the Master Course

{CODENATION}

JAVASCRIPT FUNDAMENTALS

Dot Notation

{CODENATION}

Learning Objectives

To understand what Dot Notation is.

To understand different Data Types

To be able to create a simple Random Number Generator Program

JS

What is Javascript?

The language of the Full Stack Developer

& not just limited to websites!

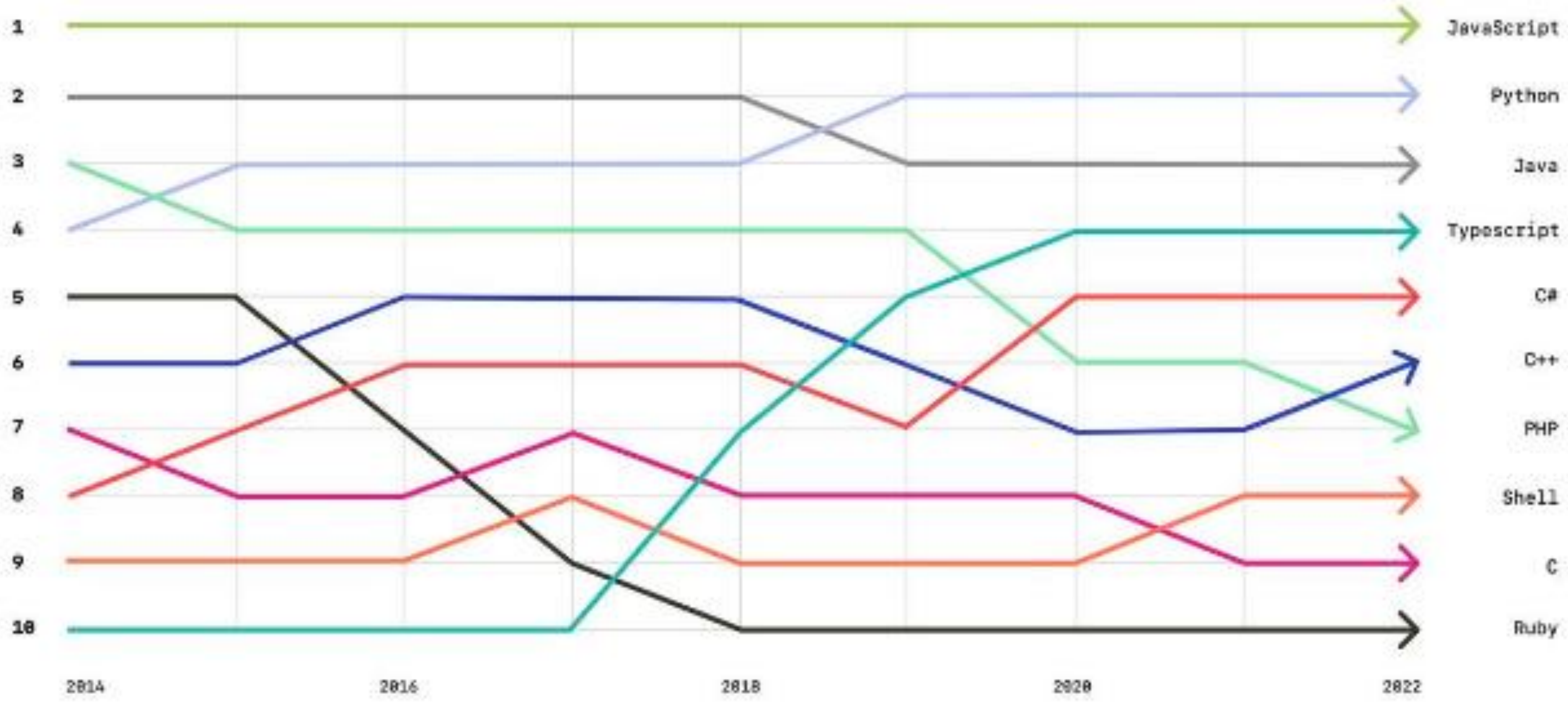
{CODENATION}

JS

It’s also the most popular language in the world

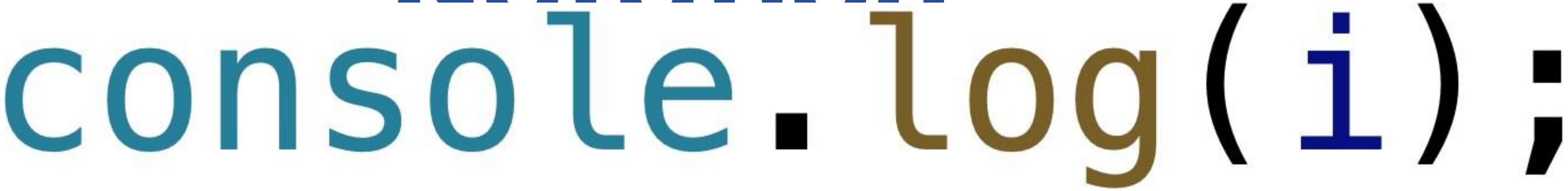
{CODENATION}

https://solutionshub.epam.com/blog/post/programming-language-popularity-on-github



Top programming languages on GitHub in 2022 (Source: GitHub)

Dot Notation



```
console.log(i);
```

Dot

```
console.log(i);

object.property
```

```
car.startEngine();

dev.makeCoffee();
```

If we break it down, coding in its simplest and snappiest is all about METHODS and DATA.

So this is where we're going to start.

JS

METHODS and DATA

...are **intimately** linked

{CODENATION}

JS

Lets look at some...

Data Types

{CODENATION}

But hang on...

.. what **ARE** data types?

Working with Data Types

Data Types refer to the kind of data that we are asking the computer to work with.

Simple, right?

Strings

… for representing **text**

Boolean

… for **true** and **false**

Numbers

… for representing **numbers** (decimals & integers)

Null

… for **nothing**

Undefined

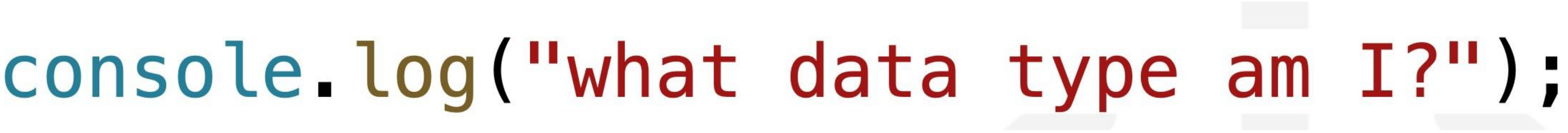
… for when a data type **isn't determined**

Symbol

… this data type is used as the key for an object property when the property is intended to be private.

JS

What data type am I?



```
console.log("what data type am I?");
```

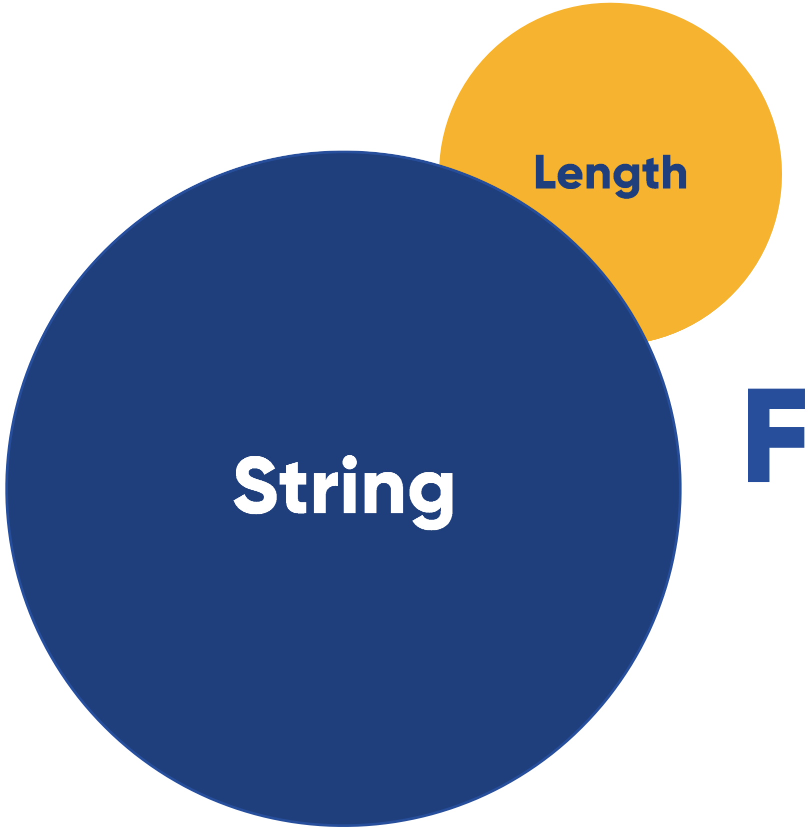
{CODENATION}

String

```
console.log("what data type am I?");
```

All data has properties

… additional data or information that is available?

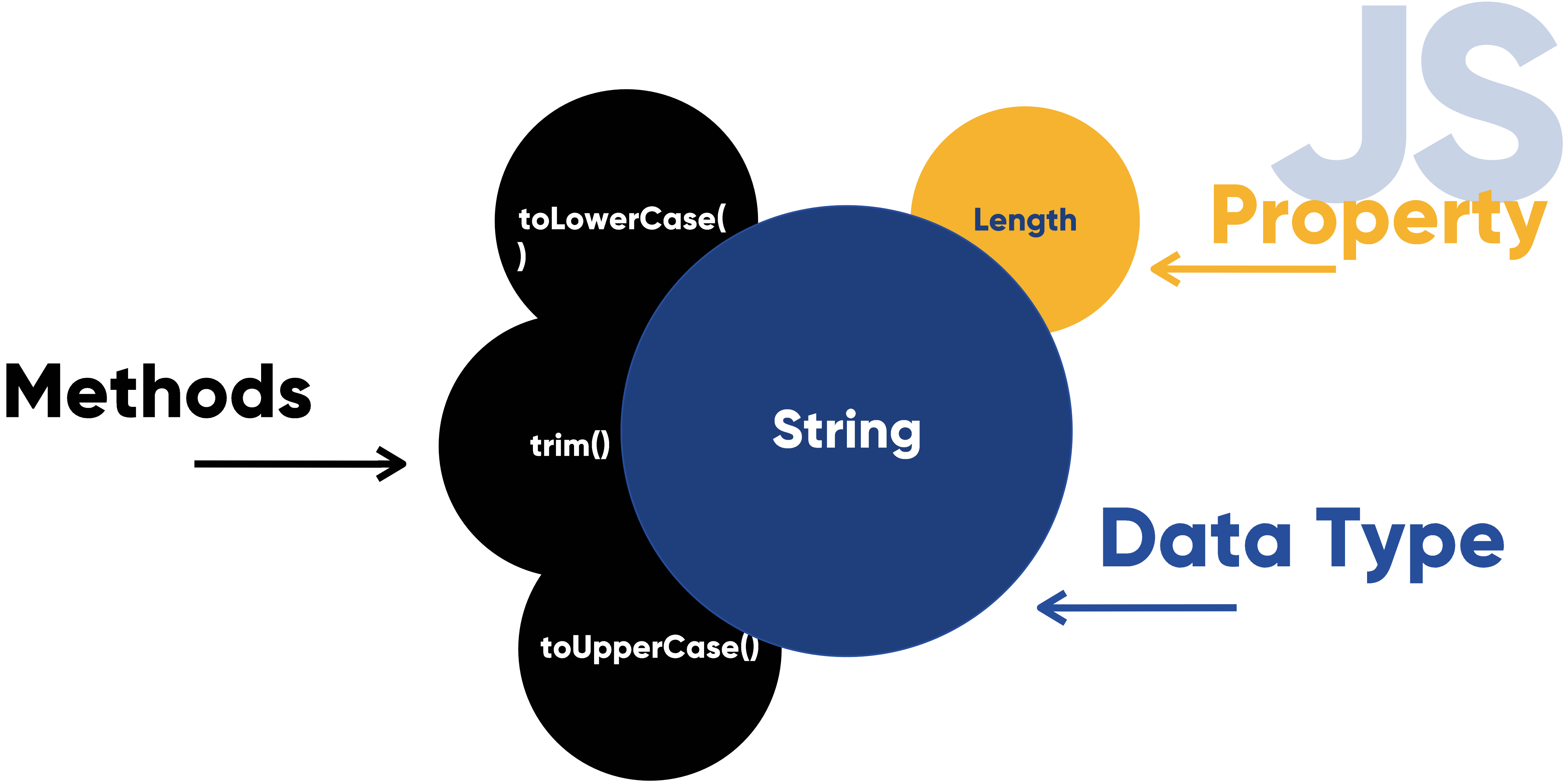


For example ‘length’

This will tell us **how long** the string is

All data also has methods

… methods allow us to **manipulate** the data type


JS
toLowerCase(
)
Length
Property
Methods
trim()
String
Data Type
toUpperCase()

{CODENATION}

The Terminal

cd, ls, mkdir, touch,
rm

Node.js

http://nodejs.org

Activity

Using what you have just learnt I want you to use the terminal to go inside of your '**CodeNation' folder made in week 1**.

Inside that folder, create another folder called **'Week2'.**

Inside that folder, create a file called **'dotNotation.js'**

You can ONLY use the Terminal

{CODENATION}

Try this

JS

Hello World!

```
console.log(“Hello World”)
```

is node working?

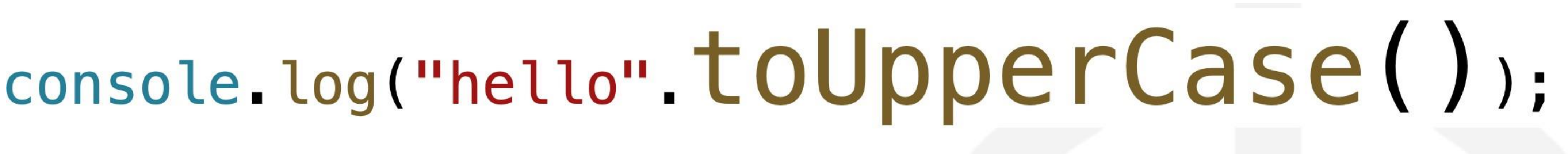
...in your terminal, type in node **dotNotation.js**

{CODENATION}

Why do we even need node?!

Node allows us to **run our Javascript code in our own Terminal**. Without this, we'd have to set up multiple files, use Google Chrome's Console and link up our files.

Try this



```
console.log("hello".toUpperCase());
```

...what happens?

Well done!

... you just used your first string method!

Shh! Libraries

In coding, libraries give us access to a **bunch of features** that thankfully we don’t have to code ourselves!

So far…

Dot notation

```
console.log("Hello");
```

Parameters

… we've stuck to the console library when using **console.log** and that's about it.

Try this

Math Library

Dot notation

```
console.log(Math.random());
```

Parameters

What happens?

How can we make this better?

Try this

Dot notation

```
console.log(Math.random()*10);
```

Parameters

What happens?

How can we make this EVEN better?

JS

MDN Web Docs

moz://a

Go to the link for **MDN Math Library** and find out **HOW** we could round this number down.

Dot notation

```
console.log(Math.floor(Math.random()*10));
```

Parameters

Math.floor

...returns an integer **less than or equal** to the specified number

Math.ceil()

... will always **round a number UP** to the next largest integer

Math.round( )

... returns the value of a number rounded **to the nearest integer**

Math.floor()

...returns an integer **less than or equal** to the specified number

{CODENATION}

Learning Objectives

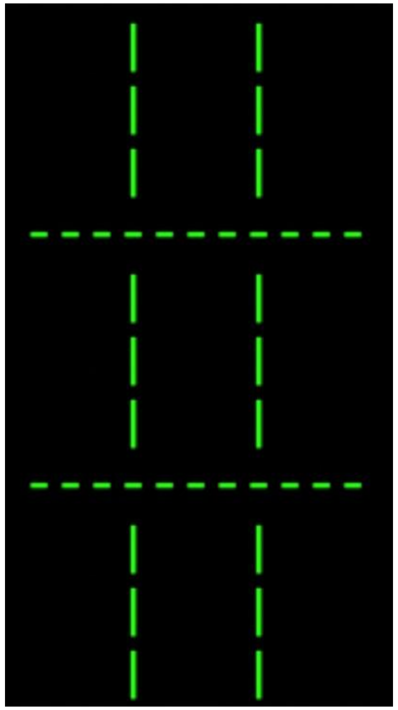
To understand what Dot Notation is.

To understand different Data Types

To be able to create a simple Random Number Generator Program

Activity

Have a go at **logging a grid** like this to the console.



```
   |   |
   |   |
   |   |
-----------
   |   |
   |   |
   |   |
-----------
   |   |
   |   |
   |   |
```

Stretch

If you figure it out, try researching **arrays** and **loops** and see if you can do it that way.

For next time...

... take a look at **variables** and **mathematical operators.**

https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/Variables

https://www.youtube.com/watch?v=XgSjoHgy3Rk

https://www.w3schools.com/js/js_arithmetic.asp

What is the difference between **let** & **const?**
What mathematical operator **returns the remainder**?

{CODENATION}