bringing designers & developers closer together
I used to be a reckless designer.
THE ROAD TRIP IS LIVE!
Follow along as the Flaming Lips attempt to break a world record.

MEMPHIS  CLARKSDALE  OXFORD  JACKSON  HATTIESBURG  BILOXI  BATON ROUGE  NEW ORLEANS

VOTING IS STILL OPEN IN SOME CATEGORIES! ENTER YOUR CODE TO UNLOCK VOTING:

NOW PLAYING:
Channel 1: The Bus Route
We’re at the fourth stop on our route. Watch us avoid the rain and see what the locals have to say.

THE LIVE FEEDS:

@OMusicAwards
Lady Gaga has won the title of Best Artist With A Cameraphone!
#winners

ThatGuyDave
Lovin’ Yeah Dog on Camera 3.
#omashow

5 minutes ago

TeamTokioHotel
Tokio Hotel better win Fan Army
FTW or else I’ll cry!!!!!!!!!! #omashow

5 minutes ago
REQUESTS: 136
PAGE WEIGHT: 5.9MB
LOAD TIME: 2M 46S
Server Delays Experiment: Results

<table>
<thead>
<tr>
<th>Time to Click (increase in ms)</th>
<th>Distinct Queries/User</th>
<th>Query</th>
<th>Refinement</th>
<th>Revenue/User</th>
<th>Any Clicks</th>
<th>Satisfaction</th>
<th>Time to Click (increase in ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50ms</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>200ms</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-0.3%</td>
<td>-0.4%</td>
<td>500</td>
</tr>
<tr>
<td>500ms</td>
<td>-</td>
<td>-0.6%</td>
<td>-1.2%</td>
<td>-1.0%</td>
<td>-0.9%</td>
<td>-</td>
<td>1200</td>
</tr>
<tr>
<td>1000ms</td>
<td>-0.7%</td>
<td>-0.9%</td>
<td>-2.8%</td>
<td>-1.9%</td>
<td>-1.6%</td>
<td>-</td>
<td>1900</td>
</tr>
<tr>
<td>2000ms</td>
<td>-1.8%</td>
<td>-2.1%</td>
<td>-4.3%</td>
<td>-4.4%</td>
<td>-3.8%</td>
<td>-</td>
<td>3100</td>
</tr>
</tbody>
</table>

- Means no statistically significant change

- Strong negative impacts
- Roughly linear changes with increasing delay
- Time to Click changed by roughly double the delay
“We made the new platform 60% faster and this resulted in a 14% increase in donation conversions.”
"To stay in Google's good graces, websites must be designed so they load quickly on mobile devices."

SAN FRANCISCO (AP) — Google is about to change the way it ranks websites, a shift that's expected to sway where millions of people shop, eat and book travel.

The revised formula, scheduled to be released Tuesday, will favor websites that Google defines as "mobile-friendly." Websites that don't fit the description will be demoted in Google's search results on smartphones while those meeting the criteria will be more likely to appear at the top of the rankings — a prized position that can translate into more visitors and money.

Although Google's new formula won't affect searches on desktop and laptop computers, it will have a huge influence on how and where people spend their money, given that more people are relying on their smartphones to compare products in stores and look for restaurants. That's why Google's new rating system is being billed by some search experts as "Mobile-geddon."

"Some sites are going to be in for a big surprise when they find a drastic change in the amount of people visiting them from mobile devices," said Itai Sadan, CEO of website-building service Duda.
designers & developers often work in silos

some designs are hard to make fast

being fast is important
small interdisciplinary teams
guiding principles
Speed is more important than design embellishment.

People are filling small gaps in their day with news. It must load fast on all touchpoints.

The design should feel light and nimble, always fresh and up to date. Never heavy, slow to load or clogged up with content.

Users expect sites to render in under 2 seconds.
prototype early
measure performance from the start
## START RENDER TIME BUDGET

<table>
<thead>
<tr>
<th>Current Start Render</th>
<th>Start Render Budget</th>
<th>Increase over 30 days (42%)</th>
<th>Over budget (-13%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.7s</td>
<td>1.5s</td>
<td>0.5s</td>
<td>-0.2s</td>
</tr>
</tbody>
</table>

**Graph:**
- Red line: Current Start Render (1.7s)
- Green line: Start Render Budget (1.5s)
- Increase over 30 days (0.5s, 42%)
- Over budget (-0.2s, -13%)

**Marked Point:**
- 15 Apr
- Start Render: 2.2s
in-page reminders
show what’s beaconed
bookmarklets
window.onload is not the best metric for measuring website speed

We haven't actually been “ignoring” this issue. We've acknowledged it, but we haven't coordinated our efforts to come up with a better replacement. Let's do that.

window.onload is so Web 1.0

What we're after is a metric that captures the user's perception of load time. The new, perception.ready() isn't on any browser's roadmap, alas. Back then, pages were mostly HTML and images. JavaScript was being loaded after the delays and blocked rendering they introduce. It wasn't perfect, but window.onload was close enough. Plus it had other desirable attributes:

- standard across browsers – window.onload means the same thing across all browsers. (The only exception I'm aware of is that IE 6-9 don't wait for async scripts before firing window.onload, while most other browsers do.)
- measurable by 3rd parties – window.onload is a page milestone that can be measured by someone other than the website owner, e.g., metrics services like Keynote Systems and tools like Boomerang. It doesn't require website owners to add custom code to their pages.
- measurable for real users – Measuring window.onload is a lightweight operation, so it can be performed on real user traffic without harming the user experience.

Web 2.0 is more dynamic

Fast forward to today and we see that window.onload doesn't reflect the user perception as well as it once did.
overly optimistic

onload: 3.9s

98% ATF rendered: 4.7s

too critical

99% ATF rendered: 2.0s

onload: 9.7s
### Performance Results (Median Run)

<table>
<thead>
<tr>
<th></th>
<th>Load Time</th>
<th>First Byte</th>
<th>Start Render</th>
<th>Speed Index</th>
<th>DOM Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>First View (Run 3)</td>
<td>3.62s</td>
<td>0.466s</td>
<td>1.093s</td>
<td>1717</td>
<td>3285</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Document Complete</th>
<th>Fully Loaded</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time</td>
<td>Requests</td>
</tr>
<tr>
<td></td>
<td>9.69s</td>
<td>122</td>
</tr>
</tbody>
</table>

### Test Results

Run 1:

- **Waterfall**
- **Screen Shot**
- **Video**
  - Filmstrip View
  - Watch Video
This would be great except for one little detail, it is unbounded. If a page spins for 10 seconds after reaching visually complete the score would keep increasing. Using the "area above the graph" and calculating the unrendered portion of the page over time instead gives us a nicely bounded area that ends when the page is 100% complete and approaches 0 as the page gets faster:

\[
\text{Speed Index} = \int_0^{\text{end}} 1 - \frac{VC}{100} \, dt
\]

end = end time in milliseconds
VC = % visually complete

The Speed Index is the "area above the curve" calculated in ms and using 0.0-1.0 for the range of visually complete. The calculation looks at each 0.1s interval and calculates \( \text{IntervalScore} = \text{Interval} \times (1.0 - (\text{Completeness}/100)) \) where \( \text{Completeness} \) is the % Visually complete for that frame and \( \text{Interval} \) is the elapsed time for that video frame in ms (100 in this case). The overall score is just a sum of the individual intervals: \( \text{SUM(IntervalScore)} \)
video
custom metrics

define *most important* elements on the page

measure using User Timing

track with RUM *and* synthetic
Improving performance on twitter.com

Tuesday, May 29, 2012 | By Twitter (@twitter) 05/29/2012 - 21:23

To connect you to information in real time, it's important for Twitter to be fast. That's why we've been reviewing our entire technology stack to optimize for speed.

When we shipped #NewTwitter in September 2010, we built it around a web application architecture that pushed all of the UI rendering and logic to JavaScript running on our users' browsers and consumed the Twitter REST API directly, in a similar way to our mobile clients. That architecture broke new ground by offering a number of advantages over a more traditional approach, but it lacked support for various optimizations available only on the server.

Reducing time to first tweet

Before starting any of this work we added instrumentation to find the performance pain points and identify which categories of users we could serve better. The most important metric we used was "time to first Tweet". This is a measurement we took from a sample of users, (using the Navigation Timing API) of the amount of time it takes from navigation (clicking the link) to viewing the first Tweet on each page's timeline. The metric gives us a good idea of how snappy the site feels.
Eric Lawrence @ericlaw · 45m
If you’re going to mangle your JPEG into a 256 color PNG, at least have the decency to use Zopfl to deflate it.

Almaer @dalmaer · 1h
What does my “if only a minute don’t keep bugging me” setting @GoogleMaps? #MEOError #ux

Yehuda Katz retweeted
James Kyle @thejameskyle · 3h
Breaking News: #ThoughtLeaders still unsure about this new JavaScript thing. @wycats reporting live at the scene.

Fastly @fastly · 2h
Good morning #rsac! We’ll be here all week, so come visit us at booth #2736 and hear about how we can help your site.
Image Custom Metric

RESOURCE TIMING: 1516 ms
IMAGE ONLOAD: 3199 ms
MUTATION OBSERVER: 3174 ms
offsetHeight POLLING: 3253 ms
INLINE SCRIPT: 5295 ms

actual image display: ~5200 ms
Image Custom Metric

RESOURCE TIMING: 1516 ms

performance
  .getEntriesByName("hero.jpg")[0].duration

actual image display: ~5200 ms
<img src="hero.jpg" onload="performance.mark('hero')">
var observer = new MutationObserver(obsCallback);
observer.observe(document, {
  childList: true, attributes: true, subtree: true });
function imagePolling() {
  var hero = document.getElementById('hero');
  if ( hero.offsetHeight ) {
    performance.mark('hero'); }
  else { 
    setTimeout(imagePolling, 100); } 
}

OFFSETHEIGHT POLLING: 3253 ms
INLINE SCRIPT: 5295 ms

actual image display: ~5200 ms
Image Custom Metric

actual image display: ~5200 ms

INLINE SCRIPT: 5295 ms
Image Custom Metric

RESOURCE TIMING: 7510 ms
IMAGE ONLOAD: 7244 ms
MUTATION OBSERVER: 3161 ms
offsetHeight POLLING: 3232 ms
INLINE SCRIPT: 5184 ms
when is image displayed?

max(image onload, inline script)

```html
<img src="hero.jpg"
    onload="performance.mark('hero1')">

<script>
    performance.mark("hero2");
</script>
```
**custom metrics**

### ALL ADS RENDERED

<table>
<thead>
<tr>
<th></th>
<th>Current</th>
<th>Change over 30 days</th>
<th>Change over 30 days</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ALL ADS RENDERED</strong></td>
<td><strong>9.3s</strong></td>
<td><strong>1.9s</strong></td>
<td><strong>26%</strong></td>
</tr>
</tbody>
</table>

![Graph showing the trend of ALL ADS RENDERED over time.](image)

### APP BEGIN

<table>
<thead>
<tr>
<th></th>
<th>Current</th>
<th>Change over 30 days</th>
<th>Change over 30 days</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>APP BEGIN</strong></td>
<td><strong>2.9s</strong></td>
<td><strong>0.6s</strong></td>
<td><strong>26%</strong></td>
</tr>
</tbody>
</table>

![Graph showing the trend of APP BEGIN over time.](image)
small, interdisciplinary teams
guiding principles
prototype early
measure performance from the start
performance budgets
UX metrics: start render, Speed Index
filmstrips, video
custom metrics

Attend the Velocity web performance and DevOps conference to join the engineers, developers, business technology, and operations pros defining the modern, IT-driven business.
THANK YOU

@souders
http://stevesouders.com/talks.php
• https://speakerdeck.com/yeseniaperezcruz/design-decisions-through-the-lens-of-a-performance-budget

• http://larahogan.me/design/


• https://www.youtube.com/watch?v=DFImM0r4EpE

• http://www.slideshare.net/bluesmoon/beyond-page-level-metrics

• http://bradfrost.com/blog/post/performance-as-design/