Ajax Performance Tuning and Best Practice

Greg Murray  Doris Chen Ph.D.

Netflix  Sun Microsystems, Inc.
Senior UI Engineer  Staff Engineer
Agenda

> Optimization Strategies and Process
> General Coding Best Practices
> Measuring Performance
> Performance Optimization Recommendations
> Frameworks and Case Study
> Summary and Resources
Optimization Strategies and Process
Performance Challenges

> Not much control over the deployment environment
  
  • Load balancers, choice of application servers, server configuration

> Browser incompatibilities

> Large development teams

> Complex projects dependencies

> Application spans client and server

> Hard to diagnose and debug
1. Analysis
   - Framework

3. Coding
   - Best Practices
   - Rules

4. Performance Tests
   - Single User

5. Use Tools
   - Find

6. Tuning / Refactor
   - Best practices
   - Guidelines / Rules
   - Better Algorithms
   - Creativity

7. Tests
   - Multiple Users Tests

8. QA

9. Ship it

No

Yes

Performance Ok

Performance Ok

Yes

Ok

Yes
General Coding Best Practices
JavaScript Best Practices

> Provide a clean separation of content, CSS, and JavaScript
> De-reference unused objects
> Think Asynchronous
> Working with Objects
Separation of content, CSS, and JavaScript

> A rich web application user interface is made up of
  • content (HTML/XHTML)
  • styles (CSS)
  • JavaScript
> Place CSS and JavaScript code in separate files
  • Optimize the bandwidth usage by having CSS and JavaScript file loaded only once
> Variables to decide using external or inline
  • multiple page views per user per session
  • many of pages re-use the same scripts and stylesheets
Inline JavaScript and CSS in front page, but dynamically download the external files after the page has finished loading

```
<style>#Styles</style>

<script type="text/javascript">
// JavaScript logic
<script>
<body>The quick brown fox...</body>
```

For multiple views per user, and reusable scripts/css:

```
<link rel="stylesheet" type="text/css" href="cart.css">

<body>The quick brown fox...</body>

<script type="text/javascript" src="cart.js">
```
De-reference unused objects

//delete objects
var foo='Delete Me'; //do something with foo
delete foo;

// detach listeners (IE / W3C difference)
someElement.removeEventListener(type, fn, false); // W3C
someElement.detachEvent(type,fn); // IE

// remove DOM elements
someElement.parentNode.removeChild(someElement);

// page onunload
window.onunload= function() { /*do something*/}
Think Asynchronous

> Prevents browsers from halting execution of a code block that might take a long time
> Semi-Asynchronous – Like an Ajax request
  • Use a callback
> use setTimeout() in conjunction with a callback

```javascript
function longProcess({ name : value}, callback) {
  // do what needs to be done
  callback.apply({});
}

setTimeout(function() {
  longProcess({ name : 'greg'}, function() {
    alert('all done');
  }, 0);
}, 0);
```
Working with Objects (I)

```javascript
var i;
for (i = 0; i < divs.length; i += 1) {
    divs[i].style.color = "black";
    divs[i].style.border = thickness + 'px solid blue';
    divs[i].style.backgroundColor = "white";
}
```

---

```javascript
var border = thickness + 'px solid blue';
var nrDivs = divs.length;
var ds, i;
for (i = 0; i < nrDivs; i += 1) {
    ds = divs[i].style;
    ds.color = "black";
    ds.border = border;
    ds.backgroundColor = "white";
}
```

Good

Bad
window.tablebuilder = function() {
    var _header, _rows = [];
    this.setHeader = function(_headers) {
        header = "<tr><th>" +
            _headers.join("</th><th>" ) + "</tr>");
    };
    this.addRow = function(_cells) {
        rows.push("<tr><td>" +
            _cells.join("</td><td>" ) + "</td></tr>");
    };
    this.toString = function() {
        return "<table>" + _header +
            "<tbody>" + _rows.join("" ) + "</tbody>" +
            "</table>";
    };
};
Demo
Measuring Performance
function myFunctionToTest() {
    var start = new Date().getTime();
    ... // body of the function
    var totalTime = new Date().getTime() - start;
}
Tool: Firebug and YSlow

Grade C
Overall performance score 72
Ruleset applied: Classic(V1) URL: http://video.google.com/?...
YSlow

- Performance lint tool
  - Analyzes and measures web page performance
    - All components on the page, including components dynamically created by JavaScript
  - Offers suggestion for improvement
- Scores web pages for each rule
- Firefox add-on integrated with Firebug
- Open source license
- http://developer.yahoo.com/yslow
Demo
Performance Optimization Recommendations
Recommendations

> Make fewer HTTP requests
> Put stylesheets at the top
> Move scripts to the bottom
> Minify JS
> Defer Loading Resources
> Add an Expires header
> Configure Etags
> Gzip Resources
Make fewer HTTP requests (content)

> Simply page design
> CSS sprites (best for components)
  - http://alistapart.com/articles/sprites
> Combined scripts, combined stylesheets
> combining six scripts into one eliminates five HTTP requests
> Make JS and CSS external
Put stylesheets at the top (css)

> Want the browser to display whatever content it has as soon as possible
  * Avoids flash of unstyled content or blank white screen

> CSS at the bottom:
  * prohibits progressive rendering in many browsers, including Internet Explorer
  * browsers block rendering to avoid having to redraw elements of the page if their styles change

> Solution: put stylesheets in document head
  * allows the page to render progressively
Move scripts to the bottom (javascript)

- Scripts block parallel downloads across all hostnames

- Scripts block rendering of everything below them in the page

- Script *defer* attribute is not a solution
  - IE only, not supported in other browsers
Minify JavaScript (javascript, css)

> Minification is the practice of:
> • removing comments
> • removing white space characters (space, newline, and tab)
> • condensing variable names (optional)

> Popular tools for minifying JavaScript code
> • JSMin [http://crockford.com/javascript/jsmin](http://crockford.com/javascript/jsmin)
> • YUI Compressor: can also minify CSS
> • Dean Edwards Packer

> Minifying both external and inlined scripts and styles
> • minifying will still reduce the size by 5% or more even after you gzip
Defer Loading Resources

> If you have a large library or set of libraries, you don't need to load everything when a page is loaded

> Resources can be html, scripts, or css
  - Use Ajax Request
  - Add script/css links using DOM
Example: Defer Loading Resources

```html
<script>
var deferredLibs = [ '/resources/jquery.js', '/resources/base.js'];
addLibraries(libs : deferredLibs,
    function(args) { // initialize components
    });
</script>
```
function addLibraries( libs, callback) {
    for(var i=0; i < libs.length; i+=1) {
        var head =
        document.getElementsByTagName("head")[0];
        var s = document.createElement("script");
        // add callback tracking logic
        s.src = libs[i];
        head.appendChild(s);
    }
}
Add an Expires header (server)

- **Expires** response header tells the client how long a component can be cached
  - Browsers use a cache to reduce the number and size of HTTP requests
  - Avoids unnecessary subsequent HTTP requests
  - Expires headers most often used with images, but should be used for scripts, stylesheets, and Flash
  - For static components: set a far future **Expires** header
  - For dynamic components: use an appropriate **Cache-Control** header to help the browser with conditional requests
How to Implement

> May configure on the server conf file
> Implement in a servlet or a servlet filter

```java
long maxAge = 1209600L;
SimpleDateFormat sdf =
    new SimpleDateFormat("EEE, d MMM yyyy HH:mm:ss z");
long created = System.currentTimeMillis();
Date expiredate = new Date(created + (maxAge* 1000) );
sdf.setTimeZone (TimeZone.getTimeZone("GMT"));
String expires = sdf.format(expiredate);
--------------------------------------------
resp.setHeader("Expires", expires);
resp.setHeader("Cache-Control", "public,max-age=" + maxAge);
```
Configure Etags (server)

> Etags: validate entities, unique identifier returned in response
   ETag: "10c24bc-4ab-457e1c1f"

> Used in conditional GET requests
   GET /i/yahoo.gif HTTP/1.1
   Host: us.yimg.com
   If-Modified-Since: Tue, 12 Dec 2008 03:03:59 GMT
   If-None-Match: "10c24bc-4ab-457e1c1f"
   HTTP/1.1 304 Not Modified

> If ETag doesn't match, can't send 304
Etag Implementation

```java
String etag = cr.getContentHash();
// get the If-None-Match header
String ifNoneMatch = req.getHeader("If-None-Match");
if (etag != null && ifNoneMatch != null && ifNoneMatch.equals(etag)) {
    resp.setStatus(HttpServletResponse.SC_NOT_MODIFIED);
}
if (etag != null) {
    resp.setHeader("ETag", etag);
}
```
Gzip components (server)

- You can affect users' download times
  - Gzip supported in more browsers
  - Gzip generally reduces the response size by 70%
    - Not just for html, gzip all scripts, stylesheets, XML, JSON but **not** images, PDF
  
  Content-Encoding: gzip

- Gzip configuration
  - HTTP request
    
    Accept-Encoding: gzip, deflate
  - HTTP response
    
    Content-Encoding: gzip
How to Implement gzip?

```java
boolean canGzip = false;
Enumeration<String> hnum = req.getHeaders("Accept -Encoding");
while (hnum.hasMoreElements()) {
    String acceptType = hnum.nextElement();
    if (acceptType != null && acceptType.indexOf("gzip") != -1) {
        canGzip = true;
        break;
    }
}
if (canGzip) {
    resp.setHeader("Vary", "Accept-Encoding");
    resp.setHeader("Content-Encoding", "gzip");
    OutputStream out = resp.getOutputStream();
    OutputStream bis = // get gzip outputstream of content
    IOUtil.writeBinaryResource(bis, out);
}
```
Frameworks and Case Study
Protorabbit

> A CSS based templating and Ajax optimization framework
  > Browsers and JavaScript neutral
  > Out of the box uses the BluePrint CSS framework
    > You can easily substitute for extend the underlying framework
  > Defined in JSON format
    > HTML based template that has properties that can be substituted with strings or the contents of a file
    > specify the underlying HTML template, cache time for the template, scripts, CSS links, properties
    > may extend any number of other templates with the child properties inheriting from their ancestors
Protorabbit (cont.)

- Protorabbit Engine will set:
  - correct headers, combines CSS and JavaScript assets (if specified)
  - gzip those resources as the page is rendered
  - surrounding final pages are also cached and gzipped for as long as you specify in your template
  - can defer loading of scripts, styles or page fragments
  - Includes HttpClient that can fetch content from external sources
  - Different templates may override the cache times of their parents or just inherit them

How to get Started?

> Include **protorabit jar**, and **org.json.jar** files in **WEB-INF/lib** directory

> Include the **servlet reference**, **servlet mapping**, **template references**, other parameters in **web.xml**

> Include the CSS template files (Blueprint CSS or Yahoo Grid template frameworks)

> Include the a **JSON template** (**templates.json**)

```xml
<context-param>
   <description> Template Definitions</description>
   <param-name>prt-templates</param-name>
   <param-value>/WEB-INF/templates.json,/resources/blueprint/blueprint-templates.json</param-value>
</context-param>
```

</context-param>
Demo
Summary and Resources
Summary

> Performance optimization is an iterative process:
  > Starts with a good architecture and page design
  > Understand the page processing
  > Client coding best practices matters
> Optimization should be measured
> Use optimization recommendations, frameworks and solutions
Resources

> Best practices and Guidelines
  - https://blueprints.dev.java.net/bpcatalog/conventions/javascript-recommendations.html

> Tool
  - http://developer.yahoo.com/yslow

> Protorabbit

> Other Sites
  - http://stevesouders.com/
  - http://javascript.crockford.com/