> Menu

App Router > ... > File Conventions > template.js

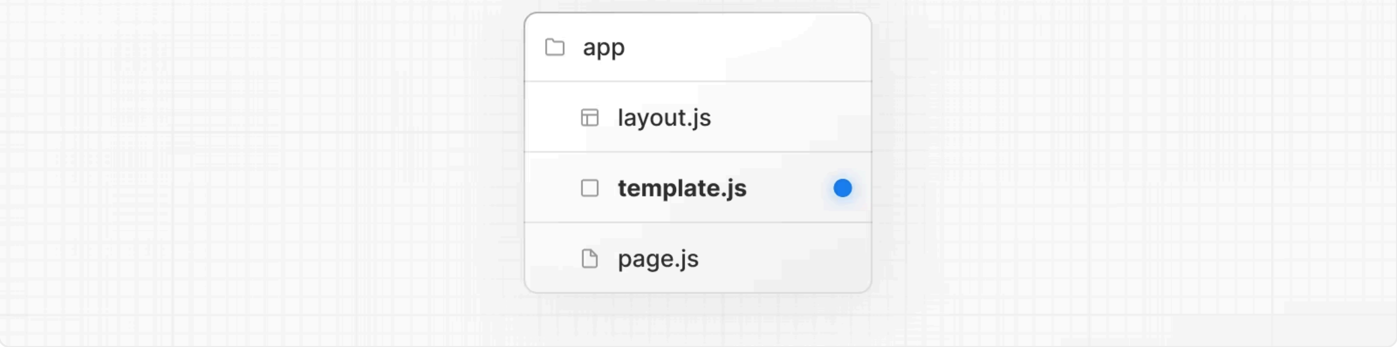
template.js

A **template** file is similar to a layout in that it wraps a layout or page. Unlike layouts that persist across routes and maintain state, templates are given a unique key, meaning children Client Components reset their state on navigation.

app/template.tsx TypeScript

```tsx
export default function Template({ children }: { children: React.ReactNode }) {
  return <div>{children}</div>
}
```

NEXT.js



While less common, you might choose to use a template over a layout if you want:

- Features that rely on `useEffect` (e.g logging page views) and `useState` (e.g a per-page feedback form).
- To change the default framework behavior. For example, Suspense Boundaries inside layouts only show the fallback the first time the Layout is loaded and not when switching

pages. For templates, the fallback is shown on each navigation.

Props

`children` (required)

Template accepts a `children` prop. For example:

Output

```
<Layout>
  {/* Note that the template is automatically given a unique key. */}
  <Template key={routeParam}>{children}</Template>
</Layout>
```

Good to know:

- By default, `template` is a Server Component, but can also be used as a Client Component through the `"use client"` directive.
- When a user navigates between routes that share a `template`, a new instance of the component is mounted, DOM elements are recreated, state is **not** preserved in Client Components, and effects are re-synchronized.

Version History

Version	Changes
`v13.0.0`	`template` introduced.

Previous
Route Segment Config

Next
Metadata Files

Was this helpful? 🤩 🙂 🙁 😭

Vercel

Resources
Docs
Learn
Showcase
Blog
Analytics
Next.js Conf
Previews

More
Next.js Commerce
Contact Sales
GitHub
Releases
Telemetry
Governance

About Vercel
Next.js + Vercel
Open Source Software
GitHub
X

Legal
Privacy Policy

Subscribe to our newsletter

Stay updated on new releases and features, guides, and case studies.

you@domain.com Subscribe