|  |
| --- |
| Circle Language Spec: Black Boxes |

## Object-Bound, Reference-Bound & Access Control

For the basics: see the article *Object-Bound & Reference-Bound* in the *System Objects* article group.

#### Key Example: Value Get

If the Value aspect were reference-bound and made read-only by a reference and you could still Get the Object, you could assign the Object to another reference and write the Value there. Read-only protecting the Value aspect would be non-waterproof and more like a *suggested* access control, rather than *real* access control. By realizing that the Value aspect is object-bound, you can not make a read-only Value aspect all of a sudden writable through another reference.

Reference-related aspects only count for a single reference and not for the whole object, and can be freely overridden by another reference. Object-bound aspects count for all references and can not be overridden by another reference.

By binding some aspects to the object and some aspects to a reference, each aspect can be safely access controlled without any loop-holes through other references.

#### Another Example: Use As Class

Use As Class is basically an Object Get, access-controlled for the specific purpose of using the object as a class. By making Use As Class object-bound it is possible to design the system so that it indeed is not possible to use the object as the source in a class assignment. So things like Use As Class can also be made safe now they are object-bound. When Object Get is Public, Use As Class can still be Private.

#### Object-Bound Access Control

Object-bound access control is visible through each reference and can be controlled through each reference, but object-bound aspects always apply to the whole object.

Object-bound aspects and reference bound-aspects do not require two separate friend declarations. With one friend declaration the Friend can access the object-bound aspects of the object, and access the reference-bound aspects of its sub-objects. It is like a combo-deal.

#### Reference can Access Control Object-Bound

Even though object-bound aspects are primarily access-controlled by the object, a reference can impose additional access control over them. So object-bound aspects can be access controlled by a parent to further protect what the object can already protect. But that is part of the *Deeper Exclusion* concept, explained elsewhere.

#### Object has Access to its Privates

The parent object automatically has friend access to its own object-bound aspects and to the reference-bound aspects of its sub-objects. That is logical: the object automatically has access to its own privates. The access connectors express potential connections that can be made by other objects and are displayed on the outside of the object.

(But compared to the traditional approach: when an object has a command (a method), the command must be made friend in order to access the privates of the object.)