Chào mừng tới lớp Java core cho thành viên CLB HIT 2021

Leader: Nguyễn Đình Huân

Support:

- Bùi Việt Hoàng
- Trần Khắc Bình Dương
- Ngô Ngọc Sáng
- Tô Văn Sang
- Phạm Ánh Trường
- Nguyễn Nhật Minh
- Nguyễn Văn Thùy
- Nguyễn Xuân Giang
- Hà Văn Phòng
- Vũ Văn Doan
- Nguyễn Việt Trung

Lộ trình học: Huân mở cái doc ra cho mn xem
Lộ trình Java_HIT_2021.docx - Google Tài liệu

Làm quen với GitHub

Leader giới thiệu và hướng dẫn mọi người.

Các lệnh:

git config --global user.email "email-github" === config email github
git config --global user.name "github-name" === config name github
git config --global --list === xem list config
git init === tạo file .git
git status === xem status
git add . === add tất cả các file lên github (git add tenfolder -- add folder, git add abc.xyz -- add file)
git commit -m"" === commit file đã add
git log --oneline === xem log
git remote add origin https://..... === thêm remote
git remote -v === xem remote
git push origin master === push lên github
Ngoài ra còn có config .gitignore để loại bỏ các file - folder không muốn push lên.

Tổng quan về Java

Giới thiệu về Java

Java là gì?

- Java là một ngôn ngữ lập trình cao cấp, ban đầu được phát triển bởi Sun Microsystems và được phát hành vào năm 1995. Java chạy trên nhiều nền tảng, chẳng hạn như Windows, Mac OS và các phiên bản khác nhau của UNIX.
- Phiên bản mới nhất của Java Standard Edition là Java SE 17. Với sự tiến bộ của Java và sự phổ biến rộng rãi của nó, nhiều cấu hình đã được xây dựng để phù hợp với nhiều loại nền tảng khác nhau.

Biểu tượng của Java

- Java là tên gọi của một hòn đảo ở Indonesia, Đây là nơi nhóm nghiên cứu phát triển đã chọn để đặt tên cho ngôn ngữ lập trình Java trong một chuyến đi tham quan và làm việc trên hòn đảo này.
- Hòn đảo Java này là nơi rất nổi tiếng với nhiều khu vườn trồng cafe, đó chính là lý do chúng ta thường thấy biểu tượng ly café trong nhiều sản phẩm phần mềm, công cụ lập trình Java của Sun cũng như một số hãng phần mềm khác đưa ra.

Đặc trưng của Java

- Hướng đối tượng – Trong Java, mọi thứ đều là một Object.Java có thể dễ dàng mở rộng vì nó được dựa trên mô hình Object.
- Độc lập nền tảng – Không giống nhiều ngôn ngữ lập trình khác như C và C ++, khi Java được biên dịch, nó không được biên dịch vào nền tảng máy tính cụ thể, thay vào đó là mã byte nền tảng độc lập. Mã byte này được phân phát trên web và được thông dịch bởi Virtual Machine (JVM) trên nền tảng nào đó mà nó đang chạy.
- Đơn giản – Java được thiết kế rất dễ học. Nếu bạn hiểu khái niệm cơ bản của OOP Java, bạn sẽ rất dễ làm chủ nó.
- Bảo mật – Với tính năng an toàn của Java, nó cho phép phát triển các hệ thống không có virus. Các kỹ thuật xác thực dựa trên key mã hoá khóa công khai.

Đặc trưng của Java

- Di động – Là kiến trúc trung lập và không bị phụ thuộc làm cho Java có thể mang đi dễ dàng. Trình biên dịch trong Java được viết bằng ANSI C với khả năng di chuyển sạch, đó là một tập hợp con POSIX.
- Mạnh mẽ – Java nỗ lực để loại trừ các tình huống dễ bị lỗi bằng cách nhấn mạnh việc kiểm tra lỗi thời gian biên dịch và kiểm tra thời gian chạy.
- Đa luồng – Với tính năng đa luồng của Java, có thể viết các chương trình có thể thực hiện nhiều tác vụ đồng thời. Tính năng thiết kế này cho phép các nhà phát triển xây dựng các ứng dụng tương tác có thể chạy trơn tru.

Đặc trưng của Java

- Phiên dịch – Mã byte Java được dịch trực tiếp tới các hướng dẫn máy tính và không được lưu trữ ở bất cứ đâu. Quá trình phát triển nhanh hơn và phân tích hơn.
- Hiệu năng cao – Với việc sử dụng trình biên dịch Just-In-Time, Java cho phép thực hiện chương trình với hiệu năng cao.
- Phân phối – Java được thiết kế cho môi trường phân tán của internet.
- Năng động – Java được xem là năng động hơn C hoặc C ++ vì nó được thiết kế để thích nghi với môi trường đang phát triển.
- Các chương trình Java có thể mang một lượng lớn thông tin run-time, có thể được sử dụng để xác minh và giải quyết các truy cập vào các đối tượng trong thời gian chạy.

Một số khái niệm

- SDK (Software Development Kit) thường là khái niệm chỉ một bộ công cụ lập trình, tiện ích, tài liệu và các thư viện API. Đối với java ta có JDK
- JVM (Java virtual machine) - Máy ảo java.
- JRE (Java Runtime Environment) – môi trường thực thi Java, cung cấp các Java API, máy ảo Java (Java Virtual Machine hay JVM) và các thành phần cần thiết khác.
- Java Runtime library là thư viện các class đã được compiled sẵn và đặt trong file jre\lib\rt.jar. Các class java.lang.String, java.lang.Object, ... nằm trong gói jar này.
- JRE = JVM + Java Runtime Library: bạn sẽ thấy java.exe, javaw.exe và rt.jar trong thư mục cài đặt jre.
- JDK = JRE + tools (javac.exe, keytool.exe, ...) + document (help, samples,..) + src.zip.

Java làm được những gì

8 Things You Can Create with Java

- Mobile Applications.
- Internet of Things (IoT) Devices.
- Cloud Applications.
- Web Applications.
- Chatbots.
- Games.
- Enterprise Applications.
- Scientific Applications.

Cấu trúc chương trình

```java
package demo;
//Thư viện được sử dụng (cần phải import thư viện thì mới sử dụng được)
import java.util.Map;
import java.util.Scanner;
import java.util.Set;

public class Main {
    //Main
    public static void main(String[] args) {
        System.out.println("Hello World!!!");
    }

    //Các hàm
    void function(){}

}
```

Xuất

Sử dụng câu lệnh System.out.print(nội dung hiển thị);

```
System.out.println("Hello World!!!");
String str = "!!!!!";
System.out.println("Hello " + "World" + str);
int songuyen = 10;
System.out.println("Số nguyên là: " + songuyen);
```

```
Hello World!!!
Hello World!!!!!
Số nguyên là: 10
```

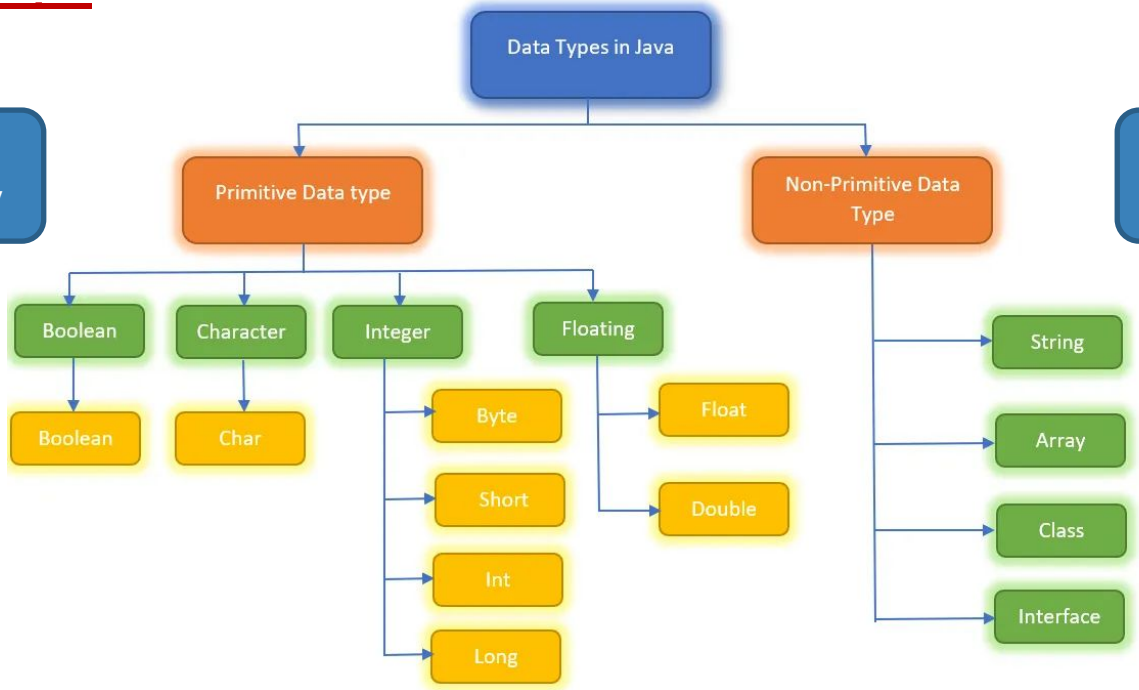
Định dạng xuất

Sử dụng ‘ .printf ’ hoặc ‘ .format ’

```
long n = 461012;
System.out.format("%n%d%n", n); // --> "461012"
System.out.format("%08d%n", n); // --> "00461012"
System.out.format("%8d%n", n); // ---> " 461012"
System.out.format("%+8d%n", n); // --> " +461012"
System.out.format("%,8d%n", n); // --> " 461,012"
System.out.format("%+,8d%n%n", n); // --> "+461,012"
```

Kiểu dữ liệu

Kiểu dữ liệu nguyên thủy



Kiểu dữ liệu không nguyên thủy

Kiểu	Mô tả	Mặc định	Cỡ	Ví dụ
boolean	true hoặc false	FALSE	1 bit	true, false
byte	Số nguyên từ -128 .. 127	0	8 bits	123
char	Ký tự Unicode	\u0000	16 bits	'a', '\u0041', '\101', '\\', '\'', '\n', 'ß'
short	Số nguyên giá trị từ -32768 - 32767	0	16 bits	1000
int	Số nguyên -2,147,483,648 - 2,147,483,647	0	32 bits	-2, -1, 0, 1, 2
long	Số nguyên dài	0	64 bits	-2L, -1L, 0L, 1L, 2L
float	Số thực	0	32 bits	1.23e100f, -1.23e-100f, .3f, 3.14F
double	Số thực	0	64 bits	1.23456e300d, -1.23456e-300d, 1e1d

Toán tử

Giả sử chúng ta có biến số nguyên a = 10 và b = 20.

Toán tử	Mô tả	Ví dụ
+	Cộng Trả về giá trị là tổng của hai toán hạng	a + b sẽ là 30
-	Trừ Trả về kết quả là hiệu của hai toán hạng.	a - b sẽ là -10
*	Nhân Trả về giá trị là tích của hai toán hạng.	a * b sẽ là 200
/	Chia Trả về giá trị là thương của phép chia.	b / a sẽ là 2
%	Phép lấy modul Giá trị trả về là phần dư của phép chia	b % a sẽ là 0
++	Tăng dần Tăng giá trị của biến lên 1. Ví dụ a++ tương đương với a = a + 1	a++ sẽ là 11
--	Giảm dần Giảm giá trị của biến 1 đơn vị. Ví dụ a-- tương đương với a = a - 1	a-- sẽ là 9

+=	Cộng và gán giá trị Cộng các giá trị của toán hạng bên trái vào toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c += a tương đương c = c + a	a += 2 sẽ là 12
-=	rừ và gán giá trị Trừ các giá trị của toán hạng bên trái vào toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c -= a tương đương với c = c - a	a -= 2 sẽ là 8
*=	Nhân và gán Nhân các giá trị của toán hạng bên trái với toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c *= a tương đương với c = c*a	a *= 2 sẽ là 20
/=	Chia và gán Chia giá trị của toán hạng bên trái cho toán toán hạng bên phải và gán giá trị trả về vào toán hạng bên trái. Ví dụ c /= a tương đương với c = c/a	a /= 2 sẽ là 5
%=	Lấy số dư và gán Chia giá trị của toán hạng bên trái cho toán toán hạng bên phải và gán giá trị số dư vào toán hạng bên trái. Ví dụ c %= a tương đương với c = c%a	a %= 8 sẽ là 2

Toán tử quan hệ

Toán tử	Mô tả
==	So sánh bằng
	Toán tử này kiểm tra sự tương đương của hai toán hạng
!=	So sánh khác
	Toán tử này kiểm tra sự khác nhau của hai toán hạng
>	Lớn hơn
	Kiểm tra giá trị của toán hạng bên phải lớn hơn toán hạng bên trái hay không
<	Nhỏ hơn
	Kiểm tra giá trị của toán hạng bên phải có nhỏ hơn toán hạng bên trái hay không
>=	Lớn hơn hoặc bằng
	Kiểm tra giá trị của toán hạng bên phải có lớn hơn hoặc bằng toán hạng bên trái hay không
<=	Nhỏ hơn hoặc bằng
	Kiểm tra giá trị của toán hạng bên phải có nhỏ hơn hoặc bằng toán hạng bên trái hay không

Toán tử bit

Toán tử	Mô tả
~	Phủ định NOT
	Trả về giá trị phủ định của một bít.
&	Toán tử AND
	Trả về giá trị là 1 nếu các toán hạng là 1 và 0 trong các trường hợp khác
|	Toán tử OR
	Trả về giá trị là 1 nếu một trong các toán hạng là 1 và 0 trong các trường hợp khác.
^	Toán tử Exclusive OR
	Trả về giá trị là 1 nếu chỉ một trong các toán hạng là 1 và trả về 0 trong các trường hợp khác.
>>	Dịch phải
	Chuyển toàn bộ các bít cuả một số sang phải một vị trí, giữ nguyên dấu của số âm. Toán hạng bên trái là số bị dịch còn số bên phải chỉ số vị trí mà các bít cần dịch.
<<	Dịch trái
	Chuyển toàn bộ các bít cuả một số sang trái một vị trí, giữ nguyên dấu cuả số âm. Toán hạng bên trái là số bị dịch còn số bên phải chỉ số vị trí mà các bít cần dịch.

Toán tử logic

Toán tử	Mô tả
&&	Toán tử và (AND)
	Trả về một giá trị "Đúng" (True) nếu chỉ khi cả hai toán tử có giá trị "True"
||	Toán tử hoặc (OR)
	Trả về giá trị "True" nếu ít nhất một giá trị là True
^	Toán tử XOR
	Trả về giá trị True nếu và chỉ nếu chỉ một trong các giá trị là True, các trường hợp còn lại cho giá trị False (sai)
!	Toán tử phủ định (NOT)
	Toán hạng đơn tử NOT. Chuyển giá trị từ True sang False và ngược lại.

Toán tử 3 ngôi

- Toán tử ba ngôi, ký hiệu là (?:). Nó nhận vào ba toán hạng
- Cú pháp:

Toán tử ba ngôi

```
expression1 ? expression2 : expression3;
```

• Ví dụ: Tìm giá trị lớn nhất trong 2 số nguyên

```
max = ((a > b) ? a : b);
```

Tương đương:

```
if (a>b)
    max=a;
else
    max =b;
```

Thứ tự ưu tiên

Toán tử	Mô tả
1	Các toán tử đơn như +,-,++,--
2	Các toán tử số học và các toán tử dịch như *,/,+,-,<<,>>
3	Các toán tử quan hệ như >,<,>=,<=,= =,!=
4	Các toán tử logic và Bit như &&,||,&,|,^
5	Các toán tử gán như =,*=,/=,+=,-=

CẤU TRÚC ĐIỀU KHIỂN

- Cấu trúc rẽ nhánh
  - If..., if...else..., switch...case...
- Cấu trúc lặp
  - for..., while..., do...while...
- Các lệnh chuyển điều khiển
  - break, continue, return.

CẤU TRÚC ĐIỀU KHIỂN

- Ra quyết định thực hiện một khối lệnh khi có một điều kiện đúng (rẽ nhánh).
- Lặp lại một khối lệnh khi một điều kiện còn đúng (vòng lặp).
- Tất cả các môi trường phát triển ứng dụng đều cung cấp một cách thức ra quyết định (decision - making) được gọi là các câu lệnh điều khiển luồng mà nó chỉ đạo thực thi ứng dụng.

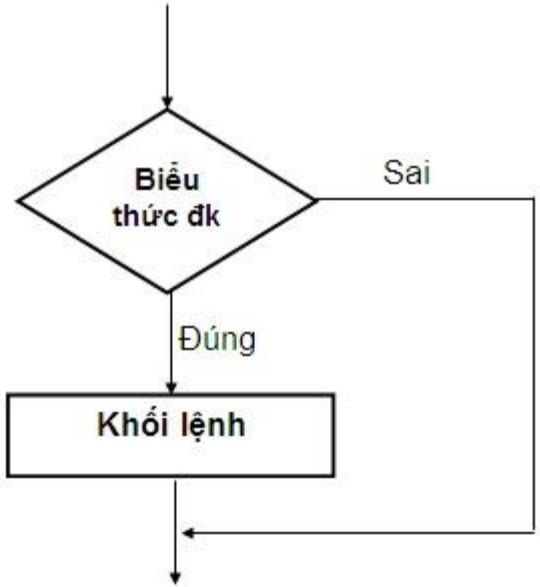
Java hỗ trợ các loại câu lệnh điều khiển rẽ nhánh sau:

- Câu lệnh if
- Câu lệnh if-else-if
- Câu lệnh switch-case

Câu lệnh if

Dạng khuyết

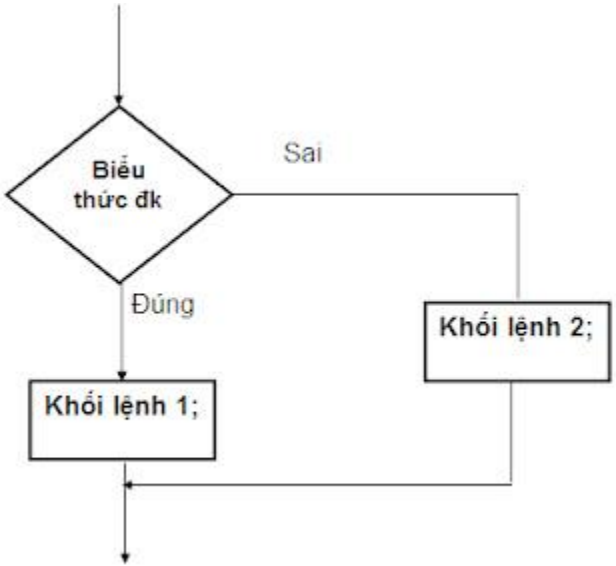
```
If  (biểu thức điều kiện)
    {
       Khối lệnh;
    }
```



Câu lệnh if

Dạng đầy đủ

```
if (biểu thức điều kiện)
   {
      Khối lệnh 1;
   }
else
   {
      Khối lệnh 2;
   }
```



If-else lồng nhau

- Cấu trúc đa câu lệnh if được biết đến như là if-else-if từng bậc
- Khi điều kiện đúng được tìm thấy, câu lệnh được liên kết với điều kiện đúng được thực thi.

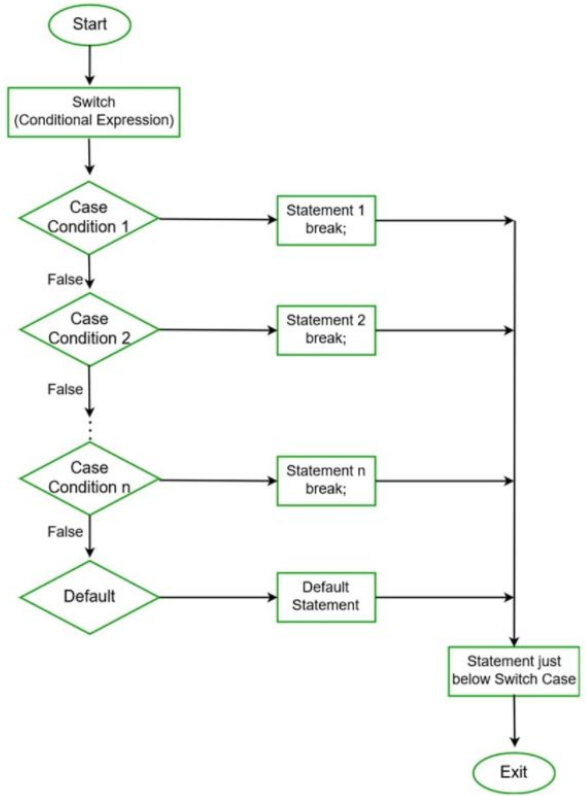
```
if (điều kiện 1) {
        Khối lệnh 1;
} else if(điều kiện 2) {
        Khối lệnh 2;
} else {
        Khối lệnh F;
}
```

- Ví dụ: bài toán tính tiền điện sinh hoạt của một hộ gia đình trong một tháng.

Câu lệnh switch-case

- Câu lệnh switch – case được sử dụng khi một biến cần phải được so sánh trở lại với các giá trị khác.
- Câu lệnh switch thực thi case tương ứng với giá trị của biểu thức.

```
// variable: Một biến để kiểm tra.
switch ( variable ) {
    case  value1:
        // Làm gì đó tại đây ...
        break;
    case  value2:
        // Làm gì đó tại đây ...
        break;
    default:
        // Làm gì đó tại đây ...
}
```



Ví dụ:

```java
int day = 4;
String str;
switch (day){
    case 0: str = "Sunday";
        break;
    case 1: str = "Monday";
        break;
    case 2: str = "Tuesday";
        break;
    case 3: str = "Wednesday";
        break;
    case 4: str = "Thursday";
        break;
    case 5: str = "Friday";
        break;
    case 6: str = "Saturday";
        break;
    default: str = "Invalid day";
}
System.out.println(str);
```

Các câu lệnh lặp được hỗ trợ bởi ngôn ngữ lập trình Java

– while
– do-while
– for

Câu lệnh While

- Câu lệnh while được sử dụng để thực thi một hay nhiều câu lệnh trong khi điều kiện liên quan là “Đúng”(true).
- Điều kiện được kiểm tra trước khi các câu lệnh được thực thi.
- Cú pháp:

```
while (điều kiện)
{
    khối lệnh;
}
```

- Ví dụ: Đếm số chữ số của một số nguyên dương.

Các luật
– Các biến được sử dụng trong biểu thức điều kiện cần phải được khởi tạo ở trước vòng lặp.
– Thân của vòng lặp phải có một lệnh mà nó làm thay đổi giá trị của biến điều kiện.

Câu lệnh do_while

- Câu lệnh do-while kiểm tra điều kiện ở cuối của vòng lặp thay vì ở phía đầu để chắc chắn rằng vòng lặp được thực thi ít nhất một lần.
- Cú pháp:

```
do{
        khối lệnh;
} while (điều kiện);
```

- Ví dụ: Kiểm tra tính đúng đắn của dữ liệu nhập vào.

Câu lệnh for

- Câu lệnh for giống như câu while ở chức năng của nó.
- Khi số lần lặp được biết trước, câu lệnh for được sử dụng.
- Cú pháp:

```
for (exp1; exp2; exp3){
    khối lệnh;
}
```

- Trong đó:
  – exp1 là biểu thức khởi tạo giá trị của biến điều khiển vòng lăp.
  – exp2 là biểu thức kiểm tra điều kiện dừng vòng lặp.
  – exp3 là biểu thức định nghĩa cách thức thay đổi giá trị của biến điều khiển vòng lặp.

- Lệnh khởi tạo sẽ được chạy trước tiên và chỉ 1 lần duy nhất
- Sau đó biểu thức điều kiện sẽ được xem xét, nếu điều kiện đúng thì khối lệnh trong thân vòng lặp sẽ được thực thi. Nếu điều kiện sai thì thoát vòng lặp
- Sau khi thực thi xong khối lệnh trong thân vòng lặp, vòng for quay trở lên, chạy lệnh tăng giảm
- Sau đó lại ktra đk và chạy tiếp than vòng lặp nếu dk đúng. Thoát nếu dk sai

For lồng nhau

- Đặt một vòng lặp bên trong thân của một vòng lặp khác được gọi là lồng nhau (nested loops).
- Khi lồng hai vòng lặp, vòng lặp bên ngoài sẽ điều khiển số lần thực thi vòng lặp bên trong.
- Các vòng lặp lồng nhau hay dùng nhất là vòng lặp for.

Các lện chuyển điều khiển

- break;
- continue;
- return;