**text-scrambler**

An isomorphic text scrambling utility for both Browser & NodeJS environments.

**Setup**:

|  |
| --- |
| $> npm i @kcak11/text-scrambler |

**Loading the text-scrambler module in a *Browser* environment:**

After running the above install command, pick the file **text-scrambler.js** from the below path and place it in your static assets folder:

|  |
| --- |
| node\_modules/@kcak11/text-scrambler/bin/**text-scrambler.js** |

Then in your web page include it inside a script tag.

|  |
| --- |
| <script src="assets/scripts/**text-scrambler.js**"></script> |

**Loading the text-scrambler module in *NodeJS* environment:**

After running the install command, do the below to load the text-scrambler module:

|  |
| --- |
| /\* Place the below line at the top of your JS module. \*/  require("**@kcak11/text-scrambler**"); |

**Code Examples (for both Browser & NodeJS environments):**

Scrambling the code:

|  |
| --- |
| var message = "This is a secret message that needs to be scrambled";  var key = "MY\_SECRET\_KEY";  var scrambled;  scrambled = scramble(message, key).raw();  console.log(scrambled); /\* This will output the scrambled text in **raw** format \*/  scrambled = scramble(message, key).encode();  console.log(scrambled); /\* This will output the scrambled text in **urlencoded** format. \*/  scrambled = scramble(message, key).base64encode();  console.log(scrambled); /\* This will output the scrambled text in **base64** encoded format. \*/  /\*  Note: The encode() & base64encode() functions return output that can be copy-pasted and persisted in a file for use later on.  The raw() function outputs the raw data which coule be lost after a copy-paste operation. Hence it is recommended to use the encode() or base64encode() functions if you want to persist the scrambled data.  \*/ |

For *unscrambling* you should follow similar steps as above:

|  |
| --- |
| var key = "MY\_SECRET\_KEY", scrambledMessage, originalMessage;  /\*  CASE 1:  When the scrambledMessage contains **raw** data obtained by  earlier usage of scramble(...).**raw**() function  \*/  scrambledMessage = . . . . // "This contains the scrambled message in **raw** format.";  /\* This will return the original message if the key matches. \*/  originalMessage = scramble(scrambledMessage, key).raw();  /\*  CASE 2:  When the scrambledMessage contains **base64** encoded data obtained by  earlier usage of scramble(...).**base64encode**() function  \*/  scrambledMessage = . . . . // "This contains the scrambled message in **base64** format.";  /\* This will return the original message if the key matches. \*/  originalMessage = scramble(PipedData.base64decode(scrambledMessage), key).raw();  /\*  CASE 3:  When the scrambledMessage contains **urlencoded** data obtained by  earlier usage of scramble(...).encode() function  \*/  scrambledMessage = . . . . // "This contains the scrambled message in **urlencoded** format.";  /\* This will return the original message if the key matches. \*/  originalMessage = scramble(PipedData.decode(scrambledMessage), key).raw(); |

Copyright © 2020 – Ashish’s Web.

License: **MIT** [ https://mit-license.kcak11.com ]