



EXPLORER

REACT-QUERY-DEMO

> node_modules

> public

src

components

JS Home.js

JS InfiniteQueries.js

JS PaginatedQueries...

JS PostDetailsRQ.js

JS PostsRQ.js

JS PostsTraditional.js

App.css

JS App.js

index.css

JS index.js

JS reportWebVitals.js

JS setupTests.js

> OUTLINE

> TIMELINE

JS InfiniteQueries.js X

src > components > JS InfiniteQueries.js > InfiniteQueries

```
    _page=${pageParam}`);
  }

  const InfiniteQueries = () => {
    const { data, isLoading, isError, error } = useInfiniteQuery({
      queryKey: ["fruits"],
      queryFn: fetchFruits,
      initialPageParam: 1,
      getNextPageParam: (_lastPage, allPages) => {
        if (allPages.length < 5) {
          return allPages.length + 1
        } else {
          return undefined
        }
      }
    })

    if (isLoading) {
      return <h2>Page is Loading...</h2>
    }
  }
}
```

1

EXPLORER

...

✓ REACT-QUERY-DEMO

> node_modules

> public

> src

> components

JS Home.js

JS InfiniteQueries.js

JS PaginatedQueries...

JS PostDetailsRQ.js

JS PostsRQ.js

JS PostsTraditional.js

App.css

JS App.js

index.css

JS index.js

JS reportWebVitals.js

JS setupTests.js

> OUTLINE

> TIMELINE

JS InfiniteQueries.js •

infiniteQueries.js > InfiniteQueries > data.pages.map() callback > page.data.map() callback

9 const InfiniteQueries = () => {

23

24 console.log(data, "data")

25

26 if (isLoading) {

27 | return <h2>Page is Loading...</h2>

28 }

29

30 if (isError) {

31 | return <h1>{error.message}</h1>

32 }

33

34 return (

35 | <div className='container'>

36 | | {data?.pages?.map(page => {

37 | | | return page?.data.map(fruit => {

38 | | | | return <div key={fruit.id}>

39 | | | | |

40 | | | | </div>

41 | | | }

42 | | }

43 | </div>



EXPLORER



▼ REACT-QUERY-DEMO

> node_modules

> public

▼ src

▼ components

JS Home.js

JS InfiniteQuerie... 1

JS PaginatedQueries...

JS PostDetailsRQ.js

JS PostsRQ.js

JS PostsTraditional.js

App.css

JS App.js

index.css

JS index.js

JS reportWebVitals.js

JS setupTests.js

> OUTLINE

> TIMELINE

JS InfiniteQueries.js 1 X



src > components > JS InfiniteQueries.js > InfiniteQueries > fetchNextPage

```
1  import { useInfiniteQuery } from '@tanstack/react-query'
2  import axios from 'axios'
3  import React from 'react'
4
5  const fetchFruits = ({ pageParam }) => {
6    return axios.get(`http://localhost:4000/fruits/?_limit=4&
   _page=${pageParam}`);
7  }
8
9  const InfiniteQueries = () => {
10
11    const [ data, isLoading, isError, error, fetchNextPage ] =
      useInfiniteQuery({
12      queryKey: ["fruits"],
13      queryFn: fetchFruits,
14      initialPageParam: 1,
15      getNextPageParam: (_lastPage, allPages) => {
16        if (allPages.length < 5) {
17          return allPages.length + 1
18        } else {
19          return undefined
20        }
21      }
22    })
23  }
```



EXPLORER

REACT-QUERY-DEMO

> node_modules

> public

> src

components

JS Home.js

JS InfiniteQueries.js

JS PaginatedQueries...

JS PostDetailsRQ.js

JS PostsRQ.js

JS PostsTraditional.js

App.css

JS App.js

index.css

JS index.js

JS reportWebVitals.js

JS setupTests.js

> OUTLINE

> TIMELINE

JS InfiniteQueries.js X

src > components > JS InfiniteQueries.js > InfiniteQueries

```
9  const InfiniteQueries = () => {
28
29
30    if (isError) {
31      return <h1>{error.message}</h1>
32    }
33
34    return (
35      <div className='container'>
36        {data?.pages?.map(page => {
37          return page?.data.map(fruit => {
38            return <div className='fruit-item' key={fruit.
39              id}>
40              {fruit.name}
41            </div>
42          })
43        })}
44        <button onClick={fetchNextPage}>Load More..</button>
45      </div>
46    )
47
48    export default InfiniteQueries
```

EXPLORER

REACT-QUERY-DEMO

- > node_modules
- > public
- ✓ src
 - ✓ components
 - JS Home.js
 - JS InfiniteQueries.js
 - JS PaginatedQueries...
 - JS PostDetailsRQ.js
 - JS PostsRQ.js
 - JS PostsTraditional.js
 - # App.css
 - JS App.js
 - # index.css
 - JS index.js
 - JS reportWebVitals.js
 - JS setupTests.js
- > OUTLINE
- > TIMELINE

src > components > JS InfiniteQueries.js > InfiniteQueries > hasNextPage

```
8
9  const InfiniteQueries = () => {
10
11    const { data, isLoading, isError, error, fetchNextPage,
12      hasNextPage } = useInfiniteQuery({
13      queryKey: ["fruits"],
14      queryFn: fetchFruits,
15      initialPageParam: 1,
16      getNextPageParam: (_lastPage, allPages) => {
17        if (allPages.length < 5) {
18          return allPages.length + 1
19        } else {
20          return undefined
21        }
22      })
23
24    console.log(data, "data")
25
26    if (isLoading) {
27      return <h2>Page is Loading...</h2>
28    }
29  }
```

Ln 11, Col 72 (11 selected) Spaces: 4 UTF-8 LF {} JavaScript Go Live

