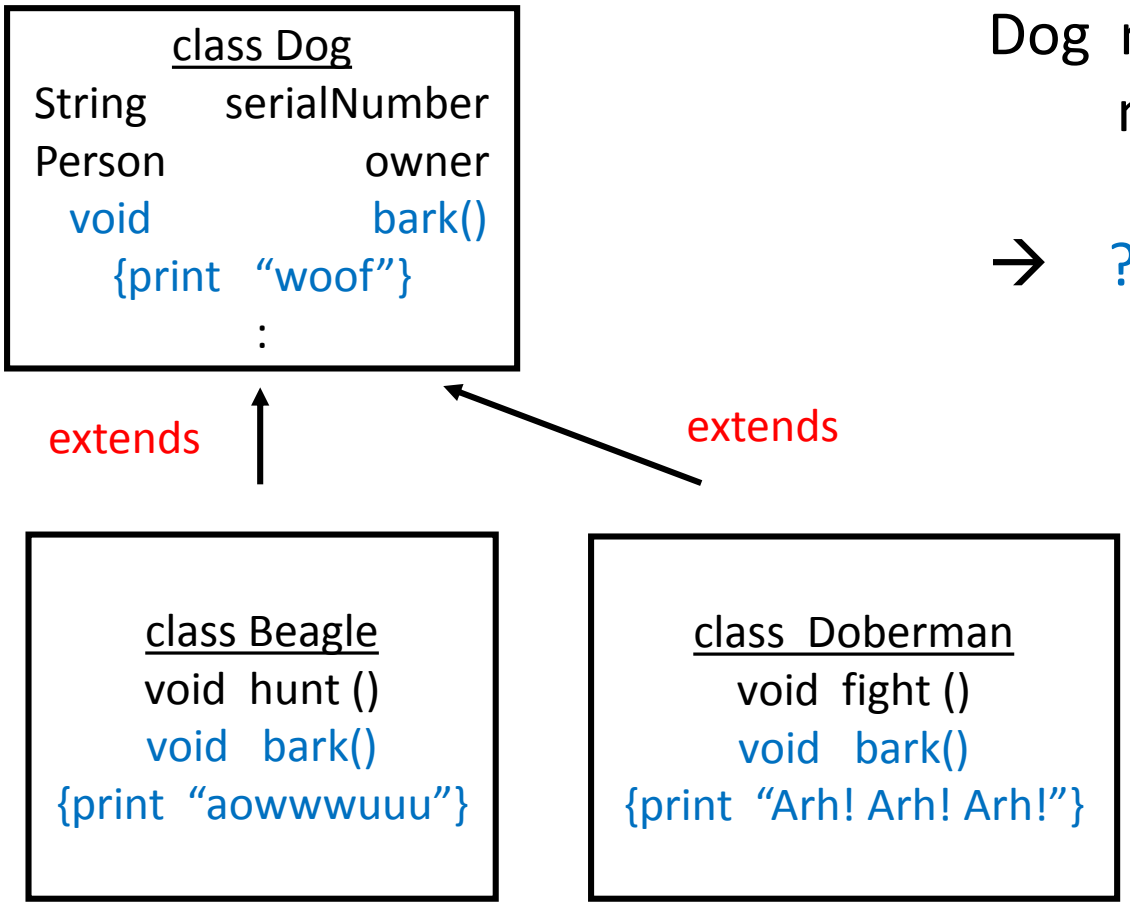
COMP 250

Lecture 32

polymorphism

Nov. 25, 2016

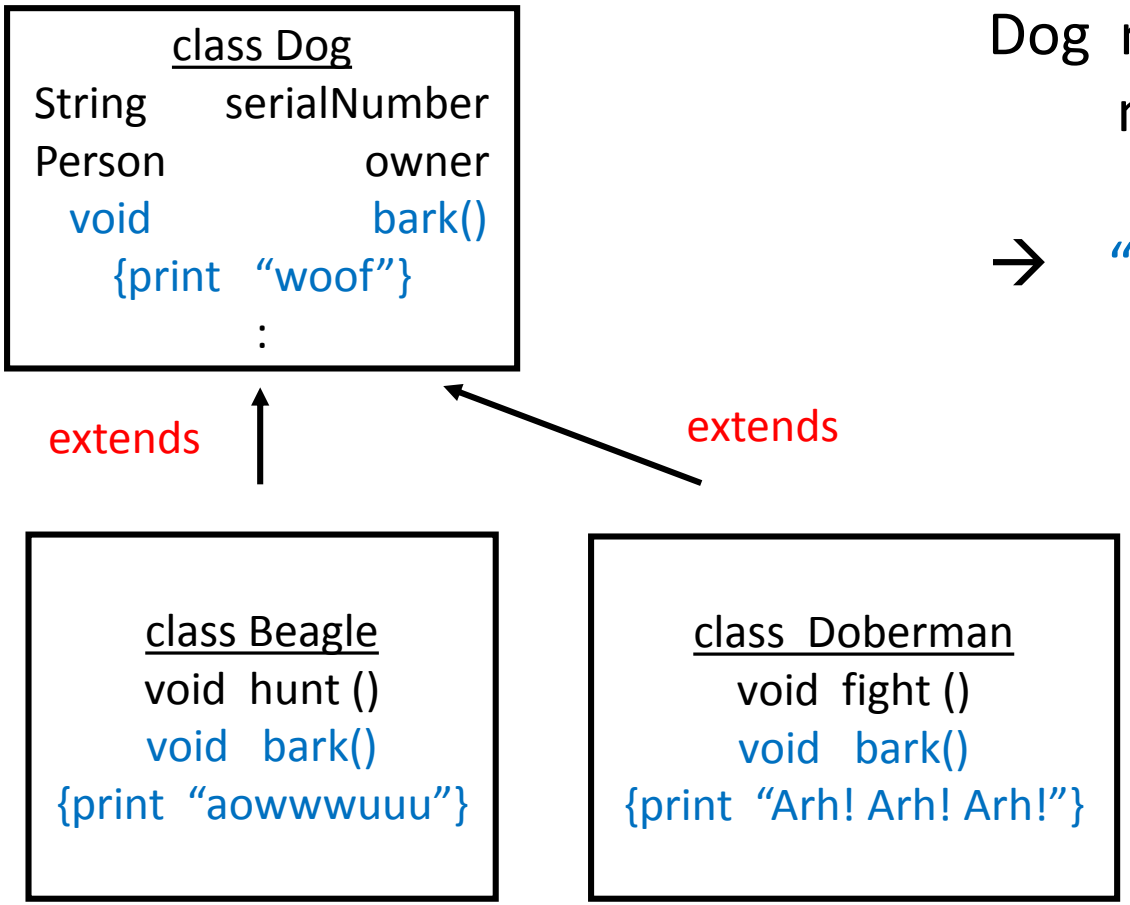
Recall example from lecture 30




```
Dog  myDog = new Beagle();
     myDog.bark();
```

→ ??????

Recall example from lecture 30




```
Dog  myDog = new Beagle();
     myDog.bark();
```

→ "aowwwuuu"

Polymorphism

“poly” = multiple

“morph” = form

We will look at “sub-type” polymorphism.

More general discussion about this in higher level courses

e.g. COMP 302.

Polymorphism

Compile time:

Suppose a reference variable has a declared type: class C.

```
C  var;
```

Runtime:

That reference variable can reference any object of class C or that extends class C.

Polymorphism

Compile time:

Suppose a reference variable has a declared type: abstract class A.

```
A var;
```

Runtime:

That reference variable can reference any object whose class extends abstract class A.

Polymorphism

Compile time:

Suppose a reference variable has a declared type: interface I.

I var;

Runtime:

That reference variable can reference any object whose class implements interface I.

```
boolean  b;
Object    obj;

:

if ( b )
     obj = new float[23];
else
     obj = new Dog();

System.out.print( obj  );     //  calls  toString()
```

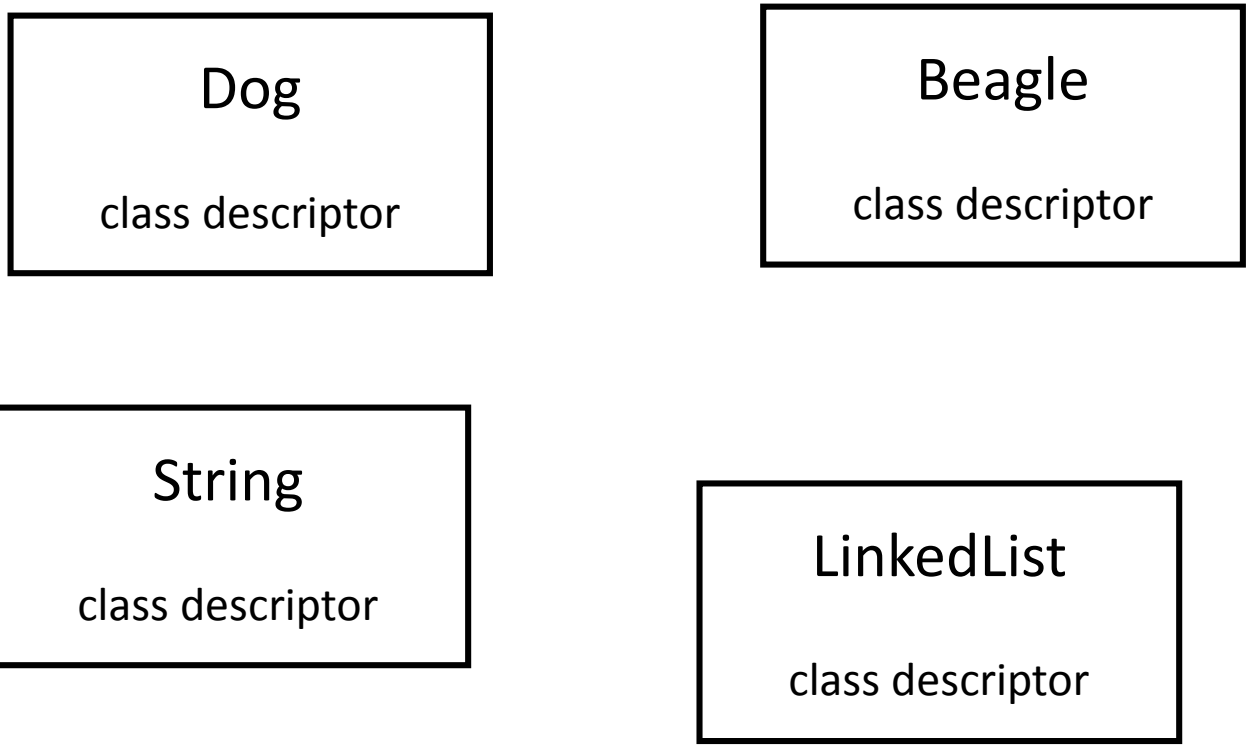
How does polymorphism work?

How does all this class relationship stuff work in a running program ?

(Sketch only.)

Java Class Descriptors

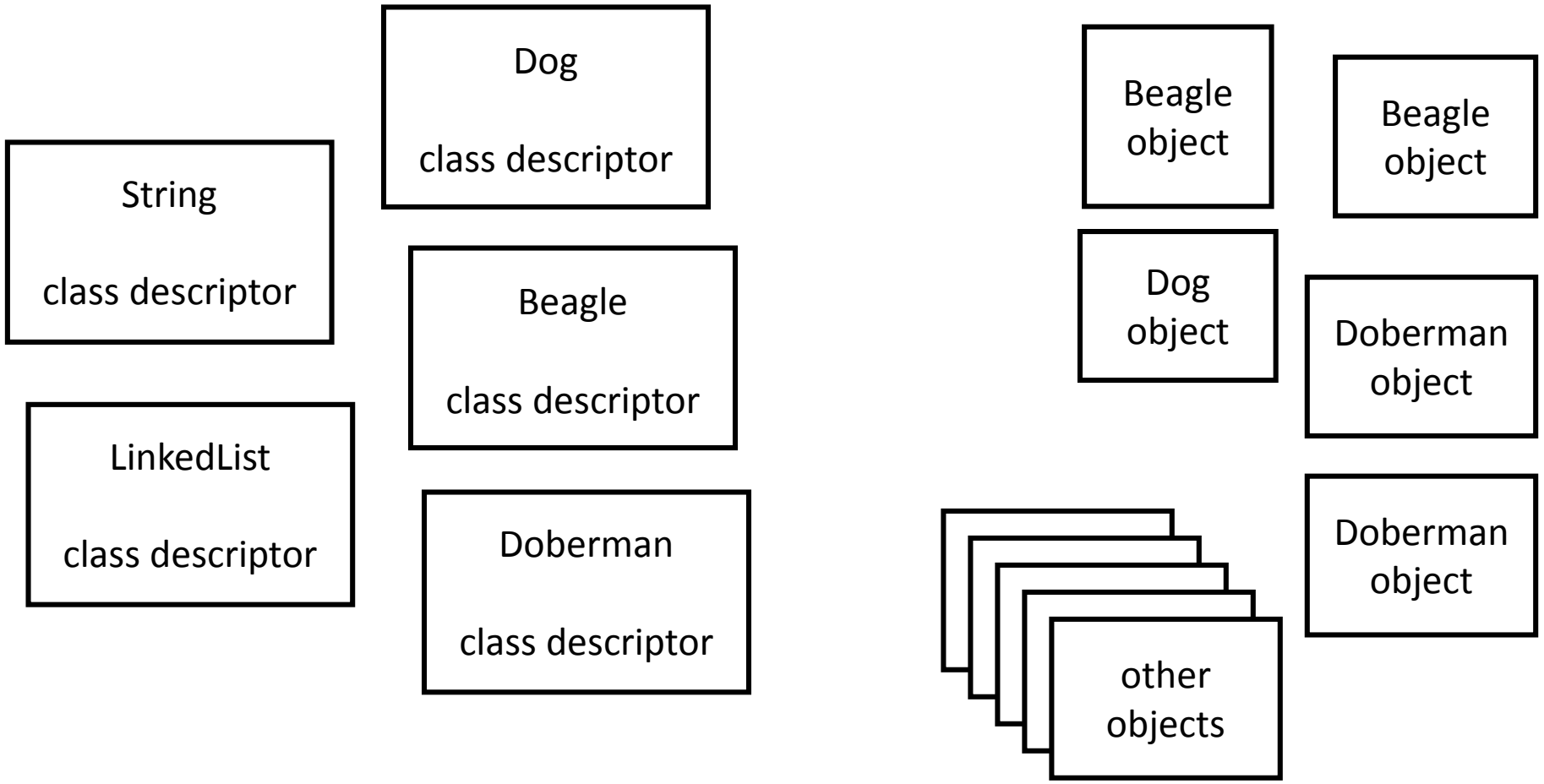
A class descriptor is an *object* that contains all the information about a class that is used in a running program.



Class Descriptors

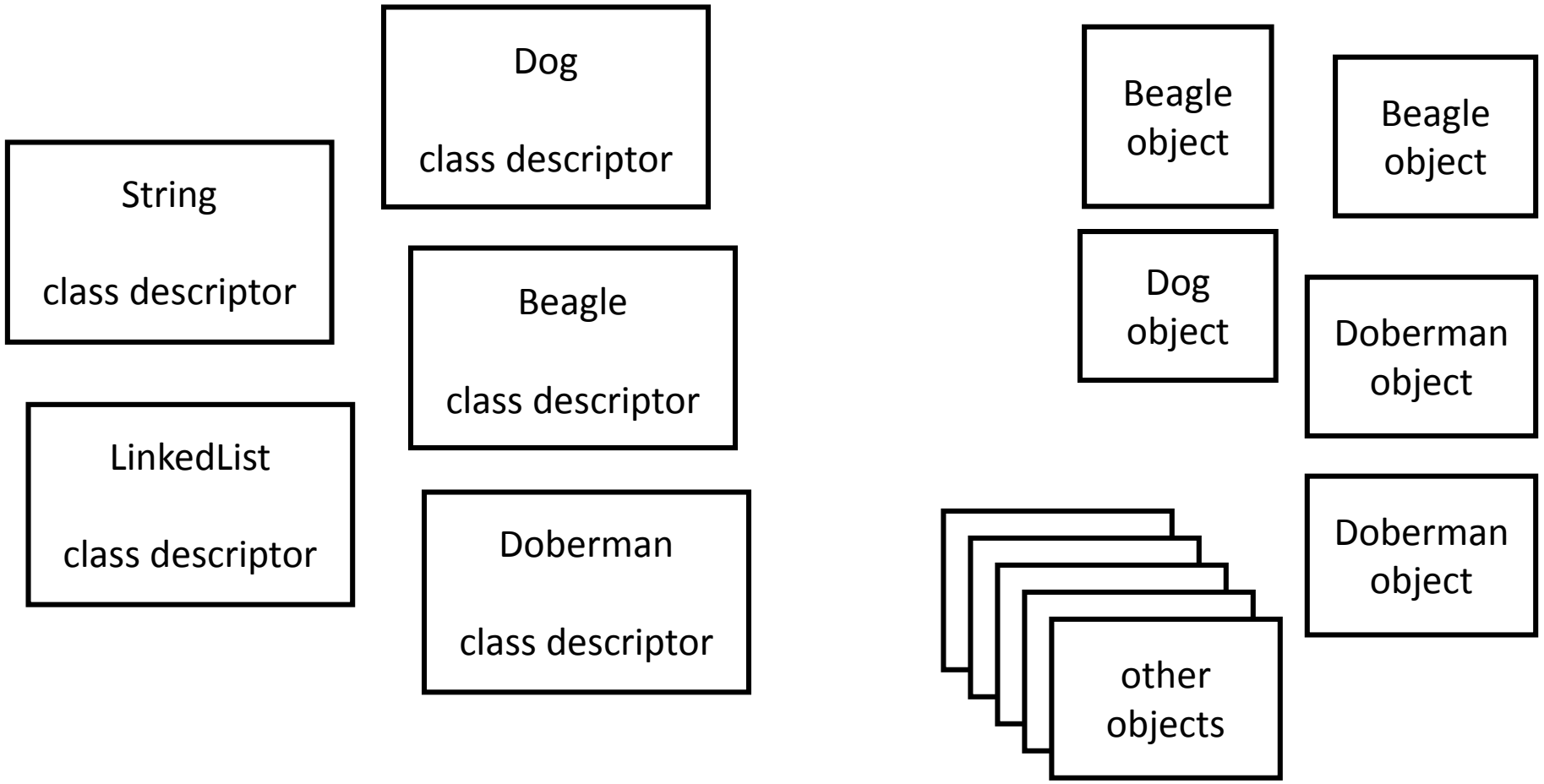
- class name
- fields (names, types)
- methods (names, parameters, implementation)
- reference to superclass
- ….

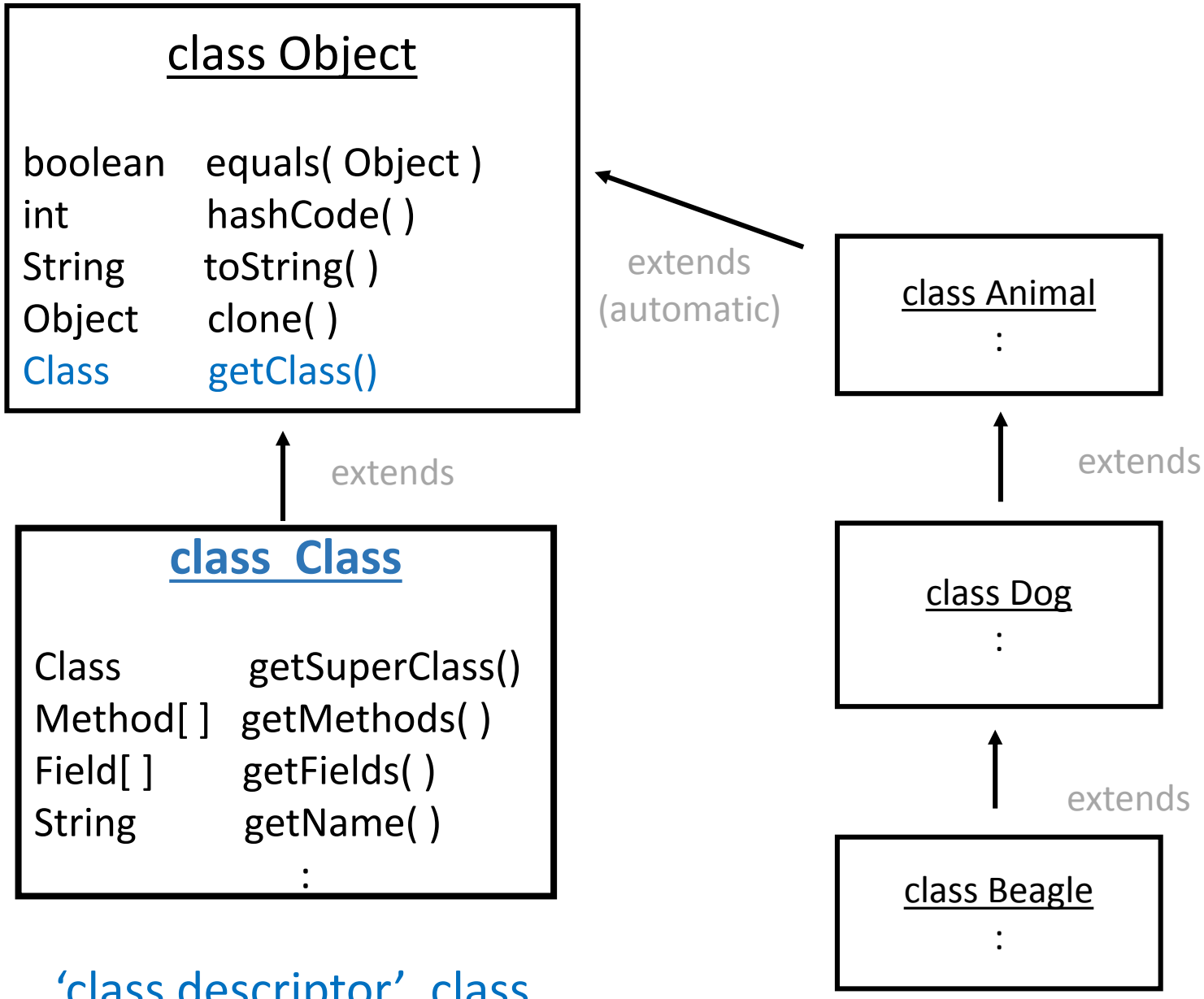
Q: Each object is an instance of a class. So, if class descriptors are objects, then what class(es) are they instances of ?



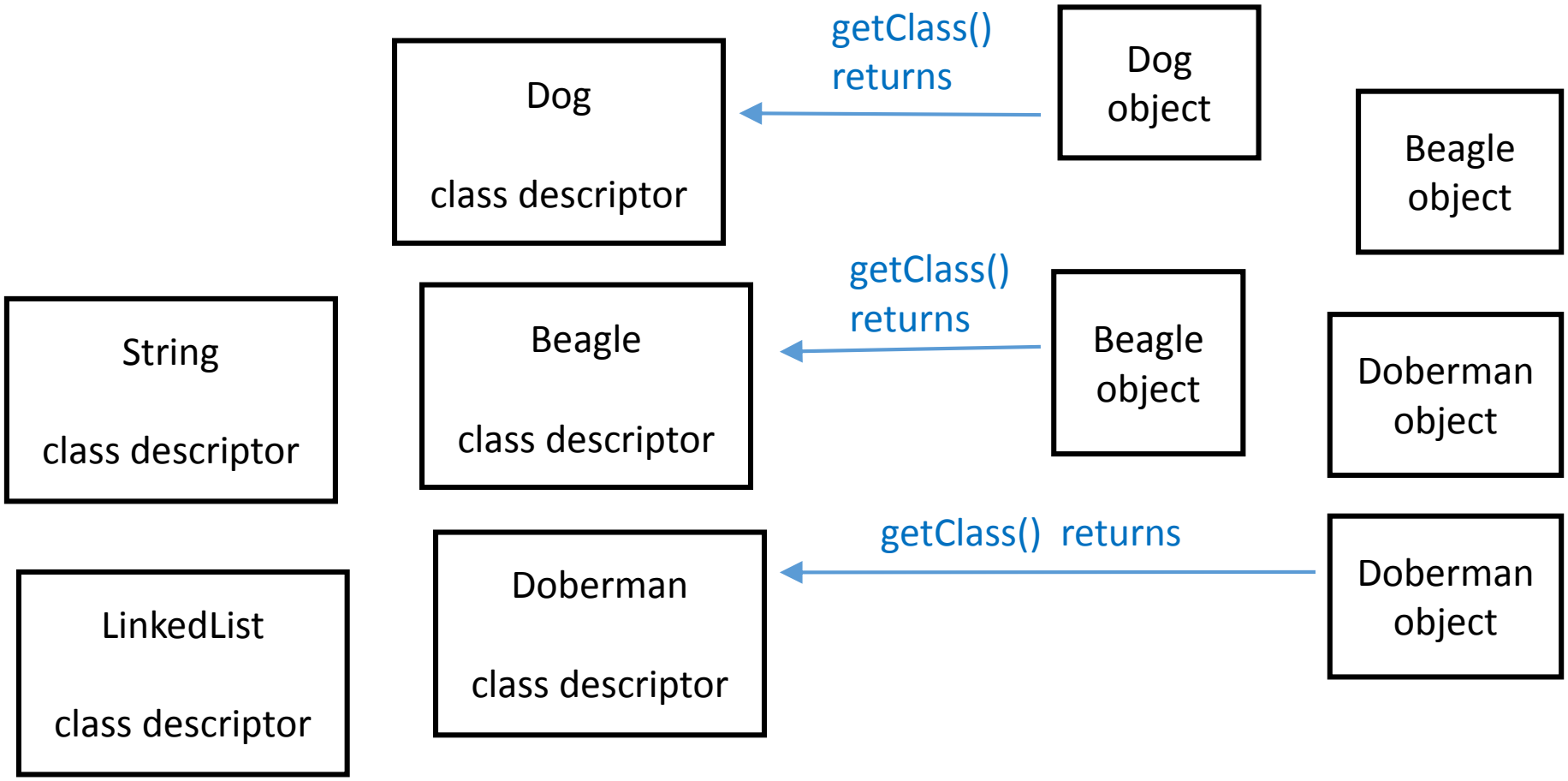
Q: Each object is an instance of a class. So, if class descriptors are objects, then what class(es) are they instances of ?

A: the Class class https://docs.oracle.com/javase/8/docs/api/java/lang/Class.html

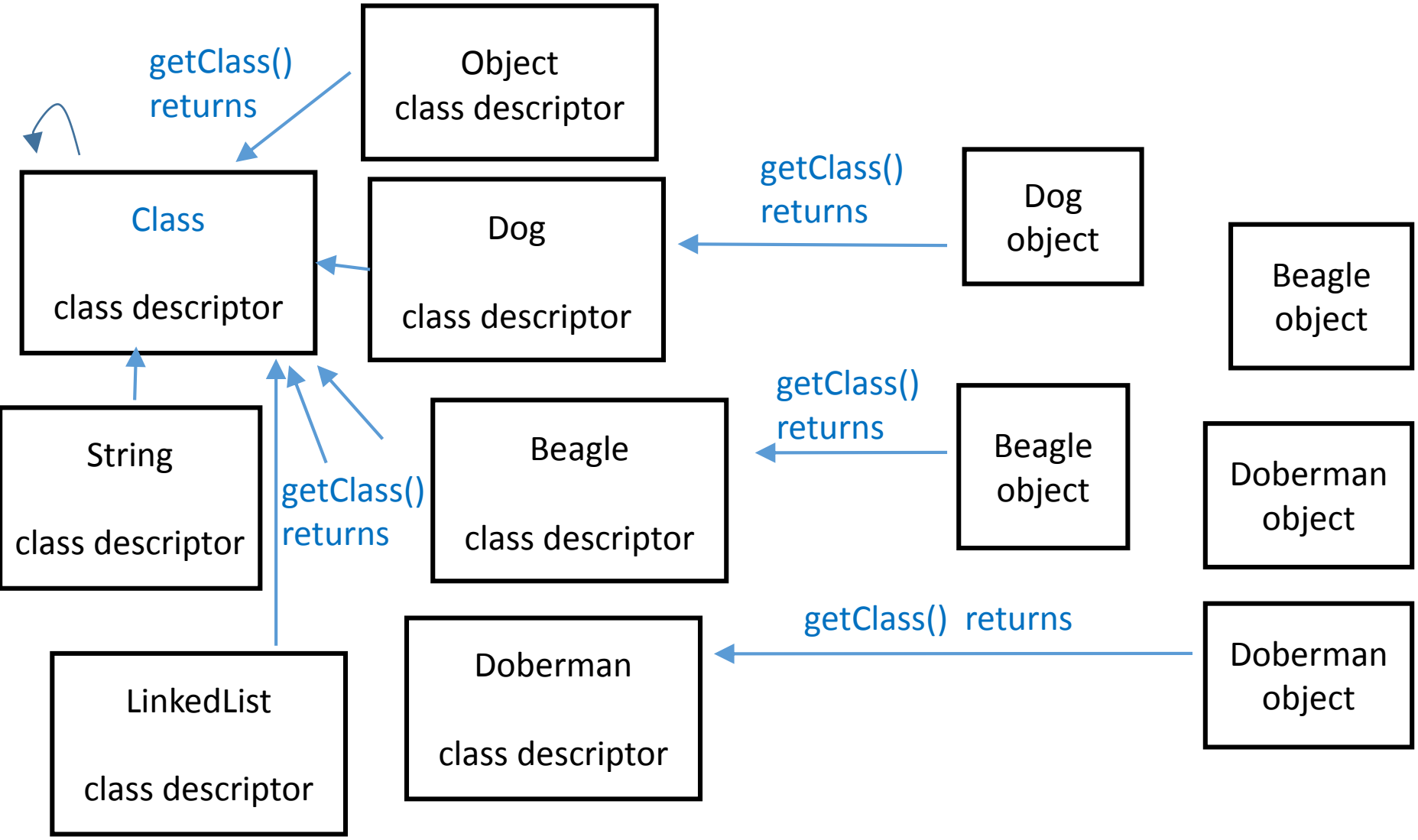


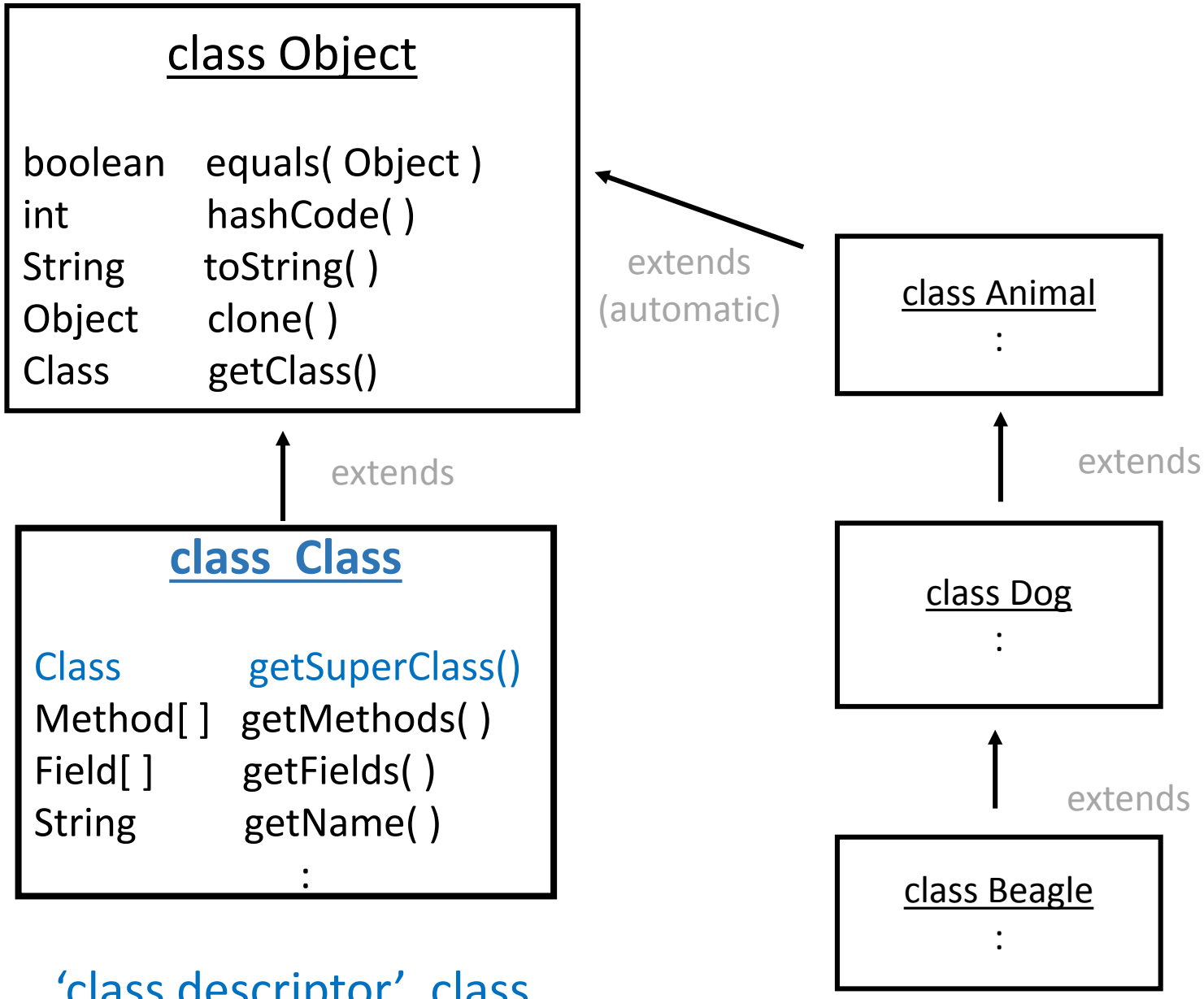

class Object
boolean equals(Object)
int hashCode( )
String toString( )
Object clone( )
Class getClass()
extends
(automatic)
class Animal
:
extends
class Class
Class getSuperClass()
Method[ ] getMethods( )
Field[ ] getFields( )
String getName( )
:
extends
class Dog
:
extends
class Beagle
:
'class descriptor' class

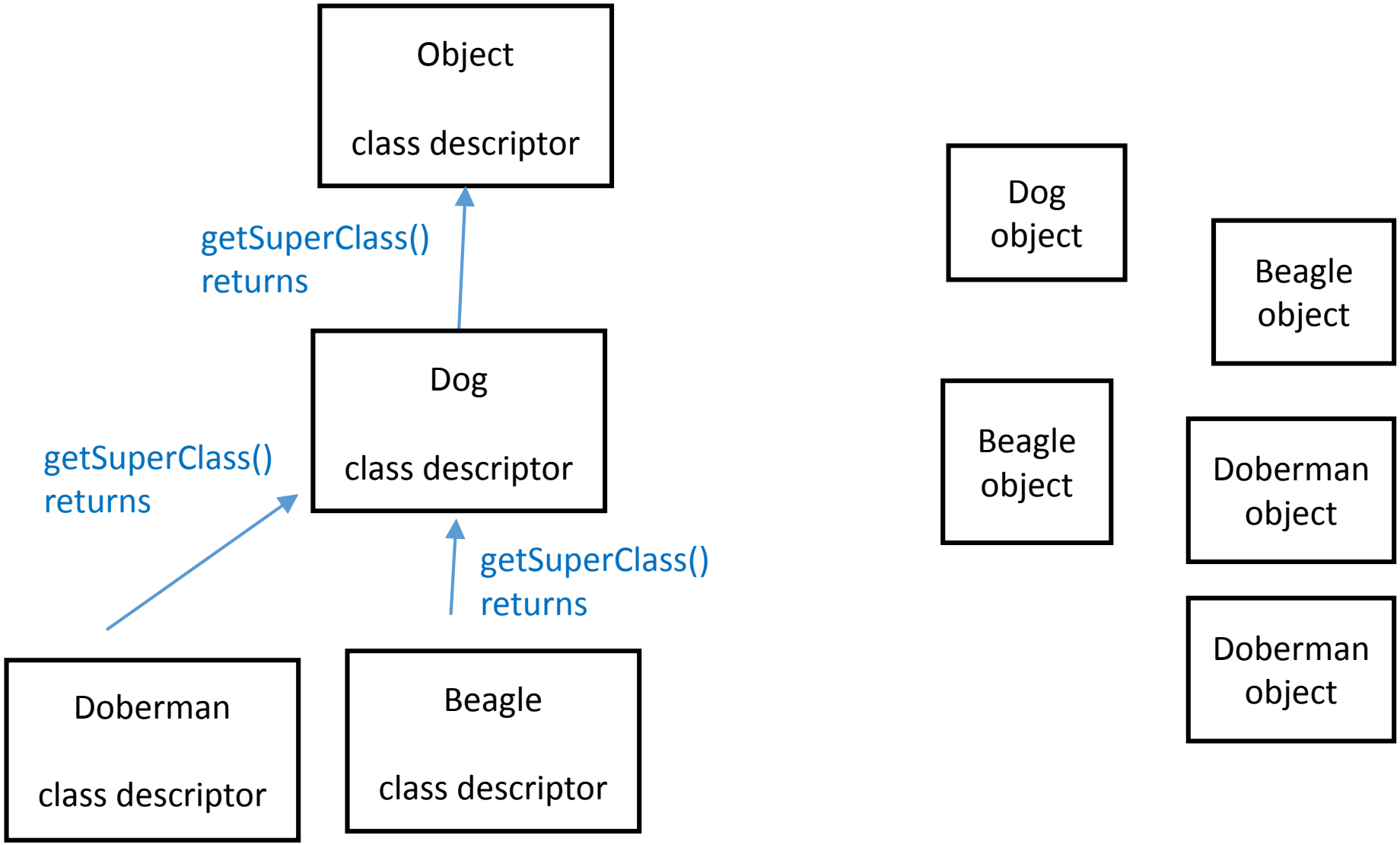
All objects inherit the Object.getClass() method.
This method returns the class descriptor for that object.

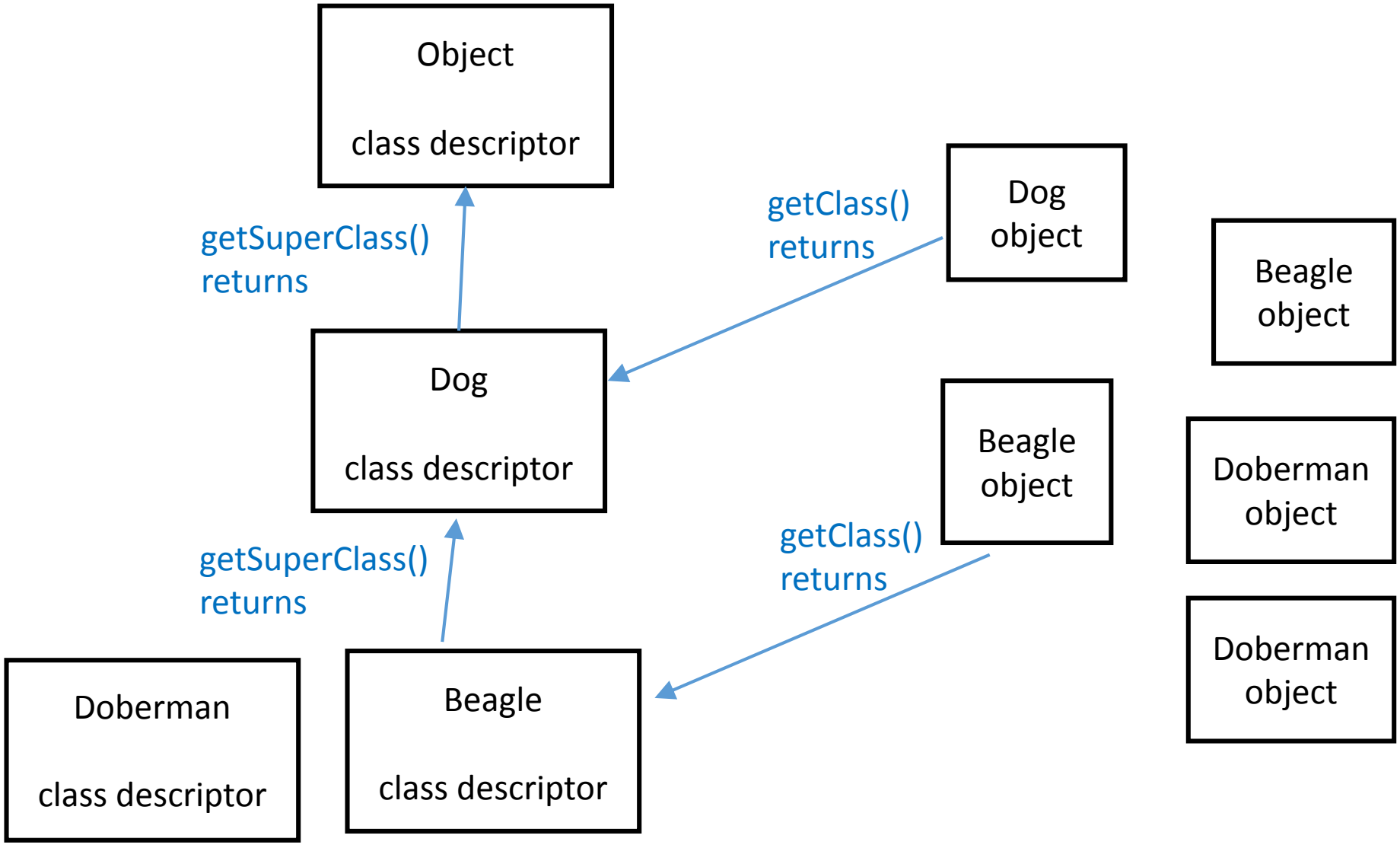


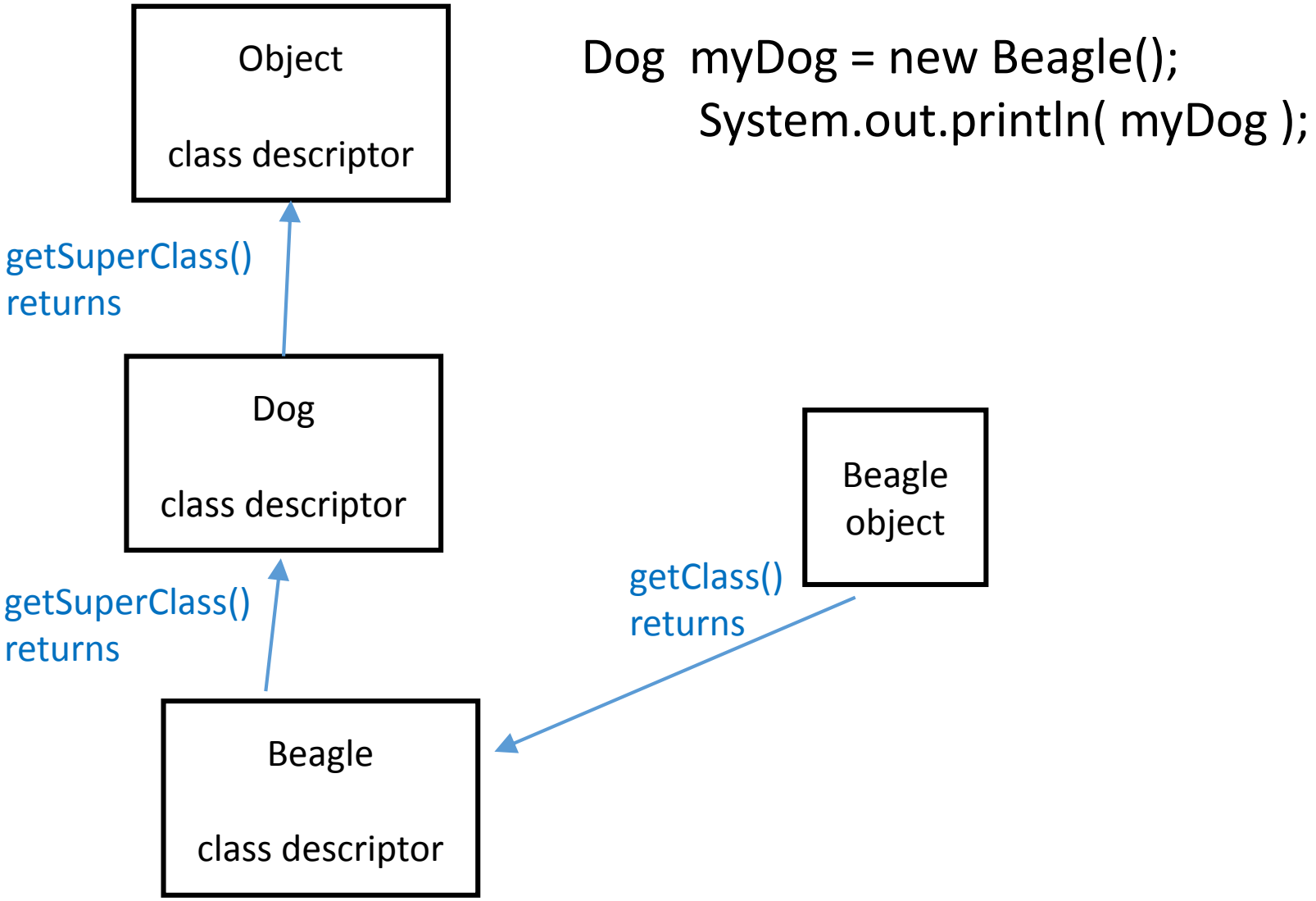
All objects inherit the Object.getClass() method.
This method returns the class descriptor for that object.




class Object
boolean equals(Object)
int hashCode( )
String toString( )
Object clone( )
Class getClass()
extends
(automatic)
class Animal
:
extends
class Class
Class getSuperClass()
Method[ ] getMethods( )
Field[ ] getFields( )
String getName( )
:
extends
class Dog
:
extends
class Beagle
:
'class descriptor' class


Object
class descriptor
getSuperClass()
returns
Dog
class descriptor
getSuperClass()
returns
getSuperClass()
returns
Doberman
class descriptor
Beagle
class descriptor
Dog
object
Beagle
object
Beagle
object
Doberman
object
Doberman
object


Object
class descriptor
getSuperClass()
returns
Dog
class descriptor
getSuperClass()
returns
Doberman
class descriptor
Beagle
class descriptor
getClass()
returns
Dog
object
getClass()
returns
Beagle
object
Beagle
object
Doberman
object
Doberman
object


Object
class descriptor
getSuperClass()
returns
Dog
class descriptor
getSuperClass()
returns
Beagle
class descriptor
Beagle
object
getClass()
returns
Dog myDog = new Beagle();
System.out.println(myDog);

Object

class descriptor

Dog

class descriptor

Beagle

class descriptor

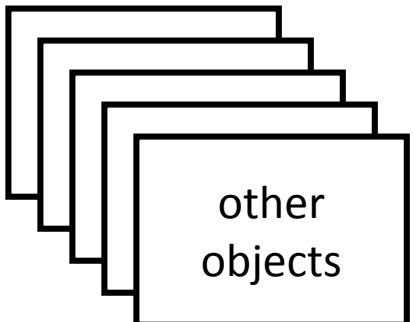
Test

class descriptor

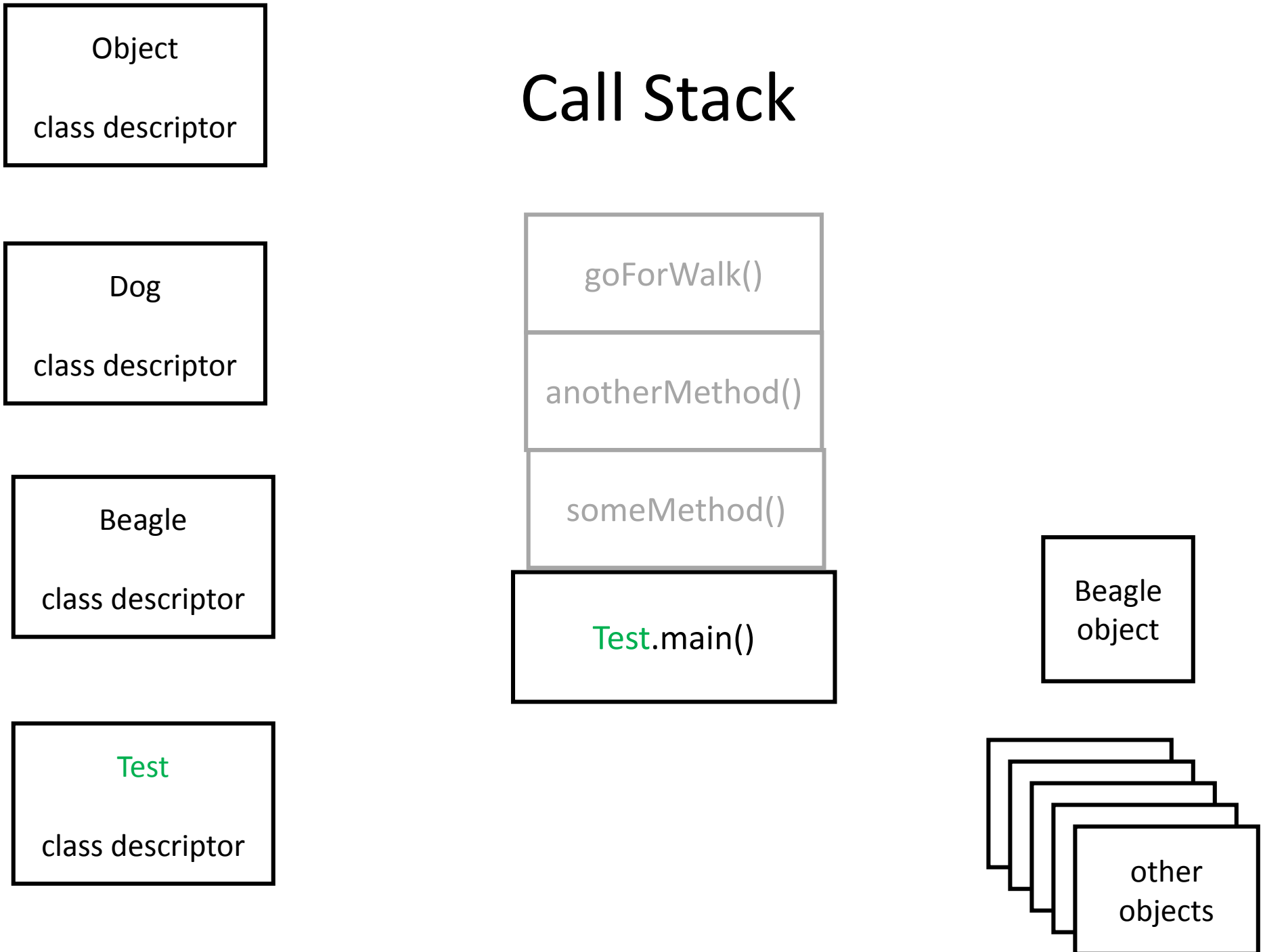
What else do we need to run a program?

e.g. Consider running a Test class which has a main() method.

Beagle object



Call Stack


Object
class descriptor
Dog
class descriptor
Beagle
class descriptor
Test
class descriptor
goForWalk()
anotherMethod()
someMethod()
Test.main()
Beagle
object
other
objects

Object

class descriptor

Dog

class descriptor

Beagle

class descriptor

Test

class descriptor

Call Stack

```
Dog    myDog = new Beagle();
//  in main() method
```

Beagle()

Test.main()

Dog myDog

other objects

Object

class descriptor

Dog

class descriptor

Beagle

class descriptor

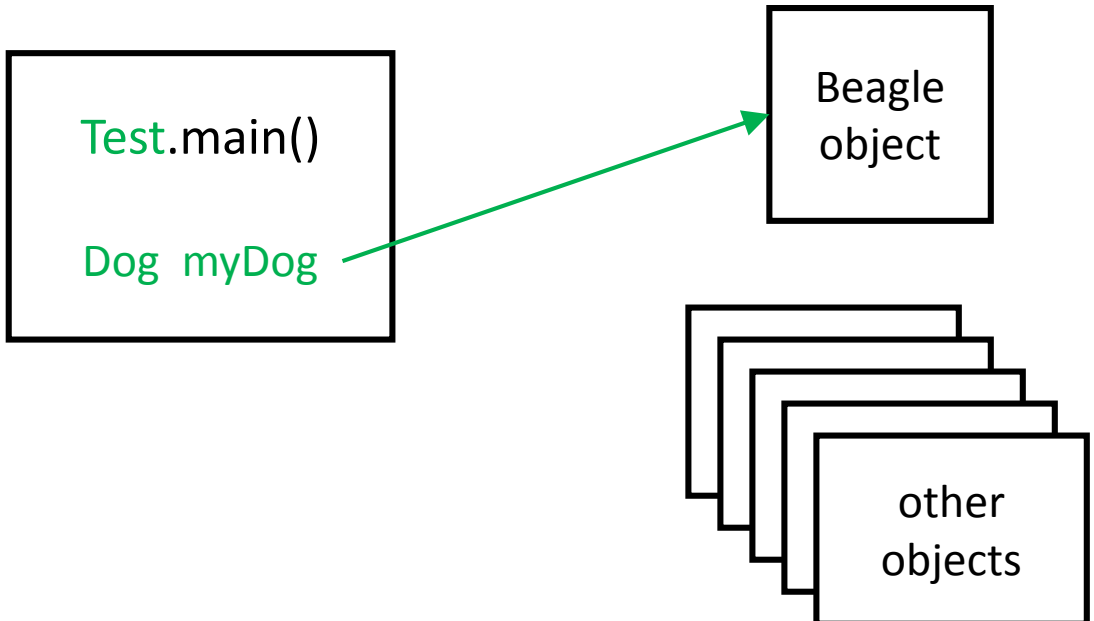
Test

class descriptor

Call Stack

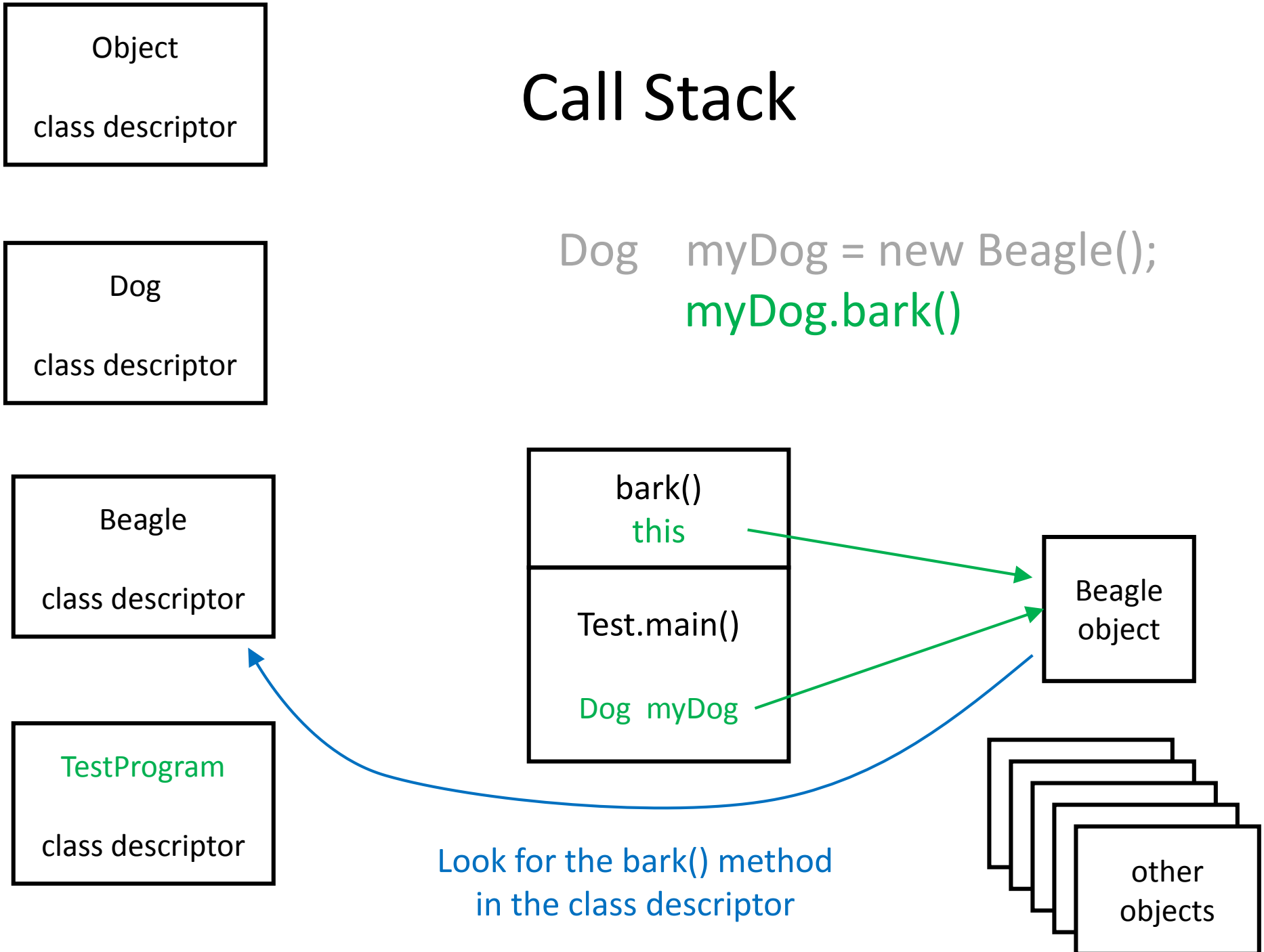
```
Dog    myDog = new Beagle();

//  after instruction is done….
```



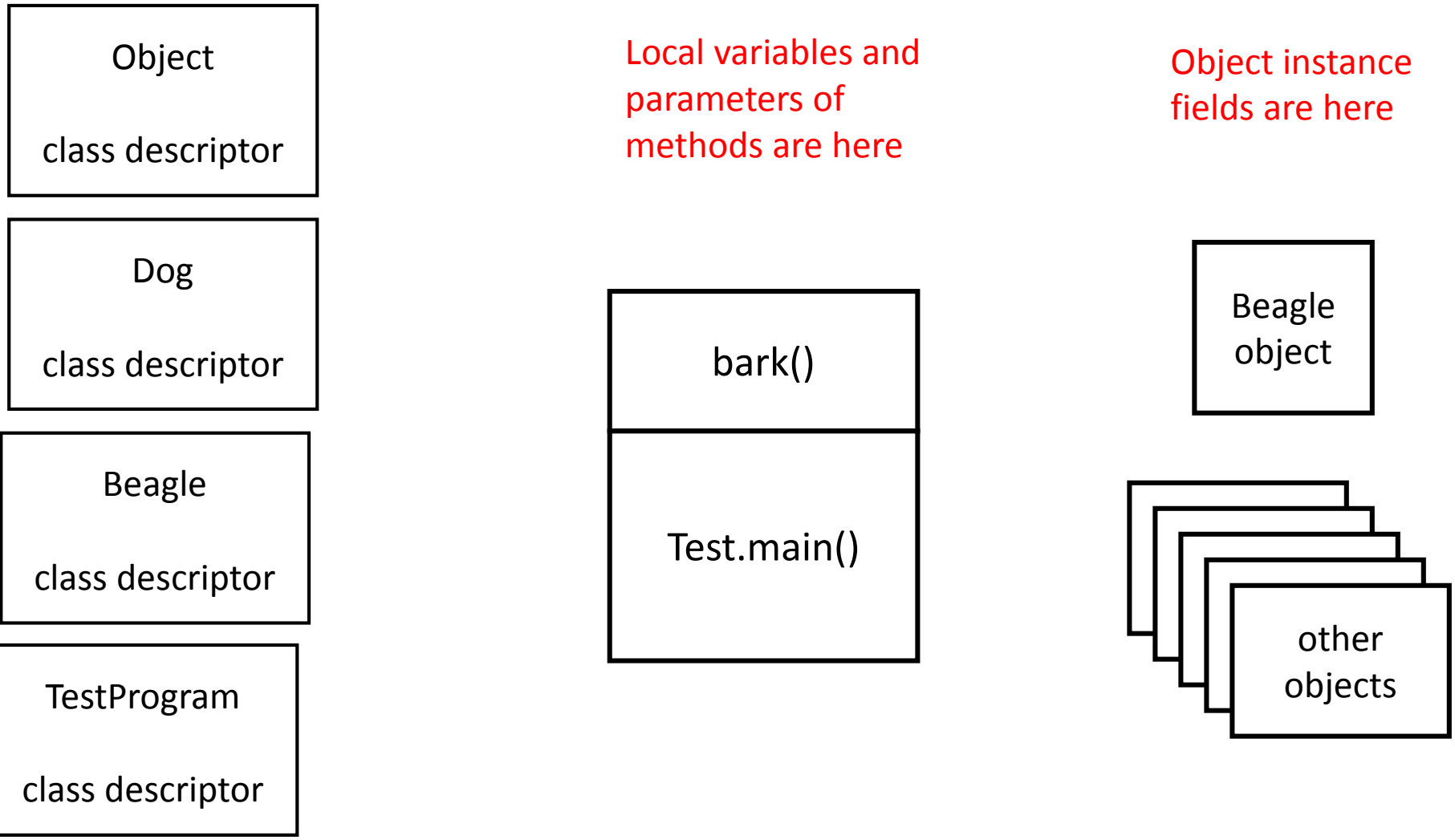
Call Stack

Dog myDog = new Beagle();
myDog.bark()



Class Descriptors

Methods are here



Call Stack

Local variables and parameters of methods are here

Objects

Object instance fields are here