



DelphiDay
italian conference

Delphi vs rest of the world



PAOLO ROSSI

WINTech ITALIA - CTO

SENCHA & EMB. MVP



 blog.paolorossi.net

 paolo@paolorossi.net

 twitter.com/awebguy

 github.com/paolo-rossi

 linkedin.com/in/paolo-rossi-pc



DelphiDay
italian conference

11-12 Giugno 2024
Piacenza



wintech
italia



GITHUB PROJECTS



github.com/paolo-rossi



Delphi JWT

JSON Web Token Library



WiRL

REST Library for Delphi



Linux Daemon

Real Linux daemons



Delphi Neon

JSON Serialization Library



OpenAPI-Delphi

OpenAPI 3.0 Library



NATS Delphi

NATS Client Library for Delphi



AGENDA

1. New languages: the trend today
2. Programming paradigms: the trend today
3. Death by features
4. The case of Go
5. The case of V
6. Conclusions



Programming Languages

1



“OLD” LANGUAGES

- 1991 – Python
- 1991 – Visual Basic
- 1993 – Lua
- 1993 – R
- **1995 – Ruby**
- **1995 – Ada 95**
- **1995 – Java**
- **1995 – Delphi (Object Pascal)**
- **1995 – JavaScript**
- **1995 – PHP**
- 1996 – OCaml
- 1997 – Rebol



NEW LANGUAGES

- 2000 – ActionScript
- 2001 – C#
- 2001 – D
- 2003 – Scala
- **2009 – Go**
- 2011 – Dart
- 2011 – Kotlin
- 2012 – TypeScript
- 2012 – Elixir
- 2014 – Swift
- 2014 – Hack
- 2015 – Rust
- 2016 – Zig
- **2019 – V (Vlang)**



Paradigms

2



PARADIGMS

- Imperative
- Procedural
- Functional
- Reactive
- Declarative
- Object-oriented
- Concurrent



PARADIGMS TREND TODAY

- Imperative
- Procedural 🖐️
- Functional 👍
- Reactive 🖐️
- Declarative
- Object-oriented 🖐️ (traditional)
- Concurrent 👍



Death by Features

3



DEATH BY FEATURES

- Feature death is a phenomenon where the excitement and usage of a feature only lasts for a short while after its launch, and then diminishes over time
- It can lead to a cycle where a product development team becomes (only) reactive



DEATH BY FEATURES

- Happens when product decisions are based on a biased niche of users
- Implement only features that are in line with your product vision
- Of course, there is also death by stagnation!



The case of Go

4



The case of Go

- My first encounter with Go
 - ◆ NATS library for Delphi
 - ◆ Sqids library for Delphi
- I noticed that I was able to easily convert Go code to Delphi code
 - ◆ Using records and functions
- I could easily understand Go code: types, records, functions, function receivers, pointers (yes pointers!)



Demo:

Delphi vs Go

- Sqids Library



Why a new language??

- To solve real world code problems (a Google)
- C/C++ too complex, slow builds
- Java, C# too complex / slow
- Dynamic languages (Python, Javascript, ect...)
 - ◆ Too slow
- Every language listes has some “pain points”



(Language) Pain Points

- Slow builds
- Uncontrolled dependencies
- Each programmer using a different subset of the language
- Poor program understanding
 - ◆ Code hard to read
 - ◆ Poorly documented
- Cost (man hours) of updates



Go features

- “(Go) is a language that could be learned in a few days”
 - ◆ Well probably not only “few” but...
- Simplicity
- Fast compile times
- Strong typing
- Concurrency
- Self-contained binaries
- Implicit Interfaces



Applications written in Go

- NATS
- Docker
- Kubernetes
- Caddy
- Prometheus
- Grafana
- InfluxDB
- Traefik
- Filebeat
- Podman
- Terraform
- PocketBase
- Github Cli
- Hugo
- CockroachDB
- Vault



Go roots

- Defined in 2009 at Google
 - ◆ Robert Griesemer
 - ◆ Rob Pike
 - ◆ Ken Thompson (inventor of B, the predecessor of C)
- Often referred as “Google version of C”
 - ◆ Not true!



Go roots

- Go at Google: Language Design in the Service of Software Engineering
 - ◆ <https://go.dev/talks/2012/splash.article>
 - ◆ <https://go.dev/talks/>
- GopherCon 2015: Robert Griesemer - The Evolution of Go
 - ◆ <https://www.youtube.com/watch?v=0ReKdcpNyQg>

Robert Griesemer - The Evolution of Go





Demo:

Oberon vs Go

- Oberon source <-> Go source



The case of V

5



The case of V

- Defined in 2019
 - ◆ Alexander Medvednikov
 - ◆ Contributions from Go devteam
- Probably will not be a mainstream language
- Designed after Go
 - ◆ Influenced by other languages
- Let's see what languages
 - ◆ <https://vlang.io/compare>



THANK YOU