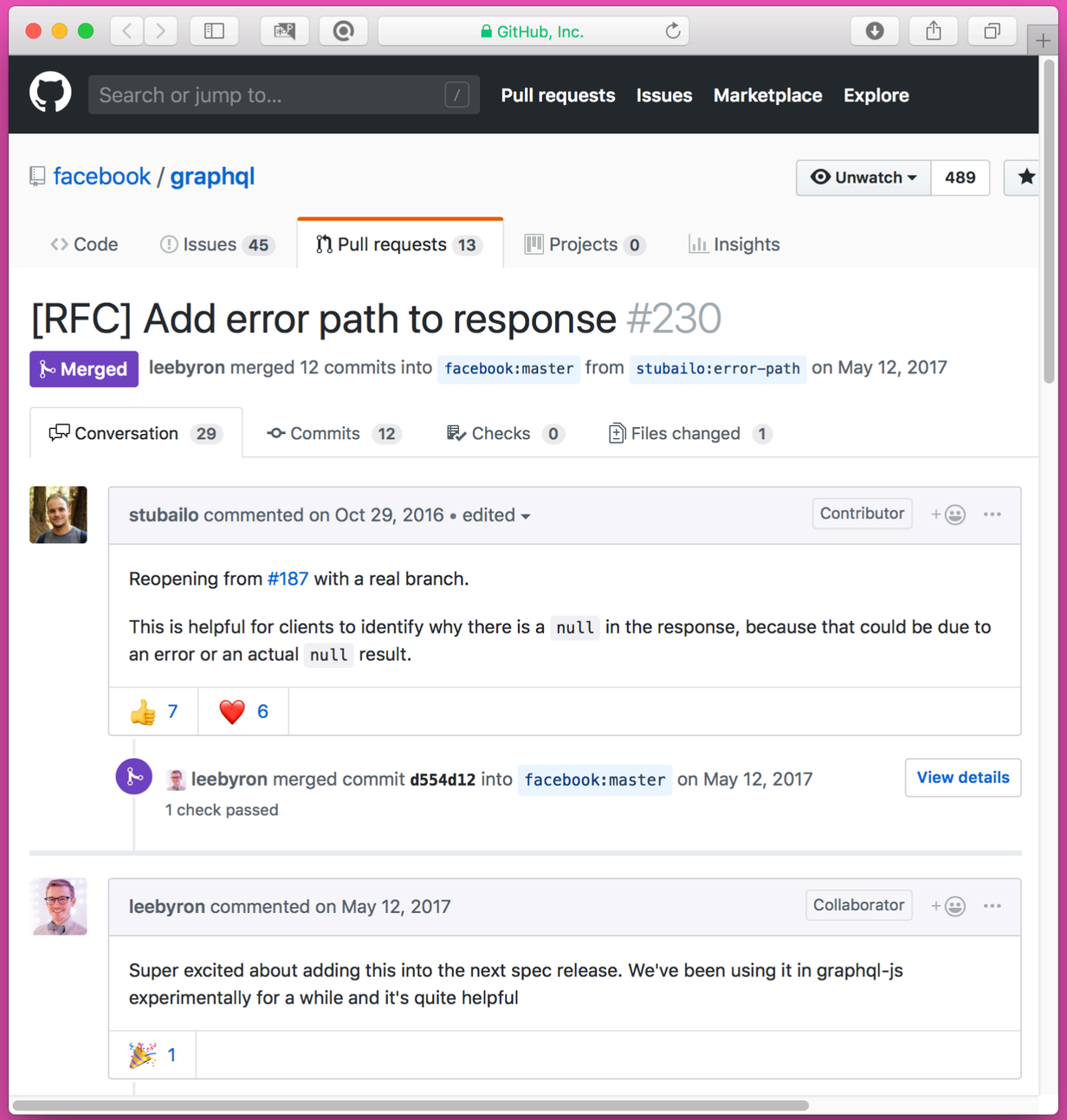
Where art thou, my error?

@alloy - artsy.net

https://github.com/alloy/graphql-finland

Still early days


GitHub, Inc.
Search or jump to...
Pull requests
Issues
Marketplace
Explore
facebook / graphql
Unwatch
489
Code
Issues 45
Pull requests 13
Projects 0
Insights
[RFC] Add error path to response #230
Merged
leebyron merged 12 commits into facebook:master from stubailo:error-path on May 12, 2017
Conversation 29
Commits 12
Checks 0
Files changed 1
stubailo commented on Oct 29, 2016 • edited
Contributor
Reopening from #187 with a real branch.
This is helpful for clients to identify why there is a null in the response, because that could be due to an error or an actual null result.
7
6
leebyron merged commit d554d12 into facebook:master on May 12, 2017
View details
1 check passed
leebyron commented on May 12, 2017
Collaborator
Super excited about adding this into the next spec release. We've been using it in graphql-js experimentally for a while and it's quite helpful
1

Errors *vs* Errors

GraphQL Errors

GraphQL Errors

```
{
  data: {
    artwork: {
      artist: {
        name: "Vincent van Gogh",
        leftEarSize: null
      }
    }
  },
  errors: [
    {
      message: "An unexpected error occurred",
      path: ["artwork", "artist", "leftEarSize"]
    }
  ]
}
```

Exceptions

What’s the problem?

Partial data

artsy.net

Search by artist, gallery, etc.

ARTWORKS AUCTIONS MORE 4

Banksy

British, 100,382 followers

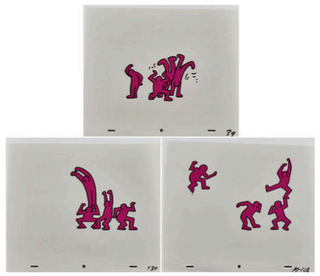
Following

Whether plastering cities with his trademark parachuting rat, painting imagined openings in the West Bank barrier in Israel, or stenciling "We're bored of fish" above a penguins' zoo enclosure, Banksy creates street art with an irreverent wit and an international reputation that precedes his anonymous identity. "TV … Read more

Recently viewed

Sold

Sold



Bidding closed

artsy.net

Search by artist, gallery, etc.

ARTWORKS AUCTIONS MORE 4

Banksy

British, 100,382 followers

Following

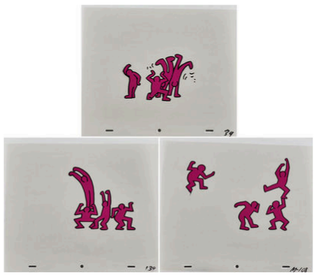
Whether plastering cities with his trademark parachuting rat, painting imagined openings in the West Bank barrier in Israel, or stenciling "We're bored of fish" above a penguins' zoo enclosure, Banksy creates street art with an irreverent wit and an international reputation that precedes his anonymous identity. "TV … Read more

Component unrelated to artist page

Recently viewed

Sold

Sold



Bidding closed

artsy.net

Search by artist, gallery, etc.

ARTWORKS AUCTIONS MORE 4

Ways to Buy

- [ ] Buy now
- [ ] Bid
- [ ] Inquire

Medium

- ( ) Prints
- ( ) Painting
- ( ) Sculpture
- ( ) Work on Paper
- ( ) Design
- ( ) Installation
- ( ) Drawing

Gallery

Institution

Time period

Sort: **Default**

$89,500
Banksy
CHOOSE YOUR WEAPON (SKY BL…
Gallery Art

$55,000
Banksy
Stop & Search, 2007
David Benrimon Fine Art

Jack & Jill (Police Kids), 2005
David Benrimon Fine Art

artsy.net

Search by artist, gallery, etc.

ARTWORKS AUCTIONS MORE 4

Ways to Buy

- [ ] Buy now
- [ ] Bid
- [ ] Inquire

Medium

- ( ) Prints
- ( ) Painting
- ( ) Sculpture
- ( ) Work on Paper
- ( ) Design
- ( ) Installation
- ( ) Drawing

Gallery

Institution

Time period

Sort: **Default**

$89,500
Banksy
CHOOSE YOUR WEAPON (SKY BL…
Gallery Art

$55,000
Banksy
Stop & Search, 2007
David Benrimon Fine Art

Jack & Jill (Police Kids), 2005
David Benrimon Fine Art

Can you spot the missing artwork brick? This is fine 👍

Communicate error

staging.artsy.net

ARTSY

Shipping › Payment › Review

- (●) Add shipping address
- ( ) Arrange for pickup (free)

Full name

Eloy Durán

Country

United States

Postal code

Add postal code

This field is required

Address line 1

401 Broadway

Address line 2 (optional)

25th floor

City

New York

State, province, or region

NY

Phone number

Required for shipping logistics

0123456789

Continue

Dabeiyuzhou | 1st Geno…
SundayTestArtworks No P…
Invoicing Demo Partner
New York, NY, US

Price	$400,000.00
Shipping	—
Tax	—
Total	**$400,000.00**

Have a question? Read our FAQ or ask a specialist.

Possible solutions

GraphQL Errors and reject entire response

https://u.nu/atrh

https://u.nu/pjv-

GraphQL Errors with metadata

https://u.nu/r5-2

GraphQL Errors with metadata

```
import { UserInputError } from 'apollo-server';

const resolvers = {
  Query: {
    events(root, { zipCode }) {
      if (!isValidZipCode(zipCode)) {
        throw new UserInputError(
          'Failed to get events due to validation errors',
          { validationErrors: {
            zipCode: 'This is not a valid zipcode'
          }}
        );
      }
      return getEventsByZipcode(zipCode);
    }
  }
}
```

GraphQL Errors with metadata

While convenient, the weakness of this approach is that the format of the validation error messages is not captured by your schema, making it brittle to changes. Unless you maintain tight control of both server and client, you should keep the error responses as simple as possible.
For mutations, it can be worthwhile defining these validation errors as first class citizens within your schema.

Extra (mutation) error fields

https://u.nu/th7y

Extra (mutation) error fields

```
type UpdateArtworkMutationResponse {
  success: Boolean!
  message: String!
  artwork: Artwork
}
```

Extra (mutation) error fields

https://u.nu/th7y

Extra error field & type

https://u.nu/-3pw

Extra error field & type

```
type GenericError {
  message: String!
}

type UpdateArtworkMutationResponse {
  error: GenericError
  artwork: Artwork
}
```

Extra error field & type

```
type PublishedArtworkNotification {
  artwork: Artwork
}

type PublishedArtworkNotificationsPayload {
  error: GenericError
  notifications: [PublishedArtworkNotification]
}

type Query {
  publishedArtworkNotificationsPayload:
    PublishedArtworkNotificationsPayload!
}
```

Extra error field & type

https://u.nu/-3pw

Side-note: https://u.nu/i0yg

Recap

- Use GraphQL
- In context
- All operations
- Explicit status

Exceptions as first-class citizens

Exceptions as first-class citizens

```
type Artwork {
  title: String!
}

type HTTPError {
  message: String!
  statusCode: Int!
}

union ArtworkOrError = Artwork | HTTPError

type Query {
  artworkOrError(id: ID!): ArtworkOrError
}

query {
  artworkOrError("mona-lisa") {
    ... on Artwork {
      title
    }
    ... on HTTPError {
      statusCode
    }
  }
}
```

Exceptions as first-class citizens

```
type Artist {
  artworksOrErrors: [ArtworkOrError]
}

type Query {
  artist(id: ID!): Artist
}

query {
  artist("leonardo-da-vinci") {
    artworksOrErrors {
      ... on Artwork {
        title
      }
      ... on HTTPError {
        statusCode
      }
    }
  }
}
```

Exceptions as first-class citizens

```
type UpdateArtworkMutationResponse {
  artworkOrError: ArtworkOrError
}
```

Exceptions as first-class citizens

```
query {
  artworkOrError("mona-lisa") {
    ... on Artwork {
      title
    }
  }
}
```

How we use it

Types

```
interface Error {
  message: String!
}

interface HTTPError {
  message: String!
  statusCode: Int!
}

type HTTPErrorType implements Error & HTTPError {
  message: String!
  statusCode: Int!
}

type Artwork {
  title: String!
}

union ArtworkOrError = Artwork | HTTPErrorType

type Query {
  artworkOrError(id: ID!): ArtworkOrError
}
```

Types

```
interface Error {
  message: String!
}

interface HTTPError {
  message: String!
  statusCode: Int!
}

type HTTPErrorType implements Error & HTTPError {
  message: String!
  statusCode: Int!
}

type Artwork {
  title: String!
}

union ArtworkOrError = Artwork | HTTPErrorType

type Query {
  artworkOrError(id: ID!): ArtworkOrError
}
```

Types

```
query {
  artworkOrError("mona-lisa") {
    ... on Artwork {
      title
    }
    ... on HTTPError {
      message
      statusCode
    }
  }
}
```

Types

```
query {
  artworkOrError("mona-lisa") {
    ... on Artwork {
      title
    }
    ...GenericErrorComponent
    ...GenericHTTPErrorComponent
  }
}

fragment GenericErrorComponent on Error {
  message
}

fragment GenericErrorComponent on HTTPError {
  message
  statusCode
}
```

Types

```
interface Error {
  message: String!
}

interface HTTPError {
  message: String!
  statusCode: Int!
}

type HTTPErrorType implements Error & HTTPError {
  message: String!
  statusCode: Int!
}

type Artwork {
  title: String!
}

union ArtworkOrError = Artwork | HTTPErrorType

type Query {
  artworkOrError(id: ID!): ArtworkOrError
}
```

Types

Side-note: https://u.nu/c1ye

Field naming

‘something’ *or* error

Field naming

```
query {
  artwork("mona-lisa") {
    title
  }
}
```

Field naming

```
query {
  artworkOrError("mona-lisa") {
    ... on Artwork {
      title
    }
    ... on HTTPError {
      statusCode
    }
  }
}
```

Field naming

```
type Query {
  artworks: [artwork]
  artworksConnection: ArtworksConnection
}
```

Downside of union

```
type ArtworkPurchasableBox {
  value: Boolean!
}

union ArtworkPurchasableOrError = ArtworkPurchasableBox | HTTPError

type Artwork {
  currentlyPurchasableOrError: ArtworkPurchasableOrError
}
```

Downside of union

Side-note: https://u.nu/c29p

Example of query usage

```
import { OrderStatus_order } from "__generated__/OrderStatus_order.graphql"
import { createFragmentContainer, graphql } from "react-relay"

interface Props { order: OrderStatus_order }

const OrderStatus: React.SFC<Props> = ({ order: orderStatusOrError }) =>
  orderStatusOrError.__typename === "OrderStatus" ? (
    <div>
      {orderStatusOrError.deliveryDispatched
        ? "Your order has been dispatched."
        : "Your order has not been dispatched yet."}
    </div>
  ) : (
    <div className="error">
      {orderStatusOrError.code === "unpublished"
        ? "Please contact gallery services."
        : `An unexpected error occurred: ${orderStatusOrError.message}`}
    </div>
  )

export const OrderStatusContainer = createFragmentContainer(
  OrderStatus,
  graphql`
    fragment OrderStatus_order on Order {
      orderStatusOrError {
        __typename
        ... on OrderStatus {
          deliveryDispatched
        }
        ... on OrderError {
          message
          code
        }
      }
    }
  `
)
```

Example of mutation usage

```
import { SubmitOrder_order } from "__generated__/SubmitOrder_order.graphql"
import { SubmitOrderMutation } from "__generated__/SubmitOrderMutation.graphql"
import { Router } from "found-relay"
import { commitMutation, createFragmentContainer, graphql, RelayProp } from "react-relay"

interface Props { order: SubmitOrder_order, relay: RelayProp, router: Router }

const SubmitOrder: React.SFC<Props> = props => (
  <button
    onClick={() => {
      commitMutation<SubmitOrderMutation>(props.relay.environment, {
        mutation: graphql`
          mutation SubmitOrderMutation($input: SubmitOrder!) {
            submitOrder(input: $input) {
              orderStatusOrError {
                __typename
                ... on OrderStatus {
                  submitted
                }
                ... on OrderError {
                  message
                  code
                }
              }
            }
          }
        `,
        variables: { input: { orderID: props.order.id } },
        onCompleted: ({ submitOrder: { orderStatusOrError } }, errors) => {
          if (orderStatusOrError.__typename === "OrderStatus") {
            props.router.push(
              `/orders/${props.order.id}/${
                orderStatusOrError.submitted ? "submitted" : "pending"
              }`
            )
          } else {
            alert(
              orderStatusOrError.code === "unpublished"
                ? "Please contact gallery services."
                : `An unexpected error occurred: ${orderStatusOrError.message}`
            )
          }
        },
      })
    }}
  />
)

export const SubmitOrderContainer = createFragmentContainer(…)
```

Final thoughts

Final thoughts

 @alloy

Final thoughts

Final thoughts

...to ‘REST’ 😉

twitter.com

Home 99+ Notifications 1 Messages Search Twitter Tweet

Eloy “DurÃ¡n” Durán @alloy · Jul 17

Anyone know of any good posts on modelling user-errors as part of your GraphQL schema? We've started doing it using a union (github.com/artsy/metaphys …) just looking for some more perspectives from people with experience. cc @leeb

Adds union type for returning success and failure …

Marking this a WIP since I want to get some feedback before I write tests. This represents yet another version of error handling... informed heavily by this discussio...

github.com

2 2 8

Lee Byron
@leeb

Following

Replying to @alloy

That diff makes a lot of sense to me. I've also seen user errors as a field on the mutation result, but I like that union makes it explicit that there was either success or failure and in the case of failure provides rich information that's in your app's domain.

11:16 PM - 19 Jul 2018

2 Likes

1 2

Lee Byron
@leeb

I make things @R
previously @Face
Immutable.js, Wel
Nonsense.

San Francisco
leebyron.com
Joined May 20