THCS 4: LẬP TRÌNH CƠ BẢN VỚI JAVA

Lớp và đối tượng

Nguyễn Thị Tâm
nguyenthitam.hus@gmail.com

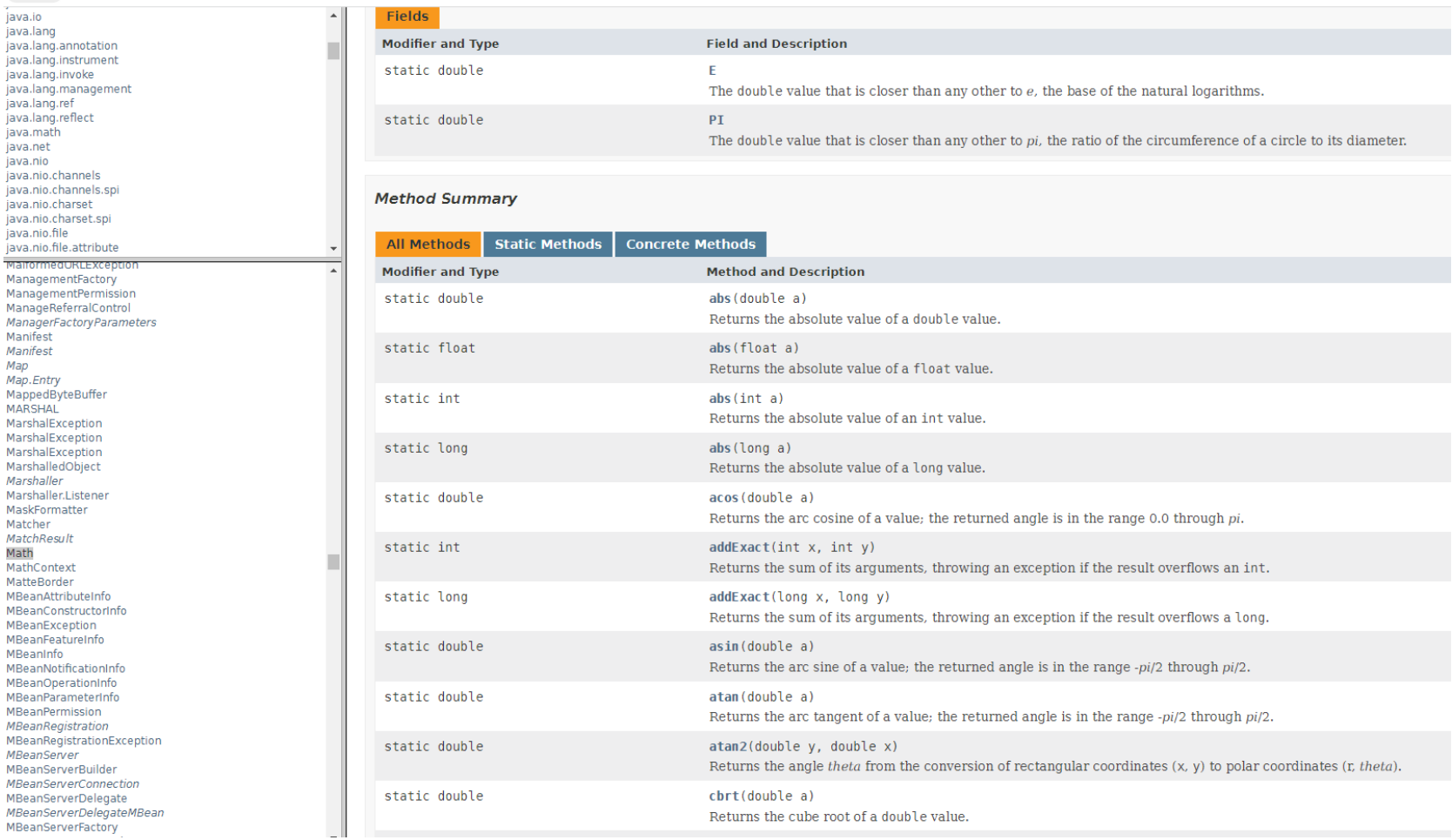
Ngày 8 tháng 11 năm 2024

Application Programming Interface - API

API

- API: API là một tập hợp các quy tắc và công cụ cho phép các phần mềm ứng dụng tương tác với nhau.
- Tài liệu, SE8: `https://docs.oracle.com/javase/8/docs/api/`
- API trong Java: là một thư viện các lớp, gói và giao diện Java có sẵn
  - Thư viện được tổ chức thành các gói (package)
  - Lớp **Scanner** là một phần của Java API
  - Tập trung vào một số thư viện hay dùng: Math, Integer, Double, String (gói java.lang)
  - Tra cứu thông tin (đầu vào/ra, phiên bản, mục đích ...) thông qua tài liệu được kết xuất kèm (theo mẫu ở phần Java documentation).

Lớp Math (1)


java.io
java.lang
java.lang.annotation
java.lang.instrument
java.lang.invoke
java.lang.management
java.lang.ref
java.lang.reflect
java.math
java.net
java.nio
java.nio.channels
java.nio.channels.spi
java.nio.charset
java.nio.charset.spi
java.nio.file
java.nio.file.attribute
MalformedURLException
ManagementFactory
ManagementPermission
ManageReferralControl
ManagerFactoryParameters
Manifest
Manifest
Map
Map.Entry
MappedByteBuffer
MARSHAL
MarshalException
MarshalException
MarshalException
MarshalledObject
Marshaller
Marshaller.Listener
MaskFormatter
Matcher
MatchResult
Math
MathContext
MatteBorder
MBeanAttributeInfo
MBeanConstructorInfo
MBeanException
MBeanFeatureInfo
MBeanInfo
MBeanNotificationInfo
MBeanOperationInfo
MBeanParameterInfo
MBeanPermission
MBeanRegistration
MBeanRegistrationException
MBeanServer
MBeanServerBuilder
MBeanServerConnection
MBeanServerDelegate
MBeanServerDelegateMBean
MBeanServerFactory
Fields
Modifier and Type
Field and Description
static double
E
The double value that is closer than any other to e, the base of the natural logarithms.
static double
PI
The double value that is closer than any other to pi, the ratio of the circumference of a circle to its diameter.
Method Summary
All Methods
Static Methods
Concrete Methods
Modifier and Type
Method and Description
static double
abs(double a)
Returns the absolute value of a double value.
static float
abs(float a)
Returns the absolute value of a float value.
static int
abs(int a)
Returns the absolute value of an int value.
static long
abs(long a)
Returns the absolute value of a long value.
static double
acos(double a)
Returns the arc cosine of a value; the returned angle is in the range 0.0 through pi.
static int
addExact(int x, int y)
Returns the sum of its arguments, throwing an exception if the result overflows an int.
static long
addExact(long x, long y)
Returns the sum of its arguments, throwing an exception if the result overflows a long.
static double
asin(double a)
Returns the arc sine of a value; the returned angle is in the range -pi/2 through pi/2.
static double
atan(double a)
Returns the arc tangent of a value; the returned angle is in the range -pi/2 through pi/2.
static double
atan2(double y, double x)
Returns the angle theta from the conversion of rectangular coordinates (x, y) to polar coordinates (r, theta).
static double
cbrt(double a)
Returns the cube root of a double value.

Lớp Math (2)

- Pakage: java.lang
- Một số phương thức trong lớp Math:
  - abs()
  - max()
  - min()
  - pow()
  - sqrt()
  - ..

Lớp Math (3)

- Lớp Math có hai thuộc tính (class attributes)

Fields

Modifier and Type	Field and Description
static double	**E** The double value that is closer than any other to *e*, the base of the natural logarithms.
static double	**PI** The double value that is closer than any other to *pi*, the ratio of the circumference of a circle to its diameter.

- Sử dụng thuộc tính của lớp như thế nào

```
double area = Math.PI * Math.pow(radius,2);
```

Lớp Math (4)

```
/* To find the area of the largest circle inscribed in a square,
    given the square's area.*/
   import.java.util.Scanner;
   public class TestMath{
       public static void main (String []args)
       {
           Scanner sc = new Scanner (System.in);
           System.out.println("Enter area of a square: ");
           double areaSquare = sc.nextDouble();

           double radius = Math.sqrt(areaSquare)/2;
           double areaCircle = Math.PI * Math.pow(radius, 2);

           System.out.println("Area of circle = %.2f\n", areaCircle);

       }
   }
```

Lớp String (1)

- API documentation: `https://docs.oracle.com/javase/8/docs/api/`
- `import java.lang.String`

Method Summary

All Methods | **Static Methods** | **Instance Methods** | **Concrete Methods** | **Deprecated Methods**

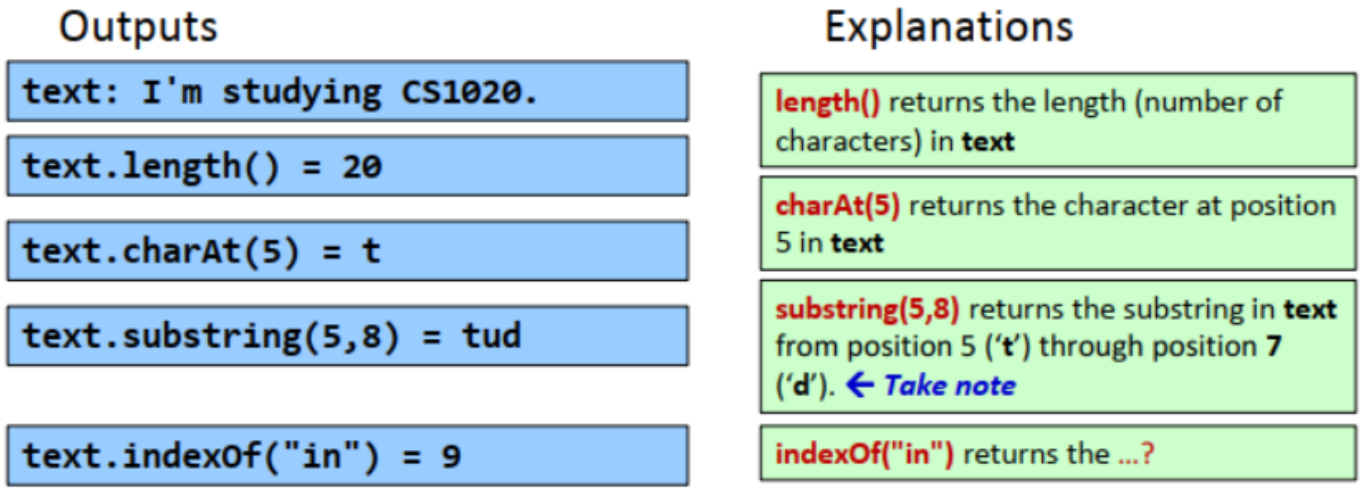
Modifier and Type	Method and Description
char	**charAt**(int index) Returns the char value at the specified index.
int	**codePointAt**(int index) Returns the character (Unicode code point) at the specified index.
int	**codePointBefore**(int index) Returns the character (Unicode code point) before the specified index.
int	**codePointCount**(int beginIndex, int endIndex) Returns the number of Unicode code points in the specified text range of this String.
int	**compareTo**(String anotherString) Compares two strings lexicographically.
int	**compareToIgnoreCase**(String str) Compares two strings lexicographically, ignoring case differences.
String	**concat**(String str) Concatenates the specified string to the end of this string.
boolean	**contains**(CharSequence s) Returns true if and only if this string contains the specified sequence of char values.
boolean	**contentEquals**(CharSequence cs) Compares this string to the specified CharSequence.
boolean	**contentEquals**(StringBuffer sb) Compares this string to the specified StringBuffer.
static String	**copyValueOf**(char[] data) Equivalent to **valueOf(char[])**.
static String	**copyValueOf**(char[] data, int offset, int count) Equivalent to **valueOf(char[], int, int)**.
boolean	**endsWith**(String suffix) Tests if this string ends with the specified suffix.
boolean	**equals**(Object anObject) Compares this string to the specified object.

Lớp String (2)

```
public class TestString {
    public static void main (String []args){
        String text = new String ("I'm studyding CS1020");
        // or String text = "I'm studyding CS1020"
        System.out.println("text: " + text);
        System.out.println("text.length() = " + text.length());
        System.out.println("text.charAt(5) = " + text.charAt(5));
        System.out.println("text.substring(5, 8) = "
            +text.substring(5, 8 );
        System.out.println("text.indexOf("in") = "+
            text.indexOf("in"));
    }

}
```

Lớp String



- string là các đối tượng, không sử dụng == để kiểm tra hai chuỗi có bằng nhau hay không?
- Sử dụng phương thức equals() để kiểm tra

```
String str1 = sc.nextLine();
String str2 = sc.nextLine();
System.out.println(str1.equals(str2));
```

Tạo Java documentation

- Java documentation (javadoc): công cụ sinh tài liệu hướng dẫn - thuyết minh chương trình
- Quy tắc cú pháp:
  - Phần thông tin chương trình: Tác giả, phiên bản, ngày tháng cập nhật
  - Phần mô tả các thành phần (hàm/phương thức) trong chương trình: Tên, tham biến, giá trị trả lại, mô tả hoạt động
- Cú pháp câu lệnh ở cửa sổ lệnh:

  javadoc <FILE_NAME.java>
- Xem thêm chi tiết tại link: http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html

Lớp và đối tượng (1)

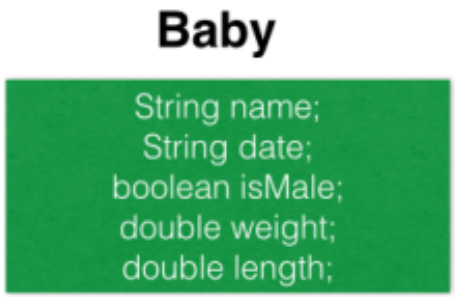
- Mô hình hoá các đối tượng trong thế giới thực
  - Baby: tên, ngày sinh, giới tính, cân nặng,...

Lớp và đối tượng (1)

- Mô hình hoá các đối tượng trong thế giới thực
  - Baby: tên, ngày sinh, giới tính, cân nặng,...
  - Vehicle: tên, hãng sản xuất, năm sản xuất,...

Lớp và đối tượng (1)

- Mô hình hoá các đối tượng trong thế giới thực
  - Baby: tên, ngày sinh, giới tính, cân nặng,...
  - Vehicle: tên, hãng sản xuất, năm sản xuất,...
  - Student: tên, ngày sinh, giới tính, mã sinh viên,...
- Chương trình thực hiện tương tác giữa các đối tượng
- Chương trình Java thao tác trên các dữ liệu kiểu
  - Nguyên thuỷ: `int`, `double`, `char`,...
  - Đối tượng: `String`,...



Khái niệm lớp

- Lớp (class) xác định kiểu đối tượng: dữ liệu của đối tượng, các thao tác trên dữ liệu đó
- Tại sao sử dụng lớp
  - Có thể hiểu lớp là kiểu dữ liệu có cấu trúc
  - Nếu chỉ sử dụng kiểu dữ liệu cơ bản: quản lý 500 babies ???

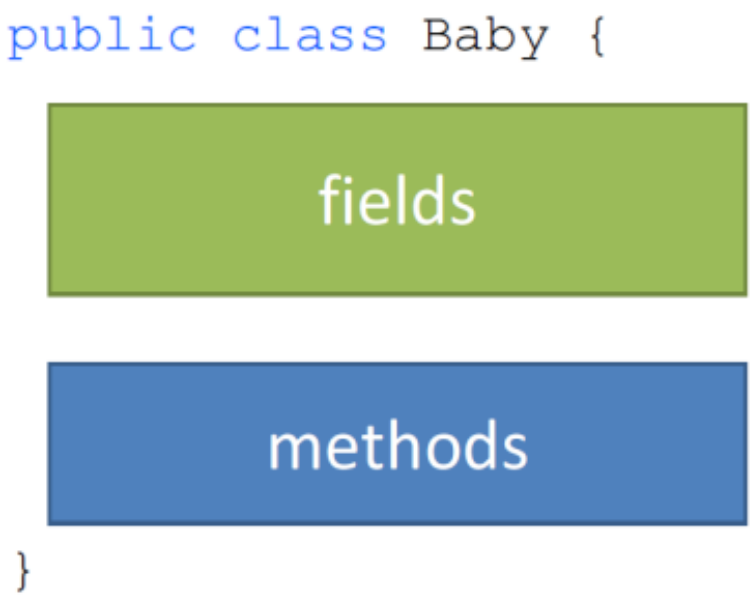
```
String name[];
String date[];
boolean isMale[];
double weight[];
double height[];
```

Sắp xếp 500 trẻ nhỏ theo thứ tự tên?

Định nghĩa lớp

```
public class Baby{
    String name;
    String date;
    boolean isMale;
    double weight;
    double height;
    public void sayHi(){
        // say hi to everybody
    }
}
```

Các thành phần khai báo trong lớp



Lưu ý

- Tên lớp: quy ước luôn viết hoa chữ cái đầu.
- Các tên lớp thư viện đã từng dùng: String, System, Math, Scanner
- Mỗi lớp được lưu trong một tệp riêng (<ClassName>.java)
- Lớp chứa phương thức `public static void main (String [] args)` là lớp để thực thi

các trường dữ liệu

- Là các biến lưu trữ các thuộc tính của mỗi đối tượng
- Khai báo như các biến

```
public class Baby {

    TYPE var_name;
    TYPE var_name = some_value;


}
```

các trường dữ liệu

```
public class Baby{
    String name;
    String date;
    boolean isMale;
    double weight = 3.2;
    double length;

    // methods
}
```

Tạo đối tượng của lớp

- `Baby aBaby = new Baby();`
- Hàm dựng: được gọi khi tạo đối tượng, thường dùng để khởi tạo giá trị cho các thuộc tính
- Mọi lớp ClassName đều có hàm dựng ngầm định là ClassName(); Khi sử dụng hàm dựng ngầm định, các thuộc tính của đối tượng có giá trị như trong khai báo/khởi gán
- Có thể viết nhiều hàm dựng phục vụ việc tạo đối tượng khác nhau tuỳ mục đích

  `ClassName(arg)`

Hàm dựng

```
public class ClassName{
    // Constructor definition: no return value
    // No "static" keyword
    public ClassName(){

    }
    public ClassName(args){

    }
}
// Use constructors to create objects in another method

ClassName obj1 = new ClassName();
ClassName obj2 = new ClassName(args);
```

Hàm dựng của lớp Baby

```
public class Baby{
   String name;
   boolean isMale;
   // other fields/attributes..
   public Baby (String myname, boolean maleBaby){
        name = myname
        isMale = maleBaby;
   }
```

Các phương thức

- Hàm dựng
- Các phương thức trong lớp thông thường sẽ không phải phương thức tĩnh (có từ khoá **static**)

```
public class Baby{
        String name = "Alex";
        double weight = 3.2
        //...
        public void sayHi(){
            System.out.println("Hi, my name is " + name);
        }
        public void eat(double foodWeight){
            if (foodWeight >= 0 && foodWeight < weight)
                weight = weight + foodWeight;
        }
    }
```

Sử dụng lớp

- Định nghĩa lớp: `public class Baby..`
- Tạo ra các thực thể (đối tượng) của lớp
  - Baby bb1 = new Baby ("Alex", true); bb2 = new Baby ("Julia", false);
- Truy cập trường dữ liệu: Object.FieldName

```
Baby bb1 = new Baby("Alex", true);
System.out.println(bb1.name);
System.out.println(bb1.weight);
}
```

- Gọi phương thức: Obj.MethodName([args])

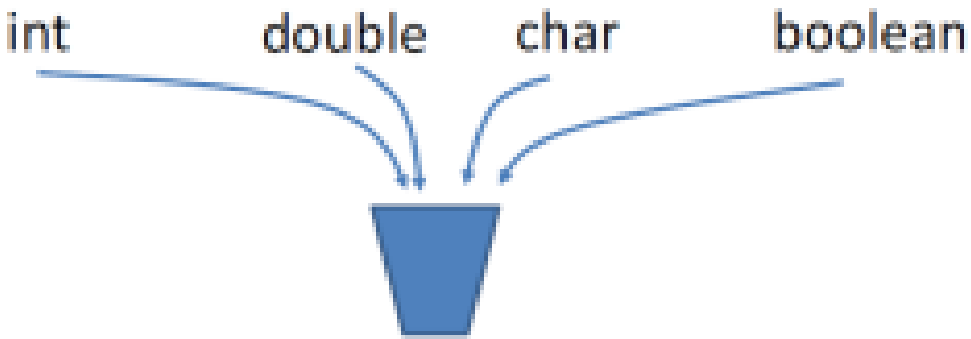
```
Baby bb1 = new Baby ("Alex", true);
bb1.sayHi();
bb1.eat(1);
```

Tham chiếu và giá trị

Tham chiếu và giá trị (1)

Kiểu dữ liệu trong Java: Kiểu nguyên thuỷ (primitive types)

- boolean, int, double, char,...
- Giá trị thực tế được lưu trực tiếp trong biểu diễn



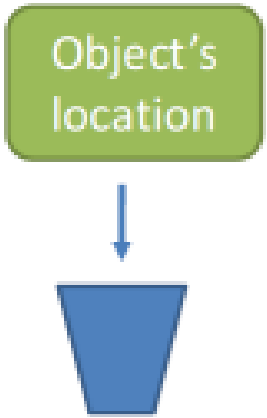
Tham chiếu và giá trị (2)

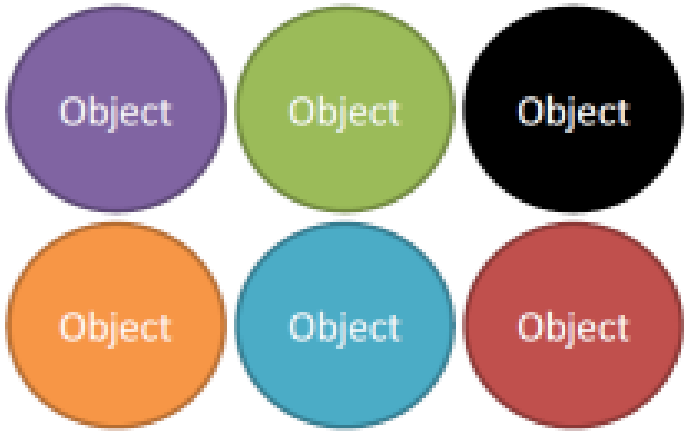
Kiểu dữ liệu trong Java: Kiểu tham chiếu (reference types)

- Kiểu mảng và các kiểu đối tượng: int[] , String,
- Biến lưu trữ địa chỉ của vùng nhớ lưu trữ đối tượng

Tham chiếu và giá trị (3)

- Biến lưu trữ địa chỉ của vùng nhớ lưu trữ đối tượng





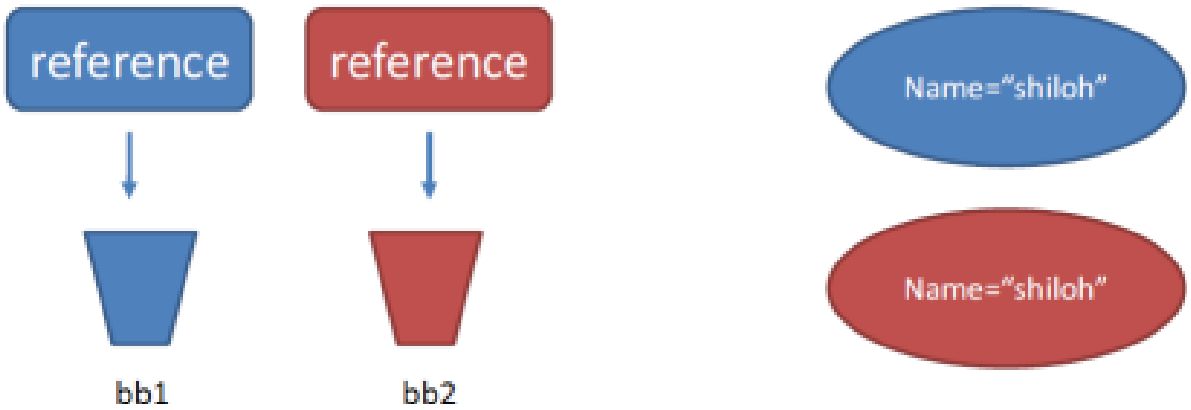
- Phép toán ==

```
Baby bb1 = new Baby ("An")
Baby bb2 = new Baby ("An")
bb1 == bb2
???
```

Tham chiếu và giá trị (4)

- Biến lưu trữ địa chỉ của vùng nhớ lưu trữ đối tượng
- Phép toán ==
  - Baby bb1 = new Baby("An");
  - Baby bb1 = new Baby("An")
  - bb1==bb2

FALSE



Tham chiếu và giá trị (5)

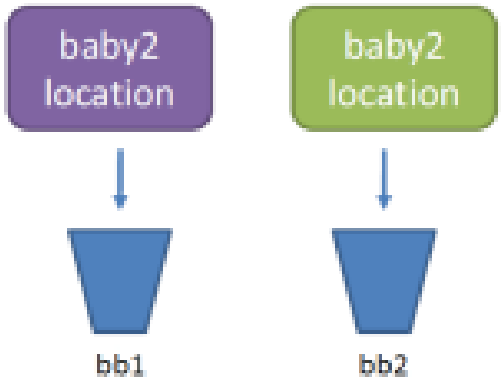
```
Baby mybaby = new Baby ("davy", true);
mybaby.name = "david"
```

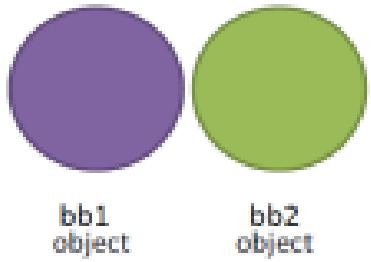
Tham chiếu và giá trị (6)

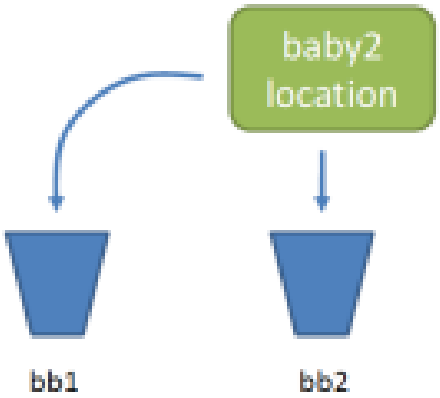
```
Baby mybaby = new Baby ("davy", true);
mybaby.name = "david"
```

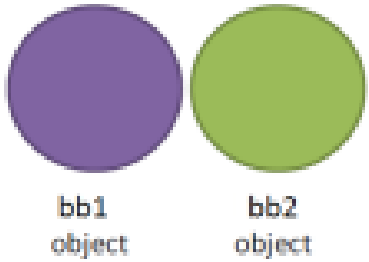
Tham chiếu và giá trị (7)

- Sử dụng phép gán = để cập nhật tham chiếu
- bb1 = bb2









Tham chiếu và giá trị (8)

```
public static void doSomething (int x, int []ys, Baby b){
    x  = 99;
    ys[0] = 99;
    b.name = "99";
}

//....in main
int i = 0;
int []j = {0};
Baby bb1 = new Baby ("50",true);
doSomething(i,j,bb1);
```

i = ?, j = ?, bb1 = ?

Biến và phương thức tĩnh

- Từ khoá: **static**
- Trường dữ liệu và phương thức tĩnh:
  - Thuộc tính và phương thức của lớp
  - Dùng chung cho tất cả các đối tượng của lớp được tạo ra
  - Có thể truy cập qua tên lớp, không cần tạo đối tượng
    - ClassName.staticField
    - ClassName.staticMethod(args)

Biến tĩnh

```
public static Baby{
    static int numBabiesMade = 0
}
//...
Baby.numbabiesMade = 100;
Baby bb1 = new Baby();
Baby bb2 = new Baby();
Baby.numBabiesMade = 2;
System.out.println(bb1.numBabiesMade); //???
System.out.println(bb1.numBabiesMade); //???
```

Phương thức tĩnh (2)

- Các phương thức không tĩnh có thể gọi đến phương thức tĩnh trong cùng một lớp
- Ngược lại, các phương thức tĩnh không để gọi tới phương thức không tĩnh trong cùng một lớp

```
public class Baby{
    String name = "Julien"

    public static void whoAmI()
    {
        System.out.println(name);
    }
}
```

- Phương thức main phải là phương thức tĩnh (giải thích tại sao?)
- Các phương thức đã dùng trong lớp Math là các phương thức tĩnh, trường dữ liệu *out* của lớp *System* là biến tĩnh.

Bài tập

- Tạo lớp Baby với 5 thuộc tính đã cho trên slides bài giảng, cùng với phương thức sayHi và eat đã cho trong bài giảng. Viết 3 hàm dựng:
  - Baby(),
  - Baby(String myName, bolean myGender),
  - Baby(String myName, String myDate, bolean myGender, double myWeight, double myLength)

  để tạo đối tượng trong 3 trường hợp: 1) không khởi gán giá trị; 2) Khởi gán tên và giới tính; 3) Khởi gán cả 5 thuộc tính (trường dữ liệu).
- Tạo lớp TestBaby trong đó có phương thức chính main và phương thức sortBabies(Baby[] babies). Trong phương thức main cần thực hiện:
  - Tạo ra một mảng 5 phần tử có tên là babies, mỗi phần tử là một đối tượng thuộc lớp Baby. Mỗi phần tử được tạo bằng cách nhập giá trị cả 5 thuộc tính rồi gọi hàm dựng với 5 đối đã xây dựng ở lớp Baby.
  - Sau đó, gọi đến phương thức sortBabies(Baby[] babies), phương thức này thực hiện sắp xếp mảng babies theo thứ tự tên em bé tăng dần (thứ tự từ điển).
  - In ra danh sách các em bé đã sắp xếp theo tên, với đầy đủ thông tin của 5 trường dữ liệu.