an open-source library to help with creating expandable text

jackyzha0.github.io/telescopic-text/

88 stars 3 forks 5 watching Branches Activity
Tags

Public repository

main 4 Branches 10 Tags Go to file t Go to file Add file Code

spencerc99 ACTUALLY BUILD IT 2a84c45 · 2 years ago

.github/workflows	update to use npm publish	3 years ago
assets	rm .DS_Store	3 years ago
demo	oops use the actual library	3 years ago
lib	ACTUALLY BUILD IT	2 years ago
src	fix duplicate text	2 years ago
.gitignore	rm .DS_Store	3 years ago
README.md	update readme	3 years ago
local-test.html	fix customization of replacement fns	3 years ago
package-lock.json	1.2.2	3 years ago
package.json	ACTUALLY BUILD IT	2 years ago
tsconfig.json	fix example	3 years ago

Telescopic Text

npm v1.2.6

An open-source library to help with creating expandable text.

Inspired by StretchText and TelescopicText

README

Background

I've been thinking a lot about creating a browsable store of knowledge that provides something useful at all distance scales (e.g. viewing the entire library, just a subcategory, a single file, etc.) and concepts like Telescopic Text are a first step in creating more information scales than just a single document level.

This library is meant to be the start for anyone to create telescopic text, including those who are non-technical.

Creating a telescopic text

To create some telescopic text, you can use your favorite note-taking app or text editor that supports bullet lists and start by writing a full sentence or two in a starting bullet:

Head to https://poems.verses.xyz/test to use an interactive playground for writing in bullet lists and get the corresponding code that leverages this library to generate the interactive text for use on your own website.

NOTE: the parsing logic is robust to different indentation levels, but for most compatible experience, you should normalize the indentations so that each nested bullet is differentiated by a standard set of spaces. We also currently only support `*`, `-`, and `+` bullet indicators.

Usage

Create some expandable text using the bullet list format shown above. You can then test out how your poem looks and feels and get a basic code snippet that recreates it using the test bed!

See the full instructions below:

You can load it directly via CDN as follows: Put this anywhere inside the `head` of your HTML file.

```
<head>
  ...
  <script src="https://unpkg.com/telescopic-text/lib/index.js"></script>
  <link
    href="https://unpkg.com/telescopic-text/lib/index.css"
    rel="stylesheet"
  />
</head>
```

or if you prefer to do the manual way, you can include the `lib/index.js` and `lib/index.css` files in your project.

The package exports a function called `createTelescopicTextFromBulletedList` that parses a bulleted list and returns a HTMLNode with your telescopic text inside.

Basic usage may look something like this:

```
<head>
  <script src="https://unpkg.com/telescopic-text/lib/index.js"></script>
  <link
    href="https://unpkg.com/telescopic-text/lib/index.css"
    rel="stylesheet"
  />
</head>
<body>
  <div id="text-container"></div>

  <script>
    const content = `
  * I
    * Yawning, I
  * made tea`;
    const node = createTelescopicTextFromBulletedList(content);
    const container = document.getElementById("text-container");
    container.appendChild(node);
  </script>
</body>
```

Advanced Usage Options

For further, customization, we provide a configuration object that can be passed into the function for different behavior.

```
// The configuration for how you want telescopic text to be parsed and rendered
interface Config {
  /**
   * Character used to separate entries on the same level. Defaults to a single space (" ")
   */
```

```
  separator?: string;
  /**
   * If true, allows sections to expand automatically on mouse over rather than requiring a click. Defaults to false.
   */
  shouldExpandOnMouseOver?: boolean;
  /**
   * A mode that designates what form the input text is in and should be interpreted as. Defaults to 'text'.
   */
  textMode?: TextMode;
  /**
   * Determines the wrapper element type for HTML elements. Defaults to 'span'.
   */
  htmlContainerTag?: keyof HTMLElementTagNameMap;
  /**
   * Only valid when textMode is 'text'. Used to insert HTML element like blockquotes, line breaks, bold, and emphasis in plain
   */
  specialCharacters?: TextReplacements;
}
```

You would use this by passing a custom configuration object into the function in order to override any of the defaults above. For example, this is how you would create telescopic text with custom HTML tags:

```
const content = `
* Some <b>rich</b> text
  * with <i>nested</i> <b>rich</b> text
`;
const config = { textMode: TextMode.Html };
const poemContent = createTelescopicTextFromBulletedList(content, config);
```

You can check out a more detailed example in `demo/index.html`

If you are using plain 'text' as the textMode, you can also define an object containing special characters and the rules for how to replace them.

```
interface TextReplacements {
  // Each entry is keyed by its regex string match
  // It defines a function that takes in the current line of text as well as its parent node
  // and
  [key: string]: (lineText: string) => HTMLElement
}

// for example, here's a text replacement rule for bolding text that is wrapped with *
"\\*(.*)\\*": (lineText) => {
  const el = document.createElement("strong");
  el.appendChild(document.createTextNode(lineText));
  return el;
}
```

By default, only these special characters have text replacements

- line breaks (`---`)
- bold (`*...*`)
- emphasis (`_..._`) To disable this, you can pass in an empty object for special characters:

```
const poemContent = createTelescopicTextFromBulletedList(content, {
  specialCharacters: {},
});
```

Types

```
// Default function to create a new `<div>` node containing the
// telescoping text from bullet points
function createTelescopicTextFromBulletedList(content: string, config?: Config);
```

Future Work

See our issues page for all the features we're thinking about exploring. Some examples include:

- Supporting infinite expansion with `...`
- Concept of shapeshifting text in general... these are not always expansions.

Development

Releases 8

v1.2.4 Latest
on Mar 16, 2022

+ 7 releases

Packages

No packages published

Contributors 3

 **jackyzha0** Jacky Zhao

 **spencerc99** Spencer Chang

 **jakeisnt** Jake Chvatal

Deployments 33

github-pages 2 years ago

+ 32 deployments

Languages

TypeScript 75.5% **HTML** 18.2% **CSS** 6.3%