Announcements

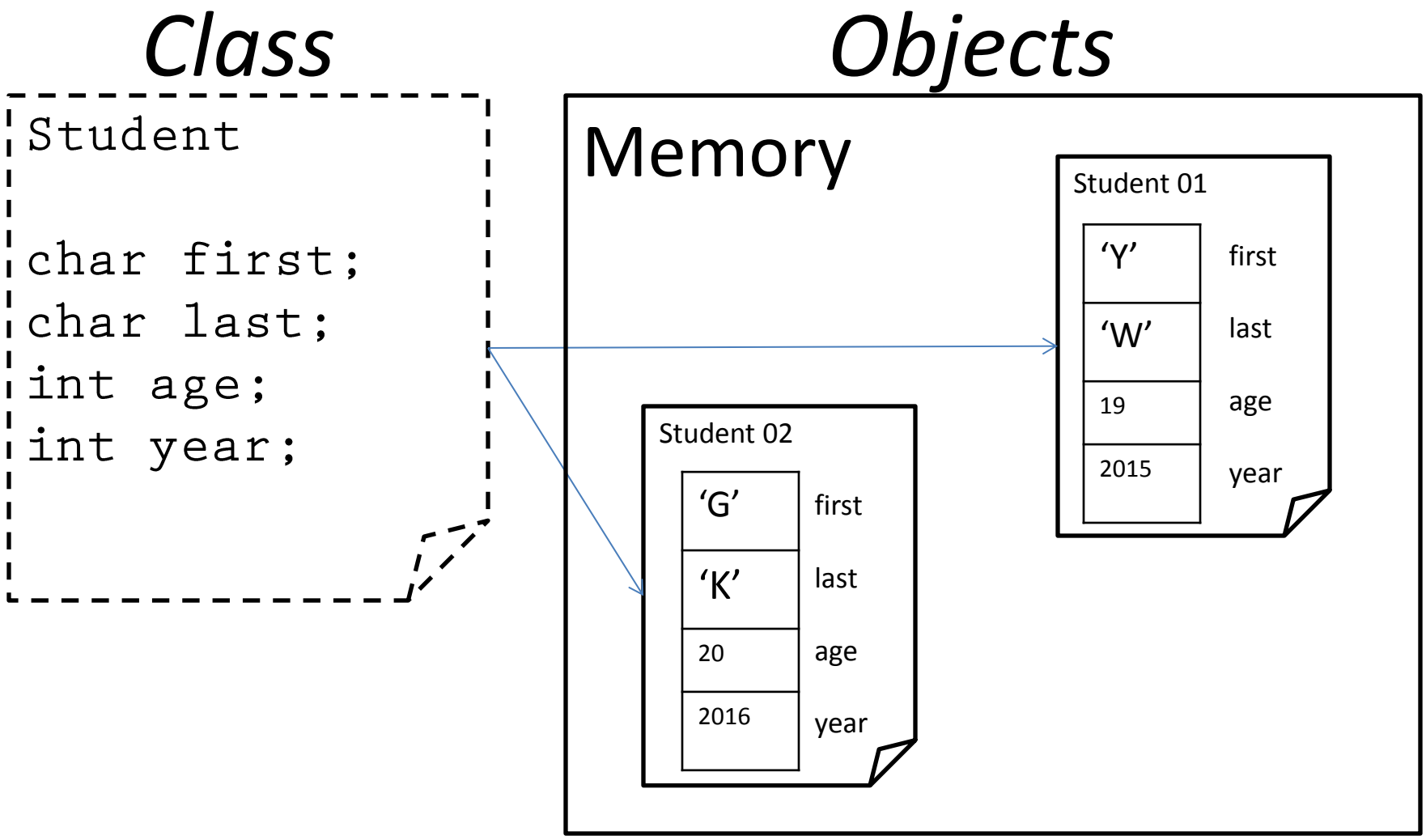
- Project #1 due tomorrow evening - Java version issues
- Midterm #1 on Friday in class
- Project #2 posted by tomorrow evening, due next Tuesday the 18th
- New links on the class resource page
  - Hackers delight
  - Computer's don't argue
- Font size?

Basics of Object Oriented programming

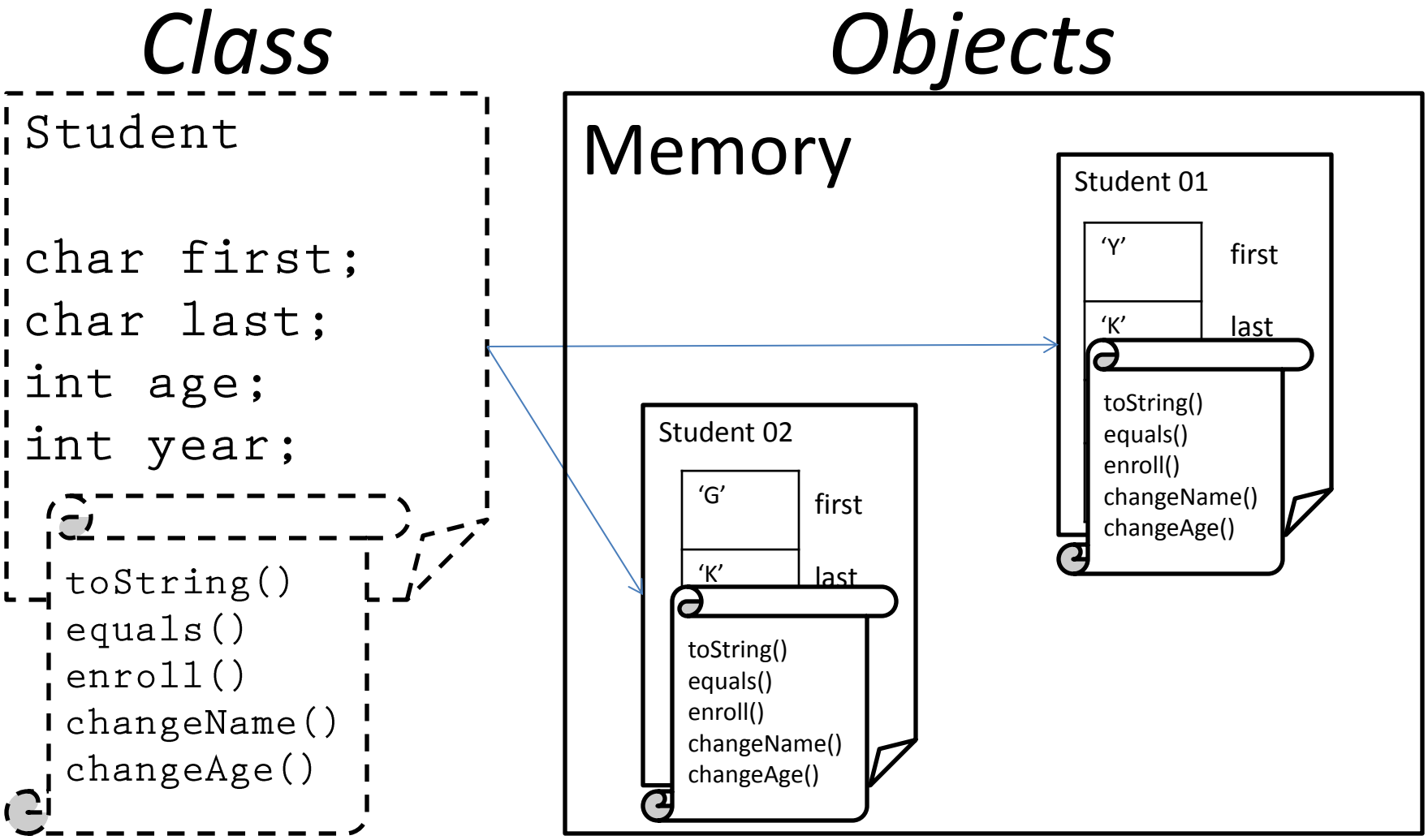
Hard coding data structure

```
//Data for student 01
String student_01_Name;
int student_01_Age;
int student_01_Year;
String student_01_Major;

//Data for student 02
String student_02_Name;
…

//Data for student 03
String student_03_Name;
…
```

Structured data



Associated methods



Classes vs Objects

- Class
  - A “Template” or “Blue-print” for individual objects
    - Like the floor-plan of a house
  - Defined by you, the programmer
  - “Fields”: variables that store data
  - “Methods”: instructions for manipulating data
- Object
  - An “instantiation” of the blue-print
    - Like the physical house constructed according to the floor plan
  - Multiple copies
    - Same fields (although perhaps different data in those fields)
    - Same methods

Basic usage

- Constructing a new instance:

```
Student stu01 = new Student();
```

- Referencing data fields:

```
int x = stu01.age;
stu01.age = 13;
```

- Calling methods:

```
stu01.enroll();
String str = stu01.toString();
boolean same = stu01.equals(stu02);
```

Defining your class

```
//Convention: class names start with Capital
public class Student {

        //Declare fields
        char first, last;
        int age, year;

        //Declare constructor
        public Student() {
                //Initialize fields
        }

        //Declare methods
        public String toString() {
                …
        }
}
```

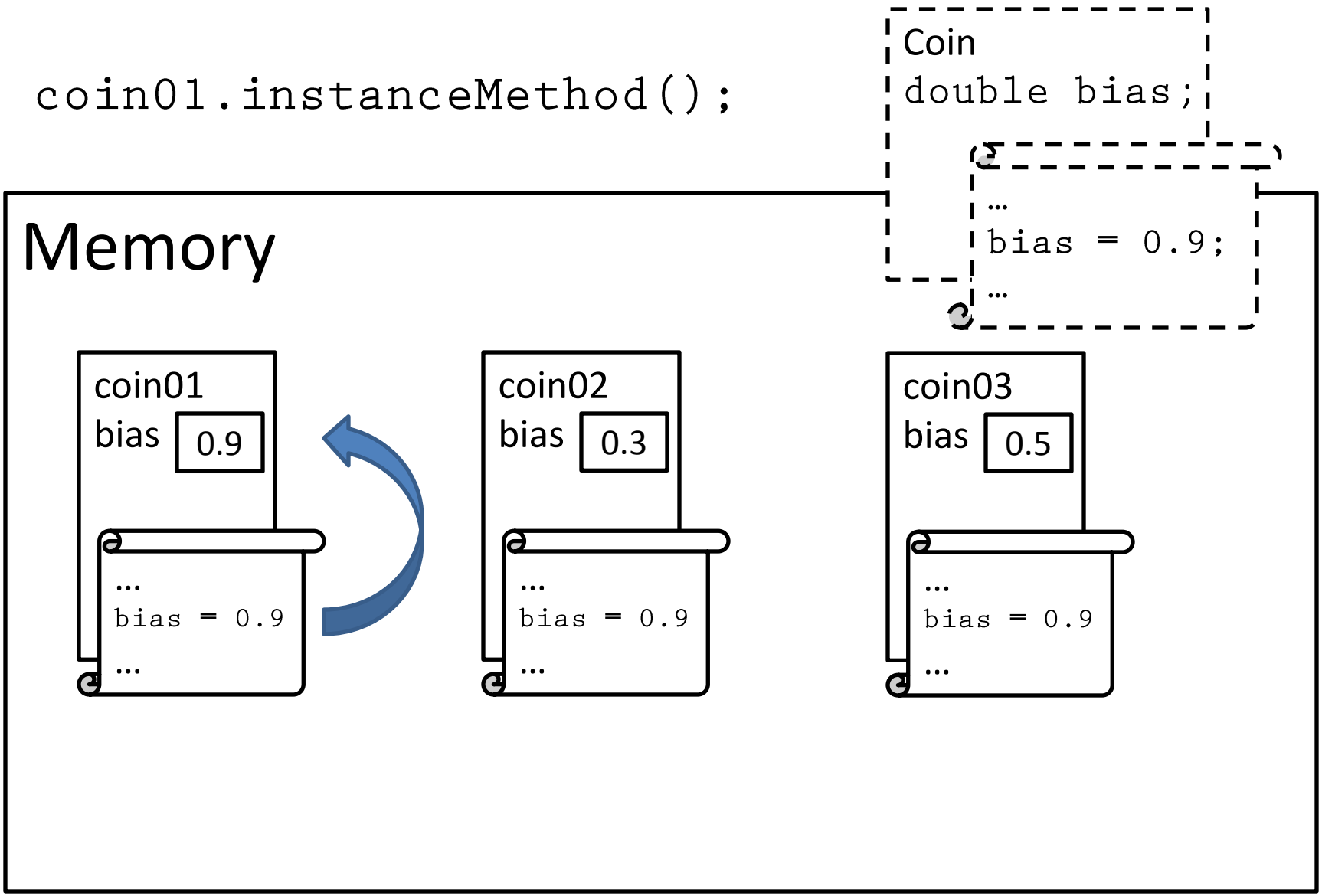
Constructor

- Method with same name as the class
- Lets you initialize the fields of an individual instance

```
public class Integer {
    int intValue;
    public Integer(int value) {
        intValue = value;
    }
 }
```

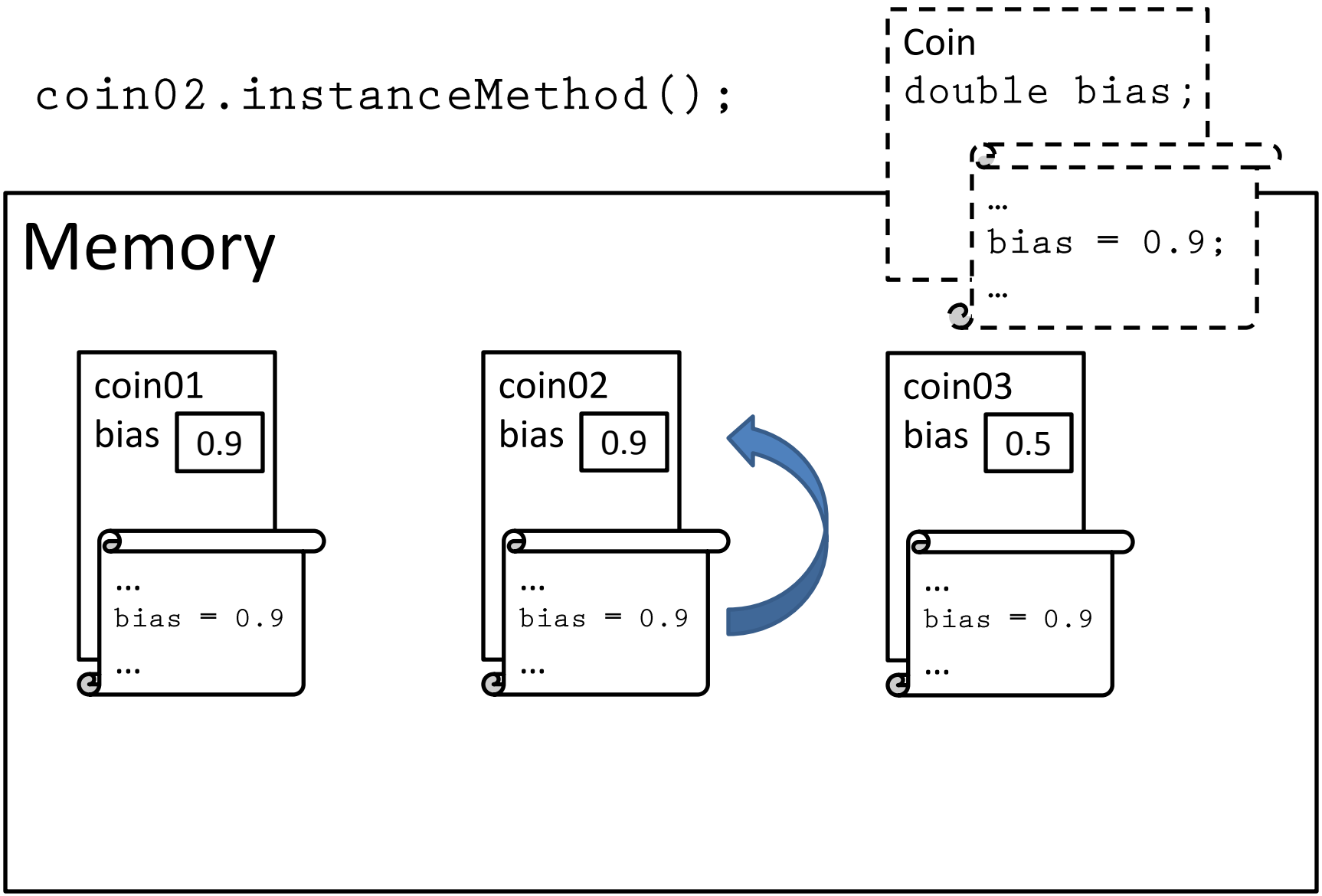
Instance Methods

```
coin01.instanceMethod();
```



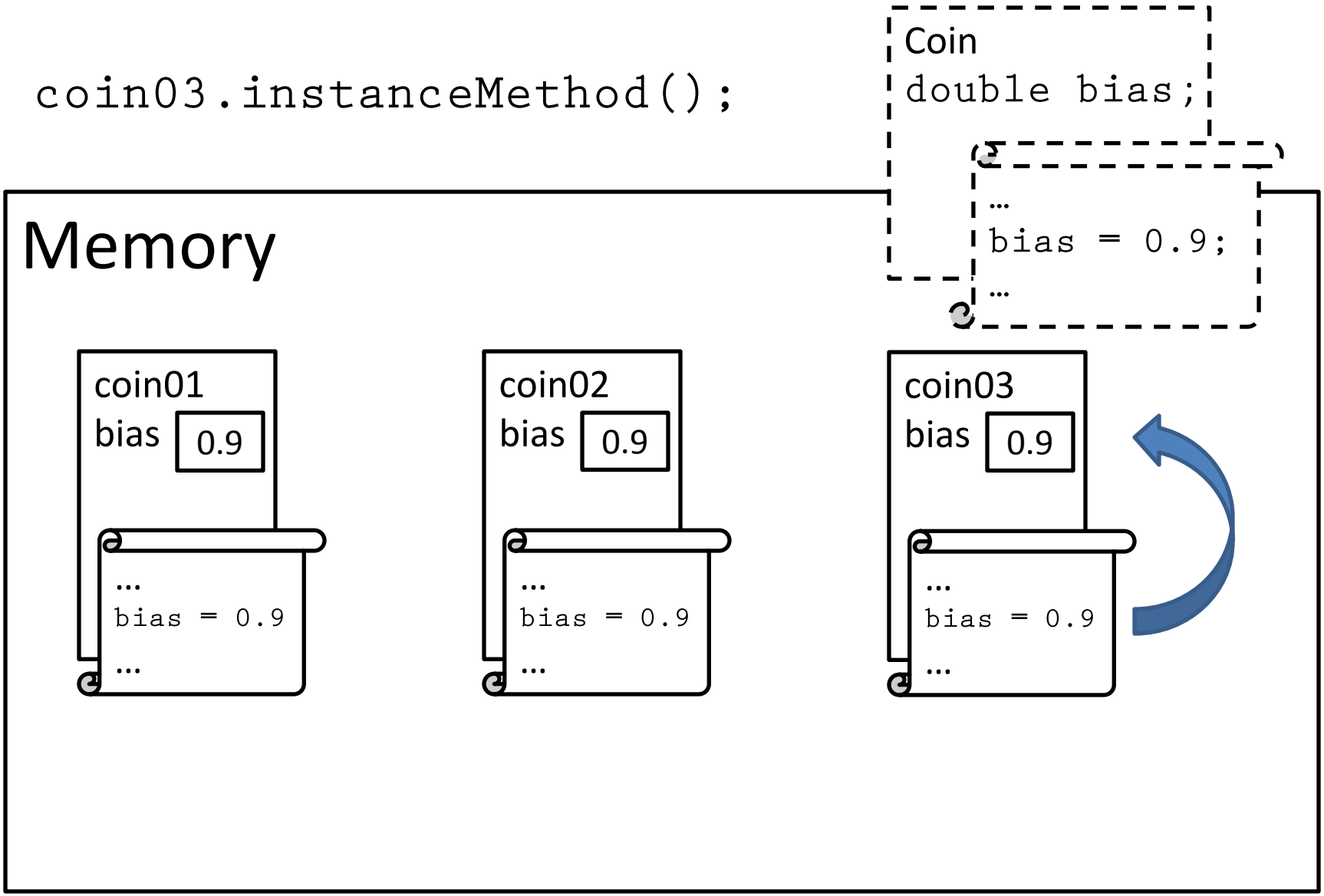
Instance Methods

```
coin02.instanceMethod();
```

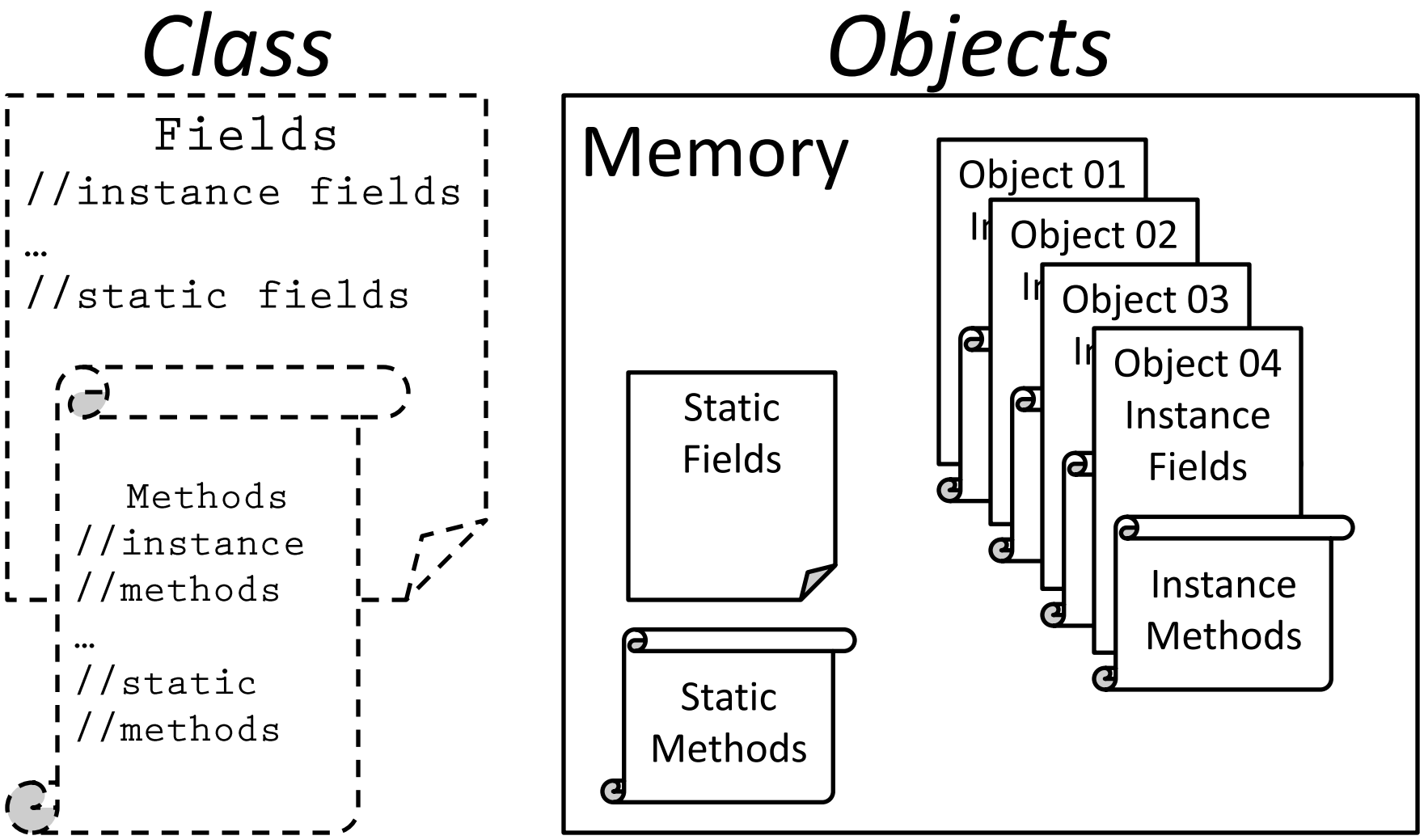


Instance Methods

```
coin03.instanceMethod();
```



Static fields and methods



Static fields and methods

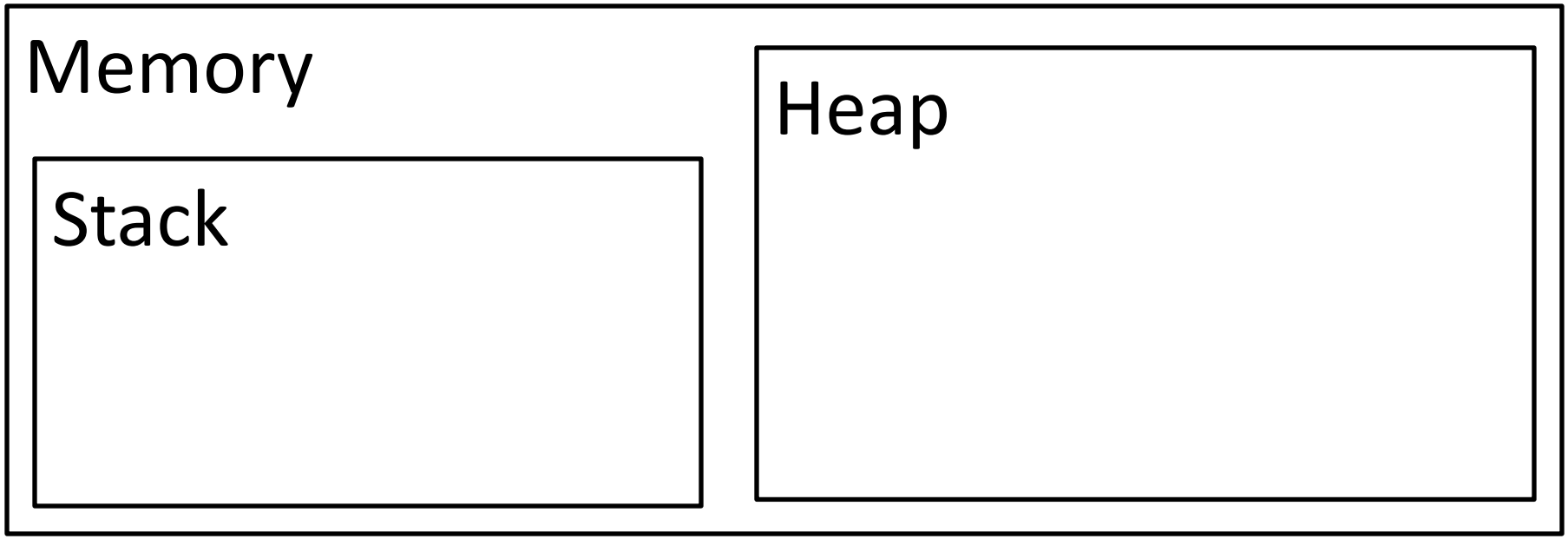
- Instance fields and methods are specific to each individual
- Static fields and methods are global entities that do not depend on the particular data in any one object

Values and references

- Primitive type variables like ints and booleans require a fixed number of bytes, and store basic values like numbers or truth values.
- The value of an Object variable is a *memory address*, also called a *reference*.
  - Aliasing: when two reference variables contain the same memory address (Problems??)
- Stack: the portion of memory containing fixed-size data, like primitive types and memory addresses
- Heap: the portion of memory containing object data (where the memory addresses point to)

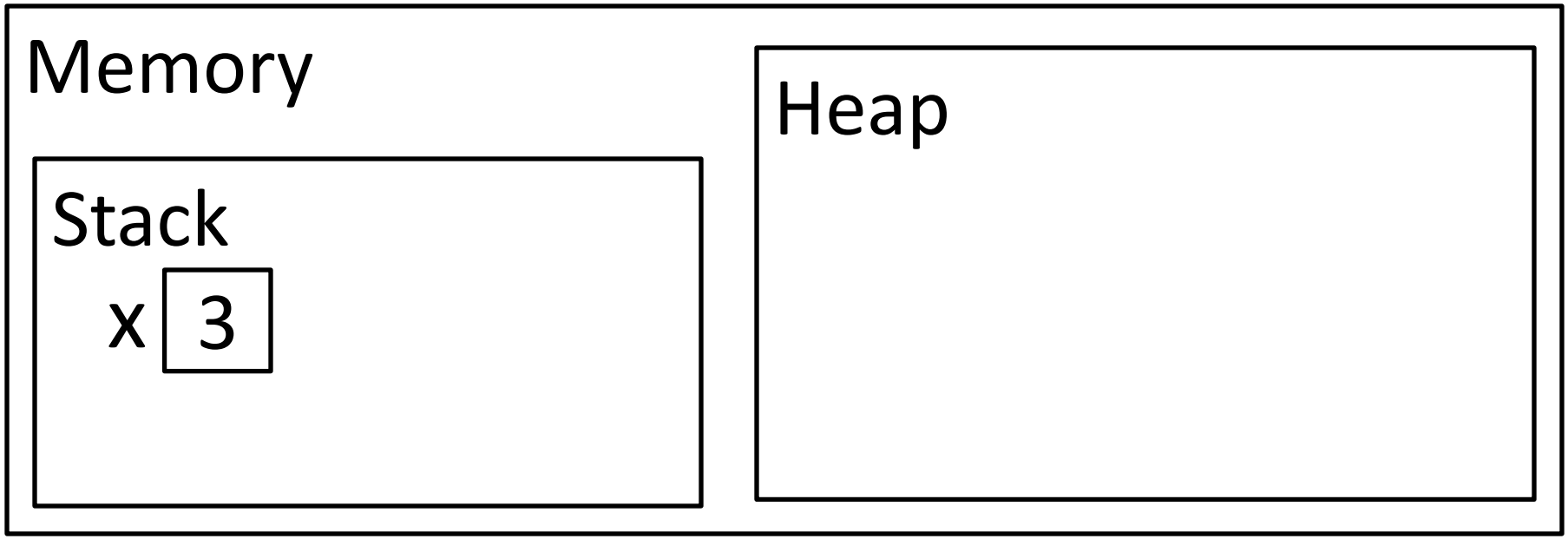
Values and References

```
public static void main(String[] args) {
        int x = 3;
        int y = x;
        String str1 = “Hello!”
        String str2 = str1;
}
```



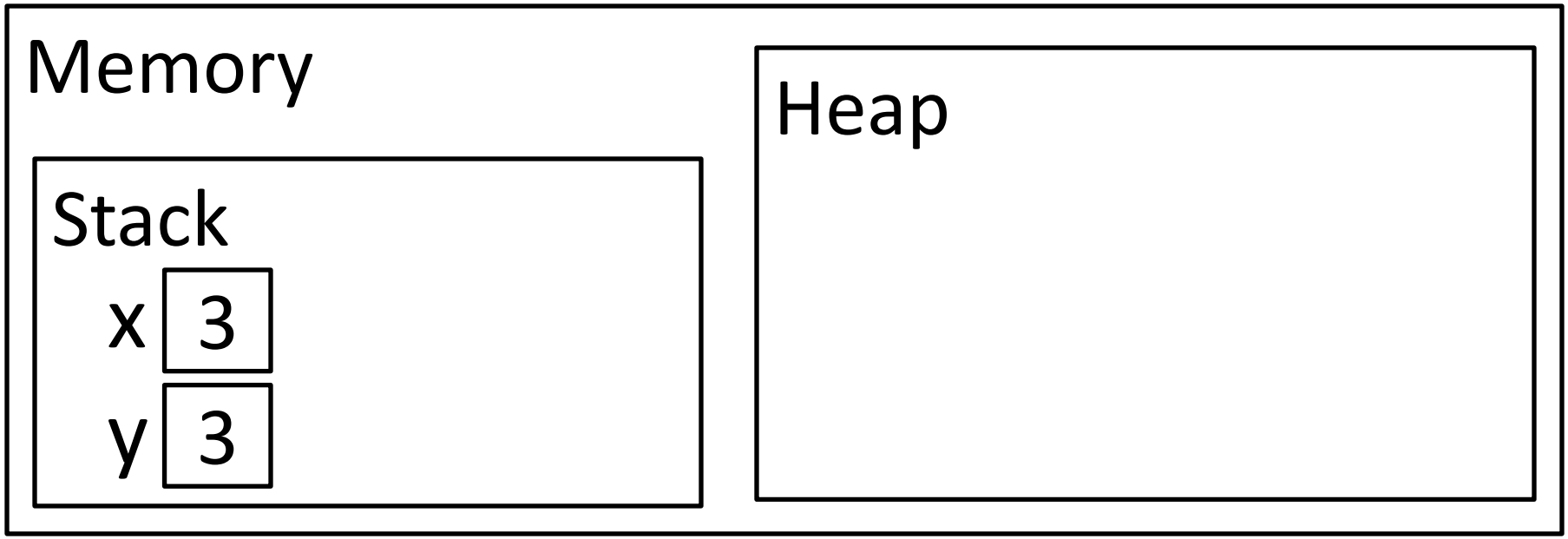
Values and References

```
public static void main(String[] args) {
        int x = 3;
        int y = x;
        String str1 = “Hello!”
        String str2 = str1;
}
```



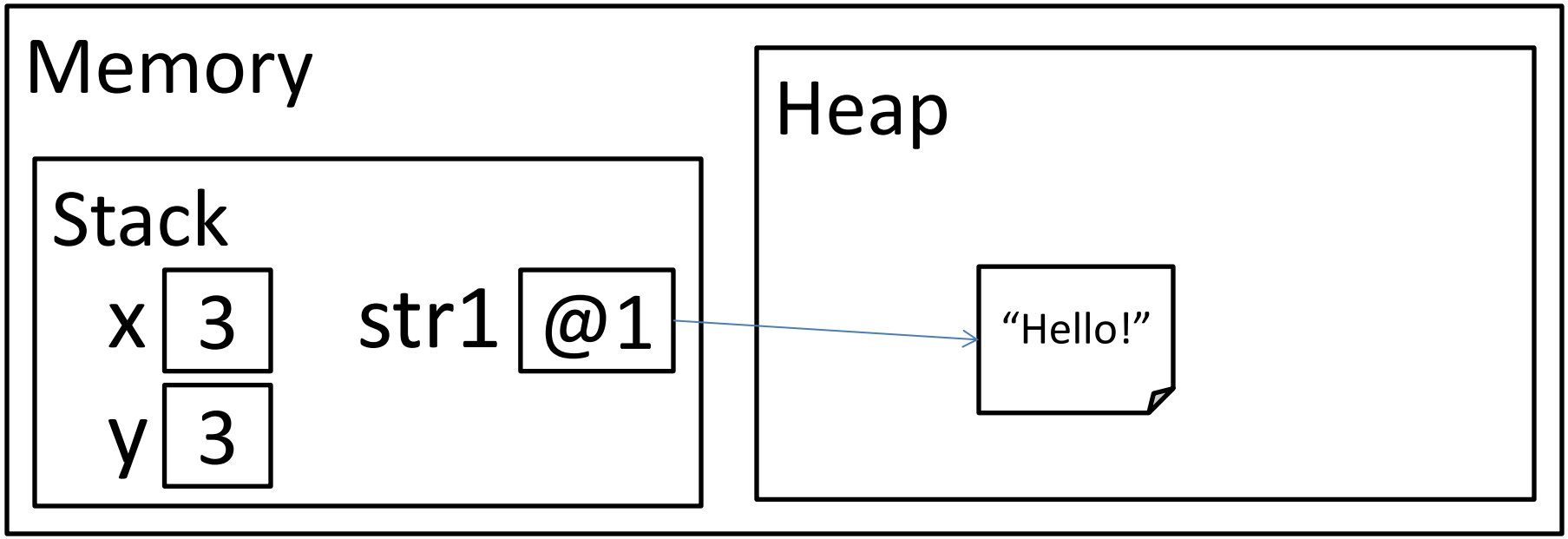
Values and References

```
public static void main(String[] args) {
        int x = 3;
        int y = x;
        String str1 = “Hello!”
        String str2 = str1;
}
```



Values and References

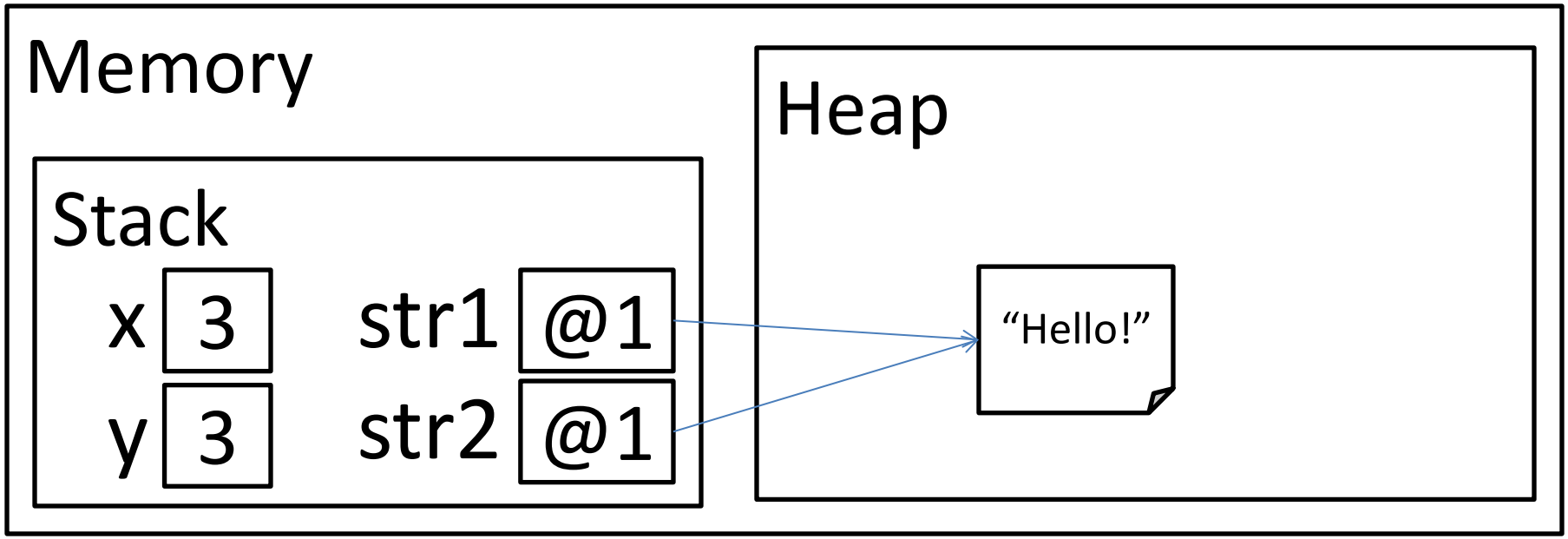
```
public static void main(String[] args) {
        int x = 3;
        int y = x;
        String str1 = “Hello!”
        String str2 = str1;
}
```



Values and References

```
public static void main(String[] args) {
        int x = 3;
        int y = x;
        String str1 = “Hello!”
        String str2 = str1;
}
```

“Aliasing”

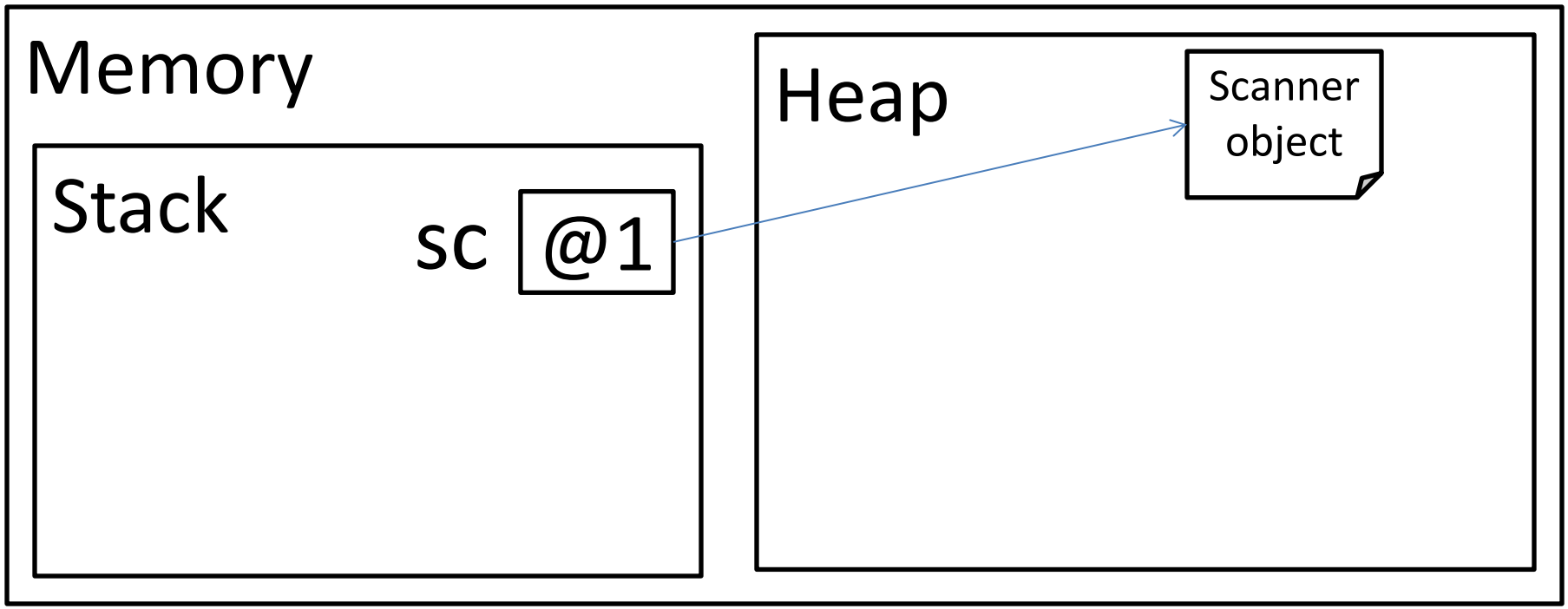


The `new` keyword

- Must be used with constructor
- Tells Java to allocate space on the heap for the new object

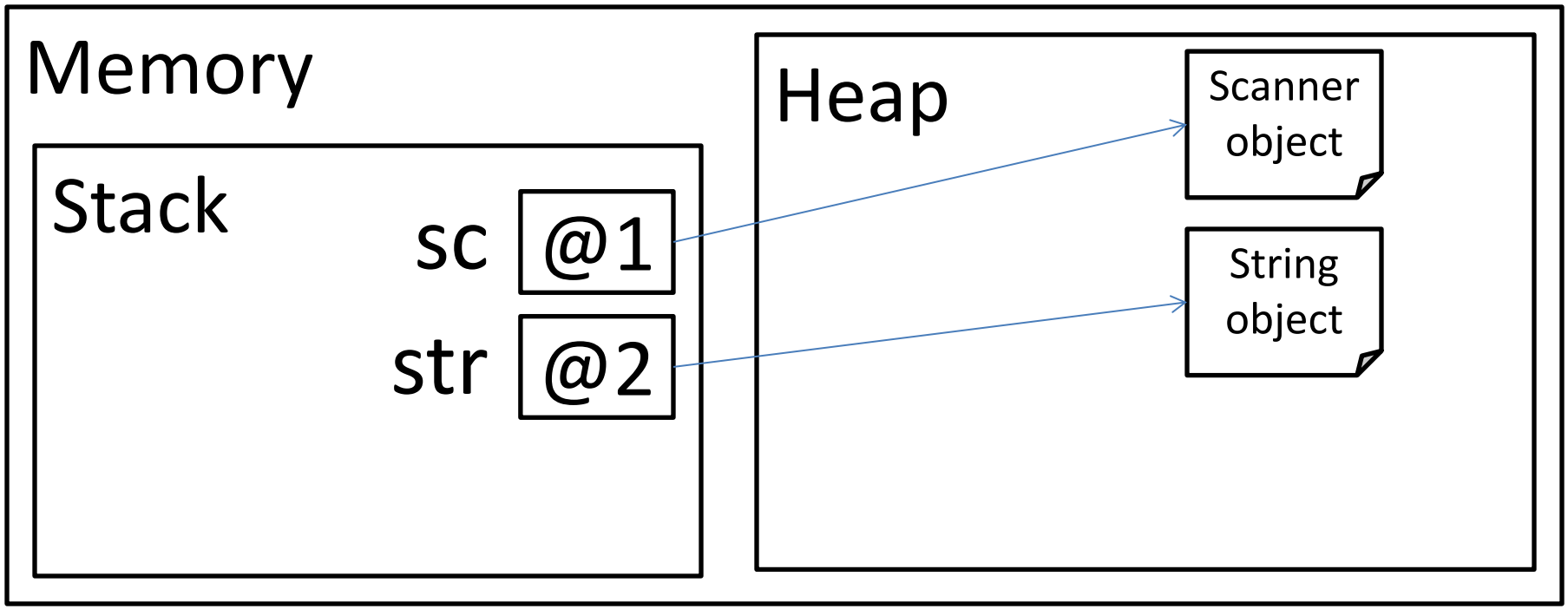
The `new` keyword

```
public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String str = new String("Hello");
        Integer i = new Integer(3);
}
```



The `new` keyword

```
public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String str = new String("Hello");
        Integer i = new Integer(3);
}
```



The `new` keyword

```
public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String str = new String("Hello");
        Integer i = new Integer(3);
}
```