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

## Friend Declarations in Instances

The points made in the section *Notations of Private* are demonstrated here again in a different situation, where objects have external definitions. To summarize the rules:

- Privates visible in definitions

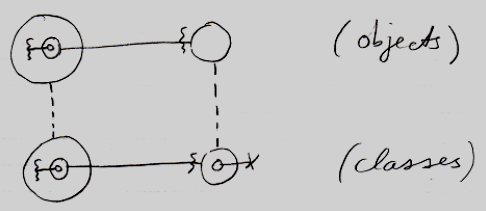
- Privates invisible in normal references

- Privates visible in friend references

In this section of the documentation it is demostrated how this looks if objects have an external class, as opposed to defining their own contents and behavior.

The main thing clarified here is: “when do you show privates and when do you not”. The same rules apply here as in traditional text code, but now they are demonstrated in a diagram.

#### In a Unary Relation

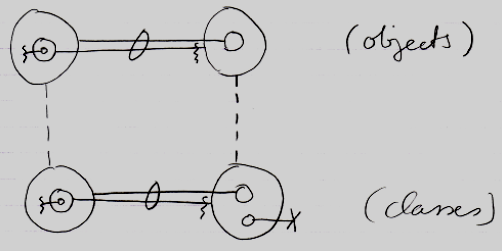


The classes show privates, but the objects do not, unless they are Friend.

The effect that you see, is that wavy access connectors stay there in the objects, while cross connectors disappear in the objects. So the main rule is: crosses disappear, wavy lines stay visible.

#### In a Dual Relation

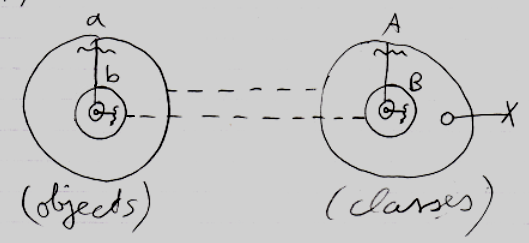
The main point to the diagram below is that the classes show privates, but the objects do not, unless they are Friend. It shows the situation in case of a dual relation.



The effect that you see, is that wavy access connectors stay there in the objects, while cross connectors disappear in the objects.

#### Container is Friend

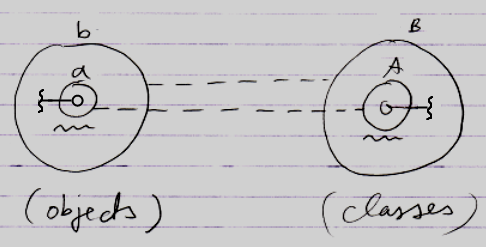
The main point to the diagram below is that the classes show privates, but the objects do not, unless they are Friend. It shows the situation where the *container* is Friend.



The effect that you see, is that wavy connectors stay there in the objects, while cross connectors disappear in the objects.

#### The Contained is Friend

The main point to the diagram below is that the classes show privates, but the objects do not, unless they are Friend. It shows the situation where the *contained* is Friend.



Here something different is going on in the notation. Earlier it was stated that: crosses disappear, wavy lines stay visible. But the picture above contains no crosses at all.

In the classes on the right, something unusual is going on.

The problem is, that *A* is both a definition and a reference at the same time.

A is:

- a definition

and also:

- B's (Friend) reference to A.

In a definition it should be a cross. In a Friend reference it should be a wavy line.

To express the member’s accessibility, the choice was made to display it with a wavy line and not with a cross.

This is simply a side-effect of the notation where *the contained is Friend*.