# 概述

AST（抽象语法树），可用来解析golang源文件，或者源码字符串。

AST所有节点都实现了ast.Node接口。

|  |
| --- |
| // All node types implement the Node interface.  type Node interface {  Pos() token.Pos // position of first character belonging to the node  End() token.Pos // position of first character immediately after the node  } |

AST节点主体分为3类（实际4类）：

* Expressions and type nodes
* statement nodes
* declaration nodes
* Spec Node(constant, type, or variable declaration)

|  |
| --- |
| type Node interface {  Pos() token.Pos // position of first character belonging to the node  End() token.Pos // position of first character immediately after the node  }  // All expression nodes implement the Expr interface.  type Expr interface {  Node  exprNode()  }  // All statement nodes implement the Stmt interface.  type Stmt interface {  Node  stmtNode()  }  // All declaration nodes implement the Decl interface.  type Decl interface {  Node  declNode()  } |

# 基础结构说明

## Expression & Types Node(都划分成Expr接口)

BadExpr 用来表示错误表达式的占位符

Ident 比如包名,函数名,变量名

Ellipsis 省略号表达式,比如参数列表的最后一个可以写成arg...

BasicLit 基本字面值,数字或者字符串

FuncLit 函数定义

CompositeLit 构造类型,比如{1,2,3,4}

ParenExpr 括号表达式,被括号包裹的表达式

SelectorExpr 选择结构,类似于a.b的结构

IndexExpr 下标结构,类似这样的结构 expr[expr]

SliceExpr 切片表达式,类似这样 expr[low:mid:high]

TypeAssertExpr 类型断言类似于 X.(type)

CallExpr 调用类型,类似于 expr()

StarExpr 指针表达式,类似于 \*X

UnaryExpr 一元表达式

BinaryExpr 二元表达式

KeyValueExp 键值表达式 key:value

ArrayType 数组类型

StructType 结构体类型

FuncType 函数类型

InterfaceType 接口类型

MapType map类型

ChanType 管道类型

## Statements Node

BadStmt 错误的语句

DeclStmt 在语句列表里的申明

EmptyStmt 空语句

LabeledStmt 标签语句类似于 indent:stmt

ExprStmt 包含单独的表达式语句

SendStmt chan发送语句

IncDecStmt 自增或者自减语句

AssignStmt 赋值语句

GoStmt Go语句

DeferStmt 延迟语句

ReturnStmt return 语句

BranchStmt 分支语句 例如break continue

BlockStmt 块语句 {} 包裹

IfStmt If 语句

CaseClause case 语句

SwitchStmt switch 语句

TypeSwitchStmt 类型switch 语句 switch x:=y.(type)

CommClause 发送或者接受的case语句,类似于 case x <-:

SelectStmt select 语句

ForStmt for 语句

RangeStmt range 语句

## Declarations Node

BadDecl 错误申明

GenDecl 一般申明(和Spec相关, 表示import, const，type或变量声明)

FuncDecl 函数申明

## Spec Node

Import Spec 单独import语句

Value Spec 常量或变量声明

Type Spec type声明

## Common Node（不属于以上四类的）

Comment 表示一行注释 // 或者 / /

CommentGroup 表示多行注释

Field 表示结构体中的一个定义或者变量,或者函数签名当中的参数或者返回值

FieldList 表示以”{}”或者”()”包围的Filed列表

File 代表一个源文件节点,包含了顶级元素.

Package 代表一个包,包含了很多文件.