7-1 VDOM 的四大问题：what、why、where、how

常见面试题：什么是虚拟 DOM？说一下 React 或者 Vue 的 DIFF 算法？

虚拟 DOM

虚拟 DOM (Virtual DOM，简称 VDOM)是一种编程概念。在这个概念里，UI 以一种理想化的，或者说“虚拟的”表现形式被保存于内存中，并通过如 ReactDOM 等库使之与“真实的”DOM 同步。这个同步的过程，在 React 中就是所谓的**协调**(reconcile)，而协调的核心则是所谓的 DIFF 算法。

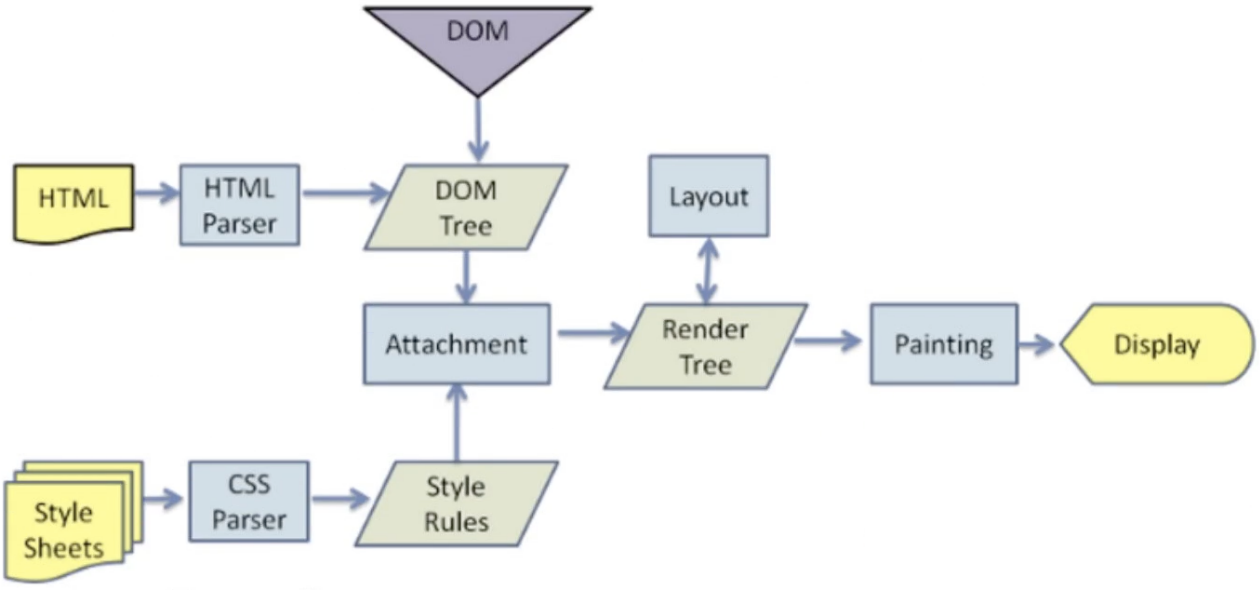
VDOM 这个概念是由 React 于 2013 年率先开拓，随后被许多不同的框架采用，包括 Vue。

注意：DIFF 是指 VDOM DIFF，不是 DOM DIFF！

what: 什么是 VDOM

用 JavaScript 对象表示 DOM 信息和结构，当状态变更的时候，重新渲染这个 JavaScript 的对象结构。这个 JavaScript 对象称为 virtual dom；

传统 dom 渲染流程



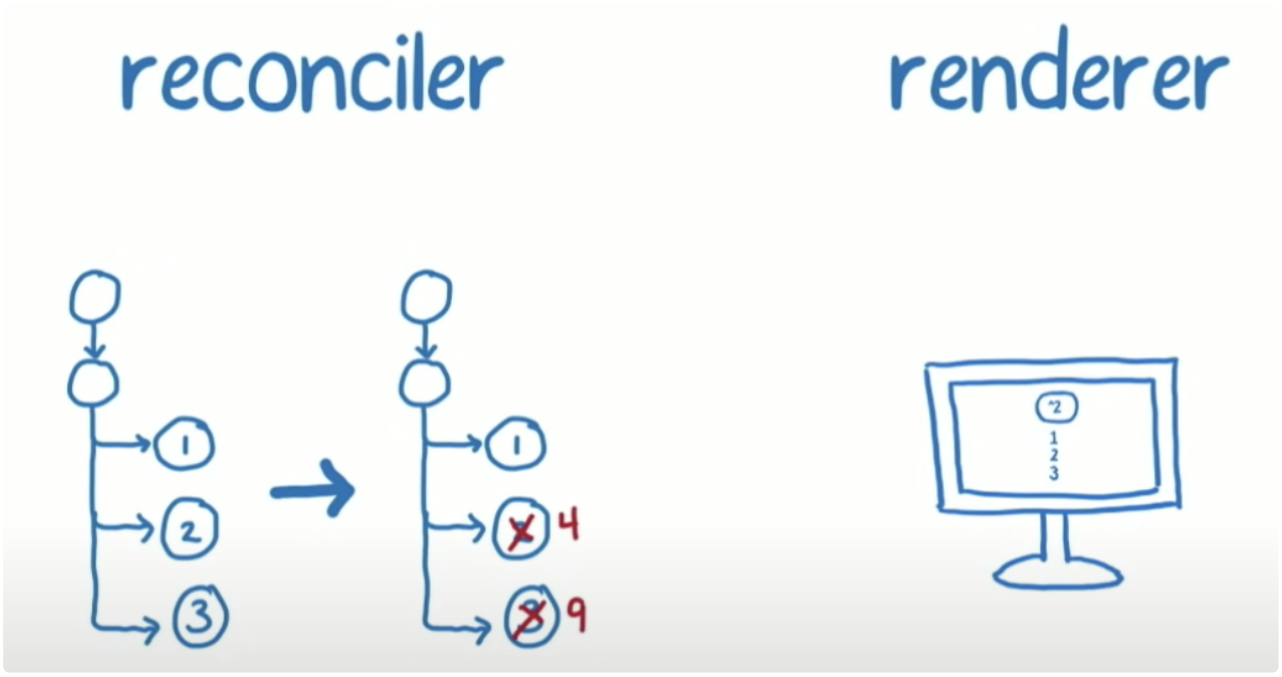

```JavaScript
let div = document.cre
let str = '';
for(let key in div) {
  str += ' ' + key;
}
console.log(str)
```

把以上代码在浏览器中执行，执行部分截图如右：

```
 align title lang translate dir hidden accessKey draggable spellcheck autocapitalize  VM23717:6
contentEditable isContentEditable inputMode offsetParent offsetTop offsetLeft offsetWidth
offsetHeight style innerText outerText oncopy oncut onpaste onabort onblur oncancel oncanplay
oncanplaythrough onchange onclick onclose oncontextmenu oncuechange ondblclick ondrag ondragend
ondragenter ondragleave ondragover ondragstart ondrop ondurationchange onemptied onended onerror
onfocus oninput oninvalid onkeydown onkeypress onkeyup onload onloadeddata onloadedmetadata
onloadstart onmousedown onmouseenter onmouseleave onmousemove onmouseout onmouseover onmouseup
onmousewheel onpause onplay onplaying onprogress onratechange onreset onresize onscroll onseeked
onseeking onselect onstalled onsubmit onsuspend ontimeupdate ontoggle onvolumechange onwaiting
onwheel onauxclick ongotpointercapture onlostpointercapture onpointerdown onpointermove
onpointerup onpointercancel onpointerover onpointerout onpointerenter onpointerleave
onselectstart onselectionchange dataset nonce tabIndex click focus blur enterKeyHint onformdata
onpointerrawupdate attachInternals namespaceURI prefix localName tagName id className classList
slot part attributes shadowRoot assignedSlot innerHTML outerHTML scrollTop scrollLeft
scrollWidth scrollHeight clientTop clientLeft clientWidth clientHeight attributeStyleMap
onbeforecopy onbeforecut onbeforepaste onsearch previousElementSibling nextElementSibling
children firstElementChild lastElementChild childElementCount onfullscreenchange
onfullscreenerror onwebkitfullscreenchange onwebkitfullscreenerror setPointerCapture
releasePointerCapture hasPointerCapture hasAttributes getAttributeNames getAttribute
getAttributeNS setAttribute setAttributeNS removeAttribute removeAttributeNS hasAttribute
hasAttributeNS toggleAttribute getAttributeNode getAttributeNodeNS setAttributeNode
setAttributeNodeNS removeAttributeNode closest matches webkitMatchesSelector attachShadow
getElementsByTagName getElementsByTagNameNS getElementsByClassName insertAdjacentElement
insertAdjacentText insertAdjacentHTML requestPointerLock getClientRects getBoundingClientRect
scrollIntoView scroll scrollTo scrollBy scrollIntoViewIfNeeded animate computedStyleMap before
after replaceWith remove prepend append querySelector querySelectorAll requestFullscreen
webkitRequestFullScreen webkitRequestFullscreen createShadowRoot getDestinationInsertionPoints
elementTiming ELEMENT_NODE ATTRIBUTE_NODE TEXT_NODE CDATA_SECTION_NODE ENTITY_REFERENCE_NODE
ENTITY_NODE PROCESSING_INSTRUCTION_NODE COMMENT_NODE DOCUMENT_NODE DOCUMENT_TYPE_NODE
DOCUMENT_FRAGMENT_NODE NOTATION_NODE DOCUMENT_POSITION_DISCONNECTED DOCUMENT_POSITION_PRECEDING
```

why 为什么要用 VDOM

DOM 操作很慢，轻微的操作都可能导致页面重新排版，非常耗性能。相对于 DOM 对象，js 对象处理起来更快，而且更简单。通过 DIFF 算法对比新旧 VDOM 的差异，可以批量的、最小化的执行 DOM 操作，从而提升用户体验。



where：React 中哪里用到了 VDOM

React 中用 JSX 语法描述视图(View)。在 React17 之前，JSX 必须经过 babel-loader 转译为 `React.createElement(...)` 调用，该函数返回 ReactElement。

在 React17 以及之后，新的 JSX 转换**不会将 JSX 转换为 React.createElement**，而是自动从 React 的 package 中引入新的入口函数并调用。

How：React 中是如何使用 VDOM 的

JSX

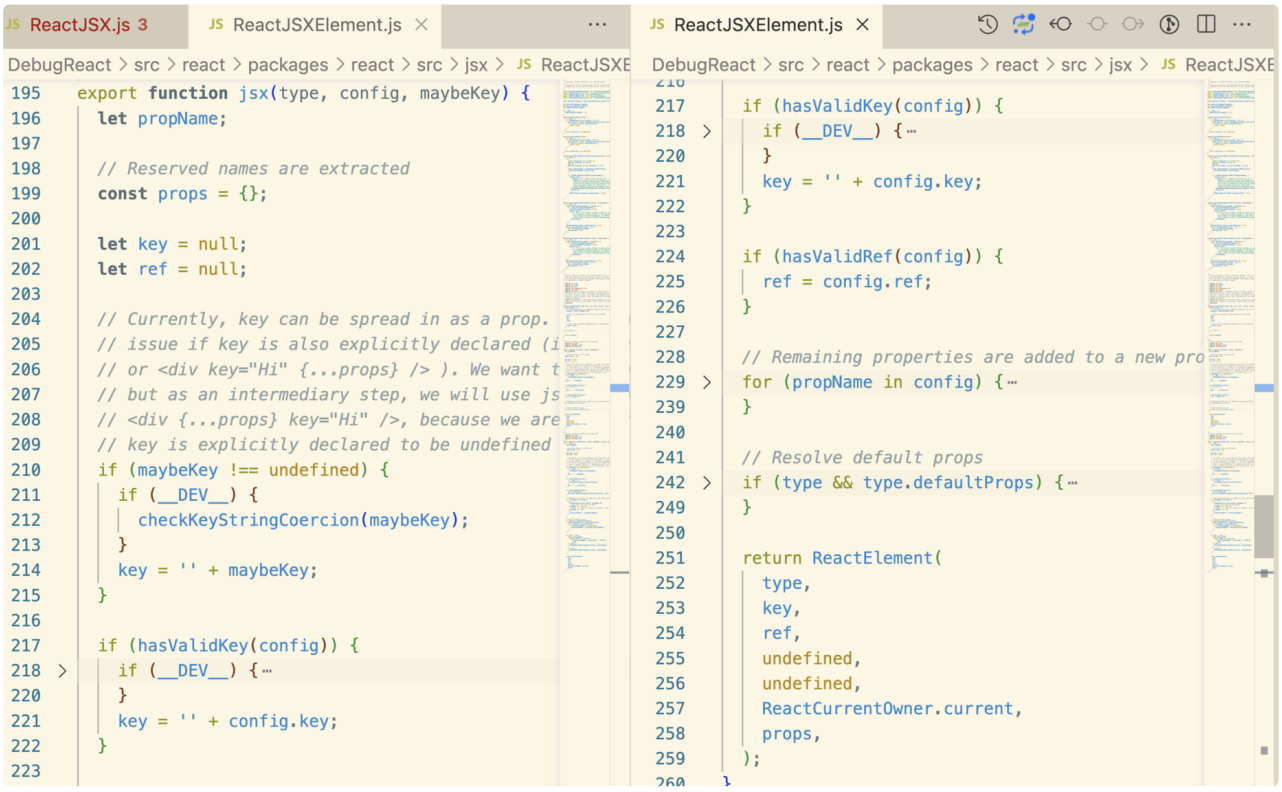
JSX 实际上是一种语法扩展，它允许开发者用类似 HTML 的语法来编写 React 组件。

VDOM 是 React 内部使用的一种技术，用于在内存中表示真实 DOM 的轻量级副本，以便进行高效的 DOM 操作。

1. 什么是 JSX

> 语法糖
>
> React 使用 JSX 来替代常规的 JavaScript。
>
> JSX 是一个看起来很像 XML 的 JavaScript 语法扩展。

2. 为什么需要 JSX
   - 开发效率：使用 JSX 编写模板简单快速。
   - 执行效率：JSX 编译为 JavaScript 代码后进行了优化，执行更快。
   - 类型安全：在编译过程中就能发现错误。
3. 与 vue 的异同：
   - react 中 vdom+jsx 的设计一开始就有，vue 则是演进过程中才出现的
   - jsx 本来就是 js 扩展，转义过程简单直接的多；vue 把 template 编译为 render 函数的过程需要复杂的编译器转换字符串–ast–js 函数字符串



ReactElement

ReactElement 是 React 中用来描述 UI 的对象，它是构建 React 组件树的基本单元。


ReactJSX.js 3
ReactJSXElement.js
DebugReact > src > react > packages > react > src > jsx > ReactJSXElement.js > ...
133 * different from the `owner` when React.createElement is called, so that we
134 * can warn. We want to get rid of owner and replace string `ref`s with arrow
135 * functions, and as long as `this` and owner are the same, there will be no
136 * change in behavior.
137 * @param {*} source An annotation object (added by a transpiler or otherwise)
138 * indicating filename, line number, and/or other information.
139 * @internal
140 */
141 function ReactElement(type, key, ref, self, source, owner, props) {
Sebastian Markbåge, 上周 | 3 authors (Luna Ruan and others)
142 const element = {
143 // This tag allows us to uniquely identify this as a React Element
144 $$typeof: REACT_ELEMENT_TYPE,
145
146 // Built-in properties that belong on the element
147 type,
148 key,
149 ref,
150 props,
151
152 // Record the component responsible for creating this element.
153 _owner: owner,
154 };
155
156 if (__DEV__) {…
184 }
185
186 return element;
187 }

Fiber


ReactJSX.js 3
ReactJSXElement.js
ReactInternalTypes.js 9+
DebugReact > src > react > packages > react-reconciler > src > ReactInternalTypes.js > Fiber
79 // A Fiber is work on a Component that needs to be done or was done. There can
80 // be more than one per component.
81 export type Fiber = {
82 // These first fields are conceptually members of an Instance. This used to
83 // be split into a separate type and intersected with the other Fiber fields,
84 // but until Flow fixes its intersection bugs, we've merged them into a
85 // single type.
86
87 // An Instance is shared between all versions of a component. We can easily
88 // break this out into a separate object to avoid copying so much to the
89 // alternate versions of the tree. We put this on a single object for now to
90 // minimize the number of objects created during the initial render.
91
92 // Tag identifying the type of fiber.
93 tag: WorkTag,
94
95 // Unique identifier of this child.
96 key: null | string,
97
98 // The value of element.type which is used to preserve the identity during
99 // reconciliation of this child.
100 elementType: any,
101
102 // The resolved function/class/ associated with this fiber.
103 type: any,
104
105 // The local state associated with this fiber.
106 stateNode: any,