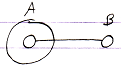
|  |
| --- |
| Circle Language Spec: Black Boxes |

## Friend Declaration

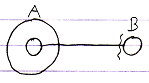
This article covers friend declaration in several situations.

#### Friend Declaration, in a Unary Relation

The following is a unary relation:



In the following picture object A is declared *friend* of object B:



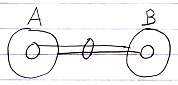
Object A can now access anything private of object B through the reference contained inside object A.

It is object B who declares itself Friend, so object B gives access to its private members.

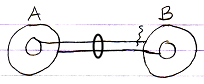
But the friend declaration is part of **A**’s reference to B, because it makes the reference *special*, using the friend access symbol.

#### Friend Declaration, in a Dual Relation

The following picture expresses a dual relation:



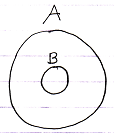
In the following picture B is declared friend of A:



A can access everything of B, even the Private stuff. It does not go both ways: B can only access the Public contents of A.

#### Friend Declaration, Container is Friend

In the following picture, a related object is only displayed as contained inside another object:



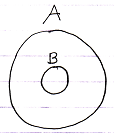
To declare A as friend of B you can use the following notation:



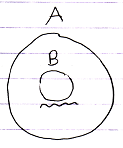
Now B can access everything of A.

#### Friend Declaration, the Contained is Friend

In the following picuture a related object is only displayed as the container of another object:



To declare B as friend of A you can use the following notation:

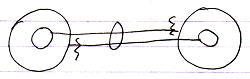


A can access anything of B.

There is no line to place the friendship declaration on. The relation is implied by the fact that something is contained inside something else.

#### Mutual Friendship Referential Notation

Mutual friendship can be expressed as follows in a referential notation:



#### Mutual Friendship Containment Notation

Mutual friendship in a containment notation would look as follows:

