

# Running Splits Calculator - Documentation

By, Isabelle Bernal

## I. Purpose

- This website should be able to calculate your splits for a certain event. Here are some examples
  - Running 1600m (about 1 mile) at 6 min pace and splits are given every 400m
    - 400m = 1:29.48
    - 800m = 2:58.95
    - 1200m = 4:28.43
    - 1600m = 5:57.91
  - The split should tell you that you have to run every 400m in about 1 min and 29.48 sec.

Total Distance:1600meters▼orselect distance▼

Total Time:

hr :

min :

sec

clear time&pace

or

Pace:6:00min/mile▼m:s or m.m format

Splits Every:400meter▼

Calculate Splits (this window)

Calculate Splits (new window)

In order to run 1600 me  
at 6:00 min/mi  
your split every 400 me must be:

Split Distance	Split Time	Total Distance	Total Time
400 me	1:29.48	400 me	1:29.48
400 me	1:29.48	800 me	2:58.95
400 me	1:29.48	1200 me	4:28.43
400 me	1:29.48	1600 me	5:57.91

Run Well!

The picture above was taken from this website: [Splits Calculator](#), useful for what user input is needed but should look more unique and creative.

## II. Basic Design:

“Find your splits for an event” box

“Total Distance”	“Select distance” (?) (dec)	Drop down box (meters/miles)
Total time:	Hr	min
Pace:	sec	min/mile
Splits every: _____ meter/mile		

Output

Total distance	Total time
400 meters	1:30
800 meters	3:00

ETC ETC

## III. Testing and Calculation

1. Figuring out Calculation

bank for fee! if (fee:

Calculation: mile to meter

① total time → secs : miles → meters

② divide <sup>mileage</sup> ~~secs~~ equally for every split = # of splits total

③ divide secs by total # of splits

ex.

Total Distance: 10 miles + 16,093.4 meters

Total time: 1 hr 30 min 22 sec + 5,422 secs

Splits every: 400 meters

400 m 2:14.77

800 m 4:29.54

16093.4 meter : 1 mile

1 meter : 0.000621371 miles

16093.4 / 400

40.23

5422 / 40.23

134.775

2 min 15.55

2 min : 14.77

Calculation: Meter to miles

① total time → secs : meter → miles

Total Distance: ~~10 miles~~ 1 meter : 0.000621371 miles

4900 meters / 1609.34 = 3.04 miles

Total time: 20 mins 27 sec + 1227 secs

Splits every: 1 mile

3.04 / 1 = 3.04

1227 / 3.04 = 403.618421

= 403.618421

1 mile = 6 min : 42.9988

Calculation: Meter to Meter

① total time → secs : Meter → meter

Total Distance: 2500 meters

Total Time: 14 min 7 sec + 847 secs

Splits every: 500 meters

2500 / 500 = 5

847 / 5 = 169.4

secs

500 m = 2 min : 49.4 secs

500 m = 5 min : 39.8 sec

Calculation miles to miles

① Total Distance: 10 miles

Total Time: 1hr 20 minutes 23 secs

Splits: every 2 miles

② convert time to secs  $80 \text{ min} \cdot 60 = 4800 + 23 = 4823 \text{ secs}$

$$4823 / (10/2) = 964.6$$

2 miles 16 min : 46 secs

4 miles 32 min : 9.2 sec

6 miles 48 min : 13.8 sec

8 miles 64 min : 18.4 sec

10 miles 80 min : 23 sec

num1 = dist num2 = hr:min:sec num3 = splits

num3 = pace

var name = document.getElementById("...").value;

2. Documentation on Process of Coding/Testing the Math Portion and Other Challenges:

I decided to approach the splits calculator by first coding it in a language where I feel confident which was Java and then taking that foundation to help me build my code in Javascript. This was very helpful because I was able to write and test my code very quickly. If you compare my Java and Javascript code you can definitely tell I added more to my Javascript code as I was thinking of more and more scenarios that could possibly break my code, I added a little more to the math portion in regards to when the user inputs pace or total time, and I had to add a couple more lines of code to help with the formatting of the output. However, I needed that foundation from my Java code to help me be as efficient as possible with coding in Javascript.

From coding the math portion of the splits calculator in Java to later translating it into javascript to put on the webpage, I have now realized that Javascript and Java may have some more differences than I thought. For instance, javascript only has three data types: const, var, and let, which I found can make it difficult to have an int data type for one specific number. Another major difference is the way you print in Java is way different in javascript and required me to go through a lot of trial and error to figure out how to do it. Once I was able to figure out how to print on the webpage in javascript it was really just an extra step in the beginning of my code and changing the syntax. The last major difference that stood out to me was the user input because not only did I have to change the syntax but since there are only three data types in Javascript I had to make sure to change any string from the user input into an actual number. Using `parseInt()` was very useful throughout my program due to this.

Although there were some major differences between Java to Javascript there were still a couple of similarities that made my Java code very useful. For instance, Javascript and Java have the same syntax when it comes to for loops and if statements so I was able to have a good foundation going into my javascript code. Another similarity is the math operators and as long as all the user inputs were converted into numbers correctly and not left as strings, I was able to move over the majority of the math portion from my Java code into my Javascript code. Converting the user inputs into numbers was very important because if the user inputs are left as strings it can mess up your entire code since it will just put the numbers side by side instead of executing the right math operation.

In my program, I was originally going to make only one text box for each user input but I found it difficult in javascript to grab the times as one string and then try to convert it into a number since the string would be “mm:ss” format. To add on, there is not one singular number in that string there are two numbers and two colons so I decided to make a text box for each number inputted. This made converting from the string value to a number much easier and made calculating easier as well. Something else I wanted to point out as well is that I originally was not going to have a refresh button but I decided to add one that will refresh the page entirely because I realized I did not account for if the user wants to start over after clicking the button. I feel that the refresh button is also convenient as well if the user decides they want to start over with the values they inputted but don't want to go through each text box to change it.

#### **IV. Task Timeline:**

- 1-2 weeks to work on design
- 1 week math/logic/other coding

Soft due date: (august 14)

- Gives time to do testing/work out any bugs

Real Due date: before school starts (august 29)

Whats left: (Done)

-> if there is invalid input after alert it needs to terminate

-> after one full splits calculation given needs to terminate so it can't keep pressing the button

(needs to refresh if wants to enter splits again)

-> add box at the bottom of the code with my print statements at the bottom

What is left design-wise: (Done)

-> name

-> background

-> upload to github