



Keynote

- > The goal of React has always been to easily build a great User Experience.
 - ✓ Fast startup
 - ✓ Responsive interactions
 - ✓ Delightful extras and polish
- > To make it dramatically easier, we need to focus on the Developer Experience.
 - ✓ Low barrier to entry for basic features and sophisticated UI
 - ✓ High productivity
 - ✓ Ability to scale
- > Agenda
 - 1 State of React
 - 2 React 19
 - 3 What's Next

State of React

- ✓ Over 1 billion times npm downloads/year
- ✓ 4 million React Developer Tools users
- ✓ Started to recommend react frameworks to start an app: Remix, NextJS, RedwoodJS, Expo

React 19

- ✓ React Server Components make declarative, composable UI work seamlessly across client and server
- ✓ Server Rendering & Suspense

- Declarative, composable HTML, Metadata, Scripts, Stylesheets

Document Metadata can float on up....

```
function MyComponent() {  
  return (  
    <>  
      <title>Story Time!</title>  
      <link rel="author" href="..." />  
      <meta  
        name="viewport"  
        content="..."  
      />  
      <div>...</div>  
    </>  
  );  
}
```

```
<!DOCTYPE html>  
<html>  
  <head>  
    <title>Story Time!</title>  
    <link rel="author" href="...">  
    <meta name="viewport" content="...">  
  </head>  
  <body>  
    ...  
    <div>...</div>  
    ...  
  </body>  
</html>
```

✓ Hydration

- Better error reporting: visualize diffs on the props instead of multiple errors

React 19 🎉

```
<App>  
  <div className="App">  
    <main data-extra="prop">  
      <h1>  
        <MyComponent>  
          <span  
            + data-prop="client"  
            - data-prop="server"  
          >  
            Hello from the client  
          + <article>  
          - <div>
```

✓ Actions

- React Actions are not exclusive to Server Components frameworks - they also work on the client, even without a framework
- React Actions are first-class pattern for asynchronous data updates in response to user input
- React Actions are integrated with features like streaming, selective hydration, Suspense and Transitions

✓ Simple “reload everything” model

✓ Works without JS

✗ Limited feedback to user input

✗ Hard to add client behavior

```
// HTML Action  
<form action="/checkout">
```

✓ Instant feedback to user input

✓ Mix client and server behavior

✗ Async data management is hard

✗ Not interactive until JS loads

```
// Event handler  
<form  
  onSubmit={function onCheckout() {  
    //...  
  }}  
</form>
```

```
// HTML Action  
<form action="/checkout" />
```

```
// React Action  
<form  
  action={async (data: FormData) => {  
    await submitDataToServer(data);  
    redirect('/orders');  
  }}  
</form>
```

```
// React Server Action (in a server module)
<form
  action={async (data: FormData) => {
    'use server';
    await updateCartInDatabase(data);
    redirect('/orders');
  }}
/>
```

- ✓ JSX improvements
 - ➔ Ref is a prop, no need to use forwardRef()

What's Next

- ✓ React Compiler - automatically optimizes components and hooks, no need of memo() or useMemo()