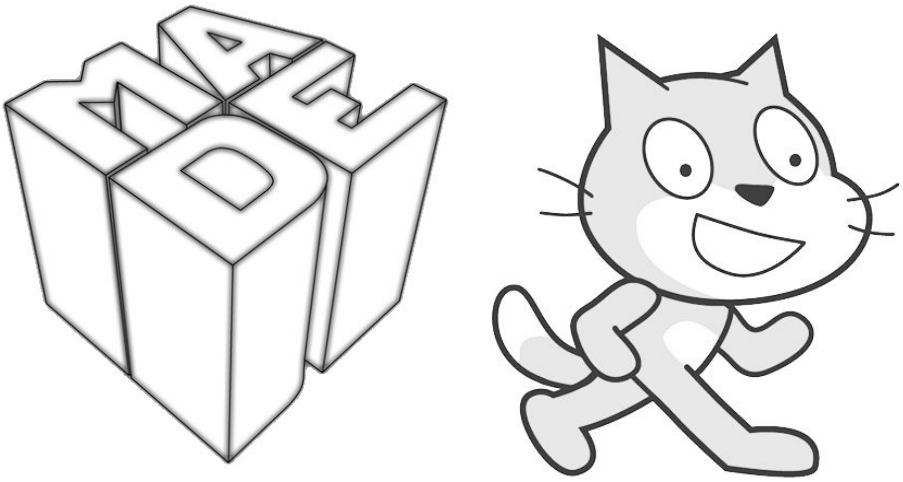
Scratch Class Handbook



The Museum of Art and Digital Entertainment
34000 Broadway, Oakland, CA 94611

The MADE is the only all-playable video game museum in the world. We were the first dedicated open to the public video game museum in the United States. Our collection houses over 5,300 playable games. The MADE is a 501c3 nonprofit dedicated to the preservation of video game history, and to educating the public on how video games are created. Our goal is to inspire the next generation of game developers.

Edited by Al Sweigart al@inventwithpython.com
Version 1, Last modified 2018/11/7

This document contains information relevant to the Saturday Scratch Programming class at the Museum of Digital Art and Entertainment (the MADE), but it could be relevant for anyone teaching a programming class or after school coding club.

This document covers Scratch 2. Version 3 of Scratch will be released January 1, 2019 and while it has some new features, this document gives advice on teaching Scratch, rather than Scratch programming itself, and will still be relevant.

Setting Up the Computer Lab

Whether you have a laptop cart or students are bringing in their own computers, go through the following steps to set up the computer for Scratch programming.

- Download and install the offline Scratch editor from https://scratch.mit.edu/download/. (You must also download and install Adobe Air from that page before installing Scratch.) The offline editor looks identical to the browser-based editor, but will continue working if you lose wifi, though you can still upload projects to a Scratch account. (On the other hand, using the browser-based will automatically save projects in case the browser crashes, while you must remember to manually save using the offline editor.)
- Go to https://scratch.mit.edu and set up a Scratch account. These accounts are free and only require an email address (which is used only for password recovery.) Students can use their own Scratch accounts, but you should set up Scratch accounts beforehand for students to use. You can print and hand out small slips of paper with the Scratch URL, username, and password on them for students to use. (You can use the same email address when signing up for multiple accounts.)
- Alternatively, you can set up a single account for the class and share the password (however, this does mean any student can change the password and lock everyone else out of the account.) Change the password (by signing in, clicking your username in the upper right, then go to Account Settings, then click the Password tab) to something simple before the start of the class (i.e. “applejack”) so that students can use it to upload their projects, and then change it back to a secret, strong password (i.e. “6Hq3CcVeiyf”) at the end of class.

Pre-Class Checklist

Follow these steps before the start of each class:

- Open Scratch 2 offline editor and **maximize the window** (students forget to maximize the window and continue to work in a small window).
- Turn off volume on each computer. (Sounds are distracting.)
- Make sure each computer has a mouse and mousepad. Students may be used to the touchpad from using the touch screen, but the mouse is a much more effective tool for dragging code blocks in the editor.

- Have pen & notepad to write down improvement class ideas as you have them. (You'll forget them by the end of class.) If you write these down on your phone, it looks like you're goofing off on your phone to the students who will probably want to do the same.
- Download any images for the project to the desktop. It's too time-consuming to have students download images for projects themselves. You don't want students searching for images to use on Google Image Search since the adult filter is not always 100% effective. If possible, stick to the images in the Scratch sprite and backdrop library.
- If you are projecting from your laptop, set max zoom for Scratch editor on the projector computer. Click the "plus magnifying glass" in the lower right corner of the Scratch editor to zoom in. This increases the block size and makes it easier to see from the back of the class. (Although a self-paced classes are better than lecture classes. See the "Class Style" section in this document.)
- On Windows, go to the Control Panel or Mouse Settings, set mouse pointer scheme to "Windows Aero (extra large)" for better visibility. This makes the mouse cursors extra large and easier to see.
- From the back of the room, look at the projector to remind yourself how far away it looks for students.
- Make sure each computer's screen saver/auto log off is disabled. These tend to interrupt the student's work if they haven't moved the mouse in a few minutes.

Start of Class Checklist

Follow these steps at the start of each class.

- Tell parents that they can sit with their kids and work along with them, or wait in the lobby.
- Have the students introduce themselves by saying their name and favorite video game. As an example, teachers should introduce themselves first. Make the effort to memorize names, or write them down. It's hard to get students attention if you don't know their name!
- Demo the completed project that they'll make.

Teaching Tips

Keep these tips in mind while teaching the class.

- KEEP YOUR HANDS OFF THE STUDENT'S KEYBOARD AND MOUSE. Point to the screen where they should click or the keyboard keys they should press. Don't take over the computer and do the task for them. This is slower, but doing the task for them means they won't get the experience of coding. They learn from doing, not watching you do it.
- Remember to have the students save their work every 15 minutes or so. Make an announcement. The filename format should be "<student name> - <project name>" so

that there aren't multiple projects with the same name on the computer. (This doesn't apply if they're using the browser-based Scratch editor instead of the offline editor.)

- If you are typing on your computer, be sure to face towards the class. Don't talk at the front wall. If you are on one of the desktops, turn the monitor & keyboard around to face the class.
- If the class is loud and you need to get people's attention, a classic teacher trick is to say "if you can hear my voice, clap once. <You clap once too.> If you can hear my voice, clap twice. <You clap twice too.>" Try to only use this trick once per class.
- The point of this class is **not** so that the students learn to program but that 1) they think programming is cool and 2) they think programming is something they're capable of.
- **ENCOURAGE and PRAISE them.** Say "Good job", "Yeah, you got it", & "Nice drawing".
- Students have a hard time seeing the projector from the back. Make sure the editor zoom is at max zoom and stays that way.
- Keep in mind that stuff at the top of the projector screen is easy to see; stuff at the bottom is hard to see.
- Don't block the projector. Students should be able to always see you **and** the screen at all times.
- Uninstall the 1.4 offline editor if you find it on a machine. This is the old version of Scratch and there's no reason to have it. Students may be confused if they launch the wrong version of Scratch.

Class Styles

Originally, Scratch classes at the MADE were 90 minutes long and an instructor on a projector would guide students through a small project. By the end of the class, they would have a small video game or other program that they made. The downside of this format is that the class can only move as fast as the slowest student, leaving other students open to distraction. If they had any bugs in their code, it could take a while to discover them.

As a better alternative, we created web-based handouts with self-paced instructions for each step of the project. Prototypes of these are available at https://inventwithscratch.com/tutorials/. As students worked through the steps, instructors could float and offer help as needed. The downside is that it takes a while to create these handouts.

The best format for these handouts were short, looped, animated gifs demonstrating each of these steps. These animated gifs were made with the LICEcap program for Windows, and placed into small web pages. If you don't know how to create web pages, they could also be placed into PowerPoint slides. Unlike static images, they show the exact process of adding blocks or clicking in the editor. Animated gifs avoid the problem of videos in that students don't have to constantly pause and rewind to review instructions. Animated gifs are also better than written instructions: "A picture is worth a thousand words". However, being animated, these cannot be printed out on paper.

If you are interested in creating more of these projects to distribute to other instructors, please contact Al Sweigart at al@inventwithpython.com.

Project Tips

When choosing a programming project for the class, keep in mind the following tips.

- **Keep in mind your project will take longer to make than you think.** The point of this class is to make Scratch familiar and seem cool; they don't actually have to learn coding.
- **Video tutorials of various Scratch projects can be found at https://inventwithscratch.com.** You can also find Al Sweigart's Scratch projects at https://scratch.mit.edu/users/AlSweigart/.
- **The book, "Scratch Programming Playground" has several projects and is free to download from https://inventwithscratch.com.**
- Avoid games that require a camera view (i.e. camerax and cameray variables). This is a complex programming concept for students.
- Programs should have at most three or four variables. Requiring more than this may be a sign that your program is too complicated.
- If you have a sprite named "foo", don't use "foo" for variable or broadcast message names.
- Professional software developers might design their code to be "generalized" and "elegant", but this means it's probably too hard or abstract to understand. It's okay to have copy/pasted code because it'll be easier to understand.
- Code that uses indirection/abstract concepts (indexes to lists, magic numbers, etc) is hard to understand.
- Be a part of the vowel generation: Use variable names like "string compare", not "strcmp".
- Don't use literal guns in the game. Alternatives: bow & arrow, "energy balls", lightning bolts, large cannons or catapults that shoot bowling balls.
- If you have shooting targets, make sure they aren't living things. Have the students shoot inanimate objects (asteroids, targets, apples, balloons, stars, sprites from the "Things" category in the sprite library) instead.
- Stick to the images in the Sprite Library and Backdrop Library. If the students have to draw their own sprite, make sure it is a simple picture. Drawing sprites eats up a lot of time, the size of the sprite can be too big/small, or the costume center is off, etc.
- Avoid changing the costume center if necessary. (In fact, avoid needing the paint editor as much as possible.)
- **Never** have the students find their own images on Google Image Search, even if the "filter adult images" option is enabled on the browser. The filter isn't 100% effective. Even if this isn't an issue, students will spend a lot of time searching and choosing images.
- If using non-library images, don't have the student download the image. Download it to the desktop at the start of the class.

- For project ideas, use mini-games that appear in real video games. Or take an item from a Zelda game like the bow, hookshot, lantern, or boomerang and make a game that uses just that item.
- Think about what the “Minimum Viable Product” would be for your project: Have the students design a basic game that works, and then let them add additional features later. This way, even if the class runs behind, they’ll still end with a playable game instead of an unfinished project.

Common Coding Mistakes

These are common mistakes that students make when coding for themselves or even copying code from someone else.

- Make sure “For all sprites” or “For this sprite only” is properly selected. On the Stage, “For this sprite only” variables will have the sprite name in front of the variable name.
- If the “goto x y” code seems to make the sprite move somewhere else, the costume center might be off.
- Students use “set” instead of “change” for the “set/change x” or “set/change variable” or “set/change color effect” blocks.
- Students mix up the <, =, and > blocks.
- Students will add code to the wrong sprite. Check which sprite is selected when looking at their code.
- Students use “broadcast” instead of “broadcast and wait”.
- When using <, =, >, make sure whitespace isn’t accidentally entered in the text fields.

Post Class Checklist

Follow these steps at the end of each class.

- Go over the bullet list of what different features they made. Say something about how “it doesn’t take a lot to make a game that is fun”.
- Thank students by name for coming to the class.
- Hand out post-class cards (preferably to parents if they’re there.) These are cards that tell the students how to find the Scratch website and show their games to other people. (This is an important part; letting students show off their work makes them more enthusiastic to continue it.)
- If you need help getting students out of the class, ask them if they’ve saved their project, then tell them they can shut down the Scratch editor and close the laptop lid. This will keep them from lagging behind.
- Record the number of students, their names, the instructors, the project, and other details in a class spreadsheet. This information is useful for demonstrating the efficacy of your class to fund raisers and donors.