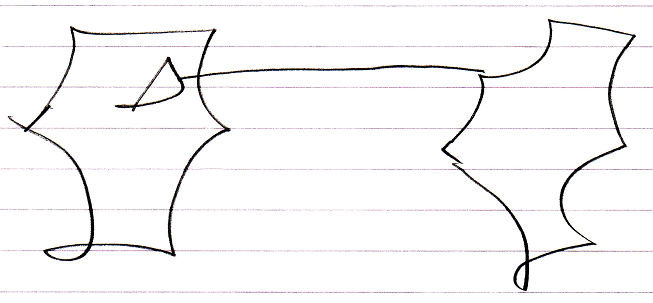
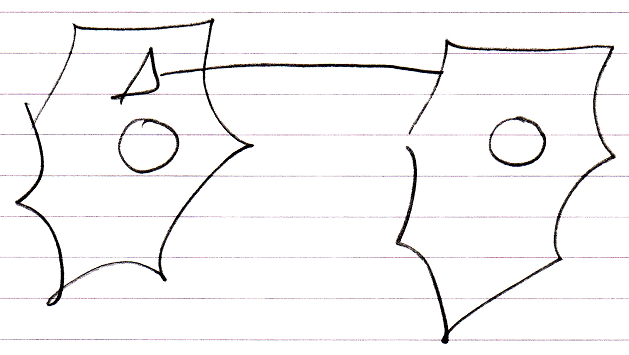
|  |
| --- |
| Circle Language Spec: Inheritance |

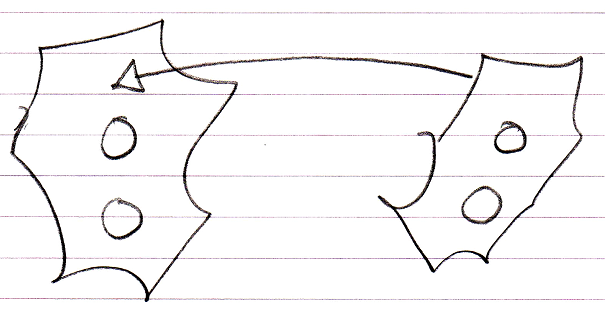
## List Inheritance

List inheritance is a special form of inheritance where one list inherits from another.

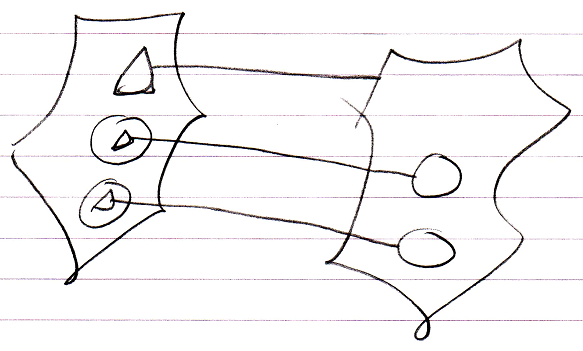


What is special about it, is that when an object is added to the base list, an object is also added to the derived list.





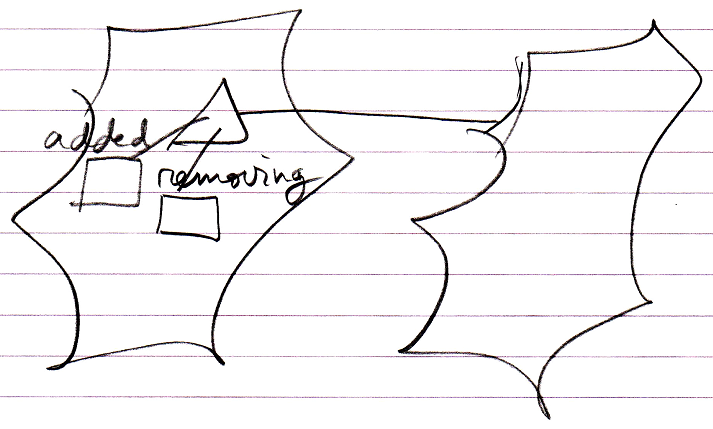
And the list item added to the derived list will inherit directly from the list item in the base list.



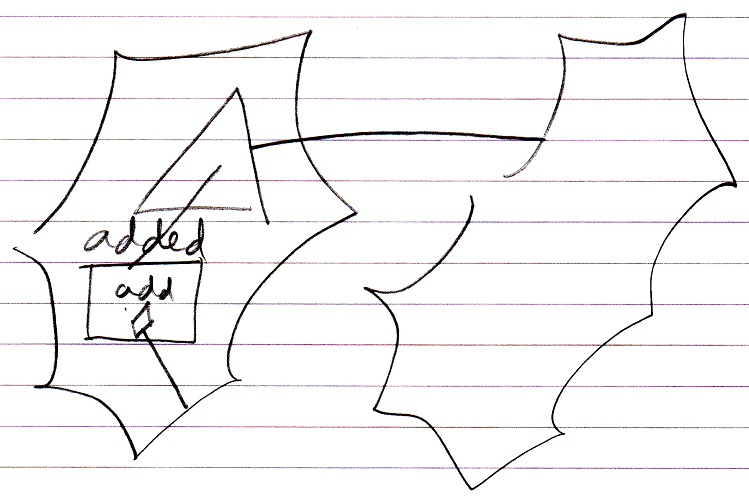
List inheritance is analogus to defining a *parallel list*, adding extra columns to an existing list.

This behavior is specific to list inheritance and does not apply to any of the other forms of inheritance.

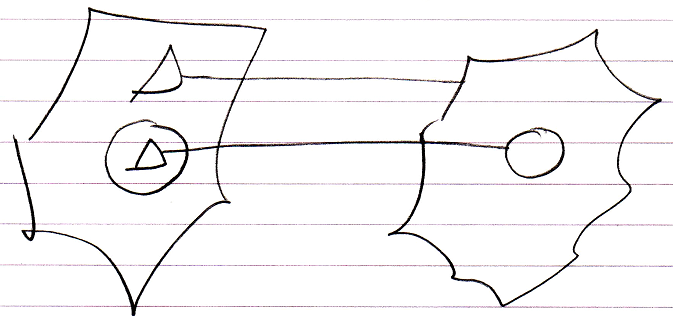
The list inheritance construct realizes such behavior by responding to the List events of the base list: the Added and Removing events. These events are *system events*, that were introduced by the *Events* chapter.



These events make sure that what happened in the base list, will also happen in the derived list.

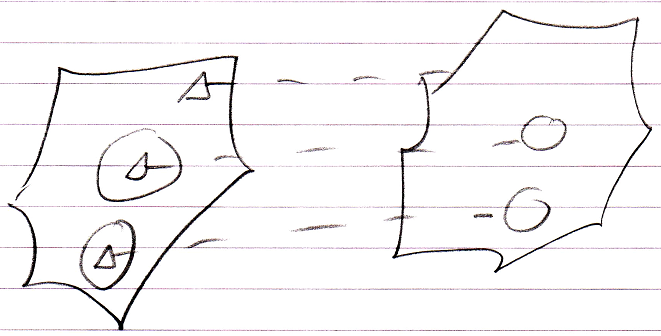


And on top of that, the derived list item will be linked to the base list item through inheritance.



List inheritance also makes a distinction between object inheritance and class inheritance. The default is object inheritance, which causes the behavior you are most likely to expect.

But *class* inheritance is also possible for lists, in case of which the lists items will be linked with class inheritance:



This behavior is not as intuitive as object list inheritance.

For an explanation about how different list objects themselves, such as linked list and stack, can inherit from eachother, see the article *List Concept*. The main idea is that you would have to open up the system interface for that.

The following might be something you could get confused about. A normal object is also a *list* of members. However one object’s inheriting from another object does not cause the same behavior as list inheritance: extra members are not added to the derived object when members are added to the base object. So do not get confused over this.

