迁移现有SPAs项目

Migrating existing SPAs

If you're interested in migrating existing SPAs into a single-spa, you'll need to do three things:

1. Create a [single spa config](https://single-spa.js.org/docs/configuration)
2. [Convert your SPA or SPAs to be registered applications](https://single-spa.js.org/docs/migrating-existing-spas#converting-spas-into-registered-applications)
3. Adjust your html file so that your single spa config is the new boss in town. See [docs](https://single-spa.js.org/docs/configuration#indexhtml-file).

如果你想将现有的 SPA 应用迁移到 single-spa 框架中，需要做三件事：

1. 创建一个[single spa 配置](https://zh-hans.single-spa.js.org/docs/configuration)
2. [将 SPA 应用转换为注册应用](https://zh-hans.single-spa.js.org/docs/migrating-existing-spas#converting-spas-into-registered-applications)
3. 调整 html 文件，使 single-spa 配置生效，查看[文档](https://zh-hans.single-spa.js.org/docs/configuration#indexhtml-file)

##### 将 SPA 应用转换为注册应用

[#](https://single-spa.js.org/docs/migrating-existing-spas#converting-spas-into-registered-applications)Converting SPAs into registered applications

Your existing SPAs, whether they be Angular, React, or something else, probably are not used to unmounting themselves from the DOM. Also, they probably have had the luxury of controlling the entire html page themselves, instead of being purely javascript applications that don't have sole control over `<script>` tags and `<link>`  tags. So in order to convert them into single-spa registered applications, they will need to overcome those obstacles while implementing lifecycle functions.

你现有的 SPA 应用，无论是基于 Angular，React 还是其他技术，可能都无法从 DOM 中将自身卸载。而且，可能都是控制整个 html 页面，而不是一个不对 `<script>` 标签和

`<link>`标签进行唯一控制的单纯 JavaScript 应用程序。因此，为了将它们转换为 single-spa 注册应用，需要在实现其生命周期功能时处理掉这些问题。

##### (1) 实现生命周期

[#](https://single-spa.js.org/docs/migrating-existing-spas#1-implementing-lifecycle-functions)(1) Implementing lifecycle functions

See the [registered application lifecycle](https://single-spa.js.org/docs/building-applications#registered-application-lifecycle) docs to see what you need to do. The hardest part will almost certainly be the `unmount`  lifecycle, since most SPAs aren't accustomed to going dormant and unmounting themselves from the DOM. When implementing your lifecycle functions, first check out the [ecosystem](https://single-spa.js.org/docs/ecosystem) docs before reinventing the wheel yourself. If that doesn't have everything you need, you'll have to make sure that your SPA can clean up its DOM, DOM event listeners (all of them, but especially hashchange and popstate), and memory.

请参阅[注册应用的生命周期文档](https://zh-hans.single-spa.js.org/docs/building-applications#registered-application-lifecycle)，以了解需要做什么。最难的部分可能就是 `unmount`

这个生命周期，因为大多数 SPA 都不会休眠并从 DOM 卸载自身。在实现生命周期功能时，请先查看[生态系统文档](https://zh-hans.single-spa.js.org/docs/ecosystem)。如果文档中没有列出你所使用的技术栈，则必须确保在你自己实现的 `unmount`中，能够清理其 DOM 节点，DOM 事件侦听（所有的事件侦听，尤其是 `hashchange` 和 `popstate`）以及释放内存。

##### (2) 使 CSS、字体和 `<script>`等依赖生效

[#](https://single-spa.js.org/docs/migrating-existing-spas#2-getting-the-css-fonts-script-dependencies-to-work)(2) Getting the CSS, fonts, `<script>` dependencies to work

Since existing SPAs are used to having an index.html file for their css, fonts, third party script-tags, etc., it's likely that you'll have to do some work to make sure all of those keep on working when your SPA becomes an html-less [application](https://single-spa.js.org/docs/building-applications). It is best to try to put all that you can into the javascript bundle, but your escape hatch is to put the things you need into your [single spa config](https://single-spa.js.org/docs/configuration).

由于现有的 SPA 应用中， CSS、字体、第三方脚本等依赖都在 index.html 文件中，因此你可能必须做一些工作，以确保当你的 SPA 应用变成了[无 HTML 应用程序](https://zh-hans.single-spa.js.org/docs/building-applications)之后，这些依赖可以正常工作。最好是将所有的资源都打包到 js 文件中，但是你也可以将所需要的资源列在[single spa 配置](https://zh-hans.single-spa.js.org/docs/configuration)中。