|  |
| --- |
| Circle Language Spec: System Objects |

## System Interfaces

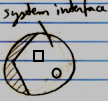
### Main Concept

#### System Interface Notation

Usually you see an object’s members:



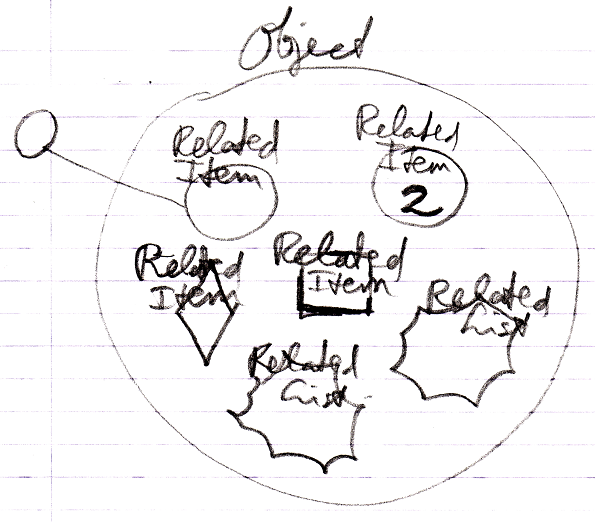
But all objects, references and lists are based on system objects. You can also show the members of the system object instead. You can do that by breaking open the inner workings of the objects and show the *system interface*:



When the system interface is shown, the normal members are not visible anymore.

#### Example: System Interface of an Object

An Object might normally look like this:



But if you open up an Object's system interface, then the related items and related lists are shown as the Related Items & Related Lists collections:



The Related Items and Related Lists are now displayed circles: normal objects, even when they are *commands* and even when they are *lists*. This only represents the exact way an Object actually internally works.

#### Aspects

Next to controlling *sub-objects*, the system interface also controls *aspects*, for instance, the Value aspect:



#### System Interface = Publics of System Objects

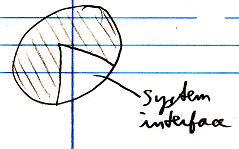
Everything about system interfaces is based on the fact that the system interface is nothing more than the public members of a system object. Even when the explanation about system interfaces is incomplete, what is missing can be derived from the fact, that the system interface will simply show all the public members of a system object. You should also realize that the *private* workings of system objects are not shown in the system interface.

#### Origin of the System Interface Notation

The notation of the *system interface* is derived from the *interface* notation. When an object has several interfaces, each interface is shown as a triangle inside the object:

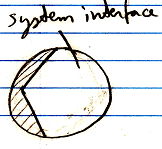


But showing the system interface is like showing the interior ‘of the symbol itself’. Therefore, the triangle of the system interface is now stuck to the border of the symbol:



That way the interface is more part of the object itself, rather than being a sub-object.

To make more room inside the system interface the following, exaggerated notation is used:



It also looks as if you have broken open the shell of the symbol to see its inner workings. It looks like you have opened up the system and can see its internal wiring. The system interface is like seeing the setup of the symbol machine.