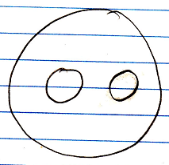
|  |
| --- |
| Circle Language Spec: System Objects |

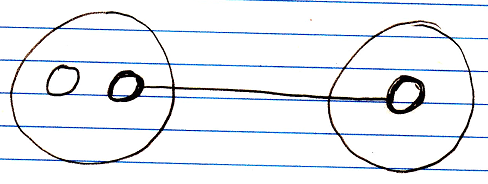
## Objects Floating Around

Objects are never directly accessed. They are always floating around somewhere you cannot touch. You are always accessing an object through an *object reference*. You are always dealing with *references* to objects, never with the object directly.



The smaller, contained circles are *references* to objects, even though the bigger circle seems to be the sole container of the objects themselves. An object does not really contain sub-objects. An object contains pointers to its sub-objects. Even when the object seems the sole container of the other objects, the other objects are really only referenced. You do not see the actual object. You are only seeing references to it.

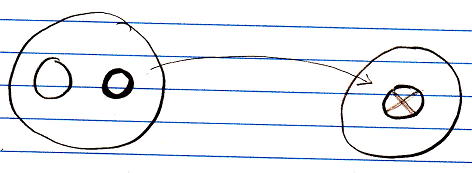
Another symbol can start referring to the same object, making the object all of a sudden not part of a unique container anymore.

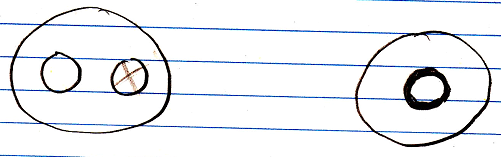


When you annul the object reference in the original parent, the second parent all of a sudden becomes the sole container of the object.



The object has moved from one parent to another.





Objects are always just floating around like that. They do not have to stay in a fixed spot.

In reality the objects do not move at all. They are physically stored in the same spot all the time, no matter where they appear to be. An object can just be freely referenced from anywhere, because objects are always accessed through references.

Even when you *create* an object, you are not directly in touch with the object. The object is immediately assigned to an object reference. Also: when you assign a *value* to an object, you do not assign the value directly to the object, but you assign it through an object reference.

Each object reference gets its own identifier, even when an object reference is Nothing. An object itself, does not have an identifier. An object can be given a Name attribute, though.