the POWER of JAVA™

Java™ Technology Techniques for Developing AJAX Applications

Bruce Johnson and Joel Webber

Software Engineers
Google, Inc.
http://www.google.com

TS-3376

Learning to Love AJAX

Like Sausage, It's Easier to Love When Properly Abstracted

Learn how to apply some of Google's techniques for building AJAX applications to your own projects

Learning to Love AJAX

Witty/Hilarious Introduction

You and AJAX: A Love/Hate Relationship

An Approach We've Found Useful…

To Spend More Time Loving AJAX

And Less Time Not Loving It

Q&A

Extremely Funny Joke #1

Stop Me if You’ve Heard This One

- Patient: “It hurts when I write AJAX code”
- Doctor: “So, don’t write AJAX code”
- Okay, that wasn’t funny
- Users actually want (good) AJAX functionality

Extremely Funny Joke #2

Other Dubious Sociological Equations

- `!love ≠ hate`
- `!love = indifference`
- `love ≈ hate`
- Never more true than when building AJAX code

Learning to Love AJAX

Witty/Hilarious Introduction

You and AJAX: A Love/Hate Relationship

An Approach We've Found Useful…

To Spend More Time Loving AJAX

And Less Time Not Loving It

Q&A

AJAX: Easy for Users to Love

What's Not to Like?

- No installation
  - Every application is just a URL away
  - No such thing as DLL Hell
- Secure…the mantra we teach our parents
  - installing things = mostly unsafe
  - surfing the web = mostly safe
- Has web simplicity
  - AJAX pages have the familiar web look and feel
  - Web simplicity—back, forward, buttons, links, URLs

AJAX: Easy for Developers to Loathe

AJAX Is Cool…Sort Of

- No installation…sort of
  - In a sense, it's also always-reinstall (better be small)
  - Modularizing JavaScript™ technology is really tricky
- Secure…sort of
  - Hard enough just to make AJAX work at all!
  - Lots of JavaScript code = lots of attack surface
- Has web simplicity…sort of
  - History, bookmarks, and even hyperlinks misbehave
  - Badly coded AJAX is worse than traditional HTML

AJAX: Easy for Developers to Loathe

The Good Thing About Standards Is…

- You need regexps to list them all on one page
  - HTTPS?, [DX]?HTML (3.2|4.0), CSS[1-3]
  - DOM Level[0-3]
  - (Java|ECMA|J|VB)Script
  - (X|VR?|Math)ML
  - SVG, Canvas, Flash
  - JSON, SOAP, XML-RPC
- And, darn it, they really don’t interoperate well
- Far less coherent than the Java technology world

AJAX: Easy for Developers to Loathe

Goodbye, Software Engineering?

- Hey, what happened to object-oriented programming?
  - (IMHO, faking it with JavaScript-based prototypes is tricky to get right)
- Static type checking?
- Design patterns?
- Unit testing?
- Code reuse?
- **IDEs?!?!**
- **Debugging?!?!**

Learning to Love AJAX

Witty/Hilarious Introduction

You and AJAX: A Love/Hate Relationship

An Approach We’ve Found Useful…

To Spend More Time Loving AJAX

And Less Time Not Loving It

Q&A

Wrangling AJAX—Goals

What We Wanted From an AJAX Toolset

- Make great AJAX apps that are still very webby
- Leverage the Java language and technologies
- Minimize special-case coding
- Facilitate reuse at the Java language level
- Rich remote procedure call (RPC) semantics
- Scalability…server-side session state is a bear
- Basically: the impossible

Wrangling AJAX—Solution

Our Solution: Write Client-Side AJAX Code in Java Technology

- Why not translate Java-based source code into JavaScript-based code?
- Hey, this really works…

Code Sample—Hello, AJAX

```
public class HelloAjax implements Module {

  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

Learning to Love AJAX

Witty/Hilarious Introduction

You and AJAX: A Love/Hate Relationship

An Approach We've Found Useful…

To Spend More Time Loving AJAX

And Less Time Not Loving It

Q&A

Wow, That’s So Much Easier

If You Can Make Everything Look Like a Nail, Then All You Need Is a Hammer

- Redefining the problem has been fruitful
- Session state? All client, not a server issue
- UI event handling? All client, no round trips
- Deployment? No fancy server, just compiled JavaScript-based code
- Leverage for the biggest AJAX headaches
  - Mantra: Solve the problem once and wrap it in a class
  - History? Create a History class
  - Cross-browser? Create an abstract DOM class
  - RPC? Create an all-Java-based RPC mechanism

Stop Bugs Before They Start

Reclaim the Presumption of Workingness

- typos + expandos = bug-o-s
- Imagine this gem on line 5912 of your script

```
x.compnent = document.getElementById("x");
// a "spelling bug" that will bite much later
```

- There’s a reason static type checking was invented
- Java-based source code instead of JavaScript-based code = fewer bugs to begin with
- And reuse is a good way to not write bugs
- And don’t forget code completion
- And this starts to matter a lot for big projects

Java IDEs and Debuggers

Reclaim Your Beloved Tools

- Sadly, this is what often suffices for debugging in the AJAX world

```
alert("made it here 1");
doSomeStuff();
alert("made it here 2");
doSomeMoreStuff();
alert("made it here 3");
```

- More Java-based coding means more IDE time
- Automated refactoring, naturally

Code Reuse and Quality

Reclaim Your Inner Software Engineer

- Ripe for building frameworks
- Share AJAX libraries as JARs
- Share RPC interfaces with a Java vocabulary
- Mature OO patterns in your AJAX designs
- JUnit for testing AJAX code

Learning to Love AJAX

Witty/Hilarious Introduction

You and AJAX: A Love/Hate Relationship

An Approach We've Found Useful…

To Spend More Time Loving AJAX

And Less Time Not Loving It

Q&A

Use Case: Widgets and Layout

A UI Object Model Instead of a Document Object Model

- Build cross-browser widgets in straight Java-based code
- Constraint-based layout with panels
- Create new widgets from existing ones

```
public class InboxWidget extends Composite {
   private EmailList list = new EmailList();
   private EmailPreview pvw = new EmailPreview();
   // combine them together in a simple panel to
   // create a new, reusable composite widget
}
```

- Styled with CSS

DEMO

Widgets and Layout

Use Case: Remote Procedure Calls

A Key Moving Part

- Many solutions out there (JSON, XML-RPC,…)
- Loose binding adds exposure to runtime errors
- A pure Java-based RPC interface sure is nice!

```
interface SpellService extends RemoteService {
  String[] suggest(String word)
}
```

- Client and server can speak the same language
- Inner classes make it easy to deal with asynchronous RPCs

Use Case: Remote Procedure Calls

Pass Around Type-Checked Java-Based Objects

```
SpellServiceAsync spell = GWT.create(SpellService.class);

spell.suggest("compnent", new AsyncCallback() {

  void onSuccess(Object result) {
    String[] alts = (String[])result;
    if (alts.length > 0)
      showSuggestionsInGui(alts);
  }

  void onFailure(Throwable e) {
    reportProblemInGui(e);
  }

});
```

DEMO

Remote Procedure Calls

Use Case: Real History

Historically Flaky

- History is the first thing to go in most AJAX apps
- Hidden IFRAME and/or timer tricks
- Different solutions work best in each browser
- Solve it once and reuse

```
History.addHistoryListener(myController);
```

- History support leads to bookmark support

  http://google.com/gulp.html#beta_carroty

DEMO

Real History

Summary

- AJAX is hard: you tame it…or vice-versa
  - You need leverage to take full advantage of AJAX
- PhD in browser quirks is no longer a prereq
- May you wield the sword of AJAX wisely
  - Very easy to make apps clunky, slow, and unintuitive
- 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!

Q&A

Java™ Technology Techniques for Developing AJAX Applications

Bruce Johnson and Joel Webber

Software Engineers
Google, Inc.
http://www.google.com

TS-3376