141-143
    defined, 137
Events blocks, 35
  broadcasts
    broadcast management program, 145-148
    creating, 145
explained, 143
removing, 145
use cases, 144
waiting for completion, 145
digital versus analog events, 139-140
list of, 137-138
video sensors
  list of, 149
  popping balloons program, 151-152
exception handling, 89-90, 226
executable files, exporting Snap! projects as, 312-315
expanding comments, 269
Explore area, 21-22
Explore button (home page), 14
exponentiation, 175-177
exporting assets
  manually, 340
  practicing with, 341-342
  Snap! projects, as executable files, 312-315
expressions, defined, 173
Eyedropper tool, 121

F

fair use, 238
Favorites button (project page), 37, 285
Featured Projects (home page), 14
Featured Studios button (home page), 14
file formats
  for audio, 238-239
  for executable files, 313
  for photos, 234
  for Scratch usage, 307-310
  for video, 243
  viewing extensions, 307
File menu
  GIMP tool, 350
  Scratch Project Editor, 29
filtering projects, 284
finding
  new sprites, 48
  sprite location, 31, 49, 52-54
fisheye effect, 86
flaming, 290
Flash, 245
Flash Blocks mode, 321
Flatten command (GIMP), 350
Flickr, 235
Flip buttons (Paint Editor), 51
flipbook animation, 163
floating-point numbers, 174
floor function, 174
.flv file format, 243
fonts, installing, 401
Forever If block, 323
forums. See Scratch Forums
function calls, defined, 256
functions, 126
  defined, 256
  for lists, 209
How can we make this index more useful? Email us at indexes@samspublishing.com
galleries. See studios

generating random numbers, 115

ghost effect, 86

GIF animation

converting video to, 243-244
creating, 358
importing, 357-358

GIF Brewery, 244

gif file format, 234

GIFFun, 244

GIFMaker.me, 244

GifQuickMaker, 358

GIMP tool, 235, 244, 348, 371

animated GIF creation, 358
backdrops
creating, 353-355
scaling images, 356
custom sprites, creating, 359-362
downloading, 348
navigating, 349-350
origin of term, 350
portable version, 355
global variables, 202
creating, 205
local variables versus, 202

Globe button (Scratch Project Editor menu bar), 29

GNU, 350

Google, 294

Google Blockly, 82-84

Google Power Search
(Spencer), 295

graphics. See bitmap graphics; vector graphics

Green Flag (Sprites list), 30

Grow button (Scratch Project Editor menu bar), 29, 72-73

H

hack, defined, 250

handling exceptions, 88, 226

hardware
list of, 383
MaKey MaKey, 389
obtaining, 390
practicing with, 391-393
usage suggestions, 390

PicOBord sensors
blocks for, 387-388
device driver installation, 384-385
practicing with, 385-387
resources from PicoCricket, 387
sensors on, 383-384
smoke testing, 385
Scratch compatibility, 381

Hat blocks, 35
in Control blocks palette, 155
in Events palette, 137
in More Blocks palette, 256
video sensors, 149
visual clues in, 138

Help button (home page), 14

Help page, 20

help system (Tips panel), 326-328
hide <variable> block, 113

Hide block, 70, 86

hiding sprites, 70

high scores lists, creating, 226-228

history of Scratch programming language, 10-14, 300-301

HTML5, 300, 312, 334

hyperlinks, 246

I

Image menu (GIMP), 350

images. See GIF animation; GIMP tool; photos

imgflip, 244

importing
animated GIF, 357-358
assets
manually, 340
practicing with, 341-342

inappropriate content, reporting, 334

Information panel (sprites), 48-50

input parameters in custom blocks, 260-263

installing
device drivers, 384-385
fonts, 401
GIMP tool, 348
Scratch 1.4, 382
Scratch 2.0 Offline Editor, 59, 301

Instructions area (project page), 37, 278, 285

instruments, virtual
drum types, 105-106
list of, 100-101
playing drums, 106-108

integers, 174

integrating Scratch 1.4 and 2.0 projects, 310-311

interactivity with embedded Flash, 245

interface. See navigating

Internet forums. See Scratch Forums

is.gd hyperlinks, 4-5
iterative control structures, 158-162
iterative software development, 409

J-K
JavaScript, 334
Join Scratch button (home page), 14
Joylabz website, 390
joysticks, 383, 392-393
JoyTail Scratch extension, 383
.jpg file format, 234
.json file format, 308-309
Jump custom block example, 257-260
keyboard, animating sprites, 66-67
Kickstarter, 390

L
labels
on multistate buttons, creating, 374-375
text labels, creating, 130-131
latency, 104
layers
blocks, list of, 251
explained, 250-251
practicing with, 252-254
viewing, 251
learning programming languages, 8-10

legal issues
downloading music, 109
fair use, 238
LEGOLego Mindstorms NXT, 383
LEGO WeDo, 383
Lightroom, 235
linked comments, 268
lists
combining with variables, 213
Data blocks, list of, 207
defined, 207
functions, 209
high scores lists, creating, 226-228
practicing with, 209-212
Stage monitors, 208-209
In function, 175
Load More button (My Stuff page), 40
local variables, 202
creating, 206
echoing between sprites, 206-207
global variables versus, 202
location
determining with Sensing blocks, 192-193
sprites
coordinate system, explained, 52-54
finding, 31, 49
resetting, 57
log function, 175
logical errors, 318, 324
logical operators, 171
Logo programming language, 9
Looking blocks, 35
for layers, 251
list of, 81
showing/hiding sprites, 70
looping, 159-163
“Lorum Ipsum” text, 376
Loudness parameter (analog input), 140

Love button (project page), 37, 285

M
Magnification tools (Scripts area), 31
MaKey MaKey, 383, 389
obtaining, 390
practicing with, 391-393
usage suggestions, 390
managing sprites, 48-50
mapping sprites to mouse, 114
mental math, 180-184
menu bar (Scratch Project Editor), 28-29
Mesh, 144
message boards. See Scratch Forums
messages between sprites, 90-92.
See also broadcasts
Messages icon (navigation bar), 288
Messages screen, 18
methods. See functions
Microphone volume (Sounds pane), 110
microphones, 110

How can we make this index more useful? Email us at indexes@samspublishing.com
Microsoft GIF Animator, 244, 358
Microsoft Paint, 127, 371
.mid file format, 239
MIDI virtual instruments
drum types, 105-106
list of, 100-101
playing drums, 106-108
mods, 263, 349
monitoring. See Stage monitors
More Blocks, 35, 255-256. See also custom blocks
Motion blocks, 35, 65. See also animating
motion detection with video sensors
list of sensors, 149
popping balloons program, 151-152
.mouse
animating sprites, 72-74
drawing with, 193
mapping sprites to, 114
.mov file format, 243
movies. See video
moving sprites between rooms, 91-93. See also animating
.mp3 file format, 238
.mp4 file format, 243
.mpg file format, 243
.multimedia
audio
adding, 238
Audacity, 242-243
ing, 110-112, 239-242
fair use, 238
file formats, 238-239
recording, 110-112, 241
Sound Library, 108-110
uploading, 239-242
defined, 233
embedded Flash content, 245
hyperlinks, 245
photos
file formats, 234
processing for Scratch, 234-235
uploading, 235-237
Sensing blocks with, 194-195
video
converting to animated GIF, 243-244
file formats, 243
limitations, 243
Scratch Movie Player Morph, 246
uploading, 244-246
multiple screens
creating, 367-371
in Dodgeball Challenge, 400-401
as Stage backdrops, 367
storyboarding, 366-367
multistate buttons
activating, 376-378
creating, 373-374
labeling, 374-375
as Stage backdrops, 371-372
music, downloading, 109
musical instruments
drum types, 105-106
list of, 100-101
playing drums, 106-108
musical note values, 101-102
My Stuff button (Scratch Project Editor menu bar), 29, 39
My Stuff page, navigating, 39-42
N
naming. See renaming
navigating
.GIMP tool, 349-350
My Stuff page, 39-42
project gallery, 21-23
project page, 36-39, 284-285
Scratch 1.4, 382
Scratch 2.0 Offline Editor, 302-303
Scratch home page, 14
Scratch Project Editor, 27-31
Sounds pane, 109-110
studios, 343-344
nesting
C blocks, 156-157
Reporter and Boolean blocks, 90
netiquette, 287-288
New Block dialog box, 255
New Project button (My Stuff page), 40
new Scratcher accounts
limits on, 218-219
promoting to Scratcher accounts, 219-220
purpose of, 218
Scratcher accounts versus, 218
New Sprite from Camera button (Sprites list), 47
New Studio button (My Stuff page), 40
nondestructive image editing, 355
NOT operator, 179
note values (music), 101-102
Notes and Credits area (project page), 37, 279, 285
numbers, concatenating with strings, 176

obsolete blocks, troubleshooting, 323

Offline Editor. See Scratch 2.0 Offline Editor
online help, 326-328
open source, explained, 348-349
operators, defined, 173
Operators blocks, 35
arguments, 173-174
with Boolean and Control blocks, 157-158
comparison operators, 179
exponentiation, 175-177
list of, 171
practicing with, 179-184
strings in, 178-179
trigonometric functions, 174-175
OR operator, 179, 194
organizing scripts, 321-322
origin point (Sprites list), 30
orphaned assets, troubleshooting, 325

Paint (Windows), 127, 371
Paint Bucket tool, 130
Paint Editor, 50-51, 127
drawing sprites, 128-130
limitations, 347-348
Paint New Sprite button (Sprites list), 46
Paintbrush, 371
painting tools (GIMP), 350
paper joystick, creating, 392-393
Pen blocks, 35
changing values, 122-123
color, 121
creating drawing program, 126-133
drawing with, 75-76
list of, 120-121
practicing with, 123-126
shade, 122
size, 122
permissions, 153
photos
file formats, 234
processing for Scratch, 234-235
uploading, 235-237
Photoshop Lightroom, 235
Picasion, 244, 358
PicoBoard sensors, 140-141, 383
blocks for, 387-388
device driver installation, 384-385
practicing with, 385-387
resources from PicoCricket, 387
sensors on, 383-384
smoke testing, 385
PicoCricket, 385, 387
pictures. See photos
pixels
defined, 53, 122
pen size and, 122
placing sprites on Stage, 54
Play option (Sounds pane), 110
player control
animating sprites, 68-69
asking for input, 87-89
.png file format, 234
Podcasting with Audacity, 243
poll example, 223-226
popping balloons program, 151-152
popularity
of projects, 270
of scripts, 270
Press Start 2P font, 401
previewing projects, 35-39
private variables. See local variables
profile page, updating, 17-18, 294
programming languages, defined, 9
Project Editor. See Scratch Project Editor
project gallery, navigating, 21-23
project page
comments on
advantages of, 286-287
deleting, 290
netiquette, 287-288
practicing with, 288-290
reporting, 288
navigating, 36-39, 284-285
Project tags (project page), 37
projects
adding to studios, 344-345
Android player, 312
announcing, 286
balloon popping project, 151-152
blocks. See blocks
cloning, 69
compiling from Snap!, 312-315

How can we make this index more useful? Email us at indexes@samspublishing.com
copying, 332-333
deleting, 40-42
documentation. See documentation

Dodgeball Challenge
  animating sprites in, 404-408
  extra features, 399
  game navigation code, 401-403
  laying out screens, 400-401
  main game logic, 408-409
  purpose of, 397-398
  remixing, 410
  sprites in, 404-407
  storyboarding, 398
  testing, 409-411
  timer in, 407
downloading
to Offline Editor, 306-307
  Scratch 1.4 projects, 388
filtering, 284
HTML5 player, 312
integrating Scratch 1.4 and 2.0 projects, 310-311
My Stuff page, navigating, 39-42
obtaining permission, 332
popularity of, 270
previewing, 35-39
pseudocode, 55-56
retrieving deleted, 42
remixing, 227, 332
  practicing with, 336-338
  remix tree (project page), 37, 334-336
  with Scratch 2.0 Backpack, 338-342
random numbers
  defined, 174
  generating, 115
raster graphics. See bitmap graphics
reacting to analog events, 141-143
realistic bounce effects, 409
receiving broadcasts, 90-92
Record New Sound (Sounds pane), 109
Record option (Sounds pane), 110
recording audio, 110-112, 241
remixes, defined, 22, 331
Remixes icon (project page), 285
remixing projects, 227, 332
  Dodgeball Challenge, 410
  practicing with, 336-338
  remix tree (project page), 37, 334-336
  with Scratch 2.0 Backpack, 338-342
What the Community Is Remixing feature, 342-343
removing. See deleting

Q-R

question mark icon (Scratch Editor), 327-328
racing game example, 195-198
Rename the Sound Clip (Sounds pane), 109
renaming
broadcasts, 93
costumes, 51
sprites, 49
repeating
actions, 70-71
block stacks, 260
repetitive code, avoiding, 126
replying to topics (in discussion forums), 293
Reporter blocks, 35, 82
in custom blocks, dragging into targets, 263
in Data palette, 203, 207
in Looks palette, 81
in Motion palette, 65
nesting into Boolean blocks, 90
in Sensing palette, 187-189
in Sound palette, 99
Stage monitors, turning on/off, 64-66, 84
username block, 222-223, 228-229
video sensors, 149
reporting
comments, 288
projects, 333-334
requirements
Scratch 2.0, 5-6
Scratch 2.0 Offline Editor, 301
resetting
environment, 86
sprite position, 57
resizing
comments, 269
images, 356
photos, 235
sprites, 72-73, 85, 373
resolution of photos, 235
restoring. See recovering
reusing assets with Scratch 2.0 Backpack, 338-342
reverse engineering, 276-277
right-clicking
Sounds pane, 109
sprites, 46
room metaphor for Scratch, 214, 249
rotating sprites, 86
Rotation Style control (sprite information panel), 49
royalty-free artwork, 367
Run Without Screen Refresh option (custom blocks), 261
runtime errors, 318
Save status (Scratch Project Editor menu bar), 29
saving projects, 31-32
say blocks, think blocks versus, 85
.sb file format, 308
.sb2 file format, 308
scaling images, 356
scope of variables, 202
Scratch 1.4, 11-12
color-coded script execution, 318-319
compatibility with Scratch 2.0, 13, 300-301
downloading projects, 388
.file formats, 307-310
hardware compatibility, 381
installing, 382
integration with Scratch 2.0, 310-311
Mesh, 144
navigating, 382
obsolete blocks from, 323
Single Stepping mode versus Turbo mode, 319-321
source code, 349
Stage monitors, 321-322
Scratch 2.0, 12
compatibility with Scratch 1.4, 13, 300-301
file formats, 307-310
integration with Scratch 1.4, 310-311
new features, 300-301
Scratch 2.0 Offline Editor, 12-13
Adobe AIR, 301
Backpack feature, 105
compatibility with Scratch 1.4, 382
installing, 59, 301
navigating, 302-303
projects
downloading to Offline Editor, 306-307
uploading to Scratch website, 304-306
public documentation, 277-279
system requirements, 301
updating, 303
Scratch button
home page, 14
Scratch 2.0 Offline Editor, 303
Scratch Project Editor, 29
Scratch Cards, 327

How can we make this index more useful? Email us at indexes@samspublishing.com
Scratch Cat, defined, 46. See also sprites
Scratch Community Guidelines, 333-334
Scratch Ed website, 20, 369
Scratch Forums
categories of, 291-292
practicing with, 291-294
pseudocode in, 271-273
reporting inappropriate content, 334
Scratch Movie Player Morph, 245
Scratch programming language blocks. See blocks
compatibility between versions, 13, 300-301
discussion forums, 18-19
file formats, 307-310
Help page, 20
history of, 10-14, 300-301
home page navigation, 14
project gallery, navigating, 21-23
reasons for learning, 10
room metaphor, 214, 249
Scratch Ed website, 20, 369
Scratch Wiki. See Scratch Wiki
Scratch Project Editor
navigating, 27-31
projects. See projects
Scratch Cat, animating, 33
Scratch Resources website, 48, 368-369
Scratch Wiki, 20-21
audio file formats, 239
avoiding spam, 296
Block Plugin syntax, 273
broadcasts, 91
built-in timer, 324
close collision detection, 164
exponentiation, 175
hardware, list of, 383
high scores lists, 227
HTML5 player, 312
mods, 263
new Scratcher accounts, 218
online chat program, 222
pixels, 122
requesting contributor access, 273
Scratch 1.4 source code, 349
Scratch Movie Player Morph, 245
sprite layer information, 251
username block, 228
scratchblocks2 (Block Plugin), 271-274
Scratcher accounts
contacting, 332
new Scratcher accounts versus, 218
requirements for promotion, 219-220
screen captures, 399
screen flicker, reducing, 261
screens, multiple
creating, 367-371
in Dodgeball Challenge, 400-401
as Stage backdrops, 367
storyboarding, 366-367
screenshots, 399
Script assets indicator (project page), 37
scripts
color-coded execution, 318-319
organizing, 321-322
popularity of, 270
Scripts area (Scratch Project Editor), 28, 30-31
cleaning up, 77
copying scripts, 104-105
deleting blocks from, 60, 74
Search button (home page), 14
search engine optimization (SEO), 294-296
Secrets of Mental Math (Benjamin and Shermer), 180
See Examples button (home page), 14
See Inside button
My Stuff page, 40
project page, 37
See Project Page button (Scripts area), 31, 36
Select menu (GIMP), 350
Select tools (GIMP), 350
sending
broadcasts, 90-92
bug reports, 411
Sensing blocks, 35
collision detection, 194
color-based collision detection, 189-192
distance, detecting, 194
list of, 187-189
location, determining, 192-193
with multimedia, 194-195
for PicoBoard sensors, 387-388
practicing with, 195-198
testing conditions, 93
sensors
MaKey MaKey, 383, 389
obtaining, 390
practicing with, 391-393
usage suggestions, 390
PicoBoard sensors, 140-141, 383
blocks for, 387-388
device driver installation, 384-385
list of, 383-384
practicing with, 385-387
resources from PicoCricket, 387
smoke testing, 385
video sensors
list of, 149
popping balloons program, 151-152
Watch Me Move! technology, 149

SEO (search engine optimization), 294-296
SEO Made Easy (Bailyn), 295
set <variable> block, 113
Set Costume Center button (Paint Editor), 51
Set Instrument block, 100-101
shade of Pen tools, 122
shapes of blocks, significance of, 71
Share button
My Stuff page, 40, 283
project page, 36
Scripts area (Scratch Project Editor), 31
Share to Website command (Scratch 2.0 Offline Editor), 305
sharing projects, 283-285
Show block, 70
Show property (sprite information panel), 49
show variable <variable> block, 113
Shrink button (Scratch Project Editor menu bar), 29, 72-73
Shrink tool, 130
Sign In button (home page), 14
sin function, 174
singing, 102-104
Single Stepping mode, Turbo mode versus, 319-321
size
limitations on projects, 239
of Pen tools, 122
Small Stage layout command (Scratch 2.0 Offline Editor), 303
smoke testing, 325-326, 385
Snap!, 349
compiling projects, 312-315
custom blocks, 263
Snapz Pro X, 399
Sol Me Re, 239
songs
choir singing, 102-104
note values, 101-102
Sort By button (My Stuff page), 40
Sound blocks, 35. See also audio
choir singing, 102-104
list of, 99
MIDI virtual instruments
drum types, 105-106
list of, 100-101
playing drums, 106-108
multiple clips, 110
note values, 101-102
sound clips, downloading, 307.
See also assets
Sound Jay, 240
Sound Library, 108-110
Sounds pane, navigating, 109-110
spaghetti code, 321
spam, avoiding, 286, 296
SparkFun, 384
spelling errors, 400
Sprite Library, adding sprites to projects, 47-48
.sprite2 file extension, 359
sprites. See also assets
adding to projects, 47-48
animating, 33
bouncing, 69-72
collision detection, 73-75
complete project, 56-58
with costumes, 50-51
in Dodgeball Challenge, 404-408
with keyboard, 66-67
with mouse, 72-74
by player control, 68-69
repeating actions, 70-71
appearance, changing, 84-86
cloning, 115, 164
collision detection, 164
duplicating versus, 164
practicing with, 164-166
collision detection, 190
color, changing, 86
costumes. See costumes
custom sprites
adding, 357
creating, 359-362
defined, 46
deleting, 48
in Dodgeball Challenge, 404-407
downloading, 306-307
drawing, 128-130
drawing with, 75-76
duplicating, 130, 372
cloning versus, 164
practicing with, 164-166
echoing private variables, 206-207
environment, resetting, 86
finding new, 48
hiding, 70
importing as animated GIF, 357-358
jumping, 257-260
layers, viewing, 251
location
coordinate system, explained, 52-54
finding, 31, 49
resetting, 57
managing, 48-50
mapping to mouse, 114
messages between, 90-92
moving between rooms, 91-93
photos as, 235-237
placing on Stage, 54
recovering deleted, 48
renaming, 49
resizing, 72-73, 85, 373
right-clicking, 46
rotating, 86
singing, 102-104
testing conditions, 72
text labels, creating, 130-131
text sprites, creating, 146
timing problems, troubleshooting, 324-325
viewing information, 48-50

Sprites list (Scratch Project Editor), 28-30, 46-47
sqrt function, 174
squares, creating, 129
Stack blocks, 35, 82
in Control blocks palette, 155
in custom blocks, 256
in Data palette, 203, 207
in Events palette, 137
for layers, 251
in Looks palette, 81
in Motion palette, 65
in Pen palette, 120
in Sensing palette, 187-189
in Sound palette, 99
video sensors, 149
Stage (Scratch Project Editor), 28
Stage area (project page), 37
backdrops
adding, 54-55, 367-368
creating, 353-355
in Dodgeball Challenge, 400-401
multiple screens as, 367
multistate buttons as, 371-372
scaling images, 356
centralizing code on, 321-322
coordinate system, 52-54
text, adding, 370
Stage button (Sprites list), 30
Stage monitors
debugging with, 321-322
displaying variables, 183-184, 203
double-clicking, 86
for lists, 208-209
for PicoBoard sensors, 388
turning on/off, 64-66, 84
Stamp tool, 103
"sticky" topics in Scratch Forums, 292
Stop All block, 323
Stop button (Sprites list), 30
stop motion animation
blocks for, 57-58
defined, 50
Stop option (Sounds pane), 110
Stop Script block, 323
storyboards
building via reverse engineering, 276-277
creating with Balsamiq Mockups, 275-276
defined, 275
for Dodgeball Challenge, 398
for multiple screens, 366-367
string manipulation operators, 171
strings
concatenating with numbers, 176
with variables, 206
in Operators blocks, 178-179
studios
administrative model, 344
creating, 344-345
defined, 311
navigating, 343-344
Studios icon (project page), 285
survey example, 223-226
.svg file format, 234
synchronization, 144, 163
syntax errors, 318, 400
system requirements
Scratch 2.0, 5-6
Scratch 2.0 Offline Editor, 301

T
tag clouds, 279
tags, 279, 285
tan function, 175
tempo, 106-108
testing conditions, 72, 88-89, 93
tests
  beta testing, 410-411
  on Dodgeball Challenge project, 409-411
  by double-clicking blocks, 210
  smoke testing, 325-326, 385
  unit testing, 133, 319, 409
text
  adding to Stage, 370
  concatenating with numbers, 176
  in Operators blocks, 178-179
text labels, creating, 130-131
text sprites, creating, 146
Text tool (GIMP), 350
think blocks, say blocks versus, 85
thumbnail view (sprite information panel), 49
time/date in mental math example, 179-184
Timer parameter (analog input), 140	
timers
  built-in timer, 149
  creating, 182
  in Dodgeball Challenge, 407
  practicing with, 325-326
  troubleshooting, 324-325
Tips button (Scratch Project Editor menu bar), 29
Tips panel (Scratch Editor), 326-327
Title field (Sprites list), 30
title link (My Stuff page), 40
title of projects, changing, 36, 278
topics (in discussion forums)
  creating, 292
  defined, 292
  replying to, 293
  “sticky” topics, 292
Total Views button (project page), 37, 285
transparency
  adding to custom sprites, 359-362
  checkerboard pattern, 362
trash. See deleting
trigonometric functions, 174-175
troubleshooting, See also
debbuging; exception handling
  with breakpoints, 320
  with color-coded script execution, 318-319
  logical errors, 324
  obsolete blocks, 323
  with online help, 326-328
  orphaned assets, 325
  Single Stepping mode versus Turbo mode, 319-321
  with smoke testing, 325-326, 385
  spelling and syntax errors, 400
  with Stage monitors, 321-322
  timing problems, 324-325
  with unit testing, 133, 319
Try It Out button (home page), 14
Turbo mode, Single Stepping mode versus, 319-321
Turbo Mode command (Scratch 2.0 Offline Editor), 303
undeleting. See recovering
Undo button (Paint Editor), 51
Undo option (Sounds pane), 109
unit testing, 133, 319, 409
Unshare link (My Stuff page), 284
uploading
  assets
    manually, 340
    practicing with, 341-342
  audio, 239-242
  photos, 235-237
  projects, 304-306
  video, 244-246
URL shortening services, 5
user accounts
  creating, 15-16
  profile page, updating, 17-18, 294
user control
  animating sprites, 68-69
  asking for input, 87-89
user interface. See navigating
User menu (Scratch Project Editor menu bar), 29
user ranks, 220
username Reporter block, 222-223, 228-229
Using Gimp (Pyles), 348

How can we make this index more useful? Email us at indexes@samspublishing.com
<variable> Reporter block, 113

variables
arrays, 207
cloud variables
creating, 220-221
data types allowed, 221-222
practicing with high scores
lists, 226-228
practicing with surveys, 223-226
username Reporter block, 222-223, 228-229
combining with lists, 213
concatenating strings
with, 206
Contact! game, 113-115
creating, 112-113
Data blocks, list of, 202-203
defined, 201-202
lists. See lists
local variables, echoing
between sprites, 206-207
practicing with, 204-206
scope, 202
in Stage monitors, displaying,
183-184, 203

vector graphics
bitmap graphics versus, 351
converting to bitmap graphics,
352
video. See also assets
converting to animated GIF,
243-244
file formats, 243
limitations, 243
Scratch Movie Player
Morph, 245
uploading, 244-246

Video motion parameter (analog
input), 140

video sensors
list of, 149
popping balloons program,
151-152
Watch Me Move! technology, 149

View filter (My Stuff page), 40

viewing
file extensions, 307
sprite information, 48-50
sprite layers, 251

virtual instruments
drum types, 105-106
list of, 100-101
playing drums, 106-108

volume
changing, 103
reacting to, 141-143

web pages, refreshing, 345

webcams
Camera Capture tool, 236
Watch Me Move! technology, 149

What the Community Is Remixing
feature, 342-343

wiki, defined, 20

Windows Paint, 127, 371

wireframes
building via reverse engineering, 276-277
creating with Balsamiq
Mockups, 275-276
defined, 275
.wma file format, 239

wrap blocks. See C blocks

X-Z

xy-grid backdrop, 192

.zip file format, 308