StockChartX HTML5

stxSocial Image Creation and Sharing Library

Version 4-2013

# About

The StockChartX stxSocial HTML5 library is an add-on to the StockChartX HTML5 library. It provides functions for creating images from stock charts and for sharing those charts via a server upload.

# Contents

stxSocial.js.zip contains the following files:

stxSocial.js – The source client library. Include this from within your application.

upload.py – An reference server side program to accept and serve shared images.

sample.html from the StockChartX HTML5 library contains sample code for using stxSocial.js.

# Usage

The library has two main functions:

STXSocial.createImage(stx, widthPX, heightPX, decorationObj, cb)

Use this method to create an image from the currently loaded chart. Width and height are optional parameters. The decorator is also optional and described below. This method is asynchronous. The image will be passed back through the callback parameter. This must be a function that you've provided to accept the image, such as:

function cb(dataImage){

}

The dataImage that is passed back is a base64 encoded representation of a png image. This data can be passed to a server or it can be transformed into an actual image for display in the browser (see sample.html for code to transform to an image).

STXSocial.uploadImage(dataImage, url, payload, uploadCb)

dataImage is the image received previously from createImage. Url is the url of the server to upload. Payload is an optional object that contains additional data to send to the server.

UploadImage is asynchronous so you must provide a callback function in the following format:

function uploadCb(err, response){

}

Images are uploaded to the server via ajax. The callback function will have err=null if the upload was successful. Response will contain the response from the server (by convention this would be the new url for the shared image).

# Server

Included is a reference sever called upload.py. This is a server written in python that accepts uploaded images, stores them on disk, returns the (relative) url of the image, and then serves up any images that have been stored. You must have python 2.x installed to use this application. Run it with:

python upload.py

It is set to run on port 8000 but you can change this to any port you desire. This is a very basic server written using Python's wsgi interface. We do not recommend the use of this server specifically although using modwsgi you can plug this into apache and use it off the shelf for storing images (we recommend disabling serve\_static in this scenario).

STXSocial.uploadImage will always upload a JSON object that contains a member called “image” which contains the base64 representation of the png image. If you've passed in a payload object, then this member will be added to that object. Payload objects can be used to pass metadata to the server.

In our example code, the client creates a random url using UniqueID() and the server accepts that url. You may choose to implement your server so that it determines the url. Likewise, in our example code the url is relative whereas you may desire for the server to create an absolute url. The example code is meant only as a very simple example.

# Decorator

Plain old images aren't much fun. The STXSocial.Decoration object can be used to add headers and footers to the image. The default decorator adds a header that contains the symbol, date range and periodicity. You can copy and paste the default implementation and create your own decorator if you'd like to share something more complex, or add a footer that contains your company's branding.

The decorator takes an initialize() function which should be used to set the headerPX and footerPX sizes. This is critical for getting the size of the image correct.

decorate() does the actual decoration. This method draws directly on the canvas and is called after the chart has been rendered on the canvas. As such any drawings that are done outside of the allocated header and footer zones will display on top of the chart.

The default decorator uses convenience functions for pulling fonts from a style sheet and for plotting lines. Any StockChartX canvas tools can be used here but you can also use raw HTML canvas drawing methods as well.