- In C#, there is one type of reference that can be used for all objects. It is aptly called Object.  
- Object is at the top of any class’ [inheritance](https://www.codecademy.com/resources/docs/c-sharp/inheritance) hierarchy.  
- Since references can be upcast to any type in its inheritance hierarchy, then all types can be referenced as Objects:

A computer code with text

Description automatically generated with medium confidence

- If that’s so, why not use Object references for everything? Because the functionality of an object is limited by its reference type.   
- We lose all of a type’s specific functionality when we reference it as an Object type.   
- We would also lose the automatic type-checking that helps prevent type errors.

- When we do use them, Object references can be very useful! For example, if we’re not sure what type a variable is, we can safely store it as an Object.   
- We can also assume that any object has access to the standard Object members for basic manipulation.

A diagram of different types of objects

Description automatically generated

**The Object Type:**

- Every class derives from Object — even [classes](https://www.codecademy.com/resources/docs/c-sharp/classes) that we define ourselves!  
- When you create a class, C# implicitly makes it inherit Object. So when we write this code:



- C# assumes we mean:

A close up of a text

Description automatically generated

**Object Members:**

- Object has several useful members that, by the nature of [inheritance](https://www.codecademy.com/resources/docs/c-sharp/inheritance), are accessible by every type. Here are some important ones:  
- **Equals(Object**) - returns true if the current instance and the argument are equal (using value equality for value types and referential equality for reference types)  
- **GetType()** - returns the type of the object  
- **ToString()** - returns a string describing the object when a non-string object is printed to the console with Console.WriteLine(),

A screen shot of a computer code

Description automatically generated

- Remember that we can access inherited members from a derived class. In this case, every type inherits from Object, so every type can access these members!

**Overriding Object Members:**

- The Equals() and ToString() [methods](https://www.codecademy.com/resources/docs/c-sharp/methods) in Object are virtual, so they can be overridden.

A computer screen shot of text

Description automatically generated

- Now any Novel instance will use this version of the method:

A screen shot of a computer code

Description automatically generated

- You can override ToString() in your class to return a string that represents the object’s properties in a more readable way. This is especially useful when you want to easily display or log the state of an object.

A computer screen with colorful text

Description automatically generated