expvar

package | standard library

Version: go1.20.1 Latest | Published: Feb 14, 2023 | License: BSD-3-Clause | Imports: 12 | Imported by: 9,974

Details Valid go.mod file ? Redistributable license ? Tagged version ? Stable version ?

Learn more

Repository cs.opensource.google/go/go

Links Report a Vulnerability

Documentation

Documentation

Overview

Package expvar provides a standardized interface to public variables, such as operation counters in servers. It exposes these variables via HTTP at /debug/vars in JSON format.

Operations to set or modify these public variables are atomic.

In addition to adding the HTTP handler, this package registers the following variables:

```
cmdline   os.Args
memstats  runtime.Memstats
```

The package is sometimes only imported for the side effect of registering its HTTP handler and the above variables. To use it this way, link this package into your program:

```
import _ "expvar"
```

Index

func (v *Float) Set(value float64)
func (v *Float) String() string
func (v *Float) Value() float64
type Func
func (f Func) String() string
func (f Func) Value() any
type Int
func NewInt(name string) *Int
func (v *Int) Add(delta int64)
func (v *Int) Set(value int64)
func (v *Int) String() string
func (v *Int) Value() int64
type KeyValue
type Map
func NewMap(name string) *Map
func (v *Map) Add(key string, delta int64)
func (v *Map) AddFloat(key string, delta float64)
func (v *Map) Delete(key string)
func (v *Map) Do(f func(KeyValue))
func (v *Map) Get(key string) Var
func (v *Map) Init() *Map
func (v *Map) Set(key string, av Var)
func (v *Map) String() string
type String
func NewString(name string) *String
func (v *String) Set(value string)
func (v *String) String() string
func (v *String) Value() string
type Var
func Get(name string) Var

Constants

This section is empty.

Variables

This section is empty.

Functions

func Do

```
func Do(f func(KeyValue))
```

Do calls f for each exported variable. The global variable map is locked during the iteration, but existing entries may be concurrently updated.

func Handler

added in go1.8

```
func Handler() http.Handler
```

Handler returns the expvar HTTP Handler.

This is only needed to install the handler in a non-standard location.

func Publish

```
func Publish(name string, v Var)
```

Publish declares a named exported variable. This should be called from a package's init function when it creates its Vars. If the name is already registered then this will log.Panic.

Types

type Float

```
type Float struct {
    // contains filtered or unexported fields
}
```

Float is a 64-bit float variable that satisfies the Var interface.

func NewFloat

```
func NewFloat(name string) *Float
```

func (*Float) Add

```
func (v *Float) Add(delta float64)
```

Add adds delta to v.

func (*Float) Set

```
func (v *Float) Set(value float64)
```

Set sets v to value.

func (*Float) String

```
func (v *Float) String() string
```

func (*Float) Value

added in go1.8

```
func (v *Float) Value() float64
```

type Func

```
type Func func() any
```

Func implements Var by calling the function and formatting the returned value using JSON.

func (Func) String

```
func (f Func) String() string
```

func (Func) Value

added in go1.8

```
func (f Func) Value() any
```

type Int

```
type Int struct {
    // contains filtered or unexported fields
}
```

Int is a 64-bit integer variable that satisfies the Var interface.

func NewInt

```
func NewInt(name string) *Int
```

func (*Int) Add

```
func (v *Int) Add(delta int64)
```

func (*Int) Set

```
func (v *Int) Set(value int64)
```

func (*Int) String

```
func (v *Int) String() string
```

func (*Int) Value

added in go1.8

```
func (v *Int) Value() int64
```

type KeyValue

```
type KeyValue struct {
    Key   string
    Value Var
}
```

KeyValue represents a single entry in a Map.

type Map

```
type Map struct {
    // contains filtered or unexported fields
}
```

Map is a string-to-Var map variable that satisfies the Var interface.

func NewMap

```
func NewMap(name string) *Map
```

func (*Map) Add

```
func (v *Map) Add(key string, delta int64)
```

Add adds delta to the *Int value stored under the given map key.

func (*Map) AddFloat

```
func (v *Map) AddFloat(key string, delta float64)
```

AddFloat adds delta to the *Float value stored under the given map key.

func (*Map) Delete

added in go1.12

```
func (v *Map) Delete(key string)
```

Delete deletes the given key from the map.

func (*Map) Do

```
func (v *Map) Do(f func(KeyValue))
```

Do calls f for each entry in the map. The map is locked during the iteration, but existing entries may be concurrently updated.

func (*Map) Get

```
func (v *Map) Get(key string) Var
```

func (*Map) Init

```
func (v *Map) Init() *Map
```

Init removes all keys from the map.

func (*Map) Set

```
func (v *Map) Set(key string, av Var)
```

func (*Map) String

```
func (v *Map) String() string
```

type String

```
type String struct {
    // contains filtered or unexported fields
}
```

String is a string variable, and satisfies the Var interface.

func NewString

```
func NewString(name string) *String
```

func (*String) Set

```
func (v *String) Set(value string)
```

func (*String) String

```
func (v *String) String() string
```

String implements the Var interface. To get the unquoted string use Value.

func (*String) Value

added in go1.8

```
func (v *String) Value() string
```

type Var

```
type Var interface {
    // String returns a valid JSON value for the variable.
    // Types with String methods that do not return valid JSON
    // (such as time.Time) must not be used as a Var.
```

```
    String() string
}
```

Var is an abstract type for all exported variables.

func Get

```
func Get(name string) Var
```

Get retrieves a named exported variable. It returns nil if the name has not been registered.

Source Files

View all

expvar.go