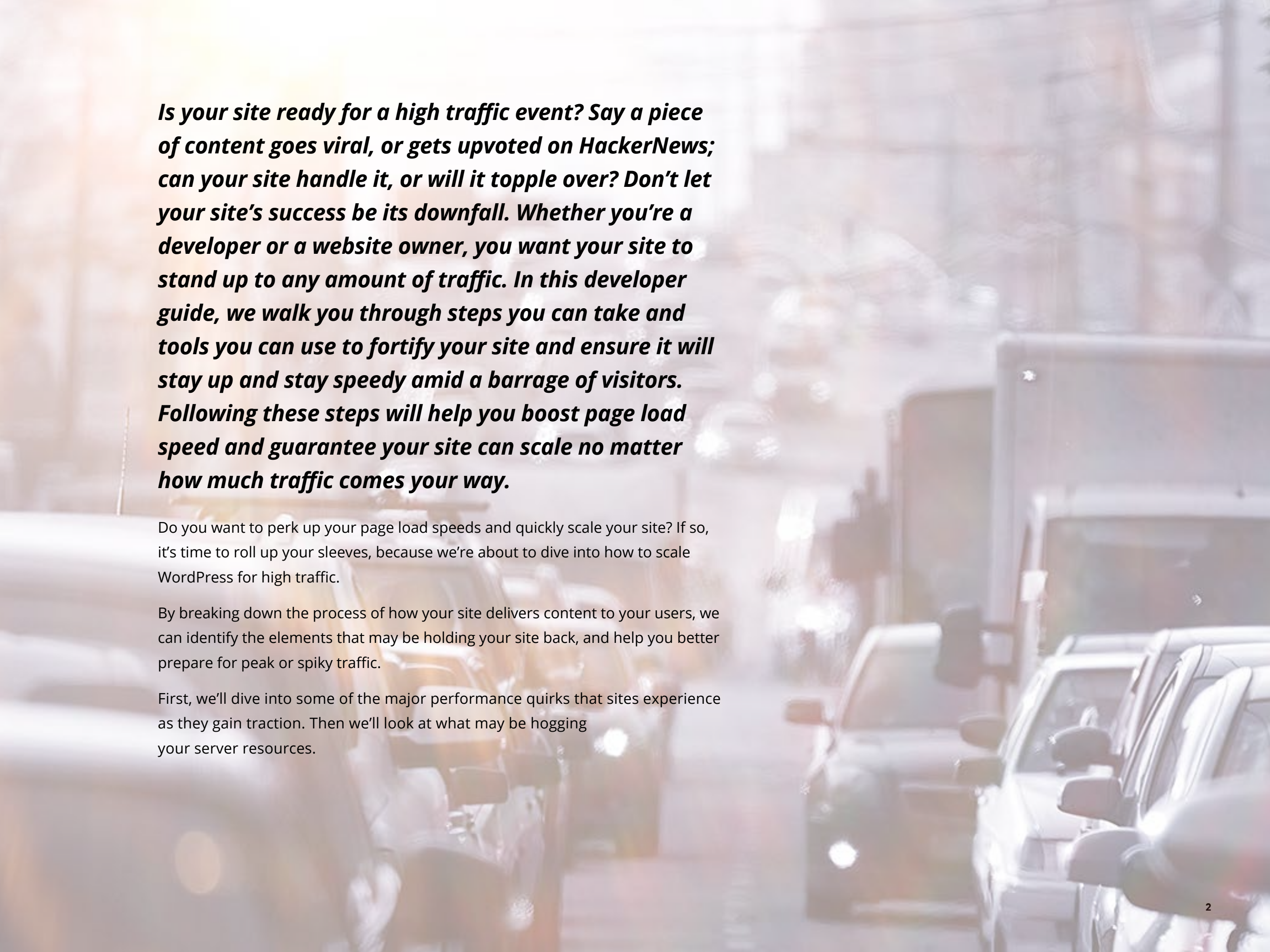


SCALING WORDPRESS FOR HIGH TRAFFIC

By Ryan Oeltjenbruns, Developer

DEVELOPER GUIDE





Is your site ready for a high traffic event? Say a piece of content goes viral, or gets upvoted on HackerNews; can your site handle it, or will it topple over? Don't let your site's success be its downfall. Whether you're a developer or a website owner, you want your site to stand up to any amount of traffic. In this developer guide, we walk you through steps you can take and tools you can use to fortify your site and ensure it will stay up and stay speedy amid a barrage of visitors. Following these steps will help you boost page load speed and guarantee your site can scale no matter how much traffic comes your way.

Do you want to perk up your page load speeds and quickly scale your site? If so, it's time to roll up your sleeves, because we're about to dive into how to scale WordPress for high traffic.

By breaking down the process of how your site delivers content to your users, we can identify the elements that may be holding your site back, and help you better prepare for peak or spiky traffic.

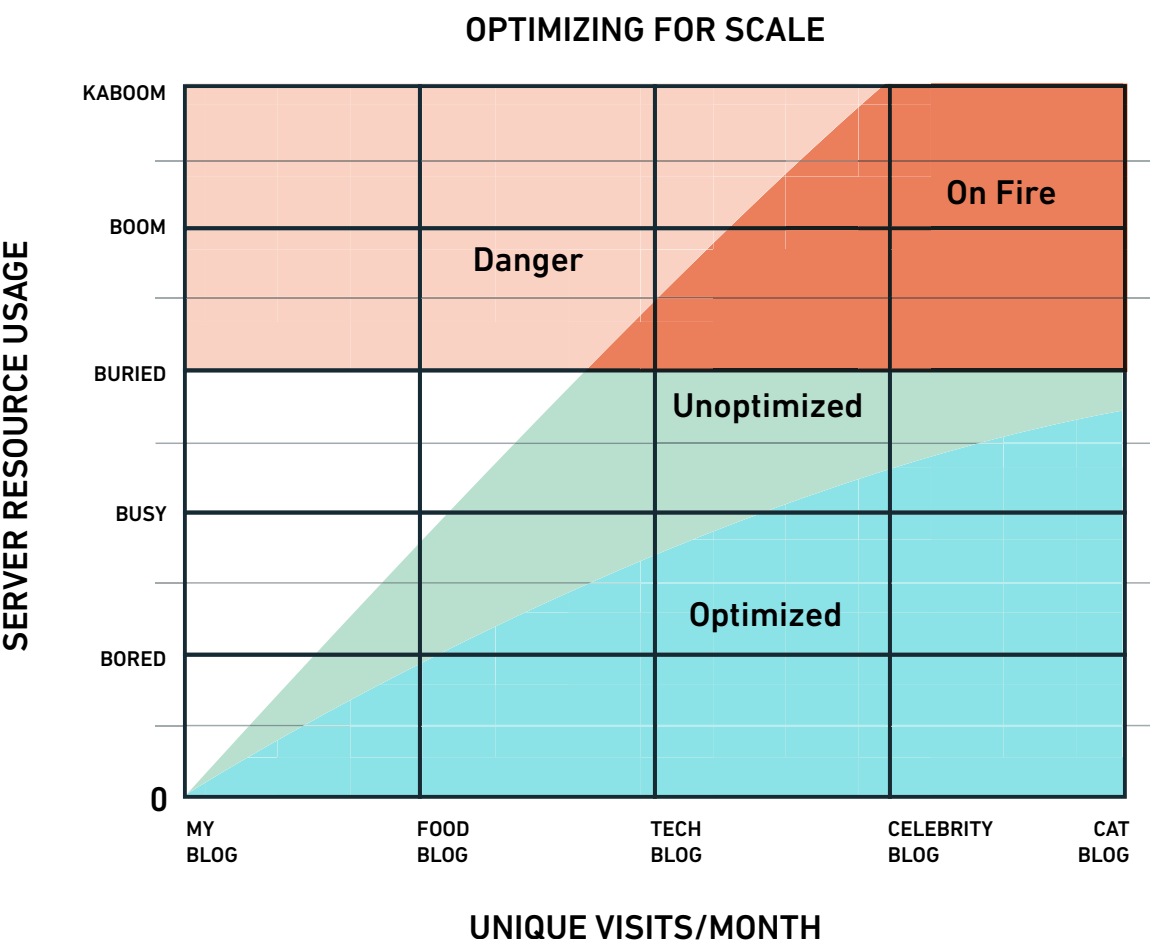
First, we'll dive into some of the major performance quirks that sites experience as they gain traction. Then we'll look at what may be hogging your server resources.

UNDERSTANDING HIGH TRAFFIC

Bottlenecking traffic

Plugins and themes can sometimes perform fine at certain traffic levels, but as visits increase and peak times hit, server resources can be pushed beyond their breaking points.

Think of it like this: there is only so much a server can work through. Eventually the server will have more work to do than it can accomplish, and it will become overloaded.



Load Testing Your Site

Your website performance is a combination of the server environment, your code, and your traffic pattern. If you are preparing for a high traffic event, we strongly recommend you perform a load test to understand how your site will scale with your anticipated traffic profile. WP Engine does not perform custom load tests for our customers. We recommend working with a load testing partner such as *Load Storm* to customize a load test to your specific requirements. Please share the results with WP Engine Support so we can collaborate on any optimization opportunities you identify.

Serving high traffic

*Hightrafficaphobia got you down?
Never! The solution to scale is
simple: less is more.*

Ensure that the request to your site requires as few server resources and as little effort as possible to render and serve. To do this: serve less, serve more efficiently, or both. Let's look at both the backend (e.g. MySQL) and the frontend to see where we can start tuning.

THE BACKEND

MySQL, the Database that Powers WordPress

One of the biggest server resource contenders is MySQL, the database WordPress uses. WordPress lives and breathes by MySQL; without it, your site doesn't know who it is or what it has to say. Total and utter amnesia!

When PHP sends a query to the MySQL server, the PHP process waits to do anything else until the database responds with data. So the longer a MySQL query takes to run, the longer it will take your entire site to load. That's why minimizing each page load's impact on your database is one of the most effective things you can do to get your site loading faster for your users.

TACTICS FOR RESOLVING MYSQL PERFORMANCE ISSUES AND IMPROVING PAGE LOAD SPEED

Defining queries

In WordPress, it is crucial to identify where your slowest queries come from. A great way to know what is going on is by *inserting "Define" into your wp-config.php file* so you can analyze them later, like this:

```
define( 'SAVEQUERIES', true );
```

Debug Objects plugin

To view the query data, use a plugin like *Debug Objects* to quickly identify which queries take the longest and hold your page loads hostage.

I know it hurts, but sometimes removing the big offenders can be the best thing you can do for yourself.

***Like I always say: "A query saved is a query earned."
I always say that. Seriously.***

Know your queries (and get rid of the long ones)

What is a query?
A query is a request to create, read, update, or delete (CRUD) data from the database.

KEEP YOUR OPTIONS TABLE UNDER CONTROL

MySQL stores a lot of data in the *Options table*: plugins, themes, and the WordPress core all use that data.

Overloading the Options table

In the Options table, the option_value is built to be a *LONGTEXT column*. In MySQL, this translates to columns that can store up to 4GB of data in a single row, but just because you can store that much data in one column doesn't mean you should.

Autoloaded queries

Autoloaded queries can be the silent page speed killer. If you're having page load speed issues, try identifying how many queries are being autoloaded.

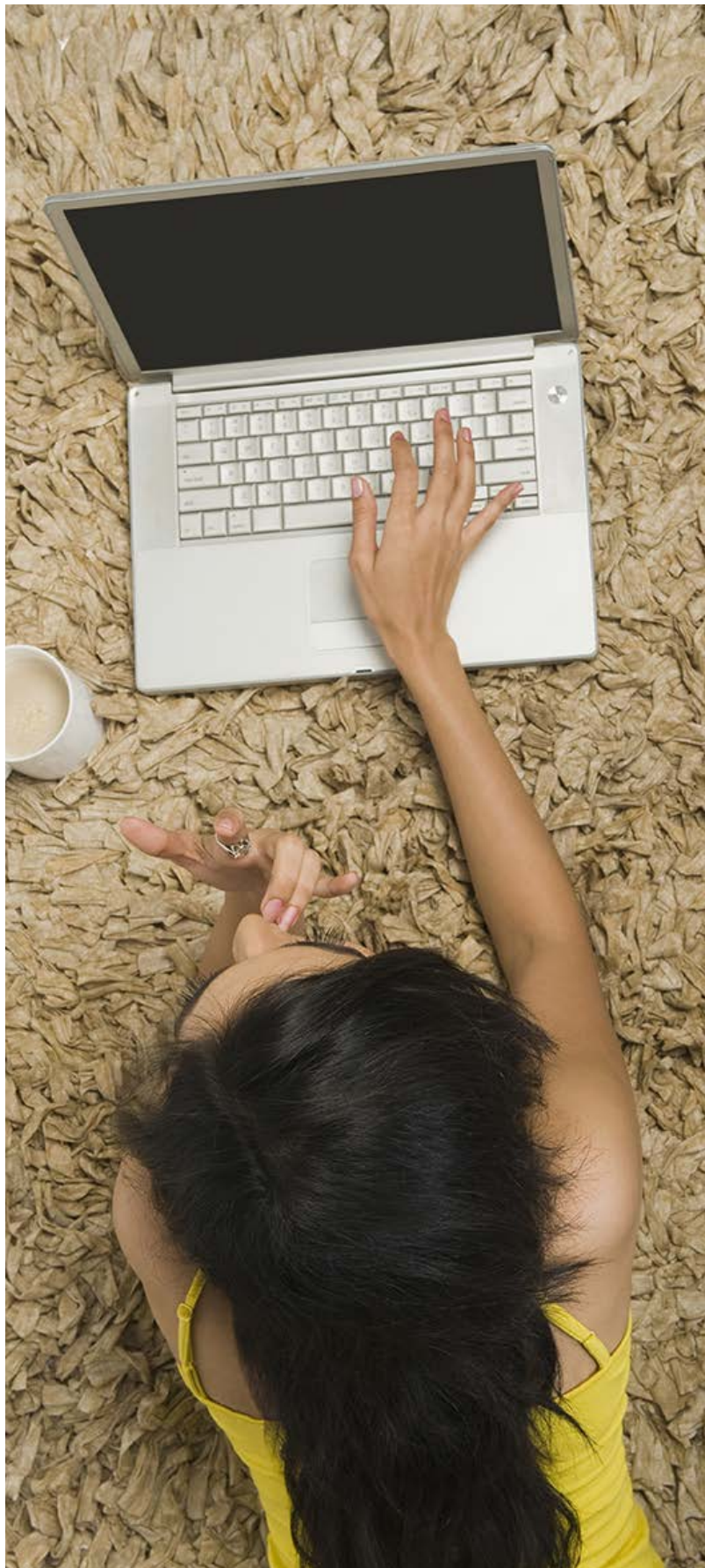
These queries assume the default table prefix of "wp_"

```
mysql> SELECT count(*) FROM wp_options WHERE  
autoload='yes';
```

A good rule of thumb is to shoot for fewer than 200 autoloaded queries on any given page (but having more than 200 isn't the end of the world). Usually, an excessive amount of autoloaded queries indicates an established blog that still carries weight from plugins and themes of yesteryear. Help your blog chug along by getting rid of those old options!

Addressing autoloaded queries is time consuming, detail-oriented work. It's extremely beneficial for your site from a troubleshooting perspective, since it can highlight what is negatively impacting your site's health, but it doesn't always result in a dramatic improvement in page load speed. Fixing autoloaded queries is important and necessary - but it's just one piece of the puzzle.





Weeding out the options

Next up it's time to analyze the Options table and determine the size of the options that are being autoloaded by using the "sum" and "length" functions in MySQL:

```
mysql> SELECT SUM(LENGTH(option_value)) as autoload_size FROM wp_options  
WHERE autoload='yes';
```

If you'd like to see which options specifically are causing the most trauma, try this:

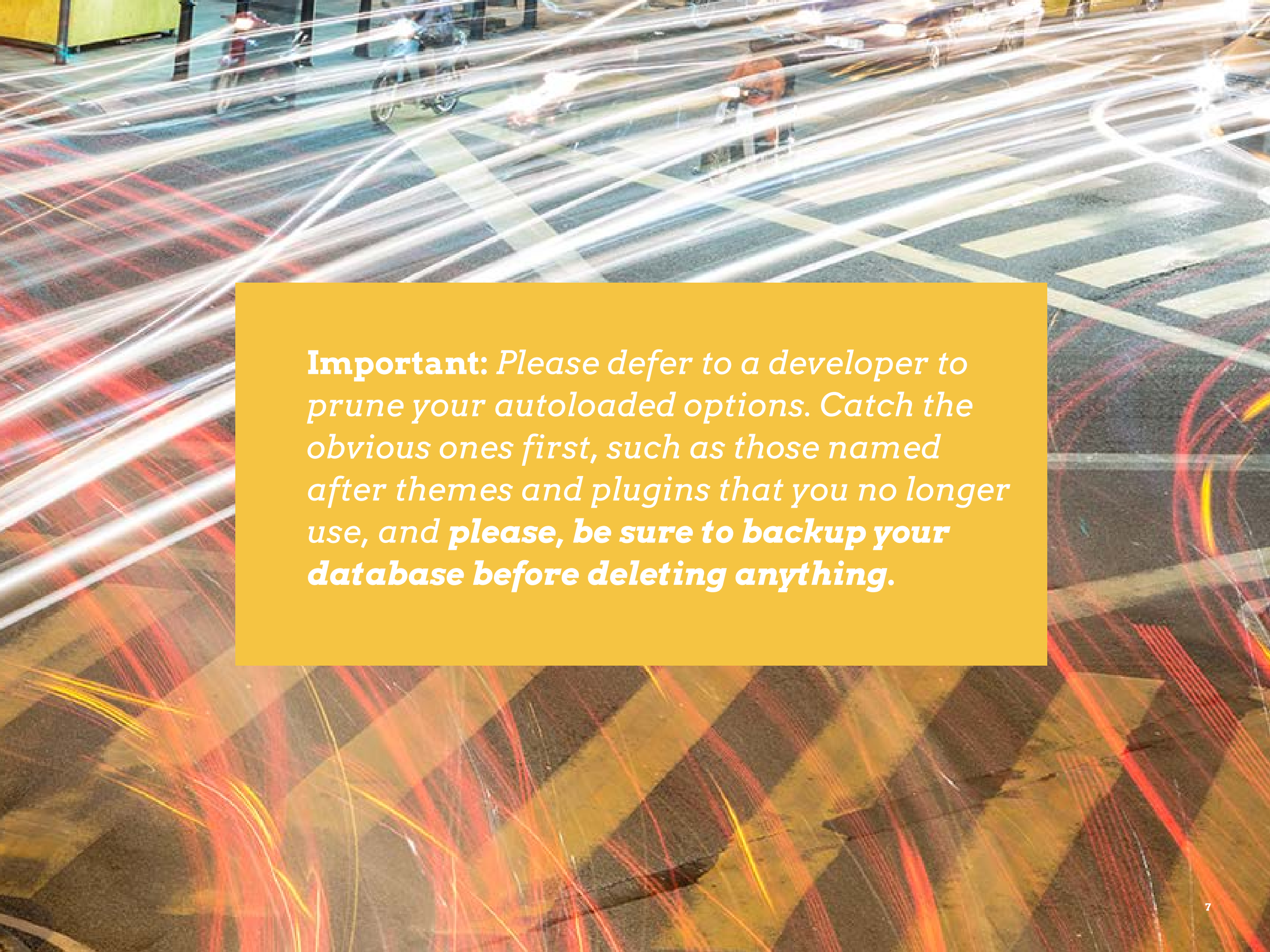
```
mysql> SELECT option_name, length(option_value) AS option_value_length  
FROM wp_options WHERE autoload='yes' ORDER BY option_value_length DESC  
LIMIT 10;
```

If your site is experiencing slowness and you're seeing an autoload_size in the MBs, removing the plugins or themes responsible could be an easy solution to speed up your page load time.

Try disabling any of the plugins or themes that might be indicated in the larger options.

If the "autoload" option is equal to "yes" in your Options table, that option could remain there and continue to plague your site's performance long after the plugin or theme is gone (this is also true if the option is not autoloaded).

A great way to test if a plugin's or theme's options are slowing down page load speed is to first disable the theme or plugin in question, set the offending option's autoload value to "no," and then test to determine if your site experiences better performance.



Important: *Please defer to a developer to prune your autoloaded options. Catch the obvious ones first, such as those named after themes and plugins that you no longer use, and **please, be sure to backup your database before deleting anything.***

Number of options

Another thing that may be slowing your blog is the number of options in the **table**. Although this is a less typical scenario, there's a chance this might be the single thing causing your site trouble. While not common, **when the number of options is a problem, it's a BIG problem.**

You can check to see exactly how long it takes to grab all of the autoload options like this (again, this assumes that you are using the table prefix "wp_"):

```
mysql> SELECT option_name, option_value FROM wp_options WHERE
autoload = 'yes';
```

However long that query takes to execute is how much time it's adding to each page load. While having a lot of options is generally a painful way to run, in the unlikely event that this query is taking the majority of a second or more, one solution is to add an index on the autoload column like this:

```
mysql> CREATE INDEX autoload ON wp_options (autoload);
```

If you notice that this doesn't speed up your query, you can drop the index like this:

```
mysql> DROP INDEX autoload ON wp_options;
```

Even with a master/slave database setup complete with all the bells and whistles, it's very common to see WordPress grind to a halt when it's used to track visitor data at scale. Instead, use a product or service that allows you to load a JavaScript library that handles the tracking automatically like Google Analytics, Piwik, or Clicky.

Logging visitor data doesn't work well on WordPress because MySQL locks its tables significantly longer to perform an INSERT than it does for a SELECT. When this happens, MySQL is getting requests from visitors who are asking to read from and write to the database — **but they can't access it immediately because it's locked from the request that was received before them, so they're stuck waiting.**



There are two things to be cautious of when doing this:

1. Adding indexes on a production database can lock your database for extended periods of time. Be sure to wait for a time when traffic is low before making any changes.
2. Adding an index isn't always the right move. In some scenarios it can actually add time to your page load. Before you make any changes, be sure to do some basic benchmarking so you can compare page load speeds before and after. Otherwise, dropping the index you just made may be in your future (see left).

Issues to keep an eye on

1. Misbehaving plugins or themes can cause the cron option to become bloated. When that's the case, you'll probably see it in your top 10 largest autoloaded options (yep, you guessed it - the option_name is "cron").
2. Watch out for plugins that track data— they may be storing their logs in the wp_options table and reading/writing to them on a per-page load basis (and even autoload them!).

Tracking visitor data with WordPress works fine for low traffic sites, but at scale it can cause a site to melt.

A warning about using WordPress to track visitor data

Do not use WordPress to track visitor data. Ever.

Thoughts on the backend

For some sites, having other bits of information — such as a secondary or micro blogroll in sidebars or footers — can make sense and increase a reader's engagement, but remember that at scale everything adds up. Consider generating the micro blogroll once and then storing it as a transient for simple retrieval later rather than rebuilding it from scratch each time.

PROGRAMMATIC WAYS TO REDUCE TOTAL QUERIES

Sometimes even after optimizing the Options table and getting your plugins under control, your site still won't perform at the levels you want. Perhaps you're seeing a bunch of queries coming from your theme, but it's working properly and looks great. There still may be room to cut out a few more queries on the road to recovery.

MAKING YOUR DATA COUNT

WordPress already brings plenty of data to your theme's doorstep — so make sure it counts!

Sometimes users just don't leverage the full power of WordPress, but that's easy to change, just use:

The `pre_get_posts` hook

Instead of making a new `WP_Query($best_args_ever)` or using `get_posts($same_amazing_args)` to get the content for the page, and perhaps disregarding what WordPress has already brought you, `pre_get_posts` allows you to craft and control the main query and shape it to your liking before it runs.

In doing this, you get what you need for the request and your page load only uses `WP_Query` once, which is a big win since it hits your database multiple times every time it loads. Some great examples on *how to use `pre_get_posts`* are right there in the WordPress codex.

THE FRONTEND

We've already attacked issues that can plague MySQL. Now let's focus on frontend optimization for WordPress and see what improvements we can make there.

KNOW YOUR PAGE LOAD

It's really simple these days to run your site through a page speed test and see 100 different dissections of how a site performs.

There are several great free options to test your page load speed:

- WebPageTest: <http://www.webpagetest.org/>
- Pingdom: <http://tools.pingdom.com/fpt/>
- Google: <http://developers.google.com/speed/pagespeed/insights/>
- GT Metrics: <http://gtmetrix.com/>

**Whichever tool you use,
the first thing you should
look at is the time to first byte.**

If your time to first byte is still more than 0.75 seconds, you should consider spending some more time massaging MySQL and improving things on the backend.

A note on minification plugins

If you install a minification plugin and portions of your styles or scripts are breaking, there's a good chance that the affected styles and/or scripts are not being enqueued properly using `wp_register_style/wp_register_script` and `wp_enqueue_style/wp_enqueue_script`.

In this event, verify that the plugin/theme is enqueueing and listing all dependencies correctly or contact the author/developer to let them know what's happening.



SIMPLE SOLUTIONS FOR IMAGES

When you look at your requests, are most of them images? If so, a great way to speed up page loads is to turn your images into CSS sprites.

CSS sprites

If you've ever delved into CSS sprites, you know that they can be tedious, frustrating, and time consuming. Fear not, though! There are many free resources out there that will do the work for you. Try out a couple and see what makes the most sense for your workflow.

When generating CSS sprites, make sure that you're combining images that are generally used together, such as all of the icons your site uses, the images that make up your navigation, or the images that make up your header and footer.

This will get you the greatest effect for your effort.

Smushing metadata

If images comprise the biggest portion of data in size from a page load, there are some great plugins to help mitigate that. Check out [*WP Smush.it*](#), [*FWWW Image Optimizer Cloud*](#), or [*Kraken Image Optimizer*](#).

These tools will keep the graphical integrity of each pixel! You will not lose any clarity in your images from smushing; only the metadata stored in the image is "smushed" (i.e.: what camera model snapped the pic, timestamp of snapshot, etc.).

Image lazy loader

If smushing didn't quite get you there, consider an alternative to that like an image lazy loader. These plugins can automate all of the work. The concept is that you only load the images that are visible on the user's screen and new images are only loaded as the user scrolls and needs them. This makes initial page loads much faster, especially on longer pages with a lot of images. It can also save high traffic sites from incurring additional bandwidth charges.

MINIFICATION

The biggest, quickest, and easiest win is to combat the number of total requests.

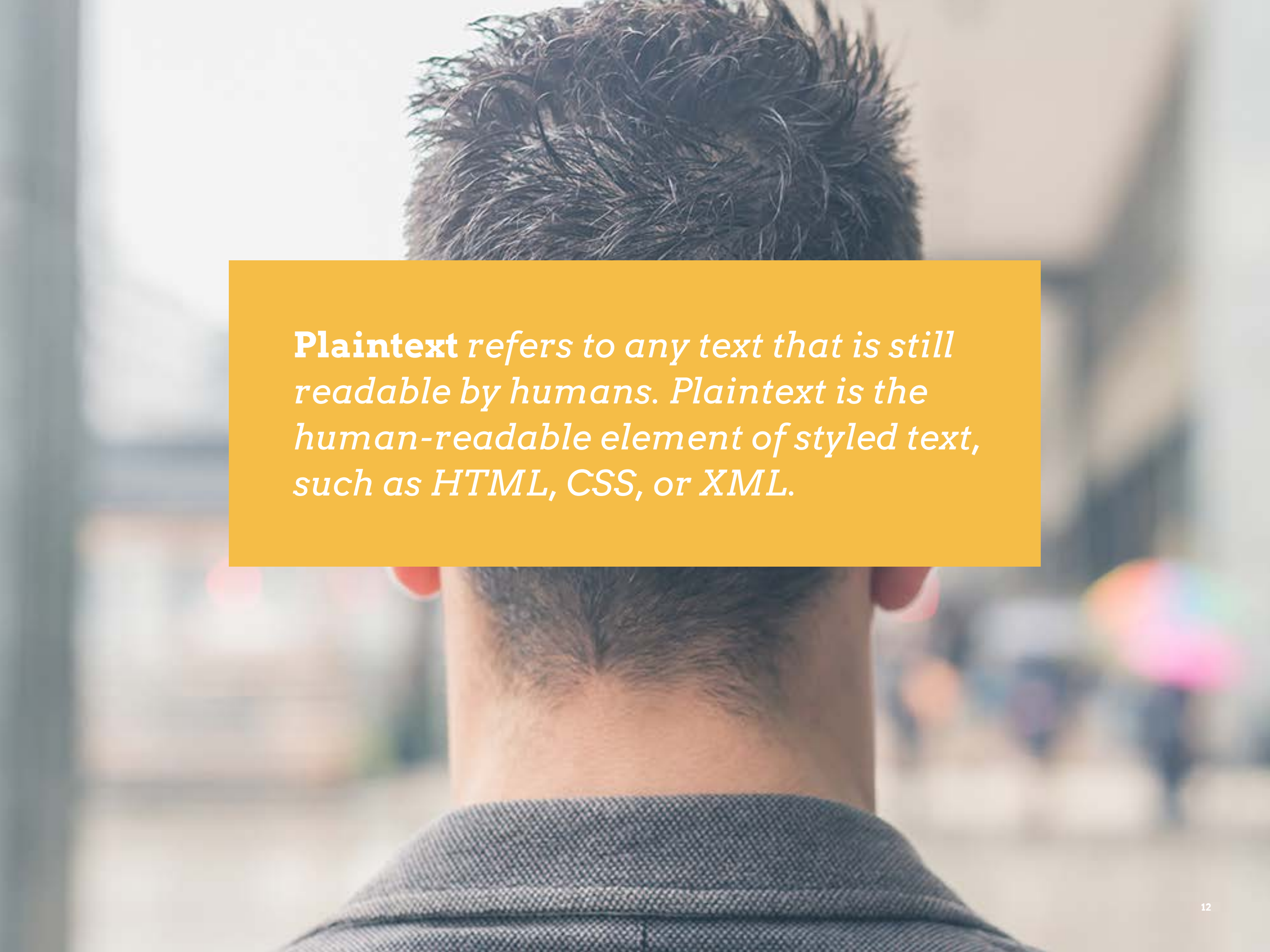
It's one thing if your page takes 20 requests to completely load, but if you're closer to 100 requests (or even more – yikes!), chances are that adding a plugin like [*BWP Minify*](#) will speed things up, especially if a good portion of those requests is between stylesheets and JavaScript files.

Turning many requests into just a few

BWP Minify's main purpose is to take all of the individual scripts and style sheets and mash each type into a single request of its own, lowering the number of overall requests.

By doing this, the visitor's browser doesn't have to make multiple requests to get all of the stylings or all of the scripts.

This is one of the easiest to implement high-impact wins for frontend performance.



Plaintext *refers to any text that is still readable by humans. Plaintext is the human-readable element of styled text, such as HTML, CSS, or XML.*

MINIMIZE ADMIN-AJAX ABUSE

The second biggest win from a scalability standpoint is minimizing the use of admin-ajax.php requests per page load, but it's a little more complicated to implement than basic minification.

When WordPress gets a request through /wp-admin/admin-ajax.php, a good portion of WordPress is loading along with it. While admin-ajax is designed to be as lightweight as possible, it still means another request to the server.

If you're seeing three, four, or even five separate requests to admin-ajax per page load, see what you can do to combine them into one (which does require some custom development work), or be sure that you absolutely need them. If you don't, get rid of the extra requests.

UTILIZING CDNS

The next step is to set up a content delivery network (CDN). CloudFlare provides an easy solution. Otherwise, there are many other great providers like MaxCDN or CDN.net.

A CDN allows you to serve the larger files of a request from a server that is better suited for it.

This typically also has the added benefit of allowing content to be served to your visitor from a location that is geographically closer to them which also helps improve page load speed.

ENSURE GZIP IS ENABLED

The last improvement that makes quick and easy work of all of the HTML, CSS, and JavaScript files (anything that is basically uncompressed plaintext) is to ensure that your server is gzipping the things it serves.

Most modern day hosts should have this enabled by default as it's in their best interest, but it's best to verify for yourself.

If you're familiar with looking at the headers of an http request, you can find out if you have gzip enabled there. One of the headers should say: "Content-Encoding: gzip." Otherwise, a quick and easy way to find out is to check your site using a tool like the [*http compression test*](#).





Avoid Disaster.

SOUND AS A POUND

By not preparing your WordPress site to serve heavy traffic, you can be setting yourself up for a disaster. Especially if it happens when a record number of people are interested in your site or business. Optimizing the frontend may seem like a lot of work, but remember that it's much easier for a potential customer to move on to a competitor's site when yours is down than it is for them to wait for your website to work again.

Even though this is intended to be a substantial and actionable list of things you can do today to improve your site's ability to scale, this is just the tip of the iceberg. Following these guidelines will help your site weather a storm of viral traffic, but you need to make sure **you also are running on a server that can handle a high number of requests from hundreds of concurrent users.**

THAT'S WHERE WP ENGINE COMES IN.

We've created an environment specifically designed to run and scale WordPress. If you've pushed site optimization to the limits and you're still looking for ways to speed up WordPress, our managed WordPress hosting platform can help.

For more helpful tips, check out our recordings from our webinar series:

"How To Handle Peak Capacity & Speed Up WordPress Sites."

THANK YOU.

