



# PONY PROGRAMMING LANGUAGE

Presented by Winnie Delinois

# AGENDA

## INTRODUCTION



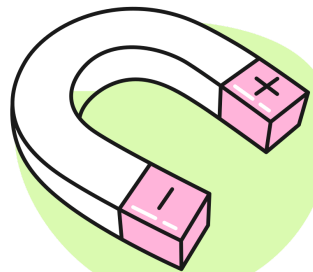
What is Pony? How can it be installed?

## LEARNING LANGUAGE



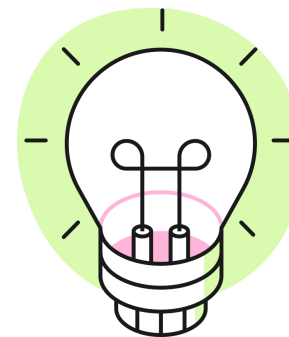
Sample code

## NEXT



Efficiency?

## CONCLUSION



Pros and cons of Pony?  
Comparison to other languages? Would I recommend it?

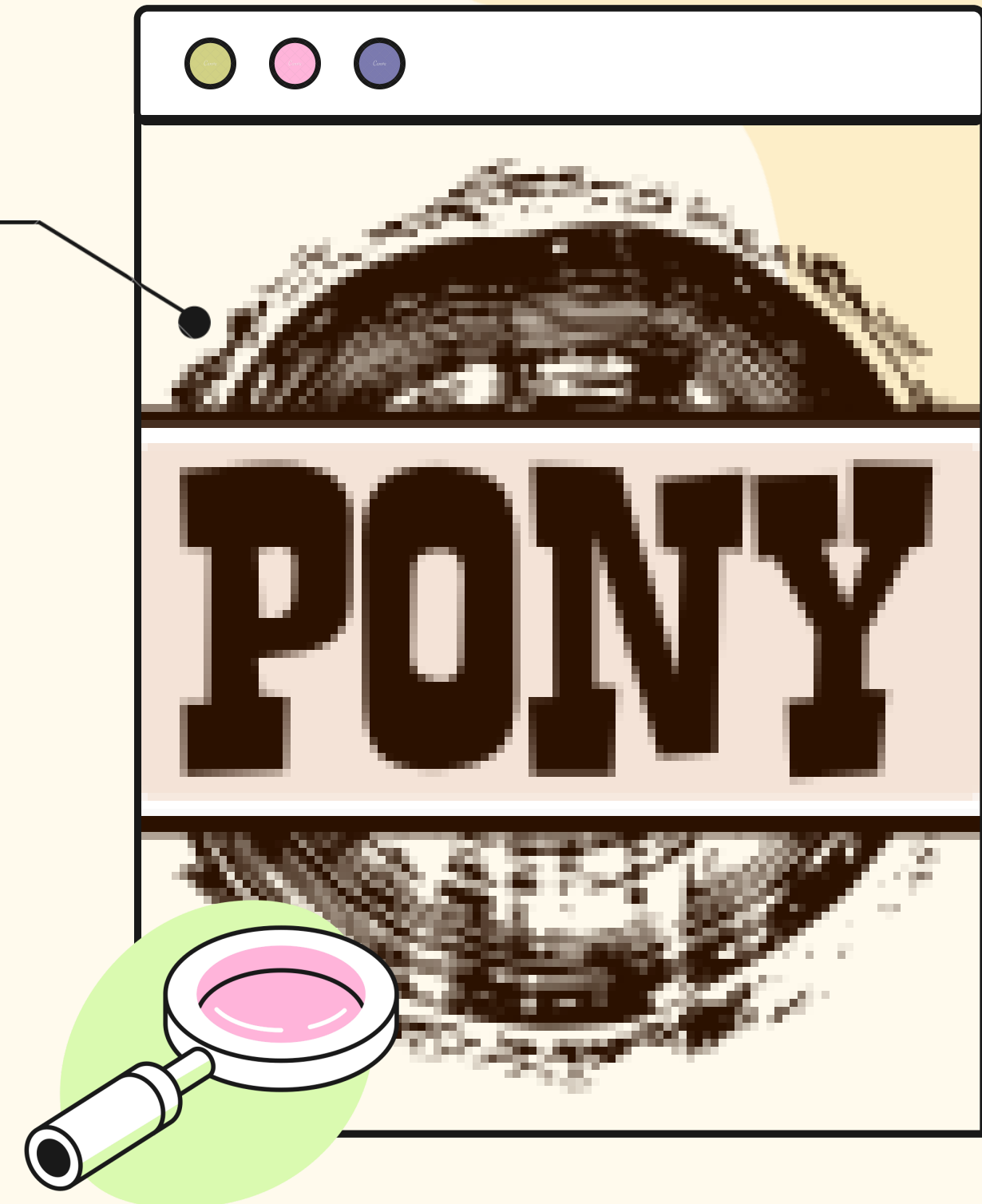
# INTRODUCTION

## WHAT IS PONY?

As described by the official website, Pony is an "open source, object-oriented, actor model, capabilities-secure, high performance" programming language.

## DESCRIPTION AND CLAIMS?

Pony claims to be "type safe, memory safe, and data-race free". The language also uses native code and is compatible with the language C, since it can call its languages using the foreign function interface.



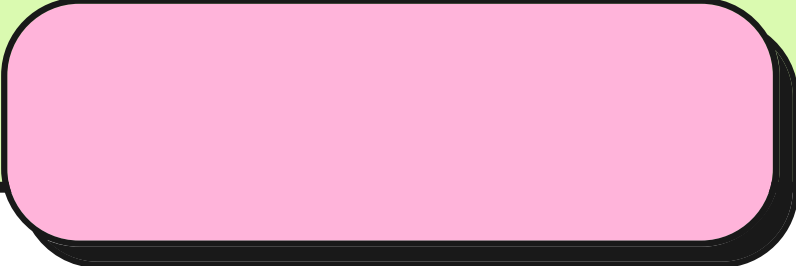
# INTRODUCTION

## INSTALLATION?

Instruction to install Pony can be found on the official website. It provides options for multiple devices such as Windows, Linux, and Mac. For this project, using the compiler on my browser would be sufficient.



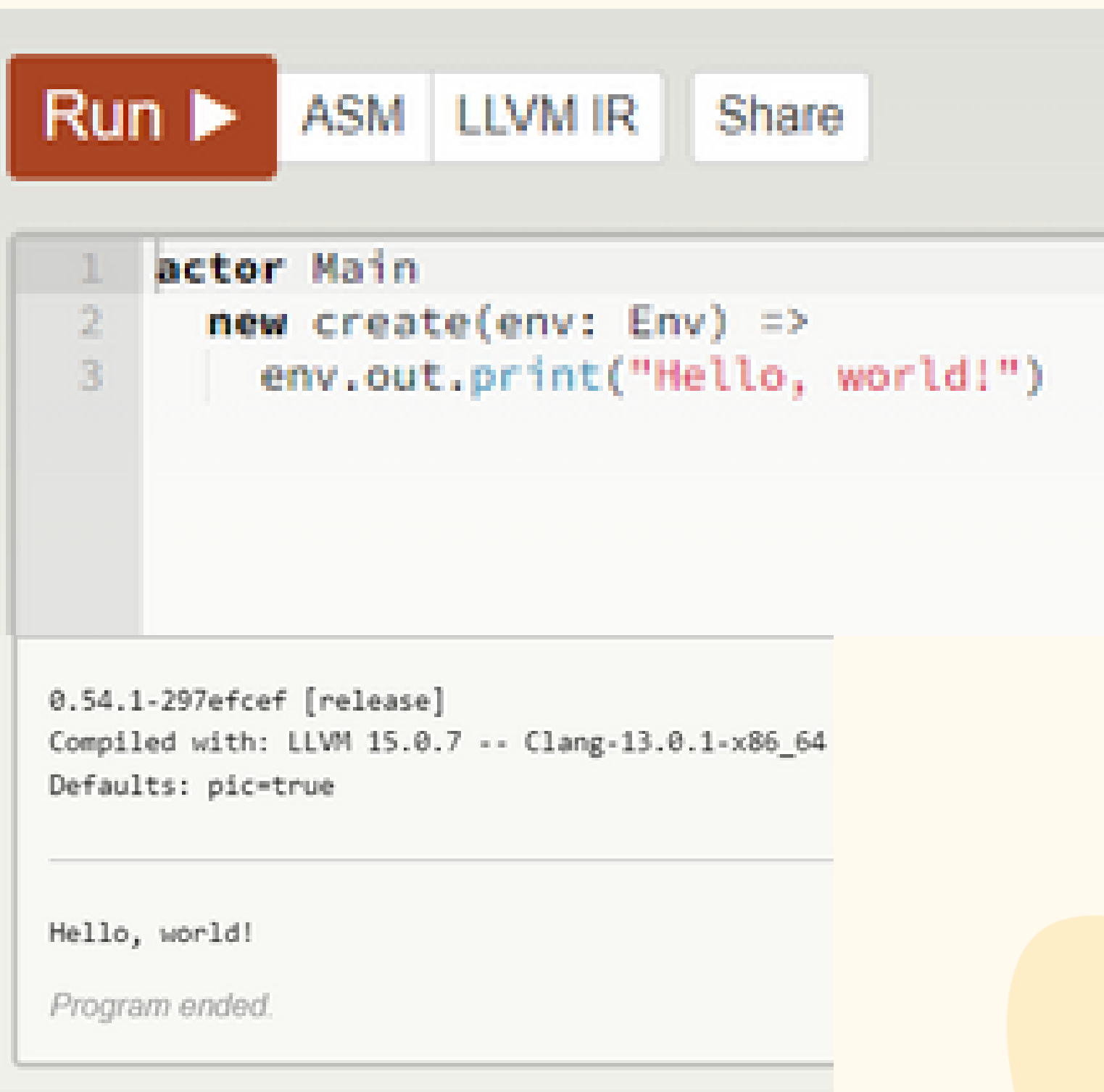
# LEARNING LANGUAGE: HELLO WORLD, VARIABLES, AND TYPE EXPRESSIONS



In this section, I wanted to do a quick summary of different aspects of the language, to help familiarize the audience with it. The main attributes I'll discuss are the "Hello World" expression, type expressions, and methods

# LEARNING LANGUAGE: HELLO WORLD

## Sample Code (GitHub)



The screenshot shows a web-based code editor with a light gray background. At the top, there are four buttons: 'Run' (orange with a play icon), 'ASM' (light blue), 'LLVM IR' (light blue), and 'Share' (light blue). Below the buttons is a code editor with three lines of code:

```
1 actor Main
2   new create(env: Env) =>
3     env.out.print("Hello, world!")
```

Below the code editor is a terminal window with a light gray background. It shows the following output:

```
0.54.1-297efcef [release]
Compiled with: LLVM 15.0.7 -- Clang-13.0.1-x86_64
Defaults: pic=true

Hello, world!
Program ended.
```

The **actor Main** in Pony is defined as a type declaration. As supported by Pony's official Github, the keyword **actor** is used to define an actor, much like how you'd describe a class in Python or in Java. In order for a Pony program to run, it needs to have a Main actor, similar to a main method in Java.

The second line of code describes a **constructor**. The keyword **new** means "it's a function that creates a new instance of the type". In this language, constructors have names, which is why this constructor is labeled **create**.

Next, the constructor has a parameter called **env** to reference the environment that the program is in and is afterward separated by a colon.

Finally, the last line prints out the string within the print statement from the environment

# LEARNING LANGUAGE: VARIABLES

## Sample Code



A lot like other languages, Pony allows you to store data in variables. Within Pony, there are different types of variables that have different lifetimes and purposes. In order to define a **local variable** the **var** keyword is used and afterwards comes the variable's name. Optionally, you can place a colon after the name.

Local variables can be declared with either **let** or **var**. **Var** can be assigned and reassigned as often as desired, while **let** "means the variable can only be assigned once".

**Fields** in Pony are variables that live within objects and have the same lifetime as the object they're in instead of being scoped. According to the Github fields are "set up by the object constructor and disposed of along with the object". Field assignments can be **var** or **let** and the rules for field assignments differ from variable assignments.

```
Run ▶ ASM LLVM IR Share
1 actor Main
2   new create(env: Env) =>
3     var y: String = "OMG this actually works, crazy"
4     env.out.print(y)
```

OMG this actually works, crazy

```
1 actor Main
2   new create(env: Env) =>
3     var x: U32 = 7
4     var y: U32 = 12
5     let z: U32 = 13
6     z = 9
7     env.out.print(z)
```

```
0.54.1-297efcef [release]
Compiled with: LLVM 15.0.7 -- Clang-13.0.1-x86_64
Defaults: pic=true
Error:
main.pony:6:5: can't reassign to a let local
    z = 9
    ^
Error:
```

# LEARNING LANGUAGE: VARIABLES CONTINUED

## Sample Code (GitHub)



No matter what type of field is used, either

1. "an initial value has to be assigned in their definition"

or

2. "an initial value has to be assigned in the constructor method"

```
class Wombat
  let name: String = "Fantastibat"
  var _hunger_level: U32 = 0
```

Alternatively, these fields could be assigned in the constructor method:

```
class Wombat
  let name: String
  var _hunger_level: U32

  new create(hunger: U32) =>
    name = "Fantastibat"
    _hunger_level = hunger
```



# LEARNING LANGUAGE: TYPE EXPRESSIONS



Pony has three types of type expressions. These are **tuples**, **unions**, and **intersections**.

## Tuples:

A tuple type is a sequence of types. For example, if you wanted to write String followed by U64 it would look like this. All type expressions are written in parentheses and the elements of the tuple are separated by commas

## Unions:

A union is written like a type but uses | (or) instead of using commas between the elements

## Intersections:

An intersection uses & between its elements, and represents the opposite of a union. It's a "single value that is all of the specified types"

## Sample Code (GitHub)

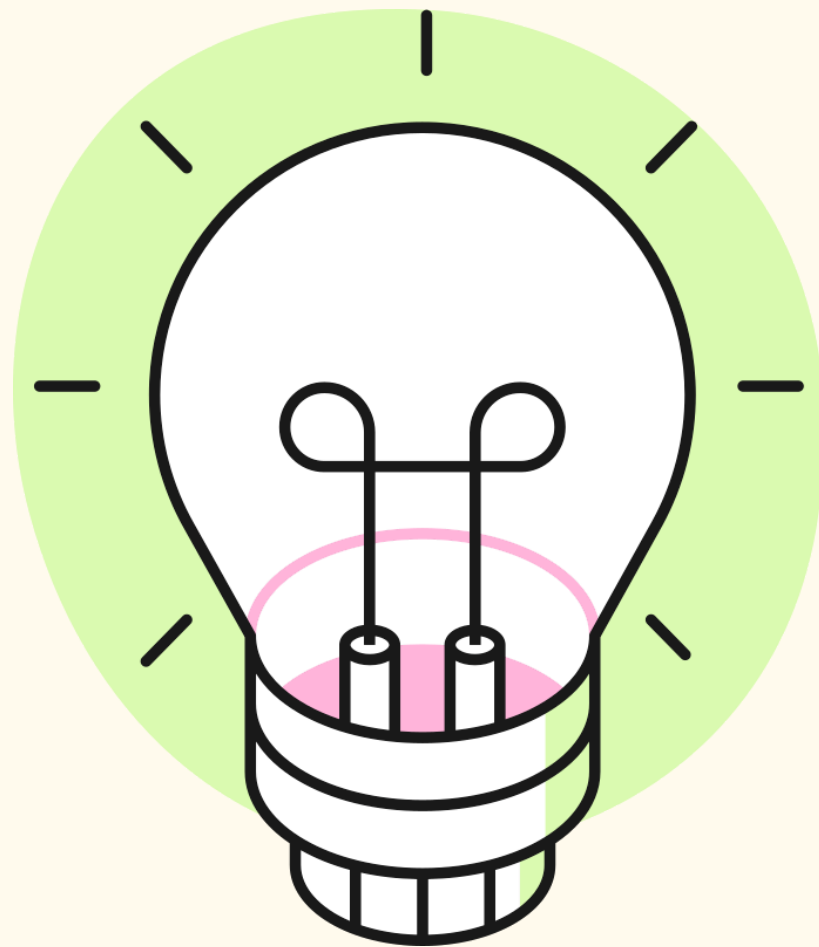
```
var x: (String, U64)
x = ("hi", 3)
x = ("bye", 7)
```

```
var x: (String | None)
```

```
type Map[K: (Hashable box & Comparable[K] box), V] is HashMap[K, V, HashEq[K]]
```

## Efficiency?

I think in terms of efficiency, Pony is pretty easy to follow. The examples provided on GitHub provide adequate instructions to follow and provide a clear way to code. The code examples are also readable



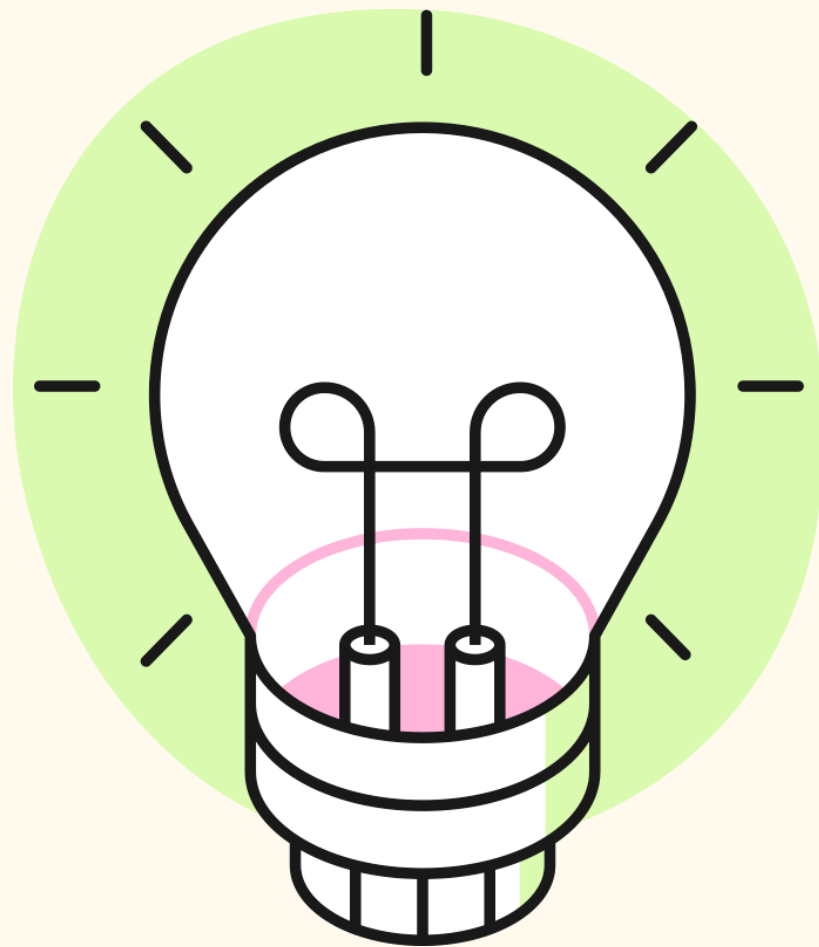
# CONCLUSION

## Pros:

- Easy to read
- Simple Grammar
- Has some similar attributes to other object-oriented languages, so there's a sense of familiarity

## Cons:

- There aren't any videos online to follow along to
- You'd have to do a thorough search through the GitHub docs to get a decent understanding
- The GitHub docs aren't very organized so it's a bit difficult to learn the language in chronological order
- Had a lot of trouble downloading the Pony repository to my computer



## CONCLUSION

### Would I Recommend it?

- I think it's easier to understand than Java at times, but like I mentioned before it would be great if they had video tutorials
- If you have the time to learn I would recommend it. If not, you could use Python or Java to save yourself the trouble of having to find multiple resources.



## WORKS CITED

**ALLEN, SEANT. "PONY-TUTORIAL/TYPE-EXPRESSIONS.MD AT MAIN · PONYLANG/PONY-TUTORIAL." GITHUB, 10 MAR. 2021, [HTTPS://GITHUB.COM/PONYLANG/PONY-TUTORIAL/BLOB/MAIN/DOCS/TYPES/TYPE-EXPRESSIONS.MD](https://github.com/PonyLang/Pony-Tutorial/blob/main/docs/types/type-expressions.md).**

**---. "PONY-TUTORIAL/VARIABLES.MD AT MAIN · PONYLANG/PONY-TUTORIAL." GITHUB, 23 MAY 2021, [GITHUB.COM/PONYLANG/PONY-TUTORIAL/BLOB/MAIN/DOCS/EXPRESSIONS/VARIABLES.MD](https://github.com/PonyLang/Pony-Tutorial/blob/main/docs/expressions/variables.md).**

**PONYLANG. "PONY-TUTORIAL/HOW-IT-WORKS.MD AT MAIN · PONYLANG/PONY-TUTORIAL." GITHUB, [GITHUB.COM/PONYLANG/PONY-TUTORIAL/BLOB/MAIN/DOCS/GETTING-STARTED/HOW-IT-WORKS.MD](https://github.com/PonyLang/Pony-Tutorial/blob/main/docs/getting-started/how-it-works.md).**

**PONY, [HTTPS://WWW.PONYLANG.IO/](https://www.ponylang.io/).**