runWithRenderContext

August 18, 2026

runWithRenderContext

Kind: Function

Source: `src/jsx/context.ts`

Part of: Jsx

Establish the request-scoped context store for a render.

`resumeStore` continues a suspended subtree in the same store on the fallback path (ignored when `AsyncLocalStorage` is available, where isolation is automatic).

Without `AsyncLocalStorage` a render can't be followed across `await`, so the store lives in `fallbackStore` only during synchronous work (mirroring React's request storage). Reading context after `await` then finds no store and falls back to the default value — never another request's value.

`runWithRenderContext` creates the request-scoped context store used while rendering a JSX tree. When `AsyncLocalStorage` is available, the store remains isolated across asynchronous work; otherwise, the fallback store exists only for synchronous rendering and context reads after `await` return their default values.

Signature

```
function runWithRenderContext(callback: () => T, resumeStore: RenderStore): T
```

Parameters

Name	Type
`callback`	`() => T`
`resumeStore`	`RenderStore`

Returns: `T`

Diagram

```
graph LR
  Render[Render callback] --> Run[runWithRenderContext]
  Run --> ALS{AsyncLocalStorage available?}
  ALS -->|Yes| RequestStore[Request-scoped async store]
  ALS -->|No| FallbackStore[Temporary fallback store]
  RequestStore --> ContextRead[Context reads]
  FallbackStore --> ContextRead
  FallbackStore --> Cleanup[Restore previous fallback store]
```

Usage

```
import { runWithRenderContext } from './jsx/context';
import { renderPage } from './render';

const html = runWithRenderContext(() => {
  return renderPage(<App />);
});

// Context reads performed during renderPage use this render's store.
return new Response(html, {
  headers: { 'content-type': 'text/html' },
});
```

AI Coding Instructions

- Wrap each top-level render entry point with `runWithRenderContext` so context values are scoped to that render.
- Keep fallback-mode context-dependent rendering synchronous; context reads after `await` use the context default value when `AsyncLocalStorage` is unavailable.
- Pass `resumeStore` only when resuming a suspended subtree that must continue with its prior context values.
- Restore the previous fallback store after callback execution so one render cannot expose context values to another render.

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)

- `Props` — `src/jsx/base.ts`:27