The class, the object, static & instance fields and methods

In the previous video, I talked about local variables and scope.

Local variables are a way to store and manipulate temporary data.

In addition to local variables, we can set up data to be defined and used as part of a class or an object.

I'll be discussing these concepts now at a cursory level for several reasons.

The class, the object, static & instance fields and methods

First, attributes on classes are another way to store data.

Second, I want to introduce you to some static methods on the wrapper classes, which are classes we previously looked at. We haven't used any methods on these classes yet.

These methods will help parse strings into numeric values.

And finally, I want to introduce you to a special class for reading input, which I'll be using in the last part of this section to create an interactive program.

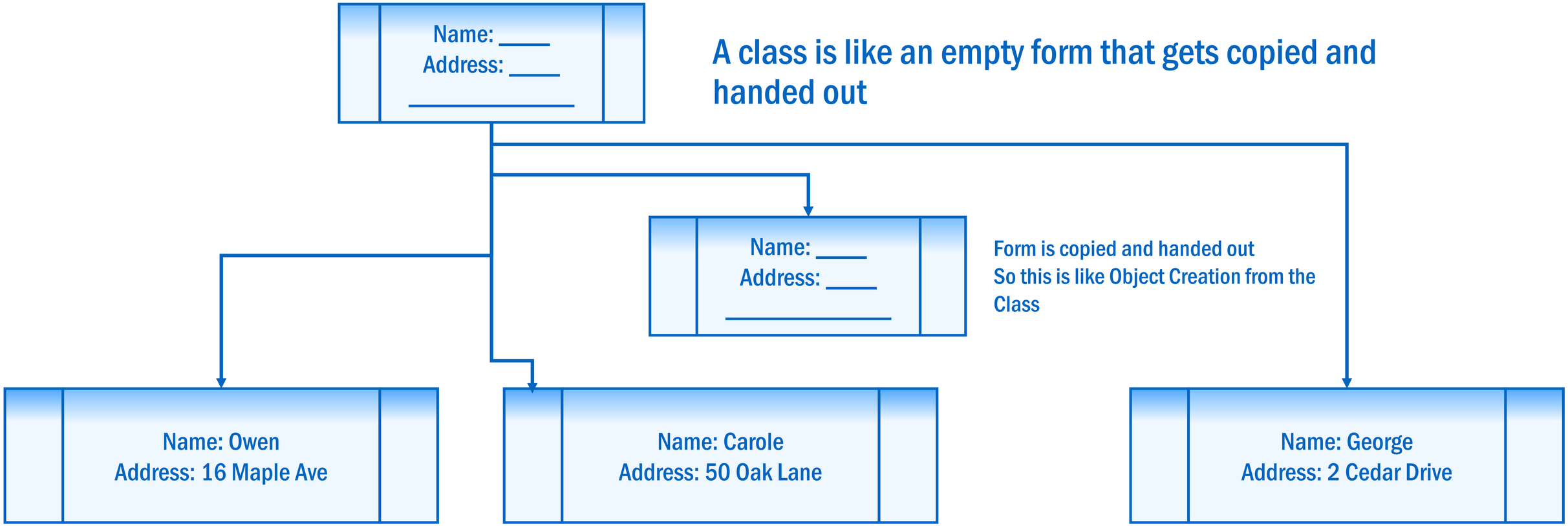
Before we use that class, it will help if we understand some very basic concepts with classes.

A class

A class can be described as:

- a custom data type.
- a special code block that contains methods.

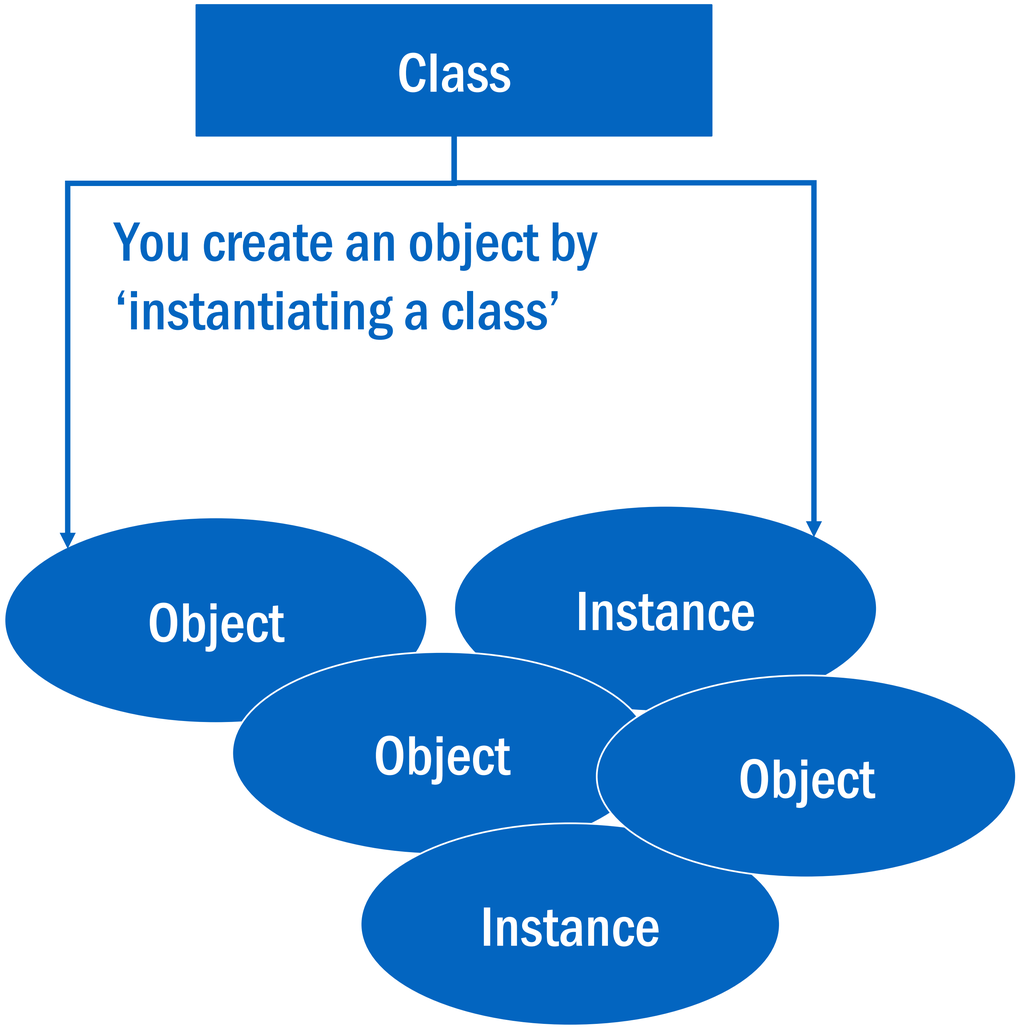
The class is a template for creating objects



An Object

An object is called an instance of a particular class.

A class and objects



You can create many objects using a single class. Each may have unique attributes or values

Object and instance are interchangeable terms

Declaring and instantiating a new object from a Class

The most common way to create an object is to use the new keyword.

The new keyword creates an instance of a class, and you can optionally pass data when creating that instance to set up data on that object.

Looking at the String, it's actually a class. But it holds a special place in the Java language, because we can create a String just by using a literal which we've seen.

```
String s = "Hello";
```

But we could also use new.

```
String s = new String("Hello");
```

static and instance fields

Static Field	Instance Field
Requires ‘static’ keyword when declared on the class.	Omits ‘static’ keyword when declared on the class.
Value of the field is stored in special memory location and only in one place.	Value of the field is not allocated any memory and has no value until the object is created.
Value is accessed by ClassName.fieldname Example: Integer.MAX_VALUE	Value is accessed by ObjectVariable.fieldname Example myObject.myFieldName (myObject is our variable name for an object we create and myFieldName is an attribute on the class.

static and instance methods

Static Method	Instance Method
Requires ‘static’ keyword when declared on the class.	Omits ‘static’ keyword when declared on the class.
Method is accessed by ClassName.methodName Example: `Integer.parseInt("123");` A method called parseInt is called directly from the Class, Integer.	Method is accessed by ObjectVariable.methodName Example: `"hello".toUpperCase();` A method called toUpperCase is called on the instance of a String with value “hello”.