Mais

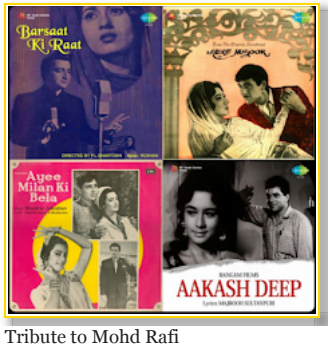
RJV 's Blog

Selecionar idioma Tecnologia do Google Tradutor

Catch - 22 | An unfinished novel | Random rhymes | KuljaSimSim | Ludo | Bricks | TicTacToe | GREYcells Instograf | Home

Short stories - Reluctant blackmailers | Awsss - animated gif of the Bay of Bengal sea from...

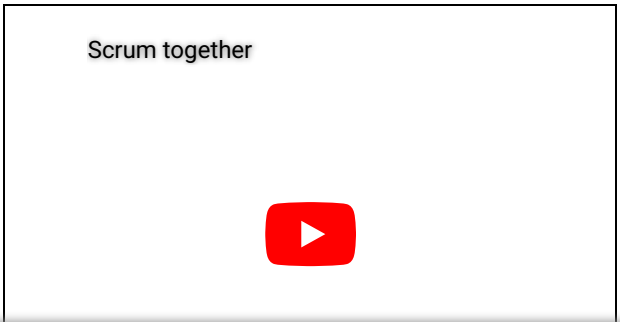
41 everlasting melodies of Mohd Rafi



Tribute to Mohd Rafi

Friday, 13 March 2020

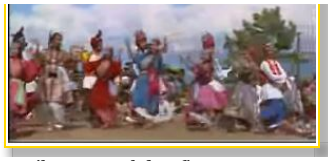
Scrum together



Google Ver esta página em: Português Traduzir Desactivar para: Inglês Opções ▼

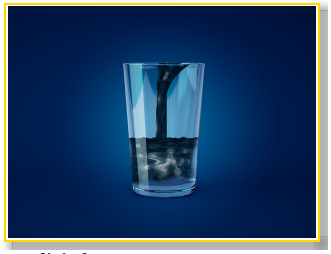
endereço IP e agente do utilizador são partilhados com a Google, bem como o desempenho e a métrica de segurança, para assegurar a qualidade do serviço, gerar as estatísticas de utilização e detetar e resolver abusos de endereço.

OBTER MAIS INFORMAÇÕES OK



A tribute to Mohd Rafi

All About Agile



My digital resume

Blog Archive

Posted by RJV at 17:05 No comments:

Thursday, 24 October 2019

Automating BDD - CI/CD, BizDevOps

39 everlasting melodies of Mohammed Rafi

39 everlasting melodies of Mohammed Rafi

Pages

- Short stories - Reluctant blackmailers
- An unfinished novel
- Catch - 22
- Random rhymes

Certified Safe 4 Agilist

SA

Certified ScrumMaster

CSM

Simplifying the complexity in the two very important workings of any software development venture is the goal of Demystifying the anomalies between this complexity and the business is the need for BizDevOps.

I had tried implementing something similar in a complex environment, when the transformation of the organizati I was leading the implementation of the changes, when I proposed an end-to-end framework that would enable th requirement right from the Excel sheet of the Business Analyst through unit tests that would map each requireme each item would have a traceability in audits.

So, if an auditor clicked on a deployed item, the entire sequence of unit and functional and other associated tests history,

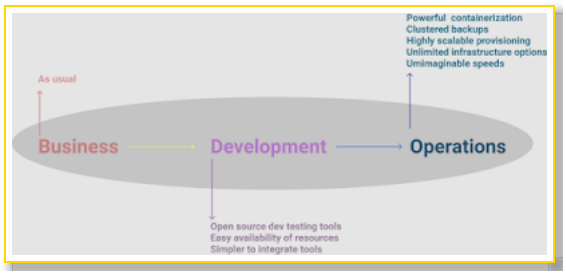
Since those days of 2008-9, tools have evolved to make that kind of a dream possible but the takers have not been that having every business rule or requirement clearly elaborated through tests into implementation as features is of ensuring zero waste in the development efforts of the team and a high productivity result for the organization.

That said, as I evolved from .Net to open source to Node.js to the cloud, the power of the testing tools in the mark the simple N/J-Behave, fitnesse-slim tools that could be used to elaborate features and scenarios, respectively, in names like Coffee, Cucumber and forks signaled that no longer were techies concerned about the conventional an open source world even forced the mighty Microsoft world of proprietary software to bundle scripting language in Visual Studio dev environment!

This was the sign of things to come and Visual Studio Code kind of reiterated that having your product on the des developer is more important than just selling - I wonder how it would be if Rolls Royce came up with the poor ma as Visual Studio Code was for many as it brought the years of expertise of the IDE that clearly defined a developm unlike other IDEs (even Eclipse, the PDE) and brought user-friendliness to the world of developers as they strive same in their own software products.

I had to elaborate the progression of products that redefined many aspects of software development to attain the the world sees today so that the need for the maturity that saw the drive happening can define the standard of qua that is expected of all software by any kind of user - poor or rich, educated or uneducated - in the incredibly fast s

And so, as I felt in 2008, Agile or no Agile movement, every organization, unless the whole world of business wen want to see that the business requirements were as clearly translated to the development stream as test cases wer the head start of venturing into BDD then with slim tools like SLIM, I found the new testing tools in mocha, chai, just ideal to create a tightly integrated flow between Business (Biz), Development, Testing and Operations - BizDe



Imagine a senior executive typing a sudden idea of a feature while commuting by a metro in Chennai and before t feature has been tested, deployed for approval, approved and the development team sitting in Ukraine (because o time zones) have started implementing the feature is under way !

It is not the swag in it that should be appealing but the lack of awareness to not make the best use of it and wait fo revolution to happen before sane models are adopted as standards which is.

The IT operations were always considered as the river of misery in a software development team, always promisin delivering due to process and network bandwidth constraints. This huge bottleneck has now been resolved with cl providing the kind of speed and infrastructural freedom that is more a dream than a reality that is widely accepte

The fact is, if the business executive cannot check-in their (plural used for gender parity) idea and if the technolog check-in to be instantly broadcast across the team and if there were no cloud computing available on the mobile t not be the reality that it is today.

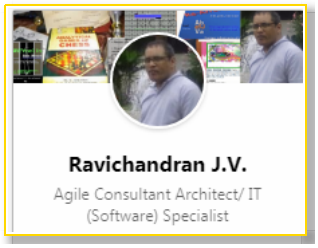
The sheer potential of the available technological advancement is what makes the BiZDevOps an absolute essenti forward with for any organization wishing to achieve Agile standards of software development.

Another great innovation in the software development technology field, aside from the sublime cloud computing, technology - cloud functions (Lambdas and stuff for the AWS cloud folks and Azure functions for the Microsoft de freedom to design scalable and seamless architecture at will.

MCP - Transcript id: 660863, Access code: 2004jvmcp

MCP (id - 3156859) in ASP.Net with C#/VB.Net

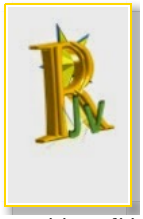
RJV's LinkedIn Profile



StackOverlow profile

user2347763
461
●2 ●10

RJV - on tumblr



Practising politics everyday with or without patrons, makes not a politician but the whole set into pests.

Agile Test Driven Design

Safari Books Online

Facebook - unplugged

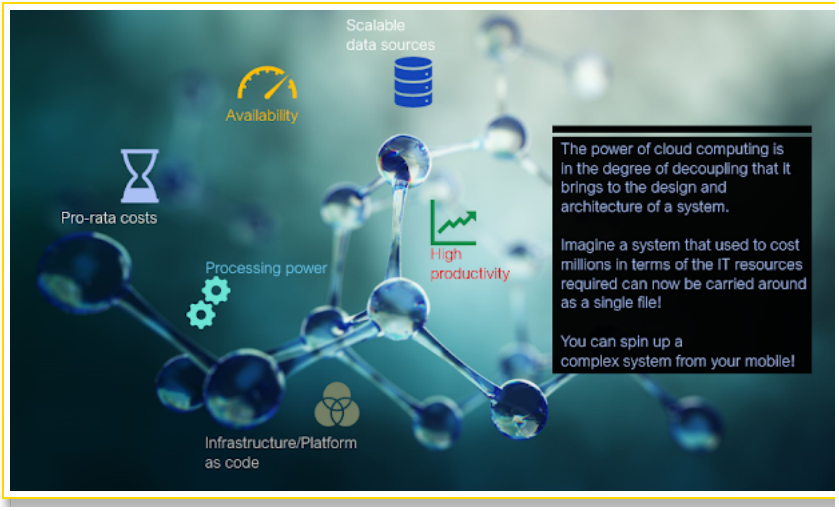
Know Facebook better!

NUnit Fix for platform detection

Open Source Contribution - 2009

Sudoku 3 + 1 - game testing for

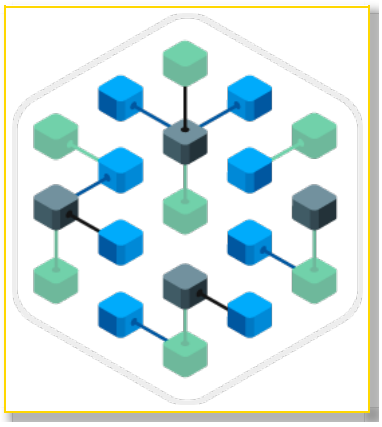
This is what I achieved in 2018 with the help of Google Cloud Functions and the Google Cloud Platform to exploit architecture that effortlessly converts a few hundred lines of Rest Api code in node.js into a powerhouse of logic.



This, in short, is the crux of the matter, when we design a cloud based architecture! Immense savings in processin maintenance with the freedom to be on top of the competition all the time, with no extra costs in periodic upgrad infrastructure would demand aside from the savings in having to invest in the know-how and skilled resources.

Serverless is the keyword in today's world of computing and when you combine it with the demands of an ever ch business requirements and communication needs to stay in business, the cloud is the best platform to survive the

A function is the smallest unit of every software system. And when this small unit is made to perform all on its ow a program or an entity to drive it, the quantum of the potential for speed increases in magnitude and scale. This is of **cloud functions**.



The ability to decouple all dependencies in functional requirements to provide a seamless platform of execution - architecture.

Disparate devices through IoT, sub-systems, disparate platforms, languages or data models work together withou force that is the essence of executing a software in an operating system.

It models the modern, digitized world as is - a business interacting in a language that is not the native language of or an architect but they have to overcome the communication dysfunctions that may arise out of it and develop a

BizDevOps, too, is the representation of the same solution.

And so, extending on the previous post on Automating BDD, I managed to implement a serverless model meant f of business analysts, developers, architects, testers and ops.

The architecture is based on a Rest Api deployed as an Api Gateway on the AWS cloud and the logic in the form o with the datastore hosted on the Atlas MongoDb on the cloud.

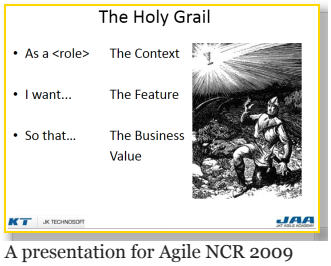
But this is only the design of the software architecture!

There is another architecture that has a say in how the software solution would perform - the technical architectu

VodQA

Unit test with Jasmine - VodQA 2012

BDD + UADD

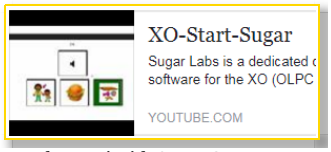


A presentation for Agile NCR 2009

NPM package

Remove duplicates

Raspi-Sugar-XO



Raspberry Pi with Sugar GUI

Conference footprints

1. Mr. Ravichandran J.V. delivers lecture on "Agile Methodology"

2. Software Testing | ThoughtWorks

3. Group Manager, Microsoft Technologies Mr. Ravichandran JV, at Amity University

4. Agile NCR 2010 – at Ansal Institute of Technology, Sector 56, GurgaonProduced by Xebia India in cooperation with ASCI.

5. agiletour.org – Agile tour 2010 New Delhi NCR Role - Chief organizer on behalf of ASCI

My Chess profile on Chess.com

My Chess profile @ Chess.com

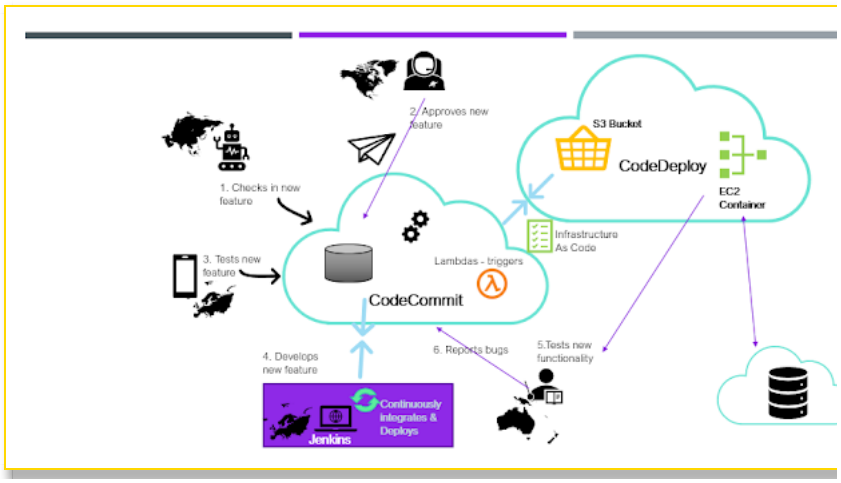
GitHub additions

- Automated NodeJs, gulp, mongodb, express Test
- Spying a ExpressJs route

infrastructure bindings! In the modern world of software development, it is almost a given that a software team is

This distributed form of a team requires the execution of the development process in such a way as to ensure 24x7 code under development. This is achieved through what is known as Continuous Integration, Continuous Deployr Delivery.

And because my solution is designed on the Amazon Cloud, I found using AWS Code Commit and Code Deploy as selection for implementing DevOps and since, I had already invested time and effort in creating an automated sol Jenkins, Jenkins it had to be for CI/CD and not AWS Code Build!



Stay tuned in for my video demonstrating how the entire architecture is implemented with live check-ins of the fe subsequent flow of development and the automated testing, integrating and deploying of the new feature here soc

Posted by RJV at 08:01 No comments:
Labels: aws, BDD, cloudcommit, clouddeploy, serverless

Saturday, 18 May 2019

Automating BDD ...

...with Jenkins, typescript, jest, Jest-Cucumber on node.js

Assumption: Jenkins and NodeJs installed in your local machine.

Continuing with a previous post on BDD, this post is on using your **GitHub** repository as the SCM provider for **J** scenario tests written using the **Gherkin** syntax of *given, when, then* and executing the tests, written in **Typescr** cucumber on Node.Js!

The scenario is

1. Your development and the business teams are located distributed across the world and you are managing th process remotely.
2. The development team waits for the BDD process to pass the requirements by runniing scenario tests to tes
3. This post elaborates how to setup the BDD environment so that all features envisioned by the business own appropriate acceptance criteria and if the tests pass then the next phase, TDD, begins.

The above scenario assumes that the features, written in Gherkin syntax, are uploaded to a GitHub repository (Th travis although this is a Jenkins example, because I had used this repository with travis so the name is so!)

- Unit testing Express JS Routes
- UADD with SlimJs
- TDD with Mocha

Search This Blog

Search

as a Writer (on LinkedIn, tumblr)

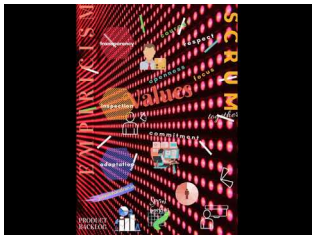
- How solid is SOLID?
- ...thoughtful regards...
- The Sum of all lies
- An awesome person
- Pictoerty - A wretched state

Older posts

- 2020 (1)
  - March (1)
    - Scrum together
- 2019 (6)
- 2018 (9)
- 2017 (12)
- 2016 (5)
- 2015 (33)
- 2014 (22)
- 2013 (38)
- 2012 (39)
- 2011 (2)
- 2010 (6)
- 2009 (18)
- 2008 (2)
- 2007 (1)

Featured post

Scrum together



Total visitors


91,534

Angular Js - Level 3

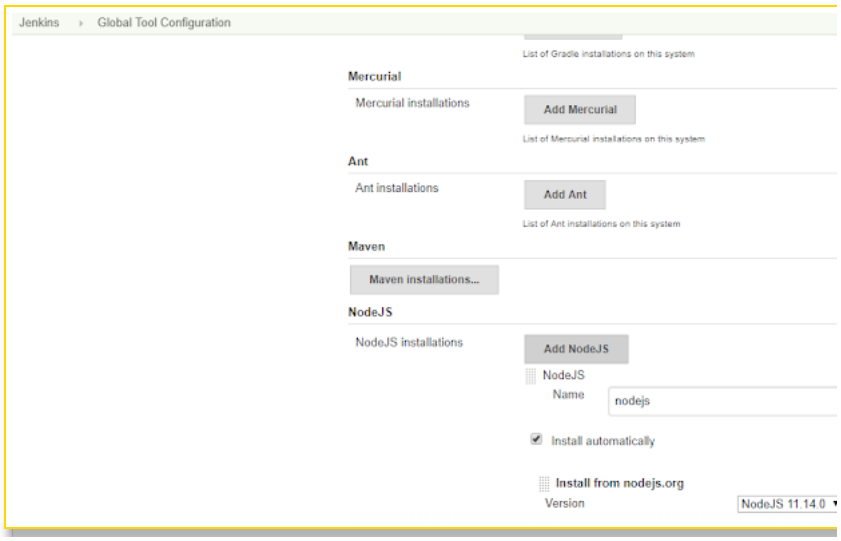
Blends well with the color scheme ;)

At the PlayStore - EasySudoku (2013)

First step, is to tell Jenkins that you are going to build a NodeJs app.

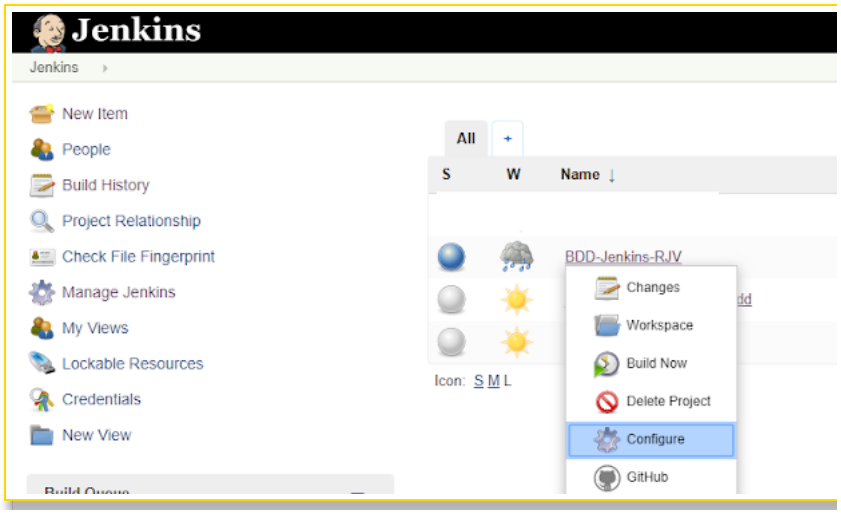
Configure your Jenkins project to include a NodeJs plugin - **Jenkins->Manage Jenkins -> Manage plugins** plugin.

Once the plug in is installed, navigate to the Manage Jenkins page and click on Global Tools Config

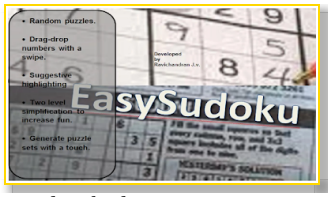


and add a new NodeJs installation and your Jenkins page should like in the above screenshot.

Once the nodejs installation is complete, head to your project and select Configure to configure the GitHub repo d process the **Continuous Integration** with the BDD repository.

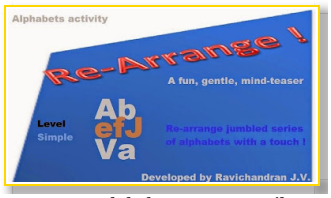


and specify the GitHub repository URL.



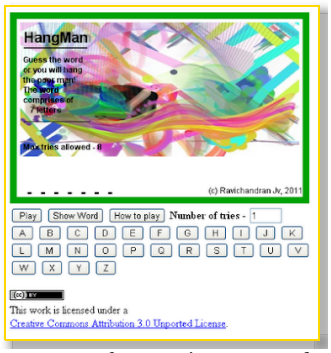
Free download

ReArrange - Fun for kids with Alphabets (2013)



ReArrange Alphabets - temporarily unavailable at the Google Play Store

Hangman (2011) - at GitHub

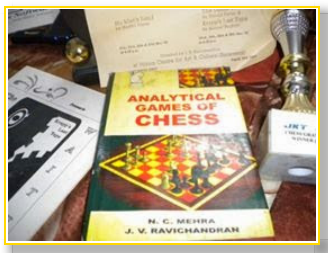


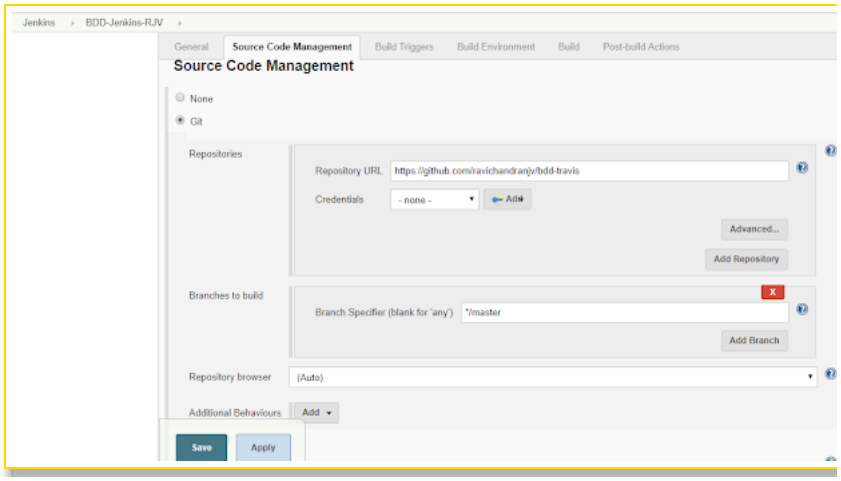
HangMan - the game, in HTML5 and Javascript

Receiving the JKT Chess Grandmaster award from Mr. A. Singhania

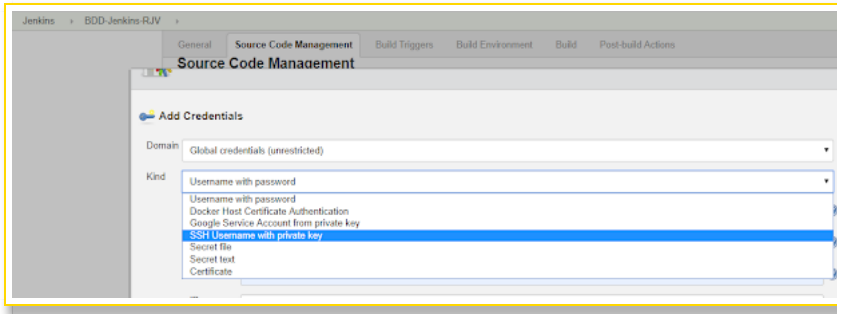
with Scott Meyers

My book on Chess





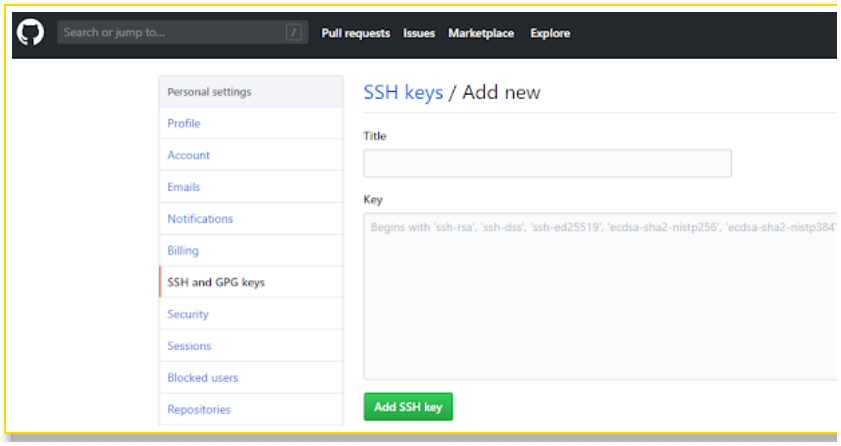
Next step is to configure the access for Jenkins. Click on the Add button next to **Credentials and select SSH u password.**



Enter your GitHub username and a private key. This private key is the SSH key that you could obtain from any SS Putty that will give you a public and private key.

Enter any passphrase that you may have created while generating the SSH key.

Next step is to add the SSH key in your GitHub repository.



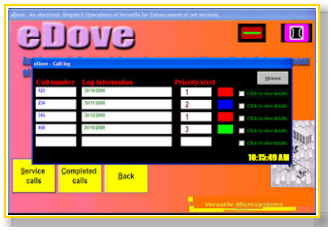
Note: The above steps of adding the SSH key is optional if you are simply using your local machine.

Once everything is set, your Jenkins dashboard show up the files from the GitHub repository in the project works

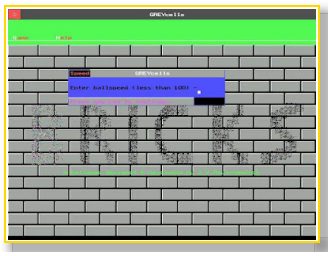
Analytical Games of Chess - book review by The Hindu (over-exaggerated and a little misquoted ;P)

eDove - demo screen (2001)

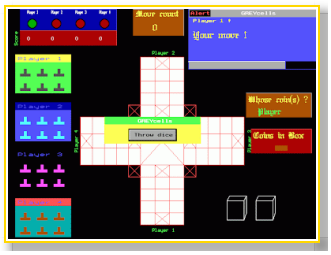
eDove



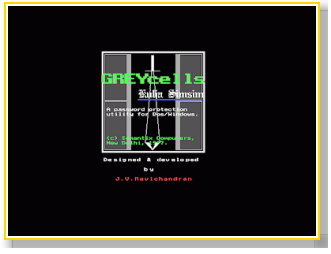
GREYcells Bricks



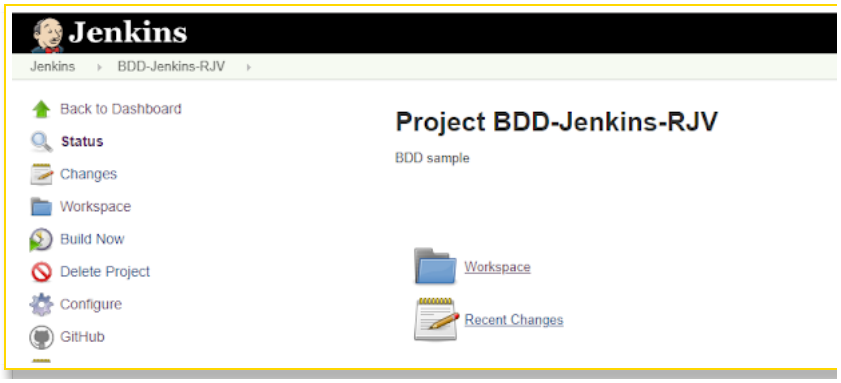
GREYcells Ludo

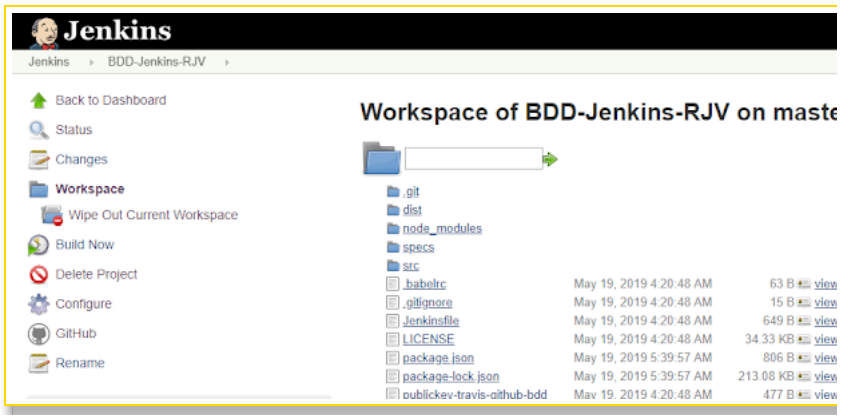


GREYcells KuljaSimSim

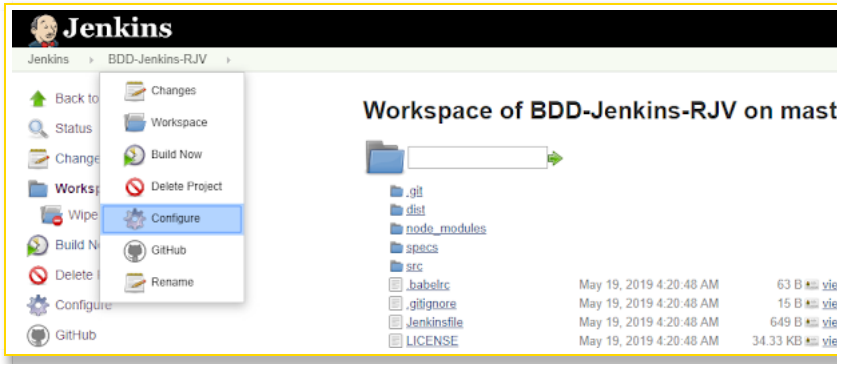


GREYcells Setup

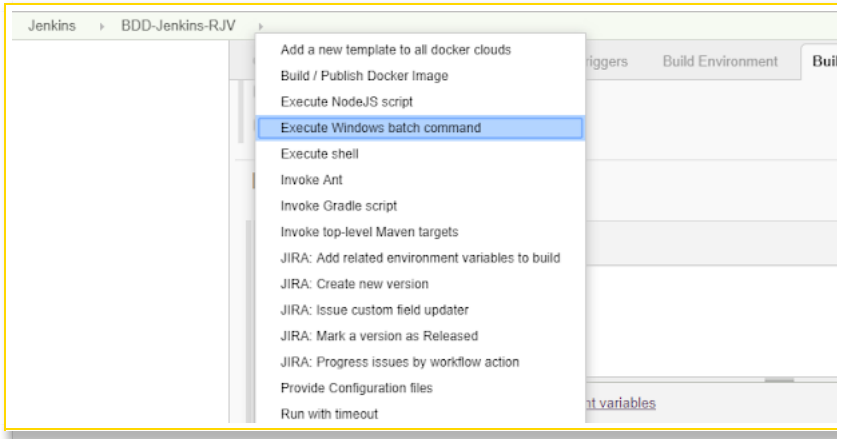




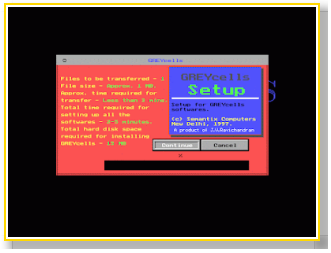
Now to build the workspace.



Under the Build tab, select the Execute Windows batch command under Execute NodeJs Script. The check will let is a NodeJs installation in your Jenkins.

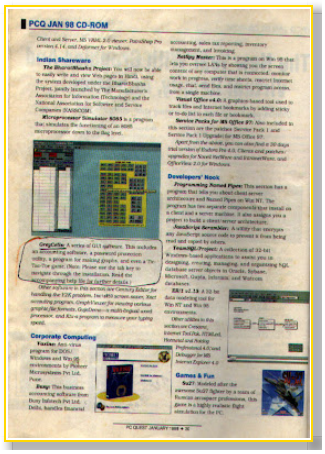


Type "**npm test**" into the textbox - the same command that you will use on your local NodeJs command prompt



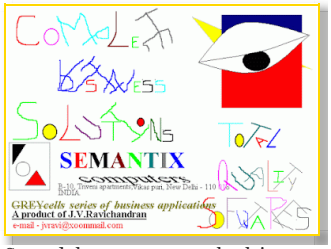
GREYcells PCQuest news

GREYcells in PCQuest news



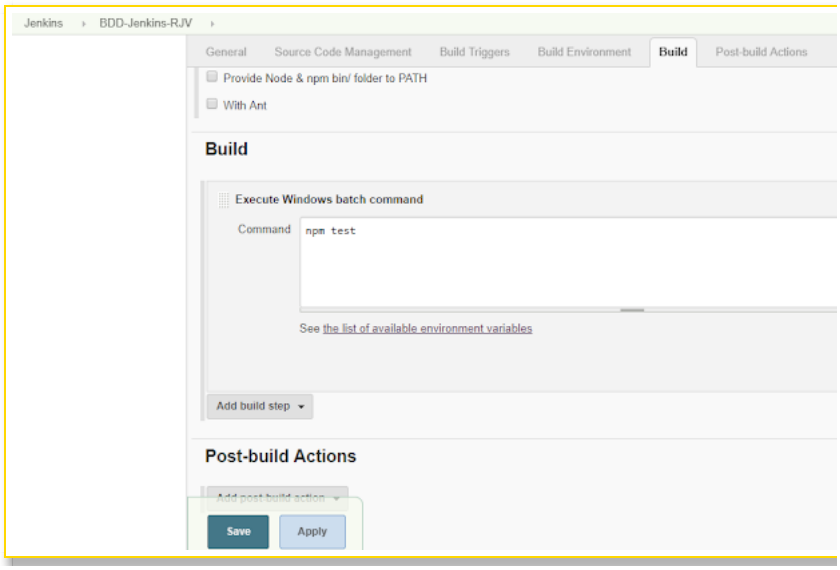
Me as a kid :P

Semantix



Started the company way back in 1992...you were probably not even born then ;P

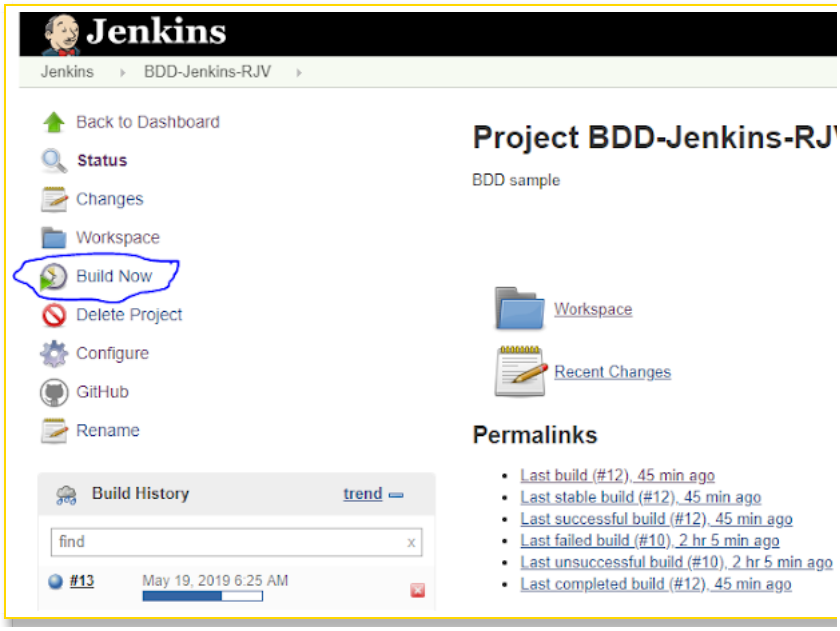
Package.json test script and click Save.



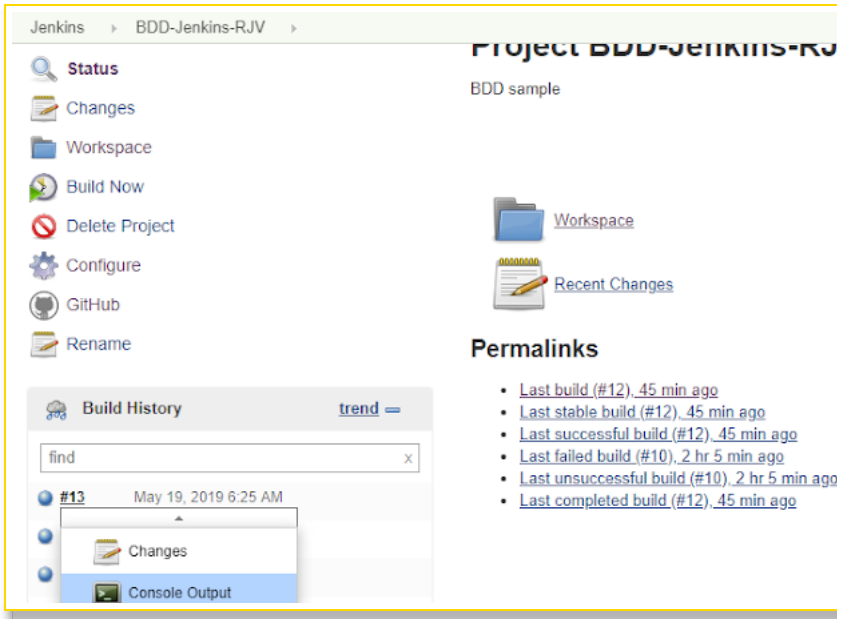
Of course, for first time, you may want to run "**npm install**" to install all nodejs packages required by your GitHu

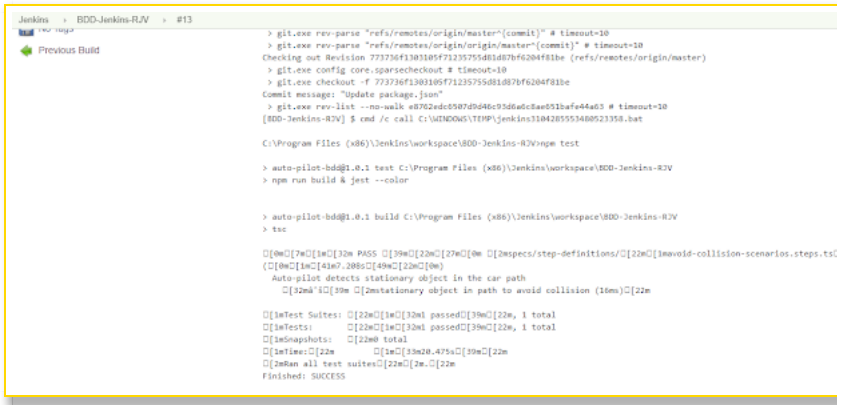
Remember, the execution of the project will happen in your local machine (**or the machine/Docker/Cloud st have installed Jenkins**).

Now, browse back to your project in Jenkins and click **Build now**.



Once the progress bar completes, click the down arrow next to the build # and click **Console output**





The Jenkins workspace will now refresh itself with any changes made to the GitHub and so, as and when newer fe scenario tests, Jenkins will run the tests.

So, as a remote product owner or a stakeholder, all you need to do is run the Jenkins build and check out the resu new features.

And a remote development team, waiting to start its TDD process based on the passed (Approved features), woul and sprint towards its goal of delivering a marketable feature frequently and fully tested!

Happy Automating BDD ! 😀😄😃

Posted by RJV at 18:10 No comments:
Labels: Automation, BDD, Github, jenkins, NodeJs

Wednesday, 15 May 2019

CI/CD - Travis, Jenkins, GitHub, Cloud, DevOps

DevOps is the latest sensation happening in the software development realm not just because of the fascinating a to be used to implement a **SCM** strategy but in the immense challenge that it presents to teams that are on the **Ag** development.

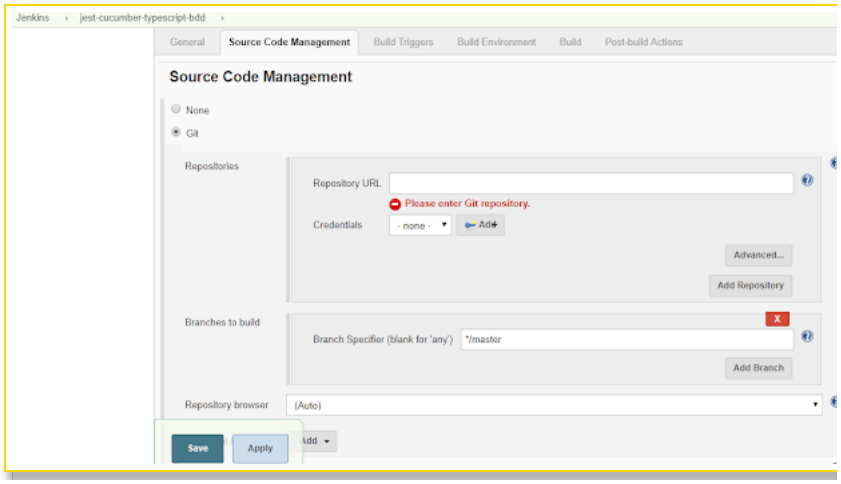
In frameworks like **SAFe**, that has a systems view, the importance of DevOps is immense especially when SAFe c **Behavior Driven Development (BDD)**, **Test Driven Development (TDD)**, Scrum, **XP**, **Kanban** in itself

There are many ways of integrating code in a SCM repository but the key factor is in selecting the right tools to ha pipelines to enable all aspects of the **Agile Software Development Lifecycle**.

Traditionally, **Continuous Integration(CI)** means checking out, checking in, committing code and to ensure th

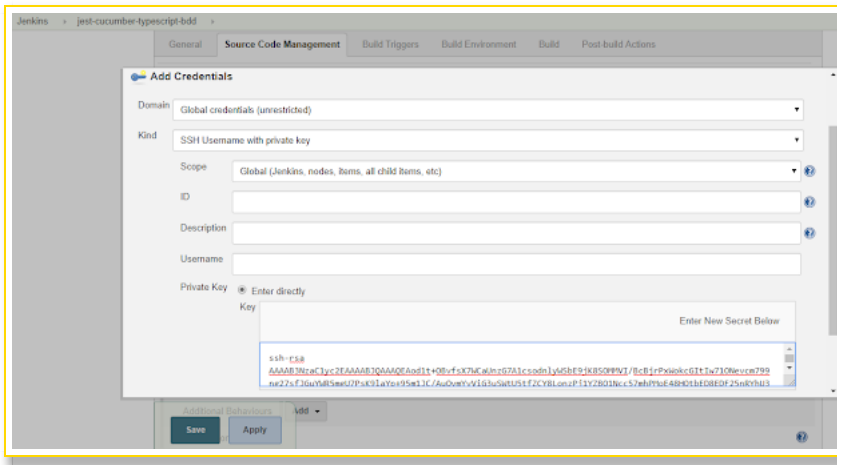
succeeds and the code base is clean for the entire team to use, whenever, wherever and whichever tool they want t from. However, things are not as hunky-dory as it looks because when you check out the tools that are available ir the Cloud based ones, they appear to be free and easy to use so, at first glance, it does look all rosy.

For instance, **Jenkins** has a cool interface that clearly tells the user on which SCM repo to use.

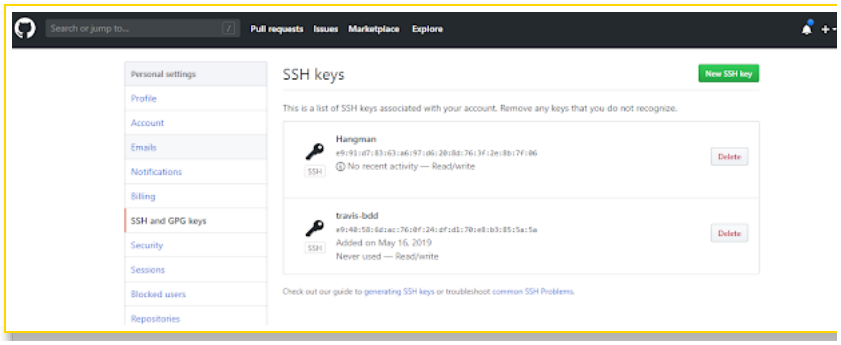


But the real trick is in enabling the access to the SCM for the tool.

Jenkins is an integrator that works on a different machine from the one where your code is hosted - eg., **GitHub** the CI tool to access the repository's status, code, scripts important and since Jenkins is a separate server by itself passed as is. So, what is required is for a secret that only the two tools know so that they could authenticate agains using a standard cryptographic algorithm like a **SSH** key.



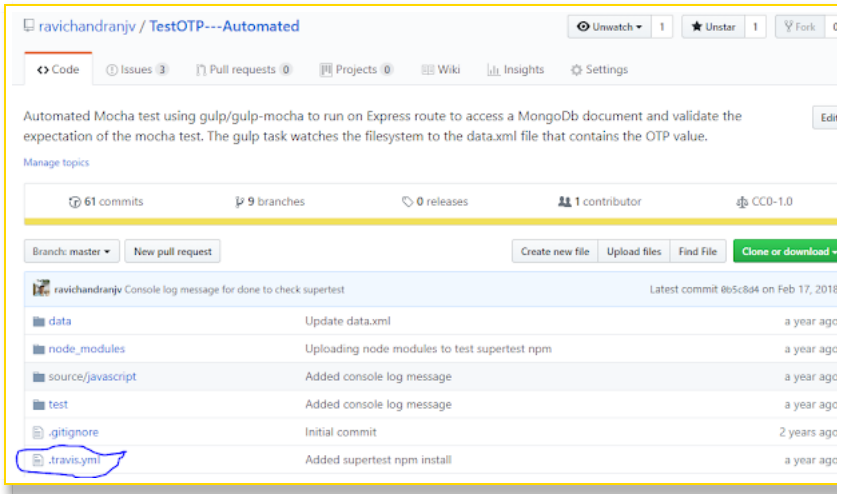
The same key is with the GitHub repository as part of your configuration of the GitHub account.



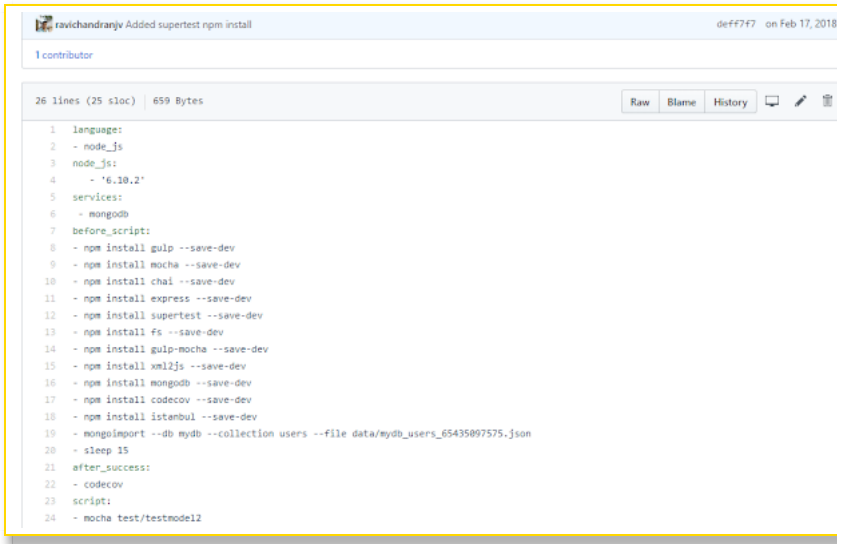
The difference between the various tools like Jenkins, **Travis**, **Azure DevOps** etc is in the way your build config be.

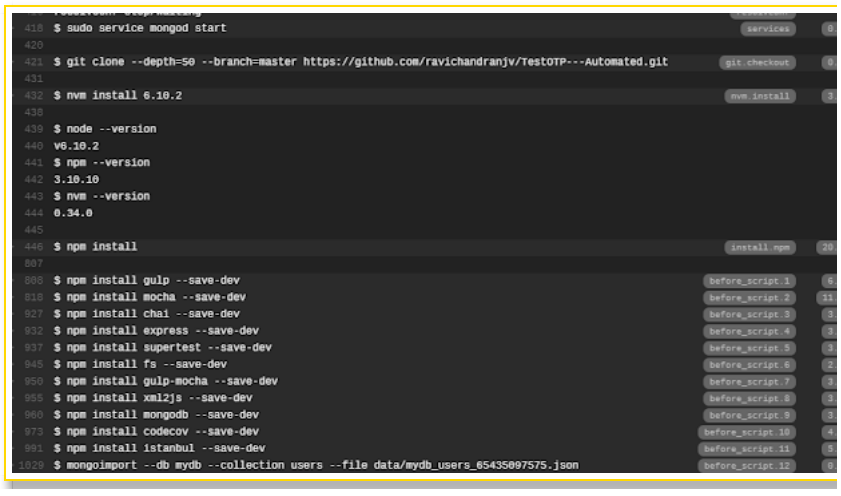
In **Travis**, for example, the **travis.yml** contains all the build steps in the form of a yaml script, which means that machine understands the script, you can just about configure anything, from code coverage to tests to installation not, to running a *mongod* instance and test results as an **Istanbul** report, for example.

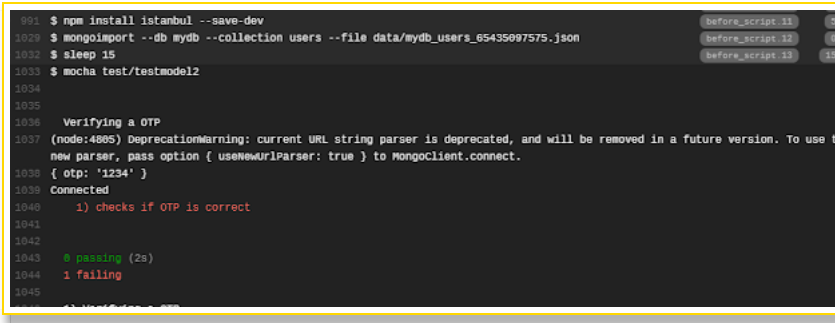
This configuration script is expected by Travis to be part of your code repository to which it is synchronized with. *travis.yml* then Travis will default to **Ruby**.



The travis.yml contains the script that tells the Travis CI container how to run the build.

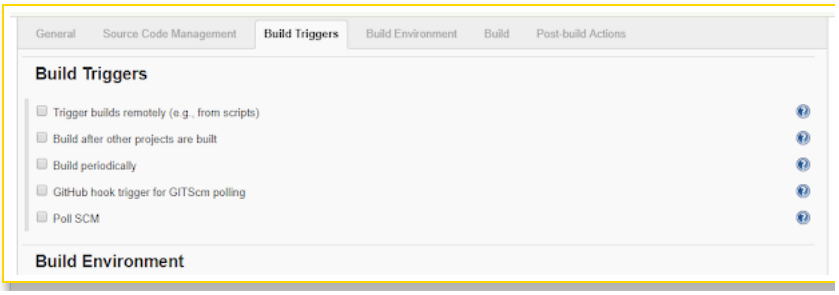


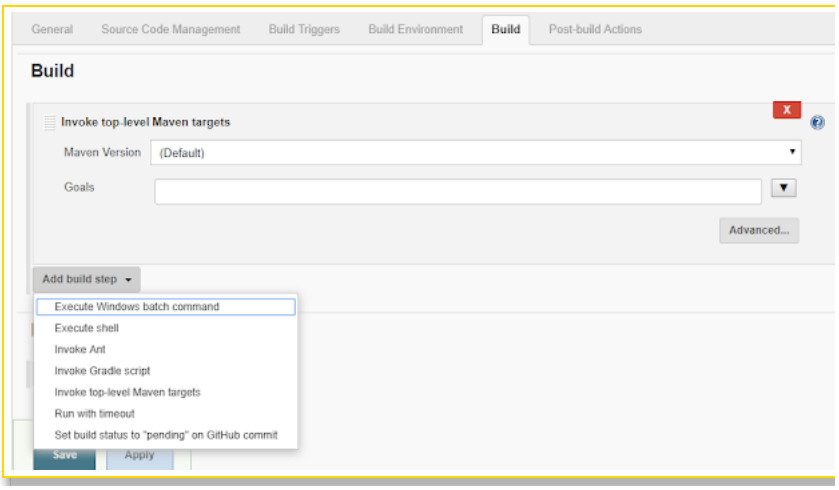


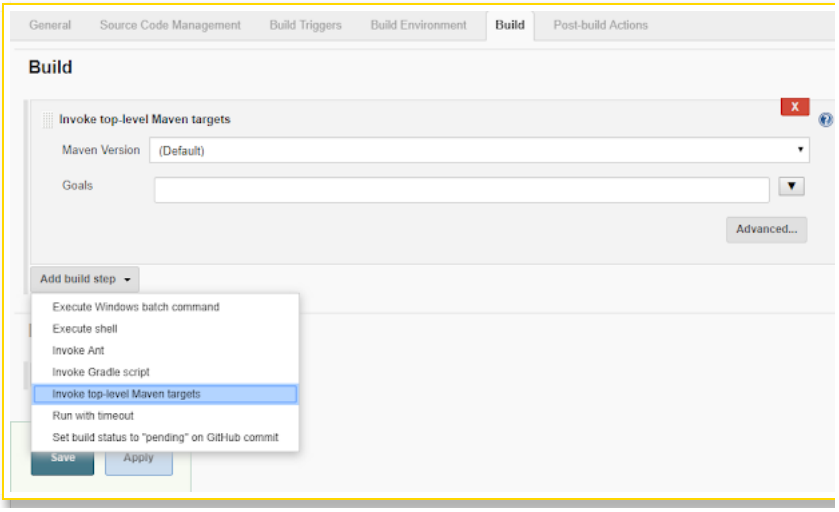


And the travis.yml script does many things like installing **nodejs** packages (which can also be set to ignore with a repo), connecting to a mongodb instance and importing some json data to running a mocha test and providing a c

Jenkins, on the other hand, provides for the build environment, build triggers and build actions and post actions And since Jenkins gels well with **Maven** and the Java environment, its usage requires a good knowledge of the Ja environment like maven, the JDK and the scripts to execute the build.







But unlike *Travis*, which is more suited for open source projects, *Jenkins* is a more specialized CI tool.

The commonality in all DevOps tools is in the integration of a CI tool with a SCM repository and with a Cloud stor
enable Continuous Deployment (CD), the usage, though, will differ as per the depth in the software development

Below are some important questions that could serve as a checklist when deciding on what kind of a DevOps envi
setup for your organization/team?

1. Does the team wish to map its BDD or TDD into an automated environment?
2. Is there an automated build that is wired with the testing framework and the code coverage tool ?
3. Are deployments to the cloud monitored or automated and to what extent is quality ensured in the release t
4. How often is the release to deploy planned?
5. What are the planned pipelines in the build that cannot be ignored or skipped and who are the responsible
6. Does the responsibility or accountability in anyway compromise the agility in the team?
7. What are the recovery mechanisms or policies in place for the CI/CD pipelines?
8. To what extent does the organization want traceability from requirements to deployment?

More on the DevOps Deployment stage in the next post.

Posted by RJV at 18:56 No comments:
Labels: Agile, automated build, automated tests, azure, BDD, code coverage, continuous deployment, Continuous Integration, dev
travis

Home

Subscribe to: Posts (Atom)