

Java technology zone technical podcast series: Season 3
Jonathan Stark on cross-platform mobile development
Episode date: 11-29-2011

GLOVER: I'm Andy Glover, and this is the Java Technical Series of the developerWorks podcast. My guest this time is Jonathan Stark. He's the author of multiple mobile development books for O'Reilly. He's recognized as a cross-platform mobile development expert who used to maintain jQuery Touch. He's an all-around interesting guy.

I had trouble tracking him down, but I've read his books, I've actually been on webinars that he's participated in. So I'm so happy that Jonathan, you were able to join us today and I was able to get through that without too many tongue ties.

STARK: Thanks, Andy. Glad to be here.

GLOVER: So, you know, mobile is huge. It's a very interesting space. There's so much going on there. So I thought I'd kick it off by kind of asking you, what does it mean to be a mobile web developer? What does cross-platform development mean? Kind of what's the landscape, what's the lay of the land here?

STARK: Right. So, I do a lot of work with a company called Mobiquity, who has enterprise-class mobile clients. So they work with the Fortune 2,000 and help them with everything from strategy to design, development, deployment, maintenance of enterprise-class mobile apps.

So, when you're working with large organizations like that, they typically don't have the leisure of just marrying one platform like iOS or Android; they have to support for B2C initiatives, they have to deploy apps that huge groups of people are going to be able to use. And of course, you can't control what kind of phones those people have.

And then even internally for B2E apps -- which is a huge opportunity as well -- a lot of companies are supporting a bring your own device model where they're not necessarily issuing BlackBerries anymore, people are demanding that they can use their iPhone on a corporate network and the corporate systems.

So, really large organizations like this don't have the luxury of pulling an iOS-only move. So where I come in is kind of connect-the-dots between their back-end systems which were usually created organically through the kind of desktop era of the nineties to be...they're not exactly real time, let's just put it that way and lots of data silos and that kind of thing.

So there's that on the back end and then the marketing department or their IT department or human resources has this mobile project where they want to deliver a mobile experience to this huge group of users with all sorts of different devices, whether they're phones or tablets or whatever, and they need to come up with a way that's feasible to actually get from Point A to Point B, and that's where I come in.

GLOVER: So, how do you get from Point A to Point B and like you said, support the.... Well, at this point there's basically two major platforms, right, in terms of hotness. You've got iOS and Android, and then everyone's kind of biting at the ankles for third place in terms of Microsoft and I guess the BlackBerry. There was the hope that web OS, a reemergence, but that appears not to have happened. I mean, what are you seeing? How do you approach this?

STARK: Yes, there's a couple of different things that...I mean, every project's different, for sure. But there are some things that seem to, you know, certain patterns keep cropping up. So, usually after analyzing the use case, the application stack, what they want to release and what all mess they had going on in the back end, usually a couple of things will come up.

One is that they need to kind of stitch together their back-end systems with some kind of an API that is usually a simple RESTful-based API, lives in the cloud, that can knit together these back end systems and at least approximate a real-time type of experience because mobile, you know, you can't release a mobile app and expect to get any traction if it's...if there's like a nightly batch script involved.

It's like, that cut it back in the day, but...and was a reasonable design decision, architecture decision back in the day, but you can't do that now and expect your application to get any traction.

So, usually there's some kind of either a consolidated API or a federated group of APIs that have a sort of central documentation either in the data center or third party hosting that the enterprise...the IT department would basically coordinate if not build it directly.

And then you'd start off with this first mobile application that would be targeted to probably a subset of the full range of smartphone devices in the market. So there would be some analysis maybe on their existing website or whatever data that they have and they'd say, okay, we need to deliver...

Ninety percent of our traffic comes from iOS, whether, could probably...a surprising amount of traffic from iPads already, iPhones, maybe a couple of versions of iPhone hardware, a raft of Android devices and probably some BlackBerry thrown in there. So those are usually the ones that they need to really address.

So, with that level of information, it's like, okay, so the front end, we did this API in the middle that we can talk to, and we can create client applications for these different phones. And I start...I generally always start with a web-first approach.

So I look at the design spec for the application and use cases and features that they want to support, and say, okay, can we do this as a web application or essentially a finger friendly mobile website, essentially. So if that's the case, then that's probably the way to go, because you've got the...you can address by far the hugest amount of market that way.

GLOVER: Let's go back to mobile web. You know, I think that maybe when you say mobile web people think, oh, HTML5, you know, Safari and Chrome. How does that affect BlackBerry?

STARK: Well, I mean, it depends on the experience you're trying to deliver. So if it's BlackBerry, the browser is not as up to date, let's say. So you know, if you want to make it super polished with like the CSS3 animations, it's probably not going to translate on BlackBerry.

But as web developers, we've been dealing with this for years. Basically either degrade gracefully or progressive enhancement to create a useful experience for all the platforms that you need to support, but the newer and shinier ones get a nicer experience.

GLOVER: Okay.

STARK: So if an application can be delivered that way, it's by far the lowest total cost of ownership approach. So that's, so for something like, you know, a human resources application that you're going to be delivering to your employees where they're essentially connecting to your intranet and managing their health care elections, or their 401(k) distributions or whatever.

Something like that that's not going in the app store, it's probably fine if it's not super fancy; it just needs to be responsive and well designed and clear and useful. And if you've got that, then why bother dealing with, you know, Apple, frankly? You know?

GLOVER: Okay.

STARK: So, but that's not always the case that the spec allows web app, because of course the big downside of using a mobile web browser to access your content and services is that they can't do things like take a picture, or maybe update your contact list on the phone, or access a calendar.

GLOVER: So what happens if you're offline?

STARK: Well, you can run a web app offline these days in the more modern browser. So like iOS and Android have support for client-side local storage which is kind of like cookies but it doesn't get sent over in adders with the requests. And web SQL databases supported on both of those phones, so it's another progressive enhancement approach for people who have those devices.

GLOVER: And these are HTML5-specific features?

STARK: Well, they're in the family of HTML5 features, yes.

GLOVER: Okay.

STARK: Technically they're not in that spec, but yes.

And then there's also offline application cache, which allows you to take static assets and store them locally on device, so essentially installing your static assets on the phone.

GLOVER: Okay.

STARK: So you can create an offline experience at the browser, which is something that people...a lot of people don't know. So it kind of ruffles my feathers when people are like, oh, we were going to go with the web, but we needed offline support so we're going to go native. And it just makes me want to scream.

GLOVER: Well, one more question, there. Would the offline support work on a web app for a BlackBerry?

STARK: Well, no, not really. There might be, the PlayBook I'm not sure about, to tell you the truth. It's definitely a bleeding-edge type of case, and that's why I said it's a progressive enhancement, so.

GLOVER: Are you aware of, what is it, Windows Mobile or Microsoft Mobile, their support?

STARK: Not for that particular feature. Actually, I just got the update for my Windows phone and I haven't tested it yet, but yes, so again, progressive enhancement. So if the feature is there, then you can use it and if it's not you can't. So, you know, it's going to be a moving target for, it changes on a daily basis, so it's just one of those things that you build in and if it works it works, and if it doesn't, it doesn't, so.

But like I said, if you need access to things like the camera or the address book or those sorts of things, you need to have some kind of native installation because the browsers are a sandbox away from those sorts of things.

So, as I'm looking at the spec for the application, I say, oh, okay, well, all you need is like a little bit of camera access to maybe do a barcode scan once in a while; it's not like the core feature of the application. So then I start thinking about what's called a hybrid approach, which really essentially is a native app for all intents and purposes but the guts of the application are written with HTML CSS and JavaScript.

So the advantage that that gives you is that you can write the guts of the application, so most of the work that you're going to do can be done first of all by web teams, which, it's a lot easier to find web developers than it is to find iOS Android, Windows phone, BlackBerry developers.

And a lot of organizations at least that we work with already have web talent in house, and they're typically pretty excited about going into mobile and learning the new skills and approaches that are appropriate for mobile. So everybody's happy, it makes everyone happy.

And if you can deliver the experience that you want, the user experience that you want using HTML CSS and JavaScript, you can wrap the static files in something like PhoneGap or something you create yourself which bundles the web assets inside of a native wrapper that you would then distribute just like a native app.

So either through the iTunes App Store or with ad hoc distribution -- you know, something I'm sure you know a lot about. So that allows you to write JavaScript that can access native APIs on the device. So I can write a navigator dot get picture in what's essentially a web app but it actually calls up the camera interface on the phone and allows me to take a picture and then it returns it into the web container to the JavaScript as either a URI or basically foreign coded text. So then you can immediately use that inside of the app or uploaded to a server using Ajax or whatever.

GLOVER: Now, you mentioned PhoneGap. Tell me more about PhoneGap in terms of, so, this is kind of a hybrid solution and as you and I both know, PhoneGap's a player in that market; there are some others. So I definitely, and I want to hear more about others, but since we started talking about Android, iOS and we mentioned BlackBerry, what's the level of support with these devices or others with something like PhoneGap?

STARK: PhoneGap offers the highest cross-platform support of anything I've seen, and it's the kind of thing I try and keep track of, so I'm pretty sure I'm right about that. So, a couple things.

There's a company in Vancouver named Nitobi that originally came up with PhoneGap at a hackathon and immediately outsourced it under M.I.T. license, and it's been on GitHub for a couple of years. It's extremely popular, very, very active project and enormous contributions from all over the community.

Recently -- very recently -- Nitobi was acquired by Adobe to sort of flesh out their HTML5 offerings, basically it was a talent acquisition. And to ensure that the PhoneGap project itself remained free and outsource, they contributed it to the Apache Software Foundation. And part of that process is that they're going to have to rename it. It's probably going to be called Apache Callback but that's still up for grabs.

GLOVER: And you say "it's up for grabs," so it's, obviously we're having this conversation on November 4th; are you aware of a timeframe or anything like that?

STARK: No, no. Unfortunately. I'm actually writing a book about it [as well], I've got a big copy/paste, find and replace operation in my future, I think.

GLOVER: So tell me more about this book you're writing.

STARK: Well, actually to be specific, it's the second edition of my Android Apps book. So I wrote a book called Building Android Apps with

HTML CSS and JavaScript, and there's a chapter on PhoneGap that I'm updating. So, that's where...

GLOVER: So what other...okay, so there's PhoneGap and maybe it will be called Callback by the time someone listens to this.

STARK: Right.

GLOVER: And as far as you're aware, and I agree with you, it has "the" most support in terms of mobile devices. But it's certainly not the only framework out there. What are other frameworks? Is PhoneGap the best one for all scenarios, just like you were saying, sometimes maybe native does make sense, but if you were going to go hybrid, is PhoneGap.... You know, is PhoneGap great for making games, for example? Are there are other platforms?

STARK: So the way that I answer that is, so, two questions there. One is, is PhoneGap good for everything, I guess, and the answer...the answer that I give people is, if you can build your app with HTML CSS and JavaScript, then you probably should. And that's not to say web apps. There's two different things.

Like, there's a web app and there's a hybrid app. They're both built with HTML, but one's distributed through the web and viewed in a web browser and the other one could be installed naively on the device. So I like to draw a strong distinction between a web app and an HTML app.

So, if you can build your app with HTML CSS and JavaScript, you probably should, but conversely, if you can't then you shouldn't. So if you're a web developer and you're sitting there and you have an idea for an app and you can immediately envision how you would put that together as a web app, then you should do that.

If you want to write, I don't know, Infinity Blade, and you're a web developer, I guarantee you there's no way in your mind that you're thinking you can do that. So, I mean, it all starts, the whole decision tree starts with, what am I trying to build?

And you look at that user experience and then you can make a decision. And like I said, if you can build it with HTML, then that's probably the way to go because worst case scenario -- worst case scenario -- you're going to have a killer high fidelity prototype that you can deliver to a native development team.

GLOVER: That makes sense, yes. And so, and you're talking about kind of if you can build it with the web, and you did mention JavaScript. And earlier I brought up jQTouch. There, you know, there are so many different projects and acronyms and just things to be aware of in the mobile space. You know, we talked about just at a high level just the different platforms, and then you've got the different ways you can go about it, building a web app or hybrid or native.

If you go the web app route, there's myriad JavaScript frameworks out there.

STARK: Right.

GLOVER: Where does one start?

STARK: It's a good time to ask me this question, because I recently did a long post on it that compares all the ones that I hear talked about.

And so I kind of go through it at a high level. So, and you've kind of asked this already, you know, what should I consider versus PhoneGap. Really PhoneGap I see the main thing that people compare PhoneGap against is Titanium Mobile, which is from a company called Appcelerator.

And Appcelerator's doing some really interesting things and they have a lot of traction in the development community, but it's not the same thing as PhoneGap at all. So comparing them is completely apples and oranges.

GLOVER: Oh, great. Okay, okay.

STARK: Yes, so PhoneGap is what we described. It's basically source code for a browser on steroids that you drop HTML, CSS and JavaScript into. Titanium Mobile, on the other hand, is sort of an environment where you write valid JavaScript...you write using JavaScript as your language but that's sort of incidental, and that JavaScript is compiled against their proprietary framework down into native code for iOS and Android.

So the code that you're, air quotes, the JavaScript you're writing wouldn't run in a browser. You're not building a web app. You're not really building an HTML app. And in fact, you're not even building a JavaScript app. You're writing code that could be compiled. It could be Java if they had picked that.

GLOVER: Got it. So it's interesting because it's in some ways like, you know, the Google web toolkit in that...

STARK: Very much...

GLOVER: ...you know, with GWT you write it in Java and they compile that down to JavaScript. In this case, what you're saying...

STARK: It's the reverse.

GLOVER: Yes, it's the reverse. You'd write JavaScript and they'll compile it down to, you know, Java if it's, you know, targeting Android. So I guess then with the Appcelerator or Titanium you have to select which platform you're building it to but it is still the same app.

STARK: You're, well, I guess you can compile it down to iPhone and Android. And they briefly had support for BlackBerry, but it seems to have been removed. I don't know what's going on there. But and I believe they're also working on outputting it, outputting the...basically outputting an HTML5 app as well.

And that's all really interesting. They have tons of support and they have this recently announced like a marketplace type of thing where you can, if you're a Titanium mobile developer, you can build kind of like little widgets or code snippets and sell them as add-ons for other people to use.

It's its own thing that I'm not...I'm only sort of casually aware of it. I haven't been in a situation where it was the right tool for the jobs that I do. So I haven't...but I've done Hello Worlds with it. It's really easy to get a demo app running.

GLOVER: Now, what about licensing in terms of, you know, PhoneGap's free, right?

STARK: Yes, M.I.T. license.

GLOVER: Okay. And what about Appcel...I always say Appcelerator, but Titanium?

STARK: Right. I don't, you know, I don't know, I haven't kept up to date with that. So that's something that people would probably have to check at appcelerator.com. My understanding is that the tools are proprietary but that the apps are basically yours to do with as you like. I mean, you're writing JavaScript that gets compiled by their framework and you know, I'd be shocked if they were holding any ownership over that.

GLOVER: Fair enough. Fair enough. So are there other tools that people should be aware of?

STARK: Sure. There are a lot of different tools. Another one that's frequently compared to Titanium Mobile and PhoneGap, at least back in the day was called Rhodes Mobile.

GLOVER: Okay.

STARK: And that was, and the reason these all get compared is because they were all marketed to web developers so even though they were all three completely different things. So row mobile is like a full stack MEAP solution -- so, Mobile Enterprise Application Platform.

GLOVER: Is that what MEAP stands for?

STARK: Yes. Yes. M-E-A-P. And the environment is, at least at it's most basic, is that you write Ruby code, you're basically writing... It feels like you're writing a Rails app. And but it's got components that live on the server and live on the client, you know, the phone itself, and it has built in sync management so offline support is extremely strong and powerful and like right out of the box.

There are all sorts of, I mean, it's an enterprise grade solution. And the code base was open source but they got acquired by, I want to say Motorola about a year ago which...yes, I think it was Motorola because I remember raising my eyebrows when Google bought Motorola. So it might be, Rhodes Mobile might technically be owned by Google now.

But it's for web developers who want to build cross-platform mobile apps, Rhodes Mobile is definitely worth looking at because it's such a familiar...it's such a familiar environment. And it gives you, you know, it will feel exactly like building a Rails app basically.

GLOVER: Okay. And then going back to JavaScript frameworks, there's Sencha Touch, there's jQ Mobile or...yes, no, not JQ Mobile, jQuery Mobile, excuse me. jQTouch. In fact, the list goes on. Those with, you know, the three that just came to my mind.

STARK: Right.

GLOVER: But I'm sure if I Googled it, I would have pages and pages of it.

STARK: Yes, absolutely. So those are the big three. Sencha Touch is probably the most...Sencha Touch is the most powerful. It's based on XJS. If people are familiar with that. It's kind of like similar to jQuery in concept. But maybe a little more enterprisy.

So basically Sencha Touch seems to be the mobile solution if you're building, you know, web apps either for delivery over the web or inside of a PhoneGap type container that automatically...they automatically reconfigure themselves for tablet and smartphone screen sizes. It's extremely slick.

The coding approach that...the problem with Sencha Touch there's really like two knocks against Sencha Touch. One is that there's a fairly high learning curve because the code that you write is like pure JavaScript. So if you are not a killer JavaScripter or at least intermittent level JavaScripter, you're going to be frustrated getting started with it.

The other thing about Sencha Touch that makes it not right for every job is that it's fairly large, which can be a problem when, you know, when bandwidth is an issue or memory on a device is not as much as you'd like.

So in a restricted, constrained environment type of situation it can be...it's not always, it's not always great. It's not always the right tool for the job. That said, if you're trying to make a rich web experience, you owe it to yourself to at least look at Sencha Touch and see if it fits what you're trying to build, because it's from a polished standpoint it's absolutely without peer. It is unbelievable.

So you can make absolutely gorgeous stuff with it. The closest competitor that I would keep my eye on is SproutCore, but I still think SproutCore is, you know, a team... SproutCore is basically the same concept but it came about from the desktop side. In fact, Apple used it to build the first version of Mobile Me.

And it's, it never, it didn't...I don't know, it wasn't purpose-built for, you know, tablets and phones, it was purpose-built for the desktop. So it's got as making rich Internet applications, but I just feel like it last behind Sencha Touch in like basically every category that you can...you know.

GLOVER: So clearly we have to take a deeper look at Sencha Touch?

STARK: Yes. It's definitely worth looking at. It's high learning curve and it's got a decent size footprint. So those are the two knocks against Sencha Touch.

Then coming down a notch, you look at, I guess jQTouch is the next one to look at. jQTouch is originally is an open source, also MIT license, JavaScript library created by David Kaneda who now works at Sencha Touch, to basically it's a jQuery plug-in, so it's built on top of jQuery. And it was kind of like jQuery UI for mobile, if you're familiar with jQuery UI.

So that's basically what it was. And it was, as soon as I saw it, I immediately jumped on the project and started contributing code and I maintained it for a little while. The general concept is that you write simple markup and just include jQuery and jQTouch in, you know, the head of your HTML and it like magically creates a smartphone-style experience.

It's fully customizable, you know, at the...when you, at the top of the document you call the jQTouch constructor, and you can pass in all these arguments to modify the behavior. There are all sorts of custom events like on swipe and on orientation change, all of these things that you want automatically handles a bunch of Ajax operations. It's really, really slick, totally theme-able. You can create your own themes.

And it doesn't mess with your code. It's, it feels, it's easy for sort of garden-variety web designers and developers to quickly create a really rich web experience. The problem with it, the knock against jQTouch, is that it's specifically optimized for web kit browsers on small screen devices. So it doesn't do anything fancy on an iPad or a, you know, a Touchpad, it just looks like a big, you know, just scales up.

GLOVER: I see.

STARK: And it...and you know, it's just web kit. So it performs fairly poorly in Firefox, Opera Mobile, Opera Mini, all the other browsers. So it's really not great for a public, I mean, it's fine but it's not...it doesn't degrade gracefully for those other browsers. So it's probably not the best choice in you're creating a web app for mobile.

But if you're creating a hybrid app that you know you're only going to put in iTunes in the Android market, then you know what devices you're going to be on, and it's a great choice because it's very lean because it's only targeted at those particular devices.

GLOVER: So that's interesting. Okay. So jQTouch, because it is a lightweight, you know, easy to pick up framework if you were doing that with PhoneGap or something like that and you were just targeting iOS, then you could get started and have an app, perhaps, quicker than you did...you could if you decided to go the Sencha without and you hadn't touched Sencha before?

STARK: Right. Exactly. And so, and then here comes jQuery Mobile. So jQuery Mobile bridges that gap. So if you, I usually recommend jQuery Mobile for those cases where you're not going with a hybrid app and you do want to use a market based approach to create your mobile web app because it's...you know, the jQuery team is all about the one web type of mantra.

And although it's impossible to test and support every mobile browser out there, there's hundreds of them, they do an admirable job of covering the A, B and C level browsers. So we're not going to get, perhaps, I mean, the UI is gorgeous but it's not as smooth as jQuery...as jQTouch is on web kit, but it's much better than jQTouch is on Firefox and Opera and these other browsers. So if you don't know who is going to be viewing your app, you probably want to go with jQuery Mobile.

GLOVER: Okay. All right. So then let me ask this. So we've talked about, you know, peer web apps. We've covered the middle ground with the hybrids. Do you find that there's scenarios where it makes sense, you know, back to the beginning of your statement when, you know, you're an enterprise and you were looking at, you know, all these different block forms.

Are there use cases where it makes sense to go off and build the iOS native app and then do the same thing for Android, obviously doing it in Java and then maybe doing some other platform? Maybe it's BlackBerry and that's Java again?

STARK: Yes, yes, but different, different tools and everything. So yes, I mean, sometimes you have to do it. You know, sometimes the experience demands that you have to go native because you need something that's really memory intensive or oppose you just have to have compiled codes to pull off the experience that you want.

The examples that always come to mind are like Nike or like Burton Snowboards or somebody who has a lot of animation, a lot of video, a lot of sophisticated design that is just super...in fact, it's the stuff that would have been flashed on the web, right?

Like people who want the utmost control over the user experience and, you know, and they don't want it to be, they want it pixel perfect across all platforms. So it's kind of like that stuff.

There are tons of use cases, tons of different cases where we go pure native, but you just have to realize that the market share that you can hope to reach is going to be very small in comparison. And that the total cost of ownership is incredibly high because you have to manage the whole lifecycle of the app on both platforms.

You know, just imagine you've released iPhone, iPad, Android, Android Tablet, maybe a couple different...couple different platforms of Android, Windows Phone and PlayBook, native versions of a particular app. And a bug shows up. You know, like how do you even, first of all it's not the same team it's different teams that built every single one of those apps.

So just the communication latency between all those teams and their tools and systems that they use, their release patterns and everything, I mean, it's like a nightmare. So then, okay, let's say you find the bug in every platform that exists and you fix it, and now I go to release it. Now you've got, you know, the approval policies for the different app stores to deal with.

I mean, you'd have that PhoneGap as well. PhoneGap approach because you still have to go through whatever your native distribution mechanism is, unless you're distributing it to employees, which is a whole different can of worms. But it's just impossible to maintain all of this. I shouldn't say impossible. It's very rare that it's worth the cost. That's probably the right way to say it.

GLOVER: Okay.

STARK: But what I recommend that people do in those cases where they do decide to, you know, situation where somebody deploys 5,000 iPads to their external, you know, their outside sales force, they know they only have to build an iPad app, right, because that's the device it's going on.

So in a case like that, I would say, okay, in order to be future friendly about this, you want to put as much of the smarts of that application on the server side where we could potentially leverage it in a Touchpad or BlackBerry PlayBook or some other \$35 Android Tablet down the road, you know.

But you know, so you want to keep as much of it centralized as possible without creating a bottleneck of course, and have as little of the logic as you can in the application itself. So you don't have to go through an update process all the way to the...you know, the ends user's devices.

I mean, we went through this in the desktop era. It's a nightmare dealing with, you know, air quotes desktop software, whether it's on a phone or a computer. It's really hard.

GLOVER: Yup. Yes, I agree. I agree. So where, so you did mention that you're updating one of your books. You know, where can people find more information about you and, you know, if they had more questions or just kind of where can they get more information about everything we just talked about? Where would... I'm a developer. I want to be a mobile developer. I listen to this podcast. I'm intrigued. Where do I start?

STARK: Definitely jonathanstark.com. I have links to free versions of both my O'Reilly Books. O'Reilly is just an amazing publisher and they get the future. So you can actually read my books for free online or of course, you know, Tim O'Reilly would probably be happier if you bought them. But you don't have to.

Where else? I'm very like hopelessly addicted to Twitter and I post a lot of links to things that I think are relevant for the future.

GLOVER: What's your Twitter handle?

STARK: Jonathan...so, @jonathanstark.

GLOVER: That's easy.

STARK: Yes, so if you just go to jonathanstark.com all my contact information and all that, so it's not a very elaborate website. It's just a few pages.

GLOVER: And you also speak. What about some conferences? Any upcoming conferences?

STARK: Yes, I'm actually doing a keynote in Washington, D.C. on Tuesday but I think it's already sold out. Then I have another one coming up in California in December, but I just got that one, so you'll have to go on my talks page and see exactly what the date and location is. I'm not sure if I even know it yet.

And then the next big one that's coming up is South by Southwest interactive in March. So that will be interesting. That one, in particular, I'm talking about, I don't know if you're aware of the Starbucks card mobile payment. Yes, I'll be speaking about sort of future of mobile payments and what happened with the Jonathan's Card Starbucks mobile payment experiment at that. So...

GLOVER: And actually, can you briefly, for the audience that may not have read that, in fact, I read that story I think in like Inc. Magazine or something.

STARK: It's crazy.

GLOVER: But can you give us the two-minute overview of what happened there?

STARK: Sure. Yes. I was doing research for a large fast casual restaurant chain in the U.S. We were working on a mobile payments solution for them. So I was testing the existing apps out there and Starbucks had a really, really good and clever loyalty card application.

So you could download this Starbucks mobile app and you could, you basically go online to Starbucks.com and link your credit card to your loyalty card and then when you have the app installed you just go into the store and you pull up a picture of your barcode on your iPhone and they could scan it at the POS and just pay for your coffee or whatever like that.

So at the time they didn't have an Android app, so, and I carry a lot of different phones, I wanted to test it on Android, too. So I took a screen shot of my iPhone and emailed it to my Android phone and sure enough the static picture worked, which kinds of blew my mind because I was like, man, I just emailed currency to myself.

And so of course I did what any good geek would do, I blogged about it and posted the bar code on my blog. And before you knew it, people were

downloading the picture to their phones and buying coffee but more importantly, donating money to the card so that it turned into this global pay it forward kind of, you know, take a penny, leave the penny, but for coffee.

And it was crazy, man. It was like...the thing about it was I didn't want people going down to a Starbucks to get a coffee if they didn't know if there was any money on the card, and since it was a static photo they couldn't tell from the app.

So I wrote a screen scraper that pulled the balance from Starbucks site because you can look it up online if you have the password. So I would just scrape it. And every time the balance changed, I would tweet the balance.

So people were like watching the Twitter stream like it was TV. People were like, I can't go to bed, I have to keep watching this Twitter screen. And the balance was just bouncing all over the place.

And we created a Facebook page. And it was like a full-time job managing the community on Facebook. It was totally insane. The media covered it. I mean, I was like talking to the producers at the Today Show when the whole thing got cancelled. I mean, it was crazy. It was totally crazy.

GLOVER: That's awesome. Yes, that was, I highly encourage for the listeners out there to just Google search it. I'm sure you...I can't remember if you.... I'm sure on your web site you've got some coverage there as well?

STARK: Yes. Just Google for Jonathan's Card and a thousand articles come up.

GLOVER: Very interesting. So that's very exciting. So you said that's at South by Southwest other somewhere else, you're going to be talking about mobile?

STARK: Yes. I've got some more talks like that in the future of mobile payments coming up. The one in Washington next week is about dealing with, basically dealing with content and managing experience in the era of ubiquitous computing that we're basically walking into.

Yes, so the Starbucks one is South by Southwest. And I'm doing sort of a hybrid of that in I think it's in Santa Clara in December. I'm going to talk about mobile, but really mobile's not...mobile's just a catalyst for this ubiquitous computing paradigms, this like continuous client where...

Like I'm talking right now through my computer. I'm surrounded by one, two, three, four, five screens, and I've got a box of about 20 phones next to me. And that's going to be, I mean, that's a little extreme, but I think most households are going to have a pile of different screens with all different platforms, all different sizes, all different bandwidths.

And we're just entering into this era, I mean, fragmentation isn't even going to describe where we're going. I mean, it's going to be everywhere.

GLOVER: These are interesting times, my friend.

STARK: Absolutely.

GLOVER: It's a good thing that people like you are out there, you know, helping educate the rest of us on what to expect and how to survive here. So I sincerely appreciate your time today. And I know I speak for all our listeners when I say, thank you very much, and, you know, keep up the great work.

STARK: Well, thank you very much for having me. It was fun.

GLOVER: So my guest again has been Jonathan Stark and I'm Andy Glover. And this is a Java Technical Series in the developerWorks podcast. Thanks for listening.

[END OF SEGMENT]