45.受控组件和非受控组件的区别和选型场景?

React 受控与非受控组件面试题

Q1: React 中，受控组件和非受控组件最核心的区别是什么?
A1: 最核心的区别在于**数据源（Single Source of Truth）**不同。

- **受控组件**：其表单数据由 **React State** 管理和控制。State 是唯一的数据源。
- **非受控组件**：其表单数据由 **DOM 节点本身**管理。DOM 是数据源。

Q2: 请描述一下受控组件的数据流是怎样的?
A2: 受控组件遵循一个清晰的单向数据流闭环：

1. **State 驱动 UI**：`input` 等表单元素的 `value` 属性被绑定到 React 组件的 state 上。
2. **用户输入触发事件**：用户在 `input` 中输入内容，触发其 `onChange` 事件。
3. **事件处理器更新 State**：`onChange` 事件的回调函数被执行，它通过 `setState` (或 `useState` 的 `setValue` 函数) 来更新 state。
4. **State 更新触发重渲染**：state 的更新导致组件重新渲染，`input` 的 `value` 也随之更新为最新的 state 值。

Q3: 在非受控组件中，我们通常如何获取表单元素的值？使用了哪个 React Hook?
A3: 在非受控组件中，我们通常使用 `useRef` 这个 Hook 来获取表单元素的值。

1. 使用 `useRef()` 创建一个 ref 对象。
2. 将这个 ref 对象附加到表单元素的 `ref` 属性上（例如 `<input ref={inputRef} />`）。
3. 在需要获取值的时候（例如在提交按钮的点击事件中），通过访问 ref 的 `current.value` 属性来直接从 DOM 节点上读取当前的值。

Q4: 如果一个表单需要实现实时输入验证（例如，当用户输入时，实时提示用户名是否过长），你应该选择受控组件还是非受控组件？为什么?
A4: 应该选择**受控组件**。
原因：因为实时验证的需求意味着我们需要在用户输入的**过程中**，能够立即响应数据的变化。

- 受控组件的每一次输入都会更新 React 的 state，并触发重新渲染。
- 这使得我们可以在 `onChange` 的处理函数中或者在 `render` 逻辑中，实时地访问到最新的输入值，并根据这个值来执行验证逻辑、更新提示信息或UI状态（比如禁用按钮）。
- 非受控组件只有在特定事件（如点击提交）触发时才会去读取值，无法满足这种实时响应的需求。

Q5: 在什么场景下，使用非受控组件会是更合适的选择？请列举至少两个场景。
A5: 以下场景使用非受控组件可能更合适：

1. **简单的、一次性的表单提交**：例如一个简单的搜索框或登录表单。在这类场景下，我们只关心用户最终提交的值，而不需要在输入过程中进行任何操作。使用非受控组件可以减少代码量，避免每次输入都触发重渲染。
2. **文件上传（`<input type="file" />`）**：由于安全原因，文件输入框的 `value` 属性是只读的，不能通过 React State 来设置。因此，我们只能通过 `ref` 来获取其 `FileList` 对象，这使其天然适用于非受控模式。
3. **与非 React 库集成**：当需要将 React 与一些需要直接操作 DOM 的第三方库（如某些 jQuery 插件）集成时，使用 `ref` 来获取 DOM 节点会非常方便，此时非受控组件是很好的桥梁。
4. **性能优化**：对于包含大量输入字段的复杂表单，如果每个字段都用受控组件，频繁的输入可能会导致大量的重渲染，对性能造成影响。此时，改用非受控组件可以避免这个问题。

Q6: 如果你为一个 `input` 元素设置了 `value` 属性，但忘记提供 `onChange` 处理器，会发生什么?
A6: 如果为 `input` 元素提供了 `value` 属性但没有提供 `onChange` 处理器，这个 `input` 将会变成一个**只读（read-only）**输入框。用户将无法在输入框中输入或修改任何内容。这是因为 React 会根据 `value` 属性的值来渲染输入框，但由于没有 `onChange` 处理器来更新 state，`value` 的值永远不会改变，所以 React 会一直强制输入框显示初始的 `value` 值。

Q7: 从实现方式上看，受控组件和非受控组件分别依赖哪些核心的 React 特性或 Hooks?
A7:

- **受控组件**：主要依赖 `useState` Hook 来创建和管理状态，以及 `onChange` 事件处理器来更新状态。
- **非受控组件**：主要依赖 `useRef` Hook 来创建一个引用，并将其直接附加到 DOM 元素上以便后续读取其值。

Q8: 请总结一下，你在项目中做技术选型时，判断使用受控还是非受控组件的关键决策点是什么?
A8: 决策的关键点在于：**我是否需要在用户输入的过程中，对数据进行实时的响应、处理或控制?**

- **如果答案是“是”**：比如需要实时验证、强制格式、动态禁用按钮、根据输入A来改变B等场景，就必须使用**受控组件**，因为我需要将数据纳入 React State 的管理体系中。
- **如果答案是“否”**：比如我只需要在表单最终提交时获取一次数据，中间过程完全由DOM自己处理即可，那么使用**非受控组件**会更简单、代码更少，且可能性能更好。