createContext

August 17, 2026

createContext

Kind: Function

Source: `src/jsx/context.ts`

Part of: Jsx

Create a context whose value can be provided with `<Context.Provider>` and read with useContext.

Server-side renders are isolated per request, so a provided value never leaks into a concurrent request — even across `await` in an async component, when `AsyncLocalStorage` is available (Node.js >= 20.16, Deno, Bun, Cloudflare Workers with `nodejs_compat`). Without it, reading context after `await` returns the default value; synchronous components and `use()`-based suspension are unaffected.

`createContext` creates a context object with a default value, a `<Context.Provider>` component, and a value that `useContext` can read. During server rendering, context state is isolated by request; reading context after `await` in an async component depends on `AsyncLocalStorage` availability.

Signature

```
function createContext(defaultValue: T): Context<T>
```

Parameters

Name	Type
`defaultValue`	`T`

Returns: `Context<T>`

Diagram

```
graph LR
  Default[Default value] --> Context[createContext]
  Context --> Provider[Context.Provider]
  Provider --> Tree[Component tree]
  Tree --> Consumer[useContext]
  Consumer --> Value[Current context value]
```

Usage

```tsx
import { createContext, useContext } from "./jsx";

const ThemeContext = createContext("light");

function ThemeLabel() {
  const theme = useContext(ThemeContext);

  return <span>Theme: {theme}</span>;
}

export function App() {
  return (
    <ThemeContext.Provider value="dark">
      <ThemeLabel />
    </ThemeContext.Provider>
  );
}
```

AI Coding Instructions

- Create contexts outside component render functions so consumers reference the same context object.
- Wrap consumers with `<Context.Provider value={...}>` when they need a value other than the default.
- Use `useContext(Context)` only inside a component render or supported hook execution path.
- In async server components, avoid relying on context reads after `await` when `AsyncLocalStorage` is unavailable.

Used by

2 references from 2 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (2)

- `Props` — `src/jsx/base.ts`:27
- `StreamingContext` — `src/jsx/streaming.ts`:30