LESSON 1: DATA & VARIABLES

PRIMITIVE

- Simple to use and understand
- Dependable, always there for you
- Is the foundation for every system, no matter how complex.

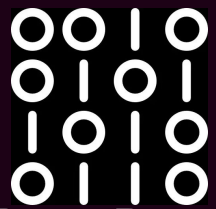
4

Restricts what you can store in this block to an accepted, interpretable range.

TYPE

programming language

Given to you by OS
32 or 64 bit blocks
stored in RAM.



Raw Data

COMPLEX

- Powerful ability to systemize ideas.
- Sparkly, shiny, alluring.
- Not very nice to newbies, comes at a cost.

2

DATA TYPES 1-3

int

Whole numbers

float

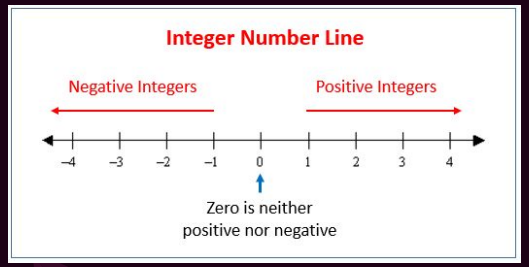
Whole and decimal numbers

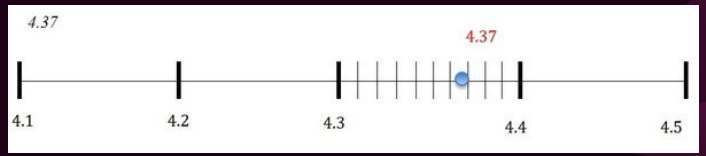
(but has drawbacks)

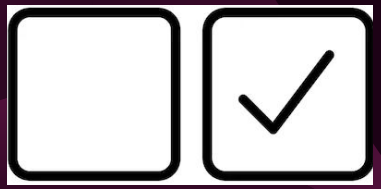
bool

true or false

(and that's it)



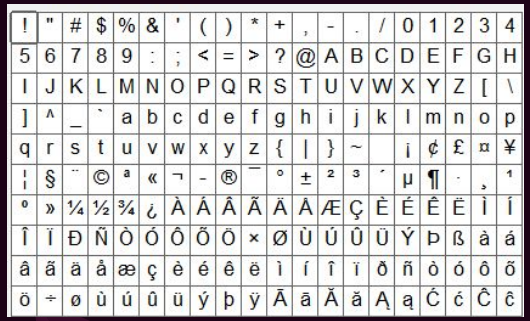




DATA TYPES 4-6

String

Text

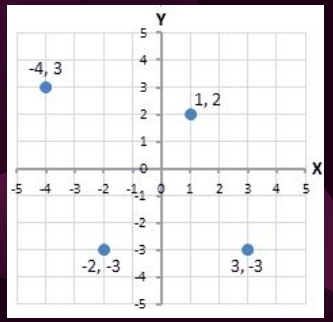


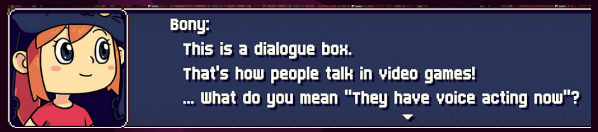
enum

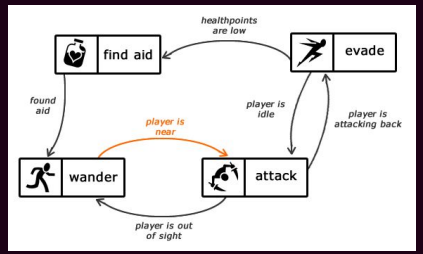
A category of options defined by the programmer

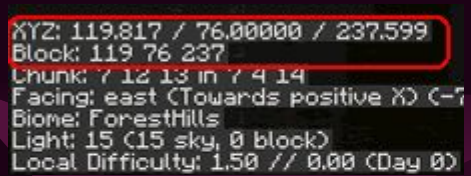
Vector2

2 floats, called “X” and “Y”

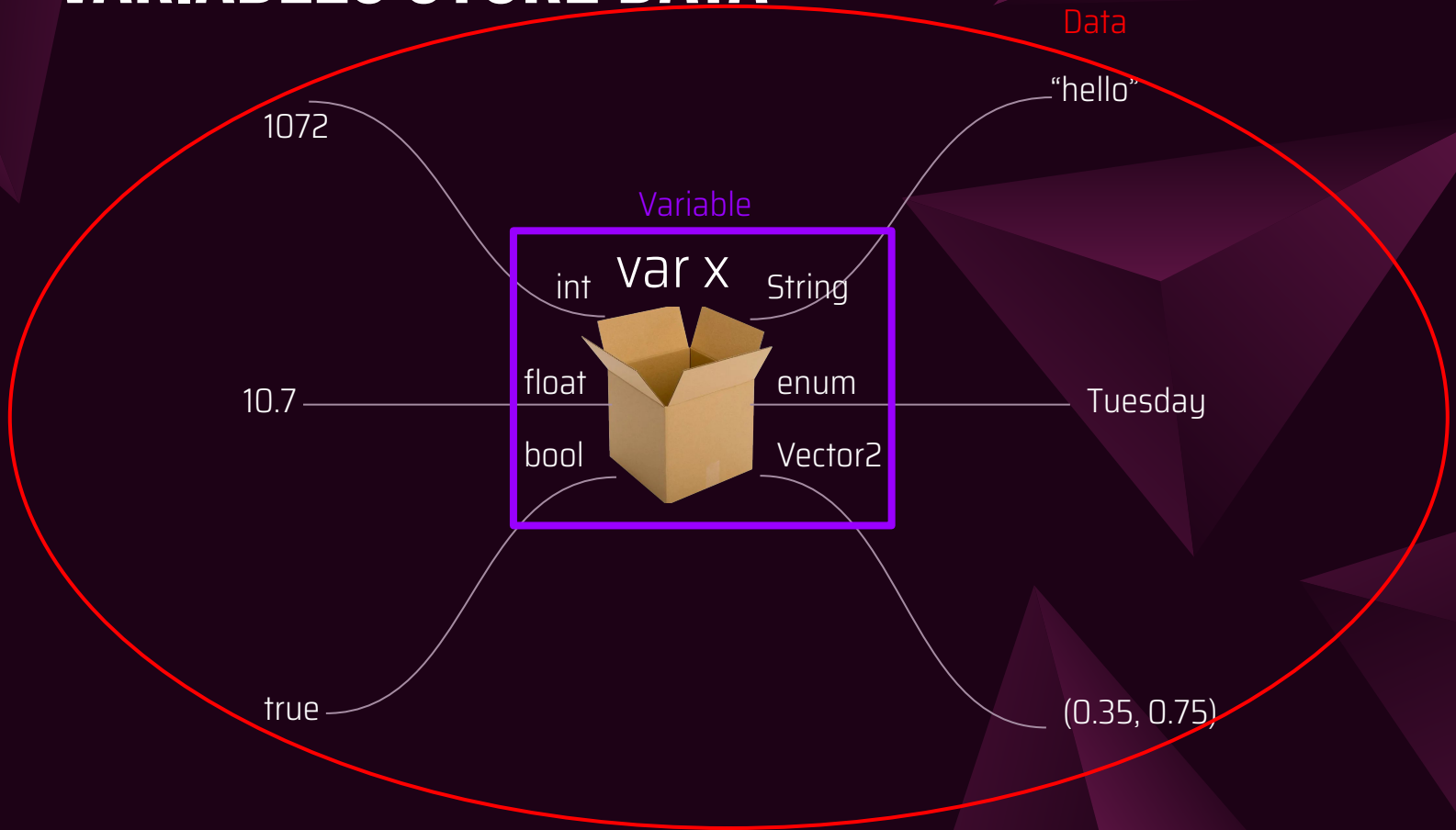








VARIABLES STORE DATA



VARIABLE DECLARATION

How do I create a variable to store some data?

Normally you see it like this:

```
enum EightBall {
    Yes,
    MostLikely,
    OutlookGood,
    AskAgainLater,
    ReplyHazyTryAgain,
    No,
    VeryDoubtful,
    DontCountOnIt
}

var iMadeAnInt    :int     = 5
var heresAFloat   :float   = 1.5
var boringBool    :bool    = false
var aString       :String  = "data"
var enumsAreHard  :int     = EightBall.AskAgainLater
var position      :Vector2 = Vector2(0.5, 0.5)
```

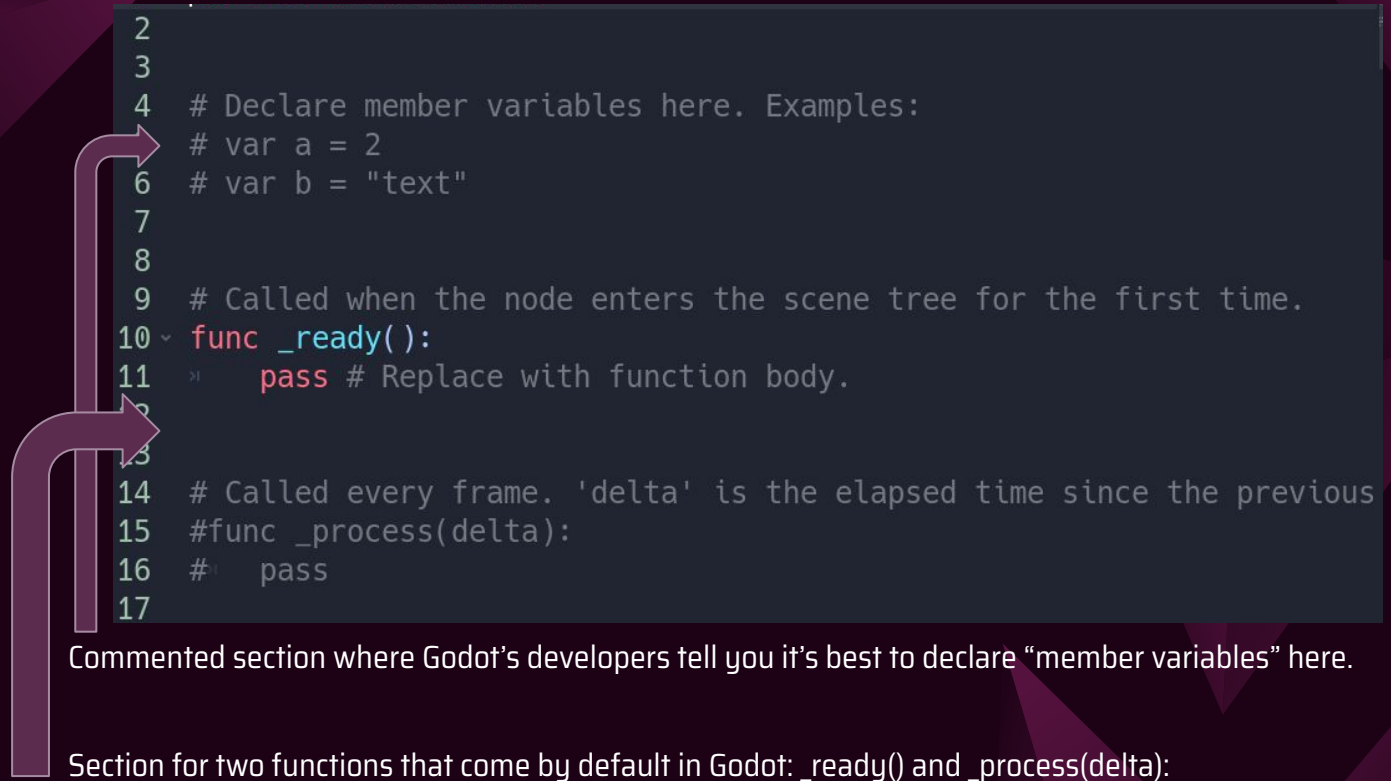
One of each variable type:

General form distilled into...

Form to declare a variable `var name :type = data`

^and this is optional

GODOT STUFF



Commented section where Godot's developers tell you it's best to declare "member variables" here.

Section for two functions that come by default in Godot: _ready() and _process(delta):

Implies the questions: What's a "member variable"? What happens if I declare a variable in ready or process?

VARIABLE DECLARATION 2

```
1  extends Node2D
2
3
4  var memberVariable = 1
5
6
7  func _ready():
8      var functionVariable = "uwu"
9
10     functionVariable += memberVariable
11
12     memberVariable = 10
13
14     pass
15
16 var memberVariableTwo = memberVariable + 10
17
18 memberVariableTwo += 10
19
20
21
22
23
24 #End of script
```

In Godot, most of the script you write must be written inside of a function--except for variable declaration.

Vars declared outside of a function are called “Member Variables,” and vars declared inside are “Function Variables.”

Vars only need to be declared once, then they can be referred to without the ‘var’ prefix word.

Member vs Function Variables

Member Variables

- Shared through every function in a script.
- Can be exposed to be accessible by other scripts.
- Can be exported to be settable through the Godot editor.

Function Variables

- Only usable in the function that created it.
- Is destroyed when the function is over
- Generally regarded as temporary holding zones for calculations and processing.

Open Godot and show instantiating a member vs function variable.

EXPORTING MEMBER VARIABLES

```
export var iMadeAnInt   :int    = 5
export var heresAFloat  :float  = 1.5
export var boringBool   :bool   = false
export var aString      :String = "data"
export var enumsAreHard :int    = EightBall.AskAgainLater
export var position     :Vector2= Vector2(0.5, 0.5)
```

Exporting member variables puts them in the Godot inspector. There, you can use GUI fields to set the initial state of that variable. Useful for...

- Variables you need to change frequently to tune game design
- Entities you're going to clone often and want different starting parameters on some instances

VARIABLE SCOPE

```gdscript
extends Node2D


var memberVariable = 1


func _ready():
    var functionVariable = "uwu"

    functionVariable += memberVariable

    memberVariable = 10

    pass

var memberVariableTwo = memberVariable + 10









#End of script
```

Variables track the line and indentation level they were declared on.

Variables cannot be used above the line they were created on, and they are deleted when the indentation level they were created on moves back to the left.

OPERATORS

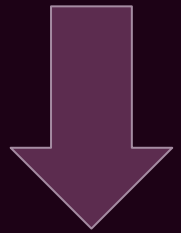
Operators are commonly used symbols that perform... well, operations, on variables.

We will be learning more operators as they become relevant (this is not an exhaustive list).

Name	Symbol	Applies to	Description
Set	=	All	Sets the value of the var E.g: x = 7, x = “foo”
Math	+ - * /	Numeric	Performs math E.g: x = 72*126, x = y + z
Modulo	%	Numeric	Divides, but returns remainder. E.g: x = 7 % 3
Append or Increment	+= -= *= /=	Numeric	Performs math, but uses the var itself as the left side of the symbol. x += 7, x *= 2, Are identical to... x = x + 7, x = x * 2

PRINT

```
print("My member variable is:", memberVariable)
print("My function variable is:", functionVariable)
```



Output:

```
--- Debugging process started ---
Godot Engine v3.2.1.stable.official - https://godotengine.org
OpenGL ES 3.0 Renderer: AMD Radeon R9 200 Series

My member variable is:1
My function variable is:uwu
```

Output Debugger Search Results Audio Animation

CHALLENGES

1: Declare a string, an int, and a float Member Variable using the ‘export’ tag, then save your script and switch to the editor to view your input fields.

2: In the _ready function, declare a typeless function variable. Set your function variable equal to one of the two variables from part 1. Then, use the += operator to append the second variable from part one. Finally, print the function variable.