Enum: If we want to represent a group of named constants, Then We should go for enum

enum Month {

JAN, FEB, MAR, DEC;

}

The main objective of enum is to define our own data types (enumerated Data Types)

Enum concept introduced in 1.5 version, when compared with old languages enum java enum is more powerful.

Internal implementation of enum:

Every enum is internally implemented by using class concept.

Every enum constant is always public, static, and final

Every enum constant represents an object type of the enum.

enum Beer {

KF,RC

}

Class Beer {

Public static final Beer KF = new Beer ();

Public static final Beer RC = new Beer ();

}

Enum Declaration and usage:

Enum Beer {

KF, KO,RC,RO;

}

Every enum constant is always by default are public static final and hence we can access the enum constant by using enum name.

enum Beer {

KF, RC

}

Class Test {

p.s.v.m () {

Beer b = Beer.RC;

Sysout (b) ;-----> RC

}

}

Note: Inside enum, to string method internally implemented to return name of the constant.

We can declare enum either with in the class or outside the class but not inside a method.

If we are trying to declare an enum inside method then we will get the compile time error.

Example: enum types must not be local.

Scenario valid/Invalid

enum x { valid

}

class y {

}

Class x { valid

enum y {

}

}

Class x { CE: enum types must not be local

Public void m1 () {

Enum y {

}

}

}

If we declare enum outside of the class applicable modifiers are public, default, strictfp

If we declare an enum inside a class, the applicable modifiers are public, default, strictfp, and private, protected, static.

Enum vs switch:

Until 1.4 version, the allowed argument types for the switch statement are byte, short char, and int

However, from 1.5 version onwards, corresponding wrapper classes and enum types are allowed.

From 1.7 version onwards, String data type also allowed.

Switch(x) {

}

1.4 version 1.5 version 1.7 version

byte Byte String

short Short

char Character

Int Integer

enum

Hence, from 1.5 version onwards we can pass enum type as argument to switch.

enum Beer {

KF,KO,KC,RC

}

Class Test {

Public static void main(String[] args) {

Beer b = new Beer.RC;

switch(b) {

case KF:

sysout(“it is children’s brand”);

break;

case KO:

sysout(“it is KO”);

break;

case RC:

sysout(“it is RC”);

break;

case FO:

sysout(“FO);

break;

}

}

}

Op: It is RC.

If we pass enum type as arg to switch statement then every case label should be valid enum constant.

Otherwise, we will get compile time error.

Switch (b) {

case KF:

case ko:

case RC

case FO:

case kalyani:

ce:unqualified .enumeration constant name required.

}