1 介绍JSX（Introducing JSX）

1.1 JSX中的嵌入式表达式（Embedding Expressions in JSX）

JSX中的javascript代码用{ }包裹。

JSX分隔成多行代码去写的时候要，建议放在括号中，避免自动分号插入的陷阱，如：

const element = ( //括号

<h1>

Hello, {formatName(user)}!

</h1>

);

1.2 JSX也是表达式（JSX is an Expression Too）

编译后，JSX表达式成为常规的javascript对象。

意味着可以在if语句和for循环中使用JSX，可以将其分配给变量，将它作为函数的参数和返回值。

1.3 在JSX中指定属性（Specifying Attributes with JSX）

可以使用引号指定字符串字面量作为属性。

const element = <div tabIndex="0"></div>;

也可以使用花括号{ } 嵌入javascript表达式作为属性。

const element = <img src={user.avatarUrl}></img>;

当使用嵌入式javascript表达式作为属性时，不能再使用“”。

1.4 Specifying Children with JSX

JSX标签可能包含children，如：

const element = (

<div>

<h1>Hello!</h1>

<h2>Good to see you here.</h2>

</div>

);

注意：React DOM使用camelCase属性命名约定而不是HTML属性名称。例如，类在JSX中变为className，tabindex变为tabIndex。

1.5 JSX防止注入攻击（JSX Prevents Injection Attacks）

默认情况下，React DOM会在渲染之前转义嵌入在JSX中的任何值。 因此，它确保您永远不会注入任何未明确写入应用程序的内容。 在呈现之前，所有内容都将转换为字符串。 这有助于防止XSS（跨站点脚本）攻击。

1.6 JSX表示对象（JSX Represents Objects）

Babel将JSX编译成React.createElement()调用：

const element = (

<h1 className="greeting">

Hello, world!

</h1>

);

等价于=>>

const element = React.createElement(

'h1',

{className: 'greeting'},

'Hello, world!'

);

React.createElement（）执行一些检查以帮助您编写无错误代码，但基本上它会创建一个如下所示的对象：

// Note: this structure is simplified

const element = {

type: 'h1',

props: {

className: 'greeting',

children: 'Hello, world'

}

};

2 渲染元素（Rendering Elements）

3 组件和属性（Components and Props）

4 State and Lifecycle

5 Handling Events

6 Conditional Rendering

7 Lists and Keys

8 Forms

9 Lifting State Up

10 Composition vs Inheritance

11 Thinking In React