
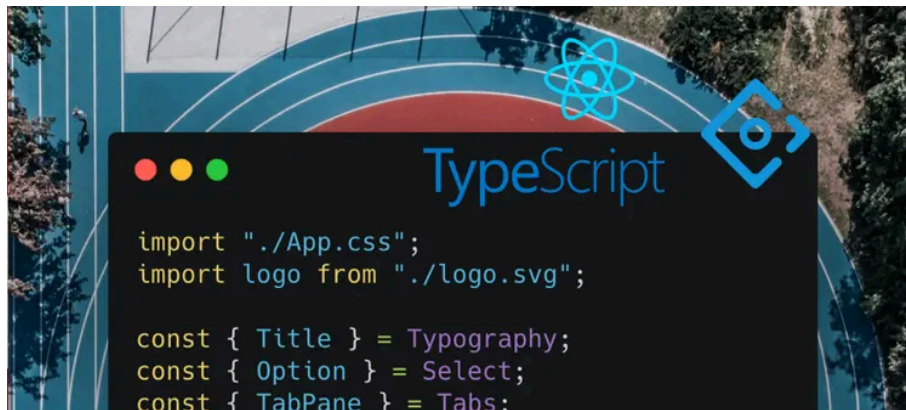


类型即正义：TypeScript 从入门到实践（序章）

 一只图雀 2020-04-06 阅读 9 分钟



作者：一只图雀

仓库：[Github](#)、[Gitee](#)

图雀社区主站（首发）：[图雀社区](#)

博客：[掘金](#)、[知乎](#)、[慕课](#)

公众号：[图雀社区](#)

联系我：关注公众号后可以加图雀酱微信哦

原创不易，❤️ 点赞+评论+收藏 ❤️ 三连，鼓励作者写出更好的教程

准备代码

因为需要尽可能全且精炼的讲解 TypeScript 语法知识，所以我们需要一个恰到好处的实战项目，这一小节主要是用于讲解我们准备初始 TypeScript 版本的 React 项目代码的过程，在下一个小节中我们将会结合 React 项目代码，真正开始 TypeScript 语法的讲解。

本文所涉及的源代码都放在了 [Github](#) 或者 [Gitee](#) 上, 如果您觉得我们写得还不错, 希望您能给 ❤️ [这篇文章点赞Github 或 Gitee 仓库加星](#) ❤️ 哦~

此教程属于 [React 前端工程师学习路线](#) 的一部分, 欢迎来 Star 一波, 鼓励我们继续创作出更好的教程, 持续更新中~

前提条件

1. 确保你已经安装了 Node.js, 可以访问官网安装: [官网地址](#)。
2. 确保你已经了解基本的 React 开发知识, 图雀社区有一篇很好的 [React 入门教程](#), 你可以通过学习它很快的上手 React。
3. 确保你有一定的命令行使用基础, 包括使用 Npm (Node.js 包管理工具) 来安装包。

初始化应用

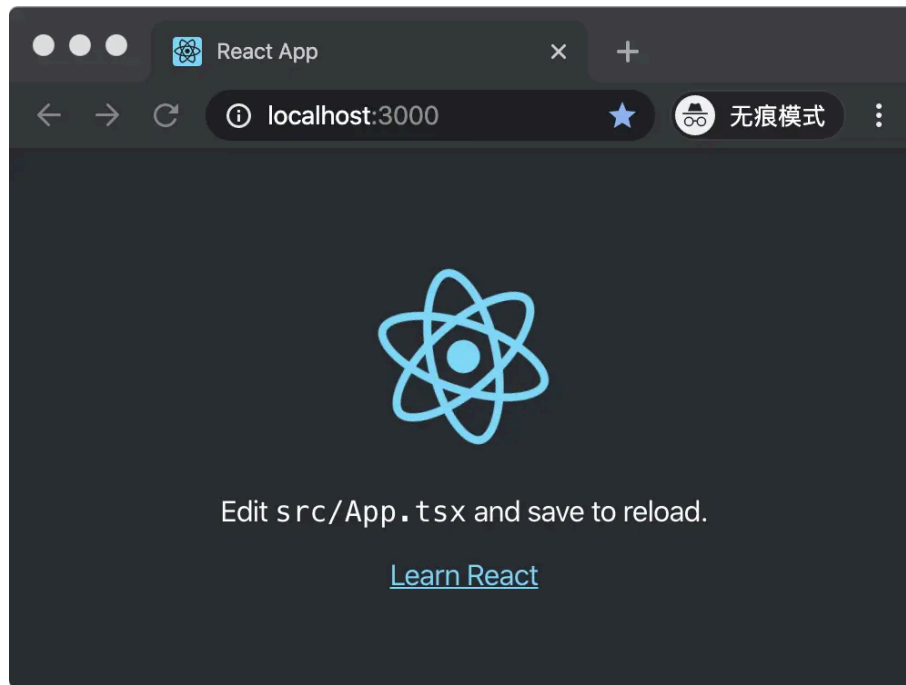
初始一个 React 应用的最佳方式那么一定是 React 官方维护的 [Create React App](#) 脚手架了, 我们打开终端, 运行如下命令来初始化一个 TypeScript 版本的 React 应用:

```
$ npx create-react-app typescript-tea --template typescript
```

运行如上命令, 命令行里面应该会有一系列输出, 等待几分钟, 就会提示已经初始化完成, 并提供了对于的命令来帮助你开启项目, 我们根据提示输入如下命令来开启项目:

```
$ cd typescript-tea  
$ npm start
```

运行如上命令之后, 会自动开启 Webpack 开发服务器, 并打开浏览器窗口, 访问 <http://localhost:3000/> 来展示你的应用初始界面:



如果看到这个界面，恭喜你 🎉！成功创建一个 TypeScript 版本的 React 应用！

提示

在下文中，为了简化语言，我们统一称 TypeScript 为 TS。

引入 antd 组件库

实战驱动的技术学习能带给我们成就感，便捷好用的包可以加快我们的开发效率，好看的界面可以提高我们的审美能力，缓解学习疲劳。在这篇教程的讲解过程中，我们将通过 **Ant Design** 对应的 React 组件库 **antd** 来辅助我们项目的编写，使得我们可以专注于讲解 TS 的核心知识，而不被繁杂的界面语言所干扰，还能做出对应相应完成的目标功能。

提示

Ant Design 是蚂蚁金服孵化的一套企业级产品设计体系，提供了完备的 TS 类型定义，使得我们可以很方便的在 TS 项目中使用，在最近发布了 4.0 版本，致力于创造高效愉悦的工作体验。

除此之外 Ant Design 的周边生态也很丰富：

- 包括新一代数据可视化解决方案：**AntV**
- 一个基于 Preact / React / React Native 的 UI 组件库：**Ant Design Mobile**
- 开箱即用的中台前端/设计解决方案：**Ant Design Pro**

- 插画设计: [海兔](#)
- 一款为设计者提升工作效率的 Sketch 工具集: [Kitchen](#)

后面图雀社区计划围绕 Ant Design 生态撰写一系列教程, 帮助大家提高设计、开发效率, 敬请期待! 🙌

安装依赖

好了, 大致介绍了 antd 组件库及 Ant Design 周边之后, 我们马上来写代码引入 antd, 打开命令行, 在其中输入如下命令:

```
$ npm install antd
```

运行上面的命令安装完依赖之后就可以在项目中使用了, 但是为了更好的定制样式和按需引用以减小打包之后的包体积, 我们还需要做一点定制化的操作, 打开命令行, 依次安装如下依赖:

```
$ npm install react-app-rewired customize-cra babel-plugin-import
```



注意到上面我们安装了很多包, 我们来依次解释一下上面各种包的意思:

- react-app-rewired: 用来定制化 Create React App (CRA)脚手架的一些配置, 比如 Webpack、Babel 等, 因为 CRA 它是一个封闭的黑盒, 不允许开发者直接定制, 但有时候我们需要对配置做一些修改, 比如这里需要配置 antd 的按需引用。
- customize-cra: 是 CRA 在发布 2.0 之后出来的一个辅助 react-app-rewired 更方便定制 CRA 的 Webpack 配置的一个库, 它提供了一些开箱即用的 API。
- babel-plugin-import: 是配置可供开发者按需引用 antd 组件的一个 Babel 插件
- less 和 less-loader: 是我们用于定制化 antd 的主题需要的 Webpack loader, 因为 antd 使用 less 作为样式化语言。

最后我们安装一个在 Ant Design 4.0 拆分出去的 icons 包, 可以用来按需引用 icons, 进一步减少最后的打包体积, 继续在命令行运行如下命令:

```
$ npm install @ant-design/icons
```

大功告成！现在我们所有的依赖以及安装完成。接下来就需要改写一下 CRA 之前通过 `react-scripts` 跑开发构建的流程，用我们安装的 `react-app-rewired` 脚本来替换它，当安装完了所以依赖，以及用 `react-app-rewired` 替换 `react-scripts` 之后，我们的 `package.json` 文件应该是下面的样子：

```
// ...
"version": "0.1.0",
"private": true,
"dependencies": {
  "@ant-design/icons": "^4.0.2",
  "@testing-library/jest-dom": "^4.2.4",
  "@testing-library/react": "^9.3.2",
  "@testing-library/user-event": "^7.1.2",
  "@types/jest": "^24.0.0",
  "@types/node": "^12.0.0",
  "@types/react": "^16.9.0",
  "@types/react-dom": "^16.9.0",
  "antd": "^4.0.0",
  "babel-plugin-import": "^1.13.0",
  "customize-cra": "^0.9.1",
  "less": "^3.11.1",
  "less-loader": "^5.0.0",
  "react": "^16.13.0",
  "react-app-rewired": "^2.1.5",
  "react-dom": "^16.13.0",
  "react-scripts": "3.4.0",
  "typescript": "~3.7.2"
},
"scripts": {
  "start": "react-app-rewired start",
  "build": "react-app-rewired build",
```

修改配置

安装完依赖之后，我们要确保对应改写 CRA 流程的配置生效，我们需要根据 `react-app-rewired` 的文档说明在根目录下建立 `config-overrides.js` 文件，并在其中编写如下的内容：

```
const { override, fixBabelImports, addLessLoader } = require("customize-cra");
const darkThemeVars = require("antd/dist/dark-theme");

module.exports = override(
  fixBabelImports("import", {
    libraryName: "antd",
    libraryDirectory: "es",
```

```

    style: true
  )),
  addLessLoader({
    javascriptEnabled: true,
    modifyVars: {
      hack: `true;@import "${require.resolve(
        "antd/lib/style/color/colorPalette.less"
      )}";`,
      ...darkThemeVars,
      "@primary-color": "#02b875"
    }
  })
);

```

可以看到，上面的代码主要是导出一个用于修改 Webpack 配置的对象，使用 `override` API，接收两个修改配置的函数调用，`fixBabelImports` 用于配置 antd 的按需引用，`addLessLoader` 用于配置 antd 的主题，这里我们使用了 Ant Design 4.0 新带来的 Dark Mode（暗色模式），然后配置了主题色为图雀社区的主题色：`#02b875`，代表希望的绿色。😄

自此，我们就引入了 antd 组件库，并进行了按需配置使用以及配置主题色和使用 Ant Design 最新的暗色主题 -- Dark Mode。

编写初始代码

准备逻辑部分

接下来，我们将使用 antd 帮助我们快速的编写一下我们即将实现的待办事项的界面，打开 `src/App.tsx`，对其中的代码做出对应的修改如下：

```

import React, { useState, useRef } from "react";
import {
  List,
  Avatar,
  Button,
  Typography,
  Form,
  Input,
  Select,
  DatePicker,
  Menu,
  Dropdown,
  Tabs

```

```
} from "antd";
import { DownOutlined } from "@ant-design/icons";

import "./App.css";
import logo from "./logo.svg";

const { Title } = Typography;
const { Option } = Select;
const { TabPane } = Tabs;

const todoListData = [
  {
```

上面的代码主要就是一系列初始数据的准备, antd 组件的使用, 编写起来的大致轮廓, 还没有涉及到任何的 TS 语法, 但这个是我们开始项目的基础, 读者只需要进行简单的复制放进现有的 `typescript-tea` 项目中对应的 `src/App.tsx` 中即可。

准备样式部分

准备了逻辑代码之后, 为了让我们最后的待办事项在样式上更美观一点, 也到了我们介绍时的操作, 我们需要给项目加上样式, 打开

`src/App.css` 对其中的代码做出对应的修改如下:

```
.App {
  display: flex;
  flex-direction: column;
  align-items: center;
  padding-top: 60px;
}

.container {
  width: 600px;
}

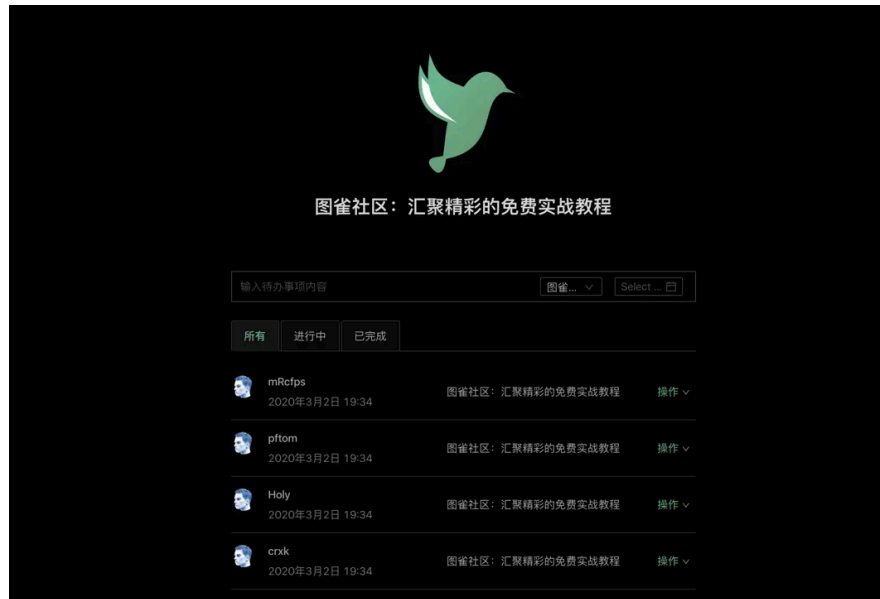
.header {
  text-align: center;
  margin-bottom: 56px;
}

.header img {
  width: 160px;
  height: 160px;
  margin-bottom: 24px;
}

.todoInput {
```

```
display: flex;  
flex-direction: row;  
align-items: center;
```

这个时候如果你的服务器还在运行，那么你应该可以看到如下效果：



好了！所有的准备工作已经就绪，在开始下一节真正的 TS 学习之前，我们先来回顾一下我们在这个小节中所完成的工作：

- 使用 CRA 的 TypeScript 脚本初始化了一个 TS 版的 React 项目
- 安装了 antd 组件库，并使用 `react-app-rewired` 替换默认的 `react-scripts` 来完成对 CRA 的 Webpack 配置进行修改，以是我们可以获得 antd 组件的按需引用和主题定制的功能
- 准备了初始待办事项代码的逻辑部分和样式部分

我们在前面铺垫了大量的 TypeScript 的优点以及花了不少笔墨来准备初始代码，想必读到这里的读者们可能已经等不及要马上见识一下 TS 的庐山真面目了吧！马上就来啦！

想要学习更多精彩的实战技术教程？来[图雀社区](#)逛逛吧。

本文所涉及的源代码都放在了 [Github](#) 或者 [Gitee](#) 上，如果您觉得我们写得还不错，希望您能给 [这篇文章](#) 点赞 [Github](#) 或 [Gitee](#) 仓库加星

❤ 哦~



图雀社区

[注册登录](#)[主页](#) [关于](#) [RSS](#)[typescript](#) [react.js](#) [antd](#)[👍 赞 2](#)[🔖 收藏 2](#)[🔗 分享](#)

阅读 1.6k • 发布于 2020-04-06



一只图雀

863 声望 • 1.2k 粉丝

我们图雀社区是一个供大家分享用 Tuture 写作工具撰写教程的一个平台。在这里，读者们可以尽情享受高质量的实战教...

[关注作者](#)[« 上一篇](#)[下一篇 »](#)[全栈“食”代：Django + Nuxt 实现...](#) [类型即正义：TypeScript 从入门到...](#)

引用和评论

被 2 篇内容引用[类型即正义：TypeScript 从入门到实践（三）：类型别名和类](#)[💬 2](#)[类型即正义：TypeScript 从入门到实践（二）](#)

推荐阅读

**Taro 小程序开发大型实战 (九)：使用 Authing 打造具有微信登录的企业级用户系统**

一只图雀 · 赞 1 · 阅读 4.4k

**文件导出**

热饭班长 · 赞 8 · 阅读 3k

**TS-react: react中常用的类型整理**

wxp686 · 赞 1 · 阅读 7k

**vscode的jsconfig.json和tsconfig.js**

热饭班长 · 赞 3 · 阅读 4.5k

**【从前端入门到全栈】Node.js之大文件分片上传**

野生程序猿江辰 · 赞 3 · 阅读 312 · 评论 1

**react组件解耦**

热饭班长 · 赞 3 · 阅读 4k

**react 踩坑**

assassin_cike · 赞 1 · 阅读 3.5k

1 条评论

得票

最新



撰写评论 ...



提交评论


评论支持部分 Markdown 语法: ****粗体**** *_斜体_* [链接]
(<http://example.com>) `代码` - 列表 > 引用。你还可以使用 @
来通知其他用户。

**指尖泛出的繁华：牛逼**

👍 · 回复 · 2020-04-06

©2024 图雀社区

除特别声明外，作品采用《署名-非商业性使用-禁止演绎 4.0 国际》进行许可

 使用 SegmentFault 发布

SegmentFault - 凝聚集体智慧，推动技术进步

服务协议 · 隐私政策 · 浙ICP备15005796号-2 · 浙公网安备33010602002000号