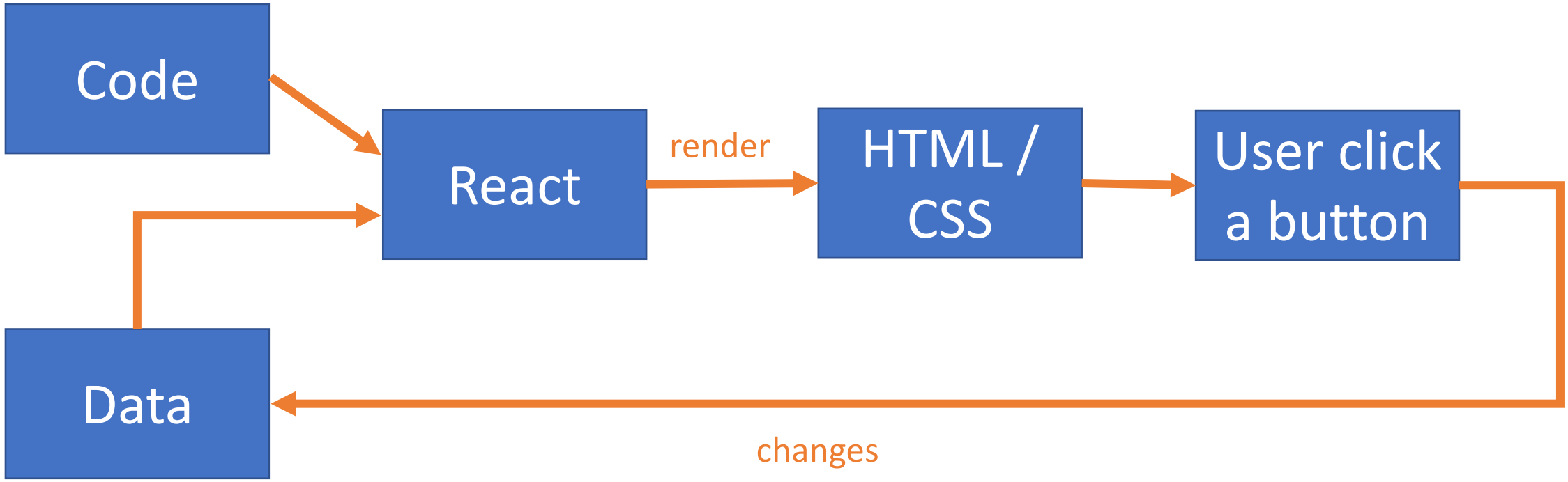
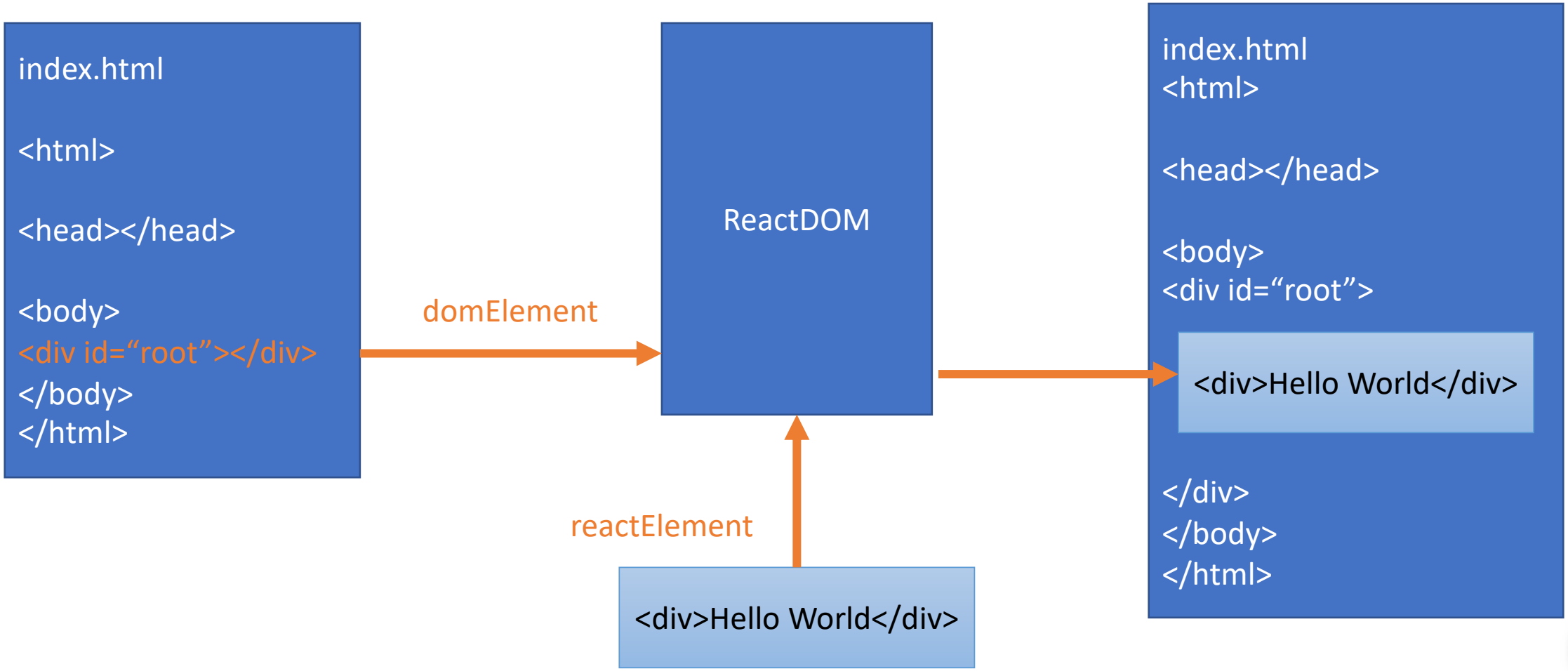
React JS Supplements

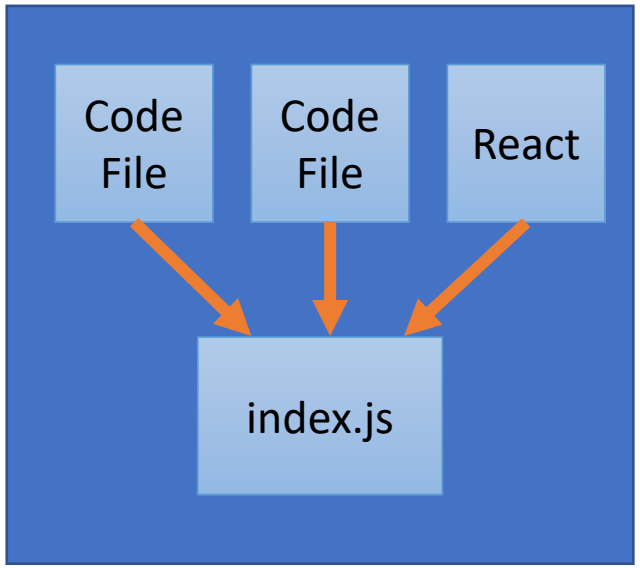
How React work?



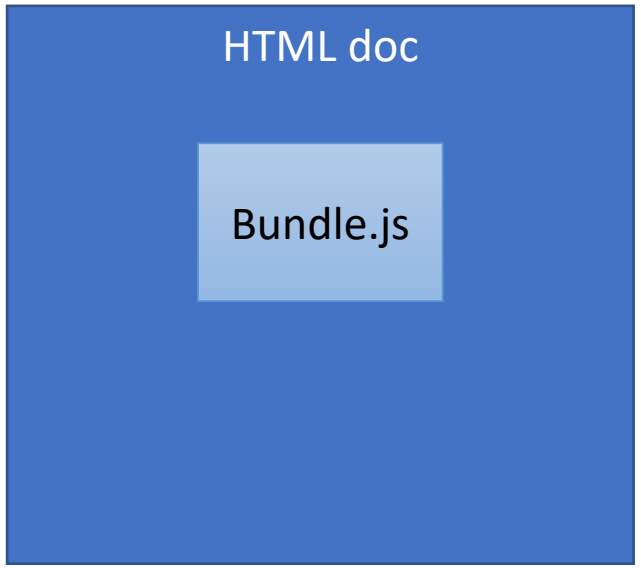
How Create React App start works?



How Create React App build works?



Create-React-App
- Turn ES2015/16/17 code to ES5
- Turn JSX to JS



input

output

React JS Supplements

- Design Single Page Application
- Fundamentals of Testing
- Test React Components with react-testing-library
- Redux
- Axios
- Synchronize data between Frontend and Backend
- Design Routing Table
- Convert a React Form to Formik

How to design React app from UI/UX?

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

ui

0 0

Leave the code cleaner than you found it.

From Clean Code: always leave the code cleaner than it was before.

craftsmanship clean code

0 0

Never say : "I've done, it works on my machine !" #itworksonmymachine

0 0

Always use === in JavaScript!

javascript

0 0

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

ui

0 0

Leave the code cleaner than you found it.

From Clean Code: always leave the code cleaner than it was before.

craftsmanship clean code

0 0

Never say : "I've done, it works on my machine !" #itworksonmymachine

0 0

Always use === in JavaScript!

javascript

0 0

RuleList View

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

Rule View

ui

0 0

Leave the code cleaner than you found it.

From Clean Code: always leave the code cleaner than it was before.

Rule View

craftsmanship clean code

0 0

Never say : "I've done, it works on my machine !" #itworksonmymachine

Rule View

0 0

Always use === in JavaScript!

Rule View

javascript

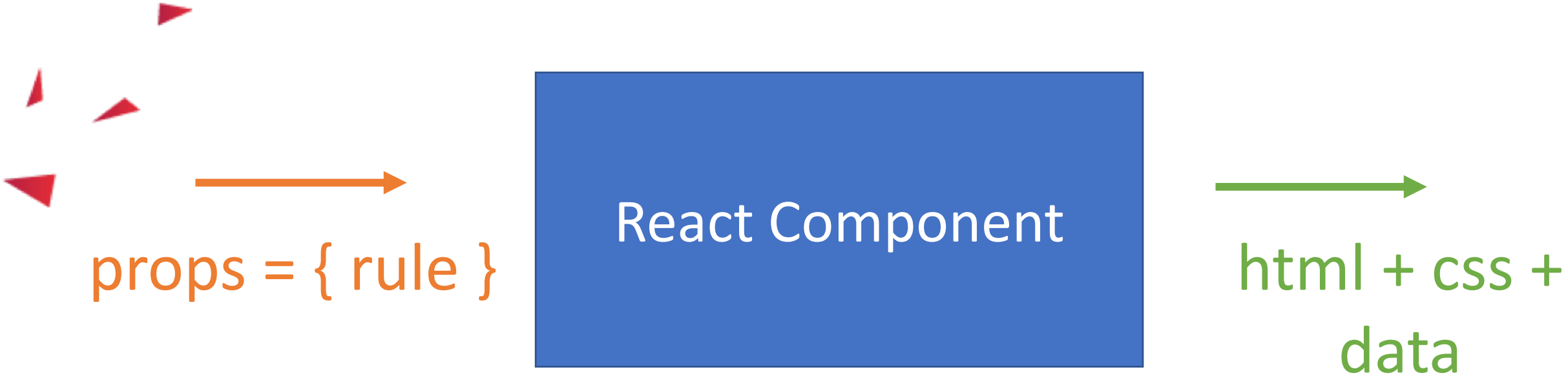
0 0

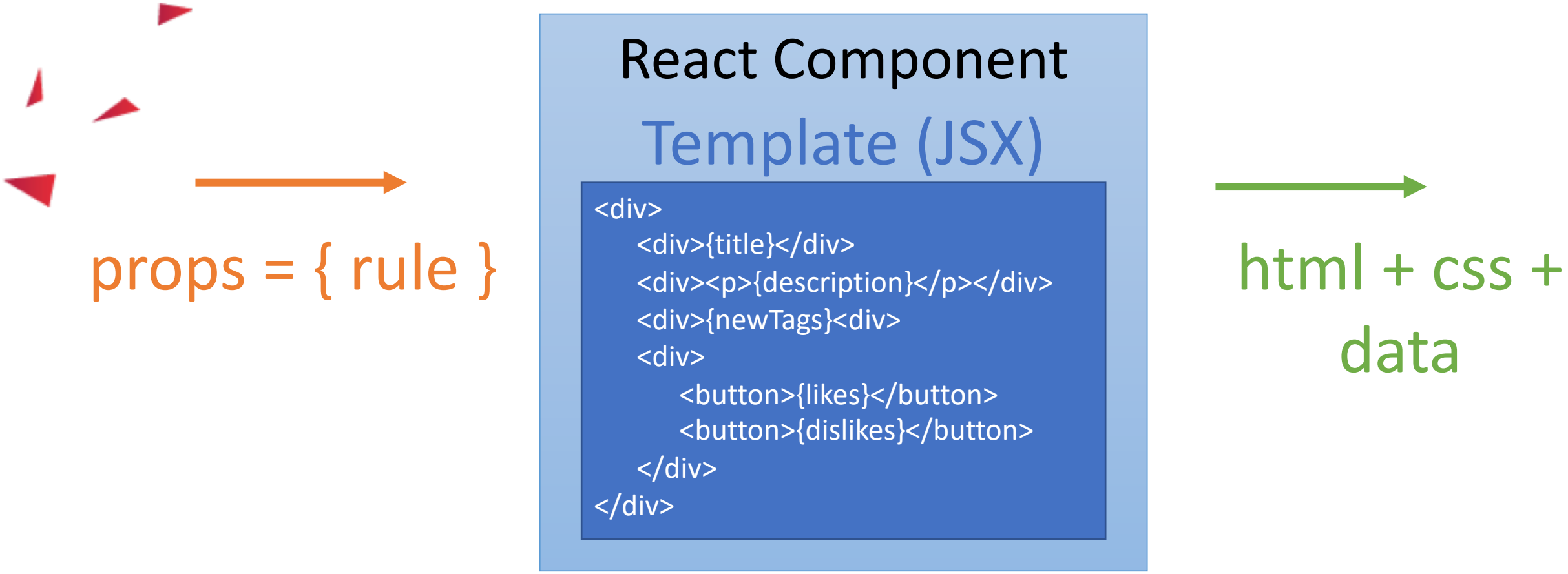

If you don't have a mobile website, you don't have a website.
title
In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.
description
ui
tags
0
0
likes
dislikes

likes

dislikes

	Rule model	Data Type	isRequired
data	id title description likes dislikes tag	number string string number number array of string	yes yes no no no no no
logic	increment() - increase likes or dislikes counter by 1		

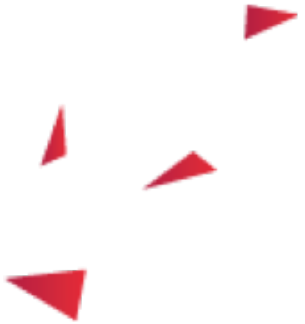

props = { rule }
React Component
html + css + data


React Component
Template (JSX)

{title}
{description}
{newTags}

<button>{likes}</button>
<button>{dislikes}</button>

props = { rule }
html + css + data



bind data and template

React Component

Template (JSX)

```
<div>
    <div>{title}</div>
    <div><p>{description}</p></div>
    <div>{newTags}<div>
    <div>
        <button>{likes}</button>
        <button>{dislikes}</button>
    </div>
</div>
```

render html + css + data

```
rule = {
    "id": 1,
    "title": "If you don't have a ...",
    "description": "In 2014, 50% of ...",
    "likes": 0,
    "dislikes": 0,
    "tags": ["ui"]
}
```

```
<div>
    <div>If you don't have a ...</div>
    <div><p>In 2014, 50% ...</p></div>
    <div>{newTags}<div>
    <div>
        <button>0</button>
        <button>0</button>
    </div>
</div>
```

```
<div>
    <div>If you don't have a ...</div>
    <div><p>In 2014, 50% ...</p></div>
    <div>{newTags}<div>
    <div>
        <button>0</button>
        <button>0</button>
    </div>
</div>
```

display html and css in browser

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

ul

0 0

Fundamentals of Testing

- Throw an Error with a Simple Test
- Build a JavaScript Assertion Library
- Abstract Test Assertions
- Build a JavaScript Testing Framework
- Encapsulate and Isolate Tests

Throw an Error with a Simple Test

```
const sum = (x, y) => x + y;

const result = sum(3, 7)
const expected = 5

if (result !== expected) {
  throw new Error(`${result} is not equal to ${expected}`)
}
```

Build a JavaScript Assertion Library

```
const expect = actual => {
  return {
    toBe(expected) {
      if (actual !== expected) {
        throw new Error(`${actual} is not equal to ${expected}`)
      }
    }
  }
}
```

Abstract Test Assertions

```
const sum = (x, y) => x + y;

const result = sum(3, 7)
const expected = 5

expect(result).toBe(expected)
```

Build a JavaScript Testing Framework

```
const test = (title, callback) =>{
  try {
    callback()
    console.log(`✓ ${title}`)
  } catch (error) {
    console.error(`✗ ${title}`)
    console.error(error)
  }
}
```

Encapsulate and Isolate Tests

```
test('sum adds numbers', () => {
  const result = sum(3, 7)
  const expected = 5
  expect(result).toBe(expected)
})
```

Test React Components with react-testing-library

react-testing-library

- Render a React component
- Avoid Memory leaks
- Debug the DOM state
- Test React Component Event Handlers
- Assert rendered text

Render a React component

```
import 'jest-dom/extend-expect'
import React from 'react'
import {render} from 'react-testing-library'
import MyComponent from  './MyComponent'

test('renders MyComponent', () => {
  const {getByLabelText} = render(< MyComponent />)
  const input = getByLabelText(/favorite number/i)
  expect(input).toHaveAttribute('type', 'number')
})
```

Avoid Memory leaks #1

```
import React from "react";
import { cleanup } from "@testing-library/react";
import "@testing-library/jest-dom/extend-expect";
import  MyComponent  from "./ MyComponent ";

describe(" MyComponent", () => {
  afterEach(cleanup);
  test("should increment counter", () => { });
});
```

Avoid Memory leaks #2

```
import React from "react";
import { fireEvent, render } from "@testing-library/react";
import "@testing-library/jest-dom/extend-expect";
import "@testing-library/react/cleanup-after-each";
import  MyComponent  from "./ MyComponent ";

describe(" MyComponent", () => {
  test("should increment counter", () => { });
});
```

Debug the DOM state

```
test("should increment counter", () => { });
 const {getByLabelText, debug} = render(<MyComponent />)
 const input = getByLabelText(/favorite number/i)
 expect(input).toHaveAttribute('type', 'number')
 debug(input)
})
```

Test React Component Event Handlers

```
import { cleanup, fireEvent, render } from "@testing-library/react";

describe("LikeBtn", () => {
  test("should increment counter", () => {
    const { getByTitle } = render(<LikeBtn type={"up"} counter={0} />);
    const likeButtonElement = getByTitle("+1");
    fireEvent.click(likeButtonElement);
  });
});
```

Assert rendered text

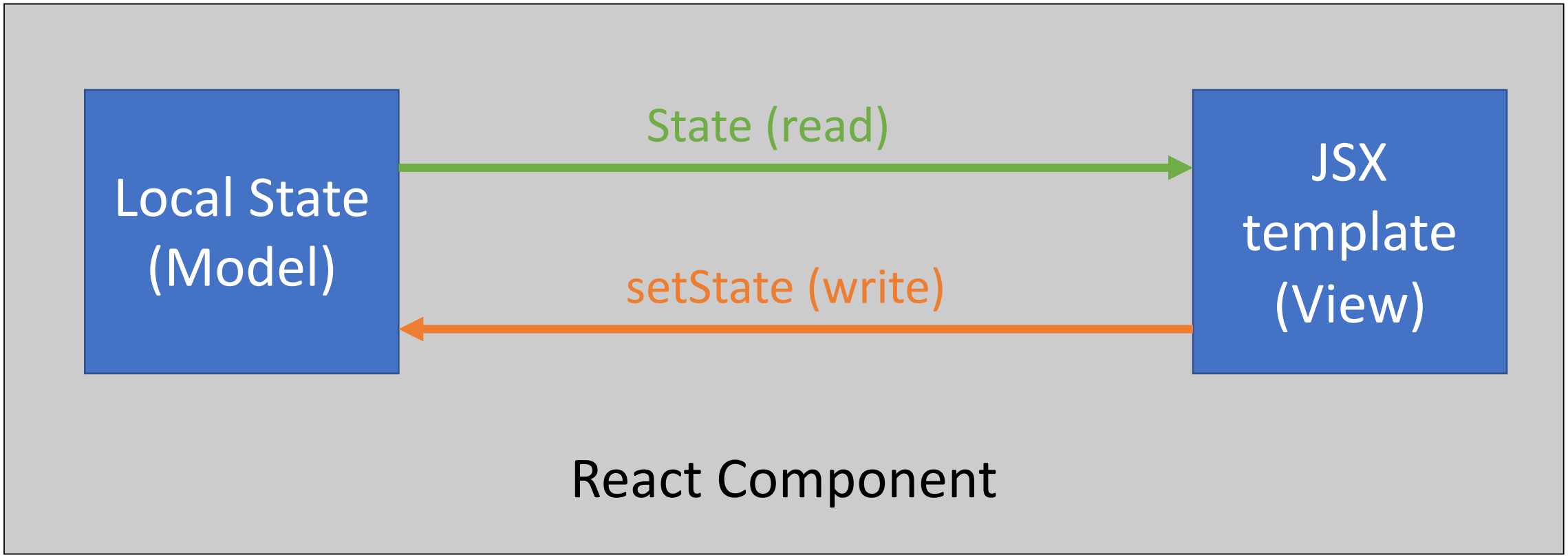
```
test("should increment counter", () => { });
  const {getByLabelText, getByTestId} = render(<MyComponent />)
  const input = getByLabelText(/favorite number/i)
  expect(getByTestId('error-message')).toHaveTextContent(
    /the number is invalid/i,
  )
})
```

Redux

Redux

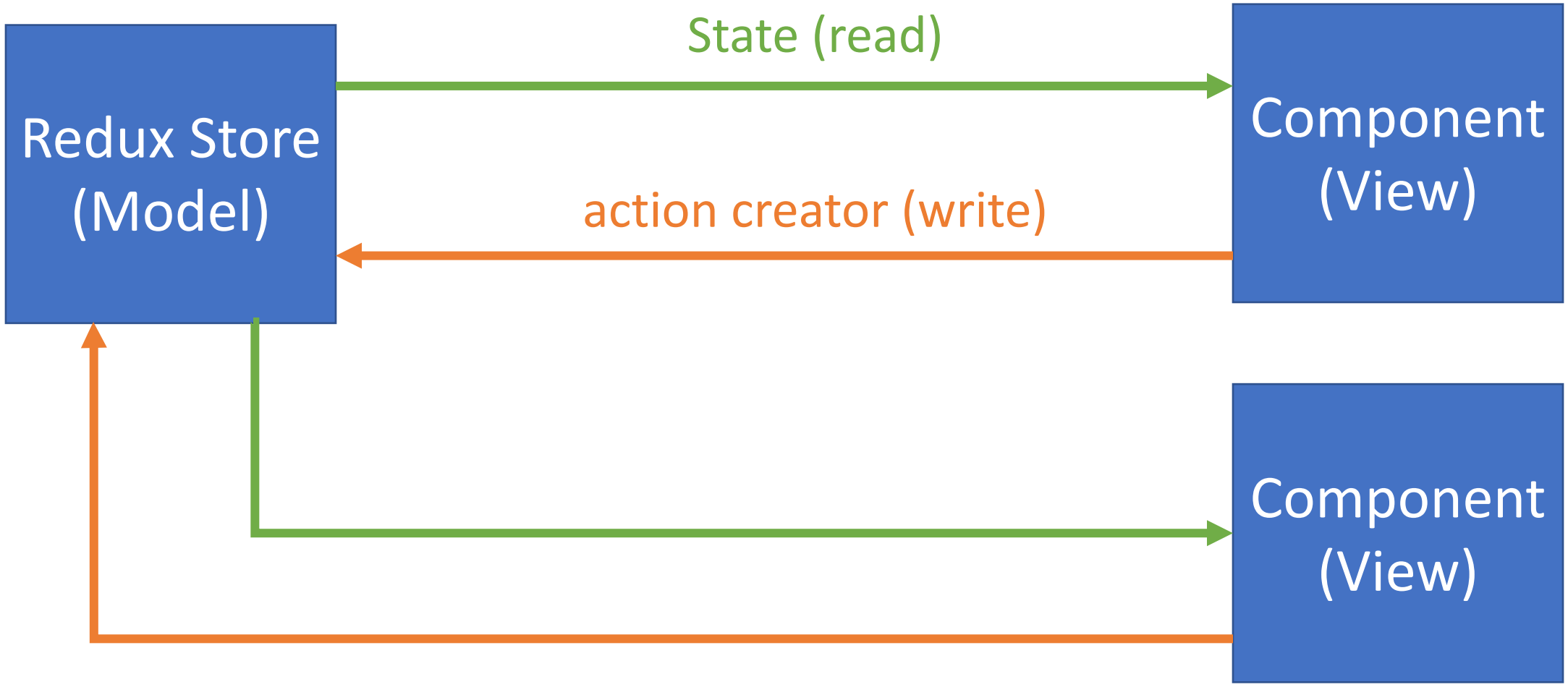
- Local State
- Global State
- Redux Cycle
- React-Redux
- Async Problem due to remote request

Local State




```
const [state, setState] = useState(initialValues);
```

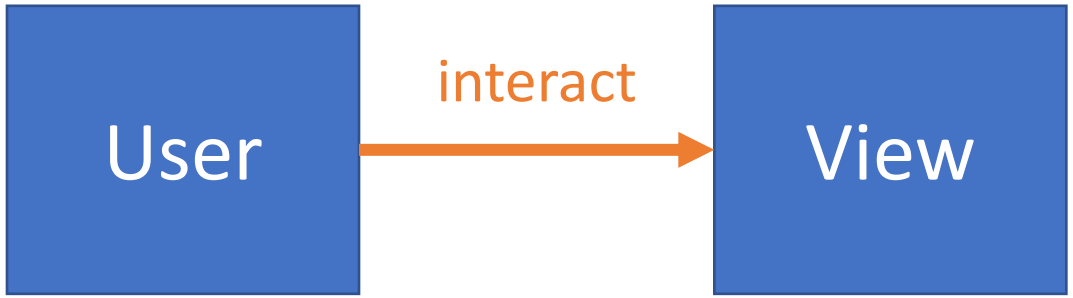
Global State



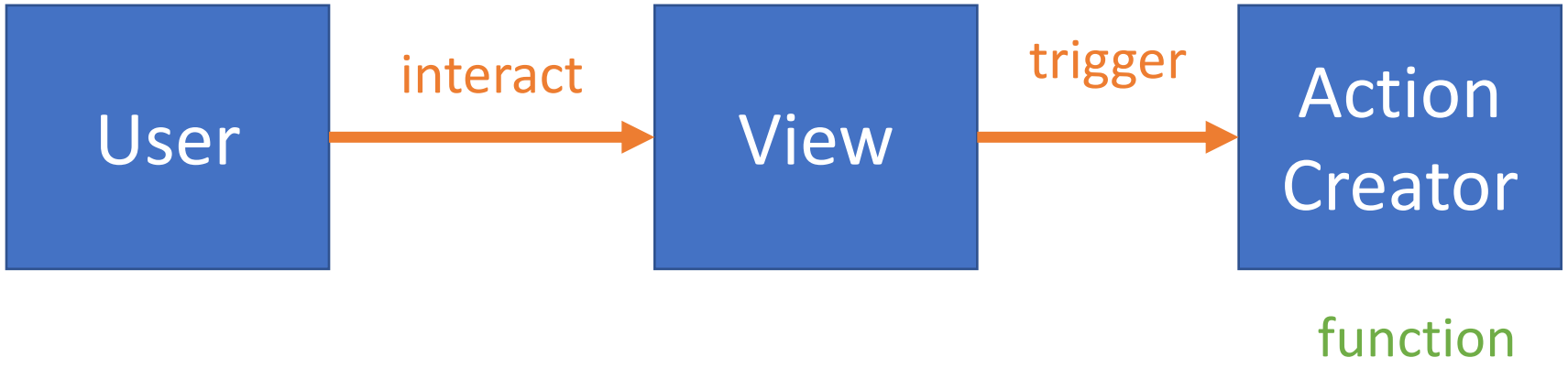
Redux Cycle

User

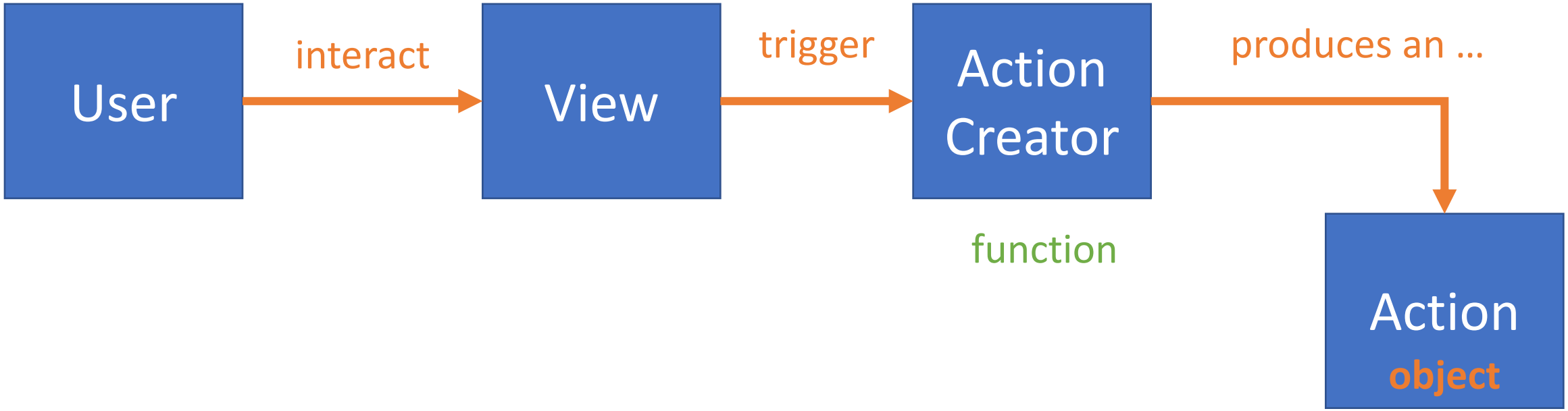
Redux Cycle



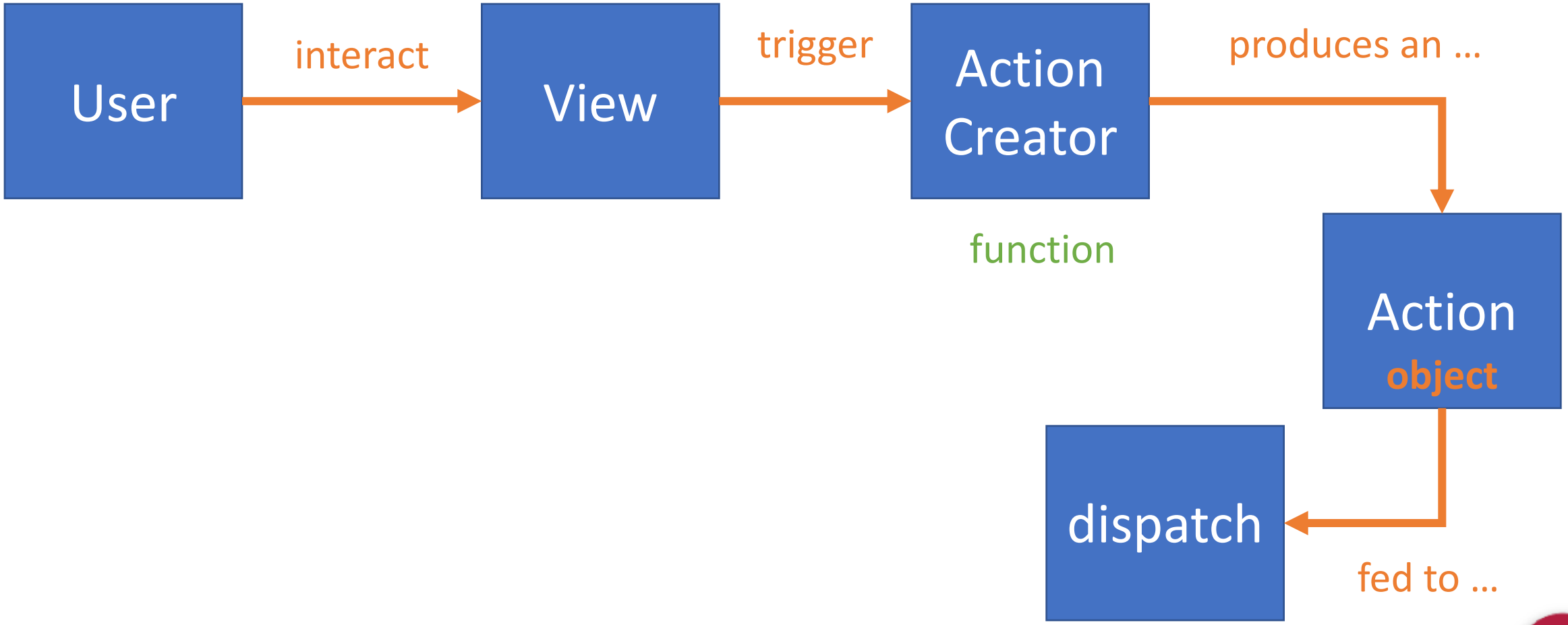
Redux Cycle



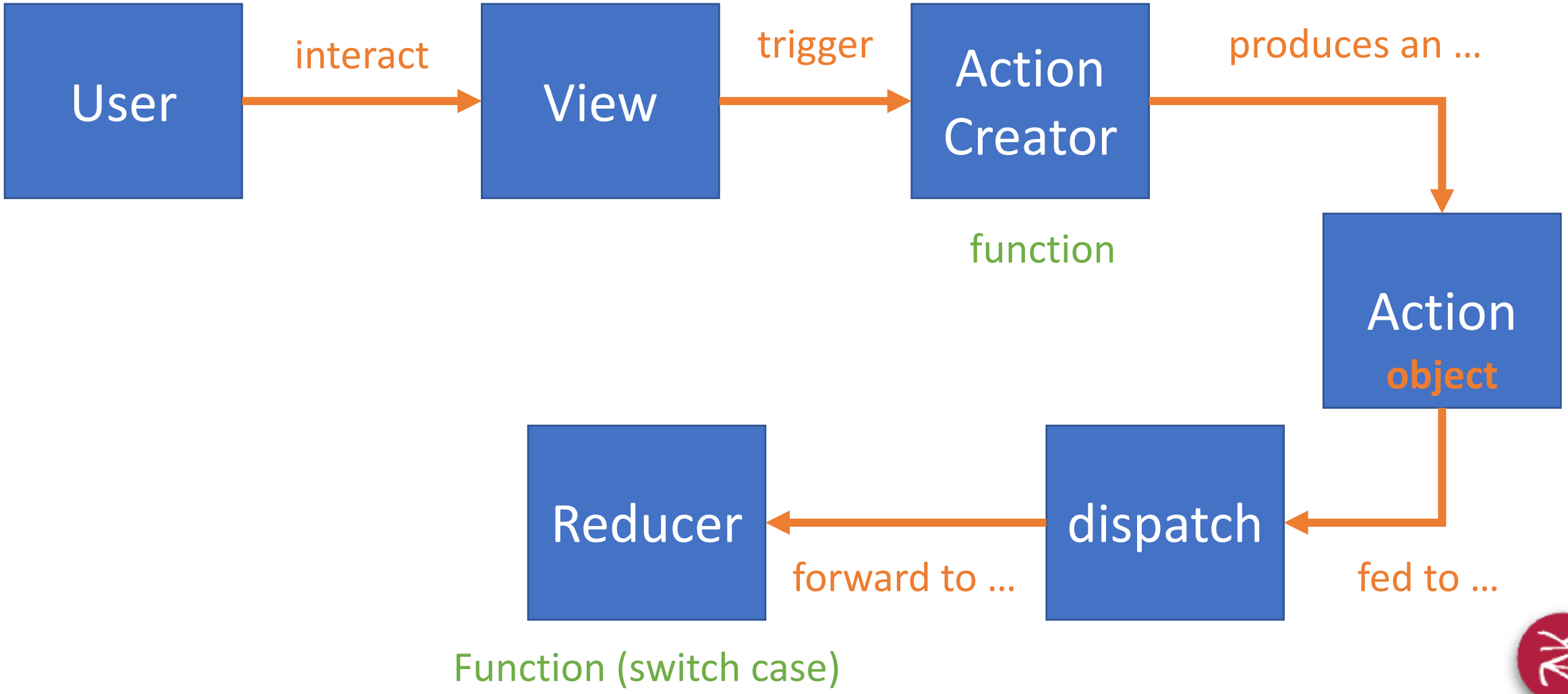
Redux Cycle


User
interact
View
trigger
Action Creator
function
produces an ...
Action
object

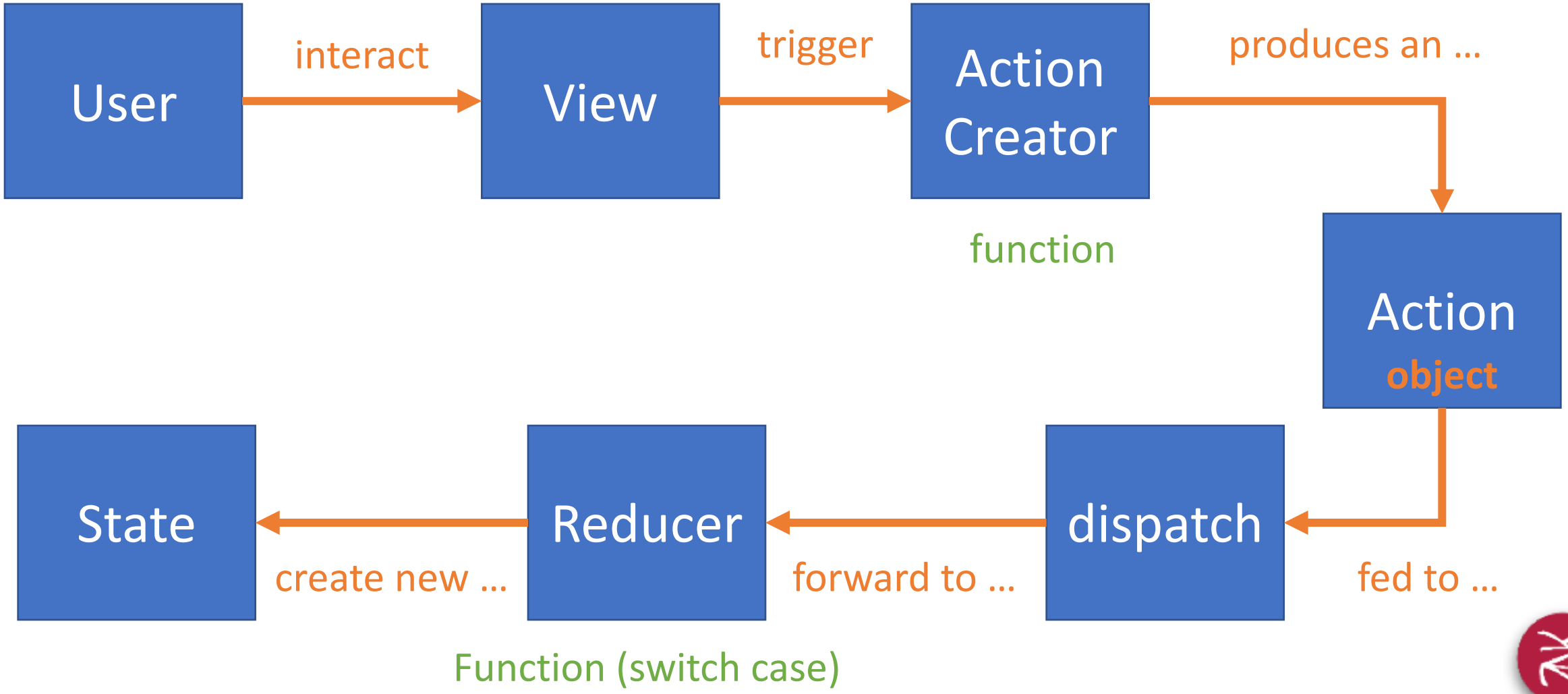
Redux Cycle


User
interact
View
trigger
Action Creator
function
produces an ...
Action
object
fed to ...
dispatch

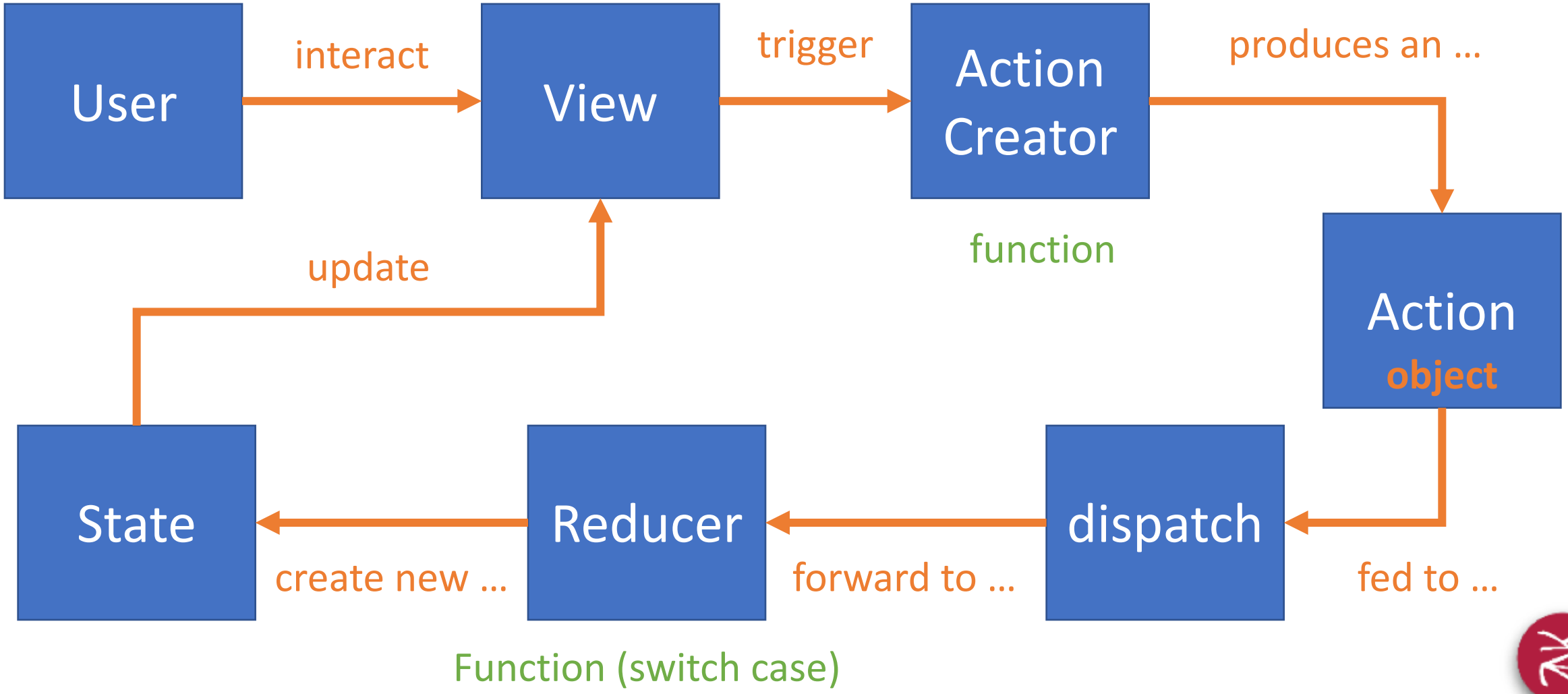
Redux Cycle


User
interact
View
trigger
Action Creator
function
produces an ...
Action
object
fed to ...
dispatch
forward to ...
Reducer
Function (switch case)

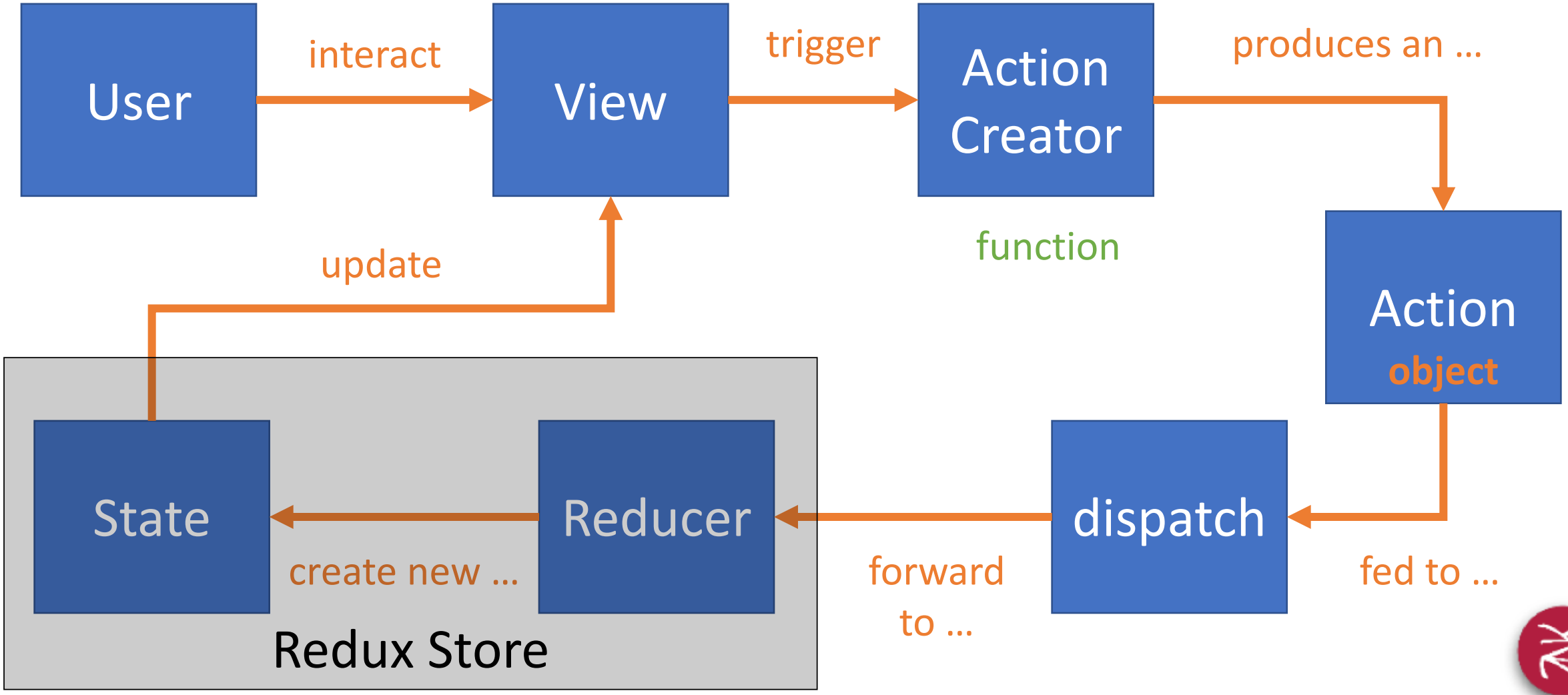
Redux Cycle


User
interact
View
trigger
Action Creator
function
produces an ...
Action
object
fed to ...
dispatch
forward to ...
Reducer
Function (switch case)
create new ...
State

Redux Cycle

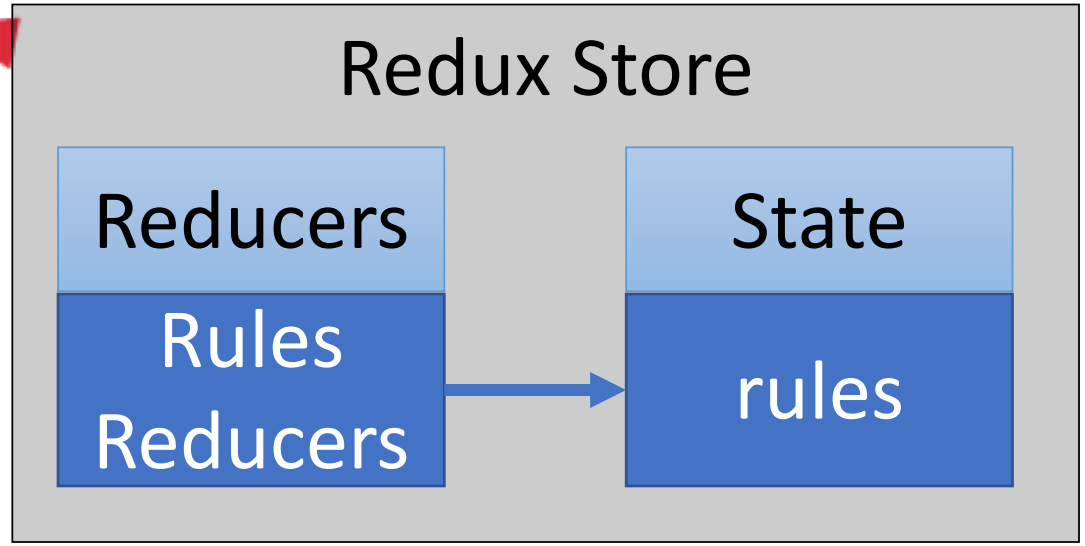

User
interact
View
trigger
Action Creator
function
produces an ...
Action object
fed to ...
dispatch
forward to ...
Reducer
Function (switch case)
create new ...
State
update

Redux Cycle

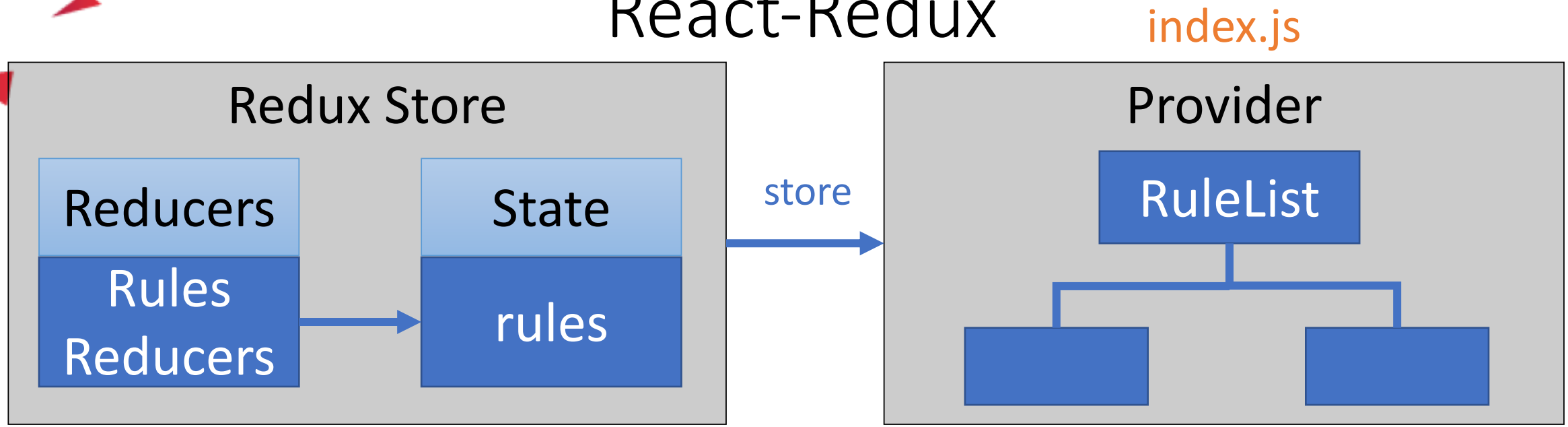

User
interact
View
trigger
Action Creator
produces an ...
function
Action object
fed to ...
dispatch
forward to ...
Reducer
create new ...
State
update
Redux Store

React-Redux

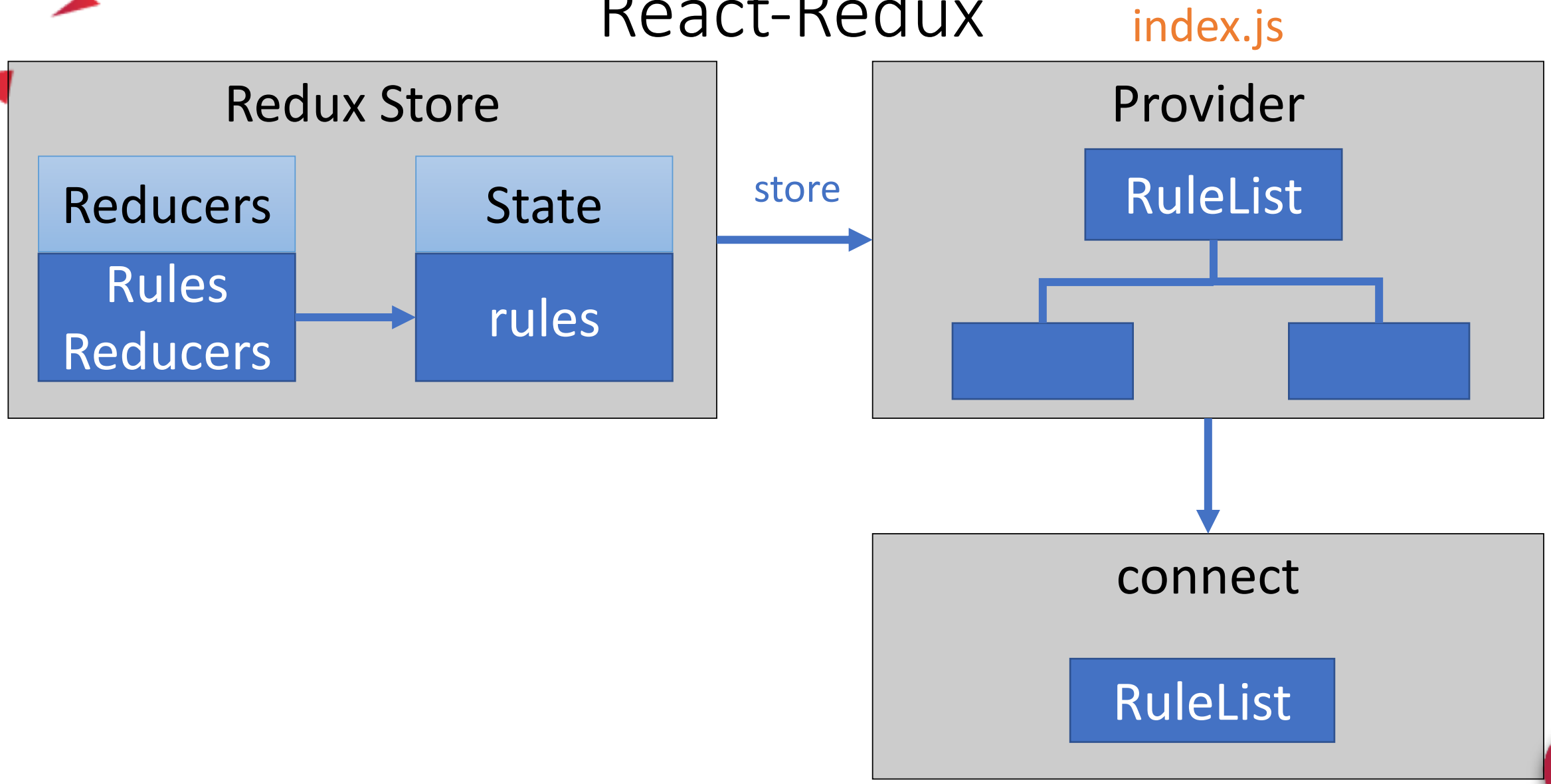
React-Redux



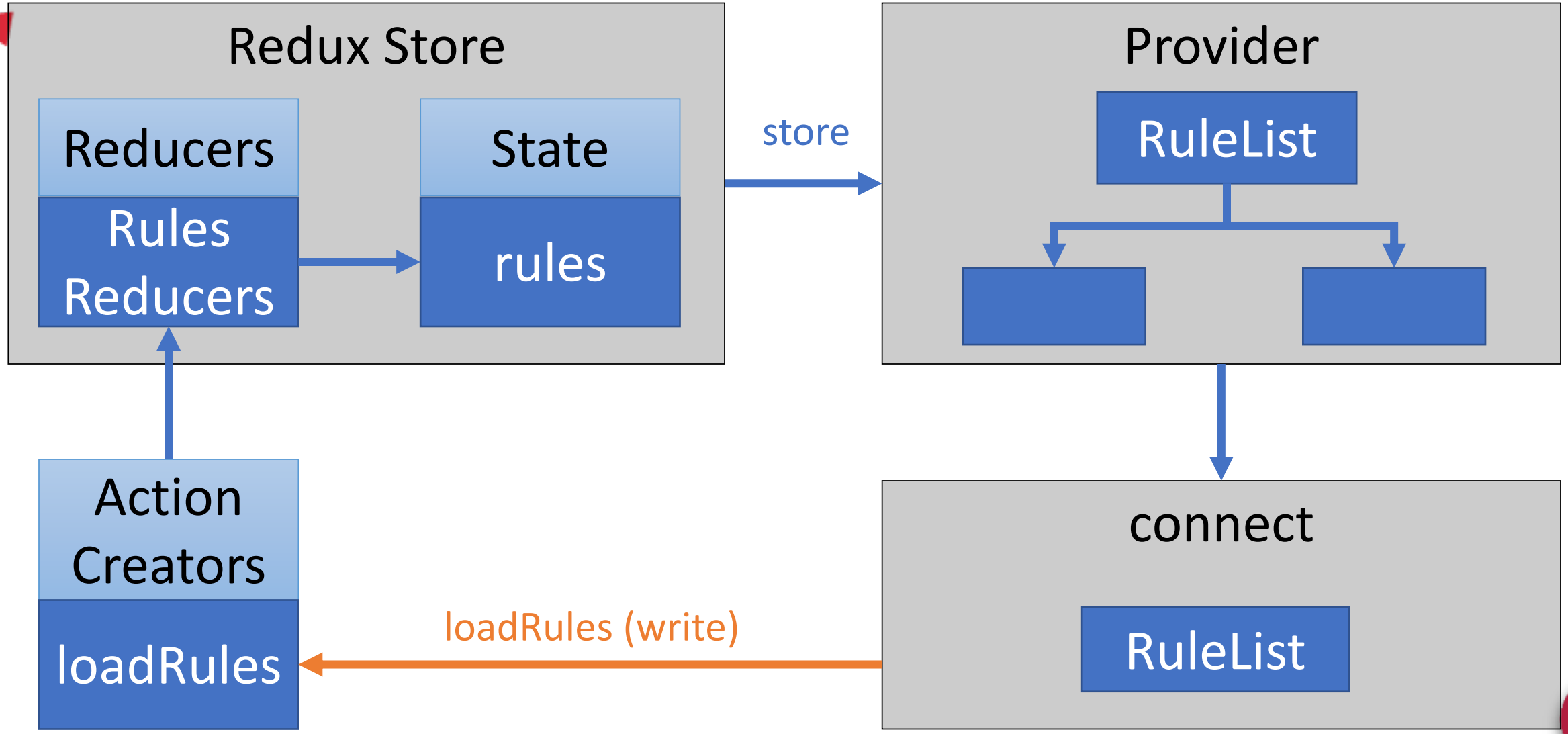
React-Redux


index.js
Redux Store
Reducers
State
Rules
Reducers
rules
store
Provider
RuleList

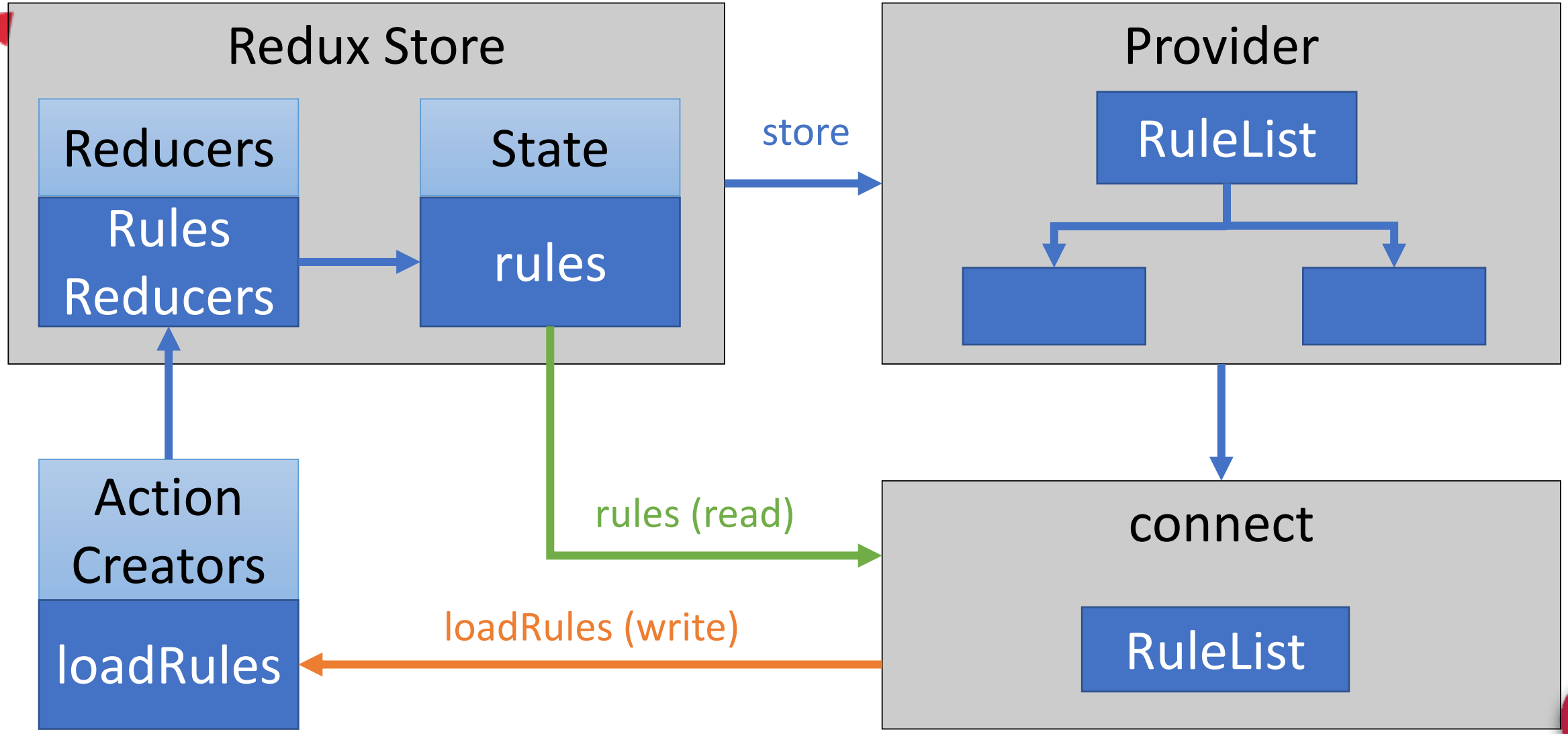
React-Redux


index.js
Redux Store
Reducers
State
Rules
Reducers
rules
store
Provider
RuleList
connect
RuleList

React-Redux: mapDispatchToProps

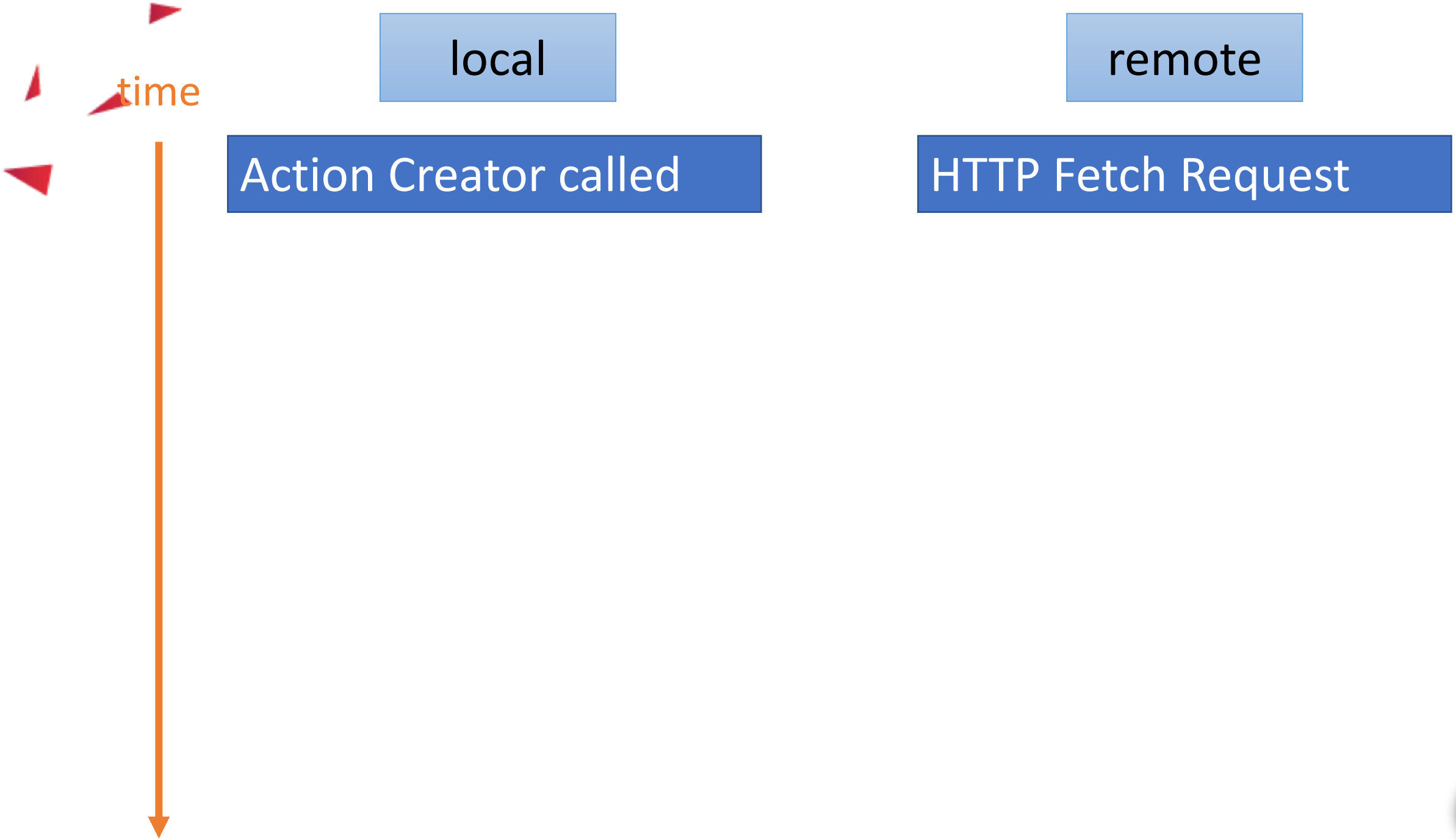


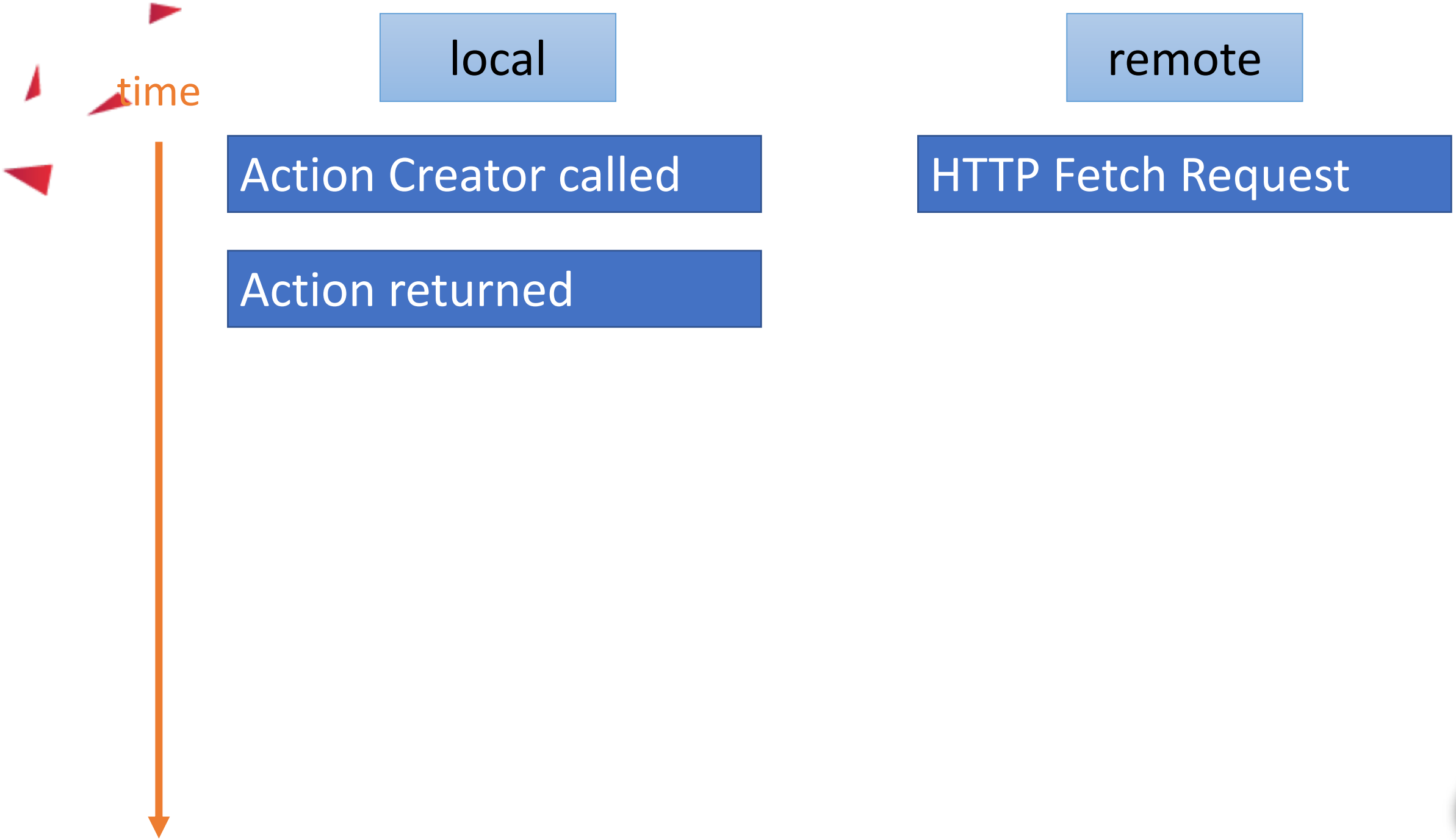
React-Redux: mapStateToProps

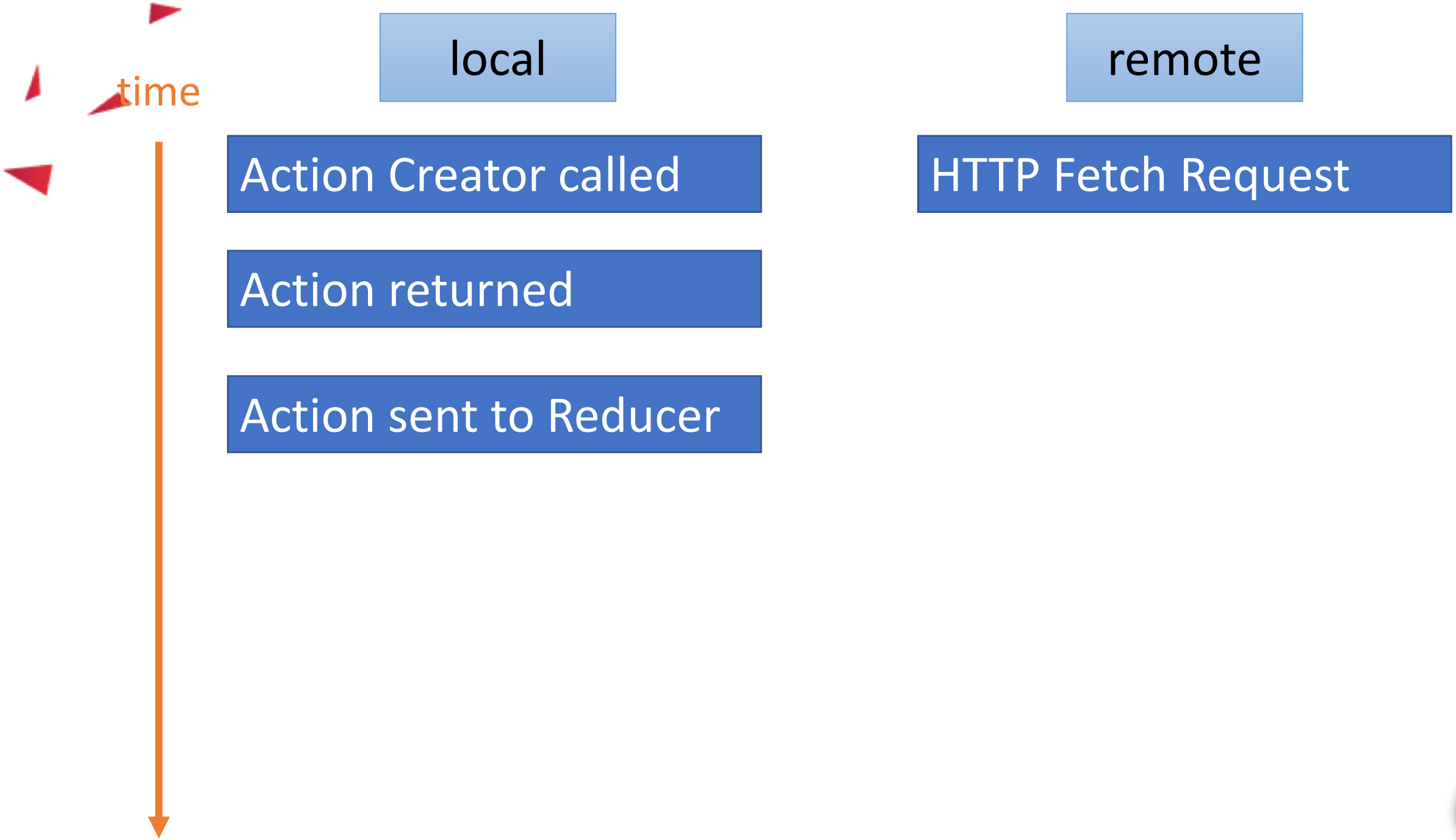


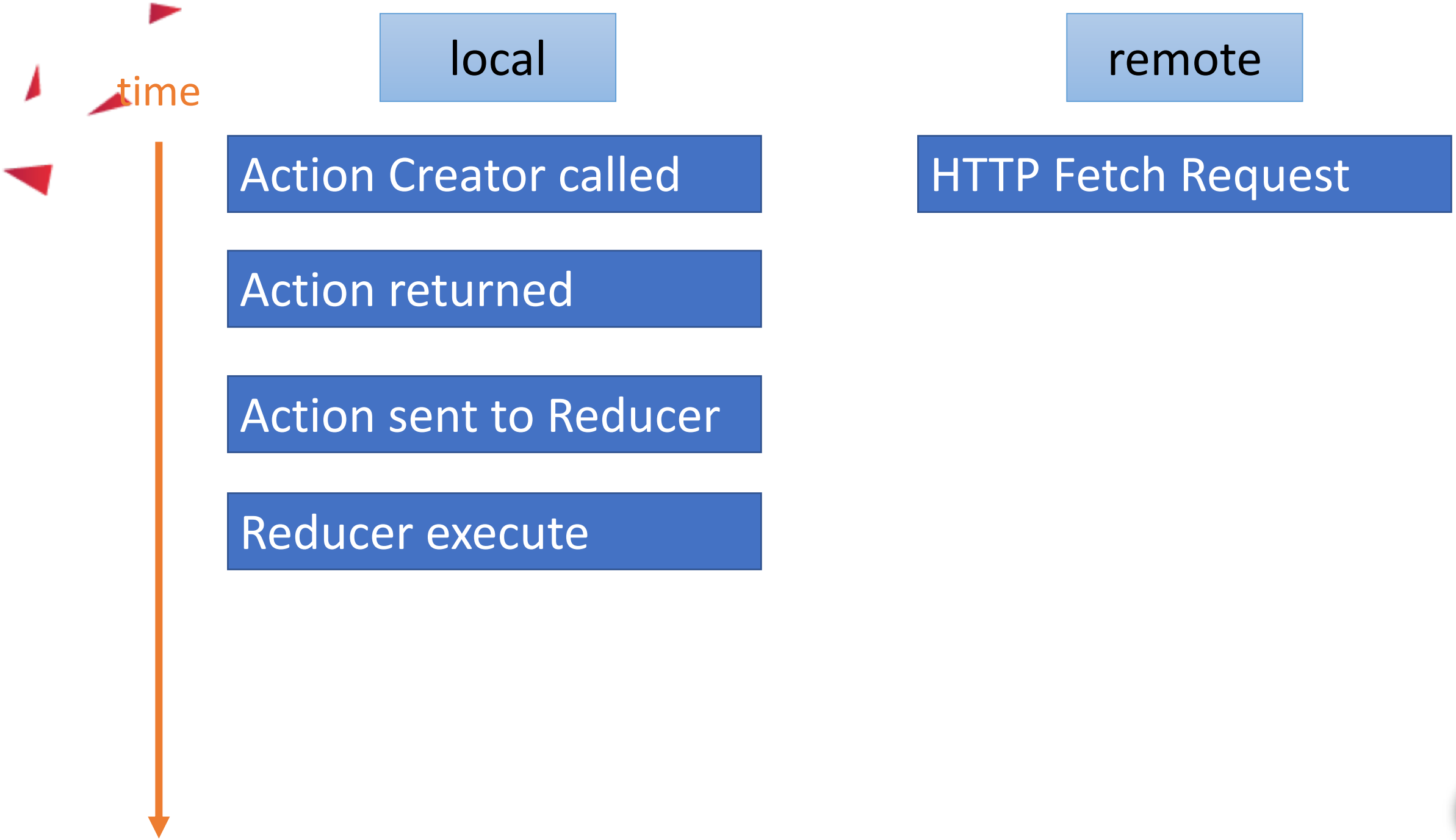
Async Problem due to remote request

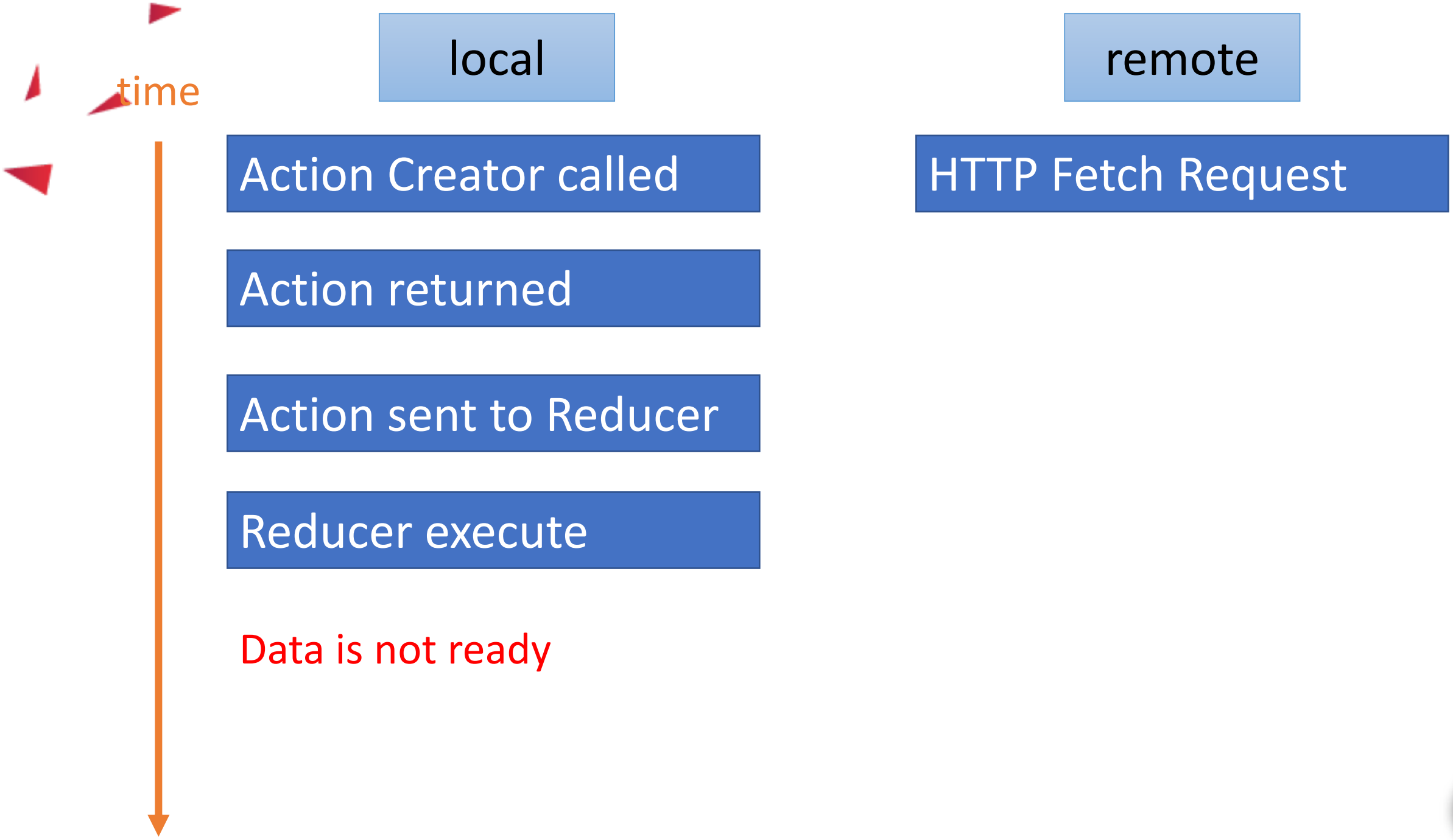
action gets consumed in fractions of a sec

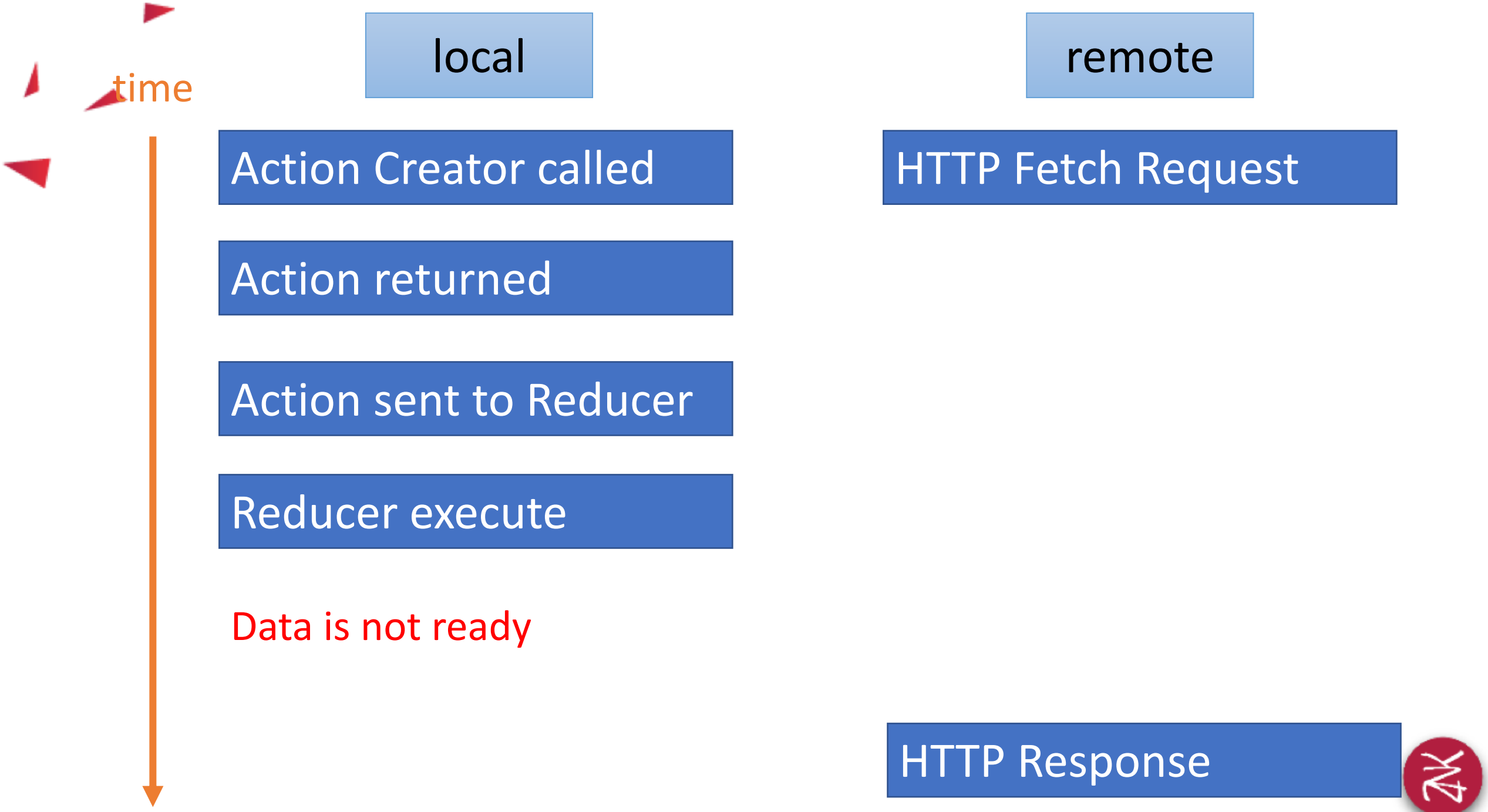

time
local
remote
Action Creator called
HTTP Fetch Request


time
local
remote
Action Creator called
HTTP Fetch Request
Action returned

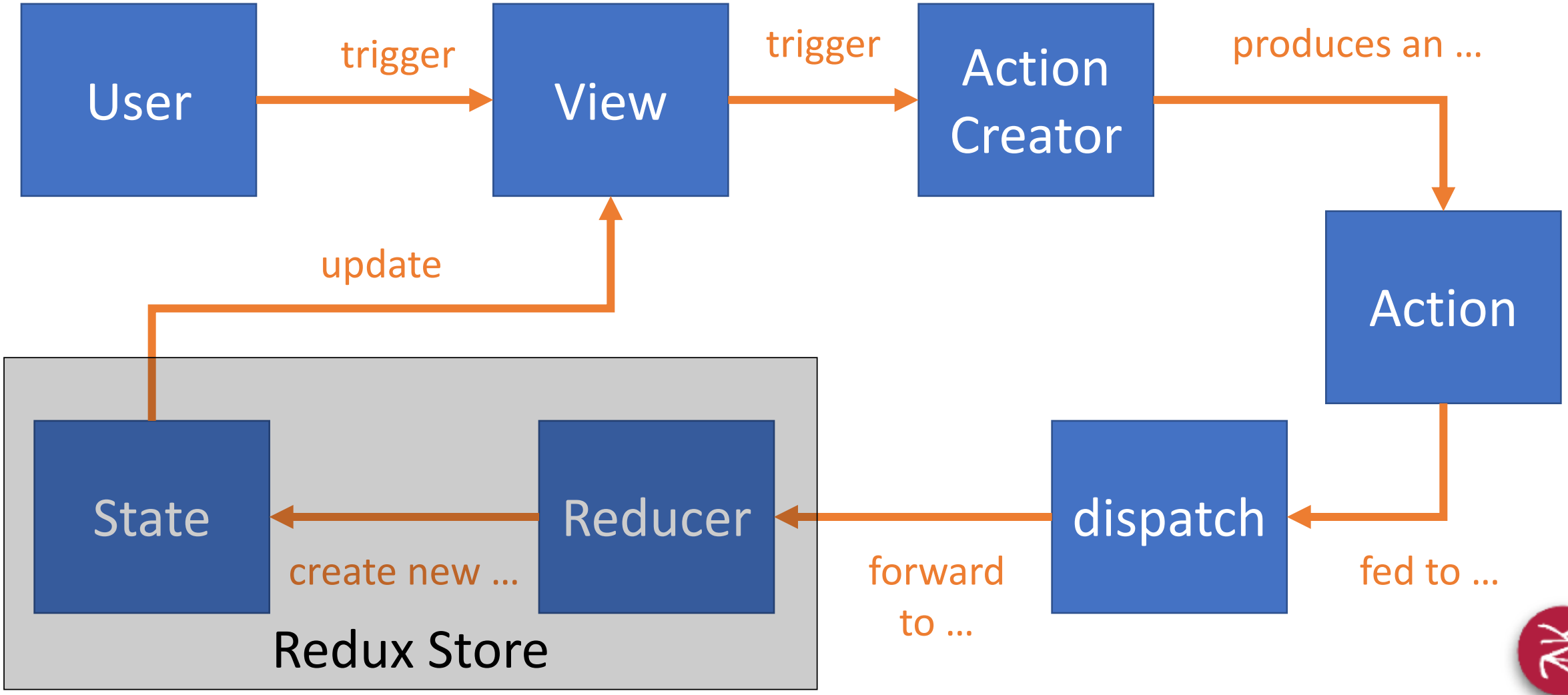

time
local
remote
Action Creator called
HTTP Fetch Request
Action returned
Action sent to Reducer


time
local
remote
Action Creator called
HTTP Fetch Request
Action returned
Action sent to Reducer
Reducer execute

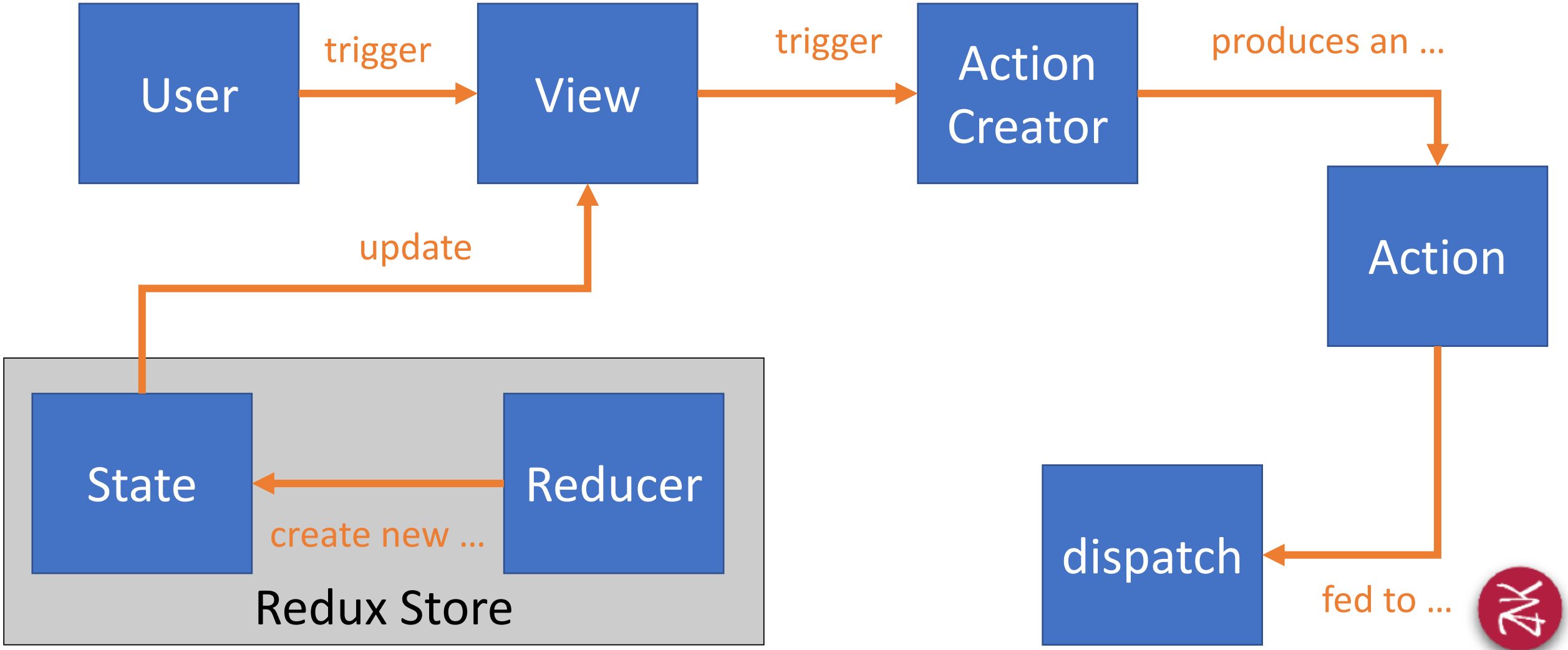

time
local
remote
Action Creator called
HTTP Fetch Request
Action returned
Action sent to Reducer
Reducer execute
Data is not ready


time
local
remote
Action Creator called
HTTP Fetch Request
Action returned
Action sent to Reducer
Reducer execute
Data is not ready
HTTP Response

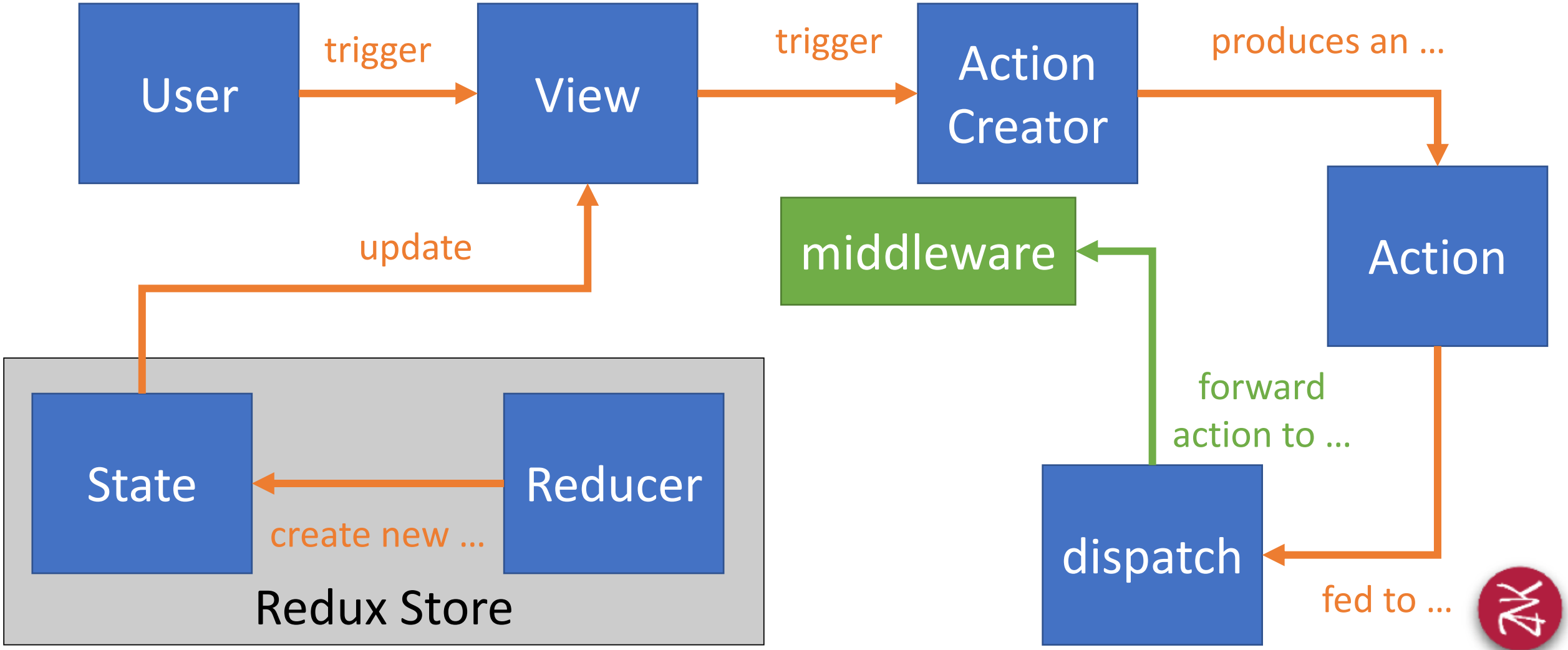
Redux Cycle


User
trigger
View
trigger
Action Creator
produces an ...
Action
fed to ...
dispatch
forward to ...
Reducer
create new ...
State
update
Redux Store

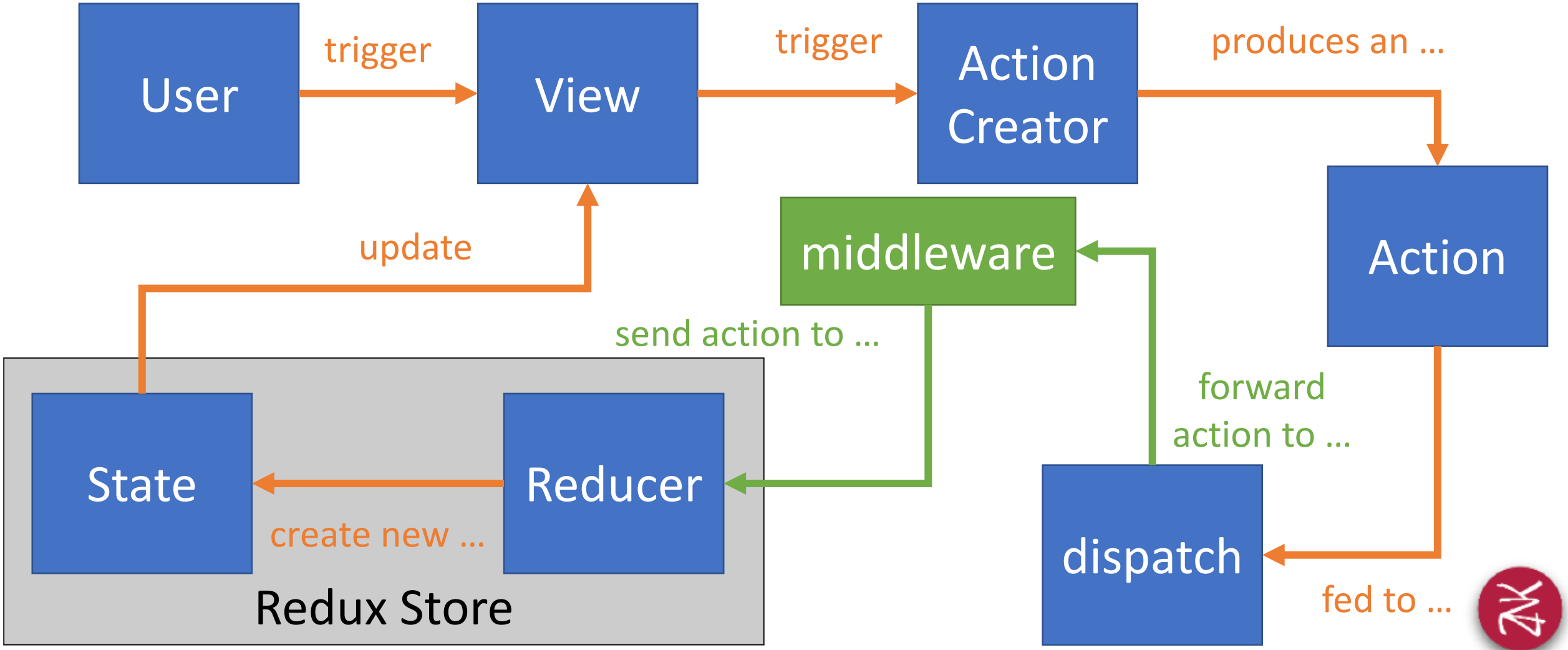
Redux Cycle


User
trigger
View
trigger
Action Creator
produces an ...
Action
fed to ...
dispatch
update
State
Reducer
create new ...
Redux Store

Redux Cycle


User
trigger
View
trigger
Action Creator
produces an ...
Action
middleware
update
forward action to ...
State
Reducer
create new ...
Redux Store
dispatch
fed to ...

Redux Cycle


User
trigger
View
trigger
Action Creator
produces an ...
Action
update
middleware
send action to ...
forward action to ...
State
Reducer
create new ...
Redux Store
dispatch
fed to ...

Axios

Axios

- Performing a GET request
- Performing a POST request
- Backend API

Performing a GET request

```
const getUser = () => {
  try {
    const response = await axios.get('/user/1');
    console.log(response);
  } catch (error) {
    console.error(error);
  }
}
```

Performing a POST request

```
const getUser = () => {
  try {
    const response = await axios.post('/user’, {
        firstName: 'Fred’,
        lastName: 'Flintstone'
    });
    console.log(response);
  } catch (error) {
    console.error(error);
  }
}
```

Backend API

User Feature	Backend API
List all users	axios.get('/users');
Get one user detail	axios.get('/users/1');
Create new user	axios.post('/users’, { … });
Update user	axios.put('/users/1’, { … });
Delete user	axios.delete('/users/1’);

Synchronize data between Frontend and Backend

post
rest/rules/1/likes

[{ id: 1,
Likes: 0},
{ },
{ }]

Frontend
Redux Store

[{ id: 1,
Likes: 0},
{ },
{ }]

Backend
Database

response

[{ id: 1,
Likes: 0},
{ },
{ }]

Frontend
Redux Store

[{ id: 1,
Likes: 1},
{ },
{ }]

Backend
Database

[{ id: 1,
Likes: 0},
{ },
{ }]

Frontend
Redux Store

{
type: DO_LIKE,
payload: 1
}

[{ id: 1,
Likes: 1},
{ },
{ }]

Backend
Database

[{ id: 1,
Likes: 1},
{ },
{ }]

Frontend
Redux Store

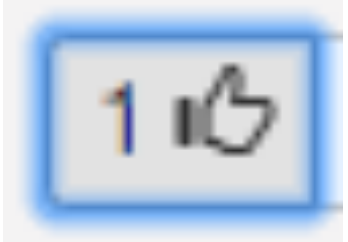
{
type: DO_LIKE,
payload: 1
}

[{ id: 1,
Likes: 1},
{ },
{ }]

Backend
Database

[{ id: 1,
Likes: 1},
{ },
{ }]

Frontend
Redux Store



[{ id: 1,
Likes: 1},
{ },
{ }]

Backend
Database

Design Routing Table

Developers rules

Home

New

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

ui

0 0

Leave the code cleaner than you found it.

From Clean Code: always leave the code cleaner than it was before.

craftsmanship clean code

0 0

Never say : "I've done, it works on my machine !" #itworksonmymachine

0 0

Always use === in JavaScript!

javascript

0 0

Developers rules Home New

New rule

Title

Title

Description

Description

Submit

Developers rules Home New

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

ui

0 0

Leave the code cleaner than you found it.

From Clean Code: always leave the code cleaner than it was before.

craftsmanship clean code

0 0

Never say : "I've done, it works on my machine !" "it works on my machine"

0 0

Always use === in JavaScript!

javascript

0 0

Region

(Layout View)

Layout Component

Developers rules Home New

Region (Header View)

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

ui

0 0

Leave the code cleaner than you found it.

From Clean Code: always leave the code cleaner than it was before.

craftsmanship clean code

0 0

Region (Router)

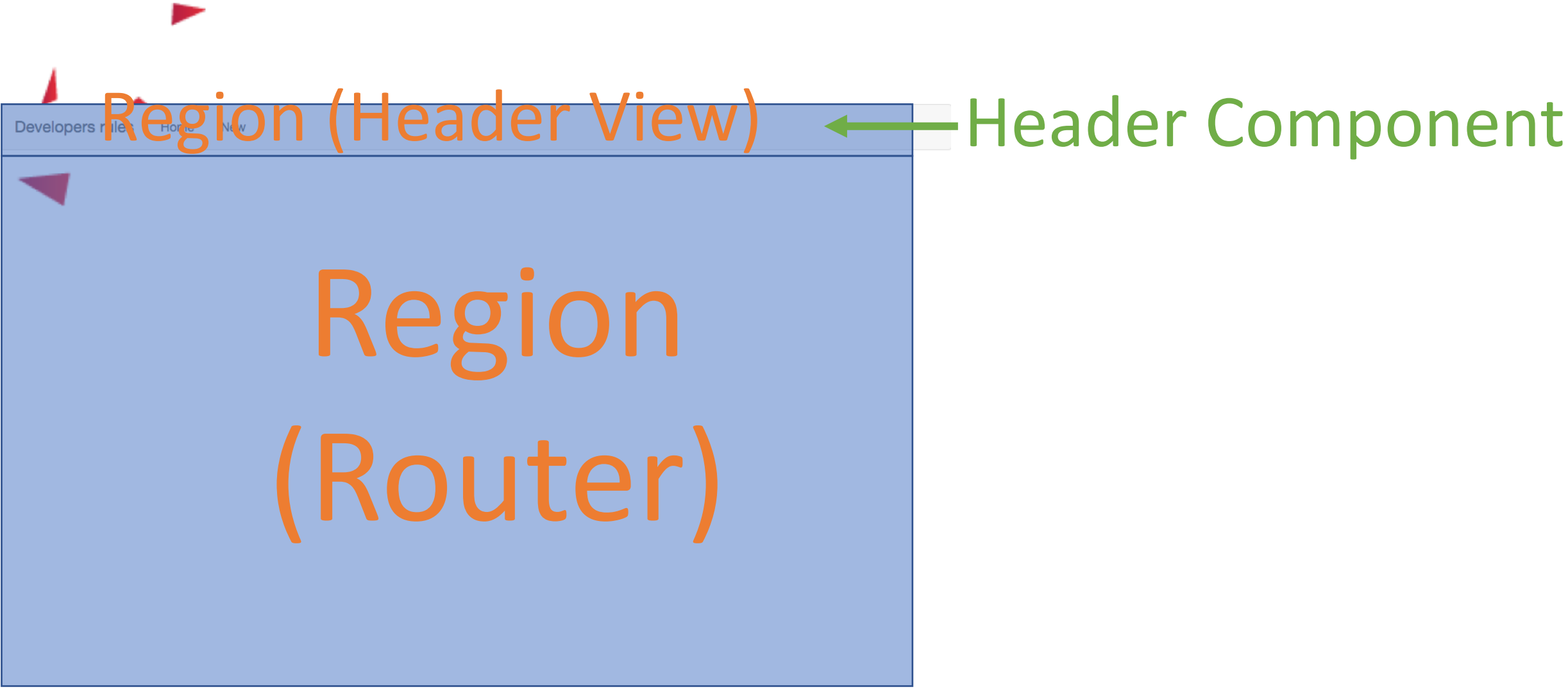
Never say : "I've done, it works on my machine" #itworksonmymachine

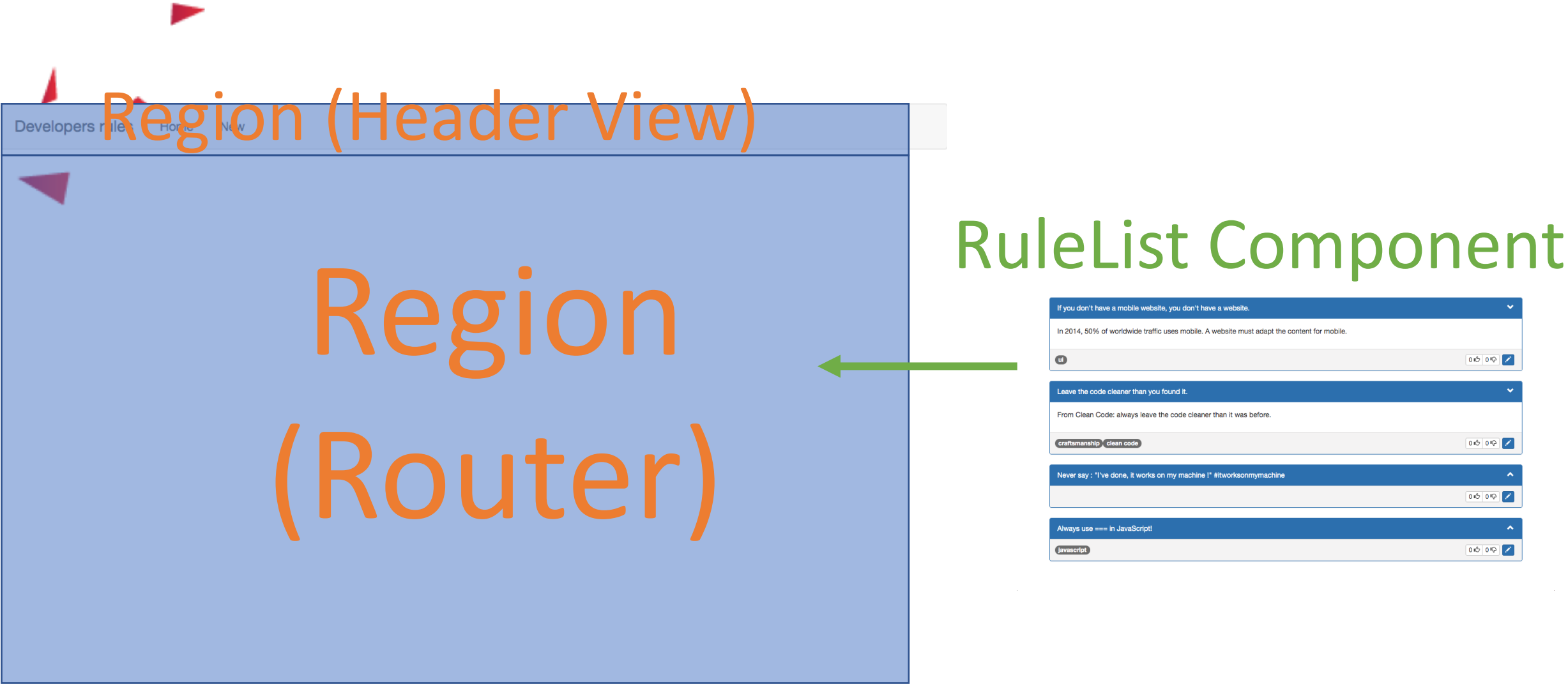
0 0

Always use === in JavaScript!

javascript

0 0


Region (Header View)
Header Component
Developers rules
Home
New
Region
(Router)


Region (Header View)
Developers rules
Home
New
Region
(Router)
RuleList Component
If you don't have a mobile website, you don't have a website.
In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.
ui
Leave the code cleaner than you found it.
From Clean Code: always leave the code cleaner than it was before.
craftsmanship
clean code
Never say : "I've done, it works on my machine !" #itworksonmymachine
Always use === in JavaScript!
javascript

Region (Header View)

Region (Router)

RuleList Component

If you don't have a mobile website, you don't have a website.

In 2014, 50% of worldwide traffic uses mobile. A website must adapt the content for mobile.

ui

Leave the code cleaner than you found it.

From Clean Code: always leave the code cleaner than it was before.

craftsmanship clean code

Never say : "I've done, it works on my machine !" #itworksonmymachine

Always use === in JavaScript!

javascript

RuleForm Component

New rule

Title

Description

Submit

Routing Table

Route	Component
/	Layout
/	RuleList
/new	RuleForm
/edit/:id	RuleForm

Convert a React Form to Formik

Formik and Yup

- Simple To do List
- Create html form
- Use local state
- onChange event
- onSubmit event

Simple To do List

To do List:

Item:

Learn React

Add Item

To do List:

- Learn React

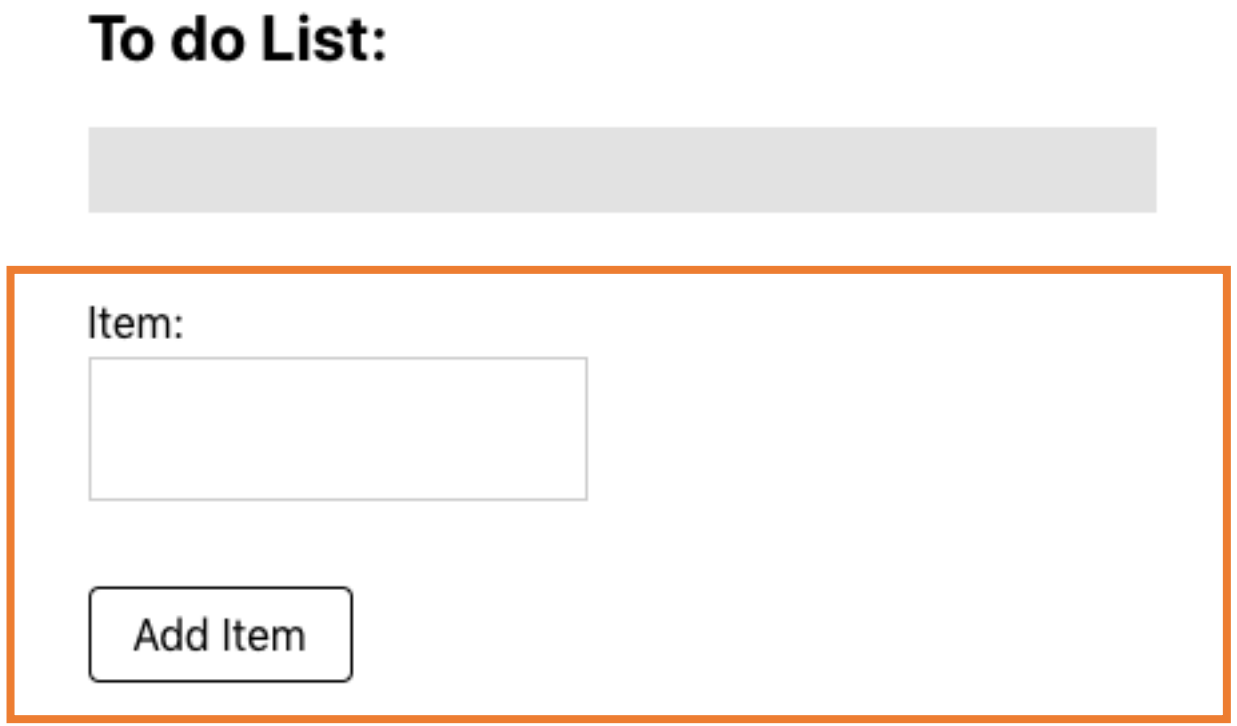
Item:

Learn React

Add Item

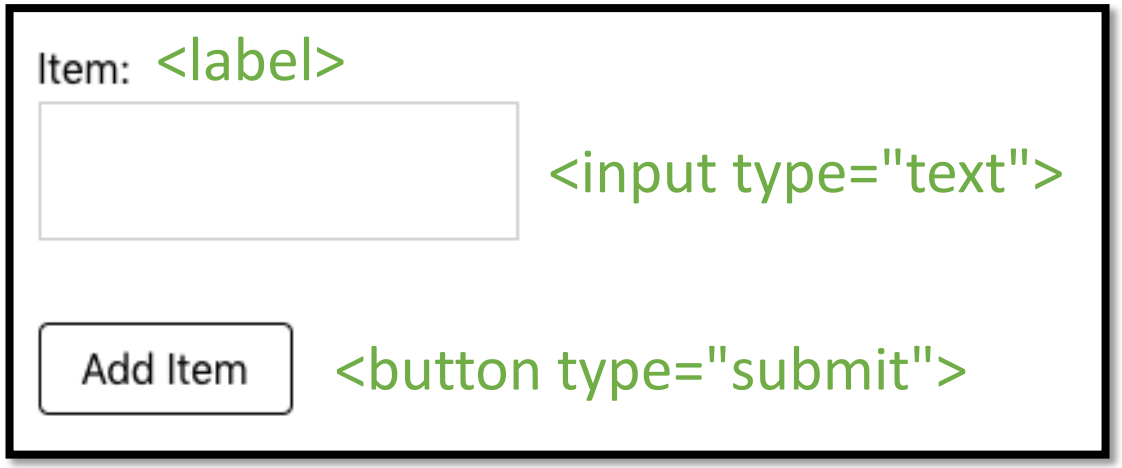
Simple To do List

html form



Create html form

<form>



</form>

```
<form onSubmit={handleSubmit}>
    <label htmlFor="name">
        Item:
    </label>
    <input type="text"
        value={item}
        onChange={handleChange} />
    <button type="submit">
        Add Item
    </button>
</form>
```

Use local state

variable	state
items	[]
item	""

```
const [items, setItems] = useState([]);
const [item, setItem] = useState("");
```

Item:

Add Item

Set onChange event handler

variable	state
items	[]
item	""

Item:

Add Item

```
<form onSubmit={handleSubmit}>
    <label htmlFor="name">
        Item:
    </label>
    <input type="text"
        value={item}
        onChange={handleChange} />
    <button type="submit">
        Add Item
    </button>
</form>
```

onChange event handler

variable	state
items	[ ]
item	"Learn React"

Item:

Learn React

Add Item

```
const handleChange =
({ target: { value } }) => {
    setItem(value);
};
```

Set onSubmit event handler

variable	state
items	[]
item	""

Item:

Add Item

```
<form onSubmit={handleSubmit}>
    <label htmlFor="name">
        Item:
    </label>
    <input type="text"
        value={item}
        onChange={handleChange} />
    <button type="submit">
        Add Item
    </button>
</form>
```

onSubmit event handler

variable	prevState	newState
items	[ ]	["Learn React"]
item	"Learn React"	""


```
const handleSubmit = event => {
  event.preventDefault();
  setItems(prevState
          => [...prevState, item]);
  setItem("");
};
```

html form issues

- One input field needs
  - One local state
  - One onChange event handler
- Need to set initial values
- Need to implement form validation

Convert to Formik (use Form and Field)

```
<form onSubmit={handleSubmit}>
    <label htmlFor="name">Item:</label>
    <input type="text"
           value={item}
           onChange={handleChange} />
    <button type="submit">Add Item</button>
</form>
```

```
<Formik onSubmit={handleSubmit}>
  <Form>
    <label htmlFor="name">Item:</label>
    <Field type="text"
           name="item" />
    <button type="submit">Add Item</button>
  </Form>
</Formik>
```

Formik Properties

```
<Formik
      initialValues={initialValues}
      validationSchema={validationSchema}
      onSubmit={ … }
    >
    <Form>
      <label htmlFor="name">Item:</label>
      <Field type="text" name="item" />
      <ErrorMessage name="item" />
      <button type="submit">Add Item</button>
    </Form>
</Formik>
```

Set initial values

```
const [item, setItem] = useState("");
```

```
<form onSubmit={handleSubmit}>
    <label htmlFor="name">Item:</label>
    <input type="text"
            value={item}
            onChange={handleChange} />
    <button type="submit">Add Item</button>
</form>
```

```
const initialValues = { item: "" };

<Formik onSubmit={handleSubmit}
        initialValues={initialValues}
>
  <Form>
    <label htmlFor="name">Item:</label>
    <Field type="text" name="item" />
    <button type="submit">Add Item</button>
  </Form>
</Formik>
```

Yup - object schema validator

```
const validationSchema = Yup.object().shape({
  item: Yup.string().required("Item name is required")
});
<Formik
      onSubmit={handleSubmit}
      initialValues={initialValues}
      validationSchema={validationSchema}
    >
      <Form>
        <label htmlFor="name">Item:</label>
        <Field type="text" name="item" />
        <ErrorMessage name="item" />
        <button type="submit">Add Item</button>
      </Form>
    </Formik>
```

Html form vs formik

html form	formik
One input field needs • One local state • One onChange event handler const [item, setItem] = useState(""); <form onSubmit={handleSubmit}> <label htmlFor="name">Item:</label> <input type="text" value={item} onChange={handleChange} /> <button type="submit">Add Item</button> </form>	• Handle by Field component <Form> <label htmlFor="name">Item:</label> <Field type="text" name="item" /> <button type="submit">Add Item</button> </Form>
• Need to set initial values	• Handle by formik initialValues props const initialValues = { item: "" }; <Formik initialValues={initialValues}></Formik>

Html form vs formik

html form	formik

• Need to implement form validation
 	
const validateItem = item => {
if (!item) return "Item name is required";
return undefined;
};
<Formik>
<Form>
<label htmlFor="name">Item:</label>
<Field type="text" name="item"
validate={validateItem}
/>
<ErrorMessage name="item" />
<button type="submit">Add Item</button>
</Form>
</Formik>
	• Validate by Yup and formik validationSchema props
	• Display error with ErrorMessage component
	
	const validationSchema = Yup.object().shape({
	item: Yup.string().required("Item name is required")
	});
	<Formik validationSchema={validationSchema}>
	<Form>
	<label htmlFor="name">Item:</label>
	<Field type="text" name="item" />
	<ErrorMessage name="item" />
	<button type="submit">Add Item</button>
	</Form>
	</Formik>