Comparison between Languages

Shark Deng

March 4 2019

1 Introduction

Development of compilers, such as clang and llvm, booms many computer languages. To name a few, java, swift, php, go, javascript, html, python, c, c++, c#. In this article, we will go through building blocks of a language. With these elements, we may pick up a new language very quickly or even can create one by llvm.

2 Overview

Languages can be divided into these categories:

C Famility C, C++, C#, Objective-C

Based on C Java, Javascript, Swift, Php, Python

ML XML, HTML, YAML

SQL(Structured Query Language) MySql

Shell AppleScript

Other Latex, Markdown

2.1 Oriented

Object Oriented Programming(OOP)[1], composed of attributes and behaviors, includes four aspects: encapsulation, inheritance, and polymorphism.

Encapsulation	Hiding the private details of a class from other objects.
Inheritance	A process of using details from a new class without modifying existing class.
Polymorphism	A concept of using common operation in different ways for different data input.

Table 1: Source: https://www.programiz.com/python-programming/object-oriented-programming

Object-oriented C++, C#, Objective-C, Java, Javascript, Python,Php

Procedure-oriented C

Protocol-oriented Swift

Graphic-oriented Swift

[1] https://www.programiz.com/python-programming/object-oriented-programming

2.2 Strong vs Weak Type

Strong type means all data must be typed while weak not. **Swift** is strong type because it's designed to run fast. So most of type check works is on programmers and IDE. **Php**, however, use Zend engine especially universal variable *zval* to parse variables, so its speed will be mush slower. **C** combines strong and weak types. In normal programming, it needs designate type but in micro programming, type is missed. Other languages, such as **Java** is strong type.

Strong Type Swift, Java, C

Weak Type Php, Python, C(micro)

2.3 Static vs Dynamic

It depends on whether the type check is conducted at compile-time or run-time.

Static

Dynamic

Static & Dynamic Java

2.4 Generics

Support Java, Swift

Not support

2.5 Compile

2.5.1 Java

Code on terminal None
Compile file on terminal

```
1     $ javac HelloWorld.java
2     $ java HelloWorld
```

Code on IDE IntelliJ

2.5.2 Swift

Code on terminal

```
1     $ swift
2 Welcome to Apple Swift version 4.2.1. Type :help for assistance.
3     1> var a = 88
4 a: Int = 88
5     2> let b = 66
6 b: Int = 66
7     3> let c = "Hello!"
8 c: String = "Hello!"
9     4> let d = c + String(b)
```

```
d: String = "Hello!66"
    5> import Foundation
```

Control + d to exit.
Compile file on terminal
Code on IDE Xcode

2.5.3 Python

Code on terminal

```
$ python
>>> print("Hello World")
>>> a = 10
>>> import tensorflow
>>> quit()
```

Compile file on terminal

```
$ touch a.py
$ echo "print('Hello World')" >  a.py
$ cat a.py
$ python a.py
$ rm a.py
```

Code on IDE Pycharm

2.5.4 Objective-C

Compile file on terminal

```
$ touch source.m
$ gcc -framework Foundation source.m -o source
$ ./source
```

Code on IDE Xcode

2.5.5 Php

Code on IDE PhpStorm

2.6 Scalability

2.6.1 Java

Some languages have rich libraries.

```
```

Dependency Management
Maven.

2.6.2 Swift

```
1  import Foundation
```

Dependency Management
Codpods

2.6.3 Python

Dependency Management
.yml

2.7 Compile System

2.8 namespace

3 Variable

4 Primitive Data Type

4.1 Java

No.	Type	Description	Range	Default
1	byte	1-byte (8-bit) / signed / 2's complement integer	-128 - 127	0
2	short	2-byte (16-bit) / signed / 2's complement	-32768 - 32767	0
3	int	4-byte (32-bit) / signed / 2's complement	$-2^{31} - 2^{31} - 1$	0
4	long	8-byte (64-bit) / signed 2's complement	$-2^{63} - 2^{63} - 1$	0
5	float	(32-bit) single precision floating number		0.0f
6	double	(64-bit) double precision floating number		0.0d
7	boolean	logically just a single bit	true, false	false
8	char	2-byte (16-bit) Unicode character	0 - 65535	0

4.2 Python3

No.	Type	Description	Range	Default
1	Number	int float bool complex		
2	String	Non-changeable		
3	Tuple	Non-changeable		
4	List	Changeable		
5	Set	Changeable		
6	Dictionary	Changeable		

Use the code to check type

```
type(10) # <class 'int'>
type(5.5) # <class 'float'>
type(True) # <class 'bool'>
type(4+3i) # <class 'complex'>
isinstance(10, int) # True
```

4.3 Swift

4.4 Objective-C

[2]

4.5 C

5 Data Structure

5.1 Common

Tuple, array, dict, set, map.

In next section, we will discuss specific data structures in these language.

5.2 Java

5.3 Array

```
public class ArrayTest {
    public static void main(String[] args) {
        // declare: don't allocate memory
        double[] a;

        // initialize 1: allocate memory
        a = new double[10];

        // give value
        a[0] = 0.0;
        a[1] = 0.1;
        a[2] = 0.2;
        a[3] = 0.3;
        a[4] = 0.4;

        // initialize 2
        double[] b = {0.1, 0.4, 0.6, 0.3};


        // display 1
```

[2] $https://en.wikipedia.org/wiki/Cdata_types$

```
21          for (int i=0; i<a.length; i++){
22              System.out.println(a[i]);
23          }
24
25          // display 2
26          for (double ele: b){
27              System.out.println(ele);
28          }
29
30
31          double[] c = a; // same memory address, two references.
32          c = new double[7];
33      }
34  }
```

5.4 List

5.4.1 ArrayList

ArrayList is dynamic array, which means its length can dynamically increase.But it is not thread safe.

5.4.2 HashMap

5.4.3 LindedHashMap

5.4.4 TreeMap

stMap.java

6 Control Flow - 5

6.1 Selection - 3

6.1.1 If then else

6.1.2 Switch

6.1.3 Try ... except ...

6.2 Iteration - 2

6.2.1 For

6.2.2 While

7 Function

7.1 Define

7.1.1 Python

```
def main():
    print('hello')

main()
```

7.2 Lambda

Also called callback.

8 Object & Class

8.1 Access Level

8.1.1 Swift - 5

name	specifier	access	example
open		outside module (read and modify)	
public		outside module (read)	
internal		inside module	
fileprivate		inside file	
private		inside class	

8.1.2 Java - 4

name	specifier	access	example
public		-	
protected		inside this class and its children	
private		inside this class	
default(package private)			

8.1.3 Python - 3

name	specifier	access	example
public	-	-	self.public = 10
protected	single underline	-	self._protected = 10
private	double underlines	inside this class	self.__private = 10

8.2 Define

A class has **constructor**, **destructor**. These will be detailed in following table of responding languages:

8.2.1 Python

Example code for Magic Methods is available here Test5.py

Magic Methods	Meaning
__new__ __init__ __del__	create a new instance constructor(initialize a new instance) desctructor
__str__ __repr__	print(obj) obj(on terminal)
__getitem__ __setitem__	
__cmp__ __eq__ __ne__ __lt__ __gt__ __le__ __ge__	 = != < > <= >=
__add__ __sub__ __floordiv__ __truediv__ __mod__ __pow__ __lshift__ __rshift__ __and__ __xor__ __or__	+ - // / % ** « » &

Table 2: Magic methods

Python use **decorator** to realize static class and so on. Pre-made decorators are listed in the following table 3 and relevant example is Test1.py. Custom decorator is exampled here Test6.py.

Method Decorator	Meaning	Example
@staticmethod		
@classmethod		
@property	getter and setter	Test3.py

Table 3: Pre-made decorators

8.3 Inheritance

8.3.1 java

8.4 Interface

8.4.1 Java

has

8.4.2 C

Because C is procedure-oriented. It doesn't have interface.

8.4.3 C++

8.4.4 C#

8.4.5 Objective-C

8.4.6 Swift

8.4.7 Php

8.5 Annotation

9 Characteristics

9.1 Objective-C

9.1.1 Category

9.1.2 Extension

9.1.3 Protocol

10 Enum

11 Singleton

12 JavaFXApplication

13 Code Standards

Listings