ATOM + Single-SPA

调用链

主应用 atomfe/wheeljack

wheeljack.init();

wheeljack.layout(layout)

wheeljack.start –> ihome/trasformer$$start –> **os-single-spa$$startSpa(){{0}}** + render()

>> render

ReactDom.render(RootComponent, DOM)

RootComponent –> App –> ReactRouter –> Route(/)–> Layout

>> Layout

getLayoutRoutes –> getSubApplicationRoutes+ATOM-POLYMER$$components –> getSubApplicationRoute –> **Route + SubApplication{{子应用 url 匹配}}**

>> SubApplication

MicroApp+ATOM-POLYMER$$component

>> MicroApp

render –> getSubApplication –> render –> Application

>> render –> ReactElement(class="atom-react-app") –> **ref-node**

>> componentDidmount

this.app = createMicroApp –> createApplication –> createAppInstance –> new Application

>> Application {load, mount, update, unmount}

>> load

createAppLoader –>

appInstance = extractModule–> loadBundle[] –> fetch –> eval

–> 子应用{bootstrap, mount, unmount, update}+reactLifecycles

remoteApp –> **{appInstance, bootstrap,mount,unmount,update}**

+ 子应用{bootstrap, mount, unmount, update}+reactLifecycles{{3}}

>> **mount{{2}}**

parcel = os–single–spa$$mountRootParcel(

{bootstrap,mount,unmount,update}, {domElement: **ref–node**}

) +remoteApp$${bootstrap,mount,unmount,update} –> mountPromise

>> update

parcel.update

>> unmount

parcel.unmount

load –> load(this.app) –> this.app.load()

mount{{1}} –> mount(this.app) –> this.app.mount()

>> componentWillUnmount

unmount–> unmount(this.app) –> this.app.unmount()

>> componentDidUpdate

update –> update(this.app) –> this.app.update

子应用 packages/react–portal/src/mount.tsx

entry –>

single-spa-react -> reactLifecycles -> {bootstrap, mount, unmount, update}+reactLifecycles

说明

ATOM 采用 Parcel 模式

https://juejin.cn/post/6955342063235760164#5-6

我们对 **application** 和 **parcel** 模式，做一个简单的对比

标题	application	parcel
路由控制	有	无
UI 渲染	有	有
生命周期方法	single-spa 管理	用户自己管理
应用场景	多个子应用聚合	跨框架使用组件

调用链串联

1. Route + SubApplication{{子应用 url 匹配}}
2. MicroApp ---> Application -> render 生成一个节点，通过 ref 记录下来
3. MicroApp ---> Application -> componentDidMount
    a. load -> loadBundle -> 子应用代码执行，返回 single-spa-react 生命周期 **{{3}}**
        - single-spa-react 返回的 reactLifecycles.mount 里的实现，应该是 ReactDom(ReactElment, DOM) https://github.com/single-spa/single-spa-react/blob/main/src/single-spa-react.js
    b. mount**{{1}} {{2}}** -> 将输入的声明周期 **{{3}}**，运行了一遍
        - ref: https://juejin.cn/post/6857783003646623751]