

The following is an attempt to write a denotational semantics for Nu's scripting system, based on <https://www.youtube.com/watch?v=bmKYiUOEo2A>. This presentation uses a more improvised style of syntax than the Conal's, however.

Axiomatic Denotations - When a  $\mu$  is defined in terms of itself, we consider it axiomatic and irreducible directly in this context.

```
 $\mu$ :Value<a> =  
|  $\mu$ :Value< $\mu$ :Relation>  
|  $\mu$ :Value< $\mu$ :Address>  
|  $\mu$ :Value< $\mu$ :Name>  
|  $\mu$ :Value< $\mu$ :String>  
|  $\mu$ :Value< $\mu$ :Bool>  
|  $\mu$ :Value< $\mu$ :Unit>  
  
 $\mu$ :Stream<a> =  
|  $\mu$ :Address ->  $\mu$ :Stream<a>  
|  $\mu$ :Name ->  $\mu$ :Relation ->  $\mu$ :Stream<a>  
|  $\mu$ :Stream< $\alpha\alpha$  -> a) ->  $\mu$ :Stream<a>  
|  $\mu$ :Stream< $\alpha\mu$ :Stream< $\beta\mu$ :Stream<a when a =  $\alpha$  *  $\beta$ >  
|  $\mu$ :Stream< $\alpha\mu$ :Stream< $\beta\mu$ :Stream<a when a =  $\alpha$  |  $\beta$ >
```

$\mu$ :Effect =  $\mu$ :Effect // transforms the environment

$\mu$ :Declare a =  $\mu$ :Name ->  $\mu$ :Declare a // augments the environment

### Derived Denotations

$\mu$ :Get<a> =  $\mu$ :Name ->  $\mu$ :Relation ->  $\mu$ :Value<a>

$\mu$ :Set<a> =  $\mu$ :Name ->  $\mu$ :Relation ->  $\mu$ :Value<a> ->  $\mu$ :Effect

$\mu$ :Command<a> =  $\mu$ :Value<a> ->  $\mu$ :Effect

$\mu$ :Fold<a b> = ( $\mu$ :Value<a> -> b) ->  $\mu$ :Stream<a> -> b

$\mu$ :Define<a> =  $\mu$ :Declare< $\mu$ :Value<a>>

$\mu$ :Variable<a> =  $\mu$ :Declare< $\mu$ :Stream<a>>

$\mu$ :Equate<a> = \name -> \rel ->  $\mu$ :Stream<a> ->  $\mu$ :Fold ( $\mu$ :Set<a> name rel)

$\mu$ :Handle<a> =  $\mu$ :Stream<a> ->  $\mu$ :Fold  $\mu$ : Command <a>