

# Chương 1: GIỚI THIỆU NNLT JAVA

Khoa CNTT

ĐH GTVT TP.HCM

## TUYÊN BỐ SỨ MỆNH

Trường Đại học Giao thông vận tải thành phố Hồ Chí Minh đào tạo đội ngũ cán bộ khoa học kỹ thuật, cán bộ quản lý trình độ đại học và sau đại học; Tổ chức nghiên cứu khoa học, chuyển giao công nghệ về giao thông vận tải và các lĩnh vực liên quan phục vụ cho sự nghiệp Công nghiệp hóa – Hiện đại hóa đất nước và hội nhập quốc tế.

## TẦM NHÌN

Đến năm 2030, Trường phấn đấu trở thành trường đại học lớn, đa ngành của Việt Nam, là trung tâm đào tạo, bồi dưỡng, nghiên cứu khoa học hàng đầu về giao thông vận tải và các lĩnh vực liên quan, có uy tín, hoà nhập với các trường đại học trong khu vực và vững vàng tiếp cận trình độ các trường đại học tiên tiến trên thế giới.

## MỤC TIÊU CHẤT LƯỢNG

Trường phấn đấu trở thành trường đại học đào tạo nhân lực trình độ cao theo hướng ứng dụng trong lĩnh vực giao thông vận tải của khu vực phía Nam và cả nước. Hoàn thiện cơ cấu ngành nghề và trình độ đào tạo phù hợp với quy hoạch phát triển nhân lực của Bộ Giao thông vận tải và quốc gia, trong đó, có một số ngành ngang tầm khu vực và quốc tế. Tăng số lượng và chất lượng các đề tài nghiên cứu, chuyển giao công nghệ có hàm lượng khoa học cao, đảm bảo tính ứng dụng và triển khai thực tế.

(Theo Nghị quyết số 14/NQ-HĐTĐHGTVT ngày 11 tháng 01 năm 2021 của Hội đồng trường Trường Đại học Giao thông vận tải TP. Hồ Chí Minh)

# Mục tiêu học phần

<b>Mục tiêu [1]</b>	<b>Mô tả [2] Học phần này trang bị cho sinh viên:</b>	<b>Chuẩn đầu ra CTĐT [3]</b>
CO1	Áp dụng các khái niệm, phương pháp lập trình hướng đối tượng, xử lý sự kiện, đa luồng và tương tác cơ sở dữ liệu giải quyết các bài toán lập trình phần mềm	PLO2
CO2	Kết hợp các kỹ năng, phương pháp, công cụ lập trình trên Java để lập trình phần mềm	PLO2
CO3	Tích cực tự học theo xu thế phát triển của lĩnh vực	PLO7

# Mục tiêu học phần

Mục tiêu HP [1]	CDR HP [2]	Mô tả CDR [3]	Chuẩn đầu ra CTĐT [4]
CO1	CLO1.1	Áp dụng thực tế các khái niệm, phương pháp lập trình hướng đối tượng giải quyết các bài toán lập trình phần mềm	PI2.1 PI2.2 PI2.4
	CLO1.2	Phân tích phương pháp xử lý sự kiện, lập trình đa luồng và lập trình tương tác cơ sở dữ liệu trong giải quyết các bài toán lập trình phần mềm	PI2.2 PI2.4
CO2	CLO2.1	Áp dụng công cụ, môi trường lập trình Java một cách phù hợp	PI2.1 PI2.4
	CLO2.2	Thực hiện chính xác các kỹ năng, phương pháp lập trình trên Java để xây dựng phần mềm	PI2.2 PI2.4
CO3	CLO3.1	Tích cực tham gia vào quá trình học	PI7.1
	CLO3.2	Hình thành thói quen làm việc độc lập hoặc theo nhóm với các hoạt động và hình thức được qui định	PI7.2

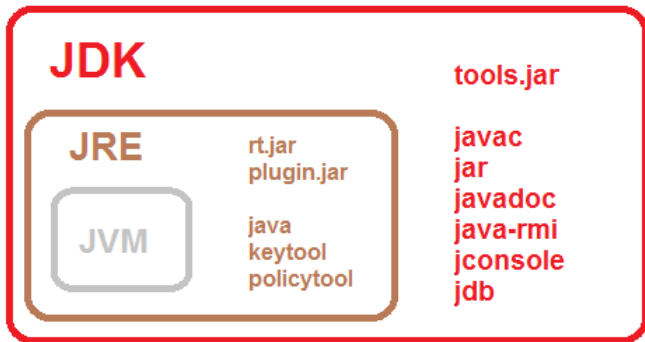
- ❶ Đặc điểm của Java
- ❷ Java platform
- ❸ Kiến trúc Java SE
- ❹ Các kiểu dữ liệu cơ sở
- ❺ Các toán tử
- ❻ Các cấu trúc điều khiển
- ❼ Mảng, chuỗi và lớp bao

# Đặc điểm của NNLT Java

- Simple
- Familiar
- Object-Oriented
- Robust
- Secure
- High performance
- Multithreaded
- Platform Independence

- J2SE (Java 2 Standard Edition): platform tối giản, đủ dùng cho các ứng dụng nhỏ và trung bình.
- J2EE (Java 2 Enterprise Edition): platform mạnh nhất, nó cho phép xây dựng bất kỳ ứng dụng nào, nhất là các ứng dụng lớn.
- J2ME (Java 2 Mobile Edition): platform phục vụ viết các ứng dụng chạy trên các thiết bị di động.



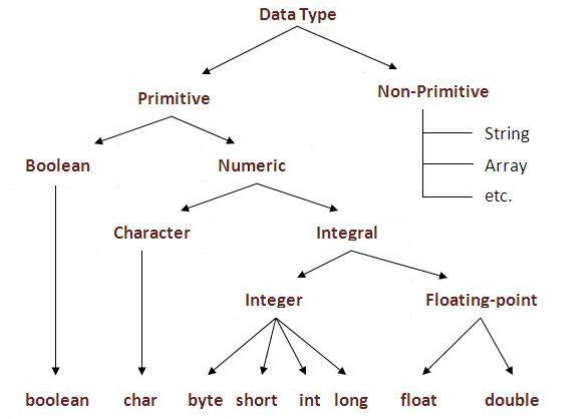


[www.codejava.net](http://www.codejava.net)

Hình 1: Kiến trúc Java SE

- \* JVM (Java Virtual Machine): is the virtual machine that runs Java applications. The JVM makes Java platform-independence.
- \* JRE (Java Runtime Environment) = JVM + standard libraries: provides environment for executing Java applications.
- \* JDK (Java Development Kit) = JRE + development tools for compiling and debugging Java applications.

# Các kiểu dữ liệu cơ sở



Hình 2: Phân cấp kiểu dữ liệu cơ sở

Precedence	Operator	Operand type	Description
1	++, --	Arithmetic	Increment and decrement
1	+, -	Arithmetic	Unary plus and minus
1	~	Integral	Bitwise complement
1	!	Boolean	Logical complement
1	( type )	Any	Cast
2	*, /, %	Arithmetic	Multiplication, division, remainder
3	+, -	Arithmetic	Addition and subtraction
3	+	String	String concatenation
4	<<	Integral	Left shift
4	>>	Integral	Right shift with sign extension
4	>>>	Integral	Right shift with no extension
5	<, <=, >, >=	Arithmetic	Numeric comparison
5	instanceof	Object	Type comparison
6	==, !=	Primitive	Equality and inequality of value
6	==, !=	Object	Equality and inequality of reference
7	&	Integral	Bitwise AND
7	&	Boolean	Boolean AND
8	^	Integral	Bitwise XOR
8	^	Boolean	Boolean XOR
9		Integral	Bitwise OR
9		Boolean	Boolean OR
10	&&	Boolean	Conditional AND
11		Boolean	Conditional OR
12	?:	N/A	Conditional ternary operator
13	=	Any	Assignment

Hình 3: Các toán tử

# Các cấu trúc điều khiển

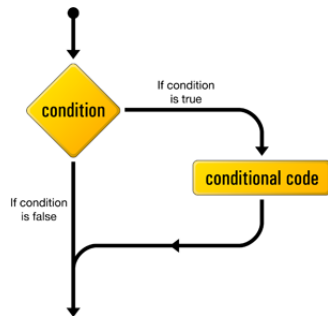
## Cấu trúc tuần tự

```
{  
    statement 1;  
    statement 2;  
    ...  
    statement k;  
}
```

# Các cấu trúc điều khiển

## Syntax

```
if (booleanExpression) {  
    statement(s)  
}  
else {  
    statement(s)  
}
```



Hình 4: Cấu trúc if

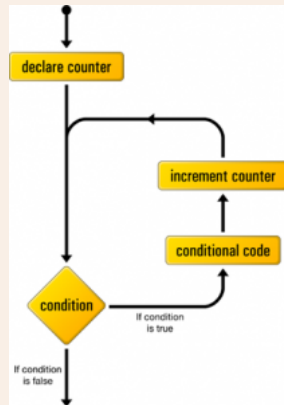
## Cấu trúc rẽ nhánh - Lệnh SWITCH

```
switch (expression) {  
    case value_1 :  
        statement(s);  
        break;  
    case value_2 :  
        statement(s);  
        break;  
    ...  
    ...  
    case value_n :  
        statement(s);  
        break;  
    default:  
        statement(s);  
}
```

# Các cấu trúc điều khiển

## Cấu trúc lặp - Lệnh FOR

```
for (initialization; condition ; increment) {  
    // body of loop  
}
```



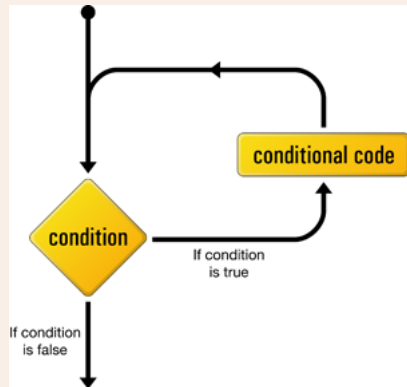
Hình 5: Cấu trúc for



# Các cấu trúc điều khiển

## Cấu trúc lặp - Lệnh WHILE

```
while (condition){  
    //body of loop  
}
```

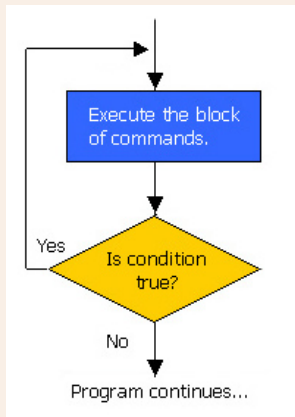


Hình 6: Cấu trúc while

# Các cấu trúc điều khiển

## Cấu trúc lặp - Lệnh DO

```
do{  
  //body of loop  
}while(condition);
```

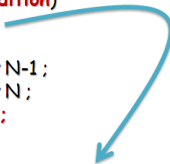


Hình 7: Cấu trúc do

# Các cấu trúc điều khiển

## Lệnh BREAK

```
Initialization;  
while (condition)  
{  
    Statement 1 ;  
    Statement 2 ;  
    Statement 3 ;  
    .....  
    .....  
    if ( If Condition)  
        break;  
    Statement N-1 ;  
    Statement N ;  
    Increment;  
}  
OutsideStatement 1;
```



Hình 8: break

## Lệnh CONTINUE

```
while (var < 10)
```

```
{
```

```
    www.c4learn.com
```

```
    continue;
```

```
    _____  
    _____  
    _____
```

```
    www.c4learn.com
```

```
}
```



These Statements  
are  
Skipped

## One Dimensional array

Initialization `int a[] = new int [12];`

Value

1	2	3	4	5	6	7	8	9	10	11	12
---	---	---	---	---	---	---	---	---	----	----	----

Index

↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑
a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]	a[10]	a[11]

`System.out.print(a[5]);`

Output: 6

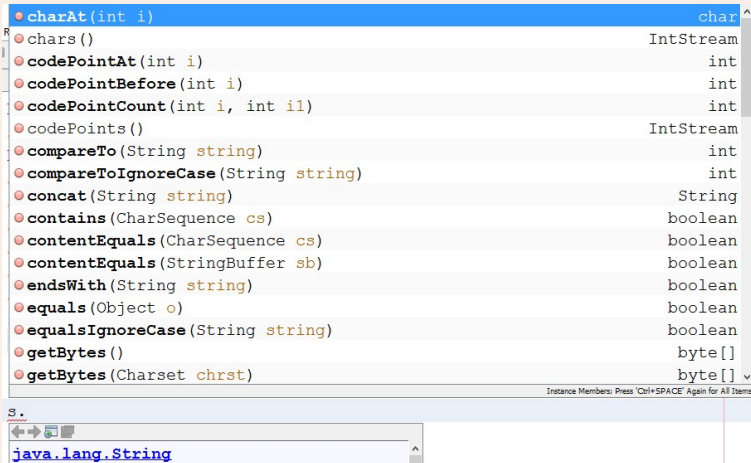
# Chuỗi - String

## Các cách khởi tạo chuỗi trong Java

```
String s = new String();  
String s1 = "Xin chào";  
char chars[] = {'i', 'B', 'y', 't', 'e', 'C', 'o', 'd', 'e'};  
String s2 = new String(chars);  
byte byteAscii[] = {65, 66, 67, 68, 69, 70};  
String strAscii = new String(byteAscii);  
byte bytes[] = {'w', 'o', 'r', 'l', 'd'};  
String s3 = new String(bytes);  
String s4 = new String(bytes, 1, 3);  
StringBuffer sb = new StringBuffer("String in java");  
String s5 = new String(sb);
```

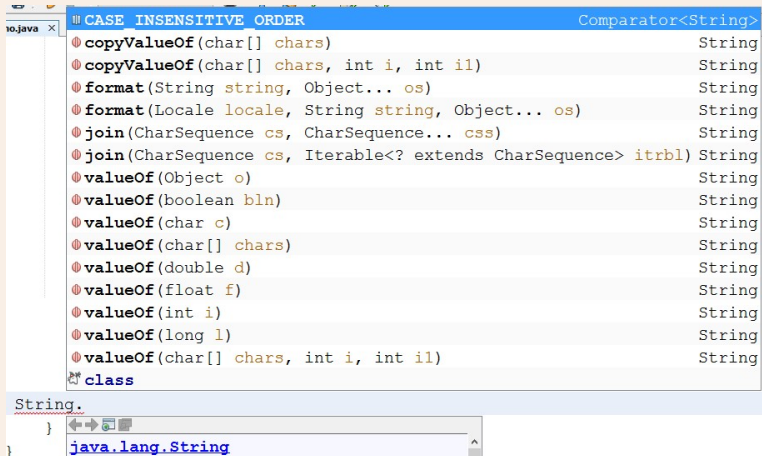
# Chuỗi - String

## Các thành phần non-static



# Chuỗi - String

## Các thành phần static



The screenshot shows an IDE window with the title "io.java". The main editor displays the static methods of the `String` class, organized under the heading "CASE INSENSITIVE ORDER" and "Comparator<String>". The methods listed are:

- `copyValueOf(char[] chars)` returns `String`
- `copyValueOf(char[] chars, int i, int il)` returns `String`
- `format(String string, Object... os)` returns `String`
- `format(Locale locale, String string, Object... os)` returns `String`
- `join(CharSequence cs, CharSequence... css)` returns `String`
- `join(CharSequence cs, Iterable<? extends CharSequence> itrbl)` returns `String`
- `valueOf(Object o)` returns `String`
- `valueOf(boolean bln)` returns `String`
- `valueOf(char c)` returns `String`
- `valueOf(char[] chars)` returns `String`
- `valueOf(double d)` returns `String`
- `valueOf(float f)` returns `String`
- `valueOf(int i)` returns `String`
- `valueOf(long l)` returns `String`
- `valueOf(char[] chars, int i, int il)` returns `String`

Below the list of methods, the keyword `class` is visible. At the bottom of the editor, the text `String.` is shown, followed by a dropdown menu displaying `java.lang.String`.



## Tại sao sử dụng wrapper class?

- \* Since java is object oriented language in which every single element should be treated as object.
- \* Primitive data types which are not actual objects and we cannot pass them by reference, they are passed by value and also we cannot make two references which refer to same data.
- \* Java only uses these primitive data types for performance reasons and hence there should a way in which we can convert them into objects and for this designers create Wrapper Classes.

Basically there are two important uses of wrapper classes:

- \* To convert a primitive data types into objects, that is to give them an object form.
- \* To convert strings into data types which is known as the parsing operations in which various methods such as `parseInt()`, `parseFloat()` etc. are used.

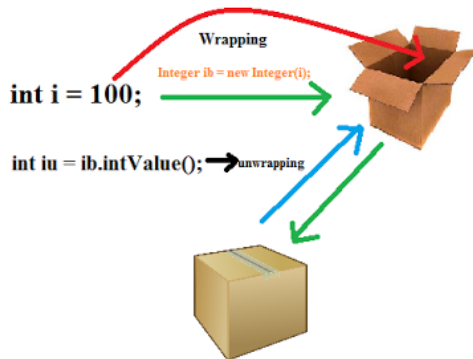
# Wrapper class

List of wrapper classes:

Data Type	Wrapper Class
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean

# Wrapper class

## Wrapping & Unwrapping



# Wrapper class

In the new versions of java there is way for autoBoxing and autoUnboxing:

```
//AutoBoxing  
int i = 100;  
Integer ib = i;  
//AutoUnboxing  
int iu = ib;
```

—Hết—