# Address:

<https://www.youtube.com/watch?v=5CA07hcesAQ>

# Inner Class trong 1 class:

## Code:

**public** **class** OuterClass {

**private** **int** x=200;

**class** InnerClass{

**private** **int** ans;

**public** **void** getValue(){

display();

System.***out***.println("Outer Class Value: " + x);

}

}

**public** **void** display() {

System.***out***.println("Outer Class Display");

}

**public** **void** accessInner(){

InnerClass obj = **new** InnerClass();

obj.ans = 500;

System.***out***.println("Inner Class Value: " + obj.ans);

}

**public** **static** **void** main(String[] args) {

OuterClass outobj = **new** OuterClass();

OuterClass.InnerClass inobj = outobj.**new** InnerClass();

OuterClass.InnerClass inobj2 =**new** OuterClass().**new** InnerClass();

inobj.getValue();

}

}

## Explanation:

Có 2 cách khai bao 1 inner class trong 1 class:

Cách 1:

OuterClass outobj = **new** OuterClass();

OuterClass.InnerClass inobj = outobj.**new** InnerClass();

Cách 2:

OuterClass.InnerClass inobj2 =**new** OuterClass().**new** InnerClass();

## Result:



## Nếu trong InnerClass ta khai báo thêm 1 hàm display thì sao:

**public** **class** OuterClass {

**private** **int** x=200;

**class** InnerClass{

**private** **int** ans;

**public** **void** getValue(){

display();

System.***out***.println("Outer Class Value: " + x);

}

**public** **void** display(){

System.***out***.println("Inner Class Value: " + x);

}

}

**public** **void** display() {

System.***out***.println("Outer Class Display");

}

**public** **void** accessInner(){

InnerClass obj = **new** InnerClass();

obj.ans = 500;

System.***out***.println("Inner Class Value: " + obj.ans);

}

**public** **static** **void** main(String[] args) {

OuterClass outobj = **new** OuterClass();

OuterClass.InnerClass inobj = outobj.**new** InnerClass();

OuterClass.InnerClass inobj2 =**new** OuterClass().**new** InnerClass();

inobj.getValue();

}

}



Như thế obj sẽ hiểu là ta sử dụng hàm display() nằm trong InnerClass

## Cách sử dụng hàm display ở bên ngoài:

**public** **class** OuterClass {

**private** **int** x=200;

**class** InnerClass{

**private** **int** ans;

**public** **void** getValue(){

display();

System.***out***.println("Outer Class Value: " + x);

}

**public** **void** display(){

OuterClass.**this**.display();

System.***out***.println("Inner Class Value: " + x);

}

}

**public** **void** display() {

System.***out***.println("Outer Class Display");

}

**public** **void** accessInner(){

InnerClass obj = **new** InnerClass();

obj.ans = 500;

System.***out***.println("Inner Class Value: " + obj.ans);

}

**public** **static** **void** main(String[] args) {

OuterClass outobj = **new** OuterClass();

OuterClass.InnerClass inobj = outobj.**new** InnerClass();

OuterClass.InnerClass inobj2 =**new** OuterClass().**new** InnerClass();

inobj.getValue();

}

}

Ta sử dụng OuterClass.**this**.display();



# Inner Class trong 1 method:

## Code:

**public** **class** OuterClass {

**private** **int** x = 200;

**public** **void** Method(){

**int** rel = 500;

//only final & abstract

**class** InnerClass{

**private** **int** ans;

**public** **void** getValue(){

System.***out***.println("Outer Class Value: " + x);

}

}

**new** InnerClass().getValue();

InnerClass innerClass = **new** InnerClass();

innerClass.getValue();

}

**public** **static** **void** main(String[] args) {

OuterClass obj = **new** OuterClass();

obj.Method();

}

}

## Explanation:

Tương tự ta có 2 cách để khai báo Inner Class trong method:

Cách 1: Xài luôn method nằm trong InnerClass mà không declare hay construct

**new** InnerClass().getValue();

Cách 2: Declare, construct xong xài method.

InnerClass innerClass = **new** InnerClass();

innerClass.getValue();

## Result:



# Anonymous Class:

Một Inner Class được khai báo mà không kèm theo tên lớp được gọi là Anonymous Inner Class. Trong trường hợp như thế, chúng ta khai báo và khởi tạo chúng tại cùng một thời điểm. Nói chung, bạn nên sử dụng loại Inner Class này khi bạn muốn ghi đè phương thức của lớp hoặc Interface. Bạn theo dõi cú pháp sau để hiểu chi tiết hơn:

## Code:

**class** OuterClass {

**public** **void** display() {

System.***out***.println("Outer Class");

}

**public** **static** **void** main(String[] args) {

InnerClass myobj = **new** InnerClass();

myobj.obj.display();

}

}

**class** InnerClass {

OuterClass obj = **new** OuterClass() {

**public** **void** display() {

System.***out***.println("Anonymous Class");

}

};

}

## Explanation:

Ta thấy đáng lý ra kết quả là “Anonymous Class” vì ta Declare trong main là InnerClass sau đó nó đã override lại display() của Outer Class.

## Result:



## Nếu sử dụng thêm method của InnerClass:

Ta sử dụng thêm display1()

**class** OuterClass {

**public** **void** display() {

System.***out***.println("Outer Class");

}

**public** **static** **void** main(String[] args) {

InnerClass myobj = **new** InnerClass();

myobj.obj.display();

}

}

**class** InnerClass {

OuterClass obj = **new** OuterClass() {

**public** **void** display() {

System.***out***.println("Anonymous Class");

display1();

}

**public** **void** display1() {

System.***out***.println("Extra Method");

}

};

}



# Static Modifier Inner Class:

## Code:

**class** OuterClass {

**int** rel = 500;

**static** **class** InnerClass{

**public** **void** display(){

System.***out***.println("Inner Class");

}

}

**public** **static** **void** main(String[] args) {

OuterClass.InnerClass obj = **new** OuterClass.InnerClass();

obj.display();

}

}

## Explanation:

Ở đây ta không thể sử dụng cái variable rel cho InnerClass được vì nó không phải static.

## Result:

