**STATICALLY TYPED VS DYNAMICALLY TYPED:**

|  |  |
| --- | --- |
| **Statically typed** | **Dynamically typed** |
| Statically typed programming languages do type checking (i.e. the process of verifying and enforcing the constraints of types) at compile-time as opposed to run-time. | Dynamically typed programming languages do type checking at run-time as opposed to compile-time. |
| In a **statically typed language** variables' types are static, meaning once you set a variable to a type, you cannot change it. That is because typing is associated with the variable rather than the value it refers to.  For example in Java:  String str = "Hello"; **//variable str statically typed as string**  str = 5; **//would throw an error since str is supposed to be a string only** | In a **dynamically typed language** variables' types are dynamic, meaning after you set a variable to a type, you CAN change it. That is because typing is associated with the value it assumes rather than the variable itself.  For example in Python:  str = "Hello" # **variable str is linked to a string value**  str = 5 # **now it is linked to an integer value**; perfectly OK  So, it is best to think of variables in dynamically typed languages as just generic pointers to typed values. |
| (**int** a; a can take only integer type values at runtime) | (**var** a; a can take any kind of values at runtime) |
| Example:  Go does type-checking at compile time:  package main  import ("fmt"  )  func silly(a int) {  if (a > 0) {  fmt.Println("Hi")  } else {  fmt.Println("3" + 5)  }  }  func main() {  silly(2)  }  **The above will not compile, with the following error:**  **invalid operation: "3" + 5 (mismatched types string and int)** | Example:  Python does type checking at run time,  def silly(a):  if a > 0:  print 'Hi'  else:  print 5 + '3'  silly(2)  Runs perfectly fine, and produces the expected output Hi**. Error is only raised if the problematic line is hit**:  silly(-1)  Produces  **TypeError: unsupported operand type(s) for +: 'int' and 'str'**  because the relevant line was actually executed. |
| Examples of statically typed languages are :- Java, C, C++ | Examples of dynamically typed languages are :- Perl, Ruby, Python, PHP, JavaScript |
| Takeaways:  usually better performance  faster compile error feedback  better IDE support  not suited for working with undefined data formats  harder to start a development when model is not defined when  longer compilation time  in many cases requires to write more code  ======================= | Takeaways:  lower performance  faster development  some bugs might be detected only later in run-time  good for undefined data formats (meta programming) |