

1. **Setup Fiddler**
2. **Open all links in tabs**
   1. <http://localhost:50485/>
   2. <http://www.softwareishard.com/har/viewer/>
   3. <https://tinypng.com/>
   4. chrome://net-internals/#prerender
   5. <http://octane-benchmark.googlecode.com/svn/latest/index.html>
   6. chrome://tracing
3. **Open ANTS**

****

1. **Intro Mascots App**
   1. Demo all pages, ending on Texas
2. **Capture w/ Network tab on League Page**
   1. Filters
   2. Totals – load is red line, DOMContentLoaded is blue line
   3. Size & Content – size is what was transferred on the wire, content is size of actual useable asset

SKIP THIS DEMO

* 1. Time & Latency – time is total time, latency is waiting time where we weren’t getting bytes. Also the more transparent part of the timeline
  2. Select a resource, show headers and preview
  3. Show preserve log – make additional requests
  4. Copy all as HAR, display in HAR Viewer, show page timeline and hide statistics

1. **Audits & PageSpeed**
   1. Audit Texas League
   2. Explain that I have an extension, others exist
   3. Red should be fixed first, and so on
   4. PageSpeed is another extension, that is similar to Audits, but has separate features
2. **Fix**
   1. Combine/Minify - Change comments in \_Layout and show BundleConfig
   2. Compression - Change two values from false to true in web.config
   3. HTTP Caching - Uncomment caching section in web.config
   4. Sprites - Generate sprites for Texas
   5. Sprites - Rename .png.css file to .css to avoid bug
   6. Sprites – Add .css file to Layout.cshtml
   7. Sprites – League.cshtml to use class=”@mascot.Slug”
   8. Image Opt - Show Sprite, and optimize with <http://tinypng.com/>
   9. Show savings in Network tab
   10. DataUri - Use Visual Studio to change the ball to a DataUri in screen.css
3. **Show Changes to Request Count and byte size**
4. **Revert changes & Disable Fiddler**



1. **Async Script Demo**
   1. Uncomment ThirdPartyScript – these we can’t always control the placement of
   2. Reload page and notice the 3 second delay due to blocked parsing
   3. Replace with snippet
   4. Refresh



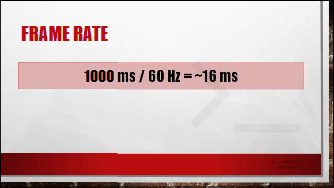
1. **Prerender Demo**
   1. Close Dev Tools!
   2. View Source on Home Page to show link tag
   3. Open chrome://net-internals/#prerender and Task Manager (Shift+Esc) to show hidden tab being used



1. **Profile Application**
   1. Stop IIS Express
   2. Cover types of Profiling Modes
   3. Use IIS Express, Chrome, and Yellow Option
   4. Allow CPU to flatten after start – mention this should be automated
   5. Click on All Leagues
   6. Stop Profiler
   7. Examine Client HTTP Requests
   8. Select the All hump and describe the hit count and time w/ children.
   9. Toggle percent and milliseconds
   10. Right click on the Call Tree and “Expand the most expensive call stack”
   11. Find the 160 queries, switch to DB view
   12. See the problem in the decompiled code.
   13. Fix the problem with a .Include(m => m.Team.League) in the All() method
   14. Re-profile and far fewer DB Queries



1. **Profile Application**
   1. Explain lack of CPU intensive JS in MiLB Mascots and browse to http://octane-benchmark.googlecode.com/svn/latest/index.html
   2. Show similarities
   3. MS or % view
   4. Show off flame chart – time across stack vertically



1. **Timeline Demo**
   1. Record All Page
   2. Show Events
   3. Show Frames
   4. Zoom In on a Few Segments



1. **Paint Demo 1**
   1. Enable “Show Paint Rectangles” & Show composited layer boarders
   2. Scroll and show red boxes + blue boxes and explain the difference
   3. Add -webkit-transform:translateZ(0) to promote layer
   4. Record in chrome://tracing and show off layers there, stress how beta it is



1. **Paint Demo 2**
   1. Enable “Continuous page repainting” while “Show paint rectangles” is on
   2. See the whole page is red
   3. Turn off show rectangles
   4. Select a .mascot-card and start to toggle them on and off to see time drop
   5. Dig in and disable box-shadow and border-radius