# WORK LOG OF JUNE 19

DANIEL R. BARRERO R.

## 1

The comonad defnition as a typeclass is

```
class Functor w => Comonad w where
  extract :: w a -> a
  duplicate :: w a -> w (w a)
```

And in Milewski's tic-tac-toe, *the evaluator* will be a comonad insntance, apparently. This evaluator defines a tree-to-vector behavior:

```
newtype W f a = W {runW :: forall n. f n -> Vec n a}
```

The following typings follow:

The type of `runW` is

```
runW :: W f a -> forall (n :: Nat). f n -> Vec n a
```

And that of `W` is

```
W :: (forall (n :: Nat). f n -> Vec n a) -> W f a
```

Also, `ghci` gives the following answer when asked for the kind of `W`:

```
:k W
> (Nat -> *) -> * -> *
```

*Date*: June 19, 2025.