# راهکارهایی برای بهینه سازی سایت

برای سایت های پر بازدید، بهینه بودن سایت از اهمیت زیادی برخورداره. بهینه بودن یعنی اینکه سایت در کمترین زمان ممکن لود بشه و مصرف منابع سرور هم به حداقل ممکن برسه.
میشه فرایند بهینه سازی رو به دو بخش سمت کاربر و سمت سرور تقسیم کرد.
بهینه سازی سمت سرور یعنی بهینه کردن برنامه و کد های سمت سرور برای اینکه حداقل منابع رو مصرف کنه و در حداقل زمان ممکن اجرا بشه. که این قسمت از بهینه سازی فعلا موضوع بحث ما نیست.
و اما بهینه سازی سمت کاربر شامل کلیه اقداماتی است که برای کاهش مصرف پهنای باند وافزایش سرعت لود شدن انجام میشه.
در ادامه این مقاله که در واقع برگرفته از کتاب «High Performance Web Sites» هست می پردازیم به اهمیت و راهکارهای بهینه سازی سمت کاربر.
لازم به ذکر است که این کتاب نوشتهی Steve Souders مسئول performance در یاهو می باشد.
اهمیت بهینه سازی سمت کاربر
از ملاک های اصلی جلب رضایت کاربران، کارایی و سرعت سایت است. بررسی ها نشان داده که کمتر از ۲۰ درصد از زمان بارگزاری سایت صرف لود شدن html میشود و مابقی یعنی بیش از ۸۰ درصد از زمان، صرف بارگزاری آنچه درون صفحه است میشود. که توجه به بهینه سازی این قسمت (ملحقات صفحه) تا حدود زیادی سرعت و کارایی سایت را افزایش میدهد.
سه دلیل عمده برای اهمیت بهینه سازی ملحقات صفحه وجود دارد:

۱. پتانسیل بیشتری برای بهینه سازی ملحقات وجود دارد. کاهش پنجاه درصدی آن حدود ۴۰ درصد در افزایش سرعت سایت تاثیر دارد درحالی که کاهش ۵۰ درصدی حجم html تاثیرش کمتر از ده درصد است.
۲. بهینه سازی ملحقات صفحه، زمان و منابع کمتری نیاز دارد نسبت به بهینه سازی سمت سرور (شامل تغییر معماری و کدنویسی مجدد، یافتن و رفع مشکلات و گلوگاه ها در کد، اصلاحات سخت افزاری و قرار دادن دیتابیس ها بر روی سرور های مجزا و ... )
۳. در عمل ثابت شده که این روش خوب جواب می دهد. بیش از ۵۰ تیم در یاهو به این روش توانستند حدود ۲۵ درصد سرعت سایت ها را افزایش دهند.
و اما راهکارها:
۱. کاهش تعداد درخواست های HTTP
o استفاده از تکنیک CSS Sprites که موجب کاهش تعداد تصاویر استفاده شده در css میشود.
o الحاق فایلهای css به یکدیگر و همچنین اسکریپت ها
o استفاده از تکنیک inline image (البته این تکنیک در IE پشتیبانی نمیشود)
۲. استفاده از CDN
یک CDN مجموعه ای از وبسرورهاست که در نقاط مختلف دنیا توزیع شده اند و معمولا محتوای static رو میزبانی میکنند. تا بازدید کنندهها بتوانند از نزدیکترین سرور فایل ها را دریافت کنند. این تکنیک برای سایت های کوچک مقرون به صرفه نیست.
۳. افزودن Expires Header
مرورگرها میتوانند پاسخ درخواست های HTTP را کش کنند. با ارسال هدر Expires میتوان به مرورگر گفت که پاسخ را تا چه زمانی کش کند. برای مثال برای محتوای استاتیک میتوان این مقدار را بر روی یکسال گذاشت تا در درخواست های بعدی مرورگر از کش استفاده نماید.
۴. استفاده از gzip
تقریبا اکثر مرورگرها از gzip پشتیبانی میکنند. فشرده سازی gzip بر روی محتوای متنی نظیر css ها و اسکریپت ها و html کارایی خوبی دارد و بیش از ۵۰ درصد حجم درخواست ها را کاهش میدهد.
۵. stylesheet ها را در ابتدای صفحه قرار دهید.
در بعضی مرورگرها درصورتی که stylesheet در انتهای صفحه باشه باعث میشه که محتوای صفحه دوبار رندر بشه و زمان بیشتری صرف بشه و ضمنا دفعه اول بدون style رندر میشه که ظاهر خوشایندی نداره. در بعضی مرورگرها هم موجب میشه رندر بعد از لود شدن محتوا و ملحقات صفحه انجام بشه که باعث میشه ابتدا یک صفحه خالی نمایش داده بشه و رندر با کمی تاخیر انجام بشه.
ضمنا قرار دادن stylesheet ها در ابتدای صفحه موجب میشه رندر به شکل مطلوبی و به صورت تدریجی همزمان با لود شدن محتوا انجام بشه.
۶. اسکریپتها را در انتهای صفحه قرار دهید.

قرار دادن اسکریپتها در ابتدای صفحه باعث میشه جلوی رندر شدن و دانلود شدن سایر محتویات صفحه تا اتمام لود شدن اسکریپت گرفته بشه.
و اما قرار دادن اسکریپتها در انتهای صفحه باعث میشه ابتدا محتوا رندر بشه و سپس سایر ملحقات و اسکریپت ها در پس زمینه لود بشه.
۷. از CSS Expression ها پرهیز کنید
CSS Expression ها فقط در IE پشتیبانی میشوند. با استفاده از این قابلیت میتوان برای مقدار دادن به یک خصوصیت در CSS از جاوااسکریپت استفاده کرد. و این کد اسکریپت در مواقعی نظیر resize شدن صفحه و یا حرکت موس مجددا مقدار رو محاسبه میکنه. این اجرای مکرر میتونه باعث کاهش کیفیت و یا حتی هنگ شدن مرورگر بشه. پس بهتره از این CSS Expression ها استفاده نکنید.
۸. جاوااسکریپت ها و CSS ها را از HTML جدا کنید
گرچه اینکار قاعده شماره ۱ رو نقض میکنه ولی با توجه به اینکه با جدا کردن CSS و JS ها و ذخیره اونها در فایلهای استاتیک میشه قواعد ۲ و ۳ و ۴ رو در موردشون بکار بست پس این کار به صرفه است.
۹. تعداد DNS Lookup ها را کم کنید
مرورگرها برای ارتباط با سرورها از IP اونها استفاده میکنند بنابراین مجبورند به ازای هر Hostname ابتدا یک درخواست به DNS سرور ارسال کنند و IP رو بدست بیارند. ضمن اینکه مرورگر میتونه از قابلیت Keep-Alive در مورد درخواستهایی که مربوط به یک Hostname هستند استفاده کنه که باعث میشه چندین درخواست از طریق یک کانکشن انجام بشه.
پس بهتره محتویات سایتتون از Hostname های مختلف نباشه. (دقت کنید که www.example.com و example.com هم دو Hostname متفاوت هستند)
۱۰. جاوا اسکریپت ها را بچلانید
چلاندن(Minify) یعنی حذف اضافات (شامل کامنتها، فضاها و اینترهای اضافی) از درون اسکریپتها برای کاهش حجم فایل. البته یک گام فراتر هم میشه رفت با استفاده از پیچوندن (Obfuscation) که در این روش علاوه بر حذف اضافات، شناسههای استفاده شده در کد رو هم با شناسههای کوتاهتر جایگزین میکنند.
روش چلاندن رو در مورد CSS هم میشه به کار برد ولی تاثیرش در مورد CSS ها کمتره. البته در مورد CSS میشه با بهینه کردن کد تا حدی از حجم CSS کم کرد. (در این مورد آقایان الوانی و ستاری به طور مفصل توضیح دادند.)
۱۱. از Redirect کردن پرهیز کنید
هر Redirect مستلزم یک درخواست اضافه است و ضمنا این درخواست به صورت موازی هم نیست یعنی تا کامل نشده درخواستهای بعدی ارسال نمیشوند. پس تا جائیکه ممکنه از Redirect پرهیز کنید.
۱۲. اسکریپتهای تکراری را حذف کنید
گاهی ممکنه پیش بیاد که یک اسکریپت دو بار در صفحه درج شده باشه مثلا توی کار تیمی ممکنه یک نفر یک اسکریپت رو در یک جای صفحه درج کنه و یک نفر دیگه که از این موضوع خبر نداره مجدد اون رو در جای دیگه ای درج کنه.
این کار دو تا مشکل ایجاد میکنه. یکی اینکه تعداد درخواستهای HTTP افزایش پیدا میکنه و دو اینکه زمان اضافه ای صرف اجرای مجدد اون اسکریپت میشه. البته مورد اول فقط در IE و اون هم در حالتی که اسکریپت قابل کش شدن نباشه اتفاق میافته.
۱۳. پیکربندی ETag ها
مرورگرها از هدر ETag برای کش کردن محتوای استاتیک تا زمانی که تغییر نکرده استفاده میکنند (مشابه هدر Last-Modified). ولی ETag در صورتی که سایت بر روی بیش از یک سرور میزبانی شده باشد میتواند مشکل ساز شود. مثلا آپاچی به طور پیشفرض برای تولید ETag از «inode – حجم – زمان ویرایش» استفاده میکنه و inode در مورد یک فایل روی سرورهای مختلف فرق میکنه و در صورتی که مرورگر یک فایل رو از یک سرور دریافت کنه و در مراجعه بعدی به اون فایل برای چک کردن تغییرات از سرور دیگری استفاده کنه، با توجه به اختلاف ETag ها، سرور مجددا فایل رو میفرسته که باعث میشه مرورگر از کش استفاده نکنه. IIS هم مشکل مشابهی داره.
اما راهکار اینه که یا کلا ETag رو حذف کنید و یا اینکه اصلاحش کنید برای مثال در آپاچی باید inode رو از ETag حذف کنید. (این کار رو با htaccess میشه انجام داد.)
۱۴. Ajax را قابل کش شدن کنید
یک برنامهی وب دویی را در نظر بگیرید که مثلا برای لود کردن لیست ایمیلها از Ajax استفاده میکند. در این حالت میتوان تا زمانی که لیست آدرس ها تغییر نکرده این درخواست را با استفاده از هدر Expires قابل کش کرد.
ضمنا قواعد ۴، ۹، ۱۰، ۱۱ و ۱۳ هم در مورد درخواستهای اجکسی قابل استفاده هستند.
اینها مواردی بود که در کتاب ذکر شده بودند. بعدها موارد دیگه ای هم به اینها اضافه شد و الان تعدادش به ۳۴ مورد رسیده که میتونید اینجا ببینید.
یاهو یک پلاگین برای فایرفاکس برای چک کردن این فاکتورها نوشته به نام YSlow که البته با Firebug کار میکنه.

این پلاگین بر اساس این فاکتورها کیفیت سایت شما رو چک میکنه و مواردی که رعایت نشدند رو مشخص میکنه و یک نمره هم از ۱۰۰ به سایت شما میده.
ابزارهای مختلفی وجود داره که پیاده سازی این موارد رو ساده میکنه.
مثلا برای چلاندن جاواسکریپتها (مورد ۱۰) میتونید از ابزارهایی مثل JSMin و Packer و YUI Compressor استفاده کنید.
ابزارهایی هم هستند که با نصب اونها روی سایت به صورت خودکار بعضی از این موارد روی سایت شما اعمال میشوند. مثل Minify و PHPSpeedy و SmartOptimizer.
برای مثال SmartOptimizer میتونه موارد ۱ و ۳ و ۴ و ۱۰ و ۱۳ رو روی سایت شما اعمال کنه و نمره YSlow شما رو حدود ۲۰-۱۰ واحد افزایش بده.