

Client js v 1.2.3 Beta

GSLIB

Document version : 1.2.3. Beta | Date : 24th July 2020 10:32am

**ABOUT THIS DOCUMENT AND LIBRARY**

This JavaScript library is a result of perseverance and hard work of the developers and good will from the library community of users. It started as a simple mobile based function or method oriented script to enable quick and fast code usage and avoid code repetition back in 2017.

The initial version (gslib\_main.js) has simple and basic usages but still did the job quite well. This library will at its best simplify most of your work into small methods that you can call and manage easily.

We are still managing or updating the library as of now and if you would like to be part of the development community email your suggestions and code attachment to the following email : [developers@greenswitchdigital.xyz](mailto:developers@greenswitchdigital.xyz) with Subject as “GSLIB\_CLIENT UPDATE APPLICATION”. Your code will be reviewed and when it passes it will be included as a patch in the later versions and then you will be notified per such changes.

**TABLE OF CONTENTS**

[Introduction 4](#_Toc42592773)

[Chapter 1 : get and set 5](#_Toc42592774)

[Chapter 1.1 get properties 5](#_Toc42592775)

[Chapter 2. Toast 6](#_Toc42592776)

[Toast Usage 7](#_Toc42592777)

[Object Direct Manipulation Functions[ODMF] 7](#_Toc42592778)

[Chapter 3 : Storage Class 8](#_Toc42592779)

[Chapter 4 : connect() function for AJax 9](#_Toc42592780)

[Chapter 5 : create method 9](#_Toc42592781)

[Chapter 6 : include("css.animations ") [Deprecated] 10](#_Toc42592782)

[Chapter 7 : \_ready() or \_main() 10](#_Toc42592783)

[Chapter 8 : onVisible() function 10](#_Toc42592784)

[Chapter 9 : createProgress() 11](#_Toc42592785)

[The Progress Object 11](#_Toc42592786)

[CHAPTER 10 (new) : Signature Pane 12](#_Toc42592787)

# Introduction

Green Switch javascript library is a light weight javascript library that aims at simplifying your web app development time by bringing to you complex visual presentation of html components with just a few lines of code. Green Switch was also designed to simplify html manipulation through simplified methods / functions that are easy to understand and implement.

This library, hence forth can be used for, android hybrid applications, desktop applications, websites or any web based system. Developed by me, (Wongani Kaluwa), this is a free gift to you and hence share the gift to others.

**Systems Running the Library**

1. Apps265.com
2. greenswitchdigital.xyz
3. Links Antivirus 2.0.3
4. Kodi Studio IDE
5. Digital Footprints Website
6. Sports 265
7. Links Antivirus website
8. Links me In Application
9. And many more…

# Chapter 1 : get and set

-get(“object\_id or object\_class\_name”)

Info : Gets an HTML object tag from within a document either from ID or ClassName

Return type : returns HTMLObject

Arguments : id or className

Example:

var c=get(“.myClassName”)

var I=get(“#myId”) or get(“myId”)

-set(“parent\_object”,”html\_object”)

Info : appends a child html\_object to another object

Return type : none or null

Arguments: parent\_html\_object, object\_to \_append

Example:

var html\_object=get(“myId”)

var newH3=create(“h3”)

newH3.innerHTML=”Hello my world”

set(html\_object,newH3)

## Chapter 1.1 get properties

Info : when ever a ‘get’ method is called, a set of easy to use properties have been included in the new 1.2.2 greenswitch version that will also allow you to perform different tasks easily and faster. This also works great for the create() function

For example, calling a get method

var m=get(“myId”)

m.hide()

This will eventually hide the html element with myId id.

Other methods or functions include the following,

|  |  |
| --- | --- |
| Method or Function | Action |
| .show() | Displays the element |
| .hide() | Hides the element |
| .fadeOut() | Slowly fades the element out |
| .fadeIn() | Slowly fades in the element |
| .slide(width,delay) | Animates the width property with defined delay, default is 300ms |
| .getWidth() | Returns object width in pixels |
| .getHeight() | Returns height |
| .setColor(“rgb | Hex | rgba”) | Sets font color |
| . setBackgroundColor (“color”) | Sets background color |
| .setText(“text here”) | Sets the specified text despite it state |
| .animate(“myanim 2s linear”) | Sets css animation keyframes to object |
| .setWidth(width) | Sets the width of the object |
| .setHeight(height) | Sets the height of the object |
| .longClick(duration,function) | Creates a long click event Listener for the object set to this method. For Duration you can also use Toast dutation timers eg. SHORT,MEDIUM and LONG  The duration is the actual delay to call you event after long pressing the element. |

# Chapter 2. Toast

Greenswitch now introduces a new Toast feature similary to the android toast. This is a cross platform way of displaying toast messages to the user interface for a specific period of times.

## Toast Usage

Toast(“Message you need to toast, can be html or not”, duration\_in\_seconds)

Example:

Toast(“Hello GS Toast”,3)

This will show a 3 seconds Toast in your user interface

However, the last argument is not a mandatory as 3 is the default value if it is forgotten. Other arguments includes SHORT,MEDIUM,LONG are the greenswitch default values ready to be parsed e.g

Toast(“Hello default Digit”,MEDIUM)

# Object Direct Manipulation Functions[ODMF]

Green Switch also brings to you a quicker and faster ways of manipulating elements

|  |  |
| --- | --- |
| Method | Description |
| setValue(“id/class”,data) | Sets text to element |
| getValue(“id/class”) | Returns text from object |
| setTransition(“id/class”,delay\_in\_ms) | Sets transition delay for an object (Universal) |
| slide(“id/class”,width,delay) | Creates an animation effect for a sliding object, good for menus or submenus |
| slideObj(html\_obj,width,delay) | Slides the object, same as above |
| scale(“id/class”,size) | Scales the object |
| setBg(“id/class’,”color”) | Sets Background Color |
| printf(“text”) | Debugging sets text onto ui, sometimes clears the whole screen for you |
| fadeIn(“id/class”) | fadeOut(“id/class”) | Fades in and out of an element |
| hide(“id/class”) || destroy(“id/class”) | Hides an element |
| setImage(“id/class”,”location”) | Changes image location of an Image element |
| setBackgroundImage(“id/class”,”location”) | Sets background Image of an element |
| marginTop(“id/class”,value) | Sets MarginTop of an element |
| ab(value) | Returns absolute value of an element |
| randomInt(min,max) | Generates a random integer within specified range, return integer (non decimal value) |
| random(min,max) | Return a random float number, despite the value |
|  |  |

# Chapter 3 : Storage Class

This class, originally developed for Kodi Studio was then ported to greenswitch library to allow you to read and manage cookies and local storage data efficiently

To use this class, you need to instatiate it

var storage=new Storage()

Methods includes the following

|  |  |
| --- | --- |
| .write(“dataKey”,”dataValue”) | Stores data to local storage with dataKey to be used for refferencing and dataValue and the data itself |
| .read(“dataKey”) | Reads local storage data (dataValue) with given dataKey |
| .writeCookie(“dataKey”,”dataValue”) | Stores a cookie with dataKey to be used for refferencing and dataValue and the data itself |
| .readCookie(“dataKey”) | Reads cookie data (dataValue) with given dataKey |
|  |  |

**Example :**

var storage=new Storage()

storage.write("username","myusername@thisdata.com")

Toast(storage.read("username"),LONG)

This will create a new local storage data with key username and data [myusername@thisdata.com](mailto:myusername@thisdata.com)

NOTE: Using this method might put your websites private data at risk as hackers can access this data, it is recommended to encrypt your information before saving it on user’s machine and frequently clear cache and history.

# Chapter 4 : connect() function for AJax

This method has went through a lot till this next update where you get to control what you receive from other pages. Want to do a background process while giving your user a sense of safety? Use this method as it connects to different files, pages or even website!

This method receives four important arguments,

1. The Link
2. Loading panel
3. Debug on / off
4. Callback method after the whole process

Example :

connect ("http://my\_file.txt ", " ",false, function(outcome){

//outcome variable is the response from the my\_file.txt which can be in json format, html, or plain text

})

//see example connect from the gslib example folder

# Chapter 5 : connect\_s() function for Ajax

The new update comes in with this secure and easy to use AJAX method that you can use for different purposes including file uploading options. In order to use this method or feature, you need to create a secure connection object that comes with a few distinct arguments and properties.

eg. let connection = {

url: ‘Link to your php, txt, html page’,

arg: {‘name’ : ’Jon Doe’, ’file’:getFiles(‘#file\_input’)},

success: (response)=>{

//handle response here

},

error : (error\_message)=>{

//handle the error message to client here

}

}

connect\_s(connection)

# Chapter 6 : create method

Create html object with this method easily

Eg. var h3=create(“h3”)

This will create an anonymous h3 tag ready to be appended on a parent, the arguments includes a string value depicting the element tag to be created.

Paste this object on any element within the document using the set method

# Chapter 7 : include("css.animations ") [Deprecated]

This function allows you to import predefined css animations into your page, that help into animating elements of the whole document. To use it efficiently, make sure it is called before the page loads

# Chapter 8 : \_ready() or \_main()

Every system has an entry point and we havent left you out alone, so introducing greenswitch’s own entry point. This function must be overriden in you javascript page and what it does is to be executed once the whole document is ready!

Usage

function \_ready(){

//this will be called once the page is loaded

Toast("Am ready now ")

}

You can use \_ready() or \_main() for your function, its your call

# Chapter 9 : onVisible() function

Ever wondered why some elements seems to know when they are out of view or not? Its simple now with the onVisible method which detects if the parsed element in within view or out of view by presenting its visibility in percentages, this is when you can do further manipulation after appearing 50% on screen

Example

onVisible("id/class ",function(state,percentage){

//state is either true for visible on screen or false when out of view

//percentage is the value in percent of how the element is visible of the browser

} );

# Chapter 10 : createProgress()

This is one of the most powerfull feature of greenswitch that even calls me to continue using it. The create progress method will animate, and create a beautiful animated circular progress bar that you can maintain and display on your interface through the connect method too!. This method receives one object variable called the progress data and it is the created on the specified element.

Usage

createProgress (progress)

You might be wondering where the variable progress is meaning, well it is the object having the way your progress looks ,et.

## The Progress Object

This are some of the progress bar object methods and properties

|  |  |
| --- | --- |
| Property / function | Description |
| Id | The element id |
| Title | Title of the progress bar, shown as a tooltip |
| Subtitle | Subtitle of the element |
| titleEnabled [true | false] | Enables the title to be visible or not, default is false |
| bgColor | Background Color of the element, default is white |
| progressColor | Color of the progress bar, default is #a50080 |
| value | Curent progress value |
| maxValue | Maximum value, default is 100% |
| thickness | The thickness of the progress bar |

Usage

Var progress={

id: "myId",

title: "Title here",

bgColor: "red",

progressColor: "#ccc",

value:20,

maxValue:27

}

createProgress(p)

# CHAPTER 12 (new) : Signature Pane

When is the right time to create a panel where you can capture users signature on almost every device? Simple, with gslib\_client version 1.2.3 a new feature was introduced where you can by all means with few code create a panel to capture signatures for your system then upload it on your server.

Usage:

First of all create a canvas element from html

HTML:

<canvas id=”mycanvas” >Sorry your browser does not support the canvas</canvas>

Then write these few lines of code in javascript and you are done

//create the signature pane object

var signaturePane=new SignaturePane()

//initialize the canvas with an id element of the canvas you created earlier

signaturePane.init(“mycanvas”)

//once you initialize is, you can draw the canvas with your mouse.

NOTE: You can setup manual buttons that will either request to save the image or clear the screen by using the following methods

//when this button is clicked, the signature panel will be erased or cleaned

signaturePane.setClearButton(“id\_of\_button”)

//when the set button is clicked, a function will be called parsing the image data back

signaturePane.setSaveButton(“id\_of\_button”,function(image){

//you can set the src of a button by pointing it at the image passed in this function

setImage(“my\_image\_id”,image)

})

This is part of the documentation, If you like what you see and would like to support the team of developers through support@greenswitchdigital.xyz or Whatsapp

+265 88 293 13 67 or +265 99 79 38 102

Don’t forget to try greenswitch java-js API