



JavaOne

# Fast, Beautiful, Easy: Pick Three

Bruce Johnson and Joel Webber

Google, Inc.

<http://code.google.com/webtoolkit>

TS-6475



# Developing With Google Web Toolkit

Focus on the user and all else will follow

Building no compromise AJAX  
applications with Java™ technology  
using Google Web Toolkit



# Today's Topics

The potential of AJAX

GWT  $\equiv$  software engineering for AJAX

No compromise usability

Fast is better than slow

Wrap-up

Q&A

# Today's Topics

## The potential of AJAX

GWT  $\equiv$  software engineering for AJAX

No compromise usability

Fast is better than slow

Wrap-up

Q&A

# Necessity Is the Mother of Invention

A polite way of saying that the status quo sucks

Browsers are treated like HTML dumb terminals

**Everything** is an HTTP round trip and history entry

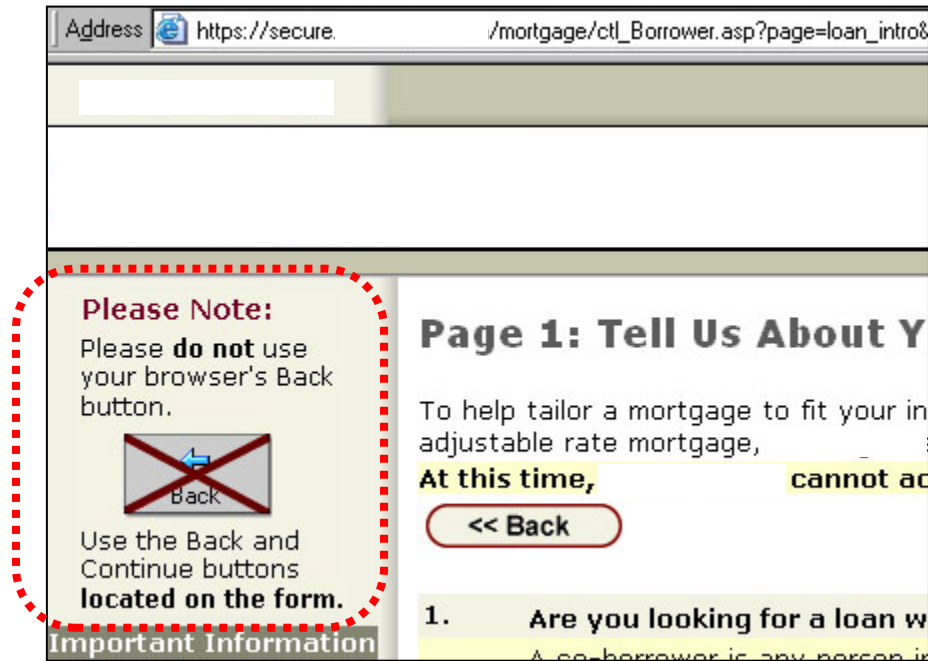
Every...page...is...so...sluggish...

...and...disconnected...that...I...

...keep...forgetting...where...I...am...

# Real World Example #1

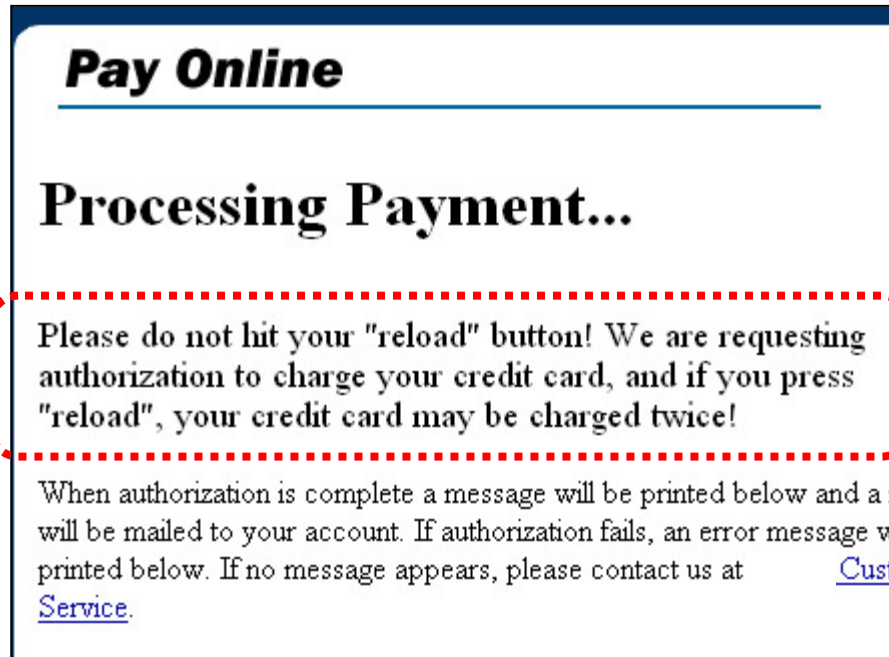
Please **do not** use your browser's Back button



But what if I **do** click Back?  
AJAX can (in theory) solve this

# Real World Example #2

My credit card may be charged twice?



**Pay Online**

---

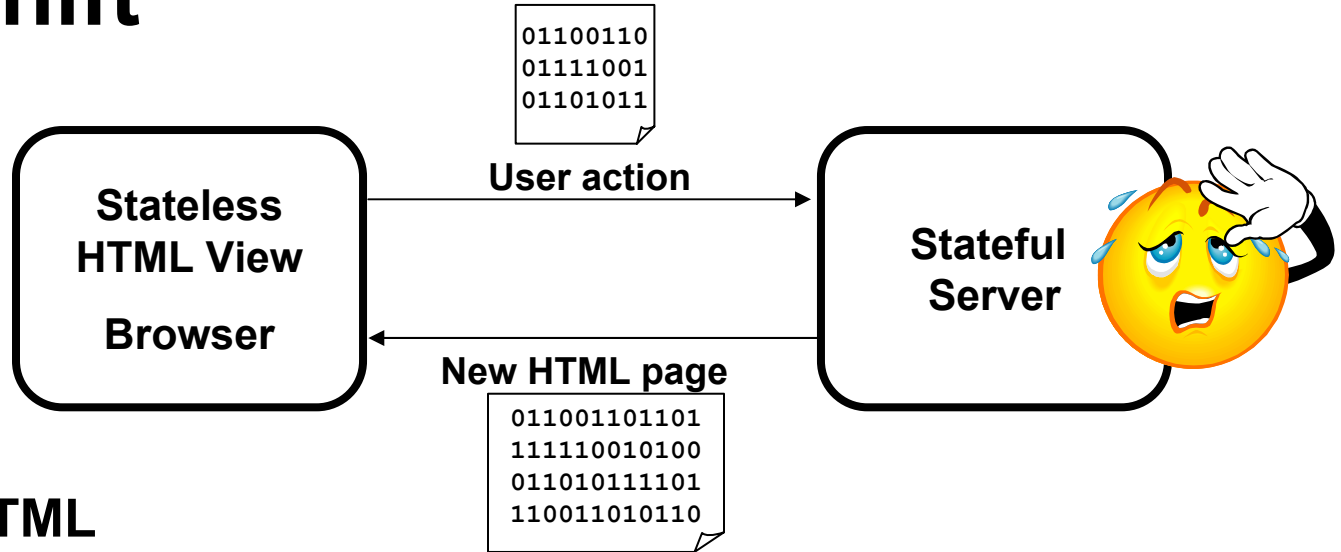
**Processing Payment...**

Please do not hit your "reload" button! We are requesting authorization to charge your credit card, and if you press "reload", your credit card may be charged twice!

When authorization is complete a message will be printed below and a message will be mailed to your account. If authorization fails, an error message will be printed below. If no message appears, please contact us at [Customer Service](#).

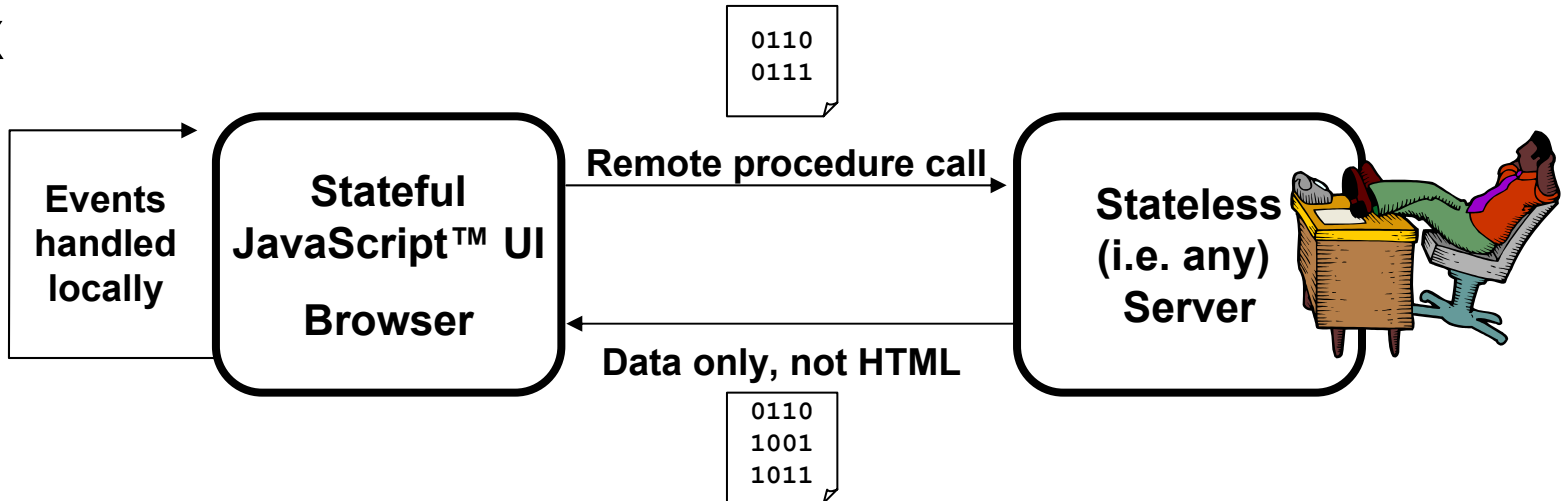
**What if the network hangs? What should I do?  
AJAX can (in theory) solve this**

# AJAX Shift



## Traditional HTML

## AJAX





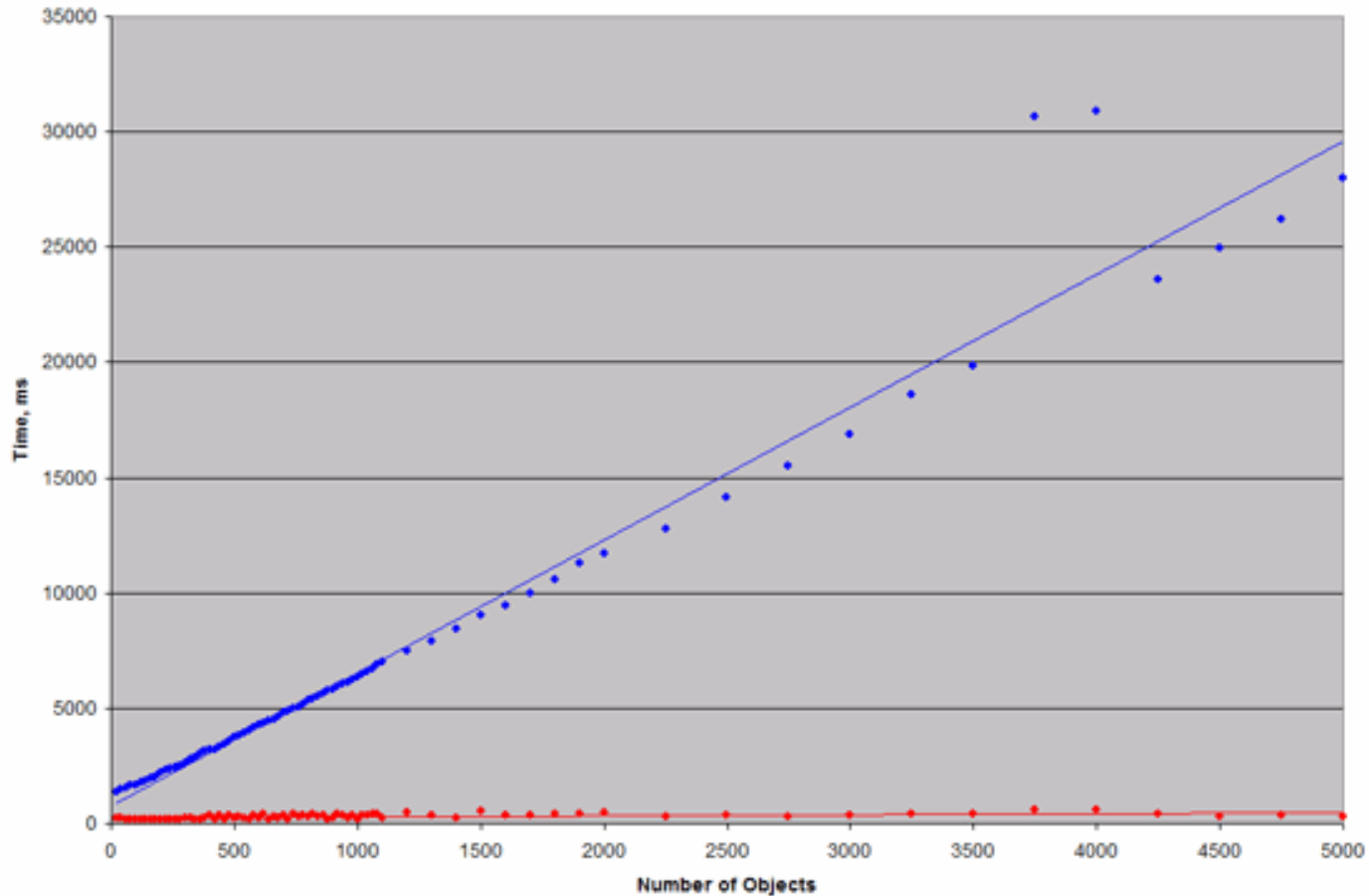
# AJAX Slippery Slope

As told by a forlorn engineer

- I begin experimenting with JavaScript™ technology
- The salespeople love it
- We're an AJAX shop?
- Oh yeah, we can't just support Internet Explorer
- Wait, this is hard
- I quit! Find another sucker to maintain this spaghetti!

# Real World Example #3

A popular browser slows down as your app gets bigger



# To Be Crystal Clear

It is very easy to slip into making  
a poorly planned AJAX investment

...but you'll live  
with the consequences  
for a **long, long** time

# Today's Topics

The potential of AJAX

**GWT ≡ software engineering for AJAX**

No compromise usability

Fast is better than slow

Wrap-up

Q&A

# What Is Google Web Toolkit (GWT)?

- Build AJAX apps with Java technology
- What makes GWT interesting?
- GWT  $\neq$  applets
- GWT is much more than a compiler

# Hello, AJAX

```
public class Hello implements EntryPoint {
    public void onModuleLoad() {
        Button b = new Button("Click me",
            new ClickListener() {
                public void onClick(Widget sender) {
                    Window.alert("Hello, AJAX");
                }
            }
        );

        RootPanel.get().add(b);
    }
}
```



# DEMO

Hello, AJAX



# GWT ≡ AJAX Software Engineering

Reclaim your inner software engineer

- Any Java IDE (or no IDE...you rebel, you)
- Rapid edit/test/debug/refactor cycle
- Unit test
- Re-use jars
- Design patterns (e.g., client-side MVC)
- Javadoc™ tool
- Compile-time errors



# Lightweight Development Cycle

## Refreshing recompiles

- The GWT dev cycle mimics normal web dev:
- Open GWT's **hosted** web browser
- Browse to your app under development
- Edit your source with the browser running
- Refresh the browser
- Changes are picked up instantly



# DEMO

Refreshing in Hello, AJAX



# Today's Topics

The potential of AJAX

GWT  $\equiv$  software engineering for AJAX

**No compromise usability**

Fast is better than slow

Wrap-up

Q&A

# The GWT Mission

No compromise usability—technology isn't the point

To radically improve the web experience for users by enabling developers to use existing Java tools to build no compromise AJAX for any modern browser

- Users first
- Developers second
  - Whenever possible, a very close second
- Gee-whiz technology is a distant third
- If you share this perspective, you'll like GWT

# Web Usability Fundamentals Intact

Let's not re-teach our parents how to use the web

- Focus on basics
- Prefer native UI elements
- Support keyboard-only use
- Honor font size
- Keep user in control of browser
- Emphasize speed, especially at start-up
- Ideal: feels like a traditional web app, just better



# DEMO

Kitchen Sink and User Admin



# History Class

Fits nicely with client-side MVC

- Simple history API
- Listening to history events
  - Implement `HistoryListener`
  - `History.addHistoryListener(controller)`
- Creating history events
  - `History.newItem("settings")`
- History tokens form the basis of linkable URLs
  - `http://example.org/email.html#settings`
- `Hyperlink` class automatically adds entries

# Messages Interface

## Efficient internationalization

- Use interfaces to define templates

```
interface ErrorMessage extends Messages {  
    String accessDenied(  
        int errorCode, String username);  
}
```

- Store localized data in properties files

```
permissionDenied = \  
Error {0}: User {1} cannot access {2}
```

- Bind them with **compile-time** code generation

```
Window.alert(msgs.accessDenied(515, user))
```

- The above wouldn't compile :-)



# Style with CSS

## Separating code and presentation

- Widgets publish CSS style names

```
public ListBox(String caption) {  
    ...  
    setStyleName("gwt-ListBox");  
}
```

- Write CSS rules that bind to widgets

```
.gwt-ListBox {  
    background-color: yellow;  
    color: black;  
}
```

- Loose coupling between CSS and code

# Today's Topics

The potential of AJAX

GWT  $\equiv$  software engineering for AJAX

No compromise usability

**Fast is better than slow**

Wrap-up

Q&A

# Fast Is Better Than Slow

No compromise speed

- Speed is a critical component of usability
- Building AJAX apps = tinkering with code
- But are we ensuring the apps are great for users?
- Performance metrics are a nice start
- GWT 1.4 includes benchmarking subsystem
- Let's see how this plays out in a real app...



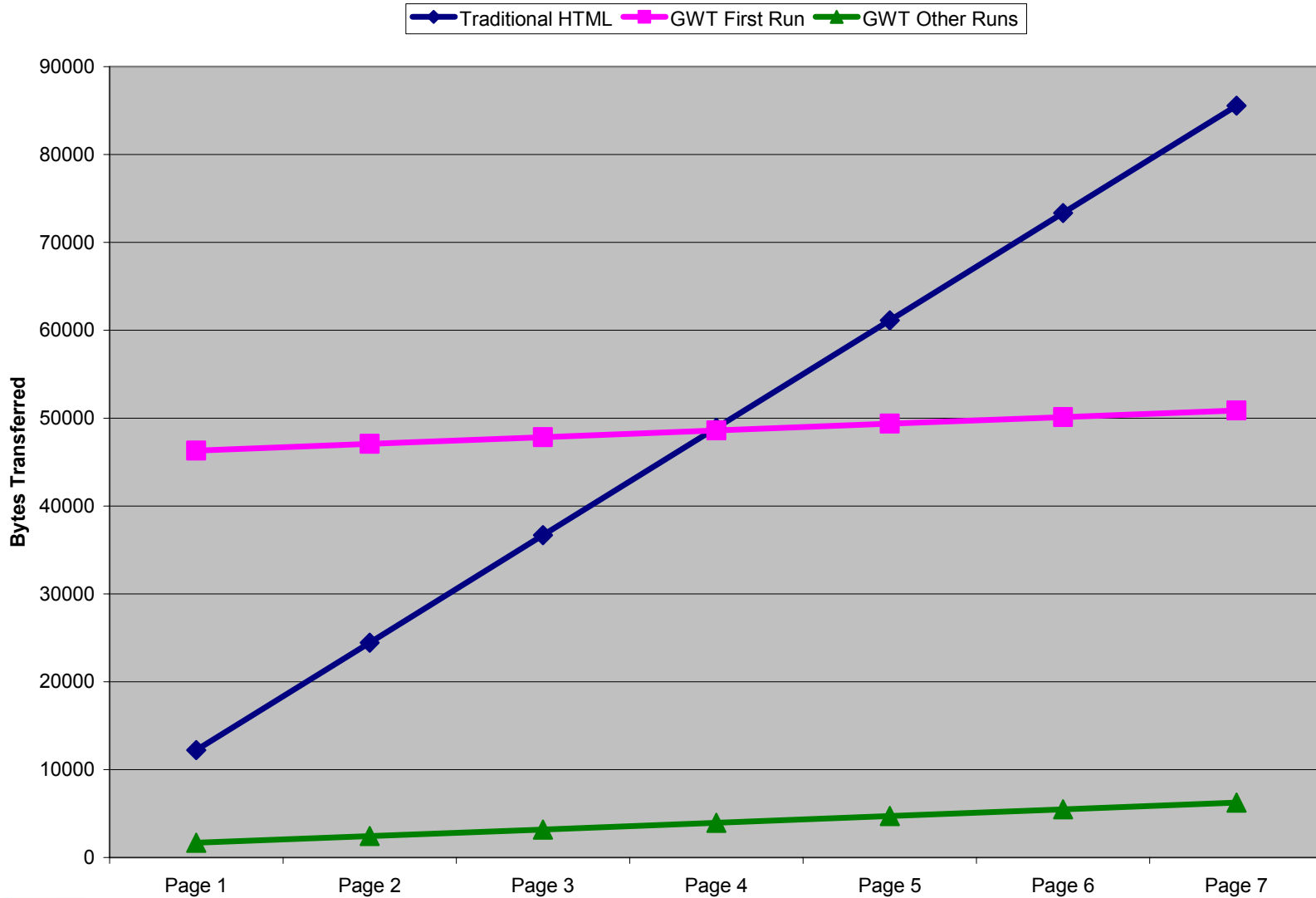
# DEMO

RPC: DynaTable

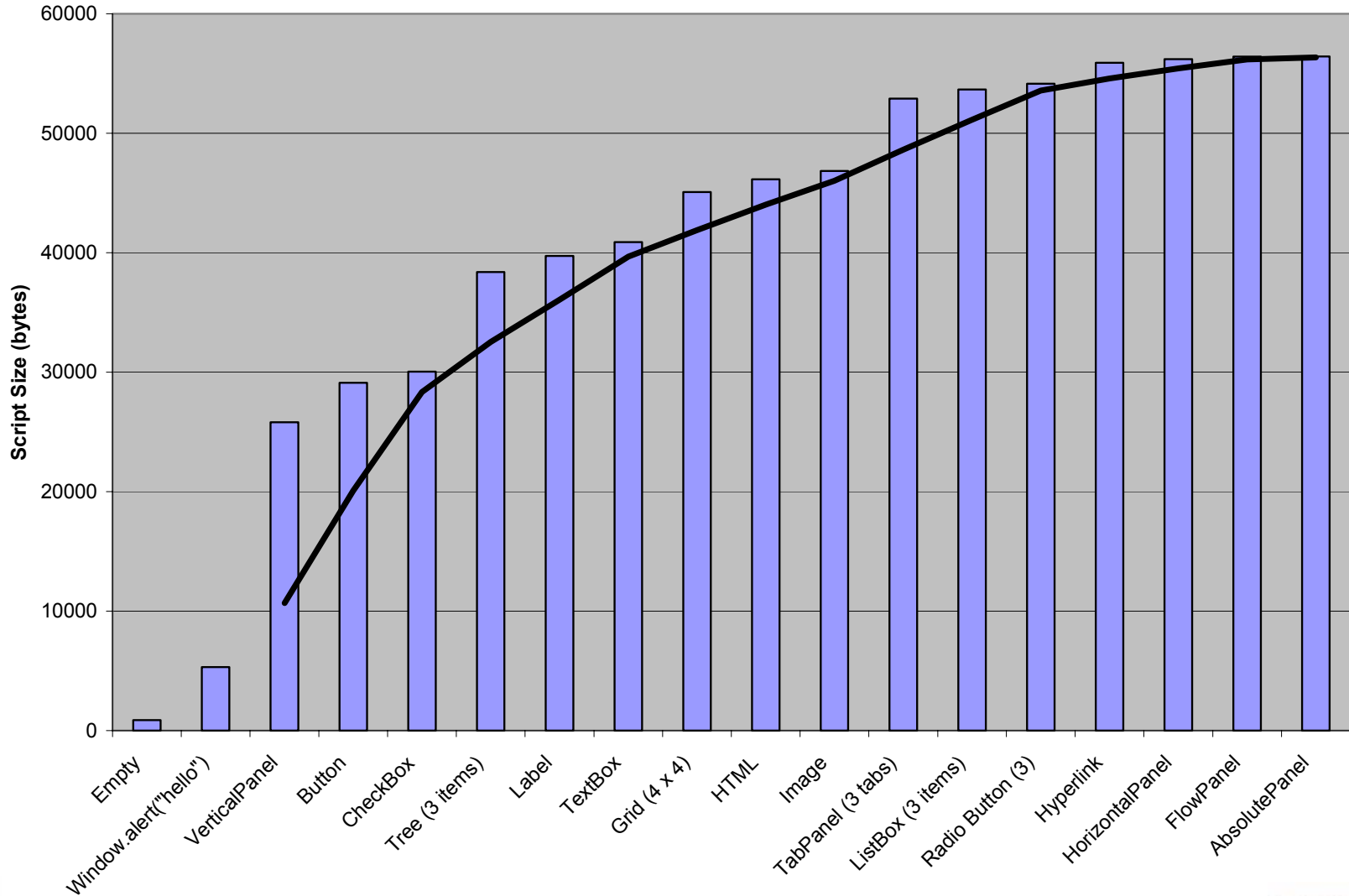




# Network Usage Over Time



# Rate of Script Size Growth



# GWT Eats Its Own Dog Food

Tastes very a-nice-eh

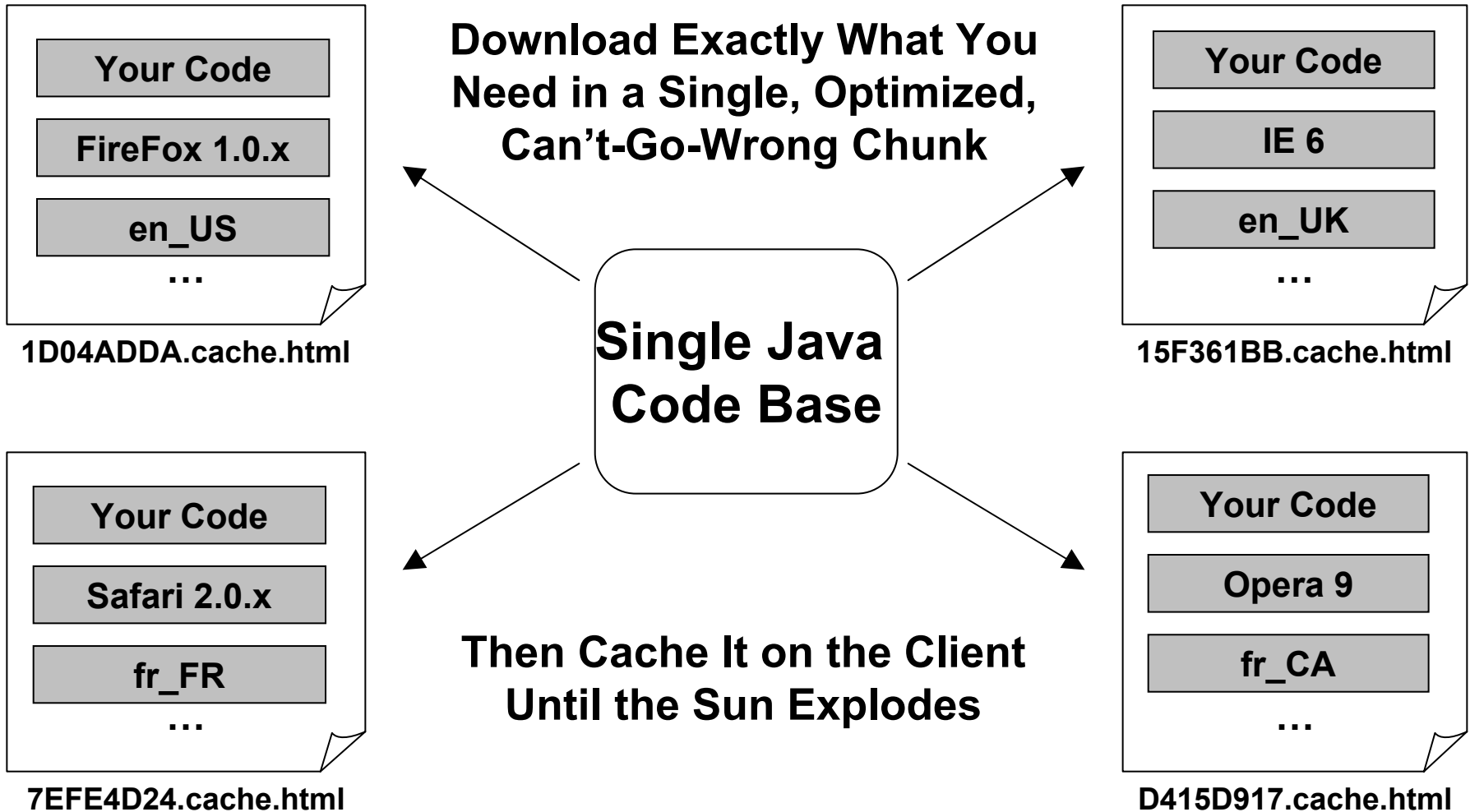
- Only a few magic GWT facilities
  - JSNI, compile-time class selection and code generation
- All GWT libraries built with them
  - Portability, i18n, rich text, image bundles, RPC...
- You can use them yourself
  - Write your own UI library
  - Just kidding...but you could
- Compiler optimizations improve everything
  - Core GWT libraries, generated code
  - Your application code

# Non-Trivial Compiler Optimizations

- Three words: Whole program optimization
- For example, type tightening to eliminate polymorphism
  - `Shape s = new Circle(2); // radius of 2`
  - `double a = s.getArea();`
  - can become
    - `Circle s = new Circle(2);`
    - `double a = (s.radius * s.radius * Math.PI);`
  - which, if Circle has no side effects, can become
    - `double a = 12.5663706143591;`
- Leverages static type information
- Imagine optimizations like that across your entire app
- Decreasing script size and increasing speed are **complementary** goals



# Optimized Permutations



# JavaScript Native Interface (JSNI)

With great power comes great responsibility

- Implement **native** methods using JavaScript technology

```
// This is Java source
private native String flipName(String name) /*- {
    // This is JavaScript source
    var re = /(\w+)\s(\w+)/;
    return name.replace(re, "$2, $1");
}-*/;
```

- Call back and forth from Java source to JavaScript technology
- Great for interop or hand-tweaking script

# Today's Topics

The potential of AJAX

GWT  $\equiv$  software engineering for AJAX

No compromise usability

Fast is better than slow

**Wrap-up**

Q&A

# Not Enough Time to Demo Everything

- Automatic, dynamic dependency inclusion
  - External CSS and JavaScript technology
- Everything is cross-browser
  - IE6+, FF 1.0.x, FF 1.5.x, Safari 2.0.x, Opera 9.x
- Your choice of development platforms
  - Mac OS X, Linux, Windows
- Your choice of IDEs
  - IntelliJ IDEA, Eclipse, NetBeans™ Integrated Development Environment, JCreator, JBuilder

# GWT Is Open Source

w00t!

- Licensed under Apache 2.0
  - Source available on Google Code project hosting
- Making GWT Better
  - The spirit of GWT, including design axioms
- Great community
  - 100,000+ downloads of the RCs for GWT 1.3
  - Great discussion on G-W-T and G-W-T-C lists
  - 7,500+ members of the developer forum
  - 300+ developers on the contributors list
  - Patches are rolling in!

# Documentation Included

## Getting Started Guide


## Widget Gallery

**Building a Sample Application**

All the sample applications are in the `samples/` directory in your GWT package. Each sample has a script you can run to start it in `hosted mode` and a script you can use to compile it into JavaScript and HTML, to run it in `web mode`.

**Running in Hosted Mode**


To run the `Kitchen Sink` example in `hosted mode`, navigate to the `samples/KitchenSink/` directory and run the `KitchenSink-ahell` script. This will open the GWT browser with the Kitchen Sink application running inside.



Since you're running in `hosted mode`, the application is running in the Java Virtual Machine (JVM). This is typically the mode you'll use to debug your applications.

**Running in Web Mode**

To run the application in `web mode`, compile the application by running the `KitchenSink-compile` script. The GWT compiler will generate a number of JavaScript and HTML files from the Kitchen Sink Java source code in the `war/` subdirectory. To see the application, open the file `www/com.google.gwt.sample.kitchensink.KitchenSink/KitchenSink.html` in your favorite web browser.



## Developer Guide

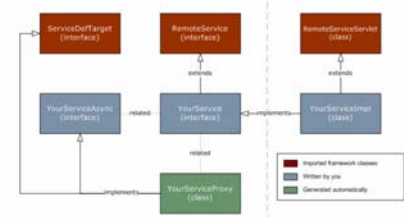
## Class Reference

Google Web Toolkit (Beta)

Google Code Home > Google Web Toolkit > Developer Guide > Remote Procedure Calls > RPC Plumbing Diagram

**RPC Plumbing Diagram**

The section outlines the moving parts required to invoke a service. Each service has a small family of helper interfaces and classes. Some of these classes, such as the service proxy, are automatically generated behind the scenes and you generally will never realize they exist. The pattern for helper classes is identical for every service that you implement, so it is a good idea to spend a few moments to familiarize yourself with the terminology and purpose of each layer in server call processing. If you are familiar with traditional remote procedure call (RPC) mechanisms, you will recognize most of this terminology already.



**Translatable Java code (runs as JavaScript on client)**

- YourServiceProxy (class)
- YourServiceInterface (class)

**Standard Java code (runs as bytecode on server)**

- RemoteService (interface)
- YourServiceImpl (class)
- RemoteServiceProxy (class)

Legend:
 

- Imported framework classes
- Written by you
- Generated automatically

Google Web Toolkit (Beta)

Google Code Home > Google Web Toolkit > Developer Guide > GWT Class Reference > com.google.gwt.user.client.ui > Hyperlink

**Class Hyperlink**

```

public class Hyperlink
extends Widget
implements HasText, HasColor, HasStyleName

A widget that serves as an "internal" hyperlink. That is, it is a link to another state of the running application. When clicked, it will create a new history item using History.newItem(), but without reloading the page.

Being a true hyperlink, it is also possible for the user to "right-click, open link in new window", which will cause the application to be loaded in a new window at the state specified by the hyperlink.


```

**Example**

```

public class MyPageExample implements EntryPoint, RemoteServiceProxy {
    private Label lbl = new Label();

    public void onModuleLoad() {
        // Create three hyperlinks that change the application's history.
        Hyperlink link1 = new Hyperlink("Link to foo", "foo");
        Hyperlink link2 = new Hyperlink("Link to bar", "bar");
        Hyperlink link3 = new Hyperlink("Link to baz", "baz");

        // If the application starts with no history items, what is set in the
        // "foo" state.
        History.newItem("foo", RemoteServiceProxy.DEFAULT);
    }
}

```

# Summary

- You need leverage to use AJAX well with low risk
- Ph.D. in browser quirks is no longer a hiring prereq
- Turn AJAX development into software engineering
- GWT rewards using good engineering practices
- We will share our best work and ideas with you, and we hope you will return the favor
- Much more to come...see you online!

# Today's Topics

The potential of AJAX

GWT  $\equiv$  software engineering for AJAX

No compromise usability

Fast is better than slow

**Q&A**



# GWT Is Not All-or-Nothing

- Only use what you want
- Don't pay for what you don't use
- Integrate with other technology as needed
- **Enabling** technology, not a sticky spiderweb

# GWT Does Not Force You to Start Over

Attach code to existing pages with a `<meta>` tag

```

<html>
...<meta name="gwt:module" content="..." />
...<h1>Welcome to GWTravel Services</h1>
...<div id="reservationWizard">
...</div>
...</html>
  
```

Interact with the page DOM however you see fit

```

Panel p = RootPanel.get("reservationWizard");
Wizard wiz = new ReservationWizard();
p.add(wiz);
  
```

As loosely coupled as you need it to be

# It Isn't GWT vs. Everything Else

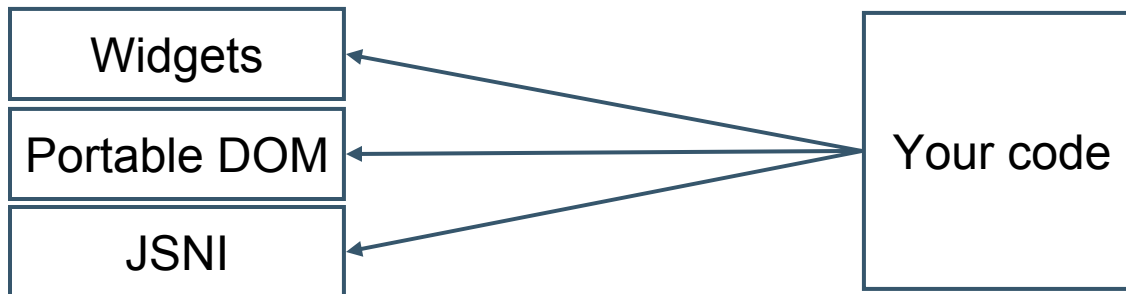
- Not sure why some people insist on this
- Using GWT  $\neq$  foregoing other approaches
- Not into smackdowns
- Ability to mix and match is important
- We've tried to make integration easy (JSNI)

# It Isn't Java vs. JavaScript

- No language wars!
- It's about leverage
- The goal isn't to hide JavaScript technology per se
- GWT is leverage for JavaScript technology and DHTML

# We Know That Abstractions Leak

- Two kinds of abstractions...
  - Those that leak a lot
  - Those that leak a little
- Embracing that fact makes better educated users
- UI leaks a lot, so we don't attempt to hide it



- RPC only leaks a little (async)

# Don't Overlook the Hosted Browser

- Feels like normal browser development
- GWT's embedded Tomcat is just for convenience
- You can debug against any backend
- The magic switch is **-noserver**



# Q&A

<code />



JavaOne

# Fast, Beautiful, Easy: Pick Three

Bruce Johnson and Joel Webber

Google, Inc.

<http://code.google.com/webtoolkit>

TS-6475