Software Developer Guide

SigWeb

Version 1.5
Update: April 17, 2018

Table of Contents

Overview

SigWeb™ is a Browser API that allows for rapid integration of signature capture functionality into a web app using Firefox, Chrome, and Internet Explorer 11 web browsers under Windows.

The SigWebTablet.js File

The SigWeb Browser API requires the SigWebTablet.js file to function. You will need to add a <script> tag to your page to reference SigWebTablet.js. You will find the latest SigWebTablet.js file at this link: **http://www.sigplusweb.com/SigWebTablet.js**

An example <script> tag can be found below. This example assumes that the .js file is in the same location as the page using SigWeb on the server. Note: For best results and to allow support, Topaz recommends that you do not alter SigWebTablet.js.

```
<script type="text/javascript" src="SigWebTablet.js"></script>
```

SigWeb Functions

AutoKeyAddData()

```
public void AutoKeyAddData(
  string KeyData
);
```

Remarks: Adds data to the auto key generation function. SigWeb uses this data to generate a key which will ultimately encrypt the signature to the data.

Parameters: String containing the data, to be added to the key generation.

ClearSigWindow()

```
public void ClearSigWindow(
  short Inside
);
```

Remarks: Erases data either inside or outside of sig window based on value of short inside.

Parameters: inside-if=0 then signature data is erased (in window), if =1 then data outside sig window is erased.

ClearTablet()

```
public void ClearTablet();
```

Remarks: Clears the signature object of ink.

GetDisplayPenWidth()

public int GetDisplayPenWidth();

Return Value: Current pen ink width for the displayed signature in pixels.

GetEncryptionMode()

public int GetEncryptionMode();

Remarks: Returns current EncryptionMode.

Return Value: Numeric value of encryption mode, 0 = no encryption, 1 = medium encryption, 2 = higher security encryption mode.

GetImagePenWidth()

public int GetImagePenWidth();

Remarks: Gets current pen ink width.

Return Value: Pen ink width for Image.

GetImageXSize()

public int GetImageXSize();

Remarks: Gets the current width in X pixels of the image.

Return Value: Number of X pixels of image.

GetImageYSize()

public int GetImageYSize();

Remarks: Gets the current height in Y pixels of the image.

Return Value: Number of Y pixels of image height.

GetJustifyMode()

public int GetJustifyMode();

Remarks: Gets the current justification mode – how the signature is sized and positioned in the signature box.

Return Value: Justification mode, 0-normal no justification, 1-justify and zoom signature (upper left corner) 2-justify and zoom signature (upper right corner) 3-justify and zoom signature (lower left corner) 4-justify and zoom signature (lower right corner) 5-justify and zoom signature (center of control).

GetJustifyX()

public int GetJustifyX();

Return Value: Justification X buffer size in pixels for display.

GetJustifyY()

public int GetJustifyY();

Return Value: Justification Y buffer size in pixels for display.

GetKeyString()

public string GetKeyString();

Remarks: Provides hash of the encryption data in ASCII compatible format.

Return Value: Hash of encryption data.

GetLCDCaptureMode()

public int GetLCDCaptureMode();

Remarks: Gets the current LCD Capture Mode for the tablet.

Return Value: Mode the LCD is set to capture signatures in, Mode 0 no LCD commands are sent to the tablet, Mode 1-sets capture mode to be active with Autoerase in the tablet, Mode 2-sets the tablet to persistent ink capture without autoerase, Mode 3-signature ink is displayed inverted on a suitable dark background set using the Graphic functions.

GetSigCompressionMode()

public int GetSigCompressionMode();

Remarks: Returns compression mode for signatures.

Return Value: Mode for compression of signature, where 0= no compression, 1= lossless compression with compacted data format, 2-8= compression ratio of signature stored in in .sig file where 2=1KB typ, 4=500 byte typ, and 8=250 byte typ. Topaz Systems does not recommend compressing beyond setting 1 unless size is more important than signature quality.

GetSigImageB64()

public void GetSigImageB64 (callback)

Remarks: Returns the signature as a PNG image in base64 string format. Argument is a callback function name with which to pass the image string.

Example: *An example of the call to GetSigImageB64 is below:*

```
GetSigImageB64(SigImageCallback);
```

An example of the callback function getting the image string:

```
function SigImageCallback( str )
{
  document.FORM1.sigImageData.value = str; //assign Base64 string to form field
}
```

Return Value: String.

Note: Please be sure to add the callback function directly as javascript to your webpage in a <script> tag.

GetSigString()

public string GetSigString();

Return Value: SigString as ASCII hex string.

GetTabletBaudRate()

public int GetTabletBaudRate();

Return Value: Current TabletBaudRate.

GetTabletComPort()

public int GetTabletComPort ();

Return Value: Current COM port setting.

GetTabletComTest()

public int GetTabletComTest ();

Remarks: Gets current hardware check mode, can be used to determine if tablet is connected.

Return Value: Current hardware check mode, True if active, False if not active.

GetTabletLogicalXSize()

public int GetTabletLogicalXSize();

Return Value: Current horizontal values used in representing signatures in Logical Tablet Coordinates.

GetTabletLogicalYSize()

public int GetTabletLogicalYSize();

Return Value: Current vertical values used in representing signatures in Logical Tablet Coordinates.

GetTabletResolution()

public int GetTabletResolution();

Return Value: Current TabletResolution.

GetTabletState()

public int GetTabletState();

Remarks: Indicates capture state of the tablet.

Return Value: Value of 1 enables the component to access the selected COM or USB port and access the tablet for signature capture, 0 disables the tablet for capture.

GetTabletType()

public int GetTabletType();

Remarks: Gets TabletType value.

Return Value: Integer value of TabletType. See **SetTabletType**.

GetTabletXStart()

public int GetTabletXStart () ;

Return Value: Current X position in Logical Tablet Coordinates of the upper left hand corner of the component signature box.

GetTabletXStop()

public int GetTabletXStop();

Remarks: Gets the X pos in Logical Tablet Coordinates of the right most X pixel.

Return Value: The X pos of the right most pixel.

GetTabletYStart()

public int GetTabletYStart();

Remarks: Gets the current Y pos in Logical Tablet Coordinates of the top most X pixel.

Return Value: The Y pos of the top most pixel.

GetTabletYStop()

public int GetTabletYStop();

Remarks: Gets the current Y pos in Logical Tablet Coordinates of the bottom most X pixel.

Return Value: The Y pos of the bottom most pixel.

KeyPadAddHotSpot()

```
public void KeyPadAddHotSpot(
  short KeyCode,
  short CoordToUse,
  short XPos,
  short YPos,
  short XSize,
  short YSize
);
```

Remarks: Defines in software the location of a tablet HotSpot which is used by the developer to detect user pen taps.

Parameters:
KeyCode-Integer value defining the HotSpot.
CoordToUse-Coordinate system used for this HotSpot.
XPos-Location (upper left- 0,0)
YPos-Same
XSize-X size in pixels.
YSize-Y size in pixels.

KeyPadClearHotSpotList()

public void KeyPadClearHotSpotList();

Remarks: Clears the controls internal list of HotSpots created using **KeyPadAddHotSpot**.

KeyPadQueryHotSpot()

```
public short KeyPadQueryHotSpot(
  short KeyCode
);
```

Remarks: Queries whether the specified HotSpot has been tapped by the user. Returns a true if the control contains data that is within the definition of the keyCode on the tablet.

Parameters: KeyCode-Mapped Logical Tablet Coordinates.

Return Value: Number of points within the KeyCode definition.

LCDRefresh()

```
public bool LCDRefresh(
  int Mode,
  int XPos,
  int YPos,
  int XSize,
  int YSize
);
```

Remarks: Sends tablet a refresh command with 4 possible modes. Mode 0-Clear, display is cleared at the specified location. Mode 1-Complement, complements display at the specified location. Mode 2-WriteOpaque, transfers contents of the background memory to the LCD display, overwriting the content of the LCD display. Mode 3-WriteTransparent, transfers contents of the background memory in the tablet to the LCD display and combined in the contents of the LCD display.

Parameters:
Mode-Defined as above (0-4)
XPos-Location in LCD Coordinates (upper left-0,0)
YPos-Same
XSize-X size in LCD pixels
YSize-Y size in LCD pixels

Return Value: True if checksum received and verified, False if no or incorrect checksum received from tablet.

LCDSendGraphicUrl()

```
public void LCDSendGraphicUrl(
  int Dest,
  int Mode,
  int XPos,
  int YPos,
  Bitmap BitmapData
);
```

Remarks: This writes a bitmap image to the LCD from the URL specified.

Parameters:
Dest- 0=Foreground,1=Background memory in tablet
Mode-0-3 as defined in LCDWriteBitmap
XPos-Location in LCD coordinates (upper left- 0,0)
YPos-Same
BitmapData-URL to bitmap

Return Value: None.

LCDSetWindow()

```
public bool LCDSetWindow(
  int XPos,
  int YPos,
  int XSize,
  int YSize
);
```

Remarks: Sets a signature window that restricts the ink of the SigPlus object to said window on the LCD itself (see **SetLCDCaptureMode**).

Parameters:
XPos – Location in LCD coordinates (upper left – 0,0)
YPos – Same
XSize – X size in LCD pixels
YSize – Y size in LCD pixels
Return Value: True if checksum received and verified, False if no or incorrect checksum received from tablet.

LCDStringWidth()

```
public int LCDStringWidth(
  Font DrawFont,
  string Str
);
```

Remarks: Takes a string and a .NET font and hand back how wide the string is in pixels.

Parameters:
DrawFont - .NET font
Str – String

LCDWriteString()

```
public void LCDWriteString(
  int Dest,
  int Mode,
  int XPos,
  int YPos,
  Font WebFont,
  int FontHeight
  string Text
);
```

Remarks: Used to write a string to the LCD Display. The data is written at the location specified by the combination of Dest, XPos, and YPos. The Mode determines how the data is written.

Mode 0 - Clear: The Display is cleared at the specified location.
Mode 1 - Complement: The Display is complemented at the specified location.
Mode 2 - WriteOpaque: The contents of the background memory in the tablet are transferred to the LCD display, overwriting the contents of the LCD display.
Mode 3 - WriteTransparent: The contents of the background memory in the tablet are combined with and transferred to the visible LCD memory

Parameters:
Dest-0 = Foreground, 1 = Background memory in tablet
Mode-0, 1, 2, 3 as defined above
XPos-Location in LCD coords to draw at
YPos-Same
WebFont-Example: “10pt Arial”
Int-Height
Str-String to write to LCD
Return Value: None.

NumberOfTabletPoints()

```
public int NumberOfTabletPoints();
```

Remarks: Returns the total number of points in the current signature, can be used to detect if a signature is present or not.

Return Value: Decimal value of number of points in the signature.

SetDisplayMode()

```
public void SetDisplayMode(
  int Mode
);
```

Remarks: *NOT CURRENTLY IMPLEMENTED.*

SetDisplayPenWidth()

```
public void SetDisplayPenWidth(
  int PenWidth
);
```

Remarks: Sets pen ink width for the displayed signature in pixels.

Parameters: PenWidth – Pen width for the displayed signature in pixels.

SetDisplayXSize()

```
public void SetDisplayXSize(
  int pixels
);
```

Remarks: Sets the pixel width of the display for SigWeb.

Parameters: Int – Width in pixels.

SetDisplayYSize()

```
public void SetDisplayYSize(
  int pixels
);
```

Remarks: Sets the pixel height of the display for SigWeb.

Parameters: Int – Height in pixels.

SetEncryptionMode()

```
public void SetEncryptionMode(
  int EncryptionMode
);
```

Remarks: Sets EncryptionMode.

Parameters: EncryptionMode-0= no encryption, 1= medium encryption, 2=higher security encryption mode.

SetImagePenWidth()

```
public void SetImagePenWidth(
  int ImagePenWidth
);
```

Remarks: Sets pen ink width.

Parameters: ImagePenWidth-Pen ink width for Image.

SetImageXSize()

```
public void SetImageXSize(
  int ImageXSize
);
```

Remarks: Set the number of X pixels in the image.

Parameters: ImageXSize – Size in X pixels of the Image width.

SetImageYSize()

```
public void SetImageYSize(
  int ImageYSize
);
```

Remarks: Set the number of Y pixels in the image height.

Parameters: ImageYSize – Size in Y pixels of the Image height.

SetJustifyMode()

```
public void SetJustifyMode(
  int JustifyMode
);
```

Remarks: Sets the current justification mode- how the signature is sized and positioned in the signature box.

Parameters: JustifyMode - Justification mode, 0-normal no justification, 1-justify and zoom signature (upper left corner) 2-justify and zoom signature (upper right corner) 3-justify and zoom signature (lower left corner) 4-justify and zoom signature (lower right corner) 5-justify and zoom signature (center of control).

SetJustifyX()

```
public void SetJustifyX(
  int JustifyX
);
```

Remarks: Sets the buffer size in Logical Tablet Coordinates of "dead space" of left and right edge of SigPlus object if **JustifyMode** is 1-5. This method functions for both the signature box.

Parameters: JustifyX-Justification X buffer size in pixels to be set.

SetJustifyY()

```
public void SetJustifyY(
  int JustifyY
);
```

Remarks: Sets the buffer size in Logical Tablet Coordinates of "dead space" of top and bottom edge of SigPlus object if **JustifyMode** is 1-5. This method functions for both the signature box.

Parameters: JustifyY-Justification Y buffer size in pixels to be set.

SetKeyString()

```
public void SetKeyString(
  string KeyString
);
```

Remarks: Sets the Key String into the SigPlus component.

Parameters: KeyString - Hash of the data used to encrypt/decrypt the signature, key internally generated by SigPlus.

SetLCDCaptureMode()

```
public void SetLCDCaptureMode(
  int CaptureMode
);
```

Remarks: Sets the current LCD Capture Mode for the tablet.

Parameters:
CaptureMode-Mode the LCD is set to capture signatures in,
Mode 0=no LCD commands are sent to the tablet
Mode 1=sets capture mode to be active with Autoerase in the tablet
Mode 2=sets the tablet to persistent ink capture without autoerase
Mode 3=signature ink is displayed inverted on a suitable dark background set using the Graphic functions.

SetSigCompressionMode()

```
public void SetSigCompressionMode(
  int CompressionMode
);
```

Remarks: Sets the current compression mode for signatures.

Parameters: CompressionMode-Mode for compression of signature, where 0= no compression, 1= lossless compression with compacted data format, 2-8= compression ratio of signature stored in in .sig file where 2=1KB typ, 4=500 byte typ, and 8=250 byte typ. Topaz Systems does not recommend compressing beyond setting 1 unless size is more important than signature quality.

SetSigString()

```
public void SetSigString(
  string SigString
);
```

Remarks: Puts signature into the component.

Parameters: SigString – Signature in ASCII format

SetSigWindow()

```
public void SetSigWindow(
  short Coords,
  short NewXPos,
  short NewYPos,
  short NewXSize,
  short NewYSize
);
```

Remarks: This function sets a window in the logical tablet space that restricts the operation of some functions to the specified window. The functions behave as follows: **JustifyMode** will only operate on points inside of this window. **SigString** only operates on points inside of the window. **ClearTablet** will only clear in the window. This behavior is enabled by setting the start and stop values to non-zero. The window defaults to (0,0,0,0). The window can be enabled at one spot, re-enabled at another, etc., without disabling in between, and then disabled when the various parts of the tablet data have been separated and stored. To determine the logical values in the control for the installed tablet, see the **TabletLogicalXSize** and **TabletLogicalYSize** properties.

Parameters:
Coords-Coordinate system used for this hot spot, 0 = Logical tablet coordinates, 1 = LCD Coordinates.
NewXPos-Location in logical tablet coordinates (upper left - 0,0).
NewYPos-Same
NewXSize-XSize in logical tablet pixels
NewYSize-YSize in logical tablet pixels

SetTabletBaudRate()

```
public void SetTabletBaudRate(
  int BaudRate
);
```

Remarks: Sets TabletBaudRate, an internal property associated with tablet model.

Parameters: BaudRate – internal tablet property.

SetTabletComPort()

```
public void SetTabletComPort(
  int Port
);
```

Remarks: Sets the COM port to use using a string. The SigPlus.NET component does not lock up a port as is the case with mouse-type drivers. Only set COM port when tablet state is OFF.

Parameters: Port-

SetTabletLogicalXSize()

```
public void SetTabletLogicalXSize(
  int XSize
);
```

Remarks: Sets the range of horizontal values to be used in representing signatures. This has no relation to the displayed, image file, or tablet sizes. This is the X-range used for the Topaz vector format, and the internally used format.

Parameters: XSize – Integer value of Tablet logical size. Default is 2150.

SetTabletLogicalYSize()

```
public void SetTabletLogicalYSize(
  int YSize
);
```

Remarks: Sets the range of vertical values to be used in representing signatures. This has no relation to the displayed, image file, or tablet sizes. This is the Y-range used for the Topaz vector format, and the internally used format.

Parameters: YSize – Integer value of Tablet logical size. Default is 1400.

SetTabletResolution()

```
public void SetTabletResolution(
  int Resolution
);
```

Remarks: Sets TabletResolution, an internal property associated with tablet model and set by the TabletModel property, based on hardware tablet resolution is 410 dpi, (excluding ClipGem which is 275 dpi) but can be changed at the risk of affecting signature capture.

Parameters: Resolution – internal tablet property.

SetTabletState()

```
public void SetTabletState(
  int State
);
```

Remarks: Enables tablet to access the COM or USB port to capture signatures or not.

Parameters: State-setting to 1 enables the tablet to capture signatures as above, setting to 0 disables signature capture.

SetTabletType()

```
public void SetTabletType(
  int TabletType
);
```

Remarks: Determines if the tablet will accept data from a com port, WinTab driver, USB driver or other method of data input. If WinTab support is not available, it will not do anything when in the active state. Conversely, if WinTab is present and the mode is not correct, the tablet will also do nothing when active, because the WinTab driver takes exclusive possession of the Com Port. This is the preferred way of setting TabletMode for tablet input.

Parameters:
TabletType-Default is 0
0=Normal mode. When tablet is activated it will accept input from the selected com port.
1=WinTab mode, when the tablet is activated, it will accept data from the Topaz WinTab driver only.
2=USB mode, when the tablet is activated, it will accept data from the Topaz USB driver
3=TracGemPOST signature format, Transparent Mode (CTRL T)
4=Older-model SigLite touch tablet format, an obsolete mode
6=HSB tablet (USB mode for tablets using HID driver)

SetTabletXStart()

```
public void SetTabletXStart(
  int XStart
);
```

Remarks: Sets the X position in Logical Tablet Coordinates of the upper left hand corner of the bean signature box.

Parameters: XStart – X coordinate of the upper left corner of the signature box.

SetTabletXStop()

```
public void SetTabletXStop(
  int XStop
);
```

Remarks: Sets the X position in Logical Tablet Coordinates of the lower right hand corner of the bean signature box.

Parameters: XStop – X coordinate for the lower right corner of the signature box.

SetTabletYStart()

```
public void SetTabletYStart(
  int YStart
);
```

Remarks: Sets the Y position in Logical Tablet Coordinates of the upper left hand corner of the bean signature box.

Parameters: YStart – Y coordinate for the upper left corner of the signature box.

SetTabletYStop()

```
public void SetTabletYStop(
  int YStop
);
```

Remarks: Sets the Y position in Logical Tablet Coordinates of the lower right hand corner of the bean signature box.

Parameters: YStop – Y coordinate for the lower right corner of the signature box.

SigWebSetDisplayTarget()

```
public void SigWebSetDisplayTarget(
  canvas
);
```

Remarks: The canvas' context used to display the signature in real time.

Parameters: Canvas context.