readme.md for @push.rocks/smartntml

lit-html for the backend

Install

To install `@push.rocks/smartntml`, you need Node.js and npm installed on your machine. If you have them ready, open your terminal and run the following command:

```
npm install @push.rocks/smartntml --save
```

This will add `@push.rocks/smartntml` as a dependency to your project and you're ready to use lit-html in your backend applications.

Usage

The core idea behind `@push.rocks/smartntml` is to bring the convenience and developer experience of `lit-html` to server-side rendering. This package leverages the power of `lit-html`'s syntax and data-binding capabilities in a Node.js environment, especially focusing on backend applications that require rendering HTML.

To start using `@push.rocks/smartntml`, you need to familiarize yourself with TypeScript and ESM syntax, as this module is designed to offer the best experience within this ecosystem.

Setting Up Your Project with TypeScript and ESM

First and foremost, ensure your project is set up to use TypeScript and ECMAScript Modules (ESM). Your `tsconfig.json` should have the following settings:

```
{
  "compilerOptions": {
    "target": "ESNext",
    "module": "ESNext",
    "moduleResolution": "node",
    "esModuleInterop": true,
    "allowSyntheticDefaultImports": true
  }
}
```

Basic Usage

`@push.rocks/smartntml` simplifies the process of creating and rendering templates. The following example demonstrates how to create a template and render it:

1. **Import the necessary modules**

First, import `Smartntml` and any other module you wish to use:

```
import { Smartntml, deesElement } from '@push.rocks/smartntml';
```

`deesElement` is leveraged for its `TemplateResult` type and `html` tag function, allowing us to define our templates with ease.

2. **Create an instance of Smartntml**

```
const smartNTML = new Smartntml();
```

3. **Define a Template**

You can define a template using the `html` function provided by `deesElement`. This function allows you to embed dynamic content in a concise and readable manner.

```
const myTemplate = deesElement.html`
<div>Welcome to Smartntml!</div>
`;
```

4. **Render the Template**

With a template defined, you can render it to a string. This string can then be sent as a response to a client in a web application or stored for later use.

```
async function renderMyTemplate() {
  const renderedString = await smartNTML.renderTemplateResult(myTemplate);
  console.log(renderedString);
}
renderMyTemplate();
```

Working with Dynamic Content

`@push.rocks/smartntml` excels in rendering dynamic content. Assume you have an array of items that you want to display in a list:

```
const items = ['Apple', 'Banana', 'Cherry'];
const listTemplate = deesElement.html`
<ul>
  ${items.map(item => deesElement.html`<li>${item}</li>`)}
</ul>
`;
```

You can dynamically render arrays or any functional JavaScript within your templates, making it incredibly powerful for generating dynamic HTML content.

Advanced Usage

`@push.rocks/smartntml` can also integrate with other backend services and capabilities. For instance, using it alongside your server-side logic to render user-specific data, handling form submissions, or creating email templates.

Note: Always sanitize any user input before rendering it with `smartntml` to prevent XSS (Cross-Site Scripting) attacks.

Support and Contribution

`@push.rocks/smartntml` is an open-source project and welcomes contributions from the community. Whether it's improving the documentation, adding new features, or reporting bugs, every contribution helps make `smartntml` better for everyone.

Before contributing to the project, make sure to discuss your ideas or issues on the repository's issue tracker. This ensures that your contributions align with the project's goals and don't overlap with existing efforts.

Final Thoughts

`@push.rocks/smartntml` provides a streamlined and efficient way to use `lit-html`-like syntax for server-rendered templates in Node.js applications. It offers a blend of simplicity and power, enabling developers to create dynamic, data-driven HTML content on the backend with minimal effort. By integrating `@push.rocks/smartntml` into your project, you embrace modern JavaScript practices and significantly enhance your server-side rendering capabilities.

For more detailed usage examples and advanced features, make sure to check out the official documentation and explore the `@push.rocks/smartntml` codebase. Happy coding!

License and Legal Information

Trademarks

Company Information

Task Venture Capital GmbH
Registered at District court Bremen HRB 35230 HB, Germany

For any legal inquiries or if you require further information, please contact us via email at hello@task.vc.

Revision #3
Created 2026-03-28 11:11:36 UTC by foss.global Team

Updated 2026-03-28 12:18:27 UTC by foss.global Team