# 关于 lab（实验室）

此依赖包包含了一些还在开发中的组件，它们还不能移到 core（核心）库中。

核心库（core）和实验室（lab）的主要区别在于对组件进行版本管理的方式。 单独的 lab （实验室）库允许我们在必要时可以发布一些破坏性的更改，而 core（核心）库是遵循 较慢的发布策略。

程序员在使用和测试组件后向项目报漏洞，维护者就知道更多关于组件的缺点：如缺少功能，访问问题、漏洞，API 设计等等。 一个组建被使用的时间越久，发现一个新的问题，以及因此需要引入重大更改的可能性就越小。

对于那些准备放到核心库里的组件，需要考虑以下几点：

- 它需要**被使用过**。 MUI 团队在文档中使用 Google Analytics（在其他指标中）来评估每个组件的使用情况。 如果一个实验室组件使用率过低，那就说明这个组件并没有完成或者需求不高。
- 它的**代码质量**要和核心组件保持一致。 并不是说要做到完美，但这个组件至少要足够可靠，让开发者们可以放心地使用。
  - 每个组件必须**类型定义**。 就目前来说，一个实验室组件不需要定义类型，但是当搬到核心代码之后就需要定义好类型了。
  - 需要较高的**测试覆盖率**。 有一些实验室组件目前不带有一些综合的测试。
- 该组件是否可以作为**杠杆**来激励用户升级到最新的主要版本？ 一个社区的分裂程度越低越好。
- 它需要尽量避免在中短期内发生**破坏性更改**。 例如，如果它需要一个新的功能并因此将有可能引入重大更改，那么最好推迟将其纳入核心库的进程。

## 安装

请在您的项目目录中用以下方式安装依赖包：

```
// 用 npm 安装
npm install @mui/lab

// 用 yarn 安装
yarn add @mui/lab
```

该 lab 和那些核心组件是对等依赖的。 若您还未在项目中使用 MUI，那可以按照如下方式安装它：

```
// 用 npm 安装
npm install @mui/material

// 用 yarn 安装
yarn add @mui/material
```

## TypeScript

为了从 theme 中的 CSS 重载 和 默认的属性自定义 中获益，使用 TypeScript 的用户需要引入如下类型的 types。 TypeScript 在内部将实验室里可用的扩展组件和 模块扩展(module augmentation) 一起使用，这样可以拓展默认的主题（theme）结构。

```
// 当使用 TypeScript 4.x 或更高版本时
import type {} from '@mui/lab/themeAugmentation';
// 当使用 TypeScript 3.x 或更低版本时
```

```
import '@mui/lab/themeAugmentation';

const theme = createTheme({
  components: {
    MuiTimeline: {
      styleOverrides: {
        root: {
          backgroundColor: 'red',
        },
      },
    },
  },
});
```