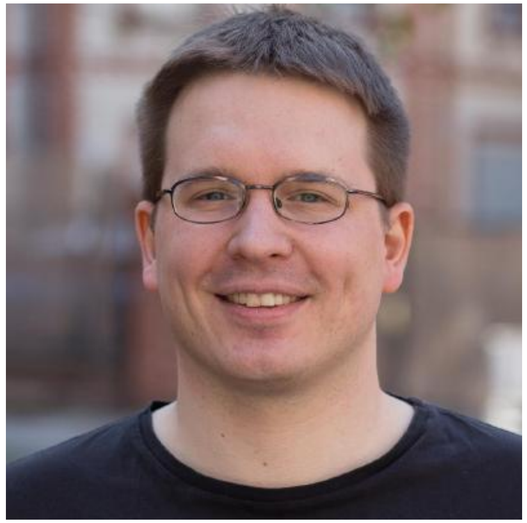
The future of custom Scalars

@andimarek



Andi Marek

@andimarek

graphql.de

ATLASSIAN

graphql-java.com

Agenda

- Custom scalars challenges
- The Specification Url
- Sneak preview: specification hosting

Custom Scalars

- Reminder: Scalars are the “atomic” types of GraphQL (next to enums)
- Build-in: String, Int, Boolean, Float and ID
- Customer Scalars are non build in Scalars
- Support for custom Scalars available since the beginning
- Common custom scalars: Long, Url, Json and DateTime

The DateTime motivation

- DateTime is a very common Scalar
- Often explained with ISO 8601 or RFC 3339
- But none of these standards is sufficient to really describe the Scalar in detail
- Example: "2017-01-10T10:00Z"
  Node => Invalid input
  Java => "2017-01-10T10:00:00Z"

The challenge with custom Scalars

- Black box implementation across languages and servers
- Only way to describe a custom Scalar is the `description` property and the name itself
- No overview of custom Scalars available or frequently used

What now?

- Define new build-in Scalars?
- How do we define the “right” custom Scalars?
- What about the existing ones?
- Why is Scalar A more important than Scalar B?

GraphQL Working Group

- Group meeting every month (online)
- Focus on GraphQL standard specification
- Everybody can join and contribute: https://github.com/graphql/graphql-wg
- Organized by the GraphQL Foundation

Custom Scalar Url

- Instead of trying to define new Scalars, we decided to improve the tooling around custom Scalars
- New “specificationUrl” property per custom Scalar
- scalar MyCoolScalar @specifiedBy(url: “https://my-cool-scalar.com”)
- Available via Introspection
- Championed by Evan Huss and myself, but ultimately it is the whole GraphQL community who made this possible
- All the Details: https://github.com/graphql/graphql-spec/issues/635

Specification Url Rules

- Url should point to a human readable specification of the Scalar
- The specification should be stable (non breaking)
- Changes to the Specification should mean new Url
- The specification doesn’t dictate the Scalar name:
  e.g. two DateTime can have different specification Urls
  or DateTime and OffsetDateTime can have the same specification Url

How does this solve our Problems?

- It doesn’t directly solve our problem, but makes it rather possible to solve them
- Problems are avoided because the Scalar name is decoupled from the implementation
- The specification Url is machine readable which allows tooling around it

Specification Url status

- GraphQL specification PR is accepted and will be merged soon
- Specification Url is already implemented in some server
- Tooling around it will hopefully improve over time

Next step: Specifications ecosystem

- The specification Url is unstructured: can be point to anything
- How do you write a good specification?
- Where do you host it?
- Which specifications are available?
- Which one are used frequently?

Sneak preview: scalars.graphql.org

- We are working on an easy way to contribute and write custom Scalar specifications
- Specifications will be available at https://scalars.graphql.org/<user>/<spec-name>/<version>
- Example: https://scalars.graphql.org/andimarek/datetime/1
- Will provide guidelines and templates
- Warning: this is not finalized

Thank you

@andimarek

- Blog how to implement Custom Scalars: https://www.graphql.de/blog/scalars-in-depth/
- GraphQL Spec PR for specification Url: https://github.com/graphql/graphql-spec/pull/649
- GraphQL Working Group: https://github.com/graphql/graphql-wg
- Scalars hosting repo: https://github.com/graphql/graphql-scalars