



Introduction to Objects

January 31, 2012

CMPSCI 121, Spring 2012

Introduction to Problem Solving with Computers

Prof. Learned-Miller

Logistics

Email

1. Send emails through OWL, not directly to me or TAs.
2. Include as much detail as possible
 1. What exactly did you do?
 2. What exact problem did you have?
 3. Other info:
 1. Machine type (Mac, PC, Linux)
 2. Versions of software: Java 1.2, etc.

Wrong way to do it

- Emails:

“Yo dude! My thing don’t work.
Wazzup?”

“I can’t get DrJava to work.”

Right way to do it

- Emails:

“Esteemed Prof., Master of all things Java, I have a Mac and after downloading and installing DrJava according to the instructions on the course web page, when I double click on the DrJava icon, nothing happens. Can you help?”

Embedded Exercises: How to see your current totals

The screenshot shows the OWL iJava 1.3 interface. At the top left is the OWL logo. To its right is the text "iJava 1.3". Below the logo is a navigation bar with two dropdown menus: "Chapter 1: Java Introduction" and "Java Introduction". To the right of these menus is a "Chapter Overview" button. Below the navigation bar is a "Reference Tools" button with a dropdown arrow. On the left side, under the heading "In This Chapter...", there is a paragraph of text. On the right side, there is a "Chapter Outline" box containing a list of links.

OWL iJava 1.3

Chapter 1: Java Introduction

Java Introduction

Chapter Overview

Reference Tools

In This Chapter...

This chapter begins with a discussion of hardware and software, and continues with a discussion of the role of high-level languages such as Java in computing. Then compilation is discussed, and Java's unusual two-stage compilation process is described. Next, Java's basic print statements are introduced, along with Java's multi-use + operator. Lastly the concept of a Java program, or application, is presented.

Chapter Outline

- 1.1 Hardware, Software, and Languages
- 1.2 The Nature of Java
- 1.3 Starting Java
- 1.4 First Programs
- 1.4.a Worked Example: String Literals
- Chapter Summary

Next

Embedded Exercises: How to see your current totals

The screenshot shows the OWL iJava 1.3 interface. At the top left is the OWL logo. To its right is the text 'iJava 1.3'. Below the logo is a navigation bar with two dropdown menus: 'Chapter 1: Java Introduction' and 'Java Introduction'. To the right of these menus is a 'Chapter Overview' button. Below the navigation bar, on the right side, is a 'Reference Tools' button with a dropdown arrow, which is circled in red. Below the 'Reference Tools' button is a green 'Next' button with a right-pointing arrow. On the left side of the main content area, under the heading 'In This Chapter...', there is a paragraph of text. On the right side, there is a 'Chapter Outline' section with a list of links: '1.1 Hardware, Software, and Languages', '1.2 The Nature of Java', '1.3 Starting Java', '1.4 First Programs', '1.4.a Worked Example: String Literals', and 'Chapter Summary'.

OWL iJava 1.3

Chapter 1: Java Introduction

Java Introduction

Chapter Overview

Reference Tools

Next

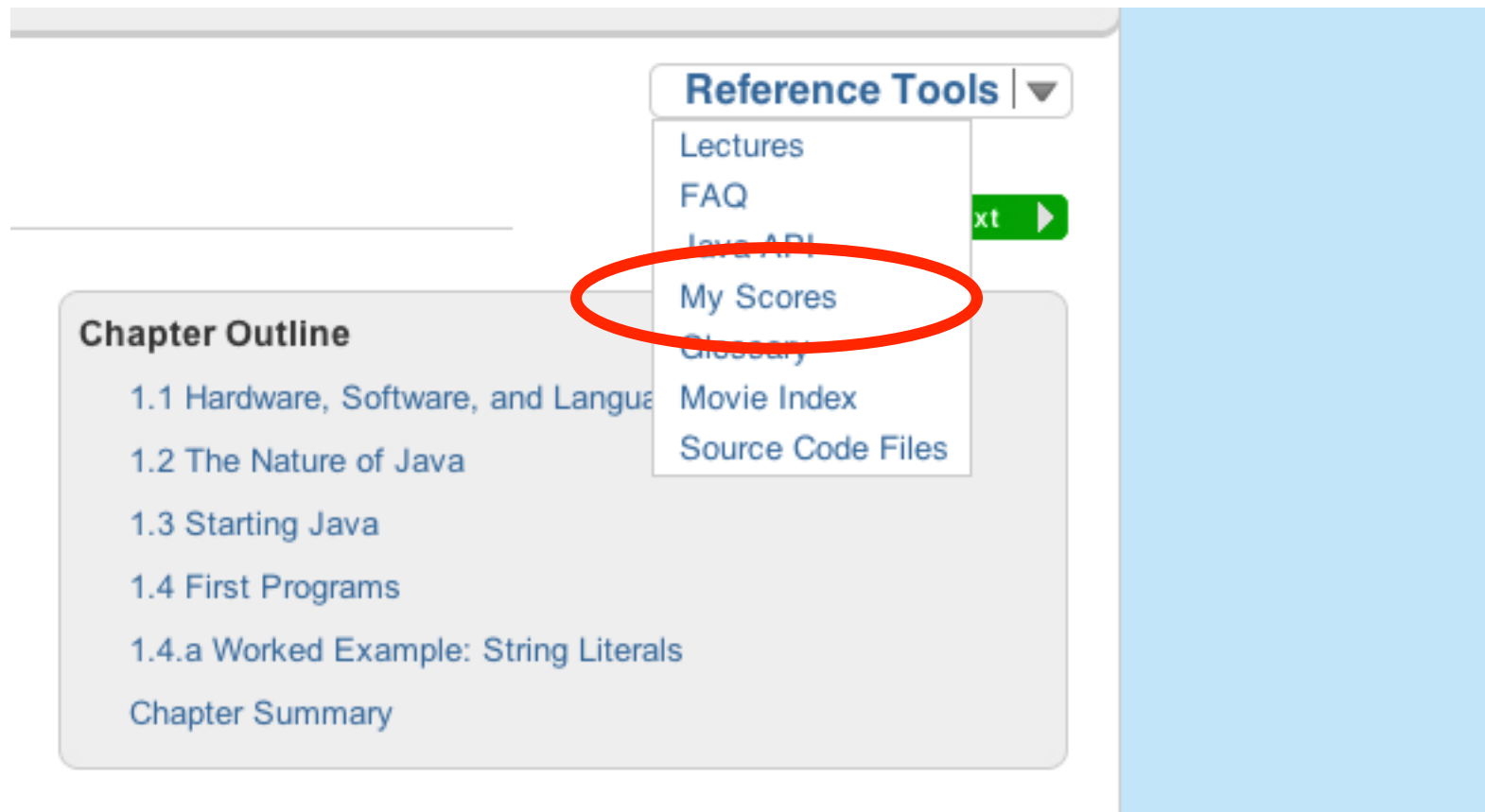
In This Chapter...

This chapter begins with a discussion of hardware and software, and continues with a discussion of the role of high-level languages such as Java in computing. Then compilation is discussed, and Java's unusual two-stage compilation process is described. Next, Java's basic print statements are introduced, along with Java's multi-use + operator. Lastly the concept of a Java program, or application, is presented.

Chapter Outline

- 1.1 Hardware, Software, and Languages
- 1.2 The Nature of Java
- 1.3 Starting Java
- 1.4 First Programs
- 1.4.a Worked Example: String Literals
- Chapter Summary

Embedded Exercises: How to see your current totals



Embedded Exercises: How to see your current totals

Chapter	Correct/Total
Chapter 1	2 /5
Chapter 2	0 /11
Chapter 3	0 /19
Chapter 4	0 /30
Chapter 5	1 /10
Chapter 6	0 /9

Some reminders

- Previous lectures on course web page
- No late assignments accepted
 - Can drop 2 exercises, and 1 programming
- Video of lectures on line (thru OWL menu).



Assignments due today, 11:30pm!

Assignment	Due Date	Test Stage
✓ Configuration Tester	1/31/2012 11:30 PM	1 of 1
✓ eBook - Chapter 0: Title Page and Preface	1/31/2012 11:30 PM	NA
✓ Owl Tutorial	1/31/2012 11:30 PM	8 of 8
Introductory Survey	1/31/2012 11:30 PM	<i>not started</i>

OWL and DrJava Problems

- If you are having DrJava Problems or OWL problems, you better go to office hours today and get them resolved. Bring your computer to office hours if you can.

More Assignments!

	 eBook - Chapter 1: Java Introduction	2/1/2012 11:30 PM
	Name and Height	2/2/2012 11:30 PM
	 eBook - Chapter 2: Objects and Classes	2/2/2012 11:30 PM
	Chapter 1 Exercises	2/3/2012 11:30 PM
	Chapter 2 Exercises	2/3/2012 11:30 PM
	eBook - Chapter 3: Classes, Strings, and I.O.	2/7/2012 11:30 PM

The Bulletin Board

- Instructions under “resources”
- http://bb-edlab.cs.umass.edu/cs121_S12/
- Off-campus access:
 - Username: cs121
 - Password: EL121p



CS121 Bulletin Board S07

[FAQ](#) [Search](#) [Memberlist](#) [Usergroups](#) [Register](#)
[Profile](#) [Log in to check your private messages](#) [Log in](#)

The time now is Mon Feb 05, 2007 2:42 pm
[CS121 Bulletin Board S07 Forum Index](#)

[View unanswered posts](#)

The Bulletin Board

yourdomain.com • Index page

http://bb-edlab.cs.umass.edu/cs121_S09/index.php

SESAME - GAMES 121 MassTraveler Admits edlab umass cal me umc face det1 http://devel...ogGuide.pdf

yourdomain.com • Index page



phpBB yourdomain.com
creating communities A short text to describe your forum

Search... Search
Advanced search

[Board index](#) [User Control Panel \(0 new messages\)](#) • [View your posts](#) [FAQ](#) [Members](#) [Logout \[elm \]](#)

It is currently February 4th, 2009, 10:17 pm Last visit was: February 4th, 2009, 9:59 pm

[View unanswered posts](#) • [View new posts](#) • [View active topics](#) [Mark forums read](#)

YOUR FIRST CATEGORY	TOPICS	POSTS	LAST POST
 Your first forum Description of your first forum.	2	2	by elm  January 28th, 2009, 3:02 pm

WHO IS ONLINE

In total there are **2** users online :: 1 registered, 0 hidden and 1 guest (based on users active over the past 5 minutes)
Most users ever online was **2** on January 26th, 2009, 9:34 pm


Registered users: elm
Legend: *Administrators*, *Global moderators*

STATISTICS

Total posts **2** • Total topics **2** • Total members **2** • Our newest member **elm**

[Board index](#) The team • Delete all board cookies • All times are UTC

The Bulletin Board




CS121 Bul

[FAQ](#) [Search](#) [Me](#)
[Profile](#) [Log in to che](#)

OWL Problem Exercises

Moderators: None

Users browsing this forum: None

 [CS121 Bulletin Board S07 Forum Index -> OWL Problem](#)

Topics	Replies	Author
--------	---------	--------

The Bulletin Board

DO NOT POST CODE!

Go to office hours
instead

General questions only

Comments about “Name and Height”

- Write a program to print your name and your height in centimeters.
- It's just going to be a few println statements, and one multiplication.
 - You must do the multiplication in the program.

Back to Java

The “main” method

```
1 public class Greetings{  
2     public static void main(String[] args){  
3         System.out.print("Hello ");  
4         System.out.print("out ");  
5         System.out.print("there");  
6     }  
7 }
```

The “main” method

```
1 public class Greetings{
2     public static void main(String[] args){
3         System.out.print("Hello ");
4         System.out.print("out ");
5         System.out.print("there");
6     }
7 }
```

The “main” method

```
1 public class Greetings{  
2     public static void main(String[] args){  
3         System.out.print("Hello ");  
4         System.out.print("out ");  
5         System.out.print("there");  
6     }  
7 }
```

main is where the program starts

Review

Strings and numbers

- `System.out.println("Stuff I want to say.");`
Stuff I want to say.
- `System.out.println("Sch" + "wing");`
Schwing
- `System.out.println(3+5);`
8
- `System.out.println("3+5");`
3+5

Strings and numbers

- Mixtures of strings and numbers.
 - From left to right, combine strings and numbers.
 - number + number → number
 - string + string → string
 - number + string → string
 - string + number → string (how can you remember this?)
- What will this do?
 - `System.out.println(3+5+"a"+3+5);`

Terms you need to learn **NOW!**

- *declarations*
- *variables, variable names, identifiers*
- *assignments*
- *types*
- *strings*

Be prepared for an AVALANCHE of terminology. This is a huge part of computer science.

Declarations, Types, and Variable Names

- `int x;` // An integer.
- `String name;` // A group characters.
- `double mileage;` // A decimal number.

Types

■ Native Types

- `int` // An integer.
- `String` // A group of characters.
- `double` // A decimal number.
- `char` // A single character.

■ User-defined Types (objects!)

- `Infant`
- `EriksFancyTaxForm`

Assignments

- `x = 3;`
- `name = "Erik";`

Combining declarations and assignments

- `int x = 3;`
- `String name = "Erik";`
- `double myDecimal = 123.456;`

10 Minutes with the Doctor

- “Play. Then *predict.*”

The rest of today

- Classes, objects, attributes, methods, ...



Here is a program...

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Here is a program...

Where does it start?

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Here is a program...
Where does it start?

```
1 public class InfantTester{  
2  
3     public static void main (String[] args){  
4         Infant myKid = new Infant("Lizzie",4);  
5         int lizAge = myKid.getAge();  
6         System.out.println("my kid's name is " + myKid.getName());  
7         myKid.anotherMonth();  
8         System.out.println("my kid is now " + myKid.getAge() + " months");  
9     }  
10 }
```

Here is a program...
Where does it start?

```
1 public class InfantTester{  
2  
3     public static void main (String[] args){  
4         Infant myKid = new Infant("Lizzie",4);  
5         int lizAge = myKid.getAge();  
6         System.out.println("my kid's name is " + myKid.getName());  
7         myKid.anotherMonth();  
8         System.out.println("my kid is now " + myKid.getAge() + " months");  
9     }  
10 }
```

“main is where the program starts.”

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

There are 5 statements inside the *body* of this program.

Each statement can be thought of as a single step, or as a program itself.

“Get some barbecue sauce.”

A single step or a whole recipe?

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Line 4: Create a “file folder” for an Infant.

Record the name of the Infant as “Lizzie”.

Record the age of the Infant as 4.

Finally, make a name for the FOLDER, (not the infant), called myKid.

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Line 5: Go to the “file folder” called myKid.
Get the number that was stored there for the
Infant’s age.
Store the age of the Infant in a variable called lizAge.

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Line 6: A println statement!

Print something that says “my kid’s name is “,
followed by the name of the Infant, which is stored
in the “file folder” called myKid.

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Line 7: Go to the “file folder” called myKid, and do something that adds one month to the age of the kid in the folder.

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Line 8: Another println statement!

Print out a string that says “my kids is now “,
followed by the age of the Infant stored in the file
folder myKid,
followed by “ months”

Objects

- To understand this program, we must start to learn about objects.
- We are going to start by thinking of objects as “containers” of information.

Containers

- **Box:**
 - Contains stuff
- **Briefcase**
 - Contains office papers and office supplies
- **Folder**
 - Contains papers and forms
- **Form**
 - “Contains” information

[illegible]

Containers (continued)

- Individual containers: (*instances*)
 - A specific box
 - A specific briefcase
 - A specific form
- A *class* of containers: (*types*)
 - Shoe boxes for Nike Air Treads
 - Gucci briefcases for 2012
 - 1040EZ Federal income tax form for 2011

Quiz



How many classes, how many instances?

Quiz



Answers:

3 **CLASSES** of containers.

6 **INSTANCES** of containers.

Today's featured container



Where do containers come from?

Where do containers come from?

- Answer: They must be *designed*.

Container designers

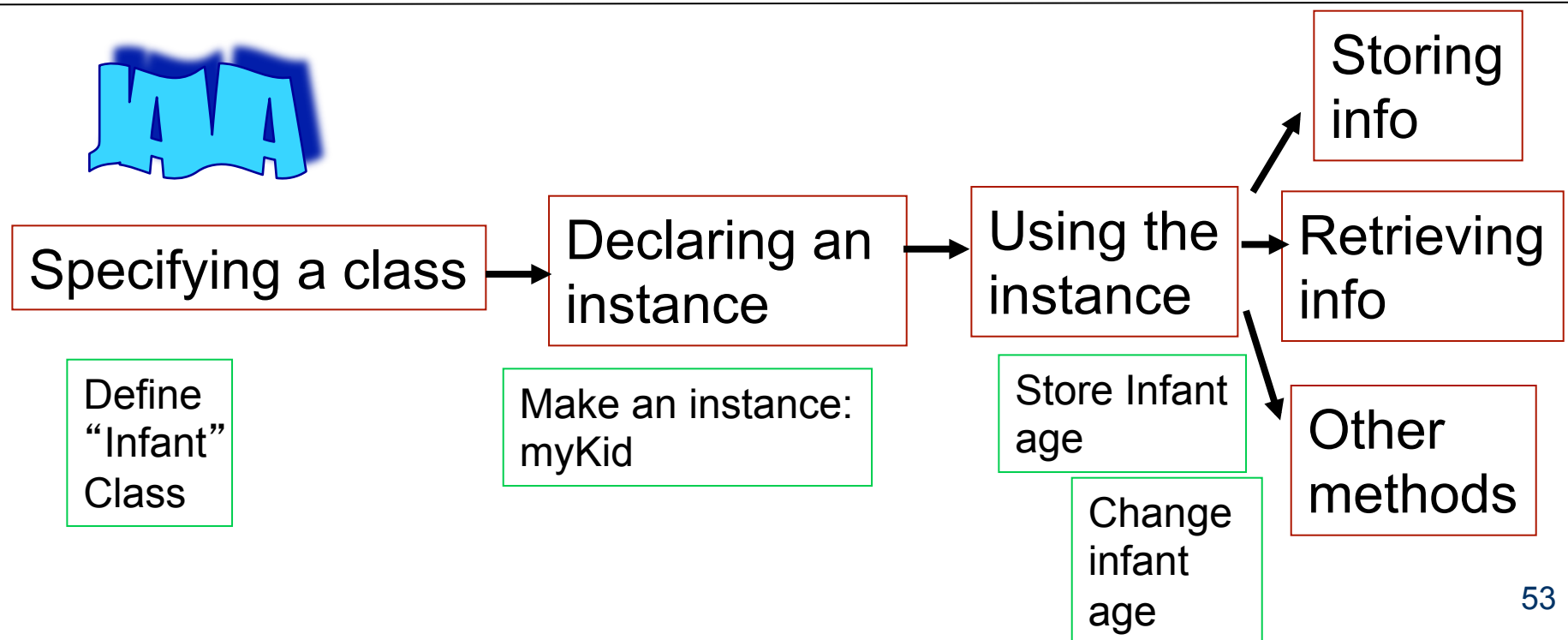
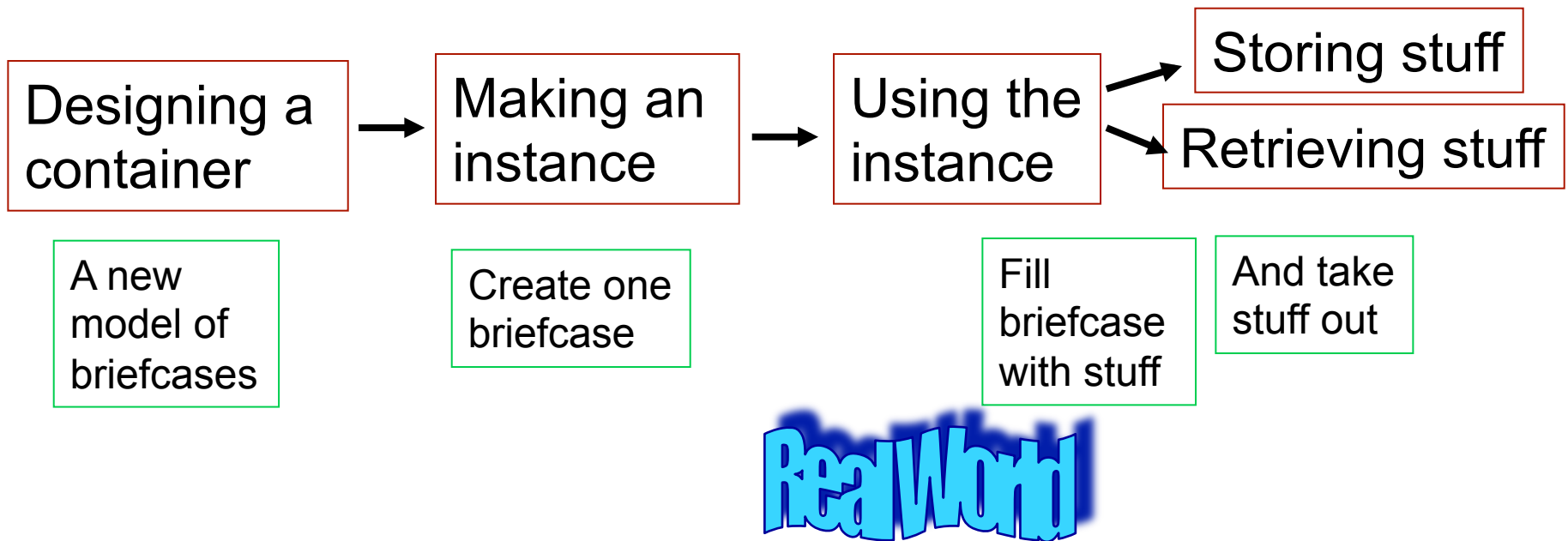
- Pocketbook designer for Gucci
 - A place for glasses? Make-up kit?...
- Backpack designer for NorthFace
 - How many bottle holders?
 - How many sub-compartments?
- Forms designer for IRS?
 - Place for SSN? For current residence?

You will be...

- Designing containers (*types of objects*) by creating *classes* in Java.
- Creating *instances* of objects using the *classes* that you designed, or that someone else designed.
- Using the *instances* of objects by storing information, retrieving information, and doing other things with them.

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```



Objects

- A way of staying organized.
 - “Containers” for information.
- Kind of like file folders.
 - Anywhere you can put information or store information.
 - Other examples:
 - Cardboard storage boxes
 - File cabinet
 - Name and address form
 - Tax form
- Why do we use “containers”?

Imagine an Office with No File Folders

- “Please get me Erik’ s address...
and Erik’ s medical history...
and Erik’ s dental insurance...
and Erik’ s financial records...
and”
- Folders allow commands like:
 - “Get me all of Erik’ s information”

Programs without objects

```
String ErikLastName="Learned-Miller";  
int ErikAge=42;  
int ErikHeightInInches=75;  
  
String VeditLastName="Jain";  
int VeditAge=27;  
int VeditHeightInInches=48;  
  
String ArmitaLastName="Kaboli";  
int ArmitaAge=26;  
int ArmitaHeightInInches=65;
```

Programs with Objects

- Person Erik;
- Person Vidit;
- Person Armita;
 - What information do we have on Armita?
 - Armita.LastName="Kaboli";
 - Armita.Age=26;
 - Armita.HeightInInches=65;

Java definitions

- **Class:** a type for objects
 - Infant (think of “InfantRecord”)
 - Person (think of “PersonRecord”)
 - Car (think of “CarRecord”)
- **Attribute:** one place to store info in an object
 - A slot for a pen in a briefcase
 - A field for your first name in a form
- **Method:** A way to tell the object to store or retrieve something, or do something else
 - Remember this person’s name
 - Give me this person’s age
 - Figure out when this person was born and tell me.

Java definitions

- **Constructor:** Create an instance of an object
 - Create one car from a car class
 - Create one tax form from a tax form class

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

Design (or blueprint) of an object:

The *Definition*

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

Design for an object: The *Definition*

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

Attributes: properties of the object (places to put stuff)

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

What's **public**, **private**?

- Basically:

- **public** : accessible from outside the class
- **private** : not accessible outside the class

Methods: stuff the object can do.

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

Methods: stuff the object can do.

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

Methods: stuff the object can do.

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

Constructor: A Special Method to Create the Object

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

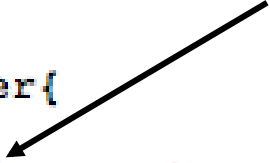
```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

```
1 public class Infant{
2
3     private String name;
4     private int age;    // age in months
5
6     public Infant(String who, int months){
7         name = who;
8         age = months;
9     }
10
11     public String getName(){return name;}
12
13     public int getAge(){return age;}
14
15     public void anotherMonth(){age = age + 1;}
16 }
```

Some Code

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Some Code



```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Some Code

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

A Declaration
Performing an Assignment
Calling a Constructor



Declarations and Assignments

```
Infant myKid = new Infant("Liz",4);
```

Could have done these separately:

```
Infant myKid; ← Declaration
```

```
myKid = new Infant("Liz",4);
```

← Assignment

Declarations and Assignments

```
Infant myKid = new Infant("Liz",4);
```

Could have done these separately:

```
Infant myKid; ← Declaration
```

```
myKid = new Infant("Liz",4);
```

← Assignment

Calling a method

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Something we recognize!

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Another method call

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

Some Code

```
1 public class InfantTester{
2
3     public static void main (String[] args){
4         Infant myKid = new Infant("Lizzie",4);
5         int lizAge = myKid.getAge();
6         System.out.println("my kid's name is " + myKid.getName());
7         myKid.anotherMonth();
8         System.out.println("my kid is now " + myKid.getAge() + " months");
9     }
10 }
```

That' s it for today