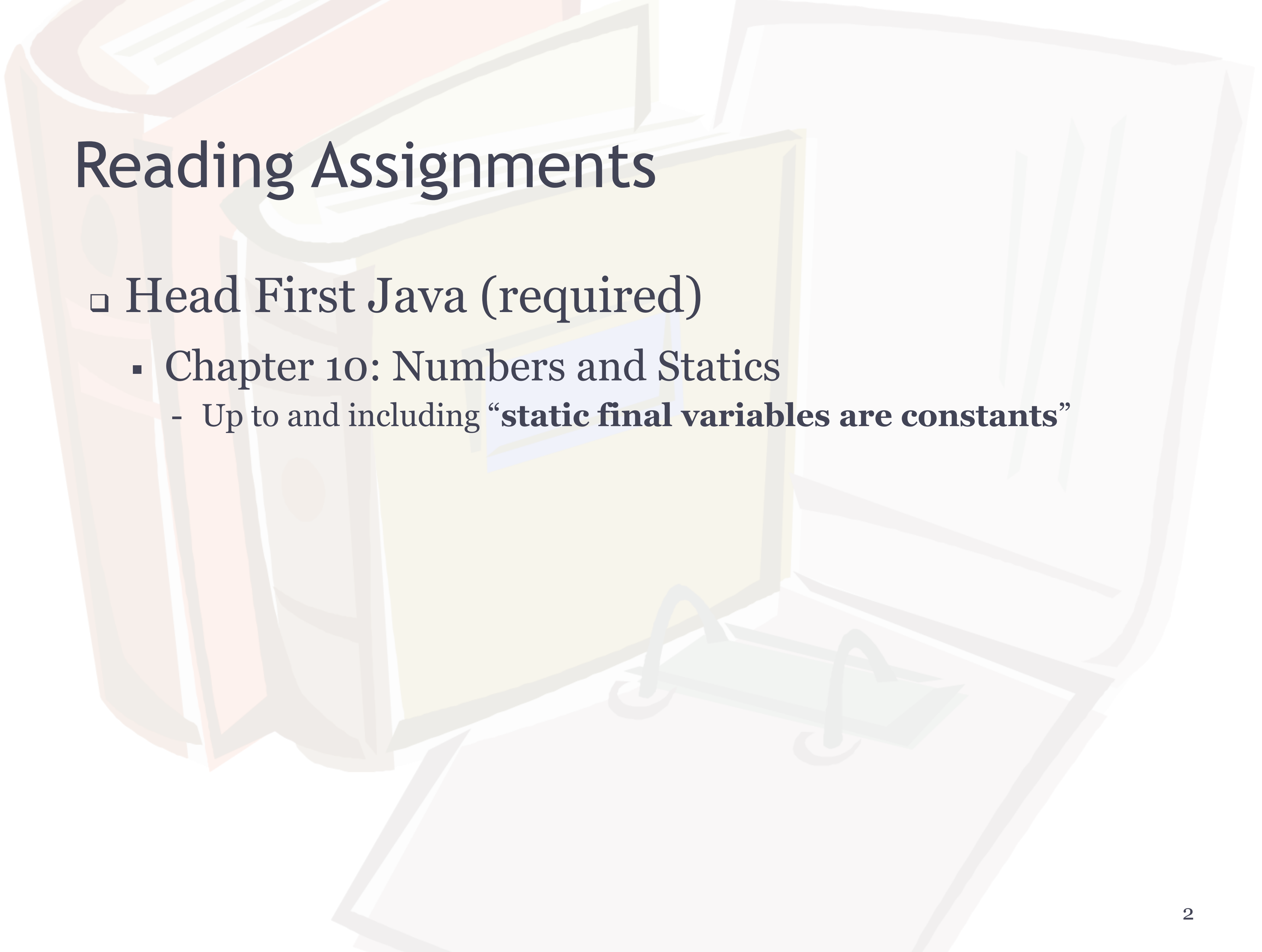
Object-Oriented Programming I

The ‘static’ keyword

Slides by Magdin Stoica
Updates by Georg Feil

Learning Outcomes

1. Explore the use and definition of static methods
2. Compare and contrast instance methods with static methods from both a syntax and semantic point of view
3. Explore the use and definition of static fields
4. Compare and contrast instance fields with static fields from both a syntax and semantic point of view



Reading Assignments

- Head First Java (required)
  - Chapter 10: Numbers and Statics
    - Up to and including "**static final variables are constants**"

The Static Keyword

(Finally !)

Static Methods

Methods that are shared by all objects of the same class type, and independent from any particular object (instance)

Defining a static method

- Static methods do not belong to a specific object, they are shared by all objects created from the same class
  - Static methods belong to the class, not the object
  - Static methods do NOT have access to normal field variables since field variables belong to specific objects, instances
  - Can't call a method that's not static directly
  - Static method DO have access to static field variables and static methods
- Static methods are declared using the keyword "static", which must follow the visibility modifier
  - public static void main(…) {…}
  - public static int getMaxGuess { return s_maxGuess; }
- Static methods are called using the class name not the object name since they do not belong to a specific object
  - GuessingGame.getMaxGuess(); // GuessingGame is the class name

Static methods

Method	Can Access normal	Can Access static
normal	yes	yes
static	no	yes

Examples from the Java library

- Math methods
  - Math.abs(-30);
  - Math.max(12, 35);
  - Math.sqrt(2.0);
  - Math.round(23.4);
  - See http://docs.oracle.com/javase/7/docs/api/java/lang/Math.html

- Parsing methods
  - int value = Integer.parseInt("123");
  - double salary = Double.parseDouble("123.45");
  - boolean playAgain = Boolean.parseBoolean("true");

Static vs. Instance Methods

Static Methods	Instance Methods
Declared with the keyword "static"	No special keyword is required to declared the method
Shared by all objects	Specific to each object
Invoked using <class name>.<method name>()	Invoked using <obj name>.<method name>()
Cannot access field variables	Can access field variables
Can access static field variables	Can access static field variables
Does not need an object to be created first	Needs the object to be created first before the method can be called

Static Fields

Field variables that are shared by all objects of the same class type, and independent from any particular object (instance)

Defining a static field

- Static fields do not belong to a specific object, they are shared by all objects created from the same class
  - Static fields belong to the class, not the object
- Again use the keyword "static", must follow the visibility modifier
  - public static final int MAX_GUESS = 11;
- Static fields are accessed using the class name not the object name since they do not belong to a specific object
  - GuessingGame.MAX_GUESS;   // GuessingGame is the class name
- You should avoid using static fields that are not 'final' unless there's a very good reason
  - Changing a field variable that's accessible by more than one object can lead to bad design and cause serious bugs that are hard to find

Example: static fields

Class declaration

Class definition

```
public class GuessingGame
{
    private static int _numCalls = 0;


    public void doSomething() {
        _numCalls++;
        …
    }
}
```

Makes the field shared between all GuessingGame objects

This variable is not 'final' so it can be changed by any class instance

Only field variables can be static!

Local variables cannot be static!

Instance vs. Static Fields

Static Fields	**Instance Fields**
❑ All instances of the same class share the same value	❑ Each class instance can have a different value (has own copy)
❑ Declared with keyword static	❑ No additional keyword
❑ <class name>.<field name>	❑ <object name>.<field name>
❑ Names are prefixed with "s_" by convention (sometimes)	❑ Names are prefixed with "_" (underscore) by convention
❑ Initialized by default to "zero"	❑ Initialized by default to "zero"
❑ Visibility modifiers allowed	❑ Visibility modifiers allowed
❑ *Used very rarely (except with 'final')*	❑ Used often as they provide the object's identity

Exercise

- Find your latest version of the Barking Dogs program
- Add a bark counter to the program which counts how many times any dog object “barked”
  - Print out the counter value just before the program ends