Developing Large-Scale Websites

Scandinavian Web Development Conference 2010

2 June 2010

Stefan Pettersson

@stpe

stefpet@gmail.com

http://stpe.se

netlight


hitta
.se

AFTONBLADET

(this slide was not included in the live presentation)

Large Scale JavaScript Web Sites?

Typical scenario...

Lots of traffic

Lots of content

Lots of functionality

Complex implementation

Lots of legacy code

Going from...

Traditional request-response

to...

Modern frontend architecture

Performance

Make fewer HTTP requests

Stylesheets at the top

Scripts at the bottom

Use a CDN

Set Expires Header

Gzip Compression

Optimize CSS expressions

Reduce DNS lookups

Minify JavaScript

Concat JavaScript

Make JS and CSS external

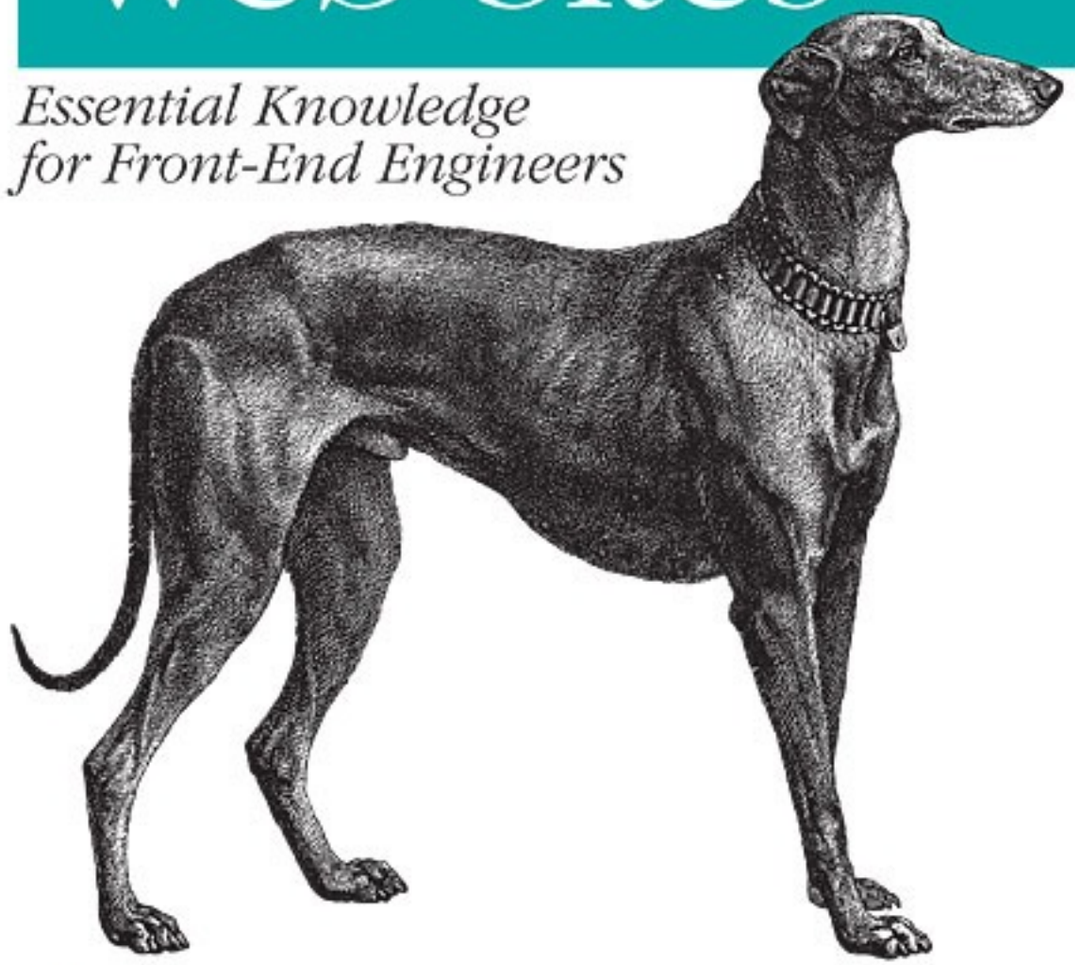
Avoid redirects

Lots to remember...?

14 Steps to Faster-Loading Web Sites

High Performance Web Sites

Essential Knowledge for Front-End Engineers



O'REILLY®

Steve Souders
Foreword by Nate Koechley

Thought that was enough?

Essential Knowlege for Frontend Engineers

Even Faster Web Sites

O'REILLY®

Steve Souders

Essential Knowlege for Frontend Engineers

Even More Fasterer Web Sites

O'REILLY®

Steve Souders

...and that's just about performance.

So...

...a good idea is to get this right the first time.

Establish...

an architecture...

best practices...

...and a way to work.

Architecture

JavaScript Loading

What may be loaded

Load on demand

Load after render

Cache renewal

Data Exchange

AJAX

Same domain or not?

XML or JSON?

XHR or JSONP?

REST

JavaScript architecture

Framework

Plugins

Widgets

Dependencies

Infrastructure

Where to put what?

Static content server

Cookieless

Cache friendly headers

Compression

Optimized for static content

Best Practices

Unobtrusiveness

```
$('#link').click(function(e) {
  e.preventDefault();
  //...
});
```

```
<a href="#" onclick="hello();"
<a href="/my/link/" id="link">
    style="color: red;">
```

```
#link {
  color: red;
}
```

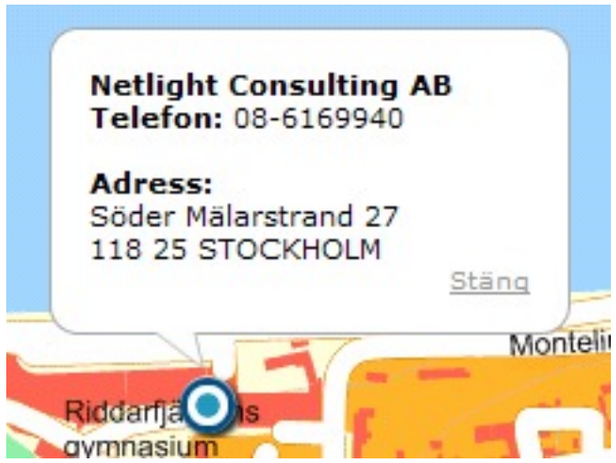
Progressive Enhancements

Graceful degradation

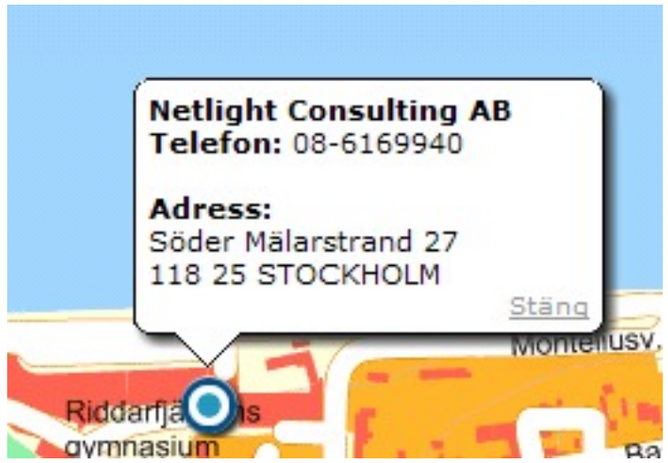
Putting your effort where it matters

“Pixel perfect in all browsers? Of course.”

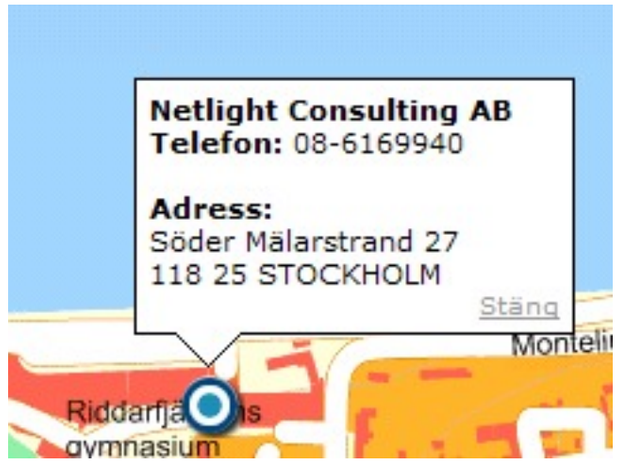
Drop it!



Complex IE6 pixel perfect implementation



Modern CSS3 dead simple implementation



Modern implementation degraded in IE6

Less complex code

Quicker implementation

Better performance

Less to test

Less that can go wrong

Namespaces and modules

Namespace pollution

```
function iAmGlobal() {
}
```

Namespace Using Object Literals

```
var foo = {
    x: 3,
    msg: "X is: ",
    say: function() {
        return foo.msg + foo.x;
    }
};

foo.say();
```

Module Pattern

```
var obj = function () {
    var key = 17;

    return {
        get: function() {
            return key;
        }
    }
}();
```

Public and Private

```
var obj = function () {
    var key = 17; // private

    function hello() {
       // private
    }

    return {
       get: function() {
            // public
       }
    }
}();
```

Know your frontend!

Scope

Inheritance (no classes)

Closures

DOM

Can’t I just use GWT?

Want to develop on the client?

Know the client!
Embrace the client!

How to share competence internally...

Peer Code Reviews

Build & Deploy

Aggregation

Minify

(Preprocessor)

Automation

...but don't cripple development process

ANT

```
<target name="concatenate" description="Concatenate all js files">
    <concat destfile="build/application.js">
        <fileset dir="src/js" includes="*.js" />
    </concat>
</target>

<target name="compress" depends="concatenate" description="Compress
app.js to app-min.js">
    <apply executable="java" parallel="false">
        <filelist dir="build" files="application.js" />
        <arg line="-jar" />
        <arg path="path/to/yuicompressor-2.4.2.jar" />
        <srcfile />
        <arg line="-o" />
        <mapper type="glob" from="*.js" to="build/*-min.js" />
        <targetfile />
    </apply>
</target>
```

Maven

yuicompressor-maven-plugin

EXPANSIVE WORLDS

Stefan Pettersson
CEO @ Expansive Worlds AB
http://www.expansiveworlds.com