Query Functions

All query functions are accessed through the `screen` object in a test. These query functions *always* begin with one of the following names: `getBy` , `getAllBy` , `queryBy` , `queryAllBy` , `findBy` , `findAllBy` .

Start of Function Name	Examples
getBy	getByRole, getByText
getAllBy	getAllByText, getByDisplayValue
queryBy	queryByDisplayValue, queryByTitle
queryAllBy	queryAllByTitle, queryAllByText
findBy	findByRole, findBytext
findAllBy	findAllByText, findAllByDisplayValue

These names indicate the following:

1. Whether the function will return an element or an array of elements
2. What happens if the function finds 0, 1, or > 1 of the targeted element
3. Whether the function runs instantly (synchronously) or looks for an element over a span of time (asynchronously)

Looking for a Single Element?

Name	0 matches	1 match	> 1 match	Notes
getBy	Throw	Element	Throw	
queryBy	null	Element	Throw	
findBy	Throw	Element	Throw	Looks for an element over the span of 1 second

Looking for Multiple Elements?

Name	0 matches	1 match	> 1 match	Notes
getAllBy	Throw	[]Element	[]Element	
queryAllBy	[ ]	[]Element	[]Element	
findAllBy	Throw	[]Element	[]Element	Looks for elements over the span of 1 second

When to use each

Goal of test	Use
Prove an element exists	getBy, getAllBy
Prove an element does **not** exist	queryBy, queryAllBy
Make sure an element eventually exists	findBy, findAllBy

```jsx
import { render, screen } from '@testing-library/react';

function ColorList() {
  return (
    <ul>
      <li>Red</li>
      <li>Blue</li>
      <li>Green</li>
    </ul>
  );
}

render(<ColorList />);
```

+ Code + Text

```jsx
test('getBy, queryBy, findBy finding 0 elements', async () => {
  render(<ColorList />);

  expect(
    () => screen.getByRole('textbox')
  ).toThrow();

  expect(screen.queryByRole('textbox')).toEqual(null);

  let errorThrown = false;
  try {
    await screen.findByRole('textbox');
  } catch (err) {
    errorThrown = true;
  }
  expect(errorThrown).toEqual(true);
});

```

+ Code + Text

```jsx
test('getBy, queryBy, findBy when they find 1 element', async () => {
  render(<ColorList />);

  expect(
    screen.getByRole('list')
  ).toBeInTheDocument();
  expect(
    screen.queryByRole('list')
  ).toBeInTheDocument()
  expect(
    await screen.findByRole('list')
  ).toBeInTheDocument()
});
```

+ Code + Text

```
test('getBy, queryBy, findBy when finding > 1 elements', async () => {
  render(<ColorList />);

  expect(
    () => screen.getByRole('listitem')
  ).toThrow();

  expect(
    () => screen.queryByRole('listitem')
  ).toThrow();

  let errorThrown = false;
  try {
    await screen.findByRole('listitem');
  } catch (err) {
    errorThrown = true;
  }
  expect(errorThrown).toEqual(true);
});
```

+ Code + Text

```
test('getAllBy, queryAllBy, findAllBy', async () => {
  render(<ColorList />);

  expect(
    screen.getAllByRole('listitem')
  ).toHaveLength(3);

  expect(
    screen.queryAllByRole('listitem')
  ).toHaveLength(3);

  expect(
    await screen.findAllByRole('listitem')
  ).toHaveLength(3);
});
```

+ Code + Text

```
test('favor using getBy to prove an element exists', () => {
  render(<ColorList />);

  const element = screen.getByRole('list');

  expect(element).toBeInTheDocument();
});
```

+ Code + Text

```
test('favor queryBy when proving an element does not exist', () => {
  render(<ColorList />);

  const element = screen.queryByRole('textbox');

  expect(element).not.toBeInTheDocument();
});
```

+ Code + Text

```
import { useState, useEffect } from 'react';

function fakeFetchColors() {
  return Promise.resolve(
    ['red', 'green', 'blue']
  );
}

function LoadableColorList() {
  const [colors, setColors] = useState([]);

  useEffect(() => {
    fakeFetchColors()
      .then(c => setColors(c));
  }, []);

  const renderedColors = colors.map(color => {
    return <li key={color}>{color}</li>
  });

  return <ul>{renderedColors}</ul>
}

render(<LoadableColorList />);
```

+ Code + Text

```
test('Favor findBy or findAllBy when data fetching', async () => {
  render(<LoadableColorList />);

  const els = await screen.findAllByRole('listitem');

  expect(els).toHaveLength(3);
});
```

+ Code + Text