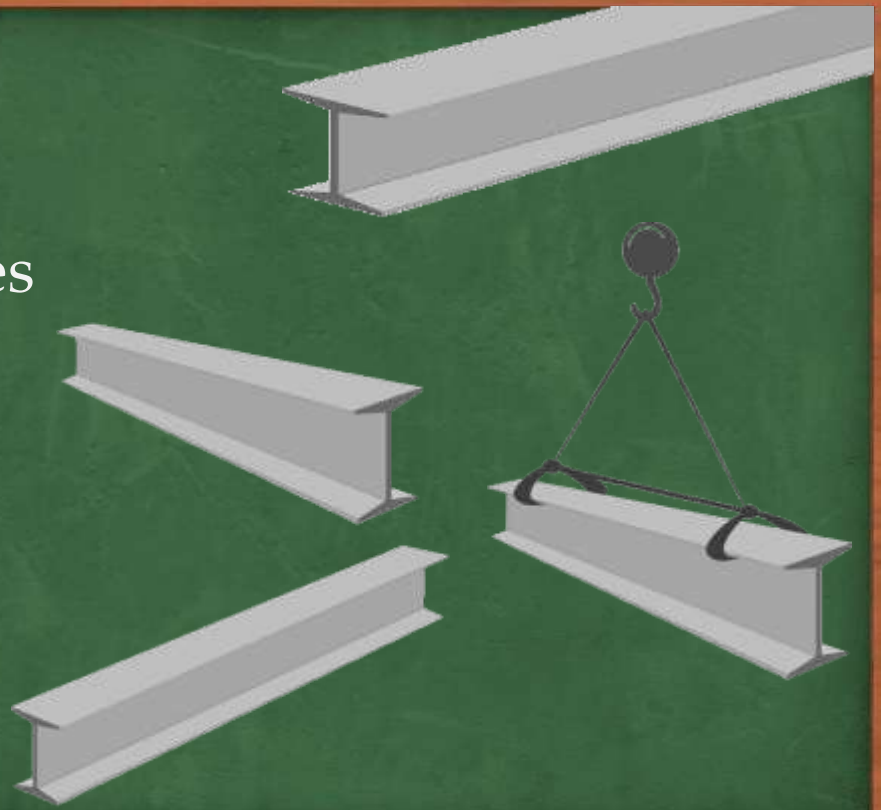
Structure

- Group Primitives into Classes
- “new” and the Heap
- Pointers
- Garbage Collection
- Thinking in Pointers



See Also

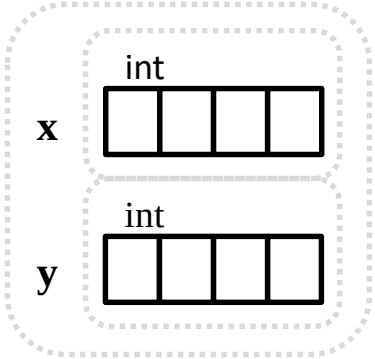
http://docs.oracle.com/javase/7/docs/api/java/lang/Class.html

http://www.tutorialspoint.com/java/java_object_classes.htm

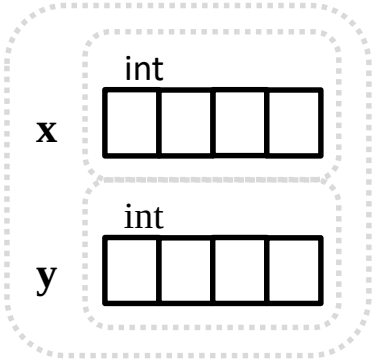
http://programmers.stackexchange.com/questions/207196/do-pointers-really-exist-in-java

Grouping Primitives

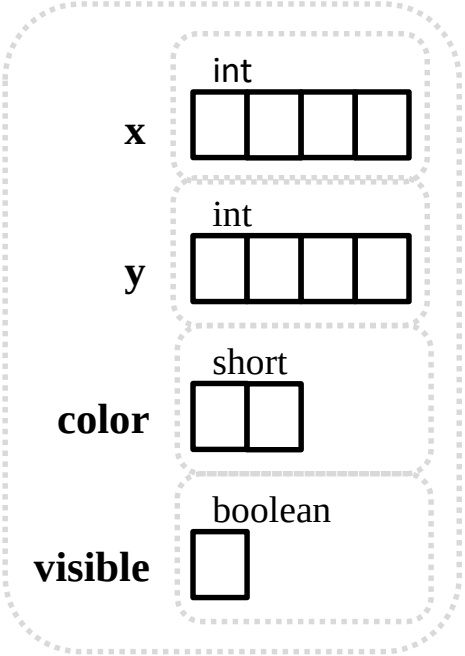
- Primitives can be pieced together to form more complex data structures
- You can access these pieces by name



Point a



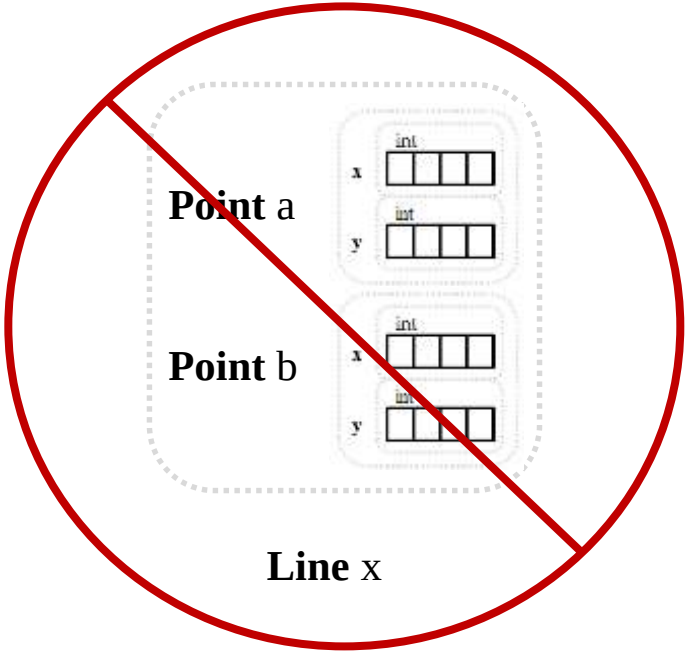
Point b



Pixel p

Groups as Primitives

- Languages like C and C++ allow you to use structures as primitives to make other structures
- NOT JAVA
- You CAN compose with pointers (coming up)



Classes

- In C you group primitives into “structures” (struct)
- In Java the keyword is “class”
- No “static” on these variables

```
class Point {
    int x;
    int y;
}
```

```
class Pixel {
    int x;
    int y;
    short color;
    boolean visible;
}
```

- “int” is a type of data that is 4 bytes
- A class defines a new memory footprint
  - A new “type” of data
  - We say a new “class” of data

Using "new"

- You use the class to "stamp out" new instances of the data structure in memory
- We call these new instances "objects". We say they are "instances" of class "Point".

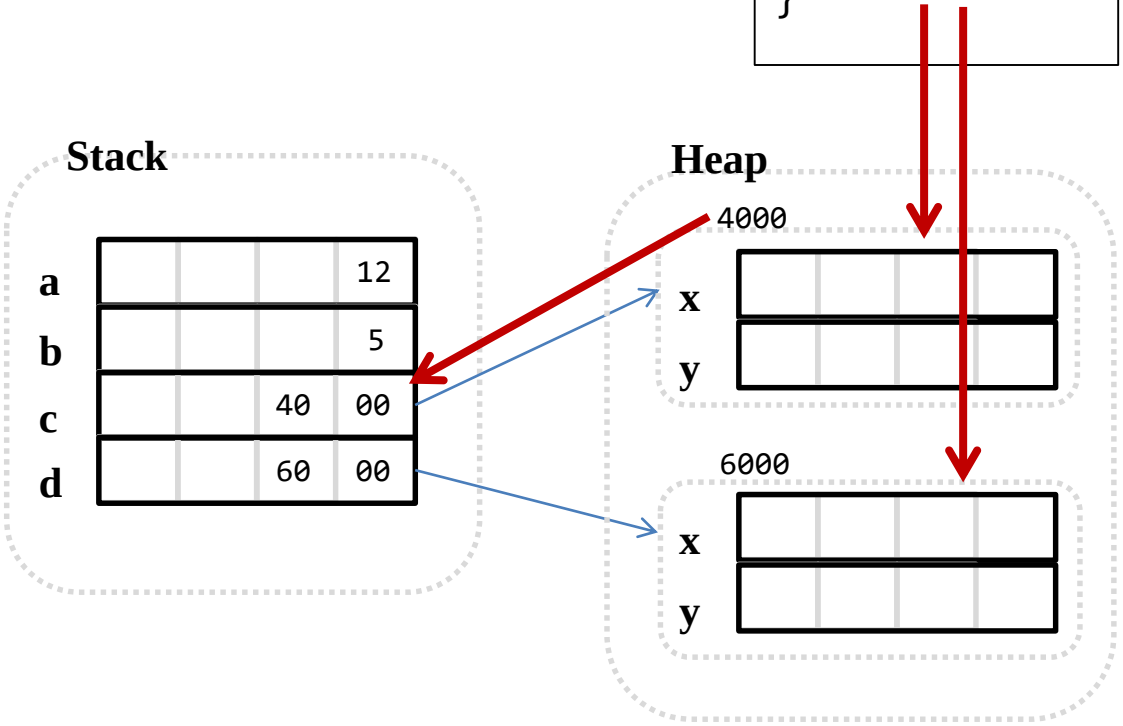
```
public static void main(String [] args) {

    int a = 12;

    int b = 5;

    Point c = new Point();

    Point d = new Point();

}
```

```
class Point {
    int x;
    int y;
}
```

Stack and Heap

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

    int a = 12;

    int b = 5;

    Point c = new Point();

    Point d = new Point();

}
```

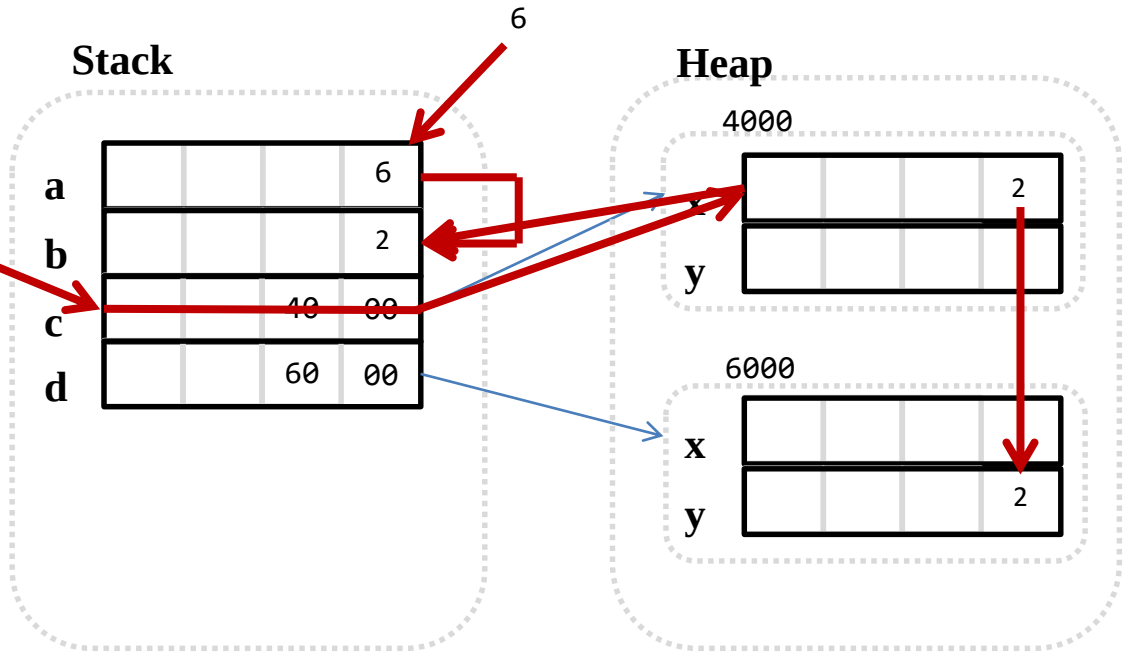
```
class Point {
    int x;
    int y;
}
```



Dot operator – "follow pointer"

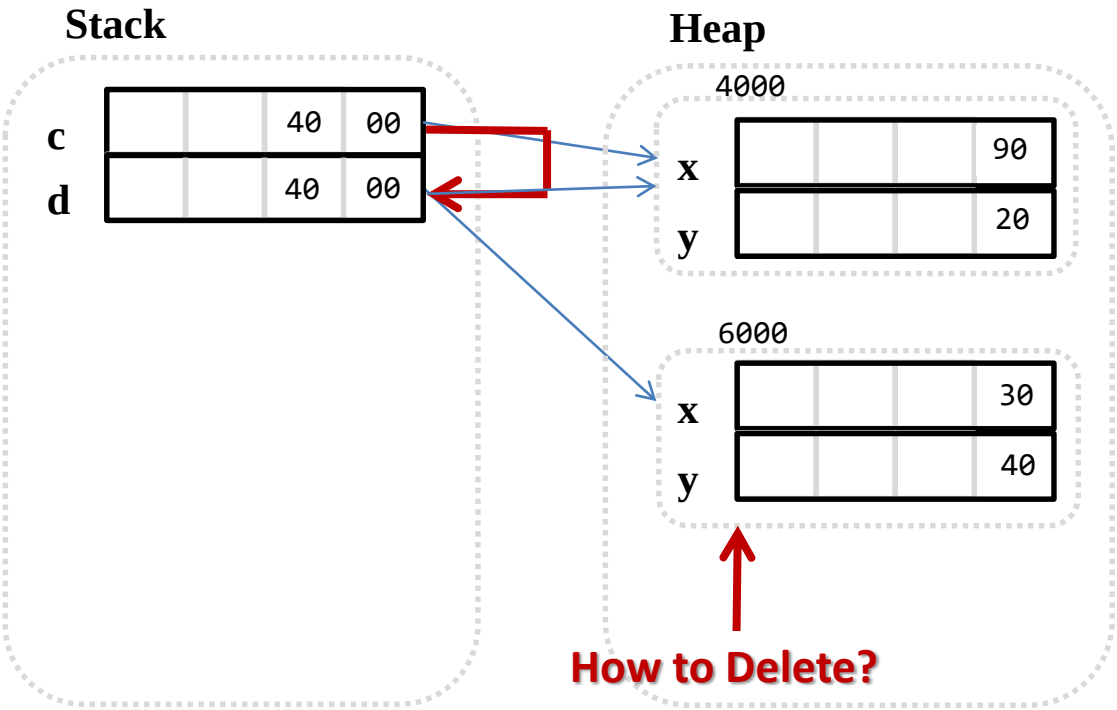
```
b = a;
a = 6;

c.x = 2;
b   = c.x;
d.y = c.x;
```

- The "." means "follow the pointer"
- C++ has a more descriptive "->" as in c->x = 2
- Exercise in indirection



Changing Pointers

```
Point c = new Point();
c.x = 10;
c.y = 20;

Point d = new Point();
d.x = 30;
d.y = 40;

d = c; // both point to same

d.x = 90;

System.out.println(c.x);

// “90" !!
```



Garbage Collection

- There is no “delete” keyword
- The VM keeps up with who points to what
- When nobody has a pointer to an object then that object becomes a CANDIDATE for collection
- Garbage collection is a background process that can stall your program
- You cannot force garbage collection
- Garbage collection may never happen

Tinkering

- Create the “Point.java” and “Line.java” from this lesson
- Create “Tinker.java” with a “main”
- Create some lines and points and wire them up
- Try setting a breakpoint and use Eclipse’s watch window to chase the pointers