Practical Performance Tips and Tricks to Make Your HTML/JavaScript Faster

Doris Chen Ph.D.
Senior Developer Evangelist
Microsoft

Meet Doris Chen

- Developer Evangelist at Microsoft based in Silicon Valley, CA
  - Blog: https://blogs.msdn.microsoft.com/dorischen/
  - Video: https://channel9.msdn.com/Niners/dorischen
  - Twitter @doristchen
  - Email: doris.chen@microsoft.com
- Speaks at numerous international conferences and user groups including O'Reilly OSCON, Fluent, PHP, Dev Nexus, HTML5 Dev Conference, JavaOne, and worldwide User Groups
- Received her Ph.D. from the University of California at Los Angeles (UCLA) in computer engineering

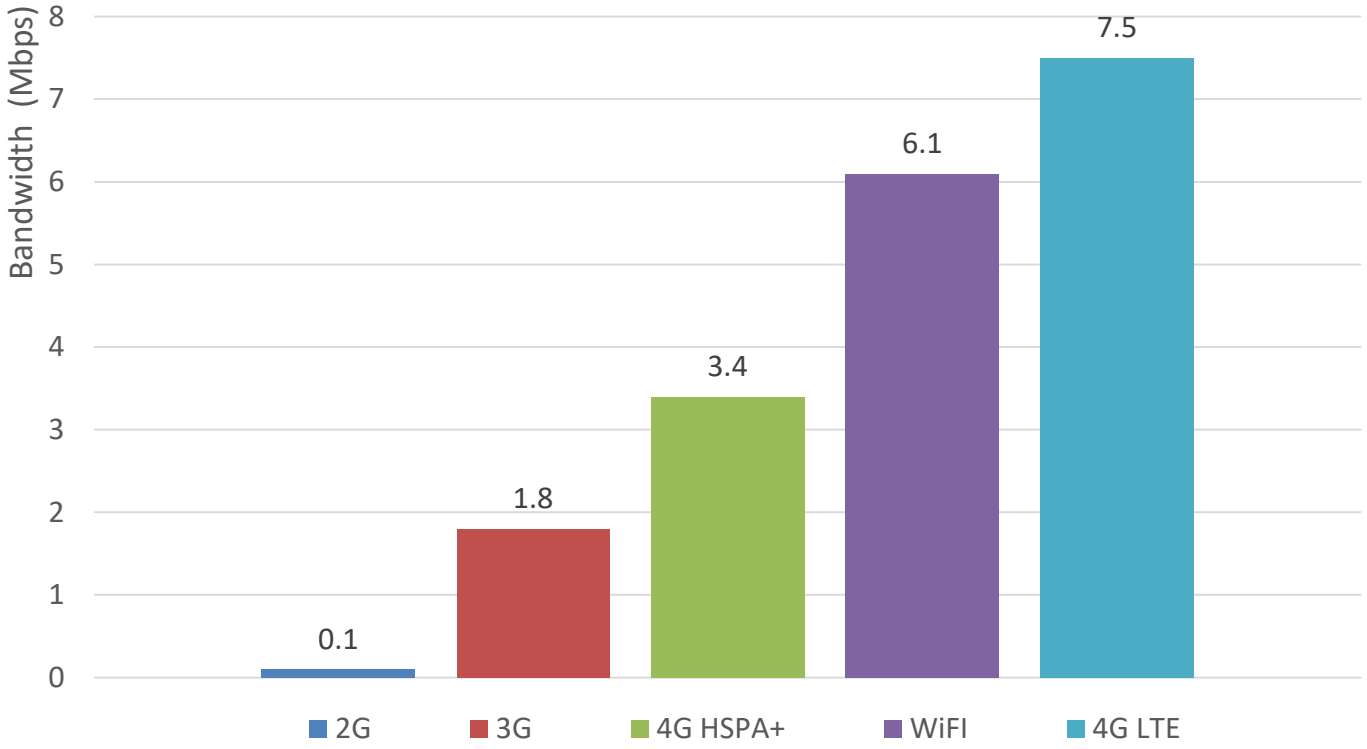
Agenda

- What to measure
- Tools
- Tips and Tricks
  - Quickly Respond to Network Requests
  - Minimize Bytes Downloaded
  - Optimize Media Usage
  - Use CSS3
  - Efficiently Structure Markup
  - Know What Your Application is Doing
  - Writing Fast JavaScript

WHAT TO MEASURE?

Network Utilization and Limits

Bandwidth, latency, and limited-data plans affect network performance



Source: OpenSignal

Power Consumption

Let it rest! Power efficiency can drain your users' battery and decrease satisfaction with your application

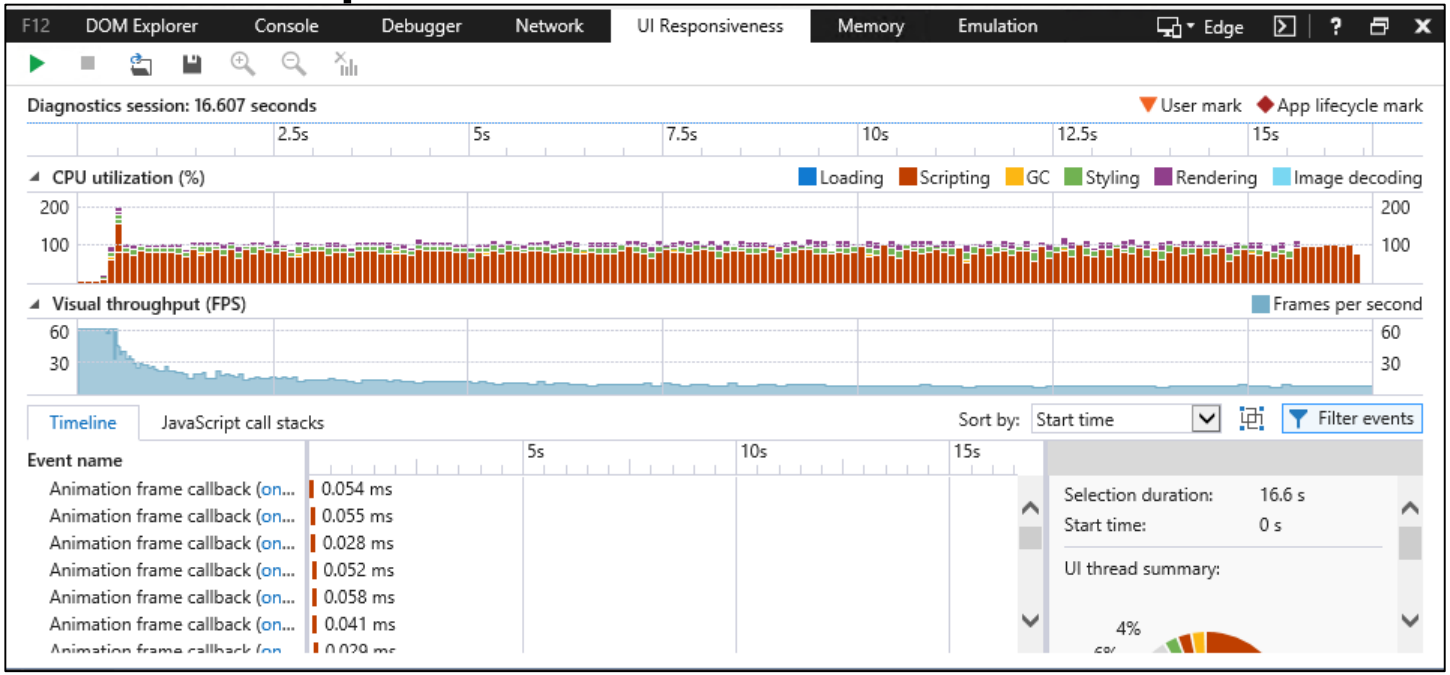
TOOLS

Tools

- F12 Tools
  - Collection of developer tools for working with DOM, Debugging, networking, JavaScript Performance, UI Responsiveness, Memory profiling, and Emulation
- Task Manager

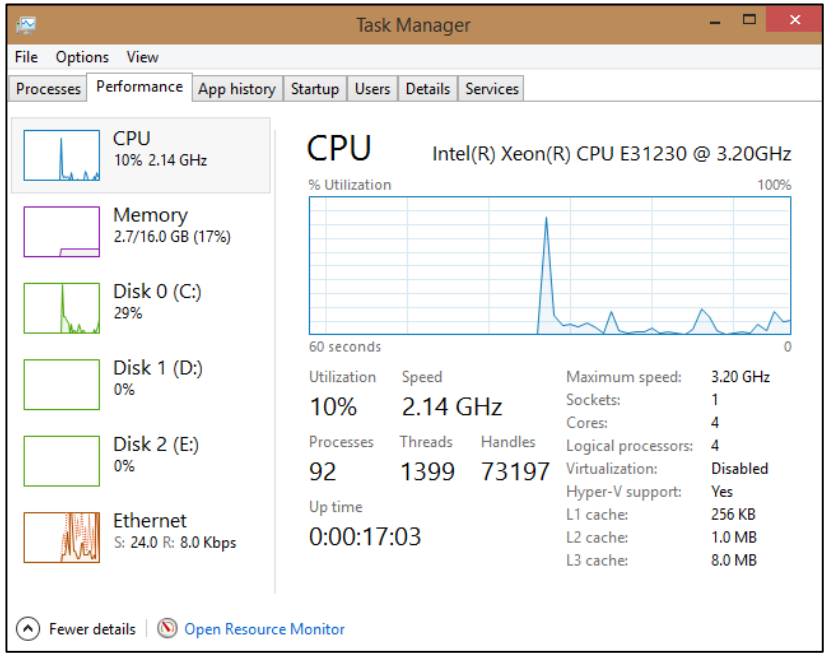
Edge F12 Developer Tools

- UI Responsiveness Tab



Old School Task Manager

- Monitoring CPU and Memory



Other Tools

- Chrome developer tools
  - https://developer.chrome.com/devtools
- Firebug Lite
  - http://getfirebug.com/firebuglite
- Fiddler
  - http://www.telerik.com/fiddler
- Web Page Test
  - http://www.webpagetest.org/
- Windows Performance Toolkit Overview
  - https://msdn.microsoft.com/en-us/library/windows/hardware/hh162981.aspx

QUICKLY RESPOND TO NETWORK REQUESTS

Avoid 3xx Redirection

Quickly Respond to Network Requests

Avoid 3xx Redirection

Quickly Respond to Network Requests

Request

GET / HTTP/1.1

Host: www.news.com

Avoid 3xx Redirection

Quickly Respond to Network Requests

Request

GET / HTTP/1.1

Host: www.news.com

Response

HTTP/1.1 303 See Other

Location: http://homepage.news.com/

Avoid 3xx Redirection

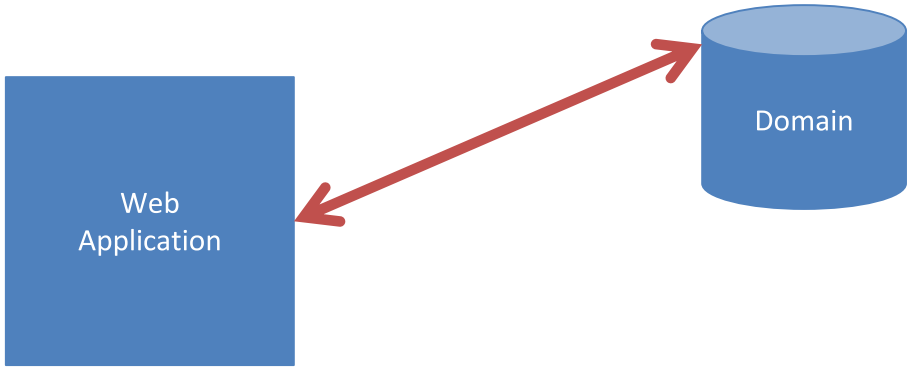
Quickly Respond to Network Requests

63%

of top 1000 websites
worldwide contain 3xx redirect

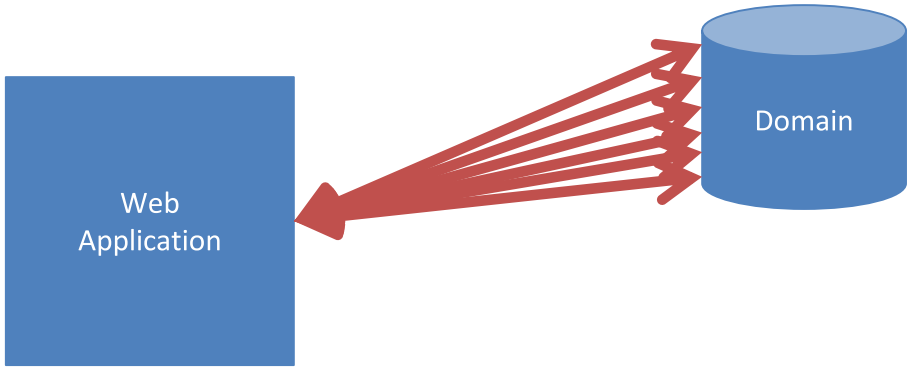
Maximize Concurrent Connections

Quickly Respond to Network Requests



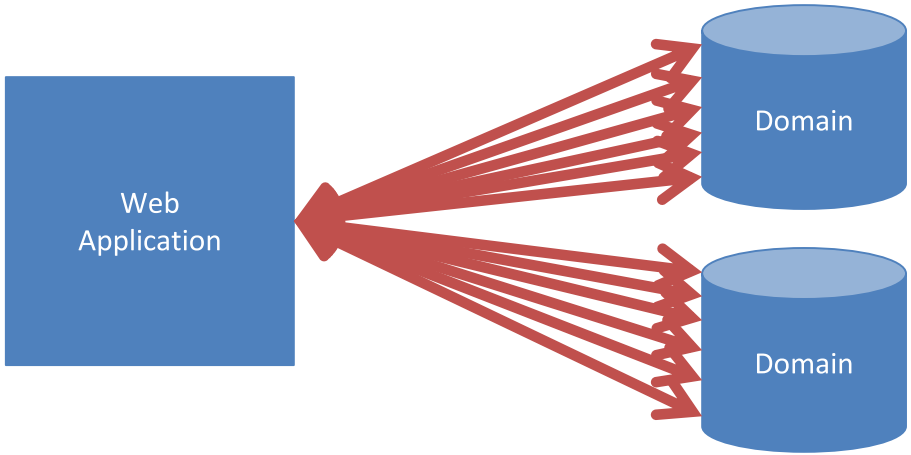
Maximize Concurrent Connections

Quickly Respond to Network Requests



Maximize Concurrent Connections

Quickly Respond to Network Requests



Reuse Connections

Quickly Respond to Network Requests

HTTP Response

HTTP/1.1 200 OK
Content-Type: application/javascript
Content-Length: 230848
Connection: close

Reuse Connections

Quickly Respond to Network Requests

HTTP Response

HTTP/1.1 200 OK

Content-Type: application/javascript

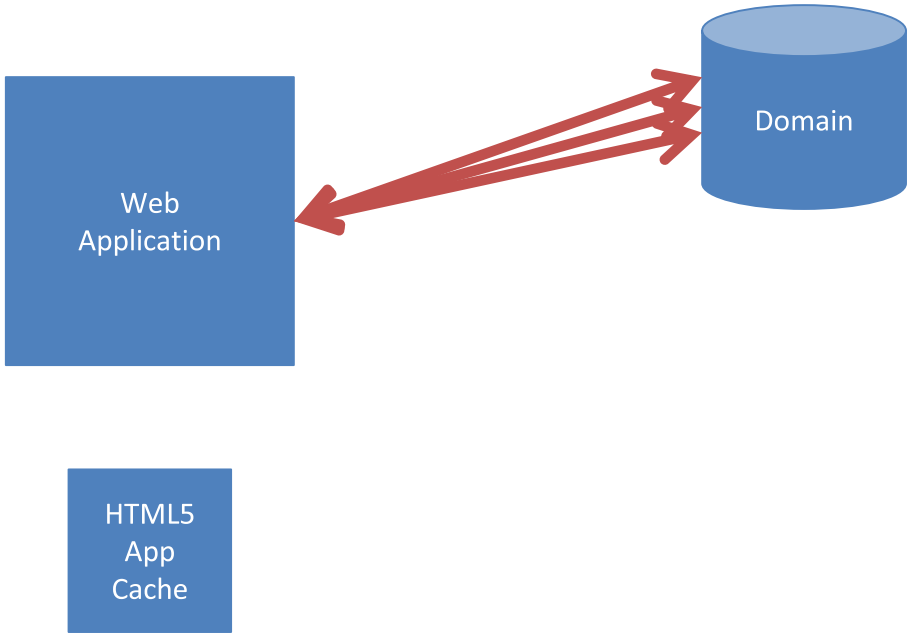
Content-Length: 230848

Connection: close

MINIMIZE BYTES DOWNLOADED

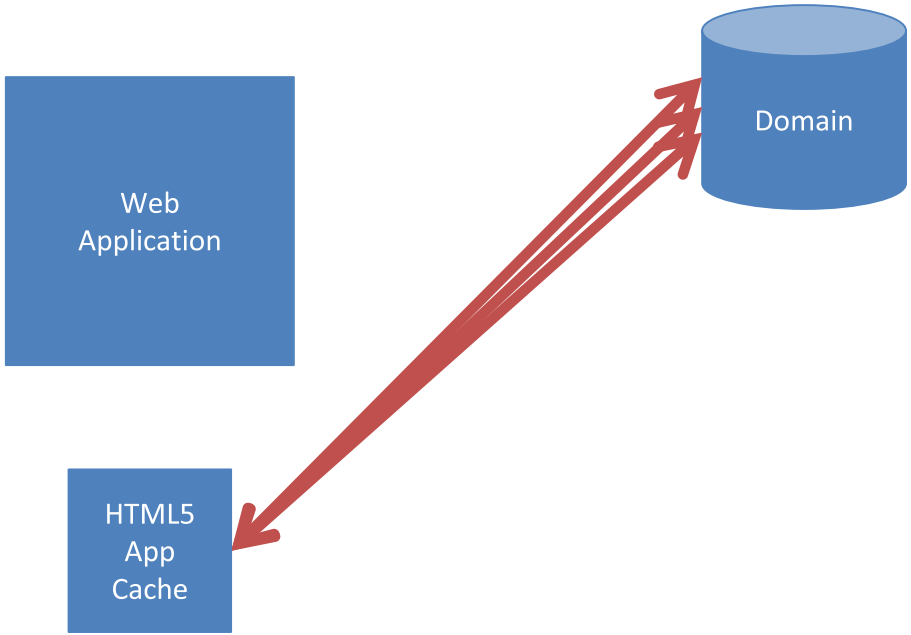
Cache Dynamic Resources in App Cache

Minimize Bytes Downloaded



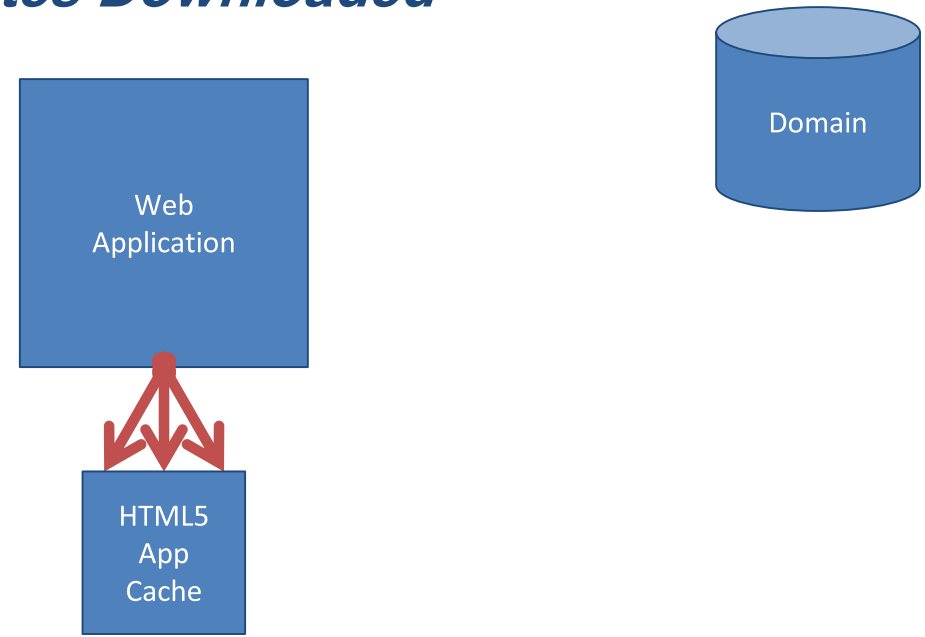
Cache Dynamic Resources in App Cache

Minimize Bytes Downloaded



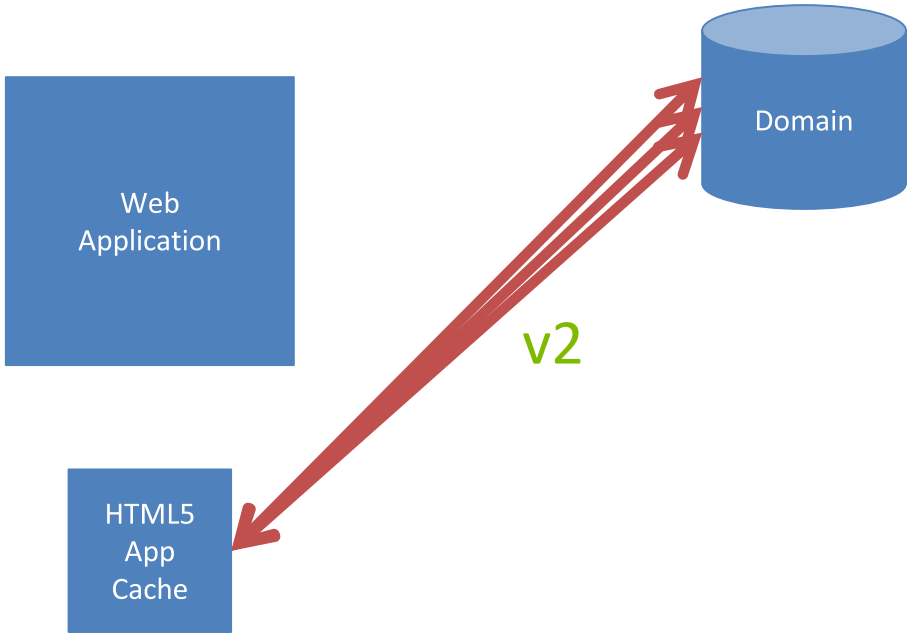
Cache Dynamic Resources in App Cache

Minimize Bytes Downloaded



Cache Dynamic Resources in App Cache

Minimize Bytes Downloaded



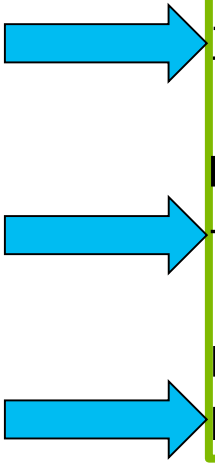
HTML5 App Cache

- Manifest-based caching to enable offline scenarios, improve performance, and reduce operating expenses
- Allows webpages to cache resources locally
- images, script libraries, style sheets
- Resynchronize files by updating the manifest
- Steps to do:
  - Create a manifest file that defines the resources you want to save
  - Reference the manifest file in each webpage designed to use cached resources

Usage of App Cache via manifest file

HTML File

```
<html manifest="test.appcache">
  <head>…</head>
  <body>
      <img src="logo.png" …>
      </img>
      …
      <video … src="fish.mp4" …>
      </video>
      …
      <img src="kid.png" … />
  </body>
</html>
```



Manifest File

```
Cache Manifest
#7/20/2011 v3
Cache:
logo.png

Network:
fish.mp4

Fallback:
kid.png noImg.png
```

MIME Type: text/cache-manifest

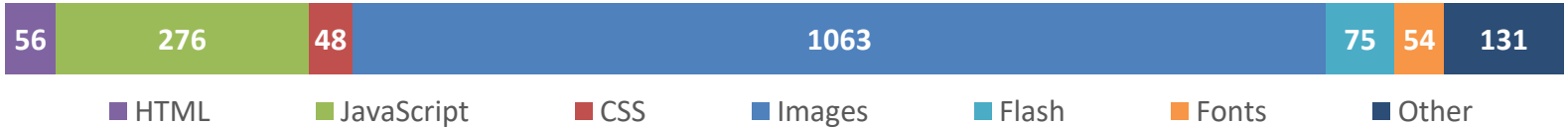
OPTIMIZE MEDIA USAGE

Average Payload

Source: http archive

- 93 resource requests, 16 Hosts

- 1.7 MB Total Size/Page, 46% Cacheable



Use Native Image Resolutions

Optimize Media Usage

```
<html>
  <head>
    <title>Test</title>
  </head>
  <body>
    …
    <!-- logo.gif dimensions: 500 x 400 -->
    <img src="logo.png" width="50" height="40" />
    …
  </body>
</html>
```

Use Native Image Resolutions

Optimize Media Usage

```
<html>
  <head>
    <title>Test</title>
  </head>
  <body>
    …
    <!-- logo.gif dimensions: 500 x 400 -->
    <img src="logo.png" width="50" height="40" />
    …
  </body>
</html>
```

Use Native Image Resolutions

Optimize Media Usage

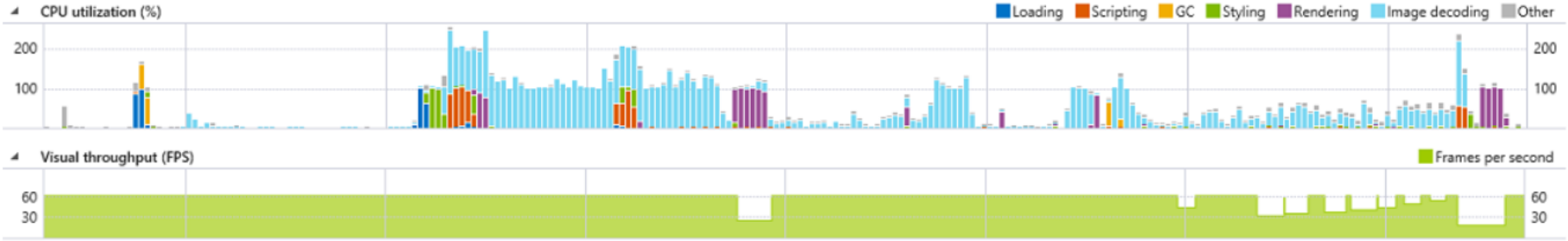
```
<html>
  <head>
    <title>Test</title>
  </head>
  <body>
    …
    <!-- logo.gif dimensions: 500 x 400 -->
    <img src="logo.png" width="50" height="40" />
    …
  </body>
</html>
```

Test Site

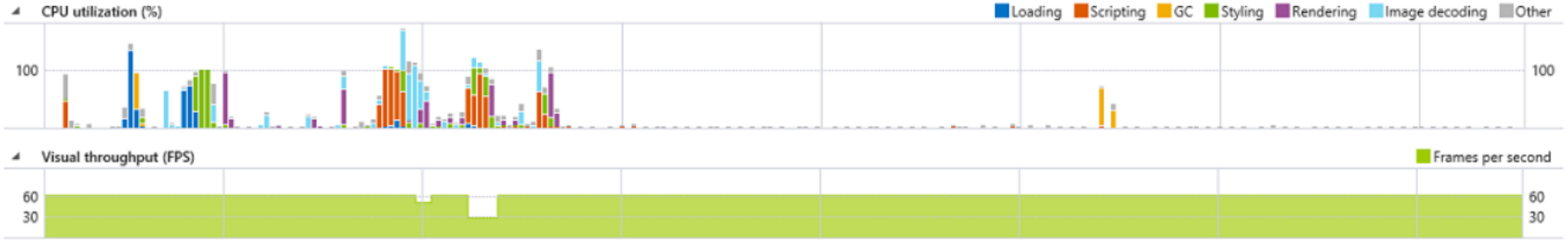
Use Native Image Resolutions

Right-size your images

Scaled: 1,850 milliseconds



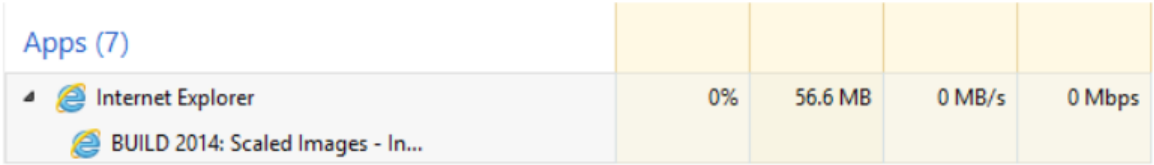
Right-sized: 625 milliseconds



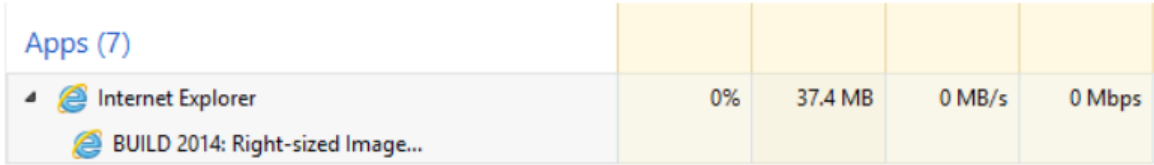
Use Native Image Resolutions

Right-size your images (again!)

Scaled: 66 MB



Right-sized: 37.4 MB



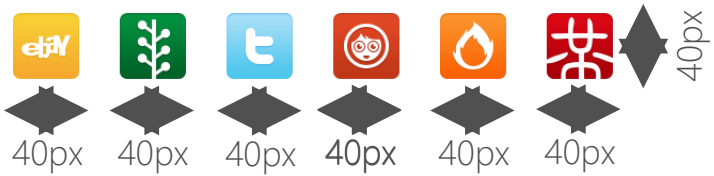
Avoid Death by 1,000 Images

Optimize Media Usage

Use Image Sprites

Optimize Media Usage

Scenario #1 – Multiple Files



6 Images
6 Connections
96k Download

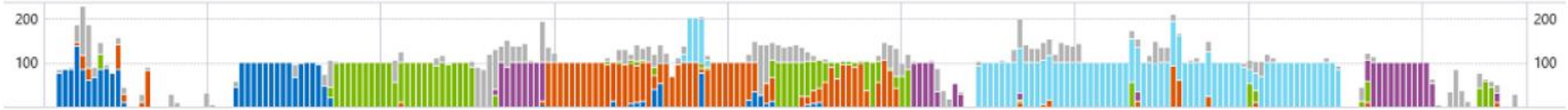
Scenario #2 - Image Sprite

1 Image
1 Connection
21k Download

Reduce Number of downloads

Reduce the number of downloads

Uncobmined: 210 milliseconds



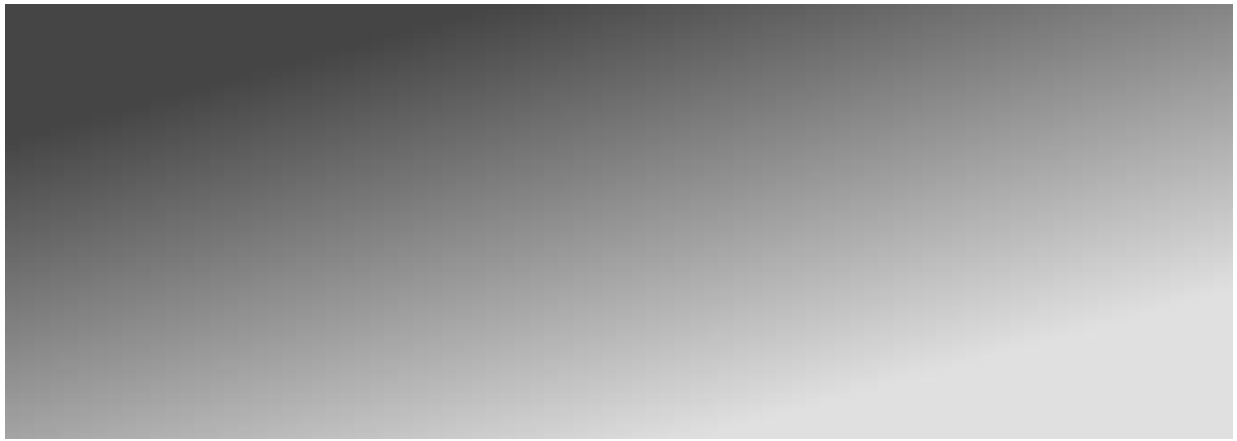
Combined: 164 milliseconds

Use CSS3

USE CSS3

Replace Images with CSS3 Gradients

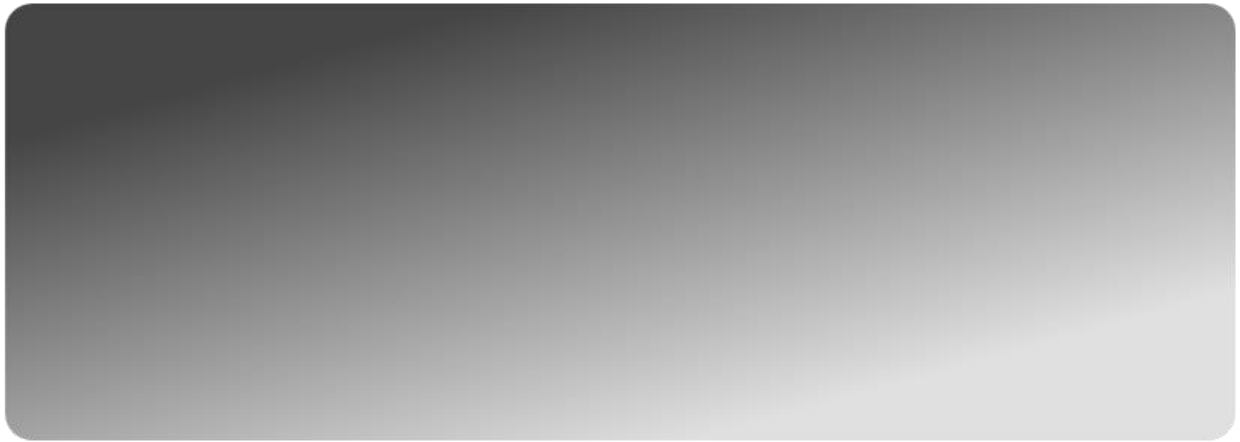
Optimize Media Usage



-ms-gradient(linear, 50% 50%, 0% 34%, from(#666666), to(#666666), color-stop(.3,#333333))
-webkit-gradient(linear, 50% 50%, 0% 34%, from(#666666), to(#666666), color-stop(.3,#333333))
-moz-gradient(linear, 50% 50%, 0% 34%, from(#666666), to(#666666), color-stop(.3,#333333))
gradient(linear, 50% 50%, 0% 34%, from(#666666), to(#666666), color-stop(.3,#333333))

Replace Images with CSS3 Border Radius

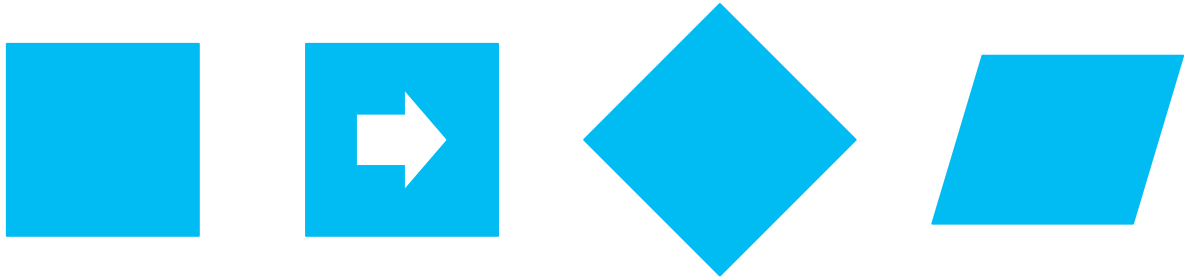
Optimize Media Usage



```
border-radius:100px;
-ms-border-radius: 100px;
-webkit-border-radius: 100px;
-moz-border-radius: 100px;
```

Leverage CSS3 Transforms

Optimize Media Usage

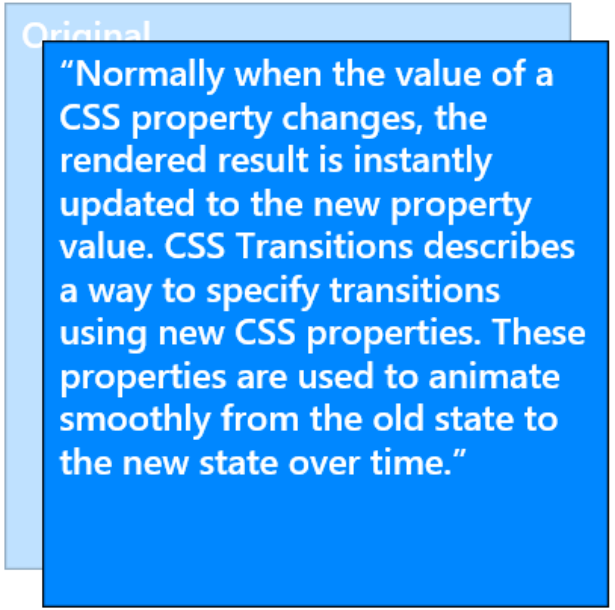


Element Move Rotate Skew

```
-ms-transform: translate(75px, 100px) scaleY(.75) skewY(10deg);
-webkit-transform: translate(75px, 100px) scaleY(.75) skewY(10deg);
-moz-transform: translate(75px, 100px) scaleY(.75) skewY(10deg);
-o-transform: translate(75px, 100px) scaleY(.75) skewY(10deg);
transform: translate(75px, 100px) scaleY(.75) skewY(10deg);
```

Leverage CSS Transitions

-ms-transition: all 1s ease-in-out;
-webkit-transition: all 1s ease-in-out;
-moz-transition: all 1s ease-in-out;
-o-transition: all 1s ease-in-out;
transition: all 1s ease-in-out;



CSS Animation

```
@keyframes demo {
  from {
    animation-timing-function: ease;
  }
  50% {
    animation-timing-function: ease;
  }
  to {
    animation-timing-function: ease;
  }
}
```

"CSS Animations allow an author to modify CSS property values over time. In a simple transition, a single timing function and duration determine the intermediate values of the animating property. When finer control of the intermediate values is required, keyframes can be used. Keyframes are specified using a specialized CSS at-rule."

EFFICIENTLY STRUCTURE MARKUP

Link Style Sheets at Top of Page

Efficiently Structure Markup

```
<html>
   <head>
      <title>Test</title>
      <link rel="stylesheet" type="text/css" href="class.css"
 />
   </head>
   <body>
      …
      …
      …
   </body>
</html>
```

Only Include Necessary Styles

Efficiently Structure Markup

```
/* These styles are for the home page. */
HomePage
{
  color: red;
}

/* These styles are for the content page. */
ContentPage
{
  color: green;
}
```

Only Include Necessary Styles

Efficiently Structure Markup

```
/* These styles are for the home page. */
HomePage
{
  color: red;
}

/* These styles are for the content page. */
ContentPage
{
  color: green;
}
```

Only Include Necessary Styles

Efficiently Structure Markup

```
/* These styles are for the home page. */
HomePage
{
  color: red;
}

/* These styles are for the content page. */
ContentPage
{
  color: green;
}
```

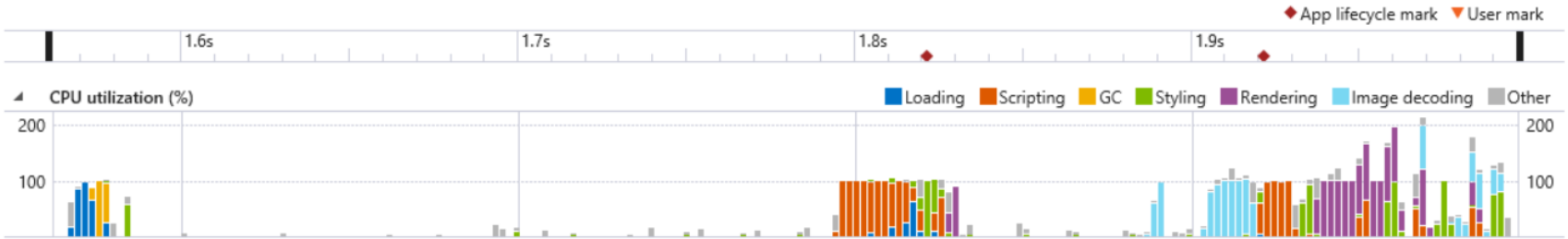
Always Link JavaScript at End of File

Efficiently Structure Markup

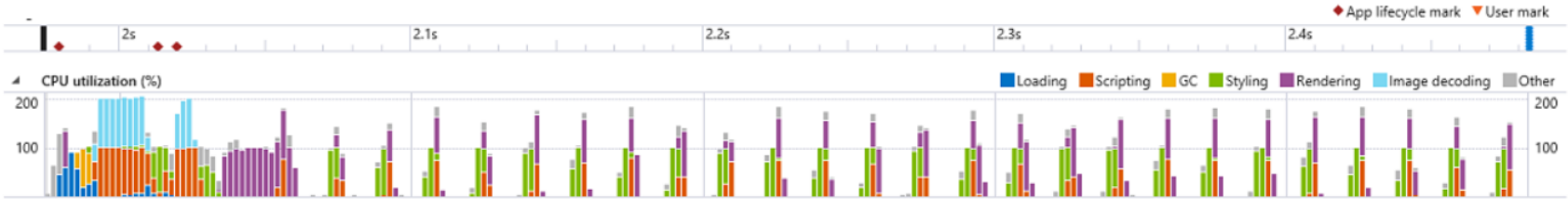
```
<html>
   <head>
      <title>Test</title>
   </head>
   <body>
      …
      …
      …
      <script src="myscript.js" … ></script>
   </body>
</html>
```

Defer script execution

Scripts in Header: 438 milliseconds



Scripts at end of document: 84 milliseconds



Remove Duplicate Code

Efficiently Structure Markup

```
<html>
   <head>
      <title>Test</title>
   </head>
   <body>
      …
      <script src="jquery.js" … ></script>
      <script src="myscript.js" … ></script>
      <script src="navigation.js" … ></script>
      <script src="jquery.js" … ></script>
   </body>
</html>
```

Remove Duplicate Code

Efficiently Structure Markup

```
<html>
   <head>
      <title>Test</title>
   </head>
   <body>
      …
      <script src="jquery.js" … ></script>
      <script src="myscript.js" … ></script>
      <script src="navigation.js" … ></script>
      <script src="jquery.js" … ></script>
   </body>
</html>
```

Remove Duplicate Code

Efficiently Structure Markup

52%

of the pages on the web
have duplicate code

Standardize on a Single Framework

Efficiently Structure Markup

```
<script src="jquery.js" … ></script>
<script src="prototype.js" … ></script>
<script src="dojo.js" … ></script>
<script src="animater.js" … ></script>
<script src="extjs.js" … ></script>
<script src="yahooui.js" … ></script>
<script src="mochikit.js" … ></script>
<script src="lightbox.js" … ></script>
<script src="jslibs.js" … ></script>
<script src="gsel.js" … ></script>
…
```

Don’t Include Script To Be Cool

Efficiently Structure Markup

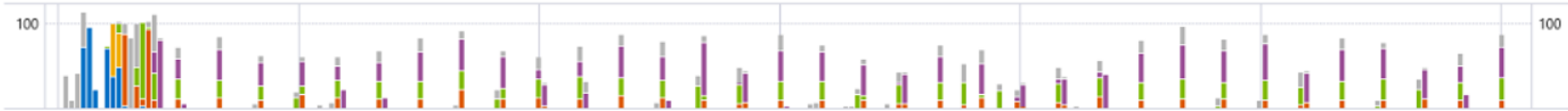
```
<script src="facebookconnect.js" … ></script>
<script src="facebooklike.js" … ></script>
<script src="facebookstats.js" … ></script>
<script src="tweetmeme.js" … ></script>
<script src="tweeter.js" … ></script>
<script src="tweetingly.js" … ></script>
<script src="googleanalytics.js" … ></script>
<script src="doubleclick.js" … ></script>
<script src="monitor.js" … ></script>
<script src="digg.js" … ></script>
…
```

Reduce Number of Frameworks

Reduce the number of frameworks

Many Frameworks: 617 milliseconds

Single Framework: 35 milliseconds



KNOW WHAT YOUR APPLICATION IS DOING

Understand JavaScript Timers

Know What Your Application is Doing

setTimeout

```
setTimeout(callback, ms);
```

setInterval

```
var test = setInterval(callback, ms);
clearInterval(test);
```

Paint as much as your users can see

Align timers to display frames

```
setInterval(animate, 0);
setTimeout(animate,  0);
```

MORE WORK

```
requestAnimationFrame(animate);

setInterval(animate, 1000 / 60);
setTimeout(animate,  1000 / 60);
```

LESS WORK

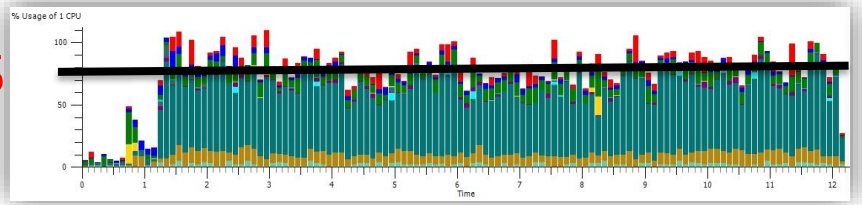
Results

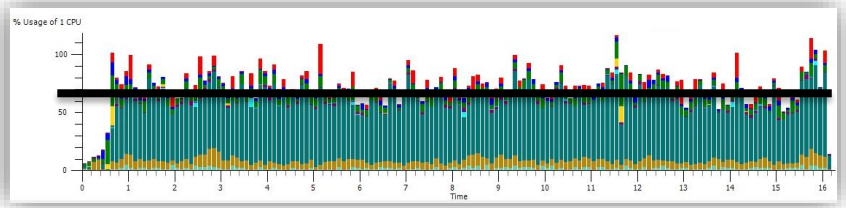
Save CPU cycles

```
setTimeout(animate,  0);
```

```
requestAnimationFrame(animate);
```

75 %





65 %

JSON Always Faster than XML for Data

WRITE FAST JAVASCRIPT

JSON Always Faster than XML for Data

XML Representation

```
<!DOCTYPE glossary PUBLIC "DocBook V3.1">
   <glossary><title>example glossary</title>
      <GlossDiv><title>S</title>
         <GlossList>
          <GlossEntry ID="SGML" SortAs="SGML">
             <GlossTerm>Markup Language</GlossTerm>
             <Acronym>SGML</Acronym>
             <Abbrev>ISO 8879:1986</Abbrev>
             <GlossDef>
                <para>meta-markup language</para>
                <GlossSeeAlso OtherTerm="GML">
                <GlossSeeAlso OtherTerm="XML">
             </GlossDef>
          <GlossSee OtherTerm="markup">
       </GlossEntry>
    </GlossList>
   </GlossDiv>
</glossary>
```

JSON Representation

```
"glossary":{
    "title": "example glossary", "GlossDiv":{
        "title": "S", "GlossList": {
            "GlossEntry": {
                "ID": "SGML",
                "SortAs": "SGML",
                "GlossTerm": "Markup Language",
                "Acronym": "SGML",
                "Abbrev": "ISO 8879:1986",
                "GlossDef": {
                    "para": "meta-markup language",
                    "GlossSeeAlso": ["GML", "XML"] },
                "GlossSee": "markup" }
            }
        }
    }
```

Use Built-in JSON Methods

Write Fast JavaScript

JSON Methods

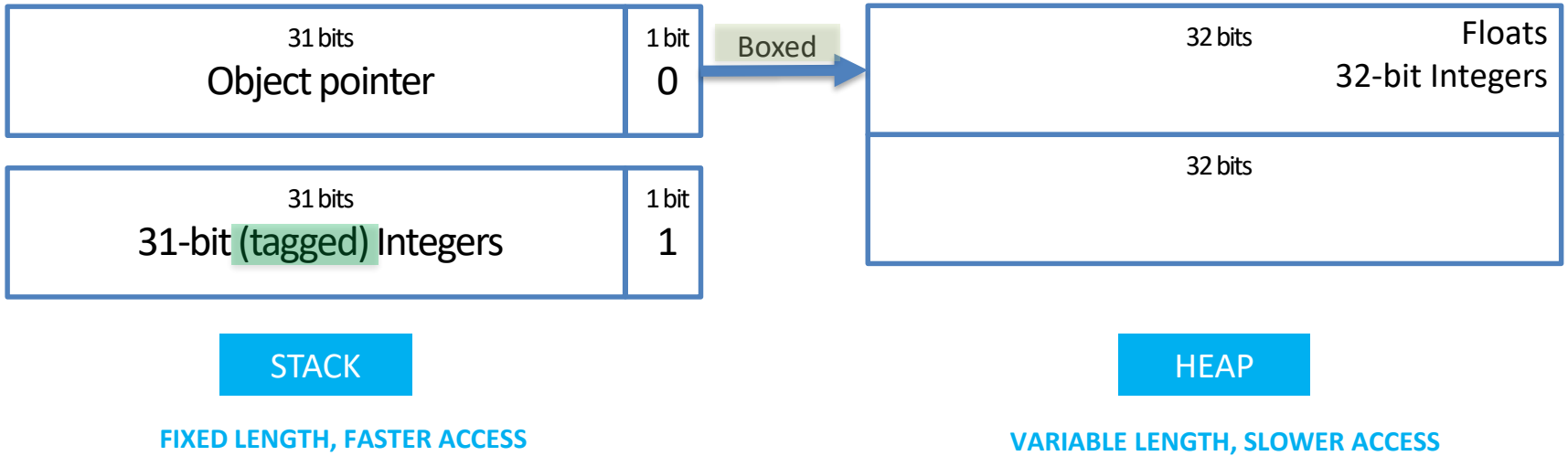
```
var jsObjStringParsed = JSON.parse(jsObjString);
var jsObjStringBack = JSON.stringify(jsObjStringParsed);
```

Numbers in JavaScript

WRITE FAST JAVASCRIPT

Numbers in JavaScript

- All numbers are IEEE 64-bit floating point numbers
  - Great for flexibility
  - Performance and optimization challenge



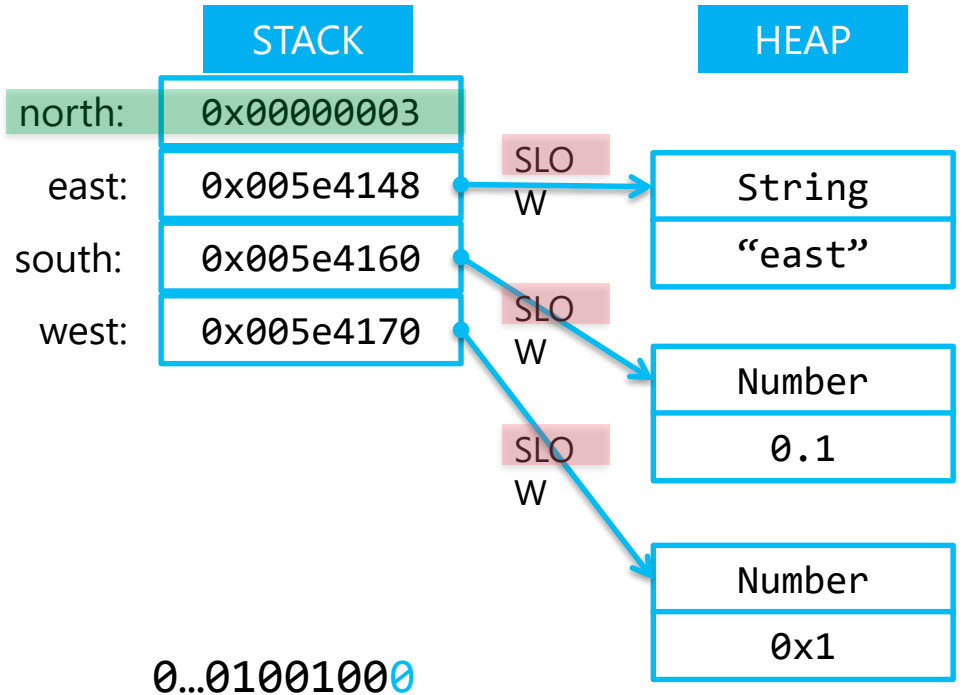
Use 31-bit integers for faster arithmetic

```
var north = 1;

var east = "east";
var south = 0.1;
var west = 0x1;


function Player(north, south, east, west)
{
    ...
}

var p = new Player(north,south,east,west);
```




```
0x005e4148:          0…01001000
0x03 represents 1:   0…00000011
```

Avoid creating floats if they are not needed

Fastest way to indicate integer math is |0

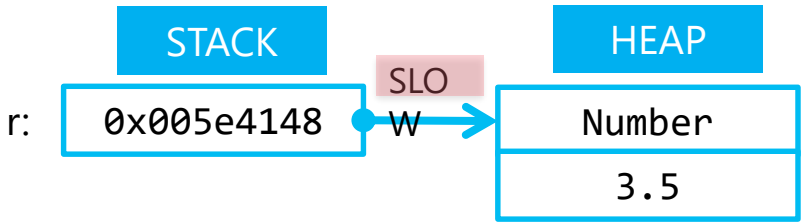
```
var r = 0;

function doMath(){
  var a = 5;
  var b = 2;
  r = ((a + b) / 2);                    // r = 3.5
}
...
var intR = Math.floor(r);
```

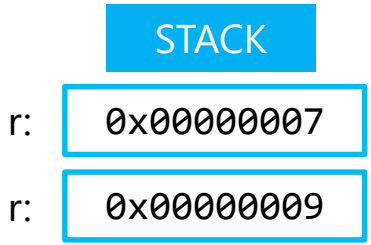
```
var r = 0;

function doMath(){
  var a = 5;
  var b = 2;
  r = ((a + b) / 2) |0 ;                // r = 3
  r = Math.round((a + b) / 2);          // r = 4
}
```

SLOW



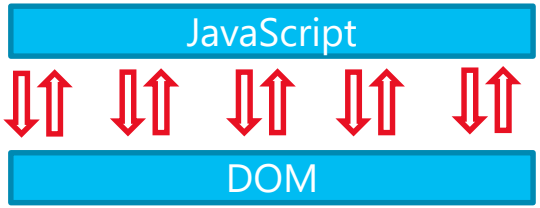
FAST



Working with DOM
WRITE FAST JAVASCRIPT

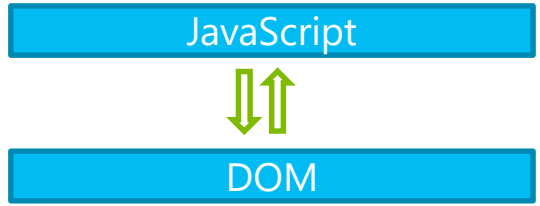
Avoid chattiness with the DOM

```
...
//for each rotation
document.getElementById(elID).classList.remove(oldClass)
document.getElementById(elID).classList.add(newClass)
...
```




```
var element = document.getElementById(elID).classList;

//for each rotation
element.remove(oldClass)
element.add(newClass)
...
```



Avoid automatic conversions of DOM values

Values from DOM are strings by default

```
this.boardSize = document.getElementById("benchmarkBox").value;

for (var i = 0; i < this.boardSize; i++) {
  //this.boardSize is “25”
  for (var j = 0; j < this.boardSize; j++) {
    //this.boardSize is “25”
    ...
  }
}
```

SLOW

```
this.boardSize = parseInt(document.getElementById("benchmarkBox").value);

for (var i = 0; i < this.boardSize; i++) {
  //this.boardSize is 25
  for (var j = 0; j < this.boardSize; j++) {
    //this.boardSize is 25
    ...
  }
}
```

FAST
(25% marshalling cost reduction in init function)

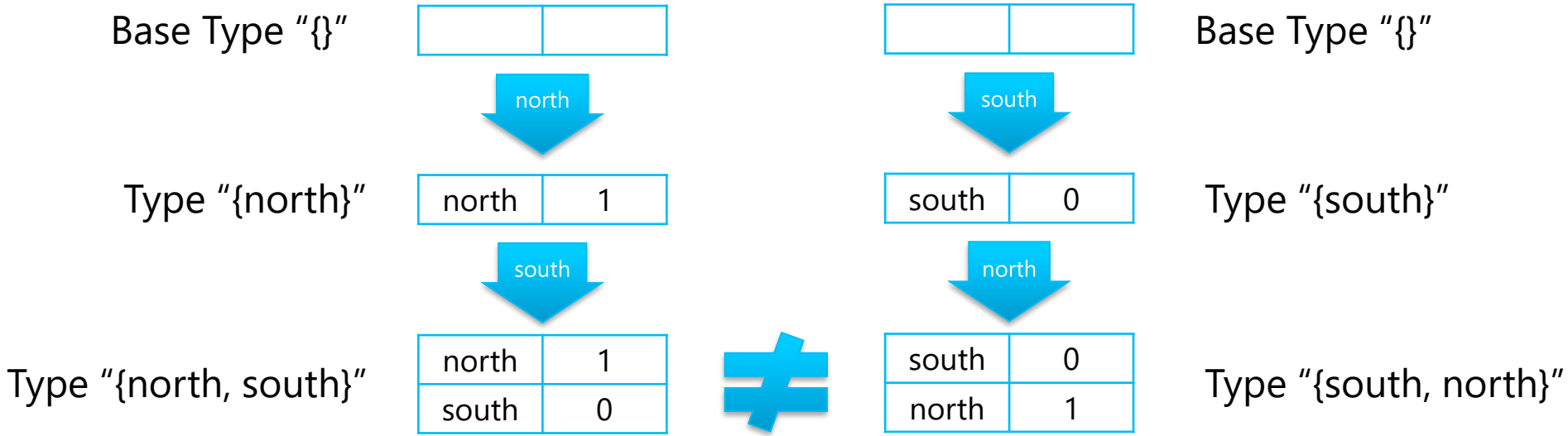
Use fast objects and manipulations

WRITE FAST JAVASCRIPT

Internal Type System: Fast Object Types

```
var p1;
p1.north = 1;
p1.south = 0;
```

```
var p2;
p2.south = 0;
p2.north = 1;
```



Create fast types and avoid type mismatches

Don't add properties conditionally

```
function Player(direction) {
    if (direction == "NE") {
        this.n = 1;
        this.e = 1;
    }
    else if (direction == "ES") {
        this.e = 1;
        this.s = 1;
    }
    ...
}

var p1 = new Player("NE");      // p1 type
{n,e}
var p2 = new Player("ES");      // p2 type
{e,s}
```

p1.type != p2.type

```
function Player(north,east,south,west) {
    this.n = north;
    this.e = east;
    this.s = south;
    this.w = west;
}

var p1 = new Player(1,1,0,0);//p1 type
{n,e,s,w}
var p2 = new Player(0,0,1,1);//p2 type
{n,e,s,w}
```

p1.type == p2.type

Avoid conversion from fast type to slower property bags

Deleting properties forces conversion

```
function Player(north,east,south,west)
{
  this.n = north;
  this.e = east;
  this.s = south;
  this.w = west;
}
var p1 = new Player();

delete p1.n;
```

SLOW

```
function Player(north,east,south,west)
{
  this.n = north;
  this.e = east;
  this.s = south;
  this.w = west;
}
var p1 = new Player();

p1.n = 0;              // or undefined
```

FAST

Avoid creating slower property bags

Restrict using getters, setters and property descriptors in perf critical paths

```
function Player(north, east, south, west) {
    Object.defineProperty(this, "n", {
        get : function() { return nVal; },
        set : function(value) { nVal=value;
},
        enumerable: true, configurable: true
    });
    Object.defineProperty(this, "e", {
        get : function() { return eVal; },
        set : function(value) { eVal=value;
},
        enumerable: true, configurable: true
    });
    ...
}
var p = new Player(1,1,0,0);
var n = p.n;
p.n = 0;
...
```

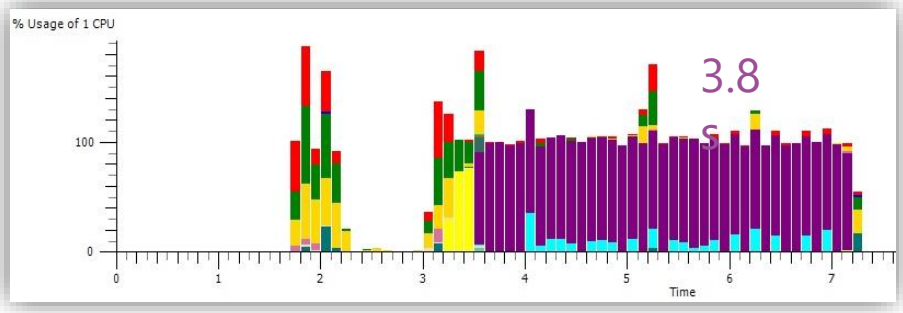
SLOW

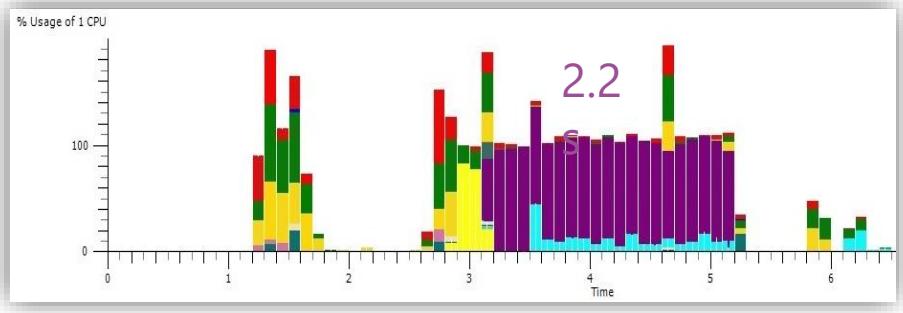
```
function Player(north, east, south, west) {
  this.n = north;
  this.e = east;
  this.s = south;
  this.w = west;
  ...
}

var p = new Player(1,1,0,0);
var n = p.n;
p.n = 0;
...
```

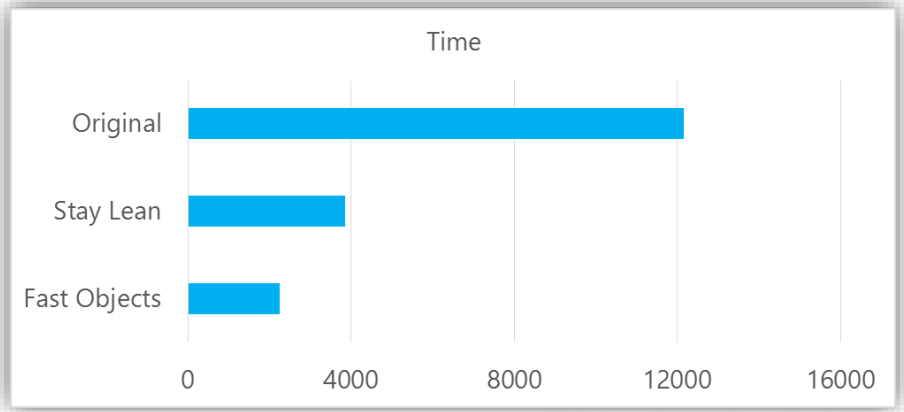
FAST

Results





- Time in script execution reduced ~30%
- Raw JS performance improved ~30%



Video: https://channel9.msdn.com/Series/htmlperf
Demo: https://github.com/doristchen/HTMLJS_Perf

Practical Performance Tips to Make Your HTML/JavaScript Faster	
01 | Web Performance 101	05 | Strategies and Principles: Memory, Markup, Execution
02 | Tools and Measurement	06 | Write Fast JavaScript
03 | Strategies and Principles: Network Requests, Bytes Downloaded	07 | Case Study: Casual Game Performance Tuning
04 | Strategies and Principles: Media Usage	

Resources

Overview Concepts

- High Performance Websites
  Steve Souders, September 2007
- Event Faster Websites: Best Practices
  Steve Souders, June 2009
- High Performance Browser Networking
  Ilya Grigorik, September 2013

JavaScript Patterns

- High Performance JavaScript
  Nicholas Zakas, March 2010
- JavaScript the Good Parts
  Douglas Crockford, May 2008
- JavaScript Patterns
  Stoyan Stefanov, September 2010
- JavaScript Cookbook
  Shelley Powers, July 2010

Microsoft Guidance

- Windows Store App: JavaScript Best Practices
- Internet Explorer Architectural Overview

W3C Web Performance

- Web Performance Working Group Homepage
- Navigation Timing Specification

Blog Posts

- Key Advances to JavaScript Performance in Windows 10
- Measuring Browser Performance in Lab Environments
- What Common Benchmarks Measure

Tools

- Debugging/Tuning Browser Performance with the Windows Performance Tools
- How to use F12 tool