

Developers

Eliminate Kubernetes Complexity For Developers

Edward Ionel and Rags Srinivas



01



Intro

02

NoSQL

03

Cassandra/Astra/
k8ssandra

04

Quarkus

05

Microservices =
Quarkus+Cassandra

06

Live Coding

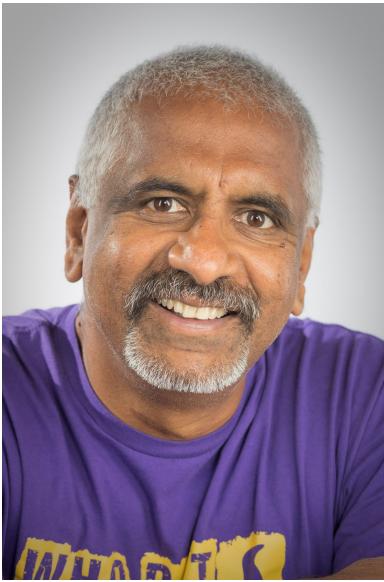
07

Resources



Agenda

Developer Advocate



- Developer/Architect
- Mechanical Engineer (so many moons ago)
- Distributed systems
- Love to teach and communicate
- Inner loop == developer productivity



Raghavan "Rags" Srinivas



@rags

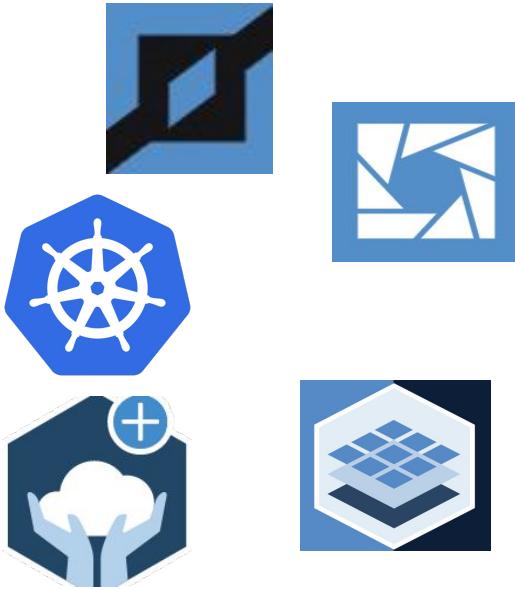


@ragsns



@ragss

Developer Advocate



@edward-ionel



@eionel



@loneleward

- Marketer/Developer
- Community Builder
- Team Lens
- Eliminate Kubernetes complexity
 - = My goal



Edward Ionel



S



Cedrick
Lunven



David
Dieruf



Rags
Srinivas



Artem
Chebotko



Stefano
Lottini



Aleksandr
Volochnev



Aaron
Ploetz



S



Jack
Fryer



Kirsten
Hunter



Gary
Harvey



Mary
Grygleski



Ryan
Welford



David
Gilardi

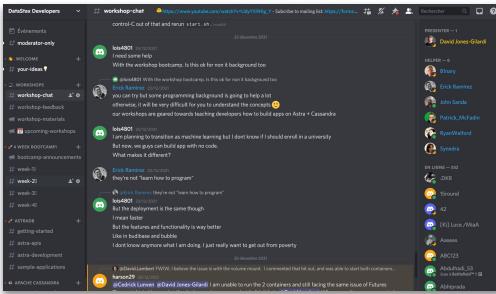


DataStax Developers Crew

Livestream: youtube.com/DataStaxDevs

Questions: <https://dtsx.io/discord>

Agenda



YouTube
(with nighbot)

Discord
(#workshop-chat)



Games and quizzes: menti.com

How much experience do you have with the Spring Framework ?



!menti

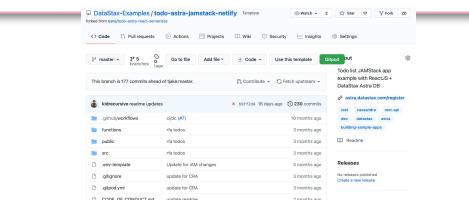


Live Sessions



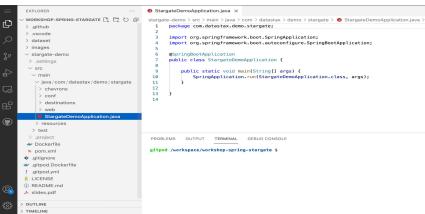
Nothing to install !

Source code + exercises + slides



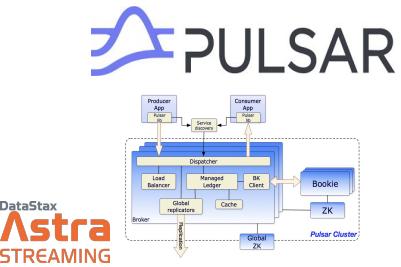
 GitHub

IDE

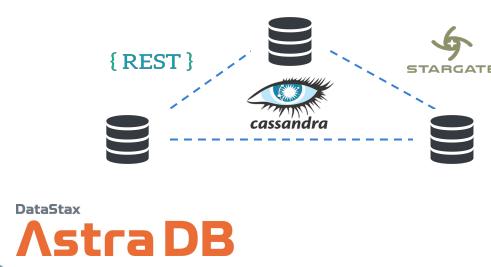


 Gitpod

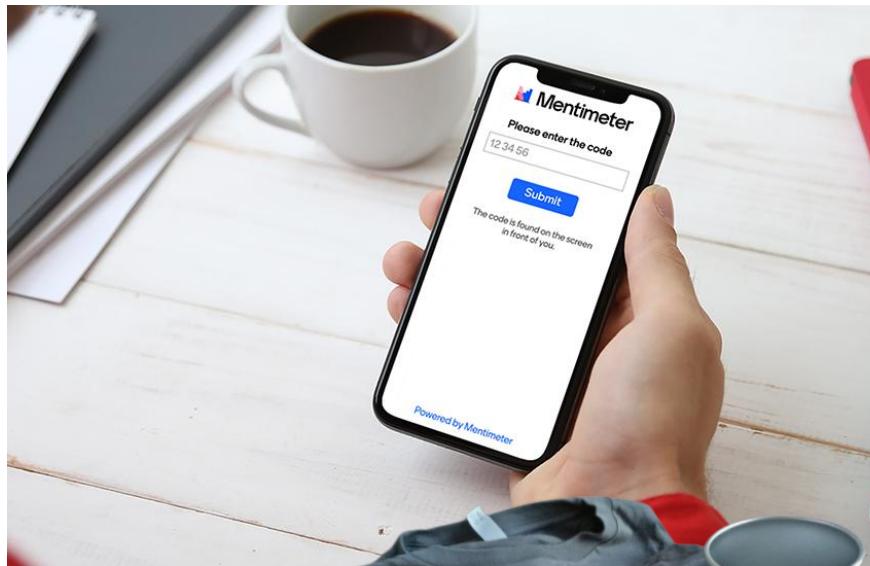
Event Streaming



Database + REST



Hands-On Housekeeping



**menti . com ⇒ enter code
Don't answer in YT chat
Look at phone (not at YT)**

Quiz on "Menti" !

Objective

- Creating an inner development workflow as a developer
- Tools that help a Kubernetes developer, particularly the inner loop
- Eliminate the need for developers mastering kubectl



todos

- What needs to be done?
- Week1, Learn about Apache Cassandra
 - Week2, Apache Cassandra Data Model
 - How to Connect to Cassandra

1 item left

All Active Completed

[Clear completed \(2\)](#)

Double-click to edit a todo

Written by [Addy Osmani](#), modified by [Pete Hodgson](#) to be integrated with a [TodoBackend API](#).

Part of [TodoMVC](#) & [TodoBackend](#)

Today's challenge: Todo App

01

Lens



02

NoSQL

04

Quarkus

03

Cassandra/Astra/
k8ssandra

05

**Microservices =
Quarkus+Cassandra**

06

Live Coding

07

Resources



Agenda

The Way The World Runs Kubernetes

Stats Updated: June 9, 2022

#1

KUBERNETES IDE

750k+

UNIQUE USERS

19.5k+

GITHUB STARGAZERS



LENS USERS ARE FROM SOME OF THE MOST ICONIC ORGANIZATIONS IN THE WORLD

1M+

CLUSTERS

3.5M+

MACHINES

Lens Desktop Overview

- All you need to take control of Kubernetes
- Standalone desktop app; no in-cluster components required; connection via K8S APIs
- Provides complete situational awareness of all clusters and workloads on-premises, public clouds and beyond
- Designed for devs & ops who need to work with Kubernetes on a daily basis
 - Lower the barrier of entry for people just getting started
 - Radically improve productivity and quality of life for people with more experience
- Works with any certified Kubernetes distributions
- Built on open source. Free of charge.



Free of charge | Open Source | MIT license | Built binaries available for:



How Value Is Measured With Lens?

Recent Lens User Survey Results; 522 Responses

1

We Accelerate
Kubernetes
Time to Value

2

We Solve
Kubernetes
Complexity

3

We Increase
Kubernetes
Productivity

95%

**Users of Lens Learn and
Adopt Kubernetes Faster**

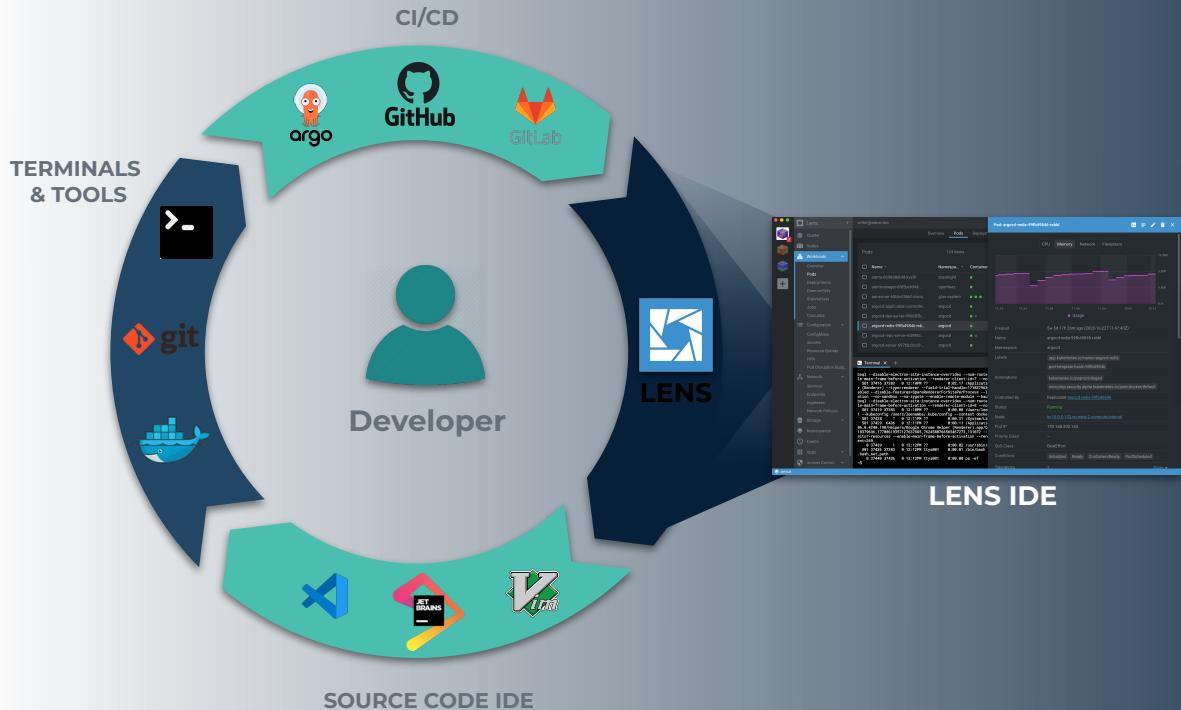
75%

**Less Tools Needed When
Working With Lens**

20%

**Time Saved Per Day /
Week When Using Lens**

Development with Lens



Deployment

Observability

Testing

Troubleshooting



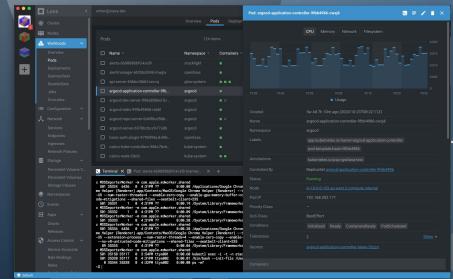
Operations with Lens



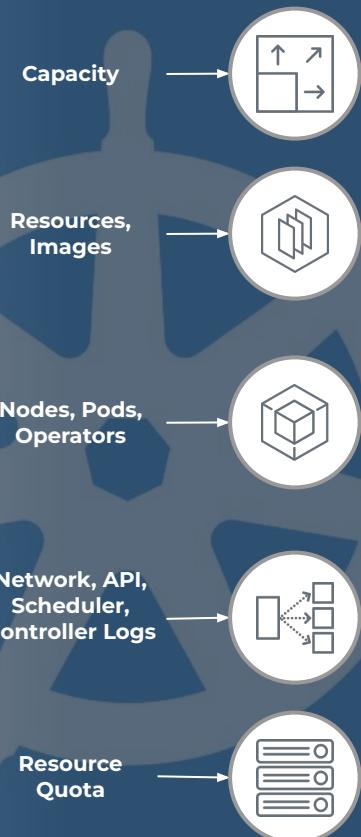
Kubernetes
SRE

The screenshot displays several panels from the Lens IDE:

- Daily Operations:** Three circular dashboards showing the status of Pods (124), Deployments (56), and Stateful Sets (6). Legend: Running (green), Succeeded (light green), Pending (orange), Failed (red).
- Health and Status:** A chart showing CPU, Memory, Network, and Filesystem usage over time (15:05 to 15:59). Legend: Receive (blue), Transmit (green).
- Events:** A list of recent events.
- Lifecycle Management:** A chart showing Filesystem usage (286.1Mi to 190.7Mi).
- Trouble shooting:** A terminal window showing system logs for a pod named calico-node-mndncd.



LENS IDE

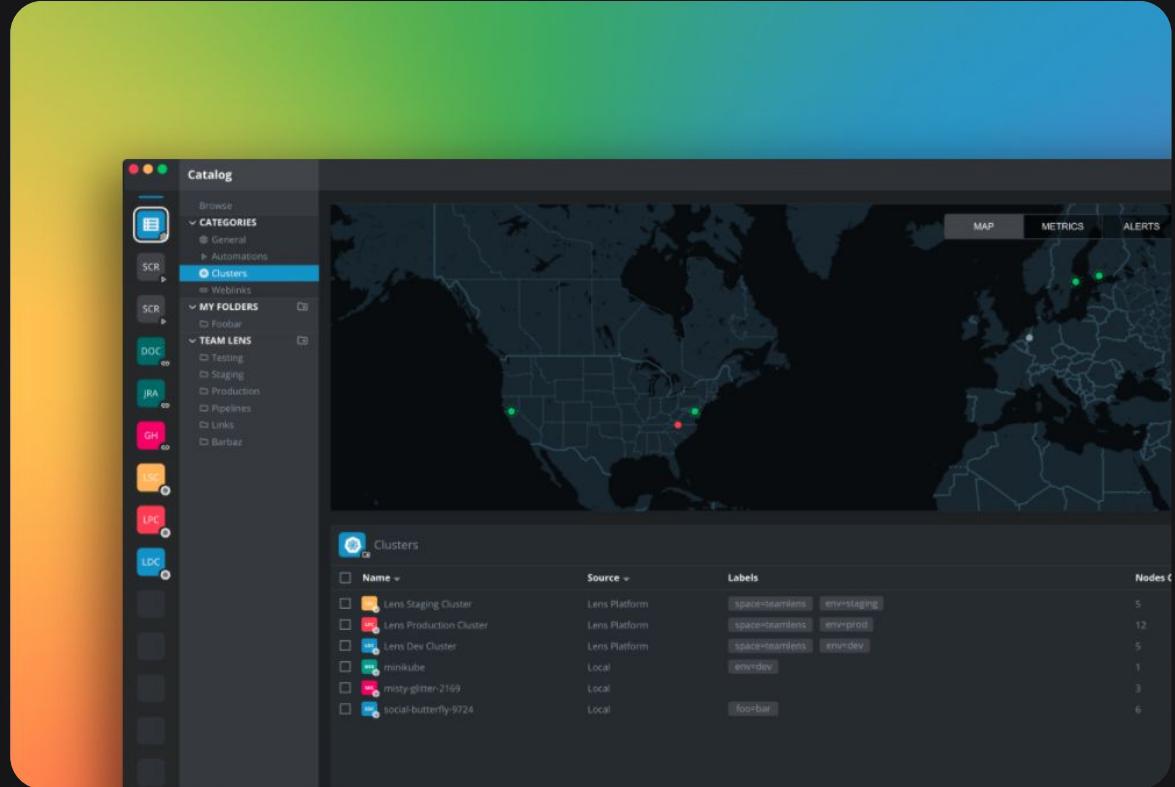


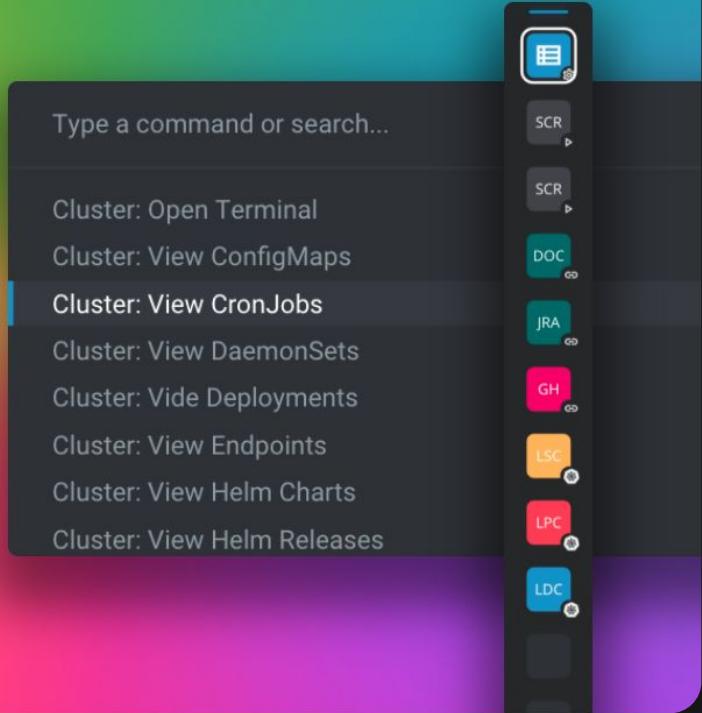
Features at a Glance

Your Cloud Native Technology Stack. Accessible.

Unified Catalog. Bring all your clusters, services, workloads, tools, automations, and related resources together for easy access.

Browse & Organize. Use search, filters, categories and labels to access the resources you need to work with easier than ever before.





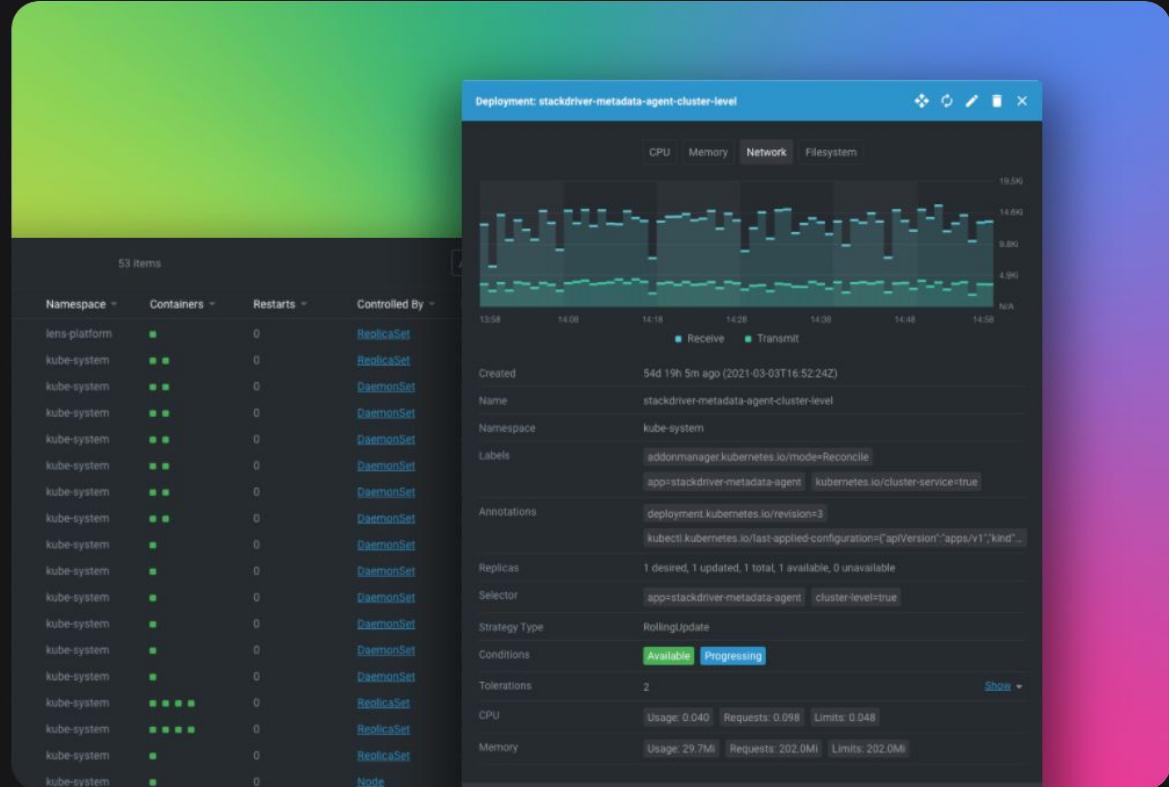
Get What You Want. Fast.

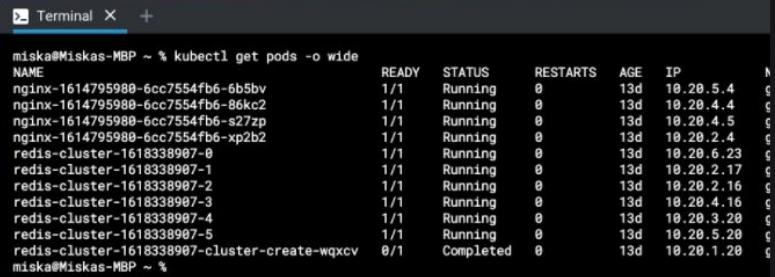
Hotbars. The main navigation, allowing users to build their “workflows” and “automation” within the desktop application. Users can customize items in the Hotbar by assigning different labels, colors, and icons for easy recall.

Command Palette. Command Palette allows users to perform specific keyboard shortcuts to make the most common tasks easier. Improving accessibility and efficiency when working with Lens.

See Everything. In Right Context. Real Time.

Built-In Visualizations. Lens integrates with Prometheus to visualize and see trends in resource usage metrics, including CPU, memory, network and disk, with the total capacity, actual usage, requests and limits. Detailed visualization is automatically generated for each k8s resource.





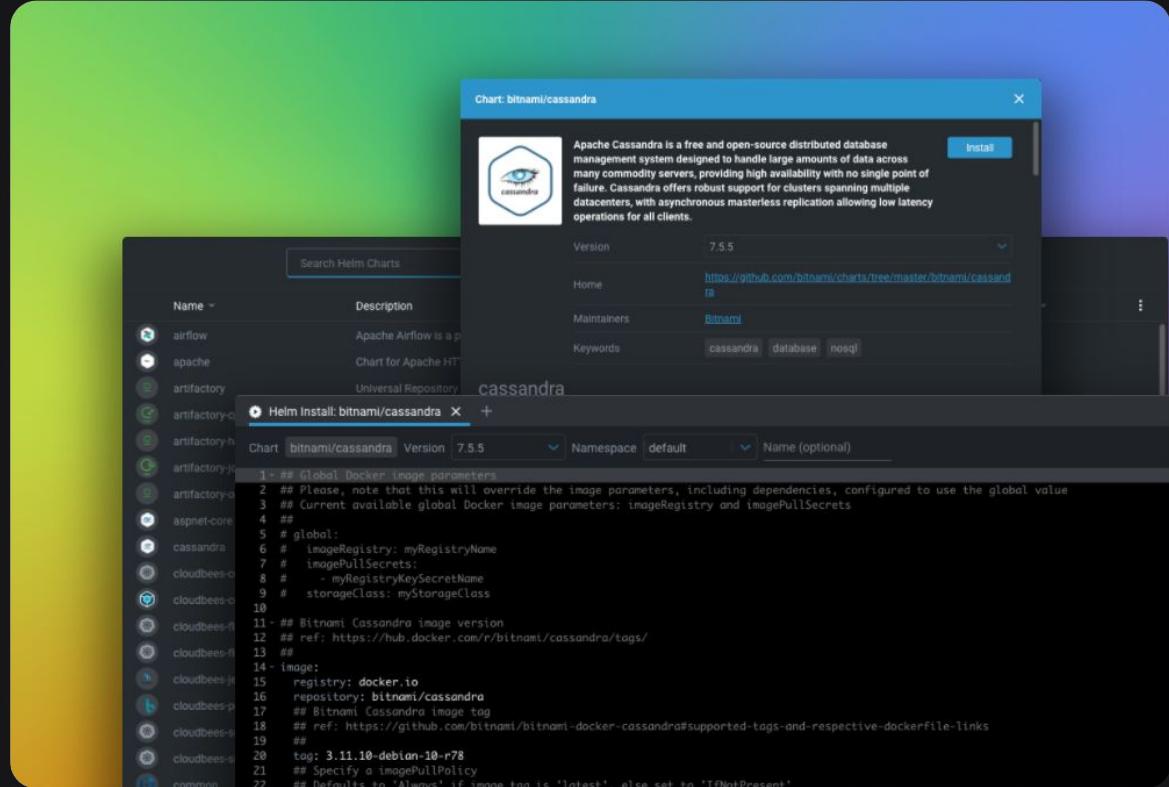
```
miska@Miskas-MBP ~ % kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE     IP          NODE
nginx-1614795980-6cc7554fb6-6b5bv   1/1    Running   0          13d    10.20.5.4   %
nginx-1614795980-6cc7554fb6-86kc2   1/1    Running   0          13d    10.20.4.4   %
nginx-1614795980-6cc7554fb6-s27zp   1/1    Running   0          13d    10.20.4.5   %
nginx-1614795980-6cc7554fb6-xp2b2   1/1    Running   0          13d    10.20.2.4   %
redis-cluster-1618338987-0           1/1    Running   0          13d    10.20.6.23   %
redis-cluster-1618338987-1           1/1    Running   0          13d    10.20.2.17   %
redis-cluster-1618338987-2           1/1    Running   0          13d    10.20.2.16   %
redis-cluster-1618338987-3           1/1    Running   0          13d    10.20.4.16   %
redis-cluster-1618338987-4           1/1    Running   0          13d    10.20.3.28   %
redis-cluster-1618338987-5           1/1    Running   0          13d    10.20.5.20   %
redis-cluster-1618338987-cluster-create-wqxcv  0/1    Completed  0          13d    10.20.1.20   %
miska@Miskas-MBP ~ %
```

Full Power at Your Fingertips. Always.

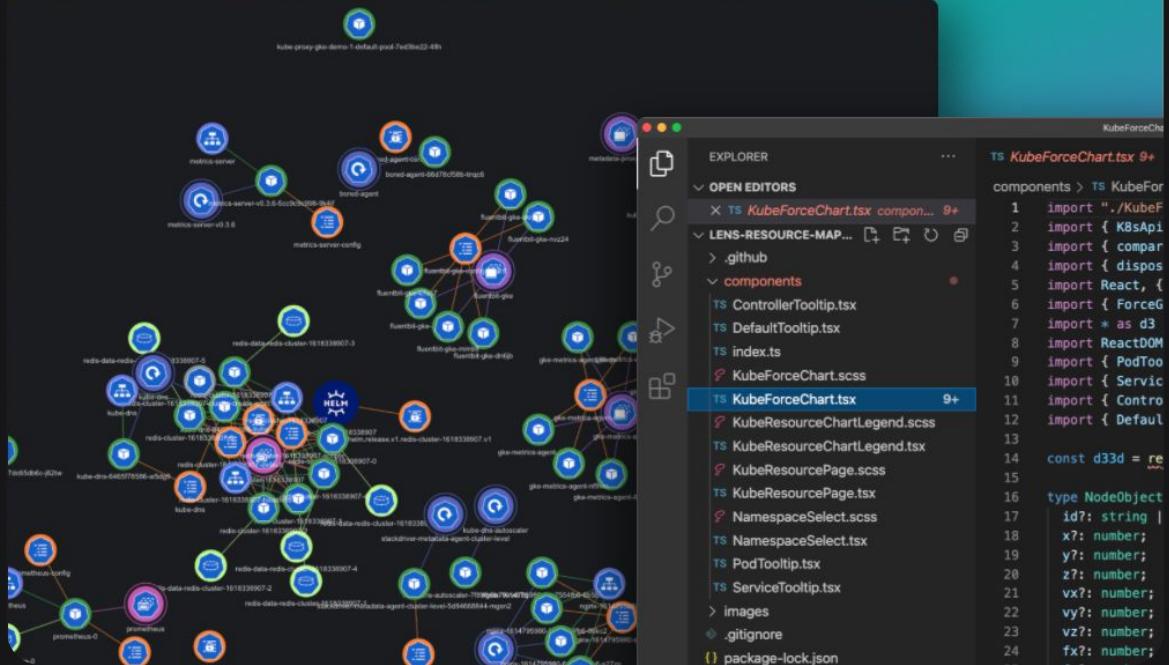
Smart Terminal. The smart terminal comes with kubectl and helm, automatically syncing the version of kubectl to match the currently selected K8S cluster API version. Lens will automatically assign the kubeconfig context to match the currently selected K8s cluster.

All Cloud Native Technologies. Deployed in a Click.

Helm Charts. Lens comes with Helm charts management allowing for the discovery and fast deployment of thousands of publicly available Helm charts and management of your own repositories. Explore installed Helm charts and their revisions and upgrade with a single click.



No Limits. Expand. Just the Way You Like It.



Extensions. Easily add Lens extensions from the community and cloud native ecosystem vendors or build your own. Lens Extensions are used to add custom functionality and services to accelerate development workflows for all the technologies that integrate with Kubernetes and other cloud-native technologies.

You are not alone.
Join the party.
Share.

Team Management. Team Space is for collaboration and centralized access. Assign teams and manage Kubernetes RBAC with ease. Take the amazing Lens experience and share it with your team!

Cloud Service. Spaces is made available to Lens users as centralized cloud based service operated by Team Lens. Signup via Lens application to get started for free!

The screenshot shows the Lens application interface for managing a Team Space named "lensapp-staging".

SHARE CLUSTER FOR COLLABORATION: A search bar labeled "Add members by username or email address".

SPACE MEMBERS AND INVITATIONS: A table listing members with their roles:

Member	Role	Actions
@miskun	Member	
@chenhungchan	Admin	
@leenamba	Admin	
@nevalla	Admin	
	Owner	
	Owner	
	Admin	
	Invite Pending	

Members: A section titled "Manage your space members in here." with a search bar and a table:

Member	Role	Actions
@miskun	Member	⋮
@chenhungchan	Admin	⋮
@leenamba	Admin	⋮
@nevalla	Admin	⋮
@msa0311	Owner	⋮
@stevejr	Owner	⋮
@nachasic	Admin	⋮
@jakolehm	Admin	⋮

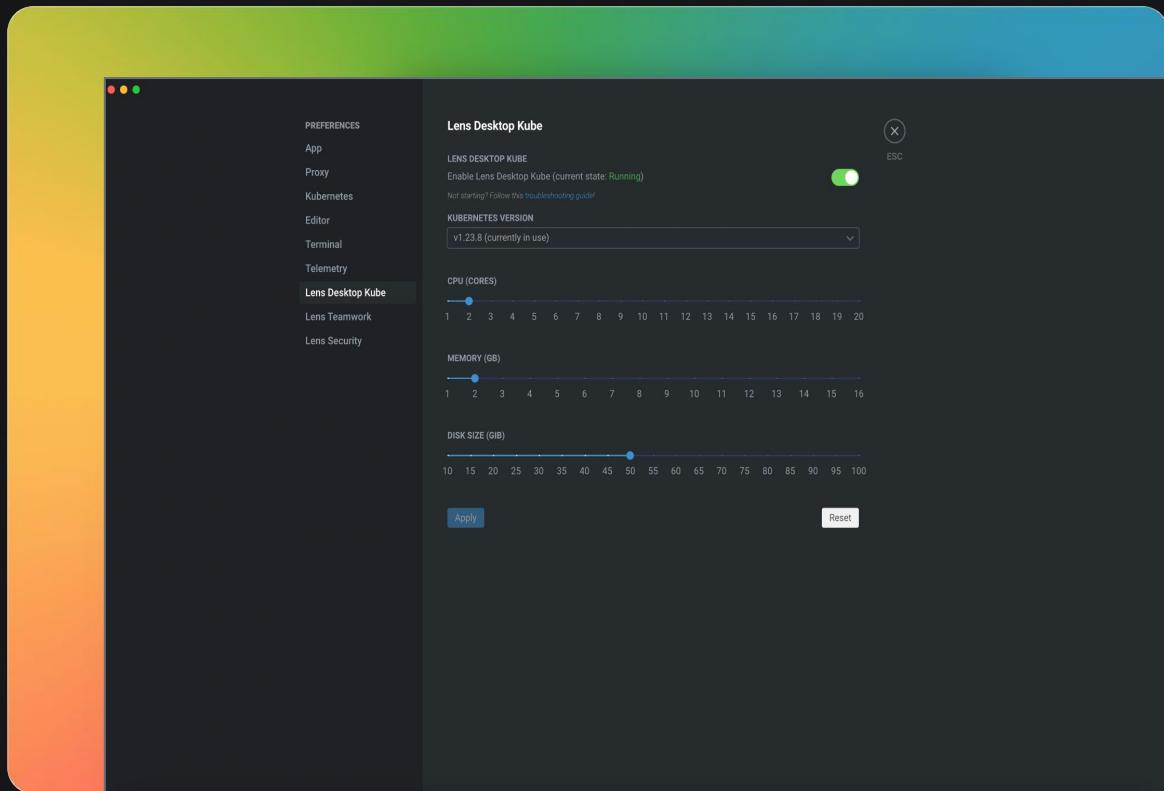
Teams: A section titled "Manage your space teams in here." with a search bar and a table:

Team	Members
lensapp-staging Admin team	@chenhungchan, @leenamba, @nevalla, @msa0311, @stevejr
lensapp-staging Owner team	@jakolehm

Lens Desktop Kubernetes

Kubernetes for Everyone: Provides a personal Kubernetes cluster to everyone. From people just getting started to seasoned cloud native developers. Zero hassle!

Intelligent Auto-Shutdown: If you forget to shut down your cluster, we will do it for you. Avoid searching for unused clusters or being surprised by large cloud provider bills. Pay only for the actual use, without any extra charges.



Resources

1. Download Lens
 - a. <https://k8slens.dev/>
2. Getting Started guide
 - a. <https://medium.com/k8slens/getting-started-with-lens-763290e3b59c>

01

Intro/Lens

02

NoSQL



03

Cassandra/Astra/
k8ssandra

04

Quarkus

05

Microservices = Quarkus
+ Cassandra

06

Live Coding

07

Resources



Agenda

Origin of the term “NoSQL”

- Meetup name on June 11, 2009 in San Francisco
 - Catchy hashtag intended to refer to databases like BigTable and DynamoDB
 - Meetup presentations: Cassandra, MongoDB, CouchDB, HBase, Voldemort, Dynomite, and Hypertable
- Sometimes referred to “Not only SQL”



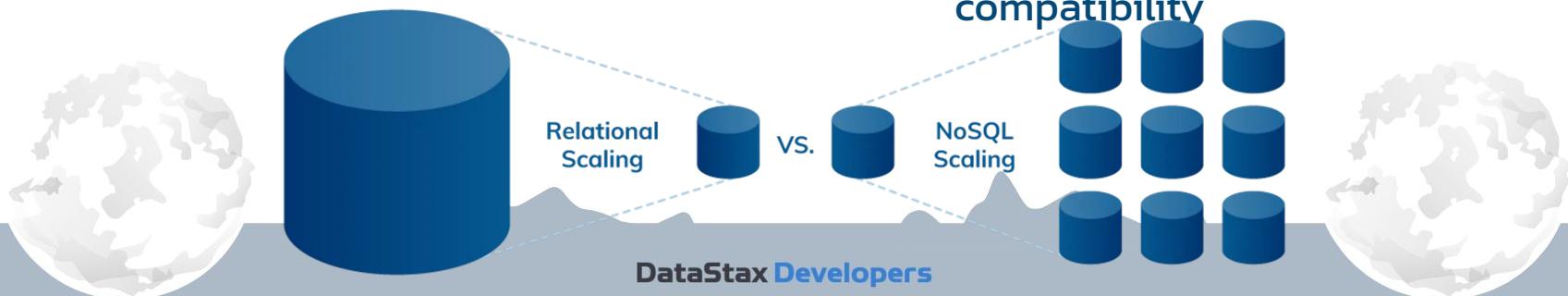
Relational vs. NoSQL

- Relational

- Standard relational data model and language SQL
- ACID transactions
- Integration database
- Designed for a single machine
- Hard to scale
- Impedance mismatch

- NoSQL

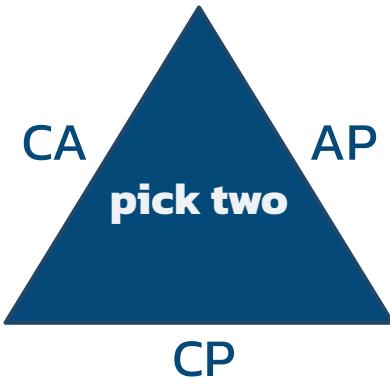
- Variety of data models and languages
- Lower-guarantee transactions
- Application database
- Designed for a cluster
- Easy to scale
- Better database-app compatibility



The CAP Theorem

Always responds,
may not always return
the most recent write

Availability



Consistency

Every read receives
the most recent write
or an error

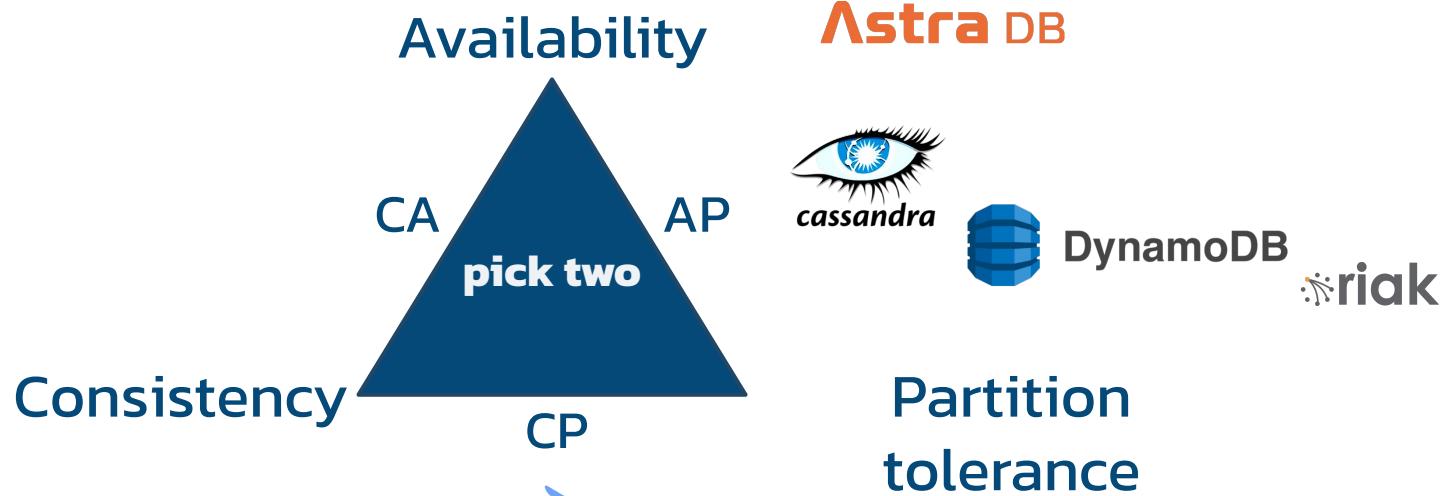


Partition tolerance

Operates in the
presence of network
partition failures



The CAP Theorem



DataStax Developers



01

Intro/Lens

02

NoSQL

04

Quarkus

05

Microservices (Quarkus + Cassandra)

03

Cassandra/Astra/k8ssandra

06

Live Coding

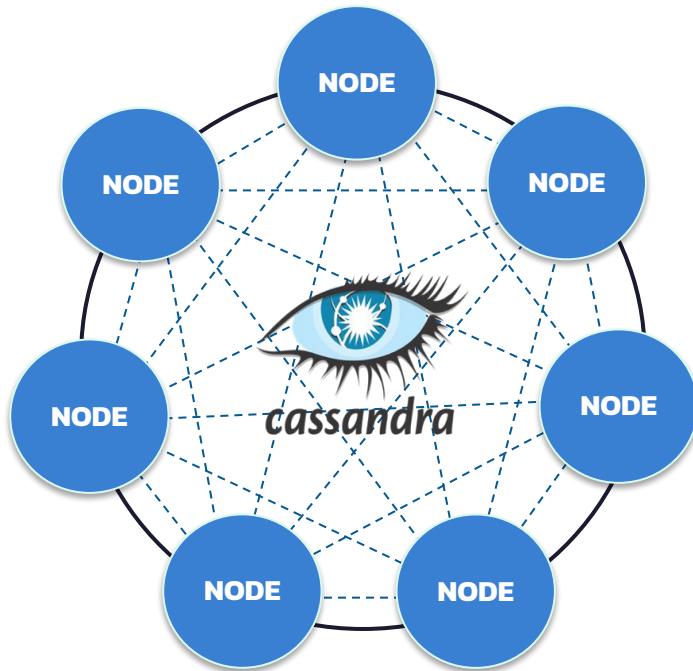
07

Resources



Agenda





1. NO Single Point of Failure
2. Scales for writes and reads
3. Application can contact any node
(in case of failure - just contact next one)



Master-less (Peer-to-Peer) Architecture

Why partitioning? Because scaling doesn't have to be [s]hard!

Big Data doesn't fit to a single server, splitting it into chunks we can easily spread them over dozens, hundreds or even thousands of servers, adding more if needed.



Cassandra is configurable consistent. In any moment of the time, for any particular query you can set the Consistency Level you require to have. It defines how many **CONFIRMATIONS** you'll wait before the response is dispatched;

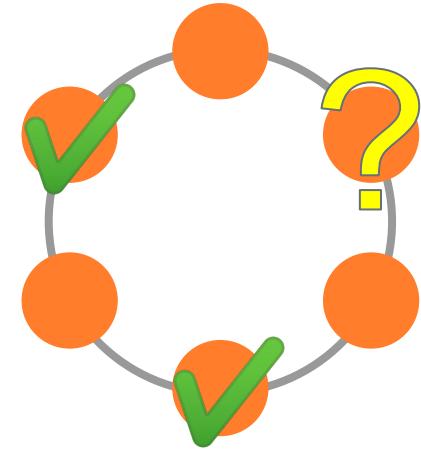
```
PreparedStatement pstmt = session.prepare(  
    "INSERT INTO product (sku, description) VALUES (?, ?)"  
);  
pstmt.setConsistencyLevel(ConsistencyLevel.ONE);
```

```
cqlsh> CONSISTENCY
```

```
Current consistency level is QUORUM.
```

```
cqlsh> CONSISTENCY ALL
```

```
Consistency level set to ALL.
```



Is Cassandra AP or CP?



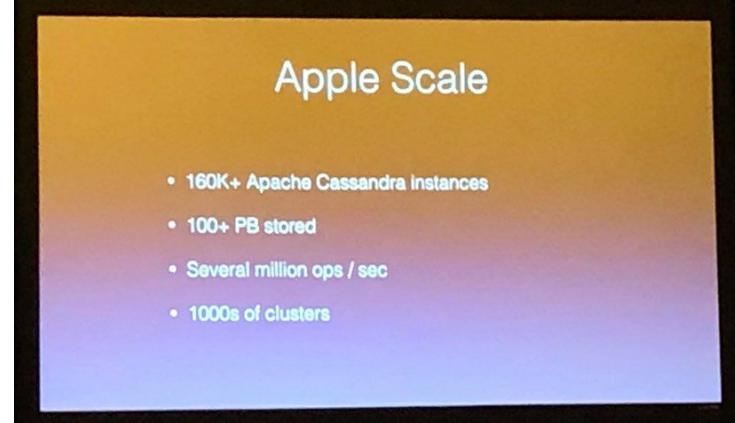
Apache Cassandra @ Netflix

- . 98% of streaming data is stored in Apache Cassandra
- . Data ranges from customer details to viewing history to billing and payments
- . Foundational datastore for serving millions of operations per second

- 30 million ops/sec on most active single cluster
- 500 TB most dense single cluster
- 9216 CPUs in biggest cluster

O(100) Clusters
O(10000) Instances
O(10,000,000) Replications per second
O(100,000,000) Operations per second
O(1,000,000,000,000,000) Petabytes of data

dtsx.io/cassandra-at-netflix

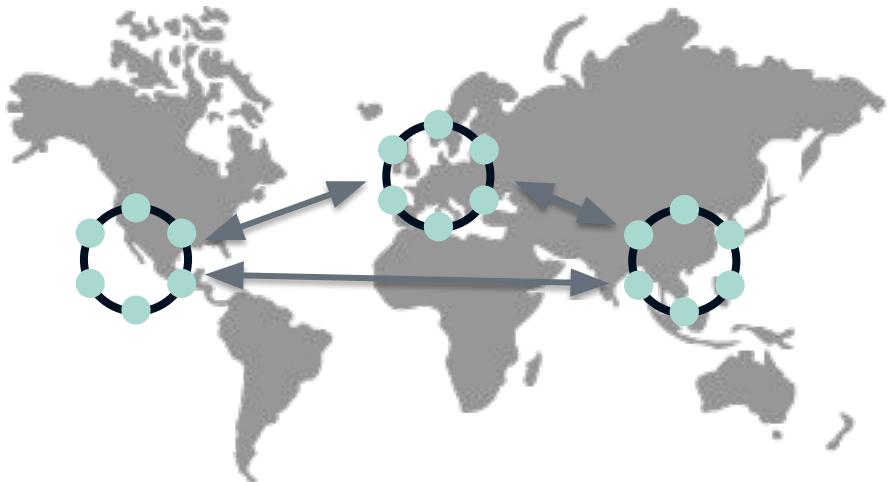


And many others...

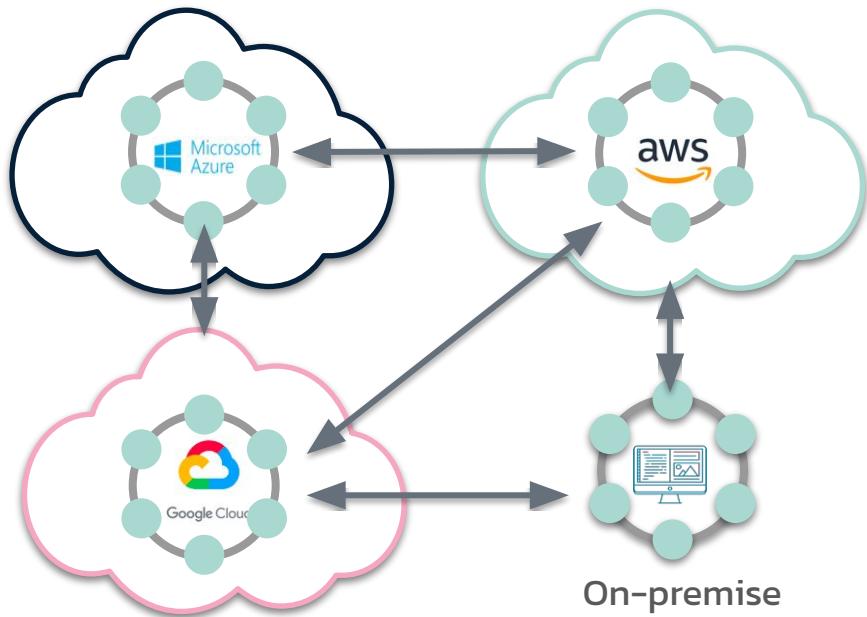


Cassandra Biggest Users (and Developers)

Geographical Distribution



Hybrid-Cloud and Multi-Cloud

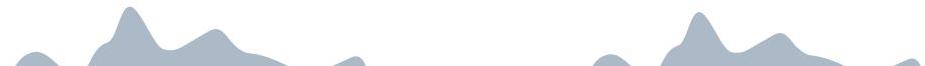


Data is globally distributed

State-of-the-art NoSQL Database

Astra DB

- DBaaS, serverless, auto-scalable
- Multi-cloud, distributed, multi-node cluster
- NoSQL, multi-model
- Tabular, document, key-value
- Based on open-source *Apache cassandra*



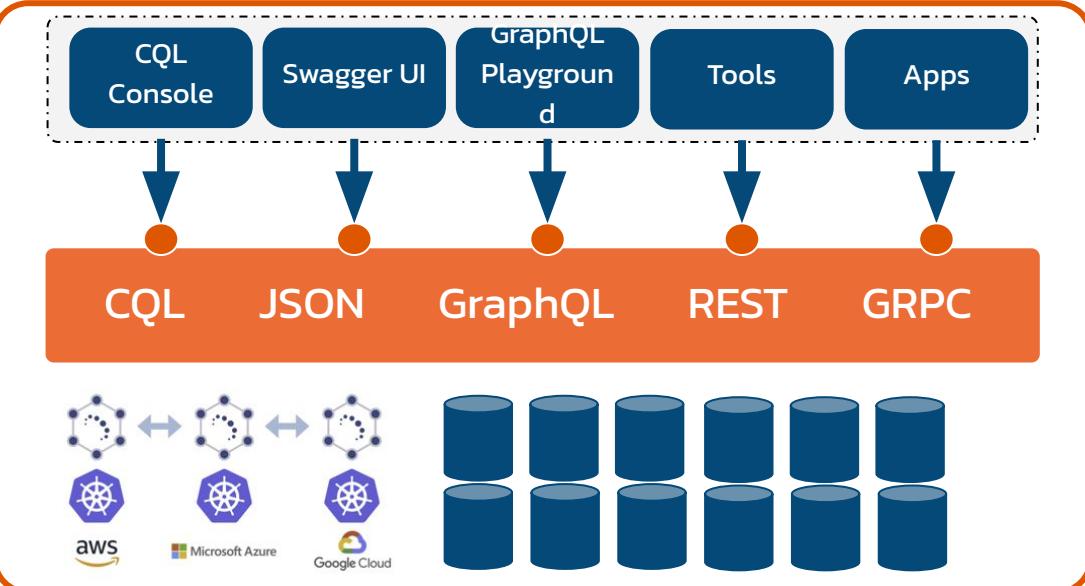
DataStax Developers



Astra DB

\$25/month credit

Launch a database in the cloud with a few clicks, no credit card required.



User Interface
Web-based developer tools and apps



OSS Stargate.io
A data gateway to allow multiple usages



OSS Apache Cassandra
A tabular NoSQL database



DataStax Developers





+



=

**K8SSANDRA**



Lab 1

Create AstraDB Instance

<https://github.com/datastaxdevs/workshop-astradb#1-create-astra-db-instance>



01

Intro./Lens

02

NoSQL

03

Cassandra/Astra/
k8ssandra

04



Quarkus

05

Microservices = Quarkus
+ Cassandra

06

Live Coding

07

Resources



Agenda

Java Designed for a Different Time



Traditional

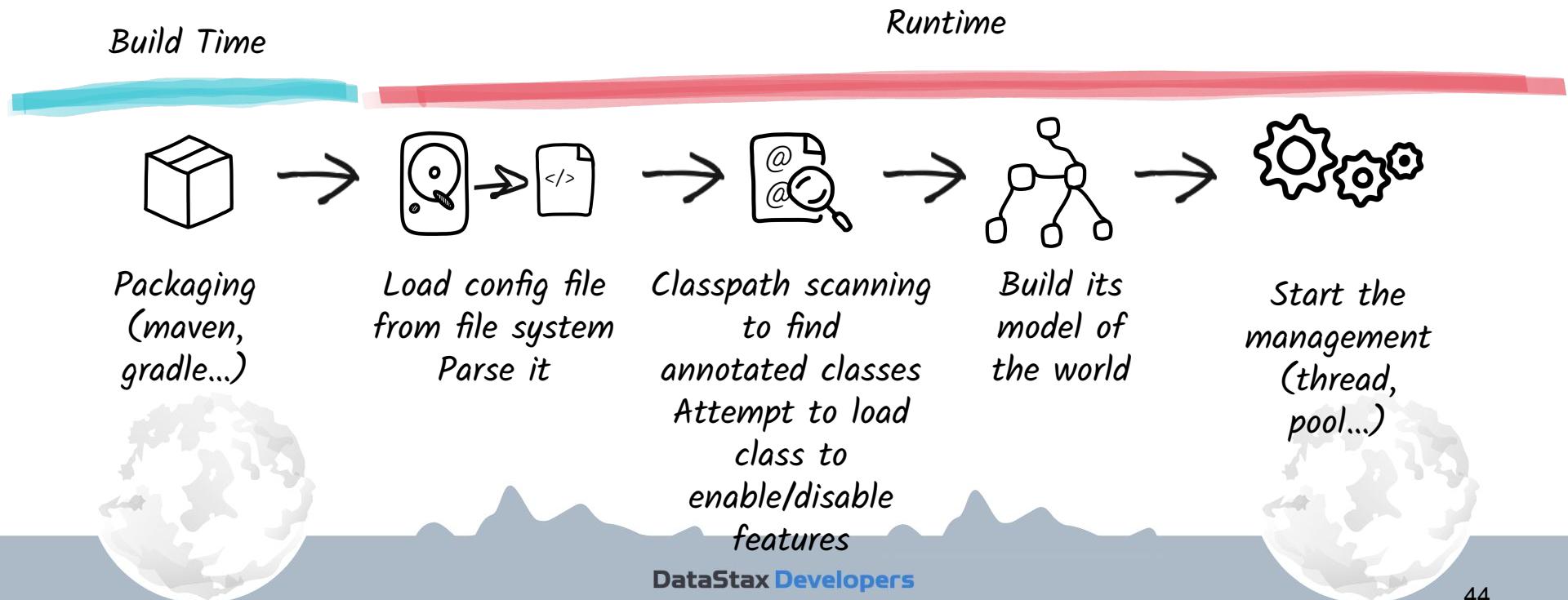
- Throughput at the expense of **footprint**
- Long running at expense of **startup speed**
- Rich, dynamic behavior for mutable systems



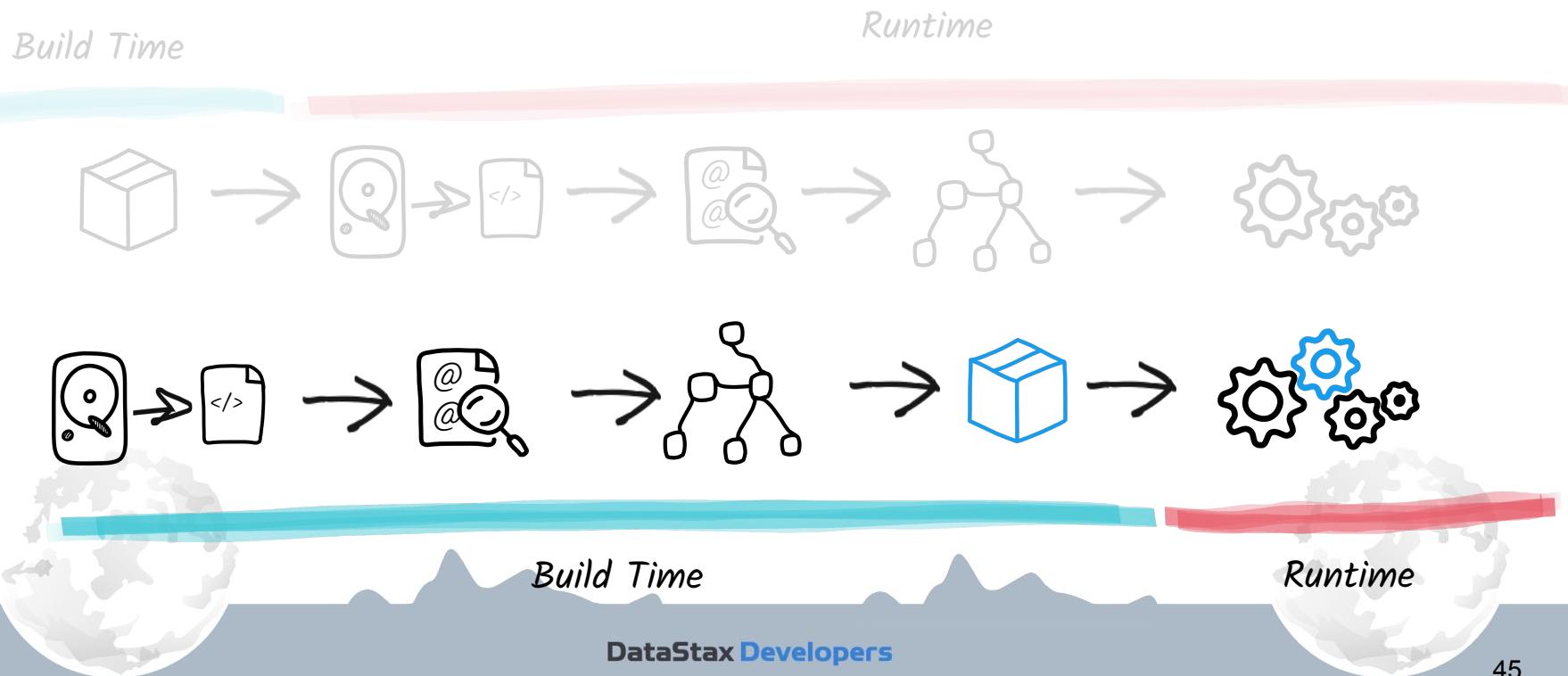
Cloud Native

- Throughput solved by scaling
- Ephemeral, immutable systems
- Footprint and performance matter

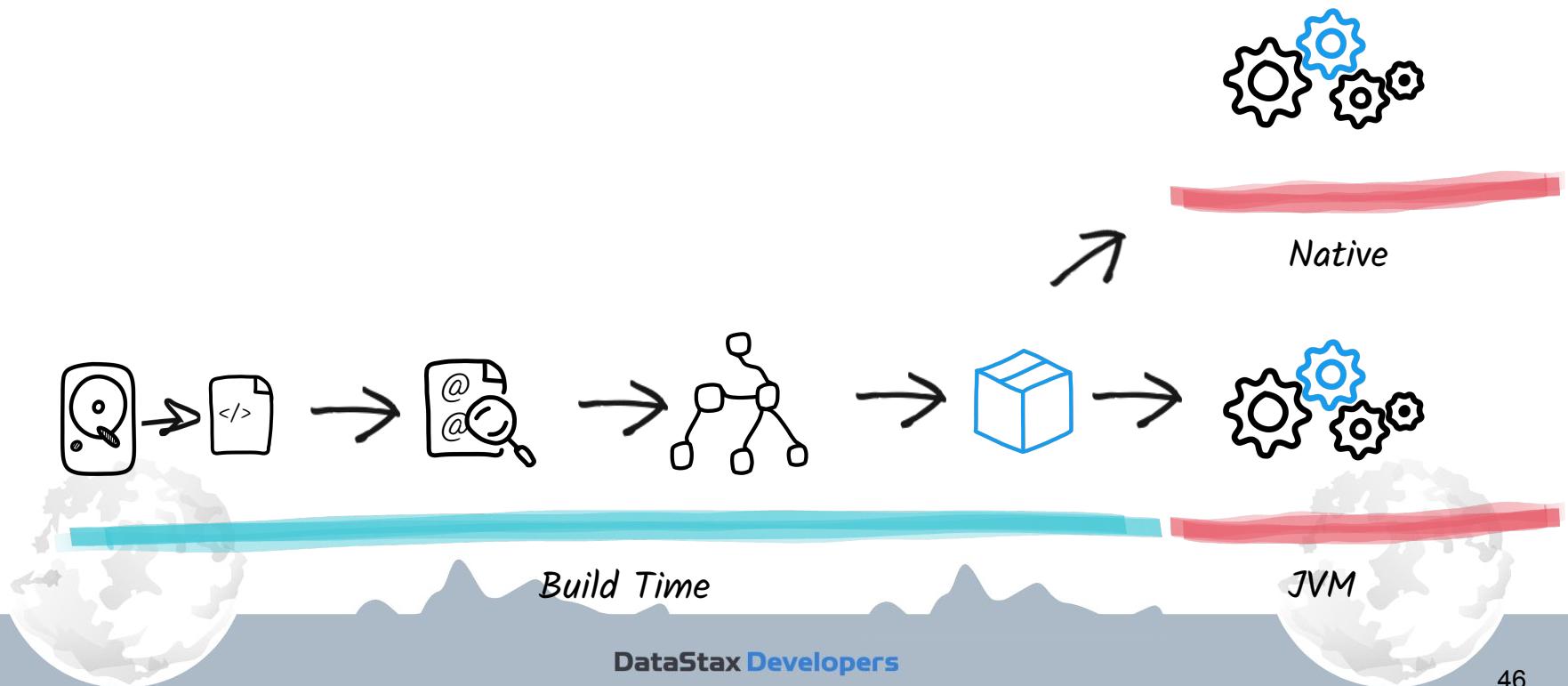
How does a Typical Java Framework Work?



The Quarkus Way



The Quarkus Way Enables Native Compilation





IT'S STILL JAVA!



Inner loop == Developer Productivity

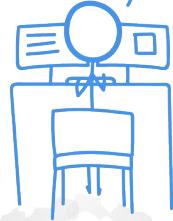
"Our developers used to wait **2 to 3 mins** to see their changes. **Live coding** does away with this!"

A cohesive platform for Microservices

- Developer productivity
 - Zero-config Live coding
 - Developer services
 - Continuous testing
 - Dev UI and CLI
- Streamlined code for the 80% common usages
- Native executable generation

WAIT.
SO YOU JUST SAVE IT,
AND YOUR CODE IS RUNNING?
AND IT'S JAVA?!

I KNOW, RIGHT?
SUPersonic Java, FTW!



Kubernetes Native Java with Quarkus

- Quarkus is an industry leader in startup time and memory utilization for native and JVM-based Java applications
- Building a native executable is easy
- Quarkus runs Java applications more efficiently instead of making a costly switch to Node.js or Golang
- Developers can utilize their existing Java ecosystem expertise with build tools and APIs like Jakarta EE, MicroProfile, Spring, and more, with imperative or reactive programming styles – or both!
- Kubernetes is a first-class deployment platform in Quarkus
- Quarkus unites the benefits of rapid development seen in scripting languages, the maturity of the Java ecosystem, and the efficiency of native compilation in a single runtime

by



Jason Greene

[FOLLOW](#)

Quarkus Co-Founder | Distinguished Engineer @ Red Hat



John Clingan

[FOLLOW](#)

Senior Principal Product Manager at Red Hat



Eric Deandrea

[FOLLOW](#)

Senior Principal Developer Advocate at Red Hat



<https://www.infoq.com/articles/native-java-with-quarkus>

01

Intro./Lens

02

NoSQL

03

Cassandra/Astra
k8ssandra

04

Quarkus

05



Microservices = Quarkus
+ Cassandra

06

Live Coding

07

Resources



Agenda

Monolith

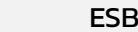
90s



Multi Tiers 2000

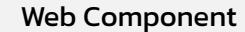


SOA (2005)



Microservices Architecture evolution

Microservices (2015)



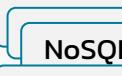
Backend for frontend

Microfrontend



Service Service

MicroServices



Data Mesh



ADVANTAGES



- Reduce Cost (Scaling, Design)
- Reduce Risk (resilience)
- Increase Release Speed
- Enable Visibility (security, monitoring)

DISADVANTAGES



- Complexity (Security, Transaction, Orchestration)
- Cultural Changes
- Bigger RUN footprint



Microservices



Quarkus Cassandra Extension

- Native Quarkus Config
- Cassandra Driver Session Support
- Cassandra Driver Object Mapper Support
- Support for Mutiny Types (Reactive Types)
- Native Image Support
- Support for DataStax Astra (Cassandra DBaaS)

Native Quarkus Config

```
quarkus.cassandra.cloud.secure-connect-bundle=/path/to/astra/bundle.zip
```

```
quarkus.cassandra.keyspace=ks1
```

```
quarkus.cassandra.auth.username=alice
```

```
quarkus.cassandra.auth.password=s3cr3t
```

```
quarkus.cassandra.request.timeout=5 seconds
```

```
quarkus.cassandra.request.consistency-level=LOCAL_ONE
```

```
quarkus.cassandra.request.page-size=1000
```

```
quarkus.cassandra.metrics.enabled=true
```

```
quarkus.cassandra.health.enabled=true
```

Cassandra Driver Session Support

```
@Inject QuarkusCqlSession session;

@GET
@Produces(MediaType.APPLICATION_JSON)
@Path("/product/{id}")
public Uni<Response> findProduct(@PathParam("id") String id) {
    return session.executeReactive("SELECT * FROM table_name WHERE key = " + id)
        .toUni()
        .map(row -> new Product(row.getString("id"), row.getString("name")))
        .map(product -> Response.ok(product).build())
        .ifNoItem().after(Duration.ofSeconds(5))
        .recoverWithItem(Response.status(Status.NOT_FOUND).build());
```

Cassandra Driver Object Mapper Support

```
@Dao interface ProductDao {  
    @Insert Uni<Void> create(Product product);  
    @Select Uni<Product> findById(String id);  
    @Select Multi<Product> findAll();  
}  
  
interface ProductMapper {  
    @DaoFactory  
    ProductDao productDao();  
}  
  
@ApplicationScoped class ProductService {  
    @Inject ProductDao dao;
```

DataStax Developers

Support for Mutiny Types

```
@GET  
@Produces(MediaType.APPLICATION_JSON)  
@Path("/product/{id}")  
public Uni<Response> findProduct(@PathParam("id") String id) {  
    return dao.findById(id)  
        .map(todo -> Response.ok(todo).build())  
        .ifNoItem().after(Duration.ofSeconds(5))  
        .recoverWithItem(Response.status(Status.NOT_FOUND).build());  
}
```

01

Intro./Lens

02

NoSQL

04

Quarkus

05



Microservices = Quarkus
+ Cassandra

03

Cassandra/Astra/
k8ssandra

06

Live Coding

07

Resources



Agenda

todos

- What needs to be done?
- ✓ Test the TodoApplication
- ✓ Create a REST API Backend
- ✓ Connect the backend to Cassandra
- ✓ Have fun
- ✓ Register to Youtube channel

4 items left All Active Completed Clear completed (1)

List all tasks

Create a new Task

Mark a task as
completed/uncomplete

Delete a task



Specification of Service layer



todoitems

user_id	TEXT	K
item_id	TIMEUUID	C↑
completed	BOOLEAN	
title	TEXT	
offset	INT	

```
CREATE TABLE todos.todoitems (
    user_id      text,
    item_id      timeuuid,
    completed    boolean,
    title        text,
    PRIMARY KEY ((user_id),item_id)
);
```

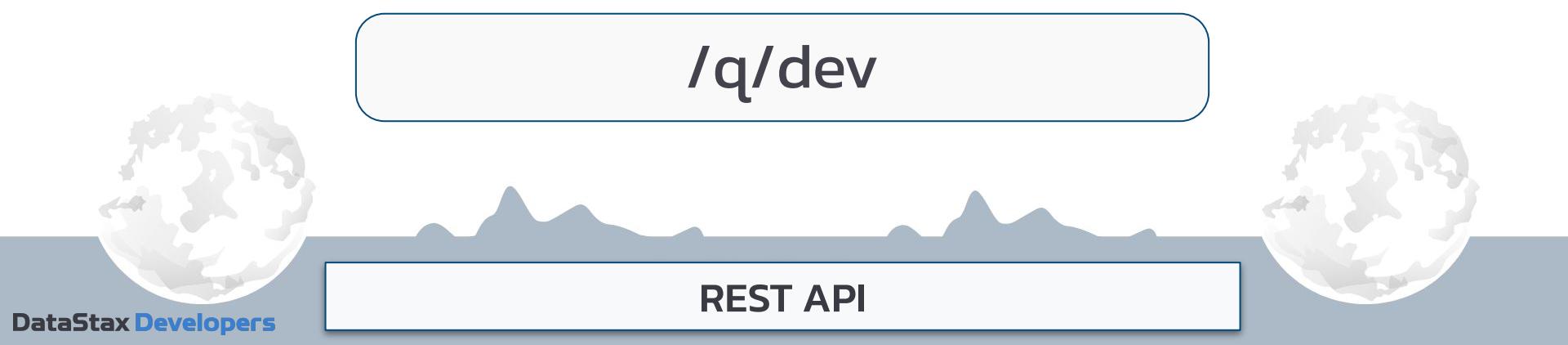


Data Model

- Our Partition key is user_id
 - We chose to have one todo list per user (avoiding any select * from table)
- Service
 - findTodos() for user
 - createTodo() for user
 - deleteTodo() from its id (userid + itemid)
 - updateTodo() from its id (both to mark it as complete and update title)
- REST API
 - The userid will appear in the URL
 - Provide major version (best practice)



/q/dev



REST API



Helping you **select** an MV* framework

Download

[View on GitHub](#)

[Blog](#)



<http://todomvc.com/examples/angularjs/>



A screenshot of a web application titled "todos". The interface includes a header with the title, a dropdown menu, and a list of tasks. The tasks are listed in a table with columns for the task name, completion status, and edit options. A footer at the bottom shows the number of items left and filtering options.

What needs to be done?		
<input type="checkbox"/>	Explain the use case	edit
<input type="checkbox"/>	Create the Data model	edit
<input type="checkbox"/>	Define the queries to perform	edit
<input type="checkbox"/>	Create the DDL	edit
<input type="checkbox"/>	Connect to Cassandra	edit
<input type="checkbox"/>	Create the CRUD repository	edit
<input type="checkbox"/>	Run the API	edit

7 items left All Active Completed



[TodoMVC.com](http://todomvc.com)



Todo-Backend

a shared example to showcase backend tech stacks

The Todo-Backend project defines a simple web API spec - for managing a todo list. Contributors implement that spec using various tech stacks. Those implementations are cataloged below. A spec runner verifies that each contribution implements the exact same API, by running an automated test suite which [defines the API](#).

The Todo-Backend project was inspired by the TodoMVC project, and some code (specifically the todo client app) was borrowed directly from TodoMVC.

Created and curated by [Pete Hodgson](#).

featuring HTTP APIs built with:



aiohttp



Akka



API Platform



Axon Framework



Azure Functions



CakePHP



Catalyst



Ceylon



Clojure



CoffeeScript



Compojure



CouchDB



Crystal



C#



django



.NET



Dropwizard



Elixir



ES6



express



express.js



Finatra



Finch

Swagger UI /q/openapi Explore quarkus-astra-intro-demo 0.01 (powered by Quarkus 2.3.1.Final)

quarkus-astra-intro-demo API 0.01 OAS3

/q/openapi

Astra TODO

- GET /api/todo/{list_id}
- POST /api/todo/{list_id}
- POST /api/todo/{list_id}/{id}
- DELETE /api/todo/{list_id}/{id}

Astra Demo CQL

- GET /hello

Schemas

```
Todo ∵ {  
    id      string  
    title   string  
    completed boolean  
}
```

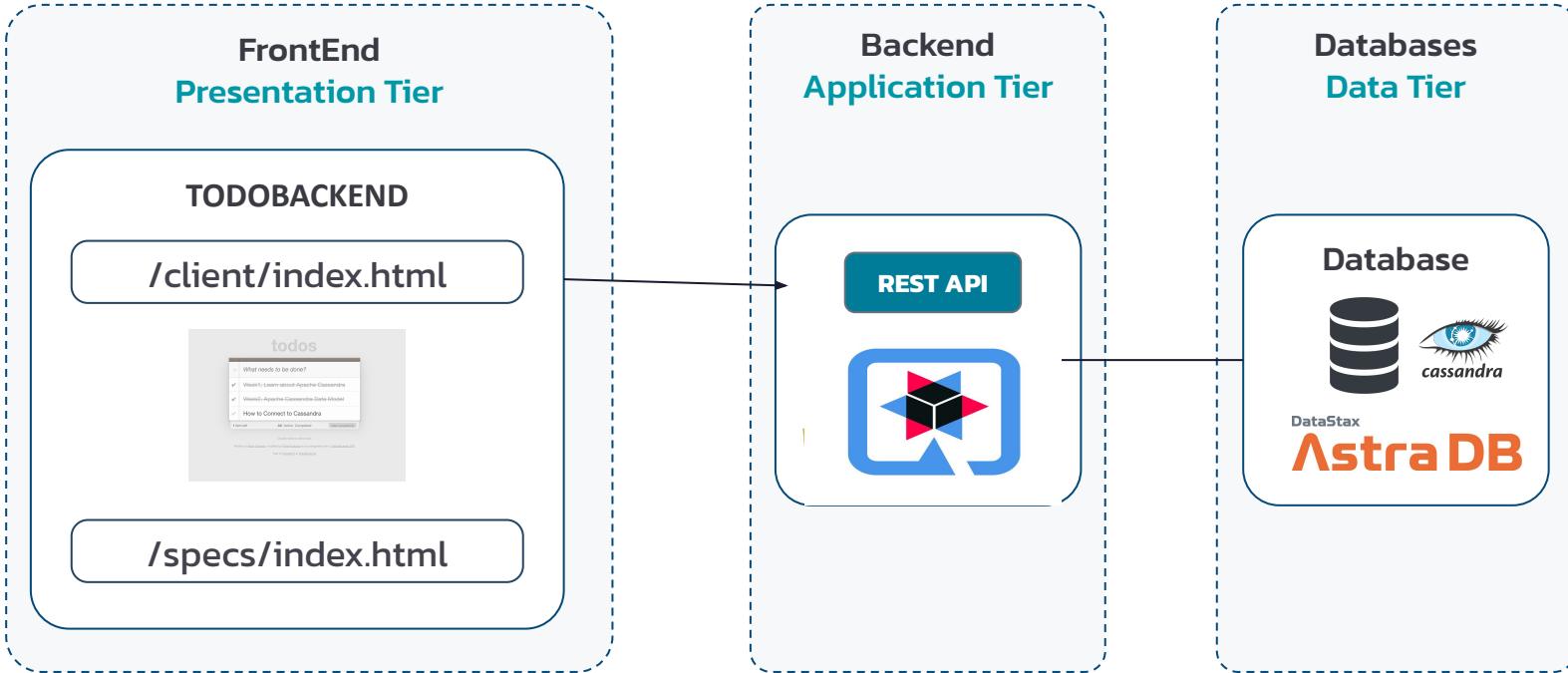
todos

What needs to be done?

- ✓ Test the TodoApplication
- ✓ Create a REST API Backend
- ✓ Connect the backend to Cassandra
- ✓ Have fun
- ✓ Register to Youtube channel

4 items left All Active Completed Clear completed (1)

Rest API Specifications (specs)



Architecture



Lab 2,3,4: Create Token ...

<https://github.com/datastaxdevs/workshop-intro-quarkus-cassandra#2-create-astra-token>





Lab 5,6: Setup App and Test connection

<https://github.com/datastaxdevs/workshop-intro-quarkus-cassandra#5-setup-your-application>





Lab 10:

Containerize, Deploy to Lens

<https://github.com/datastaxdevs/workshop-intro-quarkus-cassandra#10-containerizing>





Paused for Hands-on
(Don't miss Menti Quiz at end)



01

Intro

02

NoSQL

03

Cassandra/Astra/
k8ssandra

04

Quarkus

05



Microservices = Quarkus
+ Cassandra

06

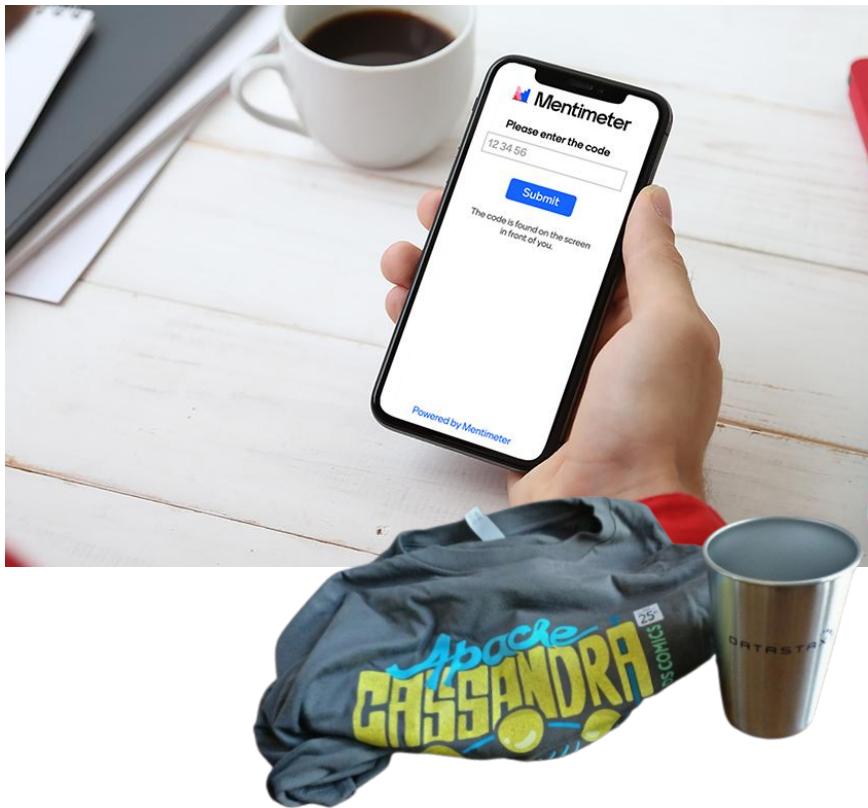
Live Coding

07

Resources



Agenda



Quiz on "Menti" !

SWAG WINNERS



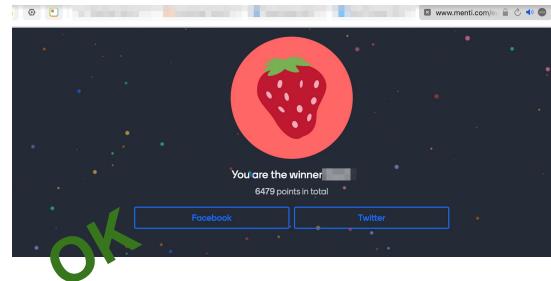
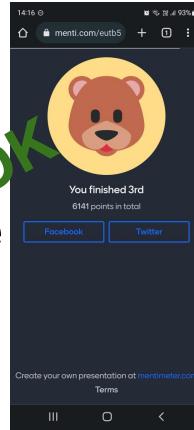
OK

Congratulations to 1st, 2nd and 3rd place
on the Menti quiz!

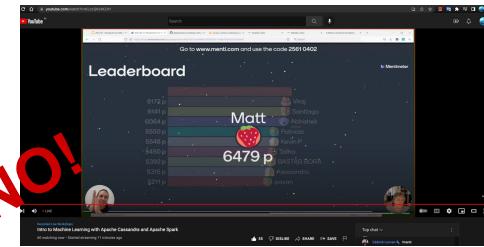
To claim your prize:

Take a screenshot of your Menti screen

Fill the form at dtsx.io/workshop-swag



OK



Swag Winners!



!discord

dtsx.io/discord

Screenshot of the DataStax Developers Discord server interface:

- Left Sidebar:** Shows categories like Événements, moderator-only, WELCOME, start-here, code-of-conduct, introductions, upcoming-events, useful-resources, memes, your-ideas, @the-stage, WORKSHOPS, workshop-chat, workshop-feedback, workshop-materials, upcoming-workshops, ASTRADB, getting-started, astra-apis, astra-development, sample-applications, and APACHE CASSANDRA.
- Center Chat:** The #workshop-chat channel. A pinned message from David Jones-Gilardi shows a screenshot of a presentation slide with nodes labeled Tagimbyba, Tegimbera, Nodobius, Argyreobius, and Drusobius.
- Messages:**
 - RIGGITYREKT (Hier à 21:14): I have a 5 node datacenter, 4 nodes are on dse version 5.1.20, one is on dse5.0.15. I am doing some mixed version testing for a class and the one node that is 5.0.15 is coming up as an analytics workload. I dont have /etc/default/dse, instead I am using /etc/init.d/dse-cassandra. how do I make that node start in cassandra workload, not in analytics?
 - RIGGITYREKT (Hier à 23:39): Okay I found out my issue, when i started DSE 5.0.15 it had endpointsnitch set to DseSimpleSnitch, the rest of my cluster is using PropertyFileSnitch, when i change it to PropertyFileSnitch, it still uses the simple snitch config. looking at the docs i see there is a way to go to GossipingPropertyFileSnitch, but i need the property file one. I can wipe this dbs, do anything with this node to get this done. how do i fix this? @here
 - Erick Ramirez (Aujourd'hui à 02:19): Okay I found out my issue, when i started DSE 5.0.15 it had endpointsnitch set to DseSimpleSnitch, the rest... mixed versions isn't supported and you're guaranteed to run into weird issues that will cause further problems down the track
 - Cedrick Lunen (Aujourd'hui à 09:01): When you start a node you have parameters -k for analytics, -g for graph and -s for search. To remove analytics check and remove -k
- Right Sidebar:** Lists PRESENTER — 1 (David Jones-Gilardi), HELPER — 7 (012345, AaronP, B1nary, Chelsea Navo, Jeremy Hanna, John Sanda, Patrick_McFadin), and EN LIGNE — 560 (-samu-, 6304-42JB, Aahlya, Abdurahim, abhi3pathi, Abhiis.s, Abhineet, Abirsh).

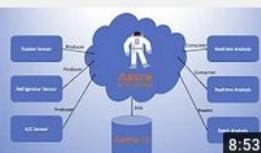


DataStax Developers Discord (18k+)



DataStax

Subscribe



Astra Streaming Demo
177 views • 2 weeks ago

Kubernetes Ingress Management with Traefik...
496 views • Streamed 2 weeks ago

Build your own TikTok clone!
1.9K views • Streamed 3 weeks ago

Build your own TikTok Clone!
4K views • Streamed 3 weeks ago

How to use the Connect Driver in Astra DB
113 views • 4 weeks ago

How to use the CQL Console in Astra DB
39 views • 4 weeks ago



How to create an Authentication Token in...
37 views • 4 weeks ago

How to use the Data Loader in Astra DB
62 views • 4 weeks ago

Astra DB Sample App Gallery
36 views • 4 weeks ago

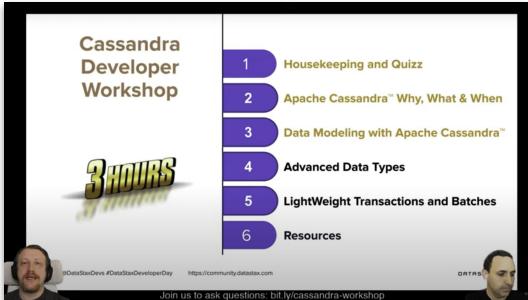
How to use Secure Connect in Astra DB
42 views • 4 weeks ago

Cassandra Day India: CL Room (Workshops)
2.4K views • Streamed 4 weeks ago

Cassandra Day India: RF Room (Talks)
1.3K views • Streamed 1 month ago

Live and interactive

Livestream: youtube.com/DataStaxDevs

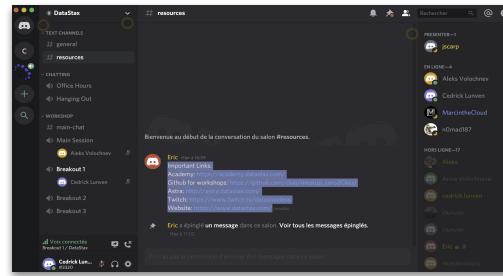


YouTube



Twitch

Questions: <https://dtsx.io/discord>



Discord

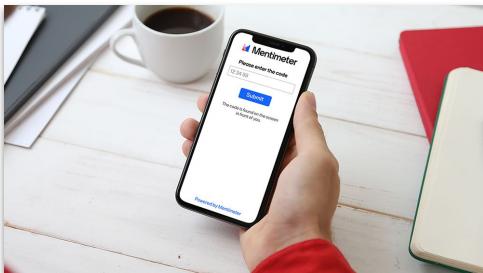


YouTube



Available on the iPhone App Store

Games menti.com



Mentimeter



GET IT ON
Google play



1

Attend the sessions

Database + GraphQL + PlayGround



This screenshot shows the Gitpod IDE interface. On the left is the 'EXPLORER' sidebar with project files like 'StargateDemoApplication.java', 'ManifestDemoApplication.java', 'resources', 'Dockerfile', 'gitpod.Dockerfile', 'README.md', and 'LICENSE'. The main area shows code snippets for 'StargateDemoApplication.java' and 'ManifestDemoApplication.java'. Below the code are tabs for 'PROBLEMS', 'OUTPUT', 'TERMINAL', and 'DEBUG CONSOLE'. The terminal shows the command 'gitpod /workspace/workshop-spring-stargate \$'. To the right are logos for 'npm', 'node.js', 'Maven', and 'Maven'. The bottom of the interface has buttons for 'Code', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'.

This screenshot shows a GitHub repository page for 'DataStax-Examples / todo-astra-jamstack-notify'. The repository summary indicates it's a 'Todo list JAMStack app' with 230 commits. The repository structure includes 'github/workflows', 'functions', 'public', 'src', 'env-template', 'gitignore', and '.gitpod.yml'. The 'Releases' section shows no releases published, with a link to 'Create a new release'. The GitHub logo is prominently displayed at the bottom.

datastax.com/workshops



3

Become a Jedi Master of Astra

- Showcase & explain Quarkus, how it enables modern Java development & the Kubernetes-native experience
- Introduce familiar Spring concepts, constructs, & conventions and how they map to Quarkus
- Equivalent code examples between Quarkus and Spring as well as emphasis on testing patterns & practices
- Chapters devoted to
 - Why the need for Quarkus in the first place?
 - Getting started
 - RESTful applications
 - Persistence
 - Event-driven services
 - Cloud environments, containers, and Kubernetes



<https://red.ht/quarkus-spring-devs>

- **Create a Quarkus Serverless Project**
 - **Run Serverless Functions Locally**
 - **Test Serverless Functions Continuously**
 - **Deploy Functions to a Knative Service on OpenShift**
- **Make Serverless Functions Run Faster With GraalVM**
- **Make Portable Functions Across Serverless Platforms**
 - **Deploy a Quarkus Funqy Application to AWS Lambda**
- **Bind CloudEvents on Knative With Quarkus Serverless Functions**



<https://dzone.com/refcardz/getting-started-with-quarkus-serverless-functions>

Try Cassandra+Quarkus

- Create your quarkus + cassandra app (code.quarkus.io or running the following):

```
$ mvn io.quarkus:quarkus-maven-plugin:2.9.1.Final:create \  
  -DprojectGroupId=io.quarkus.astra \  
  -DprojectArtifactId=quarkus-astra-demo \  
  -DprojectVersion=1.0.0 \  
  -DclassName="io.quarkus.astra" \  
  -Dextensions="resteasy-reactive-jackson, micrometer-registry-prometheus, smallrye-openapi, smallrye-health, cassandra-quarkus-client"
```

- Stand up your Astra free database (astra.datastax.com)
- Check out <https://k8ssandra.io>
- Get coding + see docs for more info and try the Quarkus + Cassandra workshop

<https://quarkus.io/guides/cassandra>

<https://github.com/datastaxdevs/workshop-intro-quarkus-cassandra/>

Thank You!

