



**HABLA**  
C O M P U T I N G



Universidad  
Rey Juan Carlos

**Embedding of Domain-Specific Languages in Scala**

*Juan M. Serrano*

*[juanmanuel.serrano@urjc.es](mailto:juanmanuel.serrano@urjc.es) | [hablapps.com](http://hablapps.com)*

October 2024

**Lambda World Workshops**

Cádiz, Spain



# We are Habla Computing

A curated team of data engineer experts and *functional programming* evangelists and trainers.

Pioneers in the use of Scala, experts in Spark and kdb+/q, with more than 13 years of experience in the big data & microservice technology landscape.





ELSEVIER

## Science of Computer Programming

Volume 190, 1 May 2020, 102395



### The optics of language-integrated query ☆

J. López-González<sup>a b</sup>  , Juan M. Serrano<sup>a b</sup> 



## Doric API

Fast & safe development API for Apache Spark



doric

Public



Type safety for spark columns



Scala



62



11

# About me

*co-founder*



*faculty staff*



Universidad  
Rey Juan Carlos

*meetup organizer*



>2300 members



# Goals

- Knowing the purpose of DSLs, its major components and its relationship to FP in general
- Being able to implement DSLs in the tagless-final style
- Knowing the Scala 3 features that allow us to implement tagless-final DSLs
- Being prepared to learn more advanced topics on DSL design

# What are DSLs?

- Domain specific languages are programming languages specifically designed for a given purpose; they contrast with *general*-purpose programming languages.

Querying databases	→	SQL, DataFrames, comprehensions, ...
Implementing Http endpoints	→	Tapir, ...
Writing parsers	→	Yacc, lex, regex(3), parsley, ...
Unit/Property-based testing	→	Scalatest, Scalacheck, Quickcheck, ...
Computing with arrays	→	APL, Q, ...
Marking up documents	→	HTML, Latex, Markdown, ...
Stream filters/transformers	→	Sed, Jq, pyjq, ...
...	→	...

# External vs. internal DSLs

- External DSLs are given their own syntax and tooling; internal DSLs are embedded into general-purpose DSLs

Querying databases	→	SQL, DataFrames, comprehensions, ...
Implementing Http endpoints	→	Tapir, ...
Writing parsers	→	Yacc, lex, regex(3), parsley, ...
Unit/Property-based testing	→	Scalatest, Scalacheck, ...
Computing with arrays	→	APL, Q, ...
Marking up documents	→	HTML, Latex, Markdown, ...
Stream filters/transformers	→	Sed, Jq, pyjq, ...
...	→	...

# DSLs & Functional programming

What is a functional architecture?

A FUNCTIONAL LANGUAGE IS A DOMAIN-SPECIFIC LANGUAGE FOR DEFINING DOMAIN-SPECIFIC LANGUAGES



<https://homepages.inf.ed.ac.uk/wadler/papers/qdsl/googlex.pdf>

DSL<sub>1</sub>

PURE DESCRIPTION 1

DSL<sub>2</sub>

PURE DESCRIPTION 2

DSL<sub>n</sub>

PURE DESCRIPTION N



(side effects)





# DSLs & Functional programming



Functional programming (FP) is based on a simple premise with far-reaching implications: we construct our programs using only *pure functions*—in other words, functions that have no *side effects*.

p. 3

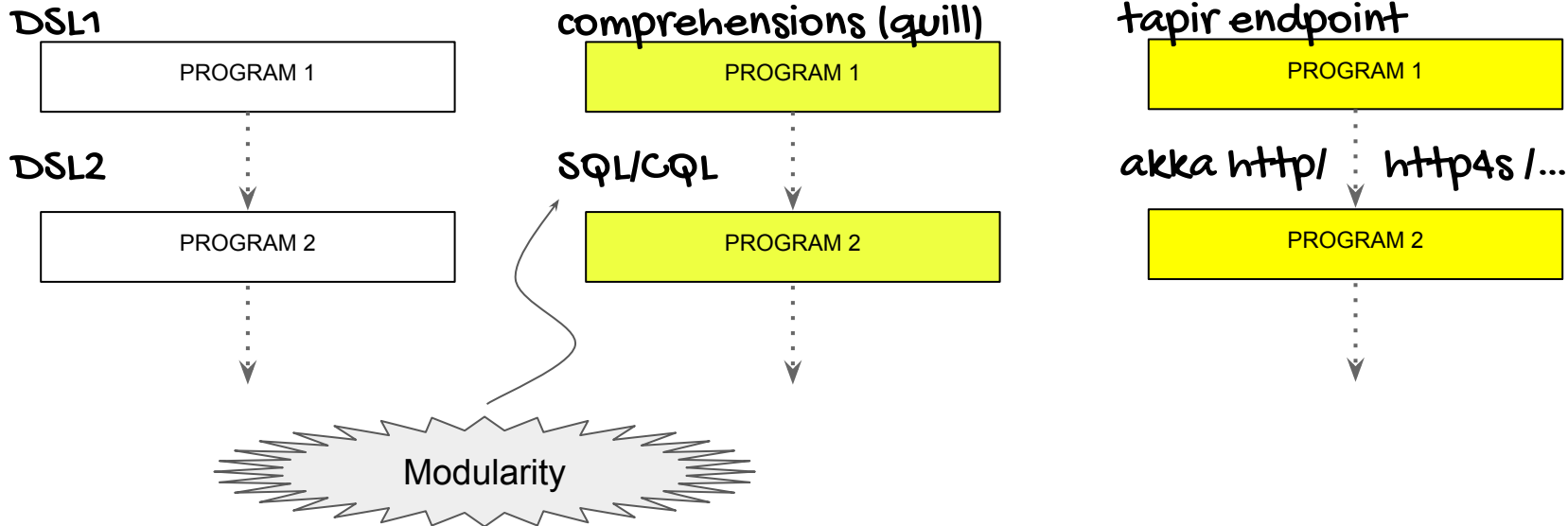
- Still, we need to speak about the effects that we need
- So, pure functions don't execute side effects, but *describe* the side effects that we need



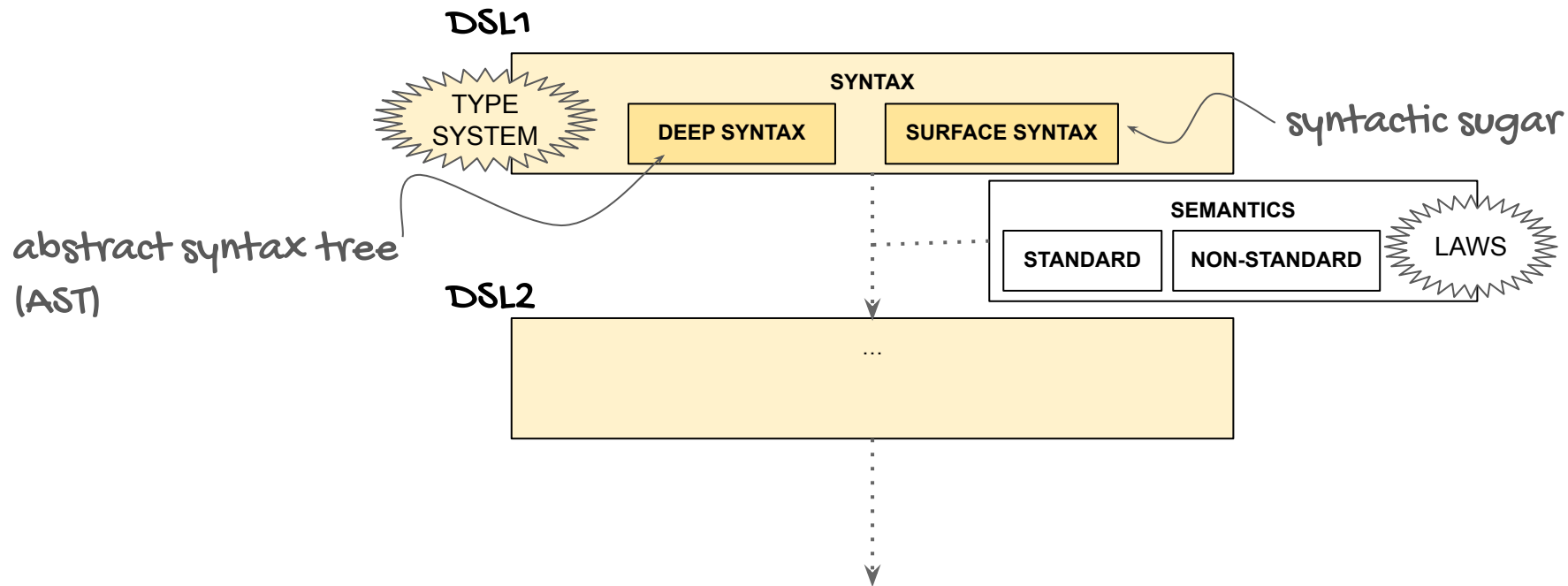
Those effects are described by a **DSL!**



# Declarative programming



# DSL components



# Implementation techniques

- Quoted DSLs
  - Builds upon macros/staging
  - Leverage the AST of the host language in full
  - Examples: Quill
- Initial DSLs
  - Builds upon algebraic data types
  - Most simple approach
  - Examples: ...
- Tagless-final DSLs
  - Builds upon type classes
  - Simple plus extensible
  - Examples: ...

**OUR CHOICE**

# Methodology

- Toy examples to introduce concepts
- Real use-case to put them in practice
- Jupyter lab FTW!
- Notebooks to introduce new concepts: with small exercises
- Notebooks with practical exercises



<https://dslcourse2024.hablapps.com>

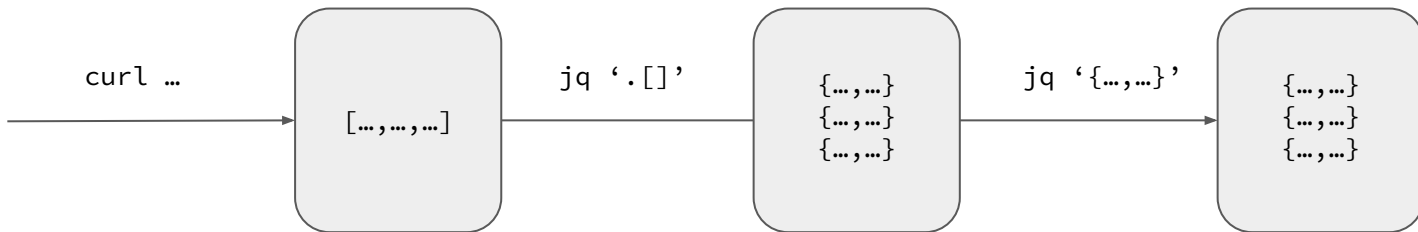
## Use case

# ./jq

jq is a lightweight and flexible command-line JSON processor.

[Download jq 1.7](#) [Try online at jqplay.org!](#)

```
$ curl 'https://api.github.com/repos/jqlang/jq/commits?per_page=5' | \  
> jq '[.[] | {message: .commit.message, name: .commit.committer.name}]'
```





## Use case

# ./jq

jq is a lightweight and flexible  
command-line JSON processor.

[Download jq 1.7](#) [Try online at jqplay.org!](#)

We aim at embedding jq in Scala so that:

- We can write jq expressions in Scala **idiomatically**
- We can use jq expressions with arbitrary **stream processors** like fs2, akka-http, zio-streams, etc.
- We can illustrate the usefulness of the **tagless-final** style and the features of **Scala 3** that facilitates this embedding



# Outline

1. **Intro**
2. **Ad-hoc Embedding**
3. **Domain Abstraction**
4. **Dynamic typing**
5. **Static Typing**

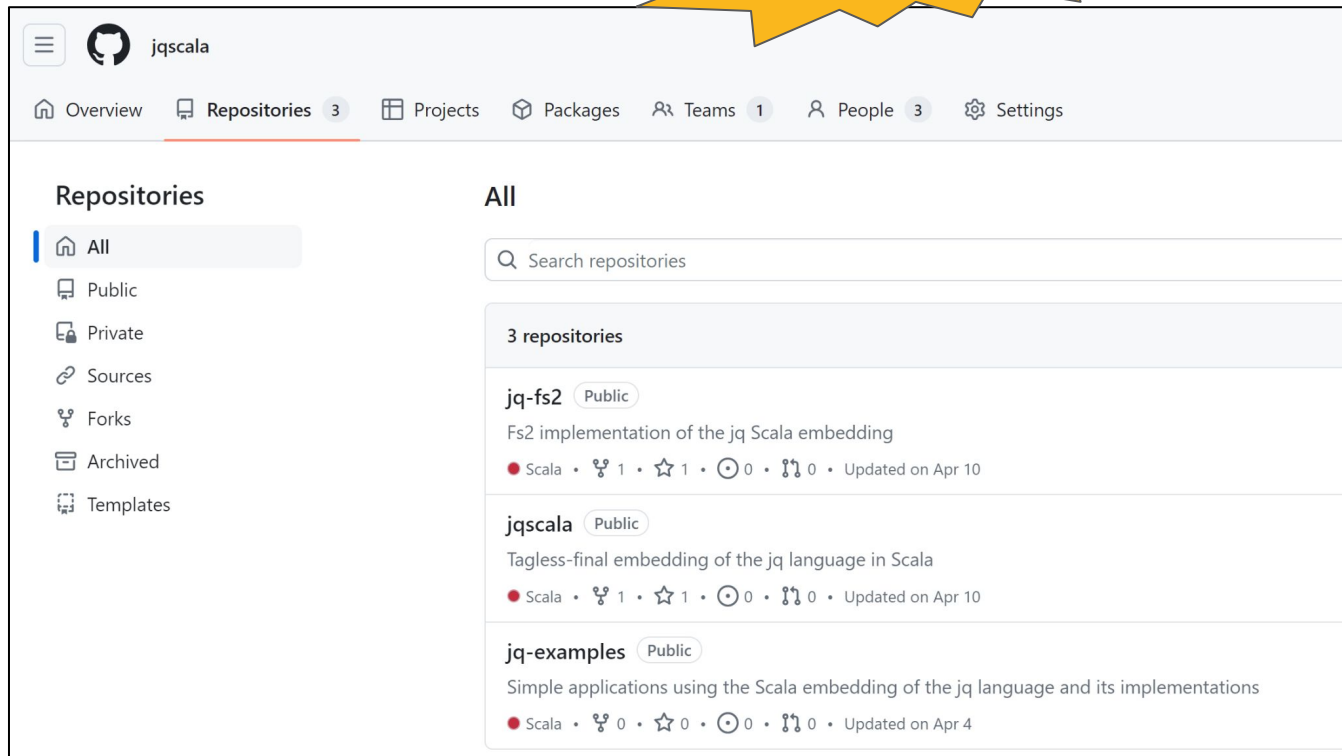


jq exercises



# References

Contributions  
welcomed!



The screenshot shows the GitHub profile page for the user 'jqscala'. The 'Repos' tab is selected, showing 3 repositories. The repositories listed are 'jq-fs2', 'jqscala', and 'jq-examples', all marked as 'Public'. Each repository entry includes a description, a list of languages (Scala), and statistics for forks, stars, and issues. The 'jqscala' repository is the most prominent, being the user's profile repository.

Repository	Visibility	Description	Language	Forks	Stars	Issues	Updated
jq-fs2	Public	Fs2 implementation of the jq Scala embedding	Scala	1	1	0	Apr 10
jqscala	Public	Tagless-final embedding of the jq language in Scala	Scala	1	1	0	Apr 10
jq-examples	Public	Simple applications using the Scala embedding of the jq language and its implementations	Scala	0	0	0	Apr 4

<https://github.com/jqscala/jqscala>



# References

- <https://okmij.org/ftp/tagless-final/>
- <https://github.com/hablapps/doric>
- <https://github.com/hablapps/tagless-final-tutorial>
- <https://github.com/hablapps/lambdas>
- <https://github.com/hablapps/optica>
- <https://docs.scala-lang.org/scala3/reference/metaprogramming/staging.html>
- <https://okmij.org/ftp/Streams.html#strymonasv2>

AVAV

# Enjoy!



**Phone us:**

+34 609 252 235



**Email us:**

[info@hablapps.com](mailto:info@hablapps.com)



**Find us:**

Spaces Río. Calle de Manzanares, 4 |  
Madrid, Spain +34 609 252 235

[info@hablapps.com](mailto:info@hablapps.com)

