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 µ is defined in terms of itself, we consider it axiomatic and irreducible directly in this context.

µ:Value<a> =

| µ:Value<µ:Relation>

| µ:Value<µ:Address>

| µ:Value<µ:Name>

| µ:Value<µ:String>

| µ:Value<µ:Bool>

| µ:Value<µ:Unit>

µ:Stream<a> =

| µ:Address -> µ:Stream<a>

| µ:Name -> µ:Relation -> µ:Stream<a>

| µ:Stream<α> -> (α -> a) -> µ:Stream<a>

| µ:Stream<α> -> µ:Stream<β> -> µ:Stream<a when a = α \* β)

| µ:Stream<α> -> µ:Stream<β> -> µ:Stream<a when a = α | β)

µ:Effect = µ:Effect // transforms the environment

µ:Declare a = µ:Name -> µ:Declaration a // augments the environment

Derived Denotations

µ:Get<a> = µ:Name -> µ:Relation -> µ:Value<a>

µ:Set<a> = µ:Name -> µ:Relation -> µ:Value<a> -> µ:Effect

µ:Command<a> = µ:Value<a> -> µ:Effect

µ:Fold<a b> = (µ:Value<a> -> b) -> µ:Stream<a> -> b

µ:Define<a> = µ:Declare<µ:Value<a>>

µ:Variable<a> = µ:Declare<µ:Stream<a>>

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

µ:Handle<a> = µ:Stream<a> -> µ:Fold µ: Command <a>