

BIZ & IT

Google talks Chrome OS, HTML5, and the future of software

Ars Technica's Jon Stokes and Ryan Paul sit down with the engineering director ...

JON STOKES - JAN 19, 2010 11:30 PM | 47



Aa TEXT SETTINGS

On the last day of November, 2009, after the initial rush of excitement around Google's Chrome OS launch had quieted a bit, Ryan Paul and I sat down with Matthew Papakipos, the engineering director for the Chrome OS project, and Eitan Bencuya, from Google PR. I had done my best to sort out the why's and wherefore's of Google's first consumer OS effort in my [initial launch coverage](#) but I still had many questions about the past, present, and future of the project.

What followed that afternoon was an interview that was so candid, in-depth, and informative about not just Chrome OS, but about the present and future of the Web as a distributed application platform, that we chose to sit on the results until the holiday and CES madness had passed. So, in this brief pause between CES and the coming iSlate hysteria, we present our Chrome OS interview.

Over the course of the interview, Papakipos and Bencuya go into considerable detail about topics that range from big-picture perspectives on how Google develops software and where it sees the Web going with HTML5, to the nuts and bolts of what Chrome OS is slated to offer in specific areas. In short, we cover the following ground:

- How and when the Chrome OS project was conceived
- The relationship between Chrome OS and Android
- How Google is trying to tackle the same "file handler" problem as Windows OLE and the registry, but in the cloud.
- Who Google sees as the target audience for Chrome OS, how did they decide which projects and features to pursue
- The convergence of the phone and the computer
- Nuts and bolts details, like native client execution, security, and UI issues
- The significance of Chrome's built-in media player

In the text below, the speakers should be obvious, but just in case you need it: JS is me (Jon Stokes), RP is Ryan Paul, MP is Matthew Papakipos, and EB is Eitan Bencuya.

ADVERTISEMENT

How Chrome OS started, and where it's going

JS: I want to ask about the history of this. Did you know that this was something you wanted to do when you started at Google, or did you just slot into this [as an existing project]... did you cook up the idea?

MP: How open can I be about this Eitan?

EB: The way that we've thought about this for a while is if you read the Chrome OS blog post that we did in July, and you read the Chrome browser blog post that we did in September in 2008, they're very similar. They're essentially the same thing. And the reason is because when we were building Chrome the browser, we realized that everyone is spending their time online. So essentially we were trying to build something that mimics this operating system feel—there's a task manager in Chrome, and that was one of the early additions to Chrome.

So there's this trajectory where we kind of realized that this is the direction that we're going, but we didn't start coding and having engineers working on a Chrome OS until this year.

MP: Yeah, we didn't start coding until this year. But yeah, I sort of was thinking about it ever since I got here to Google two years ago. The crazy notion of "why doesn't Google go make a consumer operating system?" But I think most of us assumed it wouldn't happen for a bit longer. The interesting thing was that the market was ready for it faster than we expected. I think part of it is the shift to the Web has been going faster than we expected, but there are also some interesting developments that made it look ready even a year or two earlier than we expected.

ADVERTISEMENT

Part of it is the netbook phenomenon—the fact that netbooks took off. If netbooks hadn't happened, and we were all still talking about laptops, I think we probably wouldn't be doing this yet. Part of what spurred it was people getting these really tiny portables and the fact that... well, [look at] their name, "netbook." I've been puzzling over that for a while, when I first started seeing them. Why are they called netbooks; there's nothing net-y about these; they're just laptops that are small. It's interesting that they got perceived from the beginning as being about the 'Net, and about the Web.

Your current computer is a somewhat unpleasant device: you put it on your lap, you sit on the couch, you start working, it starts burning your leg, you start running out of battery.

And I think part of it is just the way that people ended up using them. The people who were buying them were tending to use the browser for most of their apps, so people started calling them netbooks. But that made us realize that this was happening faster than we had thought.

When you buy a netbook, you don't really install Photoshop on it, or Outlook. It's too cheap—it's a \$300 machine, [and] it doesn't make sense to go install expensive software on it... which is great software, but you don't go install *Call of Duty 6* on it. It's not that kind of machine.

JS: Right. So the netbook proved the market for, not necessarily thin-client, but a sort of "svelte" client—a slimmed down...

MP: ... a lighter computer.

JS: Yeah.

MP: But it's still a real computer; just not the kind of thing that you're going to install major content creation applications on. Or sit down for a four-hour gaming session with your *Call of Duty* buddies.

ADVERTISEMENT

But there's also other interesting stuff that's happened. Processors have gotten lower-power, so that the low-power processors are very cool. Both the x86 stuff and also the ARM stuff. And I personally have been surprised by how fast some of the ARM stuff has been happening; and that's one of the other things that got us really excited about netbooks—the prospect of really low-power computers.

Your current computer is a somewhat unpleasant device: you put it on your lap, you sit on the couch, you start working, it starts burning your leg, you start running out of battery. Some of the new low-power ones really last for 8 to 10 hours on a charge, and some of them don't have fans. They're really cool, quiet machines that last for a long time, which is actually what you'd like your laptop to be, right? It's kind of a bummer that today, your laptop really is a two-and-half hour computer. You sit down, you do two hours of work and are like, "Ah, it's gonna crater," so you bring it back to your desk and plug it in, and you're like, "Now I don't have a laptop anymore." So then you find yourself sitting at your desk, working on your laptop, which is pretty lame.

JS: Well, a lot of us have gone laptop-only. I actually went MacBook Air-only, from a 15" MacBook, and I can plug this into my 30" monitor and have all the screen real-estate that I need. But I only do that if I have to mess with a spreadsheet or something.

MP: But the other thing is that you find yourself forced to do it, right? Just because you're using it somewhere else for the convenience, but then you have to plug in, and you're like, "I might as well sit at my desk and plug it all into everything, in that case."

ADVERTISEMENT

EB: And you carry around your charger cord, and that adds weight to your bag.

MP: Yeah, I was laughing at that over the weekend, because I was showing my dad one of these [Chrome OS portables], and I said, "Can you grab that one from the other room," so he brought it and the charger in. And I said, "Dad, it's a netbook... you don't need the charger" [laughs]. But it's so funny that that's reflexive, and people assume it won't last very long without the charger.

Who's the target audience for Chrome OS?

If you want a Windows machine you should buy a Windows machine. If you really want to use Photoshop, you should buy a Windows workstation.

JS: I had this section in my article about the audience for this, where I said, "Who is the netbook buyer? Is it the guy who wants a Windows PC and says [of the netbook], 'Hey, this is small and light,' or is it the guy who wants a small and light Internet client that may or may not run Windows."

MP: So what's the distinction you're making?

JS: There's this phenomenon of users returning their netbooks because they don't do everything they want for Windows, so obviously some people are buying these thinking it's a Windows portable and what they want to do is Windows...

MP: Yep.

JS: ...and it just happens to be a cheap way to do Windows. For instance, my brother-in-law is a doctor, and he would like to put a bunch of PCs in his practice, versus having to carry something around from room to room. When he first discovered netbooks at Best Buy, he was shocked and said, "I can have a Windows PC in every patient room for the price of a large portable that I might carry around from room-to-room." But the virtue of the netbook for him is that it's a cheap way to run Windows.

ADVERTISEMENT

MP: There was this thing last Christmas where OEMs sold a lot of machines that weren't running Windows and ended up getting a high return rate because they weren't very clear in their advertising and materials on the box about the fact that Windows wasn't on there. So people would bring it home and say, "Oh, I'm going to install *World of Warcraft*" and they'd go "Wait a minute, what is this?"

So there was some stuff like that out there. The thing that this has convinced us of is that we need to be really clear with people when they buy a Chrome OS machine that it's not a Windows machine. We do need to be really clear about that. We don't want you to end up getting a machine for Christmas and it's not one that's going to work out for you.

If you want a Windows machine you should buy a Windows machine. If you really want to use Photoshop, you should buy a Windows workstation.

But it is challenging—we haven't figured this all out. We really need to make it clear to you when you buy a machine online or in a store that this is a Web machine; it's not a Windows machine. It doesn't run Photoshop, it doesn't run *WoW*. What it does is get on the Web. And a lot of that we're still figuring out.

Google's a funny company, though, because we often get questions like, "What demographic is this targeted at," or "Who's your user base," but we don't really think of it that way.

EB: Our users are just people who use the web a lot.

ADVERTISEMENT

JS: So my question was more along the lines of, did you guys do some demographic research. Or did you find some people in some other part of the company that have some really detailed sense of the demographics of the user base via...

MP:... we don't really tend to think that way at Google. We're just really not wired that way. Because as soon as you get into that demographic stuff, it's sort of hand-wavey pseudo-science stuff. We're more like, let's get it working really quick, and let's give it out to 200 people and see what happens, which is what we did. We currently have about 225 Googlers who have gotten these machines, and what we're seeing is that a very large fraction of them are using it every week. We don't know why, but they like it.

So we tend to be very data-driven. We try something, then we go measure the results. Every day when I come in, I look at the graph of usage and say, "Oh, it went up. OK, good." If it starts dropping, I start worrying.

JS: I can see how it's the kind of thing you start to think of uses for after you know that it's there. Like, after it was announced, I started thinking about how it would be nice to have one downstairs in our TV room, because we're always thinking, "Oh, what other show was this guest star in?" So we pause to go get on IMDB and look it up. So it would be nice to have a machine there for things like that.

MP: Exactly. That's what I found in my house. I have three different machines at home, and I started leaving them in different rooms: one in the kitchen, one in the living room, and one in the family room. And it's amazing how often you just pick it up, look up one piece of information, and shut it off. And you're done with that whole transaction in 45 seconds. Which is awesome, because sometimes that's all you want. You're talking to someone and you're like, "What's that actor's name?" So you pull the netbook out, look it up, shut it, you're done.

ADVERTISEMENT

The funny thing I'm finding is that I use it now instead of my phone. Because the phone isn't a particularly fast way to send email or find somebody in your contact book. If I need to send a four sentence email I find it's faster to boot the Chrome OS machine, send the email, and power it off.

Convergence: phone and computer

■ We're so focused on shipping release one right now, we spend less time than the outside world speculating about what we're going to do next.

JS: This gets to another question of mine, which is more along the lines of "what is it." One of the questions I had as I thought more about a Chrome OS PC was, is it a stripped-down netbook or a beefed-up phone? Because a phone is a network client now, and that's kind of what you guys have [with the Chrome OS netbook]. But it's kind of like a phone in a netbook body, to a certain degree.

MP: A phone with a keyboard.

JS: To a certain degree, yeah. A phone with a real keyboard and a real screen. But it doesn't do voice... of course that doesn't matter anymore because nobody cares. If you're trying to talk on your iPhone in the Bay Area, it doesn't do voice either.

MP: Google Talk does voice, so you can make voice calls. You can even make voice calls to telephone lines, so you can use it as a phone. I don't think in most cases you'd want to use it as a phone, but it is cool that in Google talk if I'm chatting with my mom I can say, "Let's go to voice," and it just works.

JS: And this also gets at another issue. I know that Sergey made some comments that were widely reported about a future merging of Android and Chrome OS, and to me this seems more like a statement of the obvious. In the sense that, once the hardware is full-featured enough to support all the things that you have on Chrome OS, then why do you need something as stripped-down as Android.

ADVERTISEMENT

MP: Yeah, it's unclear. We get a lot of questions like this, and the truth is, I don't know the answer. We're so focused on shipping release one right now, we spend less time than the outside world speculating about what we're going to do next... how will we integrate this with Android... how will we do any number of things like that.

We're pretty focused on just trying to make it work, and ship release one.

EB: I think you hit the nail on the head, in terms of the way that phones and computers are converging. Sundar spoke about this at the event—there's a perfect storm of converging trends, so computers are becoming smaller and more mobile, phones are becoming more powerful and more computerlike, and they're meeting in the middle. Take HTML5 and netbooks and all this stuff, and it makes sense. And I think Sergey was alluding to that as well.

MP: There are people out there who say that it's all going to converge to one device. Some day I'll just take my phone and make a call on it, then I'll put it down on my desk and hook it up to a monitor and keyboard, and it'll do everything.

I'm skeptical about that. Sometimes I really do want to sit down with a laptop and do some work; and sometimes I want to sit down at my desk and have a really beefy monster of a machine. And that's always going to be better than my phone, or at least, for the next 20 years it is.

So I tend to think that we're always going to have a variety of devices. And who knows, maybe we'll have tablets, too. And other weird stuff. Game consoles—I don't really see those disappearing tomorrow. That's another kind of computer you have in your house.

ADVERTISEMENT

JS: The game console is actually one of the ones that works the best and causes the least headaches. It's one of the most successful machines in my house.

MP: And it's one of the least flexible. That's part of their philosophy—keep it simple, which is a good thing. That's kind of what we're trying to do with the browser. Just make a lean-and-mean browser experience. That's limiting, but on the other hand we can do it really really well, and that's most of what you need.

Applications vs. documents, and the nature of the Web

JS: On the Web, it seems that there's a spectrum of sites that exists between "document" on the one hand, and "application" on the other. Google Maps would be an example of an application, whereas Ars would be an example of a document or document collection with some navigation added on. But you guys put both documents and applications in a browser window, still.

I know you're still refining the user experience, so I guess what I want to know is, are you going to stay within the document/browser paradigm, or are Web applications going to become peers, with their own window and their own specific window chrome. So maybe [a Web app] is like this OS X window, so that the view looks like an application to the user, and not like a "webpage." [I'm essentially talking about moving from a document/browser paradigm to an application/window manager paradigm for a certain class of sites.]

MP: That's a good, insightful comment. It is weird that when you cruise the Web today most of the Web is still a fairly static web of things that fundamentally seem more like documents than they do like applications. Even CNN, The New York Times, the Wall Street Journal, etc. are really fairly static pages. And yes it's being served to you dynamically, and there are ads, and whatever, but fundamentally you're reading words off the page and it's not quite like something like Gmail, or Picasa, or even Netflix, where it really does feel like an app. Everything being served to you [on these sites] isn't really static at all—you can reorder lists by dragging things around, and it feels like an application.

ADVERTISEMENT

I live in the browser most of the day, and every time I have to leave that to run something that's not browser-based, that's actually more annoying than positive. So our current thinking is to keep it in tabs.

So far in Chrome, we haven't really made any distinction between those two things, because the Web doesn't really make any distinction between the two. You can't really tell when you get to a page what the page is supposed to be. But there is a lot of stuff that we're starting to work on in Chrome OS where we're starting to add specific features for things that really are apps.

A good example for Chrome OS is mailto links. Typically, those kick you out of the browser back into your Windows mail program, which, if you're using Gmail, is a complete hassle and annoying. So we're doing a bunch of work now to figure out, how do we make that actually launch Gmail. Or, better yet, if you're actually running Gmail, how do we get it to switch to that tab and open a new compose window, which is what you want it to do. Or in Chrome OS, maybe it should pop up a panel and you start writing in that, so as not to interrupt your flow in the site that you were on.

So yeah, we're starting to work on that stuff, and figure out how we get different link types to be handled by Web applications. For example, if I have a JPEG viewer that happens to be a Web application, how do I make it to where if someone tries to open a JPEG in Chrome or on a Chrome OS device, it kicks over to the right website to view that JPEG. Or, who knows, maybe I want to pull it into my Picasa collection, or any number of things.

ADVERTISEMENT

Here's another example: .doc files. If I click on a doc file, do I want it to go to Office Live? Maybe I do, or maybe I want it to use Gview, the stuff that we showed in the launch where we can do a preview of that kind of document type, or maybe we want to store it on a USB camera stick. How does the user say how he wants each type of file handled? So we're starting to do some work on that.

JS: It's like the problems that Windows faced early on, with OLE and the registry and such, but on a Webified level. It's the same fundamental problem, but with a different set of constraints.

MP: We want to solve it in a way that's simple but flexible. It has gotten fairly complicated in Windows these days. You've got all these warring applications fighting over who gets to display JPEGs—Quicktime is fighting with Windows Media Player, which is fighting with Chrome. It's getting very complicated and hard to manage for users. And the Web is just now starting to get these abilities, where you can register filetype handlers. So we're at the leading edge of that.

The other part of your question was, are we going to make those look like tabs or something else? Currently, at least, we're thinking they'll just be tabs, because we don't really want to have one browser window full of tabs, and a couple of other Windows that seem somehow different. It seems better to just have it all be tabs.

I live in the browser most of the day, and every time I have to leave that to run something that's not browser-based, that's actually more annoying than positive. So our current thinking is to keep it in tabs.

ADVERTISEMENT

But this is the kind of UI stuff that could change. We're still a year away from consumer launch, and lots of stuff could change. But that's our thinking right now.

JS: I know that Mailplane is popular around the Ars office. Do you ever use that, or have you seen it used?

MP: No.

JS: [I turned the Macbook screen around and gave Matthew a quick Mailplane demo]. It's just a thin wrapper around Gmail with some mail-specific widgets up top.

MP: Chrome actually has something like this. There's a way to get Gmail to use a special interface in Chrome to install an icon on your desktop that is Gmail, so it behaves more like a conventional desktop application.

EB: Any website can be opened via a conventional desktop shortcut.

MP: Google Gears had this ability to give you a desktop shortcut for an application and make a Web app look like a desktop app.

EB: And it's built into Chrome, now, that you launch Gmail or any site via an application shortcut with even less chrome...

MP: ...without the tabs or the controls...

JS: Right. There's this Fluid thing though that I used to make a Google Wave app. You give Fluid a URL and it makes an "app" out of it. Have you seen this?

MP: Yeah, that sounds like what he's describing you can do in Chrome already.

I think that kind of thing makes more sense for a conventional operating system than it does for Chrome OS.

JS: I guess my point in bringing that up was that, there are already ways of treating Web applications as *applications*, so that, from a user experience perspective, they're peers with other applications in the interface. Versus being shoehorned into this Web document, browser model. And to me, that's a good thing, versus going back to everything in the browser window.

ADVERTISEMENT

But this is a larger philosophical discussion.

MP: I think we have the opportunity with Chrome OS to merge those two concepts. Right now, it's weird because you have this set of tabs, which is browser-managed, and you have the operating system window manager which is managing the set of applications. And Chrome OS, by making them all tabs, we can make the window manager manage those tabs in any way we want.

So in some ways, the problem should get cleaner; the user model should get cleaner. Because there's not two kinds of applications. There's really just one kind of application. How we present that will vary over time, as we figure out what works best for users.

Right now, we have two ways to do it in Chrome OS. One is that you can see apps or pages as tabs, and the other is that you can use this overview mode to see multiple Chrome windows and move tabs or application between windows.

But we definitely are exploring the whole notion of "should there be things called 'apps' on the Web." And should they have a broader set of capabilities, like handling filetypes and handling mailto links.

Programming nuts and bolts

RP: There are a few things I'm interested in from a programmer's perspective. Particularly, will it be possible to use something like NACL to write native code that will interface with browser applications? Is NACL something that Google is thinking about bringing to Chrome OS?

MP: Oh yeah, definitely. We're planning on it.

RP: Are there any prototypes of that yet?

MP: In Chrome itself. So, native client right now is in the dev channel in Chrome, which means that by running Chrome with a special flag you can use experimental native client content in mainstream Chrome on Windows today.

ADVERTISEMENT

It's moving its way through the release process—that process takes a while, so it's not yet in the stable builds of Chrome. And it's still behind a flag because there are still security aspects that we're working on, and we don't want to turn the flag off until that's all fully resolved. But it's definitely moving its way through that whole process, and it'll definitely be in Chrome OS, too, once it's ready.

RP: Are there any plans to expose hardware capabilities through JavaScript to something like NACL?

MP: By hardware capabilities, what do you mean?

RP: For example, if someone wanted to access the webcam or GPS. I know there are already some APIs for things like geolocation, but what about the whole range of other peripherals that aren't supported with emerging Web standards.

MP: That's a great question. The answer is that we're trying to add all those missing devices to the set of HTML5 APIs, and then make them usable by JavaScript but also C++ in native client.

And you're right, we've still got a lot of work to do. If I list off missing devices right now, we're missing some sound capabilities, the webcam, the microphone, line-in, stuff like that. We don't have any multichannel output, like [Dolby] 5.1 for audio. Peer-to-peer networking and even client-server APIs are fairly limited right now, in terms of what you can do. So we're working on all that stuff.

That said, there's a huge amount to do. So it'll take a while to add new APIs for JavaScript and native client, but that's the path we're on and we're working to add those as fast as we can.

RP: So are there also plans at some point to bring a port of the Dalvik runtime to Chrome OS, so that people could run Android applications?

ADVERTISEMENT

MP: We're not actively working on it. We've been pretty focused just on the pure Web application thing. I wouldn't rule anything out, but we've been pretty focused on the pure Web app view of it so far.

But the native client [question] is really a good one, and I'm personally really excited about it. But I also want to be realistic with people's expectations, because it'll take a while. It's a non-trivial undertaking.

And the same is true of JavaScript. I would love for there to be a JavaScript API today for camera and microphone access. There are actually some fairly tricky and thorny security issues we have to sort out, first, such as: how does the user acquiesce to this, how do we make sure it's not spying on them, how do we make sure that some bad tab isn't peeking at you when you're wearing your bathrobe.... so there's some tricky issues here.

EB: And, of course, we want to do it in an open way, and that all takes time. The HTML5 process takes time. So, it's in the works but it takes time.

Privilege levels and permissions for Web apps

RP: I also wanted to ask if there's a way for an application to request heightened privileges for certain kinds of activities. For example, in the Mozilla ecosystem with XULRunner, they distinguish between internal application-level privileges for extensions and Web-level privileges. Is there a way to have a secure mechanism so that Web applications get higher privileges to more?

MP: I think it's a slippery slope. That's the challenging thing—that's how current operating systems work: there's explicit ECLs and permission grants. I guess the thing that I've learned from traditional OSes is, if you look at how that goes wrong, is that users tend to have a very hard time managing it.

ADVERTISEMENT

We have over 200 Googlers using this every week, and we tend to just inflict a new build on them and see if they use things more or less, and we just iterate from there.

If you contrast that with the Web model, the Web mostly takes the view of "you shouldn't be able to do anything bad from a Web application." Which mostly serves the Web really, really well. You cruise the Web without worrying too much about badness lurking out there. It's not 100 percent true, because of malware and browser exploits and stuff like that, but for the most part you just cruise the Web and don't sweat it too much.

We want to be really careful that we don't end up in the bad ECL/permission list kind of place that conventional operating systems are, so we're being careful with it. But we're definitely struggling with this right now, with the geolocation UI, which is one of the hardest ones we've had to add so far. It's fairly tricky; you need to add it carefully because you don't want users to be spied on in terms of where they are physically in the world—they clearly need to have some say in whether websites know where they are. But you also want to figure out how to make it so it's not obnoxious.

Like, some phones today, if you go to websites that use HTML5 geolocation, they're constantly popping up boxes that say, "Do you want show your location? I know you just told me an hour ago, but are you sure you want to show your location?" And that gets really obnoxious, too, and users get trained to just click "Yes" and not read the text. (My kids call it the "'yeah, whatever' button.") That's dangerous, too.

ADVERTISEMENT

So, we're trying to figure this out, and I guess what I'd say is, we're trying to keep it simple, but make sure that users know what's going on. But we don't have the total answer yet.

For the most part, the way browsers have been addressing this is info bars and dialog boxes, but I would say in general that we're moving more towards info bars and away from dialog boxes, because dialog boxes tend to lock up your browser experience. A good example is when Google Calendar pops up a dialog box saying I have an appointment, the whole browser locks up until I deal with that thing, which is not very good. So a lot of the browsers, including Firefox, have been moving for geolocation towards an info bar approach, where something pops up at the top of the browser window and notifies you; but you can keep cruising and ignore it, or you can decide to deal with it.

A good example is what Chrome does when you type in a password. It offers to remember the password, but you can just ignore it and it'll go away eventually. That's the path we're going down right now with our UI thinking in terms permission grants for some of these things, but it's still in flux and we haven't tested a lot of it with users, so we'll see what we learn when we try it.

Extending the Chrome OS UI

RP: It seems like Chrome extensions are going to be the primary point of extensibility for extending the UI and the operating system. Is that correct?

MP: In terms of the browser chrome, yeah, that's probably fair. In terms of toolbars or things like that that you might want to add, sure, that's the way you would do it.

ADVERTISEMENT

RP: So are there any plans to add extra UI integration points for extensions that are going to be specific to Chrome OS, like, to be able to customize features that are specific to the platform?

MP: We haven't thought of any, yet, but I don't know... that's an interesting idea. Did you have anything specific in mind?

RP: I don't know, I was just thinking about the menu stuff, and how, on a widescreen netbook there would be some value in having a sidebar infrastructure kind of thing that might not be as useful for a regular browser, but might be worthwhile in an environment where the browser is all you have.

MP: Yeah, that's a good point. And it is true that when you cruise the Web today on a netbook you see a lot of dead space. Particularly on news sites, there's a lot of dead space on the left and right edges of the screen, because nobody ever expected to run at these aspect ratios, and with such wide screens.

But that's an interesting idea... yeah, you're right, we could allow some sidebars. We have been looking at sidebar UIs for Chrome OS, experimenting with that, and with stuff like, should the tabs be on the top, on the side, on the left, on the right; or with panels, should they be on the bottom, or on the right? We've gone back and forth and tried a lot of different things, and we're not done trying yet. They keep moving around every week or two.

A lot of our approach is, we have over 200 Googlers using this every week, and we tend to just inflict a new build on them and see if they use things more or less, and we just iterate from there. And we've got another year of iteration left before consumers get their hands on it, so it's going to change a lot as a result.

ADVERTISEMENT

The native apps vs. Web apps issue, round II

MP: Ryan, I'd love to get your take [on Chrome OS]. Is there anything that you think is weird, or that doesn't compute in your head?

RP: Well, I'm taking a wait-and-see approach because I think there's a lot of potential here. But the main thing that sticks out in my mind is that it's hard at first to see what kind of value you can really deliver beyond what you can already get today with a regular Linux distribution running Chrome. And I think that really the integration is going to be everything.

But I'm still a little bit skeptical about the whole idea of making the browser be the core part of the user experience, and not having anything external to that. Because if you look at the way that people are interacting with Web services, increasingly they're using desktop client applications, particularly in the Twitter space, for example.

I think that it'll be interesting to see if there's a way to serve those kinds of needs while sticking with a browser model.

MP: Uh-huh. Yeah, and if we have to extend what browser apps can do to do that, we should do that.

JS: That's what I was talking about earlier with Fluid and Mailplane.

MP: I still think what we've got to do is go look at those apps and see, "why did this have to be created as a native app?" And add that capability so that you don't have to do this anymore.

A good example is that a lot of native apps end up being made in these things because people want to run things in the background, and people are still trying to figure that out [for Web apps]. There's work going on in "persistent workers," with the Web community trying to figure out, how do we make an API in HTML 5 that lets something upload photos in the background without consuming a tab and without having to have the website open?

ADVERTISEMENT

EB: Or notifications, for that matter. Take Twitter desktop clients, where people are notified every time they get a new tweet. There's a notifications API that would allow something like that within the browser.

MP: We're also looking at the issue of uploads, because that's a common use case, where I'm trying to upload photos to Picasa and I have to keep the tab open. Sometimes people install helper apps for that, so we're trying to figure out how to make that be a built-in part of the Web instead.

The built-in media player

MP: One thing that we mentioned during the launch that nobody's picked upon, but you guys might be into, is the integrated media player.

Another big aspect to what we're doing is we're integrating a whole media player into Chrome and into Chrome OS. People often get confused about this, and it's a fairly subtle but important point. Because in a sense what we're doing is integrating the equivalent of Windows Media Player into Chrome itself.

To have a computer, that's the other thing you need: you need a way to play JPEGs and MP3s and PDFs and all that stuff when you're offline. For example, you might just have a USB key that has a bunch of MP3s on it, so you want to be able to plug that in and listen to those MP3s. There might not be any controlling webpage for that activity, but it's clearly something you need to be able to do in any reasonable operating system or browser. So we're doing a lot of work to make Chrome and Chrome OS handle those use cases really well.

RP: Is your approach to that standards-based, with HTML5 audio and video?

MP: Yes, definitely. Which is not to say... we love Flash, too, so we run with Flash as well. But we also love video tag and audio tag and all that stuff.

RP: Are you going to be exposing more codecs through the video tag?

MP: We're not specifically looking at adding more codec support, but what I'm talking about is really orthogonal to the video/audio tag. Because the video/audio tag is designed for when you have a webpage that says, "I want to play this video." But there's also a use-case where you just have this USB key with an MP3 or a video file on it. How do I browse that when I'm not necessarily on any website whatsoever?

Another example is when someone mails you an MP3 in Gmail. You shouldn't have to exit the browser to listen to that. We want to make that work directly in Chrome, and make it a really clean, smooth experience in ChromeOS.

So that's the other big part of what we're doing on the UI front, is integrating a whole media player framework into the browser. Which is not unusual, because other browsers do this, too. But it's not something Chrome has done historically.

JON STOKES

47 COMMENTS

PREV STORY

NEXT STORY

MOST READ



1. **After \$380M hack, Clorox sues its "service**

2. **Two major AI coding tools wiped out user data after making cascading mistakes**

3. **Hackers—hope to defect to Russia? Don't Google "defecting to Russia."**

desk” vendor for simply giving out passwords

4. [Study sheds light on why some people keep self-sabotaging](#)
5. [iPadOS 26 preview: The rare software update that makes \(most\) old hardware feel new](#)



Ars Technica has been separating the signal from the noise for over 25 years. With our unique combination of technical savvy and wide-ranging interest in the technological arts and sciences, Ars is the trusted source in a sea of information. After all, you don't need to know everything, only what's important.



Manage Preferences

© 2025 Condé Nast. All rights reserved. Use of and/or registration on any portion of this site constitutes acceptance of our [User Agreement](#) and [Privacy Policy and Cookie Statement](#) and [Ars Technica Addendum](#) and [Your California Privacy Rights](#). Ars Technica may earn compensation on sales from links on this site. [Read our affiliate link policy](#). The material on this site may not be reproduced, distributed, transmitted, cached or otherwise used, except with the prior written permission of Condé Nast. [Ad Choices](#)

MORE FROM ARS

- [ABOUT US](#)
- [STAFF DIRECTORY](#)
- [NEWSLETTERS](#)
- [GENERAL FAQ](#)
- [POSTING GUIDELINES](#)
- [RSS FEEDS](#)

CONTACT

- [CONTACT US](#)
- [ADVERTISE WITH US](#)
- [REPRINTS](#)