Announcements

- P2 due tonight
- P3 posted today
- Quiz 2 in discussion tomorrow
  - Answers to study questions posted today
- Code on piazza policy

Follow ups

- Aliasing example (in CVS)
- Primitive “wrappers” (Integer, etc.)
  - > libraries and packages
- Where are static variables stored?
  - > garbage collection
  - > other inner-workings of Java

Primitive wrappers

- Objects to represent primitive types, and provide additional functionality beyond the built-in operators
  - Byte, Boolean, Float, Integer, etc.
- Auto-boxing/unboxing

  http://docs.oracle.com/javase/7/docs/technotes/guides/language/autoboxing.html
  - Automatic conversion between primitives and wrappers.
  - Useful when using “collections” (more later)
- Wrapper example

Libraries

- Pre-existing classes that provide common functionality
  - Strings (java.lang.String)
  - Basic math (like sqrt) (java.lang.Math)
  - User Input (java.util.Scanner)
  - Linked List (java.util.LinkedList)
  - GUI buttons (java.awt.Button)

Packages

- Groups of related libraries for organizational purposes
  - Java.lang
  - Java.util
  - Java.awt
  - list (Lab04)
    - list.IntList
    - list.IntListEntry
- Can have sub-packages nested inside packages

Using packages and libraries

- To use libraries, place an “import” statement at the top of your class definition, e.g.:

```
import java.util.Scanner;
```

- If you plan on using many classes from a single package, you can import them at once

```
import java.util.*;
```

  – Although this doesn’t import sub-packages

- The package java.lang is considered so essential that it is automatically imported.

Your own packages

- Each class in the package should have the package name at top

```
package myFirstPackage;
```

- Use an import statement to use your package in an external class

```
import myFirstPackage.*;
```

- Package example
- You can import packages from a different project:
  - Right-click on project -> Properties -> Build Path -> Projects -> Add…

package-private access

- `public` members: accessible from anywhere
- `private` members: only accessible from within the same class
- *package-private* members: only accessible from within the same package.
  - Default access level, when no access modifier is provided
  - Package-private example

Javadocs

- The standard way of documenting your own class.
- Javadocs can be generated automatically from comments in your source code.
  - Javadoc example for comment format
- By default only public members get published
- In Eclipse:

  Right-click on file -> Export -> Java -> Javadoc -> Next -> Finish

Static variable storage

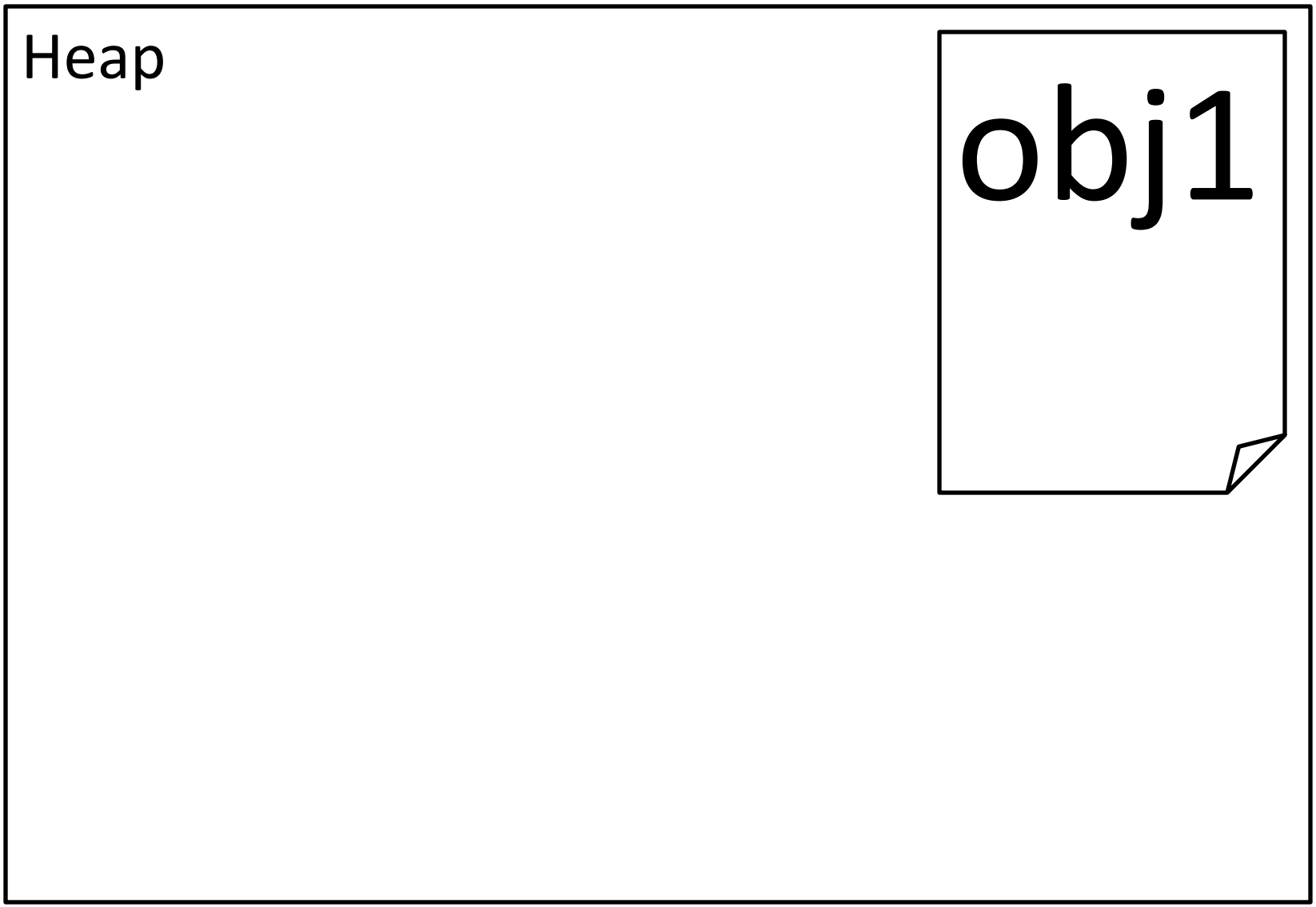
- Where are static variables stored?
  - Dr. Golub's answer
  - Stack overflow's answer

    http://stackoverflow.com/questions/3849634/static-allocation-in-java-heap-stack-and-permanent-generation
  - Oracle blogs' answer

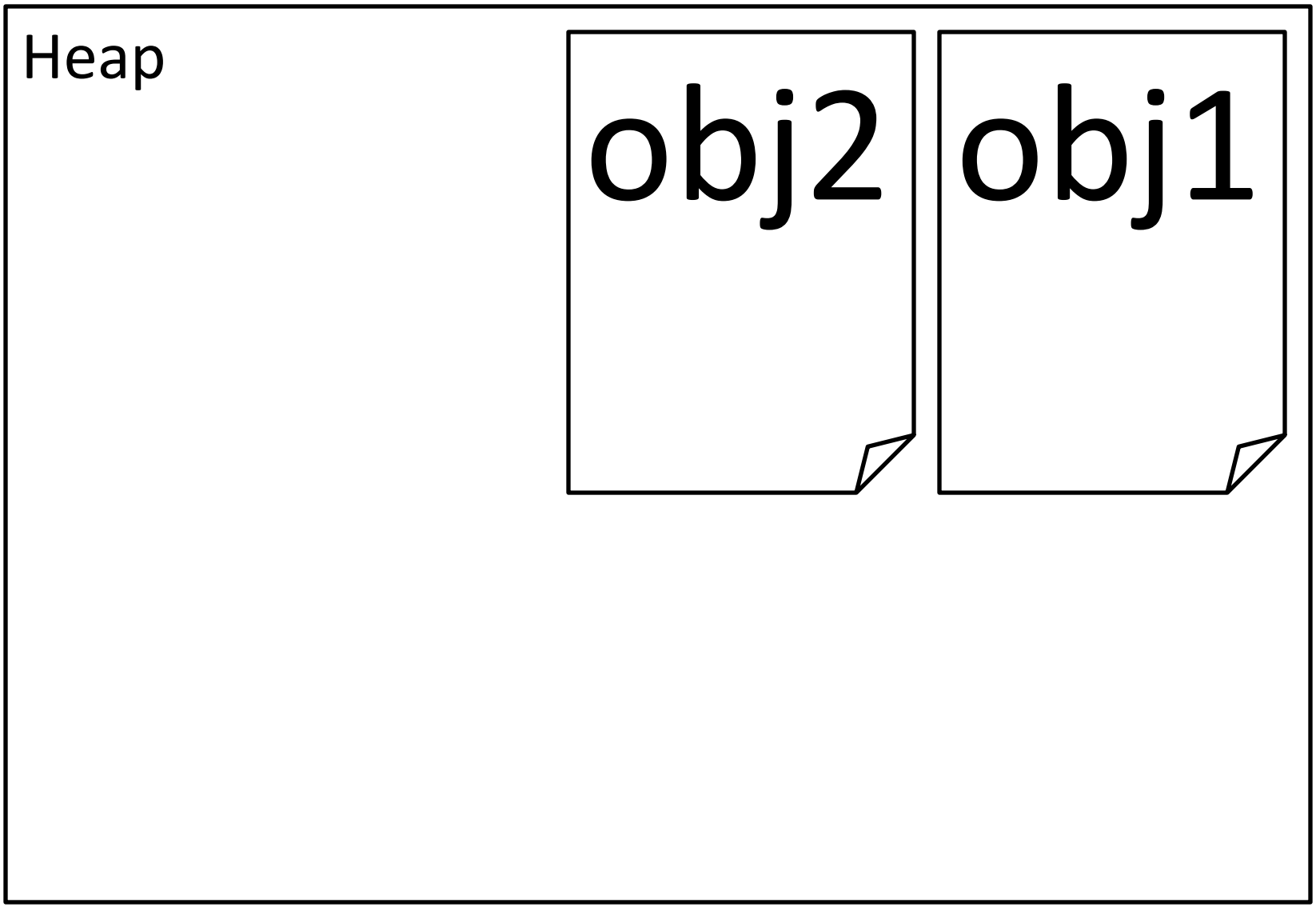
    https://blogs.oracle.com/jonthecollector/entry/presenting_the_permanent_generation
  - Java Language Spec's answer

    http://docs.oracle.com/javase/specs/jls/se7/html/jls-17.html#jls-17.4.1
- -> Garbage collection

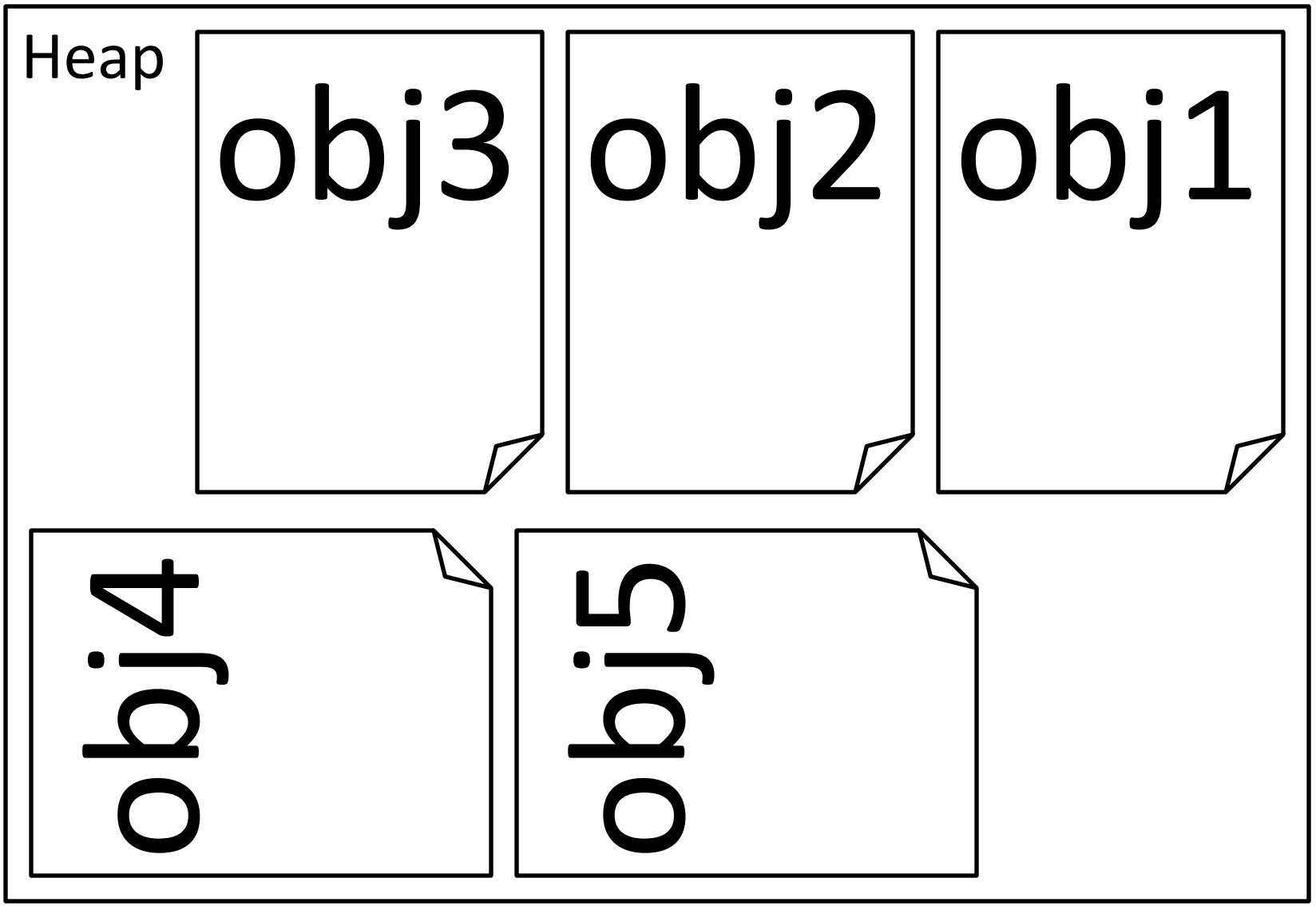
Limited memory



Limited memory



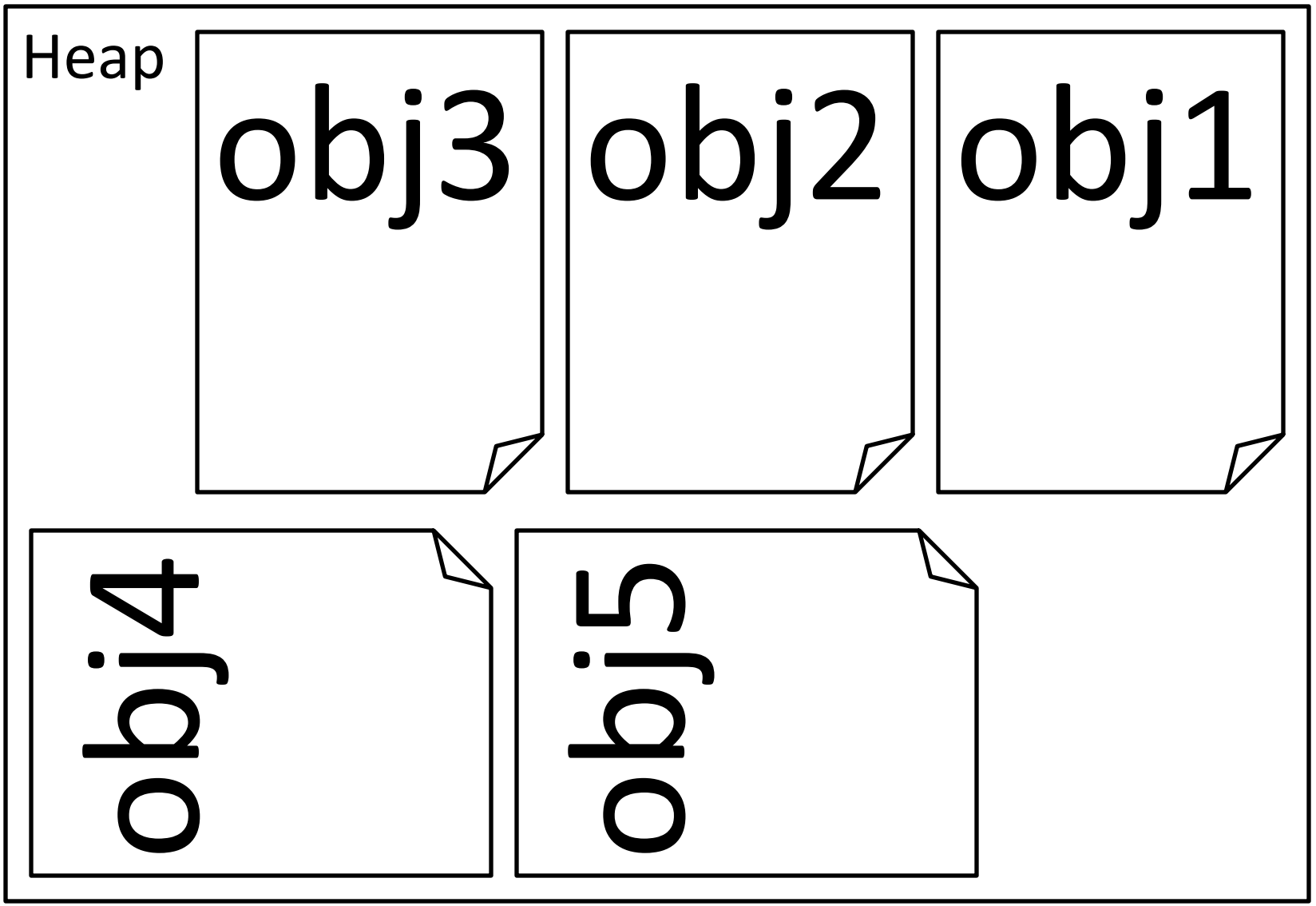
Limited memory



Garbage collection

- Garbage example
- Java will automatically identify objects that are “garbage” (no longer needed) and de-allocate the memory so that it can be used by new objects
- “Generations” of objects: how many garbage collections have they survived?
  - Used to make this process more efficient
- The permanent generation is never collected.

Garbage collection



Garbage collection

Heap

obj3

obj2

obj1

obj4

obj5

Garbage collection

Heap

obj3

obj2

obj1

obj4

obj5

Garbage collection

Heap

obj3

obj1

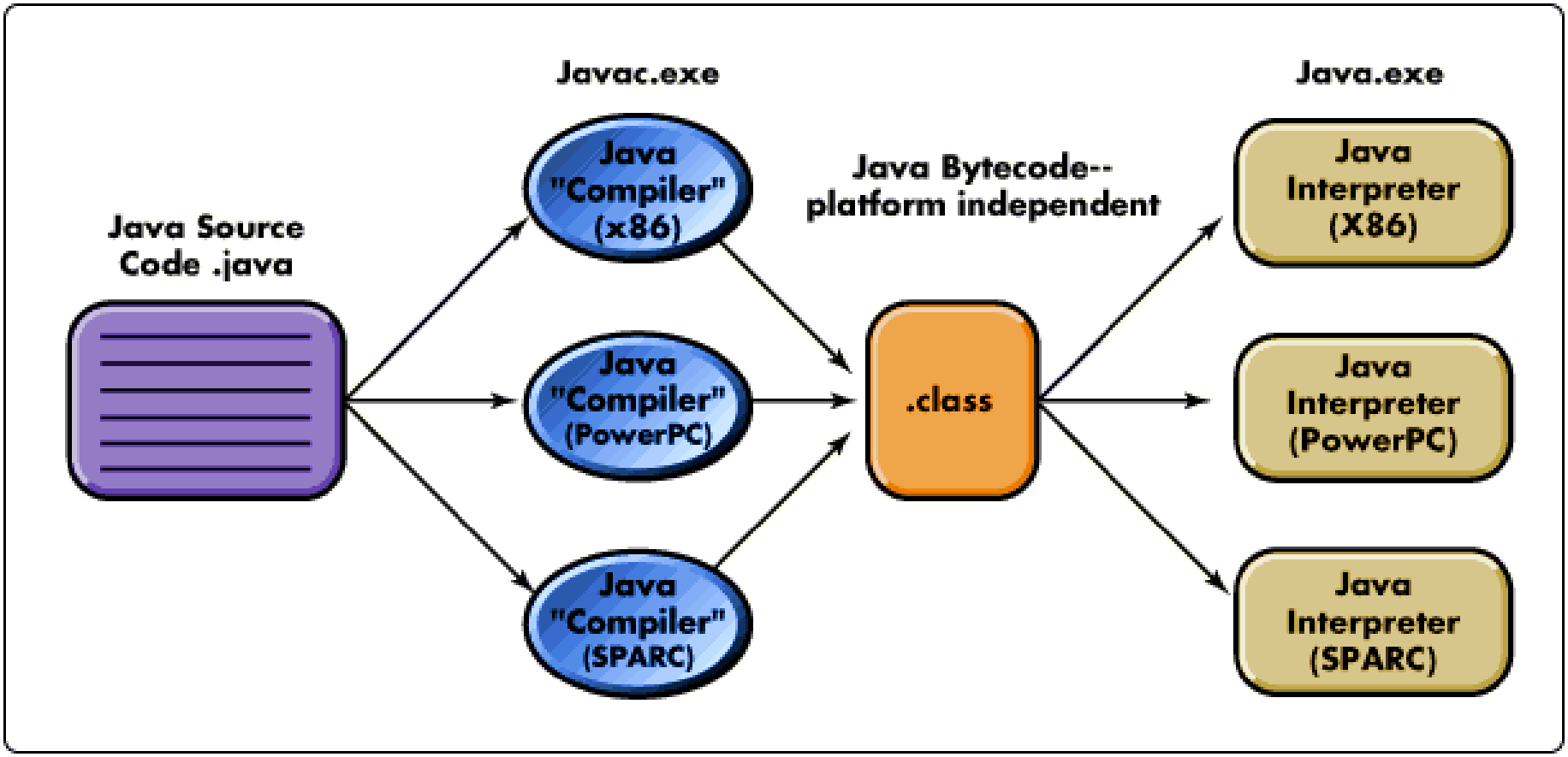
obj4

obj5

Java Virtual Machine

- Makes Java *portable*
  - Java programs can run on “any” hardware
- Simulates an imaginary piece of hardware with it’s own machine code, called “bytecode”
- .class files consist of bytecode
- The JVM translates bytecode to the actual hardware-dependent machine code of your system.

Java Virtual Machine



http://support.novell.com/techcenter/articles/img/ana1997070102.gif

Java Virtual Machine

- Interpreted or compiled?
  - Can interpret bytecode, instruction by instruction
  - Can fully compile bytecode to actual machine code
  - Can perform "Just-In-Time" compilation
    - Bytecode is grouped into segments, and the segments are compiled and executed one at a time.
    - A hybrid of interpretation and compilation.

More acronyms

- **J**ava **R**untime **E**nvironment
  - Contains the JVM
  - Contains Java's standard class libraries
- **J**ava **D**evelopment **K**it
  - Contains the JRE
  - Contains several other tools to aid in software development

A closer look at toString()

- The toString() method is used to produce a String representation of an object.
- By convention, every class should implement this method.
- Returns a String, doesn’t println.
- toString example

A closer look at toString()

- Implicit calls:

```
System.out.println(obj);
```

actually executes

```
System.out.println(obj.toString());
```

- Default toString(): calls hashCode();

http://docs.oracle.com/javase/7/docs/api/java/lang/Object.html#hashCode()

  - Typically based on memory address (but not necessarily).