Objx

build passing | go report A+ | maintainability A | test coverage 99%

used by 2.7k projects | GO reference

Objx - Go package for dealing with maps, slices, JSON and other data.

Get started:

- Install Objx with one line of code, or update it with another
- Check out the API Documentation http://godoc.org/github.com/stretchr/objx

Overview

Objx provides the `objx.Map` type, which is a `map[string]interface{}` that exposes a powerful `Get` method (among others) that allows you to easily and quickly get access to data within the map, without having to worry too much about type assertions, missing data, default values etc.

Pattern

Objx uses a preditable pattern to make access data from within `map[string]interface{}` easy. Call one of the `objx.` functions to create your `objx.Map` to get going:

```
m, err := objx.FromJSON(json)
```

NOTE: Any methods or functions with the `Must` prefix will panic if something goes wrong, the rest will be optimistic and try to figure things out without panicking.

Use `Get` to access the value you're interested in. You can use dot and array notation too:

```
 m.Get("places[0].latlng")
```

Once you have sought the `Value` you're interested in, you can use the `Is*` methods to determine its type.

```
 if m.Get("code").IsStr() { // Your code... }
```

Or you can just assume the type, and use one of the strong type methods to extract the real value:

```
m.Get("code").Int()
```

If there's no value there (or if it's the wrong type) then a default value will be returned, or you can be explicit about the default value.

```
 Get("code").Int(-1)
```

If you're dealing with a slice of data as a value, Objx provides many useful methods for iterating, manipulating and selecting that data. You can find out more by exploring the index below.

Reading data

A simple example of how to use Objx:

```
// Use MustFromJSON to make an objx.Map from some JSON
m := objx.MustFromJSON(`{"name": "Mat", "age": 30}`)

// Get the details
name := m.Get("name").Str()
age := m.Get("age").Int()

// Get their nickname (or use their name if they don't have one)
nickname := m.Get("nickname").Str(name)
```

Ranging

Since `objx.Map` is a `map[string]interface{}` you can treat it as such. For example, to `range` the data, do what you would expect:

```
m := objx.MustFromJSON(json)
for key, value := range m {
  // Your code...
}
```

Installation

To install Objx, use go get:

```
go get github.com/stretchr/objx
```

Staying up to date

To update Objx to the latest version, run:

```
go get -u github.com/stretchr/objx
```

Supported go versions

We support the lastest three major Go versions, which are 1.10, 1.11 and 1.12 at the moment.

Contributing

Please feel free to submit issues, fork the repository and send pull requests!