

Forum:[Hack Forums](#)

Thread: [How To Make Your Own Userscripts!](#)

Emylbus - July 21, 2013, 12:05 pm

Contents

Introduction

[Document Object Model](#)

[jQuery](#)

[Userscript Metadata Block](#)

[Console Logging](#)

[Adding CSS Styles](#)

[Local Data Storage](#)

[Sending Webpage Requests](#)

[Easy Examples](#)

[More Resources](#)

Userscripts are scripts that increase the functionality of websites. Maybe you want to change the background color of your favorite site or perhaps you want a full blown revamp, userscripts are for you! Throughout this tutorial I will show you everything you need to know to get into making your own userscripts :D

Getting Greasemonkey

First thing's first, you should understand that userscripts are generally run through a script manager such as Greasemonkey. These will let you organize, install, edit, and update userscripts and are absolutely necessary to have if you want to get into this wonderful world full of customization. Click on one of the links below for the correct manager for your browser:

I recommend using Chrome's Tampermonkey, I find that Chrome's developer tools are simply more useful.

Installing Scripts

The largest source of userscripts is, without a doubt, [userscripts.org](#). Once you select the script you wish to install and you go to the install page, your script manager will ask you if you wish to install it. Make sure you review the permissions you're giving the script and press ok.

Starting a New Script

Greasemonkey

1) Select the Greasemonkey button and choose the "New User Script..." option.

2) Enter the relevant information. The only important one right now is the Name, the rest you can easily change later. For a description of what each of these options are, jump to [this section of the tutorial](#).

3) Simply add your code in the editor that pops up :D

Tampermonkey

- 1) Open the Tampermonkey dashboard.
- 2) Click on the new script button.
- 3) Simply add your code in the editor that pops up :D

Writing a New Script

Unfortunately I don't have the time nor space in this tutorial to teach you how to program. Luckily JavaScript is easy enough that if you just tinker with what's out there, you'll quickly learn the ropes. Find a simple userscript and change one thing and see how that makes the output different. What I will do is explain how userscripts work and show you the tools that are available so you can learn and create the next great program.

Good luck gentlemen and gentleladies. I'll see you on the other side!

Emylbus - July 21, 2013, 12:05 pm

[Top](#) Document Object Model

The Document Object Model (DOM) is the most commonly used convention for handling and interpreting objects in HTML documents. In DOM, a webpage is broken up into a family-tree where elements within other elements are considered the children of their upper level, or parent, elements: 

Image courtesy of w3schools.com

Here is a 3D representation of the DOM tree in website as complicated as Hack Forums:

As you can see from the blurry picture, the internet is a complicated place and the DOM convention lets us choose exactly what we are looking for.

Methods

Luckily there are methods built into JavaScript which let us traverse the DOM tree. Below I have listed some I frequently use, but if you want a full list please visit [HTML DOM Document Objects | w3schools](#).

`document.body` Returns the body element of the document.

`document.getElementsByClassName(someClass)` Return the elements with the class name defined by the string `someClass`.

`document.getElementById(someID)` Return the element with the id defined by the string `someID`.
[w3school's interactive example](#)

`document.getElementsByName(someName)` Return the elements with the name defined by the string `someName`.

[w3school's interactive example](#)

`document.getElementsByTagName(someTag)` Return the elements with the tag defined by the string `someTag`.

[w3school's interactive example](#)

`document.links` Return an array of all links in the current document.

[w3school's examples](#)

`document.title` Returns or sets the title of a webpage.

PHP Code

```
<?php
```

```
GM_log(document.title); // Prints the page title
```

```
document.title = "A New Title"; // Sets the current page title
```

```
document.URL Returns the full URL of the document.
```

Despite having these tools at our disposal, using this method of DOM traversal is inefficient and often may break certain features of websites. I would strongly recommend, after understanding how DOM works, that you use the jQuery library to quickly and efficiently traverse and modify DOM elements. I will be covering jQuery in the next section.

Emylbus - July 21, 2013, 12:06 pm

[Top](#) jQuery

jQuery is an incredibly popular library designed to simplify JavaScript with numerous features. Features include the easy DOM traversal and modification, event handling, effects and animations, and various utilities. When writing userscripts, you'll find jQuery will simplify finding and manipulating the document.

You can include the jQuery into your script with the following @require line:

PHP Code

```
<?php
```

```
// @require http://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js
```

Here is a list of other versions of jQuery you may use by simply replacing the version number in the above @require line:

Code 2.0.3, 2.0.2, 2.0.1, 2.0.0, 1.10.2, 1.10.1, 1.10.0, 1.9.1, 1.9.0, 1.8.3, 1.8.2, 1.8.1, 1.8.0, 1.7.2, 1.7.1, 1.7.0, 1.6.4, 1.6.3, 1.6.2, 1.6.1, 1.6.0, 1.5.2, 1.5.1, 1.5.0, 1.4.4, 1.4.3, 1.4.2, 1.4.1, 1.4.0, 1.3.2, 1.3.1, 1.3.0, 1.2.6, 1.2.3

Filtering

Perhaps the most useful part of jQuery that we will be using frequently is the DOM filtering. Faster than doing it manually, you can specify your search by tags, class name, or id name and jQuery will return an array with everything matching your terms. You can then manipulate them however you wish.

Everything in jQuery starts with this syntax (where aString is the string containing your filter terms):

PHP Code

```
<?php
```

```
$(aString);
```

To filter for a certain tag (i.e. the <div> tag), you would simply make your search term "div". If you wished to search by class name, you would prefix your term with a period and get ".someclass". If you wished to sort by id name, you can do "#someid". Here are some examples of search terms and their results:

As you can see, some search terms are more specific than others and you'll want to be as specific as possible when determining what you want to prevent accidentally causing unplanned and unwanted changes.

You can specify a child tags/class/id in addition to their parent, helping limit what results you get back. You do this by simply delimiting the parent and child filter with a space:

There are also things called "selectors" which let you be even more specific with your filtering. However it is not within the scope of this tutorial to explain everything in jQuery, and I will redirect you to the official jQuery documentation for more information: [Selectors | jQuery API Documentation](#)

Manipulation

Once you have specified the elements you want to mess with, you can run built in functions to change their behavior. You can add or change attributes, you may change the text inside them, or do anything else you'd like. I can't possibly list all of jQuery's functions, but here are some of the ones I use most frequently:

`.append(content)` Insert content in the form of a string, element, or jQuery object at the end of the set of matched elements defined by your filter.

[jQuery API doc](#)

`.attr(attributeName, [value])` Set one or more attributes for the matched elements.

If you do not include the value, the function will return the attribute specified without changing anything.

[jQuery API doc](#)

`.html([astring])` If the astring parameter is passed, this function will replace the html of the matched elements. If it is not, then this function will return the html in the matched elements.

[jQuery API doc](#)

`.prepend(content)` Similar to `.append()`, however you insert content at the beginning of each matched element.

[jQuery API doc](#)

`.remove([selector])` This will remove the matched elements. You may include an optional selector to help specify which elements to remove.

[jQuery API doc](#)

`.text([astring])` Similar to `.html()`, however you may return or change the text within matched elements depending on whether or not you include the astring parameter.

[jQuery API doc](#)

You may find a list of all manipulation functions here: [Manipulation | jQuery API Documentation](#).

There are many other functions that may help you learn how to use jQuery that I simply can't get into in this tutorial. I would strongly recommend you check out the full documentation and learn the amazingness that is called jQuery: [jQuery API Documentation](#)

Emylbus - July 21, 2013, 12:07 pm

[Top](#) Userscript Metadata Block

The Metadata Block is an essential and necessary part for every userscript. Located at the top of every script, this section has important information including the name of the script, the author, and other vital things.

Syntax

The syntax for the block is fairly straight forward: everything is placed into comments at the top of your script.

PHP Code

```
<?php
// ==UserScript==
// @key value
// ==/UserScript==
```

Imperatives@description A brief summary of what the script does. It will be shown to the user during installation of the script.

PHP Code

```
<?php
// @description This is one awesome tutorial!
```

@downloadURL Define the URL to download updates if the userscript manager determines there is an update after checking the URL defined by **@updateURL**.

PHP Code

```
<?php
```

```
// @downloadURL https://www.example.com/superscript.user.js
```

@exclude Define which URLs your script will not be activated on. The **@exclude** may be used to let you specify sections of a certain website you do not want the code to run on.

You may find a more detailed list of rules and tricks for **@exclude** here: [Include and exclude rules - GreaseSpot Wiki](#)

PHP Code

```
<?php
```

```
// @exclude https://www.example.com/secrets/*
```

@grant Lets you include any GM API methods within your script.

Possible methods to include: [GM_info](#), [GM_deleteValue](#), [GM_getValue](#), [GM_listValues](#), [GM_setValue](#), [GM_getResourceText](#), [GM_getResourceURL](#), [GM_addStyle](#), [GM_log](#), [GM_openInTab](#), [GM_registerMenuCommand](#), [GM_setClipboard](#), [GM_xmlHttpRequest](#), [unsafeWindow](#). I go into greater detail about some of the more commonly used ones later in this tutorial and the rest you can find more info by clicking on their links.

PHP Code

```
<?php
```

```
// @grant GM_log
```

@icon Define an icon to be displayed in the script management interface. Any image will work, but 32x32px will be displayed best.

PHP Code

```
<?php
```

```
// @icon http://www.example.org/icon.png
```

@include Define what webpages you want your script to be activated on.

You may find a detailed list of rules and tricks for **@include** here: [Include and exclude rules - GreaseSpot Wiki](#)

PHP Code

```
<?php
```

```
// @include http://www.example.com/*
```

@match A safer version of **@include** which sets more strict rules for the * character.

You may find a detailed list of rules and tricks for **@match** here: [Match Patterns - Google Chrome](#).

PHP Code

```
<?php
```

```
// @match http://*.example.com/*
```

@name The name of the script.

PHP Code

```
<?php
```

```
// @name My First Script
```

@namespace **@namespace**, along with **@name**, create a unique identifier for the script. This will prevent different scripts with the same name from overwriting each other. Usually the namespace is often a URL, however it can be any unique value.

PHP Code

```
<?php
```

```
// @namespace http://www.example.com/
```

@require This will import and download other userscript libraries.

PHP Code

```
<?php
```

```
// @require http://www.example.com/example.js
```

@resource The named resources may be accessed through [GM_getResourceText](#) and [GM_getResourceURL](#).

PHP Code

```
<?php
```

```
// @resource resourceName http://www.example.com/example
```

@run-atYou may force the script to run either before the page is loaded or after with either document-end or document-start. Default behavior is to run the script after the document has been loaded.

PHP Code

```
<?php
```

```
// @run-at document-start
```

@unwrapLet your script run outside of the sandbox. Only use this for debugging purposes.

PHP Code

```
<?php
```

```
// @unwrap
```

@updateURLDefine where the userscript manager will check for updates from. The userscript manager will parse the metadata and determine if there is a newer version.

PHP Code

```
<?php
```

```
// @updateURL http://www.example.org/example.meta.js
```

@versionDefine the version number of your program. This will be used when determining if there is an update.

PHP Code

```
<?php
```

```
// @version 1.0
```

Example

PHP Code

```
<?php
```

```
// ==UserScript==
```

```
// @name HF Enhancement Suite
```

```
// @author Emylbus
```

```
// @namespace http://www.sublyme.net
```

```
// @description This userscript defines general enhancement tweaks that apply to the entire HackForums site.
```

```
// @include *hackforums.net/*
```

```
// @version 0.32
```

```
// @updateURL http://sublyme.net/site_media/userscripts/HFES.user.js
```

```
// @grant GM_addStyle
```

```
// @grant GM_xmlhttpRequest
```

```
// @grant GM_getValue
```

```
// @grant GM_setValue
```

```
// @grant GM_log
```

```
// @require http://github.com/sizzlemctwizzle/GM_config/raw/master/gm_config.js
```

```
// @require http://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js
```

```
// @require http://sublyme.net/site_media/userscripts/GM_SuperValues.user.js
```

```
// ==/UserScript==
```

Emylbus - July 21, 2013, 12:07 pm

[Top](#) Console Logging

Logging information about what is happening in your script is incredibly important, especially when debugging your program. You can print to your javascript console with GM_log():

PHP Code

```
<?php
GM_log(astring);
```

Add CSS Styles

GM_addStyle() lets you insert new CSS into the document by creating a new <style> element in the <head> of the page. Here is an example of it being used:

PHP Code

```
<?php
// Joining a list like this is a convenient way to make long strings:
var emyNotice = [
    ".emy_alert {",
    "background: #33;",
    "border-top: 1px solid #F4D639;",
    "border-bottom: 1px solid #F4D639;",
    "text-align: center;",
    "padding: 5px 20px;",
    "font-size: 11px;",
    "margin-bottom: 15px;}",
].join("");

GM_addStyle(emyNotice);
```

Emylbus - July 21, 2013, 12:09 pm

[Top](#) Local Data Storage

The GreaseMonkey API provides convenient methods to store data locally on your computer.

Store Values

You can store data with a name->value mapping. The GM_setValue() method provides this functionality:

PHP Code

```
<?php
GM_setValue(name, value);
```

Note that the name must be a String while the value may be either a String, Integer, or Boolean. GM_setValue() does not return anything.

Load Values

You can retrieve the data if you know the stored name with GM_getValue(). The syntax for getValue() is as follows:

PHP Code

```
<?php
GM_getValue(name, default);
```

The default parameter is essentially the backup in case the name hasn't been stored yet, helping you prevent issues of trying to retrieve things that haven't been set. Note that the name must be a String and the default may be either a String, Integer, or Boolean. GM_getValue() returns the value associated with the name we set with GM_setValue().

List Values

You may retrieve the complete list of stored names with `GM_listValues()`. Syntax is as follows:

PHP Code

```
<?php
```

```
GM_listValues();
```

This function will return an array of strings containing all stored names.

Delete Values

You may delete a saved name->value pair completely with `GM_deleteValue()`:

PHP Code

```
<?php
```

```
GM_deleteValue(name);
```

Example

Here's a brief example of these features working together:

PHP Code

```
<?php
```

```
GM_setValue("owner", "Omniscient"); // Store "owner"->"Omniscient"
```

```
GM_setValue("AwesomePerson", "Emylbus"); // Store "AwesomePerson"->"Emylbus"
```

```
var aval = GM_getValue("owner", "unknown");
```

```
// Store either the value of "owner" or the default string "unknown"
```

```
GM_log("The owner is: "+aval); // Print the value stored in aval
```

```
GM_log("Here are all of the stored names: "+GM_listValues());
```

```
// This should print any stored names: ["owner", "AwesomePerson"]
```

```
GM_deleteValue("owner"); // This deletes the "owner" name->value pair.
```

```
GM_log("What names are left: "+GM_listValues()); // This should only print ["AwesomePerson"].
```

GM_SuperValue

The biggest issue with the previously stated methods of storing information is that you can only store Strings, Integers, and Booleans. What if you wanted to store an array or a JSON object? Fortunately a 3rd-party library called [GM_SuperValue](http://userscripts.org/scripts/source/107941.user.js) extends the storage capability to allow you to store any type of data.

You can use this by importing the library with the `@require` metadata:

PHP Code

```
<?php
```

```
// @require http://userscripts.org/scripts/source/107941.user.js
```

I am hosting a mirror which can be accessed with this:

PHP Code

```
<?php
```

```
// @require http://www.sublyme.net/site_media/userscripts/GM_SuperValues.user.js
```

The usage is identical to `GM_setValue()` and `GM_getValue`, except with a different name:

PHP Code

```
<?php
```

```
GM_SuperValue.set(name, value);
```

```
var x = GM_SuperValue.get(name, default);
```

[Top](#) Sending Webpage Requests

To connect to another webpage we'll be using one of the GreaseMonkey APIs: `GM_xmlHttpRequest()`. This gmethod is asynchronous which means that it does not pause the execution of the rest of your program while it is working. Once the the script recieves the webpage, it will execute the function defined in the `onload` argument.

Usage

PHP Code

```
<?php
GM_xmlHttpRequest({
  method: "GET",
  url: "example.com",
  onload: function(response){
    // Put your code here that will be ran after the page is loaded.
    GM_log(response.responseText); // Print the returned HTML
  },
});
```

Arguments

`binaryBoolean` parameter and Optional

If set to true, your request to the server is sent in binary.

`contextObject` parameter and Optional

`dataString` parameter and Optional

Sent in the body of the request. Used for POST requests.

`headersObject` parameter and Optional

Define the headers you wish to send to the server, this may include any header, such as User-Agent (if not specified, the useragent will be whatever your browser uses).

`methodString` parameter and Required

Define what request method you want to make (i.e. "GET" or "POST")

`onabortFunction` parameter and Optional

The function defined will be called if the request is aborted. The function will be passed the Response Object as an argument.

`onerrorFunction` parameter and Optional

The function defined will be called if an error occurs while processing the request. The function will be passed the Response Object as an argument.

`onloadFunction` parameter and Optional

The function defined will be called once the request has been successfully completed. The function will be passed the Response Object as an argument.

`onprogressFunction` parameter and Optional

The function defined will be called when the request progress changes. The function will be passed the Response Object as an argument.

`onreadystatechangeFunction` parameter and Optional

The function defined will be called repeatedly while the request is in progress. The function will be passed the Response Object as an argument.

`ontimeoutFunction` parameter and Optional

The function defined will be called if the request timesout. The function will be passed the Response Object as an argument.

`overrideMimeTypeString` parameter and Optional

Lets you specify the MIME type with the request (i.e. "text/html; charset=ISO-8859-1")

`passwordString` parameter and Optional

Send a password used for authentication purposes.

synchronousBoolean parameter and Optional

If true, the request becomes synchronous which means your browser will freeze until the request has been completed.

timeoutNumber parameter and Optional

Defines the number of milliseconds to wait before terminating the request. Default (or 0) has no timeout.

uploadObject parameter and Optional

Pass an object with optional callback functions to monitor the upload of data. Each is passed the Response Object.

urlString parameter and Required

Defines the URL you send the request to. The URL may be relative or absolute.

userString parameter and Optional

Used for authentication purposes.

Response Object

All callback functions (onabort, onerror, onload, onprogress, onreadystatechange, and ontimeout) are passed the Response Object which contains information recieved from the server. Depending on when the function is called, the response object will have more data. The following is the different information this object has:

context

finalUrl

readyState

responseHeaders

responseText

status

statusText

Progress callbacks also have:

lengthComputable

loaded

total

Examples

These are examples provided from the Grease Monkey API documentation:

Bare Minimum

PHP Code

```
<?php
GM_xmlHttpRequest({
  method: "GET",
  url: "http://www.example.com/",
  onload: function(response) {
    alert(response.responseText);
  }
});
```

GET request

PHP Code

```
<?php
GM_xmlHttpRequest({
  method: "GET",
  url: "http://www.example.net/",
  headers: {
```

```

    "User-Agent": "Mozilla/5.0", // If not specified, navigator.userAgent will be used.
    "Accept": "text/xml" // If not specified, browser defaults will be used.
},
onload: function(response) {
    var responseXML = null;
    // Inject responseXML into existing Object (only appropriate for XML content).
    if (!response.responseXML) {
        responseXML = new DOMParser()
        .parseFromString(response.responseText, "text/xml");
    }

    GM_log([
        response.status,
        response.statusText,
        response.readyState,
        response.responseHeaders,
        response.responseText,
        response.finalUrl,
        responseXML
    ].join("\n"));
}
});

```

POST request

PHP Code

```
<?php
```

```

GM_xmlHttpRequest({
    method: "POST",
    url: "http://www.example.net/login",
    data: "username=johndoe&password=xyz123",
    headers: {
        "Content-Type": "application/x-www-form-urlencoded"
    },
    onload: function(response) {
        if (response.responseText.indexOf("Logged in as") > -1) {
            location.href = "http://www.example.net/dashboard";
        }
    }
});

```

HEAD request

PHP Code

```
<?php
```

```

GM_xmlHttpRequest({
    url: "http://www.example.com",
    method: "HEAD",
    onload: function(response) {
        GM_log(response.responseHeaders);
    }
});

```

[Top](#) Easy Examples

Example 1 - Replace HF logo

The first thing we'll do is something pretty fun and quick to make: replacing the HF logo with any image of your choice. First things first, we need to figure out exactly where in the code the image is. Open up your developer tools and click on the logo to take us to that part of the HTML:

As always we want to be as specific as possible when defining the jQuery filter, and the first thing that jumps out to me is the "logo" class. So that is what we should set our initial jQuery filter to be:

PHP Code

```
<?php  
$(".logo");
```

However that's not specific enough for us! If we can also throw another term into that filter to narrow down exactly what we're looking for! Remember that class names are preceded by a period whereas tag names are not preceded by anything:

PHP Code

```
<?php  
$(".logo img");
```

This will limit us to just that the bit of code we're looking for. Now that we have the code we need, we'll need to change the image source. We can do this with the attr() jQuery method:

PHP Code

```
<?php  
$(".logo img").attr("src","your-pic.jpg");
```

Let's try this out! Don't forget to import the jQuery code with @require!

Let's see what it looks like!

Alright, well the picture doesn't fit very well, but you can always choose your own to use. Lastly let's make sure this code doesn't run on hackforums.net because there is no logo. We can do by either changing the @include parameters in the beginning of the script to:

PHP Code

```
<?php  
// @include hackforums.net/*  
// @include hackforums.net/*
```

which will only let the script run on those two sites, or we can check the URL in script with a simple conditional which checks whether or not "nsfw." exists in the page's URL:

PHP Code

```
<?php  
if(document.URL.indexOf('nsfw.') != -1){  
    // 'nsfw.' is not in the URL  
    $(".logo img").attr("src","http://www.mainecoonsofminnesota.com/images/cat_banner.jpg");  
}
```

Example 2 - Add Custom Links

Next let's get some custom links to our favorite pages. In this case, let's add links to the right of where the link to our Private Messages is. As always we begin by looking at the source code to see what's already being done:

As we can see each link is simply listed and if we wanted to add more we could just add anything we want to the end of that <div>. Note that the <div> in question has a convenient id "panel". Since you're only supposed to have one thing associated with an id, we can specify that in our jQuery filter without having to specify more like we did in the previous example:

PHP Code

```
<?php  
$("#panel");
```

We can append pretty much anything we want to the end of this <div> with the jQuery .append() method:

PHP Code

```
<?php  
$("#panel").append("<b>Hello World</b>");
```

If we run this code in a userscript, we'll get this result:

Excellent! Now all we need to do is define our links and append those instead and we'll be in great shape. We'll store the links in variables this time:

PHP Code

```
<?php  
var myrep, lounge;  
myrep = "<a href='https://hackforums.net/reputation.php?uid=[Your UID]'>My Rep</a>";  
lounge = "<a href='https://hackforums.net/forumdisplay.php?fid=25'>Lounge</a>";  
Just append those variables (delimiting them with "|" for consistency's sake), the final code will look like this:
```

which will look like this in the end:

Example 3 - Replace All Photos

All the way back in example 1, we found out how easy it was to replace the logo with a picture of a cat, let's change EVERY picture >:D

We start off trying to think of what we're trying to do: change every image source to our own and that requires us to first filter for all image tags.

PHP Code

```
<?php  
$("img");  
We can see how many elements were tagged by using the GM_log() method we learned about earlier:
```

PHP Code

```
<?php  
GM_log($("#img"));  
Sheesh, we can see that there are a TON of images!
```

We can use the attr() method again to change the src attribute of all image tags at once!

PHP Code

```
<?php  
$("#img").attr("src","my-pic.jpg");  
Here's what my final code looks like:
```

And my final result is (drum roll please):

Beautiful.

Emylbus - July 21, 2013, 12:12 pm

[Top](#) [More Resources](#)

[Official JavaScript Documentation](#) Don't let this scare you off, the JS docs are one of the best ways to quickly get information about anything with the language.

[jQuery Documentation](#) An incredibly easy to use and understand resource for all jQuery features.

[Greasemonkey API manual](#) A full list of API tools built into your userscript manager. Take advantage of them, they are incredibly useful.

[The Ultimate Compilation of Hack Forum Userscripts](#) by [xadamxk](#) The most comprehensive list of Hack Forums specific userscripts.

Find some scripts you like from here and tinker with them. Learning through practicing and exploring is one of the best ways to learn how to program.

Emylbus - July 24, 2013, 6:11 pm

Sheesh that was a ton of information to throw you at once. Remember this isn't a time for resting your laurels, your adventure is just beginning! I hope to see great things from you all, good luck!

'Snorlax - July 25, 2013, 7:09 am

Incredibly detailed guide, Emylbus! Good job making this for this community, because a lot of people have been wondering how to make these userscripts!

Very detailed, very nice layout and a good read! Thanks for making and sharing this!

vitafortis - July 25, 2013, 7:16 am

Great thread mate, looks like you put a lot of work into it. 5 Star from me.

Sleazy P. Martini - July 25, 2013, 7:39 am

Looks good, Will read it all when i'm not busy.

Good job.

JustCraig - July 25, 2013, 7:42 am

Thanks for this man! Awesome tutorial!

+Awesome - July 25, 2013, 8:08 am

Very detailed and superb stuff! Thanks for posting. I saw this building! ;)

Trapcode - July 25, 2013, 8:20 am

Very good and superb job bro ;) Although i did not understand coding ?

The Doctor - July 25, 2013, 8:30 am

Why is this in the lounge and not in a coding section?

Emylbus - July 25, 2013, 8:33 am

(Jul 25, 2013 08:30 AM)?lixir Wrote: [\[CITE\]](#) Why is this in the lounge and not in a coding section? Because the staff member I asked thought this would be a good place. This is where userscripts are released, so it's fitting this is also where a tutorial about userscripting goes.

Though I agree, the lounge isn't a great place for it. I would encourage everyone to support the petition for a separate userscript/browser extension forum: [Userscript Sub-Section](#) by [xadamxk](#)

(Jul 25, 2013 08:20 AM)webismylife Wrote: [\[CITE\]](#) Very good and superb job bro ;) Although i did not understand coding ?

I might offer very cheap programming lessons in the near future if there is enough interest.

james2107 - July 25, 2013, 8:42 am

(Jul 25, 2013 08:33 AM)Emylbus Wrote: [\[CITE\]](#) Because the staff member I asked thought this would be a good place. This is where userscripts are released, so it's fitting this is also where a tutorial about userscripting goes.

Though I agree, the lounge isn't a great place for it. I would encourage everyone to support the petition for a separate userscript/browser extension forum: [Userscript Sub-Section](#) by [xadamxk](#)

I might offer very cheap programming lessons in the near future if there is enough interest. So this is what you talked about! Huge tutorial! I know what i'm going to do for the night.

PandaVenom - July 25, 2013, 8:43 am

Dude you could defiantly off programming lessons to people who don't know how but want to learn.

Helios. - July 25, 2013, 8:55 am

wow, great tutorial and very detailed!

Amazing effort and work!

redi - July 25, 2013, 9:43 am

Omg Emyl, Who doesn't know this? Like wtf no need to teach people how to make a userscript, it's sooooo easy.

Thank you for the tutorial man. :)

kalandershihabrocker - July 25, 2013, 9:55 am

Awesome tutorial mate! I'll bookmark this for further reference.

Breakbeat Junkie - July 25, 2013, 11:04 am

Very nice.

Bookmarked for future scripts :D

PinkPanther - July 25, 2013, 12:36 pm

Wow, thats a pretty impressive tutorial. Very thorough and full of quality.

EliteCrew - July 25, 2013, 12:42 pm

I too shall be bookmarking this. Very professional lay out and detail. Nice work.

chrisman0091 - July 25, 2013, 3:04 pm

Amazing tutorial, very HQ as always Emylbus

Emylbus - July 25, 2013, 4:12 pm

I'm glad you guys are enjoying this ^_^ If you have any questions feel free to ask!

Dyck - July 27, 2013, 2:38 pm

Emylbus, damn you are amazing O_O...

Watermelman - July 27, 2013, 2:40 pm

This is fucking amazing OP, if i could rep i would rep you. I see you put alot of time and effort into this. And i made a Logo change now C:

?XilluX - July 27, 2013, 2:43 pm

Amazing Thread, in my eyes, it deserves a sticky!

'Snorlax - July 27, 2013, 3:04 pm

(Jul 27, 2013 02:43 PM)XilluX Wrote: [\[CITE\]](#) Amazing Thread, in my eyes, it deserves a sticky!
If we only had an userscript section ?

Emylbus - July 28, 2013, 12:36 pm

(Jul 27, 2013 03:04 PM)Snorlax Wrote: [\[CITE\]](#) If we only had an userscript section ?
That'd be wayy too helpful and useful for Hackforums to have :P

Emylbus - July 31, 2013, 12:13 am

(Jul 28, 2013 06:14 PM)xadamxk Wrote: [\[CITE\]](#) One of these days I'm going to read this hahaha.

Damn good tutorial mate!

I appreciate it! I wouldn't have gotten into userscripts if you hadn't commissioned HFES in the first place. Maybe we'll see you make one in the near future!

Emylbus - August 1, 2013, 9:02 pm

(Aug 1, 2013 08:44 PM)xadamxk Wrote: [\[CITE\]](#) Where would you suggest learning Javascript?

Any good eBooks, or online tutorials?

Honestly I would suggest learning a language like Python first, once you understand the fundamentals of one language, it's very easy to jump to another. If you wish to go head first into javascript (not a very complicated language), I'm not entirely sure what tutorial out there would be best, but glancing at some ebooks this one caught my eye:

<http://eloquentjavascript.net/contents.html>

Of course you can always PM me any questions you have.

SIM-73089 - August 11, 2013, 5:51 am

Thanks for this.

I became interested in this topic after using the HFES and realizing that I should not have been able to exist without multi-quotes. :P

Ombra - August 30, 2013, 2:24 pm

Let's start :) thank you @Emylbus for giving me the starting direction

sizzlemctwizzle - September 2, 2013, 2:30 pm

(Jul 21, 2013 12:07 PM)Emylbus Wrote: [\[CITE\]](#) PHP Code

```
<?php
// ==UserScript==
// @name HF Enhancement Suite
// @author Emylbus
// @namespace http://www.sublyme.net
// @description This userscript defines general enhancement tweaks that apply to the entire HackForums site.
// @include *hackforums.net/*
// @version 0.32
// @updateURL http://sublyme.net/site_media/userscripts/HFES.user.js
// @grant GM_addStyle
// @grant GM_xmlHttpRequest
// @grant GM_getValue
// @grant GM_setValue
// @grant GM_log
// @require http://github.com/sizzlemctwizzle/GM_config/raw/master/gm_config.js
// @require http://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js
// @require http://sublyme.net/site_media/userscripts/GM_SuperValues.user.js
// ==/UserScript==
```

If you're going to link to my library (GM_config) and not mention it, the least you could do is link to [the wiki](#).

Emylbus - September 2, 2013, 2:32 pm

(Sep 2, 2013 02:30 PM)sizzlemctwizzle Wrote: [\[CITE\]](#) If you're going to link to my library (GM_config) and not mention it, the least you could do is link to [the wiki](#).

I didn't mention it in this thread because it doesn't discuss this plugin. I'll link to the wiki in the release thread of my script.

sizzlemctwizzle - September 2, 2013, 6:02 pm

(Sep 2, 2013 02:32 PM)Emylbus Wrote: [\[CITE\]](#) I didn't mention it in this thread because it doesn't discuss this plugin. I'll link to the wiki in the release thread of my script.

I figured since you mentioned the other two libraries used in that script in this thread, you might as well mention mine (I think it's something that many user script developers would find useful). You don't need to link the wiki in the release thread since that thread is about your script and not user script development.

Also,

(Jul 21, 2013 12:07 PM)Emylbus Wrote: [\[CITE\]](#) @updateURLDefine where the userscript manager will download updates from.

PHP Code

```
<?php
```

```
// @updateURL http://www.example.org/example.meta.js
```

That isn't exactly correct. @updateURL defines where to check for updates (i.e parse the metadata and compare the version number with the installed script). If an update is found it is downloaded from the location specified by @downloadURL.

Source: I implemented user script updating in Greasemonkey.

Emylbus - September 2, 2013, 6:42 pm

(Sep 2, 2013 06:02 PM)sizzlemctwizzle Wrote: [\[CITE\]](#) I figured since you mentioned the other two libraries used in that script in this thread, you might as well mention mine (I think it's something that many user script developers would find useful). You don't need to link the wiki in the release thread since that thread is about your script and not user script development.btw, the release thread for my userscript I was referencing is here: <https://hackforums.net/showthread.php?tid=3603256>

I linked to the wiki there to give credit to you for the code. Usually I'm so good at that, but it must have slipped my mind. There are a ton of libraries people might find useful and I only had a limited space so I needed to try to figure out which ones are most useful. Also the documentation for GM_Config is pretty abysmal. If there's a decent place that explains every possible option for it, please let me know. In addition to this, there are two versions (yours and JoeSimmons) which leads to confusion.

(Sep 2, 2013 06:02 PM)sizzlemctwizzle Wrote: [\[CITE\]](#) Also,

That isn't exactly correct. @updateURL defines where to check for updates (i.e parse the metadata and compare the version number with the installed script). If an update is found it is downloaded from the location specified by @downloadURL.

Source: I implemented user script updating in Greasemonkey.

Thank you for the clarification, the official docs aren't the greatest in some places.

sizzlemctwizzle - September 2, 2013, 7:11 pm

(Sep 2, 2013 06:42 PM)Emylbus Wrote: [\[CITE\]](#) Also the documentation for GM_Config is pretty abysmal. If there's a decent place that explains every possible option for it, please let me know.

All the documentation you could ever want is in [the wiki](#). Here's [a link](#) to the full list of pages. The

[fields page](#) is probably what you're really looking for.

(Sep 2, 2013 06:42 PM)Emlbus Wrote: [\[CITE\]](#) In addition to this, there are two versions (yours and JoeSimmons) which leads to confusion.

https://github.com/sizzlemctwizzle/GM_config/wiki/FAQ#isnt-there-another-version-of-gm_config-on-userscriptsorg

Emlbus - September 2, 2013, 7:18 pm

(Sep 2, 2013 07:11 PM)sizzlemctwizzle Wrote: [\[CITE\]](#) All the documentation you could ever want is in [the wiki](#). Here's [a link](#) to the full list of pages. The [fields page](#) is probably what you're really looking for.

https://github.com/sizzlemctwizzle/GM_config/wiki/FAQ#isnt-there-another-version-of-gm_config-on-userscriptsorg

Well you just have the answers for everything, don't you! Thanks, I had been looking for a list of fields for a long time now but got jumbled up with the multiple different versions.

sizzlemctwizzle - September 2, 2013, 7:34 pm

(Sep 2, 2013 07:18 PM)Emlbus Wrote: [\[CITE\]](#) Thanks, I had been looking for a list of fields for a long time now

Truth be told, the detailed documentation is pretty new. GM_config has been around since '09 and until recently the only way of knowing what was possible was looking at either GM_config's source or existing scripts using GM_config. Hopefully with the new documentation more people will decide to use it, rather than trying to build their own settings panel from scratch.

Emlbus - September 2, 2013, 8:27 pm

(Sep 2, 2013 07:34 PM)sizzlemctwizzle Wrote: [\[CITE\]](#) Truth be told, the detailed documentation is pretty new. GM_config has been around since '09 and until recently the only way of knowing what was possible was looking at either GM_config's source or existing scripts using GM_config. Hopefully with the new documentation more people will decide to use it, rather than trying to build their own settings panel from scratch.

Keep up the excellent work. I hope to see you stay around HF! I'll be sure to PM you if I have any questions.

sizzlemctwizzle - September 3, 2013, 11:10 am

Since you're already using GM_config, you can also implement GM_SuperValue like this:

PHP Code

```
<?php
var GM_SuperValue = new function () {
  this.set = function(name, val) {
    GM_config.setValue(name, GM_config.stringify({'val': val}));
  };

  this.get = function(name, def) {
    var rval = GM_config.getValue(name, def);
    return rval === def ? rval : GM_config.parser(rval).val;
  };
};
```

It has the advantage of working in user script environments that don't have GM_setValue/GM_getValue. GM_config uses localStorage as a fallback and will run in pretty much

any modern browser (in fact, here is a [web page](#) that it runs on without any modification). The only downside of using the above code is that you can't store functions, but that only works in Firefox (last time I checked) and I don't know why the hell you'd want to do that anyway.

Cletus Kasady - September 19, 2013, 2:48 pm

Amazing very high quality guide here, will be using it tomorrow with my first attempt at a user script. Thanks for the tutorial!

Ombra - December 11, 2013, 11:24 am

I'm working to understand bases, luckily in this section there's not too activity and a lot of help is provided :)

Vexane - April 21, 2014, 8:50 pm

Not sure if I've posted on this or nah, but I just want to get this back to the top of threads because it is truly an amazing resource.

I'll be using this thread VERY soon. I might even turn it into video format :)

Aura - November 10, 2014, 4:27 am

I honestly don't understand why that thread isn't stickied yet ?

Probably one of the most detailed, in-depth tutorial I've seen in that section.

Hash G. - June 30, 2015, 6:12 am

I wish I had seen that before. Very good and detailed tutorial!

B?ker - December 21, 2016, 2:14 pm

I subscribed the hell out of this thread, so I can check up on it later to understand userscript more, so thank you for this man this is awesome! <3

ven0m - February 2, 2017, 11:47 pm

(Feb 3, 2017 12:38 AM)xadamxk Wrote: [\[CITE\]](#) Would anyone be interested if I made a more beginner friendly guide?

I would. Also, make Beginner to Advance scripting tutorials.

ven0m - June 2, 2019, 5:49 am

(May 24, 2019 12:28 PM)xadamxk Wrote: [\[CITE\]](#) Still down for that?

I'd like to have a course on it but it would require some basic understanding of web dev.

I would definitely read the guides but I don't have time to enroll in a course at this moment.