【Next.js】專案結構與檔案說明

以下是根據你提供的 Next.js 專案結構，整理出各資料夾與檔案的用途與說明：

📁 專案結構與說明表格

路徑 / 檔案	類型	說明
`.next/`	資料夾	Next.js **build 輸出資料夾**，會在 `next build` 時產生。內含編譯後的 server 與靜態資源。可部署至伺服器。
├── `server/`	資料夾	Server-side compiled files（如 SSR 頁面、API Routes）。
├── `static/`	資料夾	靜態產出資源（例如圖片、CSS、JS 等）。
├── `cache/`	資料夾	編譯過程中的 cache，提高二次編譯速度。
├── `build-manifest.json`	JSON 檔案	描述每個頁面對應的 JS chunk，供 runtime 載入使用。
├── `routes-manifest.json`	JSON 檔案	儲存頁面與路由資訊的映射表。
├── `prerender-manifest.json`	JSON 檔案	與預先產生的頁面（SSG）有關，例如 `getStaticProps` 結果。
`node_modules/`	資料夾	安裝的 npm 套件。
`public/`	資料夾	放置**靜態資源**（如圖片、SVG），透過 `/` 路徑對外公開，例如：`public/globe.svg` → `/globe.svg`。
`src/app/`	資料夾	使用 **App Router 架構** 的主要入口，代表新架構（取代 `pages/` 目錄）。
├── `layout.tsx`	React 元件	每個 route 的共同 Layout 元件。App Router 中不可或缺。
├── `page.tsx`	React 元件	對應 `/` 路由的頁面內容（Home）。
├── `globals.css`	CSS 檔案	全域 CSS，需在 `layout.tsx` 中引入。
`favicon.ico`	圖示檔案	網站 favicon，會自動用在 `<head>` 中。
`.gitignore`	設定檔	Git 忽略檔案清單。通常會包含 `.next/`、`node_modules/`。
`next-env.d.ts`	TS 宣告	Next.js 產生的 type 設定檔，自動加入必要的 TypeScript 型別。
`next.config.ts`	設定檔	Next.js 專案設定檔，支援自訂建置、路徑、Plugin 等功能。
`package.json` / `package-lock.json`	設定檔	npm 套件定義與鎖定檔。記錄所有依賴與版本。
`postcss.config.mjs`	設定檔	PostCSS 設定，通常與 TailwindCSS 搭配使用。
`README.md`	說明檔	專案說明、安裝方式、開發資訊等。
`tsconfig.json`	設定檔	TypeScript 專案設定，例如路徑 alias、編譯選項等。

🔍 重點說明

分類	重點
重要目錄	`src/app/`（使用 App Router 架構）、`public/`（靜態資源）、`.next/`（build 輸出）
重要設定檔	`next.config.ts`、`tsconfig.json`、`postcss.config.mjs`
App Router 必備	`layout.tsx` 是必要頁面組件，類似傳統 `_app.tsx + _document.tsx`
部署時可忽略	`node_modules/`、`.next/cache`、`.gitignore`

說明layout.tsx 與 page.tsx 關係

`layout.tsx` 與 `page.tsx` 是 Next.js 13+（App Router 架構）中，兩個**核心檔案**，分別扮演不同但互補的角色。

簡單說明

檔案	用途	載入頻率	比喻
`layout.tsx`	定義頁面結構與共用 UI（像是外框）	只載入一次（不會重複重建）	「建築物的骨架」
`page.tsx`	定義實際頁面內容	每個頁面都會重新載入	「房間的內容、家具、住戶」

目錄架構範例

```
app/
├── layout.tsx        # 所有頁面的共用 Layout
├── page.tsx          # / 路徑的首頁內容
├── about/
│   ├── layout.tsx    # 僅 /about 使用的 layout
│   └── page.tsx      # /about 的內容
```

關係與行為

1. **一個 layout 對應多個 page**
   - 例如 `app/layout.tsx` 包住所有頁面：`/`, `/about`, `/contact`
2. **嵌套 layout**
   - 子資料夾可有自己的 `layout.tsx`，會**包在上層 layout 裡**
3. **layout 不會因為切換 page 被卸載**
   - 可以放在 `Header`, `Sidebar`, `Footer` 等共用 UI 元件，效能更佳
4. **page.tsx 是實際的頁面入口**
   - 每個 URL 都需要一個 `page.tsx` 來當作入口點

實際範例

app/layout.tsx

```tsx
export default function RootLayout({ children }: { children: React.ReactNode }) {
  return (
    <html lang="en">
      <body>
        <header>共用導覽列</header>
        <main>{children}</main>
        <footer>共用頁尾</footer>
      </body>
    </html>
  )
}
```

app/page.tsx

```tsx
export default function HomePage() {
  return <h1>這是首頁內容</h1>
}
```

 此時 `HomePage` 內容會被注入進 `layout.tsx` 的 `{children}` 中。

□ 一圖總結

```
layout.tsx
├── Header
├── <main>
│   └── page.tsx 的內容
└── Footer
```

□ 補充：layout.tsx 特性

特性	是否支援	備註
支援 `loading.tsx`	□	當子頁面在載入時顯示 loading UI
支援 `error.tsx`	□	包裹此 layout 的錯誤畫面
支援 `metadata` 設定	□	可以定義 title、description 等
自動 caching	□	layout 有自動快取，非動態頁效能極佳

需要我幫你畫出 layout/page 的巢狀結構流程圖或建立範本專案也可以說一聲。

修訂版本 #2
★由 treeman 建立於 23 C 2025 17:53:36
由 treeman 更新於 23 C 2025 18:02:19