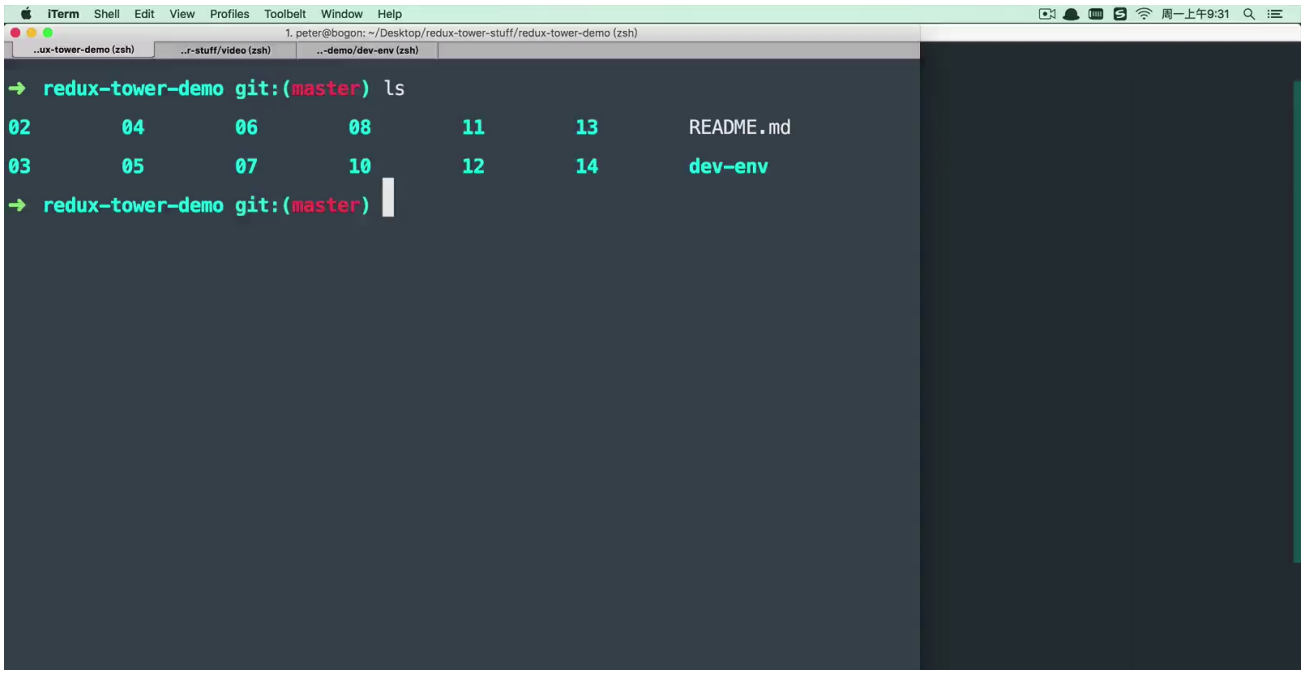

00:00 / 12:34

展示课程封面

本节课程完成的功能是把 `dev-env/src/data/courses.js` 文件中存储的课程封面信息在项目首页显示出来，同时还添加了一个蓝色的导航栏。为了实现这个功能，分别定义了三个组件：Main 组件、Courses 组件和 Course 组件。对于组件的样式则采用了

inline styles 形式，因此用到了 Radium 包。同时，还介绍了 Material-UI 组件的使用方法。

添加蓝色导航栏

新建一个目录，叫做 `components`，这个目录用来存储本项目所用到的所有组件，执行命令：

```
cd dev-env/src
mkdir components
```

然后进入 `components` 目录，新建一个 `Main.js` 文件，这就是我们的 Main 组件文件，先在 Main 组件中添加一个蓝色的导航栏，组件样式采用了 inline styles，代码如下：

```
import React, { Component } from 'react';

class Main extends Component {
  render() {
    let styles = {
      root: {
        fontFamily: 'sans-serif'
      },
      header: {
        backgroundColor: '#00bcd4',
        height: '8.4rem',
        width: '100%',
        textAlign: 'center'
      },
      logo: {
        fontWeight: '600',
        fontSize: '3rem',
        letterSpacing: '-1px',
        lineHeight: '8.4rem',
        color: '#fff'
      }
    };
    return (
```

```
        <div style={styles.root}>
          <header style={styles.header}>
            <div style={styles.logo}>Haoqicat</div>
          </header>
        </div>
      );
    }
  }

  export default Main;
```

Main 组件编写完成之后，我们还需要在 `dev-env/src/index.js` 中加载 Main 组件，才能在浏览器中看到添加的蓝色导航栏，代码如下：

```
  import React from 'react';
  import { render } from 'react-dom';

  import Main from './components/Main';

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

代码修改完成之后，就可以到浏览器中查看效果了。

Courses 组件

在项目首页还需要展示所有的课程封面，所以再编写一个 Courses 组件，组件代码存储在 `components/Courses.js` 文件，

```
  import React from 'react';
  import Radium from 'radium';

  import courses from '../data/courses';

  const Courses = React.createClass({
    render() {
      let styles = {
        root: {
          maxWidth: '1200px',
```

```
        margin: '0 auto',
        paddingTop: '6rem',
        paddingBottom: '6rem',
        display: 'flex',
        flexWrap: 'wrap'
      }
    };
    return (
      <div style={styles.root}>
        { courses.map((course, i) => <div><img src={course.image} /></d
      </div>
    )
  }
});

export default Radium(Courses);
```

所有的课程信息来自 `dev-env/src/data/courses.js` 文件中存储的 `courses` 数组，先导入数据：

```
import courses from '../data/courses';
```

得到 `courses` 数据之后，然后调用 JavaScript 的 map 接口，把每门课程的封面显示出来。

另外，代码中还用到了 Radium 包，因为在组件中使用了 `flexbox` 布局形式，所以要借助 Radium 给样式添加厂商前缀。

Courses 组件编写好之后，就可以在 Main 组件中使用了，修改 `Main.js` 组件文件，添加代码：

```
import Courses from './Courses';

<div style={styles.root}>
  ...
  <Courses />
</div>
```

Course 组件

后续视频还会给每门课程添加新功能，所以单独定义一个 Course 组件展示每门课程信息，把它和 Courses 组件分离开来。Course 组件文件代码：

```
import React, { Component } from 'react';
import { Card } from 'material-ui/Card';
import Radium from 'radium';

class Course extends Component {
  getStyles() {
    return {
      root: {
        margin: '0 2rem 4rem',
        flexBasis: '100%',
        '@media (min-width: 600px)': {
          flexBasis: 'calc(50% - 4rem)'
        }
      },
      imgWrap: {
        position: 'relative'
      },
      img: {
        width: '100%',
        display: 'block'
      }
    };
  }

  render() {
    const { course } = this.props;
    let styles = this.getStyles();
    return (
      <div style={styles.root}>
        <Card>
          <div style={styles.imgWrap}>
            <img src={course.image} alt={course.name} style={styles.img
          </div>
        </Card>
      </div>
```

```
      );
    }
  }

  export default Radium(Course);
```

Course 组件用到了 Material-UI 提供的 Card 组件，不过这个 Card 组件还不能工作，因为需要添加一套 Material-UI 的主题，才能使用 Material-UI 的组件。另外，组件布局是响应式的，在内联样式中使用了媒体查询语句，这个功能也是由 Radium 支持的，不过响应式暂且也不能生效，这要用到 Radium 提供的 `StyleRoot` 组件。引入 Material-UI 主题 和 Radium 的 `StyleRoot` 组件都需要在 Main 组件中完成。

Course 组件已经编写好了，就可以在 Courses 组件中使用它了，修改 `Courses.js` 组件文件：

```
  import Course from './Course';

  { courses.map((course, i) => <Course key={i} course={course} />)}
```

导入 Course 组件，然后通过其属性 `course`，把每门课程的信息传递给 Course 组件使用。

添加 Material-UI 主题

一般在使用 Material-UI 组件的时候，应该在 `dev-env/src/index.js` 文件中的两行代码：

```
  import injectTapEventPlugin from 'react-tap-event-plugin';
  injectTapEventPlugin();
```

react-tap-event-plugin 包能让所有的 React 组件支持 onTouchTap() 属性，一些 Material-UI 组件会依赖于它，比如说 Material-UI 的 `Tab` 组件，请查看这个欢乐的 GitHub Issue。目前为止，我们只用到了 Material-UI 的 `Card` 组件，没有上面两行代码的情况下，`Card` 组件也能正常工作。

导入 Material-UI 的主题色，需要修改 `Main.js` 组件文件，添加代码：

```
import getMuiTheme from 'material-ui/styles/getMuiTheme';

class Main extends Component {
  getChildContext() {
    return { muiTheme: getMuiTheme() };
  }
}

Main.childContextTypes = {
  muiTheme: React.PropTypes.object.isRequired,
};
```

通过上述代码，我们才能在 Main 组件及其子组件中正常使用 Material-UI 提供的组件，参考文档 Materil-UI 使用。

使用 StyleRoot 组件

继续修改 `Main.js` 组件文件，添加代码：

```
import Radium, { StyleRoot } from 'radium';
import Courses from './Courses';

<StyleRoot style={styles.root}>
  <header style={styles.header}>
    <div style={styles.logo}>Haoqicat</div>
  </header>
  <Courses />
</StyleRoot>

export default Radium(Main);
```

把 Main 组件要渲染的内容都用 `SytleRoot` 组件包裹起来，这样组件内联样式中使用的媒体查询语句才能生效，组件变成响应式组件。参考官方文档 StyleRoot 组件

至此，本节课程要实现的功能也就完成了。