



- 가
- 8가 ( ) , , , ,



- 가 4가
- 가

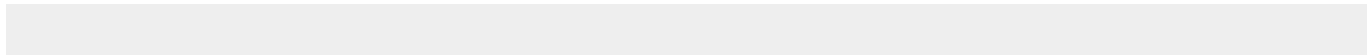


- boolean true false 1
  - : false
- char (2 byte ) 2byte
  - : \u0000
- byte 가 1byte byte.
  - : 0
- int(4 byte) (2 byte) (8 byte)
  - : 0
- float (floating-point) float
  - : 0.0F
- double float (8byte) double
  - : 0.0



- (precision)가 , 가 가
- float 7 10 7 double

- |   |   |                  |
|---|---|------------------|
|   | (Primitive Type)  | (Reference Type) |
| • | (Primitive Type) : (boolean), (char), (byte, short, int, long), (float, double) |                  |
| • | (reference type) :  | Java.lang.Object |



- 
- 2020



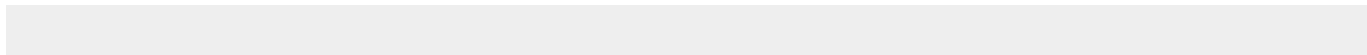
- , 2020, 123, 3.14, "ABC"



- (Immutable) (immutable class)
- 
- Ex: Java.lang.String java.awt.Color



- : (type)
- : 가



(initialization)



- year 가 = 2020 year int
- ,
- ,

1. **(explicit initialization)**

: 가 ,

## 2. (initialization block)

:

```

class ExplicitInitialization {
    static {
        /*          */
    }
    {
        /*          */
    }
}

```

- : 가
- : 가

## 3. (constructor)

:

가

```

class A {
    int instanceValue; //
    static int classValue; // (static, )
    void method(){
        int localValue = 0; //
    }
}

```

static 가 static 가 localValue ,

1.



1) **(instance variable)**

- 
- 가
- Ex: 1000 가  
가

2) **(class variable)**

- static 가 가
- 가
- (global variable) 가

```
class LottoTicket {
    public static final LOTTO_PRICE = 1000;
    ...
}
public static void main(String[] args) {
    //LottoPrice: 1000
    System.out.println("LottoPrice: "+ LottoTicket.LOTTO_PRICE);
}
```

3) **(local variable)**

- 가
- for while

```
public static void main(String[] args) {
    for (int i = 0; i < 10; i++) {
        System.out.println("i = " + i);
    }
    System.out.println("i = " + i);//Checked Exception
}
```

( )

1.



( 가 .)

2.



3.

```

class InitTest {
    static int classValue = 1;
    int instanceValue = 1;
    static {
        classValue = 2;
    }
    InitTest() {
        instanceValue = 3;
    }
}

```



- (1~3): 가 .
- (4~7): .
- \_\_\_\_\_ .

,

가 .

□ (type)operand

- .

```
double value = 123.456;
int score = (int)value;
System.out.println(value == 123.456); //true
```

- (primitive type) boolean 가 .
- , 가 ( )가 (loss of data) .

- 가 . 가 .
- , 가 가 가 .

```
byte b = 10000; // . byte -128~127 .
byte c = (byte)10000; // 가 .
```

- 가 가



1 2

1.



- ([[]]) .

2.

```
[ ] = new [ ];
```

- new
- 

## 1 & 2

- 1
- 2 가 ([ ]) 가 1 1 , 2

### 1.1

```
int[] oneDimensionalArray = new int[5]{1, 2, 3, 4, 5};
```



1

### 2.2

```
int[][] twoDimensionalArray = new int[2][2]{{1, 2}, {3, 4}};
```



2

- 2 / twoDimensionalArray 2 2 2 new int[ ][ ]

## , var

- Type Inference ( )
- (inference algorithm) 가 가
- ( )

```
static <T> T pick(T a1, T a2) { return a2; }
public static void main(String[] args) {
    Serializable d = pick("d", new ArrayList<String>());
}
```

- 가 a1, a2 T , pick
- pick T String ArrayList Serializable String ArrayList

Serializable

### Generic Methods

- generic

```
public class BoxDemo {
    public static <U> void addBox(U u, java.util.List<Box<U>> boxes) {
        Box<U> box = new Box<>();
        box.set(u);
        boxes.add(box);
    }
}

public static <U> void outputBoxes(java.util.List<Box<U>> boxes) {
    int counter = 0;
    for (Box<U> box: boxes) {
        U boxContents = box.get();
        System.out.println("Box #" + counter + " contains [" +
            boxContents.toString() + "]");
        counter++;
    }
}

public static void main(String[] args) {
    java.util.ArrayList<Box<Integer>> listOfIntegerBoxes =
        new java.util.ArrayList<>();
    BoxDemo.<Integer>addBox(Integer.valueOf(10), listOfIntegerBoxes); //---
(1)
    BoxDemo.addBox(Integer.valueOf(20), listOfIntegerBoxes); //--- (2)
    BoxDemo.addBox(Integer.valueOf(30), listOfIntegerBoxes);
    BoxDemo.outputBoxes(listOfIntegerBoxes);
}
}
```

(1) : addBox      generic                                      <Integer> type witness                      type parameter  
 (2) : Java              가                                      Integer type                                      type witness

### Generic

- Java              가                                      가                      Generic  
                                  type arguments                      type witness(<>, the diamond)

```
List<String> myList1 = new ArrayList<String>();
List<String> myList2 = new ArrayList<>();
```

## Generic

- 가 Generic/non-generic

generic

```
class MyClass<X> {
  <T> MyClass(T t) {
    // ...
  }
}

public static void main(String[] args) {
  MyClass<Integer> myInstance = new MyClass<Integer>("");
}
```

- MyClass type X Integer가 가 type T String

Java SE 7 type argument  
 Java SE 7 the diamond(<>)  
 type argument 가 generic

```
MyClass<Integer> myInstance = new MyClass<>("");
```

## Target Types

- Java generic method invocation type argument target typing
- target type ( ) Java 가

```
static <T> List<T> emptyList() { return new ArrayList<>(); }
List<String> listOne = Collections.emptyList();
```

- Collection API emptyList List<String>
- Target Type , emptyList 가 List<T>
- type argument T가 String
- , type witness

```
List<String> listOne = Collections.<String>emptyList(); // witness
```

- , Type Witness가

```
void processStringList(List<String> stringList) {
```

```

    //process stringList
}
public static void main(String[] args) {
    processStringList(Collections.emptyList());
}

```

- Java SE 7
- ? 가

List<Object> cannot be converted to List<String>

- Object value type argument T value
- Collections.emptyList() List<Object> processStringList
- Java SE 7 type witness

processStringList(Collections.<String>emptyList());

- Java SE 8 type witness Target type
- argument 가
- Java SE 8 type witness가

**var**

- Java 10 가 Local Variable Type Inference var

```

var list = new ArrayList<String>(); //infers ArrayList<String>
var stream = list.stream(); //infers Stream<String>

```

**Local Variable Type Inference**

- (enhanced for loop )

**var**

1.

```

var numbers = Arrays.asList(1, 2, 3, 4, 5);

for (var i = 0; i < numbers.size(); i++) {
    System.out.println("numbers = " + numbers.get(i));
}

```

```
}
```

## 2. forEach

```
var numbers = Arrays.asList(1, 2, 3, 4, 5);

for (var number : numbers) {
    System.out.println(number);
}
```

- Object IDE가 가 .  
var 가 .

## 3. Lambda (Java 11)

```
IntBinaryOperator plus10 = (@NonNull var one, @NonNull var two) -> one + two + 10;
```

- Java 11 var 가 ,  
가 .
- : type witness Object .

## Ref

<https://github.com/whiteship/live-study/issues/2>

From: <http://rwiki.repia.com/> - . - 2023.12

Permanent link: <http://rwiki.repia.com/doku.php?id=wiki:java:java-lecture:2week&rev=1610929470> 

Last update: **2022/03/10 19:52**