
Go-Diagrams

Fast and easy application diagrams

Go-Diagrams is a loose port of diagrams.

Contents

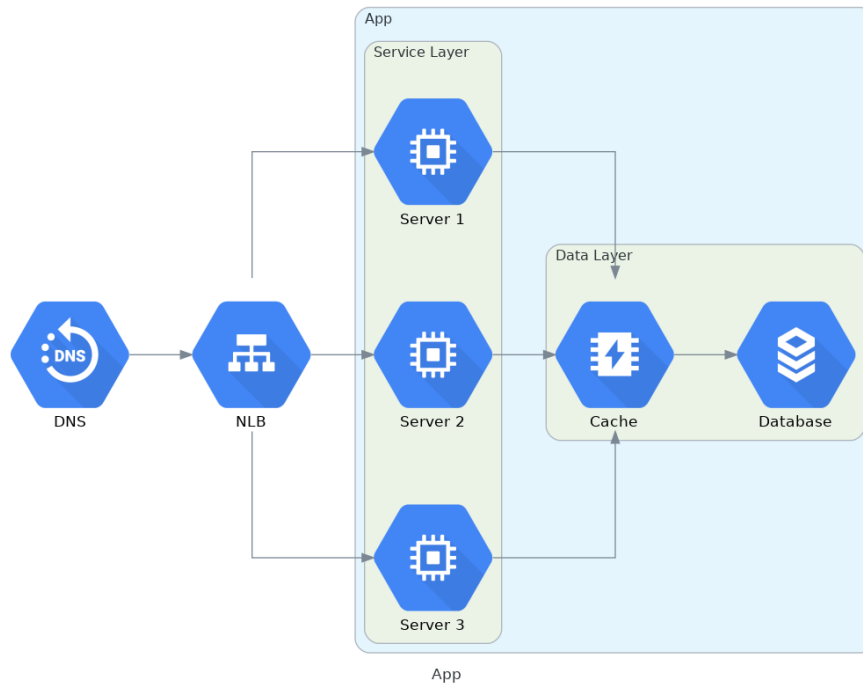
- Features
- Usage

Features

Turn this:

```
1 d, err := diagram.New(diagram.Filename("app"), diagram.Label("App"),
2   diagram.Direction("LR"))
3 if err != nil {
4     log.Fatal(err)
5 }
6 dns := gcp.Network.Dns(diagram.NodeLabel("DNS"))
7 lb := gcp.Network.LoadBalancing(diagram.NodeLabel("NLB"))
8 cache := gcp.Database.Memorystore(diagram.NodeLabel("Cache"))
9 db := gcp.Database.Sql(diagram.NodeLabel("Database"))
10
11 dc := diagram.NewGroup("GCP")
12 dc.NewGroup("services").
13     Label("Service Layer").
14     Add(
15         gcp.Compute.ComputeEngine(diagram.NodeLabel("Server 1")),
16         gcp.Compute.ComputeEngine(diagram.NodeLabel("Server 2")),
17         gcp.Compute.ComputeEngine(diagram.NodeLabel("Server 3")),
18     ).
19     ConnectAllFrom(lb.ID(), diagram.Forward()).
20     ConnectAllTo(cache.ID(), diagram.Forward())
21
22 dc.NewGroup("data").Label("Data Layer").Add(cache, db).Connect(cache,
23     db)
24 d.Connect(dns, lb, diagram.Forward()).Group(dc)
25
26 if err := d.Render(); err != nil {
27     log.Fatal(err)
28 }
```

Into this:



Usage

```
1 go get github.com/blushft/go-diagrams
```

Create a diagram:

```
1 d, err := diagram.New(diagram.Label("my-diagram"), diagram.Filename("
2   diagram"))
3 if err != nil {
4     log.Fatal(err)
5 }
6 fw := generic.Network.Firewall().Label("fw")
7 sw := generic.Network.Switch().Label("sw")
8
```

```
9 d.Connect(fw, sw)
```

Render the output:

```
1 if err := d.Render(); err != nil {  
2     log.Fatal(err)  
3 }
```

Go-Diagrams will create a folder in the current working directory with the graphviz DOT file and any image assets.

Create an output image with any graphviz compatible renderer:

```
1 dot -Tpng diagram.dot > diagram.png
```