

Pepe Fabra Valverde | Noviembre 2024

Headless UI

Una nueva manera de desarrollar interfaces

Disclaimer

- La sesión se grabará
- Se compartirán las slides al terminar la sesión
- Pregunta sin miedos

Librería/Framework

Se usa React como apoyo, pero aplicable a todo Front

Pepe Fabra Valverde

Senior Frontend Developer



Con que funcione me conformo

Historia de librerías y Desarrollo UI

Dónde empezamos

TUI, Terminal UI

Semáforos, farmacias y aeropuertos

Ejemplos cotidianos de Interfaces de Usuario

Qué buscaba el desarrollo UI

- Representar información al usuario
- Ser más accesible para usuarios menos expertos
- Limitaciones técnicas y el progreso a través de los videojuegos

HTML y CSS

Qué supuso

- Lenguaje unificado para el desarrollo de interfaces
- Una manera enriquecida de compartir información científica

Con que funcione me sirve

- Ordenadores no tan potentes
- Lenguajes no orientados al desarrollo de UI
 - o a veces no tan completos y cómodos como hoy en día

Front y Back

Más allá del servidor

- Cuándo ocurrió la separación más clara
- Front como una disciplina seria e independiente del back, no una capa más

Experiencia de Usuario

Cómo la psicología ayuda a desarrollar mejor

- UI, UX y ROI

Modifica lo que quieras

Y entonces nacieron las librerías

- Librerías opinionadas
- Qué es un software opinionado

Componente

- Cómo se diferencia del componente de arquitectura
- Enfoque para el desarrollo UI
- Contribución de React/JSX

Librerías de Componentes

- ¿Nos hacen falta? Accesibilidad, compatibilidad entre navegadores, funcionalidad compleja
- ¿Qué opiniones debería tener una librería?

Modifica lo que quieras

Sobreescribe lo que quieras

SCSS/SASS
2004, CSS



jQuery UI

2006, DOM Wrapper. 2007

Librería de comunidad



Angular y Material UI

2010 Framework. 2014 MUI



Backbone.js

2010, MVC



Bootstrap
2011, CSS, SCSS/SASS



Composition over inheritance

Composition over inheritance
¡¡De verdad!!

De dónde viene el concepto

- Object-Oriented Programming
- Herencia en componentes de UI

Por qué preferimos composición...

Componiendo delegamos
control y responsabilidad

**Componiendo delegamos
control y responsabilidad**

No seas un control freak

Ventajas de delegar el control

...y los riesgos

- Diseño y desarrollo ahora **componen** lo que necesitan
- Si no te puedes fiar de desarrollo, el problema es más grande de lo que creías

Para qué nos sirve componer interfaces

- React, y otros frameworks (Vue, Astro) están pensados desde la composición
- !important deprecado, estilas lo que necesitas
- Puedes acceder, y reutilizar, las partes que necesites (donde las necesites)

Estilos

¿Ya no usaremos CSS? ¿Cómo implementamos condicionales?

- Componer abre caminos... pero también genera problemas

Estilos

¿Ya no usaremos CSS? ¿Cómo implementamos condicionales?

- No dependemos de la estructura HTML para estilar
- Podemos usar tailwind/emotion para las clases
- Condicionales con `clsx`, `classNames` y más librerías (`stylex` en alpha)

Headless UI

Headless UI

Lógica desacoplada de la presentación

Headless User Interface Components

merrickchristensen.com/articles/headless-user-interface-components/

- 2018, Artículo original del patrón
- Merrick Christensen

Headless UI Aplicado

Cómo nos acercamos al patrón

Headless UI Aplicado

- Separamos lógica de la presentación
 - Hooks en react

Headless UI Aplicado

- Encapsulamos lógica y el consumer decide
 - Estructura en cierta medida, existirán limitaciones de dominio
 - Presentación (estilos, visibilidad, iconos)

Presentación y lógica no siempre
irán de la mano

Presentación y lógica no siempre
irán de la mano

Separarlos correctamente nos permite escalar

Tanstack Table

Tablas agnósticas

TanStack Table ^{v8}

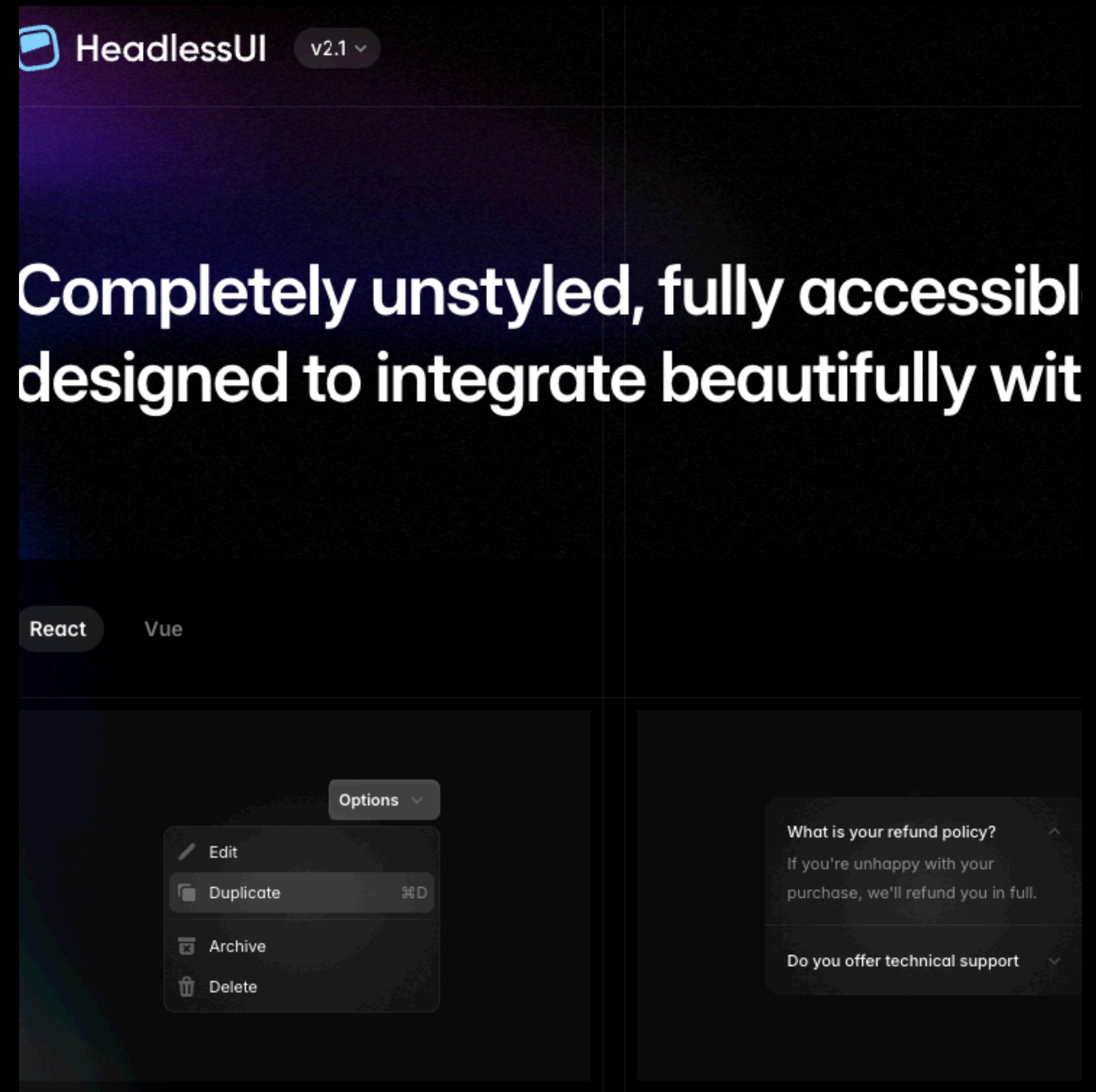
**Headless UI for building
powerful tables & datagrids**

Supercharge your tables or build a datagrid from scratch for TS/JS, React, Vue, Solid, Svelte & Lit while retaining 100% control over markup and styles.

[GET STARTED](#)

HeadlessUI

UI accessible, React y Vue



Radix UI

UI WACG completa

[Read about Radix Themes 3.0 →](#)

Start building your app now

An open source component library optimized for fast development, easy maintenance, and accessibility.
Just import and go—no configuration required.

```
import "@radix-ui/themes/styles.css";
import { Theme, Button } from "@radix-ui/themes";

export default () => (
  <Theme>
    <Button>Hey 🙌</Button>
  </Theme>
);
```

[Get started →](#)

 [Playground](#)

[Live examples](#)

Ark UI

Solid, React y Vue

New: Launching Ark UI Plus →

Fully customizable and accessible UI components

Ark UI is a headless library for building reusable, scalable Design Systems that works for a wide range of JS frameworks.

Get Started

Show Components



React



Solid



Vue

Separation of Concerns

Cómo abordarlo

1. Primero desarrollamos las piezas descentralizadas para resolver el problema
2. Y luego las juntamos como necesitemos en nuestro caso de uso
3. Usar un enfoque Outside-In*

**Usa lo que necesites, cuando lo necesites,
si lo necesitas**

5 Years of Building React Table

- Tanner Linsley
 - React table, query, virtualized, etc.
 - <https://www.youtube.com/watch?v=O4IWJcafX8c>

5 Years of Building React Table



TANNER
LINSLEY

NOZZLE.IO, USA

Domain-Driven Design y Compound Components

Domain-Driven Design

Una filosofía de diseño de software

- Eric Evans, 2003
- Domain-Driven Design: Tackling Complexity in the Heart of Software

Qué objetivo tiene DDD

- Que el software simbolice el negocio
- No perder el foco del negocio
- Lo técnico no ha de ser limitante del negocio*

**Se usa lenguaje de negocio para
escribir el software**

What is DDD?

DDD Europe

- Eric Evans
- <https://www.youtube.com/watch?v=pMuiVlnGqjk>



¿Por qué se creó?

- El foco estaba en desarrollar software, **no** en solucionar
- Sigue ocurriendo, nos olvidamos que desarrollamos producto **a través del software**

¿Cómo nos hemos desviado?

Qué no es Domain-Driven Design

- Replicar arquitectura hexagonal en Java
- Hay limitaciones técnicas, pero más de una vez la limitación es no haber usado el lenguaje de negocio para desarrollar el software

¿Cómo nos hemos desviado?

Qué sí es Domain-Driven Design

El software se adapta al negocio

**El software se adapta al
negocio... no al revés**

Bounded Context

- Partes concretas del negocio que interactúan con otras
 - Son módulos pero en lenguaje de negocio
- Tener un modelo mental ayuda a replicar fórmulas

Namespace de componentes

- Object.assign para Function Components
- “High cohesion, Low coupling”

Un-suck your React Components

- [https://www.youtube.com/
watch?v=vPRdY87_SH0](https://www.youtube.com/watch?v=vPRdY87_SH0)

COMPOSITION CONTEXT API



Compound Components

Composition by design

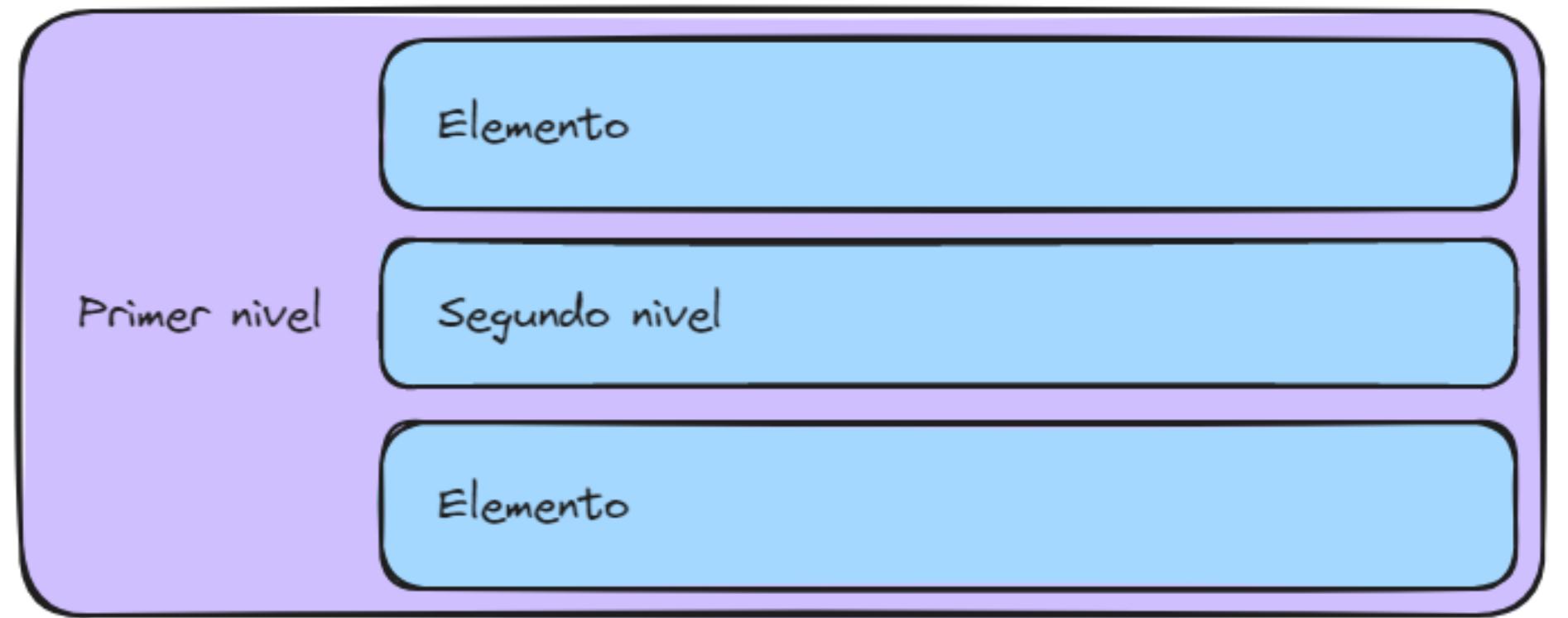
- Qué lo compone, props drilling, contextos y namespaces
- React, Vue, Solid

Cuidado con el flash
Ahora viene fondo blanco

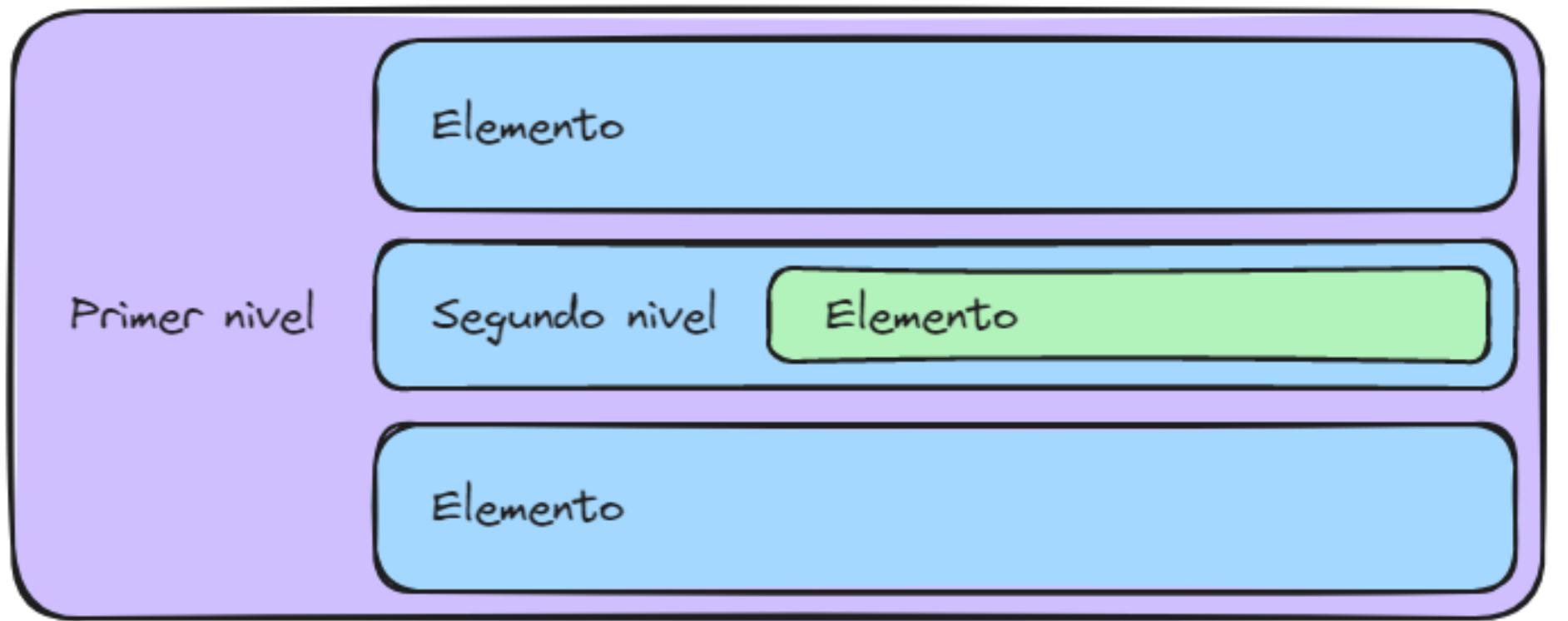
Namespace

Primer nivel

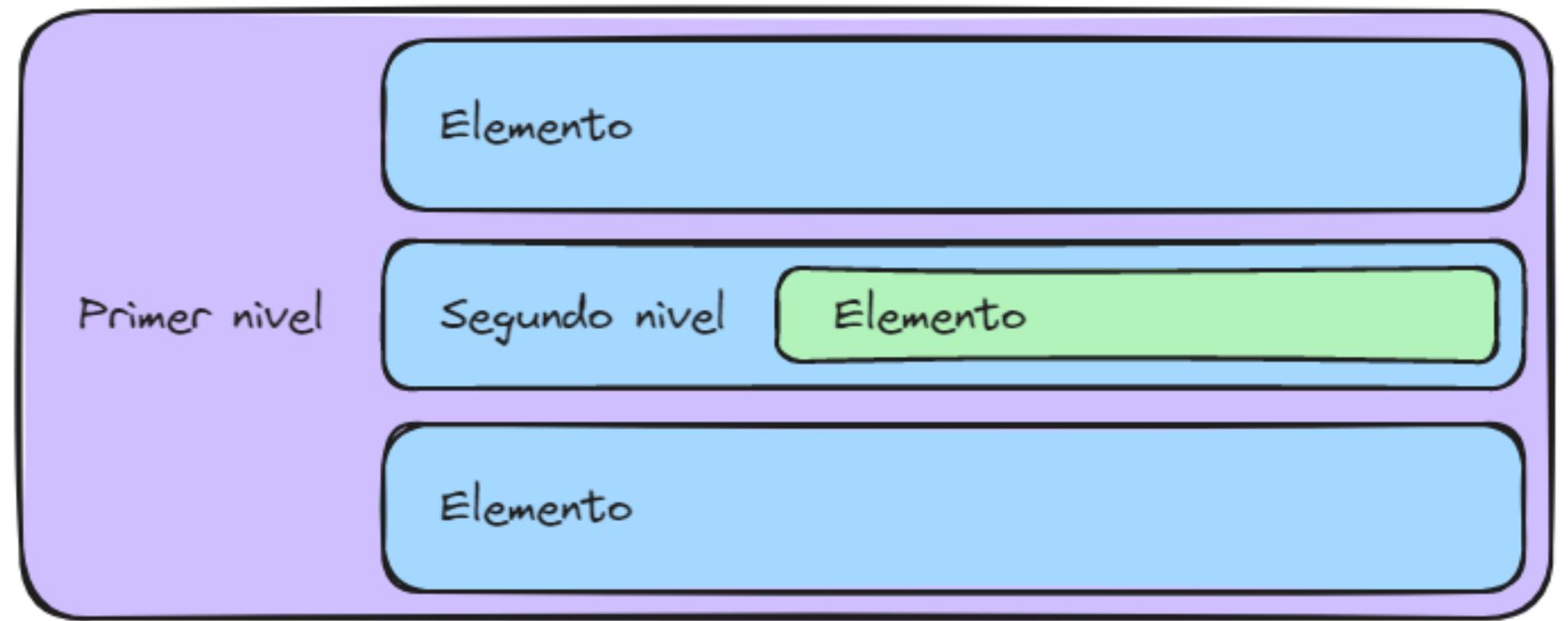
Namespace



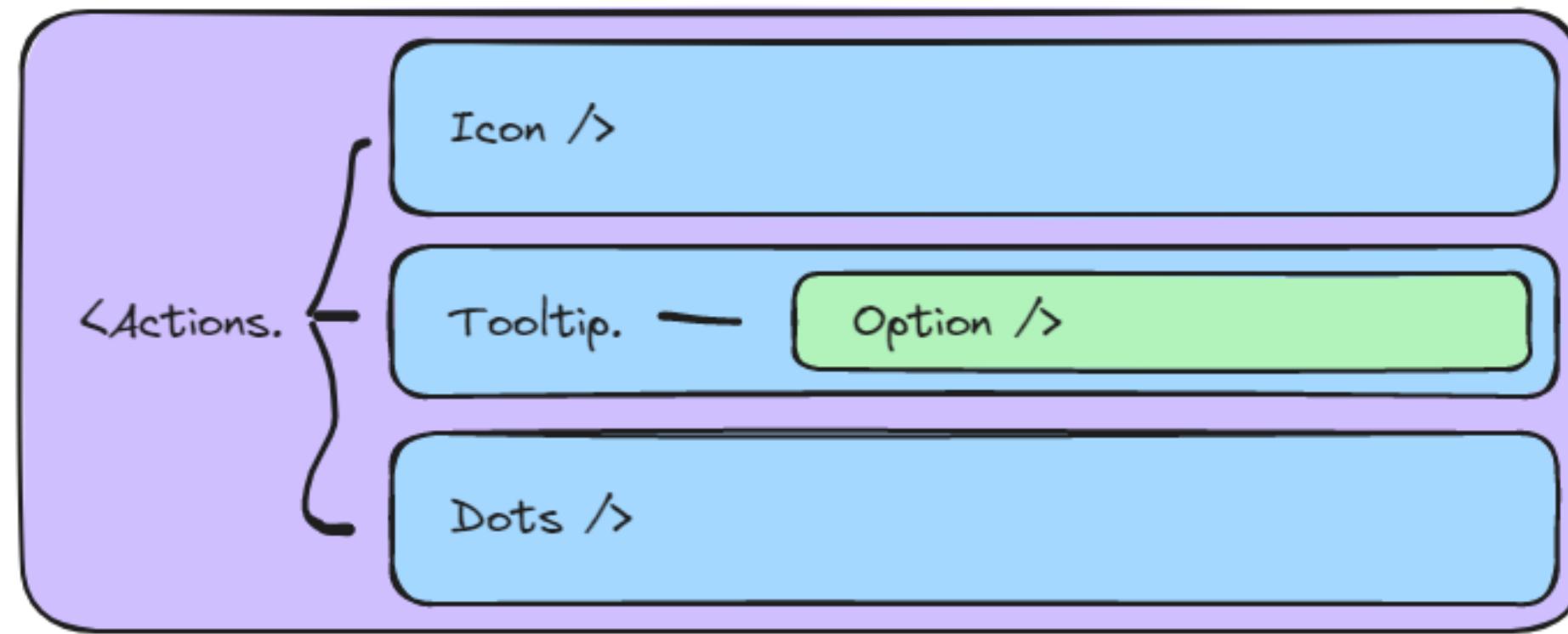
Namespace



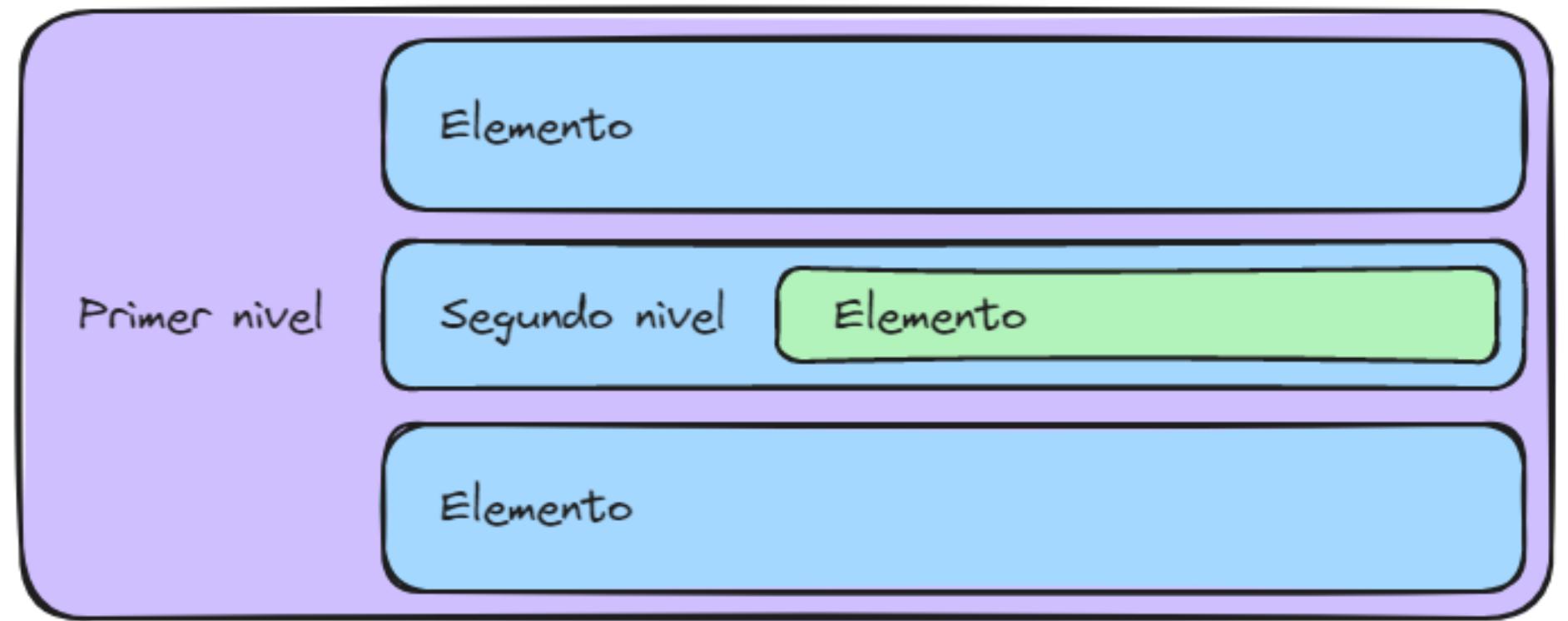
Namespace



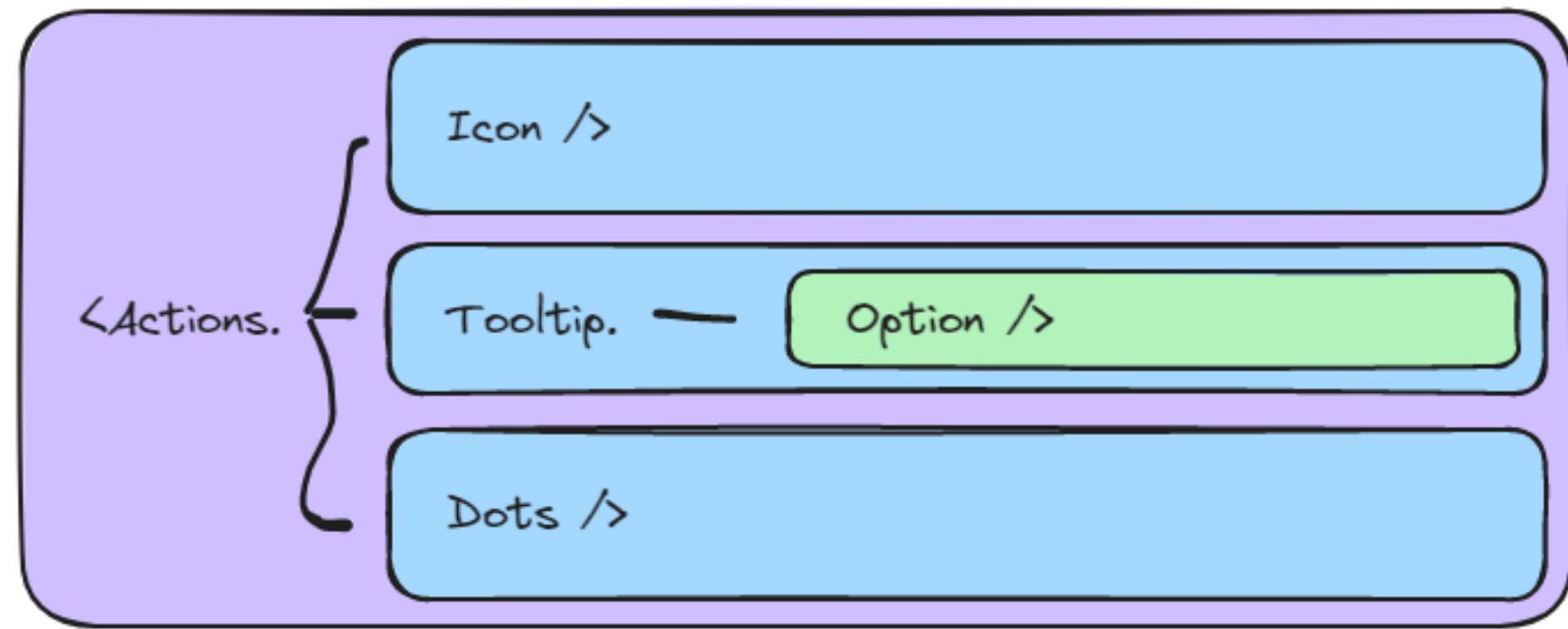
Caso de Uso



Namespace



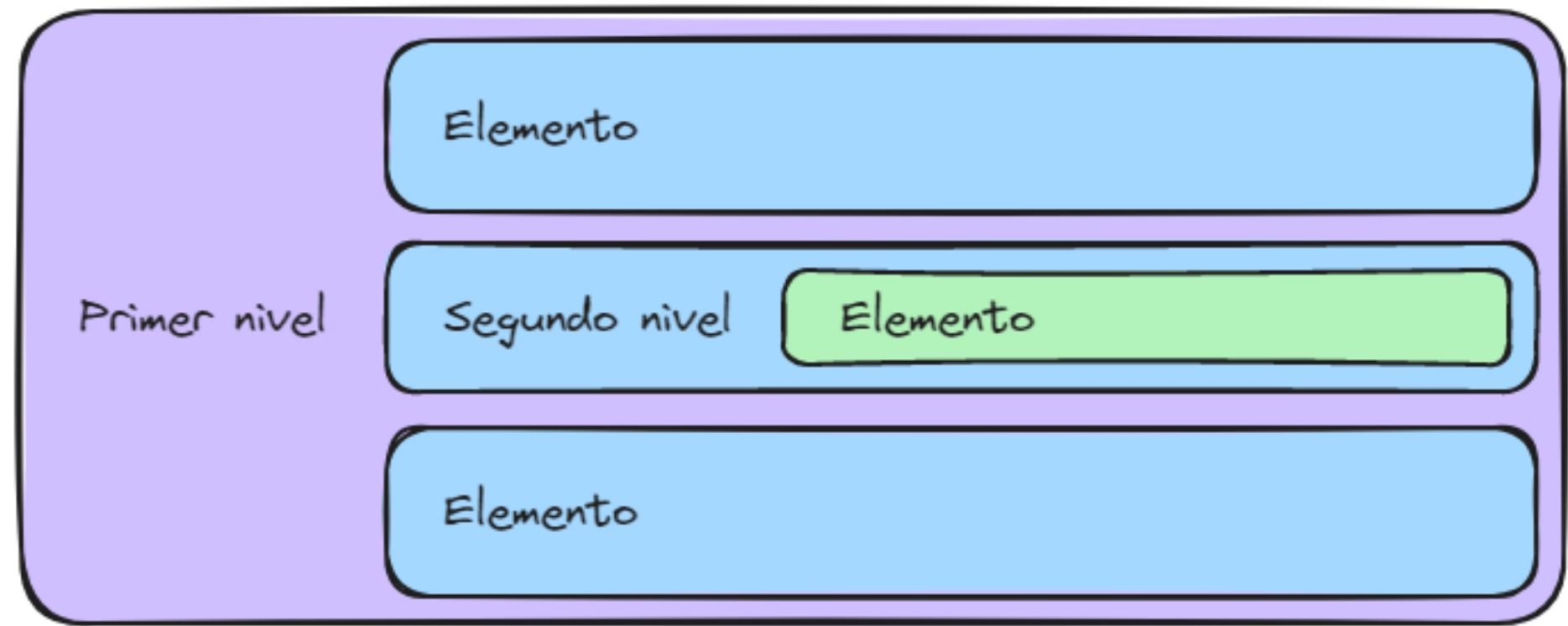
Caso de Uso



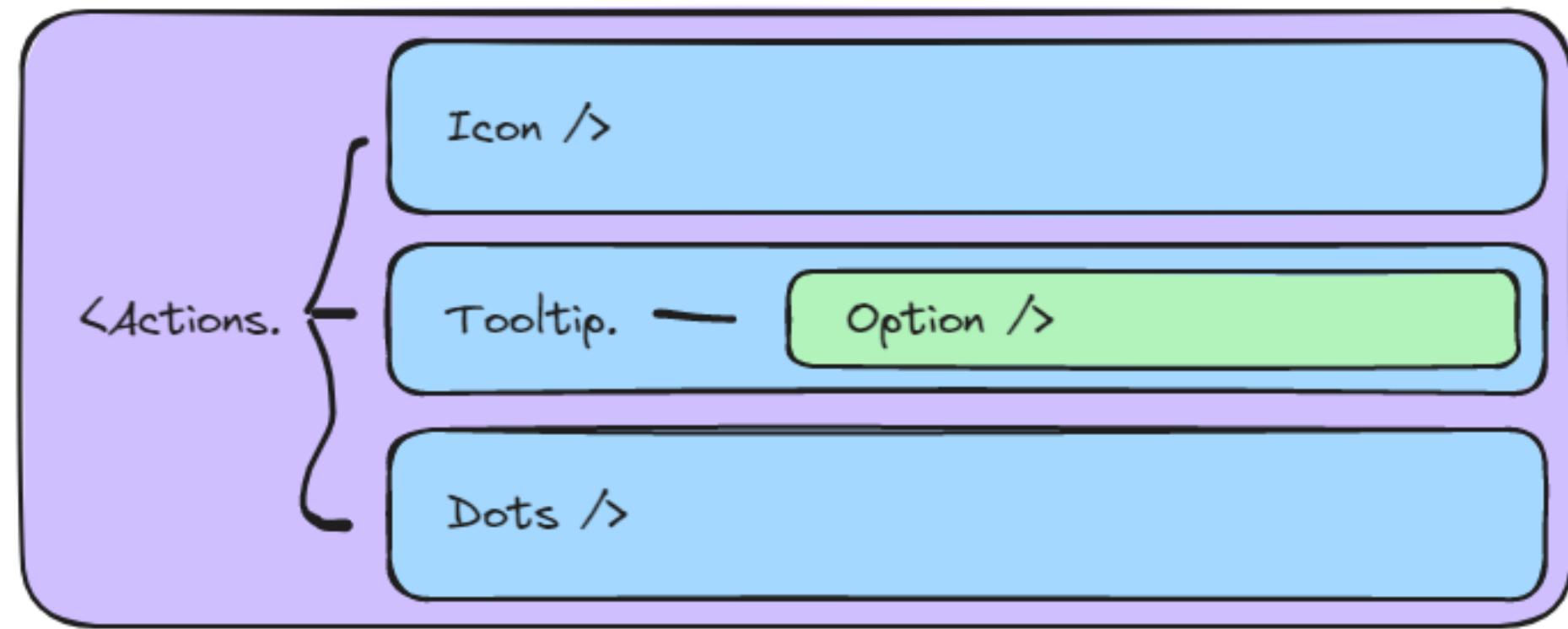
Contexto y dominio compartido

Namespace Scope

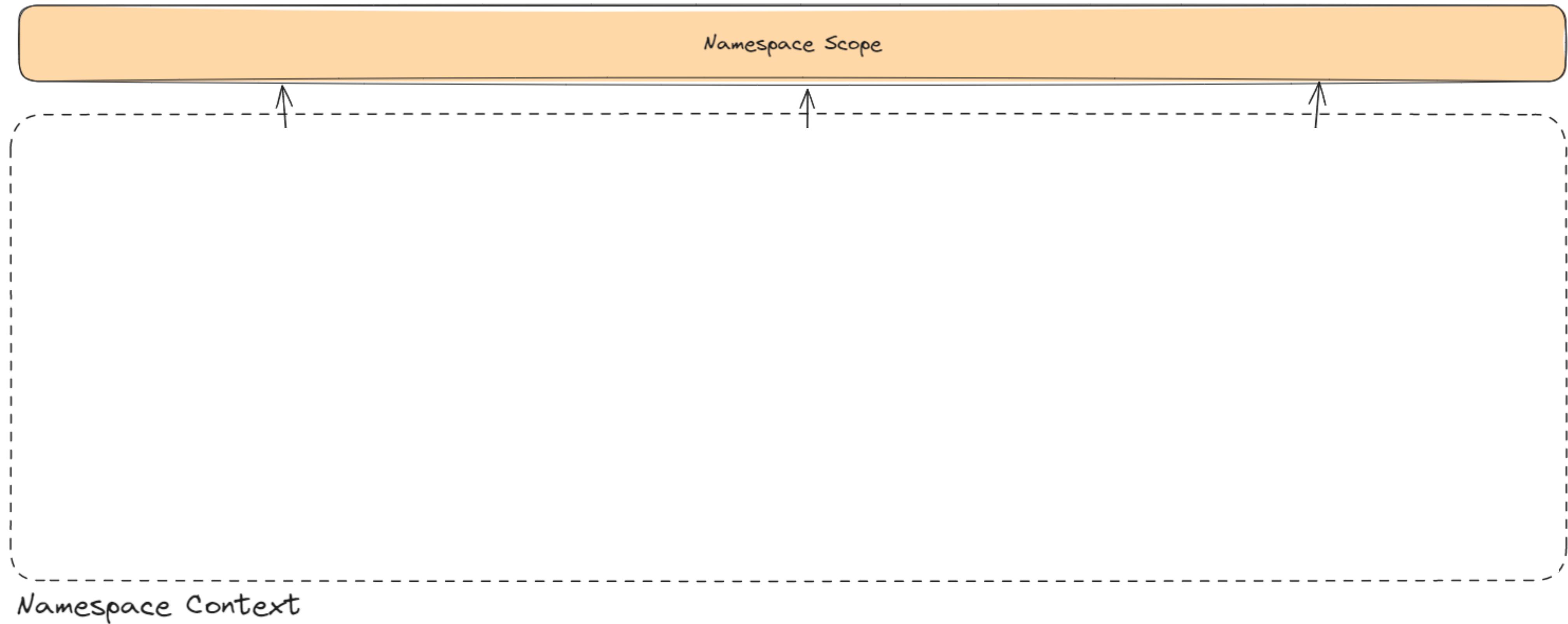
Namespace



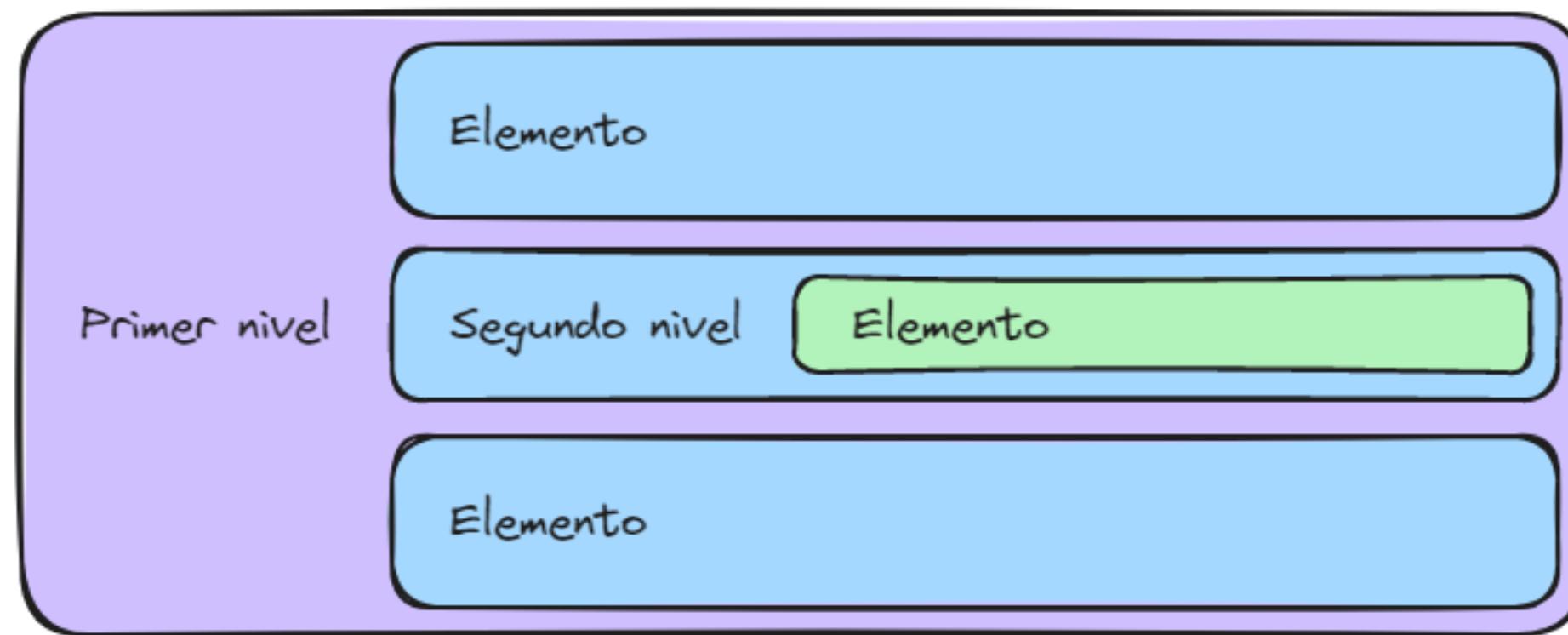
Caso de Uso



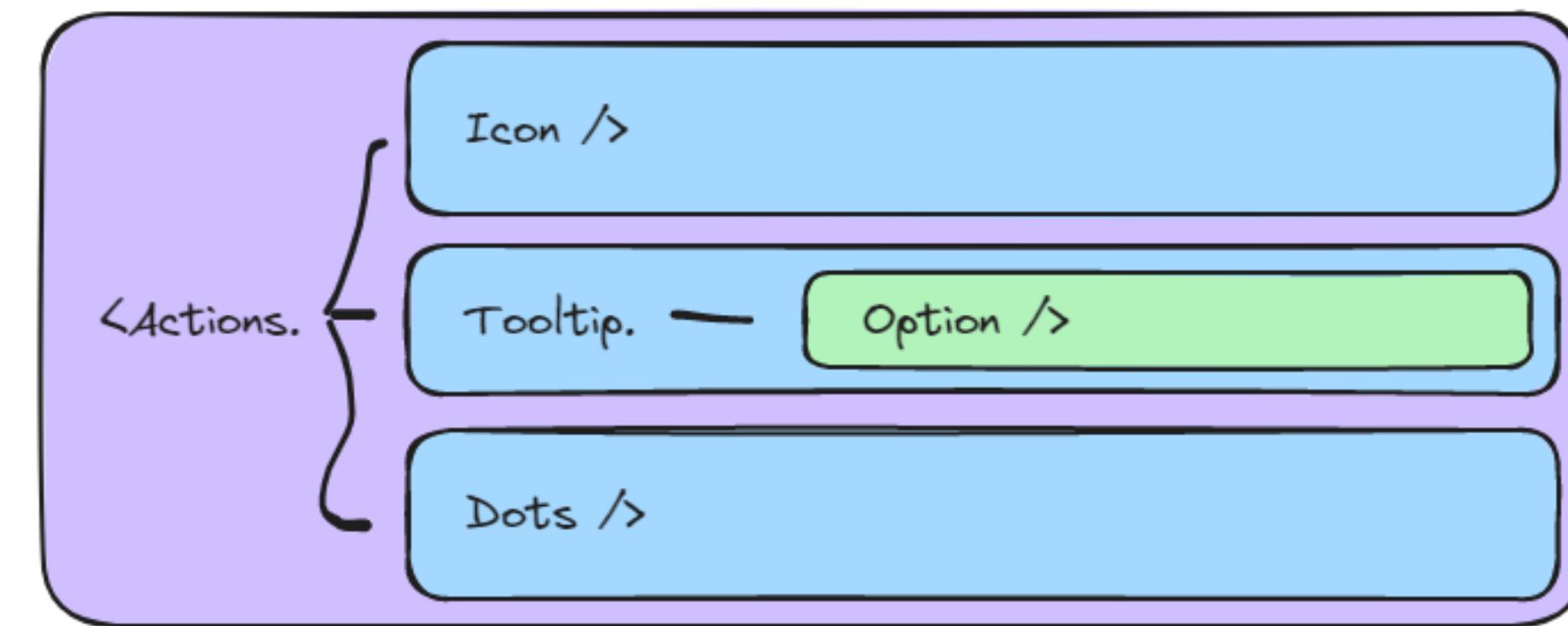
Contexto y dominio compartido



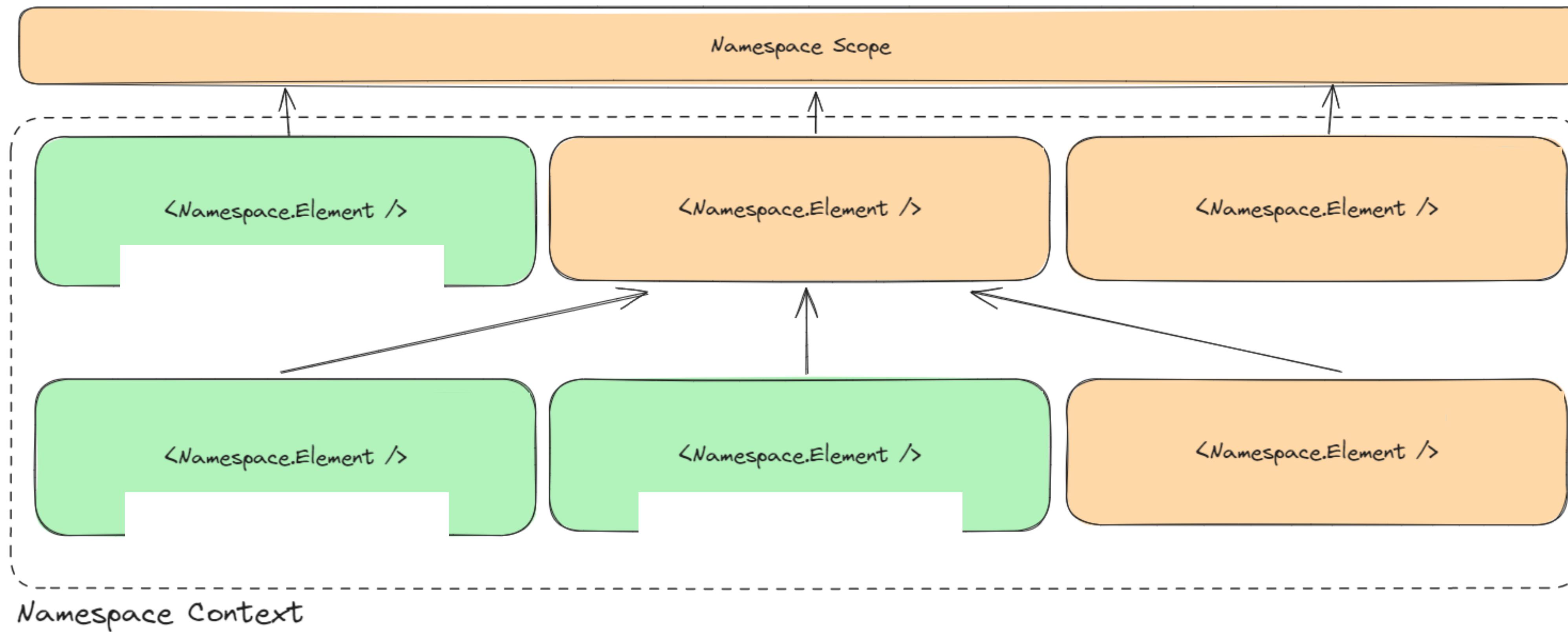
Namespace



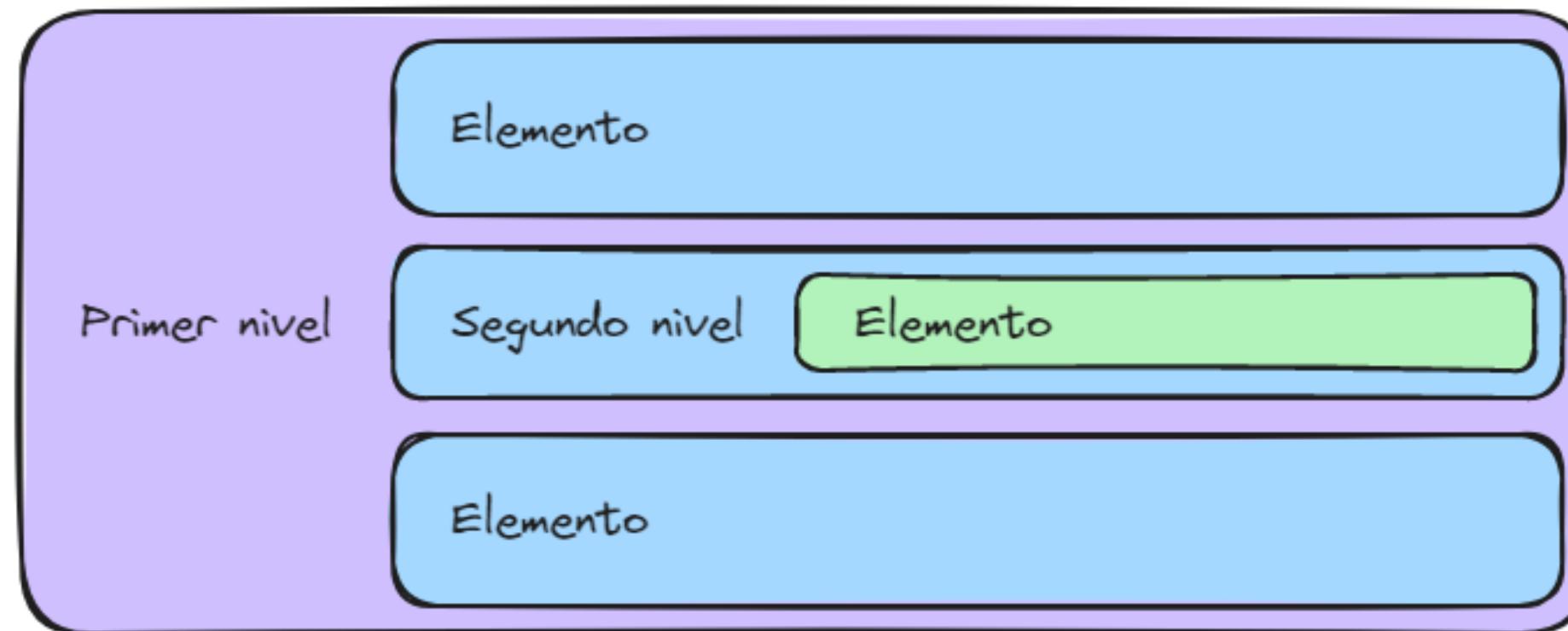
Caso de Uso



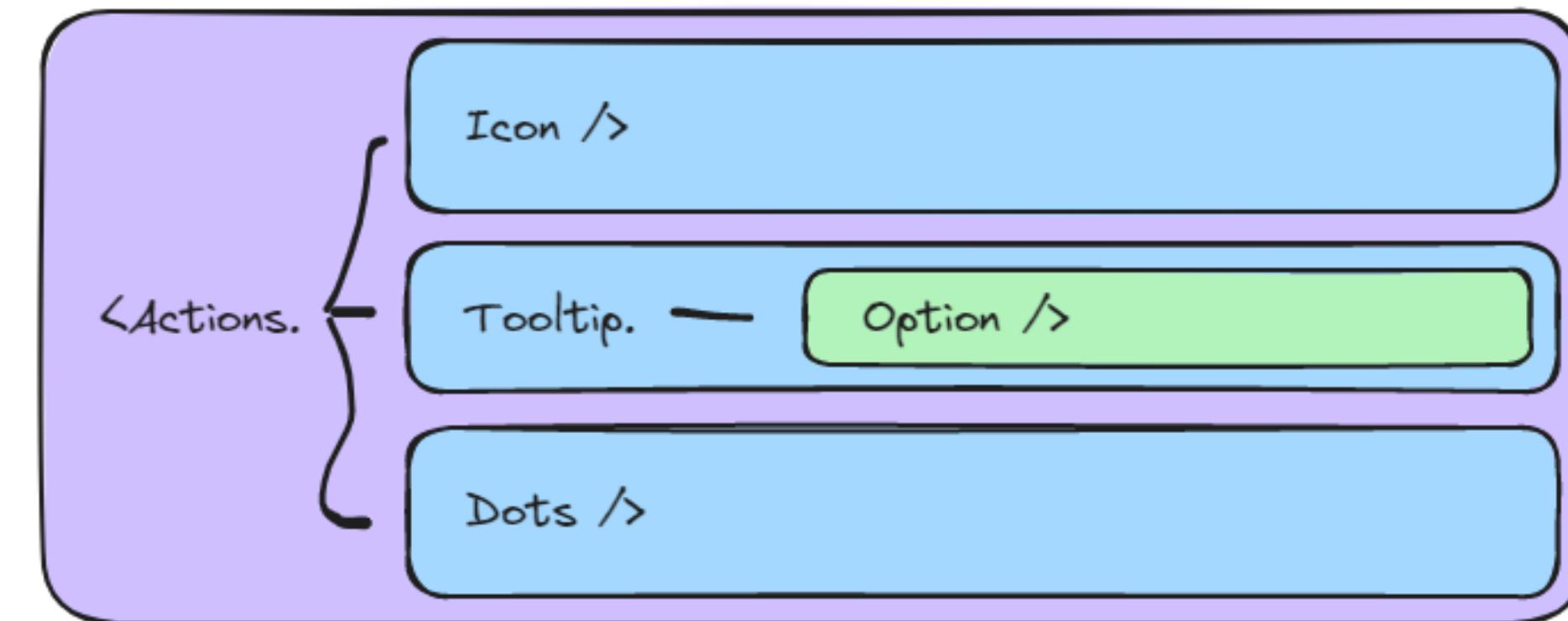
Contexto y dominio compartido



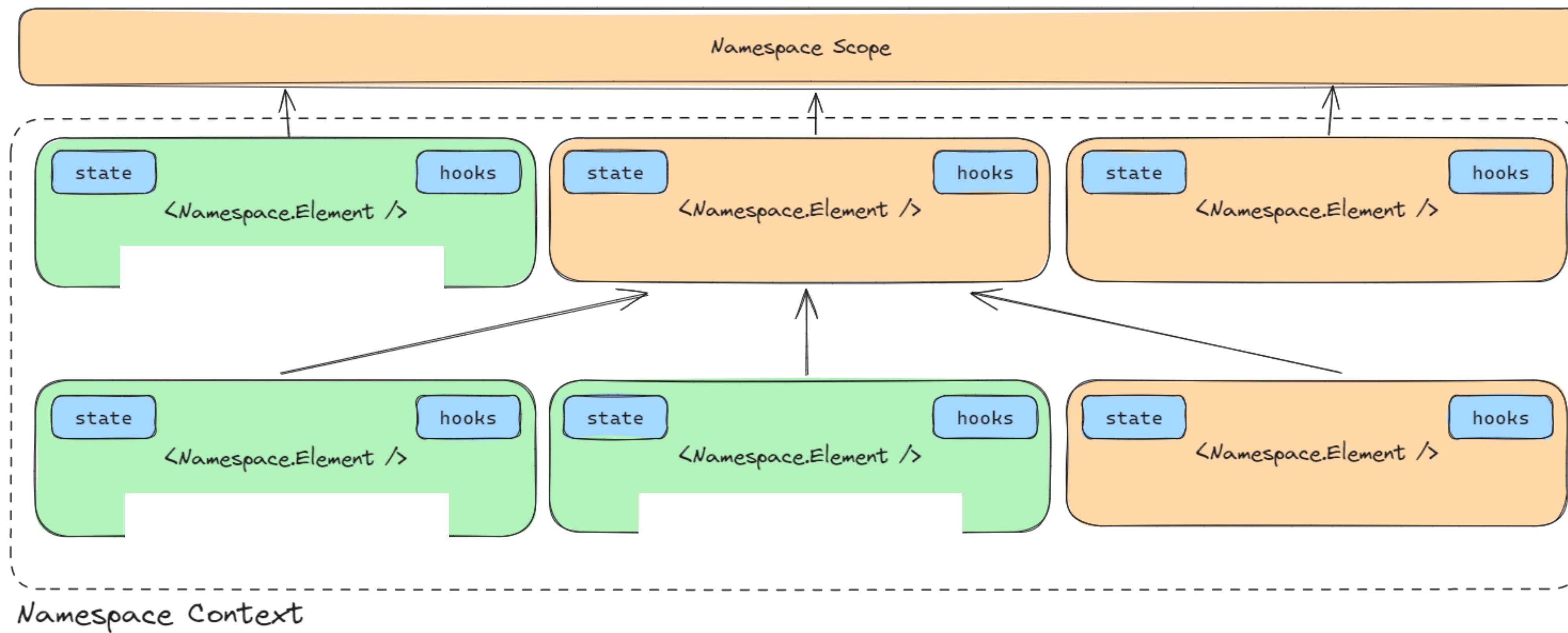
Namespace



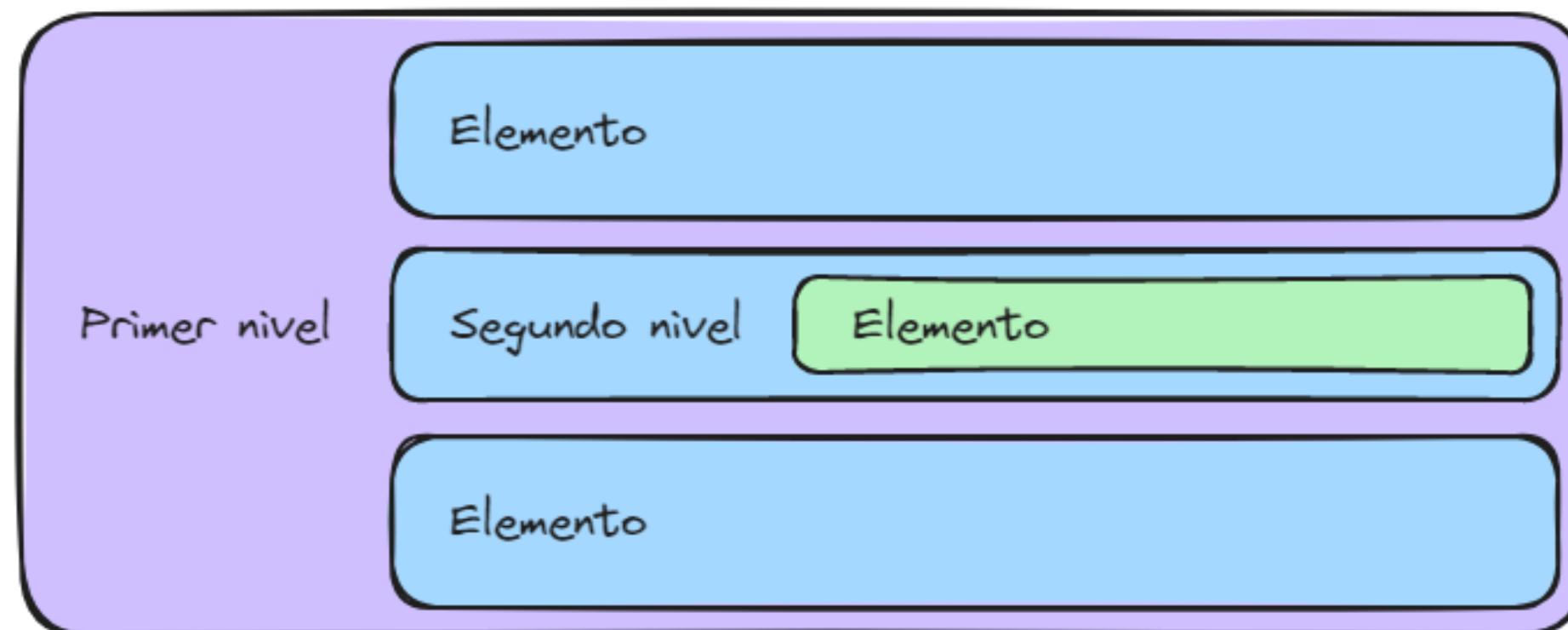
Caso de Uso



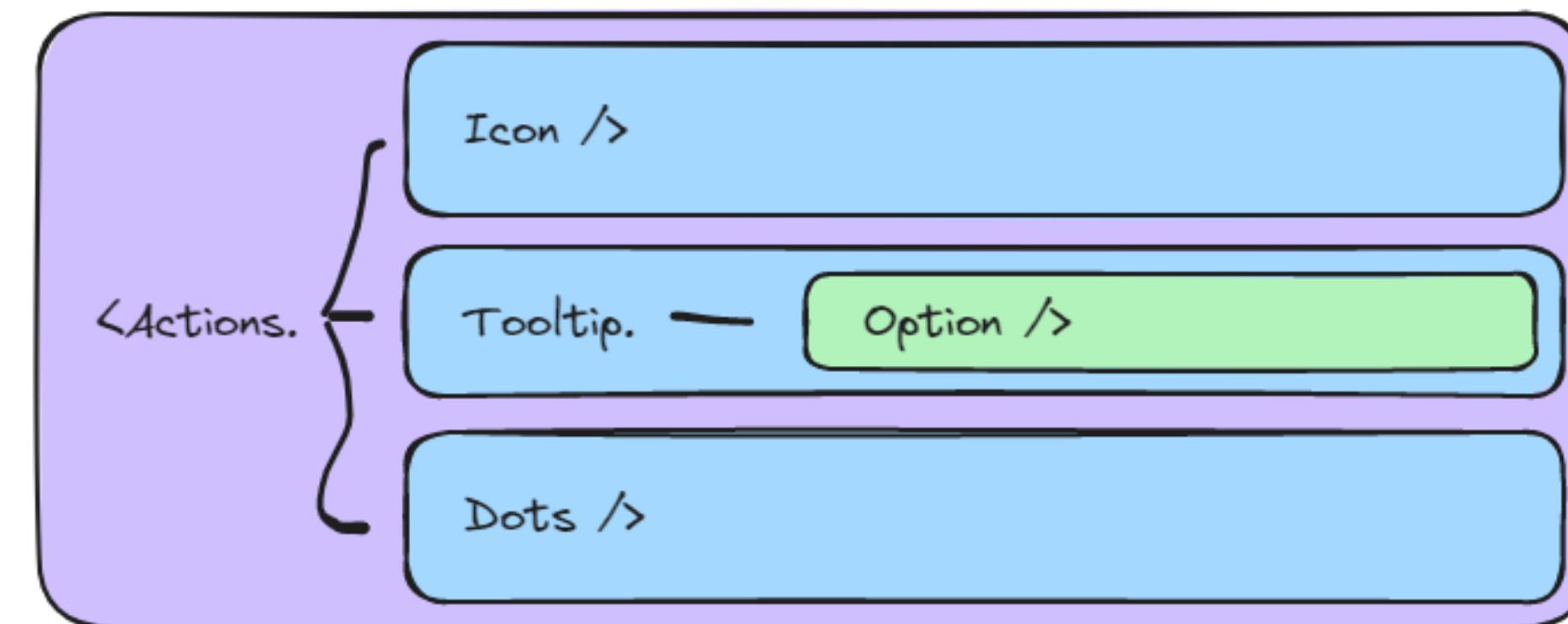
Contexto y dominio compartido



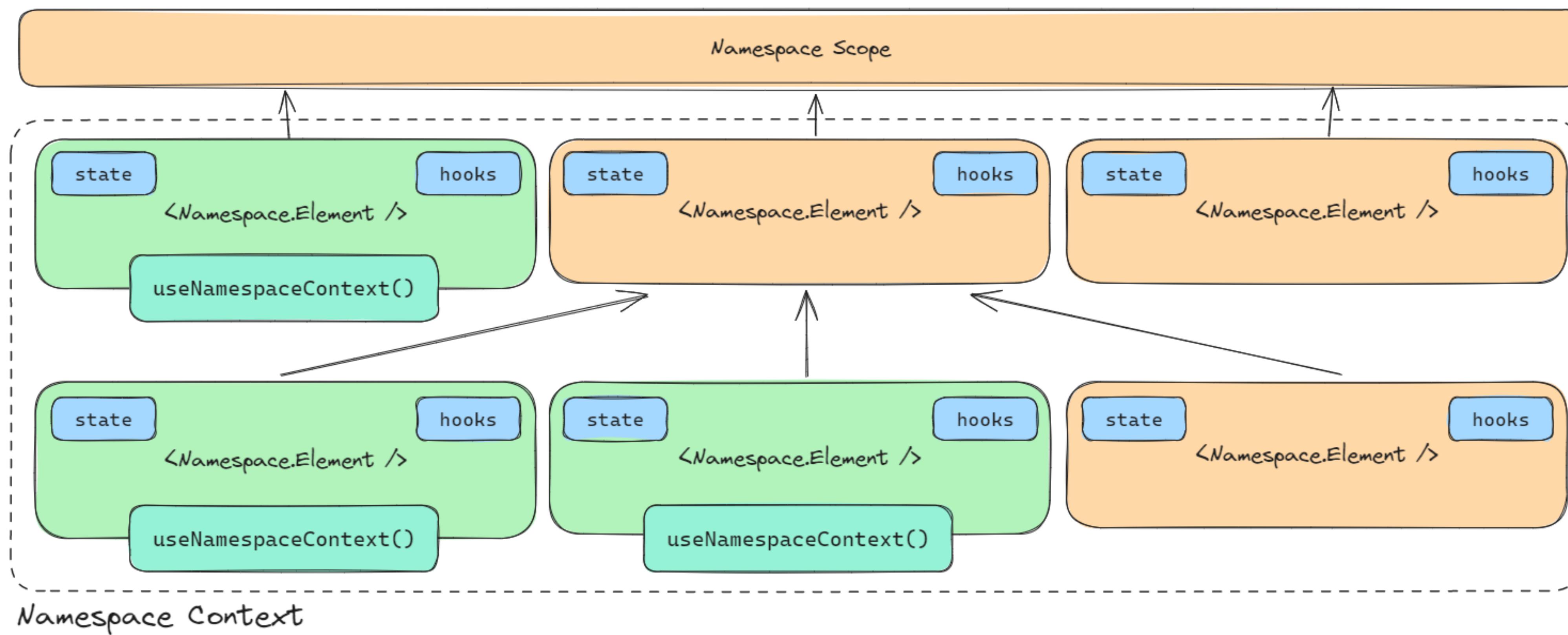
Namespace



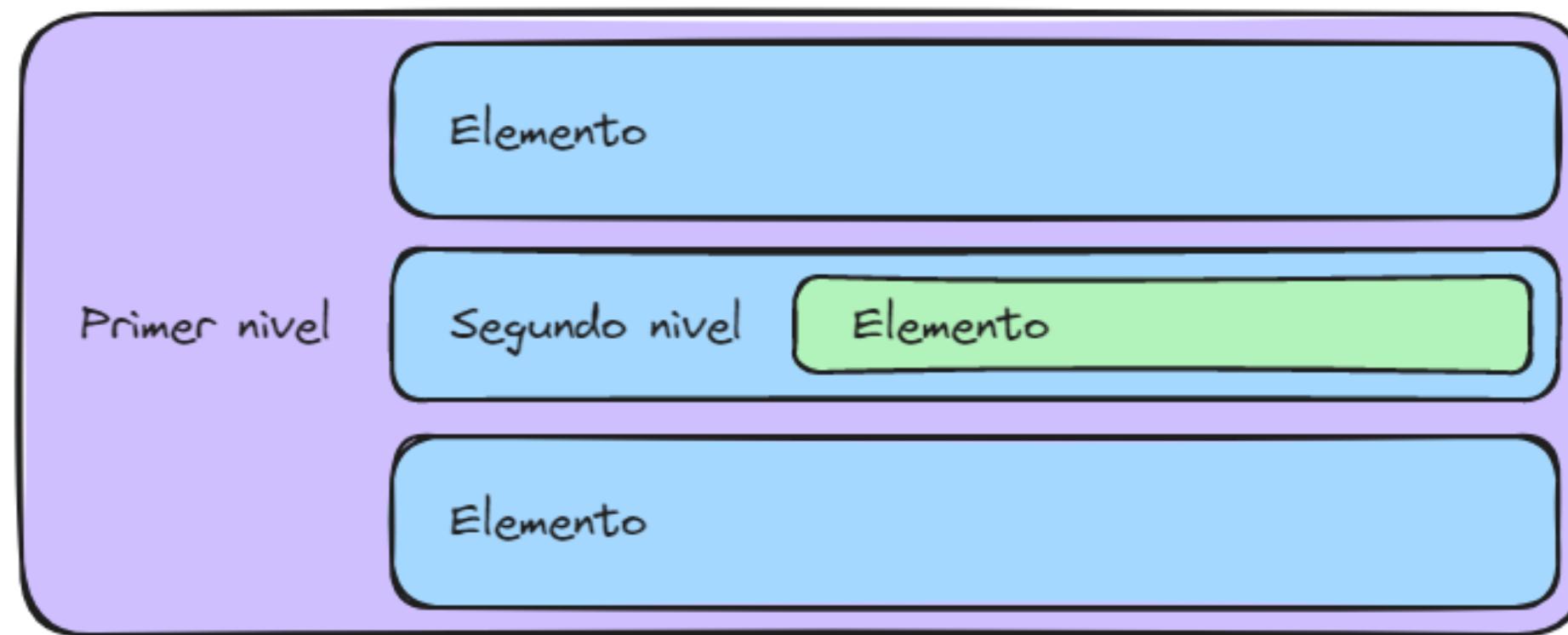
Caso de Uso



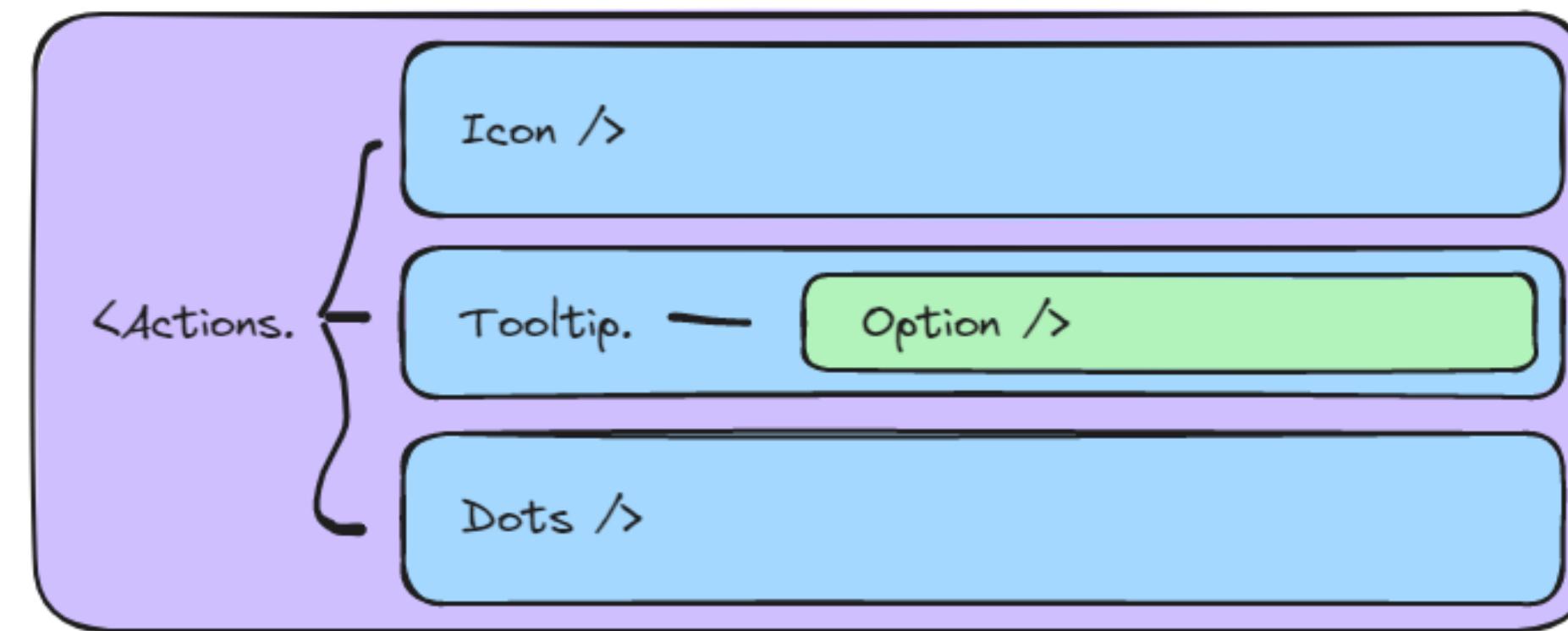
Contexto y dominio compartido



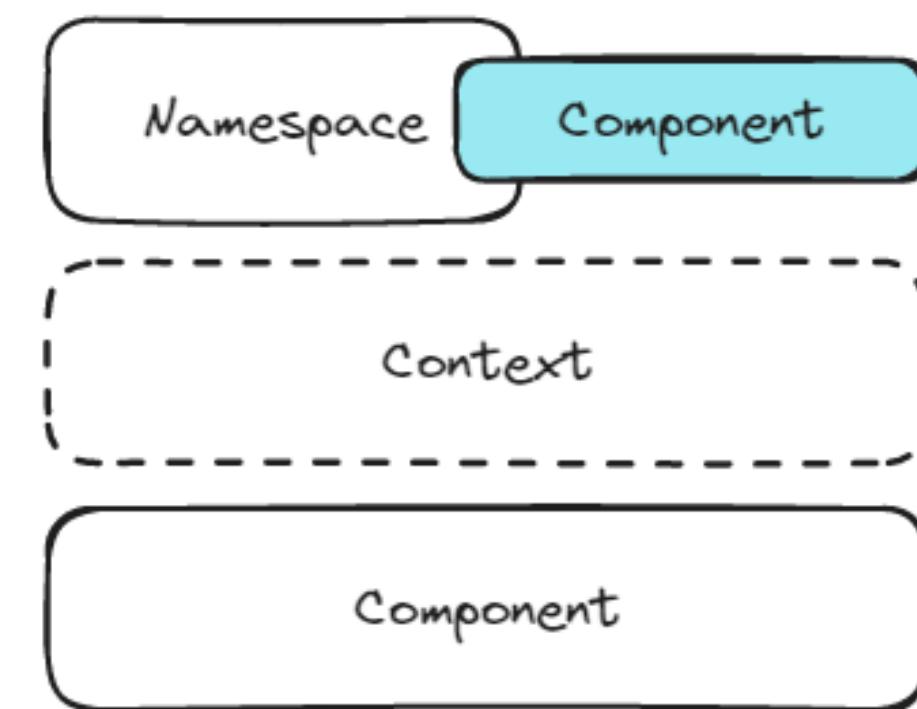
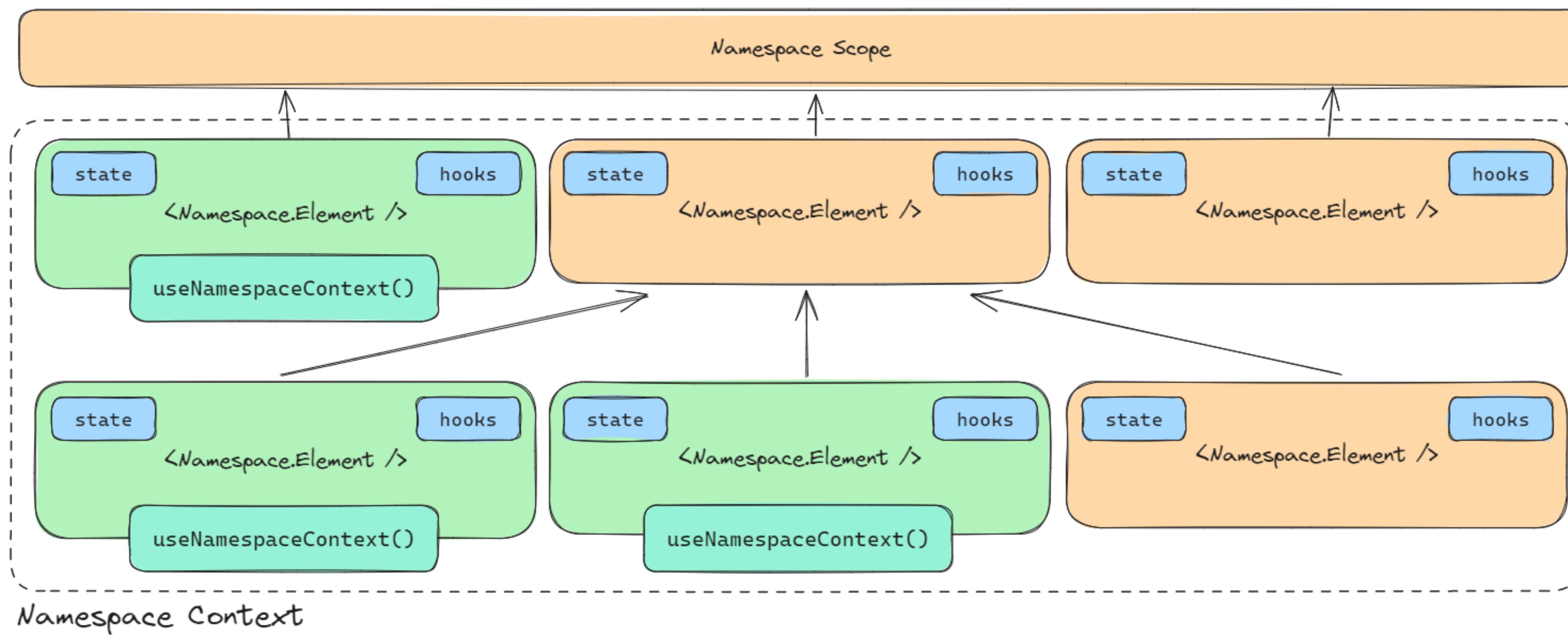
Namespace



Caso de Uso



Contexto y dominio compartido



Recapitulando

Recapitulando

Hemos visto...

- Historia de librerías UI

Recapitulando

Hemos visto...

- Historia de librerías UI
- Headless UI

Recapitulando

Hemos visto...

- Historia de librerías UI
- Headless UI
- Compound Components

Recapitulando

Hemos visto...

- Historia de librerías UI
- Headless UI
- Compound Components
- Domain-Driven Design

Headless UI

Una nueva manera de desarrollar interfaces

¡¡Gracias por la atención!!

Preguntas