

LValue

An lvalue is a syntax element that represents a target that can be assigned values. The syntax of an lvalue is a member-path: A sequence of identifiers separated by dots. Example: 'a.foo.bar.baz'. The first identifier must be a mutable [local variable](#) (defined with the `mut` keyword). Each subsequent identifier must be a member of the type of the previous identifier, assuming the previous identifier is of a [struct](#) type.

Mutability in an immutable world

A common misconception about assigning to lvalues is that it somehow changes the memory. However, Cairo's memory model is immutable. Values are immutable, and cannot be changed. However, variables can refer to different values. A variable does not represent a single memory target. Instead, it is a logical placeholder for some memory target. The meaning of assignment is that the variable now refers to a different memory target.

Member borrowing

Assigning to a member path (e.g. `a.b`) is possible since the compiler keeps track of the individual member values. Not only variables are now logical placeholders, but also members.

[6.2 Assignment statement](#) [6.4 Item statement](#) ð§