link: null
title: 珠峰架构师成长计划
description: src\react\index.js
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=93 sentences=186, words=1455

1. 初始化项目

- react16.6 (https://gitee.com/zhufengpeixun/react16.6)

```
create-react-app 1.jsx
```

2. 实现虚拟DOM

2.1 src\index.js

```
import React, { Component } from './react';
import ReactDOM from 'react-dom';

class App extends Component {
  render() {
    return (
      <div>
        <p>1p>
        <button>+button>
      div>
    )
  }
}
let element = <App />;
console.log(element);

ReactDOM.render(
  element
  , document.getElementById('root'));
```

2.2 react\index.js

src\react\index.js

```
import { createElement } from './ReactElement';
import { Component } from './ReactBaseClasses';
const React = {
    createElement
}
export {
    Component
}
export default React;
```

2.3 react\ReactElement.js

src\react\ReactElement.js

```
import ReactCurrentOwner from './ReactCurrentOwner';
import { REACT_ELEMENT_TYPE } from '../shared/ReactSymbols';
const RESERVED_PROPS = {
    key: true,
    ref: true,
    __self: true,
    __source: true,
};
function hasValidRef(config) {
    return config.ref !== undefined;
}

function hasValidKey(config) {
    return config.key !== undefined;
}
export function createElement(type, config, children) {
    let propName;

    const props = {};
    let key = null;
    let ref = null;
    let self = null;
    let source = null;

    if (config != null) {

        if (hasValidRef(config)) {
            ref = config.ref;
        }
        if (hasValidKey(config)) {
            key = '' + config.key;
        }

        self = config.__self === undefined ? null : config.__self;
        source = config.__source === undefined ? null : config.__source;

        for (propName in config) {
            if (!RESERVED_PROPS.hasOwnProperty(propName)) {
                props[propName] = config[propName];
            }
        }
    }

    const childrenLength = arguments.length - 2;
    if (childrenLength === 1) {
        props.children = children;
    } else if (childrenLength > 1) {
        const childArray = Array(childrenLength);
        for (let i = 0; i < childrenLength; i++) {
            childArray[i] = arguments[i + 2];
        }
        props.children = childArray;
    }

    if (type && type.defaultProps) {
        const defaultProps = type.defaultProps;
        for (propName in defaultProps) {
            if (props[propName] === undefined) {
                props[propName] = defaultProps[propName];
            }
        }
    }
    return ReactElement(
        type,
        key,
        ref,
        self,
        source,
        ReactCurrentOwner.current,
        props,
    );
}

const ReactElement = function (type, key, ref, self, source, owner, props) {
    const element = {

        $typeof: REACT_ELEMENT_TYPE,

        type: type,
        key: key,
        ref: ref,
        props: props,

        _owner: owner,
    };
    element._self = self;
    element._source = source;
    return element;
};
```

2.4 ReactCurrentOwner.js

src\react\ReactCurrentOwner.js

```
const ReactCurrentOwner = {
    current: null
};

export default ReactCurrentOwner;
```

src\shared\ReactSymbols.js

```
const hasSymbol = typeof Symbol === 'function' && Symbol.for;
export const REACT_ELEMENT_TYPE = hasSymbol
    ? Symbol.for('react.element')
    : 0xeac7;
export const REACT_FORWARD_REF_TYPE = hasSymbol
    ? Symbol.for('react.forward_ref')
    : 0xead0;
```

2.6 react\ReactBaseClasses.js

src\react\ReactBaseClasses.js

```
class Component {
    constructor(props, context) {
        this.props = props;
        this.context = context;
    }
}
    Component.prototype.isReactComponent = {
}
class PureComponent extends Component {
}
PureComponent.prototype.isPureReactComponent = {}

export { Component, PureComponent };
```

3. React.Children.map

- React.Children.map (https://reactjs.org/docs/react-api.html#reactchildrenmap)

```
React.Children.map(children, function[(thisArg)])
```

3.1 实现单个映射

3.1.1 src\index.js

src\index.js

```
import React, { Component } from './react';
import ReactDOM from 'react-dom';
class Child extends Component {
  render() {
    console.log(this.props.children);
    const mappedChildren = React.Children.map(
      this.props.children,
      (item, index) => (
        <div key={index}>{item}div>
      )
    );
    console.log(mappedChildren);
    return (
      <div>
        {mappedChildren}
      div>
    )
  }
}
class App extends Component {
  render() {
    return (
      <Child><span >Aspan>Child>
    )
  }
}
ReactDOM.render(<App />, document.getElementById('root'));
```

3.1.2 react\ReactChildren.js

src\react\ReactChildren.js

```
import { REACT_ELEMENT_TYPE } from '../shared/ReactSymbols';

function mapChildren(children, mapFunction, context) {
    const result = [];
    mapIntoWithKeyPrefixInternal(children, result, mapFunction, context);
    return result;
}

function mapIntoWithKeyPrefixInternal(children, result, mapFunction, context) {
    const traverseContext = { result, mapFunction, context };
    traverseAllChildren(children, mapSingleChildIntoContext, traverseContext);
}

function traverseAllChildren(children, mapSingleChildIntoContext, traverseContext) {
    const type = typeof children;
    if (type === 'string' || type === 'number' || (type === 'object' && children.$typeof === REACT_ELEMENT_TYPE)) {
        mapSingleChildIntoContext(
            traverseContext,
            children
        );
    }
}
function mapSingleChildIntoContext(traverseContext, child) {
    const { result, mapFunction, context } = traverseContext;
    let mappedChild = mapFunction.call(context, child);
    result.push(mappedChild);
}

export {
    mapChildren as map,
};
```

3.2 实现对数组的映射

3.2.1 src\index.js

src\index.js

```
import React, { Component } from './react';
import ReactDOM from 'react-dom';
class Child extends Component {
  render() {
    console.log(this.props.children);
    const mappedChildren = React.Children.map(
      this.props.children,
      (item, index) => (
        {item}
      )
    );
    console.log(mappedChildren);
    return (

        {mappedChildren}

    )
  }
}
class App extends Component {
  render() {
    return (
+      AB
    )
  }
}
ReactDOM.render(, document.getElementById('root'));
```

3.2.2 src\react\ReactChildren.js

src\react\ReactChildren.js

```
import { REACT_ELEMENT_TYPE } from '../shared/ReactSymbols';

function mapChildren(children, mapFunction, context) {
    const result = [];
    mapIntoWithKeyPrefixInternal(children, result, mapFunction, context);
    return result;
}

function mapIntoWithKeyPrefixInternal(children, result, mapFunction, context) {
    const traverseContext = { result, mapFunction, context };
    traverseAllChildren(children, mapSingleChildIntoContext, traverseContext);
}

function traverseAllChildren(children, mapSingleChildIntoContext, traverseContext) {
    const type = typeof children;
    if (type
        mapSingleChildIntoContext(
            traverseContext,
            children
        );
    }
+    if (Array.isArray(children)) {
+        for (let i = 0; i < children.length; i++) {
+            traverseAllChildren(children[i], mapSingleChildIntoContext, traverseContext);
+        }
+    }
}
function mapSingleChildIntoContext(traverseContext, child) {
    const { result, mapFunction, context } = traverseContext;
    let mappedChild = mapFunction.call(context, child);
    result.push(mappedChild);
}

export {
    mapChildren as map,
};
```

3.3 实现映射为数组

3.3.1 src\index.js

src\index.js

```
import React, { Component } from './react';
import ReactDOM from 'react-dom';
class Child extends Component {
  render() {
+    console.log(this.props.children);
+    const mappedChildren = React.Children.map(
+      this.props.children,
+      (item) => (
+        [item, [item, [item, item]]]
+      )
+    );
+    console.log(mappedChildren);
    return (

        {mappedChildren}

    )
  }
}
class App extends Component {
  render() {
    return (

        A
        B

    )
  }
}
ReactDOM.render(, document.getElementById('root'));
```

3.3.2 src\react\ReactChildren.js

src\react\ReactChildren.js

```
import { REACT_ELEMENT_TYPE } from '../shared/ReactSymbols';

function mapChildren(children, mapFunction, context) {
    const result = [];
    mapIntoWithKeyPrefixInternal(children, result, mapFunction, context);
    return result;
}

function mapIntoWithKeyPrefixInternal(children, result, mapFunction, context) {
    const traverseContext = { result, mapFunction, context };
    traverseAllChildren(children, mapSingleChildIntoContext, traverseContext);
}

function traverseAllChildren(children, mapSingleChildIntoContext, traverseContext) {
    const type = typeof children;
    if (type
        mapSingleChildIntoContext(
            traverseContext,
            children
        );
    }
    if (Array.isArray(children)) {
        for (let i = 0; i < children.length; i++) {
            traverseAllChildren(children[i], mapSingleChildIntoContext, traverseContext);
        }
    }
}
function mapSingleChildIntoContext(traverseContext, child) {
    const { result, mapFunction, context } = traverseContext;
    let mappedChild = mapFunction.call(context, child);
+    if (Array.isArray(mappedChild)) {
+        mapIntoWithKeyPrefixInternal(mappedChild, result, c => c);
+    } else if (mappedChild != null) {
+        result.push(mappedChild);
+    }
}

export {
    mapChildren as map,
};
```

3.4. 映射为数组

3.4.1 src\index.js

src\index.js

```
import React, { Component } from './react';
import ReactDOM from 'react-dom';
class Child extends Component {
  render() {
    console.log(this.props.children);
    const mappedChildren = React.Children.map(
      this.props.children,
      (item) => (
+        [item, [item, [item, item]]]
      )
    );
    console.log(mappedChildren);
    return (

        {mappedChildren}

    )
  }
}
class App extends Component {
  render() {
    return (

+        {[A, B]}
+        {[C, D]}

    )
  }
}
ReactDOM.render(, document.getElementById('root'));
```

3.4.2 src\react\ReactChildren.js

src\react\ReactChildren.js

```
import { REACT_ELEMENT_TYPE } from '../shared/ReactSymbols';
+const SEPARATOR = '.';
+const SUBSEPARATOR = ':';
function mapChildren(children, mapFunction, context) {
    const result = [];
+    mapIntoWithKeyPrefixInternal(children, result, null, mapFunction, context);
    return result;
}

function mapIntoWithKeyPrefixInternal(children, result, prefix, mapFunction, context) {
+    if (prefix != null) {
+        prefix = prefix + '/';
+    }
+    const traverseContext = { result, prefix, mapFunction, context };
+    traverseAllChildren(children, '', mapSingleChildIntoContext, traverseContext);
}

+function traverseAllChildren(children, nameSoFar, mapSingleChildIntoContext, traverseContext) {
    const type = typeof children;
    if (type
+        mapSingleChildIntoContext(
+            traverseContext,
+            children,
+            nameSoFar === '' ? SEPARATOR + getComponentKey(children, 0) : nameSoFar
+        );
    }
    if (Array.isArray(children)) {
+        let child;
+        let nextName;
+        const nextNamePrefix = nameSoFar === '' ? SEPARATOR : nameSoFar + SUBSEPARATOR;
        for (let i = 0; i < children.length; i++) {
+            child = children[i];
+            nextName = nextNamePrefix + getComponentKey(child, i);
+            traverseAllChildren(child, nextName, mapSingleChildIntoContext, traverseContext);
        }
    }
}
+function mapSingleChildIntoContext(traverseContext, child, childKey) {
+    const { result, prefix, mapFunction, context } = traverseContext;
    let mappedChild = mapFunction.call(context, child);
    if (Array.isArray(mappedChild)) {
+        mapIntoWithKeyPrefixInternal(mappedChild, result, childKey, c => c);
    } else if (mappedChild != null) {
+        result.push({ ...mappedChild, key: prefix + childKey });
    }
}
+function getComponentKey(component, index) {
+    return component.key || index.toString(36);
+}
export {
    mapChildren as map
};
```