DIEGATUS STUDIOS CALENDAR

README AND DEVELOPER GUIDE – V 1.4.5

INDEX

ABOUT THE AUTHOR:

- *Programmed and developed by* ***DIEGO VALERO ARÉVALO, DIEGATUS*** *on* ***JavaScript ECMAScript 2020****.*
- *Latest date of User and Developer Guide revision:* ***09/oct/2023****.*
- *Latest released version and date:* ***1.4.5 – 11/oct/2023****.*
- *Project available on GitHub: github.com/Diegatus/calendar.*

AN INTRODUCTION

Welcome to my first public JavaScript project! If you made it to this point, thanks for being interested!

I created this project just as a fun thing to do, to work on JavaScript and practice web programming. PLEASE, TAKE NOTE, I'M A JUNIOR AND STUDENT, so probably not every piece of code will be perfect, but I'm eager to refactor and revise the code every often to fix or improve things, also, feedback is always welcome, so if you see something weird, don't hesitate and contact me!

This project is an extension of a project I've been running for at least 7 years, the **Diegatus Studios Calendar**, a calendar released every year that helps me practice and keep working on graphic design, and also people can download It and print it to use it as is intended. For me, the main point of it is the **events**, special days that are celebrated around the world or in my country, Spain, in some days of the year. They're included as little icons or "emojis" that scatter around the calendar marking when to celebrate or congratulate someone *(Do you know any DJs? 9th March is World DJ Day! Or maybe give your dog a special treat on World Dog Day the 21st July!)*, all of them listed in one page of the calendar as a complete guide of events by month. You can check it out or download it for free at the section Files in diegatus.es/en/downloads.

That's when, as my journey as a web developer starts and grows, I thought of creating a digital events calendar, but updating it by hand is a grotesque task, so I had to think how to implement an automated program that would run it for me. And here we are!

This project, since its beginning and release of the first public version (1.4.5), had a duration of approximately 6 months.

Let's get into it!

SETTING UP THE CALENDAR

TAKE NOTE The repository for this project includes all the **JavaScript scripts files** and an extra **HTML file** named **`calendar.html`** that uses no CSS but serves as an example of the results of the calendar inner workings, so you can use it as a base to start building your own calendar design.

IMPORTANT: TO LOAD THE SCRIPTS INTO YOUR HTML YOU SHOULD WRITE THEM IN THIS SPECIFIC ORDER:

```
<script type="text/javascript" src="hideMissingFileIcon.js" defer></script>
<script type="text/javascript" src="V_X_calendarLogic.js" defer></script>
<script type="text/javascript" src="V_X_calendarEvents.js" defer></script>
<script type="text/javascript" src="V_X_calendarMoons.js" defer></script>
<script type="text/javascript" src="calendarLANG.js" defer></script>
```

Where `V_X` corresponds to the version of the script (1.4.5 = `V_1_4_5_...`) AND `calendarLANG.js` corresponds to the version of language you want to use: Spanish (`calendarES.js`) or English (`calendarEN.js`).

This calendar works with **3 main containers** and **1 optional container** that you should have in your web page so every piece of code works as intended. This guide will use the model coded at diegatus.es/en/calendar (named as "**the page**" from this point).

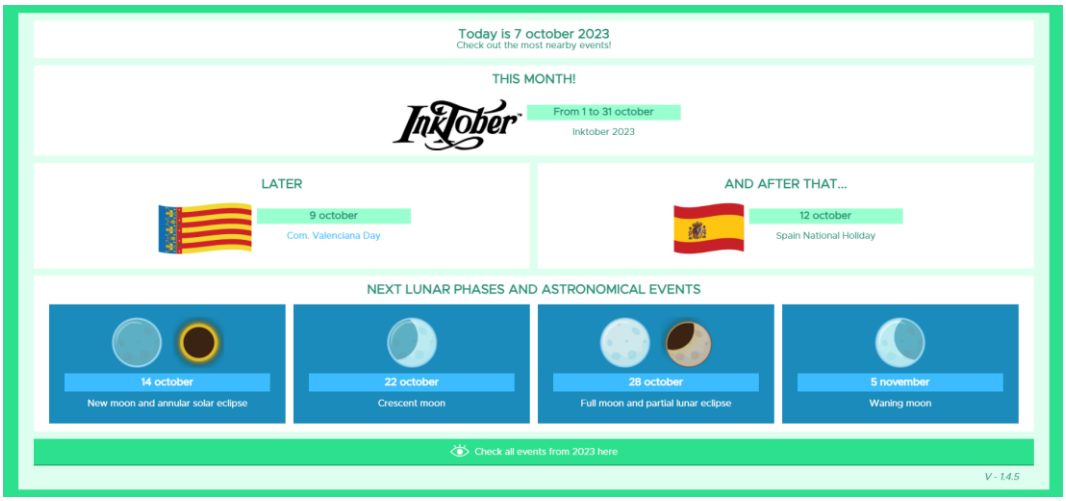
TAKE NOTE The different texts and numbers are assigned through the use of **IDs** on HTML tags that you can check on the following table:

				OPTIONAL*
	1. Global date	**2. Date for long term events**	**3. Dates for main events (2): X => A or B**	**4. Dates for moon phases and astronomical events (4): X => 1, 2, 3 or 4**
USE	Shows the current date, built by number of day, string of month and number of year.	Shows events that have a duration of more than 1 day. It **hides** or **reveals** only when a long term event is active.	Shows the nearest and next nearest event days respective to the current day.	Shows the four next moon phases or astronomical events, in order, that will take place from the current day.
Container		`calendarC`		
When is it happening?		`calendarCTitle`	`calendarXTitle`	
Icon 1 (main)		`calendarCImg1`	`calendarXImg1`	`calendarMoonXImg1`
Icon 2			`calendarXImg2`	`calendarMoonXImg2`
Icon 3			`calendarXImg3`	
Date + month	`calendarGlobalDate`	`calendarCDate`	`calendarXDate`	`calendarMoonXDate`
Event name		`calendarCDesc`	`calendarXDesc`	`calendarMoonXDesc`

*The calendar for moon phases and astronomical events is not essential for the main calendar to work. All its logic and main functions are stored on its own script (**`V_X_calendarMoons.js`**), and because most of the basic functions that it uses are stored on **`V_X_calendarLogic.js`**, you don't have to use it if you don't want to.

VISUAL DISTRIBUTION EXAMPLE AND CALENDAR ELEMENTS

An example of elements distribution at the page would be the following one:

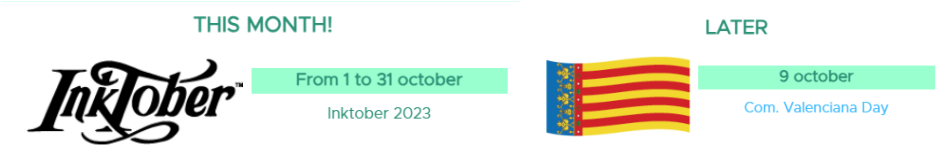


1. GLOBAL DATE

1A **“custom message”**

1A: Title that shows the current day that matches the current date & time (and by proxy, the UTC offset/locale) from the client's local environment. It shows the text **Today is**, the **number of day**, the **name of the month** that corresponds and the **number of the current year**. Under it you can add a **custom message** inside a paragraph tag (**<p>**) like in the example.

2. LONG DURATION EVENTS AND EVENTS DATES



2A	
2B1 2B2 2B3	2C
	2D

2A: Title that shows the proximity of the event. For **day A**, it will show:
- **LATER** if the day returned is not on a distance of 1 or less about the current day.
- **TOMORROW** if the day returned is on a distance of 1 about the current day.
- **TODAY!** If the day returned is equal to the current day.

For **day B** it will show:
- **AND AFTER THAT...** if the number of day returned is not on a distance of 1 or less about dayA.

- **AND TOMORROW...** if the day returned is on a distance of 1 or less about dayA and dayA is equal to the current day (dayA should be showing TODAY!).
- **AND THE NEXT DAY...** if the day returned is the very next day to dayA but dayA is not the current day (dayA should be showing TOMORROW).

For long duration events (**display C**), depending on the event that is stored manually on **`V_X_calendarEvents.js`**, it will show **THIS WEEK!** or **THIS MONTH!**.

2B: The icon or "emoji" representing the event that corresponds to the number of day returned, they'll be assigned a **src** value. Because of the fact that some events share the same day, there are 3 slots for images. **2B1** will always show an image, but **2B2** and **2B3** will hide if the **src** value is empty (see **`hideMissingFileIcon.js`**).

2C: The number of day returned and the name of the month that corresponds for it.

2D: The name of the event that corresponds to the number of day returned.

3. MOON PHASES AND ASTRONOMICAL EVENTS DATES

3A1 3AB
3B
3C

3A: The icon or "emoji" representing the moon phase or astronomical event that corresponds to the number of day returned, they'll be assigned a **src** value. Because of the fact that some events share the same day, there are 2 slots for images. **3A1** will always show an image, but **3A2** will hide if the **src** value is empty (see **`hideMissingFileIcon.js`**).

3B: The number of day returned and the name of the month that corresponds for it.

3C: The name of the moon phase and, if corresponds, the name of the astronomical event that corresponds to the number of day returned.

CODE FLOW

Now that we have set up the containers with their respective IDs, it's time we start to comprehend how the code of the calendar works for each element.

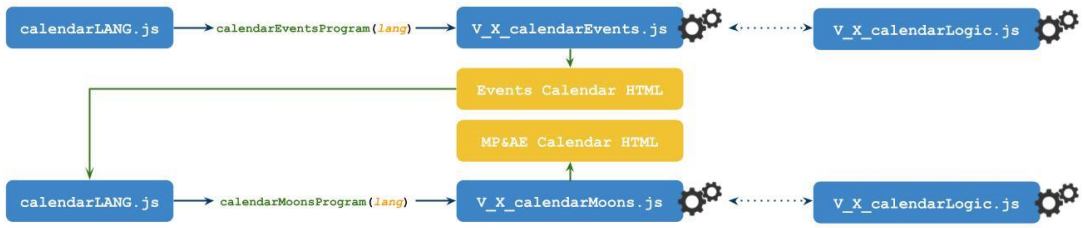
The main script is **`calendarLANG.js`**, that calls two **main functions**: **`calendarEventsProgram()`** and **`calendarMoonsProgram()`**, that must have a parameter **`lang`** that relates to the language of the text returned.

TAKE NOTE	The main functions have been coded this way because of the **language feature** to return values in one of this two options: **`"es"`** for **Spanish** or **`"en"`** for **English**. If you don't want to use this feature, check the paragraph ***ABOUT LANGUAGES***.

TAKE NOTE	Texts are **NOT** translated automatically, you have to set each option and text manually on the functions that need them.

CODE FLOW BETWEEN SCRIPTS

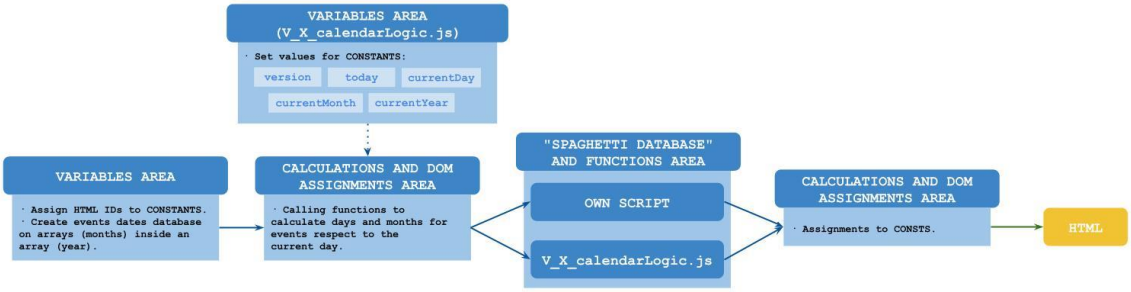
The first function to be called on **`calendarLANG.js`** will be **`calendarEventsProgram()`** and will calculate every data needed to be returned and assigned on the HTML **(see the next paragraph, *CODE FLOW INSIDE THE SCRIPTS*, for more information)**. Once all the data has been calculated and assigned to the **events calendar**, the next program to be called will be **`calendarMoonsProgram()`** for the **moon phases and astronomical events calendar (`MP&AE`)**, which working is very similar to **`calendarEventsProgram()`**.



CODE FLOW BETWEEN SCRIPTS

CODE FLOW INSIDE THE SCRIPTS

Inside each main function, it'll assign **HTML IDs** to constants and create the **events database** on the **"VARIABLES AREA"**, and it will continue through the **"CALCULATIONS AND DOM ASSIGNMENTS AREA"**. Some functions will be on the own script but others are called to the script **`V_X_calendarLogic.js`**, both on a respective area called **""SPAGHETTI DATABASE" AND FUNCTIONS AREA"**.



CODE FLOW INSIDE A SCRIPT

The **constants** **`version`**, **`today`**, **`currentDay`**, **`currentMonth`** and **`currentYear`** that are used on many of the functions are called from **`V_X_calendarLogic.js`**.

You can check every **function use**, **parameters** and **return values** on each function **JSDoc** after the function itself.

This code flow explains why we have to load the files in the order listed on the paragraph ***AN INTRODUCTION***, because those who are executed first can't work if some variables and values from other scripts haven't been loaded yet.

ABOUT THE LANGUAGE FEATURE

If you don't want to use this feature you'd have to change a massive part of the code, so it's recommended that you use the parameter value `"es"` and just **modify the information of the first parameter of all code and functions that use the function `checkLanguage()`** and **the code and functions that return its values based on language**. I know it's not a short task, but it's better than rewrite the majority of the databases code.