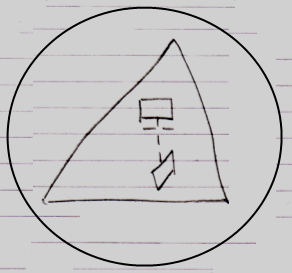
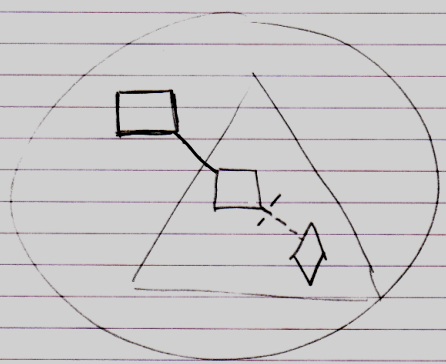
|  |
| --- |
| Circle Language Spec: Inheritance |

## Overriding

Base classes normally call their own commands:



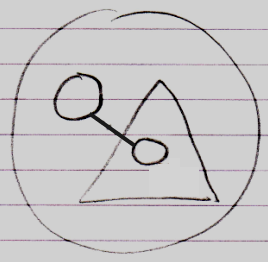
But derived classes can redefine those commands. When the base class tries to call its own command, the call is detoured to the redefined command. In other words: the original call to the command is detoured to another command.



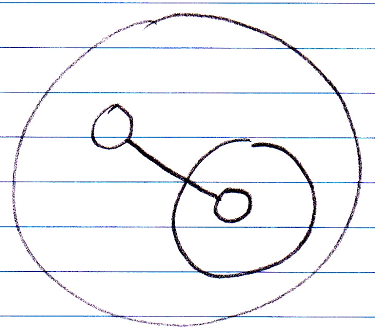
The base class can not run its original command anymore, but the command is detoured to a command defined in the derived class.

Overriding is accomplished with simple object redirection.

What should be mentioned is that you can also override objects, just like you can override commands:

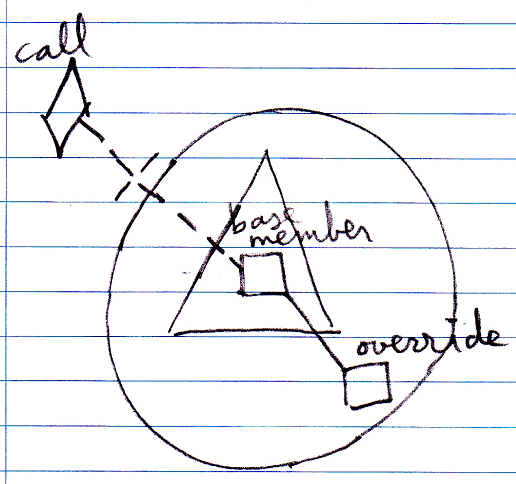


What should also be mentioned is that overriding is not specific to inheritance. It can also be done in case of regular containment:

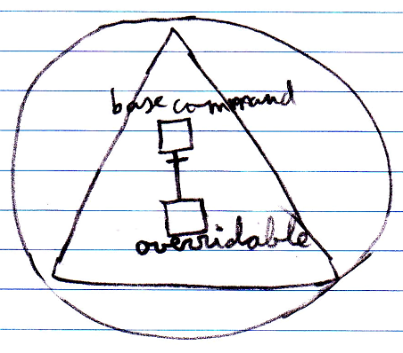


### Calling Base Class’s Version of Overridden Member

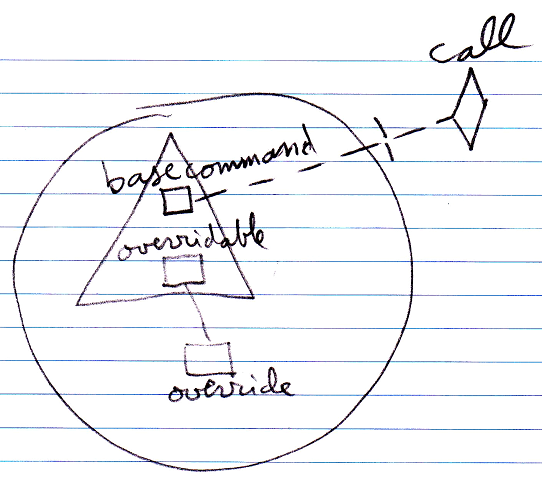
Some programming languages offer a way to call the member defined by the base class, even when the base member was overridden. In the new computer language this is not possible by default, because any call to the overridden member will redirect to the overriding member.



However, there is an alternative: define a base command and create an overridable command that is a reference to the base command.

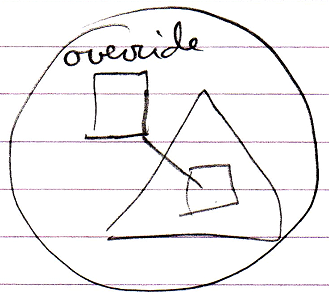


You can replace the overridable command, but you can still call the base command.



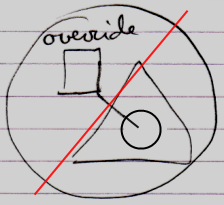
### Overriding Alternative: Event Notation

An alternative notation to overriding is the override event.



An override event is basically unnecessary, because you can already accomplish that by simple object redirection, but the notation is intuitive, and creates an analogy with the notation for pre- and post-extension, which is introduced later. The notation also offers the possibility to easily express overriding *system* commands, also explained later.

The notation does not seem to work for object overriding.

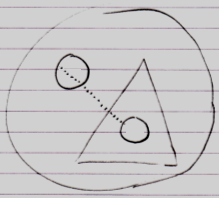


This is because object overriding redirects to an object, and since events are commands, it looks strange.

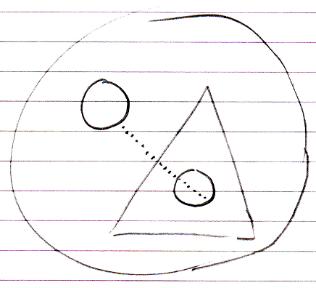
### Discarded Intrinsic Overriding Notation

There used to be an intrinsic overriding notation which looked a lot like shadowing:

*Shadowing notation:*



*Intrinsic overriding notation:*



However, the intrinsic overriding notation was discarded, because it is undefendable to have it when the same is possible with simple object redirection.