# FLASH&FLEX

DEVELOPER'S MAGAZINE

Vol.4 No.10 Monthly Issue 10/2011 (28) October ISSN 1898 -9136

# ActionScript Programming

## MASTER THE FUNDAMENTAL FEATURES OF THE LANGUAGE

## UNION PLATFORM AS THE REALTIME CONNECTIVE GLUE BETWEEN CLIENTS AND SERVER

## DEVELOPING A CHAT APPLICATION WITH XMLSOCKET AND EVENTMACHINE

## SCOREOID AND THE IMPORTANCE OF LEADERBOARDS

**STEVE LIONEL ON WHY FORTRAN STILL MATTERS**

BY BRIDGET MOORE FOR INTELLIGENCE IN SOFTWARE

**NICE GESTURE, BUT WHAT DOES IT MEAN?**

BY TIM KRIDEL FOR INTELLIGENCE IN SOFTWARE

# STREAMING + INTERACTIVITY = LIMITLESS POSSIBILITIES.

The new FMS 4 – Flash® Media Interactive Server (FMIS) & Flash Media Enterprise Server (FMEN) – offers better performance in interactive media streaming.

## Adobe Flash® Media Server 4 Advantages:

- New RTMFP Multicast P2P capability
- Reduces latency and bandwidth usage
- Interactive access to buffered media
- Faster switching with RTMP Dynamic Streaming

FMIS plans starting at $19.95/mo.
FMEN pricing, please contact us.

INFLUXIS

www.influxis.com

# STREAMING + INTERACTIVITY = LIMITLESS POSSIBILITIES.

Our new mobile streaming lets you reach your audience on the go.
Deliver your videos to smartphones and tablets with ease.

**Mobile Streaming Advantages:**

- For VOD or live streaming
- Intuitive and easy-to-use
- Optimized for all mobile platforms and social media
- HTML5-ready
- Custom mobile streaming solutions

Start with a free account.

INFLUXIS

mobile.influxis.com

# Editor's Note

## Master Your Skills

*The October issue of FFD magazine features information about programming. We gathered a few articles from different sources to give you a deep insight into this matter.*

*The cover article is written by our regular columnist Huw Collingbourne. In this new series, Huw Collingbourne explains the fundamental features of ActionScript programming. If you are new to ActionScript, this is the place to start! You will learn the essential features of an ActionScript program as well as how to use data types and variables and how to concatenate strings, add numbers and display output. This article is for all of you who have some knowledge in this filed. Before you start, make sure that you know how to use a Flash or ActionScript editor or IDE, how to compile (or publish) and Debug a program. Just go to page 12 and see how to start ActionScript programming.*

*If you want to learn more about Flex solution, just go to page 32 to read the article in our regular section Flex/ Action Script 101. Marc Pires wrote article on Developing A Chat Application With XMLSocket and EventMachine. After reading, you will learn how to use XMLSocket and how to use Ruby to develop server side apps and APIS. Marc shows how you can connect to servers via XMLSocket and presented a new topic EventMa-chine and Ruby.*

*If you are a fun of Java, you will find a very informative and useful article written by Chad Ternent entitled The Union Platform. Integrating the JavaScript and Flex Clients. He settled on the Union*

*Platform (http://www.unionplatform.com) as the real-time connective glue between clients and server. Union's Reactor client framework for Flex abstracted away the distributed messaging headaches, and now with the OrbiterMicro framework for JavaScript, in his article, he will*

*show how easy it is to integrate different types of clients to the same Union server. Page 18.*

*The next article I would recommend that we publish in FFD is on page 26.*

*This month we also present articles written by Tim Kridel for Intelligence in Software. They are very good piece of knowledge. Just go to next page to find them in table of content.*

*All articles are very interesting and they need to be marked as Need to be Read! As always!*

*Thank the Beta Testers and Proofreaders for their excellent work and dedication to help make this magazine even better. Special thanks to all authors that help me create each issue.*

*I would like to mention some supporters this time and thank Csomák Gábor, Huw Collingbourne, Marc Pires, Kevin Ruse, Chad Ternent. They work really hard to get magazine out for your to read.*

*Please keep up the great work and send in your articles, tutorials or product reviews, questions, ideas and advises.*

*Enjoy reading!*
*Ewa Dudzic & FFD team*

FLASH&FLEX

Editor in Chief: Ewa Dudzic ewa.dudzic@ffdmag.com
Proofreaders: Betsy Irvine, Patrick French

DTP Team: Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl
Art Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl
Senior Consultant/Publisher: Paweł Marciniak
Flex/ActionScript 101 Section Editor: Marc Pires marcpiresrj@gmail.com
iPhone Development Section Editor: Ryan D'Agostino
ActionScript in Action Section Editor: Huw Collingbourne
Games Section Editor: Chris Hughes
Contributing Editors: Pedro de La Rocque, Ali Raza, Csomák Gábor

Publisher: Software Press Sp. z o.o. SK
ul. Bokserska 1 02-682 Warszawa Poland Worldwide Publishing

Software Press Sp. z o.o. SK is looking for partners from all over the World.
If you are interested in cooperating with us,
please contact us by e-mail: cooperation@software.com.pl



Thanks to the most active and helping beta testers:
Russell TangChoon, Lee Graham, Jassa Amir Lang, Ed Werzyn, Yann Smith-Kielland, Justus, Csomák Gábor, Kevin Martin, Charles Wong, Ali Raza, Almog Koren, Izcoatl Armando Estanol Fuentes, Lionel Low, Michael J. Iriarte, Paula R. Mould, Rosarin Adulseranee, Sidney de Koning

To create graphs and diagrams we used smartdraw.com program by SmartDraw company.

The editors use automatic DTP system ADPOS
Mathematical formulas created by Design Science MathType™

# CONTENTS



# Tip of the issue

### The Fox is Your Friend: FireFox and FlashTracer

**by Julia Detar**

If you don't know already, there's a great Firefox plug-in called Flash Tracer which displays your flash log inside the browser! This makes it much easier to use your Flash logs because you don't have to search for them each time and it auto-updates. It will display for both online and local content. You will need a Flash player debug installed in Firefox to run this plug-in. Once installed just set up the correct path to your Flash log and bam, you're done! You can also change some settings like when to cap warnings and font size and whether to embed the window or have it float. One thing to keep in mind when using Flash Tracer… if there is other Flash content running in other windows or on the same page, it will also trace them. Close out the other Flash content to focus on your own.

## Sponsors of the Issue

# Mitigate Mobile Cross-platform Headaches

Android is now the most widely used mobile operating system in both the United States and the world, according to the research firms Canalys and Nielsen. But behind that news lurk a lot of headaches for enterprise developers.

For example, 36 percent of U.S. smartphones run Android. So unless an enterprise is comfortable alienating 2 out of every 3 employees or customers, its developers have to create apps for at least iOS, BlackBerry or Windows Mobile.

If that enterprise is multinational, then its developers can add Symbian to their to-do list.

We have European customers that say: *We know Symbian is going to die. It might take five years, but in the meantime, a large percentage of our workers have Symbian phones*, says Adam Blum, CEO of Rhomobile, which provides development tools.

If that weren't enough work for developers, there's also fragmentation within some operating systems.

For example, each Android smartphone vendor often has a unique implementation of the OS, while differences in screen size and resolution create additional variables that affect an app's user experience.

## Web or Native?

Some developers sidestep fragmentation by using *Web services* or *Web apps*, which consists of taking the phone's browser and stripping off the user interface so it appears to be an app. One drawback is that the browser isn't always able to access phone hardware, such as the GPS radio or accelerometer, thus limiting what this pseudo app can do.

*This is a very valid approach if you do not need to do anything fancy*, says Vince Chavy, product management director at RADVISION, a videoconferencing vendor whose portfolio includes endpoint apps that run on Android and iOS devices. *It is easy, and it is quite amazing what you can do with HTML5 and CSS nowadays*.

Some enterprises like Web apps because they leverage their developers' existing skills. "*What we always hear is, Because my developers have Web skills, you get all that productivity and affordability across all phones*, says Blum.

The alternative is a *native app*, whose benefits include better performance because the software doesn't have to pull everything from the Web. If the target audience is made up of customers rather than employees, then another benefit is that native apps can be distributed through app stores.

*You will be closer to the look and feel of the device*, says Chavy. *You are only limited by your imagination and the OS SDK APIs*.

## Write Once, Run Many

The big downside to native apps is that code for one OS version can't be reused to create other OS versions. The corollary to that requirement is that if the enterprise doesn't have developers versed in all of the major codes, it has to find them, which increases costs and lead time. *You need a different code base for iOS and Android*, says Chavy. *You need to find Objective C developers for your IOS client and Java/Android developers for the Android client. Those are tough to find!*

Hence the growing selection of tools that enable cross-platform app development, such as MoSync, PhoneGap and Rhodes.

*You develop in C/C++*, says Henrik von Schoultz, MoSync co-founder and vice president of marketing and sales. *We will later also use other languages, like Java Script and Lua. The developer uses MoSync APIs, and at build time we use an intermediate language and have profiles for your target devices. We compile native to the OS you want to target.*

*If a feature does not exist on an OS, the system tries to handle that. For example, if the target device doesn't have GPS, you may use positioning from the cell towers*.

So how much time could a developer reasonably expect to save by using MoSync versus developing from scratch for each OS? It depends, but in one case, a developer went from Symbian to Android in about four hours. *If you have an app with loads of logic and not so much UI, you can save up to 80 percent*, says von Schoultz. *But if it's a very UI-intense app, you may save maybe 30 percent*.

PhoneGap, meanwhile, emphasizes the importance of having both apps and a mobile-friendly website.

*They can use largely the same code in their mobile website as their native app in PhoneGap because we're just running HTML and Java Script right there*, says Andre Charland, president and co-founder of Nitobi, PhoneGap's creator. *We're compliant with W3C APIs. You're also future-proofing your app because as the mobile browsers evolve, you can take your PhoneGap applications and push more of that into the browser, while still augmenting the native app experience with native code via plug-ins from PhoneGap*.

## One Size Doesn't Fit All

Tablets and smartphones come in an ever-increasing range of screen sizes and resolutions. As a result, developers also have to ensure that their app looks and works just as well on a 3.5-inch display as a 4.3-inch display, or on both an Android phone and an Android tablet.

*The first decision is if you want the same GUI on the phone and on the tablet,* says Chavy. *Some developers decide to have the same UX. In that case, they just make sure that when they design the GUI, it can resize nicely by deciding on which area resizes and by setting proper anchors on the objects*.

The catch is that consumers and business users increasingly expect tablet apps to take advantage of the extra screen space.

*For example, in our SCOPIA Mobile application, on the phone, you can see the video or the presentation, and not both at the same time*, says Chavy. *On the tablet, since you have a bigger screen, we have layouts with both the video and the presentation*.



**TIM KRIDEL FOR INTELLIGENCE IN SOFTWARE**

special report

# The Billion Dollar Lost Laptop Problem




Photo: @iStockphoto.com/studiocasper

Every time a business laptop is lost or stolen, an organization takes a direct cost hit. But how much of a hit might surprise you. What would your organization do if it realized that each year it's losing millions of dollars in this way? Odds are, it would be far more diligent in protecting laptops.

Last year, the Ponemon Institute released a study (conducted independently and sponsored by Intel) of The Billion Dollar Lost Laptop Problem, an independent benchmark of 329 private and public-sector U.S. organizations – ranging in size from less than 1,000 to greater than 75,000 employees and representing more than 12 industry sectors – to determine the economic cost of lost or stolen laptops. What they found: The cost is huge.

Participating organizations reported that in a 12-month period 86,455 laptops were lost or otherwise went missing. That added up to 263 laptops per organization on average.

According to an earlier Ponemon Institute study (conducted independently and sponsored by Intel), The Cost of a Lost Laptop, the average value of a lost laptop is a staggering $49,246. This value is based on seven cost components: replacement cost, detection,

forensics, data breach, lost intellectual property costs, lost productivity and legal, consulting and regulatory expenses. It's important to point out that the smallest cost component is the replacement cost of the laptop.

Some of the salient findings from The Billion Dollar Lost Laptop Problem report:

- The total economic impact for 329 participating companies is $2.1 billion, or on average $6.4 million per organization.
- Out of the 263 laptops per organization that are lost or go missing, on average just 12 laptops were recovered.
- Forty-three percent of laptops were lost off-site (working from a home office or hotel room); 33 percent lost in transit or travel; and 12 percent were lost in the workplace.
- Twelve percent of organizations said they don't know where employees or contractors lose their laptops.
- Although 46 percent of the lost systems contained confidential data, 30 percent of laptops lost had disc encryption, 29 percent had backup, and just 10 percent had other anti-theft features.
- Industries that experience the highest rate of laptop loss are education and research; health and pharmaceuticals were next, followed by the public sector. Financial services firms had the lowest loss rate.
- Laptops with the most sensitive and confidential data are the most likely to be stolen. However, these laptops are also more likely to have disc encryption.
- Average loss ratio over the laptop's useful life is 7.12 percent. That means more than 7 percent of all assigned laptops in benchmarked companies will be lost or stolen.

## But Who's Minding the Data?

Not nearly enough organizations, it appears. Given the significant financial impact of missing laptops and the vulnerabilities of stolen laptop data, it is astonishing that the majority of these companies aren't taking even basic precautions to protect them.

The worst cost component is the data breach. A stolen laptop can be easily booted to reveal passwords, stored temporary files the user was even unaware of, and access to VPN connections, remote desktops, wireless encryption keys and more.

That's enough reason to do something. Here are your best options for protecting your organization's data integrity against all of that potential mayhem.

- Full Disk Encryption: Full disk encryption prevents unauthorized access to data storage. Under this scenario, nearly everything is encrypted, and the decision of which individual files to encrypt is not left up to users' discretion. But all too often, end users choose to disable the full disk encryption, probably because they incorrectly assume it significantly slows all of the processing.
- Anti-Theft Technology: Laptops can disable themselves, when the hardware observes suspicious activity, if they get lost or stolen. When the laptop is recovered, it can be easily reactivated and returned to normal operation.
- Data in the Cloud: Keeping sensitive material off your laptop by storing data in the cloud is not a viable solution, because that does nothing to protect the data. Such data is easily accessible by simply cracking the login credentials. Worse yet, the existence of a full backup actually increases the cost of a lost laptop, because backups make it easier to confirm the loss of sensitive or confidential data, resulting in greater expense from forensic diagnosis and recovery efforts.

Just like Smokey the Bear says about you and forest fires, only you can stop data loss.

**BY THE EDITORS OF INTELLIGENCE IN SOFTWARE FOR INTELLIGENCE IN SOFTWARE**

## Sencha Jump-Starts Mobile Brand Advertising with Flash-free Sencha Animator

Designers and agencies need to create rich, smooth and engaging ads that will run on the widest array of handsets/tablets, and that can be developed efficiently using familiar tools and with no change to their current workflow.


Until now, rich media advertising on the desktop web has been enabled by the Flash plugin. But with no Flash support on iOS, interactive designers have been looking for a way to bring animations and rich interactivity to mobile devices without hand-coding hundreds, if not thousands, of lines of JavaScript, CSS3, Canvas, or native code.

To solve this problem, Sencha has launched Sencha Animator, the first professional tool for creating CSS-based animations that run „plugin-free" on a wide variety of browsers and devices. This disruptive technology finally makes it possible for marketers and agencies to deliver cross-platform, immersive animations that run reliably in browsers and within apps, on websites, and across mobile ad networks.

You can read more about it here: *http://pitch.pe/177733*.

## FairFX Competition – L1,000 Prize for App Developer

FairFX.com, an online currency specialist, is offering a L1,000 prize for the winner of their *build an app* competition. There are also three other L200 prizes for those that come up with the most original, most incredible, most innovative, or downright ridiculous twists outside of their brief as well.

FairFX, provide prepaid cards for travellers to carry their spending cash abroad on and are looking for either an iPhone or Android app for their cardholders to log in and use.

Jules Bean, Technical Director at FairFX said *For a while now our customers have been asking if some of our account management functions could be performed by a smartphone app, however just being able to perform these functions is not enough to stand out in the smartphone market. This is why we are running this as a competition. We want people's creativity, new ideas and their passion for problem solving*.

In order to win the main prize, the app needs to fill minimum criteria of being able to:

- Retrieve the current balance
- Retrieve the last 5 transactions in order
- Make a top up to the card
- Show the current rates on FairFX cards.

The winner of the FairFX competition is likely to be bought on board as a contractor to make their app cross compatible, as well as winning the initial L1,000 prize.

If you are interested email *rob.taylor@fairfx.com* for your starter kit.

The competition is running from 3rd October to 31st November and full terms can conditions can be found here (*http://www.fairfx.com/appcompetitionterms*).

## RIA Unleashed FITC 2011

FITC is VERY excited to be involved with RIA Unleashed! Guests and speakers from New England and throughout North America will be coming to the 5th Annual RIA Unleashed event this October. This popular, low-cost event is accessible to every budget, and last year's event was a super success. Grab your ticket early to ensure your place within the thought leaders in RIA development. RIA Unleashed has some of the most unique and engaging presenters from the Rich Internet Application industry.

- One full day of workshops, and one full day of presentations. Come for one or both!
- Over 20 presentations and 10 workshops

### What will be covered?

Future Features in Flash Player/AIR • Optimizing Game Performance for Web, Mobile, and Desktop • HTML 5 Gaming • Customizing Mobile Flex • Introduction to jQuery • Using AmplifyJS / jQuery • Brass Monkey • Optimize your flash apps • Flex Profiling for Sissies • Introducing Robotlegs 2.0 • Flash & Flex Facebook Applications • Getting social with Flex • Data-Intensive PHP Applications using Flash Builder for PHP • Mobile Flex Skinning • Developing Performant Flex Components for Desktop and Mobile • Mobile Games: Adobe AIR and Corona • Physical Computing + Android • Writing Reusable jQuery • Flex Mobile

### Who is speaking at RIAUnleashed 2011?

Terry Ryan • Ryan Stewart • Jesse Warden • Christophe Coenraets • Mike Labriola • Michelle Yaiser • Charles Schulze • Jesse Freeman • Raymond Camden • Doug Neiner • Chris Allen • Keith Peters • Elad Elrom • Rob Rusher • Joel Hooks • Veronique Brossier • Ryan Canulla • Jeff Tapper • Andy Matthews • Brian Rinaldi • Scott Janousek • Holly Schinsky

### What's it all about anyway? Why should I go?

It starts with the presentations. Presenters have been hand-picked for their skill and talent, and all are eager to share and meet you. Most times, you will have different presentations to choose from. It's also about the networking, being able to meet and talk to people who share your passion for this industry. Meet them at the presentations, in the exhibitor hall, or at an evening party.

## The Ultrashock Creative Mega Bundle

The Ultrashock Creative Mega Bundle will expire soon, so make sure to check it out now!

Stock up on over 1K creative items (2GB total size): vectors, images, audio, flash, fonts, Photoshop files, and vouchers with great value! Items by SlideShowPro, Go Media, Influxis, Electric Rain, Miniml, Amayeta, Multidmedia, Onyro, BrushLovers, Ultrashock and more!


*http://ultra.sh/megabundleffdmag*

Get the following 22 items for free, if you pay via a tweet or Facebook wall post!

7 Digital Vector Circles + 5 Neurological Complex JPGs by Onyro,

4 Pixel Fonts by Miniml,

6 Abstract Smoke JPGs by Ultrashock.

Pay on Twitter via a tweet:

*http://ultra.sh/mbtweet*

Pay via a Facebook wall post:

*http://ultra.sh/mbfacebook*

# Starting ActionScript Programming

In this new series, Huw Collingbourne explains the fundamental features of ActionScript programming. If you are new to ActionScript, this is the place to start!

### What you will learn…

- The essential features of an ActionScript program
- How to use data types and variables
- How to concatenate strings, add numbers and display output

### What you should know…

- How to use a Flash or ActionScript editor or IDE
- How to write ActionScript code into a code file
- How to compile (or publish) and Debug a program

Usually this column deals with quite specialist topics related to Flash-based development aimed at experienced ActionScript programmers. But what about less experienced programmers? If you are not already familiar with another Object Oriented language such as Java or C#, you may find it hard to get to grips with ActionScript. This month I am starting a new series of tutorials aimed at helping newcomers to ActionScript to master the fundamental features of the language. In future columns, I'll explain the inner workings of object orientation. This month, however, I want to go right back to basics by explaining how to write a couple of simple ActionScript classes to display messages and do simple calculations.

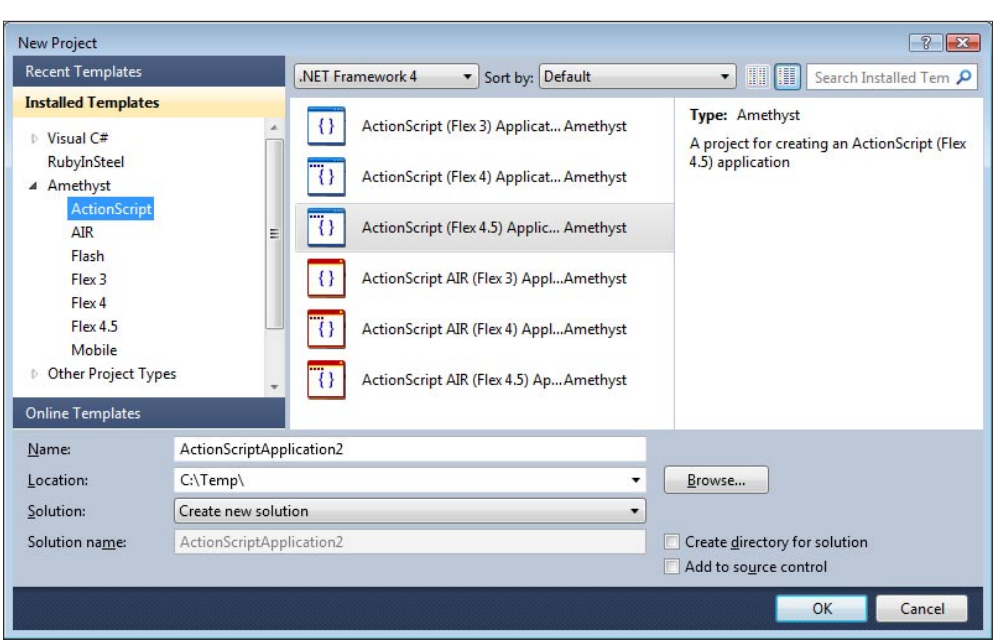



**Figure 1.** *Creating A Project*
*In Amethyst you should create an ActionScript project for whichever version of the Flex SDK you have installed*

## Getting Started

In order to begin programming in ActionScript you need to have a suitable editor or an *Integrated Development Environment* (IDE). I will assume that you will be using an IDE such as Adobe's Flash CS5 or Flash Builder or SapphireSteel Software's Amethyst.

If you are using Amethyst or Flash Builder you should being by creating an ActionScript (not a Flex) application. In the Flash IDE, you should create a Flash\ActionScript 3.0 application (Figure 1). While the programming techniques explained in this series are applicable to Flex too, my code samples do not require that you have Adobe's Flex framework installed.

Some users of the Flash IDE (e.g. Flash CS5) may be used to writing bits of program code in the Actions editor and attaching that code to specific frames in the timeline. We won't be doing that in this series. While that type of programming may be handy for coding simple things, it is not recommended when developing complex applications. For large applications, it is much better to save all your code in external files on disk. Once you've done that you must specify the main code file by entering its name as the *class* in the Properties panel of the Flash IDE. For example, if the code file is called *Text01.as* and it is in the same directory as Flash's FLA file, you would enter *Test01* as the class name. Be careful to use the same mix of upper and lowercase letters as you use for the file name itself. ActionScript is case sensitive. If you write a class called *Test01*, its file name must be *Test01.as*. Any reference you make to *test01* or *TEST01* will fail because the case of some of the letters in the name is different.

The nitty-gritty details of using a specific editor or IDE are beyond the scope of this tutorial. I will assume that you have at least a basic familiarity with the editing and compiling (or *publishing*) features of your chosen IDE.

In this first tutorial, we won't use any visual design tools. Instead, all output will be shown in the Output window or Console. In order to view that output in Amethyst or Flash Builder, you must be sure to Debug (not simply to *Run*) each program.

## Hello trace()

I'm going to begin with a simple `"Hello world"` program (Listing 1). This aims to do nothing more than to display the words `"Hello world"` in the Output window or Console. For now, the only bit of this program that is of interest to us is this line:

```
trace( „Hello world" );
```

This calls a built in routine or *function* named `trace()`. The `trace()` function simply displays some text in the Output window of your development environment When you debug the program above, this is what you should see:

```
-------------------------------------------------------
Hello world
-------------------------------------------------------
```

Let's look more closely at that line of code. This is how you run the `trace()` function:

```
trace();
```

**Listing 1.** *A simple Hello World Program*

```
package {
    /* ************************************************
     * SapphireSteel Software http://www.sapphiresteel.com
     * ActionScript sample program
     *
     * How to use trace() to display simple debugging output
     * ************************************************/
    import flash.display.*;

    public class Test01 extends MovieClip {

        public function Test01( ) {
            trace( "Hello world" );
        }
    }
}
```

Here, trace is the name of the function. When we run it, we *call* the function by its name. The semicolon (;) is a marker that separates one piece of code, such as a function-call, from another piece of code. For example, if you want to call `trace()` twice you could write this:

```
trace();trace();
```

In fact, it is generally better – clearer – to put separate bits of code onto separate lines like this:

```
trace();
trace();
```

The pair of parentheses are required when calling a function. In the case of the `trace()` function, we want to display some text. In order to do that we put the text between the parentheses like this:

```
trace( „Hello world” );
```

In programming terms, a piece of text is called a *string* – because it's a bunch of characters, `'H','e','l','l','o'` and so on, that are *strung* together. In ActionScript, a string must be enclosed by a pair of double or single quotes.

## Packages, Comments and Braces

While I've focused on a single line of code, the complete ActionScript program contains quite a bit more than that. Here is the brief overview of what it all means:

```
package {}
```

A package is like a storage area to contain your source files. Usually packages live in named folders or directories on disk. If I happened to keep my source code in a directory called `mycode`, my package would be called `mycode` too, like this:

```
package mycode {}
```

Here package is a keyword. A keyword is a *reserved word* – an identifier that means something special in the ActionScript language. As this package is in the top-level directory of my program it doesn't have a name. Notice that the keyword `package` is followed by a pair of curly braces. The opening curly brace comes right after the keyword `package`. The closing curly brace is the last character in the program. After the opening curly brace comes this:

```
/* ****************************************************
 * SapphireSteel Software http://www.sapphiresteel.com
 * ActionScript sample program
 *
 * How to use trace() to display simple debugging output
 * ****************************************************/
```

This is a comment, which is a block of documentation. Comments are entirely optional – they are inserted to explain something about the program. The ActionScript compiler ignores comments. A comment-block begins with the characters `/*` and ends with `*/`. This is sometimes called a C-style comment as it is widely used in the C programming language. You can also add single-line comments after two slash characters: `//` Everything after `//` on a single line is a comment.

```
// This comment runs up to the end of this line
```

After my comment, comes this:

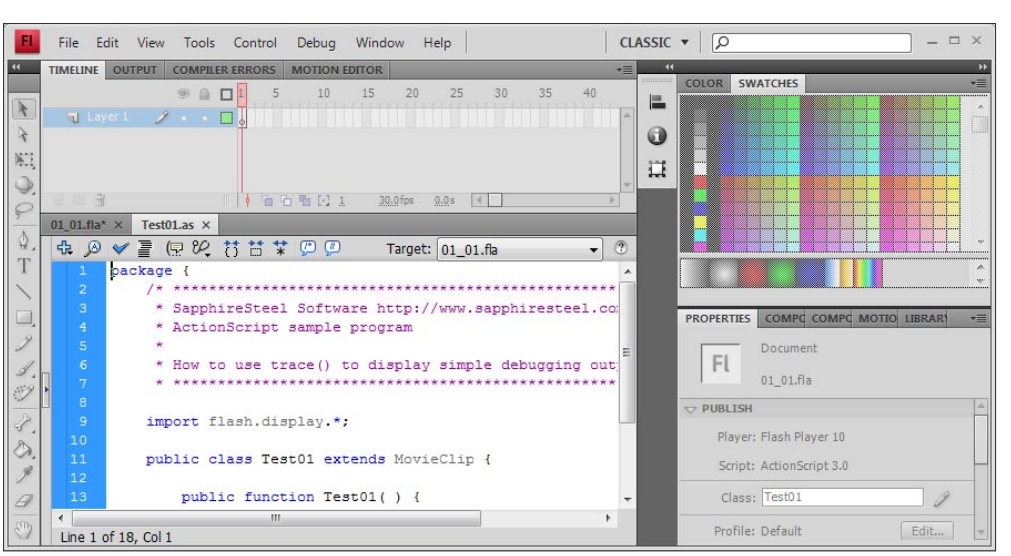



**Figure 2.** *The Flash IDE Class*
*If you are using the Flash IDE, be sure to set the 'class' name in the Properties panel to match the class name in the ActionScript code file*

```
import flash.display.*;
```

Here `import` is another keyword and `flash.display` happens to be the name of one of Adobe's *ready-to-run* packages; one that is widely used by Flash programmers. The asterisk after its name indicates that we want to import everything from that package. The package happens to contain the definition of MovieClip which is one of the core Flash *classes* that lets us display our programs on screen. I want my program to be a type of MovieClip which is why I need to import the MovieClip class. If I didn't import it, the ActionScript compiler wouldn't be able to figure out what a MovieClip is and it would display an error message. Next comes this:

```
public class Test01 extends MovieClip {

}
```

This is how I tell ActionScript that my program, which I've named *Test01*, is a type of (that is, it *extends*) MovieClip. The program here is defined as a *class*. Real-world programs may contain many different classes but, for now, I have just one class. Pay attention to those curly braces. The start and end of the *Test01* class are marked by the opening and closing braces. Inside the opening and closing braces of the class is this *function*:

```
public function Test01( ) {
    trace( „Hello world” );
}
```

A function is just a named chunk of code. We'll look at functions next month. The function here has the same name as the class, *Test01*. That is because it is a special function – a *constructor* – that runs when the program itself runs. If this is all new to you, you don't need to concern yourself with the details at this point. I'll be talking a lot about classes and their constructors later on.

## Strings

A program that displays "Hello world" may be simple to write but it certainly isn't particularly useful. Its data (here the string, "Hello world") is used just once and it is then immediately forgotten. It is often more useful to be able to store data so that the same data item can be reused later on. In order to store data we use variables.

A variable takes the form of a name or identifier. For example, I might create a variable with the name *mycat* and another variable with the name *yourcat*. The variable names must be declared before they can be used. To declare a variable you enter the keyword var followed by a name of your choice, then a colon, then the data *type* of the variable and finally a semicolon. Here are my two String variables:

```
var myCat: String;
var yourCat: String;
```

Now I need to assign values to those variables. I do that with the equals sign which is called the assignment operator. The assignment operator assigns the value to its right to the variable to its left.

```
myCat = „Flossie”;
yourCat = „Tiddles”;
```

So now I have a variable called *mycat* whose value is *Flossie* and a variable called yourcat whose value is *Tiddles*. I can use `trace()` to display the values of the variables:

```
trace( myCat );
trace( yourCat );
```

And this is the output:

```
------------------------------------------------------
Flossie
Tiddles
------------------------------------------------------
```

**Listing 2.** *Appending Strings*

```
package {

    import flash.display.*;

    public class Test02 extends MovieClip {

        public function Test02( ) {
            var aGreeting : String;
            var aName : String;
            var aPersonalGreeting: String;
            aGreeting = 'Hello';
            aName = "Frodo";
            aPersonalGreeting = aGreeting + " "
                + aName;
            trace( aPersonalGreeting );
        }
    }
}
```

The values of variables can, as the name suggests, vary. That is, having assigned a value to a variable, I can subsequently change its value. For example, I can assign a different name to the mycat variable:

```
myCat = „Felix”;
```

Now I call `trace()` to display its value:

```
trace( myCat );
```

And this shows that the `myCat` variable now has a new value:

```
------------------------------------------------------
Felix
------------------------------------------------------
```

You can think of a variable as a sort of empty box. The box has a label such as, for example, `myCat` or `sparecash` and into that box I can put different values such as *Flossie* or 100. The variable name always stays the same but its value can change throughout my program. However, while the variable *value* can change its *type* must stay the same!

Consider the variable named `aGreeting`. I declare it to be a String variable and I assign the string “Hello” to it:

```
var aGreeting : String;
aGreeting = „Hello”;
```

I can now change this value to a different String such as *Goodbye* if I wish, but I cannot change it to some other type of data. If I assign a number to it, for example, the ActionScript compiler shows an error message:

```
var aGreeting : String;
aGreeting = „Hello”;
   aGreeting = „Goodbye”; // This is OK
   aGreeting = 100;       // This is an Error!
```

## Data Types

Each bit of data – such as the number 1 or the word “hello” – has a specific data type. A whole number such as 1 or 10 is an integer. A number such as 1.5 or 10.7 is a floating point number. A piece of text such as ‘hello’ or „How are you?” between a pair of single or double quotes, is a String.

In many programming languages you are obliged to declare the type of each variable before it is used. In some cases, ActionScript allows you to omit the type declaration, but this is not (usually) good programming practice. The ActionScript compiler – the tool that translates your source code into a runnable program – may warn you if you fail to specify a type. Note that once you a declare variable to be a String it can never suddenly become an integer. Similarly an integer or floating point variable cannot become some other type of variable such as a String. That would be an error and your program wouldn’t run.

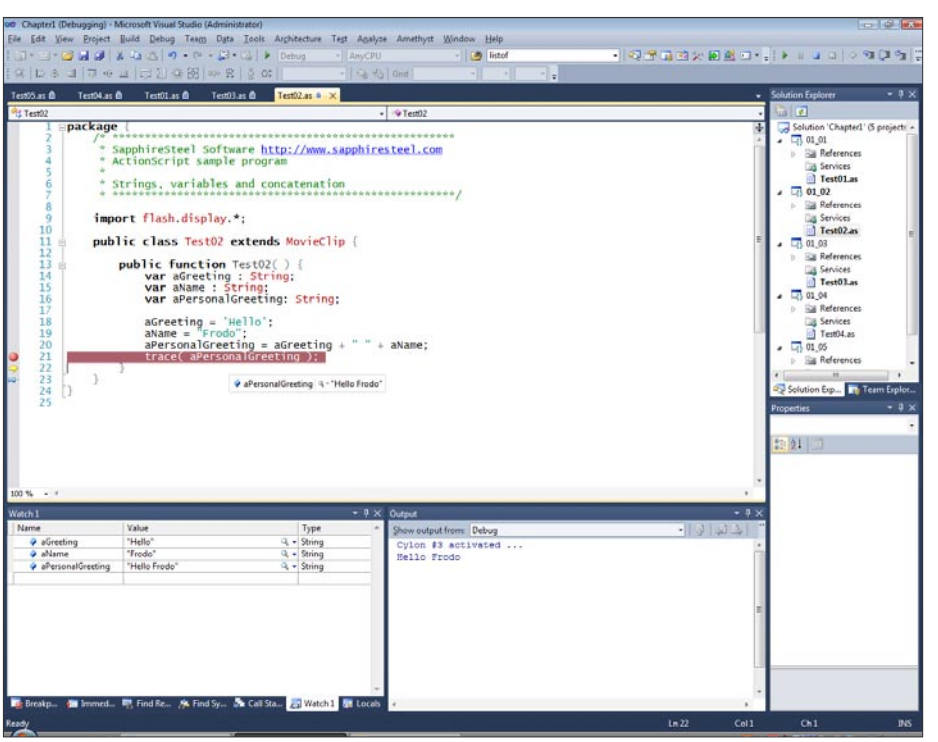



**Figure 2.** *Output and Debug*
*Use the Amethyst debugger to examine variables in the Watch window and view trace() output in the Output window*

This is an error. The `aGreeting` variable is a String but I am trying to assign an integer value to it:

```
var aGreeting : String;
aGreeting = 10;
```

This is correct. A declare `aGreeting` to be a string variable and I assign a string to it:

```
var aGreeting : String;
aGreeting = „Hello”;
```

In my next program (Listing 2) I declare three string variables, `aGreeting`, `aName` and `aPersonalGreeting`. The code assigns a string to each of the first two variables. It then concatenates the `aGreeting` variable, a single space character, “ ”, and the `aName` variable using the + concatenation operator. The three concatenated strings form a new string: “Hello Frodo”. This new string is assigned to the variable `aPersonalGreeting` and the value of this variable is displayed by `trace()`.

## Numbers

Look at Listing 3. In essence, this is very similar to the last program which created two string variables, concatenated them and assigned them to a third string variable. This time, however, the variable data types are numeric. In ActionScript, the Number data type is compatible with both integer numbers such as 100 and floating point numbers such as 100.25. If you specifically want to use only integers, you could specify the variables as `int` or `uint` like this:

**Listing 3.** *Number variables*

```
package {


    import flash.display.*;

    public class Test03 extends MovieClip {

        public function Test03( ) {
            var value1 : Number;
            var value2 : Number;
            var value3 : Number;
            value1 = 10;
            value2 = 15;
            value3 = value1 + value2;
            trace( value3 );
        }
    }
}
```

```
var value1 : int;
var value2 : uint;
```

To be strictly accurate, the ActionScript Number class defines a double-precision, floating-point number. The `int` class lets you work with the data type representing a 32-bit signed integer with a range of values from -2,147,483,648 (-2^31) to 2,147,483,647 (2^31-1). The `uint` class provides methods for working with a data type representing a 32-bit unsigned integer. An unsigned integer can only be positive with a range of values from 0 to 4,294,967,295 (2^32-1). Typically, *ints* are used for loop counters and signed integer calculations. The uint type is useful representing for things such as RGB colour values and byte counts. The default value of a Number variable is NaN (*Not a Number*). The default value of a `int` or `uint` variable is 0.

For the sake of simplicity, I’ve decided to use Number rather than `int` or `uint` here. This + operator works differently for numbers than for strings. Here it adds the two values to its left and right and returns a total. I then assign this total to the `value3` variable and display it with `trace()`:

```
var value1 : Number;
var value2 : Number;
var value3 : Number;
value1 = 10;
value2 = 15;
value3 = value1 + value2;
trace( value3 );
```

As `value1` and `value2` have been assigned 10 and 15, it will come as no surprise that value3 is here assigned 25.

In next month’s column I’ll be looking at ActionScript’s Object Orientation.


## HUW COLLINGBOURNE

***Huw Collingbourne is Director of Technology at SapphireSteel Software. Over more than 20 years, he has programmed in languages ranging from Ruby to C# and has been a technical columnist for computer magazines such as PC Pro and PC Plus. He is the software lead of the Amethyst Designer, the Flex user interface design environment of the Amethyst IDE for Visual Studio. SapphireSteel Software: http://www.sapphiresteel.com/***

# The Union Platform
## Integrating JavaScript and Flex Clients

Social gaming is all about interacting, and sometimes you have to adapt how you communicate.

**What you will learn…**
- OrbiterMicro client framework
- Union Platform and Reactor basics

**What you should know…**
- Flex / Actionscript 3
- Javascript

I used to enjoy crowding around a table with my friends every week to play hours of various role-playing games, but we've all spread apart, got families, and just don't have the time any more to devote to such an activity.

However the desire to create worlds and have adventures still exists. This inspired me to start writing a completely customizable MMORPG that runs in the browser. I quickly settled on the Union Platform (*http://www.unionplatform.com*) as the real-time connective glue between clients and server. Union's Reactor client framework for Flex abstracted away the distributed messaging headaches, and now with the OrbiterMicro framework for JavaScript I will show how easy it is to integrate different types of clients to the same Union server.

Like many other offerings, Union Platform divides itself up neatly into Rooms. A Room is just a logical grouping of clients, like topic based chat rooms, or instances of online matches of Euchre. For the purposes of my game, I made each map area into its own room. This meant the players' characters could communicate with each other when they were within the same map area, but were unable to talk across distances. For that there needed to be some sort of a global channel. Although Union provides the ability to send messages across many rooms, it also allows users to join multiple rooms at once. For me, the cleaner solution was to add a *global* chat room that everyone automatically joins.

This approach has another benefit. It makes it easy to add a JavaScript chat client, so that players can talk

**Listing 1.** *Making a Connection*

```
function init () {
  orbiter = new net.user1.orbiter.Orbiter();
  orbiter.addEventListener(net.user1.orbiter.OrbiterEvent.READY, readyListener, this);
  orbiter.connect("myserver", 9100);
}
function readyListener (e) {
  alert("Connected.");
}
```

without requiring Flash or having to enter the game. Let's start with that chat client. The Union Platform has an excellent tutorial on their site showcasing how to create a basic JavaScript chat using the OrbiterMicro client framework. I used this as the basis for my chat client; however I wanted to add the ability to log in and out, and to show how many people are online.

Connecting to a Union server using `OrbiterMicro` takes very few lines of code. For example: see Listing 1.

The `OrbiterMicro` tutorial shows how to turn that example into a full chat client, but I will jump ahead and start adding the code to tie it to my game. Since it's handy to maintain a single page for the chat, I will

**Listing 2.** *Init login listener and handler*

```
  // Register for incoming messages from Union
  msgManager = orbiter.getMessageManager();
  msgManager.addMessageListener(UPC.LOGIN_RESULT,
                    loginListener, this);
function loginListener (fromUserID, status) {
  if (status == "SUCCESS") {
    displayChatMessage("Logged in as "+fromUserID);
    // Join chat room
    joinRoom();
    // hide login
    hideLogin();
  } else {
    displayChatMessage("Login:"+status);
  }
}
```

**Listing 3.** *Init logout listener and handler*

```
  msgManager.addMessageListener(UPC.LEAVE_ROOM_
                    RESULT, leaveRoomListener, this);
function logout() {
  // Hide text input
  hideInput();
  // Announce your departure
  msgManager.sendUPC(UPC.SEND_MESSAGE_TO_ROOMS,
                    CHAT_MESSAGE, roomID,
    "true", "", username+" left");
  msgManager.sendUPC(UPC.LEAVE_ROOM, roomID);
}
function leaveRoomListener(fromRoomID, status) {
  if (status == "SUCCESS") {
    // Close the connection
    orbiter.disconnect();
  } else {
    displayChatMessage("Leave Room:"+status);
  }
}
```

**Listing 4.** *Joining the Global Chat from Flex*

```
private function loggedIn(event:AccountEvent):void
{
if (event.getAccount()!=null) {
    trace(event.getAccount().getUserID()+" has
                    logged in");
    // Join global chat room, creating it if it
                    doesn't exist
    var settings:RoomSettings = new RoomSettings();
    settings.removeOnEmpty = true;
    globalChatRoom = reactor.getRoomManager().create
                    Room("chatRoom",settings);
    globalChatRoom.join();
  }
}
```

**Listing 5.** *Tying together Global Chat with Chat Manager*

```
    // Hook into global chat room and announce
                    presence
    globalChatManager = new ChatManager(Applicatio
                    n.application.globalChatRoom,
                    chatInput, chatPanel, player.get
                    Account().getUserID());
    globalChatManager.sendMessage(player.getAccount(
                    ).getUserID()+" is online");
```

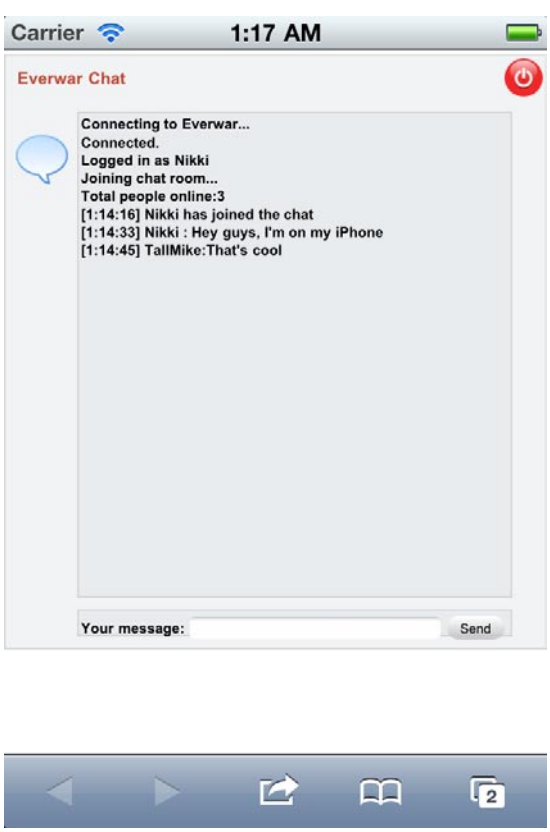



**Figure 1.** *javascriptChat*

use CSS and manipulate the styles with JavaScript to hide and show the appropriate components. What is of interest is adding authentication to the client.

The first step is to change the readyListener function.

```
function readyListener (e) {
  displayChatMessage(„Connected.”);
  // show the login field
  showLogin();
}
```

Now the user is presented with a prompt for credentials when a connection is established. Once submitted, those credentials can be passed to the server. Note that I've scoped the username variable here so that it can be referenced by other functions. That's not necessary, but Union typically references clients by their session name rather than account name.

```
function login () {
  username = document.getElementById(„username”).value;
  var password = document.getElementById(„password”).v
                  alue;
  msgManager.sendUPC(UPC.LOGIN, username, password);
}
```

The server will respond with the results, so there needs to be a listener added to the init and a new function to handle the server response (see Listing 2).

`joinRoom()` just adds the user to the global chat room in the same manner as shown in the `OrbiterMicro` tutorial. Now, to show how many people are online when someone logs in, it's as simple as adding another listener and function.

```
msgManager.addMessageListener(UPC.ROOM_SNAPSHOT,
                  roomSnapshotListener, this);
```

**Listing 6.** *Flex Chat Manager*

```
public class ChatManager
{
  public static const CHAT_MESSAGE:String = "CHAT_MESSAGE";
  private var chatPanel:ChatIncomingTextArea;
  private var chatInput:TextInput;
  private var charName:String;
  private var room:Room;

  public function ChatManager(room:Room, chatInput:TextInput, chatPanel:ChatIncomingTextArea, charName:String){
    this.room = room;
    this.chatInput = chatInput;
    this.chatPanel = chatPanel;
    this.charName = charName;
    room.addMessageListener(CHAT_MESSAGE, chatMessageListener);
  }

  private function chatMessageListener(fromClient:IClient, chatText:String ):void {
    chatPanel.displayMessage(chatText);
  }

  public function submitChat():void {
    sendMessage(charName+":"+chatInput.text);
    chatInput.text="";
  }

  public function sendMessage(message:String):void {
    room.sendMessage(CHAT_MESSAGE, true, null, message);
  }
}
```


**Figure 2.** *inGameChat*

```
function roomSnapshotListener (requestID, fromRoomID,
                    occupantCount) {
  displayChatMessage(„Total people online:
                    ”+occupantCount);
}
```

A Room Snapshot is sent by the Union server automatically whenever you join a room. It contains a complete description of room properties and the clients in it. In this case, I've only grabbed the number of occupants. Since everyone in the game joins this room, this supplies a count of everyone online. I could list out the names here, but since this is supposed to be an MMO that list might be ridiculously long.

You'll notice the use of UPC constants throughout the code. UPC is the generic protocol that the Union Platform uses to communicate with clients. This is what allows the development of clients in any language for interacting with the Union server. The complete protocol is described on the Union Platform web site, which I found useful while expanding upon the JavaScript chat.

The last thing I've added is a logout button. Although Union is perfectly happy closing connections when the user closes the window, logging out is an opportunity to notify the other users of the departure.

As before, another listener needs to be added to the init to catch leaving the room (see Listing 3).

That's it for the JavaScript client! All that's left is to join players in the game to this global chat room and give them the means to partake in the conversation.

Since the game client is written in Flex, it uses the Reactor client framework as I mentioned earlier. While the syntax may be different, the principles are still the same.

In my *Main.mxml* I added a public variable for the room:

```
public var globalChatRoom:Room;
```

When the logged in event is triggered, I have the player join the global chat room (see Listing 4).

Finally, once the player has selected their character and enters the game, I attach the controls to allow them to participate in the global chat room. Here I use the account name instead of character name to identify the player, since being able to talk across vast distances might be considered *out of character* (see Listing 5).

The ChatManager class should look somewhat familiar. Just like the JavaScript chat, it simply submits chat messages to the room and listens for messages to dump to the display (see Listing 6).

So there you have it! By adding a single new room to the game, I've enabled global chat functionality for players in the game and supplied a single point of connection for a JavaScript chat client. This demonstrates the ease and flexibility of the Union Platform. If you're doing any kind of distributed multiuser real time application, you should definitely consider the Union Platform. It's completely free up to 1000 concurrent connections.

All of the code in this article is available in its entirety in the *everwar* open source project (*https://code.google.com/p/everwar/*) and here is the completed JavaScript chat: *https://code.google.com/p/everwar/source/browse/trunk/webChat/chat.html*.


## CHAD TERNENT

***Chad Ternent is Lead Research Developer at IntelliResponse Systems Inc. where he professionally builds applications in Flex, Java, and JavaScript. Writing games has been a hobby since the late 80's.***

# Protecting What You Don't Know Is Yours

It seems self-evident to state that you cannot protect what you don't know is yours. This is the basic principle that this article will emphasize and build upon. Although the principle applies in the case of all property, including intellectual property (IP), it is often ignored in the case of copyrights.

### What you will learn…

- Open Source: defined.
- Today's software is collaborative; developers include code from many sources, including open source code.
- All open source code carries with it certain license obligations.
- Before any code is used in a software product it is crucial to locate understand the legal obligations associated with its use.
- Common misconceptions in open source.

### What you should know…

- Copyrights are automatically applied to all software.
- Just because open source software is freely distributed doesn't mean it's not copyrighted.
- Open source licenses allow authors to maintain authorship rights such as attribution and the right to the integrity of the work.
- When it comes to other's property, it is crucial to understand the legal obligations associated with its use in order to avoid legal repercussions of using someone else's protected goods without permission.

Copyrights easily go unnoticed, because they are attached to works without the author ever asking for them. Indeed, any original and creative work automatically falls under the protection of copyright law. This protection aims to encourage the production of works of lasting benefit to the world (Washingtonian Co. v. Pearson, 306 U.S. 30, 36 (1939)) and facilitates access to them (Industry Canada and Canadian Heritage, A Framework for Copyright Reform (Industry Canada and Canadian Heritage, 2001), at *http://strategis.ic.gc.ca/pics/rp/framework.pdf*). The economic basis for granting such exclusive rights is the conviction that personal gain is the best way to encourage creators and inventors to produce and publish new works (Mazer v. Stein, 347 U.S. 201, 219 (1954)). The protection of intellectual goods prompts creators to share their work immediately rather than keeping it secret, by giving them a right of action against copycats and forgeries. The public benefits both immediately from the disclosure of the work, and later from its incorporation to the public domain at the expiration of copyright protection. In today's knowledge-based economy, the benefits of IP and its value are accentuated. IP is an increasingly valuable commodity, which is bought, sold and exchanged. Here arises the first need to identify works under copyright protection: to protect the valuable goods that we own and exclude others from illegitimately using the product of our efforts. On the other hand, when it comes to others' property, it is crucial to locate its source and to understand the legal obligations associated with its use in order to avoid the legal repercussions of using someone else's protected goods without permission.

But it's not just about protecting yourself and your property! As Daniel J. Gervais elegantly reflected, *copyright is not a dam, it is a river* (Daniel J. Gervais, Towards a New Core International Copyright Norm: The Reverse Three-Step Test, 9 MARQ. INTELL. PROP. L. REV. 1, 7 (2005)). Copyright law should be viewed is a means to direct use, rather than a tool to fend off users. Savvy IP holders will therefore optimize the exploitation of their own IP, as well as those of others, to guide the flow of information and knowledge. Copyright covers all original expressions of ideas. If it is written, drawn, said, or acted, there is copyright involved. Software, as it is written, is no exception. But software is rarely written in isolation; it is generally developed collaboratively. In our modern world of fast-paced communication and knowledge-sharing, the adage proclaiming that two heads are better than one is widened: two hundred heads are better than two… two thousand heads are better than two hundred. Therein lies the power of collaborative innovation. For programmers, communication not only avoids redundancy in efforts, but it also promotes efficiency by allowing others to

scrutinize existing code and build upon it in their own work. Open source software (OSS) is a great instrument in the exploitation of such opportunities. Generally speaking, OSS is source code that is made available by its author for free redistribution and modification (See Open Source Software Initiative, The Open Source definition, available at *http://www.opensource.org/docs/osd* (describing in detail the specific elements of open source software).). It is a common misconception that OSS carries no copyright. In reality, OSS is copyrighted (just like all original works). The distinguishing mark of OSS is the permission to freely use the work. This permission is given through an OSS license, which specifies the permitted uses as well as the associated limitations and conditions of use. Open source licenses allow authors to maintain authorship rights such as attribution and the right to the integrity of the work. The licenses also enable authors to limit their liability and share their code without fear of legal repercussions. In return, users may use the source code, while respecting the specific conditions attached to its use, such as attaching modification notifications or providing a copy of the OSS license in modified versions of the code. Open source software, when adopted and managed properly, is a bountiful resource that speeds up development, reduces product costs, and contributes to a fuller world of proven, reliable software that runs every aspect of our life today. Owners and producers of IP can undoubtedly benefit from identifying the protected works that fall within their activities. To fully take advantage of the IP system, put in place to facilitate knowledge sharing, it is essential to have an accurate grasp of the protected works which are used in a collaborative software project. Identification of the external IP can be done through an internal evaluation, or by calling on the assistance of companies that provide products and services that manage software license and copyright obligations. Don't let the fear of the unknown hold you back: protect yourself and tap into this bountiful resource… Know your code, know your IP!

**EVE HEAFEY**

***Eve Heafey, LL.M., M.Sc., is a legal consultant for Protecode (www.protecode.com) and is currently completing the National Program (LL.L.-J.D.) in the Common Law section at the University of Ottawa. She obtained her Masters of Laws (LL.M.) at Columbia Law School in New York, where she earned highest honors and was named a James Kent scholar for outstanding academic achievement. She graduated summa cum laude from the University of Ottawa, where she obtained a License in Civil Law (LL.L) as well as a Masters of Science, with specialization in nanochemistry in the Scaiano research group.***

# Anatomy of a Software Code Audit Process

Software has become a major component of products that are produced by most technology companies and is rarely written from scratch.

### What you will learn…

- A software code audit establishes code ownership.
- The earlier an audit is preformed the easier, and cheaper it is to fix any problems discovered during the audit.
- Automated code scanning is the most efficient way to conduct an audit.
- Audits produce detailed reports on what is in a software product and what obligations must be met.

### What you should know…

- Resourceful developers use third party and open source code to speed up development time and reduce development costs.
- Most open source licenses have certain legal obligations that have to be met.
- Uncertainty around code ownership can stall product launches and negatively impact M&As.
- A common mistake is to start a code audit process in the last step of a transaction.

Resourceful software development organizations and developers use a combination of previously created code, commercial software and open source software, and their own creative content to produce the desired software product or functionality. Anytime a product containing software changes hands there is a need to understand its composition, its pedigree, its ownership, and any third-party (including open source software) licenses or obligations that govern its use by its new owners.

## Avoiding Uncertainties in a Technology Transaction

Technology transactions that involve software may include the launch of a product into the market, *mergers & acquisitions* of companies with software development operations, or technology transfer between organizations whether they are commercial, academic or otherwise public. Any uncertainty around either ownership of software or compliance with the licenses associated with software can:

- deter downstream users,
- reduce ability to create partnerships,
- create litigation risk to the company and the downstream users,
- increase risk and threaten closures in funding deals,
- negatively impact M&A activities,
- increase product time to market, and
- affect company valuation.

So how can all of this be avoided?

A software code audit is a good way to determine what is in your *software product*. A software code audit should not be confused with the more common place software audit process; the latter generally has to do with making sure you have paid for the software applications (e.g. Microsoft Office) you are using in your organization. Software code audits identify building blocks (files or software modules or packages, or even five lines of external code) that are used in a product or exist in the code inventory of an organization.

The audit process establishes code ownership, licensing or copyright obligations around any third party content in the code portfolio, authorship, package versions and export restrictions. Software code audits can also highlight alignment with the policies around either use or delivery of software in a particular organization. Software code audits can also pinpoint code-reuse between different portfolios within or across organizations.

The common mistake is to only start a code audit process in the last step of a transaction. Starting the audit *in anticipation* of a transaction allows for timely correction of any shortcomings detected during the audit. You certainly do not want to delay a transaction because of uncertainties uncovered during the audit.

## What Is Software Code Audit Process?

Except in simplest, smallest code portfolios, a manual audit of a company's code portfolio takes time, is

inaccurate, and expensive. Automated code scanning solutions can sift through large portfolios quickly and efficiently, detecting outside code and retrieving licensing and other attributes of external components. While automated code scanning solutions operate very fast, there is still a human element to a thorough audit project. Our experience has shown that most of the time an audit is taken by the front-end and back-end processes.

## The software code audit process usually involves

- Establishing a legal framework (NDA) between the parties involved and the auditor.
- Question and answer between the parties involved to establish:
  - the objectives of the audit, to understand the company or product that is audited,
  - the specific business of the target companies.
  - their third party software practices,
  - the software environment that is used in the target company, and
  - their open source adoption policy (if any)

In some cases, all code that must be audited is not in one place, or must be *assembled* before an audit is carried out. Depending on the size of the project, the front-end process can take can take 1-5 days.

## Software Code Scanning and Detection

Once the legal framework is in place, the code is available, and the environment discovery process is complete, an automated scanning application is set up. The complete job is broken into logically-meaningful segments (for example, identifiable subprojects and modules), and then the actual automated code scanning is carried out. Ownership warnings generated by the automated application (such as proprietary code without appropriate headers or copyright information, or conflicting license information) are brought to the attention of the staff, and either resolved or marked for further action.

The reports created by the automated solution are reviewed by an expert audit staff, and a final executive report is assembled. Depending on the size of the audit project, this step can take as little as a couple of days (small project containing thousands of files) and up to two weeks (for a very large portfolio of hundreds of thousands of software files).

## End-Results of a Software Code Audit

The end result of a software code audit is a combination of two reports.

The first is a high-level executive overview report that is custom created by the audit staff. This report defines the software code audit environment, the process used, and the major findings, in simpe graphical and tabular format. Attention is drawn to specific packages, files or licenses. Information on commercial or open source software components, a description what each piece of software does, who created it, and related references on public-domain project websites should be provided.

Important information such as copyright owners, licenses associated with the discovered software packages, and optionally encryption or export obligations, are tabulated. The text of all licenses that are discovered is included with this report. The report lists all external content, including complete third party software files, modules or projects, or snippets of code that have a code structure similar to known open source projects. The findings of a software code audit must be verifiable; therefore references or hyperlinks to all information that is discovered would be provided. The second report is a detailed machine-generated report, listing:

- all packages, files, licenses, copyrights, etc. associated with all software files in the target portfolio, and
- optionally, a license obligation report, summarizing the obligations associated with all licenses found in the portfolio.

The detailed report can be very large, and is normally provided as a back up to the high level executive report. This report is normally consulted if a specific project or file requires further scrutiny.

## How Much Does a Software Code Audit Cost?

Generally the cost of an audit is proportional to the complexity of the project, which in turn can be roughly defined as the number of files in the target portfolio, the nature of the packages (commercial or public domain) used in the portfolio, and the information that is available about those packages. Most audits (thousands and up to hundreds of thousands of software files) fall within a $5-$40K range.

If you're planning a specific transaction involving software assets, whether it's an M&A, equity investment, product introduction, demand for IP indemnity, commercialization of research or other event, conduct a software code audit as early as possible in the transaction. Knowing what's in the code can speed up transaction times and reduce costs associated with fixing problems at the last minute.

**KAMAL HASSIN**

***Kamal Hassin, Director, R&D and Product Management at Protecode, is a thought-leader in the area of open source licensing and is the author or co-author of a number of papers on Software Intellectual Property management. Kamal has a Bachelor of Engineering degree and a Masters degree in Technology Innovation Management from Carleton University. He can be reached at khassin@protecode.com.***

# Scoreoid and the Importance of Leaderboards

Games are becoming ever more popular, especially causal games on mobile devices and tablets. As games become cross platform the importance of leaderboards is multiplied.

### What you will learn…

- How to get started with Scoreoid and some of the unique benefits that Scoreoid offers
- how to create dynamic players saves and advanced leaderboards.

### What you should know…

- Basic understanding of development and HTTP request as Scoreoid is truly cross platform it doesn't matter what coding language or tools you use.

With game development there has always been one issue that repeats itself; managing leaderboards. In most cases we may use local storage, like shared objects in Flash, to save local game data, but this is neither sufficient nor provides any inherent benefits to our game. Unfortunately, developing a full sever side solution per game isn't a recommend method as it takes away valuable time which can be spent developing your game or improving it. In reality, leaderboards are not the only game feature that requires this solution; there are also game achievements, player records and general game management. At the same time maintaining the required hosting is another cost in a very competitive market and as games become cross platform this issue is multiplied, this is where Scoreoid comes in.

## But how important are leaderboards?

There is no doubt that leaderboards are extremely important to drive retention of gamers, but the degree by which they increase retention is not easy to judge. It can be driven by various factors. There are reports, though, that leaderboards can easily increase retention time by as much as 30 percent.

In a professional environment, sales executives, managers, and entrepreneurs have their own motivational methods by which they can take their performance to the next level. For gamers, the results they achieve also act as a huge booster to make them

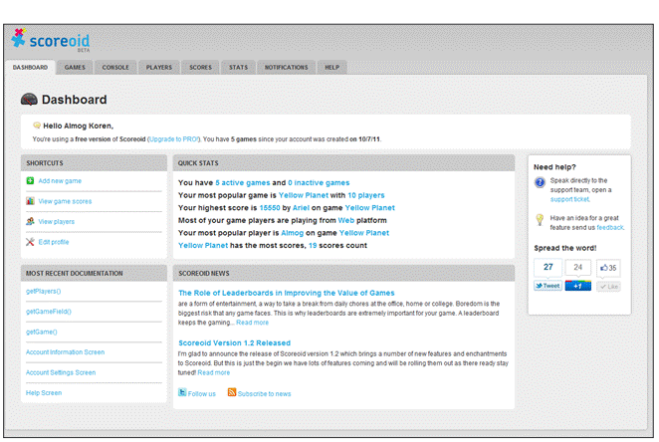



**Figure 1.** *Scoreoid Dashboard*

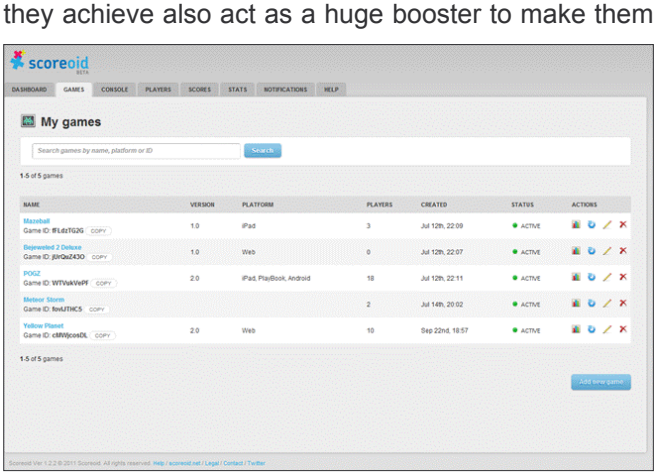


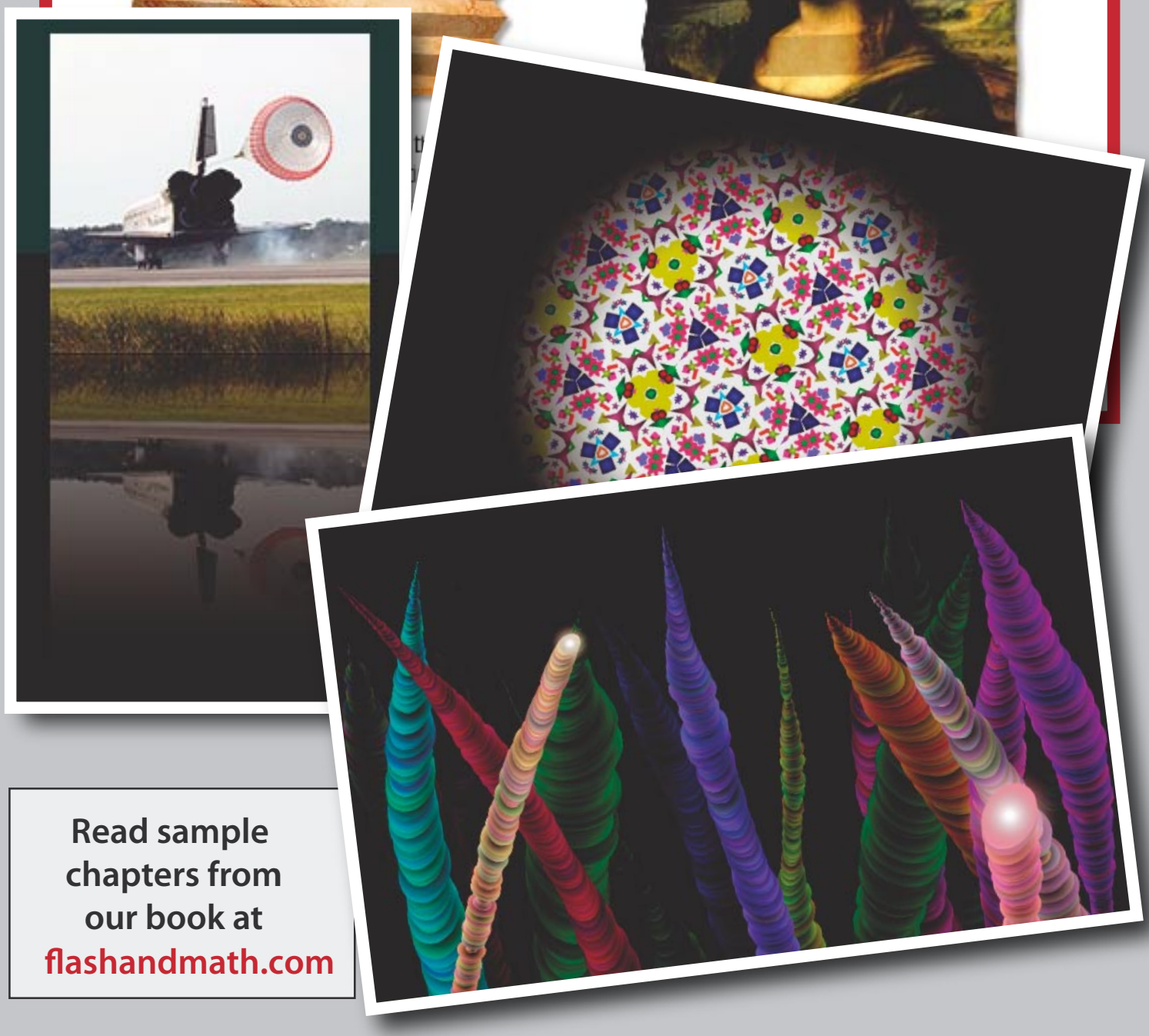

**Figure 2.** *Scoreoid Games Screen*

Flash & Math

# www.flashandmath.com

## ActionScript 3 Tutorials

## Dazzling Effects in Flash CS4 and CS5



Read sample chapters from our book at flashandmath.com

try harder and go beyond their normal abilities. In a gaming environment, the needs of gamers are met by badges, trophy rooms, and leaderboards.

Leaderboards are the tool that motivates you to reach higher scores, greater gaming levels, and better skills. The intent is to keep you fixated and interested and to make you keep coming back at every conceivable opportunity. While there is no doubt that it can be a hugely powerful tool for motivation, the intention should be to use it towards a positive objective.

## So what is Scoreoid?

Scoreoid is a non-restrictive, reliable and easy to use gaming platform designed to handle scoring, leaderboards and game management, including advanced functions for multi-platform games such as platform content awareness to advanced player management, developed by game developers for game developers.

Scoreoid's goal is to handle scoring and leaderboard functionality offering plenty of features to make games better; shortening game development time and costs, and giving developers time to work on their games.

## What makes Scoreoid unique?

Unlike similar solutions, Scoreoid is focused on providing game developers in game features to help improve their game titles and at the same time focusing on giving game developer's more time to work on their game. Scoreoid is also truly cross platform, with Scoreoid's Open Web API there is no need to download SDK's, no waiting for updates, and yes Scoreoid works on every platform.

**Listing 1.** *Here is a simple example how to add a score using AS3*

```
function createScore():void
{
    var url:String = " https://www.scoreoid.com/api/
                    createScore";
    var request:URLRequest = new URLRequest(url);
    var requestVars:URLVariables = new
                    URLVariables();
        request.data = requestVars;
        requestVars.api_key = "YOUR API KEY";
        requestVars.game_id = "YOUR GAME ID";
        requestVars.response ="XML"
        requestVars.score = 86;
        requestVars.username = "Mike";
        requestVars.platform = "iPhone";

        request.method = URLRequestMethod.POST;

    var urlLoader:URLLoader = new URLLoader();
        urlLoader = new URLLoader();
        urlLoader.dataFormat = URLLoaderDataFormat
                    .TEXT;
     urlLoader.addEventListener(Event.COMPLETE,
                    loaderCompleteHandler);

        urlLoader.load(request);
}

function loaderCompleteHandler(event:Event):void
{
    trace("responseVars: " + event.target.data);
}
```

**Listing 2.** *Here is a simple example how to get your games scores only for iPhone using AS3*

```
function getScores():void
{
    var url:String = " https://www.scoreoid.com/api/
                    createScore";
    var request:URLRequest = new URLRequest(url);
    var requestVars:URLVariables = new
                    URLVariables();
        request.data = requestVars;
        requestVars.api_key = "YOUR API KEY";
        requestVars.game_id = "YOUR GAME ID";
        requestVars.response ="XML"
        requestVars.order_by = "date";
        requestVars.platform = "iPhone";

        request.method = URLRequestMethod.POST;

    var urlLoader:URLLoader = new URLLoader();
        urlLoader = new URLLoader();
        urlLoader.dataFormat = URLLoaderDataFormat
                    .TEXT;
     urlLoader.addEventListener(Event.COMPLETE,
                    loaderCompleteHandler);

        urlLoader.load(request);
}

function loaderCompleteHandler(event:Event):void
{
    trace("responseVars: " + event.target.data);
}
```

# Save Hours Building Apps with PlugrMan

## The Universal API Toolkit

With support for **AMFPHP**, **Zend AMF** and **SabreAMF**, PlugrMan is the ideal tool for developers who need to build, work with and debug remote AMF services.

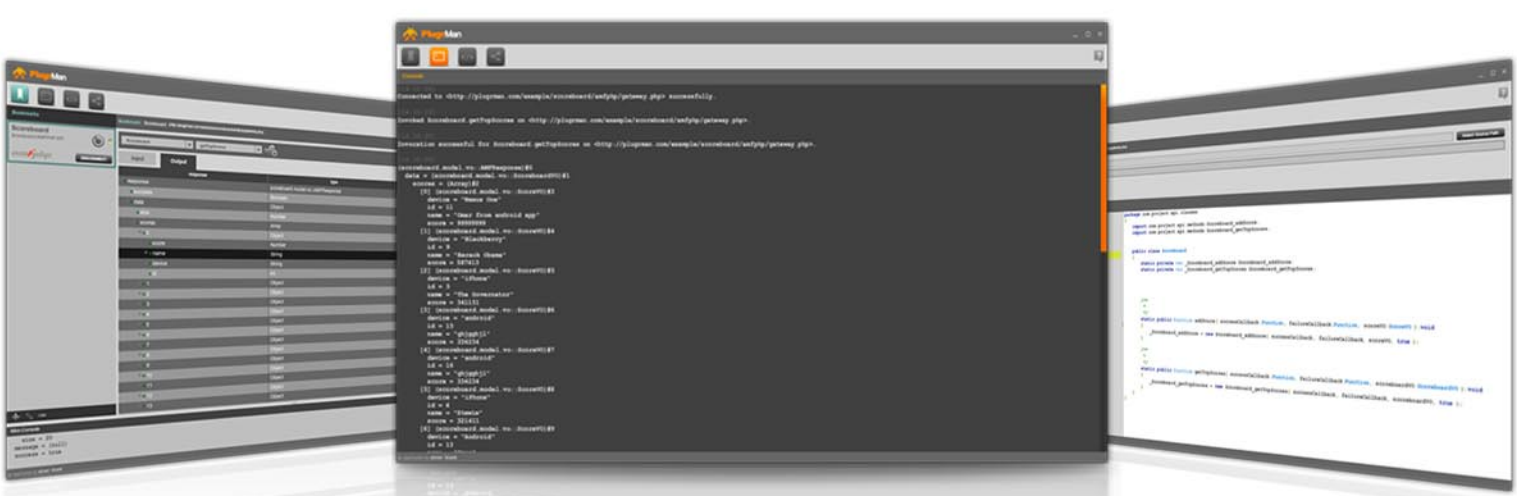
- Introspect Remote Services
- Store Test Data
- Automate Service Tests
- Generate Detailed Reports
- Generate ActionScript 3 Code for your Services

**Only $79.99!**

Get Your **FREE 30-Day Trial** at plugrman.com

Mac Windows Linux

Runs on **MacOS**, **Windows** & **Linux**!



Introspect and Test Remote Services

Generate Reports on Automated API Tests

Generate AS3 Classes for API Communication

*"It is **totally worth the price**, it will **save you time**."* - GÁBOR CSOMÁK, Flash & Flex Developer's Magazine

almer/blank laboratories

**But there is more:**

- *100% free, easy web platform* – Scoreoid is a completely free hosted solution. It includes automatic built-in upgrades, updates and new features, all of which are free.
- *Cross platform development* – Because it's cross platform you can use any game development tool you like. And multi-platform content awareness means if your game is on multiple platforms, Scoreoid can pull data based on the platform for a more competitive, fulfilling game.
- *Advanced, customizable leaderboards* – Create advanced leaderboards. Use a wide choice of options for segmenting and extracting your game data. And customize your leaderboards – our open API lets you skin data with your own UI.
- *Platform content awareness* – You can display different leaderboards depending on what platforms you're using. And players can compare scores across different platforms.
- *Advanced player management* – Manage all of your players. It's easy to see your top players and even capture potential leads for your next title.
- *In game notifications (new)* – Create dynamic in game notifications across all platforms without having to update your game, great for updating your current players on new games titles, game updates, features and much more.
- *Cut Development Time* – Scoreoid's dramatically reduces your coding burden, giving you more time to develop your game.
- *In depth game analytics and stats* – Get in depth game stats and in-game analytics without third party APIs.
- *Secure* – You don't have to worry about your data.
- *Make sure to check out the full features list at Scoreoid.net*

## Beyond the sales point; "making games better"

Scoreoid is much more than what you would expect. Using Scoreoid can not only increase your revenue but can help you create a better and more unique game. The gaming market is becoming more competitive. The more you do to make your game stand out the better your chances are. Just think about some of the possibilities you have with Scoreoid.

## Creating a brand

While reading a great blog post, *Will Make Games for Food* by Sash, one of the key points that caught my attention was building a brand. *Recognition and respect: the successes of your previous works are presumed in your latest games. People come to respect your brand and the games that you make, Sash*.

I thought this was very relevant as Scoreoid provides some unique features to help you create a brand. First, with Scoreoid you keep your player's data and information. This is great if you want to send out newsletters, updates or connect on a more personal level with your players. Scoreoid also has a built in *In Game Notification* feature which lets you set up easy to use dynamic notifications for your games. This is great for updating your current players on new games titles, game updates or features. Maybe you want to tell them about your blog or game site. You can even create promotions, raffles, and challenges. The best thing with Scoreoid's In Game Notifications is that it works across all platforms.

## Creating unique leaderboards

If you're developing games in today's market having your game on different platforms is one of the best options to increase your revenue and your games brand. With Scoreoid you can make your game very unique using the built in platform content awareness feature. You can display what platform a player is playing from, making your leaderboards very different and attractive to players showing them results based on different platform and how they compare to other players playing on a different platform.

## Easily create user saves

Everyone has played Angry Birds. You play it on your iPhone, iPad, Android and even at work on Google chrome, but what has always been my issue is having a different score across different platforms. This issue is becoming very important, since a user can play your game on the go with their mobile and then at home with their tablet why should they start over? Users are expecting to play your game everywhere but not lose their place or score; especially if they paid extra for the

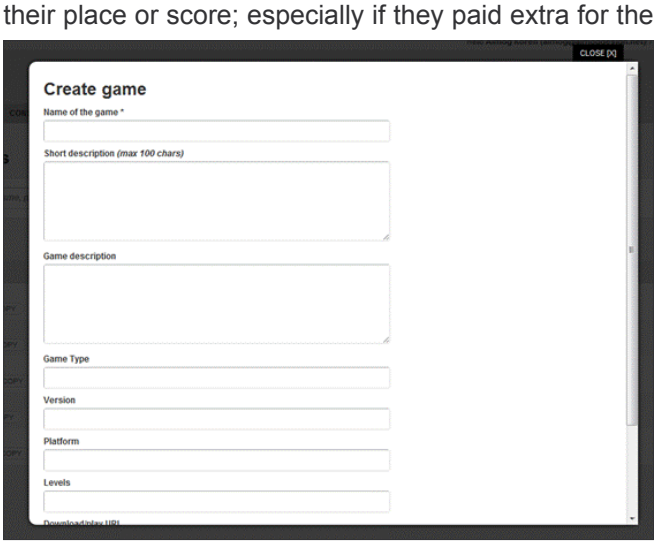



**Figure 3.** *Scoreoid Dashboard*

HD tablet version. With Scoreoid you can easily save a user's score, completed levels, achievements and much more. It's all built-in making it very ease for you and giving you more time to work and improve your game.

## Getting started with Scoreoid

So you're interested in trying Scoreoid out and I bet you're asking what you need to do. No problem! First thing you need to do is sign up at Scoreoid.com once you sign up and validated your email sign in and create you first game.

Once you added your game, all you have to do is use the Scoreoid Open Web API.

Scoreoid's Open Web API methods are RESTful HTTP/HTTPS requests which return XML or JSON responses. The Scoreoid Open Web API works with every coding language making it truly cross platform. We recommend using the built in Scoreoid console to learn and test our Open Web API.

## Conclusion

There is much more to Scoreoid and we have only scratched the surface so make sure to check out the rest of the Scoreoid's APIs and stay tuned as we roll out new features.

**ALMOG KOREN**

***Almog Koren is an Interactive Developer & Designer, working under „Almog Design". He is also the CEO of Scoreoid a game development platform start-up. He enjoys scuba diving and photography. His website: http://www.almogdesign.net/ and http://www.scoreoid.net/.***

# Let's Chat

## Developing A Chat Application With XMLSocket and EventMachine

In this article we'll take the chat example app presented in my previous article (IoC I10n)and connect it to a simple chat server developed with Ruby and Event Machine.

### What you will learn…

- How to use XMLSocket
- Use Ruby to develop server side apps
- and APIS
- How to use EventMachine

### What you should know…

- Intermediate Ruby Knowledge
- Basic network communication knowledge
- No fear form command line shells

Let's begin changing the client application. I will create a state called chatWindow to simulate login process – which we'll really implement in the upcoming articles.

I will also do a very simple validation for the user name and password, just enforcing we typed something and keep the user name to use ahead. See Listing 1.

### Connecting to Chat Server

There are two types of socket connections we can use in ActionScript 3.0:

- Binary connection
- XMLSocket connection

I chose to use XMLSocket connections for a couple of reasons:

- First and foremost because it enables us to exchange data between client and server, without needing to continually open new server connection.
- This open connection will remove latency problems is commonly used for real-time applications such as chat applications or multiplayer games.

A note about HTTP tunneling: The XMLSocket class cannot tunnel through firewalls automatically because, unlike the *Real-Time Messaging Protocol* (RTMP), XMLSocket has no HTTP tunneling capability. If you need to use HTTP tunneling, consider using Flash Remoting or Flash Media Server (which supports RTMP) instead.

**Listing 1.** *LogtoServer method*

```
/**
 * Log user to server
 *
 * For now, it does not really logs, just changes
                state
 */
  public function logToServer(event:MouseEvent):void
 {
     //A simple validation to ensure we have a
                user name
     if(this.txtUser.text == '')
     {
          Alert.show('type your user name');
          return;
     }
     //Let's keep the user name.
     this._userName = this.txtUser.text;
     this.currentState = 'chatWindow';

     this.btnSend.addEventListener(MouseEvent.CLIC
                K,this.sendMessage)
}
```

So to connect to our chat server – that I will the code soon, let's declare a private vari-able as shown in Listing 2.

I also defined two constants to hold the server host and port. In a production application I'd store this information in an XML file because we can change HOST and PORT address without need to recompile the hole app. See Listing 3.

The init method was changed as I moved all listeners to the registerListenerMethod and added the `serverConnect` method. Shown in Listing 4 and Listing 5.

**Listing 2.** *socket connection.*

```
private var _socketConnection:XMLSocket;
```

**Listing 3.** *Constants for host and server port.*

```
private const HOST:String = 'localhost';
private const PORT:uint = 54178;
```

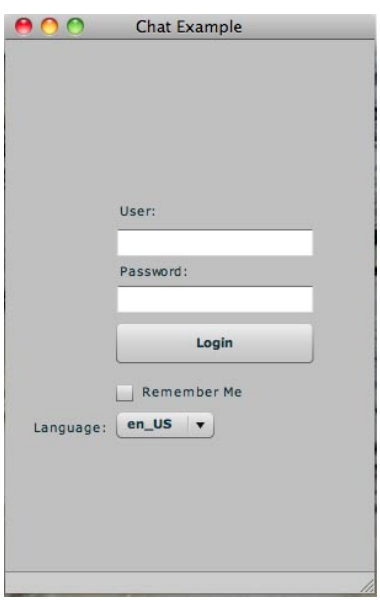



**Figure 1.** *Chat Application UI*

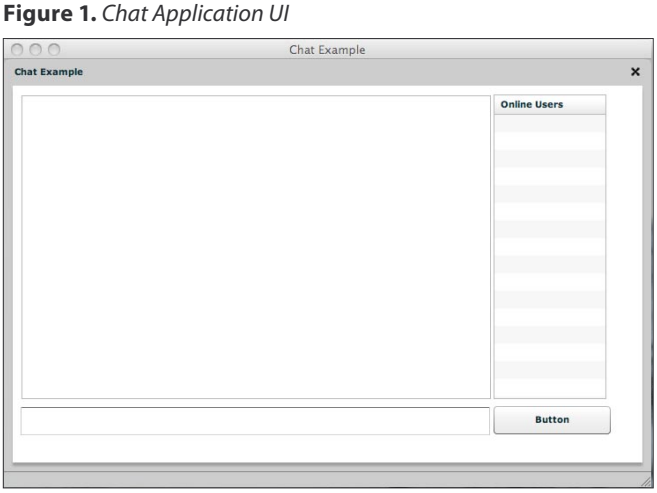



**Figure 2.** *Chat Application UI*

All of you will note that i didn't include no exception handling this time, this will be done on upcoming articles. In the `registerListeners` method, we have and event handler for the `DataEvent.DATA` event that will be dispatched as our application receives data from the server.

The next two methods are pretty easy, that's the handler to close the event Listing 7 and to print a message to the log window upon client connection.

The `onSecurityMethod` will be invoked as soon as a security error occurs and prints a message to the console. Listing 9.

## Server interaction

Interacting to the chat server via XMLSocket is a quite direct approuch. To send data we use the send from our `_socketConnection` passing as arguments the user name and the message typed. Again i added a small validation to ensure the user typed something. See Listing 10.

Receiving data could not be easier, as we registered an event listener to DataEvent, as server responds to the client, we have a lot of information about this event available. One of them is the text the server sent to our small client. So, with this information in hand we print it in the chat window using the text property of the event. See Listing 11.

**Listing 4.** *Init method*

```
/**
 * Application entry point
 */
  public function init():void
  {
     this.serverConnect();

     this.cmbLocale.addEventListener(ListEvent.CHANGE,

                    this.changeLocate);

      //Let's use this locale as default
      resourceManager.localeChain = ['en_US','ja_JP'];
      this.registerListeners();
   }
```

**Listing 5.** *Connecting to the chat server*

```
/**
 * Connect to chat server
*/
  public function serverConnect():void
  {
     this._socketConnection =
              new XMLSocket(this.HOST,this.PORT);
  }
```

That's it for our little chat client for now. Let's take a look what is going on the server side.

**Listing 6.** *Application listeners*

```
/**
 * Register the application listeners
 */
  public function registerListeners():void
  {
      this._socketConnection.addEventListener(Event.
                    CONNECT,this.onConnection);

     this._socketConnection.addEventListener(Event.C
                    LOSE,this.onClose);

     this._socketConnection.addEventListener(Secur
                    ityErrorEvent.SECURITY_ERROR,

                     this.onSecurityError);

     this._socketConnection.addEventListener(DataEve
                    nt.DATA,this.onIncomingData);

     this.btnLogin.addEventListener(MouseEvent.CLICK
                    ,this.logToServer);

   }
```

**Listing 7.** *onConnection event handler*

```
/**
 * Action upon connection
 */
  public function onConnection(event:Event):void
  {
      this.txtMessageArea.text += this._userName +
                    'Connected';

  }
```

**Listing 8.** *Close socket connection event handler*

```
/**
 * Close the connection to the chat server
 */
   public function onClose(event:CloseEvent):void
   {
       this._socketConnection.close();
   }
```

## Anatomy of a Server

Really cool stuff is going on here. First and foremost the chat server is written in Ruby, a language i just fell in love as soon as i started using it. Second this server uses the concept on non-blocking I/O.

## What's Non-Blocking I/O is all about ?

To put it very simple, with a non-blocking I/O server, we can handle a lot of connection in a single process. Our server will use EventMachine an event driven networking library for Ruby, similar to Twisted for Python.

It is an implementation of the reactor pattern, it separates networking logic from application logic. This means you don't have to worry about handling the low-level connec-tions and socket logic. Instead, you just implement callbacks for the appropriate networking events.

## So, show me the code !!

So, before i dive in the server code, just checks if have Ruby and RubyGems installed for your system,

**Listing 9.** *Security error event handler*

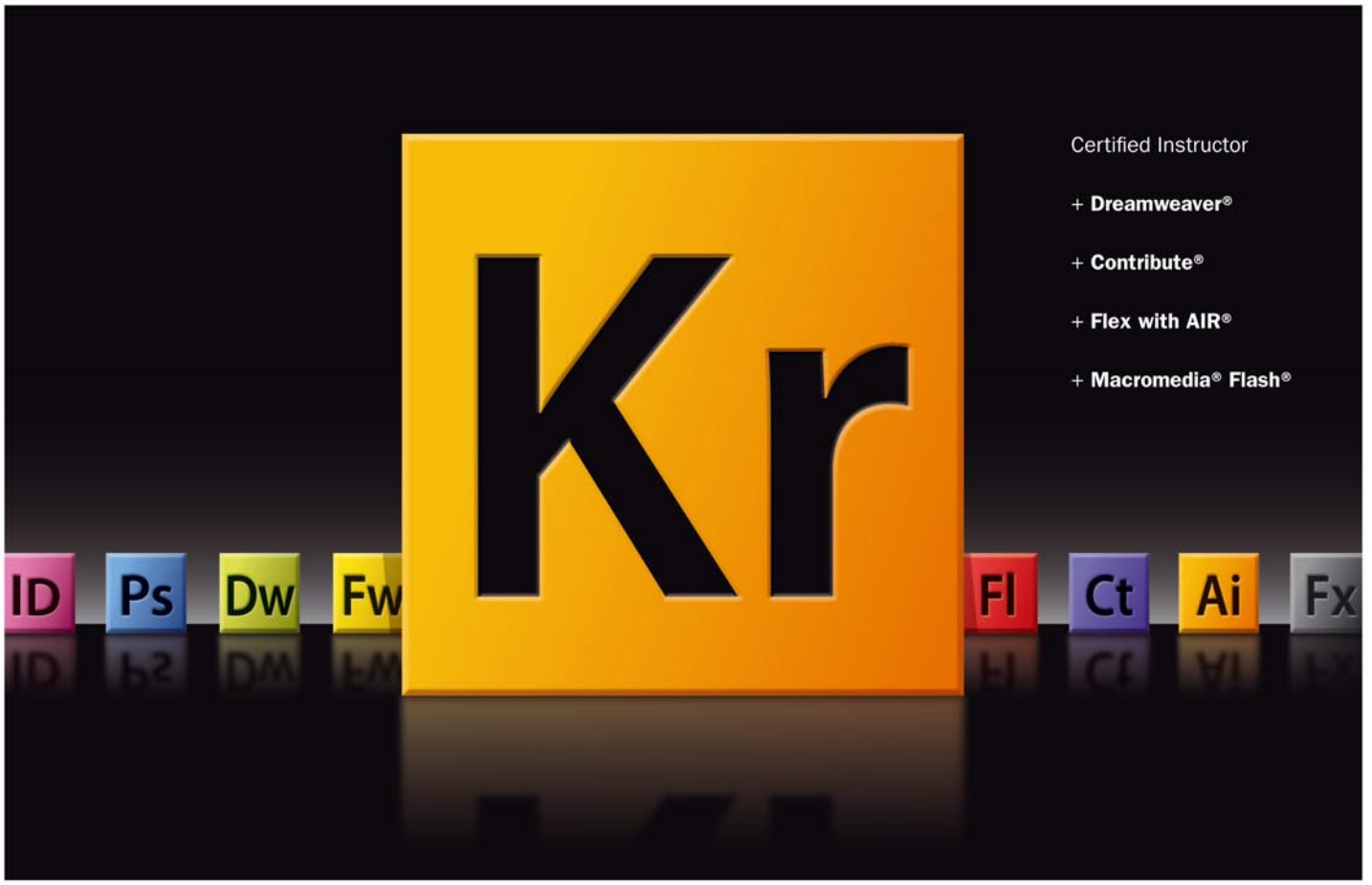
```
/**
    * Security Error
    */
     public function onSecurityError(event:
                    SecurityErrorEvent):void
    {
        trace('security error');
    }
```

**Listing 10.** *Sending data from the server*

```
    /**
     * Send message to server
     */
     public function sendMessage(event:MouseEvent):
                    void
     {
         if(this.txtMessage.text == '')
         {
             Alert.show('type your message');
             return;
         }

        this._socketConnection.send(this._userName +
        '  say: ' + this.txtMessage.text);

         this.txtMessage.text = '';
     }
```




```
/*
ask this question of prospective students
*/
```

# What do you want to learn today?

```
/*
this tells 'em what we do
*/
```

Learn Flash, Flex, Contribute, DreamWeaver, XML, XSLT, JavaScript, XHTML, CSS and ColdFusion from an Adobe certified instructor, who:

- Specializes in training beginners to advanced learners
- Conducts corporate on-site training
- Provides consulting services

```
/*
let others sing your praises
*/
```

"It's rare to find an instructor who has personality and the ability to teach complex subjects. Kevin is great!"
–Kelley Sullivan, Power Integrations

KR KEVINRUSE + ASSOCIATES

| www.kevinruse.com | 001.408.496.6846 | kevin@kevinruse.com | lstevens@kevinruse.com |
|---|---|---|---|

to be up and running. Then you'll have to install the eventmachine gem. Shown in Listing 12.

Now that you have everything installed, let's dive into the chat server code.

- I first require both eventmachine and rubygems
- Define a class called echo that inherits from `EM::Connection`

**Listing 11.** *Receiving data to the server*

```
/**
 * Method executed on incoming data
 */
public function onIncomingData(event:DataEvent):void
{
    this.txtMessageArea.text += event.text + '\n';
}
```

**Listing 12.** *Installing the eventmachine gem*

```
$ sudo gem install eventmachine
```

**Listing 13.** *An evented chat server*

```
require 'rubygems'
require 'eventmachine'

class Echo < EM::Connection

    def post_init
        send_data "Connection stablished"
end

    def receive_data(data)
        send_data(data)
    end

    def unbind
        puts "Client disconnected"
    end
end

EM.run do
   EM.start_server("0.0.0.0",54178,Echo)
   puts 'Starting chat server on port 54178'
end
```

**Listing 14.** *Stating the server*

```
EM.run do
   EM.start_server("0.0.0.0",54178,Echo)
   puts 'Starting chat server on port 54178'
```

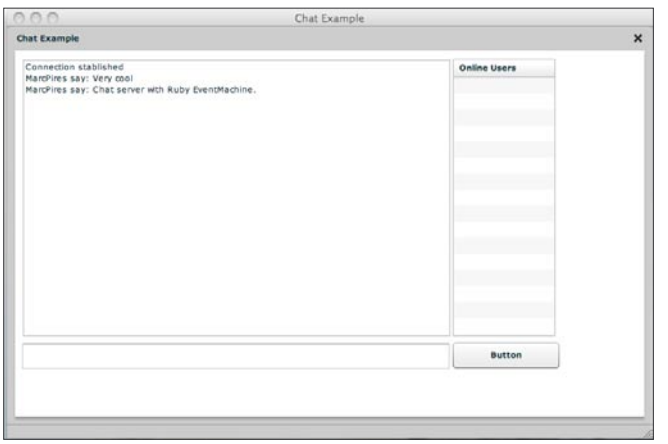



**Figure 3.** *Chat example*

- Then i define some methods that will be responsible for dealing with our little client. The post_init will just send *Connection stabelished* when our client connects to the server.
- The `receive_data` will receive and send the data posted to clients. unbind is will just send a notification that the client disconnected.
  Note that i did not close the server process in this method.
- After i start my server passing the ip address – passing 0.0.0.0 means that our server will listen for connection from any ip address, port number and a callback – in this case the Echo class. Listing 13, Listing 14 show the complete code for the chat server.

To test the chat server, go to the server directory – if you download the code from GitHub *https://github.com/marcpiresrj/XMLSocketEventMachine* and issue the command ruby server.rb see Figure 3.

Note that if you run the client before the server you will have a connection errors be-cause as before exception handling will be added in upcoming articles.

Well that's all for this article. Despite the application shown is still very simple. It shows how we can connect to servers via XMLSocket and presented a new topic EventMa-chine and Ruby. In the upcoming articles i will present more interest things that can be done with Flex communicating with Ruby/EventMachine.

This application example will be available at *https://github.com/marcpiresrj/XMLSocketEventMachine*.

Any question about this article drop a line. See you next month.


## MARC PIRES

***Marc Pires is a RIA, IOS, Ruby Developer and is heading a new startup focused on mobile ecosystems.***

***Contact information:***

***@MarcPires (Twitter)***

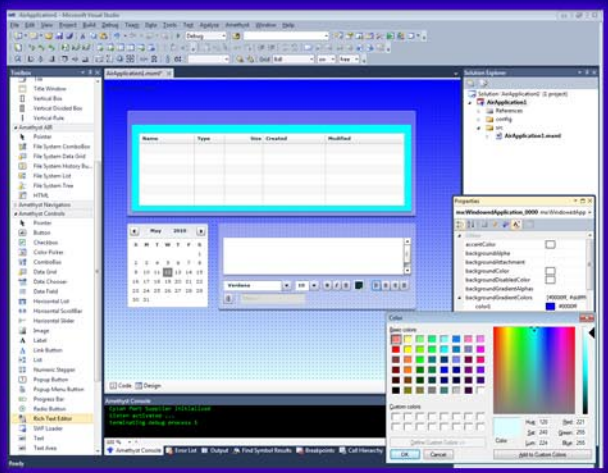
***marcpiresrj@gmail.com | IChat: marcpiresrj@aim.com***

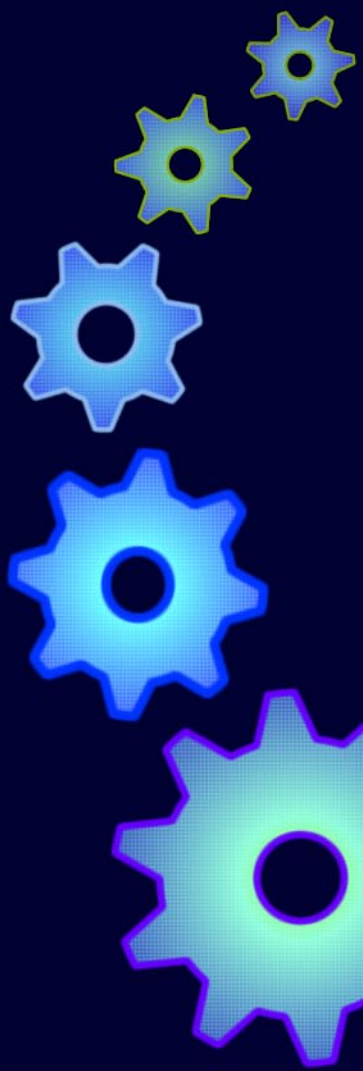
SapphireSteel Software

# amethyst

## the flash platform in visual studio

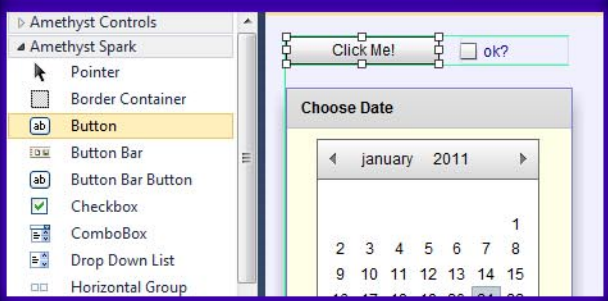
the professional solution
for Microsoft developers





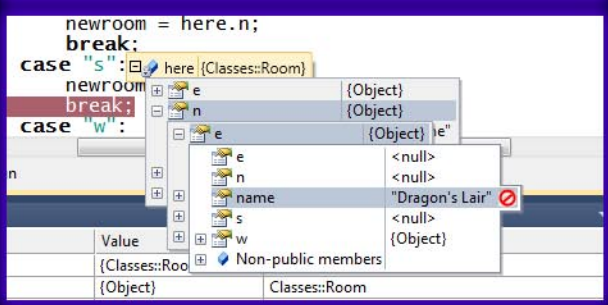
### design

flex 3 or 4
drag & drop
auto-align



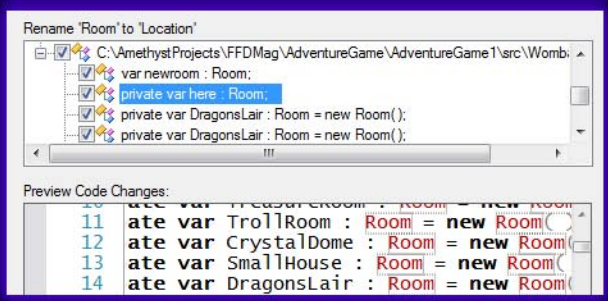

### debug

hover & drilldown
multi-swf debugging
debug Flex & .NET




### refactor

rename
getters/setters
re-order arguments




**Plus:**

IntelliSense, snippets (code templates), customizable code formatting, AIR support, share projects in Flash Builder or Flash IDE, class-browser, 76 custom code colors

http://www.SapphireSteel.com

# 10 Things Every Java Developer Should Know About Flex

Originally Posted on The Code Project

Java developers make great Flex/Flash developers. As an Adobe Certified Technical Trainer who has taught many Java programmers in the last several years, I have seen first-hand how quickly Java developers adapt to ActionScript.

As a RIA Project Manager I have found Java developers not only pick up the Flex API quickly, but they use it correctly and generally follow best practices almost instinctively. Notwithstanding the above, Java developers tend to encounter some stumbling blocks along the way. First of all there is a paradigm shift in the way they approach applications that are considered *movies* by the Flash Player runtime. Generally speaking, I would say the area of the runtime is where my classes tend to slow down and the questions come quickly. In addition to the runtime, there are some unique features of ActionScript vs. Java or ActionScript vs. JavaScript. To explore some of these differences and shed some light on some of the more unique features of the Flash Platform, I interviewed Darryl West, Chief Technology Officer of roundpeg (San Francisco), a Web Development firm.

In many ways Darryl represents the Java developer I typically find in my enterprise training classes. Darryl is extremely well-versed in Java and deeply involved in open source, as well as Spring and Hibernate. In addition to Java, Darryl actively programs to varying degrees in Ruby, Groovy, Grails, Javascript, Ruby on Rails and Flex. He is an excellent source of information regarding Flex from a Java programmers' point of view and has worked on both the back and front-end (including Swing GUI experience) in both Java and Flex. He also has had significant exposure to database work and multi-threaded environments. My own experience involves the training, developing and managing of those who create Flex applications using the Flash Platform tools, servers and runtimes. In addition to being a technical trainer that specializes in HTML, CSS and Javascript, I am also an Adobe Certified Instructor for the Flash Platform. For the past five-plus years I have trained approximately 2,500 designers and developers representing over 700 small to large organizations. To provide further perspective, five of these firms are in the Fortune 25, 12 are in the Fortune 100 and 21 are in the Fortune 500.

## The List

The result of my interview with Darryl is a list of ten things Java developers will find helpful, interesting and sometimes unique about the Flash Platform. Specifically the list includes some happy discoveries Darryl made in the course of his Flex development as well as some challenges he experienced along the way. The list also touches on some significant changes in thinking the Java developer might consider making in order to use the Flash platform most effectively. It seemed appropriate for us to include some of the similarities and differences between ActionScript and JavaScript as well. Some of the lists' topics include: the runtime, handling data access, and specifics of the ActionScript language including closures, cookies vs. shared objects, the external interface class and the local connection classes. We also looked at some specific MXML features regarding the user interface and mx components vs. spark components.

### #1 The Flash Player is Single-threaded

When I tell my students that the Flex runtime, the Flash Player is single-threaded, it often causes confusion and results in misconceptions. Developers will often erroneously conclude that new operations such as a search attempt cannot occur while some other operation is in place, such as a view state change or remote procedure call. As Darryl states *the Flash Player is not really single-threaded, but threads are not exposed*. Darryl has sometimes created pseudo-threads with various techniques (see # 10: Using the

`LocalConnection` class) such as timers, *but they don't operate like a real thread*. While at first Darryl was surprised at this he doesn't consider it a serious issue or flaw. *…with Swing you have to be real conscious of threads and cleaning them up when creating and destroying them…[Flash] eliminates the concerns a Java programmer might have such as if you begin a process, which thread to put it on…the runtime is multi-threaded but the programmer has no opportunity to use the threads*.

**Conclusion**
*Compared to Swing, Flex applications are easier to develop and test*.

## #2 Write Once and Run Anywhere

Darryl reiterated a comment I make early on in my Flex training classes. *The Flash runtime is truly write once and run just about anyplace!* The Flash runtime is cross-platform, cross-browser and cross-browser-version. Although I am aware of known issues regarding frame rate and Internet Explorer vs. Firefox, Darryl has not encountered any problems in his applications. In this regard, Flex contributes to rapid application development, particularly in comparison to HTML, JavaScript, CSS applications where much time is devoted to testing and fixing cross-browser issues.

**Conclusion**
*With [Flash Player] rendering is nearly identical across browsers.*

## #3 Draw Once and Leave It Alone

As a Java developer if you are accustomed to using Java 2D or writing applets, you draw every time you need to update. With Flex, you draw once and

leave it alone. The API has numerous transformations that can be applied – to rotate, to skew, to scale, etc. *There is no need to loop through re-draws* according to Darryl, who once again found this a welcome change as well as a time-saving feature.

**Conclusion**
*Draw once and leave it alone*.

## #4 Flex is Extremely XML-friendly

While Darryl considers Java equally xml-friendly, he has found Flex to be extremely xml-friendly right *out of the box*, eliminating the step of locating and using xml libraries. ActionScript natively supports EcmaScript for XML (also known as e4x). This provides Flex developers with an alternative to a DOM interface. e4x uses a simpler syntax for accessing XML documents treating the XML as a primitive. The effect is typically faster access, better support, and a resulting data structure for the program. On the other hand Darryl found Flex not as JSON-friendly and relies on the standard AS3corelib for JSON support.


**Conclusion**
Flex has *built-in* support for XML data exchange.

## #5 Java projects can suffer from class bloat

*While Flex projects can also suffer from class bloat, the programmer is at the mercy of the runner. SWF's larger than 300k should be considered 'huge' projects and therefore separated into Modules or separate SWFs*. I asked Darryl to elaborate on this and the differences between custom components, modules and separate swfs. *A module is in the same space as the main swf that loaded it. A separate swf is running as if it were in another browser, so it doesn't tie up the swf in the main space*. Darryl suggests that the developer does not attempt to use a typical *Java Structure* with eight or ten tabs, a lot of input forms,


etc. that result in increasing the size of your main swf to greater than 300k. Instead, split the main swf into a separate project and swf. He recently stripped out the printing feature of a main application which was pulled out into its own swf project. The application had an info tab with stats that was also moved into a separate swf as well as a user preferences panel. Darryl states that this was *not for memory but for initial load time*.

**Conclusion**

*Rethink the application in order to break things out that are not used all of the time and can live completely independently*.

## #6 Flex has closures

With a background in Groovy, Ruby and Lisp Darryl found it helpful to know that Flex has closures whereas Java programmers use inner classes.

*Every function has a special [scope] property that represents the environment it was in when it was defined. If a function is returned from another function then this reference to the old environment is closed over by the new function in a „closure"* (Caswell, Tim. Learning Javascript with Object Graphs. 30 September 2010 *http://howtonode.org/object-graphs*).

**Conclusion**

To achieve object oriented polymorphism consider the use of closures as a good alternative to inheritance.

## #7 Use Spark for lightweight components

Flex components are not a complete or pure Model View Controller architecture like Swing. With Spark components you must attach skins and scrollers to complete the component. When comparing mx to spark components, spark is similar to Swing. *Spark pulls the extra stuff out…you can load it on if you want, but you don't have to*.

```
http://ns.adobe.com/mxml/2009"
ibrary://ns.adobe.com/flex/spark"
```

**Conclusion**

Spark-built Flex applications can be made lightweight and have a much simpler API when compared with Swing.

## #8 Browser Cookies vs. Local Shared Objects

Most Javascript developers are familiar with Cookies (also known as an HTTP cookie, web cookie, or browser cookie) small text files containing name/value pairs. The cookie is sent as a field in the header of the HTTP response by a web server to a web browser and then sent back unchanged by the browser each time it accesses that server. The Flash Player provides a similar mechanism in the form of the Local Shared Object. The Flash Player runtime has a specific area where it stores files that are controlled and managed by

the runtime. The `SharedObject` class uses a static, lazy instantiation factory method called `getLocal()` that returns the `SharedObject` instance which is a proxy to the local shared object file on the client computer. The developer passes the file name reference to the getLocal method and if the file doesn't exist the Flash Player creates and opens the file for reading and writing by the application. Shared objects can be shared by all swf files within the same domain. Unlike cookies, shared objects are

*machine centric* and browser independent, so objects stored with browser A are readable by browser B.

**Conclusion**
Local Shared Objects provide the Flex developer with another nice feature for authentication, storing user preferences, data persistence, etc.

## #9 Use GZip with the ByteArray

Like most Java programmers Darryl dug through the API for relevant classes. While researching the ByteArray class (The ByteArray class provides methods and properties to optimize reading, writing, and working with binary data), he discovered the opportunity to use gzip with the ByteArray classes compress and decompress methods. While many Flex developers favor using the *Action Message Format* (AMF) object serialization for transferring data from server to client, the ByteArray class supports zlib compression and


decompression. gzip is not part of AMF, and since it's binary, it would not compress well, which is an argument for not using AMF – binary is smaller than XML or JSON text, but not as small as compressed XML/JSON. In addition XML/JSON is cross platform and language so more portable than AMF. According to Darryl, the best alternative to AMF is HTTPService, but HTTPService is non-binary, so you can't use gzip directly, you have to also base64 encode. This adds a size payload to the compressed data, so the typical net compression of XML/JSON using gzip and base64 encode is about 45 to 50%. This is about the same as using AMF, but again you gain portability (and much easier testing) with XML/JSON.

Numerous websites exist that demonstrate benchmark tests using 5,000 to 100,000 or more rows of data. However the majority of web applications that I have been involved in return far fewer records per page. Darryl also reports greater compression results on the more real-world row numbers versus compression on 100,000 + rows, resulting in performance that not only exceeds AMF, but also eliminates the need for BlazeDS or LiveCycle.

**Conclusion**
For remoting, consider using compressed XML or JSON packettes using gzip on the server and ByteArray compress/decompress on the client.

## #10 Use Local Connection class

Another helpful class is the LocalConnection class used to facilitate swf communication from browser-to-browser. This is analogous to Java's applet to applet communication in the same domain. One use Darryl discovered for this class was to overcome the lack of multiple threads by using local connection to provide swf to swf communication, even between two independent browsers. One application in particular loads the main swf while a pop-up windows, or a separate browser communicates with the main swf via messaging through local connection.

| | |
|---|---|
| **Package** | flash.net |
| **Class** | public class LocalConnection |
| **Inheritance** | LocalConnection → EventDispa |

**Conclusion**
Flex applications offer a module to module communication similar to applets but extend that capability to cross/independent browsers.

**Originally Posted on The Code Project**

**KEVIN RUSE**

***Kevin Ruse is the principal trainer at Kevin Ruse + Associates a technical training firm in Santa Clara, California. Kevin has taught at Google, YouTube, Hewlett Packard, Cisco, Disney, Applied Materials, and many other companies throughout the United States and Europe. www.kevinruse.com***

**DARRYL WEST**

***Darryl West is the Chief Technology Officer at roundpeg, a San Francisco-based software development company that focuses on dynamic, data-driven applications for the Web, Wireless, and Interactive TV space. Our vision for the future embraces new media application architectures for the next generation Web-enabled devices.***

# Flash Conferences

## Flash and the City

New York, NY


**Homepage:** *http://www.flashandthecity.com/*

## Flash Camp Brasil

Maceió, Alagoas


**Homepage:** *http://events.actioncreations.com/flashcampbrasil/english/index.html*

## FlashIsreal

Isreal


**Homepage:** *http://flashisrael.com/*

## 360|Flex Conferences

Denver, CO


**Homepage:** *http://www.360flex.com/*

***If you know the conference or event that should be listed here, please write to me at ewa.dudzic@ffdmag.com.***

## FITC Toronto 2011

Toronto, Canada


**Homepage:** *http://www.fitc.ca/Toronto*

## MIX11

Las Vegas


**Homepage:** *http://live.visitmix.com/*

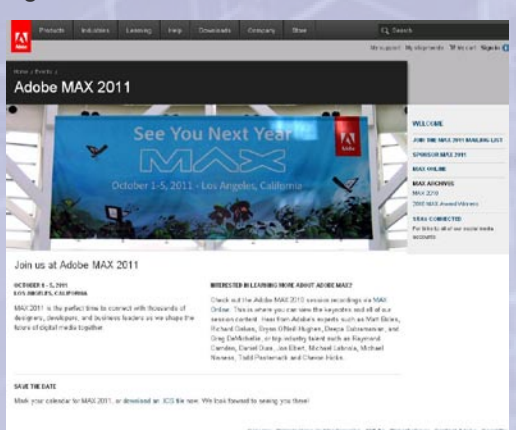
## FITC Amsterdam 2011

Amstardam, Netherlands


**Homepage:** *http://www.fitc.ca/amsterdam*

## Adobe MAX 2011

Los Angeles, California




**Homepage:** *http://max.adobe.com/*

## Geeky by Nature

New York, NY


**Homepage:** *http://www.geekybynature.com/*

## Flash GAMM 2011

Moscow

**Homepage:** *http://www.flashgamm.com/*

# Steve Lionel on Why Fortran Still Matters

Steve Lionel says Fortran, at 54 years old, remains important in an era when most programmers focus on languages like PERL, Python and Java. Lionel, also known as Dr. Fortran, is a senior member of Intel's technical staff, and has a long history with Fortran having developed Fortran compilers at Digital Equipment Corp.


**Q: Is Fortran still relevant? You don't hear many developers talking about this venerable programming language.**

Steve Lionel: It is absolutely still relevant. Fortran itself is venerable, but that doesn't mean it hasn't changed. The Fortran standard has received five updates. The last was in October, and each time new capabilities are added, some of which are vendor extensions that programmers wanted, some of which are features that became popular in other languages and got adopted.

**Q: What are some features Fortran borrowed from other languages?**

S.L.: C++ had classes and polymorphic features that were added to Fortran in 2003.

**Q: What are its other strengths?**

S.L.: Fortran 2008 has built-in (*http://www.intelligenceinsoftware.com/feature/expert_insight/parallel_programming/*) parallel programming capabilities that no other widely used language has. Other languages have parallel programming features, but Fortran incorporated modern parallel programming in ways that none of the other languages have. There is an incredible body of well-written and well-debugged routines in Fortran that are out there for reuse.

**Q: Are new applications being written in Fortran?**

S.L.: There are lots. If you look at hurricane forecasting applications, most of the modern models are written

in Fortran. The Weather Research and Forecasting Model (*http://www.wrf-model.org/development/wppb/wppb.php*) is largely written in Fortran.

There is an automobile crash simulator called (*http://www.esi-group.com/products/crash-impact-safety/pam-crash*) PAM-CRASH written in Fortran. Lots of engineering code, optical modeling, nuclear physics work is done in Fortran. Fortran remains the pre-eminent language in *high-performance computing* (HPC).

### Q: So why aren't more people using it?

S.L.: I don't want to leave the impression that everything is written in Fortran. It's a relatively smaller part of the market than 20 years ago, but it isn't dying. Yes, there's a lot of C and C++ that is more appropriate for certain things than Fortran is.

### Q: What applications are best written in Fortran?

S.L.: For something like string processing, Fortran would not be my first choice. But if you're doing number crunching, working with a lot of floating-point data, or doing parallel processing, it's an excellent choice. Its strengths in array operations – its wide variety of routines – make it attractive, and there is a huge library of freely available high-performance routines written over 40 years that still work together.

With a lot of other languages, when they update the standard, they introduce incompatibilities. The Fortran standards group is very careful not to do that.

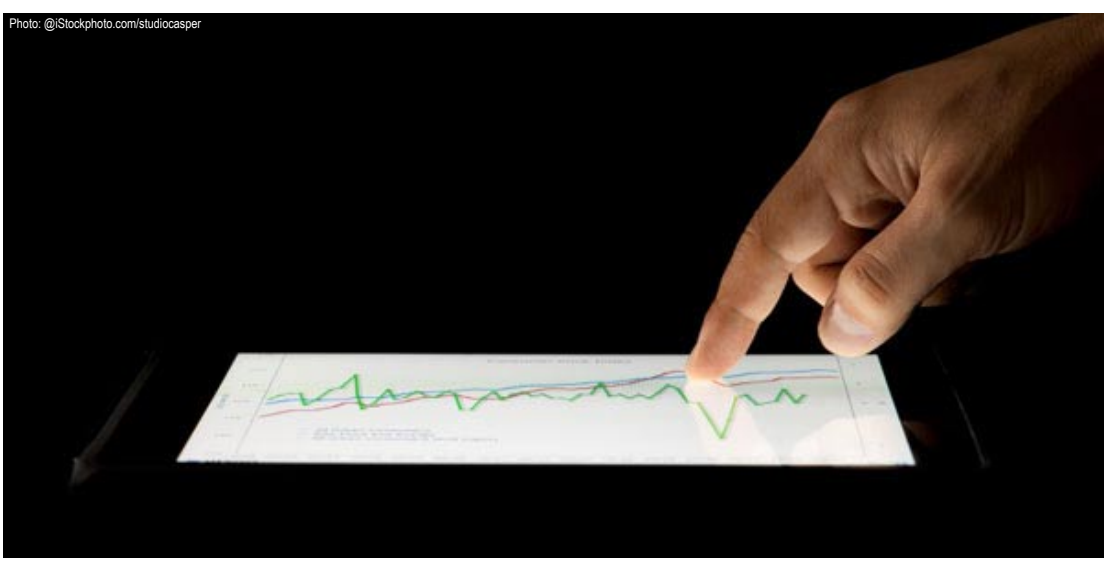


**BRIDGET MOORE FOR INTELLIGENCE IN SOFTWARE**

***Bridget Moore is a Boston-based journalist with more than 15 years experience covering business and high technology.***

# Nice Gesture, But What Does It Mean?

One step forward, two steps back. That's how Don Norman describes today's gesture-based user interfaces (UIs) for smartphones, tablets and a growing assortment of other devices.




Named one of the world's 27 most influential designers by *Business Week*, Norman laments the lack of standards, which have created a world where a finger-swipe on one device often doesn't have the same effect on another. That inconsistency often makes using gesture-based UIs as much fun as folding a fitted sheet. Norman recently spoke to us about this and more from South Korea, where he's distinguished visiting professor in the Korea Advanced Institute of Science and Technology's Department of Industrial Design.

**Q: Your colleagues from the Nielsen Norman Group recently did usability tests on Apple's iPad: "The first crop of iPad apps revived memories of Web designs from 1993, when Mosaic first introduced the image map that made it possible for any part of any picture to become a UI element. As a result, graphic designers went wild – anything they could draw could be a UI, whether it made sense or not. It's the same with iPad apps – anything you can show and touch can be a UI on this device. There are no standards and no expectations." Yet tablet and smartphone sales remain brisk.**

A: It's confusing and difficult for people. On the other hand, it's so engaging and so much fun, that it in many ways compensates for its difficulties.

I'm about to give a series of talks about the way the field has evolved. In the beginning, we were so delighted with the technology. With the passage of time, we've come to take understandability, usability and function for granted. What we want now is a good experience, some fun and delight.

That's where Apple has transformed itself as a company. It really is about experience, fun, delight and entertainment. That's what the iPhone did: By this whole new way of interacting with strokes and gestures, it was so delightful and different.

**Q: In "Gestural Interfaces: A Step Backwards in Usability," you and Jakob Nielsen identified two root causes for gesture UI inconsistencies: a lack of industry guidelines and "the misguided insistence by companies (e.g., Apple and Google) to ignore established conventions and establish ill-conceived new ones." Will we ever have standards for gesture UIs?**

A: There will be gestures that Apple will patent and refuse to let Microsoft and Google use. All of the companies will have competing methods that will just confuse the hell out of people. But we had that in the beginning of computers too.

When I was at Apple as vice president of the Advanced Technology Group in the '90s, I was fighting hard for standards. We had this wonderful meeting, and we got all of the major companies together to argue for standards: Sun, IBM, Apple. Microsoft walks in and says, "This meeting is unnecessary. If you want standards, just use Microsoft's." That's Apple's attitude today.

**Q: One downside of standards – whether it's a UI or a wireless technology – is that they sometimes don't leave companies with many opportunities for market differentiation, aside from price.**

A: Every Android phone basically looks the same. So how do you compete in a world like that, where if you really follow Google's guidelines, you can't distinguish LG from Samsung from HTC from Motorola? Nobody wants to compete on price. That's death.

With early technologies, as people are learning, it's understandable that we don't have standards and that sometimes different applications don't follow the same rules. Jakob and I wrote our joint piece to be critical yet friendly, to say that we thought gestures and the new interfaces were exciting, powerful, adding a lot to our experience, and that this early confusion can be overcome. This confusion is compounded by the lack of differentiation among the vendors, who are desperate to do anything different, and the companies don't wish to cooperate.

**Q: For now, consumers and business users seem to be willing to put up with those inconsistencies, judging by how many iOS and Android apps and devices there are. How long before the gee-whiz wears off and they start to complain that app and device UIs aren't as user-friendly as they could or should be?**

A: It's starting to wear off already. When the iPad came out, people just swooned over it. Now you're starting to see articles, sometimes by the same journalists, saying, "You know, it's got a lot of weaknesses. You can't really type on it." I think it's becoming more realistic that the pad is not a substitute for a computer. It's a wonderful device, but for its own purpose: You can read on the couch or answer a quick question. That's a reality that wasn't there at first.

**Q: You and Jakob also bemoaned the lack of "Undo" in today's gesture UIs. That absence is surprising, considering that PCs have conditioned us to expect it, and it's useful.**

A: I think because it comes a little out of the browser world, where the theory is that the "Back" button is the "Undo." But it isn't, not when you type something. On top of that, "Back" has always been very confusing. You're never sure what's going to happen when you hit "Back."

I really hate this. There's a stack of the previous locations, and "Back" takes you to the previous location. But I'm in some app, and I'm not sure where I am because it doesn't tell me, and I want to get back to what I think is the home page. I hit "Back" and, whoops, I'm on the desktop.

It shouldn't do that. You shouldn't be able to back out of the application. In the browser, when you reach the end of its list, it doesn't close the browser or take you to the desktop. It just dims the back button and doesn't work anymore.

That's what ought to happen when you hit "Back" in an application. I got really angry at one application developer. I kept saying, "Why are you doing this?" They explained that it wasn't them, that the "Back" was programmed into the operating system.

**Q: That means operating system companies have to provide a foundation that enables device vendors and app developers to make UIs more user-friendly.**

A: Yes, it does. And they know that. That's why Apple has usability guidelines. I'm not sure if Google does. There are people who have written them for Google.



**TIM KRIDEL FOR INTELLIGENCE IN SOFTWARE**

***Tim Kridel has been covering all things tech and telecom since 1998 for a variety of publications and analyst firms. Based in Columbia, Mo., he still enjoys the childhood hobby that led to a career writing about technology: ham radio.***

# FLASH&FLEX

DEVELOPER'S MAGAZINE


coverpage

Your digital publishing tool for iPad.

Easily create your interactive digital experience.
Bring your readers to a whole new level.

Created with Digital Publishing Tools MONOGRAM CoverPage™.
www.monograminteractive.com / www.coverpageapp.com


# ActionScriptJobs.com

## *KickASS Flex & Flash Jobs*

### NEED A JOB?

We work with companies to bring *KICKASS* Flash/Flex jobs to you through our Job Boards & Resume Tools.

### NEED TO HIRE?

We can *HELP*... not only do we know how to recruit, we are Flash/Flex developers ourselves.

**CREATED by ActionScript Developers ...**

**... FOR ActionScript Developers**

**www.ActionScriptJobs.com**