Learn by doing: less theory, more results

Scratch 2.0

Second Edition

Create digital stories, games, art, and animations through six unique projects

Beginner's Guide

Michael Badger

[PACKT PUBLISHING]

Scratch 2.0 Beginner's Guide
Second Edition

Create digital stories, games, art, and animations through six unique projects

Michael Badger

PACKT PUBLISHING

BIRMINGHAM - MUMBAI

Scratch 2.0 Beginner's Guide

Second Edition

First Published: July 2009

Second Edition: April 2014

Production Reference: 1080414

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

ISBN 978-1-78216-072-4

`www.packtpub.com`

Cover Image by Ross Manges (`ross@rossmanges.net`)

Credits

Author

Michael Badger

Reviewers

Samyak Bhuta

Manuel Menezes de Sequeira

Franklin Webber

Acquisition Editor

Joanne Fitzpatrick

Content Development Editor

Dayan Hyames

Technical Editors

Shubhangi Dhamgaye

Shweta Pant

Mrunmayee Patil

Aman Preet Singh

Copy Editors

Sarang Chari

Brandt D'Mello

Mradula Hegde

Project Coordinator

Binny K. Babu

Proofreaders

Simran Bhogal

Maria Gould

Ameesha Green

Paul Hindle

Indexer

Mehreen Deshmukh

Production Coordinator

Nitesh Thakur

Cover Work

Nitesh Thakur

About the Author

Michael Badger is a writer and technical communicator who has worked in a range of technical roles, including support, automated software testing, and project management. He has authored several books for Packt Publishing, including *Scratch 1.4 Beginner's Guide*. He also authors a regular Scratch column for Raspberry Pi Geek Magazine, which focuses on Scratch 1.4.

I'd like to thank the team at Packt Publishing for putting up with me and helping me make this revision the best it could be. My loving wife Christie and son Cameron also deserve credit for allowing me the flexibility to complete this book.

About the Reviewers

Samyak Bhuta is fascinated by art and technology and is always excited when they both meet. He is a software architect by profession with over a decade of experience. He started programming in his childhood with GWBasic and quickly moved over to QBasic. Professionally, he has worked on Java, JavaScript, Python, and PHP. He enjoys coding user interfaces as well as working on backend programming. Samyak believes in the open source philosophy and has been active in his local community. He loves to eat dal bati, an Indian dish, and has dreams to become a flautist.

I would like to thank Packt Publishing for keeping patience when I couldn't submit my reviews on time.

Manuel Menezes de Sequeira has been teaching programming since 1995. He started teaching programming using C, then moved to C++, and later to Java. Nowadays, in his lectures, he usually starts programming with Scratch and Snap!, and then moves on to text-based languages such as Java. Manuel teaches at the Universidade Europeia | Laureate International Universities in Lisbon, Portugal, where he also champions in CoderDojo LX, the Lisbon-based CoderDojo, where children can learn to program for free while having fun. He lives in Lisbon, Portugal, and has been involved for a few years in the translation of Scratch, SNAP!, and other projects to Portuguese.

Franklin Webber is a software professional whose professional experience comes from a testing background where he sought to automate himself out of a job. A college teaching assistant once told Frank that he was a great software developer and a terrible computer scientist, and that the software he wrote cared more for the user experience than the size of its Big O Notation. As a software developer, he became the resident generalist who was always willing to step up to learn new technologies. He now spends most of his time teaching software design to students, both young and old.

www.PacktPub.com

Support files, eBooks, discount offers, and more

You might want to visit `www.PacktPub.com` for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

`http://PacktLib.PacktPub.com`

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why Subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free Access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface

This book demystifies Scratch programming through a variety of projects. The book assumes that you have no programming experience when you begin reading, but by the time you reach the last page, you will be ready to explore your own projects and help other people with Scratch.

The projects start with simpler concepts and get progressively more complicated in terms of programming concepts and design. You will learn how to make multiple-scene stories, think through the logic of a fast-paced arcade game called *Breakout*, interact with a snarky fortune teller, and more. The book's projects tend to demonstrate a programming concept first and then discuss the concept in more detail.

You will receive a balanced introduction to Scratch and universal programming concepts as you create digital stores, animations, and games. With a firm grasp on the fundamentals, you'll be ready to take on more advanced topics and projects.

What this book covers

Chapter 1, *Welcome to Scratch 2.0*, introduces Scratch and the various types of projects covered in the book.

Chapter 2, *A Quick Start Guide to Scratch*, takes us on a tour of the online Scratch community. In this chapter, we will create our first Scratch animation while learning basic programming concepts such as loops.

Chapter 3, *Creating an Animated Birthday Card*, will guide us through how to use Scratch's built-in paint editor to draw bitmap and vector images. To create the card, we will learn important programming concepts such as project initialization, object naming, and event coordination.

Chapter 4, *Creating a Scratch Story Book*, will guide us through how to build a joke book and coordinate scene changes as a way to navigate through the book. The chapter introduces sound and coordinates as a way to move sprites.

Chapter 5, *Creating a Multimedia Slideshow*, will guide us through how to create a personalized slideshow by uploading files from our computer. We will also work on resizing images and recording slide narrations that can be played on demand.

Chapter 6, *Making an Arcade Game – Breakout (Part I)*, remixes the classic Pong game into our own brick-busting version called Breakout. We'll clone sprites, estimate direction, and create custom variables to develop the framework of the game.

Chapter 7, *Programming a Challenging Gameplay – Breakout (Part II)*, builds on our Breakout game from the previous chapter. Here, we make the gameplay more challenging by programming the ball speed and reducing the paddle size based on the gameplay. Important concepts include custom procedures, Boolean values, and cloud data.

Chapter 8, *Chatting with a Fortune Teller*, deals with our game of fortune, where a fortune teller will provide a random fortune in response to the user's typed question. We will work with lists, track intervals with mod, and split words apart to identify individual words.

Chapter 9, *Turning Geometric Patterns into Art Using the Pen Tool*, combines all the programming concepts we've learned so far to draw art using simple math equations, polygons, and string art. The projects will show you how to take user-defined values and turn them into shapes. This chapter also explains how to apply color and shades to Scratch projects.

Appendix A, *Connecting a PicoBoard to Scratch 1.4*, emphasizes on projects that use a computer's webcam and the PicoBoard, which is an add-on device capable of running on Scratch 1.4 on the Raspberry Pi. The PicoBoard project incorporates an experiment that measures the resistance of warming water using a thermistor and generates graphs for it.

What you need for this book

To create projects using the Scratch 2 project editor, you need a relatively recent web browser (Chrome 7 or later, Firefox 4 or later, or Internet Explorer 7 or later) with Adobe Flash Player Version 10.2 or later installed. Scratch 2 is designed to support a screen resolution of 1024 x 768 or larger. If your computer doesn't meet these requirements, you can try downloading and installing Scratch 1.4, which you can still use to share projects to the Scratch 2 website.

An offline Scratch 2 editor is also available. You can also still use Scratch 1.4. Note that you can have both Scratch 1.4 and 2 on your computer.

The software to download are as follows:

- The Scratch 2 offline editor can be downloaded from the following link: `http://scratch.mit.edu/scratch2download`
- The Scratch 1.4 editor can be downloaded from the following link: `http://scratch.mit.edu/scratch_1.4`

Who this book is for

The author approaches the content in this book with the belief that we are all teachers and that you are reading this book not only because you want to learn, but also because you want to share your knowledge with others. Motivated students can pick up this book and teach themselves how to program because the book takes a simple, strategic, and structured approach to learning Scratch.

Parents can grasp the fundamentals so that they can guide their children through introductory Scratch programming exercises. It's therefore perfect for homeschool families. Teachers of all disciplines from Computer Science to English can also quickly get up to speed with Scratch and adapt the projects for use in the classroom.

Conventions

In this book, you will find several headings that appear frequently.

To give clear instructions of how to complete a procedure or task, we use:

Time for action – heading

1. Action 1
2. Action 2
3. Action 3

Instructions often need some extra explanation so that they make sense, so they are followed with:

What just happened?

This heading explains the working of tasks or instructions that you have just completed.

You will also find some other learning aids in the book, including:

Pop quiz – heading

These are short multiple-choice questions intended to help you test your own understanding.

Have a go hero – heading

These practical challenges give you ideas for experimenting with what you have learned.

You will also find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "Conditional statements are used to check whether a statement is `true` or `false`. For example, `if 4 > 0` is a conditional statement."

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "The top of the page contains the **Create**, **Explore**, and **Discuss** links."

Warnings or important notes appear in a box like this.

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title through the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Downloading the color images of this book

We also provide you a PDF file that has color images of the screenshots/diagrams used in this book. The color images will help you better understand the changes in the output. You can download this file from: `https://www.packtpub.com/sites/default/files/downloads/0724OT_coloredimages.pdf`.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website, or added to any list of existing errata, under the Errata section of that title.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
Welcome to Scratch 2.0

I assume you're reading this book because you want to learn how to create interactive stories, animations, and games using Scratch, or you want to learn Scratch so that you can teach someone else how to program. It matters not whether your classroom is in a middle school, a home school environment, an after-school workshop, or a weekend coding project with your son or daughter. We are all teachers. That's the perspective of this book. You'll learn how to create projects using Scratch so that you can teach someone else, but no programming knowledge is expected.

In this chapter, we will:

- Review what Scratch is and how we can use it
- Learn more about the types of projects we will create in this book
- Explore an example project from the Scratch website and review the project editor

Whether you're 8 or 80, the Scratch programming language provides a beginner-friendly computer programming environment that enables you to create digital projects. Success with Scratch comes quickly. You won't find any quirky syntax to learn, and you won't make any typing mistakes that prevent your program from running.

Creating a project in Scratch is as easy as snapping the color-coded blocks together. This environment allows us to see the positive results quickly. In addition to this, Scratch helps turn passive users into creators.

You'll find comfort in Scratch's building-block approach to create animations, games, and stories. After using Scratch, programming will make sense. It will seem easy. It will bring a smile to your face, and you'll be able to cope with technical concepts in the future.

About Scratch

Mitch Resnick and the Lifelong Kindergarten Group at the Massachusetts Institute of Technology (MIT) in the Media Laboratory developed Scratch as a teaching language primarily for 8 – 16 year olds, but there's nothing stopping the rest of us from enjoying the Scratch experience and sharpening our creative minds.

Encouraging everyone to think programmatically

The natural reaction of people is to see Scratch as a means of teaching computer science and integrating it into classrooms of all levels. There are teachers who use Scratch across a variety of subjects as seen on the ScratchEd site. The ScratchEd site caters to the educational community and aggregates a lot of Scratch resources, including lesson plans and tips. However, the approach and thoroughness of the included material varies greatly. You can check out ScratchEd at `http://scratched.media.mit.edu/`.

While writing this book, I did not set out to write a computer science textbook. It's quite simply a tutorial for people who want to learn how to use Scratch to create stories, animations, games, or art. It primarily addresses the parents, home school families, and teachers who may not be programmers themselves but want a fun way to help their children become more digitally literate. Everyone, however, can use this tutorial to learn Scratch, and many young students have worked through the Version 1.4 of Scratch of this book. I expect young scratchers will be more than capable of working through the projects in this edition.

My underlying belief is that knowing how to program can benefit everyone, but not everyone needs to be a programmer. The mental work required to create a program inherently develops an understanding of how computers work, sharpens our critical thinking skills, and gives us lots of practice at solving problems.

There's also an increasingly popular idea that sometimes we want to create applications for personal use. Of course, system administrators have always created custom scripts to help automate repetitive tasks. However, modern applications such as Scratch or the **MIT App Inventor** (originally developed by Google) make it incredibly easy to create programs for personal use or with the intention of sharing it with a small group of friends, which counters the perspective that learning to program is synonymous with wanting to be a professional programmer or an application developer.

When you have a little bit of programming knowledge, you'll approach non-programming problems in a different way. For example, I've used programming as a marketer to manage search engine optimization and keyword research on business websites. I've also used my programming knowledge to write automated software tests.

Bottom line, programming becomes a tool in your problem-solving toolbox. This is the key to understanding how I approach this Scratch tutorial. I want the computer scientists to come along for the ride, but I'm catering to a broader audience.

Sample Scratch uses

I couldn't begin to suggest every possible way for you to use Scratch; that's why we have an imagination. However, here are a few ideas to get you started:

- Use Scratch to teach yourself or your students how to program. That's the obvious one.
- Use Scratch to demonstrate Math concepts. Scratch can also demonstrate the *x* and *y* coordinate system in an interactive way.
- Use Scratch to inspire your kids to read and write. Find a story and animate each scene or encourage them to animate the story. Turn their haiku into a Scratch project.
- Have a child who only wants to play video games? Make a deal. Your child can only play the games he creates or remixes with Scratch.

As you work through the examples in this book, write down your project ideas no matter how hard, easy, obvious, or silly they seem. The next one might be your best idea yet.

Computational thinking

Learning: we do it for life. We should help our children develop skills that will help them keep learning and solving problems in an increasingly digital environment. Using Scratch, we will learn how to design, think, collaborate, communicate, analyze, and program in a computer language.

You may frame the Scratch approach as computational thinking. According to Wikipedia:

> *Computational thinking is a problem solving method that uses computer science techniques. The term computational thinking was first used by Seymour Papert in 1996.*

By the time we make our cat dance for the first time, we'll forget all about the academic research and theories behind Scratch. Instead, we'll focus on having fun and creating the next project.