package ast

```
import "github.com/mewmew/uc/ast"
```

Package ast declares the types used to represent abstract syntax trees of µC soure code.

Index

- func (n *IfStmt) Start() int

type IndexExpr

- func (n *IndexExpr) Start() int

type Node

type ParenExpr

- func (n *ParenExpr) Start() int

type ReturnStmt

- func (n *ReturnStmt) Start() int

type Stmt

type UnaryExpr

- func (n *UnaryExpr) Start() int

type VarDecl

- func (n *VarDecl) Start() int

type WhileStmt

- func (n *WhileStmt) Start() int

Package Files (https://github.com/mewmew/uc/tree/master/ast)

ast.go (https://github.com/mewmew/uc/blob/master/ast/ast.go)

type BasicLit (https://github.com/mewmew/uc/blob/master/ast/ast.go#L176)

```
type BasicLit struct {
    // Basic literal type, one of the following.
    //
    //    token.CharLit
    //    token.IntLit
    Kind token (/github.com/mewmew/uc/token).Kind (/github.com/mewmew/uc/token#Kind)
    // Basic literal value; e.g. 123, 'a'.
    Val string (/builtin#string)
}
```

A BasicLit node represents a basic literal.

func (*BasicLit) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L242)

```
func (n *BasicLit) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type BinaryExpr (https://github.com/mewmew/uc/blob/master/ast/ast.go#L197)

```
type BinaryExpr struct {
    // First operand.
    X   Expr
    // Operator, one of the following.
    //    token.Add      // +
    //    token.Sub      // -
    //    token.Mul      // *
    //    token.Div      // /
    //    token.Lt       // <
    //    token.Gt       // >
    //    token.Le       // <=
    //    token.Ge       // >=
    //    token.Ne       // !=
    //    token.Eq       // ==
    //    token.Land     // &&
    //    token.Assign   // =
    Op  token (/github.com/mewmew/uc/token).Kind (/github.com/mewmew/uc/token#Kind)
    // Second operand.
    Y   Expr
}
```

An BinaryExpr node represents a binary expression; X op Y.

func (*BinaryExpr) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L245)

```
func (n *BinaryExpr) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type BlockItem (https://github.com/mewmew/uc/blob/master

/ast/ast.go#L143)

```
type BlockItem interface {
    Node
    // contains filtered or unexported methods
}
```

A BlockItem represents an item of a block statement, and has one of the following underlying types.

```
Decl
Stmt
```

type BlockStmt (https://github.com/mewmew/uc/blob/master/ast/ast.go#L99)

```
type BlockStmt struct {
    // List of block items contained within the block.
    Items []BlockItem
}
```

A BlockStmt node represents a block statement.

Examples.

```
{}
{ int x; x = 42; }
```

func (*BlockStmt) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L248)

```
func (n *BlockStmt) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type CallExpr (https://github.com/mewmew/uc/blob/master/ast/ast.go#L219)

```
type CallExpr struct {
    // Function name.
    Name *Ident
    // Function arguments.
    Args []Expr
}
```

A CallExpr node represents a call expression.

func (*CallExpr) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L251)

```
func (n *CallExpr) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type Decl (https://github.com/mewmew/uc/blob/master/ast/ast.go#L35)

```
type Decl interface {
    Node
    // contains filtered or unexported methods
}
```

A Decl node represents a declaration, and has one of the following underlying types.

```
*FuncDecl
*VarDecl
```

type EmptyStmt (https://github.com/mewmew/uc/blob/master/ast/ast.go#L105)

```
type EmptyStmt struct{}
```

An EmptyStmt node represents an empty statement (i.e. ";").

func (*EmptyStmt) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L254)

```
func (n *EmptyStmt) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type Expr (https://github.com/mewmew/uc/blob/master/ast/ast.go#L160)

```
type Expr interface {
    Node
    // contains filtered or unexported methods
}
```

An Expr node represents an expression, and has one of the following underlying types.

```
*BasicLit
*BinaryExpr
*CallExpr
*Ident
*IndexExpr
*ParenExpr
*UnaryExpr
```

type ExprStmt (https://github.com/mewmew/uc/blob/master/ast/ast.go#L108)

```
type ExprStmt struct {
    // Stand-alone expression.
    X Expr
}
```

An ExprStmt node represents a stand-alone expression in a statement list.

func (*ExprStmt) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L257)

```
func (n *ExprStmt) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type File (https://github.com/mewmew/uc/blob/master/ast/ast.go#L13)

```
type File struct {
    // Top-level declarations.
    Decls []Decl
}
```

A File represents a µC source file.

func (*File) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L260)

```
func (n *File) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type FuncDecl (https://github.com/mewmew/uc/blob/master/ast/ast.go#L48)

```
type FuncDecl struct {
    // Function signature.
    Type *types (/github.com/mewmew/uc/types).Func (/github.com/mewmew/uc/types#Func)
    // Function name.
    Name *Ident
    // Function body; or nil if function declaration (i.e. not function
    // definition).
    Body *BlockStmt
}
```

A FuncDecl node represents a function declaration.

Examples.

```
int add(int a, int b) { return a+b; }
int puts(char s[]);
```

func (*FuncDecl) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L263)

```
func (n *FuncDecl) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type Ident (https://github.com/mewmew/uc/blob/master/ast/ast.go#L170)

```
type Ident struct {
    // Identifier name.
    Name string (/builtin#string)
}
```

An Ident node represents an identifier.

func (*Ident) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L266)

```
func (n *Ident) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type IfStmt (https://github.com/mewmew/uc/blob/master/ast/ast.go#L114)

```
type IfStmt struct {
    // Condition.
    Cond Expr
    // True branch.
    Body Stmt
    // False branch; or nil if 1-way conditional.
    Else Stmt
}
```

An IfStmt node represents an if statement.

func (*IfStmt) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L269)

```
func (n *IfStmt) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type IndexExpr (https://github.com/mewmew/uc/blob/master/ast/ast.go#L233)

```
type IndexExpr struct {
    // Array name.
    Name *Ident
    // Array index.
    Index Expr
}
```

An IndexExpr node represents an array index expression.

func (*IndexExpr) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L272)

```
func (n *IndexExpr) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type Node (https://github.com/mewmew/uc/blob/master/ast/ast.go#L25)

```
type Node interface {
    // Start returns the start position of the node within the input stream.
    Start() int (/builtin#int)
}
```

A Node represents a node within the abstract syntax tree, and has one of the following underlying types.

```
*File
Decl
Stmt
Expr
```

type ParenExpr (https://github.com/mewmew/uc/blob/master/ast/ast.go#L227)

```
type ParenExpr struct {
    // Parenthesised expression.
    X Expr
}
```

A ParenExpr node represents a parenthesised expression.

func (*ParenExpr) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L275)

```
func (n *ParenExpr) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type ReturnStmt (https://github.com/mewmew/uc/blob/master/ast/ast.go#L132)

```
type ReturnStmt struct {
    // Result expression; or nil if void return.
    Result Expr
}
```

A ReturnStmt node represents a return statement.

func (*ReturnStmt) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L278)

```
func (n *ReturnStmt) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type Stmt (https://github.com/mewmew/uc/blob/master/ast/ast.go#L84)

```
type Stmt interface {
    Node
    // contains filtered or unexported methods
}
```

A Stmt node represents a statement, and has one of the following underlying types.

```
*BlockStmt
*EmptyStmt
*ExprStmt
*IfStmt
*ReturnStmt
*WhileStmt
```

type UnaryExpr (https://github.com/mewmew/uc/blob/master/ast/ast.go#L187)

```
type UnaryExpr struct {
    // Operator, one of the following.
    //    token.Sub   // -
    //    token.Not   // !
    Op  token (/github.com/mewmew/uc/token).Kind (/github.com/mewmew/uc/token#Kind)
    // Operand.
    X   Expr
}
```

An UnaryExpr node represents an unary expression; op X.

func (*UnaryExpr) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L281)

```
func (n *UnaryExpr) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type VarDecl (https://github.com/mewmew/uc/blob/master

/ast/ast.go#L64)

```
type VarDecl struct {
    // Variable type.
    Type types (/github.com/mewmew/uc/types).Type (/github.com/mewmew/uc/types#Type)
    // Variable name.
    Name *Ident
    // Variable value expression; or nil if variable declaration (i.e. not
    // variable definition).
    Val Expr
}
```

A VarDecl node represents a variable declaration.

Examples.

```
int x;
char buf[128];
```

func (*VarDecl) Start (https://github.com/mewmew/uc/blob/master /ast/ast.go#L284)

```
func (n *VarDecl) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

type WhileStmt (https://github.com/mewmew/uc/blob/master /ast/ast.go#L124)

```
type WhileStmt struct {
    // Condition.
    Cond Expr
    // Loop body.
    Body Stmt
}
```

A WhileStmt node represents a while statement.

func (*WhileStmt) Start (https://github.com/mewmew/uc/blob/master/ast/ast.go#L287)

```
func (n *WhileStmt) Start() int (/builtin#int)
```

Start returns the start position of the node within the input stream.

Directories

Path	Synopsis
astx (/github.com/mewmew/uc/ast/astx)	Package astx implements utility functions for generating abstract syntax trees.