2-6 React 中的组件，如函数组件、类组件等

1. React DOM 组件

https://zh-hans.react.dev/reference/react-dom/components

有对应 DOM 节点的组件，比如 div、input 等。

2. 类组件

React 目前推荐函数组件：比如示例里，现在都是用的函数组件 https://zh-hans.react.dev/learn/your-first-component。

类组件已经归类到了 Legacy React APIs 中。

React JSX

```
class Greeting extends Component {
  render() {
    return <h1>Hello, {this.props.name}!</h1>;
  }
}
```

3. 函数组件

React JSX

```jsx
function Profile() {
  return (
    <img
      src="https://i.imgur.com/MK3eW3As.jpg"
      alt="Katherine Johnson"
    />
  );
}
```

4. React 内置组件

- `<Fragment>`，也可以写作 `<>...</>`，让你可以将多个 JSX 节点组合在一起。
- `<Profiler>` 让你可以以编程方式衡量 React 树的渲染性能。
- `<Suspense>` 让你可以在子组件加载时显示后备方案。
- `<StrictMode>` 可以开启一些额外的，仅用于开发环境的检测，帮助你更早地发现 bug。

5. `SomeContext.Provider/SomeContext.Consumer`

使用 Context 的时候，分别使用 `Provider` 和 `Consumer` 来分发和消费 context value。

6. 函数：forwardRef/memo/lazy/createPortal

forwardRef

`forwardRef` 允许组件使用 ref 将 DOM 节点暴露给父组件。

```javascript
const SomeComponent = forwardRef(render)
```

`forwardRef` 返回一个可以在 JSX 中渲染的 React 组件。与作为纯函数定义的 React 组件不同，`forwardRef` 返回的组件还能够接收 `ref` 属性。

lazy

`lazy` 能够让你在组件第一次被渲染之前延迟加载组件的代码。

```javascript
const SomeComponent = lazy(load)
```

`lazy` 返回一个 React 组件，你可以在 fiber 树中渲染。当懒加载组件的代码仍在加载时，尝试渲染它将会处于 *暂停* 状态。使用 `<Suspense>` 可以在其加载时显示一个正在加载的提示。

如：

```javascript
import { lazy } from 'react';

const MarkdownPreview = lazy(() => import('./MarkdownPreview.js'));
```

memo

HOC，higher order component 高阶组件。是个函数，接收组件作为参数，返回一个新的组件。

`memo` 允许你的组件在 props 没有改变的情况下跳过重新渲染。

```JavaScript
const MemoizedComponent = memo(SomeComponent, arePropsEqual?)
```

createPortal

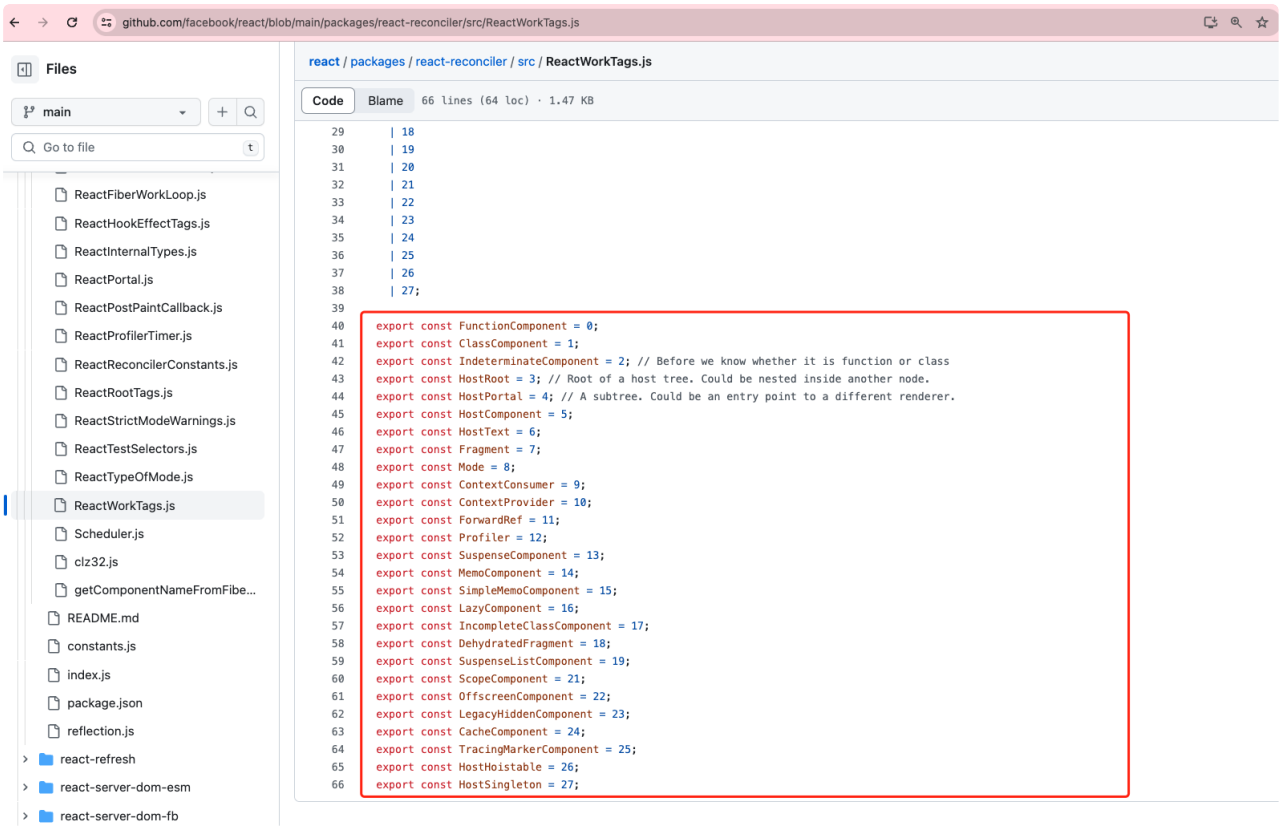
`createPortal` 允许你将 JSX 作为 children 渲染至 DOM 的不同部分。

```React JSX
<div>
  <SomeComponent />
  {createPortal(children, domNode, key?)}
</div>
```

`createPortal` 返回一个可以包含在 JSX 中或从 React 组件中返回的 React 节点。如果 React 在渲染输出中遇见它，它将把提供的 `children` 放入提供的 `domNode` 中。

React 源码中对组件的区分

图片源码地址：https://github.com/facebook/react/blob/main/packages/react-reconciler/src/ReactWorkTags.js


github.com/facebook/react/blob/main/packages/react-reconciler/src/ReactWorkTags.js
Files
main
Go to file
ReactFiberWorkLoop.js
ReactHookEffectTags.js
ReactInternalTypes.js
ReactPortal.js
ReactPostPaintCallback.js
ReactProfilerTimer.js
ReactReconcilerConstants.js
ReactRootTags.js
ReactStrictModeWarnings.js
ReactTestSelectors.js
ReactTypeOfMode.js
ReactWorkTags.js
Scheduler.js
clz32.js
getComponentNameFromFibe...
README.md
constants.js
index.js
package.json
reflection.js
react-refresh
react-server-dom-esm
react-server-dom-fb
react / packages / react-reconciler / src / ReactWorkTags.js
Code
Blame
66 lines (64 loc) · 1.47 KB
| 18
| 19
| 20
| 21
| 22
| 23
| 24
| 25
| 26
| 27;
export const FunctionComponent = 0;
export const ClassComponent = 1;
export const IndeterminateComponent = 2; // Before we know whether it is function or class
export const HostRoot = 3; // Root of a host tree. Could be nested inside another node.
export const HostPortal = 4; // A subtree. Could be an entry point to a different renderer.
export const HostComponent = 5;
export const HostText = 6;
export const Fragment = 7;
export const Mode = 8;
export const ContextConsumer = 9;
export const ContextProvider = 10;
export const ForwardRef = 11;
export const Profiler = 12;
export const SuspenseComponent = 13;
export const MemoComponent = 14;
export const SimpleMemoComponent = 15;
export const LazyComponent = 16;
export const IncompleteClassComponent = 17;
export const DehydratedFragment = 18;
export const SuspenseListComponent = 19;
export const ScopeComponent = 21;
export const OffscreenComponent = 22;
export const LegacyHiddenComponent = 23;
export const CacheComponent = 24;
export const TracingMarkerComponent = 25;
export const HostHoistable = 26;
export const HostSingleton = 27;