Layouts Directory

Nuxt provides a customizable layouts framework you can use throughout your application, ideal for extracting common UI or code patterns into reusable layout components.

Layouts are placed in the `layouts/` directory and will be automatically loaded via asynchronous import when used. Layouts are used by adding `<NuxtLayout>` to your `app.vue`, and either setting a `layout` property as part of your page metadata (if you are using the `~/pages` integration), or by manually specifying it as a prop to `<NuxtLayout>`. (**Note**: The layout name is normalized to kebab-case, so `someLayout` becomes `some-layout`.)

If you only have a single layout in your application, we recommend using app.vue instead.

Unlike other components, your layouts must have a single root element to allow Nuxt to apply transitions between layout changes - and this root element cannot be a `<slot />`.

Enabling the Default Layout

Add a `~/layouts/default.vue`:

layouts/default.vue

```
<template>
  <div>
    Some default layout shared across all pages
    <slot />
  </div>
</template>
```

In a layout file, the content of the layout will be loaded in the `<slot />`, rather than using a special component.

If you use a `app.vue` you will also need to add `<NuxtLayout>` :

app.vue
```
<template>
  <NuxtLayout>
    some page content
  </NuxtLayout>
</template>
```

Setting Another Layout

```
> -| layouts/
> ---| default.vue
> ---| custom.vue
```

You can directly override the default layout like this:

app.vue
```
<script setup lang="ts">
// You might choose this based on an API call or logged-in status
const layout = "custom";
</script>

<template>
  <NuxtLayout :name="layout">
    <NuxtPage />
  </NuxtLayout>
</template>
```

Alternatively, you can override the default layout per-page like this:

pages/index.vue app.vue layouts/custom.vue layouts/default.vue

pages/index.vue
```
<script>
// This will work in both `<script setup>` and `<script>`
definePageMeta({
  layout: "custom",
});
</script>
```

Learn more about defining page meta.

Changing the Layout Dynamically

You can also use a ref or computed property for your layout.

```
<script setup lang="ts">
function enableCustomLayout () {
  setPageLayout('custom')
}
definePageMeta({
  layout: false,
});
</script>

<template>
  <div>
    <button @click="enableCustomLayout">Update layout</button>
  </div>
</template>
```

Read and edit a live example in Docs > Examples > Features > Layouts.

Overriding a Layout on a Per-page Basis

If you are using the `~/pages` integration, you can take full control by setting `layout: false` and then using the `<NuxtLayout>` component within the page.

pages/index.vue layouts/custom.vue

pages/index.vue

```
<script setup lang="ts">
definePageMeta({
  layout: false,
});
</script>

<template>
```

```
  <div>
    <NuxtLayout name="custom">
      <template #header> Some header template content. </template>

      The rest of the page
    </NuxtLayout>
  </div>
</template>
```

If you use `<NuxtLayout>` within your pages, make sure it is not the root element (or disable layout/page transitions).

Edit on Github

Enterprise Design Kit NuxtLabs

Nuxt Studio