**Constructor: Predictor**

(

HTMLElement targetElement,

CSSClassNameString frameClass,

CSSClassNameString divClass,

CSSClassNameString selClass,

Function dataFetch,

Function singleton

[,Object lovOptions]

)

**Parameters:**

|  |  |
| --- | --- |
| targetElement | The HTML/DOM element that you wish to apply the dynamic List-of-Values functionality to.  This Element:   * Must be an “input” tag of type “text” * Must have the “id” attribute defined * Cannot be under the control of another Predictor Object |
| frameClass | The CSS class name you wish to apply to the outer container DIV for the list-of-values |
| divClass | The CSS class name you wish to apply to the inner DIV that will contain the unordered list. |
| selClass | The CSS class name you wish to apply to the individual list item that is being hovered over or selected. |
| dataFetch | The user-defined function that will populate the list.  The value of *this* in dataFetch is the instance of the Predictor object that was instantiated when calling this constructor.  The parameters dataFecth will receive are detailed later. |
| singleton | The user-defined function that will be invoked when the user has clicked on a list item.  The value of *this* in singleton is the instance of the Predictor object that was instantiated by calling this constructor.  The parameters singleton will receive are detailed later. |
| lovOptions | This is a JSON object containing the various options that can affect the behaviour of your Predictor.   |  |  | | --- | --- | | fixedWidth | Boolean. If set to true, the width of the list frame will be set to the width of the specified targetElement. Scrolling will control any overflow. If set to false, the list frame will grow as wide as is necessary to accommodate content.  The default is false. | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  | | --- | --- | | keyDetent | Integer. The number of milliseconds to pause between user keystrokes before Predictor will call the dataFetch function.  The default value is 300. | | maxRows | Integer. Specifies the maximum number of rows in the list that can be displayed at one time before vertical scrolling is required to see further list items. If there are less list items than maxRows then only those list items will be displayed.  The default is 10. | | scrollBars | Boolean. If set to true, scrollbars will be visible when required. If set to false, any scrollbars will be hidden from view.  The default is false. | | cancellable | Boolean. If set to true, it signifies that the dataFetch function is capable of being cancelled. This improves performance as any subsequent entry or deletion of characters from the target element may result in orphaned and redundant searches.  The default is true. | |

**Predictor Methods:**

**setOptions(lovOptions)** Modify the options specified during instantiation.

**hideLOV()**  Hide the list from view andreset the criteria.

**removeLOV()** Completely remove the instance of the Predictor.

**dataFetch Parameters:**

|  |  |
| --- | --- |
| targetElement | HTHLElement. This is a reference to the target input element that was passed to the Predictorconstructor. |
| appender | Javascript object. The appender lets you add rows to the list of suggestions and otherwise manipulate a specific instance of list.  Methods surfaced by each appender are described below. |

**Appender Methods:**

**Append(rowkey, rowData)**

Appends a new item to the list of matching suggestions. The list grows visibly as items are added.

|  |  |
| --- | --- |
| rowKey | This variable will be passed to the singleton function and is intended to provide a means of uniquely identifying the rowData to a database of other container. |
| rowData | The list item to be appended to the list of values. Must be of type “string”. |

**Returns:** A [Promise](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Using_promises) that resolves to the index of newly inserted list item; starting from zero. Or -1 if the list has been closed, the list has been superseded by another list, or if rowData is not a string.

**Cancel()** *Only available if the Cancellable flag was set via lovOptions*

The Cancel method aborts the population of the current list of suggestions and closes its appender. The AbortController.abort() method is invoked, therefore anyone referencing the signal returned from getSignal() will be instructed to abort operation.

**Returns:** Nothing.

**Close()**

The Close method terminates the population of the current list of suggestions by closing its appender. Any further calls to append will have no effect on the current list and return -1.

**Returns:** A [Promise](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Using_promises) that resolves to 1 if the list closure was successful or -1 if it was already closes.

**GetSignal()** *Only available if the Cancellable flag was set via lovOptions*

For all cancellable instances of a list, Predictor creates an AbortController and associates it with the list. The getSignal method allows you to retrieve the controller’s signal attribute. This cancellation token can be used for asynchronous operations on both the server and the client. For example, the below C# MVC Controller method can be cancelled if the return value from getSignal was passed to your Javascript Fetch call: -

Note that the AbortController is managed by Predictorand the Abort() method is called each time a list is superseded by the user changing the contents of the targetElement. You can also abort the list creation at anytime by calling the appender.cancel() method. For example, you may wish to specify a timeout on your server queries or provide the user with a Cancel Button.

The impressive combination of the AbortController, Fetch, IAsyncEnumerable, and Yield provide an extremely fast and resource-frugal solution to database queries!

**Returns:** <https://developer.mozilla.org/en-US/docs/Web/API/AbortSignal>

**Cancellable C# server method:**

****

**singleton Parameters:**

|  |  |
| --- | --- |
| targetElement | HTHLElement. This is a reference to the target input element that was passed to the Predictor constructor. |
| rowKey | The variable that was passed to the appender.append() method when the rowData. |
| rowData | The value of the list item String that was selected by the user. |
| selectedIndex | Ordinal position of the rowData in the list. Starting at 0. |
| appender | Javascript object. The appender lets you add rows to the list of suggestions and otherwise manipulate a specific instance of a that list.  Methods surfaced by each appender are described above. |