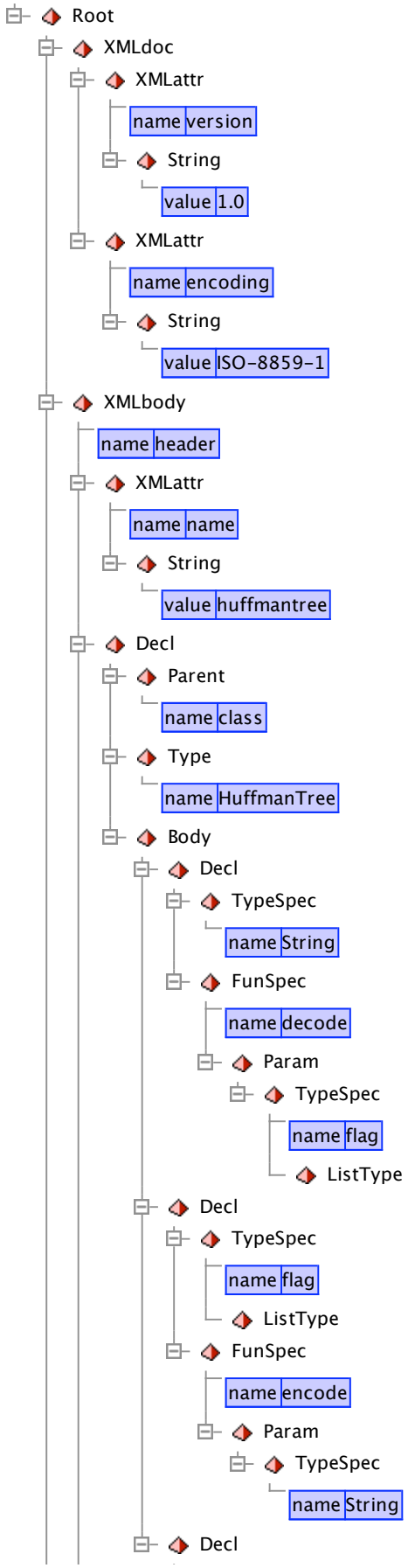
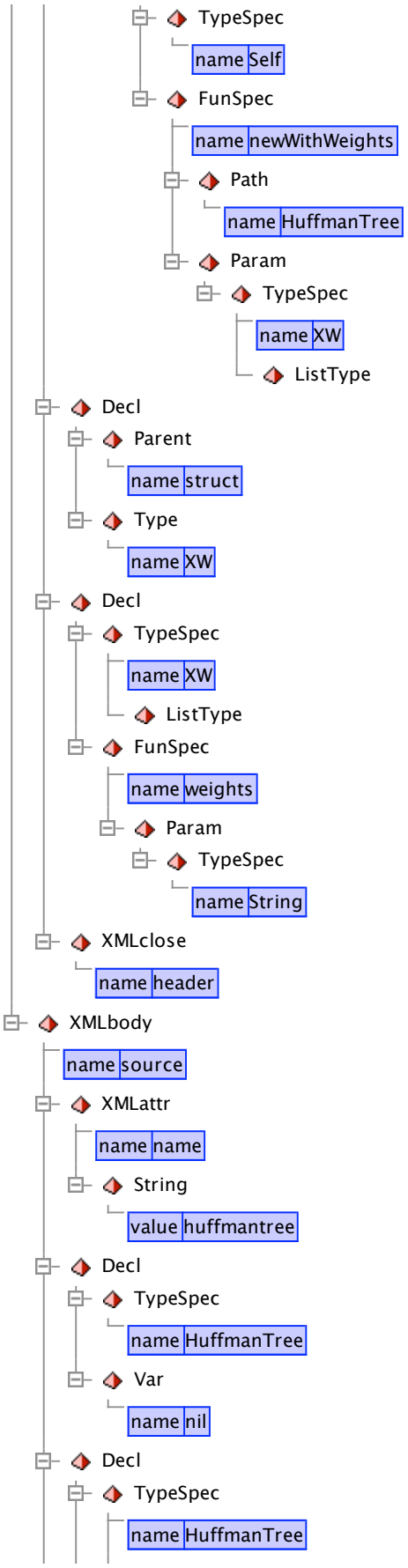
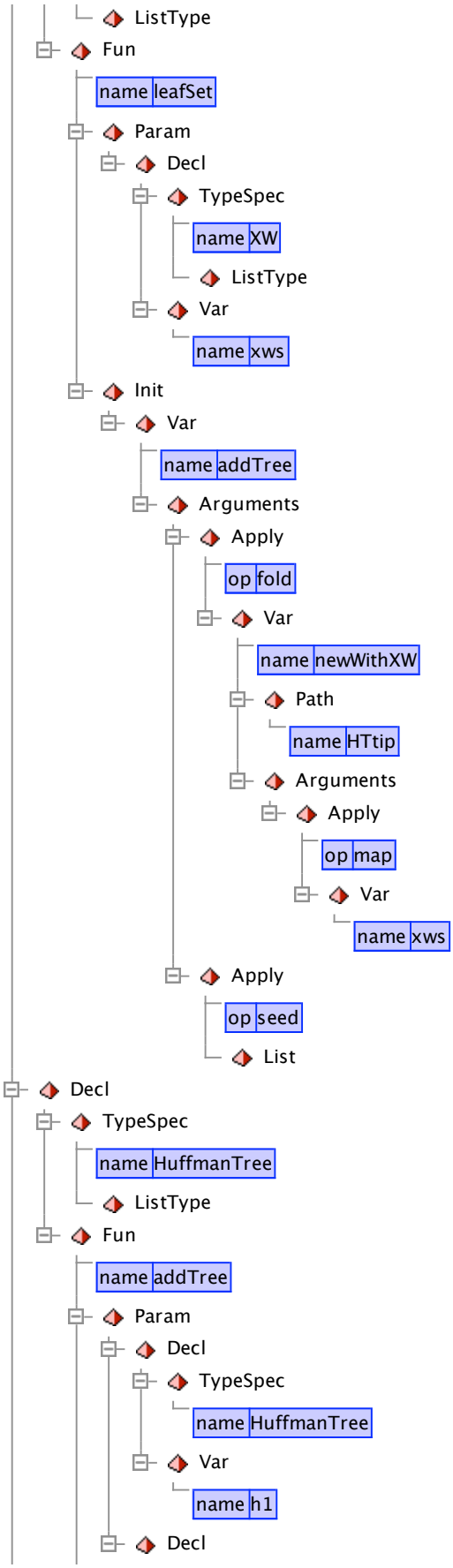
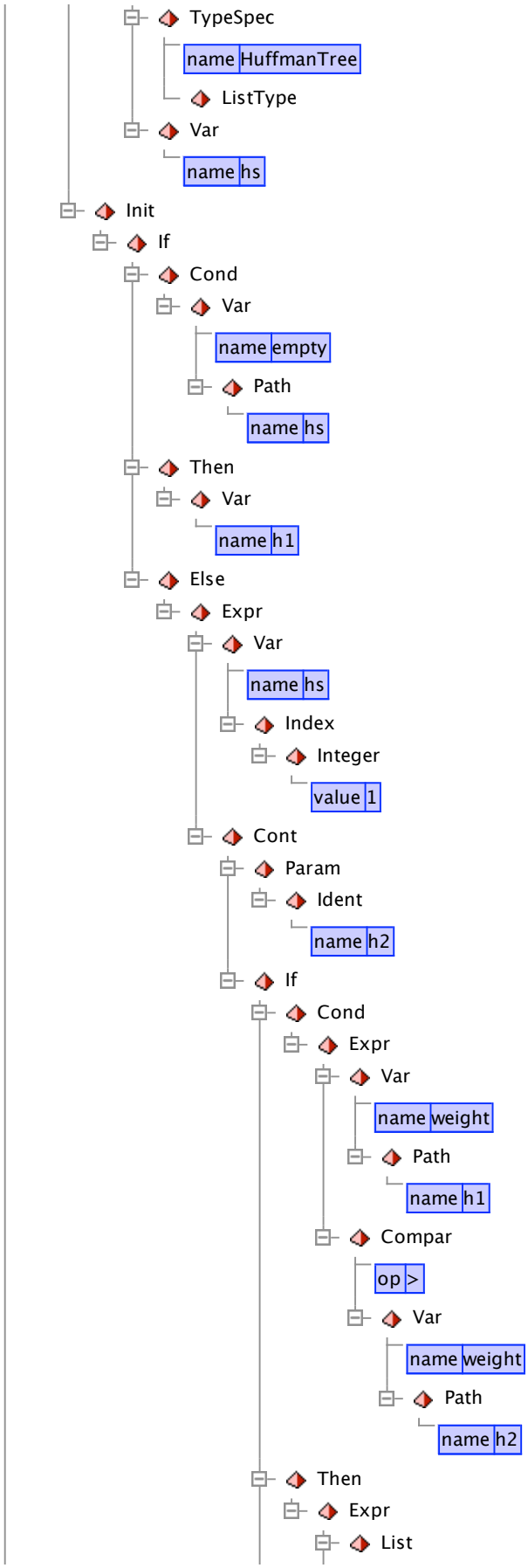
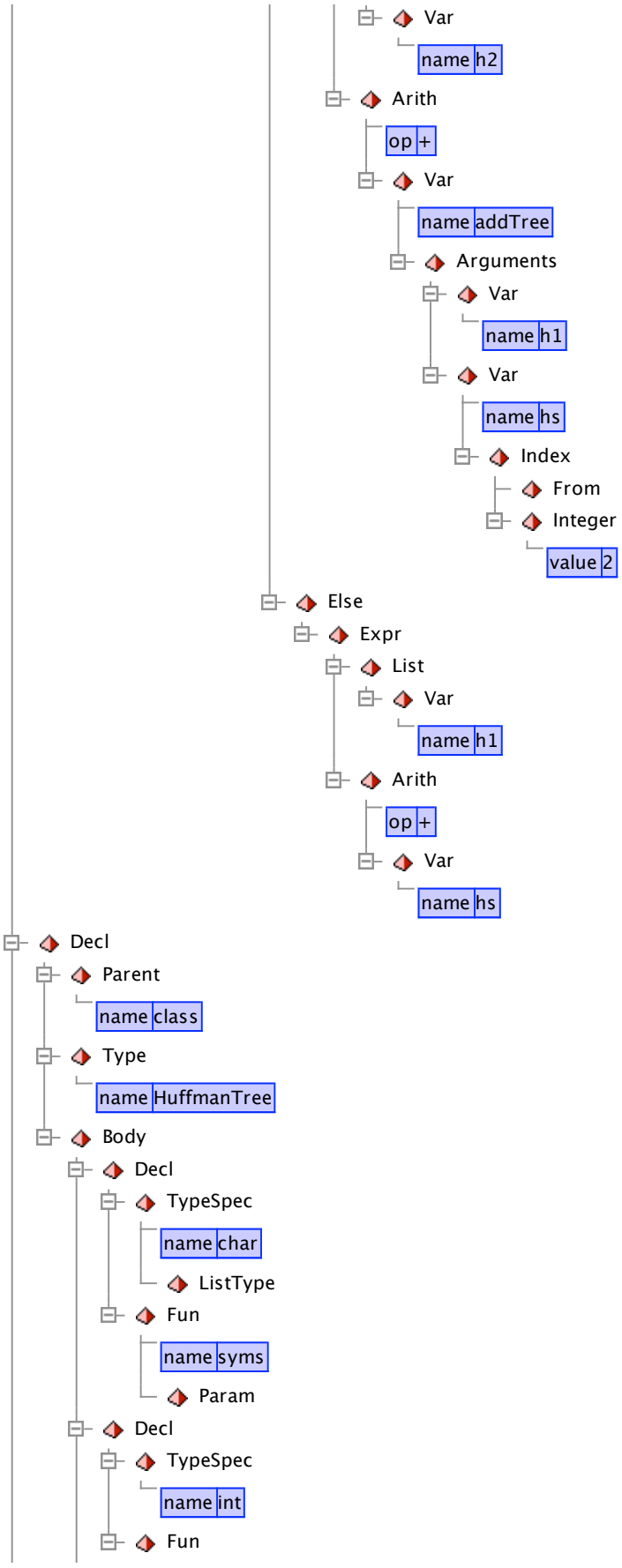
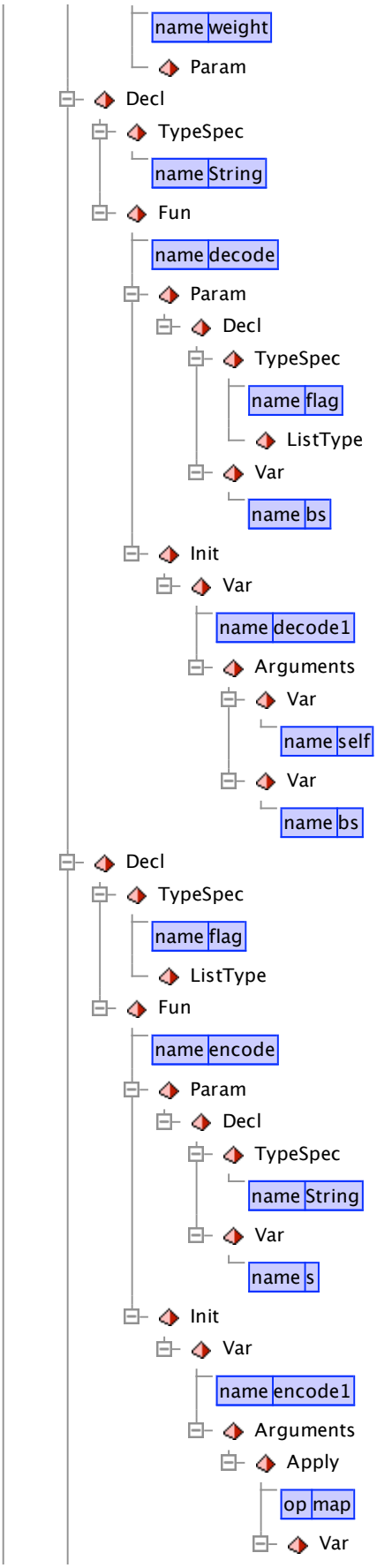

Root
XMLdoc
XMLattr
name version
String
value 1.0
XMLattr
name encoding
String
value ISO-8859-1
XMLbody
name header
XMLattr
name name
String
value huffmantree
Decl
Parent
name class
Type
name HuffmanTree
Body
Decl
TypeSpec
name String
FunSpec
name decode
Param
TypeSpec
name flag
ListType
Decl
TypeSpec
name flag
ListType
FunSpec
name encode
Param
TypeSpec
name String
Decl

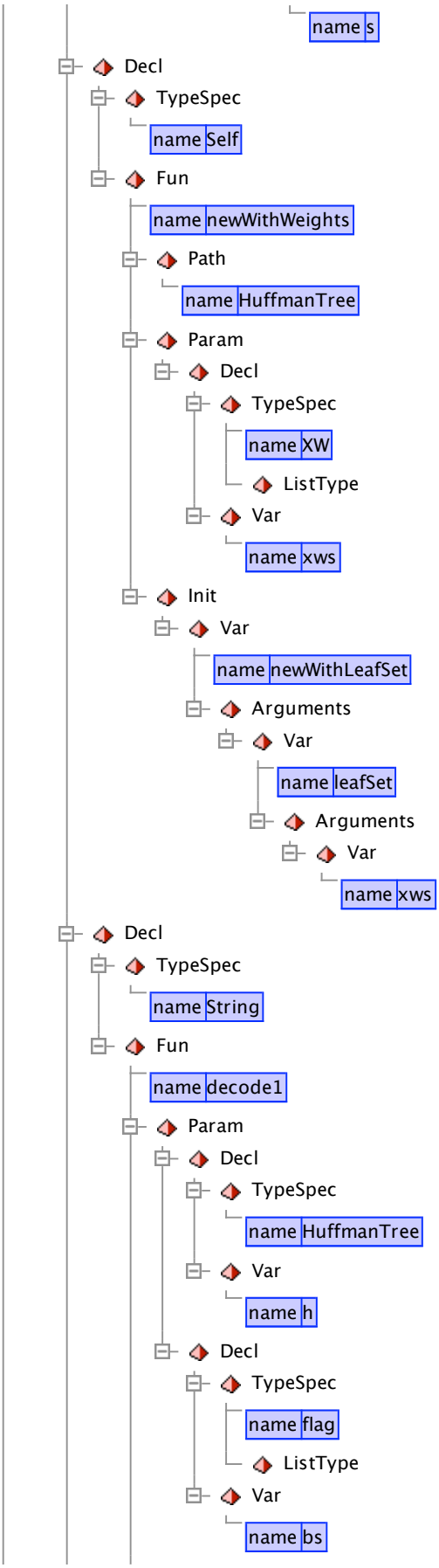

TypeSpec
name Self
FunSpec
name newWithWeights
Path
name HuffmanTree
Param
TypeSpec
name XW
ListType
Decl
Parent
name struct
Type
name XW
Decl
TypeSpec
name XW
ListType
FunSpec
name weights
Param
TypeSpec
name String
XMLclose
name header
XMLbody
name source
XMLattr
name name
String
value huffmantree
Decl
TypeSpec
name HuffmanTree
Var
name nil
Decl
TypeSpec
name HuffmanTree

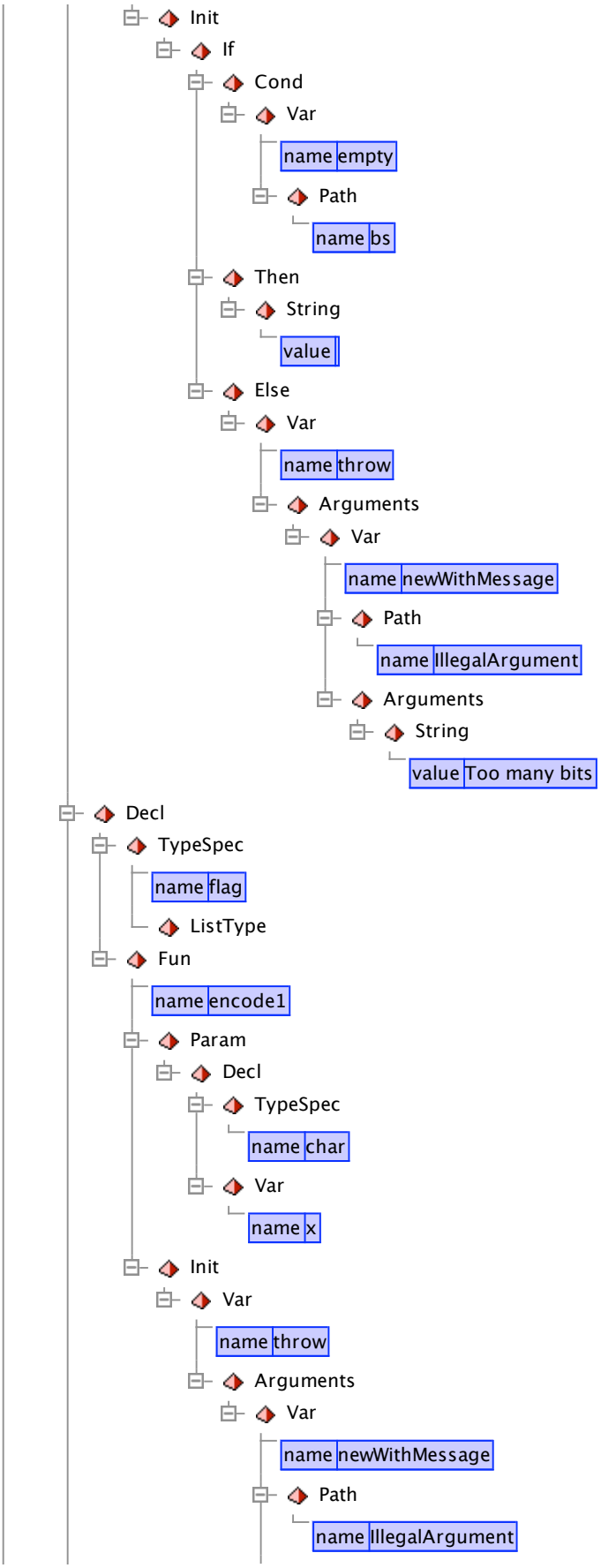

ListType
Fun
name leafSet
Param
Decl
TypeSpec
name XW
ListType
Var
name xws
Init
Var
name addTree
Arguments
Apply
op fold
Var
name newWithXW
Path
name HTtip
Arguments
Apply
op map
Var
name xws
Apply
op seed
List
Decl
TypeSpec
name HuffmanTree
ListType
Fun
name addTree
Param
Decl
TypeSpec
name HuffmanTree
Var
name h1
Decl

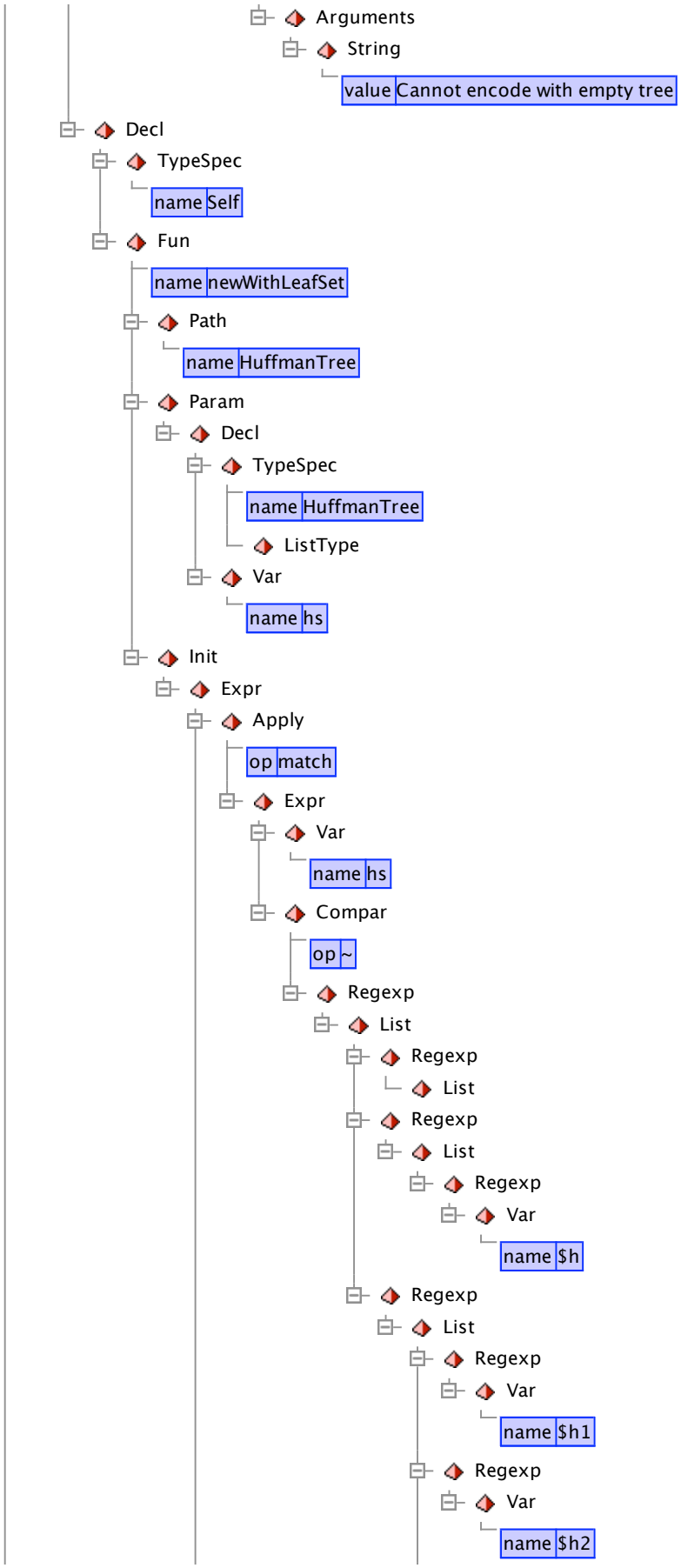

TypeSpec
name HuffmanTree
ListType
Var
name hs
Init
If
Cond
Var
name empty
Path
name hs
Then
Var
name h1
Else
Expr
Var
name hs
Index
Integer
value 1
Cont
Param
Ident
name h2
If
Cond
Expr
Var
name weight
Path
name h1
Compar
op >
Var
name weight
Path
name h2
Then
Expr
List

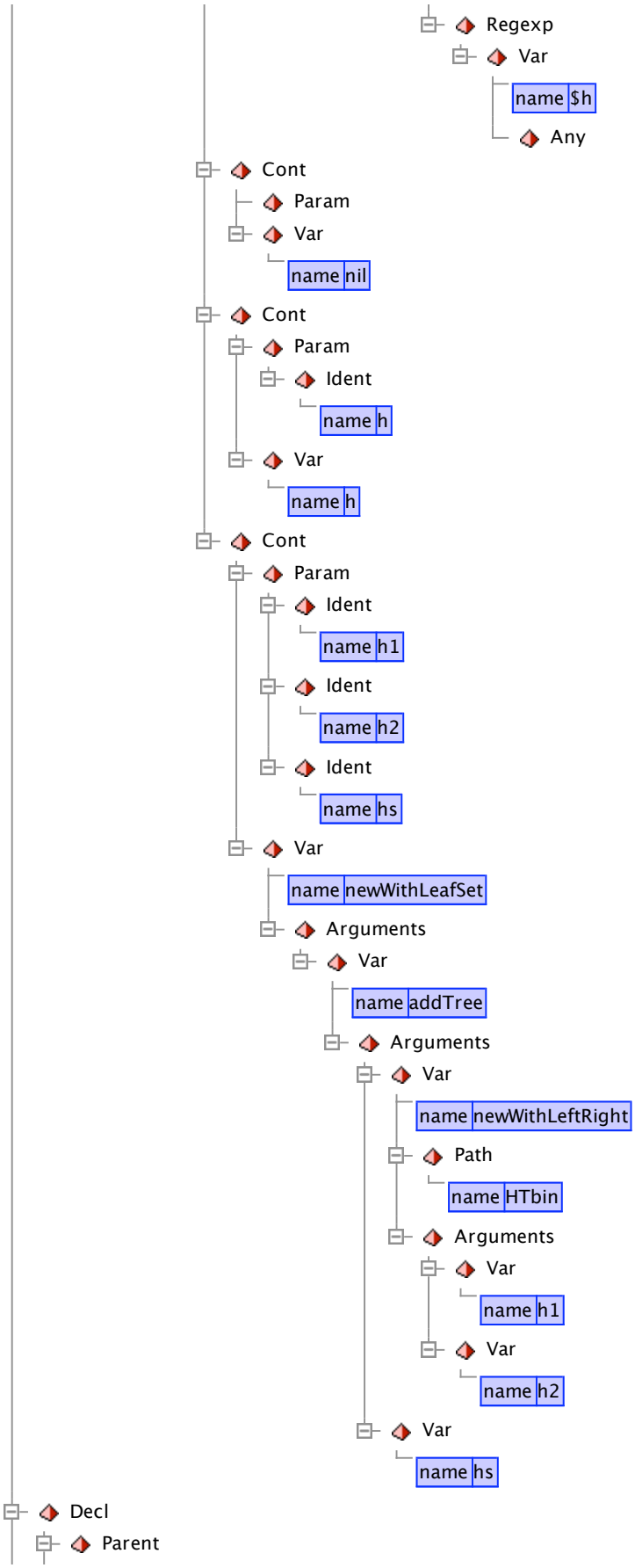

Var
name h2
Arith
op +
Var
name addTree
Arguments
Var
name h1
Var
name hs
Index
From
Integer
value 2
Else
Expr
List
Var
name h1
Arith
op +
Var
name hs
Decl
Parent
name class
Type
name HuffmanTree
Body
Decl
TypeSpec
name char
ListType
Fun
name syms
Param
Decl
TypeSpec
name int
Fun

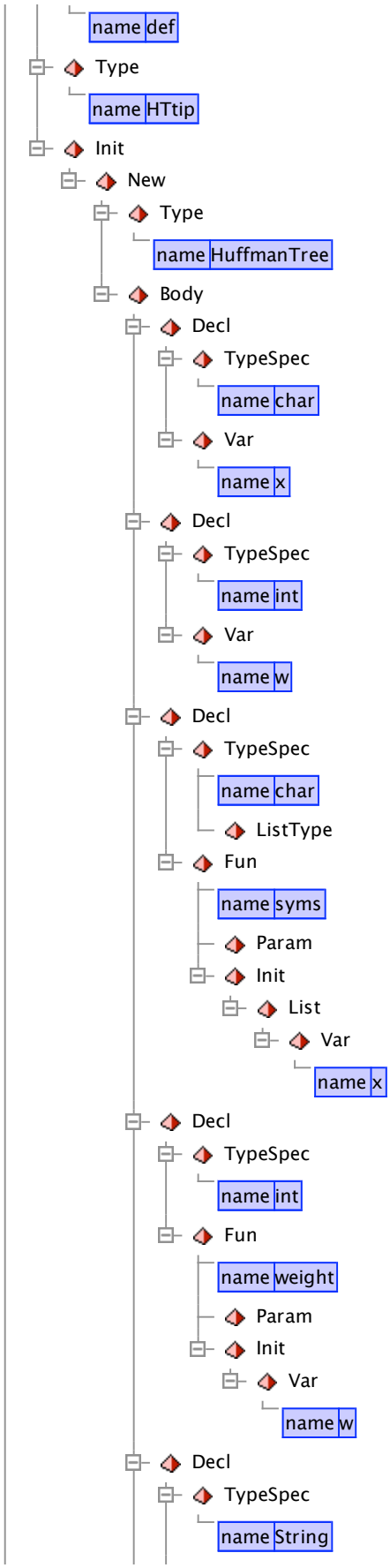

name weight
Param
Decl
TypeSpec
name String
Fun
name decode
Param
Decl
TypeSpec
name flag
ListType
Var
name bs
Init
Var
name decode1
Arguments
Var
name self
Var
name bs
Decl
TypeSpec
name flag
ListType
Fun
name encode
Param
Decl
TypeSpec
name String
Var
name s
Init
Var
name encode1
Arguments
Apply
op map
Var

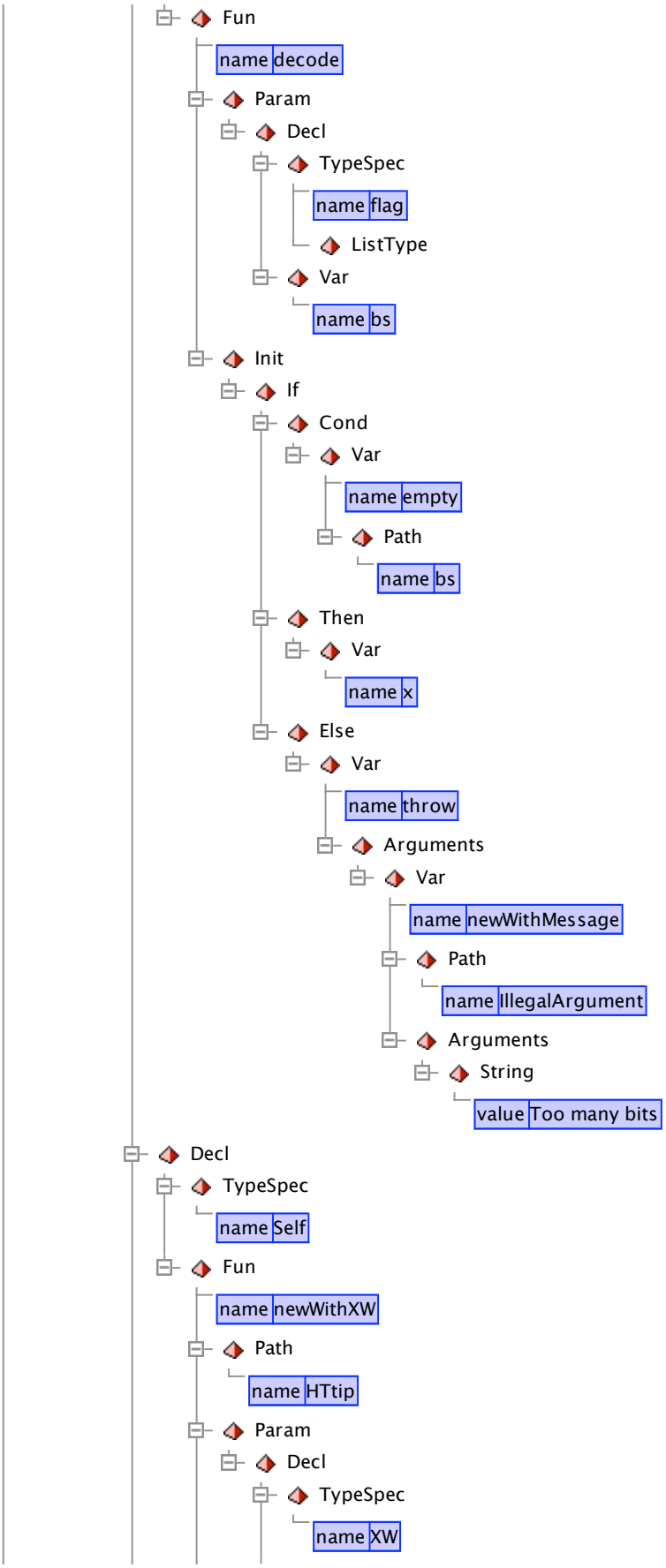

name s
Decl
TypeSpec
name Self
Fun
name newWithWeights
Path
name HuffmanTree
Param
Decl
TypeSpec
name XW
ListType
Var
name xws
Init
Var
name newWithLeafSet
Arguments
Var
name leafSet
Arguments
Var
name xws
Decl
TypeSpec
name String
Fun
name decode1
Param
Decl
TypeSpec
name HuffmanTree
Var
name h
Decl
TypeSpec
name flag
ListType
Var
name bs

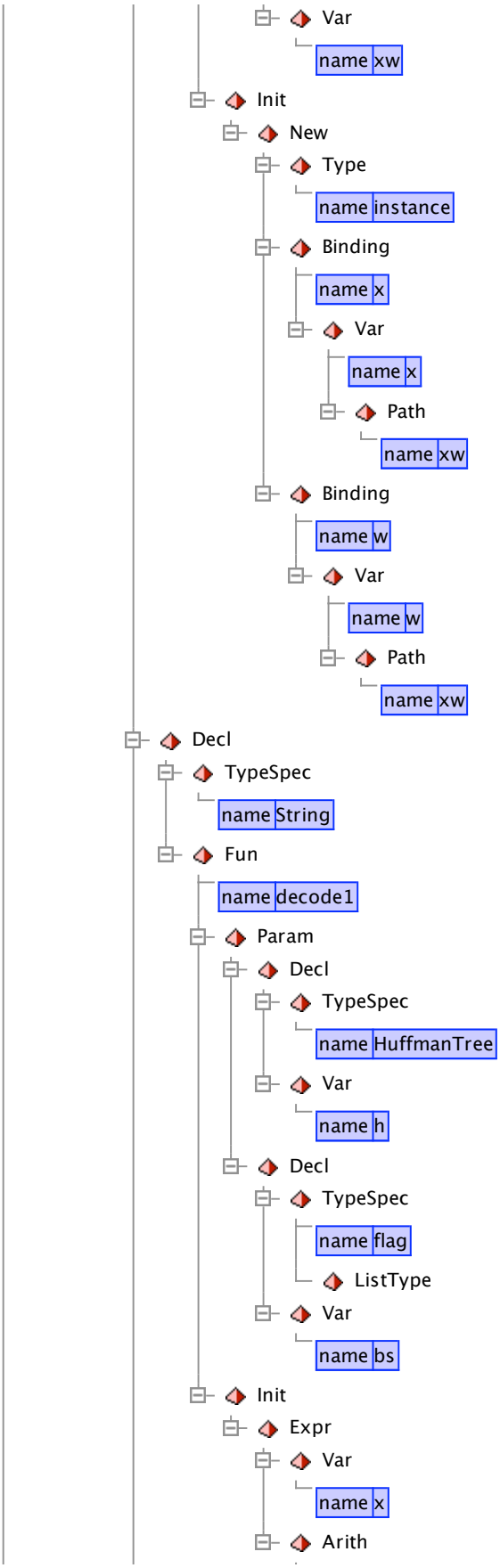

Init
If
Cond
Var
name empty
Path
name bs
Then
String
value
Else
Var
name throw
Arguments
Var
name newWithMessage
Path
name IllegalArgument
Arguments
String
value Too many bits
Decl
TypeSpec
name flag
ListType
Fun
name encode1
Param
Decl
TypeSpec
name char
Var
name x
Init
Var
name throw
Arguments
Var
name newWithMessage
Path
name IllegalArgument

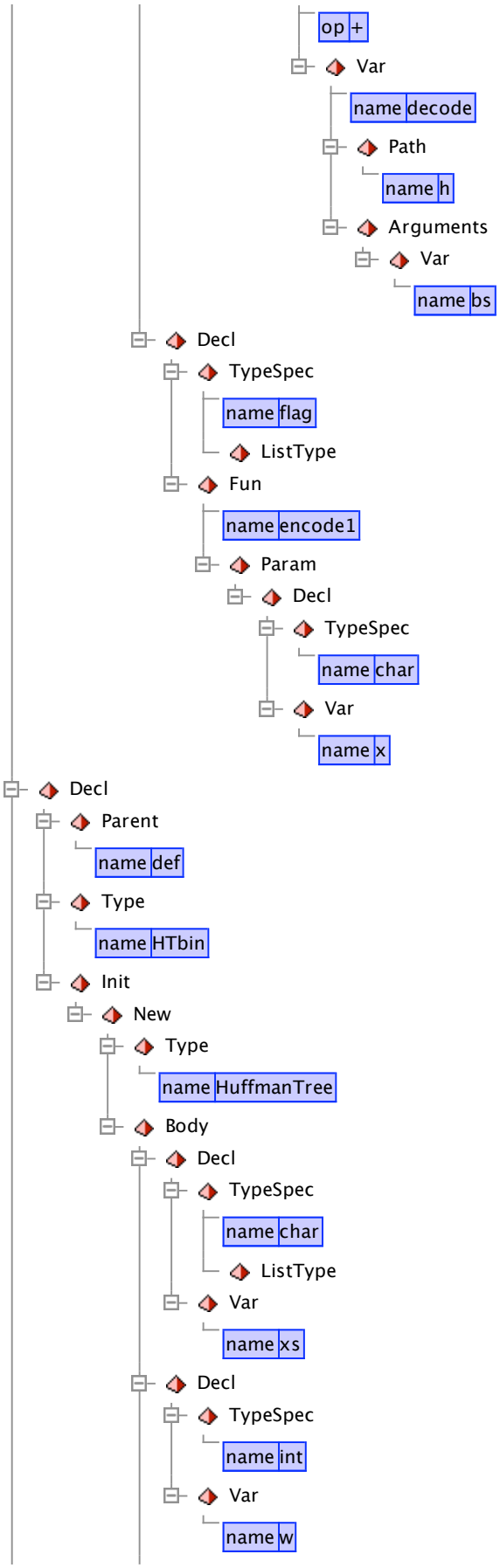

Arguments
String
value Cannot encode with empty tree
Decl
TypeSpec
name Self
Fun
name newWithLeafSet
Path
name HuffmanTree
Param
Decl
TypeSpec
name HuffmanTree
ListType
Var
name hs
Init
Expr
Apply
op match
Expr
Var
name hs
Compar
op ~
Regexp
List
Regexp
List
Regexp
List
Regexp
Var
name $h
Regexp
List
Regexp
Var
name $h1
Regexp
Var
name $h2

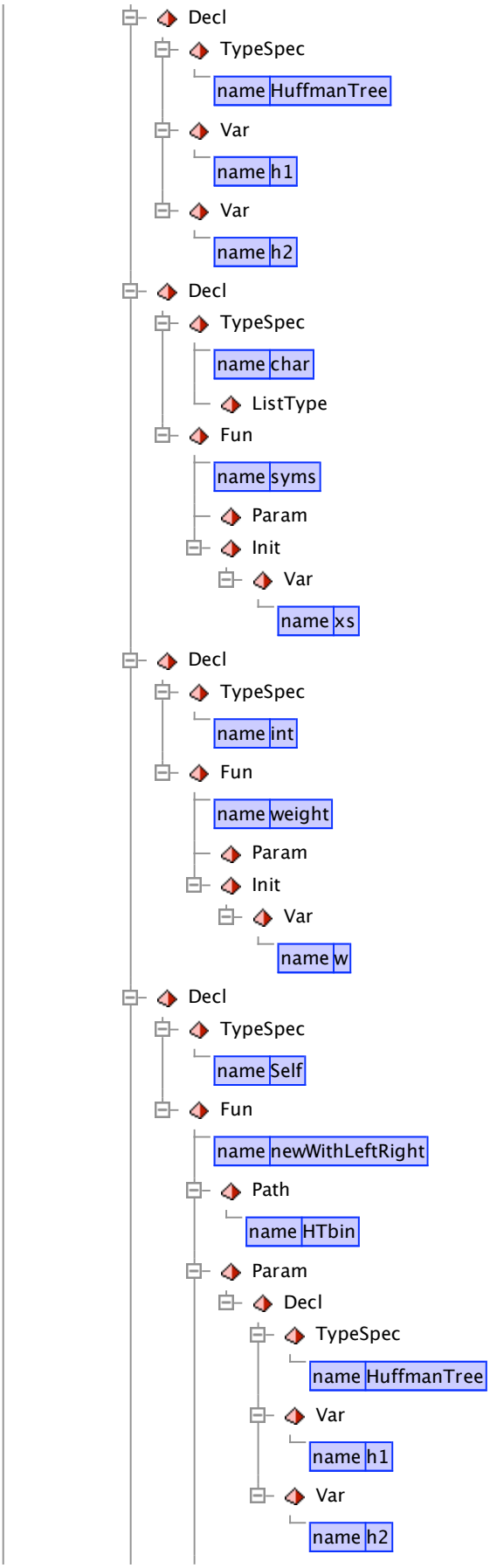

Regexp
Var
name $h
Any
Cont
Param
Var
name nil
Cont
Param
Ident
name h
Var
name h
Cont
Param
Ident
name h1
Ident
name h2
Ident
name hs
Var
name newWithLeafSet
Arguments
Var
name addTree
Arguments
Var
name newWithLeftRight
Path
name HTbin
Arguments
Var
name h1
Var
name h2
Var
name hs
Decl
Parent

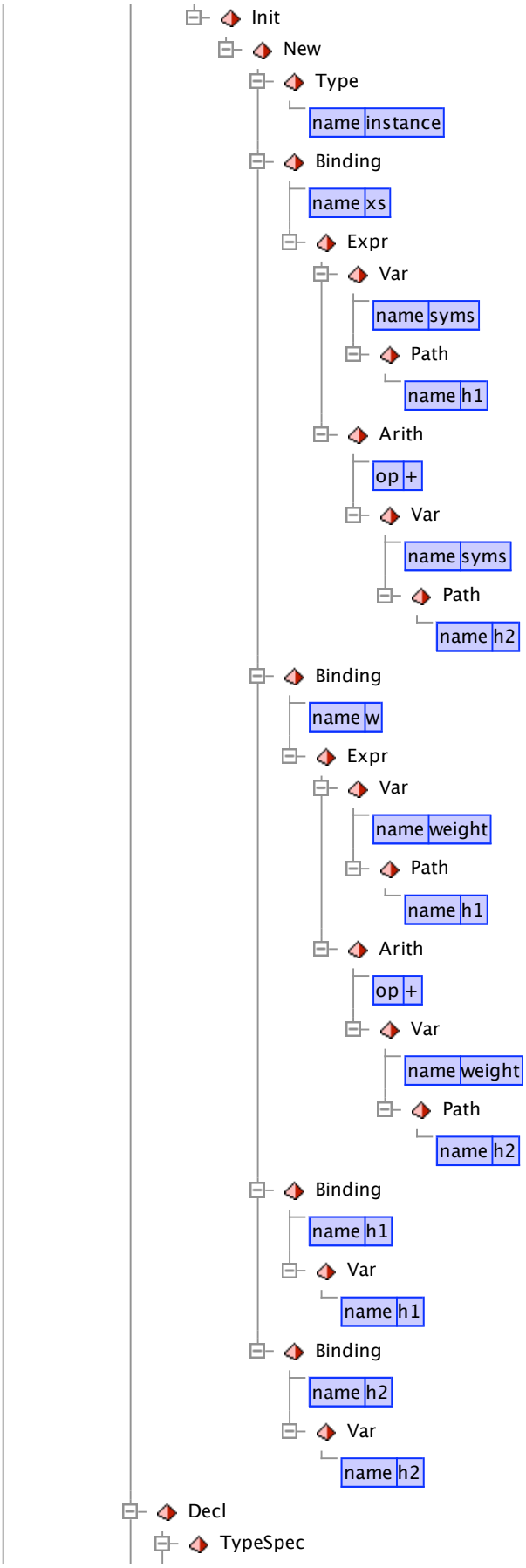

name def
Type
name HTtip
Init
New
Type
name HuffmanTree
Body
Decl
TypeSpec
name char
Var
name x
Decl
TypeSpec
name int
Var
name w
Decl
TypeSpec
name char
ListType
Fun
name syms
Param
Init
List
Var
name x
Decl
TypeSpec
name int
Fun
name weight
Param
Init
Var
name w
Decl
TypeSpec
name String

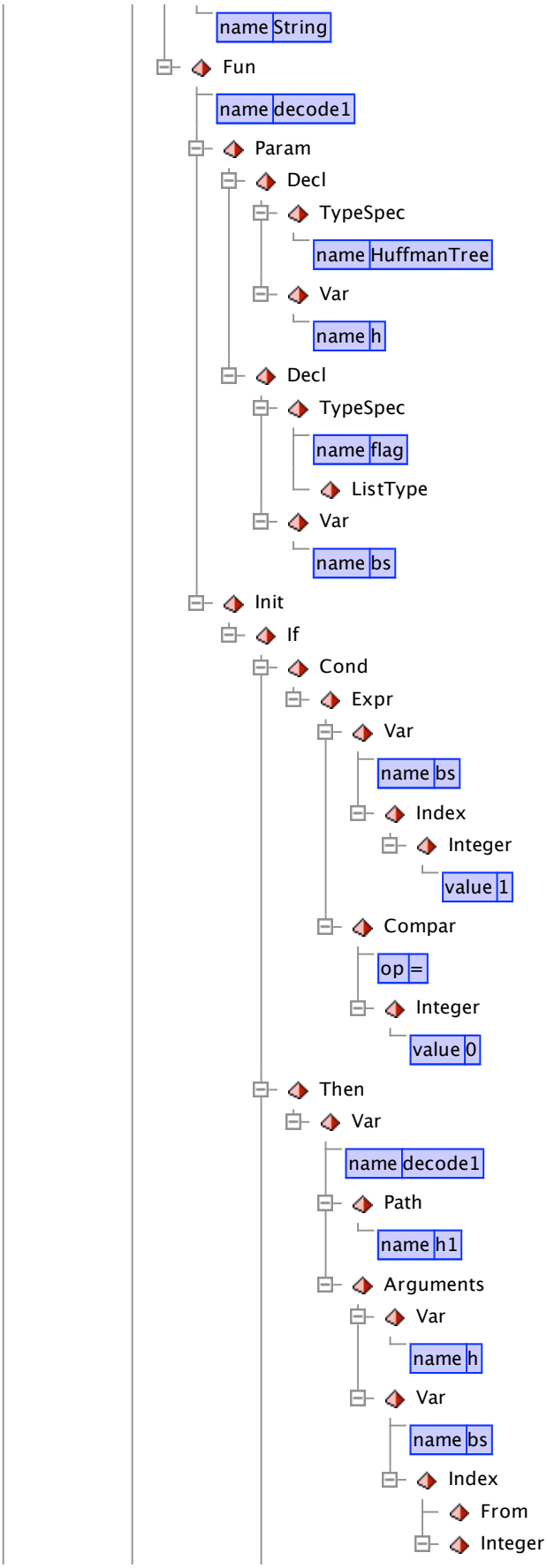

Fun
name decode
Param
Decl
TypeSpec
name flag
ListType
Var
name bs
Init
If
Cond
Var
name empty
Path
name bs
Then
Var
name x
Else
Var
name throw
Arguments
Var
name newWithMessage
Path
name IllegalArgument
Arguments
String
value Too many bits
Decl
TypeSpec
name Self
Fun
name newWithXW
Path
name HTtip
Param
Decl
TypeSpec
name XW

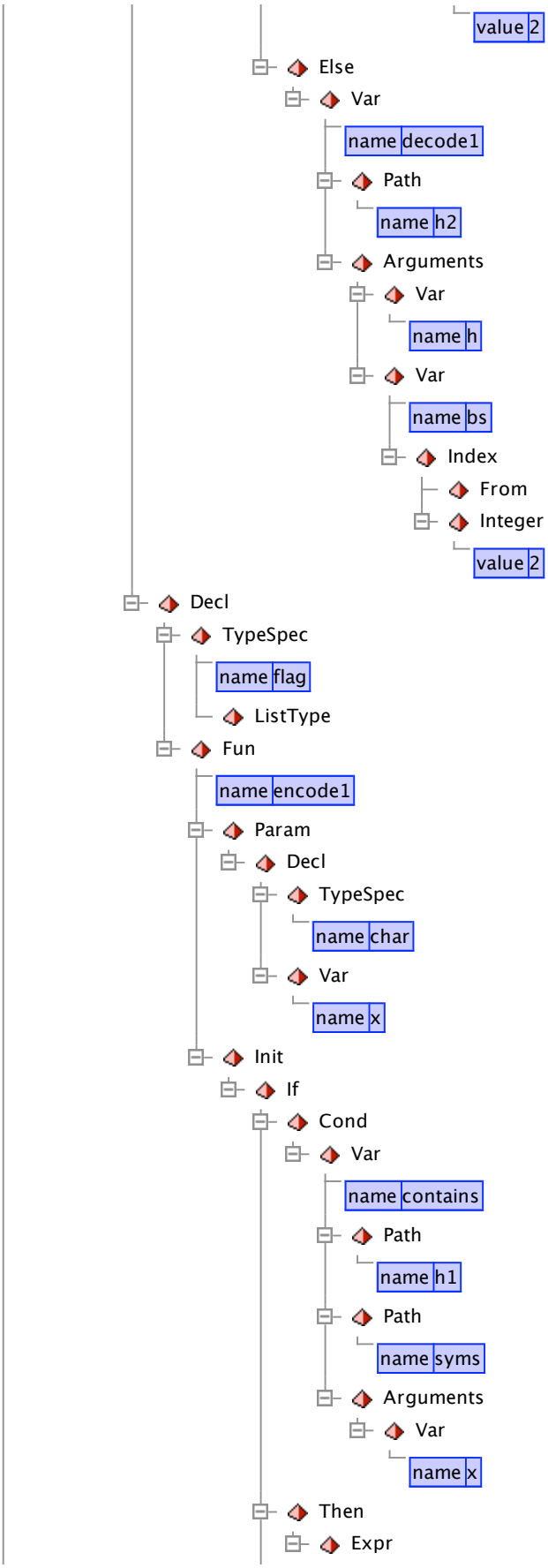

Var
name xw
Init
New
Type
name instance
Binding
name x
Var
name x
Path
name xw
Binding
name w
Var
name w
Path
name xw
Decl
TypeSpec
name String
Fun
name decode1
Param
Decl
TypeSpec
name HuffmanTree
Var
name h
Decl
TypeSpec
name flag
ListType
Var
name bs
Init
Expr
Var
name x
Arith

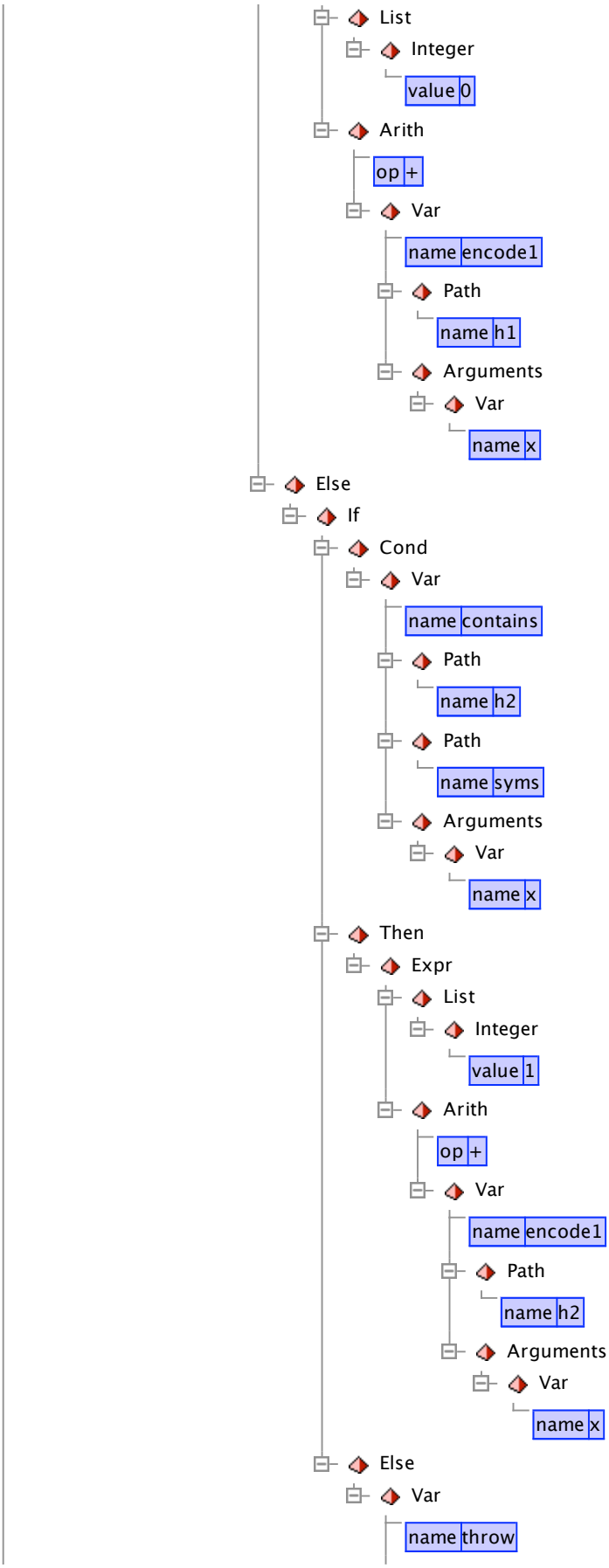

op +
Var
name decode
Path
name h
Arguments
Var
name bs
Decl
TypeSpec
name flag
ListType
Fun
name encode1
Param
Decl
TypeSpec
name char
Var
name x
Decl
Parent
name def
Type
name HTbin
Init
New
Type
name HuffmanTree
Body
Decl
TypeSpec
name char
ListType
Var
name xs
Decl
TypeSpec
name int
Var
name w

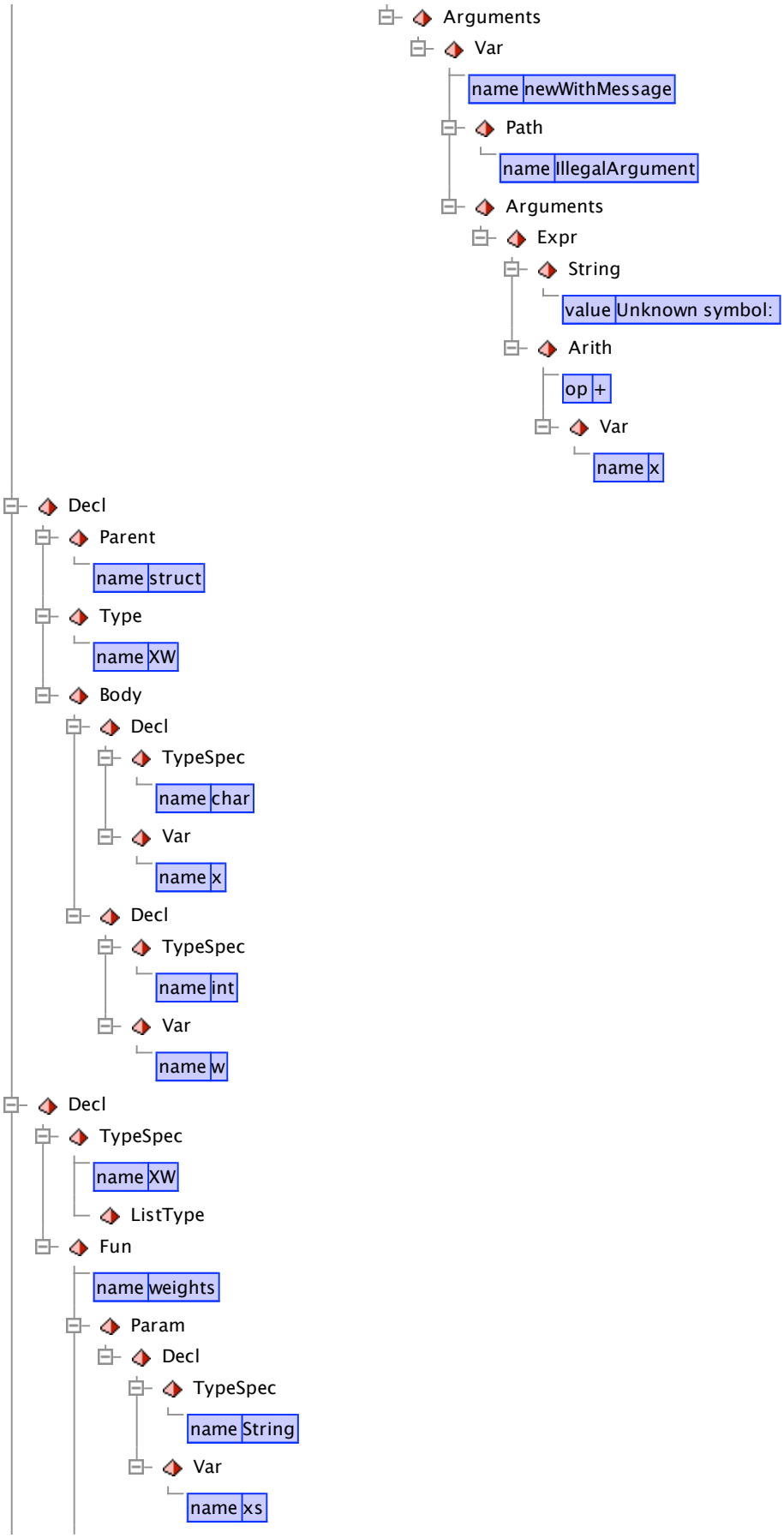

Decl
TypeSpec
name HuffmanTree
Var
name h1
Var
name h2
Decl
TypeSpec
name char
ListType
Fun
name syms
Param
Init
Var
name xs
Decl
TypeSpec
name int
Fun
name weight
Param
Init
Var
name w
Decl
TypeSpec
name Self
Fun
name newWithLeftRight
Path
name HTbin
Param
Decl
TypeSpec
name HuffmanTree
Var
name h1
Var
name h2

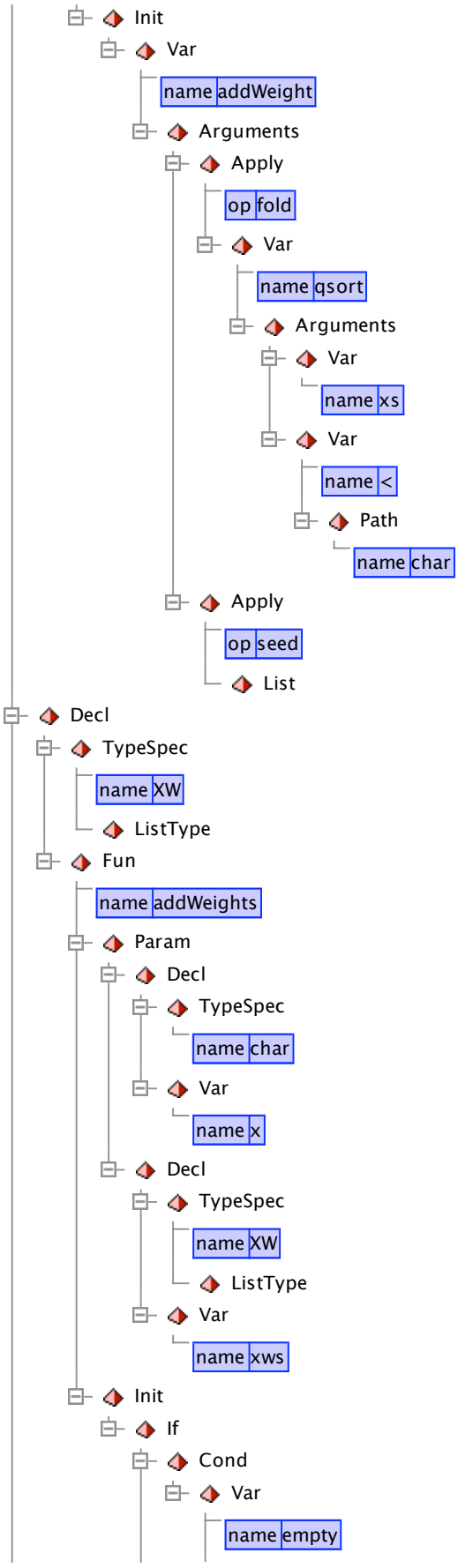

Init
New
Type
name instance
Binding
name xs
Expr
Var
name syms
Path
name h1
Arith
op +
Var
name syms
Path
name h2
Binding
name w
Expr
Var
name weight
Path
name h1
Arith
op +
Var
name weight
Path
name h2
Binding
name h1
Var
name h1
Binding
name h2
Var
name h2
Decl
TypeSpec

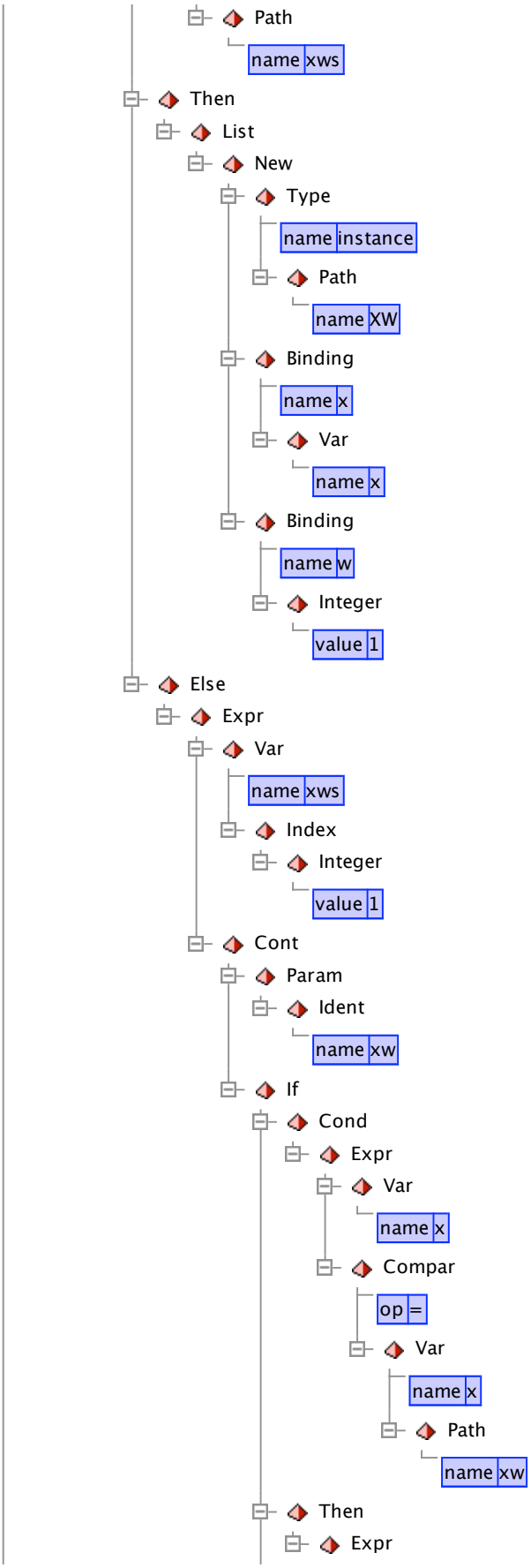

name String
Fun
name decode1
Param
Decl
TypeSpec
name HuffmanTree
Var
name h
Decl
TypeSpec
name flag
ListType
Var
name bs
Init
If
Cond
Expr
Var
name bs
Index
Integer
value 1
Compar
op =
Integer
value 0
Then
Var
name decode1
Path
name h1
Arguments
Var
name h
Var
name bs
Index
From
Integer

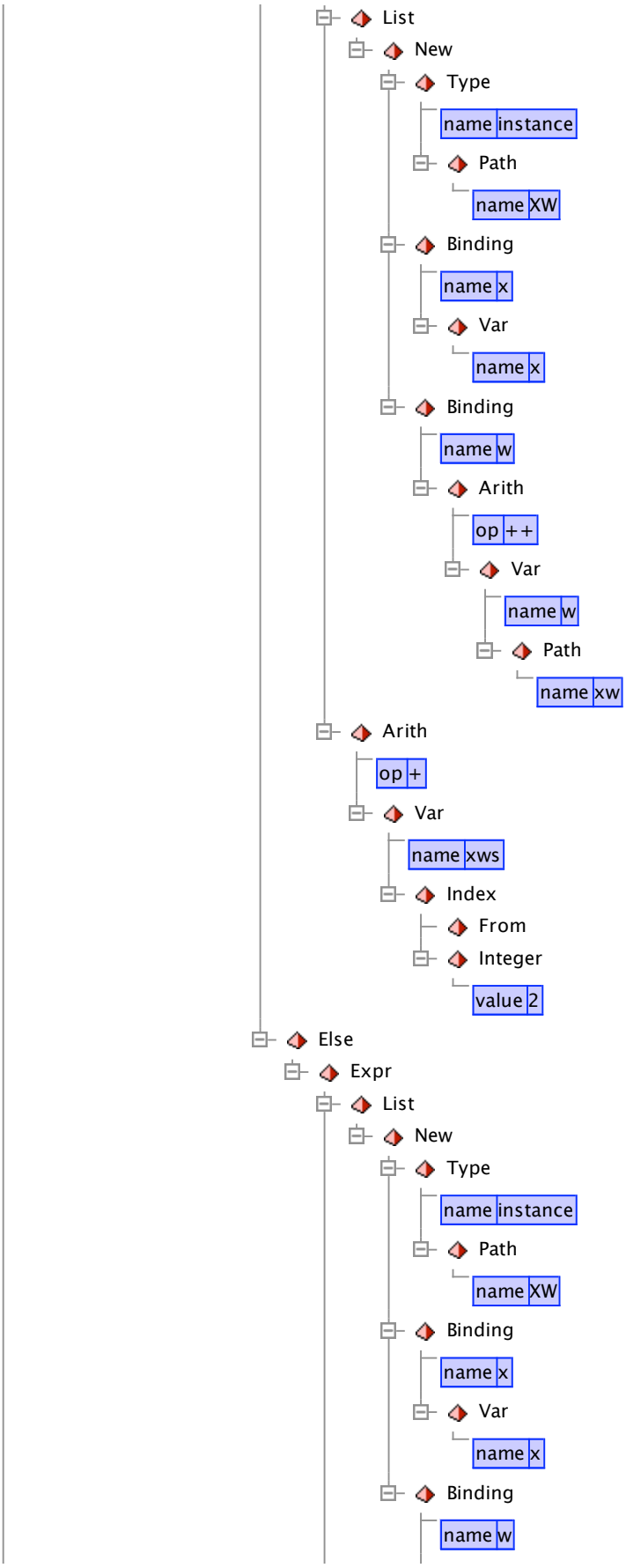

value 2
Else
Var
name decode1
Path
name h2
Arguments
Var
name h
Var
name bs
Index
From
Integer
value 2
Decl
TypeSpec
name flag
ListType
Fun
name encode1
Param
Decl
TypeSpec
name char
Var
name x
Init
If
Cond
Var
name contains
Path
name h1
Path
name syms
Arguments
Var
name x
Then
Expr

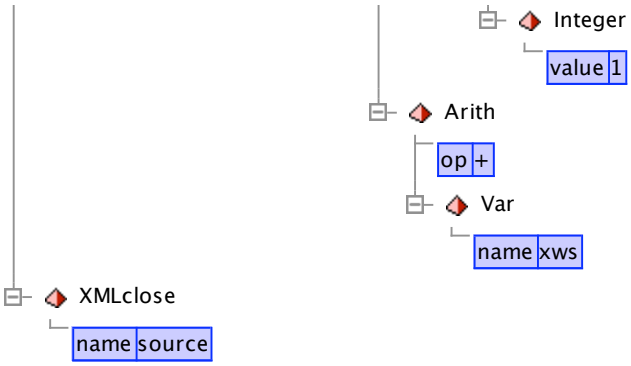

List
Integer
value 0
Arith
op +
Var
name encode1
Path
name h1
Arguments
Var
name x
Else
If
Cond
Var
name contains
Path
name h2
Path
name syms
Arguments
Var
name x
Then
Expr
List
Integer
value 1
Arith
op +
Var
name encode1
Path
name h2
Arguments
Var
name x
Else
Var
name throw


Arguments
Var
name newWithMessage
Path
name IllegalArgument
Arguments
Expr
String
value Unknown symbol:
Arith
op +
Var
name x
Decl
Parent
name struct
Type
name XW
Body
Decl
TypeSpec
name char
Var
name x
Decl
TypeSpec
name int
Var
name w
Decl
TypeSpec
name XW
ListType
Fun
name weights
Param
Decl
TypeSpec
name String
Var
name xs


Init
Var
name addWeight
Arguments
Apply
op fold
Var
name qsort
Arguments
Var
name xs
Var
name <
Path
name char
Apply
op seed
List
Decl
TypeSpec
name XW
ListType
Fun
name addWeights
Param
Decl
TypeSpec
name char
Var
name x
Decl
TypeSpec
name XW
ListType
Var
name xws
Init
If
Cond
Var
name empty


Path
name xws
Then
List
New
Type
name instance
Path
name XW
Binding
name x
Var
name x
Binding
name w
Integer
value 1
Else
Expr
Var
name xws
Index
Integer
value 1
Cont
Param
Ident
name xw
If
Cond
Expr
Var
name x
Compar
op =
Var
name x
Path
name xw
Then
Expr


List
New
Type
name instance
Path
name XW
Binding
name x
Var
name x
Binding
name w
Arith
op ++
Var
name w
Path
name xw
Arith
op +
Var
name xws
Index
From
Integer
value 2
Else
Expr
List
New
Type
name instance
Path
name XW
Binding
name x
Var
name x
Binding
name w


Integer
value
1
Arith
op
+
Var
name
xws
XMLclose
name
source