Programming workshop and computational thinking with Scratch

INDEX

Introduction

The workshop is part of the Activities for the 1st mobility in Puertollano (8th-15th April 2018), is carried out in three sessions and a last one being the presentation of elaborated projects.

The central idea of the workshop is to relate the central axis of this Erasmus +, "cities" with a form of work and dealing with ideas using programming tools and computational thinking. Proposed as tool Scratch as a free, multiplatform, suitable for any age and widespread tool with a very active and collaborative community that offers millions of networked projects.

Scratch is a programming language created by MIT and specially designed so that everyone can start in the world of programming. It's used to create interactive stories, games, animations and hot-pluggable creations with hardware; In addition to facilitating the dissemination of the final creations with others on cloud.

The name comes from the word: "Scratching" meaning in programming languages, those pieces of code that can be reused, easily combined and adapted to new uses.

What is Scratch?

Scratch is both an application that you can download to your computer (available for various operating systems: Windows, Mac, Ubuntu, Sugar) and a web application that you can run from your browser.

In both cases, on the one hand we have a series of objects or "sprites" (in the nomenclature used for Scratch) and on the other hand, a series of actions and behaviors that we can combine to get objects to react or to act in a certain way.

One of the most interesting things from Scratch is that those actions or behaviors are puzzle shaped and our mission as programmers will be cooking those pieces to obtain a certain action or behavior. So in essence programming becomes something like solving a puzzle, which eliminates one of the main barriers that newcomers have in the world of programming: the arid and complex aspect of the programming environments. Making the process of programming in something akin to a game.

Actions and behaviors are divided into categories and these are:

- Movement: to move and rotate an object around the screen.
- Appearance: Changing the display of the object: the background, making it big or small, etc.
- Sound: Playing audio streams.
- Pencil: Drawing by controlling the size of the brush color and shade of the same.
- Data: Creating variables and their assignment in the program.
- Events: Event handlers that "trigger" certain actions in a block.
- Control: Conditional: if-else, "forever", "repeat", and "stop".
- Sensors: Objects or "sprites" can interact with the environment or with elements created by the user such as a lego robot, for example.
- Operators: mathematical operators, random number generators, position operates...
- More blocks: Own blocks and controllers of external devices.

What is it useful for?

- Development of processes of thoughts and mental abilities in students
- It's perfect to get into programming
- Allows sharing projects via the web, they can be downloaded and used. They can be downloaded and used by other people

Why Scratch?

Programming is the new language that all we need to know if we want to have a good understanding of today's world and above all of which comes as well as good career opportunities. It is estimated that about 50% of the jobs that we know today will disappear and largely will be replaced by the software and the robotics industry. Thus Scratch becomes a great tool to understand the concepts and logic of programming. In addition it does addressing your learning from a playful perspective to avoid the initial rejection that, for many people, represent the most classic programming environments.

Advantages for the development of the student

If there is an environment where Scratch is especially indicated, it's in teaching of programming, for that playful component we have discussed before. We could group a series of advantages which use provides students:

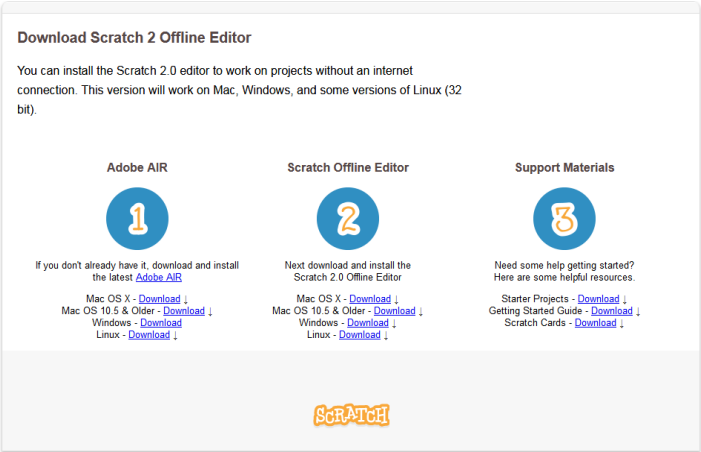
- The development of logical thinking.
- The development of problem-solving methods,
- The development of the habit of doing self-diagnosis with respect to his work
- The development of the ability to call into question the ideas of one self
- Having the chance to achieve complex results starting from simple ideas
- Working each at their own pace according to their own competencies
- Learning and assume mathematical concepts: coordinates, variables, algorithms, randomness,
- Learning the basics of programming
- Using various means: sound, image, text, graphic...
- Enabling collaborative learning through the exchange of knowledge

Conclusions

Scratch is free, cross-platform, simple and aimed at the introduction of the students (and anyone who wants to) in the world of programming.

Start, is as simple as downloading the program and start playing by changing the individual objects that come with the program with the actions and behaviors that are available.

Download and installation of Scratch 2.0

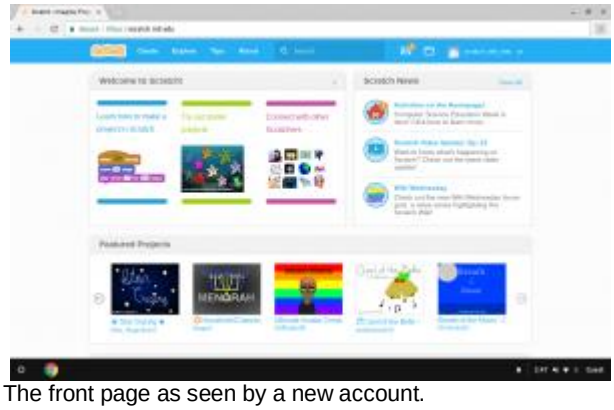


Though we can work online through an account shared with Scratch, it is more comfortable and efficient for work in the classroom or at home to download version 2.0 on the computer and once finished each one's tasks, keep them in the shared account.

Download 2.0 Scratch

Account Creation

A screenshot of the account creation page.



The front page as seen by a new account.

To create an account, go to the front page and click "Join Scratch" in the top right hand corner. If you already have an account, it will first ask you if you want to retrieve your password. When you get to the signup page, enter the following data:

- **Username**

Use one that reflects you in some way, but does not give away any personal information like your full name or address. Do not try to impersonate other users, and do not use an inappropriate username either. If you do, your account might be blocked or deleted. Also keep in mind that you will not be able to change your username in the future without making a new account.

- **Password**

A secure password should have both letters and numbers. To make it stronger, you may use symbols or a upper-lowercase combination, but make sure you can remember it! Always write your password down somewhere safe in case you forget it. But make sure not to leave a piece of paper with your password just lying around! Also remember not to write 'Scratch password' or similar on it, as that would be risky. To change the password after you've logged in, visit the Account Settings page. Scratch requires that your password be at least six characters when you create your account. A good way to make a secure password is to take a word you can easily remember, shorten it by removing vowels, replace a few of the letters with numbers, and add in punctuation. This way, your password is extremely hard to guess, but easy to remember.

- Email Address

An existing email address is required so that the email address can be confirmed and the account is confirmed. If an account is not confirmed, it will not be able to comment or share projects.

- Other data

Some of this data is optional, although your birthday, gender, and country are still required in order to create a Scratch account.

Then click Sign Up. You are now ready to post projects, comments, and use the forums.

Development of the workshop

We will divide the different times of the event to tackle various steps:

Step 1: the beginning of the workshop

In this first step we make an introduction and explanation of what are we going to do in the workshop. Here are some ideas:

- ✓ We will work in joint groups of students and we must organize into teams.
- ✓ We talk about the importance of programming for students. Maybe this TED talk by Mitch Resnick will help us.
- ✓ We can start doing an introduction to Scratch to show the interface so those who do not know, can start to take their first steps. This video could help you Scratch 2.0 Tutorial 1.
- ✓ We can deliver a few sheets cross-linked with the dimensions of the stage to allow groups to design and plan their projects. We can show some 'city'-related projects from Scratch so that attendees can see the great possibilities that the tool has and don't be fooled by the first impression.

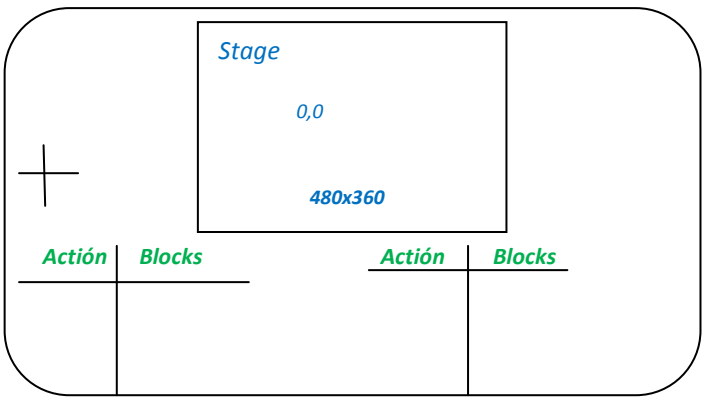
STEP 2: Programming activities

In this step, the groups need to think which project they want to program during the event. It is convenient to establish a time for the exchange of ideas, which can be addressed to:

- presentations on cities or places.
- games and activities about cities.
- interactive activities about the city.

During the two hours they will be spending time on the programming of the project with all the graphics and sound resources for programming.

To help in this task they should make use of the sheets that show them the steps to give by way of traveling. In this way, each group can go at their own pace. We leave a totally prepared card for you to use at the event:



STEP 3: Preparation of the presentation.

Finishing of the project groups and presentation tasks. Programming tasks are completed, the various activities of the project are solved and everything is organized for the presentation.

After the event

There will surely be groups who have not had the time to finish the project and others who would like to return to retake it at home. Hence the usefulness of using a shared account from Scratch from where everyone can return to the original project and modify it according to their interests and abilities.

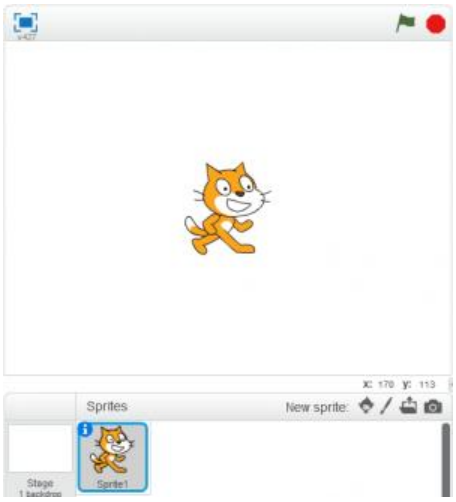
The following pages include two help attachments.

ATTACHMENT I

Getting Started with Scratch

Though Scratch is designed for people totally new to programming, it can still be a challenging program to wield. Consisting of a script, paint, and sound editor, it has complexity to its structure. This tutorial is aimed at individuals completely new to all programming and looking to understand the concepts of Scratch.

Projects



The stage and sprites pane.

Projects are animations, stories, Art (could be made in Pen Code), games – just about everything made in Scratch! The Scratch program is used to make Scratch projects, which can be shared to the world by the Scratch community. In other terms, a project is the created application that is run.

Offline Editor vs. Online Editor

Scratch offers two editors, an offline one, and an online one. Both are very similar but have minor differences. The online editor can be accessed by clicking Create tab or by clicking here. The offline editor can be downloaded here. The current version of Scratch is Scratch 2.0.

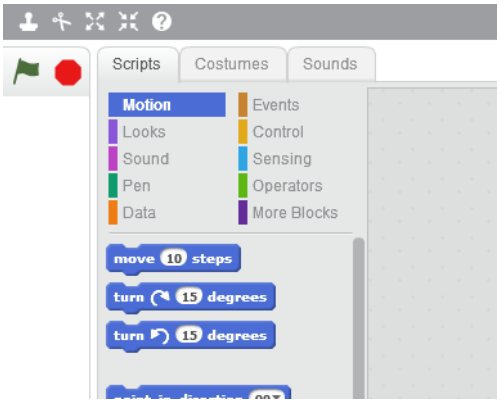
The Interface

The Scratch interface is divided into two sections: the project running environment and the project development. In the top-left of Scratch is the Stage, shown at the top of the image on the right. The Stage is where a Scratch project is physically run, so when one plays a game, the Stage is the window in which it is run. By default, the Scratch Cat is on the stage. The Scratch Cat is simply one of many sprites, or characters, buttons, etc. in a project. Characters are programmed to perform anything one wants! The flexibility of Scratch allows the creator to be imaginative and actually make the desired project. That is when programming comes into place, as it "makes things do what they should".

Programming

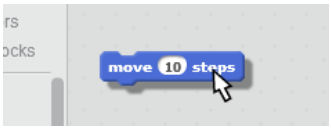
Before getting more into the interface, the quickest way to understand how sprites are programmed in Scratch is by testing things out. Follow the steps below when the Scratch program is opened with a clean, new project.

1. Access this area of the Scratch program:



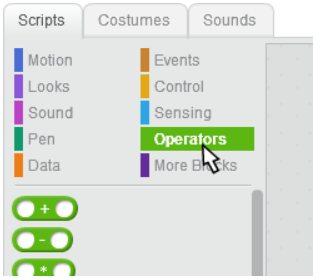
2. Select the blue "block" called move () steps, and drag it to the right.

3. Release the mouse to place the block; make sure the block is placed in the darker grey, technically called the scripts area.



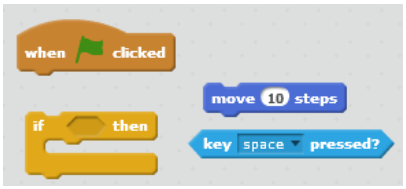
4. When done, click anywhere on the block except the white middle, and watch what happens to the Scratch Cat... it moves 10 steps!

5. Check out the other block categories and test out what each one does!

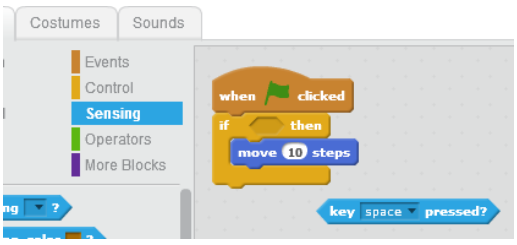


Blocks

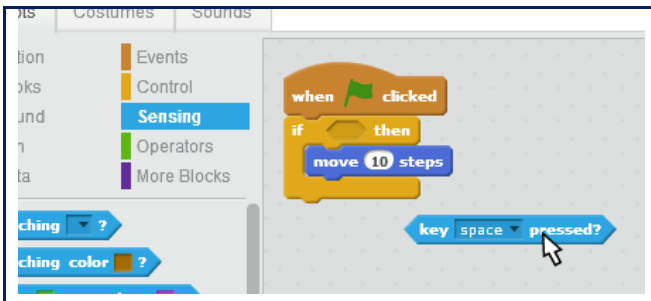
As shown above, blocks are the literal building "blocks" of a Scratch project. They have specific commands which function uniquely from one another. Some blocks can even fit inside *other* blocks, as shown below: **1.** Assemble the following "script", or connection of blocks, by accessing the various blocks by color and category.



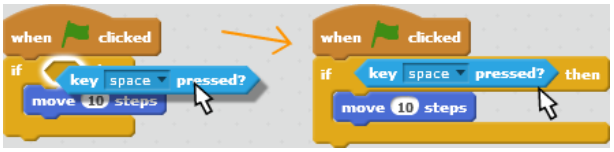
2. Assemble the blocks into this formation:



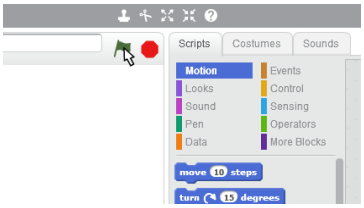
3. Grab the blue key sensing block that is still in the void.



4. Place the block into the hexagonal input area in the orange "if" block.



5. Click the Green Flag to run the project, and see what it does!



6. Unless you were holding down the space key, nothing should have happened. Why is that? Take a look at the script again; remember, a script is a connection or stack of blocks.

whenclickedifkeyspacepressed?thenmove10steps

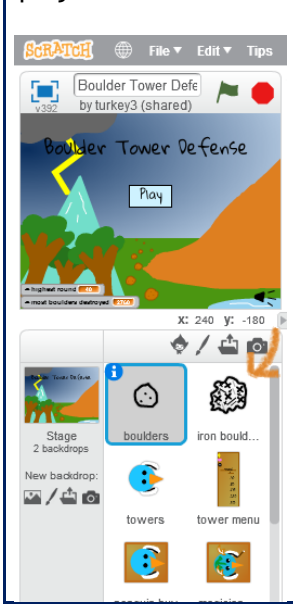
The script begins with "when green flag clicked", which was done. When the green flag is clicked, it triggers the script beginning with the "when green flag clicked" block to run. When the script ran, it first detects *if* the space key is down, and *if* it is, *then* the sprite will move 10 steps. Run the project again while holding the space key down, and the sprite will move 10 steps!

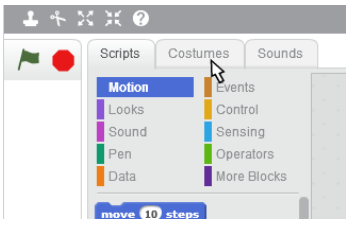
Paint/Sound Editors

Main article: Paint Editor

Main article: Sound Editor

Scratch even includes its very own paint editor and sound editor. A paint editor is a program used for designing and editing images. The Scratch paint editor can be used to draw the images for sprites (the characters, buttons, etc.). The sound editor is used for importing, recording, and modifying sounds used in a project. To access these two editors, click on the tabs above the blocks palette:





Sprites Panel

The sprites pane of a user's project. The currently selected sprite is highlighted blue.

Not all sprites do the same functions in a project, so different sprites have different appearances, scripts, and sounds stored in their data. Accessing different sprites can be done in the sprites pane, located below the stage. The currently selected sprite has a blue box around it always; by simply clicking on a different sprite, its data can be accessed. The sprites pane is shown in the image to the right.

Creating New Sprites

Most projects on Scratch have indeed more than one sprite. How can new sprites be created in Scratch? Underneath the Stage is four valuable buttons for creating a new sprite:

With these buttons, a new sprite can be imported as either a plain image or one that already has scripts. The buttons, going from left to right, do the following:

- Opens a pre-built sprite library with many choices
- Allows one to draw his or her own sprite in the paint editor
- Opens a file explorer to allow one to upload an image from his or her computer
- Turns on the computer webcam to take an image for a sprite

Project Sharing

For a more in-depth page about Project Sharing click here

Online Editor

Note You must confirm your email address in order to share.
:

One can share a project from either the unshared project's page or directly in the editor.

From an unshared project page, a user can click the "share" button in an orange bar above the project, as shown:

This project is not shared -- so only you can see it. Click share to let everyone see it!

In the project editor, in the top-right corner adjacent, the project page button is a "Share" button which appears for an unshared project. Clicking this will share the project, opening the project page as well.

Your project is now shared.

Offline Editor

In the offline editor, to share a project, click "File > Share to website" then enter the details. Then wait for a message that says "Successfully uploaded to scratch.mit.edu!"

Remixing

Main article: Remix

The Scratch Website is filled with many projects. Feel free, if you find any cool projects, to remix them. Follow these simple steps to do so.

1. Press the "See Inside" button in the corner of the project.
2. Edit the scripts inside of the project to make it your own.
3. Press the orange remix button in the top-right corner.
4. The edited project is yours!

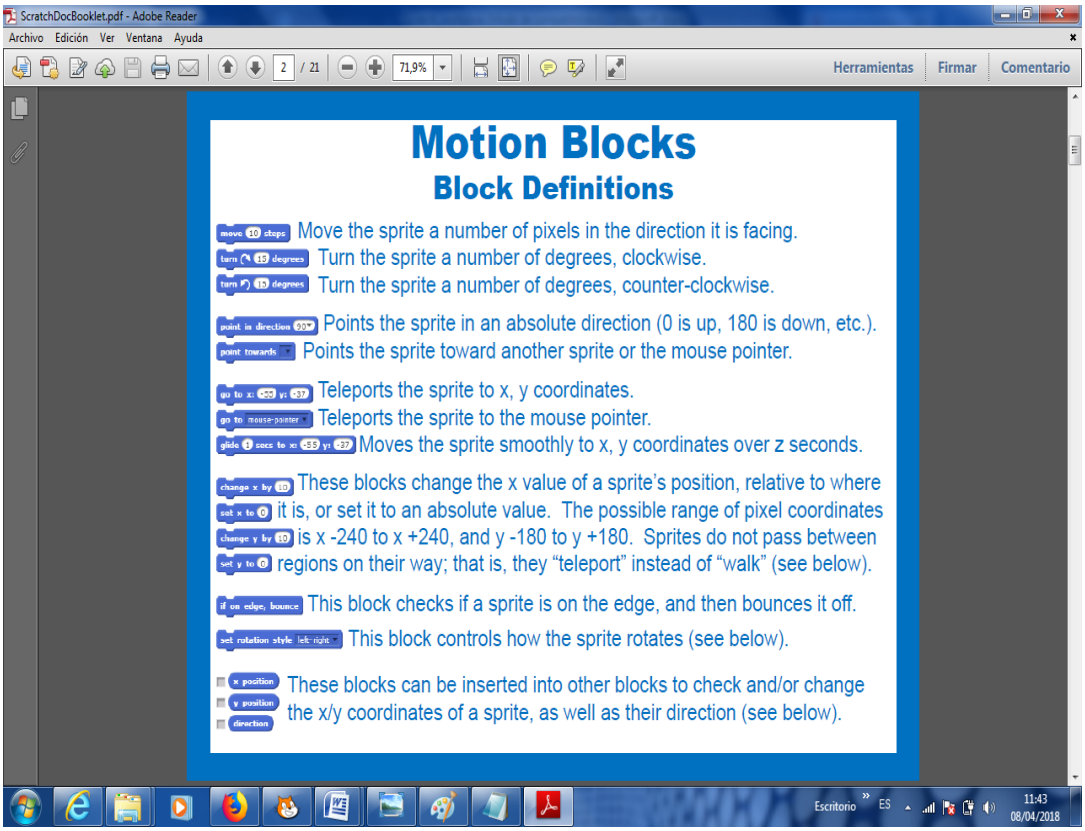
What Now?

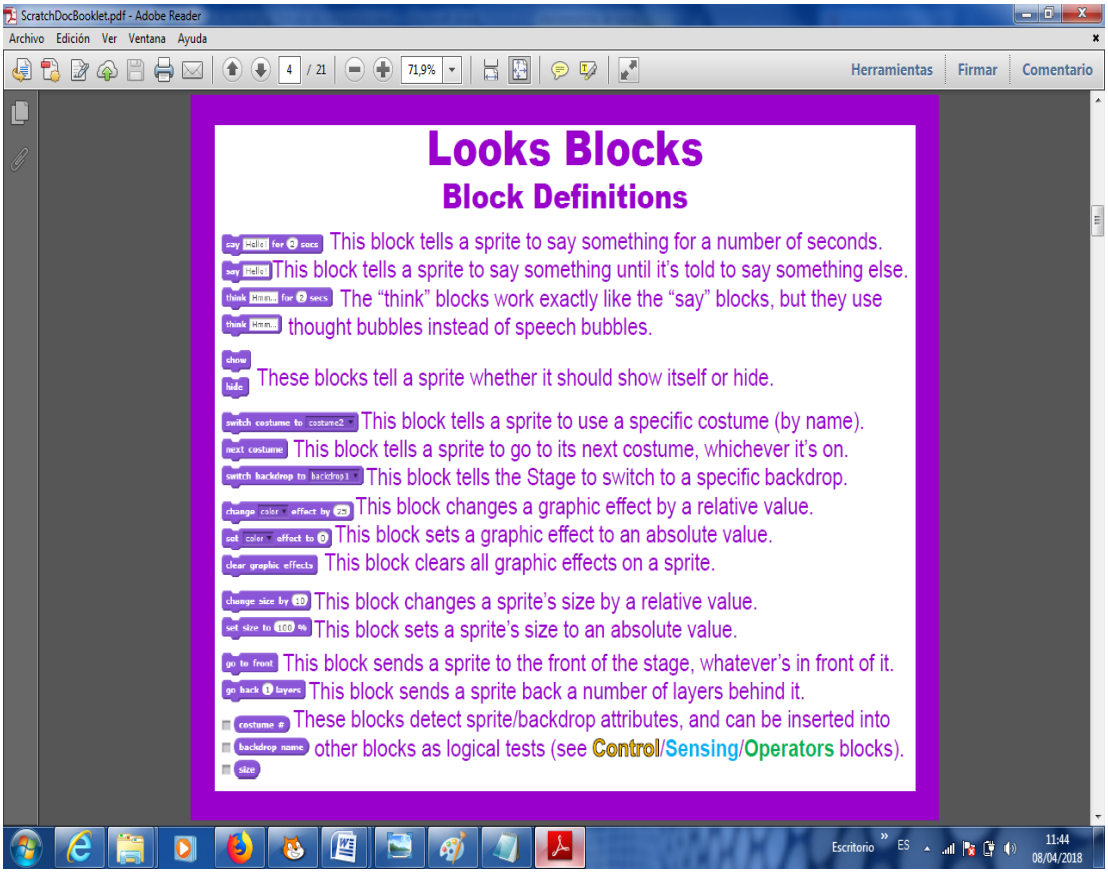
One possible way to advance a beginner's knowledge with Scratch is by playing around with it. Trying out different blocks, testing tools in the paint editor, and seeing all the nifty sound editor features can help one learn more about the program. Resources such as the Scratch forums and Scratch Wiki can be utilized as help when necessary. Creating many quality, hard-worked projects is arguably the best way to learn about Scratch.

Tutorials

You can also check out some video tutorials on Scratch. There are a wide variety of tutorials you can choose from that range from how to make your sprite move to making your own story! Here is a link to the tutorials.

ATTACHMENT II
SCRATCH BLOCKS





Sounds Blocks

Block Definitions

play sound meow — This block starts playing a sound, then moves to the next block.

play sound meow until done — This block plays a sound completely, then moves on.

stop all sounds — This block stops all sounds in the program.

play drum 1 for 0.25 beats
rest for 0.25 beats
play note 60 for 0.5 beats
set instrument to 1

These blocks are able to make music with different instruments (including drums), and are also capable of rests. See online tutorials for in-depth information on how to make your own musical compositions in Scratch.

change volume by -10
set volume to 100 %
volume

These blocks change volume by a relative value or set it to an absolute value. The **volume** block detects volume and can use it as a variable (see **Operators** blocks).

change tempo by 20
set tempo to 60 bpm
tempo

These blocks change the tempo of scripted music by a relative value, or set it to an absolute value. See online tutorials for in-depth information on how to compose music in Scratch.

Pen Blocks

Block Definitions

clear — This block erases all pen drawings from the stage.

stamp — This block makes a "stamp" of the sprite, as it looks now, on the stage.

pen down — This block places the pen down into the "draw" position.

pen up — This block lifts the pen up from the paper, so it will not draw on the stage.

set pen color to — This block sets the color of the pen to a color you choose.

change pen color by 10 — This block changes the pen's color by a relative amount.

set pen color to 0 — This block sets the pen's color to an absolute value.

change pen shade by 10
set pen shade to 50

These blocks change or set the darkness of the pen's color. Experiment with it to see how it works; the graphical effects of the pen are difficult to judge without experience.

change pen size by 1
set pen size to 1

These blocks control the size of the pen, in pixels. You can change the size by a relative value, or set it to an absolute value.

Data Blocks

Block Definitions

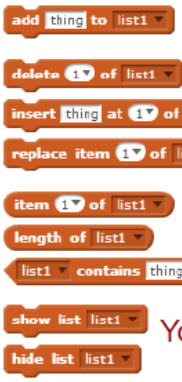
Make a Variable Use this button to create a variable. You should name it something

variable1 descriptive, after what you're using it for (e.g. "score," "lives," etc.).

These blocks set a variable to an absolute value, change it by a a relative value, and control whether it is shown or hidden. Variables like a score or number of lives should be shown, but other variables can be hidden (for "achievements" or other).

Make a List Use this button to create a list. You should name it descriptively, as in

list1 "Items" or "Names," depending on what you're using it for.



These blocks add, remove, and change list items. You can use these blocks for a wide variety of functions – experiment with them, and look up tutorials on the internet for advanced functions.

These blocks can be inserted into other blocks (see **Operators**) as all kinds of data. See the "Number Guesser" project (at https://scratch.mit.edu/projects/22179026/#editor) for ideas. Get creative with it, and experiment to learn the finer points. You will rarely want to show/hide lists, but exceptions are exceptional.

Events Blocks

Block Definitions

This block starts a script of blocks, and will start whenever the the green flag is clicked. You will probably use this block a lot.

This block starts a script of blocks when a key is pressed. You can use A-Z, 0-9, space, and arrow keys.

This block starts a script when the sprite is clicked.

This block starts a script when the stage's background switches to a certain backdrop (you can define it by name).

This block starts a script of blocks when the microphone receives a certain loudness. This requires a microphone in order to work; it only detects volume; look up tutorials for more information.

These blocks use messages to communicate "behind the scenes" information between sprites. Look up tutorials for more information, or see https://scratch.mit.edu/projects/33044982/#editor for an example.

Control Blocks

Block Definitions

wait 1 secs — This block tells a sprite to wait for some time (1 sec, .25 sec, etc.).

repeat 10 — This block tells a sprite to execute the code inside a number of times.

forever — This block tells a sprite to execute the code inside forever (all the time).

if then — This block checks a logical test, and if the test succeeds, then it will execute the code inside. If not, the code inside is skipped.

if then / else — This block checks a logical test, and if the test succeeds, then it will execute the code inside the first block. If not, then it will execute the code inside the second block (the "else" part).

wait until — The **wait until** block waits until a logical test succeeds, then executes the code inside. The **repeat until** block will repeatedly execute the code inside until a logical test succeeds; then it will stop.

repeat until

stop all — This block stops all scripts in your program. Think of it as "Quit" or "Exit."

when I start as a clone / create clone of myself / delete this clone — Clones are copies of a sprite with separate scripts of their own. Look up online tutorials to see how they work, or see the wiki: http://wiki.scratch.mit.edu/wiki/Cloning for more information.

Sensing Blocks

Block Definitions

touching ? / touching color ? / color is touching ? — These blocks check if a sprite or clone is touching another sprite or a certain color, or if one color is touching another color. Insert them into other blocks (usually **Control** blocks) to serve as logical tests.

distance to — This block measures the distance between sprites (in pixels).

ask What's your name? and wait — This block waits for user input. You can store the answer with

answer — the **answer** block as a variable or string of text for later use.

key space pressed? — This block checks if a key (A-Z, 0-9, space, or arrows) is pressed.

mouse down? — This block checks if the left mouse button is clicked.

mouse x / mouse y — These blocks measure the x and y coordinates of the mouse pointer.

loudness — This block measures the absolute volume from the microphone input.

video motion on this sprite / turn video on / set video transparency to 50 % — Scratch is able to use the webcam (if you have one), but it is complicated. Look up an online tutorial or see the wiki for more information.

timer / reset timer — Scratch has an in-game timer, but it is often quirky. In most cases, you are better off making your own timer using **Data** blocks and the **wait** block.

x position of Sprite1 — This block can measure various attributes of a sprite.

current minute / days since 2000 / username — These blocks are useful for cloud variables. See the wiki for more info.

B

Operators Blocks

Block Definitions

These blocks will perform arithmetical operations on numbers and/or variables. You can put these blocks inside other blocks, and inside each other.

This block picks a random number from a user-defined range.

These blocks test for equivalence or relative value (equals, greater than, or less than). They can be used with **Control** blocks to make logic tests.

These blocks are used for logical tests, mostly with **Control** blocks.

These blocks are used to do things to "strings" of text, but can also be used with numbers. Check the wiki for more information on this topic.

The "modulo" function reports the remainder of a division operation.

The "round" function removes the remainder from a division operation.

The "sqrt" function reports the square root of a number.

For the other operations (shown at left), you may wish to look up them up on the wiki or in a high school algebra/trigonometry textbook. They have many uses (for example, distance as reported in the **Sensing** blocks, or direction as shown in the **Motion** blocks), but are complicated if you don't know what they are. Brief definitions are given below.