TJBOT - An Open Source Cardboard Robot

Haleema Khatoon[1], Prof. Praveen Kumar Pandey[2]

[1]Student, MCA,
[1,2]Jain University, Bengaluru, Karnataka, India

ABSTRACT
TJBOT is an interactive Open Source robot which is designed to encourage All individuals to build TJBOT in fun way with cognitive services. Where TJBOT can be 3D printed and which comes with set of recipes from which we can build TJBOT. We can program the TJBOT to speak, listen, Shine Led, see, recognize, Understand the emotions, wave arms, play games: Play Rock Paper Scissors in Node-RED. Basically TJBOT was designed for two communities: makers, who enjoy the DIY aspects of building and programming novel devices, and students, who can learn about programming cognitive systems.

KEYWORDS: *Cognitive services; makers; open source; DIY; emotions; mood; robots; Internet of Things*

How to cite this paper: Haleema Khatoon | Prof. Praveen Kumar Pandey "TJBOT - An Open Source Cardboard Robot" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-4 | Issue-3, April 2020, pp.862-869, URL: www.ijtsrd.com/papers/ijtsrd30701.pdf

IJTSRD30701

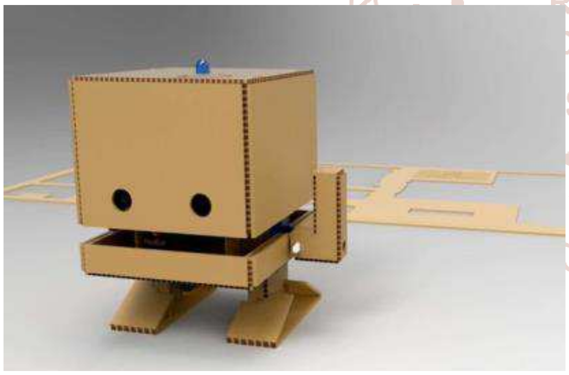


FIGURE 1: Outlook of TJBOT

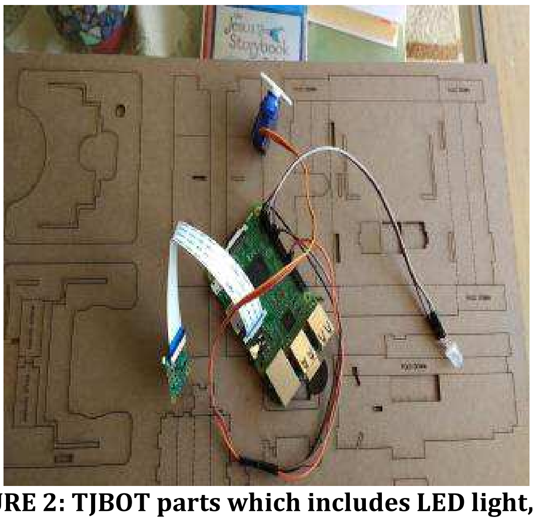


FIGURE 2: TJBOT parts which includes LED light, USB Microphone, Camera, Servo Motor

INTRODUCTION

Cognitive applications enable people to collaborate with AI technologies in order to discover information, analyze, make decisions, and solve problems.

Due to a number of recent breakthroughs in artificial intelligence, machine learning, computer vision, natural language processing, and conversational systems, The ability to create cognitive applications is becoming easier via the availability of cognitive services and APIs At IBM Research, IBM Research have been exploring applications of cognitive services with everyday objects, a.k.a cognitive objects ognitive objects are capable of sensing the environment, performing computation, and providing feedback to a person. As part of exploration, IBM have developed a new kind of object a paper robot that embodies a number of cognitive services, including visual recognition, conversation, sentiment analysis, speech to text, and text to speech. IBM designed TJBOT as an open source platform to enable people to learn and make interesting things with the IBM Watson cognitive services. In a sense,

TJBOT represents a restriction of a broad design space of cognitive objects by providing a limited set of concrete functionalities. The robot can shine its LED, wave its arm, speak (or play sounds) via its speaker, listen via its microphone, and see via its camera.

TJBOT is open source, including both the robot housing and software libraries.

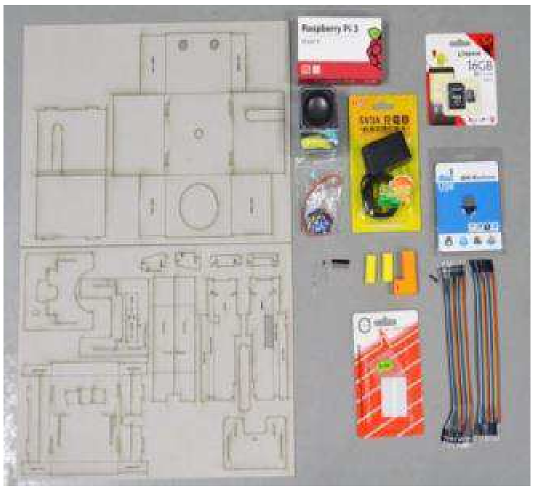


FIGURE 3: Cardboard TJBOT With its parts.

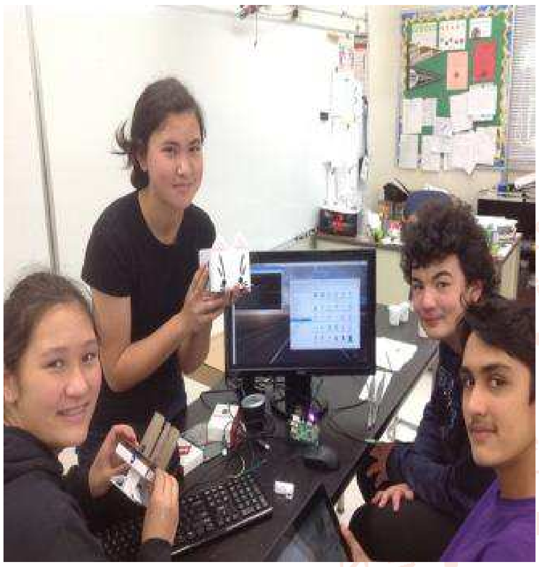


FIGURE 4: TJBOT Built by students.

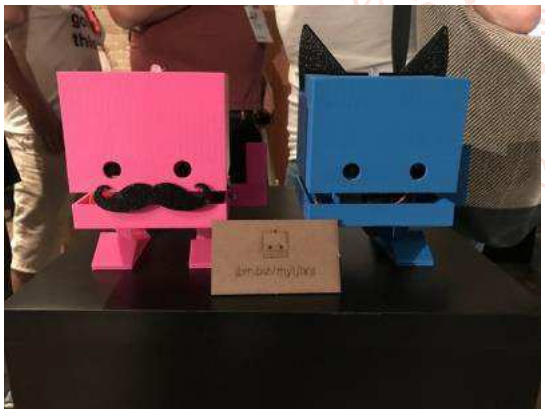
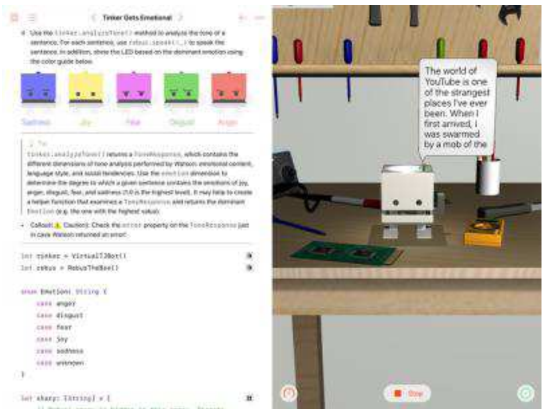


FIGURE 5(A)


Tinker Gets Emotional

Use the tinker.analyzeTone() method to analyze the tone of a sentence. For each sentence, use robus.speak(...) to speak the sentence. In addition, show the LED based on the dominant emotion using the color guide below.

tinker.analyzeTone() returns a ToneResponse, which contains the different dimensions of tone analysis performed by Watson: emotional content, language style, and social tendencies. Use the emotion dimension to determine the degree to which a given sentence contains the emotions of joy, anger, disgust, fear, and sadness (1.0 is the highest level). It may help to create a helper function that examines a ToneResponse and returns the dominant Emotion (e.g. the one with the highest value).

Callout(⚠ Caution): Check the error property on the ToneResponse just in case Watson returned an error.

```
let tinker = VirtualTJBot()
let robus = RobusTheBot()

enum Emotion: String {
    case angry
    case disgust
    case fear
    case joy
    case sadness
    case unknown
}

let sharp: [String] = [
```

The world of YouTube is one of the strangest places I've ever been. When I first arrived, I was swarmed by a mob of the

FIGURE 5 (B) FIGURE 5: Exploring AI By students in the Classroom.

OBJECTIVE OF THE PROJECT

Our goals with TJBOT are to understand

A. what kinds of new behaviors people create,

B. what the best practices are when designing interactions with the robot,

C. what kinds of capabilities people assume the robot has, and

D. whether restricting the design space motivates people to learn about cognitive services and build new things with them.

E. Create an extensible framework so TJBOT can be taught new tricks simply by orchestrating new action handlers

F. Make actions technology agnostic

G. Exercise IBM technologies: Bluemix, Watson/Watson IOT.

TJBOT HARDWARE

- TJBOT has a Raspberry Pi 3 inside. It is a small Linux computer that we can use to connect to Watson API and control TJBOT's hardware.
- The robot frame has placeholders for a Raspberry Pi camera behind the left eye, a speaker and a microphone inside the bot to enable a conversation, a servo motor to move the right arm, a RGB LED to shine and reflect emotions.
- Inside the head is left out empty to accommodate additional sensors. For example, some of the makers used TJBOT as a personal weather station..

TJBOT SOFTWARE

- In order to make TJBOT easy to program, they have created a software library (e.g. an API) that encapsulates the robot's unique capabilities and connection to cognitive services.
- The library is written for Node.js, an open-source, cross-platform runtime environment for developing applications in JavaScript.
- To get people up-and-running with TJBOT without having to write code, as well as teach people how to program TJBOT themselves, we created a set of sample instructions (called recipes) that demonstrate different capabilities.
- For example, one recipe makes TJBOT introduce himself using Text to Speech (the robot simply speaks, "Hi, I'm TJBOT").
- Another recipe allows a person to have a simple conversation with TJBOT, asking questions such as "What is your name" or "Tell me a joke."
- TJBOT can also monitor Twitter for tweets, and a third recipe monitors for tweets containing a specified keyword, performs a sentiment analysis on those tweets to determine a dominant emotion (e.g .joy, anger, sadness, etc.), and then changes the colorof his LED based on that emotion.

Deep dive on Bluemix

Introduction to Bluemix or *IBM* Cloud

computing services from IBM that offers both platform as a service (PaaS) and infrastructure as a service (IaaS).

FIGURE 6 (A): TJBOT Workshop

FIGURE 6 (B): Panning Objectives For TJBOT workshop

FIGURE 6 (C): Preparation For Setup

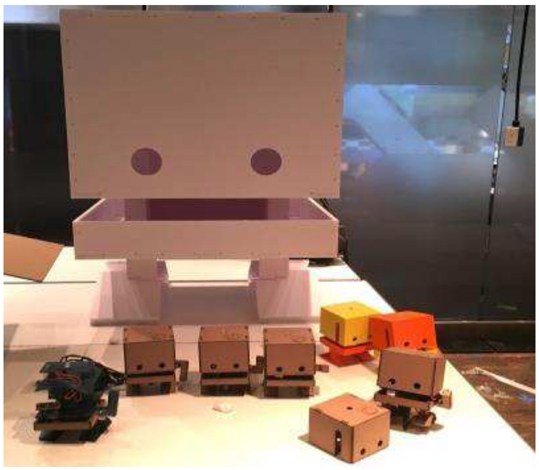


FIGURE 6 (D): TJBOT

With IBM Cloud IaaS, organizations can deploy and access virtualized IT resources -- such as compute power, storage and networking -- over the internet. For compute, organizations can choose between bare-metal or virtual servers.

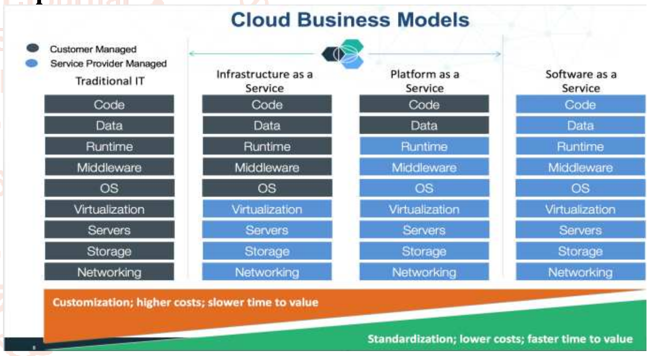
With IBM Cloud PaaS -- which is based on the open source cloud platform Cloud Foundry -- developers can use IBM services to create, manage, run and deploy various types of applications for the public cloud, as well as for local or on-premises environments. IBM Cloud supports various programming languages, such as Java, Node.js, PHP and Python and extends to support other languages.

IBM Cloud deployment models
IBM offers three deployment models for its cloud platform:
Public: A public cloud that provides access to virtual servers in a multi-tenant environment. An enterprise can choose to deploy its applications in one or multiple geographical regions.

Dedicated: A single-tenant private cloud that IBM hosts in one of its data centers. An enterprise can connect to the environment using a direct network connection or virtual private network (VPN), and IBM manages the platform.

IBM Cloud Private: A version of the IBM platform that an organization deploys as a private cloud in its own data center behind a firewall.

Top 5 benefits of IBM Bluemix



Bluemix provides developers a platform as a service (PaaS) where they can easily compose new business applications without the overhead of setting up the underlying architecture. Bluemix enables web and mobile applications to be rapidly and incrementally composed from services. These services are provided by IBM, the open source community and an ecosystem of partners.

1. Simplicity and speed:
By focusing on the DevOps model, Bluemix can reduce the downtime of redeploying applications. Continuous delivery is one way this can be provided. The integrated environment provided by Bluemix allows developers to automatically deliver code without the hassle of building and debugging installation scripts. This reduces the time needed to manage code delivery and puts it in the hands of the testers and user community faster. The application can be deployed to multiple spaces which allow segregation of environments for development, testing and production. Automatically delivering code keeps developers focused on coding, not installation.

2. Agility:
Bluemix allows developers to focus on delivering business value, rather than on maintaining the development environment, by scaling environments elastically based on business demand. Instead of manually deploying workloads, Bluemix will automatically redeploy workloads to other virtual machines (VMs) if there is an outage. To provide continuous availability, Bluemix abstracts the underlying architecture and keeps the manageability of services and applications at an easily understood level. Users are able to stop or start applications and define how much memory is associated with each application while Bluemix manages the rest.

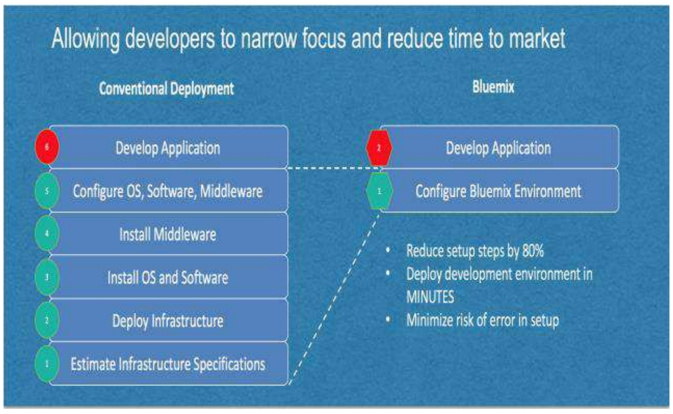
3. Tools:
With Bluemix, developers have the freedom to choose the development tools that work best for them. Developers don't always want to work with the same tool sets and Bluemix provides several options, including the following:

- **Command line:** The Cloud Foundry (CF) command line provides integration for developers that prefer coding without an integrated development environment (IDE). This is also helpful for developing automation scripts with Bluemix. The CF application programming interfaces (APIs) can be integrated with multiple languages, frameworks and services.
- **Eclipse:** Since Eclipse is widely used by developers, they can continue to use the tools with which they are comfortable. The Cloud Foundry integration can be installed from the Eclipse Marketplace. This provides integration with Bluemix from the Eclipse client.
- **Web IDE:** Developers can work with the Web IDE directly in Bluemix. This allows modification of the application without any development environment installed on the developers' laptops.

4. Source control:
Bluemix also comes with integration to several source control management (SCM) systems. These include Git, GitHub and Jazz SCM. These environments can be configured to deliver application changes continuously. Open source Cloud Foundry applications can be forked and loaded to Bluemix. This provides a great place to start development of a new project.

5. Services marketplace:
Services leverage APIs and software development kits (SDKs) that can quickly and easily be incorporated with Bluemix applications. Although IBM provides many services, Bluemix offers an open and flexible ecosystem which allows other companies to provide services that can be integrated into applications. Companies can be both providers and users of services. Two types of services are available:

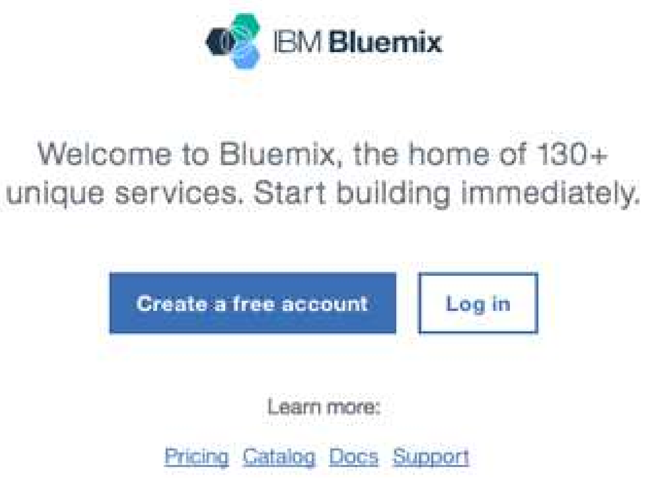
- The "Managed Service Offering" is not available in the beta, but will be available. Services in the marketplace can be at no charge or have a pay as you go (PAYG) pricing model. These services are available to all Bluemix customers.
- "User Provided Services" can be added so that organizations can share services within their organization. This promotes more reuse and standardization of services within the company.


Allowing developers to narrow focus and reduce time to market
Conventional Deployment
Bluemix
Develop Application
Configure OS, Software, Middleware
Install Middleware
Install OS and Software
Deploy Infrastructure
Estimate Infrastructure Specifications
Develop Application
Configure Bluemix Environment
Reduce setup steps by 80%
Deploy development environment in MINUTES
Minimize risk of error in setup

IBM Watson
Watson is an IBM supercomputer that combines artificial intelligence (AI) and sophisticated analytical software for optimal performance as a "question answering" machine. The supercomputer is named for IBM's founder, Thomas J. Watson.

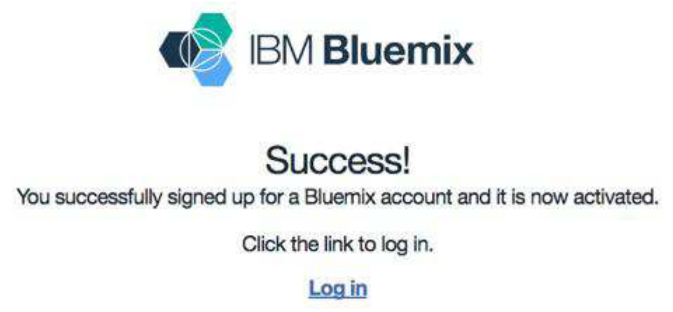
Watson's key components include:
- Apache Unstructured Information Management Architecture (UIMA) frameworks, infrastructure and other elements required for the analysis of unstructured data.
- Apache's Hadoop, a free, Java-based programming framework that supports the processing of large data sets in a distributed computing environment.
- SUSE Enterprise Linux Server 11, the fastest available Power7 processor operating system.
- 2,880 processor cores.
- 15 terabytes (TB) of RAM.
- 500 gigabytes (GB) of preprocessed information.
- IBM's DeepQA software, which is designed for information retrieval that incorporates natural language processing (NLP) and machine learning.

Create a Bluemix Account
1. Visit http://www.bluemix.net in your web browser.
2. Press the Create a free account button:



Welcome to Bluemix, the home of 130+ unique services. Start building immediately.

3. Complete the form and press the Create Account button.
4. You will receive an email asking you to confirm your email address. Check your email and click the Confirm Account link.
5. Once confirmed, you will be asked to Log in. Click on the Log in link and follow the instructions to enter your credentials and log in.


IBM Bluemix

Success!

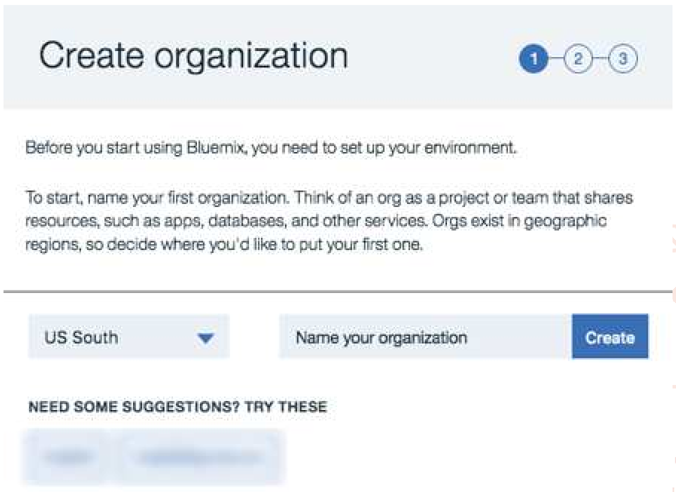
You successfully signed up for a Bluemix account and it is now activated.

Click the link to log in.

Log in

Set up your Bluemix Organization

1. You will be prompted to **Create organization**. Enter an organization name (notice that there are suggestions for you). Also select an appropriate region. Then press the Create button.


Create organization

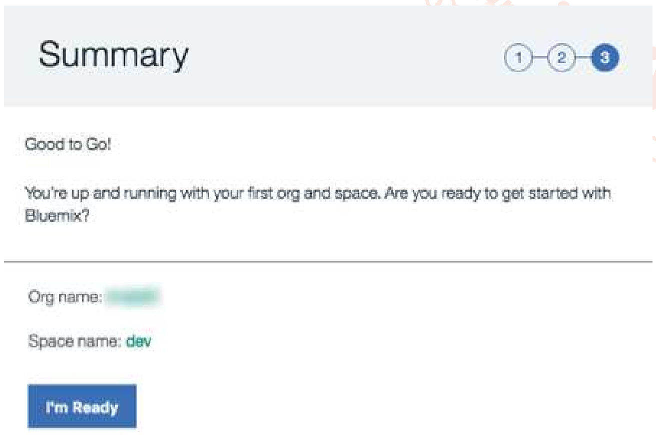
Before you start using Bluemix, you need to set up your environment.

To start, name your first organization. Think of an org as a project or team that shares resources, such as apps, databases, and other services. Orgs exist in geographic regions, so decide where you'd like to put your first one.

US South | Name your organization | Create

NEED SOME SUGGESTIONS? TRY THESE

2. Next you will be prompted to Create space. Name your space dev and click on the Create button.
3. Finally, you will see the Summary page where you can review your entries. Click the I'm Ready button.


Summary

Good to Go!

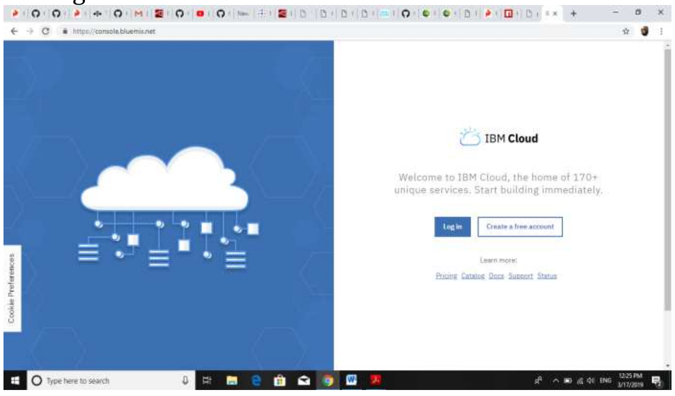
You're up and running with your first org and space. Are you ready to get started with Bluemix?

Org name:

Space name: dev

I'm Ready

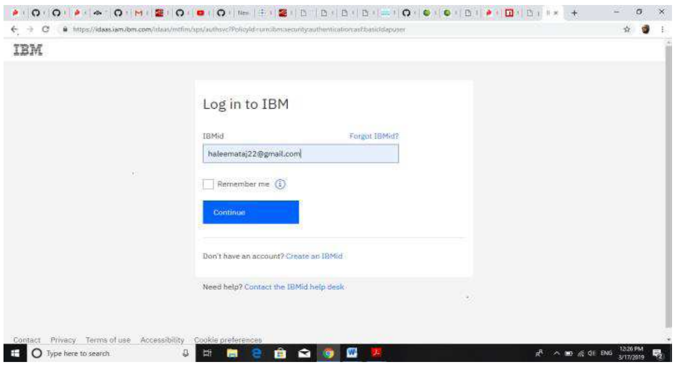
4. log in to ibm cloud


IBM Cloud

Welcome to IBM Cloud, the home of 170+ unique services. Start building immediately.

Log in | Create a free account

Learn more:
Pricing Catalog Docs Support Status


IBM

Log in to IBM

IBMid: haleemataj22@gmail.com

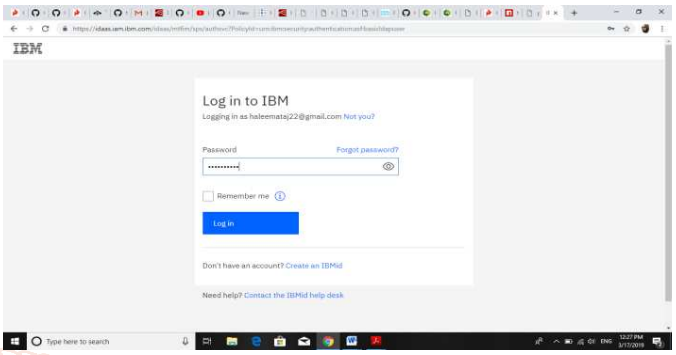
Forgot IBMid?

Remember me

Continue

Don't have an account? Create an IBMid

Need help? Contact the IBMid help desk


IBM

Log in to IBM

Logging in as haleemataj22@gmail.com Not you?

Password

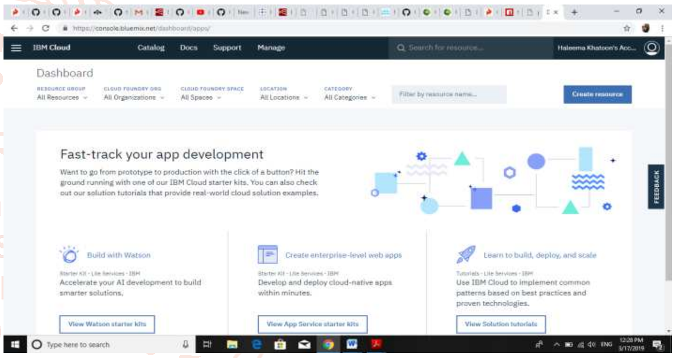
Forgot password?

Remember me

Log in

Don't have an account? Create an IBMid

Need help? Contact the IBMid help desk


IBM Cloud | Catalog | Docs | Support | Manage | Haleema Khatoon's Acc...

Dashboard

RESOURCE GROUP: All Resources | CLOUD FOUNDRY ORG: All Organizations | CLOUD FOUNDRY SPACE: All Spaces | LOCATION: All Locations | CATEGORY: All Categories

Create resource

Fast-track your app development

Want to go from prototype to production with the click of a button? Hit the ground running with one of our IBM Cloud starter kits. You can also check out our solution tutorials that provide real-world cloud solution examples.

Build with Watson
Starter Kit · Lite Services · IBM
Accelerate your AI development to build smarter solutions.
View Watson starter kits

Create enterprise-level web apps
Starter Kit · Lite Services · IBM
Develop and deploy cloud-native apps within minutes.
View App Service starter kits

Learn to build, deploy, and scale
Tutorials · Lite Services · IBM
Use IBM Cloud to implement common patterns based on best practices and proven technologies.
View Solution tutorials

Bluemix Catalog

Watson Assistant

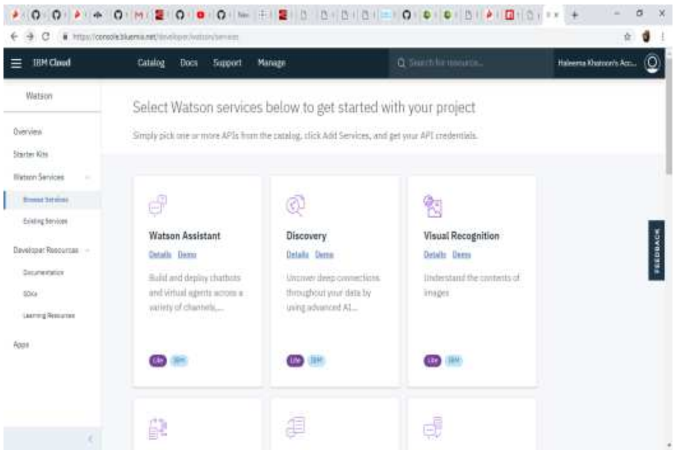
Discovery

Visual Recognition

Natural Language Classifier

Tone Analyzer

Watson Knowledge Studio


IBM Cloud | Catalog | Docs | Support | Manage

Watson: Overview, Starter Kits, Watson Services, Browse Services, Existing Services, Developer Resources, Documentation, SDKs, Learning Resources, Apps

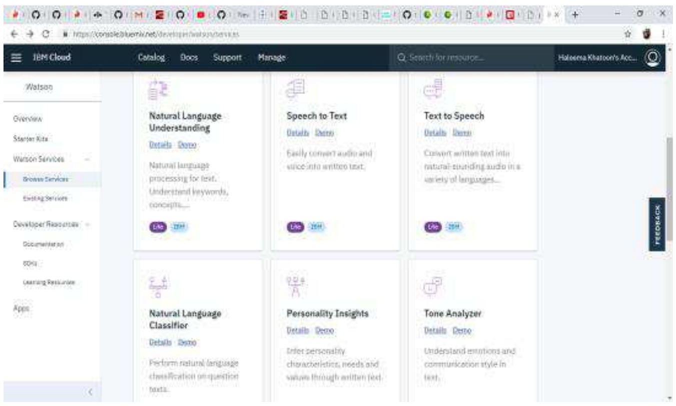
Select Watson services below to get started with your project

Simply pick one or more APIs from the catalog, click Add Services, and get your API credentials.

Watson Assistant
Details Demo
Build and deploy chatbots and virtual agents across a variety of channels,...

Discovery
Details Demo
Uncover deep connections throughout your data by using advanced AI...

Visual Recognition
Details Demo
Understand the contents of images



Natural Language Understanding
Details Demo
Natural language processing for text. Understand keywords, concepts,...

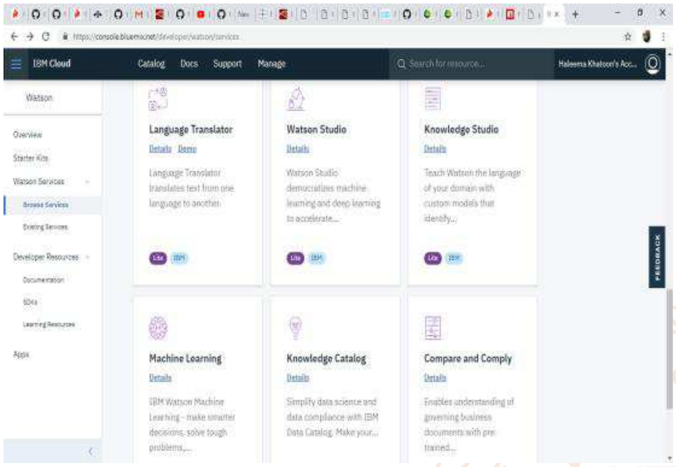
Speech to Text
Details Demo
Easily convert audio and voice into written text.

Text to Speech
Details Demo
Convert written text into natural-sounding audio in a variety of languages...

Natural Language Classifier
Details Demo
Perform natural language classification on question texts.

Personality Insights
Details Demo
Infer personality characteristics, needs and values through written text.

Tone Analyzer
Details Demo
Understand emotions and communication style in text.



Language Translator
Details Demo
Language Translator translates text from one language to another.

Watson Studio
Details
Watson Studio democratizes machine learning and deep learning to accelerate...

Knowledge Studio
Details
Teach Watson the language of your domain with custom models that identify...

Machine Learning
Details
IBM Watson Machine Learning - make smarter decisions, solve tough problems,...

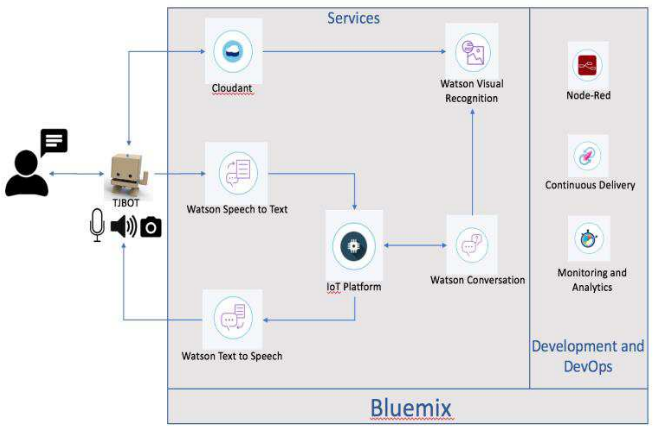
Knowledge Catalog
Details
Simplify data science and data compliance with IBM Data Catalog. Make your...

Compare and Comply
Details
Enables understanding of governing business documents with pre trained...

EXISTING SYSTEM

- TJBOT has received global attention. People in Italy and Brazil got TJBOT to speak Italian and Portuguese. They received pictures of elementary school kids in South Africa building and painting their TJBOT.
- TJBOT has been programmed as a security device, a personal weather station, to recognize sesame street characters, to control drones, or just being a charming caring companion.
- In the CHI demo session, They will walk the audience through how to get started, build their very own TJBOT, and use one of his initial recipes to bring him to life.
- They will also be able to interact with TJBOT and test out some of his capabilities. In this demo session, they can interact with Watson Conversation, Visual Recognition, and Tone Analyzer.
- TJBOT is connected to Twitter monitoring the conversations around CHI.
- He uses Watson Tone Analyzer to understand emotions and shines different colors based on that.
- For example, he shines yellow when people are happy. The audience can interact with TJBOT through Twitter but also ask questions and provide voice commands to control him.
- Some examples are: “What do you see”, “introduce yourself”, “play some music”, or “Dance”.

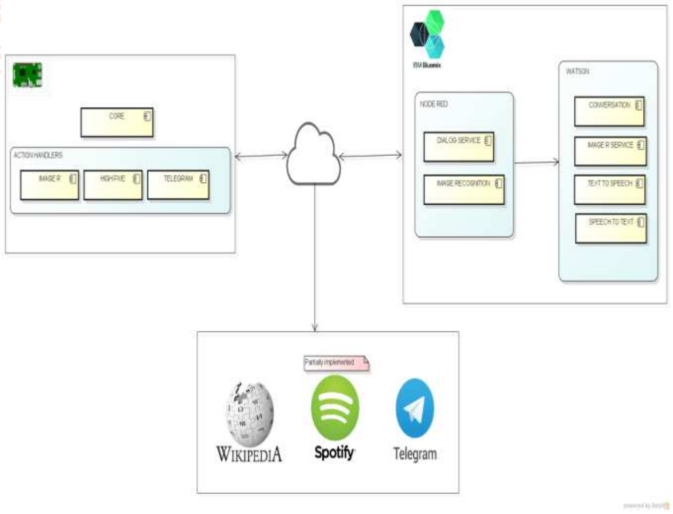
DESIGN ARCHITECTURE ARCHITECTURAL OVERVIEW



The above diagram provides an overall architectural view of the Bluemix experience demo. On this demo, Bluemix is showcase it as a development echo system where infrastructure, Watson, dev ops, object storage, IoT and multiple runtimes coexist on a single framework easy, practical and fun to use. For this demo, we put together a set of different functionalities into the mix:

- Boilerplates:
  Node-red Boilerplate: A nodejs environment with Node-red with cloudant.
- Watson Services
  Watson Speech to text
  Watson Text to Speech
  Watson Conversation
  Watson Visual Recognition
- DevOps
  Github
  Continuous delivery pipeline
  Monitoring and Analytics
- IoT
  Internet of Things Platform

Overall flow – Architecture

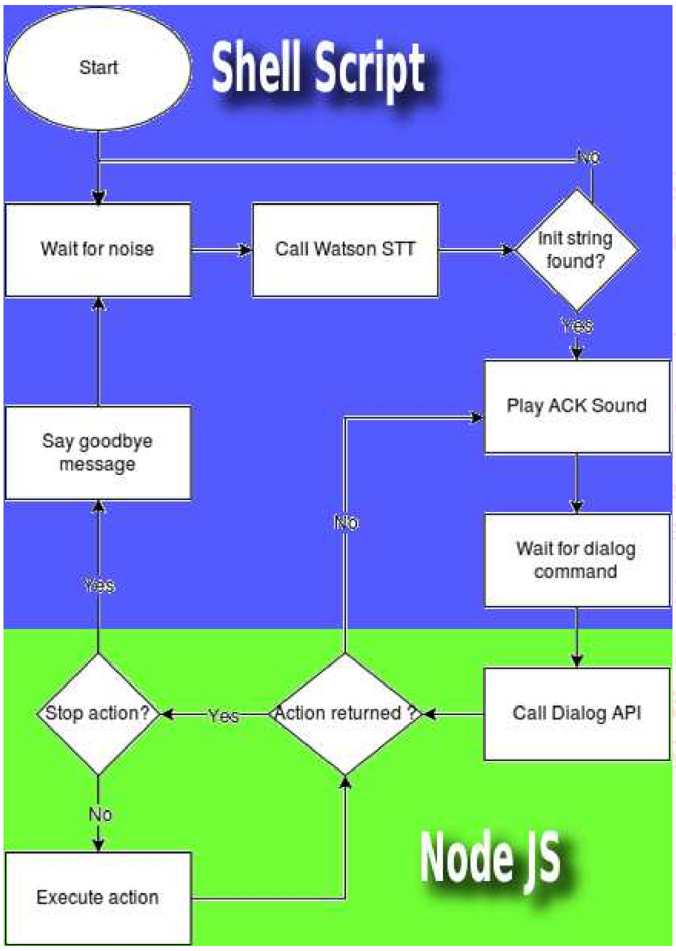


- The main daemon will be started and will continuously listen for a key word/sentence (like “Hey, TJBOT”) and will call the supporting services. Upon receiving of a message with an ACTION, it will call a factory for selecting one of the action services;
- Action handler services are the building blocks for TJBOT to do things. There will be a built-in for the action code “SAY”, so you can talk to TJBOT right out of the box;

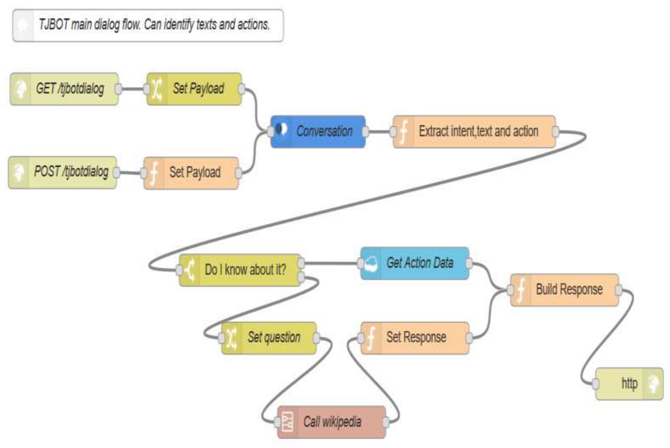
- ➢ All services will consume and reply JSON messages (technology agnostic);
- ➢ Can be implemented using any language/tool: node.js, node red, java, python;
- ➢ Will return a pre-defined status code along with the actual response;
- ➢ They can, if needed, return another action. The core will consume and execute the new action;
- ➢ Actions can be executed sequentially or even in parallel. So, for example, if you have an action for TJBOT to move its arm, it could be executed at the same time as a SAY action for doing a compliment;
- ➢ Action handlers are stateless.

FLOW CHART ACTIVITY DIAGRAM

An activity diagram depicts the flow of actions that different users do according to their respective roles.



Dialog service node-red flow



- ➢ This service will accept a text and will call Watson conversation services to understand what is being asked. The text can indicate two things:

1. Regular text which we should use conversation and start a conversation;
2. Action(s) to be taken.

- ➢ The service will also make calls to Wikipedia apis when it doesn't know about a subject. This call will be fired for two conversation intents: #whatis and #whois.

Future Enhancement

Future work is needed to focus on further automating aspects of the kit's software and hardware setup, and creating visual programming interfaces that allow younger or lower-skilled users to prototype solutions.

A. to take TJBOT massively available,

B. work together with the maker community to expand on the capabilities of TJBOT and simplify his software libraries, and

C. continue working with teachers to explore the potential of TJBOT for education.

CONCLUSION

We believe TJBOT achieves this goal in several ways. In addition, assembly can be performed by very young children under supervision, resulting in a fun artifact that they can make their own via decoration; no code or AI required.

For those with more advanced skills, even the wiring of the electronics can be performed simply, without soldering. Finally, the software developed for TJBOT hides much of the complexity of interfacing with hardware and cloud APIs, and enables one to program at the level of the bot's capabilities: speak, see, listen, shine, wave, etc.

TJBOT is an open source, interactive robot designed to encourage people to build with cognitive services in a fun way. He is a paper robot, which can also be 3D printed, and comes with an initial set of recipes that bring him to life.

REFERENCES

[1] 2017. node-red-contrib-TJBOT. Retrieved September 21, 2018 from https://github.com/jeancarl/node-red-contrib-TJBOT

[2] 2017. TJBOT-Node-RED. Retrieved September 21, 2018 from https: //github.com/johnwalicki/TJBOT-Node-RED

[3] 2017. TJBOTlib. Retrieved September 21, 2018 from https://github.com/ ibmTJBOT/TJBOTlib

[4] Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Zheng. 2016. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed

Systems. In OSDI'16 Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation. 265–283. http://arxiv.org/abs/1603.04467

[5] Google AIY. 2018. Do-it-yourself artificial intelligence. https://aiyprojects.withgoogle.com/

[6] MG Ames, J Bardzell, S Bardzell, and S Lindtner. 2014. Making cultures: empowerment, participation, and democracy-or not? CHI'14 Extended (2014). http://dl.acm.org/citation.cfm?id=2579405

[7] Chris Anderson. 2012. Makers: The new industrial revolution. Vol. 1. 559–576 pages. https://doi.org/10.1016/0016 3287(91)90079-H

[8] Maryam Ashoori. 2016. Calling all Makers: Meet TJBOT! Retrieved September 21, 2018 from https://www.ibm.com/blogs/research/2016/11/calling-makers-meet-tj-bot/

[9] Andrew H. Beck, Ankur R. Sangoi, Samuel Leung, Robert J. Marinelli, Torsten O. Nielsen, Marc J. van de Vijver, Robert B. West, Matt van deRijn, and Daphne Koller. 2011. Systematic Analysis of Breast Cancer Morphology Uncovers Stromal Features Associated with Survival. Science Translational Medicine 3, 108 (2011), 113–108. https://doi.org/10.1126/scitranslmed.3002564

[10] James Bergstra, Olivier Breuleux, Frederic FrAl'dAl'ric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math compiler in Python. Proceedings of the Python for Scientific Computing Conference (SciPy) Scipy (2010), 1–7. http://www-etud.iro.umontreal.ca/~wardefar/publications/theano_scipy2010.pdf

[11] M. Bertozzi, A. Broggi, M. Cellario, A. Fascioli, P. Lombardi, and M. Porta. 2002. Artificial vision in road vehicles. Proc. IEEE 90, 7 (7 2002), 1258–1271. https://doi.org/10.1109/JPROC.2002.801444

[12] Erik Brynjolfsson and Andrew Mcafee. 2017. The business of artificial intelligence. Harvard Business Review (2017). https://hbr.org/cover-story/2017/07/the-business-of-artificial-intelligence

[13] Leah Buechley and Michael Eisenberg. 2008. The LilyPad Arduino: Toward Wearable Engineering for Everyone. IEEE Pervasive Computing 7, 2 (4 2008), 12–15. https://doi.org/10.1109/MPRV.2008.38

[14] Leah Buechley and Benjamin Mako Hill. 2010. LilyPad in the wild: how hardware's long tail is supporting new engineering and design communities. In Proceedings of the 8th ACM Conference on Designing Interactive Systems - DIS '10. ACM Press, New York, New York, USA, 199. https://doi.org/10.1145/1858171.1858206