测试

编写测试能够预防回归问题，并能够带来更好的代码。

用户空间

通常情况下，我们建议测试应用程序时不要将测试程序与 Material-UI 捆绑得太紧。 这就是 Material-UI 组件内部的测试方式。 A library that has a first-class API for this approach is `@testing-library/react`.

例如，当渲染 `TextField` 时，你的测试用例不应该查询有关 `TextField` 的特定 Material-UI 实例，而是应该查询 `input` 或 `[role="textbox"]`。

通过不依赖 React 组件树的方式，你可以使你的测试应对 Material-UI 的内部变化时变得更加健壮，或者如果你需要快照测试时，也可以添加额外的包装器组件，如上下文提供器。 我们不建议进行快照测试。 "Effective snapshot testing" by Kent C. Dodds 详细介绍了为什么快照测试可能会对 React 组件测试产生误导。

内部

Material-UI 的 测试范围 **很广**，因此我们有信心 对组件进行迭代，例如，Argos-CI 提供的可视化回归测试已被证明非常有用。 若您想要进一步了解内部测试，您可以查看一下 README。