# FlowPaper Annotations API

The following methods are exposed by FlowPaper Annotations and can be accessed through Javascript and Adobe Flex.

## addMark(mark) : void
The mark which is passed in is drawn in the document. If it is already drawn, no action is taken. The mark is created as a note if the note property has been populated on the mark object (see Data Structures).

## addMarks([mark,...]) : void
As above except as input an array of marks is used.

## scrollToMark(mark) : void
The document is scrolled such as the passed in mark is visible.

## removeMark(mark) : void
The given mark is removed (undrawn) from the document. If it doesn't exist, no action is taken.

## clearMarks() : void
All marks are cleared from the document.

## getMarkList():[mark,...]
Returns an array of marks (See Data Structures) that has been created in the document.

## createMark(color) : mark
If a selection exists, the selection is converted to a highlight and a mark is returned. The color is added to the returned mark object. If the selection doesn't exist, a mark is returned with a reference to the current page (see Data Structures).

# Events

## onMarkClicked(mark) : void
Thrown when the user clicks on a drawn mark within FlowPaper.

## onMarkCreated(mark) : void
Thrown when a mark has been created.

## onMarkDeleted(mark) : void
Thrown when a mark has been deleted.

## onMarkChanged(mark) : void
Thrown when a mark has been changed. The javascript callback for this function is delayed by 5 seconds to avoid excessive calls over the flash/javascript bridge.

# Data Structures

A mark can be created either as a selection or as a note. The data structures below show the two different types of marks and their corresponding properties.

## Mark *(type: highlight)*

Contains the text of the selection and the necessary info required in order to draw/find the highlight again in the document.

{

**id:** A unique identifier for the mark. Will be generated if not set or set to null.
**selection_text:** The text of the selection.
**has_selection:** *true/false* based on whether text has been selected.
**color:** HEX String representing color (e.g #FFAABB)
**selection_info:** String representing a selection as such “page number;start index;end index” where start and end index is where the selection starts and ends.
**color:** Color to be used/which has been used for the highlight
**readonly:** Makes the selection read-only.
**type:** Describes the mark as “highlight”.
**displayFormat:** Used for compability and gets set to either flash or html depending on where the mark got created

}

## Mark *(type: strikeout)*

Contains the text of the selection and the necessary info required in order to draw/find the highlight again in the document.

{

**id:** A unique identifier for the mark. Will be generated if not set or set to null.
**selection_text:** The text of the selection.
**has_selection:** *true/false* based on whether text has been selected.
**color:** HEX String representing color (e.g #FFAABB)
**selection_info:** String representing a selection as such “page number;start index;end index” where start and end index is where the selection starts and ends.
**readonly:** Makes the selection read-only.
**type:** Describes the mark as “strikeout”
**displayFormat:** Used for compability and gets set to either flash or html depending on where the mark got created

}

## Mark *(type: note)*

Contains the text of the note and the necessary info required in order to position the note again in the document.

{

**id:** A unique identifier for the mark. Will be generated if not set or set to null.
**note:** The text of the note
**pageIndex:** The page of where the note is positioned
**positionX:** The X position of the note
**positionY:** The Y position of the note
**width:** The width of the note
**height:** The height of the note
**collapsed:** Indicating whether the note shall appear as collapsed (icon) or expanded
**readonly:** Makes the mark read-only.
**type:** Describes the mark as “note”
**displayFormat:** Used for compability and gets set to either flash or html depending on where the mark got created

}

# Mark *(type: drawing)*

Contains the points about a drawing

{

**id:** A unique identifier for the mark. Will be generated if not set or set to null.
**color:** HEX String representing color (e.g #FFAABB)
**pageIndex:** The page of where the drawing is positioned
**readonly:** Makes the selection read-only.
**points:** The drawing points that make up the drawing in the form of “X1,Y1:X2,Y2”
**type:** Describes the mark as “drawing”
**displayFormat:** Used for compability and gets set to either flash or html depending on where the mark got created

}