Experiences Building InfluxDB in Go

Paul Dix
paul@influxdb.com
@pauldix
Who am I?

informs my experience working with a language like Go
co-founder, CEO, programmer

InfluxDB
Languages worked with Professionally in order

- VBScript
- Delphi
- C# and VB.NET
- Ruby & Javascript
- Java
- C (only a little)
- Ruby & Javascript
- Scala
- Go
Mix of static and dynamic, but most time in dynamic languages
Open source time series database written in Go

InfluxDB
What’s a time series?
Stock trades and quotes
Analytics
Events

64.242.08.10 - - [07/Mar/2004:16:05:49 -0800] "GET /wiki/edit/Main/Double_bounce_sender?topicparent=Main.ConfigurationVariables HTTP/1.1" 401 12846
64.242.88.10 - - [07/Mar/2004:16:04:51 -0800] "GET /wiki/edit/Path/WelcomeTemplate?rev=1.3&rev2=1.2 HTTP/1.1" 200 4523
64.242.88.10 - - [07/Mar/2004:16:10:02 -0800] "GET /mailman/listinfo/hsdvision HTTP/1.1" 200 6291
64.242.88.10 - - [07/Mar/2004:16:11:58 -0800] "GET /wiki/edit/Path/TWikiWikiSyntax HTTP/1.1" 200 7352
64.242.88.10 - - [07/Mar/2004:16:20:55 -0800] "GET /wiki/edit/Path/TOCandPostFix HTTP/1.1" 200 5253
64.242.88.10 - - [07/Mar/2004:16:23:32 -0800] "GET /wiki/edit/Path/AppendixFilesystemTemplate?opsmore=1.12&n2=1.12 HTTP/1.1" 200 11382
64.242.88.10 - - [07/Mar/2004:16:24:16 -0800] "GET /wiki/edit/Path/PeterThoeny HTTP/1.1" 200 4924
64.242.88.10 - - [07/Mar/2004:16:30:29 -0800] "GET /wiki/edit/Path/Main/OfficeLocations HTTP/1.1" 401 12851
64.242.88.10 - - [07/Mar/2004:16:31:48 -0800] "GET /wiki/edit/Path/TWiki/WebTopicEditTemplate HTTP/1.1" 200 3732
64.242.88.10 - - [07/Mar/2004:16:32:50 -0800] "GET /wiki/edit/Path/Main/Redirect . HTTP/1.1" 200 40520
64.242.88.10 - - [07/Mar/2004:16:35:19 -0800] "GET /mailman/listinfo/business HTTP/1.1" 200 6379
64.242.88.10 - - [07/Mar/2004:16:36:22 -0800] "GET /wiki/edit/Path/Main/WebIndex?rev=1.2&rev2=1.1 HTTP/1.1" 200 46373
64.242.88.10 - - [07/Mar/2004:16:37:27 -0800] "GET /wiki/edit/Path/TWiki/Don'tNotify HTTP/1.1" 200 4140
64.242.88.10 - - [07/Mar/2004:16:39:24 -0800] "GET /wiki/edit/Path/Main/TokyoOffice HTTP/1.1" 200 3853
64.242.88.10 - - [07/Mar/2004:16:43:54 -0800] "GET /wiki/edit/Path/Main/mkemnlc HTTP/1.1" 200 3686
64.242.88.10 - - [07/Mar/2004:16:45:56 -0800] "GET /wiki/edit/Path/Main/PostfixCommands HTTP/1.1" 401 12846
64.242.88.10 - - [07/Mar/2004:16:47:12 -0800] "GET /robots.txt HTTP/1.1" 200 68
64.242.88.10 - - [07/Mar/2004:16:47:46 -0800] "GET /wiki/edit/Path/Main/HnkoFirst?rev=1.5&rev2=1.4 HTTP/1.1" 200 5724
64.242.88.10 - - [07/Mar/2004:16:49:04 -0800] "GET /wiki/edit/Path/Main/TWikiGroup?rev=1.2 HTTP/1.1" 200 5162
64.242.88.10 - - [07/Mar/2004:16:50:54 -0800] "GET /wiki/edit/Path/Main/ConfigurationVariables HTTP/1.1" 200 59679
64.242.88.10 - - [07/Mar/2004:16:52:35 -0800] "GET /wiki/edit/Path/Main/ConfigurationVariables HTTP/1.1" 401 12851
64.242.88.10 - - [07/Mar/2004:16:53:46 -0800] "GET /wiki/edit/Path/TWiki/TWikiRegistration HTTP/1.1" 200 34395
64.242.88.10 - - [07/Mar/2004:16:54:55 -0800] "GET /wiki/edit/Path/Main/NicholasLee HTTP/1.1" 200 7235
64.242.88.10 - - [07/Mar/2004:16:56:39 -0800] "GET /wiki/edit/Path/Main/Flash_service_name?topicparent=Main.ConfigurationVariables HTTP/1.1" 200 8545
64.242.88.10 - - [07/Mar/2004:16:58:54 -0800] "GET /mailman/listinfo/administration HTTP/1.1" 200 6459
lordgun.org - - [07/Mar/2004:17:01:53 -0800] "GET /raasr.html HTTP/1.1" 200 2869
64.242.88.10 - - [07/Mar/2004:17:09:01 -0800] "GET /wiki/edit/search/Main/SearchResult?scope=twiki&text=twiki&onsearch=0&is20=1&isachop=1-2a-2z HTTP/1.1" 200 4284
64.242.88.10 - - [07/Mar/2004:17:10:20 -0800] "GET /wiki/edit/ops/TWiki/TextFormattingRules?template=owsmore&n1=1.37&n2=1.37 HTTP/1.1" 200 11400
64.242.88.10 - - [07/Mar/2004:17:13:50 -0800] "GET /wiki/edit/Path/DefaultPlugin?0=1078889936 HTTP/1.1" 401 12846
64.242.88.10 - - [07/Mar/2004:17:16:00 -0800] "GET /wiki/edit/Path/Main/Scope?topic=Public&n2=1&n2=p HTTP/1.1" 200 3675
l1036.inkomisearch.com - - [07/Mar/2004:17:18:36 -0800] "GET /robots.txt HTTP/1.0" 200 68
Sensor data
Two kinds of time series data...
Regular time series

Samples at regular intervals
Irregular time series

Events whenever they come in
Why’d we pick Go?
Some project requirements

- Self contained binary install
Some project requirements

• Self contained binary install

• Performance
previous experience with it
faster development than working with C/C++
growing community
simplicity of the language

significant advantage for picking up new programmers and contributors
InfluxDB Project Stats

- First Commit - September 26th, 2013
- 176 Contributors
- 68,000 LOC
Team background

- Java
- Scala
- Python
- Ruby
- C++
What has been great
Static typing

can’t believe I’m saying this
Community
Performance
What surprised us
GC hasn’t been a problem
Contributors with no previous Go experience

simplicity of the language wins again
Haven’t really missed generics
except when I do...
// Sort methods

func (a Values) Len() int          { return len(a) }

func (a Values) Swap(i, j int)     { a[i], a[j] = a[j], a[i] }

func (a Values) Less(i, j int) bool {
    return a[i].Time().UnixNano() < a[j].Time().UnixNano()
}
Duplicate code for each data type

```
select percentile(90, value) from cpu
where time > now() - 1d
group by time(10m)
```
// Iterator represents an iterator over a series.

type Iterator interface {
    SeekTo(seek int64) (key int64, value interface{})
    Next() (key int64, value interface{})
    Ascending() bool
}

we have to cast this later
Costs on performance

\[
f := \text{val.}(\text{float64})
\]

// do some math
// FloatIterator represents an iterator over a series.
type FloatIterator interface {
    SeekTo(seek int64) (key int64, value float64)
    Next() (key int64, value float64)
    Ascending() bool
}
Implement for every function

```go
func MapMean(input *MapInput) interface{} {
    if len(input.Items) == 0 {
        return nil
    }

    out := &meanMapOutput{}
    for _, item := range input.Items {
        out.Count++
        switch v := item.Value.(type) {
        case float64:
            out.Total += v
        case int64:
            out.Total += float64(v)
            out.ResultType = Int64Type
        }
    }
    return out
}
```
Could use interfaces, but there’s a performance penalty there too…
What has been bad
Dependency management
Experiences migrating to 1.5
Compile times longer

on a project this size it’s noticeable
Performance more about our code
Go 1.5 issue forced us to revert

https://github.com/golang/go/issues/12233
Unit and integration tests won’t save you.

if you operate at scale, only full blown scale tests will tell you anything and even then you may not find it