StreamingContext

August 19, 2026

StreamingContext

Kind: Constant

Source: `src/jsx/streaming.ts`

Part of: Jsx

Used to specify nonce for scripts generated by `Suspense` and `ErrorBoundary`.

`StreamingContext` carries a CSP nonce for inline scripts emitted by `Suspense` and `ErrorBoundary` during JSX streaming. Wrap streamed JSX with its provider so generated scripts receive the same nonce allowed by the response's Content Security Policy.

Definition

```
JSXContext<{ scriptNonce: string } | null>
```

Value

```
createContext<{
  scriptNonce: string
} | null>(null)
```

Diagram

```
graph LR
  A[StreamingContext.Provider] --> B[Streaming JSX render]
  B --> C[Suspense]
  B --> D[ErrorBoundary]
  C --> E[Generated script with nonce]
  D --> E
```

Usage

```
import { StreamingContext } from 'hono/jsx/streaming'

const nonce = 'request-csp-nonce'

const app = (
  <StreamingContext.Provider value={{ nonce }}>
    <Suspense fallback={<p>Loading···</p>}>
      <AsyncContent />
    </Suspense>
  </StreamingContext.Provider>
)
```

AI Coding Instructions

- Set the nonce from the request's Content Security Policy configuration.
- Wrap the streamed JSX tree with `StreamingContext.Provider` before rendering.
- Keep the nonce consistent with the nonce sent in the `Content-Security-Policy` response header.
- Use this context when `Suspense` or `ErrorBoundary` can emit inline streaming scripts.

Relationships

- IMPORTS → `html`
- IMPORTS → `HtmlEscapedCallbackPhase`
- IMPORTS → `resolveCallback`
- IMPORTS → `JSXNode`
- IMPORTS → `childrenToString`
- IMPORTS → `DOM_RENDERER`
- IMPORTS → `DOM_STASH`
- IMPORTS → `captureRenderContext`
- IMPORTS → `createContext`
- IMPORTS → `useContext`
- IMPORTS → `Suspense`
- IMPORTS → `buildDataStack`

Used by

1 reference from 1 file. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (1)

- `childrenToString` — `src/jsx/components.ts`:14