GROOGLE

WHEN GOOGLE MET GROOVY

... or why use Groovy to build a DSL

PRESENTATION

Hi!

I’m Jorge Aguilera,

@pvidasoftware (PuraVida Software)

Last +-10 years using Groovy

KNOW YOUR AUDIENCE

- raise your hand if you work/know Groovy?
- raise your hand if you work/know Google API?
- raise your hand if you know what a DSL is?

AGENDA

- DSLs
- Groogle: origin, features and live demo
- Why Groovy
- Real use case of DSL

WHAT'S A DSL?

DSL: Domain Specific Language

> *DSLs are **small** languages, focused on a **particular aspect** of a software system. You can't build a whole program with a DSL, but you often use **multiple DSLs** in a system mainly written in a general purpose language.*

https://martinfowler.com/books/dsl.html

— Martin Fowler

SQL

The (maybe) most famous DSL

```
select title, author, pages from books
```

GRADLE

```
repositories {
    mavenCentral()
}
dependencies{
    api 'org.log4j:log4j:1.2.3'
}
```

FLUENT

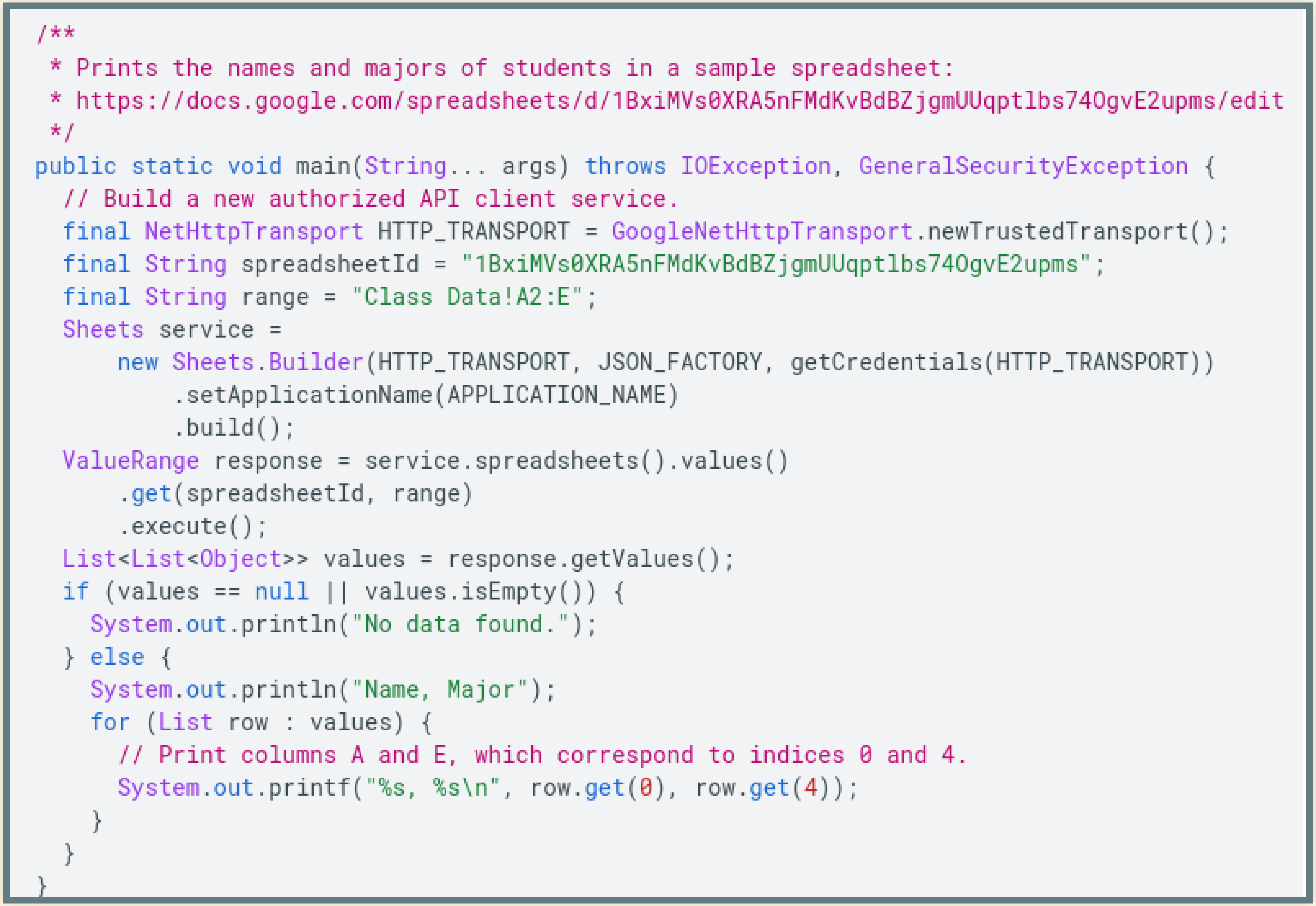
```
IntegrationFlows.from("example")
    .channel(this.inputChannel())
    .filter((Integer p) -> p > 0)
    .transform(Object::toString)
    .channel(MessageChannels.queue())
    .get();
```

GROOGLE, THE ORIGIN

Google + Groovy = Groogle

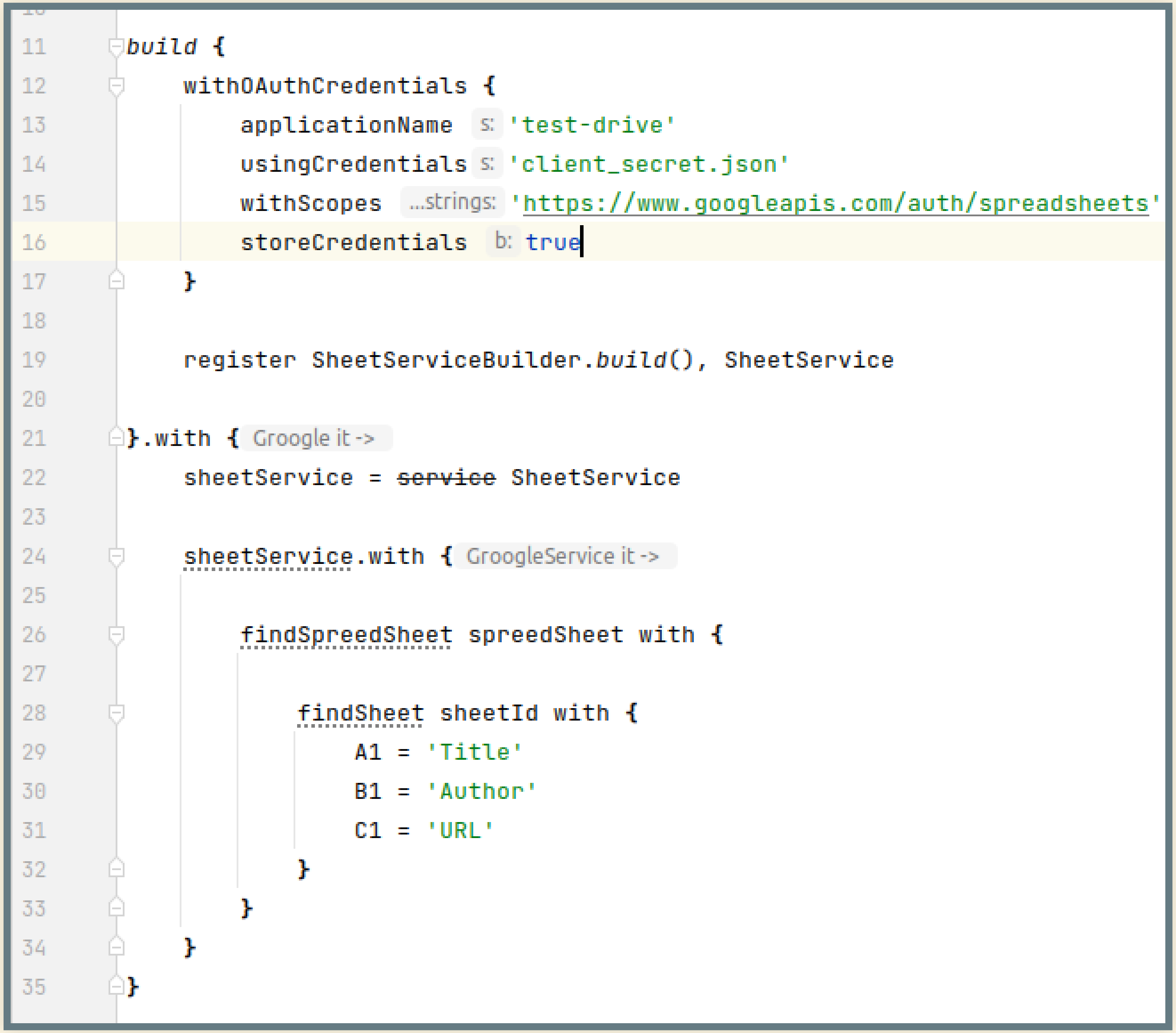
- started as a (lot of) Groovy Scripts for Google Calendar for a friend of mine
- a collection of DSL for Google APIs
- (the idea) simple to use for non-programmer users
- spoiler: my friend never used it

GOOGLE "VS" GROOGLE



```java
/**
 * Prints the names and majors of students in a sample spreadsheet:
 * https://docs.google.com/spreadsheets/d/1BxiMVs0XRA5nFMdKvBdBZjgmUUqptlbs74OgvE2upms/edit
 */
public static void main(String... args) throws IOException, GeneralSecurityException {
  // Build a new authorized API client service.
  final NetHttpTransport HTTP_TRANSPORT = GoogleNetHttpTransport.newTrustedTransport();
  final String spreadsheetId = "1BxiMVs0XRA5nFMdKvBdBZjgmUUqptlbs74OgvE2upms";
  final String range = "Class Data!A2:E";
  Sheets service =
      new Sheets.Builder(HTTP_TRANSPORT, JSON_FACTORY, getCredentials(HTTP_TRANSPORT))
          .setApplicationName(APPLICATION_NAME)
          .build();
  ValueRange response = service.spreadsheets().values()
      .get(spreadsheetId, range)
      .execute();
  List<List<Object>> values = response.getValues();
  if (values == null || values.isEmpty()) {
    System.out.println("No data found.");
  } else {
    System.out.println("Name, Major");
    for (List row : values) {
      // Print columns A and E, which correspond to indices 0 and 4.
      System.out.printf("%s, %s\n", row.get(0), row.get(4));
    }
  }
}
```

GROOGLE DSL



```
build {
    withOAuthCredentials {
        applicationName 'test-drive'
        usingCredentials 'client_secret.json'
        withScopes 'https://www.googleapis.com/auth/spreadsheets'
        storeCredentials true
    }

    register SheetServiceBuilder.build(), SheetService

}.with {
    sheetService = ~~service~~ SheetService

    sheetService.with {

        findSpreedSheet spreedSheet with {

            findSheet sheetId with {
                A1 = 'Title'
                B1 = 'Author'
                C1 = 'URL'
            }
        }
    }
}
```

GROOGLE I

- Open Source, https://gitlab.com/groogle
- Groovy 3.0.10
- Maven Central (com.puravida-software.groogle)
- Drive, Sheet, Gmail, Calendar, People, Photos (started)
- BigQuery ("in progress")

GROOGLE II

- OAuth and Service authentification
- Multiple instances, multiple services
- Documentation https://groogle.gitlab.io/groogle/latest/index.html

WARNING

> Probably the most difficult part of a DSL is the documentation

LIVE CODING

A live coding session is better than 1000 slides

(🤞 🤞 🤞)

- Retrieve Astronomy Picture of the Day (JSON) (https://apod.nasa.gov/apod/astropix.html)
- Create a Google Sheet and append the data
- Send email with data

GROOGLE DSL

- Drive (createFolder, withFolder, uploadFrom, downloadTo, removeFromDrive …)
- Sheet (createSpreadSheet, withSpreadSheet, createSheet, withSheet, …)
- Gmail (sendEmail)
- Calendar (eachCalendarInList, withPrimaryCalendar, newCalendar)
- BigQuery (rows, eachRow)
- Photos (TODO!!)

WHY GROOVY FOR DSL?

- Command chains
- Operator overloading
- Enhancing classes
- Missing method/property
- @DelegatesTo
- and more …

COMMAND CHAINS

Java

```
please(show).the(square_root).of(100)
paint( wall).with( List.of(red, green)).and(yellow)
```

Groovy

```
please show the square_root of 100
paint wall with red, green and yellow
```

OPERATOR OVERLOADING

Various operators in Groovy are mapped onto regular method calls on objects.

So you can overwrite these methods, i.e:

```
def tobacco = new Tobaco()
def tomatoes = new Tomatoes()

def tomacco = tobacco + tomatoes    1

Tobaco.plus( object ){
    return new Tomacco(nicotine:90)
}
```

1 ™ Homer Simpson

ENHANCING CLASSES

- extension and categories

```
use(TimeCategory)  {
    println 1.minute.from.now
    println 10.hours.ago

    def someDate = new Date()
    println someDate - 3.months
}
```

Categories are **lexically bound.**

MISSING METHOD/PROPERTY

(do you remember A1='Title' example?)

```
def methodMissing(String name, def args) {
    return "this is me"
}

def propertyMissing(String name, value) {
    storage[name] = value
}

def propertyMissing(String name) {
    storage[name]
}
```

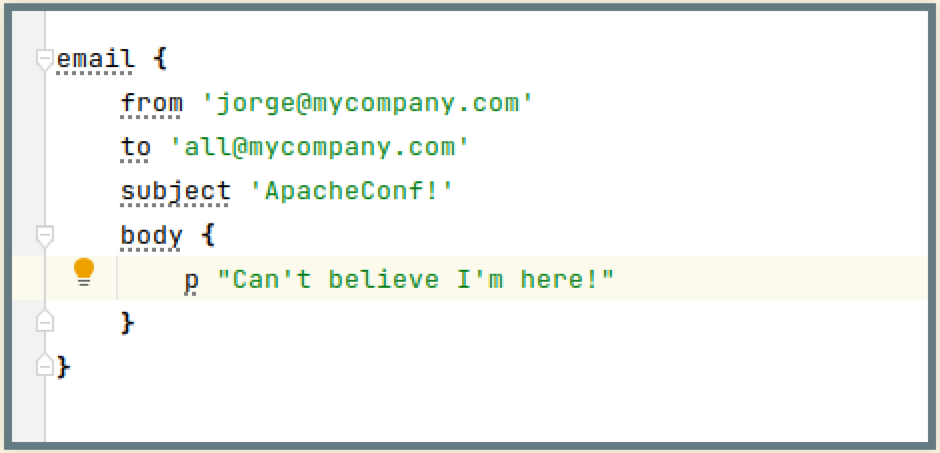
CLOSURES

@DELEGATESTO

With @DelegatesTo we can instruct to the compiler which class will execute the Closure

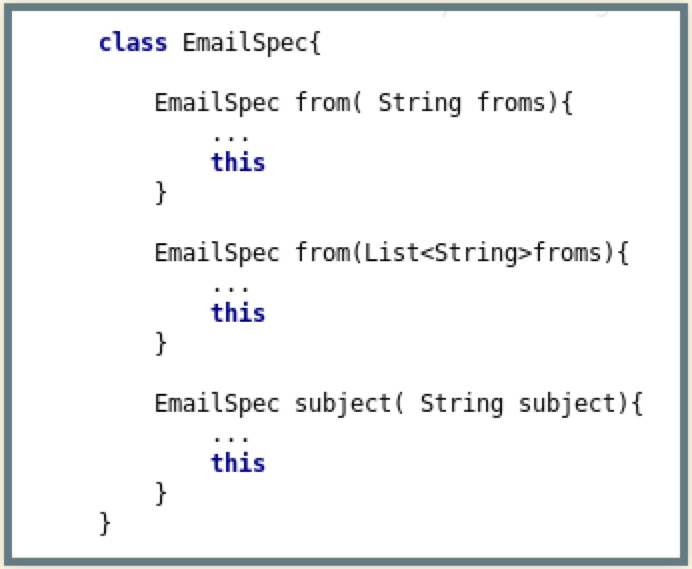
- static compile
- documentation
- IDE completion

@DELEGATESTO DSL



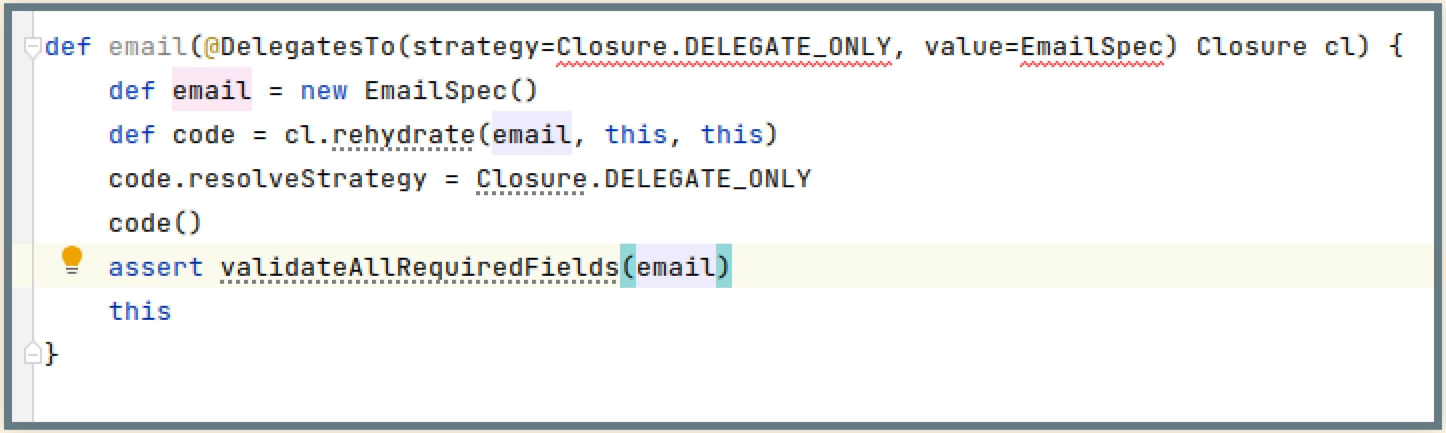
```
email {
    from 'jorge@mycompany.com'
    to 'all@mycompany.com'
    subject 'ApacheConf!'
    body {
        p "Can't believe I'm here!"
    }
}
```

@DELEGATESTO SPEC



```
class EmailSpec{

    EmailSpec from( String froms){
        ...
        this
    }

    EmailSpec from(List<String>froms){
        ...
        this
    }

    EmailSpec subject( String subject){
        ...
        this
    }
}
```

@DELEGATESTO IMPLEMENTATION



```
def email(@DelegatesTo(strategy=Closure.DELEGATE_ONLY, value=EmailSpec) Closure cl) {
    def email = new EmailSpec()
    def code = cl.rehydrate(email, this, this)
    code.resolveStrategy = Closure.DELEGATE_ONLY
    code()
    assert validateAllRequiredFields(email)
    this
}
```

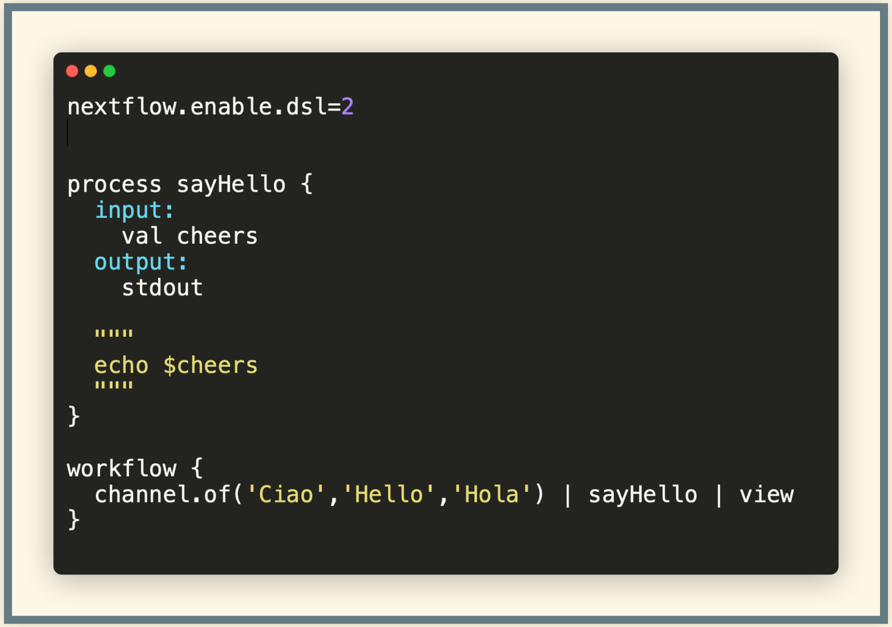
RECAP

- DSL can improve your API/Product
- Groovy has a lot of feature to help you
- @DelegatesTo as an assistant completion
- Have fun and learn from everything

Ok, nice, but show me a real use case …

REAL USE CASE: NEXTFLOW DSL.

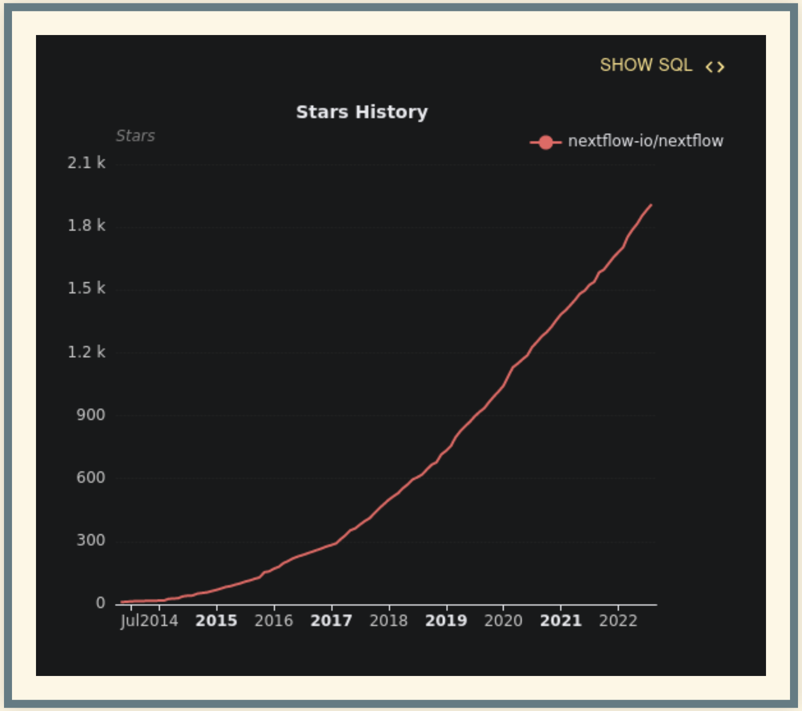
- nextflow.io
- *Nextflow enables scalable and reproducible scientific workflows using software containers.*
- Write (a pipeline) once and run wherever (local, k8s, AWS, Google, Azure,...)



```
nextflow.enable.dsl=2

process sayHello {
  input:
    val cheers
  output:
    stdout

  """
  echo $cheers
  """
}

workflow {
  channel.of('Ciao','Hello','Hola') | sayHello | view
}
```

NEXTFLOW

- DSL to orchestrate workflows
- Open Source, community driven
- Groovy 3.0.10 (4.x coming soon)
- GPars and others



DANKSCHEEN
BİYAN
SHUKRIA
TINGKI
YUSPAGARATAM
CHALTU
YAQHANYELAY
NUHUN
SPASSIBO
SNACHALHUYA
TASHAKKUR ATU
RACIAS
WABEEJA
MAITEKA
HUI
DHANYAVAADAD
SUKSAMA
GUI
ANINA
EKHMET
UNALCHEESH
HATUR
MERSI
SPASIBO
ATTO
DENKAUJA
NENACHALHYA
THANK
EKOJU
SIKOMO
ARIGATO
YOU
SANCO
KOMAPSUMNIDA
MAAKE
GRAZIE
MEHRBANI
MAKETAI
SHUKURIA
MERASTAWHY
GAEJTHO
PALDIES
LAH
TAVTAPUCH
GOZAIMASHITA
MEDAWAGSE
EFCHARISTO
AGUYJE
BAIIKA
JUSPAXAR
FAKAAUE
MINMONCHAR
BOLZÏN
MERCI

QUESTIONS ?