

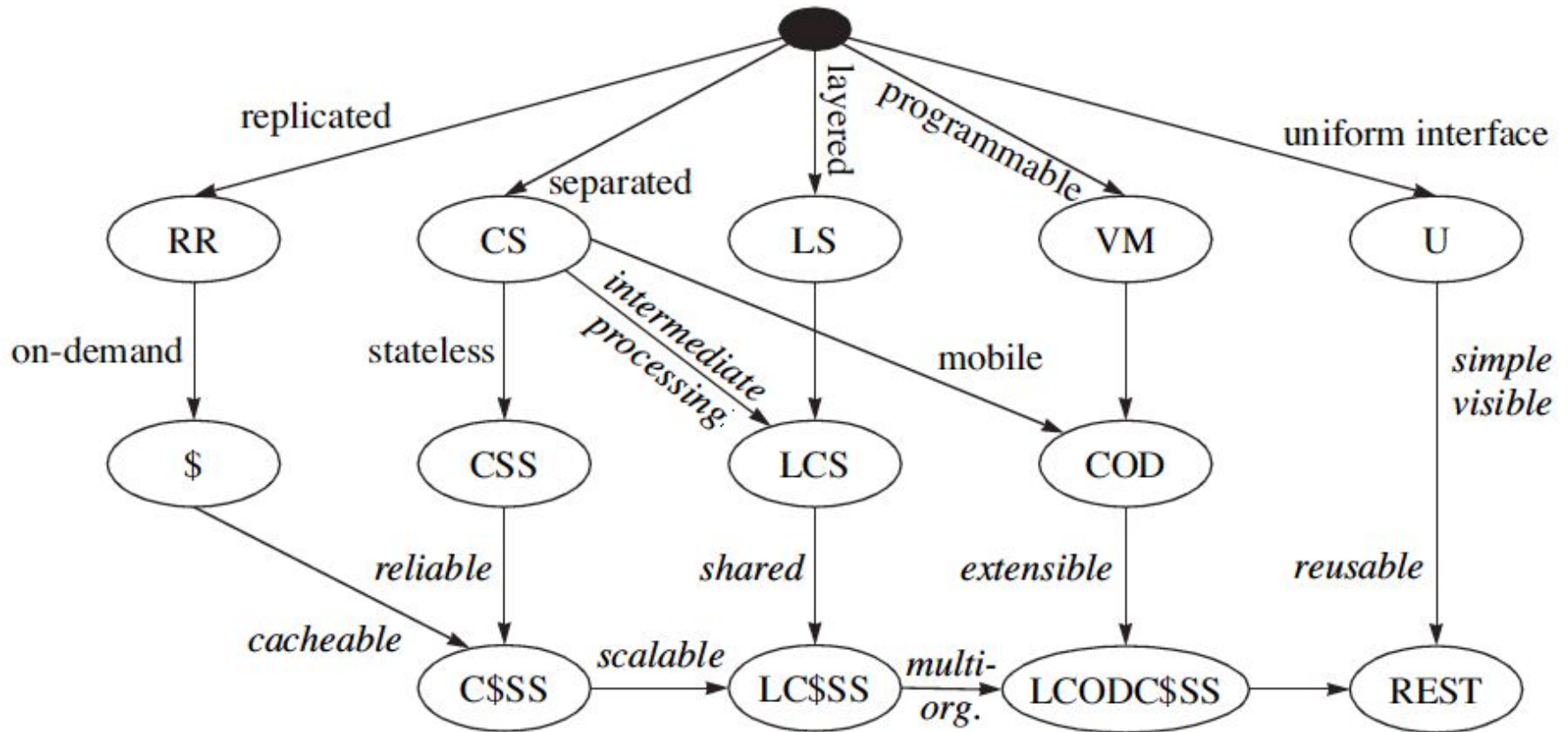
REST Continued!

Oxford University
Software Engineering Programme
April 2021



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

REST Derivation from Style Constraint



Key concepts

- Client Server
- Caching
- Replicable
- Stateless
- Layered
- Uniform interface



Cacheing

- Large scale network systems often rely on cacheing
 - Reduce traffic
 - Localised access
 - Reduced processing
 - **Akamai** and others make the web work effectively
- Caching relies on differentiating between cacheable and not cacheable traffic
 - Also understanding the lifetime and status of cacheable objects

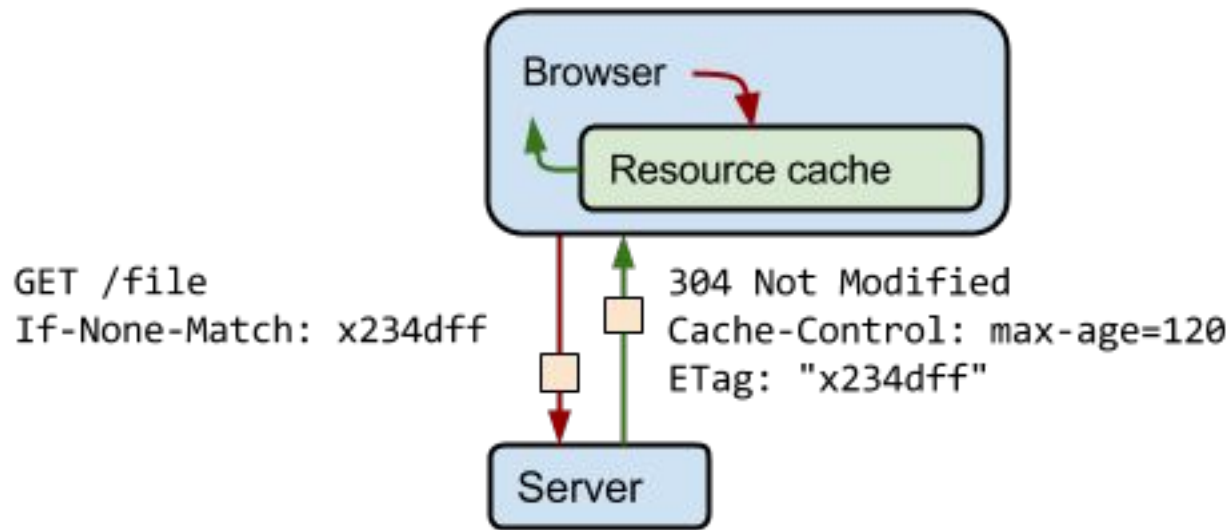


HTTP cacheing features

- Expires header
 - Cache-control header
 - If-modified-since / Not modified
 - ETags (Entity Tags)
 - Uuids for content
 - Strong and Weak
 - If None Match
-
- Unfortunately some websites are using Etags to track users instead of cookies!



ETags



<https://developers.google.com/web/fundamentals/performance/optimizing-content-efficiency/images/http-cache-control.png>



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Statelessness

- Of course there is state!
- The only question is where the state is kept.
- Traditional CS systems required the client and server to be kept in sync
- The web uses cookies



Layered systems

- Reverse proxies
- Composable
 - E.g. my business process is a service that exposes a REST interface and co-ordinates other services
- Compare with web scraping



Representation

- State of resource captured and transferred between components
- Might be current or desired future state
- Represented as data plus metadata (name–value pairs)
- Metadata includes control data, media type
- The **Content-Type** of the resource should be useful and meaningful (self-description)
- One resource might have several representations
- Selected via separate URIs, or via content negotiation



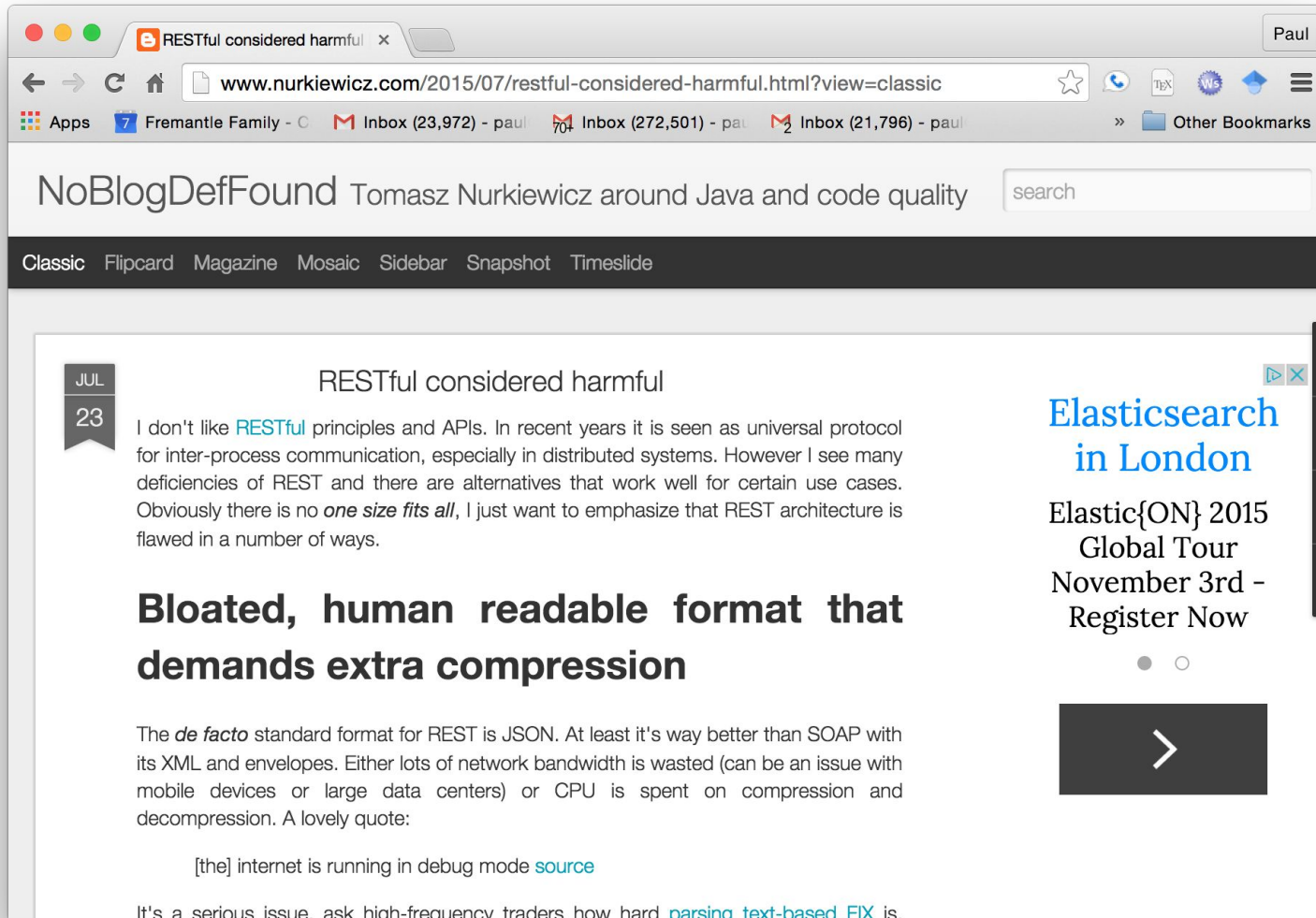
Uniform Interface

- A Uniform, Constrained Interface.
When applying REST over HTTP, stick to the methods provided by the protocol
 - GET, POST, PUT, and DELETE.
- These should be used properly
 - GET should have no side effects or change on state
 - PUT should update the resource “in-place”



Not everyone agrees:

<http://www.nurkiewicz.com/2015/07/restful-considered-harmful.html>



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

Anti-REST concerns

- Bloated formats (equally applies to SOAP)
- Neither Schema nor Contract
- APIs and discovery instead of clear published machine-readable documentation
- No inbuilt batching, paging, sorting, etc
- CRUD only
- HTTP Status codes mixed with business replies
- Temporal Coupling
- Not clear enough what is REST and what isn't!
- Backwards compatibility



Why REST Keeps Me Up At Night

News, Mobile

May. 15 2012 By [Guest Author](#)



This guest post comes from Daniel Jacobson ([@daniel_jacobson](#)), director of engineering for the [Netflix API](#). Prior to Netflix, Daniel ran application development for [NPR](#) where he created the [NPR API](#), among other things. He is also the co-author of [APIs: A Strategy Guide](#) and a frequent contributor to [ProgrammableWeb](#) and the [Netflix Tech Blog](#).

With respect to Web APIs, the industry has clearly and emphatically landed on REST as the standard way to implement these services. And for good reason... REST, which is generally implemented as a one-size-fits-all solution, is an excellent choice for a most companies who wish to expose their content to third parties, mobile app developers, partners, internal teams, etc. There are many tomes about what REST is and how best to implement it, so I won't go into detail here. But if I were to sum up the value proposition to these companies of the traditional REST solution, I would describe it as:

REST APIs are excellent at handling requests in a generic way, establishing a set of rules that allow a large number of known and unknown developers to easily consume the services that the API offers.

<http://www.programmableweb.com/news/why-rest-keeps-me-night/2012/05/15>

Alternatives to REST

- gRPC
- Async and Events
- GraphQL



gRPC and Events

Covered in other lectures



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

GraphQL



Describe your data

```
type Project {  
  name: String  
  tagline: String  
  contributors: [User]  
}
```

Ask for what you want

```
{  
  project(name: "GraphQL") {  
    tagline  
  }  
}
```

Get predictable results

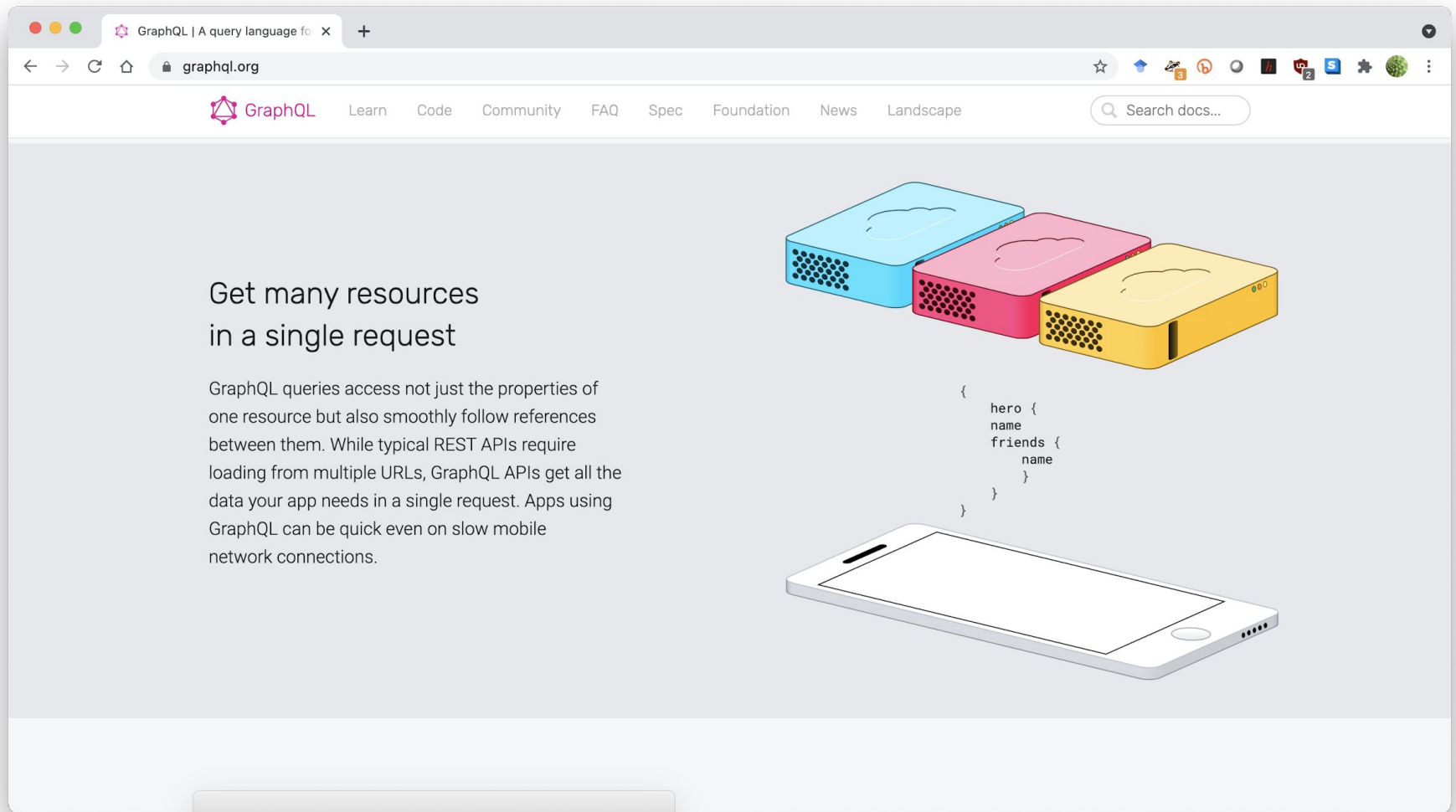
```
{  
  "project": {  
    "tagline": "A query language for APIs"  
  }  
}
```



intuit®



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>

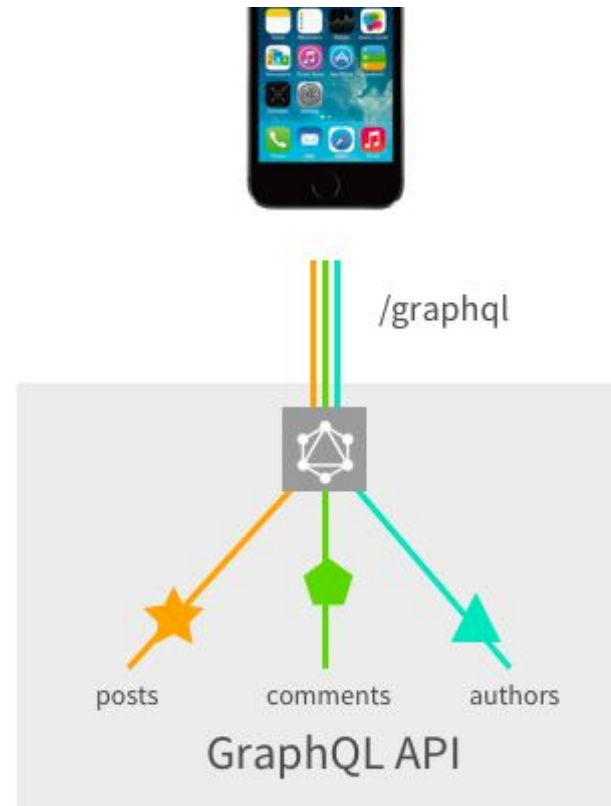


GraphQL

- A Facebook developed language
- Supports getting many resources with a single call
- Allows the caller to specify what data is needed
- A single endpoint to access multiple resources with a well defined type system



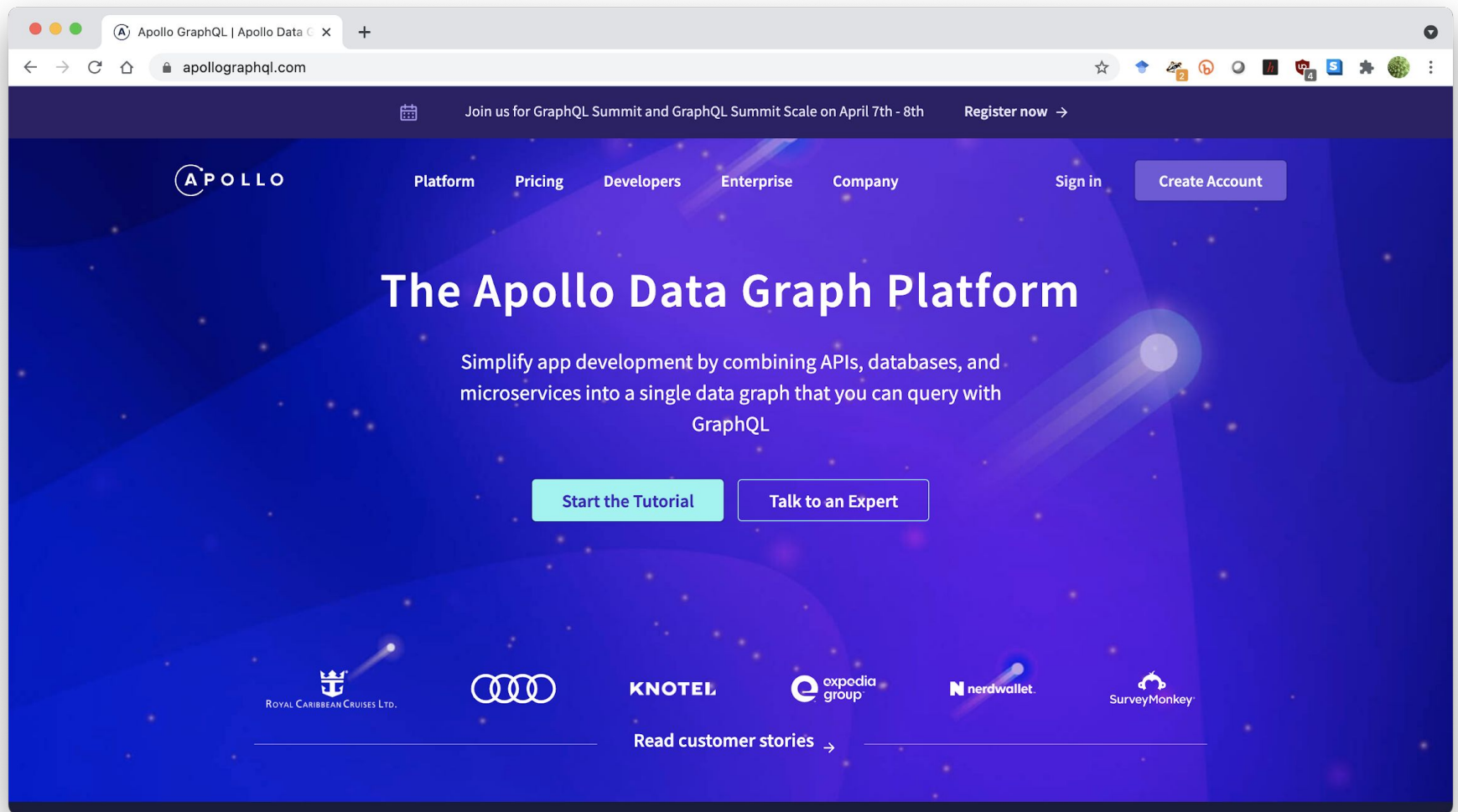
GraphQL vs REST



<https://blog.apollographql.com/graphql-vs-rest-5d425123e34b>




© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>




Hasura | Instant GraphQL APIs x +

hasura.io


The Hasura User Conference is back! Registrations and Talks submissions are open.



 **HASURA**

PRODUCTS RESOURCES PRICING CONTACT SALES LOG IN [GET STARTED FOR FREE](#)  Star 21.3k

Instant GraphQL for all your data

Hasura's open-source engine gives you instant GraphQL & REST APIs that unify your data and power modern applications



 **This website uses cookies.** By using this website, you consent to our use of cookies in accordance with our [Terms of Use](#) and [Privacy Policy](#) X 



graphql/graphql: GraphQL & t x +

github.com/graphql/graphql

README.md

GraphQL IDE Monorepo

Note: The primary maintainer @acao is on hiatus until December 2020 **Looking for the GraphQL Docs?:** This is the root of the monorepo! The full GraphQL docs are located at [packages/graphql](#)

Node.js CI passing chat 45 online coverage 66% typescript 89.4% languages 5 vulnerabilities 0 code quality: js/ts A
lgtm alerts 18 cli best practices in progress 88%

Overview

GraphQL is the reference implementation of this monorepo, GraphQL IDE, an official project under the GraphQL Foundation. The code uses the permissive MIT license.

Whether you want a simple GraphQL IDE instance for your server, or a more advanced web or desktop GraphQL IDE experience for your framework or plugin, or you want to build an IDE extension or plugin, you've come to the right place!

The purpose of this monorepo is to give the GraphQL Community:

- a to-specification official language service (see: [API Docs](#))
- a comprehensive LSP server and CLI service for use with IDEs
- a codemirror mode
- a monaco mode (in the works)
- an example of how to use this ecosystem with GraphQL.
- examples of how to implement or extend GraphQL.


Latest Stable Ecosystem



dosco/graphjin: GraphJin - Build

github.com/dosco/graphjin

README.md



GraphJin - Build APIs in 5 minutes

GODOC

REFERENCE

GO REPORT

A+

LICENSE

APACHE-2.0

DOCKER BUILD

PASSING


CHAT

17 ONLINE

GraphJin gives you a high performance GraphQL API without you having to write any code. GraphQL is automagically compiled into an efficient SQL query. Use it either as a library or a standalone service.

Sponsors

GraphJin is an Apache-licensed open source project with its ongoing development made possible entirely by the support of these awesome backers. If you'd like to join them, please consider:



Go 95.0%

JavaScript 2.7%

HTML 0.7%

Makefile 0.5%

Shell 0.2%

Roff 0.5%

Other 0.4%



Questions?



© Paul Fremantle 2016 except where credited elsewhere. This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License
See <http://creativecommons.org/licenses/by-nc-sa/4.0/>