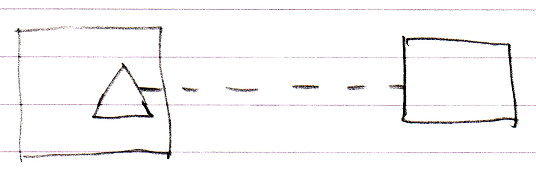
|  |
| --- |
| Circle Language Spec: Inheritance |

## Miscellaneous Inheritance Situations

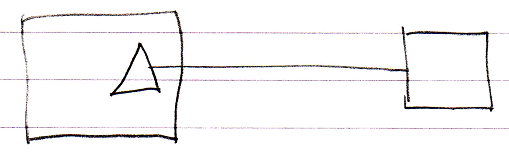
This article shows a couple of miscellaneous situations that can occur when you work with inheritance.

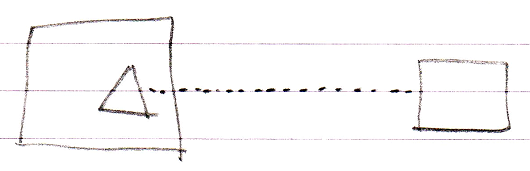
### Command Inheritance

Because commands are just objects, inheritance can also be used between commands. One command can class-inherit from another command:

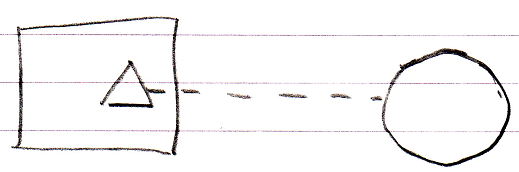


Object and interface inheritance are also possible for commands:



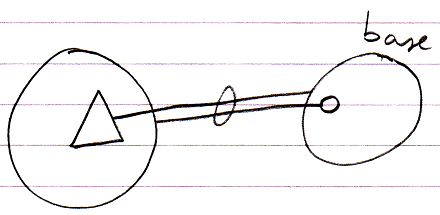


Since commands are just objects that can be executed, a command could in theory also inherit from an object:

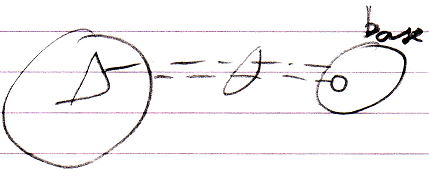


### Backward Relation to Derived Classes

Due to the duality of relationships, when a derived class refers to its base class, the base class could also point back at the derived class again.

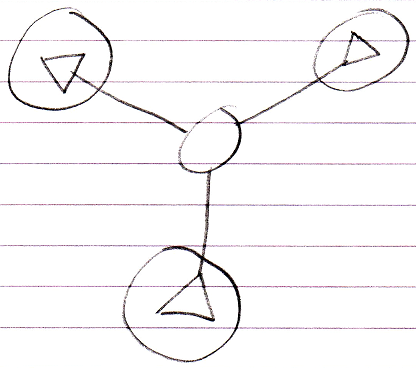


Or for class inheritance:



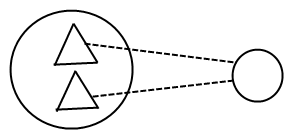
### Multiple Objects Sharing One Base Object

Because multiple objects can inherit from the *same instance* of an object, you can have a set of objects use another object as their base:



### Inheriting Multiply from the Same Class

Because you can give an object two sub-objects of the same class, you can also let a single object inherit from the class class twice:



But why would you do that? It does not matter, that’s up to you. You’re the programmer. You can reference the double members individually in the diagram, but when you want to reference a double member textually you have to disambiguate with a qualifier, which does require you to give each base its own name.