

A wide-angle photograph of Lake Tahoe. In the foreground, a small, snow-dusted island with a cluster of evergreen trees is visible in the middle of the lake. The lake's deep blue water stretches towards a range of mountains in the background. These mountains are partially covered in snow, particularly on their peaks. The sky above is a clear, pale blue.

ANALYTICS / SEO



Search engine optimization is alive
and well but gaming the system is
largely dead.

There are basics we will review but
Google and others operate on a
level that is near impossible to
game any longer.

SOME BASICS

- Control search metadata in the head of your document
- Format body of document semantically so search engines understand hierarchy of content
- Give things time! And check your results

**CONTENT IS
KING**



#1 THING

Sounds ridiculous but search engines are pretty good at getting you to good content. So...make good content. Much of the web is garbage. Don't expect good SRP position with bad content.

SEO: THINK <head>



SEO: NOT <body>



XML SITEMAPS

Once a search engine gets to your site, you can tell it the hierarchy of pages you want it to crawl through and the frequency. Search engines mostly like to find a XML-based sitemap that says this to them.

XML SITEMAPS

You'll typically see these used with
WordPress / Drupal sites.

My favorite one is Google XML
Sitemaps:

[https://wordpress.org/plugins/
google-sitemap-generator/](https://wordpress.org/plugins/google-sitemap-generator/)

Publishes to:

<http://yoururl.com/sitemap.xml>

XML SITEMAPS

There is a protocol for how these are structured:

[http://www.sitemaps.org/
protocol.html](http://www.sitemaps.org/protocol.html)

In my whole life, I've never made one of these from scratch - unlikely you will either.

CHECK ONE OUT

<http://nutritionwonderland.com/sitemap.xml>

TALK TO THE COMPUTERS

Use the head of your HTML pages to talk with the other computers that will view the page. Metadata is the next most important thing to get right, after having solid content.

GOOGLE-METADATA

Unfortunately, every major platform wants you to talk to it slightly differently. Google doesn't ask for much meta info (they will automatically find your subpages, especially if you used an XML sitemap):

```
<!-- Google meta -->
<meta name="description"
content="Your tagline here, will
appear on Google SRP">
```

GOOGLE-METADATA

Google has a great suite of tools that allow you to tweak what's going on with your sites:

<https://www.google.com/webmasters/tools/home?hl=en>

FACEBOOK METADATA

Started the Open Graph standard:
<http://ogp.me/>

Looks like this in reality:

```
<!-- Facebook meta, ridiculously complex but necessary -->
<meta property="og:title" content="Name of your site">
<meta property="og:description" content="Tagline here">
<meta property="og:type" content="website">
<meta property="og:url" content="http://domain.com/">
<meta property="og:image" content="http://domain.com/
pic.jpg">
```

FACEBOOK METADATA

Where you can test your Facebook metadata - see what they see:

[https://developers.facebook.com/tools/
debug/](https://developers.facebook.com/tools/debug/)

Your site MUST be on a public server for this to work. Be mindful of robots.txt settings - Facebook will respect those.

TWITTER METADATA

So naturally, Twitter had to come up with their almost identical standard (Twitter Cards):

<https://dev.twitter.com/cards/markup>

```
<meta name="twitter:title" content="Site title">
<meta name="twitter:description" content="Tagline">
<meta name="twitter:card" content="summary">
<meta name="twitter:site" content="@handle">
<meta name="twitter:url" content="http://example.com/">
<meta name="twitter:image" content="http://example.com/
pic.jpg">
```

TWITTER METADATA

Testing Twitter Card metadata:

<https://cards-dev.twitter.com/validator>

Must be authenticated into Twitter
already and you must also be pointing
towards a site on a public server.



CREATE LOCALIZED CONTENT

Search engines are very localized now and will serve different results based on where you are. Be where the users you want to reach physically reside.

MAJOR SEARCH ENGINES

US / Europe / Latin America: Google

US Secondary: Bing, Yahoo

Russia: Yandex

China: Baidu, WeChat (sorta)



**RESPONSIVE
DESIGN**

UNIVERSAL CODEBASE

Search engines, especially Google, heavily favor sites that work on mobile phones natively. Make sure your site is responsive.

A photograph of Bodiam Castle, a medieval moated castle in East Sussex, England. The castle is built of light-colored stone and features several towers, including a prominent round tower on the left and a square tower on the right. It is situated in a moat, with trees and foliage visible in the foreground and background. The sky is blue with scattered white clouds.

SECURITY

HTTPS EVERYWHERE

The modern web is dangerous because of it's benign design. All requests are assumed safe. The big shift is towards using HTTPS for all transactions. You can do this free now with Let's Encrypt. Use it.



ROBOTS.TXT

ROBOTS.TXT

The primary way to talk to a search engine is through a file called robots.txt. Put it in the root of your project for search engines to see it.

See more here:

<http://www.robotstxt.org/>

[https://developers.google.com/
search/reference/robots_txt](https://developers.google.com/search/reference/robots_txt)

ROBOTS.TXT

By default, all search engines 'crawl' everything on your site. If you want that behavior, simply don't do anything.

```
User-agent: *
Disallow:
```

ROBOTS.TXT

If you're making a development or something in general you don't want a search engine to see:

```
User-agent: *
Disallow: /
```

ROBOTS.TXT

To disallow certain webpages:

```
User-agent: *
Disallow: /this-directory/
Disallow: /that-directory/
Disallow: /mypage.html
```

ROBOTS.TXT

You can talk to particular search engines if you want to, but why limit yourself?

```
User-agent: googlebot
Disallow: /this-directory/
Disallow: /that-directory/
Disallow: /mypage.html
```

CHECK OUT A REAL ONE

<http://www.ltk.com/robots.txt>

REDIRECTS

Sometimes a page moves and you want to tell search engines that it moved. You have to use HTTP status codes at the server level to do this using a 301 redirect.

REDIRECTS AND SEO

If your site has been around and accumulated a lot of weight on Google, you want to maintain that history so redirects allow you tell search engines to do just that.

HTTP STATUS CODES

1xx: Informational (rarely seen)

2xx: Success (often just 200: success)

3xx: Redirect (often 301: redirect)

4xx: User Errors (often 404: not found)

5xx: Server Errors (often 503: timeout)

More here: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>

301 VS 302

Depending on the redirect HTTP status code you use, you are telling the search engine something different.

301 = permanent

302 = temporary

99% of the time, use 301. Why would you make a temporary redirect?

A WORD ON SERVERS

There are mainly two servers that power the entire internet:

Apache HTTP Server ([http://
httpd.apache.org/](http://httpd.apache.org/))

Nginx Server (pronounced engine-x)
([https://www.nginx.com/resources/wiki/
community/faq/](https://www.nginx.com/resources/wiki/community/faq/))

A WORD ON SERVERS

Apache / Nginx do the same thing but get to that goal differently. You have to figure out which one you're using if you want to do redirects at the server level.

Redirects can also be controlled via CMS systems. I've done them both by adjusting servers and CMS'.

APACHE

You have to access the `mod_alias` module within your Apache installation and put code like this in there:

```
Redirect 301 /oldlocation http://  
www.domain2.com/newlocation
```

APACHE

You have to access the `mod_alias` module within your Apache installation (typically system file called `.htaccess` at the root of your server) and put code like this in there:

```
Redirect 301 /oldlocation http://  
www.domain2.com/newlocation
```

APACHE

If you had to remap an entire site you would use mod_rewrite module to dynamically created rewritten URLs. This gets very complicated but this is a nice guide:

[https://www.digitalocean.com/
community/tutorials/how-to-set-up-
mod_rewrite](https://www.digitalocean.com/community/tutorials/how-to-set-up-mod_rewrite)



Nginx is a bit different - it looks more like Javascript:

```
server {  
    listen      80;  
    server_name example.org;  
    return      301 http://  
www.example.org$request_uri;  
}
```

NODE.JS

Things get weird with Node.JS. Node is typically used with Javascript frameworks. This requires a lot of backend configuration combined with redirects at the server level.

ANALYTICS

REVIEWING TWO

Google Analytics:

<https://www.google.com/analytics>

Intercom:

<https://www.intercom.io/>

GOOGLE ANALYTICS

- Generally makes sense if you are a website, dispensing information with very little feedback from your audience.
- It uses a page-oriented model for tracking user behavior.

INTERCOM

- Generally makes sense if you are a web app, where your audience is creating a lot of content on your site (Trello, Gmail, Facebook, etc).
- It uses an user action-based model for tracking.

HARD TO DO IN A DECK

I'm going to just have to show you how these work, so notes will be light in through here.

The general idea of all analytics is to capture everything and then segment data into useful reporting groups. We'll go through how that works for both tools.