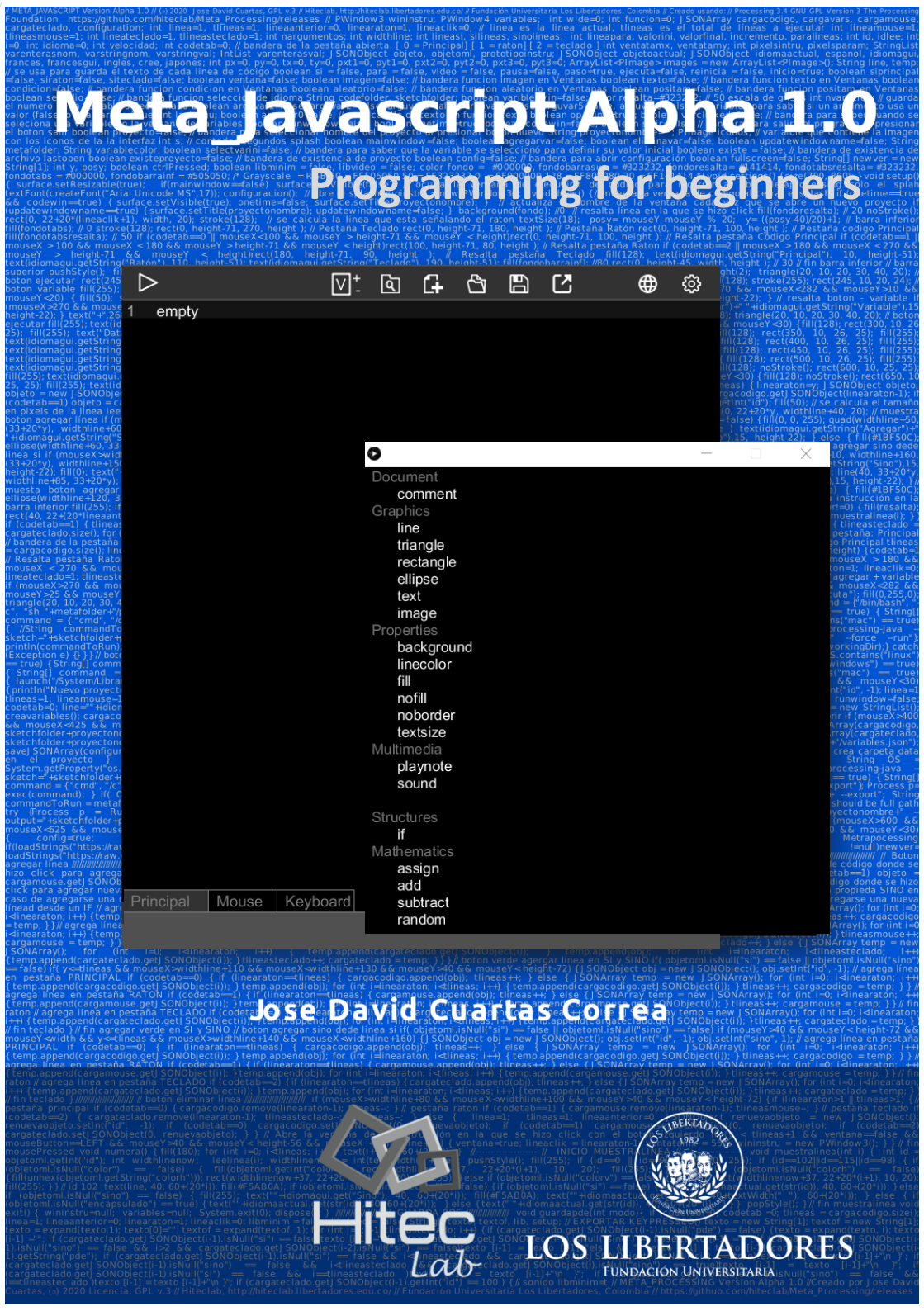

Meta_Javascript_Alpha 1.0
Programming for beginners
1 empty
Document
comment
Graphics
line
triangle
rectangle
ellipse
text
image
Properties
background
linecolor
fill
nofill
noborder
textsize
Multimedia
playnote
sound
Structures
if
Mathematics
assign
add
subtract
random
Principal
Mouse
Keyboard
Jose David Cuartas Correa
Hitec Lab
LOS LIBERTADORES
FUNDACIÓN UNIVERSITARIA

Meta_Javascript Alpha 1.0

Programming for beginners

Jose David Cuartas Correa

LOS LIBERTADORES
FUNDACIÓN UNIVERSITARIA

DRAFT EDITION 1.1: MAYO, 2020

LOS LIBERTADORES, FUNDACIÓN UNIVERSITARIA
Bogotá D. C., Colombia
Cra 16 No. 63 A - 68 / Tel. 2 54 47 50
www.ulibertadores.edu.co

Juan Manuel Linares Venegas
President of the institution

Patricia Martínez Barrios
Chancellor

Ángela María Merchán Basabe
Academic vice chancellor

Jose David Cuartas Correa
Layout and cover design

INTRODUCTION

With this text I seek to introduce the reader to the basic aspects of the first version of Meta_Javascript, a meta-programming language that I developed for the beginner of web application programming. It is based on Meta_Processing and was designed to help you create interactive Javascript code. It is a personal initiative that is influenced by my work as director of the Hypermedia Laboratory (Hitec Lab) at the Fundación Universitaria Los Libertadores (Bogotá, Colombia).

What is sought with this derivation of Meta_Processing is to create an ecosystem of applications that allow the user to program different platforms using the same programming language. Today it is almost essential to learn a different language to program each platform.

This programming environment is based on the needs that I identified during the years that guided a basic programming course for graphic designers at the Fundación Universitaria Los Libertadores. It also responds to part of my research interests since when I was doing my PhD in Design and Creation, which ended with the thesis: “Program the world in the context of free technologies and Hacker-Maker cultures. Case study: Hitec Lab ”(fourth, 2017) .

Meta_Javascript is one of several initiatives that I have been leading in the quest to demystify technologies and to help fulfill one of the unfulfilled promises of free software: Allow anyone to modify and adapt the software they use, so that they can adjust it to their particular needs and personal likes. So making programming code easier to write and read is one of the purposes of Meta_Processing and Meta_Javascript.

Jose David Cuartas Correa
Bogota Colombia
2020

Thanks:

To the unconditional support received from the Fundación Universitaria Los Libertadores who have believed in every project we develop at Hitec lab.

Dedication:

To my wife Shahzadi and my daughters Helen and Megan, who fill my existence with love and joy.

CONTENT

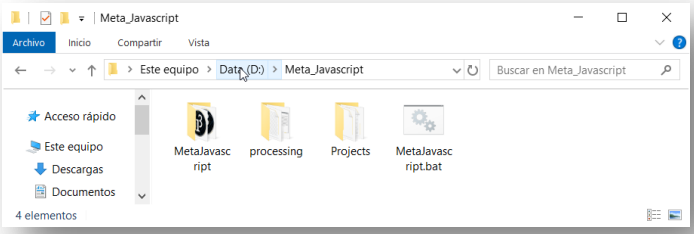
1. META_JAVASCRIPT FIRST STEPS

1.1. How to open Meta_Javascript?

Meta_Javascript was developed to work on the operating systems: Windows, Mac and GNU / Linux. The steps to open Meta_Javascript vary slightly depending on the operating system used, the steps for each system are described below:

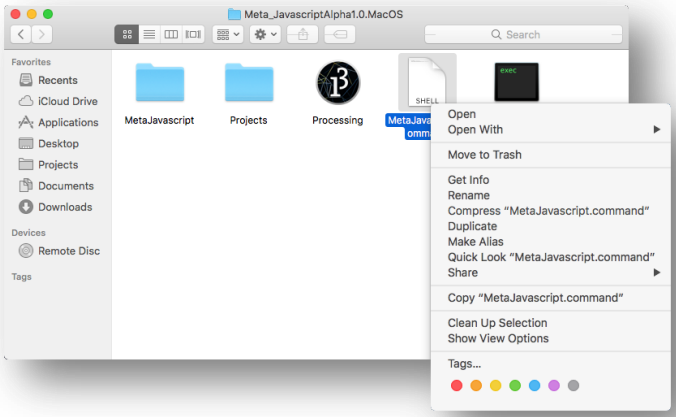
Windows

In Microsoft Windows, double-click on the file with the name: **MetaJavascript.bat**

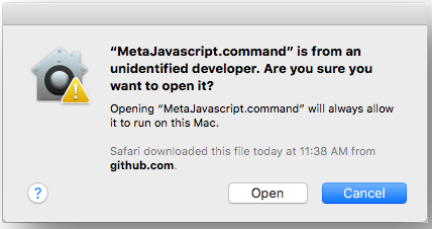


Mac OS

On Mac Os you must right-click on the file with the name **MetaJavascript.command** and select the option: **Open**

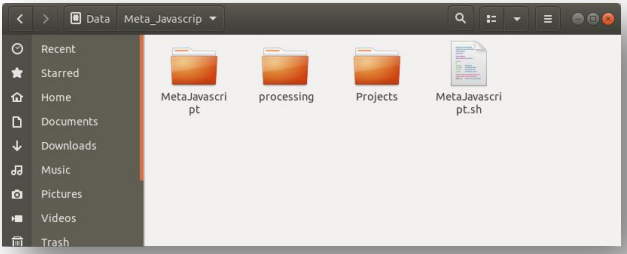


In the window that opens, select the option: **Open**

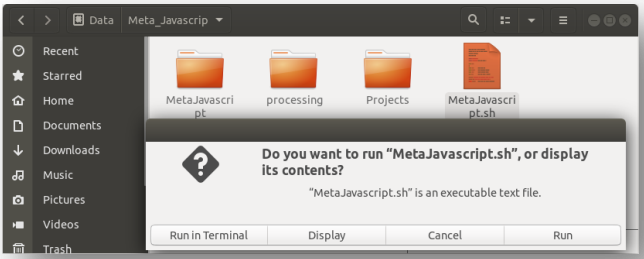


GNU / Linux

In GNU / Linux double click on the file with the name **MetaJavascript.sh**



In the window that opens you must select the option: **Run** (you can also use Run in Terminal if you want to open the terminal window of Meta_Javascript).

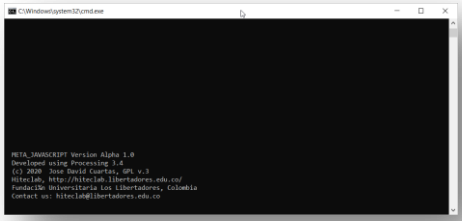


If the script does not open when you double click on it, you can try to run the following command in the Linux terminal to activate the previous dialog box.

```
gsettings set org.gnome.nautilus.preferences executable-text-activation ask
```

1.2. Windows that open when you start Meta_Prosessing

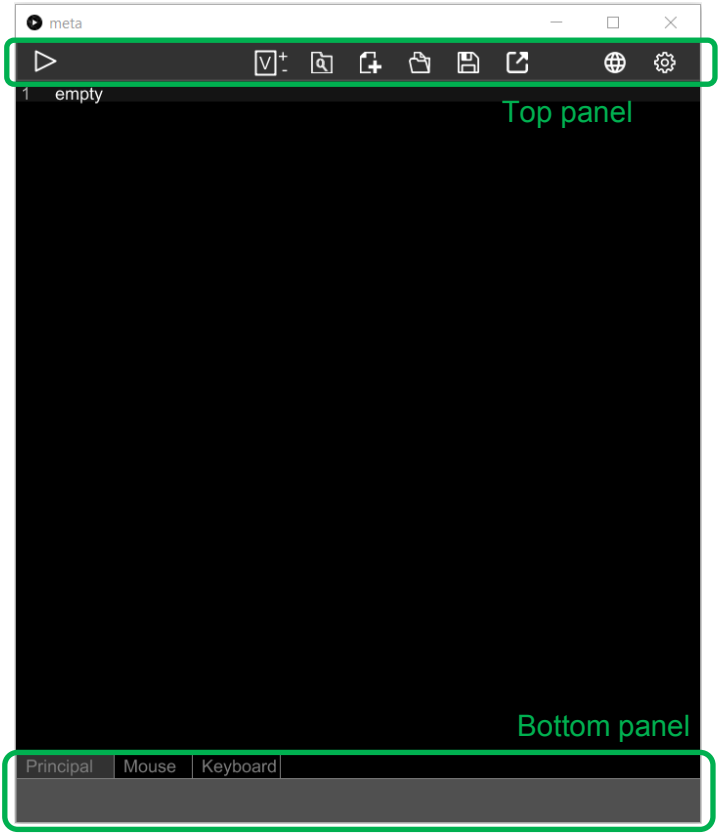
Once the Meta_Prosessing file is executed, regardless of the operating system you are working on, the terminal window opens where you can see messages that come from the main Meta_Javascript window.



Then a second window opens with the animated splash of welcome to Meta_Javascript. Clicking on or closing this window opens the main Meta_Javascript window.

1.3. **Basic elements of the interface**

In the upper panel are the buttons: Run, Variables, Data, New, Open, Save, Export, Languages and Configuration.



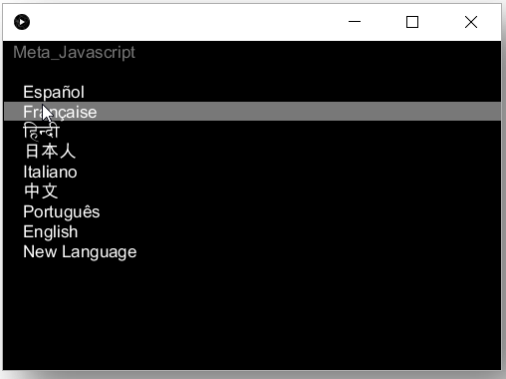
In the Bottom panel are the tabs: Principal, Mouse and Keyboard. Also there is the description bar:

where the names of the buttons and the prototype of the instructions are shown.

1.4. How to select Languages?

To change the Meta_Javascript language, click on the languages icon in the top bar.

Then in the window that opens, click on the desired language.



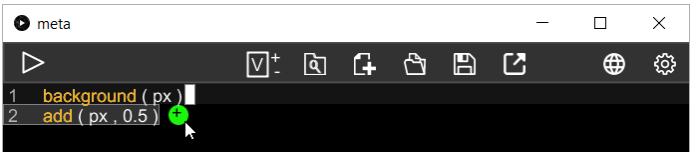
1.5. How to execute the code?

To execute the code, click on the run icon in the top bar.

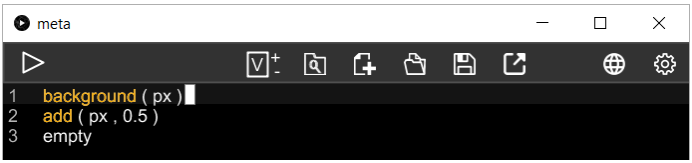
Wait few seconds and a new window should appear in which the created code will be executed.

1.6. How to add a line of code?

To add a line of code you must move the mouse cursor until a green circle appears with the plus (+) character inside it.

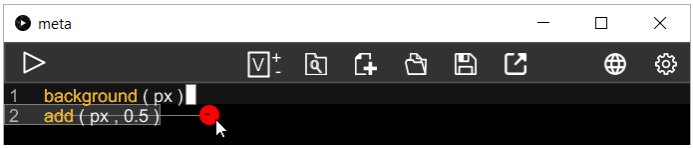


A new line of code will appear once the green circle is clicked.

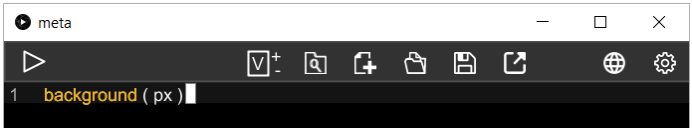


1.7. How to delete a line of code?

To remove a line of code, you must move the mouse cursor until a red circle appears with the minus character (-) inside it, and you see a gray line crossing out the entire instruction.

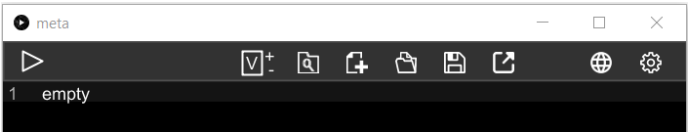


The line disappears once it is clicked.

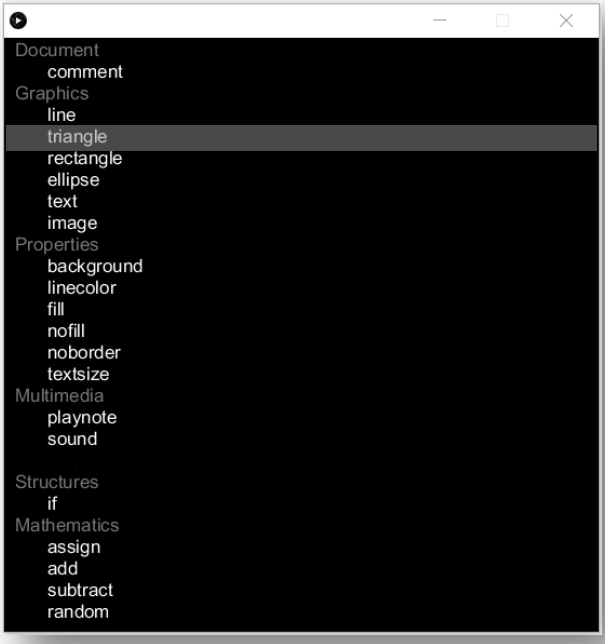


1.8. **How to add instructions?**

To add an instruction, you must click on the word that says **empty**.



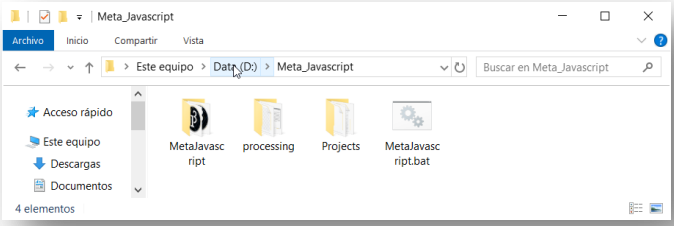
Once this is done, a new window will open where all the instructions available in Meta_Javascript will appear, organized by categories like this:



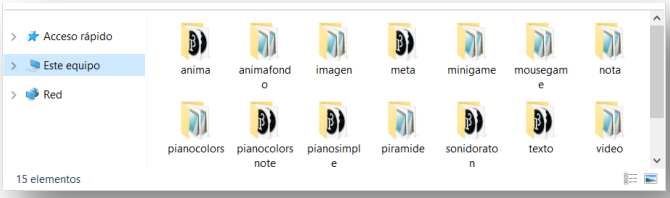
When the mouse cursor is placed on any of these instructions the instruction is highlighted, in this example you can see how the **triangle** instruction is highlighted. Clicking will open a new window where you can enter the properties of each instruction. The description of each of these instructions will be made in Chapter 2.

1.9. What is the file and folder structure in Meta_Javascript?

Inside the **Meta_Javascript** folder is the file to run Meta_Javascript and three sub-folders. The Meta_Javascript folder contains the files that allow it to function. In the processing folder there is a distribution of this language that is used to execute Meta_Javascript.

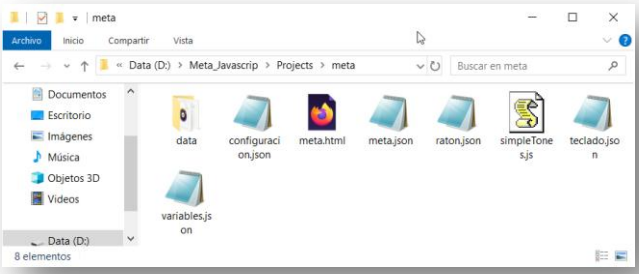


In the **Projects** folder it contains the folders of each of the program projects written using the Meta_Javascript programming environment.

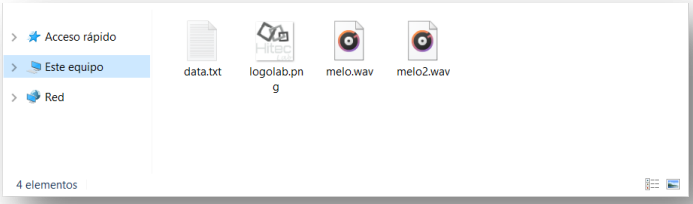


In the folder of each project there are few **.json** files, one **.js** file and one **.html** file. The **.json** files contain

the instructions in Meta_Javascript language, the **.js** file contain a sound library necessary to play notes in the web browser and the **.html** file contain the javascript code of the project, and is generated every time the **run** icon is clicked.



And in the **data** folder of each project the files that will be used in the execution of the program are stored, such as images, sounds and videos.



1.10. How to open the data folder of the current project?

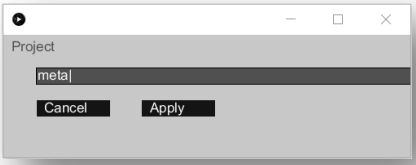
To open the data folder of the current project, click on the **data** icon in the top bar.

Once clicked, the data folder of the current project opens in another window

1.11. How to create a new project?

To create a new project, click on the **new** icon in the top bar.

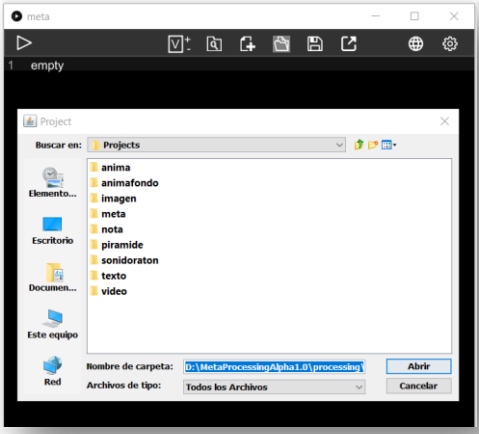
In the window that opens, you must write the name that you want to give to the new project and click on the **Apply** button.



1.12. **How to open a project?**

To open a project, click on the **open** icon on the top bar.

Then a new window opens in which you can select the project folder you want to open and click the **open** button.



1.13. How to save the current project?

To save the current project, click on the **save** icon in the top bar.

1.14. How to export the current project as application?

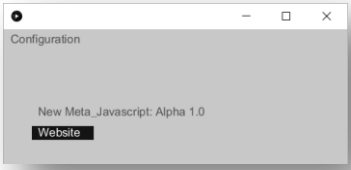
To export the current project as an application, click on the **export** icon on the top bar.

The application is saved in the subfolder called **app** inside the current project folder.

1.15. Configuration icon

To change the configuration options, click on the **configuration** icon in the top bar.

Then a new window opens offering the option for check the latest version of Meta_Javascript.



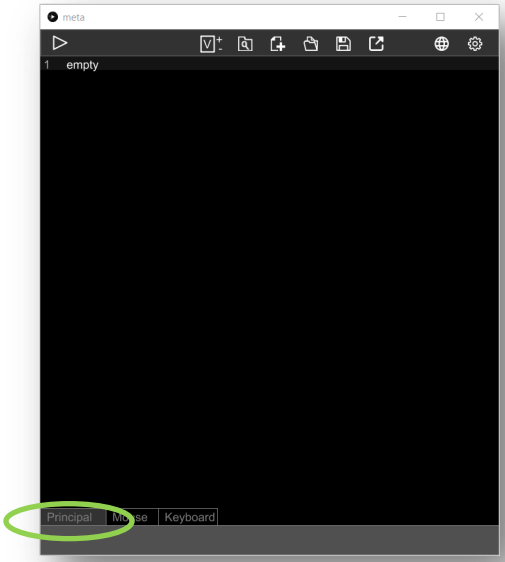
The **New Meta_Javascript** option allows you to check the latest version of Meta_Javascript published on the internet. The information that appears after the colon is the version available for download. If you want to download that new version you can click on the **Website** button that redirects to the official Meta_Javascript download website.

1.16. Functions: Principal, keyboard and mouse

To write the code in Meta_Javascript, three functions can be used: Principal, Mouse and Keyboard. Each of these functions is selected by clicking on its corresponding tab.

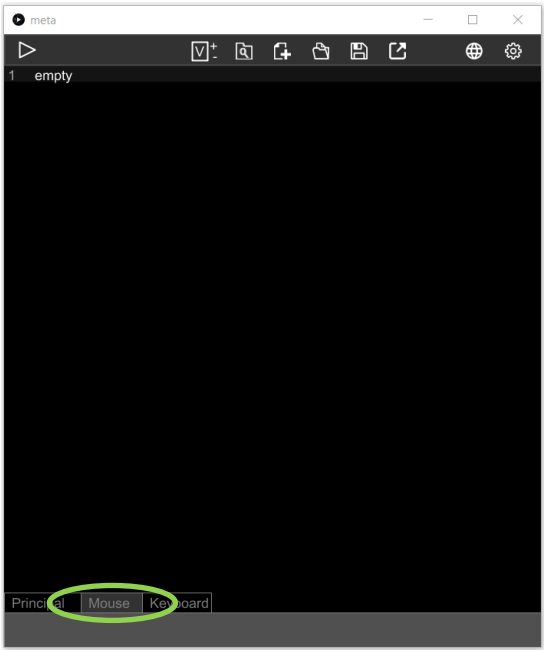
1.16.1. Principal

The code that is written to the **Principal** tab runs in an infinite loop, until the application window is closed.



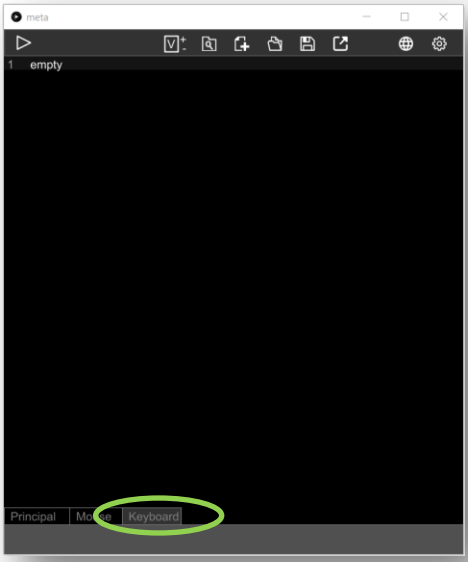
1.16.2. **Mouse**

The code that is written on the **Mouse** tab is executed the moment any mouse button is pressed.



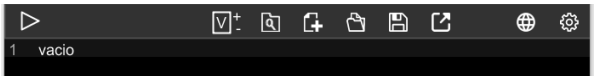
1.16.3. **Keyboard**

The code that is written on the **Keyboard** tab runs the moment any key is pressed.

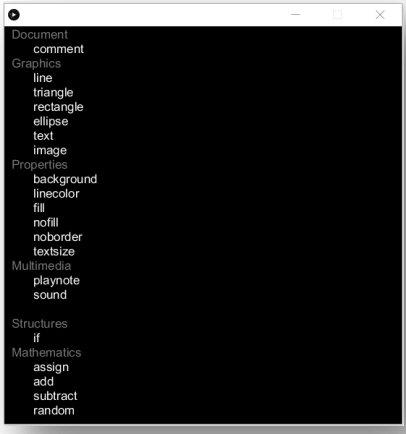

meta
1 empty
Principal
Keyboard

2. **BASIC INSTRUCTIONS**

This section will cover the basic instructions for programming with Meta_Javascript. As explained in point 1.8. to add an instruction you must click on the word that says **empty**.



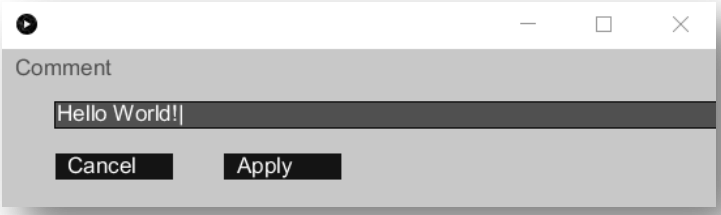
Then, a new window will open showing all the instructions available in Meta_ Javascript organized by categories like this:



2.1. Document the code

Documenting the programming code is one of the first things anyone who wants to learn to program must learn. For this purpose, all programming languages allow adding comment lines. The main feature of this line of code is not executed; it's just there to give the developer information on how that part of the code works. To add a comment, click on the **comment** option in the **Document** category.

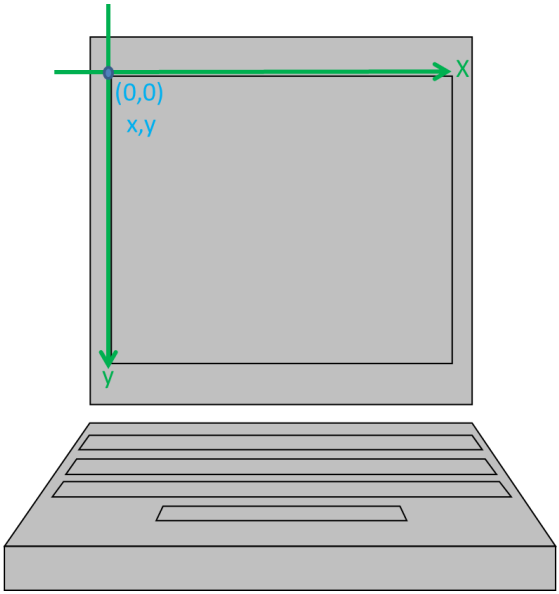
Then a new window will open in which you write the comment and click **Apply**.



2.2. Screen coordinates

In order to make graphics on the screen, it is necessary to first know how the screen coordinates work in Meta_Javascript. The positions on the screen are measured in pixels and each screen has a certain number of pixels on the X axis and on the Y axis.

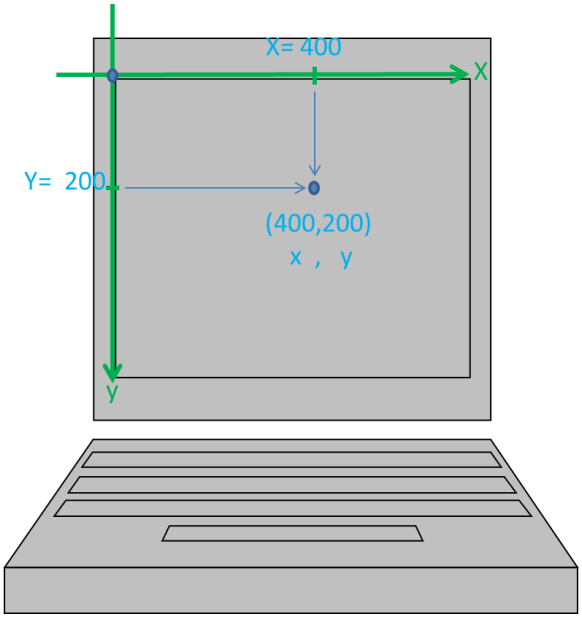
As can be seen in graph 1, the upper left corner of the screen is the point of origin of the coordinate system. This point is position 0 on the X axis and position 0 on the Y axis. Coordinates are always arranged first with value on X axis, then a comma and then the value on the Y axis, so this point is (0,0).



Graph 1 Origin point in the coordinate system

The positions on the X axis increase from left to right and the positions on the Y axis increase from top to bottom. Figure 2 shows an example to better illustrate this concept.

If you want to locate the point (400,200) on the screen, what you do is count 400 pixels to the right from the point (0,0) of the screen and count 200 pixels down from the point (0,0) of the screen. This is how the point (400,200) is located.



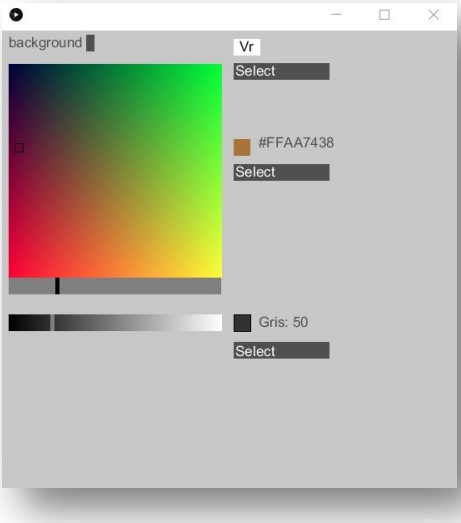
Graph 1 Point at position 400 in X and 200 in Y on the screen

2.3. On-screen graphics instructions

Some of the instructions to display on the screen include: line, triangle, rectangle, ellipse, text, image. Next pages will explain how to use each of them.

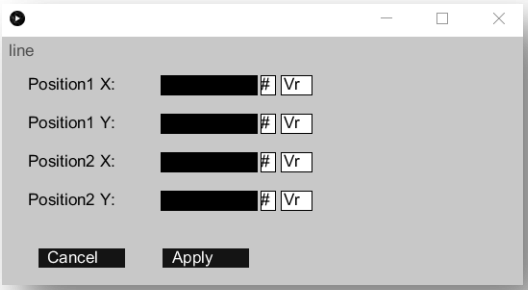
2.3.1. background

The **background** instruction is used to define the background color of the entire window of the application. This instruction erase everything that is being displayed on the screen and leaves the entire screen with the selected color.



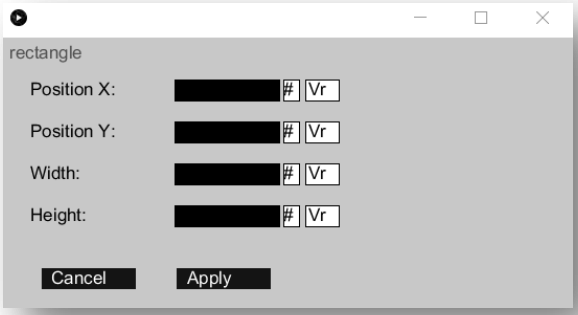
2.3.2. line

The **line** instruction is used to draw a line on the screen. Tracing a line requires defining the (x,y) position of the point where the line begins and the (x,y) position of the point where the line ends.



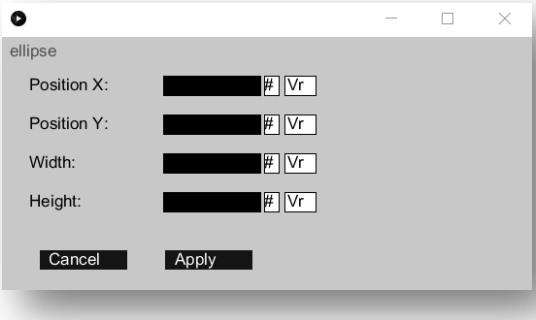
2.3.3. rectangle

The **rectangle** instruction is used to draw a rectangle on the screen. To use this instruction you must define in the first two boxes the position x,y of the upper left corner from where the square will be drawn, and in the next two boxes must be defined its width and height.



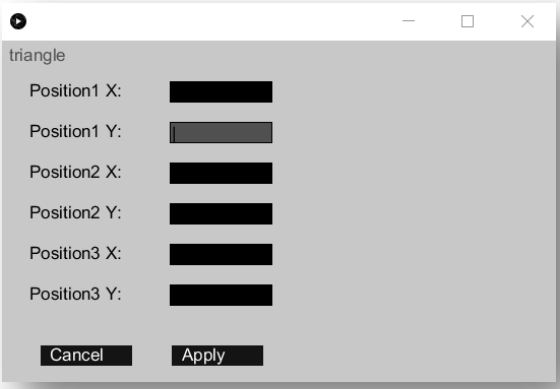
2.3.4. ellipse

The **ellipse** instruction is used to draw an ellipse on the screen. To use this instruction you must define in the first two boxes the center point x,y from where the ellipse will be drawn, and in the next two boxes define its width and height.



2.3.5. triangle

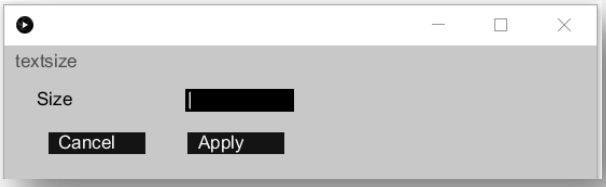
The **triangle** instruction is used to draw a triangle on the screen. To draw any triangle it is required to define the three points corresponding to each of its corners. To use this instruction, the position of the first point must be defined in the first two boxes, the position of the second point must be defined in the following two boxes, and the position of the third point must be defined in the last two boxes.



2.3.6. texsize

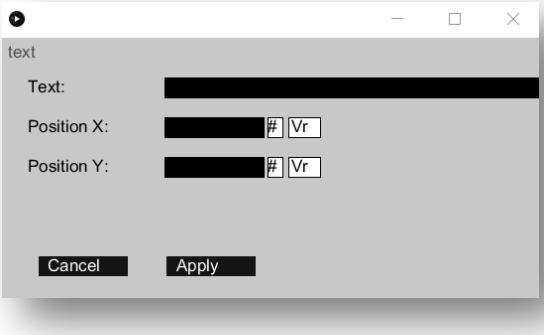
The **textsize** instruction is used to define the size of the font when displaying text on the screen. To use this instruction, the size box must be filled with the number corresponding to the font size to be applied.

For this statement to take effect, it must be added before the **text** instruction.



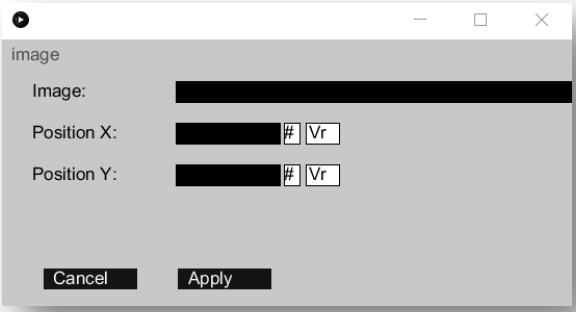
2.3.7. text

The **text** instruction is used to display text on the screen. To use this instruction, you must write in the first box the text you want to display, and define in the next boxes the x, y position of the lower left corner from where the text will start to be displayed on the screen.



2.3.8. image

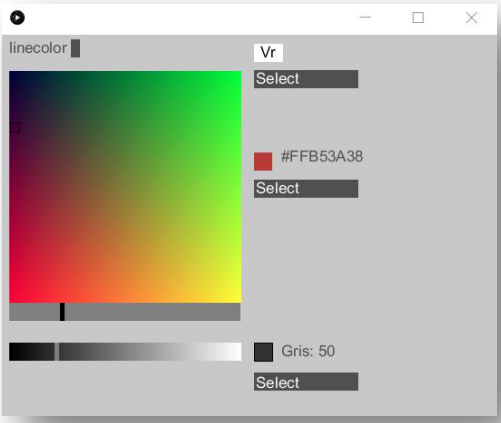
The **image** instruction is used to display an image on the screen. To use this instruction, you must first save the image you want to use inside the project's **data** folder, then select the image that was saved in the data folder, and after that in the next two boxes define the x, y position of the upper left corner from where the image will begin to be displayed on the screen.



2.3.9. linecolor

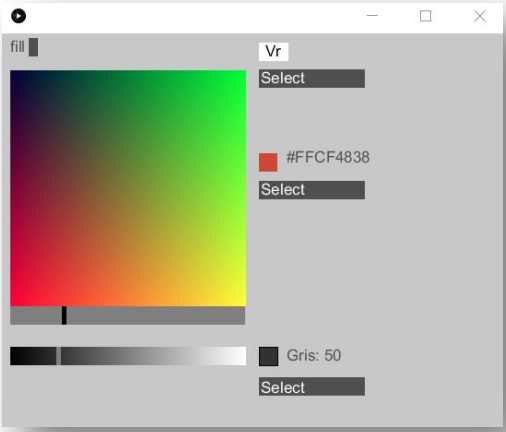
The **linecolor** instruction is used to define the color of the lines and the border color of the rectangular, ellipse and triangle figures. To assign the line color using this instruction you can, use the color picker, or the grayscale picker and click the **Apply** button. You can also use a variable to dynamically change the color. For this instruction to take effect, it must

be added before the instructions with which to draw lines or figures on the screen.



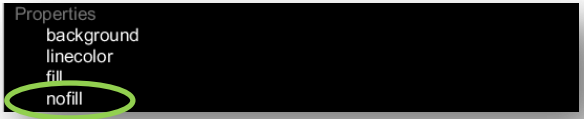
2.3.10. fill

The **fill** instruction is used to define the fill color of the rectangle, ellipse and triangle figures. To assign the fill color using this instruction you can, use the color picker, or the grayscale picker and click the **Apply** button. You can also use a variable to dynamically change the color. For this instruction to take effect, it must be added before the instructions with which to draw figures on the screen.



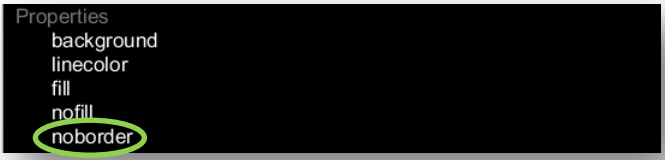
2.3.11. nofill

The **nofill** instruction is used to remove the filling of the rectangle, ellipse and triangle figures. This will look transparent and only its edge will be seen. To use this instruction, it is only necessary that in the add instruction window, you click on the **nofill** instruction within the **Properties** category and this line will automatically be added to the project code.



2.3.12. noborder

The **noborder** instruction is used to remove the border of the rectangle, ellipse and triangle figures. If this statement is added before the **line** instruction then the line will not be displayed. To use this instruction, it is only necessary that in the add instruction window, you click on the instruction **noborder** within the **Properties** category and this line will automatically be added to the project code.

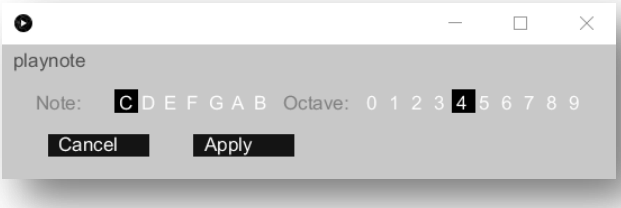


2.4. Multimedia instructions

Some of the multimedia instructions that can be used with Meta_Javascript include: **playnote**, **sound** and **video**. The next pages will explain how to use each of them.

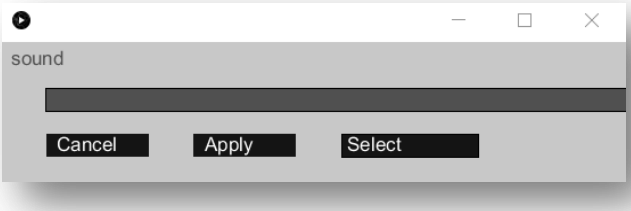
2.4.1. playnote

The **playnote** instruction is used to reproduce the sound of a note on the musical scale. To use this instruction, you must select the note that you want to play and then select the octave that you want the note to sound. Finally, click on the **Apply** button.



2.4.2. sound

The **sound** instruction is used to play a sound file in wav or mp3 format. To use this instruction you must first save the sound file you want to use inside the project's **data** folder, then click on the **Select** button to choose the file that was previously saved in the data folder and finally you must click the **Apply** button.



3. VARIABLES AND CONDITIONS

This section will address the concepts of variables and conditionals, which are fundamental when you are learning to program.

3.1. Variables

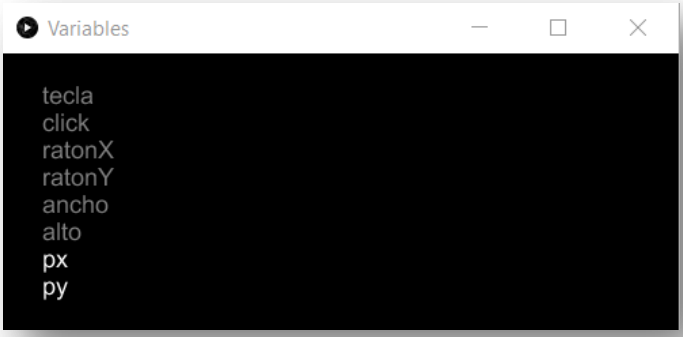
A variable is a memory space reserved for storing a value that changes while the program is on execution. In Meta_Javascript there are two types of variables: System variables and variables created by the user.

The system variables are: **tecla**, **click**, **ratonX**, **ratonY**, **ancho** and **alto**. The variable **tecla** stores the value of the last key pressed on the keyboard. The variable **click** stores the value of the last button pressed on the mouse. The variable **ratonX** stores the current mouse position on the X axis. The variable **ratonY** stores the current mouse position on the Y axis. The variable **ancho** stores the value of the width of the screen in which the code is executing. And the variable **alto** stores the value of the height of the screen in which the code is running.

3.1.1. How do you look at the list of variables?

To see the variables list of the project, click on the **variable** icon.

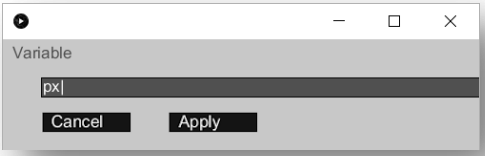
In the window that opens you will see the variables that are being used. The ones shown in gray are the system variables and the ones shown in white are the variables created by the user.



3.1.2. How is a variable created?

To create a variable, click on the plus (+) icon next to the **variable** icon.

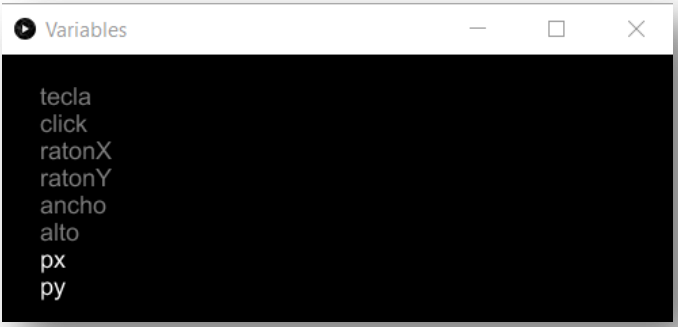
In the window that opens, you must write the name that you want to give of the variable to be created, in this example it is given the name px.



3.1.3. How is a variable removed?

To remove a variable, click on the minus icon (-) next to the **variable** icon.

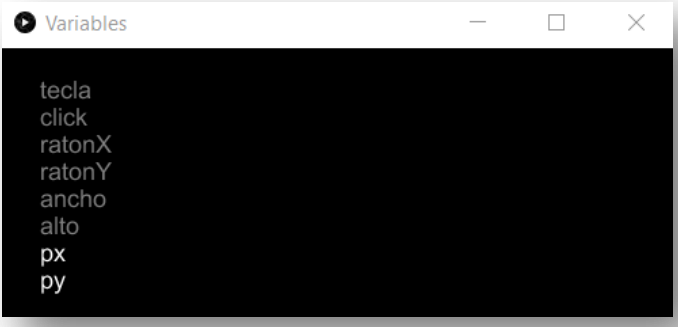
In the window that opens, click on the name of the variable to be removed. Only variables that have been created by the user can be removed, that is, only variables with white color can be removed. Variables in gray are system variables and cannot be removed.



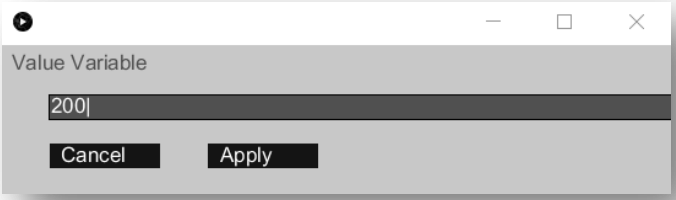
3.1.4. How is a variable initialized?

To initialize a user-created variable, click on the **variable** icon.

In the window that opens, click on the variable that you want to initialize.



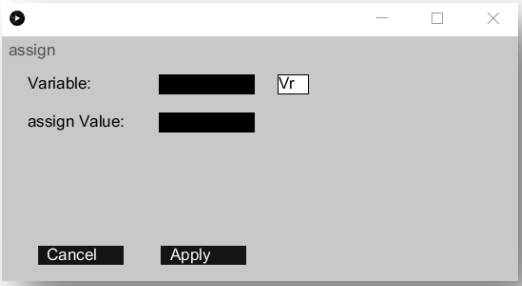
Once this is done, another window will appear in which you must write the value with which you want to initialize the variable. In this example, the variable is initialized with the value 200.



3.1.5. How to assign a new value to a variable?

Within the code you can assign a new value to a variable, for this you must add the **assign** instruction found within the **Mathematics** category in the add instruction window.

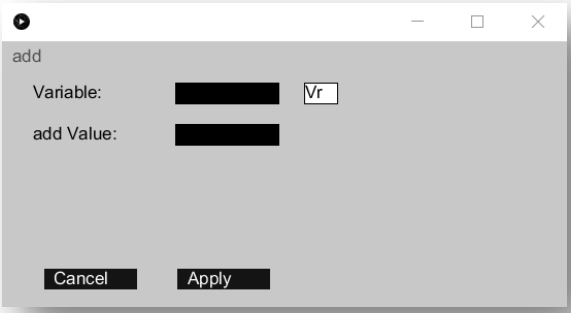
Inside the window that opens, in the first box select the variable you want to assign a new value and in the second box write the value to be assigned.



3.1.6. How do you add a value to a variable?

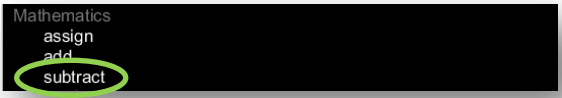
Within the code you can add a value to a variable, for this you must aggregate the **add** instruction found within the **Mathematics** category in the add instruction window.

Inside the window that opens, in the first box select the variable you want to add the value and in the second box write the value to be added.

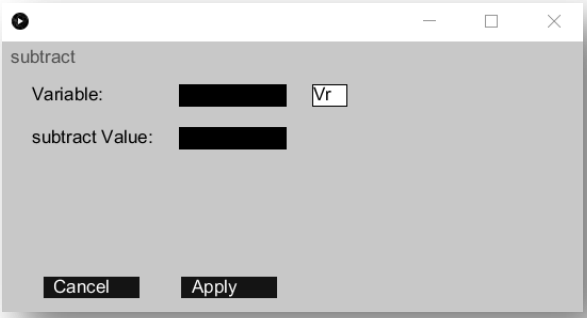


3.1.7. How do you subtract a value from a variable?

Within the code you can subtract a value from a variable, for this you must add the **subtract** instruction that is within the **Mathematics** category in the add instruction window.

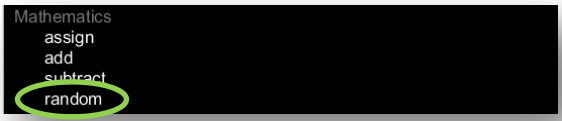


Inside the window that opens, in the first box select the variable you want to subtract the value and in the second box write the value to be subtracted.



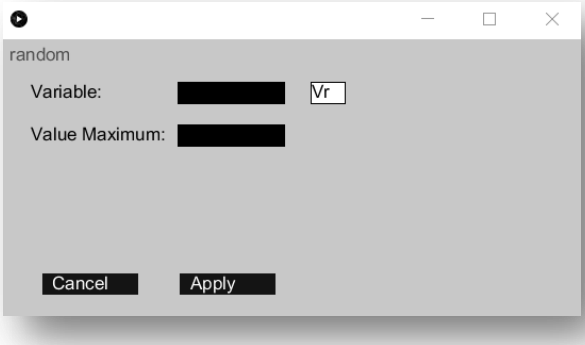
3.1.8. How is a variable assigned a random value?

Within the code you can assign a random value to a variable, for this you must add the **random** instruction that is within the **Mathematics** category in the add instruction window.



Inside the window that opens, in the first box select the variable you want to assign the random value and in the second box write the maximum value that would be generated randomly. A random value

would be generated between 0 and the maximum value defined inside the instruction.

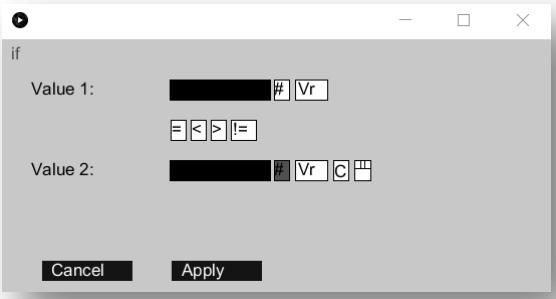


3.2. Conditionals

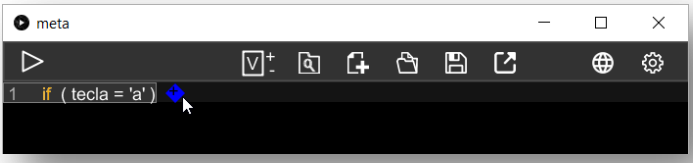
Conditions are a type of algorithmic structures that allow the program to make decisions as a condition is met.

To add a condition, you need add the **if** instruction that is within the **Structures** category in the add instruction window.

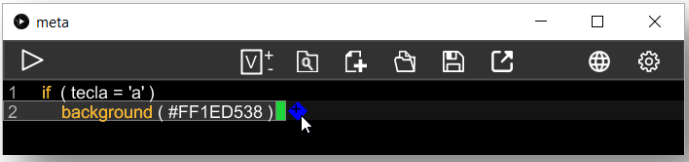
Then in the window that opens in the first box you can write a value or select a variable. Then an operator must be selected, which can be equal (=), less than (<), greater than (>) or different (¡=). And in the second box you can write a value, select a variable, choose a key or choose one of the mouse buttons.



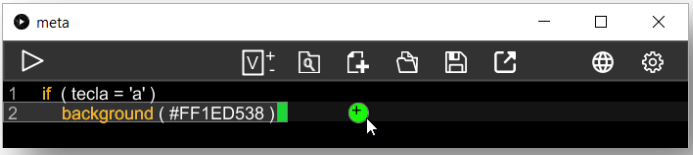
Once the **Apply** button is clicked, the condition can be seen in the Meta_Javascript code window. To add an instruction inside the condition, you must move the mouse cursor until a blue diamond appears with the plus (+) character inside it.



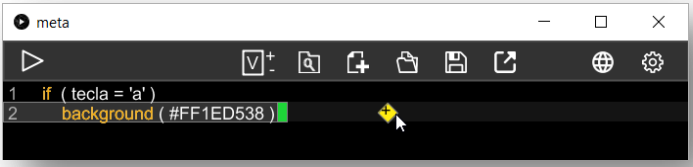
Once clicked the new empty line of code will appear and click to assign the desired instruction. If you want to create one more instruction inside the condition, then you must move the mouse cursor until a blue diamond appears with the plus (+) character inside it.



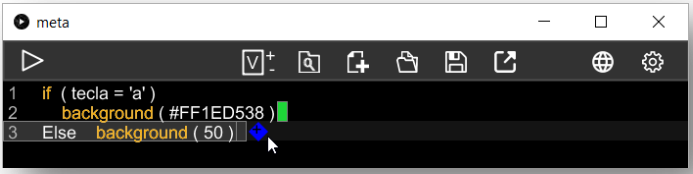
If once you finish adding the instructions inside the condition, what you want is to add another line of code but outside the condition, then you must move the mouse cursor until a green circle appears with the plus character (+) inside.



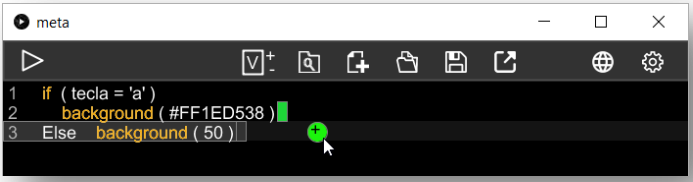
If, on the contrary, what you want is to add a line for the case when said condition is not met, then you can add lines within the Else. For this you must move the mouse cursor until a yellow diamond appears with the plus (+) character inside it.



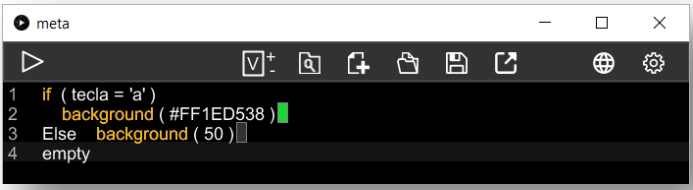
Once the instruction you want to use is selected, you would see that the word **Else** appears before the instruction. For when you want to add more instructions inside the **Else**, you must move the mouse cursor until a blue diamond appears with the plus (+) character inside it.



If, on the other hand, you want to add another line of code but outside the **Else**, then you must move the mouse cursor until a green circle appears with the plus (+) character inside it.



When you do this then a new empty line will appear outside the condition's instructions.

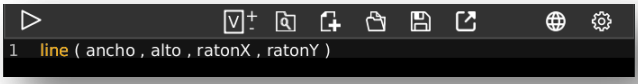


4. CODE EXAMPLES WITH META_JAVASCRIPT

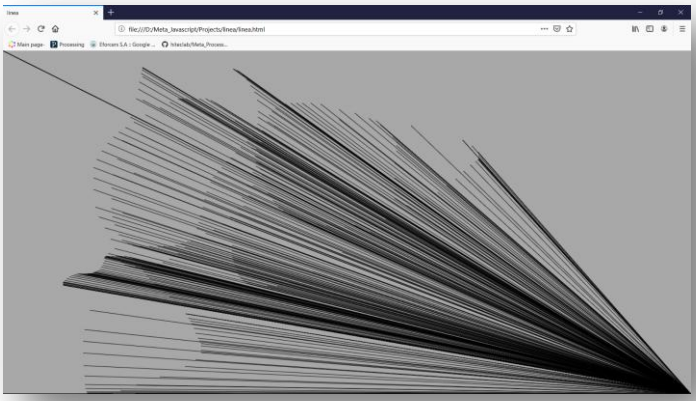
Below are a number of examples of how to use Meta_Javascript. The first of them shows how to make abstract line drawings. The second allows us to experiment with the keyboard function. The third allows us to draw circles on the screen when pressing the mouse. The fourth is an example of how an animation can be created. The fifth shows us three ways to program a piano. And the last one shows us how to create a simple mini game.

4.1. Abstract line drawings

The following code must be written on the **Principal** tab, and when the code is executed, the mouse cursor must be moved to generate the abstract drawings with lines.

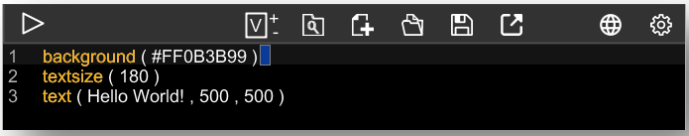


When executed, it will look like this in the browser:

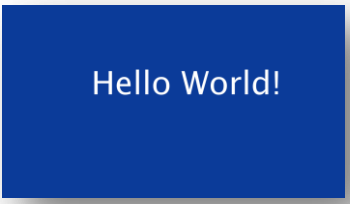


4.2. Basic example with the Keyboard

The following code must be written on the **Keyboard** tab, so it will be executed at the moment any key is pressed.

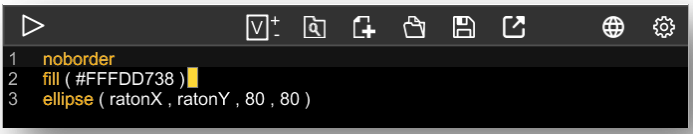


The ideas is that when you press any key, the screen turns blue and a white text appears on the screen.



4.3. Basic example with the Mouse

The following code should be written on the **Mouse** tab so it will be executed at the moment any of the mouse buttons are pressed.

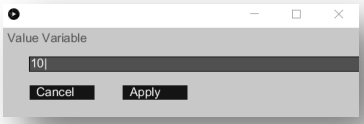


```
1  noborder
2  fill ( #FFFDD738 )
3  ellipse ( ratonX , ratonY , 80 , 80 )
```

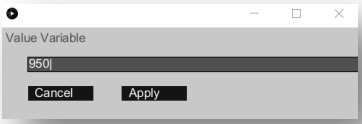
The ideas is that after pressing any of the mouse buttons yellow circles will be drawn on the screen.

4.4. Animation

This is an example of how you can make an animation in Meta_Javascript. First, you need to create two variables, one called **px** and the other called **py**. The **px** variable must be initialized, with the value 10.



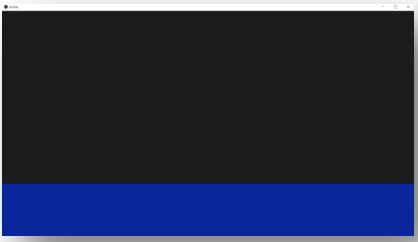
And the variable **py** must be initialized with the value 950.



Then the following code should be added in the **Principal** tab:

```
1  background ( px )
2  add ( px , 0.5 )
3  subtract ( py , 0.5 )
4  fill ( #FFF2E138 )
5  ellipse ( 200 , py , 200 , 200 )
6  fill ( #FF0B259A )
7  rectangle ( 0 , 800 , ancho , 400 )
```

After clicking on the **run** icon you will see that the animation begins by recreating a night scene in the sea, and it will slowly dawn until the firmament is fully illuminated.



4.5. Piano

Below are three examples of how to make a piano in Meta_Javascript. The first is a simple piano, the second is a piano that also changes the color of the screen, and the third is a piano that changes the color of the screen and displays text with the note being played.

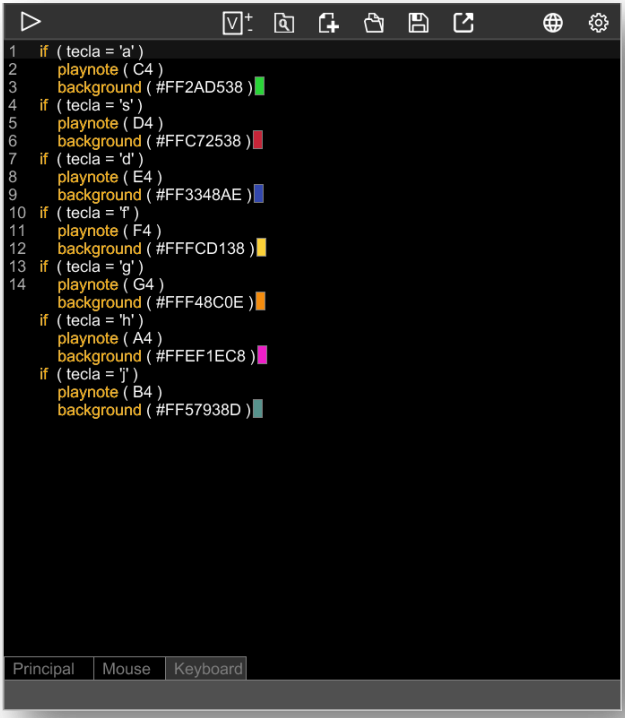
4.5.1. Simple piano

To program this piano, the following code must be added in the **Keyboard** tab:

With this code you can play the 7 musical notes using the keys a, s, d, f, g, h, j. To make it work properly the keyboard cannot be in uppercase mode.

4.5.2. Piano colors

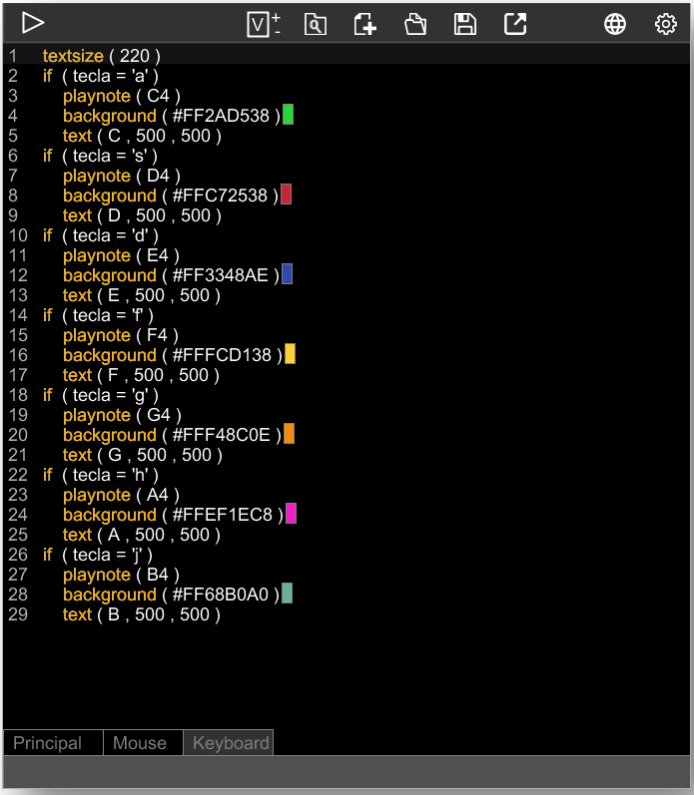
To program this piano, the following code must be added in the **Keyboard** tab:



With this code you can play the 7 musical notes using the keys a, s, d, f, g, h, j. To make it work properly the keyboard cannot be in uppercase mode.

4.5.3. Piano colors and notes on screen

To program this piano, the following code must be added in the **Keyboard** tab:

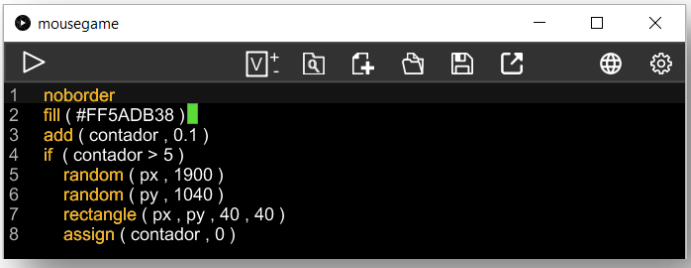


With this code you can play the 7 musical notes using the keys a, s, d, f, g, h, j. To make it work properly the keyboard cannot be in uppercase mode.

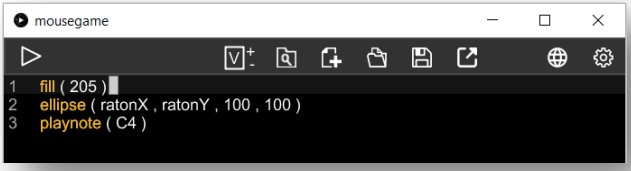
When the piano is executed would look like this:

4.6. Mini Game

This is an example of how you can make a mini game in Meta_Javascript. The following code must be added in the **Principal** tab:

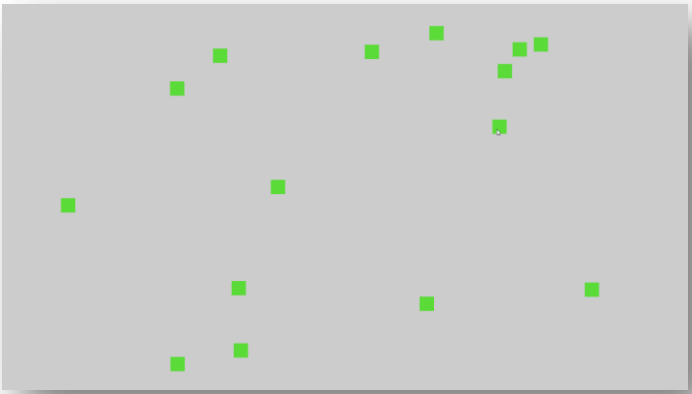


And in the **Mouse** tab you must add the following code:



The game consists of a series of green squares that appear randomly on the screen. The purpose of the game is to try to make disappear all the green squares by clicking on them.

When the game is executed would look like this:



References

Cuartas, J.D. (2014). Digitópolis I: Diseño de Aplicaciones Interactivas para Creativos y Comunicadores. Bogotá: Fundación Universitaria Los Libertadores.

Cuartas, J.D. (2017). Programar el mundo en el contexto de las tecnologías libres y las culturas Hacker-Maker. Caso de estudio: Hitec Lab. (Tesis doctoral). Doctorado en Diseño y Creación. Universidad de Caldas, Manizales.

Hitec Lab (2020). Hitec Lab Homepage. Retrieved from http://hiteclab.libertadores.edu.co/

Hitec Lab (2020). Meta_Javascript. Retrieved from https://github.com/hiteclab/Meta_Javascript

Hitec Lab (2020). Meta_Javascript. Retrieved from https://github.com/hiteclab/Meta_Javascript

Libertadores, L. (2019). Institución Universitaria Los Libertadores. Retrieved from http://www.ulibertadores.edu.co/

Processing. (2019). Processing. Retrieved from http://www.processing.org/

Meta_Javascript, is a meta-programming environment written in Processing language. It was created by the author of this book in order to provide the web application programming beginner with a tool to create code using their native language. It is based on Meta_Processing and was designed to help you write and read the same code in different languages, such as Spanish, French, Hindi, Japanese, Italian, Chinese, Portuguese and English. It´s a programming environment designed to offer a friendly experience, which prevents the user from making common syntax errors. Meta_Javascript is free software released under the GPL v3 license.

What is sought with this derivation of Meta_Processing is to create an ecosystem of applications that allow the user to program different platforms using the same programming language. Today it is almost essential to learn a different language to program each platform.

Professor Jose David Cuartas is the director and founder of the Hypermedia laboratory (Hitec Lab) at the Fundación Universitaria Los Libertadores. He is a Visual Designer and a PhD in Design and Creation from the Universidad de Caldas. He is a promoter of the use and development of free software for environments like art, design and entertainment. His research work focuses on exploring strategies to promote Hacker and Maker cultures in the university academic context, seeking that inexperienced people would discover the creative opportunities that exist when they learn to program. From the Hipermedia laboratory, he is one of the mentors of the initiative called: “Women in technology”, which seeks to bring the female audience closer to the Hacker and Maker cultures and contribute to demystifying technologies. Professor Cuartas is the author of the series of three books called Digitopolis, which is available for free download at google books. The first book in the series (Digitopolis I), allows the reader to learn to program using the Processing language. The second book (Digitopolis II), is a guide to learn how to develop 2D videogames with Gdevelop. And the third book (Digitopolis III), is for those interested in learning how to create virtual tours in Blender 3D.