

28/10/22 ⑥ Class Identifiers, Reserved Keyword, Data types in JAVA :-

Today's topic of discussion

① oops (basic introduction)

② identifiers / Variables

a. rules to write an identifier.

③ Reserved words.

④ Data types and its chart.

① Object orientation Principles :- (oops)

→ it stands for object orientation principles.
object → real time instance

→ Every object in real time will have 2 parts.

① what it has?

② what it does.

Eg Car:-

① what it has?

brand name

no. of wheels

model

speed

② what it does?

move

accelerate

brake.

Java Code → object representation

Class Car

{

// This part of an object is represented as a "Variable".

String brandName

int noOfWheels

String model

Speed

// Does part of an object is represented through "methods".

}

→ terminate statements with semi colon;

Java Code

object representation to Express object we use Class.

Class Car

{

// Has part of an object is represented as Variable.

String brandName;

int noOfWheels;

// Does part of an object is represented through Methods.

Public Void move()

{

// logic of Moving a Vehicle

}

Public Void accelerate()

{

// logic of accelerating a Vehicle

}

}

① Object - what we see physically.

Eg - ① Student → object

Class Student

{

// Has part (variables)

String name;

int id;

float height;

- Has Part

// Does part (methods)

Public Void study();

// logic of studying.

}

Public Void playing();

// logic of playing.

}

}

→ does part.

② object - object is a member (also called as instance) of a Java class. Each object has an identity, a behavior and a state.
→ the state of a object is stored in fields (variable)
→ while (methods (functions)) is play is the objects behavior.

③ Identifier :-

29/10/22

→ identifier is a name in Java program

→ it can be a class name, method name, variable name and label name.

Eg ① Class Main test

{

Public Static Void main (String [] args)

{

int x = 10;

}

}

→ String is Class in Java.

→ totally 5 identifiers.

Eg ② :- Class Test {

Public Static Void main (String [] args) {

System.out.println ("Sachin");

class

method

Variable

(instance of System class)

→ totally 7 identifiers.

Eq 3

Class Demo {

public static void main (String [args]) {

String name = 'sachin';

String result = name.toUpperCase();

System.out.println(result);

}

}

2.1 Rules for writing an identifier
↳ w.r.t to (JVM + compiler).

(i) Rule 1: The only allowed characters in java identifiers are a to z, A to Z, 0 to 9, _ (underscore), \$.

Rule 2: if we use any other character, it would result in an invalid or in complete time error.

int ^{*} = 10; (invalid) X

int total = 10; (Valid) ✓

int total# = 10; (invalid) X

int telusko1 = 100; (Valid) ✓

~~int telusko~~

Rule 3: identifiers are not allowed to start with digit.

Eq 1 int telusko1 = 100; (Valid) ✓
int 1telusko = 100; (Invalid) X

Rule 4: Java identifiers are Case Sensitive.

Eq Class Demo {

int number = 10;

int Number = 20;

int number = 30;

int NUMBER = 40;

} all these variables are different.

Rule 5: There is no length limit on java identifiers but still it is a good practice to keep the length of the identifier no more than 15 character.

Eq 0 int Priority of Thread with Min Value = 1;

→ identifiers can also start with special symbol '\$' and '-'.

Eq 0 int \$a = 10;

② int -a = 10;

→ if, while, for, else, string, try catch throw asserts.

2.2 Reserved words

- there are identifiers which have special meaning associated with. Compiler and JVM. (as it is a built in words/keywords which has already a predefined meaning to it.)
- they are Reserved for some meaning.

Eg if, while, for, else, string, try, catch, throw, throws, assert, true, false, true, false.

- they are Predefined for Compiler and JVM.

Rule (6) - We Can't use reserved words as identifiers

Eg `int if = 10;` ^{true}
↳ Compile time Error

Rule (7) - Predefined class names can be used as identifiers.

- ① String → class name ^{inbuilt}
- ② Runnable → interface (inbuilt)
- ③ student → user defined class name.

Eg ① `int string = 10;`
`System.out.println(10);`

Eg ① -

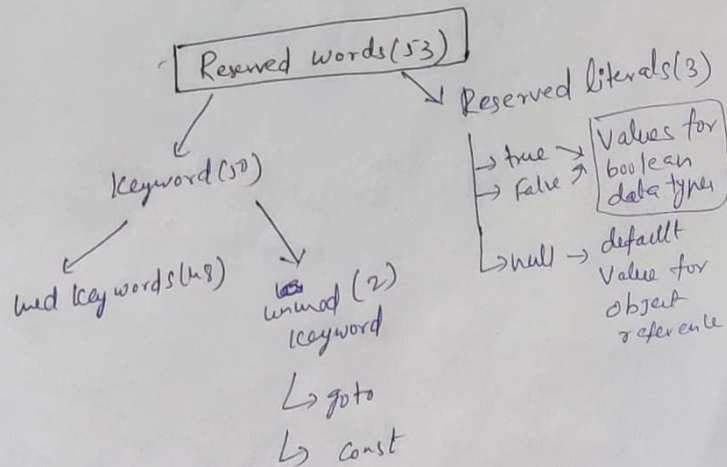
`String Runnable = "sachin";`
`System.out.println(Runnable);`

Note - Even though predefined class names can be used as identifiers, it is not a good practice to keep.

- if and I F different.

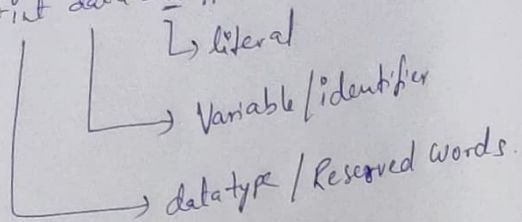
↳ Reserved word.

→ `int iint t = 10;` X
`System.out.println(iint);`
↳ Reserved word.
↳ Compile time Error.



Literal :-
→ Any constant value which can be assigned to variable is called literal.

Eg ① int data = 10 //



→ Any constant value which can be assigned to variable is called literal.

Reserved Literals (3)

- ① true
 - ② false
 - ③ null
- value of boolean data types
→ default value for object reference

Eg ① boolean result = true;
→ then literal is associated with boolean variable

② boolean flag = false;

- ③ boolean ExamResult = True;
 - ④ boolean ExamResult = true;
 - ⑤ boolean flag = 0;
- these are different for JVM
→ boolean literals

→ In C++ 0 → false, 1 → true

→ In Java 0 → means zero
1 → means one

Note - for boolean data types the only values allowed for a variable is "true" & "false", other than this if we try to keep any values it would result in "Compile Time Error".