05062019 Build Technical Keynote Rajesh Jha

**Build 2019**

**Technical Keynote**

**Rajesh Jha**

**May 6, 2019**

(Video segment.)

**ANNOUNCER:** Please welcome Executive Vice President, Experience and Devices, Microsoft, Rajesh Jha.

(Cheers, applause.)

**RAJESH JHA:** Welcome. Welcome. I'm Rajesh Jha, and I lead Windows, Office, and our devices platform and browser teams.

I'm humbled by the responsibility, and also the opportunity to be here today and share with you the work we've been doing.

Earlier this morning, Satya talked about Microsoft 365 as the world's productivity cloud. That aspiration is core to our belief, the aspiration to build experiences that transcend applications, that transcend even Windows and Office, and to help individuals and organizations do more, achieve more.

Now, if you're here today to learn more about Office and Windows, you're still in the right place. (Laughter.)

Microsoft 365 has been a journey for us and we invite you to come on this journey with us today and I want to give you an update on our progress. It is our intention with the Microsoft 365 platform to enable developers to be more effective, more efficient, and more innovative, no matter what framework you use, no matter what layer of the stack you work on.

But first, I want to go back 30 years. Mark Wiser, he was a visionary, a computer scientist, CTO at Xerox PARC in Palo Alto. In 1988, Mark conceptualized the notion of ubiquitous computing, the idea that computing was going to be pervasive in our lives. In 1991, he wrote this article in Scientific American about the computer of the 21st century.

Now, Mark talked about great technology disappearing in service of the people it serves today. Let's come to today, computing is cheap, connectivity is cheap, devices are cheap, we have computers in our homes, in our offices, in our meeting rooms, in our cars, in our pockets, in our bags. But today, they have not disappeared in the background, they haven't faded away. They reach out, they grab us, they distract us.

Now, for most of us, research has shown it takes us 25 minutes to get to peak efficiency once we start to work on something. But we get interrupted every three minutes. I'm sure this session today is going to be an exception to that interruption every three minutes.

So, look, you know, people look for their phones before they look for their wallets. I have walked into walls while being distracted by devices. (Laughter.) And so today, you know, it's not like what Mark said. Today, it's actually the device at the center.

Our world is that it's the human being in the center, it's people at the center, it's users at the center, and devices and applications are a means to an end. They are to help people work on what really matters. And the Microsoft Graph represents the people's context, it represents their documents, their meetings, their projects, their goals, who they work with, what they work on. And when devices and applications connect to the Graph, they get to be people centered no matter what application or device people work on. That is our world view.

With Microsoft 365, we build people-centered productivity experiences for work, for life, and in schools. It's a set of applications, devices and services that help people communicate, collaborate, learn, coordinate, work together, do more, achieve more.

In terms of products, we've brought together Windows 10, Office 365, mobility and security. We are working to seamlessly integrate this for users, IT professionals and for developers.

Now, Microsoft 365 works on all major platforms, on all devices. In terms of the value that we seek to deliver for customers of Microsoft 365, it starts building connected experiences and devices that connect to the graph so they're people centered, they're intelligent, they're interactive. We've taken our scaled products like Office and Windows and we've connected that to the Graph, made them people centered. We've built new applications ground up to be people and user centered with Microsoft Teams and Stream. And we've extended these experiences so as developers, you can connect business processes and line-of-business data right into where users are today.

And as Microsoft 365, we're working to bring enterprise-grade security, compliance and privacy for organizations small and large, for individuals and their families.

We are gratified by the progress we are seeing on Microsoft 365. Satya talked to some of these, let me just add to that.

Every 0.76 seconds, a Windows 10 device is coming online in the enterprise, just the enterprise alone. Every three seconds, there are four Windows 10 devices coming online. With 180 million commercial subscribers of Office 365, it is the most popular commercial SaaS service on the planet, growing amongst the fastest.

I talked about the graph providing intelligence and people-centric experiences, 100 billion transactions a month against the Microsoft Graph, applications we've written, and applications you've written.

But beyond the numbers, let's take a look at how customers are using Microsoft 365 and then we're going to come -- Mary's going to come and show you some of the experiences that power Microsoft 365. Thank you.

(Video: M365 Customer Compilation.)

(Applause.)

**MARY VOTAVA SHEPPARD:** Good morning. As Rajesh said, the Microsoft 365 experience all centers around people. With that in mind, I'm going to show you some examples of how we've invested in the user experience to help people get their work done wherever they are, collaborate with their teammates, and stay in their flow.

I'm going to start with Outlook on my mobile phone. Now, here I have some emails and meeting requests, and I can take quick action straight from my inbox. For example, clicking on RSVP in this first message from Travis gives me a micro view of my calendar, so I can see exactly where this meeting fits into my schedule. Looks good.

Now, here's one of my favorite features that not many people know about. I have a message from Jordan saying, "When can we meet?" Rather than switching over to my calendar, I can just reply to this message. And then if I click this little calendar icon on the left of my screen above the keyboard, you'll see that it lets me reply with "send availability," which lets me choose a few different time slots that work for my day, and then with a single tap, this nice email is generated that lets the recipient choose a time that works for both of us.

Now, one of the challenges that many of us face at work is actually finding time to get high-quality work done, because we spend 80 percent of our time in meetings or on the phone or writing emails. And when we do find time to work, we're often inundated with distractions.

I'm excited to announce two new features coming to Microsoft 365 that can help. The first is focus plan in MyAnalytics, which helps me carve out time for uninterrupted work.

The new MyAnalytics email digests not only provide me key insights about my work patterns, but you can see here, it's used AI to detect that I made a commitment in email where it says I'll work on the doc for Tuesday's meeting, and then proactive suggested a few different time slots in my calendar so I can actually complete this task.

This brings me to our second announcement. We now support actionable messages in Outlook Mobile. Actionable messages let users take action directly from their inbox without needing to leave their flow.

So, in this example, if I click one of these look buttons, you'll see that it's updated this message to show the time is blocked on my calendar, and I never needed to leave my inbox. Developers can now build actionable messages that work across Outlook endpoints, including mobile. Be sure to optimize for mobile first, so your designs scale well across all devices.

Now, another app that I use daily on my phone is Teams, which is my hub for teamwork. Oh, I can see that I have a meeting that's already started, and I can join this directly from my lock screen. After I've logged in, you'll see it takes me directly into the meeting lobby where I can adjust my settings before joining this call.

Hey, Travis. Now, it's often difficult to feel like an active participant when you're taking a meeting from a phone, but with the connected environment of Microsoft 365, powered by the Microsoft Graph, I'm able to access and share files from my OneDrive within a Teams meeting directly on my phone. Watch how easily I can do this.

I have a presentation that I need to share here. You can see it shows up right in recently used. And then once this loads, watch how easily I can even progress the slides just by swiping on my phone. So, I don't need to be at my PC to be fully involved in a Teams meeting.

But now I do want to move over to my Windows PC because this is where I spend the majority of my time when I really need to focus on more complex or involved work. We've been listening to your feedback and we understand that interruptions and clutter don't help with that focus, so I'll share a few subtle improvements that we've made that make Windows more effortless.

First, you'll notice we've refreshed it with a new light theme. The start menu has a sleek, one-column design with fewer and more relevant tiles by default. And I'm happy to show you that now all editions of Windows allow you to pause updates for seven days at a time, up to 35 days, so you can manage updates on your own time. And maybe you don't want it to update in the middle of a presentation, so I'm just going to go ahead and pause that.

Now, another thing that you might relate to is when you're right in the middle of something, I mean, you're really in your zone, and all of a sudden you're interrupted by a notification. I'm happy to show you that we now have the ability to manage notifications directly from the pop-up itself, making it even easier to avoid disruptions to your flow.

Speaking of not wanting to be interrupted, earlier in Outlook, I scheduled focus time. Now, that time has begun. If I switch over to teams, you can see that it's updated my status to "focused," which is a mode for do not disturb so I can work uninterrupted. Only urgent notifications or those from contacts that have priority access during my focus time will pop up.

So, let's get productive. I have a proposal that I need to create for a work project, and I'm going to start with some online research. Now, I'm using the next version of Microsoft Edge, which is built on Chromium open source, this is my new tab page. Since I'm signed in with my corporate credentials, I have access to my recently used files and documents as well as corporate resources.

But I'm going to switch over to this tab to share with you a new feature that we announced this morning. Collections is the brand-new feature that we're previewing today that allows you to collect, organize, export and share content from the web easily.

Now, here, if I open Collections, you'll see that I've already started a few different projects and I'm going to open the one that I'm working on for my work proposal.

Now, I've already collected some information already, but I want to copy over an image. Watch how easily I can just drag and drop this into my collection. There it is. And then as you saw earlier, the really cool thing about Collections is what you can do after you're done gathering all the content.

So, up here, I can export it, send it as email, or export to Excel or Word. I'm going to export to Word and show you, once again, how Edge creates this clean, fully formatted document that even includes citations.

(Cheers, applause.)

Very cool, right? And this is another example of the Microsoft Graph breaking down those barriers between apps and creating a truly connected experience.

Now that I'm here in Word, I want to show you how we're taking advantage of the Microsoft Graph and AI here. So, some of you may be familiar with Ideas in PowerPoint and Excel. And I'm pleased to announce that we are now bringing Ideas to Word. Let me show you a couple examples of what it can do.

So, I'll of course start by clicking on Ideas. Now, here, the first thing I get is some quick info about my document, such as how long it might take to read it, some acronyms that I'm going to encounter.

If I scroll further down, you'll see that ideas has given me a sort of checklist of recommended writing improvements, almost a little bit like it created a list of compile warnings for me to debug. I'm going to look at what it suggests here for inclusiveness.

Here, you can see it's suggesting I change "chairman" to "chairperson" to be more gender inclusive. This was a great catch.

Now, Ideas can also help me rewrite sentences beyond just spelling and grammar. So, up here, continually expanding feels a little bit clunky. I'm going to see what it suggests. Now, rewrite gives me a few different options, and with a single click, I can change this so the sentence reads much more smoothly.

Now, I want to show you another example of how we're using AI in Word. I sometimes leave notes inline for myself so I can stay in my flow and come back to things later.

Now, intelligence in Word will detect that my intent here is to create sort of note to self, see if it does this. Come on. Yes, there it goes. And it gives me the option to convert this into a task. And when I'm done, I can check this off and it disappears.

Now, if I want to get someone else to do this, I can at mention them right here, let's pick Alyssa. And when I do this, my colleague will receive an email with some context and the comment that I need them to respond to.

Now, I'm going to show you what that looks like in Outlook. Here, you have another example of an actionable message in Outlook. My colleague can click on this, add a comment button, and insert their response directly into the message. When they do, it updates my Word document and they never needed to leave their flow in Outlook. The most important thing to take away here is that Microsoft 365 is a people-centric experience. These were just a few examples of how we're innovating and using technology like the Microsoft Graph to help people get their work done wherever they are, collaborate with their teammates, and stay in their flow.

We're so excited about what this technology can do for our users and about what it can do for you. Now, I'll hand you back to Rajesh.

**RAJESH JHA:** Thank you, Mary, thank you. (Applause.)

So, now let's take a look at the platform that underlies these set of experiences and how you can build these AI-driven experiences, people-centric experiences in your own applications.

Now, I know that over the years, we've had multiple frameworks, multiple technologies, lots of changes. But we have clarity on the Microsoft 365 platform, and I'm excited to share that with you today.

As we spend the next hour going deep into the platform, there are four things I'd like you to keep in mind. The first, the Microsoft 365 platform is optimized for developers. And the way we do that is by building our own applications on the same platform with the mantra, "First party equals third party." Second, as you know, there's compute, intelligence and data. In the cloud, there's compute, data and intelligence. On the edge with the Microsoft 365 platform, your applications can span both the tiers.

Third, with the Microsoft 365 platform, you can take your experiences wherever users are on any device. And, finally, with the Microsoft 365 platform, you get enterprise-grade security, privacy and compliance.

Now, as we go deeper into the platform, we'll start with the intelligent cloud. We'll take a look at the innovations in Microsoft Identity and Graph and how it flows into your applications no matter what framework it is built on, whether these are native applications, web applications, or even our application tiers.

Microsoft Identity is the foundational layer. It starts with the people and it moves as the person moves from app to app, device to device, Identity secures that for them. Microsoft Graph, as we spoke about, represents the users' context, their documents, their meetings, who they've worked with, what they're working on, their conversations.

Now, on the intelligent edge, we know that end users have a choice of devices. And they're going to pick the device that best works for their given context or their task. Many developers will choose to build native applications to take full advantage of the device or optimize for the device.

Today, we'll talk about when you build native applications in Windows, how you can reuse your code across different platforms. Web is, of course, a very popular framework to build your experiences across different platforms. Today, we'll talk about how we make it easier for developers and end users.

Microsoft Teams, SharePoint and Office, are at massive scale, hundreds of millions of users. As a developer, you put your data and you put your logic into these apps, you get to go cross-platform with these apps, but more importantly, you get to reach the users that are already in these apps.

So, you know, in the next hour, we'll go to six chapters of this platform, chapters one and two on Identity and Graph, and then chapters three through six on native web and the application tiers.

The first, I wanted to share a little bit of our own journey onto the intelligent cloud. We did not start out being people centered, true confession. We were very app centric. We had an application server, Exchange and Outlook, that did meetings, that did messaging, that did tasks. We had SharePoint application server with Office that did documents and content management and workflows. Then we had Skype and Lync for real-time collaboration. And with the move to Microsoft 365 over the last five or 10 years, we have just relentlessly worked to get much more people centric, not application bound, but people centric.

And then the other thing that we've been working on is, you know, we used to have a different set of experiences for consumers. We had Hotmail and OneDrive and Skype and for organizational users, we had Exchange and we had SharePoint and we had Lync. Again, with the move to Microsoft 365, we've become people centered, whether they are at work, whether they are in school, or for using these applications for their whole lives.

So, let's start with chapter 1, Microsoft Identity. Like I said, it's the foundational layer of the Microsoft 365 platform. It allows individuals and organizations to secure their applications, their data and their devices. Microsoft Identity is at massive scale and we've done the work to unify for a developer using Microsoft Identity across all Microsoft identities, whether this is a personal identity like Microsoft Account or whether it's an identity shared by your organization's Azure Active Directory.

I'm particularly excited about the work we are doing to make Identity not only secure, but seamless for our end users with our work on passwordless authentication.

To show you how you can use Microsoft Identity in your own applications, let me invite up Aashima. Thank you.

(Applause.)

**AASHIMA NAURLA:** Hi, everybody. My name is Aashima Naurla, and I don't know about you, but I have way too many passwords.

Passwords are a high cost for the helpdesk and the weak link in the majority of data breaches. Eighty-one percent of breaches are due to poor passwords. At Microsoft, we're on a mission to rid the world of passwords using a variety of technologies like Windows Hello, Microsoft Authenticator App and FIDO2 security keys. In fact, let's look at a few of our passwordless experiences as they're available today and see how you as the developer can benefit from those with little to no work. I hope you will join us on our journey to rid the world of passwords.

I'll sign into my machine using my personal Microsoft account. Windows Hello just authenticated me without a password. Now, let's look at this sample application that I have built using the Microsoft Identity Quick Starts. Yay, I'm automatically signed in. This is the best possible end user sign-in experience. And as a developer, you have to write no extra code to benefit from single-sign-on or passwordless authentication.

Now, this is a web application, but it could very well be a mobile app or a desktop app or any kind of an app.

Next, I'm going to sign into this application using my work credential. First, I'm going to sign out of my personal Microsoft account. I'm going to sign in using my work credential. My work account has been configured to use the Microsoft Authenticator app. Microsoft just sent me a verification code on my personal device. I will complete the sign-in on my personal device without using any password. It's that simple.

As a developer, you have to write no extra code, or you can use the same code in this case to gain from authenticating across personal accounts or work or school accounts provisioned by Azure Active Directory.

Now, let's look at what it takes for you to develop this application. The first thing you need to do is register this application with Microsoft Identity. This is our new application registration experience. It is the one place for you to manage all your applications across Microsoft Identity. Microsoft Identity authenticates across all our services from Microsoft 365 to Azure to Dynamics and to Xbox.

Now, let's look at the sample application that I just demoed. All I had to configure was the display name, the supported account types, which is personal accounts or work or school accounts or both, and the redirect where my application receives the tokens. That's it.

This portal represents a significant simplification of app registration experiences, thank you to those of you who previewed and provided us feedback. Today, we're happy to announce that this new app registration experience is now generally available. (Cheers, applause.) Thank you.

Now, let's look at the underlying code for this sample application. This sample application is built using the Microsoft Authentication Libraries, or MSAL. Let's look at what it takes to sign in using MSAL.

First, you configure the MSAL library, then you instantiate user agent application and you call login pop-up. At this point, an end user is signed in.

To call an API, you call the acquired tokens function for the corresponding API scopes. MSAL takes care of acquiring, caching, refreshing and managing the tokens for you.

Visual Studio 2019.1, which I'm using here, is using MSAL.net to gain single sign-on across the Visual Studio tools. The authenticator app that I was using earlier is using MSAL. Thousands of third-party applications are in production using the MSAL libraries. Today, we're equally excited to announce that MSAL.net and MSAL.js are now generally available. (Applause.)

The MSAL libraries depend on our Open ID Connect certified end point. Partners like Apple are using end point for their upcoming release of iOS and MacOS. Apple has significantly streamlined end user experiences for Microsoft identities on Apple devices and reduced developer complexity and support costs.

My team and I are super excited about all the innovations in the unified identity stack. When you use our identities, you gain built-in benefits like advanced security, passwordless authentication, and the capability for your application to comply with conditional access policies set by IT. Your application is also eligible to be discovered on the Office 365 app gallery on the Office 365 app launcher, and within Microsoft Search on Office.com.

We are honored that there are over 1 million active third-party applications using our identities. With the generally available release of the app registration experience, the release of the MSAL libraries and the Open ID Connect certification of our end point, we're pleased to say we've made it even easier for you to securely sign in over a billion users.

I would like to personally invite all of you to develop your next application using our Unified Microsoft Identity Stack and help us rid the world of passwords. Thank you.

(Cheers, applause.)

**RAJESH JHA:** Thank you, Aashima. OK, so let's move to chapter two. We talked about Microsoft Identity as chapter one, let's talk about Microsoft Graph.

As we've discussed earlier, the Microsoft Graph has all the relevant productivity schemas for users and organizations. And today, with Microsoft 365, it is one of the most comprehensive graphs of organizational signals on the planet.

But the graph exists to serve users more contextual, more relevant, more intelligent experiences, but the users and organizations are always in control of the data with Microsoft Graph.

In terms of how as a developer you will build on the Microsoft Graph, we've built high-fidelity REST APIs and web hooks that you can call into the Graph, you can fetch the right data, annotate your business objects and new experiences and build intelligent, people-centric applications.

We also today will talk about a second way to build these kind of intelligent, interactive applications for Microsoft Graph. You can take your data and you can put that into the Graph. And when you embed your data into the Microsoft Graph, your data, your logic gets to participate in our first-party experiences, whether these be Office or Microsoft Teams or even the Windows Shell with notifications and the timeline.

Today, we'll also talk about a third way that you can build applications on Microsoft Graph. These are analytics insights applications. You take the data in the Microsoft Graph, you take the data in your application, move that to Azure Data Lake, use an AI ML tool chain of your preference and then drive the insights for customers, the customers control the compliance and the security boundary on the data.

The Microsoft Graph, though, is more than just data. We have a rich set of services on Microsoft Graph, and Satya talked about some of these today. Mary showed you Microsoft Adaptive Cards. The best way to think about Adaptive Cards is it's a standard UI on custom data that can be hosted on a bunch of different canvases. It can be hosted on our canvases, you can host it on your canvas.

We'll talk more about Microsoft Search in a bit, but it is, again, going back to the people centricity, helping people find information across any application data in first party or third party.

We talked about Microsoft Graph data connect as a means to bring your data and the graph data together to build insights in analytics apps. Satya showed you this groundbreaking Fluid Framework collaboration technology. In the next few months, we'll make that available for you to host in your applications, hyper scale, incredibly fast, multi-modal collaboration, humans to humans and humans and machines working together.

Azure Cognitive Services, industry-leading vision, speech, language understanding that you can use inside of your applications, we start to use a lot of these inside of Microsoft Office, these Azure Cognitive Services.

Let's also take a look at Bot Framework. Satya talked about the world being a multi-agent world. Customers, developers, they want to build their own virtual assistants that understand their unique data, that understand their unique business logic, have a conversational interface, intelligent and interactive. With the Microsoft Bot Framework, we now have a solution accelerator that lets you build these virtual assistants. You can easily extend these virtual assistants with custom skills using Visual Studio. These skills can tie back into the Microsoft Graph and customers all over the world are starting to build their own virtual assistants using the Bot Framework.

Sometimes, they want to talk to their employees through this virtual assistant, sometimes they want to talk to their customers.

So, as you can see, the Microsoft Graph is about providing intelligence and people centricity. We have data, rich schemas, and we have service. To show you the art of what’s possible of this Microsoft Graph and how you can use this inside of your applications, let me invite up Jeremy. (Applause.)

**JEREMY THAKE:** Thank you, Rajesh.

How many times a day do you alt-tab out of Word to jump to another application to try and get the information? Right? Imagine if you could stay inside of Word to get right at that information. Today, I'm going to show you how you can get Microsoft information into your apps and then I'll show you how you can get your data into our Microsoft apps.

Now, Aashima demoed earlier how easy it was to authenticate with a both a consumer account and an organizational account. With the Microsoft Graph, with both those accounts, I can call the REST APIs and get access to people's calendar, tasks, files and much more. All, obviously, with the consent of either the user or the organization.

Now, let me show you how easy it is to add Graph-powered services to the existing Microsoft app.

I have Visual Studio Code here on the left-hand side, and with one simple line of code, I can add a new web component to the page. This will show me frequent contacts, which is actually calling the people API. You can see if I hover over the names, I'll also get the names of all those people, too.

Now, these web components can be used not just in View like I'm using here, but they can also be used in things like React, Angular, or whatever new framework comes open in the next few years that will support this.

Now, we know that you will want to customize these templates, so we provide a baseline template here that just renders those photos. But again, we have the ability through these templates to just add some simple lines of code, where you can see here I'm going to show you the display name, the job type and the department. And right there and then, you'll see that that's been refreshed and the ability to customize it.

Now, last year, we showed you and announced a new full browser control, and in that full browser control is something we actually use internally within OneDrive and SharePoint.

Now, the benefit of this for you as a developer is that you can drag and drop those files directly into your app. And under the cover, that's uploading that into OneDrive. Now, this gives you the benefits of full previews, it gives you the benefits of things like coauthoring support and a trusted security model.

Now, if I just expand this, what we've shown you today is something new. It's called the Microsoft Graph toolkit. I'm really pleased to announce the public preview of this on GitHub today.

Now, this toolkit includes a few components which you saw in that sample app. We have an agenda control, that signing control, which actually is using the MSAL libraries under the covers as well as the people, person and task API.

I'm really looking forward to seeing what you contribute back to these web components libraries as well as seeing what templates you use in your own sample apps that you'll share with the community.

Now, there are many other really awesome ways to use the Microsoft Graph. So, I switch over to Outlook here, I love showing this example of Survey Monkey. They built a feedback system that uses the people API to suggest colleagues that you can ask feedback from. The people API takes into account things like who I collaborate with over mail, who I spend the most time coauthoring documents with, and who I spend the most time in meetings with.

Survey Monkey actually sends the mail to me as an actual message and the nice thing about that being an actual message is that right there from Outlook, I can come in here and decide I want to get feedback from Megan and Ling and click "request."

Now, as someone who is being asked for feedback, if I jump over to my iPhone here, you'll see that inside of Outlook Mobile, if I open this message, this is also sending an apps-level message. And you can see here that Amy is requesting that feedback.

Now, right here, again, I can respond and answer those questions. And, immediately, I've given that feedback into the system. Now, as Mary mentioned, Outlook Mobile now supports actual messages in general availability, which is going to be really good for developers to keep people in their flow.

Now, I'd like to switch gears a little and talk about the future and the next steps of Microsoft Graph.

So far, I've shown you how you can integrate key organizational data like files and contacts into your own apps. But now I want to show you how you can get your data into our Microsoft apps.

In the coming months, you'll soon be able to take advantage of connectors to include your own external files and user-defined types, all powered by the Microsoft Graph. The Microsoft Graph Connectors unlock some really amazing scenarios, so I'd like to show you a few things we're working on.

Now, I'm sure you're all familiar with this OneDrive and SharePoint screen here and you're using this on a day-to-day basis. But the hawk-eyes among you might actually notice that we're not just showing those OneDrive and SharePoint files. You can see I'm actually surfacing an on-premises file here from a media file server. And then below that, I'm actually showing a non-file in a search experience, and this is a user-defined type. In this instance, it's from a partner called Service Now that has a service ticketing system.

They've taken advantage of Adaptive Cards. So, when the user expands an Adaptive Card, they'll see things such as when the ticket was opened, who the ticket is owned by, and any graphics that you wish to show to make that more useful to the user.

They can also take advantage of actions as part of Adaptive Cards so that user can change the status of that ticket or even close that ticket without leaving that search experience.

Now, this same Microsoft Search experience is available elsewhere. I love the latest build of Windows 10, where if I click the Windows key, I immediately see my search results when I type in queries. And you can see here that I'm showing those user-defined types in this experience as well. This can easily be added to other experiences like Office. So, right from the top here in the top pane, you'll be able to do that same query result. And because we're in Word, we'll show just the file context first. But if I click "see more," you'll see that those same user-defined types will show up on that right-hand side.

Here, I can link to that service ticket or even reuse some of the content straight from that with one click. Now, Microsoft Search is also available inside of Bing, and so side by side with SharePoint and files, you'll be able to see those same service tickets, all in line with those internet results around that particular query string.

The great thing about your data being connected to our apps is that it's not just going to show up in search experiences. If you're anything like me, you're forever trying to find people in the organization, whether it's job title or where they are in the org chart or what building they're in.

And I often go and look to see what files people are working on. Now, if I click and look at those files on the right-hand side, what you'll actually see is that we are showing those same on-premises connected files inside of that experience, all powered by the Microsoft Graph.

This gives users the ability to see all of their files across all of the different services they're connected to, without leaving that experience. Now, my favorite place to show these connected services is inside of Word. Mary showed earlier how easy it is to add inline actions inside of the Word canvas. The cool thing for you is that as a developer, you can train and expand the Word canvas to understand more intent.

So, for instance, if I went in as a user and added "add known issues from Service Now," the word canvas can be trained to understand and then suggest results to come up. And in this case, with one single click, I can add that service ticket straight into my Word document. This opens up so many potentials from those connectors.

Now, we will be announcing a private preview of Microsoft Graph Connectors in the summer, and we're really looking forward to seeing what amazing things you do with this extensibility story. All of this is powered by the Microsoft Graph.

Now, we also know that developers want to use data to build insights. And we understand how important Microsoft 365 data is to that insight. So, rather than you go directly against our REST API and bring all of that data into your application to do that insight, we thought we'd make it easier for you. We've built a pipeline that connects Graph data and moves it into Azure, all while staying within the compliance and governance boundaries of your customers.

We call this pipeline Microsoft Graph Data Connect. Let's roll the video to see how our customers are using this today.

(Video: Microsoft Graph Data Connector/Dior.)

**RAJESH JHA:** So, now let's talk about the intelligent edge. First, let me just share a little bit about how our own applications have evolved. Today, Microsoft 365 apps that we build use all of these frameworks -- Word, Excel, PowerPoint, Outlook, as you know are native applications, Win32. OneNote is a native application built in UWP.

And in all of these cases, we've connected it to the Graph, made it people centered, we've refactored our code and we've taken that to all other devices and platforms. Microsoft Teams is a web application at its core, and then we use the Electron framework to wrap it up and make it a device native application. Microsoft SharePoint is a web application as well.

Planner is an example of an application that exists as a module in Teams and SharePoint. So, now let's go into chapter three and four about building native applications and building web apps.

Microsoft Windows, as you know, powers a host of modern devices -- 800 million end points running Windows 10 -- PCs, laptops, two-in-ones, of course. But this morning, you also saw the HoloLens Version 2 running on Windows 10.

And as you know, Windows 10 also powers our large-screen devices, the Surface Hub devices for collaboration and meetings, and three weeks ago in New York City, we launched the latest member of the Surface Hub family, the Hub 2X. Let's take a look.

(Video: The New Surface Hub.)

(Applause.)

**RAJESH JHA:** These are devices that were built ground-up for creativity, teamwork and collaboration. Today, we'll talk about how, with Windows 10, you can build native applications no matter what framework you're using --WinForms, WPF, UWP, Win32.

In either case, you can take advantage of the cloud, the Microsoft Graph, and make intelligent applications that have natural user interface, whether it be ink or speech, that can take advantage of innovations in Fluent UI.

And then we'll also talk about how you can reuse your code in these native applications across different platforms.

We'll also talk about web applications in the chapter four. The new version of Microsoft Edge, as you know, is based on the Chromium Open Source Project, it now has broad compatibility with the web. With its ability to run IE and ActiveX controls natively in a tab, it has brought compatibility with the enterprise applications. Microsoft Edge will run on all versions of Windows, Mac, iOS and Android. It is now the browser for all your needs.

Today, we'll talk about how we're going to make it easy to build a standards-based web application on the new Microsoft Edge.

And then we are investing to make the web better for everybody. We're making changes back to the Chromium Open Source Project, any browser, any framework, like Electron, that depends on it gets better.

Now, to show you how you can build rich applications on both Windows and web, I'm going to invite up Kevin. Thank you.

(Cheers, applause.)

**KEVIN GALLO:** Good morning! When you put people at the center of the experience, it frees you to dream about the most optimal experiences for your employees or customers.

So, for any given task, you can choose the right device with the right capabilities. To support this, the Microsoft 365 platform delivers a rich set of frameworks and tools so you can be productive when building these modern experiences.

Today, I want to show you how you can build great Microsoft 365 experiences for two of the most ubiquitous canvases in the world -- Windows and the web.

Let's kick it off with Windows. We're going to walk through a scenario for a regional power company. One of their goals is to help customers plan their solar panel installation. To do that, they send mobile workers out to the customer site to do in-person estimates. They wanted a device that works great while walking around. So, they selected the Surface Go because it was lightweight, portable and supports pen, touch and keyboard.

Let's take a look at the native UWP app I'm building for the field sales team.

So, I'm just going to run it right here. We built this app in C#. It uses some of our new open-source WinUI controls like navigation panel over here and the mapping control here in the grid. We also instantly logged in. You'll see it went right to the user identity and it logged in using the unified identity framework that you just saw talked about.

And because we have unlocked identity and the Microsoft Graph with logging in, I now have access to the employee’s calendar and their location information. So, I was able to bring them right to where they're supposed to go and give an aerial view of the home that they're going to go do this installation at.

Now, when we originally built the application, it was a very manual process. Let me show you. We had manual code here, you know, and I have to kind of guess as an employee kind of how big the roof was. And, you know, it was error prone, sometimes it led to incorrect estimates, and didn't close as many sales as we'd liked.

So, we're going to go and build a more optimal experience now that we have access to the pen and touch and this great device on the go.

So, I'm going to show you this smart ink experience that we've built. So, now I can sit there with the customer and I can, you know, make suggestions and say, you know, what if we put the solar panels over here in the south? I know in Seattle, we'll get more sun. Got to really know where there's sun here. And then maybe over here, I can go there with maybe another alternative location.

Then I can hit this analyze ink, and it will automatically go, recognize those rectangles, and draw them in a much more beautiful way and in a way that I can now locate the solar panel. This is using our new ink recognize Cognitive Service that we release last week.

Now, it does way more than just rectangles. It can do complex shapes, of course handwriting, it can do diagrams, timelines, flow charts, a lot of great capabilities built into this new ink service.

Now that I've got it recognized and I can go maybe take an idea saying, hey, what if we chose a location over here? And it will go automatically calculate that there are eight panels that can fit there. Great. No, I don't have to do that calculation as an employee. I can get this rich, immersive experience. Again, really bringing the customer into the experience about what they're trying to buy, and maybe you'll close the sale of it better. Everybody wants to close the sales better, of course.

And then I can maybe go here, get product information, talk about the efficiency and all those types of things. So, really, a much more delightful application and delightful experience for the customer. But how hard was it to build?

Let's hop over here to Visual Studio and look at some of the code I had to write to enhance that new experience.

Now, before I get started, I want to point out that the mapping control and the ink canvas control that I use to show the map and the drawing, all of that was free. I just used the stock controls and all of that rich interactive experience of drawing was done. Do you guys like free code? Of course we all like free code. (Applause.) No, anything I don't have to write.

But, unfortunately, we do have to write some of this code here to call the ink recognizer service. But I'll show you, it's not a lot of code and it's pretty easy to write.

So, what I'm doing here when I press the "analyze" button is I pass in that ink canvas that I use. From there, I'll just grab the strokes, one call. I need to package it up into a JSON format, so I'm going to make a REST API call, and then here I make the call, very simple call here, and what I get back is the recognized two rectangles, and then I draw them and I'm done. A few lines of code, and I get that great, immersive experience.

Now, Microsoft Office also uses this same ink recognize Cognitive Service -- this means it works at scale and it also works cross-platform. And I'm happy to announce that our first partner to utilize the ink recognizer Cognitive Service is Wacom, a leader in digital pen. (Applause.)

They're updating their popular ink-based app with some new features such as semantic understanding of ink content.

Now, back at the main office, the office staff communicated with these field agents when they're ordering the parts. And they prefer to look at the same description page to avoid mistakes. This flow, workflow is mission critical, and we need to update the app in an incremental and non-intrusive way.

Now, sometimes I hear these apps referred to as legacy, but the reality is it's active code that we just need to modernize responsibly. Let me show you how you can do this with XAML islands for any existing Windows app, whether it's WPF, WinForms, or native Win32.

So, the page I'm talking about is that last page I was on here. I want to show the part because, many times, the part's out of stock or maybe they want a replacement and they want to make sure they're ordering it correctly. So, this is a page we wanted to reuse inside of our Windows app.

So, I've just launched a WPF app here, and again, you'll see the product page here that we're going to share. I've already done the work to host it in the XAML island.

Now, the one thing to note is in this app, the text is much more dense because it's built for mouse and a could be, for productivity workers looking at many products, and to make it look more in line with the dense UI, I'm just turning on one attribute in the XAML for the WPF app, and then it will automatically keep the same layout but give me that condensed look. Much more like you'd expect inside of this application.

Let me show you now the code we used -- the code we had to write in order to reuse this UI. Now, before I show you the code inside the WPF app, we did have to do a little bit of refactoring. For any control that I want to share between a UWP app and a WPF app, I need to put in a separate library that links into both, pretty straightforward.

So, I've created this controls library for all those shared controls, and the one I'm using is data entry view.

So, I did that factoring from the UWP app into the controls library. Then I go over here to my XAML and all I need to do is add this new XAML host, Windows XAML host, name of the control, point it at the control that I want to load that's a UWP control, and we automatically wire two together and done. One line of XAML, I'm able to reuse that entire UI. And I'm happy to announce that XAML islands 1.0 will be available in the Windows 10 May update. (Cheers, applause.)

I'm going to move on and now talk about a few more capabilities that I want to add to the application. Now, another team created this simple cross-platform phone app, which helps the field agents manage their schedules. The field agents have asked me to update the Surface Go app so they can access their schedule without needing to pull out their phone.

Because the phone team's skill set was based in JavaScript, they decided to use React Native. In a nutshell, React Native is a popular solution for mobile developers that prefer a web-like development environment to create native, cross-platform apps. In the same way that Xamarin and Xamarin Forms is a popular solution for developers that prefer C#, and it could easily have been built in either. I'm excited to announce that we teamed up with the Facebook and React Native community to deliver a new, high-performance version of React Native for Windows. (Cheers, applause.)

Partnering with us are Citrix, Plex, AXI and EPAM. Let me show you how easy it is to integrate this phone UI into our UWP app using this new React Native implementation.

So, I'm going to go over here -- actually, let me show you the UI for it first. Let me show you exactly what I wanted to integrate in. So, here I have my calendar tab and I pop over here and on the right-hand side here is actually XAML that I had, but I wanted to reuse this UI. I made it green specifically so you could see it, but the reality is I probably would have kept it white, just like the phone app. But the main reason is it is literally fluidly integrated in because the controls rendering it are exactly the same controls that I would have used from XAML. I just instantiated them from JavaScript versus from C#, but the UI is fluid and smooth and exactly what I would expect.

So, how hard was it? Let's go look at the code. So, here, I've done something similar that I did with XAML islands where I had control that then points to the UI that I wanted to reuse. In this case, it's called a React Control, and I do all the wiring of it in this loaded event. So, there it is, I laid it out, I sized it, made sure it fit in the right place.

And then over here, when I want to wire it up, what I ended up doing in the load event is I set up a few parameters if I want live reload, it lets me reload it dynamically, or I can set up debugging.

But the real key here are two attributes: the bundle name and the component name. The bundle name is a way of bundling a set of React Native controls or components, and then I can reuse those components in different aspects of the UI.

So, let me hop over and show you what we need to do in the React Native product itself. Well, the reality is, nothing. It was exactly the same controls, exactly the same code. All I need to do is pull out down here the component name that I called Day View, and I reused the file name as the bundle name. So, there you go, wired it all together, simple use, and get all that reuse of that React Native code.

I've highlighted just a few of the key improvements coming to Windows for developers, many more coming as well.

But next, I want to talk to you about Microsoft Edge and the web. Fundamentally, the web is about consistency and reach. So, the next version of Microsoft Edge will work across all versions of Windows down to Windows 7. It will update frequently, and of course, we will bring it cross-platform to Mac, Android and iOS.

Now, enterprises have unique needs beyond just the fundamentals of the web. Let's take a look at a common scenario.

So, here I have a website and this is a standards-based website running in the next version of Microsoft Edge, it runs great. Now, I have some sites that use some ActiveX controls, and they're running in an IE-11 compatibility mode.

Now, typically, when I want to go view them, and here in this case I'm going to view an invoice, if I click this button, it would have popped it up into Internet Explorer in a separate window, kind of breaking my workflow and making it hard for me to manage my windows.

But as you saw here, it popped up right in another tab, allowing me to group them using tabbing in exactly the way you would expect it to work.

Now, we have another experience that we have which is our chat. And the chat, let me show you right here. I'll click. When I click on it, it pops up in a tab. I don't really want it in tab, I want to be able to chat going over multiple sets of different tabs and use it kind of as a top-level window. So, we've added some features here to make it a PWA. Now, we've also optimized in Edge detection of the PWA and making it an inline experience. So, right here, we'll get a notification saying this is a PWA and it's installable. So, I can click that, install it right here in my flow, pops up, the top-level window, no address bar, no tabs, exactly the way I would want to be able to use chat, and I can use my tabs on the left-hand side, really building an optimized, streamlined workspace where myself and every employee or every customer can go and do that for themselves.

So, now that I've shown you some of the ways that Edge has made the front-end a better experience, I want to move and talk to you about some of the improvements we have made for doing back-end development for the web.

Now, to do that, I'm going to hop over here and launch Bash. Wow. You know, Bash, it's new, it's cool, but that console experience, looking a little vintage. You know? Been around for I think 30 years. I think we can do better. What about you? Can we do better? (Cheers, applause.)

(Video: The New Windows Terminal.)

(Cheers, applause.)

**KEVIN GALLO:** Welcome to the new Windows Terminal. Let me walk you around some of the UI features that we've added here and all the great enhancements.

So, first off, tabs. We give you a rich set of tabs built with our Win UI library that allows you to pull them off, create multiple sets of windows. We've also added the ability to host multiple shells and themed them however you would choose.

So, for example, here, I have PowerShell and I chose a nice fluent blue to it. Here's CMT in a nice fluent black. And, of course, here Ubuntu that I can have with a nice purple. You can choose your themes, group them by tabs, however you choose, completely customizable.

We've also added this new shortcut feature so you can get to your shell of choice, whatever you use the most, quickly and easily. Here, I can go quickly launch Ubuntu, create another tab, I can delete it, and easily manage my workspace.

We're also going to add a capability for extensions, allowing developers to truly go and customize the full experience. Much like we do in VS Code or web extensions, allowing a marketplace for developers to share with each other and really build great experiences.

And last but not least, we've added full Unicode support. This means that east Asian fonts render correctly, it also means that we get features like ligatures, and of course fun things like emojis.

And to really kind of celebrate that, I built this file here that I'll show you. There we go. (Cheers, applause.) Windows loves developers.

Now, this is my expression of creativity, I hope you'll express your creativity probably maybe a little bit better than I do. And one of the ways you can do that is by joining us on GitHub because Terminal is going to be built in the open for all of us to give feedback and contribute and help us build those features together. So, together, we can build the best terminal ever and it's live right now. (Cheers, applause.) Don't go look at it yet, wait until after this is over.

Now that I've shown you some of the UI enhancements we've made to Terminal, let's move and talk about what we're doing for WSL, or the Windows Subsystem for Linux.

We announced that two years ago back here at Build, and since then, over 3.5 million developers use it on a regular basis. And we're excited today to announce the next version we call WSL2.

What we're doing with WSL2 is dramatically improving performance, specifically around file load, so if you're doing a compile or a git clone, git repo clone, you are getting at least 3X performance improvements for those workloads.

And of course probably one of the most important things is compatibility. We, in order to improve compatibility across the board, we have now introduced a Linux kernel directly in Windows. So, all of the distributions will run against that Linux kernel, give us this unprecedented compatibility across all of them.

Now, one of the workloads and we've had a lot of feedback that every developer wants to start using WSL mainly so they can be as close to production as possible is Docker. And I'm excited to announce that we now have native Docker support with no need for a VN. (Cheers, applause.) Pretty cool.

What this means is faster startup times, better memory usage, and it truly can be as close to production as possible when testing your environment. Let me show you what I mean. I'm going to launch VS Code here. And what this is doing is I'm running it from the Ubuntu distribution right here with all the installation and extensions that I've installed in this exact shell.

And then it's going to pop up the UI, and the UI is actually running in Windows in NT just like you'd normally expect. But the way you know it's kind of running against that instance is it's using a feature we call VS Code Remote, and down here in the corner, it indicates that it's debugging WSL directly from in this distribution.

So what this means is if I want to be a Debian or SUSE, I can launch from within one of those, and I'll be debugging from inside that environment versus inside of the Windows environment.

Now, this also gives me the ability to kind of fully do an end-to-end web development stack debugging and workflow. So I'm going to go and organize this the way that I like. I like to put VS Code on the right, I'm going to launch my web page here and go and show you it here in my debugging. This is launching against -- I didn't hit F5, yeah, this always happens. Hit F5, there we go. Kind of proves to you it's not running against it in the web, it's running directly against this instance here.

So, when I click here, it pops it up, shows it on the right-hand side, go here, I'll go and I'll move this to my left and then I can do F12, and I'll be able to show the new Edge development tools.

So, here I have running as close to production as possible my debugger, my front end, and even my front-end debugging having a great end-to-end full stack development experience. (Applause.)

So, let me recap both Windows and the web. I've shown you how easy it is on Windows to incrementally adopt new features like Microsoft Graph, Cognitive Services, XAML islands and React Native for Windows. For the web, with the new Microsoft Edge and the hot new terminal and WSL2 with Native Docker support, you have everything you need.

Thank you, back to you, Rajesh. (Cheers, applause.)

**RAJESH JHA:** Thank you, Kevin. So, let's move on to chapter five and talk about Microsoft Teams.

As you know, businesses are transforming and decision-making just seems to keep getting faster and faster. Teams are distributed, they use disjointed tools, and so there is this need to build a tool that's collaboration first, and that's what Microsoft Teams is. Built ground up with collaboration in mind.

And we are gratified by the momentum we are seeing on Microsoft Teams. It is the fastest-growing application in Microsoft history -- 181 different markets, over half a million businesses, more than nine out of 10 Fortune 100s are using Microsoft Teams today.

And Microsoft Teams is more than just chat-based collaboration. It is a hub for teamwork. It's got meetings built in, it's got telephony built in, it has Office built in, Planner, GitHub, these are all integrated. Hundreds of developer and partner apps are integrated inside of Microsoft Teams today.

Now, Microsoft Teams, you'll see in a bit, has got tremendous extensibility that allows developers to bring in business processes, line-of-business applications, workflows, right to where the users are already collaborating. And then I'm also very excited about how Microsoft Teams has been extended to not just bring information workers together, but first-line workers, folks who work in retail stores, in hospitals like nurses and doctors, flight attendants. First-line workers, information workers collaborating amongst themselves, between each other, because Microsoft Teams can be customized in this experience by role and by industry.

And to show you how as a developer you can build onto Microsoft Teams and customize it for your business processes, and as a one-stop hub for teamwork, I'm going to invite up Klorida, thank you.

(Cheers, applause.)

**KLORIDA MIRAJ:** Thank you, Rajesh. Good morning, everyone. Today, I'm going to show you how Microsoft Teams is helping airlines improve boarding time. But most importantly, making customers happy by getting them to their destination on time and stress free.

Many of you here flew to Seattle here to be with us this morning. And as you can imagine, there's a ton of collaboration and communication that happens in the background to get a flight departed without delays.

Just for this demo, I will be your chief flight attendant. And we are heading to LaGuardia. Fasten your seat belts! I'm going to use my Microsoft Teams mobile application and I'm going to show you how I communicate and collaborate and complete tasks more effectively with my team.

Using the Graph APIs, you'll see that my experience has been personalized I have a list of teams that has been automatically created per my flight schedule, that's our flight, 305, right there. And there's a list of channels here that are set up per job function.

In my channel, flight crew, the line-of-business application, Fly Bot, has already set us a card with the flight details. This Fly Bot is built on the same Microsoft Bot Framework that both Rajesh and Satya mentioned earlier. It leverages the Adaptive Card, and it also connects to the airline's back-end services.

In addition to the flight's information, I also have access to a set of channel tabs. You see here the flight plan, my flight roster, this is an Excel tab, there's a passenger experience checklist, which is built by one of our partners, Trello, and there's a lot of other information that I'm going to need for my flight.

These tabs are a great way to bring all the relevant content into a single place. They can be driven by an Office experience such as Excel, Word, Planners, a SharePoint web part, a line-of-business application, a PowerApps, or a partner web application. Everything that I need to do my job is at my fingertips. Isn't that awesome? (Cheers, applause.) Thank you.

So, in addition to having this information accessible, I also need up-to-date flight status. So, let's look at this notification. Fly Bot informs everyone in the channel that the incoming aircraft is delayed. We have less than 38 minutes to get this flight ready.

Now, even less time with this delay, so this is going to be a tight timeframe. Let me send a note to my team. We'll have to get the cabin done faster.

Being able to get these updates in real time and communicate decisions to my crew in the same context is how Teams enables us to take actions instantly and improve outcomes. It's now time for boarding, so Fly Bot is sending us updates on the boarding status. You see here I can see details about how many passengers have boarded and any other special accommodations.

I have quick access to key actions that I'm going to need later to prep the flight.

Now, one critical area that the flight attendants need to keep track of is overhead bin capacity. As soon as space starts to run out, passengers waiting to board will need to check bags at the gate. Now, today, I would have to check how much bin space is left, run to the back of the plane where the phone is, and call the gate crew to let them know. I would have to do this multiple times.

Now, let me show you how I can do this easily in Teams. I tap on bins, action, enter how many remaining bags I have, say six, six sounds good. I check all luggage at the gate, hit submit, that's all I have to do. Fly Bot does the rest. The gate crew is notified.

With Teams, I was able to do it in a transparent and effective way. Now, another common reason for delays is managing two people booked for the same seat. I'm sure you've noticed this.

Today, this involves some type of seat shuffling coordination between the onboard flight attendants and the gate crew. Now, let me show you how I can do this with Teams.

One of the key actions from the card is that I can reach out to the supervisor to ask for a quick clarification. With the push of a button, I can start a one-on-one chat. Say it's double booked. So, the supervisor sends me -- oh, the supervisor needs more information. So, what I can do is take a quick photo. I have a boarding pass right here. Let's see, we are heading to LaGuardia, and send.

Now, the supervisor is in the background doing her magic, and she already gives me a new seat assignment. This is awesome, further delays are all avoided and it's so cool, right? (Applause.) Thank you.

So, a key part of Teams is fostering an open and inclusive culture. I want to send kudos to my supervisor for quickly resolving the issue. And I can do that right in here, in this experience.

I will message extensions, which are another application capability that allows me to share rich cards with my team members. These can be powered by a line-of-business application or a partner application.

Let me show you how one of our partners, Disco, has built this experience. I choose Disco, choose a leadership value, and voila, that's it. Pretty cool and easy and simple, right?

I love how Teams allowed me to use all the applications and services that I need to get my work done, while at the same time I stayed in touch with my team.

I just showed you the mobile experience. But as you know, Teams works across platforms, on the web, iOS, Android, Windows.

I'm going to head over to my laptop and show you the Windows experience. You see here, I have the same setup for Teams and channels as I had on my mobile. IT admins can also now customize Teams for their end users by pinning partner and line-of-business applications such a Polly, ADP, Lucidchart, right here in my app bar.

This is a new feature. I am happy to announce new application policies that are available in the product today. (Applause.)

My IT admin has pinned a line-of-business application right here called FlyApp. Let's check it out.

From here, I can look at my flights, my upcoming and my past trips. I can look at my layover services. There are some training videos that I can look at, and also binge on company news and any announcements.

This experience is based on open standards so it can be adapted to meet any needs and then be brought into Teams very easily. Let me tell you, though, that airline customers are not the only ones leveraging Teams to deliver results faster and bring people together. We are working with customers across industries and job functions to bring a similar digital transformation using teams.

In manufacturing, tracking projects and stats and status delays. In banking, helping with training and education. In health care, transforming the coordination of customer care. In education, helping faculty, staff and students achieve more by working more closely together.

Dr. David Kellerman, an engineering faculty member at the University of New South Wales in Australia, observed that within three weeks of introducing Teams, class engagement was up 900 percent from the previous term.

Now, you may all be thinking, how do I build one of these, right? Let me show you how easily I can publish a new team right within the Teams experience.

We have a built-in Teams app called App Studio right here, which steps me through this process. I can tap on create a new team, I can fill in the app details, and also any of the capabilities that I need such as tabs or bots, connectors, message extensions. And these can all be built using open web standards and technologies that you're all familiar with.

There is a built-in card editor which I can use to customize my experiences. And then once my app is ready, I can go and site load it using the uploaded custom apps and test my application with Teams. We are streamlining the end-to-end application life cycle. Starting in June, you will be able to submit applications via Partner Center and create your offers for your applications. OK?

These can be purchased by partners on behalf of the customers or directly through app stores.

Then, the IT admin can deploy them using policies that I just showed you and they will appear as thin applications to our end users. It's great.

Another key feedback that we heard from our customers is the need to have visibility into the application security, data handling and compliance practices. So, I'm happy to announce the Microsoft Teams Application Certification Program. Today, we are launching a pilot, our very first step towards this direction.

The program is going to streamline the process of gathering and reviewing application information. We are already seeing great early feedback from our launch partners and customers on this program.

Everything that I showed you is available today. Thank you. (Applause.)

I want to tell you that applications in Teams are as core as applications are in Windows. I invite you all to bring your apps to Microsoft Teams so we can together amplify rapid decision-making and drive cultural change. Thank you. (Applause.)

(Video: Teams.)

(Applause.)

**RAJESH JHA:** Thank you, Klorida. So, we talked about Identity, Graph, building on Windows, web, and Teams. We're in the home stretch now. Let's talk about building on Office and SharePoint.

These apps, they reach hundreds of millions of users. SharePoint is the home in Microsoft 365 for documents and content. And it's now performant and responsive with the new web technologies based on SharePoint Framework.

As a developer, you can easily extend SharePoint now using these web technologies.

We're also excited that in SharePoint you can bring together PowerApps and Flow and use business processes, use PowerApps and Flow, use SharePoint document libraries and lists, the rich capabilities there, and bring these business processes right to where people are, because SharePoint can now also be hosted inside of Microsoft Teams.

Office has always been extensible. Now, with the web-based extensibility in Office, your extensions will get to run on Windows, on Mac, on the web. In addition to extending the UI in Office, today we'll also talk about how you can extend the logic in Office using JavaScript.

To show you how as a developer you can build on the capabilities in Office and SharePoint, I want to invite up Wey. Welcome. (Applause.)

**WEY LOVE:** Thanks, Rajesh. Hello, everyone. I'm so excited to be here today to talk to you all about Office and SharePoint.

A key part of SharePoint are its pages, the face of intranets, department sites and team sites everywhere.

We've been on a journey to help you create beautiful, engaging sites using a wide array of controls and components. Here, you can see web parts that link to Yammer, Stream and Microsoft Search-based feeds.

This site has been customized with company logos and colors, and it's responsive to any viewport. So it works great on a mobile device and in the SharePoint mobile app.

Before, a site like this might have taken weeks or even months to develop, but now using out-of-the-box web parts, it can be created in just hours. We know that your pages are personal to you and your team. Everyone's needs are different, and so sometimes you'll need to deeply customize your experience. In fact, for the rest of my time, I'm going to show you how this is done at a retail store.

With SharePoint Framework, I was able to extend these pages. Here, I built a web part that connects to our supply chain and delivery system and integrated it into our page.

Let me show you the code behind this. SharePoint Framework provides a tool chain that makes it easy to take my script, debut it locally using integrated tools, and then package and deploy to SharePoint. Now, this web part was written using the React Framework, but you can develop using a number of frameworks like Angular, Vue.js, Knockout and more.

I can even use React templates to build my web part. So that means all the power of React is available to customize your sites.

We've also recently announced you can host your SharePoint Framework customization in Microsoft Teams as a tab. And here, you can see that the same web part I showed earlier is now hosted in a table for my sales team.

I didn't have to write any additional code to make this work, which is great. Now, SharePoint also hosts document content, which is critical to how organizations run. This content is stored in lists that are powerful and flexible. At our store, we're preparing for the change of season, so it's time to update all of our retail store displays, which means we need to get rid of those teas and bring in those surf boards, bikes and puffy vests.

Now, getting our displays right is critical. So, we're using a list to track the work of updating these displays in time. Notice we have custom columns showing whether a display has the required elements. So, at a glance, I can see if any of the major items are missing from the store. These new list visuals can be created with simple script customizations, and I can reorder and rearrange them to fluidly work with the data.

Now, I want this season's launch to be as easy as possible, so I'm going to build a process using Microsoft Flow which is deeply integrated into Microsoft 365. I want to be able to ask my team to take pictures of their store displays and upload them to SharePoint using the SharePoint Mobile app. When that happens, I'll want a workflow to kick off.

Here, you can see that I've created a flow that as the pictures are uploaded, they'll be routed to Azure Cognitive Services, where I've trained a model to look for the required elements like bikes.

Flow will then take those results and update my status, which I think is really awesome.

SharePoint lists serve as the hub for documents and content processes and coupled with the rich set of Graph APIs and great integrations with Flow and PowerApps, you can use them as ingredients in your Microsoft 365 applications.

I use lists a lot with my team, and they're a big time saver for me. Now, preparing for our summer launch also requires me to track some budgeting. So, naturally, I'm going to do that in Excel.

Because Excel supports web-based add-ins, our store can easily integrate Excel calculations with our Oracle Enterprise Performance Management Cloud, utilizing Oracle's Smart View add-in, which you can see here in the ribbon tab.

Now, this add-in allows me to directly access and analyze data, all while using familiar tools. But there's a new addition. We can integrate this Oracle data directly in formulas throughout our workbook. And as you can see right here, we are calculating revenue and it looks like the Napa store has brought in the most this year.

Now, this specific function is special because it is an example of an Excel JavaScript custom function. Excel Custom Functions let developers create their own functions that extend the soul of Excel, allowing organizations to calculate or retrieve data right in their workbook.

I'm pleased to announce Excel Custom Functions are generally available as of right now. (Cheers, applause.)

My organization has built Custom Functions as well. And so I'm going to switch over to this store feedback tab, which is showing you a table of all of the customer input we've received over the past month, along with the relevant purchase information.

Now, I want to see if there's any correlation between the purchase information and the feedback that we're seeing. So, in order to do that, I'm going to use our customer happiness function, which calls an ML-powered service to perform sentiment analysis.

So, looking at this briefly, I can see here that the Napa pop-up store has all positive interactions, which may be why they have the highest revenue for the year. I wonder what they're doing differently.

Anyway, to me, what's great about this custom function is that there's no need to mess around with separate installs. This solution can be auto deployed to my organization by IT. And once deployed, anyone on the team can use these custom functions alongside the rich tools of Excel to analyze their results.

Now, before I wrap up, I want to jump into VS Code to show you how easy it was to do this.

Here, you can see that we've made our request to Azure Cognitive Services. And then we take the data that we receive and put that into Excel. Now, today, I've had the opportunity to show you how our retail store was able to make all of our tools more engaging with SharePoint, to help streamline our workflows via lists and flow, and provide deep analysis to connect the data in Excel.

Like the rest of Microsoft 365, you can use web technologies to extend and build these experiences seamlessly and host them wherever work is getting done. Thank you.

(Applause.)

**RAJESH JHA:** I'm grateful for the time that you spent with us today. I hope you share our aspiration, our excitement around building people-centered experiences.

I'm excited to see what you can do with the Microsoft 365 platform in service of our customers. Let's close with taking a look at what developers are already doing with the Microsoft 365 platform. Thank you.

(Applause.)

(Video: Developer Testimonial.)

END