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

## Black Box Introduction

The concept of *Black Box* is also known as the concept of *Public and Private*. *Private* makes something only accessible inside the object. *Public* makes something accessible outside the object. That way, the inner workings of an object are hidden away and you only work with the input and output.

The code generator version of the the computer language (version 0.9) simply borrowed the public and private concept of the target programming language for which code was generated. The *generic module* version of the new computer language (version 2.0) did not have any black boxing yet at all. The black box implementation of the version to come is worked out in this documentation folder.

The main articles about black boxing are finished, but you will also find a lot of unfinished material at the end of the documentation, which might be worked out later. Nevertheless, the main idea is worked out completely, and it is clearly denoted which material is not finished.

The idea for the future is that *user* access control and *black boxing* access control will become a single concept of ‘what has access to what’, but that will not immediately be done at first. A simpler black box method is worked out first: one much like what you see in contemporary programming languages. However, this method is adapted to the new computer language, because the new computer language has a different view on objects and commands. Which commands get access to the private contents of which objects is less obvious, because a command is not part of a specific object or class. In fact, no command gets access to the private contents of an object, unless that command is a *friend* of the object. This creates a tighter bond between the command and the object. Another thing, that is different in the new computer language, is that commands are no more than objects. They are the same thing. This really impacts the way you think about black boxing.