



链滴

React Visual - 弹窗组件

作者: [Vanessa](#)

原文链接: <https://ld246.com/article/1570754498261>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

描述

渲染一个可通过事件控制的弹窗组件。使用该组件时，只需要 import `Modal` 一次，然后通过传递给 `isVisible` 属性的一个布尔值来进行显示。

- 使用对象解构为弹窗组件的特定属性设置默认值
- 定义一个处理所有键盘事件的方法 `keydownHandler`，他可以根据你的需要进行操作（如：当按 `<kbd>Esc</kbd>` 时关闭弹窗组件）
- 使用 `Use React.useEffect()` hook 对键盘监听事件进行添加或移除，可直接调用 `keydownHandler` 方法
- 使用 `isVisible` 属性来设置弹窗组件的显示与否
- 使用 CSS 来设定弹窗组件的样式和位置

实现

```
.modal {
  position: fixed;
  top: 0;
  bottom: 0;
  left: 0;
  right: 0;
  width: 100%;
  z-index: 9999;
  display: flex;
  align-items: center;
  justify-content: center;
  background-color: rgba(0, 0, 0, 0.25);
  animation-name: appear;
  animation-duration: 300ms;
}
```

```
.modal-dialog {
  width: 100%;
  max-width: 550px;
  background: white;
  position: relative;
  margin: 0 20px;
  max-height: calc(100vh - 40px);
  text-align: left;
  display: flex;
  flex-direction: column;
  overflow: hidden;
  box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
  -webkit-animation-name: animatetop;
  -webkit-animation-duration: 0.4s;
  animation-name: slide-in;
  animation-duration: 0.5s;
}
```

```

.modal-header,
.modal-footer {
  display: flex;
  align-items: center;
  padding: 1rem;
}
.modal-header {
  border-bottom: 1px solid #dbdbdb;
  justify-content: space-between;
}
.modal-footer {
  border-top: 1px solid #dbdbdb;
  justify-content: flex-end;
}
.modal-close {
  cursor: pointer;
  padding: 1rem;
  margin: -1rem -1rem -1rem auto;
}
.modal-body {
  overflow: auto;
}
.modal-content {
  padding: 1rem;
}

```

```

@keyframes appear {
  from {
    opacity: 0;
  }
  to {
    opacity: 1;
  }
}

```

```

@keyframes slide-in {
  from {
    transform: translateY(-150px);
  }
  to {
    transform: translateY(0);
  }
}

```

```

function Modal({ isVisible = false, title, content, footer, onClose }) {
  React.useEffect(() => {
    document.addEventListener('keydown', keydownHandler);
    return () => document.removeEventListener('keydown', keydownHandler);
  });
}

```

```

function keydownHandler({ key }) {
  switch (key) {
    case 'Escape':
      onClose();
  }
}

```

```

        break;
      default:
    }
  }

  return !isVisible ? null : (
    <div className="modal" onClick={onClose}>
      <div className="modal-dialog" onClick={e => e.stopPropagation()}>
        <div className="modal-header">
          <h3 className="modal-title">{title}</h3>
          <span className="modal-close" onClick={onClose}>
            &times;
          </span>
        </div>
        <div className="modal-body">
          <div className="modal-content">{content}</div>
        </div>
        {footer && <div className="modal-footer">{footer}</div>}
      </div>
    </div>
  );
}

```

使用

```

//Add the component to the render function
function App() {
  const [isVisible, setModal] = React.useState(false);

  return (
    <React.Fragment>
      <button onClick={() => setModal(true)}>Click Here</button>
      <Modal
        isVisible={isVisible}
        title="Modal Title"
        content={<p>Add your content here</p>}
        footer={<button>Cancel</button>}
        onClose={() => setModal(false)}
      />
    </React.Fragment>
  );
}

```

```
ReactDOM.render(<App />, document.getElementById('root'));
```

返回总目录

[每天 30 秒系列之 React](#)