package types

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

Index

Package Files

basickind_string.go types.go

func Equal

```
func Equal(t, u Type) bool
```

Equal reports whether t and u are of equal type.

func IsVoid

```
func IsVoid(t Type) bool
```

IsVoid reports whether the given type is a void type.

type Array

```
type Array struct {
    // Element type.
    Elem Type
    // Array length.
    Len int
}
```

An Array represents an array type.

Examples.

```
int[]
char[128]
```

func (*Array) Equal

```
func (t *Array) Equal(u Type) bool
```

Equal reports whether t and u are of equal type.

func (*Array) String

```
func (t *Array) String() string
```

type Basic

```
type Basic struct {
    // Kind of basic type.
    Kind BasicKind
}
```

A Basic represents a basic type.

Examples.

```
char
int
```

func (*Basic) Equal

```
func (t *Basic) Equal(u Type) bool
```

Equal reports whether t and u are of equal type.

func (*Basic) IsNumerical

```
func (t *Basic) IsNumerical() bool
```

IsNumerical reports whether the given type is numerical.

func (*Basic) String

```
func (t *Basic) String() string
```

type BasicKind

```
type BasicKind int
```

BasicKind describes the kind of basic type.

```
const (
    Invalid BasicKind = iota // invalid type

    Char // "char"
    Int  // "int"
    Void // "void"
)
```

Basic type.

func (BasicKind) String

```
func (kind BasicKind) String() string
```

type Field

```
type Field struct {
    // Field type.
    Type Type
    // Field name; or empty.
    Name string
}
```

A Field represents a field declaration in a struct type, or a parameter declaration in a function signature.

Examples.

```
char
int a
```

func (*Field) String

```
func (field *Field) String() string
```

type Func

```
type Func struct {
    // Return type.
    Result Type
    // Function parameter types; or nil if void parameter.
    Params []*Field
}
```

A Func represents a function signature.

Examples.

```
int(void)
int(int a, int b)
```

func (*Func) Equal

```
func (t *Func) Equal(u Type) bool
```

Equal reports whether t and u are of equal type.

func (*Func) String

```
func (t *Func) String() string
```

type Numerical

```
type Numerical interface {
    // IsNumerical reports whether the given type is numerical.
    IsNumerical() bool
}
```

A Numerical type is numerical if so specified by IsNumerical.

type Type

```
type Type interface {
    // Equal reports whether t and u are of equal type.
    Equal(u Type) bool
    fmt.Stringer
}
```

A Type represents a type of µC, and has one of the following underlying types.

```
*Basic
*Array
*Func
```