Testing Component Rendering

Liam McLennan

PRODUCT DELIVERY CONSULTANT

@liammclennan https://withouttheloop.com

Overview

Testing with React Testing Library

Testing components directly

Testing with Test Utilities

Testing with Test Renderer

Testing with React Testing Library

The Render Function

Renders a React component

Returns container and query functions

```
const r = render(<Message content="Some stuff" isImportant={true} />);

r.getByLabelText(‘First Name’);

r.getByText(‘2017-5-13’);
```

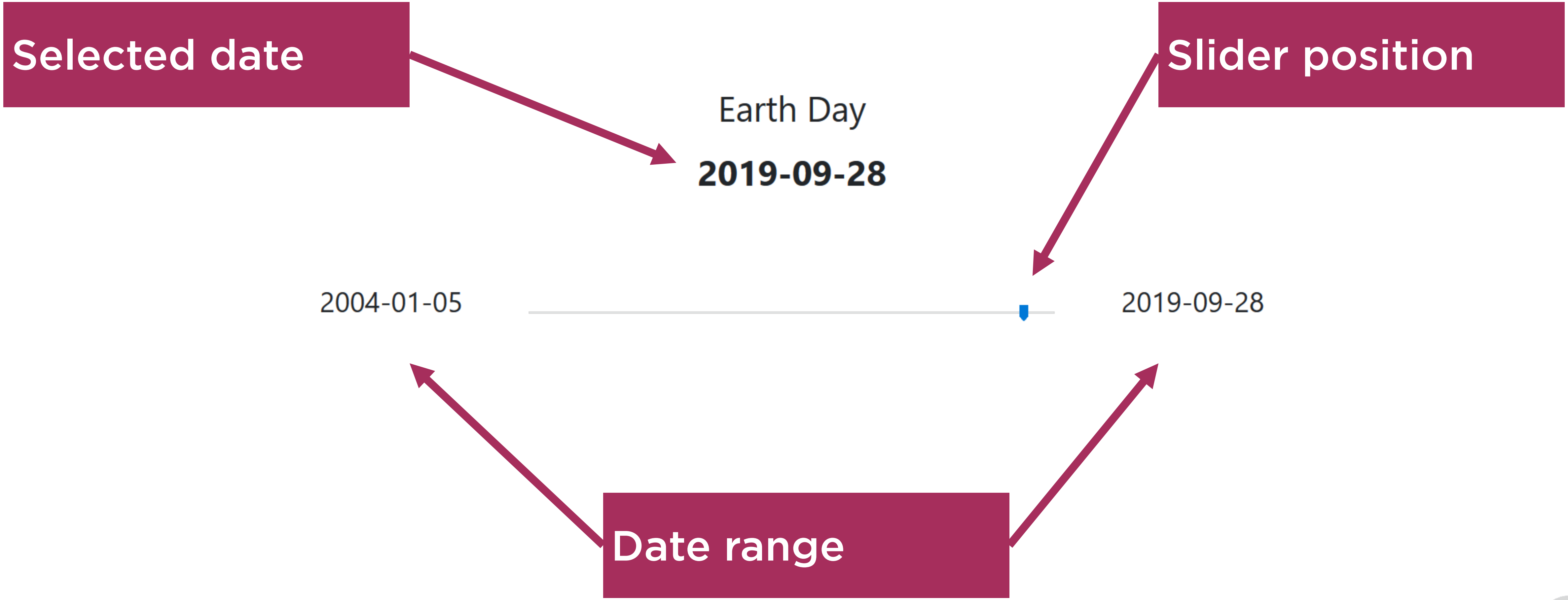
Query Functions

Principle:
Test exactly what you are trying to test and nothing more

```
const r = render(<Message content="Some stuff" isImportant={true} />);

r.getByLabelText(‘First Name’);

r.getByText(‘2017-5-13’);

r.getByText(/\d\d\d\d-\d-\d\d/);

r.getByTitle(‘introduction’);

r.getByTestId(‘target’);
```

Query Functions

Testing the DateSlider Component



Selected date

Slider position

Earth Day

2019-09-28

2004-01-05

2019-09-28

Date range

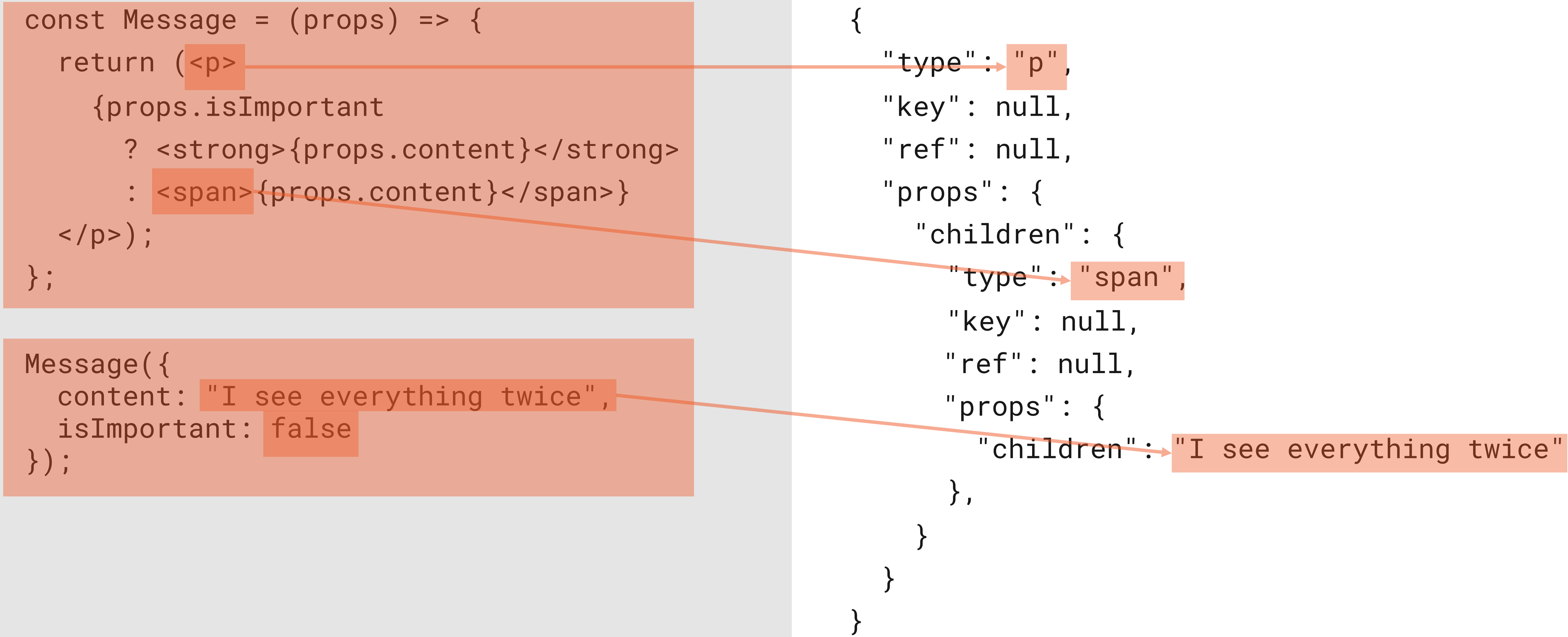
Demo

Testing the DateSlider component

Testing Components Directly

Demo

The Message Component



```
const Message = (props) => {
  return (<p>
    {props.isImportant
      ? <strong>{props.content}</strong>
      : <span>{props.content}</span>}
  </p>);
};
```

```
Message({
  content: "I see everything twice",
  isImportant: false
});
```

```
{
  "type": "p",
  "key": null,
  "ref": null,
  "props": {
    "children": {
      "type": "span",
      "key": null,
      "ref": null,
      "props": {
        "children": "I see everything twice"
      },
    }
  }
}
```

```
const Message = (props) => {
  return (<p>
    {props.isImportant
      ? <strong>{props.content}</strong>
      : <span>{props.content}</span>}
  </p>);
};


Message({
  content: "I see everything twice",
  isImportant: true
});
```

```
{
  "type": "p",
  "key": null,
  "ref": null,
  "props": {
    "children": {
      "type": "strong",
      "key": null,
      "ref": null,
      "props": {
        "children": "I see everything twice"
      },
    }
  }
}
```

Demo

Test the Message component rendering

Testing with Test Utilities

```
act(() => {
    ReactDOM.render(<Message content="Some stuff" isImportant={true} />,
container);
});

const button = container.querySelector(‘p’);

act(() => {
    button.dispatchEvent(new MouseEvent('click', {bubbles: true}));
});
```

act()

Provides a more realistic environment for rendering and updating components.

Demo

Testing the CameraSelection component

Testing with Test Renderer

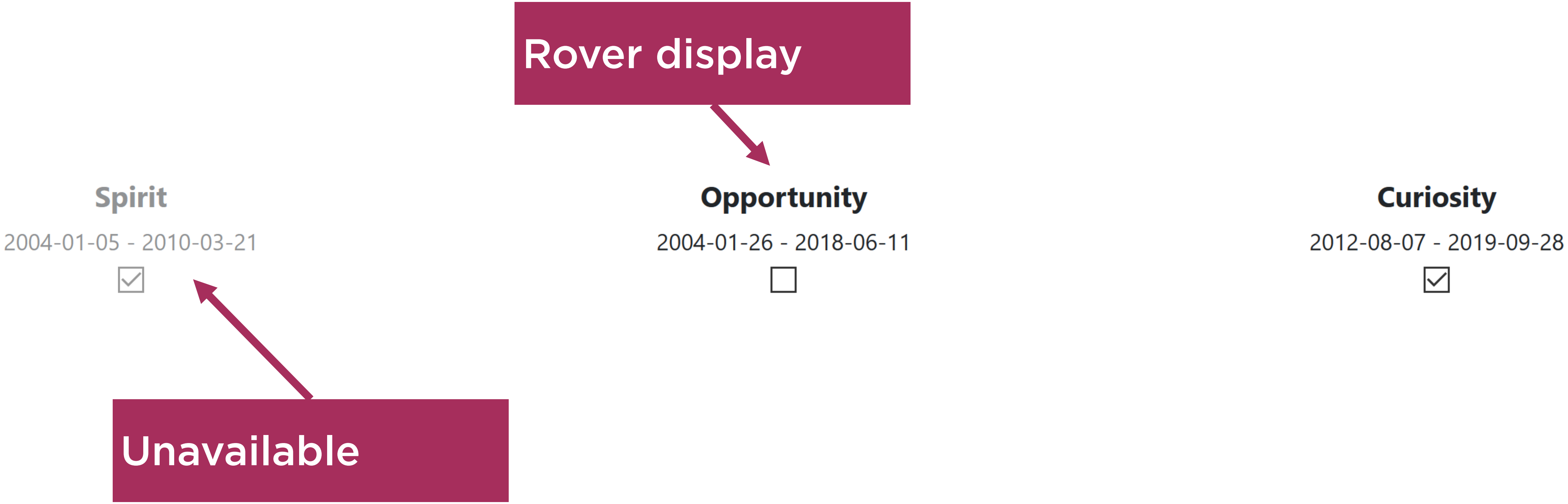
Testing with Test Renderer

- Render to objects
- Test rendering logic
- More realistic environment
- Traversal API

```
const root = TestRenderer.create(
  <Message
    content="Some stuff"
    isImportant={true} />).root;

root.findAllByType(“div”);

root.findByProps({“data-testid”: “rover-curiosity”});

root.findAll((i) => i.children.length > 0);
```

API

Testing the RoverSelector Component



Rover display

Spirit

2004-01-05 - 2010-03-21

☑

Opportunity

2004-01-26 - 2018-06-11

□

Curiosity

2012-08-07 - 2019-09-28

☑

Unavailable

Demo

Testing the RoverSelector component

Principle:
Never test through a UI component what can be tested some other way

Principle:
Never test the same thing twice

Summary

Choose low friction tools and sound test design

Test application logic directly

Principle:
Never test that your dependencies work