Traverse

Traverse

Higher-kinded TC with "inside-out" functions

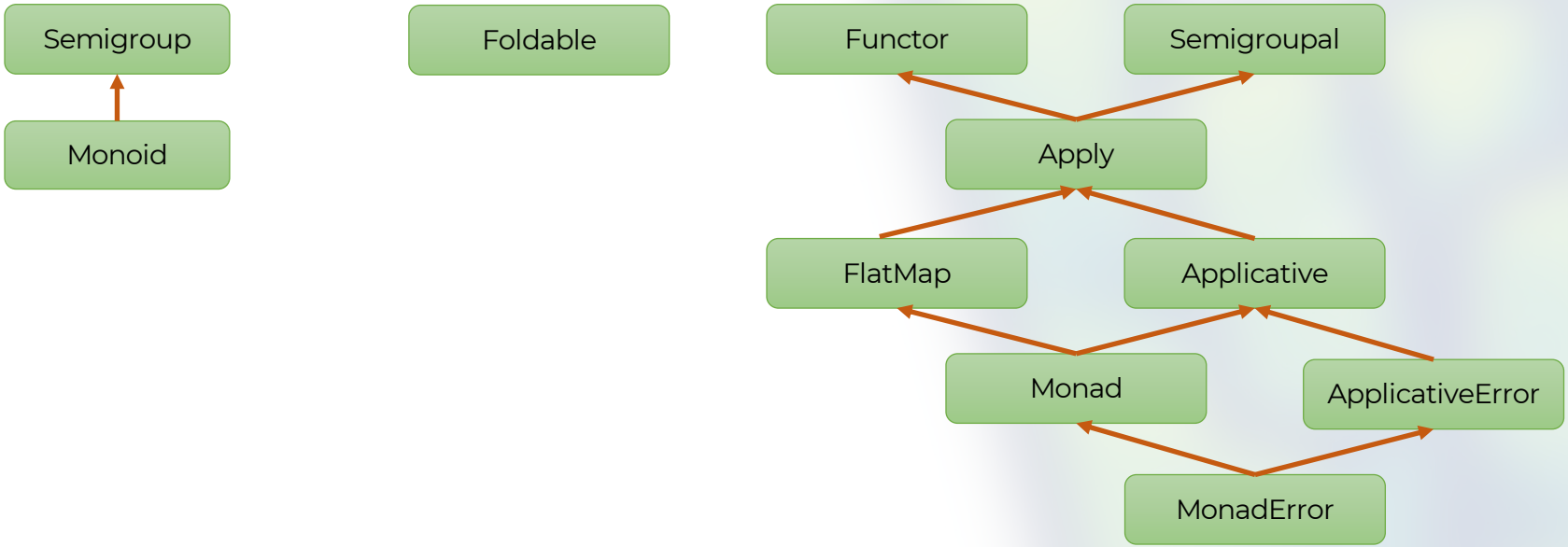
```
def traverse[F[_] : Applicative, A, B](container: L[A])(func: A => F[B]): F[L[B]]
def sequence[F[_] : Applicative, A](container: L[F[A]]): F[L[A]] =
  traverse(container)(identity)
```

Useful for

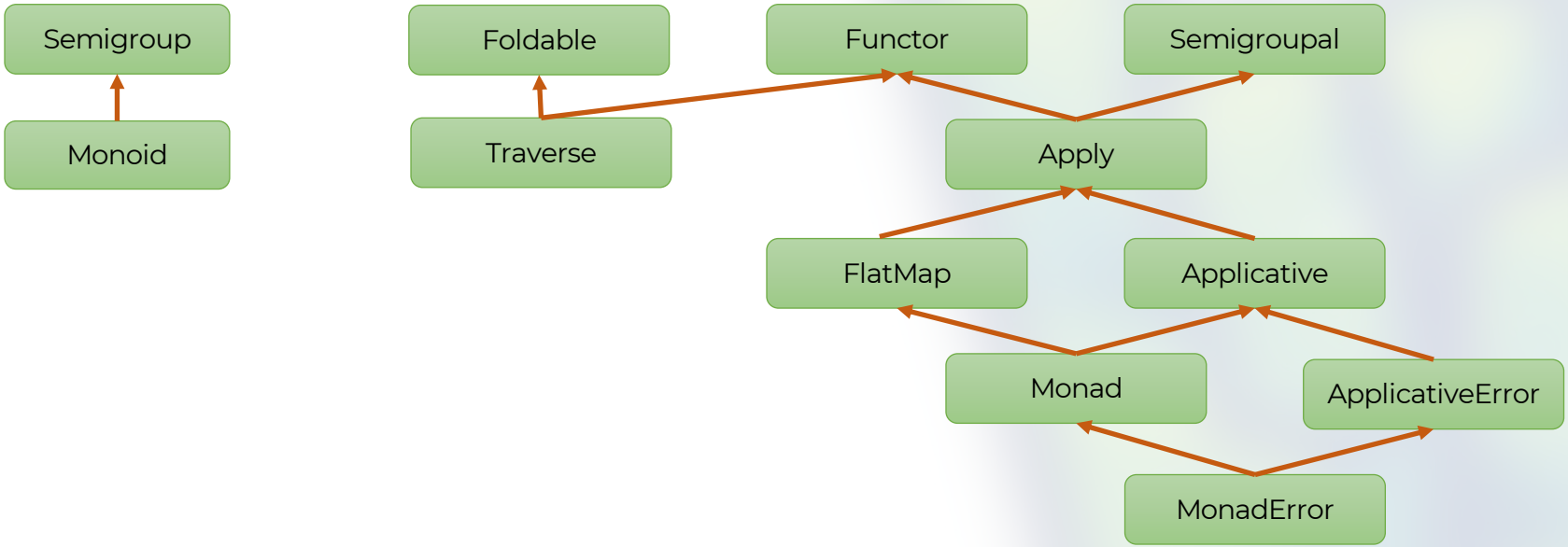
- turning nested data structures inside out
- general data combination APIs

```
// Some(List(2,4,6))
val allTrue = filterAsOption(List(2,4,6))(_ % 2 == 0)

// None
val someFalse = filterAsOption(List(1,2,3))(_ % 2 == 0)

// Valid(List(2,4,6))
val allTrueValidated = filterAsValidated(List(2,4,6))(_ % 2 == 0)

// Invalid(List("predicate for 1", "predicate for 3"))
val someFalseValidated = filterAsValidated(List(1,2,3))(_ % 2 == 0)
```

Cats TC Hierarchy



Cats TC Hierarchy



Cats rock