Watson Editor API v1.0

Watson Editor Included Features

Syntax Highlighting
Click Event
Mouse Hover Event
Insertion Bar
Line Numbers
Adding Lines
Deleting Lines

Constructor

Editor(divID, chapterName, exerciseNum, lineNumBool, syntaxHighlightingBool, lineNumStart, insertBetweenRowsBool, editable, autoSave)

Description

This function constructs the Editor. A div will need to be initialized and the constructor will find this div using the divID parameter. That div will be used to insert all the html of the editor.

Parameters

- *divID*
  Type: {number}
  the ID of the div to place the editor in
- *chapterName*
  Type: {string}
  the name of the chapter this editor is in, used for data storage
- *exerciseNum*
  Type: {number}
  the number of the exercise this editor is in, used for data storage
- *lineNumBool*
  Type: {boolean}
  if true display line numbers, if false do not
- *syntaxHighlightingBool*
  Type: {boolean}
  if true display syntax highlighting, if false do not
- *lineNumStart*
  Type: {number}
  what number line numbers should start at
- *insertBetweenRowsBool*
  Type: {boolean}
  if true a line can be inserted/deleted anywhere, if false lines can only be inserted/deleted from the end of the editor
- *editable*
  Type: {boolean}
  if true, the editor is in sandbox mode, if false, the editor is in figure mode
- *autoSave*
  Type: {boolean}

if true the editor will save/load itself, if false it will not save/load

Returns nothing

Public Functions

rowToArray(index)

Description

This function takes a row index and returns an array where each element of the array is the text from the cells of that row. Does not include line number cell or the cell between the line number and the code.

Parameters

- *index*
  Type: {number}
  the index of the row to process

Returns an array

an array of strings of the cells of the row

rowToArrayHtml(index)

Description

This function takes a row index and returns an array where each element of the array is the innerHTML from the cells of that row. Does not include line number cell or the cell between the line number and the code.

Parameters

- *index*
  Type: {number}
  the index of the row to process

Returns an array

an array of strings of the cells of the row

rowToDOMArray(index)

Description

This function takes a row index and returns an array of the DOM objects of the cells in that row. Does include line number cell or the cell between the line number and the code.

Parameters

- *index*
  Type: {number}
  the index of the row to process

Returns an array

an array of DOM objects of the cells of the row

getRowCount()

Description

This function returns the number of rows in the program definition of the editor.

Parameters

none

Returns a number

the number of rows in the editor

addRow(index, values, saveState)

Description

This function takes a row index (where you want to insert row) and an array of objects. Each object will contain a string containing desired cell text, and an array of class names.

Parameters

- *index*
  Type: {number}
  the index of the row to insert at
- *values*
  Type: {object}
  an array of objects with two things: the text of the cell and the class for syntax highlighting
- *saveState*
  Type: {boolean}
  if true, save the state of the editor, if false, do not, defaults to true

Returns nothing

addCell(cell, values)

Description

This function adds cells into a table after the one passed. So if the function is passed cell 1 and two new cells: cell 1 would be unchanged, and the new cells would be added after cell 1 in the same table.

Parameters

- *cell*
  Type: {DOM object}
  the initial cell
- *values*
  Type: {object}
  an array of objects with two things: the text of the cell and the class for syntax highlighting

Returns nothing

deleteRow(index, saveState)

Description

This function deletes the row at the specified index. The logic for when to delete a row should be handled by individual labs.

Parameters

- *index*
  Type: {number}
  the index of the row to delete
- *saveState*
  Type: {boolean}
  if true, save the state of the editor, if false, do not, defaults to true

Returns nothing

deleteCell(cell, numberOfCells)

Description

This function deletes the cells after the specified cell. So if you want to delete two cells after cell 1, cell 1 would be unchanged, and the two cells immediately after cell 1 would be deleted.

Parameters

- *cell*
  Type: {DOM object}
  the initial cell, this is NOT deleted
- *numberOfCells*
  Type: {number}
  the number of cells after the one passed to be deleted

Returns nothing

selectRowByIndex(index, performInsertionCheck)

Description

Selects a row based on the index provided and inserts a blank line there.

Parameters

- *index*
  Type: {number}
  the row to select
- *performInsertionCheck*
  Type: {boolean}
  if true check the position of the insertion bar cursor, if false do not

Returns nothing

selectAndHighlightRowByIndex(index)

Description

Selects and highlights a row based on the index provided. The selected line is also given the "running" class which overrides all other highlighting and con only be removed by calling selectAndHighlightRowByIndex() on another row, or calling clearHighlighting().

Parameters

- *index*
  Type: {number}
  the row to select

Returns nothing

setSelectedRow(index)

Description

This function sets the selected row to the value passed.

Parameters

- *index*
  Type: {number}
  the row index to set the selected row to

Returns nothing

clearHighilighting()

Description

This function manually clears all of the highlighting across the editor. Syntax coloring remains intact.

Parameters

none

Returns nothing

moveInsertionBarCursor(index)

Description

This function moves the cursor in the insertion bar, which is removed in a private mouse leave event.

Parameters

- *index*
  Type: {number}
  the index of the row to move the cursor to

Returns nothing

getSelectedRowIndex()

Description

This function returns the currently selected row's index.

Parameters

none

Returns a number

the index of the current row

setCellClickListener(clickFunc)

Description

This function sets the callback function for clicks. WARNING: this function turns off the click handlers for all of the cells of this editor

Parameters

- *clickFunc*
  Type: {function}
  the click callback function, should take a DOM object as an argument

Returns nothing

setInsertBarMouseEnterListener(mouseEnterFunc)

Description

This function sets the callback function for mouse enter. WARNING: this function turns off the mouse enter handlers for all of the cells of this editor

Parameters

- *mouseEnterFunc*
  Type: {function}
  the mouse enter callback function, should take a DOM object as an argument

Returns nothing

saveEditor(force)

Description

This function uses the Watson Data Store to save the editor based on the chapter and exercise number.

Parameters

- *force*
  Type: {boolean}
  if true a save is forced regardless of editable and autoSave, defaults to false

Returns nothing

loadEditor(oldDivID, newDivID, force)

<u>Description</u>

This function uses the Watson Data Store to load the editor based on the chapter and exercise number. It also has the capacity to change the divID across the editor to allow for loading editors that were initialized with different divIDs.

<u>Parameters</u>

- *oldDivID*
  Type: {string}
  if not null, will call changeDivID with this argument
- *newDivID*
  Type: {string}
  if not null, will call changeDivID with this argument
- *force*
  Type: {boolean}
  if true a load is forced regardless of editable and autoSave, defaults to false

<u>Returns nothing</u>

clearEditor()

<u>Description</u>

This function clears the editor and uses the Watson Data Store to clear the editor's saved data.

<u>Parameters</u>

none

<u>Returns nothing</u>

checkEditorData(force)

<u>Description</u>

This function simply wraps Watson Data Store's checkExerciseData().

<u>Parameters</u>

- *force*
  Type: {boolean}
  if true a check is forced regardless of editable and autoSave, defaults to false

<u>Returns a boolean</u>

returns true if data for this chapter name, exercise number exits, returns false otherwise

changeDivID(oldDivID, newDivID)

<u>Description</u>

This function changes the divID parameter that was passed to the editor and resets it everywhere else as well.

<u>Parameters</u>

- *oldDivID*
  Type: {string}
  the old divID to replace
- *newDivID*
  Type: {string}
  the new divID to replace the old one with

<u>Returns nothing</u>

Syntax Highlighting Classes

code

Description

Defines a normal coloring for a cell (black). Implicitly given to every cell in an editor.

Hover Behavior

Highlights this cell.

keyword

Description

Defines a color for keywords (blue).

Hover Behavior

Highlights this cell.

literal

Description

Defines a color for literals (brown).

Hover Behavior

Highlights this cell.

comment

Description

Defines a color for comments (green).

Hover Behavior

Highlights the entire line.

datatype

Description

Defines a color for datatypes (purple).

Hover Behavior

Highlights this cell

selected

Description

Defines a color for selected lines (red).

Hover Behavior

none

openParen

Description

Used for parentheses matching.

Hover Behavior

Highlights forward to the matching closeParen on the same line and the cell immediately before this one

closeParen

Description

Used for parentheses matching.

Hover Behavior

Highlights backwards to the matching closeParen on the same line and the cell immediately before the matching openParen.

openBrack

Description

Used for Bracket matching.

Hover Behavior

Highlights forward to the matching closeBracket as well as the line above this one.

closeBrack

Description

Used for Bracket matching.

Hover Behavior

Highlights backwards to the matching openBrack and the line above the matching openBrack.

startLoop

Description

Used for loop highlighting.

Hover Behavior

Highlights forward to the matching endLoop. That is all.

endLoop

Description

Used for loop highlighting.

Hover Behavior

Highlights backward to the matching startLoop. That is all.

comma

Description

Useful for functions and the like.

Hover Behavior

Highlights the cells immediately before and after this one.

API written by Andrew Duryea, Neil Vosburg, Jacob Burt, James Miltenberger, Richard Waller, and Tommy Bozeman.