


[Advertise Here](#)

SLOW WEBSITES COST VISITORS. DOES YOURS?
FIND OUT WHAT'S SLOWING YOUR SITE DOWN
WITH THIS FREE WORDPRESS SPEED TEST!

WP Engine

TEST NOW!



Load jQuery From Google CDN With Local Fallback For WordPress

Jason Witt on Nov 10th 2011 with [35 comments](#)

Tutorial Details

-
- **Program:** WordPress
-
- **Version (if applicable):** 3.0+
-
- **Difficulty:** Beginner
-
- **Estimated Completion Time:** 20 mins

In this tutorial we are going to cover why loading jQuery from the Google CDN with a local file fallback is important, and how to set this up for WordPress. If you're still trying to self-host your own copy of jQuery (or lots of other script libraries for that matter!), this is a great way to quickly improve your workflow.

Introduction

When I first took a look at [HTML5 Boilerplate](#) there was this great code snippet for loading jQuery from the Google CDN with a **local file fallback**.

[view plaincopy to clipboardprint?](#)

1. `<script src="//ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.js"></script>`

2. `<script>window.jQuery || document.write("<script src='js/libs/jquery-1.6.4.min.js'\>x3C/script>")`
`</script>`

This was the greatest thing I'd seen on the topic of loading scripts for a while... Unfortunately, this doesn't play well with WordPress, because we should be loading our JavaScripts using `wp_enqueue_script`, which we recently looked at in [this article on using enqueue](#) to properly load scripts inside WordPress.

So, here's a way to accomplish the same thing in WordPress.

Why Use a Local Fallback File?

Why would I use a local fallback file at all?

The **first** reason for using a local file fallback is for your local development environment. Internet access isn't always available when you're working on your project. Let's say you work for a web development firm and the company's internet server goes down. If you've been working on a script that is jQuery dependent, and you've been calling jQuery from the Google CDN. Well, you're out of luck. Even if you have jQuery cached, eventually you'll have to clear the cache to make sure the site is rendering properly, or switch to another browser to check cross compatibility.

At that point, you might as well take a long lunch. Which I'm sure your manager won't be too pleased with. If you're getting paid by the hour, no internet access isn't an excuse for not working.

The same could be said if you were at home and the internet went down... or if you traveled to the latest WordCamp and couldn't get internet access in your hotel. There's nothing more annoying than being in the zone and having to stop just because you can't load jQuery.

The **second** reason is what if Google goes down. Besides the usual rioting in the streets, airplanes falling from the sky, and a plague of frogs. Your site will break, and we can't have that. The rest we can deal with. Just find a, out of the way, cave and cook up some frog legs.

Seriously though, it's not likely that Google will ever go down. Google is so big that I'm sure they have backup servers for their backup servers, but you never know. Isn't better to have a backup of your own, just in case?

Step 1 Where to Start

The script is going to be placed in your `functions.php` file.

The first thing we need to do is define some variables

[view plain](#)[copy to clipboard](#)[print](#)?

1. `<?php`
2. `$url = 'http://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js'; // the URL to check against`
3. `$test_url = @fopen($url,'r'); // test parameters`
4. `?>`

`$url` is the URL the script is going to use to test against. In this case, we're testing if the jQuery file on the

Google CDN.

\$test_url is setting up the parameters for the test.

Step 2 Setting up the Test

Now, let's set up the main part of the test

[view plain](#)[copy to clipboard](#)[print?](#)

```
1. <?php
2. if($test_url !== false) { // test if the URL exists
3.     //load the external script
4. } else {
5.     //load the local script
6. }?>
```

if(\$test_url !== false) is testing to see if it can open the URL we defined in the variable \$url. If it can it will load the external script on the CDN. Else, if can't open the URL it loads the local script on your server.

Step 3 Loading the External Script Function

Here we add the function to load the external CDN file

[view plain](#)[copy to clipboard](#)[print?](#)

```
1. if($test_url !== false) { // test if the URL exists
2.     function load_external_jQuery() { // load external file
3.         wp_deregister_script( 'jquery' ); // deregisters the default WordPress jQuery
4.         wp_register_script('jquery', 'http://ajax.googleapis.com/ajax/libs/jquery/1.6.4
5.         /jquery.min.js'); // register the external file
6.         wp_enqueue_script('jquery'); // enqueue the external file
7.     }
8.     add_action('wp_enqueue_scripts', 'load_external_jQuery'); // initiate the function
9. }
```

- wp_deregister_script deregisters the default jQuery that comes with WordPress. It's usually an older version, so I like to use the the most recent version.
 - wp_register_script registers the external CDN jQuery with WordPress.
 - wp_enqueue_script will enqueue the newly register version.
 - add_action will initiate the function.
-

Step 4 Loading the Local Script Function

An else statement is used to load the local file is the test isn't able to open the external CDN file

[view plaincopy to clipboardprint?](#)

```
1. else {
2.     function load_local_jQuery() {
3.         wp_deregister_script('jquery'); // deregisters the default WordPress jQuery
4.         wp_register_script('jquery', bloginfo('template_url').'/js/libs/jquery-
    1.6.1.min.js', __FILE__, false, '1.6.4', true); // register the local file
5.         wp_enqueue_script('jquery'); // enqueue the local file
6.     }
7.     add_action('wp_enqueue_scripts', 'load_local_jQuery'); // initiate the function
8. }
```

This works the exact same way as the external file function. It deregisters the default jQuery, registers the local file, enqueues the local file, and initiates the function.

The Whole Script

Here's the completed script

[view plaincopy to clipboardprint?](#)

```
1. <?php
2. $url = 'http://ajax.googleapis.com/ajax/libs/jquery/1.6.4/jquery.min.js'; // the URL to check against
3. $test_url = @fopen($url,'r'); // test parameters
4. if($test_url !== false) { // test if the URL exists
5.     function load_external_jQuery() { // load external file
6.         wp_deregister_script( 'jquery' ); // deregisters the default WordPress jQuery
7.         wp_register_script('jquery', 'http://ajax.googleapis.com/ajax/libs/jquery/1.6.4
    /jquery.min.js'); // register the external file
8.         wp_enqueue_script('jquery'); // enqueue the external file
9.     }
10.    add_action('wp_enqueue_scripts', 'load_external_jQuery'); // initiate the function
11. } else {
12.    function load_local_jQuery() {
13.        wp_deregister_script('jquery'); // initiate the function
14.        wp_register_script('jquery', bloginfo('template_url').'/js/libs/jquery-
    1.6.1.min.js', __FILE__, false, '1.6.4', true); // register the local file
15.        wp_enqueue_script('jquery'); // enqueue the local file
16.    }
17.    add_action('wp_enqueue_scripts', 'load_local_jQuery'); // initiate the function
18. }
19. ?>
```

This Works with Other Libraries Too!

If jQuery isn't your cup of tea, there's a CDN for that as well. Be sure to check out the entire hosted scripts library that Google keeps track of: <http://code.google.com/apis/libraries/devguide.html>

. You can load everything from MooTools to Prototype in pretty much the same way that we've described above.

jQuery 1.7 Is Out!

~~Since writing this article, jQuery 1.7 has been released onto the Google CDN~~, (that's a lie, we were promised this by November 7th, but we're still in a holding pattern). Consider the [Media Temple CDN](#) location instead if you want the latest and greatest version of jQuery for your WordPress projects: <http://code.jquery.com/>.

Conclusion

You can never be too careful and this script will alleviate of your worries about the jQuery library loading.

This can be expanded to include additional CDN resources. You can just add additional URL variables and else statements for additional CDN's. You can also modified this to do the same test for your own CDN if you use it to load your CSS stylesheets.

Like

8 likes. [Sign Up](#) to see what your friends like.

Tags: [about](#) [wordpress](#) [google](#) [cdn](#) [how to](#) [wordpress](#) [show to](#) [wordpress](#) [website](#) [load](#) [jquery](#) [google](#) [Tutorial](#) [why](#) [wordpress](#) [word](#) [press](#) [wordpress](#) [wordpress](#) [3](#) [wordpress](#) [blog](#) [wordpress](#) [for](#) [free](#) [wordpress](#) [how to](#) [wordpress](#) [jquery](#) [wordpress](#) [of](#) [wordpress](#) [site](#) [wordpress](#) [that](#) [wordpress](#) [themes](#) [wordpress](#) [tutorial](#) [wordpress](#) [website](#) [wordpress](#) [wordpress](#)

By Jason Witt

My name is Jason Witt. I'm a front end Web Designer and Developer specializing in Wordpress and CSS Optimization.