2019-12-17

1. Introduction of attendees (5m)
2. Determine volunteers for note taking (2m)
3. Review agenda (2m)
4. Update on moving spec into GraphQL foundation (Ivan, 10m)
5. V1 Proposed Roadmap (Sam, 30m)
6. Review Implementations document (Sam, 10m)
7. Review Public GraphQL APIs, PR (Sam, 5m)
8. Discuss next steps (Sam, 15m)

Introduction of attendees (5m)

- Gabriel McAdams also present (not on attendees, open PR plz. :))

Review agenda (2m)

- Will be talking about the test suite during “Discuss next steps”
-
-

Update on moving spec into GraphQL foundation (Ivan, 10m)

- News, there is not currently any news!
- Pinged them and waiting for a reply.
- Last we heard they need to change the foundation to account for additional specifications.
- Asked to provide mission statement.
    - [Action Item - Ivan]: Provide one sentence mission statement.
- Ivan originally wanted to create spec under GraphQL org (but couldn’t at the time as it was owned by Facebook). We’re stuck with the “non-official” repo for the next few months. Should we move it to a new GitHub org “GraphQL-Over-HTTP” or similar, or is it okay to leave it at github/apis-guru/ ?
- Gabriel: this should be a decision for the GraphQL Foundation because the main reason is legal
- Jayden: we should move it sooner rather than later to avoid 404s and stale link issues
- Benjie: GitHub’s redirection should be fine so long as we transfer the repo rather than cloning it
- Ivan: So, should we create a new GitHub organisation, or stick with apis-guru?
- Jayden: move once and do it right [when we’re ready to move to GraphQL Foundation]
- Sam: I agree - extra moves are distracting

- Benjie: +1
- Ivan: if anyone has any concerns, open an issue.
- Jayden: lets add a paragraph at the top of the README to make it clear that this is a GraphQL official project rather than an APIs.guru branded project
    - [Action Item - Ivan] add paragraph / sentence explaining the state of the project legally migrating to GraphQL
- Gabriel: I'm satisfied by that

V1 Proposed Roadmap (Sam, 30m)

- [Shares screen] In this PR I summarised the different things we talked about; we had different perspectives so I wanted to lay out one potential roadmap for v1 so that we can work towards a shared understanding and a shared goal for v1. We might not go with this (it's a PR, not merged yet) but it opens it up for conversation.
- We should replace the vision statement with the mission statement?
    - [Action Item - Sam] ^
- Should the "will" be a "will" or a "should" regarding servers supporting later versions having to support older versions?
- Ivan: We should go for "strive for" rather than "has to"
- Versioning requires clarity, even though we're introducing it later
- Rather than referring to "V2" of the spec, can we refer to "future versions"
- Ivan: I think it'll ultimately be version 1.1 rather than version 2
- Gabriel: we might be able to black-box GraphQL entirely, so that the spec does not depend on the underlying implementation
- Jayden: if there's a functioning example GraphQL schema that supports all the things we need (mutations, etc) then that would be very useful
- Gabriel: you're right, there are things in the GraphQL spec that affect this sub-spec
- Jayden: starwars/pokemon are great, but we need a schema that has everything we need to test in it
- Sam: it might make sense for us to have our own schema that only supports our own needs
- Jayden: even being able to guarantee uptimes for ourselves would be really useful
- Ivan: GraphQL Foundation have a netlify account, uptime is good, we can make an example for express-graphql
- Ivan: if we use too many features then we exclude certain implementations - for example some implementations don't support SDL and only support class-based definition. We need to keep it to the minimum to create example servers really fast and not depend too much on GraphQL library itself for these tests.
- Jayden: Maybe a separate schema for each test might make sense.
- Gabriel: now I think we're getting into details that could be decided later
- Sam: we're getting into details, which makes it seem like we're generally happy with the approach
- Jayden: is batching okay to go into v1?

- Benjie: I think there’s 3 different implementations of batching and they all work slightly differently; I don’t think we should put it in V1
- Ivan: we should have a feature matrix. Right now the idea is to make all existing implementation valid, like Benjie called it the “Retroactive GraphQL HTTP Spec”
- Jayden: where can we read about the different versions of batched queries?
- Benjie: there’s an issue in the GitHub repo where you can read about it
- Michael: we’re implementing two versions of this in the .NET implementation; for the first version of this spec we should just keep it simple and have the most basic implementations in there
- Sam: I don’t see much pushback; please raise PRs with some feedback.
- Ivan: lets agree we’ll keep it open for a couple days and if no-one has any major objection it gets merged and people can still submit PRs later. 2 days? It’s a small document, I think that’s enough time
- Sam: I’d like to see a few more approvals, currently just Benjie and Ivan have approved it
- Jesse: I’d like to see the feature matrix as an action item. Especially if we can break down which softwares implement the different versions of things like batching.
    - [Action Item] - Feature matrix
- Sam: any volunteers?
- Jesse: I’m happy to fill in the details for Apollo; can the other people familiar with different implementations fill that out?
- Sam: it’d be great to automate a bunch of this by simply running the test suite against the different implementations
- Jesse: great point; maybe the first thing should be the example schema and some tooling around it
- Jayden: so would it be a repo with a huge bunch of folders and everyone creates their own little servers in there we can run the tests against
- Benjie: maybe using Docker?
- Sam: it’d be great to have people propose lots of different ways of running this so we can find a good way for everyone to contribute
- Spawnia: we should make it easy for the different server implementations to run the tests against their own implementations
- Michael: like a testman test suite
- Sam: would be great if we could just point the test suite at a URL and maybe turn on certain optional extensions and press go.
- Michael: there are other tools like postman that are very simple to [... breaks up because of train …]
- Ivan: a person who volunteers to test the first test stack, then we can iterate on that. To bootstrap something it’s important to start something.
- Ivan: initially we should keep the number of example servers low (3,4,5)
- Spawnia: I’d be willing to have a look at building the initial implementation. I’d like to package it up as a Docker image and use it as a GitHub action.

- Jesse: it’s not clear what the Docker images should be implementing. Maybe we should start with a google spreadsheet so that we can figure out what’s going to be in the test suite.
- Spawnia: we need a programming language to implement it because we can’t just use fixed inputs and outputs to test all the things we need to test.
- Jesse: that’s not exactly what I meant [gives way to Jayden]
- Jayden: I don’t think that we’d need to write any code specific to an implementation, it’s more DevOps: an array of URLs to the endpoints that we issue our tests against.
- Ivan: we cannot test exact snapshot because different implementations return different parsing errors, but we need to test that there is an error and that the status code is what we’d expect. We cannot use snapshot testing. Goal was to make sure that current spec text is compatible with all servers, and we cannot achieve that until we’ve written a test suite that can run the spec against all the existing servers.
- Jesse: in order to get the full spectrum of all the things that we want to propose as candidates for the things to test against.
- Jayden: both of these things are critical - we can run these in parallel - I don’t think I could contribute much to the tech stack, but I could contribute to the matrix
- Ivan: we should write a second document - “supports GET”, “Supports GET with operationName”, …
    - [Action Item] - Someone start turning the spec text into test cases
- Gabriel: one last thing before we move on; we spoke about the first version “not being a restrictive spec” - we don’t want to start right off the bat and say this is what “must” be true and not know that there’s something out there that doesn’t do it. “Should” is not testable, “must” is but we’re not after that currently. I’m not sure we should be writing a test suite right now.
-
-

Review Implementations document (Sam, 10m)

- Jayden: Should we be looking at other things too such as middleware, e.g. the graphql-upload middleware
- Benjie: also things like federation, gateways, etc that sit in the middle
- Jayden: yeah, anything that touches HTTP
- Jayden: maybe we should rename the names from “Popular Servers” to “Popular Server Implementations” to make it clear it’s not just the server itself.
- Jayden: people will complain if we have tests that don’t pass against graphql-upload middleware because it gets used in all these other Node servers.
- Ivan: we don’t want to introduce more dimensions - is it language based or ecosystem based? Supporting libraries or an additional category where it’s augmented the GraphQL HTTP spec rather than being part of the server.
- Jayden: it depends on what the intention of the list is - I think it’s just representing what the industry is up to

- Spawnia: I don't think we should be testing everything, we're just getting an idea of what the most popular things are so that we can have a first useful version of the spec, even though
- Gabriel: I disagree - we agreed that we should be making it so that everyone is compliant with V1
- Gabriel: if this is going to be a restrictive spec, then there's no way that everyone will be compliant. If we think of it in terms of what is allowed rather than what is not allowed then everything is compliant because there's no rejection; this is what we agreed previously.
- Ivan: I was working with an organisation that had implemented GraphQL over HTTP but had capitalised Query, Variables, OperationName, and so they couldn't use GraphiQL or GraphQL Playground against it because it wasn't supported out of the box.
- Ivan: we need to break some weird servers; this is actually a good thing. Playground, GraphiQL, and some other clients all behave on certain behaviour. I think we should be targeting the 97% of agreed features that clients need.
- Spawnia: we're not breaking things because they're already incompatible
- Michael: the minimal spec for me is to support what the GraphiQL example HTTP fetcher function supports.
- Michael: I think 98% of the popular servers support the basic syntax required by GraphiQL, Playground, etc
- Jayden: I'll do a PR to add some other implementations/middleware and rename the headings
- Gabriel: side note on the notes, in TC39 plenary meetings, we have 3 note takers to ensure that we don't miss anything. Perhaps we could do the same here.
- Ivan: since we have recording, someone can backfill the notes later
-

Review Public GraphQL APIs, PR (Sam, 5m)

- [Skip this?]
-
-
-
-
-
-
-

Should we include code samples in the spec?

- https://github.com/APIs-guru/graphql-over-http/pull/50/files

- Gabriel: TC39: Official code might restrict what the larger community puts forth - we should leave code up to the community, not up to the spec maintainers. Their situation is different than ours, in that code resides in proposals rather than within the spec itself, but I stand by the statement that we should not put our stamp of approval on any given code.
- Ivan:
- Ivan: one reason I disagreed with adding the code sample is that it gives the impression that GraphQL over HTTP is for browsers, but it's not - it's for everything, all different languages. Also it's quite big. And we already have the problem that people think that GraphQL is javascript-centric, and I don't want to emphasise this. But I agree if we cannot find any other way to be clear about spec text then including code should be fine.
- Gabriel: using pseudo-code in the spec is valid too
- Ivan: in the GraphQL spec we use pseudo-code, I think this works well
- Jayden: I agree with pseudocode
- Sam: I'd like to see examples more like what the HTTP request was/what the response should be
- Ivan: what you're trying to send vs what's sent over the wire
- [Action Item] Before public release we need to validate all RFC links to make sure what we're saying is correct - not just for this PR.
- Ivan: this is why it's important to have a test case - if we have a test case then we can check that the specs are correct
-

Discuss next steps (Sam, 15m)

- Tests
    - Feature matrix
        - Big features? I.e. persisted queries, etc.
        - [Action] Jesse to put info about Apollo
    - Test cases
        - [Action] OPEN List of tests that we should implement
        - Standard sample schema (for tests, not for example)
            - [Action] Jayden help
    - Technical framework for running tests
        - [Action] Sam help, others welcome!
    - Popular example servers running the sample schema
        - [Action] Ivan create graphql-js one Jayden: graphql-api-koa one
- Convert spec to spec-md
    - https://github.com/APIs-guru/graphql-over-http/issues/8
    - [Action] OPEN
- [Action] Sam to add link to the Graphql over http slack channel

Last Tuesday of the month at 19:00 - 21:00 UTC - January 28th

- Move to the GraphQL Foundation
- Set of running examples of ~5 of the most popular servers/clients with a standard, minimal GraphQL schema
- Test suite to automate testing of GraphQL servers compliance with the spec
  - Can be applied to examples of popular server or public GraphQL APIs
- Results of popular libraries and APIs compliance with current spec
- Structuring of existing spec to be easier to extend in later versions
- Fine detail focus on each of the main sections of the spec
- Update links to point to the GraphQL Foundation repos and websites not FB
- Adopt similar formatting/tooling for spec to match the GraphQL spec