JAVA LAB

2nd chapter

School of Computer and Information Sciences
University of Hyderabad

January 30, 2019

What is class?

Class is the basic building block of an "object oriented language". It is template that describes the data and behaviour associated with object(instance of class). In General, it is design plan of an object. It is a collection of data members, methods, subclasses.

syntax:

class <<classname >>{

...data members

...methods

...subclasses

}

Examples

Class Customer
Attributes:
name
address
budget
Methods:
purchase() {send a purchase request to a salesperson}
getBudget() {return budget}

Sean as an Object
Attributes:
name = Sean;
Methods:
takeOrder() {
check with warehouse on stock availability
check with warehouse on delivery schedule
if ok
then instruct warehouse to deliver stock(address, date)
return ok
else return not ok
}

What is method?

Method is same functions as in c. But in java, we have special functions, without returning anything, known as constructors. For constructors, name is same as class name. In general, object properties are declared in constructor.

syntax:

class <<classname >>{
<<classname >>() { ...
...
}
int method1(){
...
...
}
}

Examples

```
class Counter {
int number;
int reused = 0;
void add() {
number = number+1;
}
void initialize() { number = 0;
reused = reused+1;
}
}
```

What is object?

Object is a run time entity.

Object creation:

In java, object is created by **new** keyboard.

Syntax:

<<classname >><<class variable >>= new <<classname >>()

Here is the example covering all the above concepts classobject

Accessing members in class:

classvariable.method()

classvariable.datamember

Examples

```
Counter carpark; /* counter objects*/
... carpark = new Counter();
Counter entrance, exitDoor; /* newly created counter objects*/
...
entrance = new Counter(); /* Newly created instance of the Counter class
*/
exitDoor = new Counter();/* Newly created instance of the Counter class
*/
Method: initialize()
entrance.initialize();
exitDoor.initialize();
```

copy constructor

Java allows copy constructor.
Here is the sample program copy constructor

access modifiers

Access modifiers help to restrict scope of the class, method or datamember. In general, there are four access modifiers.

1. public
2. private
3. protected
4. default

public: Accessible to all.
private: Accessible to same class, which it is declared.
protected: Accessible within all classes in the same package and within subclasses in other packages.
Default: Accessible within classes in the same package.

Example

Two accessor methods getNumber() and getReused() are introduced in the above code.

```
class Counter {
private int number = 0;
private int reused = 0;
public void add() {
number = number+1;
}
public void initialize() {
number = 0;
reused = reused+1;
}
/*To access attributes */
public int getNumber() { return number; }
public int getReused() { return reused; }
}
```

Overloading

Overloading: Method name is same but differ in type of parameters or number of parameters of function. There are two types of overloading:

1. Method overloading.
2. Constructor overloading.

Here is the example of method overloading: Method overloading
Here is the example of constructor overloading: Constructor loading

Example

When a Counter to be incremented other than by 1, we could define another add() method that takes an integer parameter.

```
class Counter {
private int number = 0;
private int reused = 0;
public void add() {
number = number+1;
}
public void add(int x) {
number = number+x;
}
public void initialize() {
number = 0;
reused = reused+1;
}
public int getNumber() { return number; }
public int getReused() { return reused; }
}
```

Example of constructor method

```
class Counter {
private int number, reused;
public void add() {
number = number+1;
}
public void initialize() { number = 0;
reused = reused+1;
}
public int getNumber() { return number; }
public int getReused() { return reused; }
Counter() { number = 0; reused = 0; }
}
```

Example of constructor overloading

```
class Counter {
private int number, reused;
public void add() {
number = number+1;
}
public void initialize() { number = 0;
reused = reused+1;
}
public int getNumber() { return number; }
public int getReused() { return reused; }
Counter() { number = 0; reused = 0; }
Counter(int x) { number = x; reused = 0; }
Counter(int x, int y) { number = x; reused = y; }
Counter(float z) { number = (int) z; reused = 0; }
}
```

Type conversion and type casting

widening: Automatic type conversion.
example:
int i=10;
long l;
l=i;(no error automatic type conversion).
narrowing: Manual type casting is required.
example:
double i=10.02;
int j;
j=i;(error)
j=int(i);(manual type cast)

Questions?

Thank you