# Performance Tune Your Ajax Application

Bob Buffone

Chief Architect
Nexaweb Technologies, Inc.
http://www.nexaweb.com

TS-9646

# Goal

As the complexity of Ajax applications increase, the need for fixing performance issues will also increase; we will explore ways to performance tune your Ajax application.

# Agenda

Introduction

Start-Up Time Tuning

Runtime Tuning

Code Analysis

jsLex—Ajax Performance Profiling

Demo

# Agenda

**Introduction**

Start-Up Time Tuning

Runtime Tuning

Code Analysis

jsLex—Ajax Performance Profiling

Demo

# Introduction

## How do I know about this stuff?

- Committer on the Apache XAP Project
  - Over 700 JavaScript™ programming language files
  - 100,000+ lines of code
  - Uncompressed
  - Uses external libraries
  - Open source
    - External committers of code
- Nexaweb uses Apache XAP
- Random everyday projects

# Agenda

Introduction

**Start-Up Time Tuning**

Runtime Tuning

Code Analysis

jsLex—Ajax Performance Profiling

Demo

# Start-Up Time Tuning

- Number of requests
- Size of requests
- Time of requests
- Time it takes to run initialization code

# Number of Requests

## Why this matters?

- Highly important for applications that use lots of JavaScript programming language files
- Each request takes time and uses a valuable resource
- **Even when cached the browser still makes a request**

# Tips for Reducing Requests

- Concatenate many JavaScript programming language files into a single file or multiple files
  - **Weight decision on minimizing start-up time against decreasing application responsiveness later**
- Ways to reduce requests
  - Dev-time
  - Run-time

# Server-Side Methods

- Pluses
  - Easy depending on your server tier
  - Easy to update code
- Minuses
  - Performance load on server
    - Can be mitigated with caching
  - Hard to distribute code to other people
- Where it could be used
  - Small single projects

# Server-Side PHP Sample

```
<head>
   <script src="combiner.php"> </script>
</head>

    function combineFile($file){
      $handle = fopen($file, "r");
      if ($handle != null){
       echo fread($handle, filesize($file));
      }
    }

    //combine files
    combineFile("file_a.js");
    combineFile("file_b.js");
```

# Development and Build Methods

- Dojo
- Ant tasks
- Command line scripts

# Dojo Overview

- Pluses
  - Not server load issues
  - Easy-to-update code
  - Dependency-based
- Minuses
  - More complicated (Ant, Rhino…)
  - Hard to distribute code to other people
  - Need to change files
- Where it could be used
  - Medium to large projects

# Dojo Overview (Cont.)

- What you will need
  - Ant, Rhino (Custom) …
- Profiles
  - Determines the starting files to trace dependencies
- Incorporate require and provide in your files

```
dojo.provide("xap.application.Application");
dojo.require("xap.util.Exception");
```

- http://www.dojotoolkit.com

# Concat Ant Task Overview

```
<concat destfile="file_ab.js" force="yes">
   <!--
      Below are the list of files to
      concatenate
   -->
   <filelist dir="core"
      files="file_a.js"/>
   <fileset dir="extensions"
      includes="**/*.js"
      excludes="file_notneeded.js"/>
</concat>
```

**(Not Dependency-Based)**

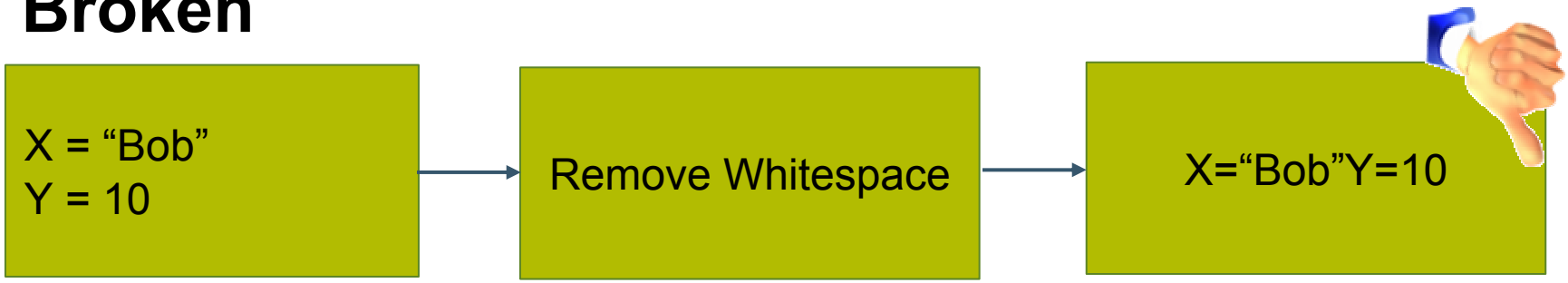
# Reducing Size of Requests

- Techniques
  - Remove white space and comments
  - gzip compression
- Tools
  - Open source, Dojo…
  - Shareware—safe compress…
- Careful of removing end of lines
  - Semicolon is optional, but needed if you remove the end of line

# Remove End of Lines

Be careful it may break your application!

## Broken




## Works

# gzip Compression

- Pluses
  - Dramatically reduces size of file
  - 330 KB to 70 KB
  - Simple to do
  - Lots of gzip tools
- Minuses
  - Lots of browser caveats

# gzip Compression Caveats

- Content-encoding must be gzip and Content-Type must be application/x-javascript; this requires web server config changes in most cases
- Browser must request HTTP 1.1 and must specify that it supports gzip
- Script tag must be between the HEADER tags in the HTML (solves IE bug with compressed JavaScript programming language)
- Some client side and/or server code will be needed to detect the user agent and request the appropriate script file—most web servers support this

# Other Start-Up Tips

- Minimize the code that is executed at start up
- Bring back data once screen is complete
- Loading images
  - Lets users know something is going on
  - Distracts the user from the time
- Want to get your application to load in under 5 seconds

# Agenda

Introduction

Start-Up Time Tuning

**Runtime Tuning**

Code Analysis

jsLex—Ajax Performance Profiling

Demo

# Runtime Tuning

- High level
  - Minimize the code
  - Use the native facilities
- Lower level
  - XML parsing
  - DOM searching
  - DOM creation
  - Coding tips

# XML Parsing

- Don’t write your own parser!
  - Tried it in XAP: 10–100x slower than native parsers
- Use the native parser
  - Encapsulated differences using parser factory
  - Does parsing
  - Normalizes DOM
- No native xPath

# XML Parsing

- Creating parsers
  - IE: `nativeDoc=new ActiveXObject("Microsoft.XMLDOM");`
  - Mozilla: `parser=new DOMParser();`
- Error handling
  - IE: check error code
  - Mozilla: check returned document
  - Error document
- White space

# DOM Creation—innerHTML

- Pluses
  - Fast
  - Code can be small

```
document.getElementById("myDiv").innerHTML =
          "<table><tr><td>New"+
          "Table</td><tr></table>";
```

- Minuses
  - Code gets messy quickly when doing more complicated snippets of HTML

# DOM Creation—innerHTML

## How YUI! does it

```
html[html.length] = "<table>";
html[html.length] =     "<tr>";
html[html.length] =     "<td>“;
html[html.length] =     “My cell text”;
html[html.length] =     "</td>“
…
domObject.innerHTML = html.join("\n");
```

# DOM Creation—Tail Recursion

- Create DOM objects

```
var table = document.createElement("table");
var tr = document.createElement("tr");
var td = document.createElement("td");
```

- Append in reverse order

```
tr.appendChild(td);
table.appendChild(tr);
onScreenElement.appendChild(table);
```

# DOM Searching

- Use native
  - getElementById()
  - getElementByTagName()
- Avoid complete DOM traversals
  - XPath - “//”
- Toolkits
  - Prototype
  - jQuery

# Coding Tips

- Use properties not getters and setters
- Avoid large string concatenation
- Look at your if statements
- Minimize string compares

# Agenda

Introduction

Start-Up Time Tuning

Runtime Tuning

**Code Analysis**

jsLex—Ajax Performance Profiling

Demo

# Code Analysis

## Finding the bottlenecks

- By hand

```
var before = new Date();
//Do something that takes awhile
var after = new Date();
alert(after - before);
```

- Pluses
  - Quick and easy
- Minuses
  - You need to know where the problem is
  - You need to change code

# Agenda

Introduction

Start-Up Time Tuning

Runtime Tuning

Code Analysis

**jsLex—Ajax Performance Profiling**

Demo

# jsLex Performance Tool

## A better way

- Systematically injects profile code
  - Ant task
  - Modified Mozilla Rhino engine
- Pluses
  - Don’t need to change your code
  - Metrics complete code base
  - Don’t need to know where the bottleneck is

# jsLex Performance Tool (Cont.)

- Pluses
  - GUI console to look through the stats
  - Save stats
  - Compare stats
  - Stack traces
- Minuses
  - Will need to change the starting file
  - Adding metric causes probing effect
  - Can slow code considerably
  - Multi-step workflow

# jsLex Workflow

1. Inject metric code into JavaScript programming language files
   - Creates function mapping file
2. Modify your HTML page to include jsLex
3. Run page
4. Open mapping file
5. Click snapshot

# DEMO

## jsLex—Performance Tool

# Summary

- Think outside the box
- It will not always be pretty
- Look for tools to make it easier
- Keep on it
- jsLex can help
- Test on all browsers

# For More Information

- Company homepage
  - http://www.nexaweb.com
- Apache project
  - http://incubator.apache.org/xap
- Home of jsLex and other cool applications
  - http://www.rockstarapps.com
- Nexaweb has a booth in the vendor area




# Q&A

# Performance Tune Your Ajax Application

Bob Buffone

Chief Architect
Nexaweb Technologies, Inc.
http://www.nexaweb.com

TS-9646