Title Page
Hi, today we're going to create our first Bluemix/Watson app. Let's get started.
Page 2
We're going to go through 8 steps in the next 20 minutes.
(1) We will complete your workstation set-up by installing nodeJS
(2) We'll get the code from the master repository
(3) We'll confirm that the Watson speech to text service is connected to your Bluemix instance and get the necessary credentials
(4) And then we'll make everything run, first on your workstation, then on Bluemix
Pages 3-5
The next three pages in this presentation tell you how to install nodejs on your workstation. There is a page for Windows, Linux and Mac OSX. If you haven't already installed nodeJS, pause the video and follow the directions to install nodejs. If you're not sure if nodejs is installed, open a Terminal or Command Prompt and type in node -v. You should get a response back like v6.4.0. If your computer doesn't recognize the node command, follow the directions on pages 3-5
Page 6-7 - go to Github.
OK, you have nodejs installed, let's get the code. We're going to get it from github and use the github fork process. The reasons for this are explained in the deck.
Go to this URL: https://github.com/rddill-IBM/ZeroToCognitive and click on the fork button in the top right hand corner your browser window. Tell GitHub where to create your version of this repository and click on 'go'.
Your browser will automatically go to the newly created repository. Click Clone and the click on Use Github Desktop.
Success!! You now have the tutorial loaded to your workstation.
Page 8 - even though nodejs is installed, your app won't yet run. Open a terminal window and navigate to your local copy of your repository. type in cd ~/Documents/GitHub/ZeroToCognitive/Chapter03/ and press enter. Type npm install and press enter. npm is the node package manager and it will read your package.json file to figure out what node modules are needed to make your application run.
install the watson sdk: npm install watson-developer-cloud --save and press enter. IF it's not already there, the node package manager will install the watson sdk. --save means to also put an entry into your package.json file.
Let's see if it works: type node index.js. Success!!! Node responds by telling you that z2c_chapter03 is running on port 6003. Go to your browser and give it this url: localhost:6003. You should see the same web page on your workstation as you see in this tutorial.
Page 9
Let's do some light programming. The background isn't all that great, so let's change it to something a little easier to work with. Most visual characteristics of a web page are managed through CSS, Cascading Style Sheets. First, we have to know what to change. We'll start with the index.html file in the HTML folder. Near the top of that file, you'll see an HTML tag for body, which I've highlighted here. There are two important parts of that definition. The first is the class=tutorial tag and the second is the onload=initPage() command. CSS looks for class tags.

Go to your HTML/CSS folder and look at row 6. .tutorial tells CSS to look for any occurrence of class=tutorial in your html files and change the background and foreground to the colors specified on line 6. Change background from 040404 to D0D0D0 and then save the file.
Go to your browser and click on the reload button while holding down the shift key. Your browser page should look like what you see in the tutorial.
Congrats! You're now creating your own version of the app.

Page 10

The other part of the body tag that helps us is the onLoad statement. Go to HTML/js and open the z2c-speech.js file. In it, you'll see an empty routine called initpage. This routine has two jobs: (1) enable and disable the microphone and stop icons based on what the user is doing and (2) connect to the Watson Speech to Text service and use it to turn spoken words into digital text.

Pages 11-12

All of the code to do this is in the Documentation/answers folder.
Open the z2c-speech_complete.js file in the Documentation/answers folder. After we've reviewed this file, we're going replace the contents of the z2c-speech.js file with this one.
Rows 7-9 add a class to the microphone and stop icons. This tells CSS to change their visual behavior and is something we can test for in javascript to know if we should respond to a click on that icon.
Row 11 tells javascript that it needs to do something every time the microphone icon is clicked.
Row 14 checks to see if the microphone is enabled. If not, exit immediately.
Rows 16-19 toggles the states of the microphone and stop icons
Row 20 calls the server to get the speech to text token ($.get) and ($.when) when that call returns, uses the returned token to
Row 22 activate the microphone and
Row 24 put the converted text into an HTML location called speech
Having turned on the microphone, it would be really nice to be able to turn it back off. Line 31 starts another on click routine, this time for stop - the stop icon on your display.
Row 33 turns off the stream if it's on
Rows 34-38 toggle the icon states.
Copy this complete routine into your z2c-speech.js file, replacing the current empty routine and save it.
OK, that's the browser piece. Now let's take care of the server connection - we have to add the code to support the $.get call in our javascript file.

Page 13

The first thing you'll need are your credentials for the Watson Speech to Text service from Bluemix.
Log in to Bluemix and go to your Zero To Cognitive space, then click on the app icon.
When the Speech to Text card displays, click on "Show Credentials" and copy this information
On your laptop, go to the Chapter03/controllers folder and open the env.json file and paste the information you just copied into this file.
Highlight and copy the user name and password values into the placeholders in the original file, then delete the rows you copied from Bluemix. Save this file.

Page 14

Open index.js and look at row 34, which tells nodejs to load and use a file called router.js, located in the controller/restapi folder. Let's open that file.
Row 5 exports 'router' and row 7 defines the path /api/speech-to-text/token which is called by the browser javascript file.
Row 3, another require statement, tells us that the code to support the router call is located in the features folder and is called 'speech_to_text.js' Let's open that file.
The essential part of this file is the (missing) code function supporting the token export on line 6. Just like we did with the browser code, we'll copy over code from the answers folder to fill in this function. Let's open the answer and look at it.

Page 15

Line 8 defines a variable which takes our credentials and extends them with a function from vcap services.
Line 10 defines a variable which stores the result of the Watson authorization method.
Lin 12 uses the authorization result to get a token which will then be used in the browser.
Lines 14-20 handle error reporting and, if there are no errors, sends the Watson speech to text token back to the browser.
Copy lines 7-22 from this file into the speech_to_text;js file in the features folder, replacing lines 5 through 8 and save this file.

Page 17

Let's test it on your laptop.
Go to your terminal window and, if node is running, press CTRL-c to cancel it.
If you're not already in the Chapter03 folder, change to it and then restart node by executing this command: node index.js.
Node should respond by saying that z2c_chapter03 is running on port 6003
Open your browser (I am doing all of this using FireFox) and type in localhost:6003 and then press enter.
You'll see your app come up like this. Click on the microphone to start Watson speech to text and tell FireFox to allow use of your microphone. If you have more than one option, select the option that matches the microphone you're going to use.
Start talking. You will, after a few seconds, begin to see your words appear in the speech window on your browser.

Page 18

Saving your work:
Stop node (CTRL-C). Type git status. Using git add, add each of the changed files to git. Type git status when you're done to ensure you didn't forget anything.
Type git commit –m 'WooHoo!! It works!!'
Type git push.
Congratulations! You've now updated your personal repository with the latest version of your code.

Page 19

Running this on Bluemix
Go to Bluemix and look carefully at the name for your speech to text service. It will look something like this: Speech to Text-aa The 'aa' part will be different.
Open the manifest.yml file and add this line after services:

- Speech to Text-aa

Replacing aa with whatever letters appear on your version of Bluemix and then save the file

- Open a Terminal Window and navigate to Chapter03 ~/Documents/GitHub/ZeroToCognitive/Chapter03/)

Delete the node-modules folder.
Type cf login and enter your IBM intranet user id and password.
Select the organization and space linked to your Zero-To-Cognitive work.
Go to your terminal window and type in cf push and press enter.
In 3-5 minutes, your app will be up and running on bluemix.
Go to your Bluemix dashboard, go to your app and click on it. You'll see a URL highlighted at the top of the page. Click on it and it will load your app!!! Congratulations. You have now successfully built and deployed a Watson app on Bluemix!!!!

Pages 20-21

In our next lesson, we will have Watson talk back to you.