The Typescript Sandwich 

Gradius has 3 layers

BUSTLE DIGITAL GROUP

Gradius has 3 layers

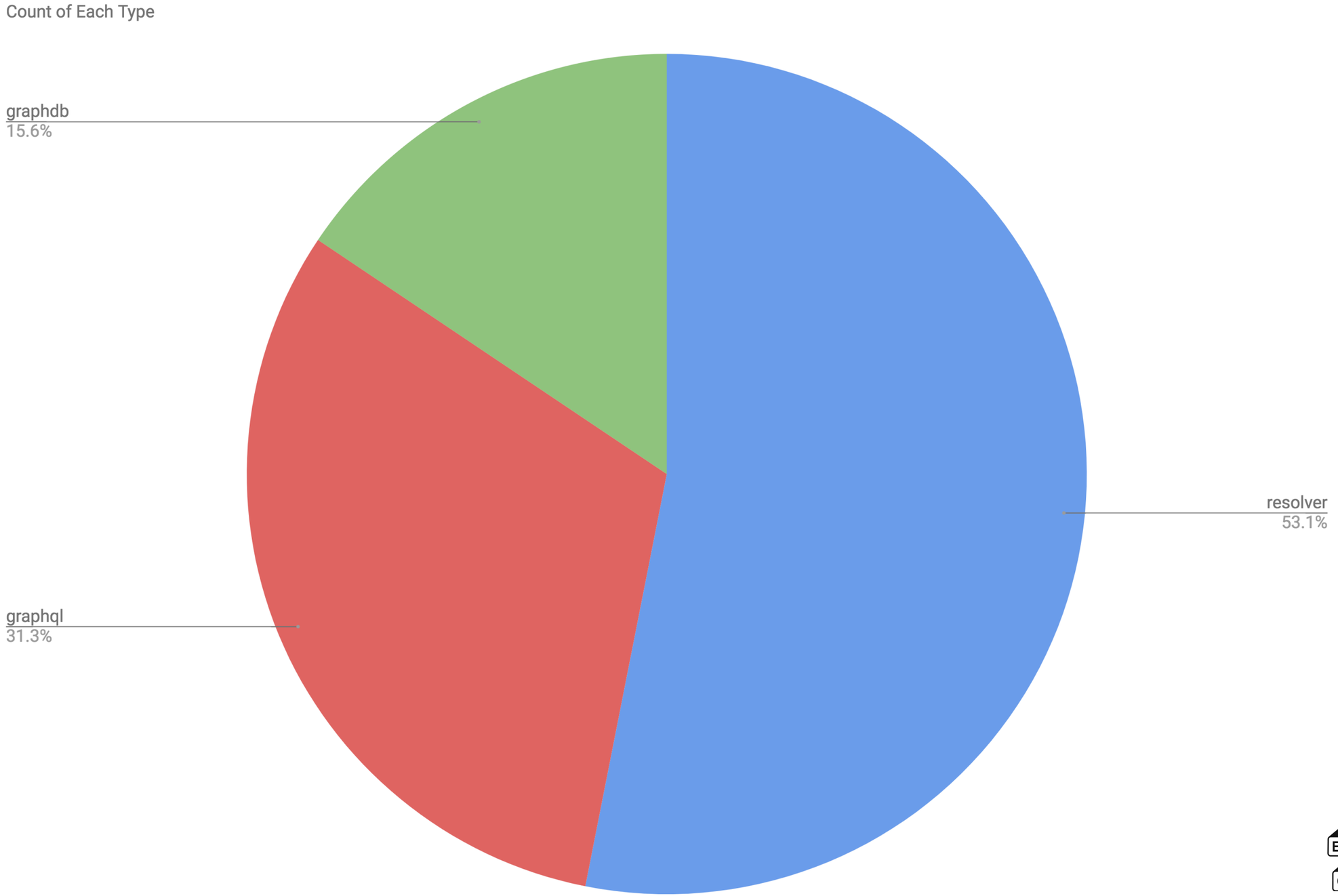
- 🍞 GraphQL Objects
- 🥩 Resolvers
- 🍞 GraphDB Objects

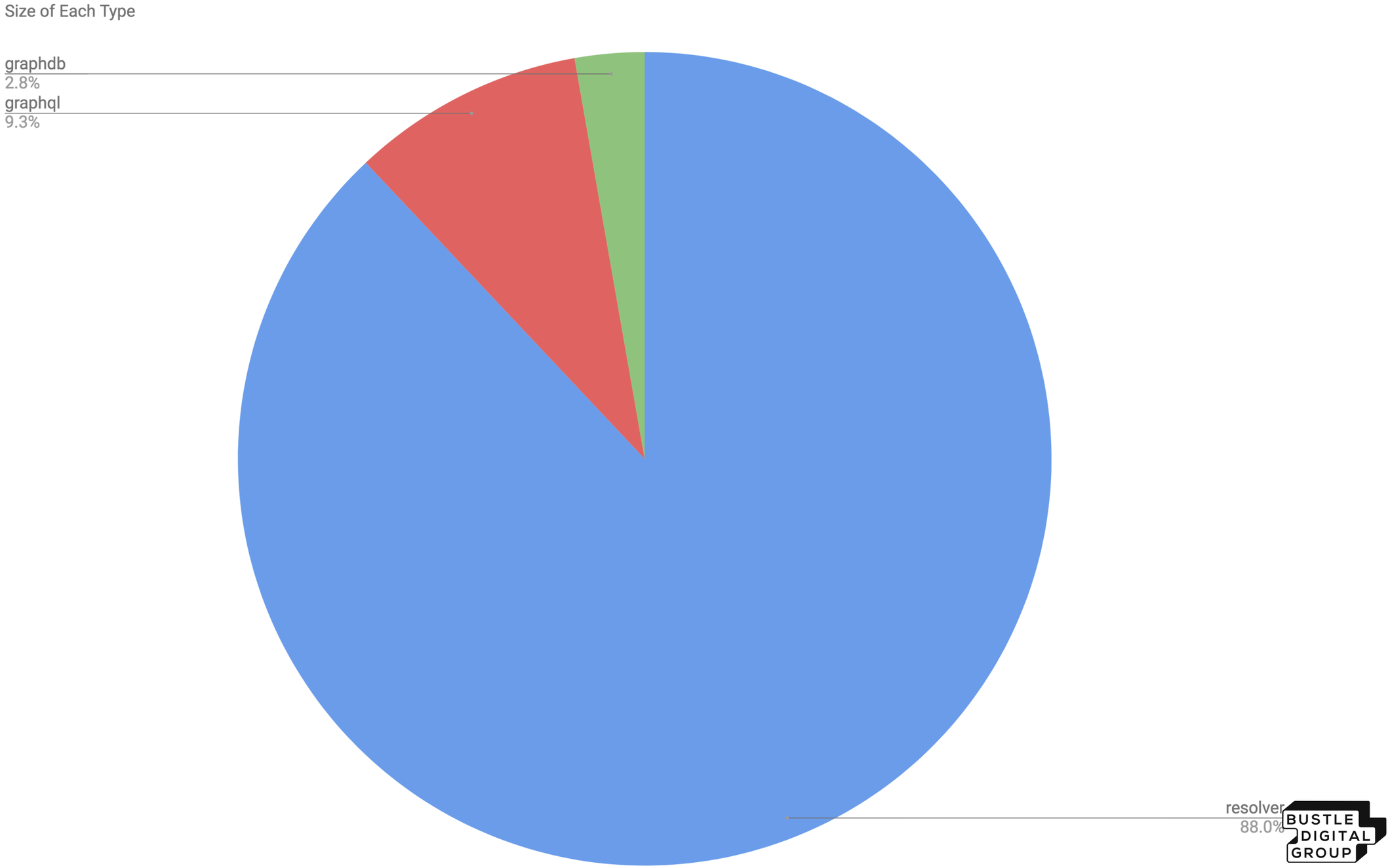
BUSTLE DIGITAL GROUP

Gradius is more like a Monte Cristo

- 🍞 GraphQL Input Objects
- 🥩 Resolvers
- 🥓 GraphDB Objects
- 🥩 Resolvers
- 🍞 GraphQL Output Objects

BUSTLE DIGITAL GROUP


Count of Each Type
graphdb
15.6%
resolver
53.1%
graphql
31.3%


Size of Each Type
graphdb
2.8%
graphql
9.3%
resolver
88.0%
BUSTLE DIGITAL GROUP

Less  in our  the better

BUSTLE DIGITAL GROUP

Lets make Resolvers Typescript?

BUSTLE DIGITAL GROUP

```
export async function updateCarouselZone(
  input,
  { skipAsyncEvents } = {}
) {
  const { id } = input
  const zone = await graph.CarouselZone.FIND(id, BadRequestError)
  const updatedZone = await graph.CarouselZone.update(input)
  if (!skipAsyncEvents) {
    await indexNodeAsync(zone)
  }
  return { zones: updatedZone }
}
```

BUSTLE DIGITAL GROUP

```
export async function updateCarouselZone(
  input: any,
  { skipAsyncEvents } = { skipAsyncEvents?: bool }
) {
  const { id }: { id: string|number } = input
  const zone: any = await graph.CarouselZone.FIND(id, BadRequestError)
  const updatedZone: any = await graph.CarouselZone.update(input)
  if (!skipAsyncEvents) {
    await indexNodeAsync(zone)
  }
  return { zones: updatedZone }
}
```

BUSTLE DIGITAL GROUP

Not helpful

BUSTLE DIGITAL GROUP

Type the Data in 

Type the Data out 

Chase the 

BUSTLE DIGITAL GROUP

Data In 

BUSTLE DIGITAL GROUP

🐦 GraphQLInput Objects

```
input UpdateCarouselZoneInput {
  id: ID!
  title: String
  description: String
  display: ZoneDisplayType
  clientMutationId: String
}
```

BUSTLE DIGITAL GROUP

 GraphQLInput Objects

```
interface UpdateCarouselZoneInput {
  id: GUID
  title?: string
  description?: string
  display?: 'FULL_BLEED' | 'INLINE'
  clientMutationId?: string
}
```

BUSTLE DIGITAL GROUP

 GraphQLOutput Objects

```
type CarouselZone implements Node & Zone {
  /* mix of code and data */
}

type UpdateCarouselZonePayload {
  zone: CarouselZone!
  clientMutationId: String
}
```

BUSTLE DIGITAL GROUP

🐦 GraphQLOutput Objects

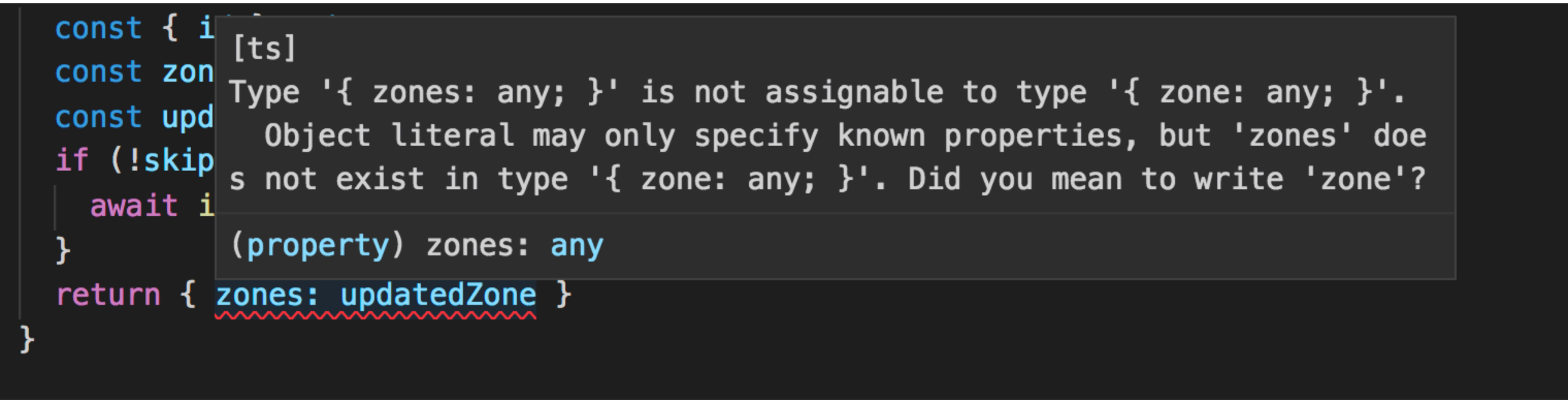
```
interface CarouselZone {
  id: GUID
  [key: string]: any // to type another day
}

interface UpdateCarouselZonePayload {
  zone: CarouselZone
  clientMutationId?: string
}
```

BUSTLE DIGITAL GROUP

```
export async function updateCarouselZone(
  input: UpdateCarouselZoneInput,
  { skipAsyncEvents } = { skipAsyncEvents?: bool }
): Promise<UpdateCarouselZonePayload> {
  const { id } = input // knows id is guaranteed and a GUID
  const zone: any = await graph.CarouselZone.FIND(id, BadRequestError)
  const updatedZone: any = await graph.CarouselZone.update(input)
  if (!skipAsyncEvents) {
    await indexNodeAsync(zone)
  }
  // ERROR! UpdateCarouselZonePayload requires a `zone` field not a `zones` field
  return { zones: updatedZone }
}
```

BUSTLE DIGITAL GROUP



BUSTLE DIGITAL GROUP

BUSTLE DIGITAL GROUP

Next Challenges 

- Mixing of data and code in view types
- Functional approach to Resolvers
- No way to Type the data input for the GraphQL Node Objects

BUSTLE DIGITAL GROUP

Next Challenges 

- GraphDB needs types in JS and TS and that's annoying
- GraphDB needs more validations then types

BUSTLE DIGITAL GROUP

BUSTLE DIGITAL GROUP