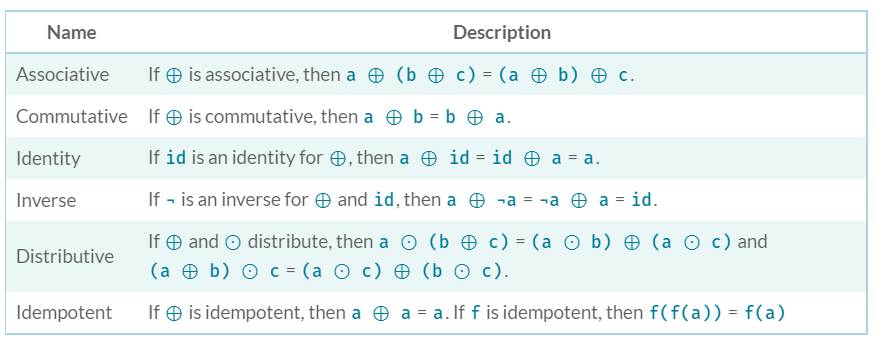
**ALGEBRAIC LAWS**

Cats uses type classes to represent algebraic structures, each representing abstract capabilities (and requirements). We then implement these generic / abstract type classes with the non-generic types that we want to implement these methods with…

**ALGEBRAIC PROPERTIES:**



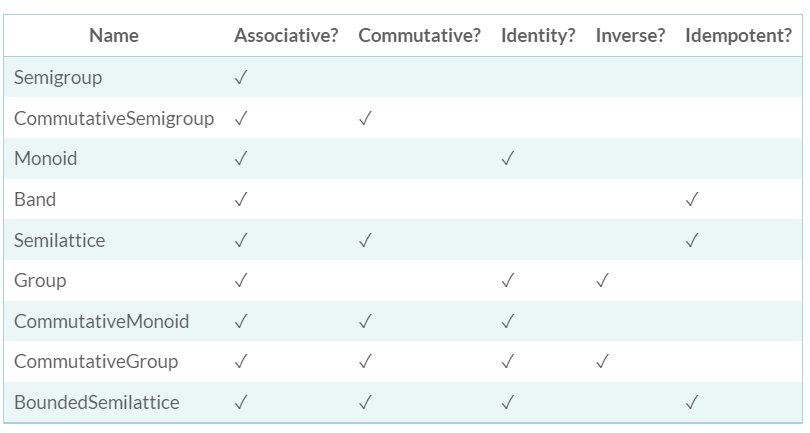
Where:

* a, b & c are values of the same type (generic type A)
* ⊕ is a function, equivalent to func() -> a . func(b) or func(a, b)
* ¬a is equivalent to g(a)  
   ¬ is a prefix representation of the unary function g

Note - for the identity property:

* It is used as the 1st argument (the initial / base value for the accumulator) in foldLeft, foldRight and recursive functions for an object.
* Examples of what identity value for different types:
  + List -> empty List
  + Strings -> “”
  + Numbers being added/summed or subtracted -> 0
  + Numbers being multiplied or divided -> 1

**BASIC ALGEBRAIC STRUCTURES:**



These are the structures found in the ***algebra*** package

All implement a combine method, which is associative.  
The identity element (if present) will be called empty.  
The inverse method (if present) will be called inverse.