18.0.0 (March 29, 2022)

Below is a list of all new features, APIs, deprecations, and breaking changes. Read React 18 release post and React 18 upgrade guide for more information.

New Features

React

- `useId` is a new hook for generating unique IDs on both the client and server, while avoiding hydration mismatches. It is primarily useful for component libraries integrating with accessibility APIs that require unique IDs. This solves an issue that already exists in React 17 and below, but it's even more important in React 18 because of how the new streaming server renderer delivers HTML out-of-order.
- `startTransition` and `useTransition` let you mark some state updates as not urgent. Other state updates are considered urgent by default. React will allow urgent state updates (for example, updating a text input) to interrupt non-urgent state updates (for example, rendering a list of search results).
- `useDeferredValue` lets you defer re-rendering a non-urgent part of the tree. It is similar to debouncing, but has a few advantages compared to it. There is no fixed time delay, so React will attempt the deferred render right after the first render is reflected on the screen. The deferred render is interruptible and doesn't block user input.
- `useSyncExternalStore` is a new hook that allows external stores to support concurrent reads by forcing updates to the store to be synchronous. It removes the need for `useEffect` when implementing subscriptions to external data sources, and is recommended for any library that integrates with state external to React.
- `useInsertionEffect` is a new hook that allows CSS-in-JS libraries to address performance issues of injecting styles in render. Unless you've already built a CSS-in-JS library we don't expect you to ever use this. This hook will run after the DOM is mutated, but before layout effects read the new layout. This solves an issue that already exists in React 17 and below, but is even more important in React 18 because React yields to the browser during concurrent rendering, giving it a chance to recalculate layout.

React DOM Client

These new APIs are now exported from `react-dom/client`:

- `createRoot`: New method to create a root to `render` or `unmount`. Use it instead of `ReactDOM.render`. New features in React 18 don't work without it.
- `hydrateRoot`: New method to hydrate a server rendered application. Use it instead of `ReactDOM.hydrate` in conjunction with the new React DOM

Server APIs. New features in React 18 don't work without it.

Both `createRoot` and `hydrateRoot` accept a new option called `onRecoverableError` in case you want to be notified when React recovers from errors during rendering or hydration for logging. By default, React will use `reportError`, or `console.error` in the older browsers.

React DOM Server

These new APIs are now exported from `react-dom/server` and have full support for streaming Suspense on the server:

- `renderToPipeableStream`: for streaming in Node environments.
- `renderToReadableStream`: for modern edge runtime environments, such as Deno and Cloudflare workers.

The existing `renderToString` method keeps working but is discouraged.

Deprecations

- `react-dom`: `ReactDOM.render` has been deprecated. Using it will warn and run your app in React 17 mode.
- `react-dom`: `ReactDOM.hydrate` has been deprecated. Using it will warn and run your app in React 17 mode.
- `react-dom`: `ReactDOM.unmountComponentAtNode` has been deprecated.
- `react-dom`: `ReactDOM.renderSubtreeIntoContainer` has been deprecated.
- `react-dom/server`: `ReactDOMServer.renderToNodeStream` has been deprecated.

Breaking Changes

React

- **Automatic batching:** This release introduces a performance improvement that changes to the way React batches updates to do more batching automatically. See Automatic batching for fewer renders in React 18 for more info. In the rare case that you need to opt out, wrap the state update in `flushSync`.
- **Stricter Strict Mode**: In the future, React will provide a feature that lets components preserve state between unmounts. To prepare for it, React 18 introduces a new development-only check to Strict Mode. React will automatically unmount and remount every component, whenever a component mounts for the first time, restoring the previous state on the second mount. If this breaks your app, consider removing Strict Mode until you can fix the components to be resilient to remounting with existing state.

- **Consistent useEffect timing**: React now always synchronously flushes effect functions if the update was triggered during a discrete user input event such as a click or a keydown event. Previously, the behavior wasn't always predictable or consistent.
- **Stricter hydration errors**: Hydration mismatches due to missing or extra text content are now treated like errors instead of warnings. React will no longer attempt to "patch up" individual nodes by inserting or deleting a node on the client in an attempt to match the server markup, and will revert to client rendering up to the closest `<Suspense>` boundary in the tree. This ensures the hydrated tree is consistent and avoids potential privacy and security holes that can be caused by hydration mismatches.
- **Suspense trees are always consistent:** If a component suspends before it's fully added to the tree, React will not add it to the tree in an incomplete state or fire its effects. Instead, React will throw away the new tree completely, wait for the asynchronous operation to finish, and then retry rendering again from scratch. React will render the retry attempt concurrently, and without blocking the browser.
- **Layout Effects with Suspense**: When a tree re-suspends and reverts to a fallback, React will now clean up layout effects, and then re-create them when the content inside the boundary is shown again. This fixes an issue which prevented component libraries from correctly measuring layout when used with Suspense.
- **New JS Environment Requirements**: React now depends on modern browsers features including `Promise`, `Symbol`, and `Object.assign`. If you support older browsers and devices such as Internet Explorer which do not provide modern browser features natively or have non-compliant implementations, consider including a global polyfill in your bundled application.

Notable Changes

React

- **Components can now render `undefined`:** React no longer throws if you return `undefined` from a component. This makes the allowed component return values consistent with values that are allowed in the middle of a component tree. We suggest to use a linter to prevent mistakes like forgetting a `return` statement before JSX.
- **In tests, `act` warnings are now opt-in:** If you're running end-to-end tests, the `act` warnings are unnecessary. We've introduced an opt-in mechanism so you can enable them only for unit tests where they are useful and beneficial.
- **No warning about `setState` on unmounted components:** Previously, React warned about memory leaks when you call `setState` on an unmounted component. This warning was added for subscriptions, but people primarily run into it in scenarios where setting state is fine, and

workarounds make the code worse. We've removed this warning.
- **No suppression of console logs:** When you use Strict Mode, React renders each component twice to help you find unexpected side effects. In React 17, we've suppressed console logs for one of the two renders to make the logs easier to read. In response to community feedback about this being confusing, we've removed the suppression. Instead, if you have React DevTools installed, the second log's renders will be displayed in grey, and there will be an option (off by default) to suppress them completely.
- **Improved memory usage:** React now cleans up more internal fields on unmount, making the impact from unfixed memory leaks that may exist in your application code less severe.

React DOM Server

- `renderToString`**:** Will no longer error when suspending on the server. Instead, it will emit the fallback HTML for the closest `<Suspense>` boundary and then retry rendering the same content on the client. It is still recommended that you switch to a streaming API like `renderToPipeableStream` or `renderToReadableStream` instead.
- `renderToStaticMarkup`**:** Will no longer error when suspending on the server. Instead, it will emit the fallback HTML for the closest `<Suspense>` boundary and retry rendering on the client.

All Changes

React

- Add `useTransition` and `useDeferredValue` to separate urgent updates from transitions. (#10426, #10715, #15593, #15272, #15578, #15769, #17058, #18796, #19121, #19703, #19719, #19724, #20672, #20976 by @acdlite, @lunaruan, @rickhanlonii, and @sebmarkbage)
- Add `useId` for generating unique IDs. (#17322, #18576, #22644, #22672, #21260 by @acdlite, @lunaruan, and @sebmarkbage)
- Add `useSyncExternalStore` to help external store libraries integrate with React. (#15022, #18000, #18771, #22211, #22292, #22239, #22347, #23150 by @acdlite, @bvaughn, and @drarmstr)
- Add `startTransition` as a version of `useTransition` without pending feedback. (#19696 by @rickhanlonii)
- Add `useInsertionEffect` for CSS-in-JS libraries. (#21913 by @rickhanlonii)
- Make Suspense remount layout effects when content reappears. (#19322, #19374, #19523, #20625, #21079 by @acdlite, @bvaughn, and @lunaruan)
- Make `<StrictMode>` re-run effects to check for restorable state. (#19523 , #21418 by @bvaughn and @lunaruan)
- Assume Symbols are always available. (#23348 by @sebmarkbage)
- Remove `object-assign` polyfill. (#23351 by @sebmarkbage)

- Remove unsupported `unstable_changedBits` API. (#20953 by @acdlite)
- Allow components to render undefined. (#21869 by @rickhanlonii)
- Flush `useEffect` resulting from discrete events like clicks synchronously. (#21150 by @acdlite)
- Suspense `fallback={undefined}` now behaves the same as `null` and isn't ignored. (#21854 by @rickhanlonii)
- Consider all `lazy()` resolving to the same component equivalent. (#20357 by @sebmarkbage)
- Don't patch console during first render. (#22308 by @lunaruan)
- Improve memory usage. (#21039 by @bgirard)
- Improve messages if string coercion throws (Temporal.*, Symbol, etc.) (#22064 by @justingrant)
- Use `setImmediate` when available over `MessageChannel`. (#20834 by @gaearon)
- Fix context failing to propagate inside suspended trees. (#23095 by @gaearon)
- Fix `useReducer` observing incorrect props by removing the eager bailout mechanism. (#22445 by @josephsavona)
- Fix `setState` being ignored in Safari when appending iframes. (#23111 by @gaearon)
- Fix a crash when rendering `ZonedDateTime` in the tree. (#20617 by @dimaqq)
- Fix a crash when document is set to `null` in tests. (#22695 by @SimenB)
- Fix `onLoad` not triggering when concurrent features are on. (#23316 by @gnoff)
- Fix a warning when a selector returns `NaN`. (#23333 by @hachibeeDI)
- Fix the generated license header. (#23004 by @vitaliemiron)
- Add `package.json` as one of the entry points. (#22954 by @Jack)
- Allow suspending outside a Suspense boundary. (#23267 by @acdlite)
- Log a recoverable error whenever hydration fails. (#23319 by @acdlite)

React DOM

- Add `createRoot` and `hydrateRoot`. (#10239, #11225, #12117, #13732, #15502, #15532, #17035, #17165, #20669, #20748, #20888, #21072, #21417, #21652, #21687, #23207, #23385 by @acdlite, @bvaughn, @gaearon, @lunaruan, @rickhanlonii, @trueadm, and @sebmarkbage)
- Add selective hydration. (#14717, #14884, #16725, #16880, #17004, #22416, #22629, #22448, #22856, #23176 by @acdlite, @gaearon, @salazarm, and @sebmarkbage)
- Add `aria-description` to the list of known ARIA attributes. (#22142 by @mahyareb)
- Add `onResize` event to video elements. (#21973 by @rileyjshaw)
- Add `imageSizes` and `imageSrcSet` to known props. (#22550 by @eps1lon)
- Allow non-string `<option>` children if `value` is provided. (#21431 by @sebmarkbage)

- Fix `aspectRatio` style not being applied. (#21100 by @gaearon)
- Warn if `renderSubtreeIntoContainer` is called. (#23355 by @acdlite)

React DOM Server

- Add the new streaming renderer. (#14144, #20970, #21056, #21255, #21200, #21257, #21276, #22443, #22450, #23247, #24025, #24030 by @sebmarkbage)
- Fix context providers in SSR when handling multiple requests. (#23171 by @frandiox)
- Revert to client render on text mismatch. (#23354 by @acdlite)
- Deprecate `renderToNodeStream`. (#23359 by @sebmarkbage)
- Fix a spurious error log in the new server renderer. (#24043 by @eps1lon)
- Fix a bug in the new server renderer. (#22617 by @shuding)
- Ignore function and symbol values inside custom elements on the server. (#21157 by @sebmarkbage)

React DOM Test Utils

- Throw when `act` is used in production. (#21686 by @acdlite)
- Support disabling spurious act warnings with `global.IS_REACT_ACT_ENVIRONMENT`. (#22561 by @acdlite)
- Expand act warning to cover all APIs that might schedule React work. (#22607 by @acdlite)
- Make `act` batch updates. (#21797 by @acdlite)
- Remove warning for dangling passive effects. (#22609 by @acdlite)

React Refresh

- Track late-mounted roots in Fast Refresh. (#22740 by @anc95)
- Add `exports` field to `package.json`. (#23087 by @otakustay)

Server Components (Experimental)

- Add Server Context support. (#23244 by @salazarm)
- Add `lazy` support. (#24068 by @gnoff)
- Update webpack plugin for webpack 5 (#22739 by @michenly)
- Fix a mistake in the Node loader. (#22537 by @btea)
- Use `globalThis` instead of `window` for edge environments. (#22777 by @huozhi)

17.0.2 (March 22, 2021)

React DOM

- Remove an unused dependency to address the `SharedArrayBuffer` cross-origin isolation warning. (@koba04 and @bvaughn in #20831, #20832, and #20840)

17.0.1 (October 22, 2020)

React DOM

- Fix a crash in IE11. (@gaearon in #20071)

17.0.0 (October 20, 2020)

Today, we are releasing React 17!

Learn more about React 17 and how to update to it on the official React blog.

React

- Add `react/jsx-runtime` and `react/jsx-dev-runtime` for the new JSX transform. (@lunaruan in #18299)
- Build component stacks from native error frames. (@sebmarkbage in #18561)
- Allow to specify `displayName` on context for improved stacks. (@eps1lon in #18224)
- Prevent `'use strict'` from leaking in the UMD bundles. (@koba04 in #19614)
- Stop using `fb.me` for redirects. (@cylim in #19598)

React DOM

- Delegate events to roots instead of `document`. (@trueadm in #18195 and others)
- Clean up all effects before running any next effects. (@bvaughn in #17947)
- Run `useEffect` cleanup functions asynchronously. (@bvaughn in #17925)
- Use browser `focusin` and `focusout` for `onFocus` and `onBlur`. (@trueadm in #19186)
- Make all `Capture` events use the browser capture phase. (@trueadm in #19221)
- Don't emulate bubbling of the `onScroll` event. (@gaearon in #19464)
- Throw if `forwardRef` or `memo` component returns `undefined`. (@gaearon in #19550)
- Remove event pooling. (@trueadm in #18969)
- Stop exposing internals that won't be needed by React Native Web. (@necolas in #18483)
- Attach all known event listeners when the root mounts. (@gaearon in #19659)
- Disable `console` in the second render pass of DEV mode double render. (@sebmarkbage in #18547)
- Deprecate the undocumented and misleading `ReactTestUtils.SimulateNative` API. (@gaearon in #13407)
- Rename private field names used in the internals. (@gaearon in #18377)

- Don't call User Timing API in development. (@gaearon in #18417)
- Disable console during the repeated render in Strict Mode. (@sebmarkbage in #18547)
- In Strict Mode, double-render components without Hooks too. (@eps1lon in #18430)
- Allow calling `ReactDOM.flushSync` during lifecycle methods (but warn). (@sebmarkbage in #18759)
- Add the `code` property to the keyboard event objects. (@bl00mber in #18287)
- Add the `disableRemotePlayback` property for `video` elements. (@tombrowndev in #18619)
- Add the `enterKeyHint` property for `input` elements. (@eps1lon in #18634)
- Warn when no `value` is provided to `<Context.Provider>`. (@charlie1404 in #19054)
- Warn when `memo` or `forwardRef` components return `undefined`. (@bvaughn in #19550)
- Improve the error message for invalid updates. (@JoviDeCroock in #18316)
- Exclude forwardRef and memo from stack frames. (@sebmarkbage in #18559)
- Improve the error message when switching between controlled and uncontrolled inputs. (@vcarl in #17070)
- Keep `onTouchStart`, `onTouchMove`, and `onWheel` passive. (@gaearon in #19654)
- Fix `setState` hanging in development inside a closed iframe. (@gaearon in #19220)
- Fix rendering bailout for lazy components with `defaultProps`. (@jddxf in #18539)
- Fix a false positive warning when `dangerouslySetInnerHTML` is `undefined`. (@eps1lon in #18676)
- Fix Test Utils with non-standard `require` implementation. (@just-boris in #18632)
- Fix `onBeforeInput` reporting an incorrect `event.type`. (@eps1lon in #19561)
- Fix `event.relatedTarget` reported as `undefined` in Firefox. (@claytercek in #19607)
- Fix "unspecified error" in IE11. (@hemakshis in #19664)
- Fix rendering into a shadow root. (@Jack-Works in #15894)
- Fix `movementX/Y` polyfill with capture events. (@gaearon in #19672)
- Use delegation for `onSubmit` and `onReset` events. (@gaearon in #19333)
- Improve memory usage. (@trueadm in #18970)

React DOM Server

- Make `useCallback` behavior consistent with `useMemo` for the server renderer. (@alexmckenley in #18783)
- Fix state leaking when a function component throws. (@pmaccart in

#19212)

React Test Renderer

- Improve `findByType` error message. (@henryqdineen in #17439)

Concurrent Mode (Experimental)

- Revamp the priority batching heuristics. (@acdlite in #18796)
- Add the `unstable_` prefix before the experimental APIs. (@acdlite in #18825)
- Remove `unstable_discreteUpdates` and `unstable_flushDiscreteUpdates`. (@trueadm in #18825)
- Remove the `timeoutMs` argument. (@acdlite in #19703)
- Disable `<div hidden />` prerendering in favor of a different future API. (@acdlite in #18917)
- Add `unstable_expectedLoadTime` to Suspense for CPU-bound trees. (@acdlite in #19936)
- Add an experimental `unstable_useOpaqueIdentifier` Hook. (@lunaruan in #17322)
- Add an experimental `unstable_startTransition` API. (@rickhanlonii in #19696)
- Using `act` in the test renderer no longer flushes Suspense fallbacks. (@acdlite in #18596)
- Use global render timeout for CPU Suspense. (@sebmarkbage in #19643)
- Clear the existing root content before mounting. (@bvaughn in #18730)
- Fix a bug with error boundaries. (@acdlite in #18265)
- Fix a bug causing dropped updates in a suspended tree. (@acdlite in #18384 and #18457)
- Fix a bug causing dropped render phase updates. (@acdlite in #18537)
- Fix a bug in SuspenseList. (@sebmarkbage in #18412)
- Fix a bug causing Suspense fallback to show too early. (@acdlite in #18411)
- Fix a bug with class components inside SuspenseList. (@sebmarkbage in #18448)
- Fix a bug with inputs that may cause updates to be dropped. (@jddxf in #18515 and @acdlite in #18535)
- Fix a bug causing Suspense fallback to get stuck. (@acdlite in #18663)
- Don't cut off the tail of a SuspenseList if hydrating. (@sebmarkbage in #18854)
- Fix a bug in `useMutableSource` that may happen when `getSnapshot` changes. (@bvaughn in #18297)
- Fix a tearing bug in `useMutableSource`. (@bvaughn in #18912)
- Warn if calling setState outside of render but before commit. (@sebmarkbage in #18838)

16.14.0 (October 14, 2020)

React

- Add support for the new JSX transform. (@lunaruan in #18299)

16.13.1 (March 19, 2020)

React DOM

- Fix bug in legacy mode Suspense where effect clean-up functions are not fired. This only affects users who use Suspense for data fetching in legacy mode, which is not technically supported. (@acdlite in #18238)
- Revert warning for cross-component updates that happen inside class render lifecycles (`componentWillReceiveProps`, `shouldComponentUpdate`, and so on). (@gaearon in #18330)

16.13.0 (February 26, 2020)

React

- Warn when a string ref is used in a manner that's not amenable to a future codemod (@lunaruan in #17864)
- Deprecate `React.createFactory()` (@trueadm in #17878)

React DOM

- Warn when changes in `style` may cause an unexpected collision (@sophiebits in #14181, #18002)
- Warn when a function component is updated during another component's render phase (@acdlite in #17099)
- Deprecate `unstable_createPortal` (@trueadm in #17880)
- Fix `onMouseEnter` being fired on disabled buttons (@AlfredoGJ in #17675)
- Call `shouldComponentUpdate` twice when developing in `StrictMode` (@bvaughn in #17942)
- Add `version` property to ReactDOM (@ealush in #15780)
- Don't call `toString()` of `dangerouslySetInnerHTML` (@sebmarkbage in #17773)
- Show component stacks in more warnings (@gaearon in #17922, #17586)

Concurrent Mode (Experimental)

- Warn for problematic usages of `ReactDOM.createRoot()` (@trueadm in #17937)
- Remove `ReactDOM.createRoot()` callback params and added warnings on usage (@bvaughn in #17916)
- Don't group Idle/Offscreen work with other work (@sebmarkbage in #17456)

- Adjust `SuspenseList` CPU bound heuristic (@sebmarkbage in #17455)
- Add missing event plugin priorities (@trueadm in #17914)
- Fix `isPending` only being true when transitioning from inside an input event (@acdlite in #17382)
- Fix `React.memo` components dropping updates when interrupted by a higher priority update (@acdlite in #18091)
- Don't warn when suspending at the wrong priority (@gaearon in #17971)
- Fix a bug with rebasing updates (@acdlite and @sebmarkbage in #17560, #17510, #17483, #17480)

16.12.0 (November 14, 2019)

React DOM

- Fix passive effects (`useEffect`) not being fired in a multi-root app. (@acdlite in #17347)

React Is

- Fix `lazy` and `memo` types considered elements instead of components (@bvaughn in #17278)

16.11.0 (October 22, 2019)

React DOM

- Fix `mouseenter` handlers from firing twice inside nested React containers. @yuanoook in #16928
- Remove `unstable_createRoot` and `unstable_createSyncRoot` experimental APIs. (These are available in the Experimental channel as `createRoot` and `createSyncRoot`.) (@acdlite in #17088)

16.10.2 (October 3, 2019)

React DOM

- Fix regression in react-native-web by restoring order of arguments in event plugin extractors (@necolas in #16978)

16.10.1 (September 28, 2019)

React DOM

- Fix regression in Next.js apps by allowing Suspense mismatch during hydration to silently proceed (@sebmarkbage in #16943)

16.10.0 (September 27, 2019)

React DOM

- Fix edge case where a hook update wasn't being memoized. (@sebmarkbage in #16359)
- Fix heuristic for determining when to hydrate, so we don't incorrectly hydrate during an update. (@sebmarkbage in #16739)
- Clear additional fiber fields during unmount to save memory. (@trueadm in #16807)
- Fix bug with required text fields in Firefox. (@halvves in #16578)
- Prefer `Object.is` instead of inline polyfill, when available. (@ku8ar in #16212)
- Fix bug when mixing Suspense and error handling. (@acdlite in #16801)

Scheduler (Experimental)

- Improve queue performance by switching its internal data structure to a min binary heap. (@acdlite in #16245)
- Use `postMessage` loop with short intervals instead of attempting to align to frame boundaries with `requestAnimationFrame`. (@acdlite in #16214)

useSubscription

- Avoid tearing issue when a mutation happens and the previous update is still in progress. (@bvaughn in #16623)

16.9.0 (August 8, 2019)

React

- Add `<React.Profiler>` API for gathering performance measurements programmatically. (@bvaughn in #15172)
- Remove `unstable_ConcurrentMode` in favor of `unstable_createRoot`. (@acdlite in #15532)

React DOM

- Deprecate old names for the `UNSAFE_*` lifecycle methods. (@bvaughn in #15186 and @threepointone in #16103)
- Deprecate `javascript:` URLs as a common attack surface. (@sebmarkbage in #15047)
- Deprecate uncommon "module pattern" (factory) components. (@sebmarkbage in #15145)
- Add support for the `disablePictureInPicture` attribute on `<video>`. (@eek in #15334)
- Add support for `onLoad` event for `<embed>`. (@cherniavskii in #15614)

- Add support for editing `useState` state from DevTools. (@bvaughn in #14906)
- Add support for toggling Suspense from DevTools. (@gaearon in #15232)
- Warn when `setState` is called from `useEffect`, creating a loop. (@gaearon in #15180)
- Fix a memory leak. (@paulshen in #16115)
- Fix a crash inside `findDOMNode` for components wrapped in `<Suspense>`. (@acdlite in #15312)
- Fix pending effects from being flushed too late. (@acdlite in #15650)
- Fix incorrect argument order in a warning message. (@brickspert in #15345)
- Fix hiding Suspense fallback nodes when there is an `!important` style. (@acdlite in #15861 and #15882)
- Slightly improve hydration performance. (@bmeurer in #15998)

React DOM Server

- Fix incorrect output for camelCase custom CSS property names. (@bedakb in #16167)

React Test Utilities and Test Renderer

- Add `act(async () => ...)` for testing asynchronous state updates. (@threepointone in #14853)
- Add support for nesting `act` from different renderers. (@threepointone in #16039 and #16042)
- Warn in Strict Mode if effects are scheduled outside an `act()` call. (@threepointone in #15763 and #16041)
- Warn when using `act` from the wrong renderer. (@threepointone in #15756)

ESLint Plugin: React Hooks

- Report Hook calls at the top level as a violation. (gaearon in #16455)

16.8.6 (March 27, 2019)

React DOM

- Fix an incorrect bailout in `useReducer()`. (@acdlite in #15124)
- Fix iframe warnings in Safari DevTools. (@renanvalentin in #15099)
- Warn if `contextType` is set to `Context.Consumer` instead of `Context`. (@aweary in #14831)
- Warn if `contextType` is set to invalid values. (@gaearon in #15142)

16.8.5 (March 22, 2019)

React DOM

- Don't set the first option as selected in select tag with `size` attribute. (@kulek1 in #14242)
- Improve the `useEffect(async () => ...)` warning message. (@gaearon in #15118)
- Improve the error message sometimes caused by duplicate React. (@jared-palmer in #15139)

React DOM Server

- Improve the `useLayoutEffect` warning message when server rendering. (@gaearon in #15158)

React Shallow Renderer

- Fix `setState` in shallow renderer to work with Hooks. (@gaearon in #15120)
- Fix shallow renderer to support `React.memo`. (@aweary in #14816)
- Fix shallow renderer to support Hooks inside `forwardRef`. (@eps1lon in #15100)

16.8.4 (March 5, 2019)

React DOM and other renderers

- Fix a bug where DevTools caused a runtime error when inspecting a component that used a `useContext` hook. (@bvaughn in #14940)

16.8.3 (February 21, 2019)

React DOM

- Fix a bug that caused inputs to behave incorrectly in UMD builds. (@gaearon in #14914)
- Fix a bug that caused render phase updates to be discarded. (@gaearon in #14852)

React DOM Server

- Unwind the context stack when a stream is destroyed without completing, to prevent incorrect values during a subsequent render. (@overlookmotel in #14706)

ESLint Plugin for React Hooks

- Add a new recommended `exhaustive-deps` rule. (@gaearon in #14636)

16.8.2 (February 14, 2019)

React DOM

- Fix `ReactDOM.render` being ignored inside `useEffect`. (@gaearon in #14799)
- Fix a crash when unmounting empty portals. (@gaearon in #14820)
- Fix `useImperativeHandle` to work correctly when no deps are specified. (@gaearon in #14801)
- Fix `crossOrigin` attribute to work in SVG `image` elements. (@aweary in #14832)
- Fix a false positive warning when using Suspense with Hooks. (@gaearon in #14821)

React Test Utils and React Test Renderer

- Include component stack into the `act()` warning. (@threepointone in #14855)

16.8.1 (February 6, 2019)

React DOM and React Test Renderer

- Fix a crash when used together with an older version of React. (@bvaughn in #14770)

React Test Utils

- Fix a crash in Node environment. (@threepointone in #14768)

16.8.0 (February 6, 2019)

React

- Add Hooks — a way to use state and other React features without writing a class. (@acdlite et al. in #13968)
- Improve the `useReducer` Hook lazy initialization API. (@acdlite in #14723)

React DOM

- Bail out of rendering on identical values for `useState` and `useReducer` Hooks. (@acdlite in #14569)
- Use `Object.is` algorithm for comparing `useState` and `useReducer` values. (@Jessidhia in #14752)
- Don't compare the first argument passed to `useEffect`/`useMemo`/`useCallback` Hooks. (@acdlite in #14594)
- Support synchronous thenables passed to `React.lazy()`. (@gaearon in #14626)

- Render components with Hooks twice in Strict Mode (DEV-only) to match class behavior. (@gaearon in #14654)
- Warn about mismatching Hook order in development. (@threepointone in #14585 and @acdlite in #14591)
- Effect clean-up functions must return either `undefined` or a function. All other values, including `null`, are not allowed. @acdlite in #14119

React Test Renderer and Test Utils

- Support Hooks in the shallow renderer. (@trueadm in #14567)
- Fix wrong state in `shouldComponentUpdate` in the presence of `getDerivedStateFromProps` for Shallow Renderer. (@chenesan in #14613)
- Add `ReactTestRenderer.act()` and `ReactTestUtils.act()` for batching updates so that tests more closely match real behavior. (@threepointone in #14744)

ESLint Plugin: React Hooks

- Initial release. (@calebmer in #13968)
- Fix reporting after encountering a loop. (@calebmer and @Yurickh in #14661)
- Don't consider throwing to be a rule violation. (@sophiebits in #14040)

16.7.0 (December 19, 2018)

React DOM

- Fix performance of `React.lazy` for large numbers of lazily-loaded components. (@acdlite in #14429)
- Clear fields on unmount to avoid memory leaks. (@trueadm in #14276)
- Fix bug with SSR and context when mixing `react-dom/server@16.6` and `react@<16.6`. (@gaearon in #14291)
- Fix a performance regression in profiling mode. (@bvaughn in #14383)

Scheduler (Experimental)

- Post to MessageChannel instead of window. (@acdlite in #14234)
- Reduce serialization overhead. (@developit in #14249)
- Fix fallback to `setTimeout` in testing environments. (@bvaughn in #14358)
- Add methods for debugging. (@mrkev in #14053)

16.6.3 (November 12, 2018)

React DOM

- Fix bugs in `Suspense` and `lazy`. (@acdlite in #14133, #14157, and #14164)

- Fix highlighting of `React.memo` updates in React DevTools. (@bvaughn in #14141)
- Fix interaction of Suspense with the React Profiler. (@bvaughn in #14065)
- Fix a false positive warning when using Suspense. (@acdlite in #14158)

React DOM Server

- Fix incorrect sharing of context state between `renderToNodeStream()` calls. (@sebmarkbage in #14182)
- Add a warning about incorrect usage of the context API. (@trueadm in #14033)

16.6.2 (November 12, 2018)

This release was published in a broken state and should be skipped.

16.6.1 (November 6, 2018)

React DOM

- Fallback should not remount every time a promise resolves. (@acdlite in #14083)
- Fix bug where Suspense keeps showing fallback even after everything finishes loading. (@acdlite in #14083)
- Fix a crash when Suspense finishes loading in IE11. (@sophiebits in #14126)
- Fix unresolved default props in lifecycle methods of a lazy component. (@gaearon in #14112)
- Fix bug when recovering from an error thrown during complete phase. (@gaearon in #14104)

Scheduler (Experimental)

- Switch from deadline object to `shouldYield` API. (@acdlite in #14025)

16.6.0 (October 23, 2018)

React

- Add `React.memo()` as an alternative to `PureComponent` for functions. (@acdlite in #13748)
- Add `React.lazy()` for code splitting components. (@acdlite in #13885)
- `React.StrictMode` now warns about legacy context API. (@bvaughn in #13760)
- `React.StrictMode` now warns about `findDOMNode`. (@sebmarkbage in #13841)
- Rename `unstable_AsyncMode` to `unstable_ConcurrentMode`. (@trueadm in #13732)

- Rename `unstable_Placeholder` to `Suspense`, and `delayMs` to `maxDuration`. (@gaearon in #13799 and @sebmarkbage in #13922)

React DOM

- Add `contextType` as a more ergonomic way to subscribe to context from a class. (@bvaughn in #13728)
- Add `getDerivedStateFromError` lifecycle method for catching errors in a future asynchronous server-side renderer. (@bvaughn in #13746)
- Warn when `<Context>` is used instead of `<Context.Consumer>`. (@trueadm in #13829)
- Fix gray overlay on iOS Safari. (@philipp-spiess in #13778)
- Fix a bug caused by overwriting `window.event` in development. (@sergei-startsev in #13697)

React DOM Server

- Add support for `React.memo()`. (@alexmckenley in #13855)
- Add support for `contextType`. (@alexmckenley and @sebmarkbage in #13889)

Scheduler (Experimental)

- Rename the package to `scheduler`. (@gaearon in #13683)
- Support priority levels, continuations, and wrapped callbacks. (@acdlite in #13720 and #13842)
- Improve the fallback mechanism in non-DOM environments. (@acdlite in #13740)
- Schedule `requestAnimationFrame` earlier. (@acdlite in #13785)
- Fix the DOM detection to be more thorough. (@trueadm in #13731)
- Fix bugs with interaction tracing. (@bvaughn in #13590)
- Add the `envify` transform to the package. (@mridgway in #13766)

16.5.2 (September 18, 2018)

React DOM

- Fixed a recent `<iframe>` regression (@JSteunou in #13650)
- Fix `updateWrapper` so that `<textarea>`s no longer re-render when data is unchanged (@joelbarbosa in #13643)

Schedule (Experimental)

- Renaming "tracking" API to "tracing" (@bvaughn in #13641)
- Add UMD production+profiling entry points (@bvaughn in #13642)
- Refactored `schedule` to remove some React-isms and improve performance for when deferred updates time out (@acdlite in #13582)

16.5.1 (September 13, 2018)

React

- Improve the warning when `React.forwardRef` receives an unexpected number of arguments. (@andresroberto in #13636)

React DOM

- Fix a regression in unstable exports used by React Native Web. (@aweary in #13598)
- Fix a crash when component defines a method called `isReactComponent`. (@gaearon in #13608)
- Fix a crash in development mode in IE9 when printing a warning. (@link-alex in #13620)
- Provide a better error message when running `react-dom/profiling` with `schedule/tracking`. (@bvaughn in #13605)
- If a `ForwardRef` component defines a `displayName`, use it in warnings. (@probablyup in #13615)

Schedule (Experimental)

- Add a separate profiling entry point at `schedule/tracking-profiling`. (@bvaughn in #13605)

16.5.0 (September 5, 2018)

React

- Add a warning if `React.forwardRef` render function doesn't take exactly two arguments (@bvaughn in #13168)
- Improve the error message when passing an element to `createElement` by mistake (@DCtheTall in #13131)
- Don't call profiler `onRender` until after mutations (@bvaughn in #13572)

React DOM

- Add support for React DevTools Profiler (@bvaughn in #13058)
- Add `react-dom/profiling` entry point alias for profiling in production (@bvaughn in #13570)
- Add `onAuxClick` event for browsers that support it (@jquense in #11571)
- Add `movementX` and `movementY` fields to mouse events (@jasonwilliams in #9018)
- Add `tangentialPressure` and `twist` fields to pointer events (@motiz88 in #13374)
- Minimally support iframes (nested browsing contexts) in selection event handling (@acusti in #12037)

- Support passing booleans to the `focusable` SVG attribute (@gaearon in #13339)
- Ignore `<noscript>` on the client when hydrating (@Ephem in #13537)
- Fix `gridArea` to be treated as a unitless CSS property (@mgol in #13550)
- Fix incorrect data in `compositionend` event when typing Korean on IE11 (@crux153 in #12563)
- Fix a crash when using dynamic `children` in the `<option>` tag (@Slowyn in #13261, @gaearon in #13465)
- Fix the `checked` attribute not getting initially set on the `input` (@dilidili in #13114)
- Fix hydration of `dangerouslySetInnerHTML` when `__html` is not a string (@gaearon in #13353)
- Fix a warning about missing controlled `onChange` to fire on falsy values too (@nicolevy in #12628)
- Fix `submit` and `reset` buttons getting an empty label (@ellsclytn in #12780)
- Fix the `onSelect` event not being triggered after drag and drop (@gaearon in #13422)
- Fix the `onClick` event not working inside a portal on iOS (@aweary in #11927)
- Fix a performance issue when thousands of roots are re-rendered (@gaearon in #13335)
- Fix a performance regression that also caused `onChange` to not fire in some cases (@gaearon in #13423)
- Handle errors in more edge cases gracefully (@gaearon in #13237 and @acdlite in #13269)
- Don't use proxies for synthetic events in development (@gaearon in #12171)
- Warn when `"false"` or `"true"` is the value of a boolean DOM prop (@motiz88 in #13372)
- Warn when `this.state` is initialized to `props` (@veekas in #11658)
- Don't compare `style` on hydration in IE due to noisy false positives (@mgol in #13534)
- Include `StrictMode` in the component stack (@gaearon in #13240)
- Don't overwrite `window.event` in IE (@ConradIrwin in #11696)
- Improve component stack for the `folder/index.js` naming convention (@gaearon in #12059)
- Improve a warning when using `getDerivedStateFromProps` without initialized state (@flxwu in #13317)
- Improve a warning about invalid textarea usage (@raunofreiberg in #13361)
- Treat invalid Symbol and function values more consistently (@raunofreiberg in #13362 and #13389)
- Allow Electron `<webview>` tag without warnings (@philipp-spiess in #13301)
- Don't show the uncaught error addendum if `e.preventDefault()` was called (@gaearon in #13384)
- Warn about rendering Generators (@gaearon in #13312)

- Remove irrelevant suggestion of a legacy method from a warning (@zx6658 in #13169)
- Remove `unstable_deferredUpdates` in favor of `unstable_scheduleWork` from `schedule` (@gaearon in #13488)
- Fix unstable asynchronous mode from doing unnecessary work when an update takes too long (@acdlite in #13503)

React DOM Server

- Fix crash with nullish children when using `dangerouslySetInnerHtml` in a selected `<option>` (@mridgway in #13078)
- Fix crash when `setTimeout` is missing (@dustinsoftware in #13088)

React Test Renderer and Test Utils

- Fix `this` in a functional component for shallow renderer to be `undefined` (@koba04 in #13144)
- Deprecate a Jest-specific `ReactTestUtils.mockComponent()` helper (@bvaughn in #13193)
- Warn about `ReactDOM.createPortal` usage within the test renderer (@bvaughn in #12895)
- Improve a confusing error message (@gaearon in #13351)

React ART

- Add support for DevTools (@yunchancho in #13173)

Schedule (Experimental)

- New package for cooperatively scheduling work in a browser environment. It's used by React internally, but its public API is not finalized yet. (@flarnie in #12624)

16.4.2 (August 1, 2018)

React DOM Server

- Fix a potential XSS vulnerability when the attacker controls an attribute name (`CVE-2018-6341`). This fix is available in the latest `react-dom@16.4.2`, as well as in previous affected minor versions: `react-dom@16.0.1`, `react-dom@16.1.2`, `react-dom@16.2.1`, and `react-dom@16.3.3`. (@gaearon in #13302)
- Fix a crash in the server renderer when an attribute is called `hasOwnProperty`. This fix is only available in `react-dom@16.4.2`. (@gaearon in #13303)

16.4.1 (June 13, 2018)

React

- You can now assign `propTypes` to components returned by `React.ForwardRef`. (@bvaughn in #12911)

React DOM

- Fix a crash when the input `type` changes from some other types to `text`. (@spirosikmd in #12135)
- Fix a crash in IE11 when restoring focus to an SVG element. (@ThaddeusJiang in #12996)
- Fix a range input not updating in some cases. (@Illu in #12939)
- Fix input validation triggering unnecessarily in Firefox. (@nhunzaker in #12925)
- Fix an incorrect `event.target` value for the `onChange` event in IE9. (@nhunzaker in #12976)
- Fix a false positive error when returning an empty `<React.Fragment />` from a component. (@philipp-spiess in #12966)

React DOM Server

- Fix an incorrect value being provided by new context API. (@ericsoderberghp in #12985, @gaearon in #13019)

React Test Renderer

- Allow multiple root children in test renderer traversal API. (@gaearon in #13017)
- Fix `getDerivedStateFromProps()` in the shallow renderer to not discard the pending state. (@fatfisz in #13030)

16.4.0 (May 23, 2018)

React

- Add a new experimental `React.unstable_Profiler` component for measuring performance. (@bvaughn in #12745)

React DOM

- Add support for the Pointer Events specification. (@philipp-spiess in #12507)
- Properly call `getDerivedStateFromProps()` regardless of the reason for re-rendering. (@acdlite in #12600 and #12802)
- Fix a bug that prevented context propagation in some cases. (@gaearon in #12708)

- Fix re-rendering of components using `forwardRef()` on a deeper `setState()`. (@gaearon in #12690)
- Fix some attributes incorrectly getting removed from custom element nodes. (@airamrguez in #12702)
- Fix context providers to not bail out on children if there's a legacy context provider above. (@gaearon in #12586)
- Add the ability to specify `propTypes` on a context provider component. (@nicolevy in #12658)
- Fix a false positive warning when using `react-lifecycles-compat` in `<StrictMode>`. (@bvaughn in #12644)
- Warn when the `forwardRef()` render function has `propTypes` or `defaultProps`. (@bvaughn in #12644)
- Improve how `forwardRef()` and context consumers are displayed in the component stack. (@sophiebits in #12777)
- Change internal event names. This can break third-party packages that rely on React internals in unsupported ways. (@philipp-spiess in #12629)

React Test Renderer

- Fix the `getDerivedStateFromProps()` support to match the new React DOM behavior. (@koba04 in #12676)
- Fix a `testInstance.parent` crash when the parent is a fragment or another special node. (@gaearon in #12813)
- `forwardRef()` components are now discoverable by the test renderer traversal methods. (@gaearon in #12725)
- Shallow renderer now ignores `setState()` updaters that return `null` or `undefined`. (@koba04 in #12756)

React ART

- Fix reading context provided from the tree managed by React DOM. (@acdlite in #12779)

React Call Return (Experimental)

- This experiment was deleted because it was affecting the bundle size and the API wasn't good enough. It's likely to come back in the future in some other form. (@gaearon in #12820)

React Reconciler (Experimental)

- The new host config shape is flat and doesn't use nested objects. (@gaearon in #12792)

16.3.3 (August 1, 2018)

React DOM Server

- Fix a potential XSS vulnerability when the attacker controls an attribute name (`CVE-2018-6341`). This fix is available in the latest `react-dom@16.4.2`, as well as in previous affected minor versions: `react-dom@16.0.1`, `react-dom@16.1.2`, `react-dom@16.2.1`, and `react-dom@16.3.3`. (@gaearon in #13302)

16.3.2 (April 16, 2018)

React

- Improve the error message when passing `null` or `undefined` to `React.cloneElement`. (@nicolevy in #12534)

React DOM

- Fix an IE crash in development when using `<StrictMode>`. (@bvaughn in #12546)
- Fix labels in User Timing measurements for new component types. (@bvaughn in #12609)
- Improve the warning about wrong component type casing. (@nicolevy in #12533)
- Improve general performance in development mode. (@gaearon in #12537)
- Improve performance of the experimental `unstable_observedBits` API with nesting. (@gaearon in #12543)

React Test Renderer

- Add a UMD build. (@bvaughn in #12594)

16.3.1 (April 3, 2018)

React

- Fix a false positive warning in IE11 when using `Fragment`. (@heikkilamarko in #12504)
- Prefix a private API. (@Andarist in #12501)
- Improve the warning when calling `setState()` in constructor. (@gaearon in #12532)

React DOM

- Fix `getDerivedStateFromProps()` not getting applied in some cases. (@acdlite in #12528)
- Fix a performance regression in development mode. (@gaearon in #12510)

- Fix error handling bugs in development mode. (@gaearon and @acdlite in #12508)
- Improve user timing API messages for profiling. (@flarnie in #12384)

Create Subscription

- Set the package version to be in sync with React releases. (@bvaughn in #12526)
- Add a peer dependency on React 16.3+. (@NMinhNguyen in #12496)

16.3.0 (March 29, 2018)

React

- Add a new officially supported context API. (@acdlite in #11818)
- Add a new `React.createRef()` API as an ergonomic alternative to callback refs. (@trueadm in #12162)
- Add a new `React.forwardRef()` API to let components forward their refs to a child. (@bvaughn in #12346)
- Fix a false positive warning in IE11 when using `React.Fragment`. (@XaveScor in #11823)
- Replace `React.unstable_AsyncComponent` with `React.unstable_AsyncMode`. (@acdlite in #12117)
- Improve the error message when calling `setState()` on an unmounted component. (@sophiebits in #12347)

React DOM

- Add a new `getDerivedStateFromProps()` lifecycle and `UNSAFE_` aliases for the legacy lifecycles. (@bvaughn in #12028)
- Add a new `getSnapshotBeforeUpdate()` lifecycle. (@bvaughn in #12404)
- Add a new `<React.StrictMode>` wrapper to help prepare apps for async rendering. (@bvaughn in #12083)
- Add support for `onLoad` and `onError` events on the `<link>` tag. (@roderickhsiao in #11825)
- Add support for `noModule` boolean attribute on the `<script>` tag. (@aweary in #11900)
- Fix minor DOM input bugs in IE and Safari. (@nhunzaker in #11534)
- Correctly detect Ctrl + Enter in `onKeyPress` in more browsers. (@nstraub in #10514)
- Fix containing elements getting focused on SSR markup mismatch. (@koba04 in #11737)
- Fix `value` and `defaultValue` to ignore Symbol values. (@nhunzaker in #11741)
- Fix refs to class components not getting cleaned up when the attribute is removed. (@bvaughn in #12178)

- Fix an IE/Edge issue when rendering inputs into a different window. (@M-ZubairAhmed in #11870)
- Throw with a meaningful message if the component runs after jsdom has been destroyed. (@gaearon in #11677)
- Don't crash if there is a global variable called `opera` with a `null` value. @alisherdavronov in #11854)
- Don't check for old versions of Opera. (@skiritsis in #11921)
- Deduplicate warning messages about `<option selected>`. (@watadarkstar in #11821)
- Deduplicate warning messages about invalid callback. (@yenshih in #11833)
- Deprecate `ReactDOM.unstable_createPortal()` in favor of `ReactDOM.createPortal()`. (@prometheansacrifice in #11747)
- Don't emit User Timing entries for context types. (@abhaynikam in #12250)
- Improve the error message when context consumer child isn't a function. (@raunofreiberg in #12267)
- Improve the error message when adding a ref to a functional component. (@skiritsis in #11782)

React DOM Server

- Prevent an infinite loop when attempting to render portals with SSR. (@gaearon in #11709)
- Warn if a class doesn't extend `React.Component`. (@wyze in #11993)
- Fix an issue with `this.state` of different components getting mixed up. (@sophiebits in #12323)
- Provide a better message when component type is undefined. (@HeroProtagonist in #11966)

React Test Renderer

- Fix handling of fragments in `toTree()`. (@maciej-ka in #12107 and @gaearon in #12154)
- Shallow renderer should assign state to `null` for components that don't set it. (@jwbay in #11965)
- Shallow renderer should filter legacy context according to `contextTypes`. (@koba04 in #11922)
- Add an unstable API for testing asynchronous rendering. (@acdlite in #12478)

React Is (New)

- First release of the new package that libraries can use to detect different React node types. (@bvaughn in #12199)
- Add `ReactIs.isValidElementType()` to help higher-order components validate their inputs. (@jamesreggio in #12483)

React Lifecycles Compat (New)

- First release of the new package to help library developers target multiple versions of React. (@bvaughn in #12105)

Create Subscription (New)

- First release of the new package to subscribe to external data sources safely for async rendering. (@bvaughn in #12325)

React Reconciler (Experimental)

- Expose `react-reconciler/persistent` for building renderers that use persistent data structures. (@gaearon in #12156)
- Pass host context to `finalizeInitialChildren()`. (@jquense in #11970)
- Remove `useSyncScheduling` from the host config. (@acdlite in #11771)

React Call Return (Experimental)

- Fix a crash on updates. (@rmhartog in #11955)

16.2.1 (August 1, 2018)

React DOM Server

- Fix a potential XSS vulnerability when the attacker controls an attribute name (`CVE-2018-6341`). This fix is available in the latest `react-dom@16.4.2`, as well as in previous affected minor versions: `react-dom@16.0.1`, `react-dom@16.1.2`, `react-dom@16.2.1`, and `react-dom@16.3.3`. (@gaearon in #13302)

16.2.0 (November 28, 2017)

React

- Add `Fragment` as named export to React. (@clemmy in #10783)
- Support experimental Call/Return types in `React.Children` utilities. (@MatteoVH in #11422)

React DOM

- Fix radio buttons not getting checked when using multiple lists of radios. (@landvibe in #11227)
- Fix radio buttons not receiving the `onChange` event in some cases. (@jquense in #11028)

React Test Renderer

- Fix `setState()` callback firing too early when called from `componentWillMount`. (@accordeiro in #11507)

React Reconciler

- Expose `react-reconciler/reflection` with utilities useful to custom renderers. (@rivenhk in #11683)

Internal Changes

- Many tests were rewritten against the public API. Big thanks to everyone who contributed!

16.1.2 (August 1, 2018)

React DOM Server

- Fix a potential XSS vulnerability when the attacker controls an attribute name (`CVE-2018-6341`). This fix is available in the latest `react-dom@16.4.2`, as well as in previous affected minor versions: `react-dom@16.0.1`, `react-dom@16.1.2`, `react-dom@16.2.1`, and `react-dom@16.3.3`. (@gaearon in #13302)

16.1.1 (November 13, 2017)

React

- Improve the warning about undefined component type. (@selbekk in #11505)

React DOM

- Support string values for the `capture` attribute. (@maxschmeling in #11424)

React DOM Server

- Don't freeze the `ReactDOMServer` public API. (@travi in #11531)
- Don't emit `autoFocus={false}` attribute on the server. (@gaearon in #11543)

React Reconciler

- Change the hydration API for better Flow typing. (@sebmarkbage in #11493)

16.1.0 (November 9, 2017)

Discontinuing Bower Releases

Starting with 16.1.0, we will no longer be publishing new releases on Bower. You can continue using Bower for old releases, or point your Bower configs to the React UMD builds hosted on unpkg that mirror npm releases and will continue to be updated.

All Packages

- Fix an accidental extra global variable in the UMD builds. (@gaearon in #10935)

React

- Add support for portals in `React.Children` utilities. (@MatteoVH in #11378)
- Warn when a class has a `render` method but doesn't extend a known base class. (@sw-yx in #11168)
- Improve the warning when accidentally returning an object from constructor. (@deanbrophy in #11395)

React DOM

- Allow `on` as a custom attribute for AMP. (@nuc in #11153)
- Fix `onMouseEnter` and `onMouseLeave` firing on wrong elements. (@gaearon in #11164)
- Fix `null` showing up in a warning instead of the component stack. (@gaearon in #10915)
- Fix IE11 crash in development mode. (@leidegre in #10921)
- Fix `tabIndex` not getting applied to SVG elements. (@gaearon in #11034)
- Fix SVG children not getting cleaned up on `dangerouslySetInnerHTML` in IE. (@OriR in #11108)
- Fix false positive text mismatch warning caused by newline normalization. (@gaearon in #11119)
- Fix `form.reset()` to respect `defaultValue` on uncontrolled `<select>`. (@aweary in #11057)
- Fix `<textarea>` placeholder not rendering on IE11. (@gaearon in #11177)
- Fix a crash rendering into shadow root. (@gaearon in #11037)
- Fix false positive warning about hydrating mixed case SVG tags. (@gaearon in #11174)
- Suppress the new unknown tag warning for `<dialog>` element. (@gaearon in #11035)
- Warn when defining a non-existent `componentDidReceiveProps` method. (@iamtommcc in #11479)
- Warn about function child no more than once. (@andreysaleba in #11120)

- Warn about nested updates no more than once. (@anushreesubramani in #11113)
- Deduplicate other warnings about updates. (@anushreesubramani in #11216)
- Include component stack into the warning about `contentEditable` and `children`. (@Ethan-Arrowood in #11208)
- Improve the warning about booleans passed to event handlers. (@NicBonetto in #11308)
- Improve the warning when a multiple `select` gets null `value`. (@Hendeca in #11141)
- Move link in the warning message to avoid redirect. (@marciovicente in #11400)
- Add a way to suppress the React DevTools installation prompt. (@gaearon in #11448)
- Remove unused code. (@gaearon in #10802, #10803)

React DOM Server

- Add a new `suppressHydrationWarning` attribute for intentional client/server text mismatches. (@sebmarkbage in #11126)
- Fix markup generation when components return strings. (@gaearon in #11109)
- Fix obscure error message when passing an invalid style value. (@iamdustan in #11173)
- Include the `autoFocus` attribute into SSR markup. (@gaearon in #11192)
- Include the component stack into more warnings. (@gaearon in #11284)

React Test Renderer and Test Utils

- Fix multiple `setState()` calls in `componentWillMount()` in shallow renderer. (@Hypnosphi in #11167)
- Fix shallow renderer to ignore `shouldComponentUpdate()` after `forceUpdate()`. (@d4rky-pl in #11239 and #11439)
- Handle `forceUpdate()` and `React.PureComponent` correctly. (@koba04 in #11440)
- Add back support for running in production mode. (@gaearon in #11112)
- Add a missing `package.json` dependency. (@gaearon in #11340)

React ART

- Add a missing `package.json` dependency. (@gaearon in #11341)
- Expose `react-art/Circle`, `react-art/Rectangle`, and `react-art/Wedge`. (@gaearon in #11343)

React Reconciler (Experimental)

- First release of the new experimental package for creating custom renderers. (@iamdustan in #10758)
- Add support for React DevTools. (@gaearon in #11463)

React Call Return (Experimental)

- First release of the new experimental package for parent-child communication. (@gaearon in #11364)

16.0.1 (August 1, 2018)

React DOM Server

- Fix a potential XSS vulnerability when the attacker controls an attribute name (`CVE-2018-6341`). This fix is available in the latest `react-dom@16.4.2`, as well as in previous affected minor versions: `react-dom@16.0.1`, `react-dom@16.1.2`, `react-dom@16.2.1`, and `react-dom@16.3.3`. (@gaearon in #13302)

16.0.0 (September 26, 2017)

New JS Environment Requirements

- React 16 depends on the collection types Map and Set, as well as requestAnimationFrame. If you support older browsers and devices which may not yet provide these natively (e.g. <IE11), you may want to include a polyfill.

New Features

- Components can now return arrays and strings from `render`. (Docs coming soon!)
- Improved error handling with introduction of "error boundaries". Error boundaries are React components that catch JavaScript errors anywhere in their child component tree, log those errors, and display a fallback UI instead of the component tree that crashed.
- First-class support for declaratively rendering a subtree into another DOM node with `ReactDOM.createPortal()`. (Docs coming soon!)
- Streaming mode for server side rendering is enabled with `ReactDOMServer.renderToNodeStream()` and `ReactDOMServer.renderToStaticNodeStream()`. (@aickin in #10425, #10044, #10039, #10024, #9264, and others.)
- React DOM now allows passing non-standard attributes. (@nhunzaker in #10385, 10564, #10495 and others)

Breaking Changes

- There are several changes to the behavior of scheduling and lifecycle methods:
  - `ReactDOM.render()` and `ReactDOM.unstable_renderIntoContainer()` now return `null` if called from inside a lifecycle method.
    - To work around this, you can either use the new portal API or refs.
  - Minor changes to `setState` behavior:
    - Calling `setState` with null no longer triggers an update. This allows you to decide in an updater function if you want to re-render.
    - Calling `setState` directly in render always causes an update. This was not previously the case. Regardless, you should not be calling `setState` from render.
    - `setState` callback (second argument) now fires immediately after `componentDidMount` / `componentDidUpdate` instead of after all components have rendered.
  - When replacing `<A />` with `<B />`, `B.componentWillMount` now always happens before `A.componentWillUnmount`. Previously, `A.componentWillUnmount` could fire first in some cases.
  - Previously, changing the `ref` to a component would always detach the ref before that component's render is called. Now, we change the `ref` later, when applying the changes to the DOM.
  - It is not safe to re-render into a container that was modified by something other than React. This worked previously in some cases but was never supported. We now emit a warning in this case. Instead you should clean up your component trees using `ReactDOM.unmountComponentAtNode`. See this example.
  - `componentDidUpdate` lifecycle no longer receives `prevContext` param. (@bvaughn in #8631)
  - Non-unique keys may now cause children to be duplicated and/or omitted. Using non-unique keys is not (and has never been) supported, but previously it was a hard error.
  - Shallow renderer no longer calls `componentDidUpdate()` because DOM refs are not available. This also makes it consistent with `componentDidMount()` (which does not get called in previous versions either).
  - Shallow renderer does not implement `unstable_batchedUpdates()` anymore.
  - `ReactDOM.unstable_batchedUpdates` now only takes one extra argument after the callback.
- The names and paths to the single-file browser builds have changed to emphasize the difference between development and production builds. For example:
  - `react/dist/react.js` → `react/umd/react.development.js`

  - `react/dist/react.min.js` → `react/umd/react.production.min.js`
  - `react-dom/dist/react-dom.js` → `react-dom/umd/react-dom.development.js`
  - `react-dom/dist/react-dom.min.js` → `react-dom/umd/react-dom.production.min.js`
- The server renderer has been completely rewritten, with some improvements:
  - Server rendering does not use markup validation anymore, and instead tries its best to attach to existing DOM, warning about inconsistencies. It also doesn't use comments for empty components and data-reactid attributes on each node anymore.
  - Hydrating a server rendered container now has an explicit API. Use `ReactDOM.hydrate` instead of `ReactDOM.render` if you're reviving server rendered HTML. Keep using `ReactDOM.render` if you're just doing client-side rendering.
- When "unknown" props are passed to DOM components, for valid values, React will now render them in the DOM. See this post for more details. (@nhunzaker in #10385, 10564, #10495 and others)
- Errors in the render and lifecycle methods now unmount the component tree by default. To prevent this, add error boundaries to the appropriate places in the UI.

Removed Deprecations

- There is no `react-with-addons.js` build anymore. All compatible addons are published separately on npm, and have single-file browser versions if you need them.
- The deprecations introduced in 15.x have been removed from the core package. `React.createClass` is now available as create-react-class, `React.PropTypes` as prop-types, `React.DOM` as react-dom-factories, react-addons-test-utils as react-dom/test-utils, and shallow renderer as react-test-renderer/shallow. See 15.5.0 and 15.6.0 blog posts for instructions on migrating code and automated codemods.

15.7.0 (October 14, 2020)

React

- Backport support for the new JSX transform to 15.x. (@lunaruan in #18299 and @gaearon in #20024)

15.6.2 (September 25, 2017)

All Packages

- Switch from BSD + Patents to MIT license

React DOM

- Fix a bug where modifying `document.documentMode` would trigger IE detection in other browsers, breaking change events. (@aweary in #10032)
- CSS Columns are treated as unitless numbers. (@aweary in #10115)
- Fix bug in QtWebKit when wrapping synthetic events in proxies. (@walrusfruitcake in #10115)
- Prevent event handlers from receiving extra argument in development. (@aweary in #10115)
- Fix cases where `onChange` would not fire with `defaultChecked` on radio inputs. (@jquense in #10156)
- Add support for `controlList` attribute to allowed DOM properties (@nhunzaker in #9940)
- Fix a bug where creating an element with a ref in a constructor did not throw an error in development. (@iansu in #10025)

15.6.1 (June 14, 2017)

React DOM

- Fix a crash on iOS Safari. (@jquense in #9960)
- Don't add `px` to custom CSS property values. (@TrySound in #9966)

15.6.0 (June 13, 2017)

React

- Downgrade deprecation warnings to use `console.warn` instead of `console.error`. (@flarnie in #9753)
- Add a deprecation warning for `React.createClass`. Points users to `create-react-class` instead. (@flarnie in #9771)
- Add deprecation warnings and separate module for `React.DOM` factory helpers. (@nhunzaker in #8356)
- Warn for deprecation of `React.createMixin` helper, which was never used. (@aweary in #8853)

React DOM

- Add support for CSS variables in `style` attribute. (@aweary in #9302)
- Add support for CSS Grid style properties. (@ericsakmar in #9185)
- Fix bug where inputs mutated value on type conversion. (@nhunzaker in #9806)
- Fix issues with `onChange` not firing properly for some inputs. (@jquense in #8575)
- Fix bug where controlled number input mistakenly allowed period. (@nhunzaker in #9584)
- Fix bug where performance entries were being cleared. (@chrisui in #9451)

React Addons

- Fix AMD support for addons depending on `react`. (@flarnie in #9919)
- Fix `isMounted()` to return `true` in `componentWillUnmount`. (@mridgway in #9638)
- Fix `react-addons-update` to not depend on native `Object.assign`. (@gaearon in #9937)
- Remove broken Google Closure Compiler annotation from `create-react-class`. (@gaearon in #9933)
- Remove unnecessary dependency from `react-linked-input`. (@gaearon in #9766)
- Point `react-addons-(css-)transition-group` to the new package. (@gaearon in #9937)

15.5.4 (April 11, 2017)

React Addons

- **Critical Bugfix:** Update the version of `prop-types` to fix critical bug. (@gaearon in 545c87f)
- Fix `react-addons-create-fragment` package to include `loose-envify` transform for Browserify users. (@mridgway in #9642)

React Test Renderer

- Fix compatibility with Enzyme by exposing `batchedUpdates` on shallow renderer. (@gaearon in 9382)

15.5.3 (April 7, 2017)

Note: this release has a critical issue and was deprecated. Please update to 15.5.4 or higher.

React Addons

- Fix `react-addons-create-fragment` package to export correct thing. (@gaearon in #9385)
- Fix `create-react-class` package to include `loose-envify` transform for Browserify users. (@mridgway in #9642)

15.5.2 (April 7, 2017)

Note: this release has a critical issue and was deprecated. Please update to 15.5.4 or higher.

React Addons

- Fix the production single-file builds to not include the development code. (@gaearon in #9385)
- Apply better minification to production single-file builds. (@gaearon in #9385)
- Add missing and remove unnecessary dependencies to packages. (@gaearon in #9385)

15.5.1 (April 7, 2017)

Note: this release has a critical issue and was deprecated. Please update to 15.5.4 or higher.

React

- Fix erroneous PropTypes access warning. (@acdlite in (ec97ebb)

15.5.0 (April 7, 2017)

Note: this release has a critical issue and was deprecated. Please update to 15.5.4 or higher.

React

- Added a deprecation warning for `React.createClass`. Points users to create-react-class instead. (@acdlite in #d9a4fa4)
- Added a deprecation warning for `React.PropTypes`. Points users to prop-types instead. (@acdlite in #043845c)
- Fixed an issue when using `ReactDOM` together with `ReactDOMServer`. (@wacii in #9005)
- Fixed issue with Closure Compiler. (@anmonteiro in #8895)
- Another fix for Closure Compiler. (@Shastel in #8882)
- Added component stack info to invalid element type warning. (@n3tr in #8495)

React DOM

- Fixed Chrome bug when backspacing in number inputs. (@nhunzaker in #7359)
- Added `react-dom/test-utils`, which exports the React Test Utils. (@bvaughn)

React Test Renderer

- Fixed bug where `componentWillUnmount` was not called for children. (@gre in #8512)

- Added `react-test-renderer/shallow`, which exports the shallow renderer. (@bvaughn)

React Addons

- Last release for addons; they will no longer be actively maintained.
- Removed `peerDependencies` so that addons continue to work indefinitely. (@acdlite and @bvaughn in 8a06cd7 and 67a8db3)
- Updated to remove references to `React.createClass` and `React.PropTypes` (@acdlite in 12a96b9)
- `react-addons-test-utils` is deprecated. Use `react-dom/test-utils` and `react-test-renderer/shallow` instead. (@bvaughn)

15.4.2 (January 6, 2017)

React

- Fixed build issues with the Brunch bundler. (@gaearon in #8686)
- Improved error messages for invalid element types. (@sophiebits in #8612)
- Removed a warning about `getInitialState` when `this.state` is set. (@bvaughn in #8594)
- Removed some dead code. (@diegomura in #8050, @dfrownfelter in #8597)

React DOM

- Fixed a decimal point issue on uncontrolled number inputs. (@nhunzaker in #7750)
- Fixed rendering of textarea placeholder in IE11. (@aweary in #8020)
- Worked around a script engine bug in IE9. (@eoin in #8018)

React Addons

- Fixed build issues in RequireJS and SystemJS environments. (@gaearon in #8686)
- Added missing package dependencies. (@kweiberth in #8467)

15.4.1 (November 22, 2016)

React

- Restructure variable assignment to work around a Rollup bug (@gaearon in #8384)

React DOM

- Fixed event handling on disabled button elements (@sophiebits in #8387)
- Fixed compatibility of browser build with AMD environments (@zpao in #8374)

15.4.0 (November 16, 2016)

React

- React package and browser build no longer "secretly" includes React DOM. (@sebmarkbage in #7164 and #7168)
- Required PropTypes now fail with specific messages for null and undefined. (@chenglou in #7291)
- Improved development performance by freezing children instead of copying. (@keyanzhang in #7455)

React DOM

- Fixed occasional test failures when React DOM is used together with shallow renderer. (@goatslacker in #8097)
- Added a warning for invalid `aria-` attributes. (@jessebeach in #7744)
- Added a warning for using `autofocus` rather than `autoFocus`. (@hkal in #7694)
- Removed an unnecessary warning about polyfilling `String.prototype.split`. (@nhunzaker in #7629)
- Clarified the warning about not calling PropTypes manually. (@jedwards1211 in #7777)
- The unstable `batchedUpdates` API now passes the wrapped function's return value through. (@bgnorlov in #7444)
- Fixed a bug with updating text in IE 8. (@mnpenner in #7832)

React Perf

- When ReactPerf is started, you can now view the relative time spent in components as a chart in Chrome Timeline. (@gaearon in #7549)

React Test Utils

- If you call `Simulate.click()` on a `<input disabled onClick={foo} />` then `foo` will get called whereas it didn't before. (@nhunzaker in #7642)

React Test Renderer

- Due to packaging changes, it no longer crashes when imported together with React DOM in the same file. (@sebmarkbage in #7164 and #7168)
- `ReactTestRenderer.create()` now accepts `{createNodeMock: element => mock}` as an optional argument so you can mock refs with snapshot testing. (@Aweary in #7649, #8261)

15.3.2 (September 19, 2016)

React

- Remove plain object warning from React.createElement & React.cloneElement. (@spudly in #7724)

React DOM

- Add `playsInline` to supported HTML attributes. (@reaperhulk in #7519)
- Add `as` to supported HTML attributes. (@kevinslin in #7582)
- Improve DOM nesting validation warning about whitespace. (@sophiebits in #7515)
- Avoid "Member not found" exception in IE10 when calling `preventDefault()` in Synthetic Events. (@g-palmer in #7411)
- Fix memory leak in `onSelect` implementation. (@AgtLucas in #7533)
- Improve robustness of `document.documentMode` checks to handle Google Tag Manager. (@SchleyB in #7594)
- Add more cases to controlled inputs warning. (@marcin-mazurek in #7544)
- Handle case of popup blockers overriding `document.createEvent`. (@Andarist in #7621)
- Fix issue with `dangerouslySetInnerHTML` and SVG in Internet Explorer. (@zpao in #7618)
- Improve handling of Japanese IME on Internet Explorer. (@msmania in #7107)

React Test Renderer

- Support error boundaries. (@millermedeiros in #7558, #7569, #7619)
- Skip null ref warning. (@Aweary in #7658)

React Perf Add-on

- Ensure lifecycle timers are stopped on errors. (@gaearon in #7548)

15.3.1 (August 19, 2016)

React

- Improve performance of development builds in various ways. (@gaearon in #7461, #7463, #7483, #7488, #7491, #7510)
- Cleanup internal hooks to improve performance of development builds. (@gaearon in #7464, #7472, #7481, #7496)
- Upgrade fbjs to pick up another performance improvement from @gaearon for development builds. (@zpao in #7532)
- Improve startup time of React in Node. (@zertosh in #7493)
- Improve error message of `React.Children.only`. (@sophiebits in #7514)

React DOM

- Avoid `<input>` validation warning from browsers when changing `type`. (@nhunzaker in #7333)
- Avoid "Member not found" exception in IE10 when calling `stopPropagation()` in Synthetic Events. (@nhunzaker in #7343)
- Fix issue resulting in inability to update some `<input>` elements in mobile browsers. (@keyanzhang in #7397)
- Fix memory leak in server rendering. (@keyanzhang in #7410)
- Fix issue resulting in `<input type="range">` values not updating when changing `min` or `max`. (@troydemonbreun in #7486)
- Add new warning for rare case of attempting to unmount a container owned by a different copy of React. (@ventuno in #7456)

React Test Renderer

- Fix ReactTestInstance::toJSON() with empty top-level components. (@Morhaus in #7523)

React Native Renderer

- Change `trackedTouchCount` invariant into a console.error for better reliability. (@yungsters in #7400)

15.3.0 (July 29, 2016)

React

- Add `React.PureComponent` - a new base class to extend, replacing `react-addons-pure-render-mixin` now that mixins don't work with ES2015 classes. (@sophiebits in #7195)
- Add new warning when modifying `this.props.children`. (@jimfb in #7001)
- Fixed issue with ref resolution order. (@gaearon in #7101)
- Warn when mixin is undefined. (@swaroopsm in #6158)
- Downgrade "unexpected batch number" invariant to a warning. (@sophiebits in #7133)
- Validate arguments to `oneOf` and `oneOfType` PropTypes sooner. (@troydemonbreun in #6316)
- Warn when calling PropTypes directly. (@Aweary in #7132, #7194)
- Improve warning when using Maps as children. (@keyanzhang in #7260)
- Add additional type information to the `PropTypes.element` warning. (@alexzherdev in #7319)
- Improve component identification in no-op `setState` warning. (@keyanzhang in #7326)

React DOM

- Fix issue with nested server rendering. (@Aweary in #7033)
- Add `xmlns`, `xmlnsXlink` to supported SVG attributes. (@salzhrani in #6471)
- Add `referrerPolicy` to supported HTML attributes. (@Aweary in #7274)
- Fix issue resulting in `<input type="range">` initial value being rounded. (@troydemonbreun in #7251)

React Test Renderer

- Initial public release of package allowing more focused testing. Install with `npm install react-test-renderer`. (@sophiebits in #6944, #7258, @iamdustan in #7362)

React Perf Add-on

- Fix issue resulting in excessive warnings when encountering an internal measurement error. (@sassanh in #7299)

React TestUtils Add-on

- Implement `type` property on for events created via `TestUtils.Simulate.*`. (@yaycmyk in #6154)
- Fix crash when running TestUtils with the production build of React. (@gaearon in #7246)

15.2.1 (July 8, 2016)

React

- Fix errant warning about missing React element. (@gaearon in #7193)
- Better removal of dev-only code, leading to a small reduction in the minified production bundle size. (@gaearon in #7188, #7189)

React DOM

- Add stack trace to null input value warning. (@jimfb in #7040)
- Fix webcomponents example. (@jalexanderfox in #7057)
- Fix `unstable_renderSubtreeIntoContainer` so that context properly updates when linked to state. (@gaearon in #7125)
- Improve invariant wording for void elements. (@starkch in #7066)
- Ensure no errors are thrown due to event handlers in server rendering. (@rricard in #7127)
- Fix regression resulting in `value`-less submit and reset inputs removing the browser-default text. (@zpao in #7197)
- Fix regression resulting in empty `name` attribute being added to inputs when not provided. (@okonet in #7199)

- Fix issue with nested server rendering. (@Aweary in #7033)

React Perf Add-on

- Make `ReactPerf.start()` work properly during lifecycle methods. (@gaearon in #7208).

React CSSTransitionGroup Add-on

- Fix issue resulting in spurious unknown property warnings. (@batusai513 in #7165)

React Native Renderer

- Improve error handling in cross-platform touch event handling. (@yungsters in #7143)

15.2.0 (July 1, 2016)

React

- Add error codes to production invariants, with links to the view the full error text. (@keyanzhang in #6948)
- Include component stack information in PropType validation warnings. (@troydemonbreun in #6398, @sophiebits in #6771)
- Include component stack information in key warnings. (@keyanzhang in #6799)
- Stop validating props at mount time, only validate at element creation. (@keyanzhang in #6824)
- New invariant providing actionable error in missing instance case. (@yungsters in #6990)
- Add `React.PropTypes.symbol` to support ES2015 Symbols as props. (@puradox in #6377)
- Fix incorrect coercion of ref or key that are undefined in development (@gaearon in #6880)
- Fix a false positive when passing other element's props to cloneElement (@ericmatthys in #6268)
- Warn if you attempt to define `childContextTypes` on a functional component (@Aweary in #6933)

React DOM

- Add warning for unknown properties on DOM elements. (@jimfb in #6800, @gm758 in #7152)
- Properly remove attributes from custom elements. (@grassator in #6748)
- Fix invalid unicode escape in attribute name regular expression. (@nbjahan in #6772)

- Add `onLoad` handling to `<link>` element. (@roderickhsiao in #6815)
- Add `onError` handling to `<source>` element. (@wadahiro in #6941)
- Handle `value` and `defaultValue` more accurately in the DOM. (@jimfb in #6406)
- Fix events issue in environments with mutated `Object.prototype`. (@Weizenlol in #6886)
- Fix issue where `is="null"` ended up in the DOM in Firefox. (@darobin in #6896)
- Improved performance of text escaping by using escape-html. (@aickin in #6862)
- Fix issue with `dangerouslySetInnerHTML` and SVG in Internet Explorer. (@joshhunt in #6982)
- Fix issue with `<textarea>` placeholders. (@jimfb in #7002)
- Fix controlled vs uncontrolled detection of `<input type="radio"/>`. (@jimfb in #7003)
- Improve performance of updating text content. (@trueadm in #7005)
- Ensure controlled `<select>` components behave the same on initial render as they do on updates. (@yiminghe in #5362)

React Perf Add-on

- Add `isRunning()` API. (@nfcampos in #6763)
- Improve accuracy of lifecycle hook timing. (@gaearon in #6858)
- Fix internal errors when using ReactPerf with portal components. (@gaearon in #6860)
- Fix performance regression. (@sophiebits in #6770)
- Add warning that ReactPerf is not enabled in production. (@sashashakun in #6884)

React CSSTransitionGroup Add-on

- Fix timing issue with `null` node. (@keyanzhang in #6958)

React Native Renderer

- Dependencies on React Native modules use CommonJS requires instead of providesModule. (@davidaurelio in #6715)

15.1.0 (May 20, 2016)

React

- Ensure we're using the latest `object-assign`, which has protection against a non-spec-compliant native `Object.assign`. (@zpao in #6681)
- Add a new warning to communicate that `props` objects passed to `createElement` must be plain objects. (@richardscarrott in #6134)

- Fix a batching bug resulting in some lifecycle methods incorrectly being called multiple times. (@sophiebits in #6650)

React DOM

- Fix regression in custom elements support. (@jscissr in #6570)
- Stop incorrectly warning about using `onScroll` event handler with server rendering. (@Aweary in #6678)
- Fix grammar in the controlled input warning. (@jakeboone02 in #6657)
- Fix issue preventing `<object>` nodes from being able to read `<param>` nodes in IE. (@syranide in #6691)
- Fix issue resulting in crash when using experimental error boundaries with server rendering. (@jimfb in #6694)
- Add additional information to the controlled input warning. (@borisyankov in #6341)

React Perf Add-on

- Completely rewritten to collect data more accurately and to be easier to maintain. (@gaearon in #6647, #6046)

React Native Renderer

- Remove some special cases for platform specific branching. (@sebmarkbage in #6660)
- Remove use of `merge` utility. (@sebmarkbage in #6634)
- Renamed some modules to better indicate usage (@javache in #6643)

15.0.2 (April 29, 2016)

React

- Removed extraneous files from npm package. (@gaearon in #6388)
- Ensure `componentWillUnmount` is only called once. (@jimfb in #6613)

ReactDOM

- Fixed bug resulting in disabled buttons responding to mouse events in IE. (@nhunzaker in #6215)
- Ensure `<option>`s are correctly selected when inside `<optgroup>`. (@trevorsmith in #6442)
- Restore support for rendering into a shadow root. (@Wildhoney in #6462)
- Ensure nested `<body>` elements are caught when warning for invalid markup. (@keyanzhang in #6469)
- Improve warning when encountering multiple elements with the same key. (@hkal in #6500)

React TestUtils Add-on

- Ensure that functional components do not have an owner. (@gaearon in #6362)
- Handle invalid arguments to `scryRenderedDOMComponentsWithClass` better. (@ipeters90 in #6529)

React Perf Add-on

- Ignore DOM operations that occur outside the batch operation. (@gaearon in #6516)

React Native Renderer

- These files are now shipped inside the React npm package. They have no impact on React core or ReactDOM.

15.0.1 (April 8, 2016)

React

- Restore `React.__spread` API to unbreak code compiled with some tools making use of this undocumented API. It is now officially deprecated. (@zpao in #6444)

ReactDOM

- Fixed issue resulting in loss of cursor position in controlled inputs. (@sophiebits in #6449)

15.0.0 (April 7, 2016)

Major changes

- **Initial render now uses `document.createElement` instead of generating HTML.** Previously we would generate a large string of HTML and then set `node.innerHTML`. At the time, this was decided to be faster than using `document.createElement` for the majority of cases and browsers that we supported. Browsers have continued to improve and so overwhelmingly this is no longer true. By using `createElement` we can make other parts of React faster. (@sophiebits in #5205)
- **`data-reactid` is no longer on every node.** As a result of using `document.createElement`, we can prime the node cache as we create DOM nodes, allowing us to skip a potential lookup (which used the `data-reactid` attribute). Root nodes will have a `data-reactroot` attribute and server generated markup will still contain `data-reactid`. (@sophiebits in #5205)
- **No more extra `<span>`s.** ReactDOM will now render plain text nodes interspersed with comment nodes that are used for demarcation. This gives

us the same ability to update individual pieces of text, without creating extra nested nodes. If you were targeting these `<span>`s in your CSS, you will need to adjust accordingly. You can always render them explicitly in your components. (@mwiencek in #5753)

- **Rendering `null` now uses comment nodes.** Previously `null` would render to `<noscript>` elements. We now use comment nodes. This may cause issues if making use of `:nth-child` CSS selectors. While we consider this rendering behavior an implementation detail of React, it's worth noting the potential problem. (@sophiebits in #5451)
- **Functional components can now return `null`.** We added support for defining stateless components as functions in React 0.14. However, React 0.14 still allowed you to define a class component without extending `React.Component` or using `React.createClass()`, so we couldn't reliably tell if your component is a function or a class, and did not allow returning `null` from it. This issue is solved in React 15, and you can now return `null` from any component, whether it is a class or a function. (@jimfb in #5884)
- **Improved SVG support.** All SVG tags are now fully supported. (Uncommon SVG tags are not present on the `React.DOM` element helper, but JSX and `React.createElement` work on all tag names.) All SVG attributes that are implemented by the browsers should be supported too. If you find any attributes that we have missed, please let us know in this issue. (@zpao in #6243)

Breaking changes

- **No more extra `<span>`s.**
- **`React.cloneElement()` now resolves `defaultProps`.** We fixed a bug in `React.cloneElement()` that some components may rely on. If some of the `props` received by `cloneElement()` are `undefined`, it used to return an element with `undefined` values for those props. We're changing it to be consistent with `createElement()`. Now any `undefined` props passed to `cloneElement()` are resolved to the corresponding component's `defaultProps`. (@truongduy134 in #5997)
- **`ReactPerf.getLastMeasurements()` is opaque.** This change won't affect applications but may break some third-party tools. We are revamping `ReactPerf` implementation and plan to release it during the 15.x cycle. The internal performance measurement format is subject to change so, for the time being, we consider the return value of `ReactPerf.getLastMeasurements()` an opaque data structure that should not be relied upon. (@gaearon in #6286)

Removed deprecations These deprecations were introduced nine months ago in v0.14 with a warning and are removed:

- Deprecated APIs are removed from the `React` top-level export:

`findDOMNode`, `render`, `renderToString`, `renderToStaticMarkup`, and `unmountComponentAtNode`. As a reminder, they are now available on `ReactDOM` and `ReactDOMServer`. (@jimfb in #5832)
- Deprecated addons are removed: `batchedUpdates` and `cloneWithProps`. (@jimfb in #5859, @zpao in #6016)
- Deprecated component instance methods are removed: `setProps`, `replaceProps`, and `getDOMNode`. (@jimfb in #5570)
- Deprecated CommonJS `react/addons` entry point is removed. As a reminder, you should use separate `react-addons-*` packages instead. This only applies if you use the CommonJS builds. (@gaearon in #6285)
- Passing `children` to void elements like `<input>` was deprecated, and now throws an error. (@jonhester in #3372)
- React-specific properties on DOM `refs` (e.g. `this.refs.div.props`) were deprecated, and are removed now. (@jimfb in #5495)

New deprecations, introduced with a warning

Each of these changes will continue to work as before with a new warning until the release of React 16 so you can upgrade your code gradually.

- `LinkedStateMixin` and `valueLink` are now deprecated due to very low popularity. If you need this, you can use a wrapper component that implements the same behavior: react-linked-input. (@jimfb in #6127)
- Future versions of React will treat `<input value={null}>` as a request to clear the input. However, React 0.14 has been ignoring `value={null}`. React 15 warns you on a `null` input value and offers you to clarify your intention. To fix the warning, you may explicitly pass an empty string to clear a controlled input, or pass `undefined` to make the input uncontrolled. (@antoaravinth in #5048)
- `ReactPerf.printDOM()` was renamed to `ReactPerf.printOperations()`, and `ReactPerf.getMeasurementsSummaryMap()` was renamed to `ReactPerf.getWasted()`. (@gaearon in #6287)

New helpful warnings

- If you use a minified copy of the *development* build, React DOM kindly encourages you to use the faster production build instead. (@sophiebits in #5083)
- React DOM: When specifying a unit-less CSS value as a string, a future version will not add `px` automatically. This version now warns in this case (ex: writing `style={{width: '300'}}`. Unitless *number* values like `width: 300` are unchanged. (@pluma in #5140)
- Synthetic Events will now warn when setting and accessing properties (which will not get cleared appropriately), as well as warn on access after an event has been returned to the pool. (@kentcdodds in #5940 and @koba04 in #5947)

- Elements will now warn when attempting to read `ref` and `key` from the props. (@prometheansacrifice in #5744)
- React will now warn if you pass a different `props` object to `super()` in the constructor. (@prometheansacrifice in #5346)
- React will now warn if you call `setState()` inside `getChildContext()`. (@raineroviir in #6121)
- React DOM now attempts to warn for mistyped event handlers on DOM elements, such as `onclick` which should be `onClick`. (@ali in #5361)
- React DOM now warns about `NaN` values in `style`. (@jontewks in #5811)
- React DOM now warns if you specify both `value` and `defaultValue` for an input. (@mgmcdermott in #5823)
- React DOM now warns if an input switches between being controlled and uncontrolled. (@TheBlasfem in #5864)
- React DOM now warns if you specify `onFocusIn` or `onFocusOut` handlers as they are unnecessary in React. (@jontewks in #6296)
- React now prints a descriptive error message when you pass an invalid callback as the last argument to `ReactDOM.render()`, `this.setState()`, or `this.forceUpdate()`. (@conorhastings in #5193 and @gaearon in #6310)
- Add-Ons: `TestUtils.Simulate()` now prints a helpful message if you attempt to use it with shallow rendering. (@conorhastings in #5358)
- PropTypes: `arrayOf()` and `objectOf()` provide better error messages for invalid arguments. (@chicoxyzzy in #5390)

Notable bug fixes

- Fixed multiple small memory leaks. (@sophiebits in #4983 and @victorhomyakov in #6309)
- Input events are handled more reliably in IE 10 and IE 11; spurious events no longer fire when using a placeholder. (@jquense in #4051)
- The `componentWillReceiveProps()` lifecycle method is now consistently called when `context` changes. (@milesj in #5787)
- `React.cloneElement()` doesn't append slash to an existing `key` when used inside `React.Children.map()`. (@ianobermiller in #5892)
- React DOM now supports the `cite` and `profile` HTML attributes. (@AprilArcus in #6094 and @saiichihashimoto in #6032)
- React DOM now supports `cssFloat`, `gridRow` and `gridColumn` CSS properties. (@stevenvachon in #6133 and @mnordick in #4779)
- React DOM now correctly handles `borderImageOutset`, `borderImageWidth`, `borderImageSlice`, `floodOpacity`, `strokeDasharray`, and `strokeMiterlimit` as unitless CSS properties. (@rofrischmann in #6210 and #6270)
- React DOM now supports the `onAnimationStart`, `onAnimationEnd`, `onAnimationIteration`, `onTransitionEnd`, and `onInvalid` events. Support for `onLoad` has been added to `object` elements. (@tomduncalf in #5187, @milesj in #6005, and @ara4n in #5781)
- React DOM now defaults to using DOM attributes instead of properties,

which fixes a few edge case bugs. Additionally the nullification of values (ex: `href={null}`) now results in the forceful removal, no longer trying to set to the default value used by browsers in the absence of a value. (@syranide in #1510)
- React DOM does not mistakenly coerce `children` to strings for Web Components. (@jimfb in #5093)
- React DOM now correctly normalizes SVG `<use>` events. (@edmellum in #5720)
- React DOM does not throw if a `<select>` is unmounted while its `onChange` handler is executing. (@sambev in #6028)
- React DOM does not throw in Windows 8 apps. (@Andrew8xx8 in #6063)
- React DOM does not throw when asynchronously unmounting a child with a `ref`. (@yiminghe in #6095)
- React DOM no longer forces synchronous layout because of scroll position tracking. (@syranide in #2271)
- `Object.is` is used in a number of places to compare values, which leads to fewer false positives, especially involving `NaN`. In particular, this affects the `shallowCompare` add-on. (@chicoxyzzy in #6132)
- Add-Ons: ReactPerf no longer instruments adding or removing an event listener because they don't really touch the DOM due to event delegation. (@antoaravinth in #5209)

Other improvements

- React now uses `loose-envify` instead of `envify` so it installs fewer transitive dependencies. (@qerub in #6303)
- Shallow renderer now exposes `getMountedInstance()`. (@glenjamin in #4918)
- Shallow renderer now returns the rendered output from `render()`. (@simonewebdesign in #5411)
- React no longer depends on ES5 *shams* for `Object.create` and `Object.freeze` in older environments. It still, however, requires ES5 *shims* in those environments. (@dgreensp in #4959)
- React DOM now allows `data-` attributes with names that start with numbers. (@nLight in #5216)
- React DOM adds a new `suppressContentEditableWarning` prop for components like Draft.js that intentionally manage `contentEditable` children with React. (@mxstbr in #6112)
- React improves the performance for `createClass()` on complex specs. (@sophiebits in #5550)

0.14.10 (October 14, 2020)

React

- Backport support for the new JSX transform to 0.14.x. (@lunaruan in #18299 and @gaearon in #20024)

0.14.8 (March 29, 2016)

React

- Fixed memory leak when rendering on the server

0.14.7 (January 28, 2016)

React

- Fixed bug with `<option>` tags when using `dangerouslySetInnerHTML`
- Fixed memory leak in synthetic event system

React TestUtils Add-on

- Fixed bug with calling `setState` in `componentWillMount` when using shallow rendering

0.14.6 (January 6, 2016)

React

- Updated `fbjs` dependency to pick up change affecting handling of undefined document.

0.14.5 (December 29, 2015)

React

- More minor internal changes for better compatibility with React Native

0.14.4 (December 29, 2015)

React

- Minor internal changes for better compatibility with React Native

React DOM

- The `autoCapitalize` and `autoCorrect` props are now set as attributes in the DOM instead of properties to improve cross-browser compatibility
- Fixed bug with controlled `<select>` elements not handling updates properly

React Perf Add-on

- Some DOM operation names have been updated for clarity in the output of `.printDOM()`

0.14.3 (November 18, 2015)

React DOM

- Added support for `nonce` attribute for `<script>` and `<style>` elements
- Added support for `reversed` attribute for `<ol>` elements

React TestUtils Add-on

- Fixed bug with shallow rendering and function refs

React CSSTransitionGroup Add-on

- Fixed bug resulting in timeouts firing incorrectly when mounting and unmounting rapidly

React on Bower

- Added `react-dom-server.js` to expose `renderToString` and `renderToStaticMarkup` for usage in the browser

0.14.2 (November 2, 2015)

React DOM

- Fixed bug with development build preventing events from firing in some versions of Internet Explorer & Edge
- Fixed bug with development build when using es5-sham in older versions of Internet Explorer
- Added support for `integrity` attribute
- Fixed bug resulting in `children` prop being coerced to a string for custom elements, which was not the desired behavior
- Moved `react` from `dependencies` to `peerDependencies` to match expectations and align with `react-addons-*` packages

0.14.1 (October 28, 2015)

React DOM

- Fixed bug where events wouldn't fire in old browsers when using React in development mode
- Fixed bug preventing use of `dangerouslySetInnerHTML` with Closure Compiler Advanced mode
- Added support for `srcLang`, `default`, and `kind` attributes for `<track>` elements
- Added support for `color` attribute
- Ensured legacy `.props` access on DOM nodes is updated on re-renders

React TestUtils Add-on

- Fixed `scryRenderedDOMComponentsWithClass` so it works with SVG

React CSSTransitionGroup Add-on

- Fix bug preventing `0` to be used as a timeout value

React on Bower

- Added `react-dom.js` to `main` to improve compatibility with tooling

0.14.0 (October 7, 2015)

Major changes

- Split the main `react` package into two: `react` and `react-dom`. This paves the way to writing components that can be shared between the web version of React and React Native. This means you will need to include both files and some functions have been moved from `React` to `ReactDOM`.
- Addons have been moved to separate packages (`react-addons-clone-with-props`, `react-addons-create-fragment`, `react-addons-css-transition-group`, `react-addons-linked-state-mixin`, `react-addons-perf`, `react-addons-pure-render-mixin`, `react-addons-shallow-compare`, `react-addons-test-utils`, `react-addons-transition-group`, `react-addons-update`, `ReactDOM.unstable_batchedUpdates`).
- Stateless functional components - React components were previously created using React.createClass or using ES6 classes. This release adds a new syntax where a user defines a single stateless render function (with one parameter: `props`) which returns a JSX element, and this function may be used as a component.
- Refs to DOM components as the DOM node itself. Previously the only useful thing you can do with a DOM component is call `getDOMNode()` to get the underlying DOM node. Starting with this release, a ref to a DOM component *is* the actual DOM node. **Note that refs to custom (user-defined) components work exactly as before; only the built-in DOM components are affected by this change.**

Breaking changes

- `React.initializeTouchEvents` is no longer necessary and has been removed completely. Touch events now work automatically.
- Add-Ons: Due to the DOM node refs change mentioned above, `TestUtils.findAllInRenderedTree` and related helpers are no longer able to take a DOM component, only a custom component.
- The `props` object is now frozen, so mutating props after creating a component element is no longer supported. In most cases, `React.cloneElement` should be used instead. This change makes your components easier to reason about and enables the compiler optimizations mentioned above.

- Plain objects are no longer supported as React children; arrays should be used instead. You can use the `createFragment` helper to migrate, which now returns an array.
- Add-Ons: `classSet` has been removed. Use classnames instead.
- Web components (custom elements) now use native property names. Eg: `class` instead of `className`.

Deprecations

- `this.getDOMNode()` is now deprecated and `ReactDOM.findDOMNode(this)` can be used instead. Note that in the common case, `findDOMNode` is now unnecessary since a ref to the DOM component is now the actual DOM node.
- `setProps` and `replaceProps` are now deprecated. Instead, call ReactDOM.render again at the top level with the new props.
- ES6 component classes must now extend `React.Component` in order to enable stateless function components. The ES3 module pattern will continue to work.
- Reusing and mutating a `style` object between renders has been deprecated. This mirrors our change to freeze the `props` object.
- Add-Ons: `cloneWithProps` is now deprecated. Use `React.cloneElement` instead (unlike `cloneWithProps`, `cloneElement` does not merge `className` or `style` automatically; you can merge them manually if needed).
- Add-Ons: To improve reliability, `CSSTransitionGroup` will no longer listen to transition events. Instead, you should specify transition durations manually using props such as `transitionEnterTimeout={500}`.

Notable enhancements

- Added `React.Children.toArray` which takes a nested children object and returns a flat array with keys assigned to each child. This helper makes it easier to manipulate collections of children in your `render` methods, especially if you want to reorder or slice `this.props.children` before passing it down. In addition, `React.Children.map` now returns plain arrays too.
- React uses `console.error` instead of `console.warn` for warnings so that browsers show a full stack trace in the console. (Our warnings appear when you use patterns that will break in future releases and for code that is likely to behave unexpectedly, so we do consider our warnings to be "must-fix" errors.)
- Previously, including untrusted objects as React children could result in an XSS security vulnerability. This problem should be avoided by properly validating input at the application layer and by never passing untrusted objects around your application code. As an additional layer of protection, React now tags elements with a specific ES2015 (ES6) `Symbol` in browsers

that support it, in order to ensure that React never considers untrusted JSON to be a valid element. If this extra security protection is important to you, you should add a `Symbol` polyfill for older browsers, such as the one included by Babel's polyfill.
- When possible, React DOM now generates XHTML-compatible markup.
- React DOM now supports these standard HTML attributes: `capture`, `challenge`, `inputMode`, `is`, `keyParams`, `keyType`, `minLength`, `summary`, `wrap`. It also now supports these non-standard attributes: `autoSave`, `results`, `security`.
- React DOM now supports these SVG attributes, which render into namespaced attributes: `xlinkActuate`, `xlinkArcrole`, `xlinkHref`, `xlinkRole`, `xlinkShow`, `xlinkTitle`, `xlinkType`, `xmlBase`, `xmlLang`, `xmlSpace`.
- The `image` SVG tag is now supported by React DOM.
- In React DOM, arbitrary attributes are supported on custom elements (those with a hyphen in the tag name or an `is="..."` attribute).
- React DOM now supports these media events on `audio` and `video` tags: `onAbort`, `onCanPlay`, `onCanPlayThrough`, `onDurationChange`, `onEmptied`, `onEncrypted`, `onEnded`, `onError`, `onLoadedData`, `onLoadedMetadata`, `onLoadStart`, `onPause`, `onPlay`, `onPlaying`, `onProgress`, `onRateChange`, `onSeeked`, `onSeeking`, `onStalled`, `onSuspend`, `onTimeUpdate`, `onVolumeChange`, `onWaiting`.
- Many small performance improvements have been made.
- Many warnings show more context than before.
- Add-Ons: A `shallowCompare` add-on has been added as a migration path for `PureRenderMixin` in ES6 classes.
- Add-Ons: `CSSTransitionGroup` can now use custom class names instead of appending `-enter-active` or similar to the transition name.

New helpful warnings

- React DOM now warns you when nesting HTML elements invalidly, which helps you avoid surprising errors during updates.
- Passing `document.body` directly as the container to `ReactDOM.render` now gives a warning as doing so can cause problems with browser extensions that modify the DOM.
- Using multiple instances of React together is not supported, so we now warn when we detect this case to help you avoid running into the resulting problems.

Notable bug fixes

- Click events are handled by React DOM more reliably in mobile browsers, particularly in Mobile Safari.
- SVG elements are created with the correct namespace in more cases.
- React DOM now renders `<option>` elements with multiple text children properly and renders `<select>` elements on the server with the correct

option selected.
- When two separate copies of React add nodes to the same document (including when a browser extension uses React), React DOM tries harder not to throw exceptions during event handling.
- Using non-lowercase HTML tag names in React DOM (e.g., `React.createElement('DIV')`) no longer causes problems, though we continue to recommend lowercase for consistency with the JSX tag name convention (lowercase names refer to built-in components, capitalized names refer to custom components).
- React DOM understands that these CSS properties are unitless and does not append "px" to their values: `animationIterationCount`, `boxOrdinalGroup`, `flexOrder`, `tabSize`, `stopOpacity`.
- Add-Ons: When using the test utils, `Simulate.mouseEnter` and `Simulate.mouseLeave` now work.
- Add-Ons: ReactTransitionGroup now correctly handles multiple nodes being removed simultaneously.

React Tools / Babel

Breaking Changes

- The `react-tools` package and `JSXTransformer.js` browser file have been deprecated. You can continue using version `0.13.3` of both, but we no longer support them and recommend migrating to Babel, which has built-in support for React and JSX.

New Features

- Babel 5.8.24 introduces **Inlining React elements:** The `optimisation.react.inlineElements` transform converts JSX elements to object literals like `{type: 'div', props: ...}` instead of calls to `React.createElement`. This should only be enabled in production, since it disables some development warnings/checks.
- Babel 5.8.24 introduces **Constant hoisting for React elements:** The `optimisation.react.constantElements` transform hoists element creation to the top level for subtrees that are fully static, which reduces calls to `React.createElement` and the resulting allocations. More importantly, it tells React that the subtree hasn't changed so React can completely skip it when reconciling. This should only be enabled in production, since it disables some development warnings/checks.

0.13.3 (May 8, 2015)

React Core

New Features

- Added `clipPath` element and attribute for SVG

- Improved warnings for deprecated methods in plain JS classes

Bug Fixes

- Loosened `dangerouslySetInnerHTML` restrictions so `{__html: undefined}` will no longer throw
- Fixed extraneous context warning with non-pure `getChildContext`
- Ensure `replaceState(obj)` retains prototype of `obj`

React with Add-ons

Bug Fixes

- Test Utils: Ensure that shallow rendering works when components define `contextTypes`

0.13.2 (April 18, 2015)

React Core

New Features

- Added `strokeDashoffset`, `flexPositive`, `flexNegative` to the list of unitless CSS properties
- Added support for more DOM properties:
  - `scoped` - for `<style>` elements
  - `high`, `low`, `optimum` - for `<meter>` elements
  - `unselectable` - IE-specific property to prevent user selection

Bug Fixes

- Fixed a case where re-rendering after rendering null didn't properly pass context
- Fixed a case where re-rendering after rendering with `style={null}` didn't properly update `style`
- Update `uglify` dependency to prevent a bug in IE8
- Improved warnings

React with Add-Ons

Bug Fixes

- Immutability Helpers: Ensure it supports `hasOwnProperty` as an object key

React Tools

- Improve documentation for new options

0.13.1 (March 16, 2015)

React Core

Bug Fixes

- Don't throw when rendering empty `<select>` elements
- Ensure updating `style` works when transitioning from `null`

React with Add-Ons

Bug Fixes

- TestUtils: Don't warn about `getDOMNode` for ES6 classes
- TestUtils: Ensure wrapped full page components (`<html>`, `<head>`, `<body>`) are treated as DOM components
- Perf: Stop double-counting DOM components

React Tools

Bug Fixes

- Fix option parsing for `--non-strict-es6module`

0.13.0 (March 10, 2015)

React Core

Breaking Changes

- Deprecated patterns that warned in 0.12 no longer work: most prominently, calling component classes without using JSX or React.createElement and using non-component functions with JSX or createElement
- Mutating `props` after an element is created is deprecated and will cause warnings in development mode; future versions of React will incorporate performance optimizations assuming that props aren't mutated
- Static methods (defined in `statics`) are no longer autobound to the component class
- `ref` resolution order has changed slightly such that a ref to a component is available immediately after its `componentDidMount` method is called; this change should be observable only if your component calls a parent component's callback within your `componentDidMount`, which is an anti-pattern and should be avoided regardless
- Calls to `setState` in life-cycle methods are now always batched and therefore asynchronous. Previously the first call on the first mount was synchronous.
- `setState` and `forceUpdate` on an unmounted component now warns instead of throwing. That avoids a possible race condition with Promises.
- Access to most internal properties has been completely removed, including `this._pendingState` and `this._rootNodeID`.

New Features

- Support for using ES6 classes to build React components; see the v0.13.0 beta 1 notes for details.
- Added new top-level API `React.findDOMNode(component)`, which should be used in place of `component.getDOMNode()`. The base class for ES6-based components will not have `getDOMNode`. This change will enable some more patterns moving forward.
- Added a new top-level API `React.cloneElement(el, props)` for making copies of React elements – see the v0.13 RC2 notes for more details.
- New `ref` style, allowing a callback to be used in place of a name: `<Photo ref={(c) => this._photo = c} />` allows you to reference the component with `this._photo` (as opposed to `ref="photo"` which gives `this.refs.photo`).
- `this.setState()` can now take a function as the first argument for transactional state updates, such as `this.setState((state, props) => ({count: state.count + 1}));` – this means that you no longer need to use `this._pendingState`, which is now gone.
- Support for iterators and immutable-js sequences as children.

Deprecations

- `ComponentClass.type` is deprecated. Just use `ComponentClass` (usually as `element.type === ComponentClass`).
- Some methods that are available on `createClass`-based components are removed or deprecated from ES6 classes (`getDOMNode`, `replaceState`, `isMounted`, `setProps`, `replaceProps`).

React with Add-Ons

New Features

- `React.addons.createFragment` was added for adding keys to entire sets of children.

Deprecations

- `React.addons.classSet` is now deprecated. This functionality can be replaced with several freely available modules. classnames is one such module.
- Calls to `React.addons.cloneWithProps` can be migrated to use `React.cloneElement` instead – make sure to merge `style` and `className` manually if desired.

React Tools

Breaking Changes

- When transforming ES6 syntax, `class` methods are no longer enumerable by default, which requires `Object.defineProperty`; if you support browsers such as IE8, you can pass `--target es3` to mirror the old behavior

New Features

- `--target` option is available on the jsx command, allowing users to specify and ECMAScript version to target.
  - `es5` is the default.
  - `es3` restores the previous default behavior. An additional transform is added here to ensure the use of reserved words as properties is safe (eg `this.static` will become `this['static']` for IE8 compatibility).
- The transform for the call spread operator has also been enabled.

JSXTransformer

Breaking Changes

- The return value of `transform` now contains `sourceMap` as a JS object already, not an instance of `SourceMapGenerator`.

JSX

Breaking Changes

- A change was made to how some JSX was parsed, specifically around the use of `>` or `}` when inside an element. Previously it would be treated as a string but now it will be treated as a parse error. The `jsx_orphaned_brackets_transformer` package on npm can be used to find and fix potential issues in your JSX code.

0.12.2 (December 18, 2014)

React Core

- Added support for more HTML attributes: `formAction`, `formEncType`, `formMethod`, `formTarget`, `marginHeight`, `marginWidth`
- Added `strokeOpacity` to the list of unitless CSS properties
- Removed trailing commas (allows npm module to be bundled and used in IE8)
- Fixed bug resulting in error when passing `undefined` to `React.createElement` - now there is a useful warning

React Tools

- JSX-related transforms now always use double quotes for props and `displayName`

0.12.1 (November 18, 2014)

React Tools

- Types transform updated with latest support
- jstransform version updated with improved ES6 transforms
- Explicit Esprima dependency removed in favor of using Esprima information exported by jstransform

0.12.0 (October 28, 2014)

React Core

Breaking Changes

- `key` and `ref` moved off props object, now accessible on the element directly
- React is now BSD licensed with accompanying Patents grant
- Default prop resolution has moved to Element creation time instead of mount time, making them effectively static
- `React.__internals` is removed - it was exposed for DevTools which no longer needs access
- Composite Component functions can no longer be called directly - they must be wrapped with `React.createFactory` first. This is handled for you when using JSX.

New Features

- Spread operator (`{...}`) introduced to deprecate `this.transferPropsTo`
- Added support for more HTML attributes: `acceptCharset`, `classID`, `manifest`

Deprecations

- `React.renderComponent` –> `React.render`
- `React.renderComponentToString` –> `React.renderToString`
- `React.renderComponentToStaticMarkup` –> `React.renderToStaticMarkup`
- `React.isValidComponent` –> `React.isValidElement`
- `React.PropTypes.component` –> `React.PropTypes.element`
- `React.PropTypes.renderable` –> `React.PropTypes.node`
- **DEPRECATED** `React.isValidClass`
- **DEPRECATED** `instance.transferPropsTo`
- **DEPRECATED** Returning `false` from event handlers to preventDefault
- **DEPRECATED** Convenience Constructor usage as function, instead wrap with `React.createFactory`
- **DEPRECATED** use of `key={null}` to assign implicit keys

Bug Fixes

- Better handling of events and updates in nested results, fixing value restoration in "layered" controlled components
- Correctly treat `event.getModifierState` as case sensitive
- Improved normalization of `event.charCode`
- Better error stacks when involving autobound methods
- Removed DevTools message when the DevTools are installed
- Correctly detect required language features across browsers
- Fixed support for some HTML attributes:
  - `list` updates correctly now
  - `scrollLeft`, `scrollTop` removed, these should not be specified as props
- Improved error messages

React With Addons

New Features

- `React.addons.batchedUpdates` added to API for hooking into update cycle

Breaking Changes

- `React.addons.update` uses `assign` instead of `copyProperties` which does `hasOwnProperty` checks. Properties on prototypes will no longer be updated correctly.

Bug Fixes

- Fixed some issues with CSS Transitions

JSX

Breaking Changes

- Enforced convention: lower case tag names are always treated as HTML tags, upper case tag names are always treated as composite components
- JSX no longer transforms to simple function calls

New Features

- `@jsx React.DOM` no longer required
- spread (`{...}`) operator introduced to allow easier use of props

Bug Fixes

- JSXTransformer: Make sourcemaps an option when using APIs directly (eg, for react-rails)

0.11.2 (September 16, 2014)

React Core

New Features

- Added support for `<dialog>` element and associated `open` attribute
- Added support for `<picture>` element and associated `media` and `sizes` attributes
- Added `React.createElement` API in preparation for React v0.12
  - `React.createDescriptor` has been deprecated as a result

JSX

- `<picture>` is now parsed into `React.DOM.picture`

React Tools

- Update `esprima` and `jstransform` for correctness fixes
- The `jsx` executable now exposes a `--strip-types` flag which can be used to remove TypeScript-like type annotations
  - This option is also exposed to `require('react-tools').transform` as `stripTypes`

0.11.1 (July 24, 2014)

React Core

Bug Fixes

- `setState` can be called inside `componentWillMount` in non-DOM environments
- `SyntheticMouseEvent.getEventModifierState` correctly renamed to `getModifierState`
- `getModifierState` correctly returns a `boolean`
- `getModifierState` is now correctly case sensitive
- Empty Text node used in IE8 `innerHTML` workaround is now removed, fixing rerendering in certain cases

JSX

- Fix duplicate variable declaration in JSXTransformer (caused issues in some browsers)

0.11.0 (July 17, 2014)

React Core

Breaking Changes

- `getDefaultProps()` is now called once per class and shared across all instances
- `MyComponent()` now returns a descriptor, not an instance
- `React.isValidComponent` and `React.PropTypes.component` validate *descriptors*, not component instances
- Custom `propType` validators should return an `Error` instead of logging directly

New Features

- Rendering to `null`
- Keyboard events include normalized `e.key` and `e.getModifierState()` properties
- New normalized `onBeforeInput` event
- `React.Children.count` has been added as a helper for counting the number of children

Bug Fixes

- Re-renders are batched in more cases
- Events: `e.view` properly normalized
- Added Support for more HTML attributes (`coords`, `crossOrigin`, `download`, `hrefLang`, `mediaGroup`, `muted`, `scrolling`, `shape`, `srcSet`, `start`, `useMap`)
- Improved SVG support
  - Changing `className` on a mounted SVG component now works correctly
  - Added support for elements `mask` and `tspan`
  - Added support for attributes `dx`, `dy`, `fillOpacity`, `fontFamily`, `fontSize`, `markerEnd`, `markerMid`, `markerStart`, `opacity`, `patternContentUnits`, `patternUnits`, `preserveAspectRatio`, `strokeDasharray`, `strokeOpacity`
- CSS property names with vendor prefixes (`Webkit`, `ms`, `Moz`, `O`) are now handled properly
- Duplicate keys no longer cause a hard error; now a warning is logged (and only one of the children with the same key is shown)
- `img` event listeners are now unbound properly, preventing the error "Two valid but unequal nodes with the same `data-reactid`"
- Added explicit warning when missing polyfills

React With Addons

- PureRenderMixin: a mixin which helps optimize "pure" components
- Perf: a new set of tools to help with performance analysis
- Update: New `$apply` command to transform values
- TransitionGroup bug fixes with null elements, Android

React NPM Module

- Now includes the pre-built packages under `dist/`.
- `envify` is properly listed as a dependency instead of a peer dependency

JSX

- Added support for namespaces, eg `<Components.Checkbox />`
- JSXTransformer
  - Enable the same `harmony` features available in the command line with `<script type="text/jsx;harmony=true">`
  - Scripts are downloaded in parallel for more speed. They are still executed in order (as you would expect with normal script tags)
  - Fixed a bug preventing sourcemaps from working in Firefox

React Tools Module

- Improved readme with usage and API information
- Improved ES6 transforms available with `--harmony` option
- Added `--source-map-inline` option to the `jsx` executable
- New `transformWithDetails` API which gives access to the raw sourcemap data

0.10.0 (March 21, 2014)

React Core

New Features

- Added warnings to help migrate towards descriptors
- Made it possible to server render without React-related markup (`data-reactid`, `data-react-checksum`). This DOM will not be mountable by React. Read the docs for `React.renderComponentToStaticMarkup`
- Added support for more attributes:
  - `srcSet` for `<img>` to specify images at different pixel ratios
  - `textAnchor` for SVG

Bug Fixes

- Ensure all void elements don't insert a closing tag into the markup.
- Ensure `className={false}` behaves consistently
- Ensure `this.refs` is defined, even if no refs are specified.

Addons

- `update` function to deal with immutable data. Read the docs

react-tools

- Added an option argument to `transform` function. The only option supported is `harmony`, which behaves the same as `jsx --harmony` on the command line. This uses the ES6 transforms from jstransform.

0.9.0 (February 20, 2014)

React Core

Breaking Changes

- The lifecycle methods `componentDidMount` and `componentDidUpdate` no longer receive the root node as a parameter; use `this.getDOMNode()` instead
- Whenever a prop is equal to `undefined`, the default value returned by `getDefaultProps` will now be used instead
- `React.unmountAndReleaseReactRootNode` was previously deprecated and has now been removed
- `React.renderComponentToString` is now synchronous and returns the generated HTML string
- Full-page rendering (that is, rendering the `<html>` tag using React) is now supported only when starting with server-rendered markup
- On mouse wheel events, `deltaY` is no longer negated
- When prop types validation fails, a warning is logged instead of an error thrown (with the production build of React, type checks are now skipped for performance)
- On `input`, `select`, and `textarea` elements, `.getValue()` is no longer supported; use `.getDOMNode().value` instead
- `this.context` on components is now reserved for internal use by React

New Features

- React now never rethrows errors, so stack traces are more accurate and Chrome's purple break-on-error stop sign now works properly
- Added support for SVG tags `defs`, `linearGradient`, `polygon`, `radialGradient`, `stop`
- Added support for more attributes:
  - `crossOrigin` for CORS requests
  - `download` and `hrefLang` for `<a>` tags
  - `mediaGroup` and `muted` for `<audio>` and `<video>` tags
  - `noValidate` and `formNoValidate` for forms
  - `property` for Open Graph `<meta>` tags
  - `sandbox`, `seamless`, and `srcDoc` for `<iframe>` tags
  - `scope` for screen readers
  - `span` for `<colgroup>` tags
- Added support for defining `propTypes` in mixins

- Added `any`, `arrayOf`, `component`, `oneOfType`, `renderable`, `shape` to `React.PropTypes`
- Added support for `statics` on component spec for static component methods
- On all events, `.currentTarget` is now properly set
- On keyboard events, `.key` is now polyfilled in all browsers for special (non-printable) keys
- On clipboard events, `.clipboardData` is now polyfilled in IE
- On drag events, `.dragTransfer` is now present
- Added support for `onMouseOver` and `onMouseOut` in addition to the existing `onMouseEnter` and `onMouseLeave` events
- Added support for `onLoad` and `onError` on `<img>` elements
- Added support for `onReset` on `<form>` elements
- The `autoFocus` attribute is now polyfilled consistently on `input`, `select`, and `textarea`

Bug Fixes

- React no longer adds an `__owner__` property to each component's `props` object; passed-in props are now never mutated
- When nesting top-level components (e.g., calling `React.renderComponent` within `componentDidMount`), events now properly bubble to the parent component
- Fixed a case where nesting top-level components would throw an error when updating
- Passing an invalid or misspelled propTypes type now throws an error
- On mouse enter/leave events, `.target`, `.relatedTarget`, and `.type` are now set properly
- On composition events, `.data` is now properly normalized in IE9 and IE10
- CSS property values no longer have `px` appended for the unitless properties `columnCount`, `flex`, `flexGrow`, `flexShrink`, `lineClamp`, `order`, `widows`
- Fixed a memory leak when unmounting children with a `componentWillUnmount` handler
- Fixed a memory leak when `renderComponentToString` would store event handlers
- Fixed an error that could be thrown when removing form elements during a click handler
- Boolean attributes such as `disabled` are rendered without a value (previously `disabled="true"`, now simply `disabled`)
- `key` values containing `.` are now supported
- Shortened `data-reactid` values for performance
- Components now always remount when the `key` property changes
- Event handlers are attached to `document` only when necessary, improving performance in some cases
- Events no longer use `.returnValue` in modern browsers, eliminating a warning in Chrome

- `scrollLeft` and `scrollTop` are no longer accessed on document.body, eliminating a warning in Chrome
- General performance fixes, memory optimizations, improvements to warnings and error messages

React with Addons

- `React.addons.TestUtils` was added to help write unit tests
- `React.addons.TransitionGroup` was renamed to `React.addons.CSSTransitionGroup`
- `React.addons.TransitionGroup` was added as a more general animation wrapper
- `React.addons.cloneWithProps` was added for cloning components and modifying their props
- Bug fix for adding back nodes during an exit transition for CSSTransitionGroup
- Bug fix for changing `transitionLeave` in CSSTransitionGroup
- Performance optimizations for CSSTransitionGroup
- On checkbox `<input>` elements, `checkedLink` is now supported for two-way binding

JSX Compiler and react-tools Package

- Whitespace normalization has changed; now space between two tags on the same line will be preserved, while newlines between two tags will be removed
- The `react-tools` npm package no longer includes the React core libraries; use the `react` package instead.
- `displayName` is now added in more cases, improving error messages and names in the React Dev Tools
- Fixed an issue where an invalid token error was thrown after a JSX closing tag
- `JSXTransformer` now uses source maps automatically in modern browsers
- `JSXTransformer` error messages now include the filename and problematic line contents when a file fails to parse

0.8.0 (December 19, 2013)

React

- Added support for more attributes:
  - `rows` & `cols` for `<textarea>`
  - `defer` & `async` for `<script>`
  - `loop` for `<audio>` & `<video>`
  - `autoCorrect` for form fields (a non-standard attribute only supported by mobile WebKit)
- Improved error messages
- Fixed Selection events in IE11

- Added `onContextMenu` events

React with Addons

- Fixed bugs with TransitionGroup when children were undefined
- Added support for `onTransition`

react-tools

- Upgraded `jstransform` and `esprima-fb`

JSXTransformer

- Added support for use in IE8
- Upgraded browserify, which reduced file size by ~65KB (16KB gzipped)

0.5.2, 0.4.2 (December 18, 2013)

React

- Fixed a potential XSS vulnerability when using user content as a `key`: CVE-2013-7035

0.5.1 (October 29, 2013)

React

- Fixed bug with `<input type="range">` and selection events.
- Fixed bug with selection and focus.
- Made it possible to unmount components from the document root.
- Fixed bug for `disabled` attribute handling on non-`<input>` elements.

React with Addons

- Fixed bug with transition and animation event detection.

0.5.0 (October 16, 2013)

React

- Memory usage improvements - reduced allocations in core which will help with GC pauses
- Performance improvements - in addition to speeding things up, we made some tweaks to stay out of slow path code in V8 and Nitro.
- Standardized prop -> DOM attribute process. This previously resulting in additional type checking and overhead as well as confusing cases for users. Now we will always convert your value to a string before inserting it into the DOM.
- Support for Selection events.

- Support for Composition events.
- Support for additional DOM properties (`charSet`, `content`, `form`, `httpEquiv`, `rowSpan`, `autoCapitalize`).
- Support for additional SVG properties (`rx`, `ry`).
- Support for using `getInitialState` and `getDefaultProps` in mixins.
- Support mounting into iframes.
- Bug fixes for controlled form components.
- Bug fixes for SVG element creation.
- Added `React.version`.
- Added `React.isValidClass` - Used to determine if a value is a valid component constructor.
- Removed `React.autoBind` - This was deprecated in v0.4 and now properly removed.
- Renamed `React.unmountAndReleaseReactRootNode` to `React.unmountComponentAtNode`.
- Began laying down work for refined performance analysis.
- Better support for server-side rendering - react-page has helped improve the stability for server-side rendering.
- Made it possible to use React in environments enforcing a strict Content Security Policy. This also makes it possible to use React to build Chrome extensions.

React with Addons (New!)

- Introduced a separate build with several "addons" which we think can help improve the React experience. We plan to deprecate this in the long-term, instead shipping each as standalone pieces. Read more in the docs.

JSX

- No longer transform `class` to `className` as part of the transform! This is a breaking change - if you were using `class`, you *must* change this to `className` or your components will be visually broken.
- Added warnings to the in-browser transformer to make it clear it is not intended for production use.
- Improved compatibility for Windows
- Improved support for maintaining line numbers when transforming.

0.4.1 (July 26, 2013)

React

- `setState` callbacks are now executed in the scope of your component.
- `click` events now work on Mobile Safari.
- Prevent a potential error in event handling if `Object.prototype` is extended.
- Don't set DOM attributes to the string `"undefined"` on update when previously defined.

- Improved support for `<iframe>` attributes.
- Added checksums to detect and correct cases where server-side rendering markup mismatches what React expects client-side.

JSXTransformer

- Improved environment detection so it can be run in a non-browser environment.

0.4.0 (July 17, 2013)

React

- Switch from using `id` attribute to `data-reactid` to track DOM nodes. This allows you to integrate with other JS and CSS libraries more easily.
- Support for more DOM elements and attributes (e.g., `<canvas>`)
- Improved server-side rendering APIs. `React.renderComponentToString(<component>, callback)` allows you to use React on the server and generate markup which can be sent down to the browser.
- `prop` improvements: validation and default values. Read our blog post for details...
- Support for the `key` prop, which allows for finer control over reconciliation. Read the docs for details...
- Removed `React.autoBind`. Read our blog post for details...
- Improvements to forms. We've written wrappers around `<input>`, `<textarea>`, `<option>`, and `<select>` in order to standardize many inconsistencies in browser implementations. This includes support for `defaultValue`, and improved implementation of the `onChange` event, and circuit completion. Read the docs for details...
- We've implemented an improved synthetic event system that conforms to the W3C spec.
- Updates to your component are batched now, which may result in a significantly faster re-render of components. `this.setState` now takes an optional callback as it's second parameter. If you were using `onClick={this.setState.bind(this, state)}` previously, you'll want to make sure you add a third parameter so that the event is not treated as the callback.

JSX

- Support for comment nodes `<div>{/* this is a comment and won't be rendered */}</div>`
- Children are now transformed directly into arguments instead of being wrapped in an array E.g. `<div><Component1/><Component2/></div>` is transformed into `React.DOM.div(null, Component1(null), Component2(null))`. Previously this would be transformed into

`React.DOM.div(null, [Component1(null), Component2(null)])`. If you were using React without JSX previously, your code should still work.

react-tools

- Fixed a number of bugs when transforming directories
- No longer re-write `require()`s to be relative unless specified

0.3.3 (June 20, 2013)

React

- Allow reusing the same DOM node to render different components. e.g. `React.renderComponent(<div/>, domNode); React.renderComponent(<span/>, domNode);` will work now.

JSX

- Improved the in-browser transformer so that transformed scripts will execute in the expected scope. The allows components to be defined and used from separate files.

react-tools

- Upgrade Commoner so `require` statements are no longer relativized when passing through the transformer. This was a feature needed when building React, but doesn't translate well for other consumers of `bin/jsx`.
- Upgraded our dependencies on Commoner and Recast so they use a different directory for their cache.
- Freeze our Esprima dependency.

0.3.2 (May 31, 2013)

JSX

- Improved compatibility with other coding styles (specifically, multiple assignments with a single `var`).

react-tools

- Switch from using the browserified build to shipping individual modules. This allows react-tools to be used with browserify.

0.3.1 (May 30, 2013)

react-tools

- Fix bug in packaging resulting in broken module.

0.3.0 (May 29, 2013)

- Initial public release