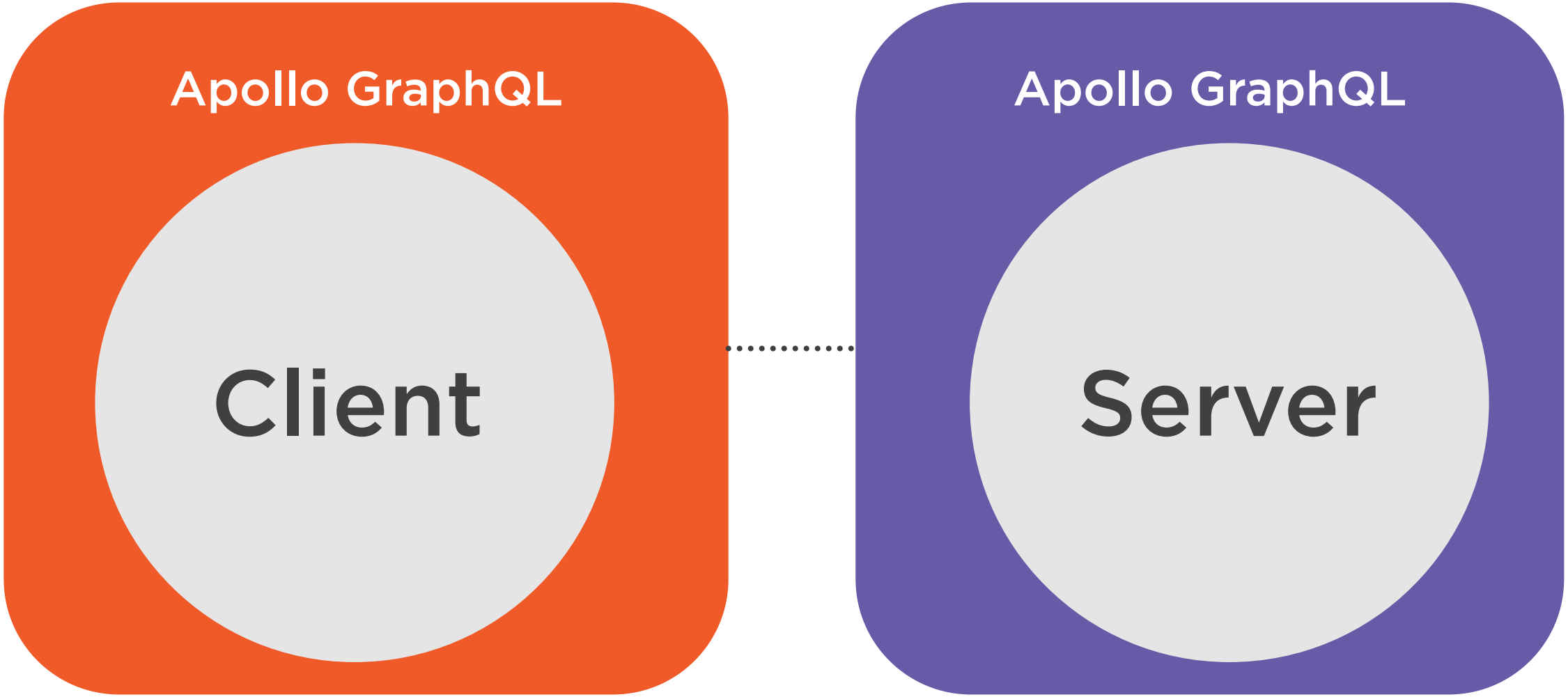
Apollo: Performance Management Playbook

BUILDING AN APOLLO GRAPHQL SERVER WITH NODE USING ONE JAVASCRIPT FILE

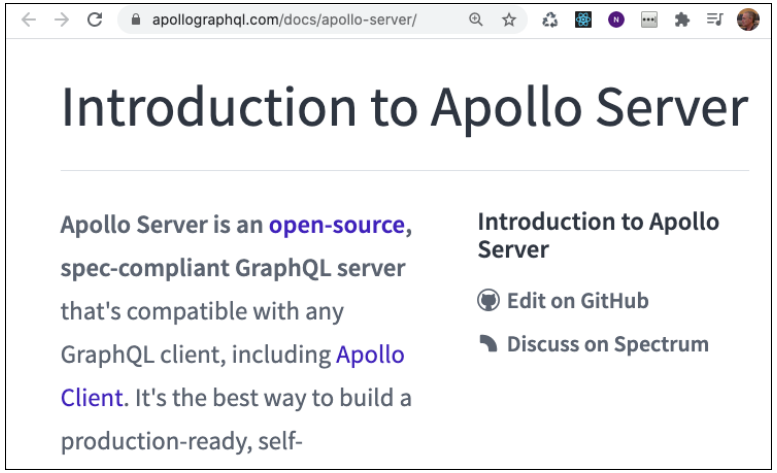
Peter Kellner
DEVELOPER, CONSULTANT AND AUTHOR

@pkellner linkedin.com/in/peterkellner99 ReactAtScale.com

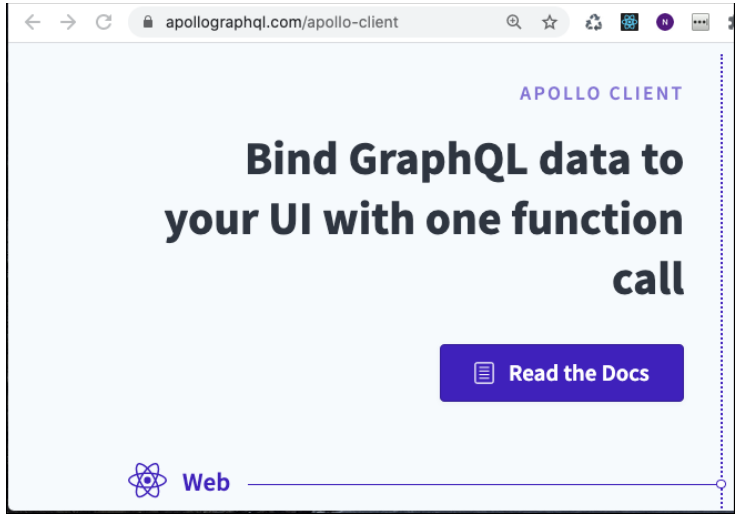
Benefits of Building Apps with Apollo



Open Source Projects from Apollo (MIT Licensed)



Apollo Server



Apollo Client

Benefits of Building Apps with Apollo

Development team productivity

Significant performance gains to developed products

Primary Benefits Breakdown

Apollo Server

Minimal setup time

Out of the box cache support

Sophisticated default resolver resolutions

Apollo Client

Intelligent caching

Declarative approach to coding features

Potential replacement for state management libraries like Redux

To learn best techniques,
need to put together both
Apollo Server and Client.

Coming up in this module,
Apollo Server setup.

Coming up in the next module, Apollo Client setup.

Coming up next,
adding read/write to our
GraphQL server by
implementing a REST
server with json-server.

Coming up next,
update our package.json
such that “start” launches
both json-server and our
GraphQL server.

Required for course

“Node”

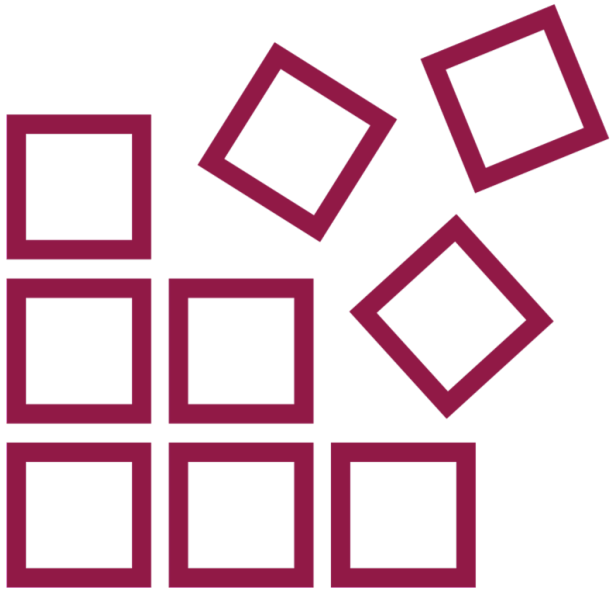
Building Our Apollo GraphQL Server

Put all JavaScript in a single file and launch with Node

Define a GraphQL Schema with GQL

Build standard and custom GraphQL resolvers

Learn how Apollo's automatic resolver resolution works



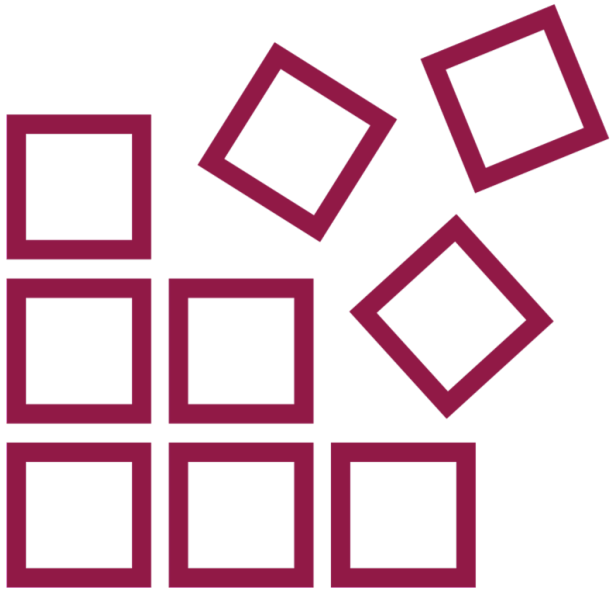
Goal to start simple and enhance

Focus on what differentiates Apollo

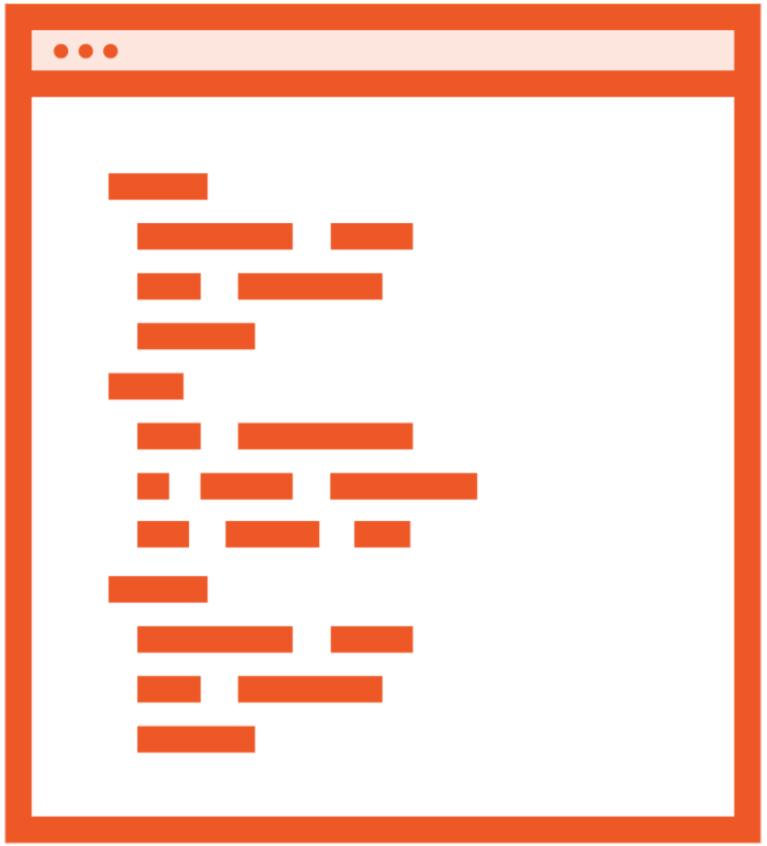
GraphQL and boiler plate code

Build a simple Apollo Server

Supporting query and mutation



Pseudo code coming...



```
Import {ApolloServer} from …


const typeDefs = gql`
  type Speaker { id: ID!, … }
  type Query { speakers: [Speaker]…}`
const resolvers = {
  Query: { speakers: (parent,args) =>
      { return speakerList }   }
const server = new ApolloServer() {
    typeDefs, resolvers };


server.listen();
```

- Import ApolloServer library
- Set up typeDefs
- Define resolvers
- Create an ApolloServer instance that includes typeDefs and resolvers
- Launch the server!

Resolvers

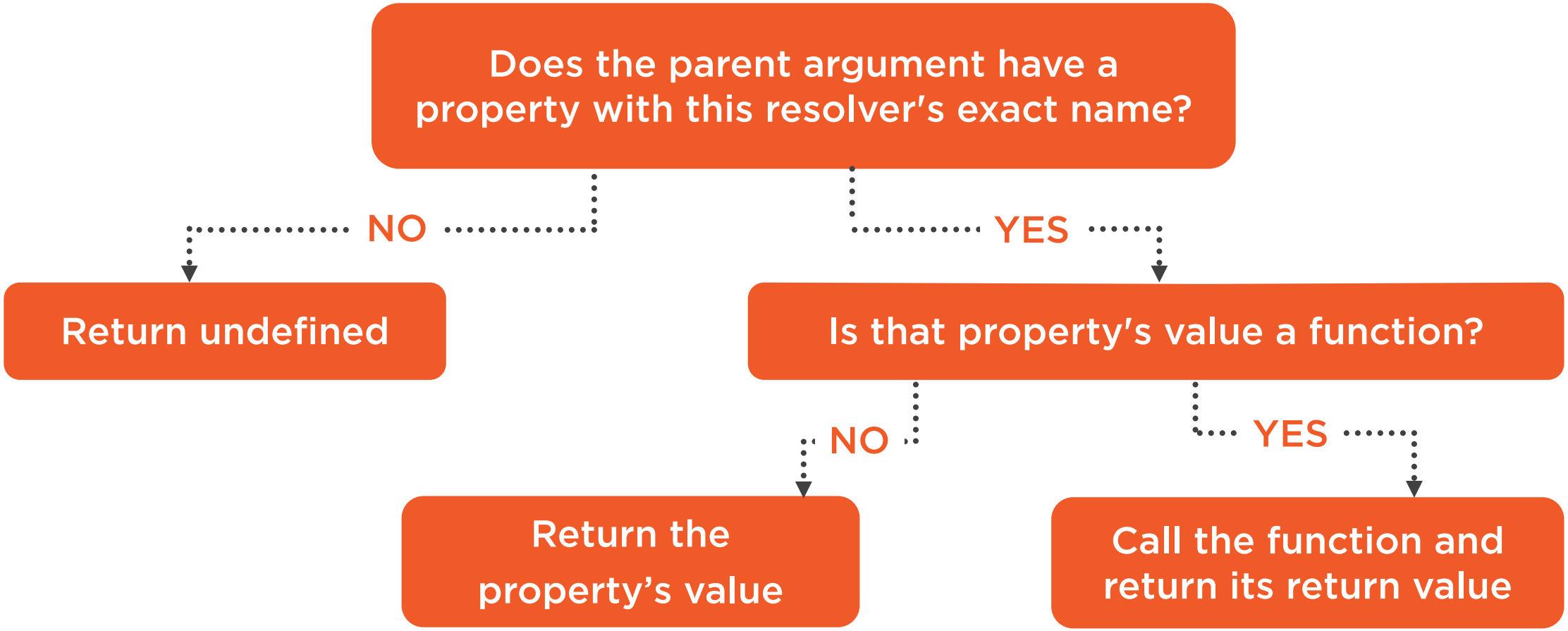
GraphQL spec

Resolvers are not part of the official GraphQL specification

One to one

Every field in the typeDefs (or schema) has a resolver field that matches it

Finding the Right Resolver in Apollo Server



Module goal: super simple Apollo GraphQL server

One single file, server.js

Apollo GraphQL Root Query Types

```
type Query {
  speakers: SpeakerResults
}

type Mutation {
  updateSpeaker(speaker: SpeakerInput): Speaker
  addSpeaker(speaker: SpeakerInput): Speaker
  deleteSpeaker(speakerId: Int!): Speaker
}
```

Persistent Store Needed for Updates

Local Storage

Database

REST server

Benefits of Using json-server in Development

No production server or copy needed

Keep ahead of server production code

REST calls spec compliant

Support both small and large data sources

Add json-server to our Apollo GraphQL server project

Next module: Apollo Client using React as well as performance details around caching.