Batching Requests

Many times we want to batch the requests a program(or component) does into one single request that fetches all the data items needed to execute(render) the program.

Lets say you have a component `<UserDetails userId={1}>` that internally fetches the user data and displays the user details. But it might be used arbitrarily on the page, for example in a user list, but also in the sidebar containing some messages and the details of the user that sent it.

The problem is that orchestrating a single fetch for all rendered user details components can be complex and has to be done in a parent component or state management module.

What we want

- We want to define a component that fetches and renders the user details by a user id.
- We want to batch the requests for user details made by all rendered components that are on the page at a given time.
- If one or more components are rendered within `10 ms` we batch those requests.

requests.

Implementation

First lets setup a batcher using @yornaath/batshit

ts

```ts
import { create, windowScheduler, keyResolver } from "@

type User = { id: number, name: string }

const users = create<User[], number>({
  // The fetcher resolves the list of queries(here just
  fetcher: async (ids: number[]) => {
    return api.users.where({
      userId_in: ids
    })
  },
  // when we call users.fetch, this will resolve the co
  resolver: keyResolver("id"),
  // this will batch all calls to users.fetch that are
  scheduler: windowScheduler(10)
})
```

Then we can define our query hook that uses the batcher. If this hook is used(rendered) in multiple places within our app within the 10 millisecond window we gave, all those requests will be batched. Caching and deduplication is of course handled by react-query.

ts

```ts
const useUser = (id: number) => {
  return useQuery([ "users", id ], async () => {
    return users.fetch(id)
  })
}
```

> *requests and can be used arbitrarily across your codebase."*

Finally lets define our user details component that consumes the hook we made.

```ts
const UserDetails = (props: {userId: number}) => {
  const {isLoading, data} = useUser(props.userId)
  return <>
    {
      isLoading ?
        <div>Loading user {props.userId}</div>
      :
        <div>
          User: {data.name}
        </div>
    }
  </>
}
```

batshit docs

For more information about how to create batchers with custom resolving and request scheduling please refer to the batshit repo readme over at the [yornaath/batshit github repository]