type
Bounds
datatype
function
datatypes
pattern
position
values
ExprF
AnnFix
constructors
generic
one
use
two
text
Expr
instance
information
functor
selection
constructor
right
value
argument
using
need
selections
example
error
Int
types
tree
left
case
functors
bounds
annotated
work
algebra
ix
editor
class
data
programming
new
range
functions
parser
expression
BareExpr
expressions
family
PositionalExpr
Zipper
Tuples
structure
zipper
algebras
node
Fix
Range
Num
parsing
Failed
definition
first
recursive
children
structural
Right
version
original
languages
model
specific
collapse
list
AST
Div
Haskell
ErrorAlg
Either
way
building
chain1
pToken
blocks
margin
HFunctor
build
child
Add
libraries
Traversable
Foldable
token
Sub
catamorphisms
language
like
positions
number
Mul
view
xy
Type
trees
every
input
fields
user
Maybe
MultiRec
section
source
fixed
many
match
make
give
available
whitespace
expr
result
take
mkErrorAlg
offsets
terms
points
look
code
also
called
errors
alg
often
Bool
Ann
arguments
always
possible
xi
OK
used
Ann
Mul
Type