THE DEVELOPER'S CONFERENCE

OpenCensus: Minha experiência em um projeto open source em Go

Vitor Casadei
Mestre em Ciência da Computação

THE DEVELOPER'S CONFERENCE

VITOR CASADEI

ENGENHEIRO DE SOFTWARE @ CESAR

MESTRE EM CIÊNCIA DA COMPUTAÇÃO

github.com/vcasadei

br.linkedin.com/in/vitorcasadei

vitor.casadei@gmail.com


C.E.S.A.R
centro de estudos e sistemas
avançados do recife
CESAR


gente
negócios
tecnologia
Para nós, inovação sustentável acontece na intersecção das dimensões gente, negócios e tecnologia

THE DEVELOPER'S CONFERENCE

OpenCensus

Easily collect telemetry like metrics and distributed traces from your services

OpenCensus

THE DEVELOPER'S CONFERENCE

Partners & Contributors

OpenCensus

THE DEVELOPER'S CONFERENCE

BACKEND	C#	C++	ERLANG	GO	JAVA	NODE.JS	PHP	PYTHON	RUBY
AWS X-Ray	-	–	–	T		–	–	–	–
Azure Monitor	T S	–	–	T	T	–	–	T	–
Datadog	–	–	T S	T S	T	–	–	–	–
Elasticsearch	-	–	–	–	T	–	–	–	–
Honeycomb	–	–	–	T	–	–	–	–	–
Instana	–	–	–	–	T	T	–	–	–
Jaeger	–	–	–	T	T	T	–	T	–
Prometheus	S	S	S	S	S	S	–	S	–
SignalFx	T	T	T	T S	T S	T	–	T	–
Stackdriver	T	T S	T	T S	T S	T S	–	T S	–
Wavefront	–	–	–	T S	–	–	–	–	–
Zipkin	T	T	T	T	T	T	–	T	–

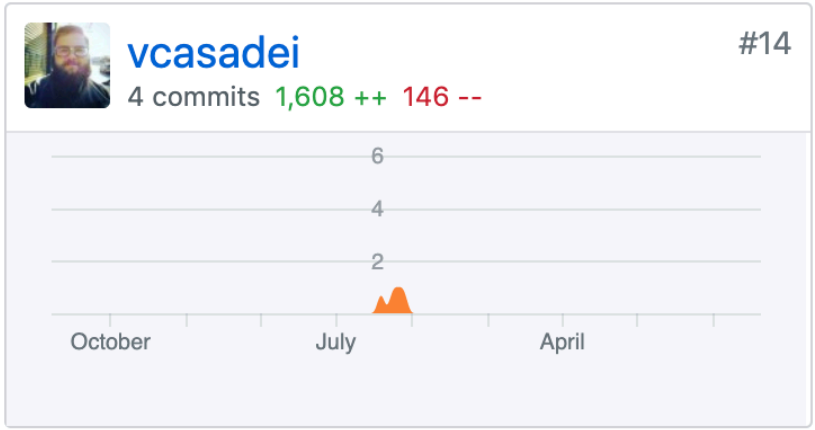
OpenCensus

THE DEVELOPER'S CONFERENCE

OpenCensus Libraries for Go

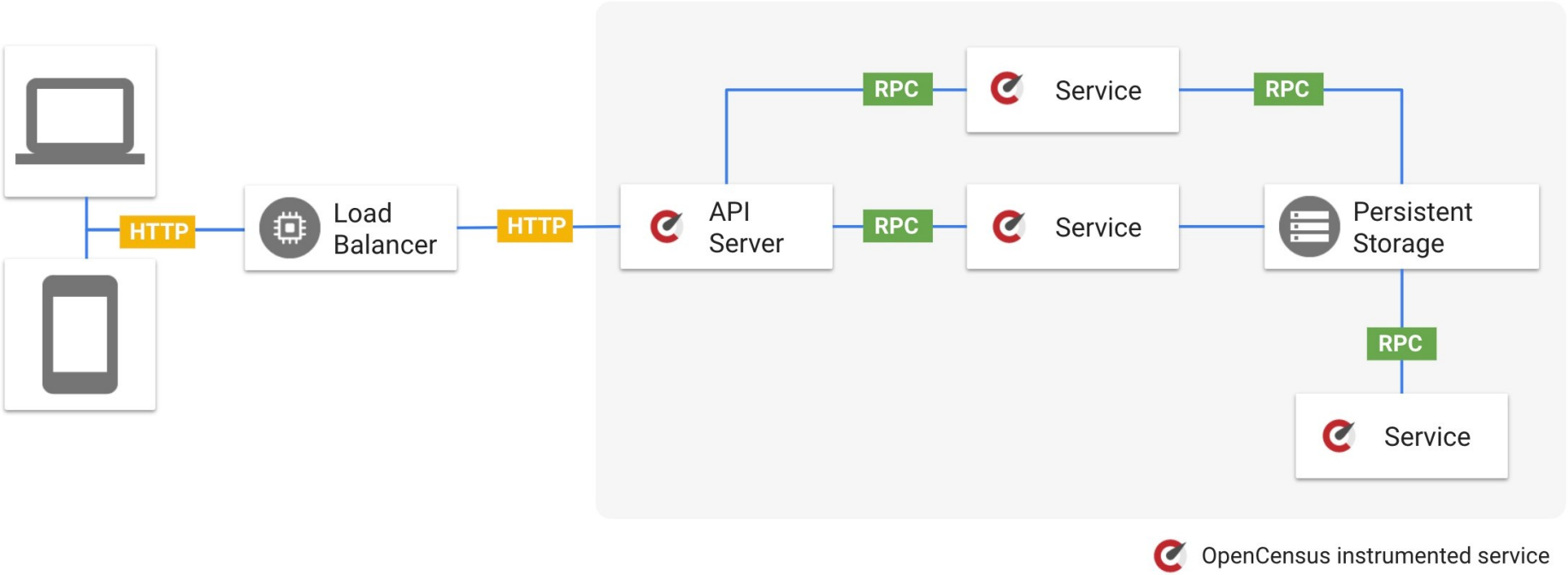
build passing | build passing | godoc reference | chat on gitter

OpenCensus Go is a Go implementation of OpenCensus, a toolkit for collecting application performance and behavior monitoring data. Currently it consists of three major components: tags, stats and tracing.



OpenCensus

THE DEVELOPER'S CONFERENCE



OpenCensus

THE DEVELOPER'S CONFERENCE

Tags

```
ctx, err := tag.New(ctx,
        tag.Insert(osKey, "macOS-10.12.5"),
        tag.Upsert(userIDKey, "cde36753ed"),
)
if err != nil {
        log.Fatal(err)
}
```

OpenCensus

THE DEVELOPER'S CONFERENCE

Stats

```
stats.Record(ctx, videoSize.M(102478))
```

OpenCensus

THE DEVELOPER'S CONFERENCE

Views

```
if err := view.Register(&view.View{
        Name:        "example.com/video_size_distribution",
        Description: "distribution of processed video size over time",
        Measure:     videoSize,
        Aggregation: view.Distribution(1<<32, 2<<32, 3<<32),
}); err != nil {
        log.Fatalf("Failed to register view: %v", err)
}
```

OpenCensus

THE DEVELOPER'S CONFERENCE

Grafana Visualization

Aprendizados

THE DEVELOPER'S CONFERENCE

- Handle Errors first!

```
type Gopher struct {
    Name     string
    AgeYears int
}
```

```
func (g *Gopher) WriteTo(w io.Writer) (size int64, err error) {
    err = binary.Write(w, binary.LittleEndian, int32(len(g.Name)))
    if err == nil {
        size += 4
        var n int
        n, err = w.Write([]byte(g.Name))
        size += int64(n)
        if err == nil {
            err = binary.Write(w, binary.LittleEndian, int64(g.AgeYears))
            if err == nil {
                size += 4
            }
            return
        }
        return
    }
    return
}
```

Aprendizados

THE DEVELOPER'S CONFERENCE

```
func (g *Gopher) WriteTo(w io.Writer) (size int64, err error) {
    err = binary.Write(w, binary.LittleEndian, int32(len(g.Name)))
    if err != nil {
        return
    }
    size += 4
    n, err := w.Write([]byte(g.Name))
    size += int64(n)
    if err != nil {
        return
    }
    err = binary.Write(w, binary.LittleEndian, int64(g.AgeYears))
    if err == nil {
        size += 4
    }
    return
}
```

Aprendizados

THE DEVELOPER'S CONFERENCE

- Avoid Repetition and use binWriter

```
func (g *Gopher) WriteTo(w io.Writer) (int64, error) {
    bw := &binWriter{w: w}
    bw.Write(int32(len(g.Name)))
    bw.Write([]byte(g.Name))
    bw.Write(int64(g.AgeYears))
    return bw.size, bw.err
}
```

Aprendizados

THE DEVELOPER'S CONFERENCE

- Switch to handle special type cases

```
func (g *Gopher) WriteTo(w io.Writer) (int64, error) {
    bw := &binWriter{w: w}
    bw.Write(g.Name)
    bw.Write(g.AgeYears)
    return bw.size, bw.err
}
```

Aprendizados

THE DEVELOPER'S CONFERENCE

```
func (w *binWriter) Write(v interface{}) {
    if w.err != nil {
        return
    }
    switch x := v.(type) {
    case string:
        w.Write(int32(len(x)))
        w.Write([]byte(x))
    case int:
        w.Write(int64(x))
    default:
        if w.err = binary.Write(w.w, binary.LittleEndian, v); w.err == nil {
            w.size += int64(binary.Size(v))
        }
    }
}
```

Aprendizados

THE DEVELOPER'S CONFERENCE

- Import only what you need!

```
import (
    "fmt"
    "io"
    "log"

    "golang.org/x/net/websocket"
)
```

Aprendizados

THE DEVELOPER'S CONFERENCE

- Avoid Concurrency when possible!

```
func doConcurrently(job string, err chan error) {
    go func() {
        fmt.Println("doing job", job)
        time.Sleep(1 * time.Second)
        err <- errors.New("something went wrong!")
    }()
}
```

```
func do(job string) error {
    fmt.Println("doing job", job)
    time.Sleep(1 * time.Second)
    return errors.New("something went wrong!")
}
```

Aprendizados

THE DEVELOPER'S CONFERENCE

- Avoid Concurrency when possible!
  - Recommended:

https://vimeo.com/49718712

Aprendizados

- Correct order of Enum

```
1  type Status uint32
2
3  const (
4          StatusOpen Status = iota
5          StatusClosed
6          StatusUnknown
7  )
```

StatusOpen = 0
StatusClosed = 1
StatusUnknown = 2

Aprendizados

- Correct order of Enum

```
{
  "Id": 1234,
  "Timestamp": 1563362390,
  "Status": 0
}
```

StatusOpen = 0
StatusClosed = 1
StatusUnknown = 2

Aprendizados

- Correct order of Enum

```
1  {
2    "Id": 1235,
3    "Timestamp": 1563362390
4  }
```

StatusOpen = 0
StatusClosed = 1
StatusUnknown = 2

Aprendizados

THE DEVELOPER'S CONFERENCE

- Correct order of Enum

```
type Status uint32

const (
        StatusUnknown Status = iota
        StatusOpen
        StatusClosed
)
```

```
{
  "Id": 1235,
  "Timestamp": 1563362390
}
```

StatusUnknown = 0
StatusOpen = 1
StatusClosed = 2

Aprendizados

- Create a meaningful stack of erros

```go
func postHandler(customer Customer) Status {
    err := insert(customer.Contract)
    if err != nil {
        log.WithError(err).Errorf("unable to serve HTTP POST request for customer %s", customer.ID)
        return Status{ok: false}
    }
    return Status{ok: true}
}
```

Aprendizados

- Create a meaningful stack of erros

```
1 func insert(contract Contract) error {
2     err := dbQuery(contract)
3     if err != nil {
4         return errors.Wrapf(err, "unable to insert customer contract %s", contract.ID)
5     }
6     return nil
7 }
```

Aprendizados

THE DEVELOPER'S CONFERENCE

- Create a meaningful stack of erros

```
1 func dbQuery(contract Contract) error {
2     // Do something then fail
3     return errors.New("unable to commit transaction")
4 }
```

unable to serve HTTP POST request for customer 1234
 |_ unable to insert customer contract abcd
    |_ unable to commit transaction

Aprendizados

- Use correct slice sizes (when possible)!

```
1 // Common way to init a slice
2 var bars []Bar
3 bars := make([]Bar, 0)
4
```

High cost on append!

Aprendizados

THE DEVELOPER'S CONFERENCE

- Use correct slice sizes (when possible)!

```
5  // Init with predefined length
6  func convert(foos []Foo) []Bar {
7      bars := make([]Bar, len(foos))
8      for i, foo := range foos {
9          bars[i] = fooToBar(foo)
10     }
11     return bars
12 }
13
14 // Init with pre-allocated length
15 func convert(foos []Foo) []Bar {
16     bars := make([]Bar, 0, len(foos))
17     for _, foo := range foos {
18         bars = append(bars, fooToBar(foo))
19     }
20     return bars
21 }
```

Aprendizados

THE DEVELOPER'S CONFERENCE

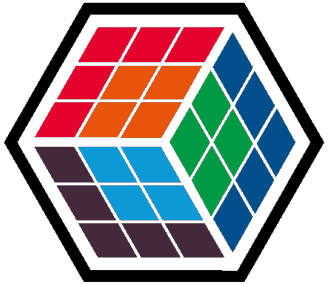
- Use correct slice sizes (when possible)!

```
5  // Init with predefined length
6  func convert(foos []Foo) []Bar {
7      bars := make([]Bar, len(foos))
8      for i, foo := range foos {
9          bars[i] = fooToBar(foo)
10     }
11     return bars
12 }
13
14 // Init with pre-allocated length
15 func convert(foos []Foo) []Bar {
16     bars := make([]Bar, 0, len(foos))
17     for _, foo := range foos {
18         bars = append(bars, fooToBar(foo))
19     }
20     return bars
21 }
```

Mais Referências

THE DEVELOPER'S CONFERENCE

- Go PerfBook - https://github.com/dgryski/go-perfbook
- Effective Go - https://golang.org/doc/effective_go.html
- Go Performance Observations - https://hashrocket.com/blog/posts/go-performance-observations (binary is faster!)
- Writing High Performance Go - https://go-talks.appspot.com/github.com/davecheney/presentations/writing-high-performance-go.slide#1



THE DEVELOPER'S CONFERENCE

THE DEVELOPER'S CONFERENCE

VITOR CASADEI

ENGENHEIRO DE SOFTWARE @ CESAR

MESTRE EM CIÊNCIA DA COMPUTAÇÃO

github.com/vcasadei

br.linkedin.com/in/vitorcasadei

vitor.casadei@gmail.com