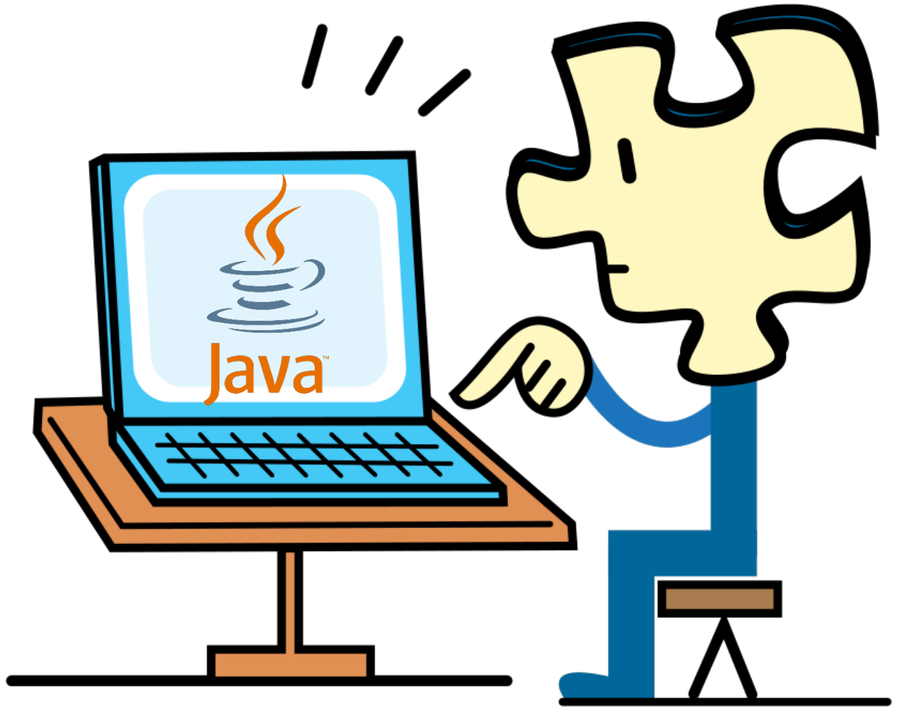


The Scanner Class

ICS 111

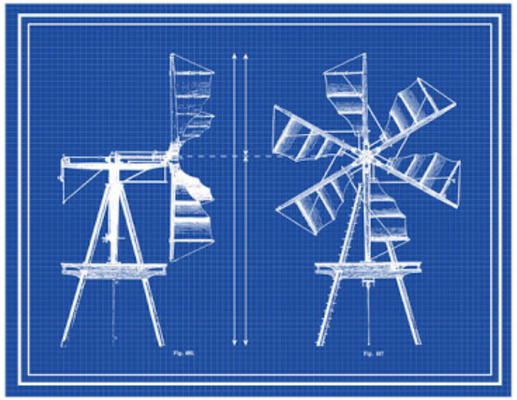
Ed Meyer

Last Time

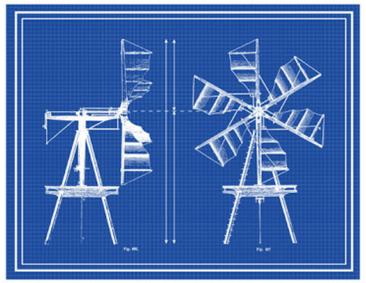
- Variables
  - Primitive Data Types
  - String object
- Arithmetic Expressions

Today

- Objects and importing classes
- Reading input from the user
- Incorporating user input into a program

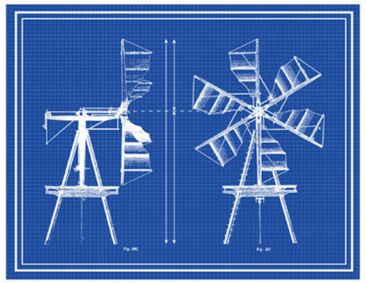


Objects and Importing Classes



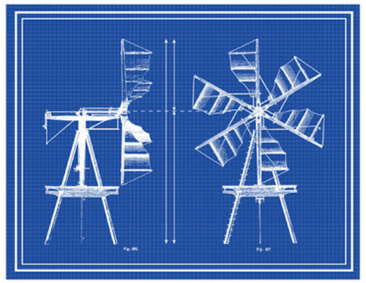
Objects

- To create an object in code, a class must be defined first



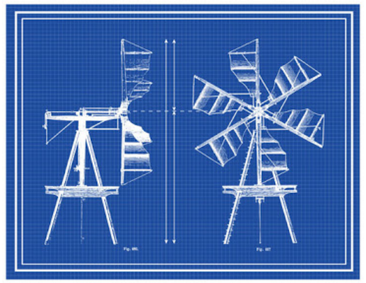
Objects

- To create an object in code, a class must be defined first
  - To create a String object, there needs to be a String class
  - To create a Scanner object, there needs to be a Scanner class



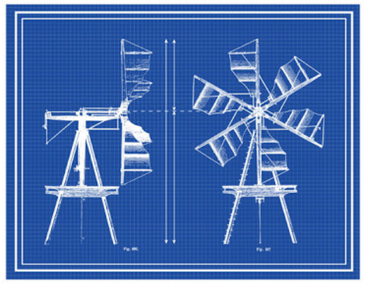
Objects

- To create an object in code, a class must be defined first
  - To create a String object, there needs to be a String class
  - To create a Scanner object, there needs to be a Scanner class
- A class is a blueprint of an object, it is an object's definition



Objects

- To create an object in code, a class must be defined first
  - To create a String object, there needs to be a String class
  - To create a Scanner object, there needs to be a Scanner class
- A class is a blueprint of an object, it is an object's definition
  - Once the blueprint is created, you can create many objects using that one blueprint

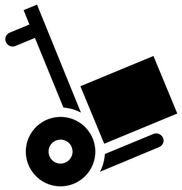


Objects

- Java has many predefined classes
- We can create objects from those predefined classes
- We can create our own custom classes and then create objects from those classes

Why Objects?

- We use objects to represent real-world things
  - String: Messages/sentences
  - Scanner: Getting input (the input device)
- Objects are more than just a value
  - Properties and methods
- An object is a variable that can store other variables



`import` Statements

- Use classes that are already defined to add functionality to your program
- Classes are organized into packages
  - A package is a group of related classes (think of a folder with a bunch of .class files)

The String Class

- To create Strings, we do not need to do anything special
- Strings are included in the main Java package
  - The Java language (`lang`) package
  - It is imported automatically
- `java.lang.String` is the package hierarchy for String

Using `import` Statements

- Use the reserved word `import` followed by the package hierarchy
  - `import java.lang.String;`
- Placed at the very top of your program
  - Above the program header
- Use the asterisk (*) to represent all classes in the package
  - Only use the * when using many classes from the package, otherwise, import specific classes

Using `import` Statements Example

```
import java.lang.*;

/**
 * Description and tags.
 */
public class ProgramName {
  public static void main(String[] args) {
    String pizzaTopping = "cheese";
  }
}
```

Using `import` Statements Example

```
import java.lang.*;

/**
 * Description and tags.
 */
public class ProgramName {
  public static void main(String[] args) {
    String pizzaTopping = "cheese";
  }
}
```

This is not needed.

java.lang.* is automatically imported when creating a new Java file.

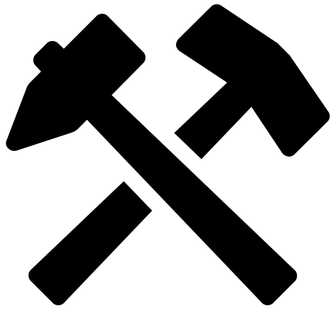
Reading User Input

Why Read from the User?

- Instead of hardcoding values, ask the user to give us some input.

How do we ask?

- The Scanner class
  - In the Java utilities (`util`) package
- The package hierarchy for Scanner is `java.util.Scanner`



The Scanner Class

- Create Scanner objects
- Used to read
  - Strings
  - Primitive Data Types
- From
  - The keyboard
  - Files
  - Strings
- Scanner Java API

Preparing to Read

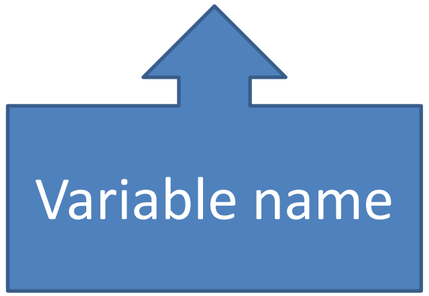
```
import java.util.Scanner;

public class ReadAndPrintInput {
  public static void main(String[] args) {
    Scanner reader = new Scanner(System.in);
  }
}
```

Creating a Scanner Object

```
Scanner reader = new Scanner(System.in);
```

Creating a Scanner Object

```
Scanner reader = new Scanner(System.in);
```



Creating a Scanner Object

```
Scanner reader = new Scanner(System.in);
```

Notice `Scanner` appears twice.

Creating a Scanner Object

```
Scanner reader = new Scanner(System.in);
```

Declares the variable type.

Constructor

Creating a Scanner Object

```
Scanner reader = new Scanner(System.in);
```

`new` is used to allocate new memory for the object data.

Creating a Scanner Object

```
Scanner reader = new Scanner(System.in);
```

Specifies where to get the input.
`System.in` is the keyboard.

`System.in`

- Represents standard input
  - The keyboard
- It is one possible argument for the `Scanner` constructor
- The opposite of `System.out`

Declaring and Initializing a `String`

- Declaring and initializing a String variable using object syntax would look like this:

```
String message = new String("I choose you, Pikachu!");
```

- Initializing an object is called "instantiating a class"
  - An instance of a class is an object.

Using a Scanner Object

```
Scanner reader = new Scanner(System.in);
```

- So now, we have this Scanner object called `reader`, how do we use it to get input from the keyboard?

`Scanner` Methods

- Methods are what you can do with the object
  - What action you can perform
- `.nextLine()` method
  - Reads an entire line as a String up to a newline
  - Does not take any arguments
    - Nothing is expected between the ( )
- Method names are lower camel case

The dot (`.`) Operator

- The dot (`.`) in `.nextLine()` is used for accessing the object's properties and methods
  - A property is information about the object itself

Other `Scanner` Methods

- What kind of data is being read in?
  - `.nextDouble()` method
    - Reads input as a double
  - `.nextInt()` method
    - Reads input as an int
  - `.next()` method
    - Reads a single word as a String

Other `Scanner` Methods

- What kind of data is being read in?
  - `.nextDouble()` method
    - Reads input as a double
  - `.nextInt()` method
    - Reads input as an int
  - `.next()` method
    - Reads a single word as a String
- When asking the user for input, you (the programmer) must think of what type of data it is

Reading

```
import java.util.Scanner;

public class ReadAndPrintInput {
  public static void main(String[] args) {
    String inputName = "";
    Scanner reader = new Scanner(System.in);

    System.out.print("Enter your name: ");
    inputName = reader.nextLine();
  }
}
```

`inputName = reader.nextLine();`

- This is an assignment statement
  - Assign `inputName` the value of `reader.nextLine();`
- `reader` is my `Scanner` object
  - As a `Scanner` object it has the `.nextLine()` method

Printing the Input

```
import java.util.Scanner;

public class ReadAndPrintInput {
  public static void main(String[] args) {
    String inputName = "";
    Scanner reader = new Scanner(System.in);

    System.out.print("Enter your name: ");
    inputName = reader.nextLine();

    System.out.println("Your name is: " + inputName);
  }
}
```

Let's do an example!

`LocalKineMadLibScanner.java`