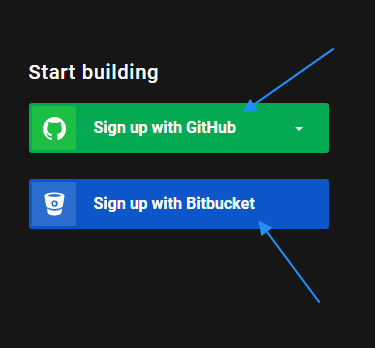
# Circle CI Hosting details

|  |  |  |  |
| --- | --- | --- | --- |
| **Hosting** | | | |
| **Cloud Hosting** | | **Server Hosting** | |
| **Build Environments** | **VCS Provider** | **Build Environments** | **VCS Provider** |
| Docker Linux  macOS  Android  Windows  Self-hosted runners | GitHub  Bitbucket | Build Environments  Docker  Linux  Android  Windows | GitHub  GitHub Enterprise |

In our hands-n session mostly cloud hosting demonstrations will be done.

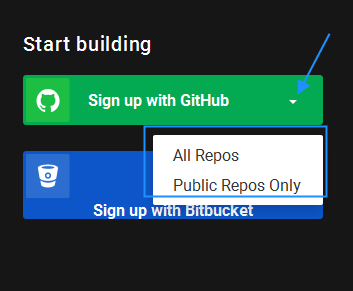
# CircleCI - Sign up & quick start

1. [Click here](https://circleci.com/signup/) to SignUp to CircleCI.
2. To Start building please choose one of VCS (Version control System provider)

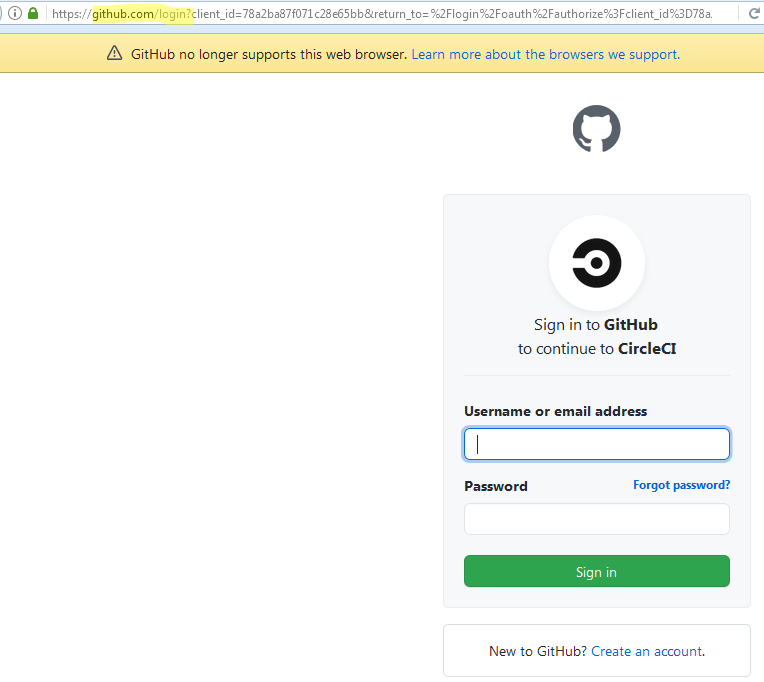


For the next few hands-on instructions GitHub will be leveraged.

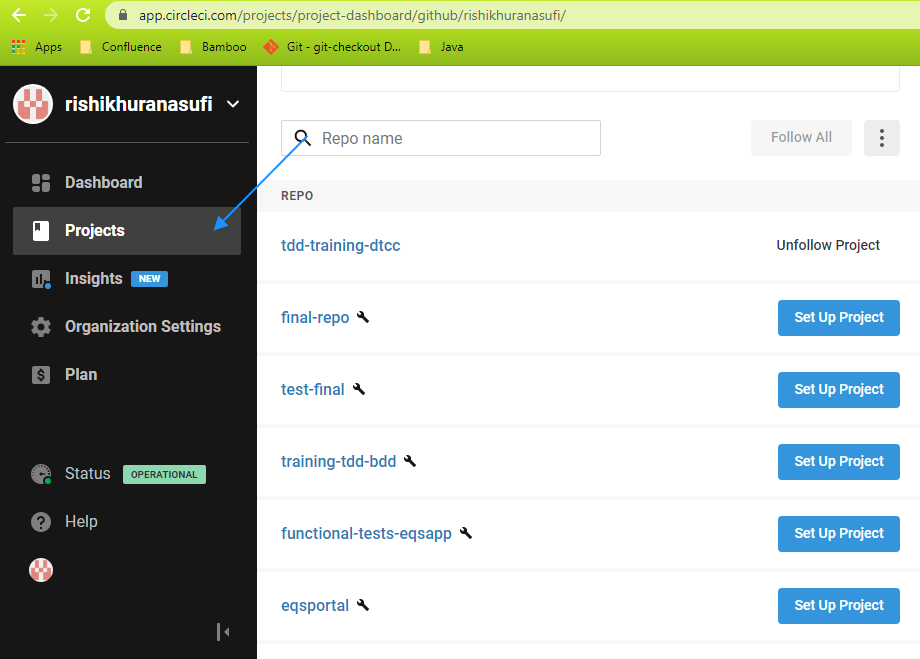
1. On Signup with GitHub link can be seen, on clicking on the these two option will be available :



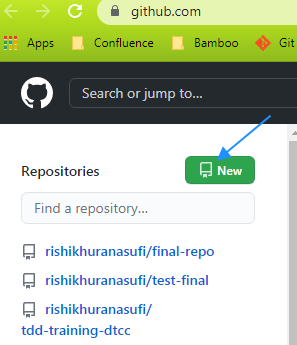
1. On click on any of item it will redirect to Sign In page of GitHub, as depicted below:



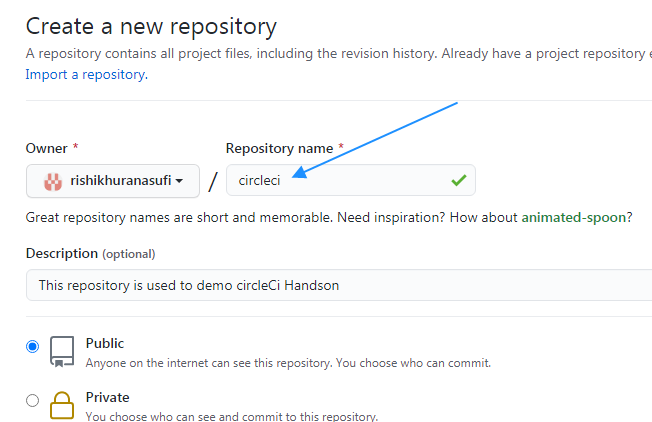
1. Enter GIThub credentials or create a new account with GitHub.
2. It will ask you to accept some permissions that will be allowed to CircleCI. Please accept and proceed.
3. Left panel will have various options available, click on Projects



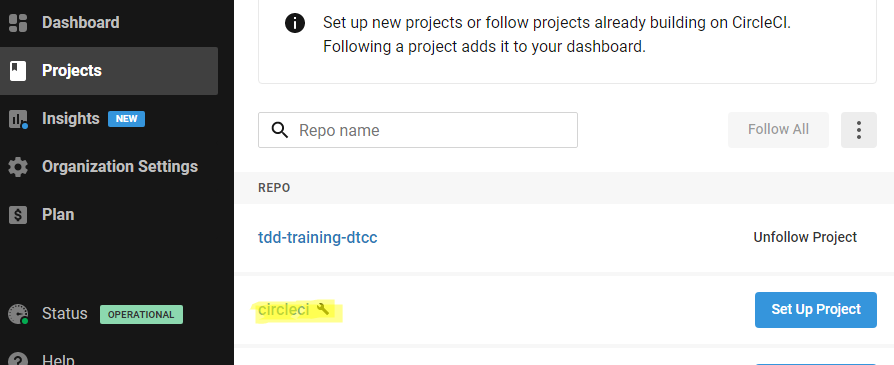
1. It will show up all available Git Repositories in CircleCI as projects.
2. Follow these steps to create one in the GIT repository. If project are not available.
   1. Go to <https://www.github.com>
   2. Click on new at left hand side(Please make sure GitHub is logged in using same credentials which is used while signing up with Circle CI)

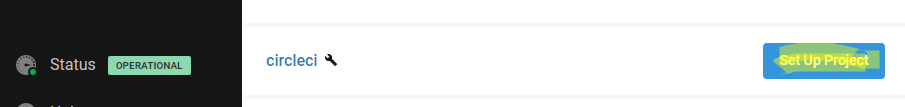


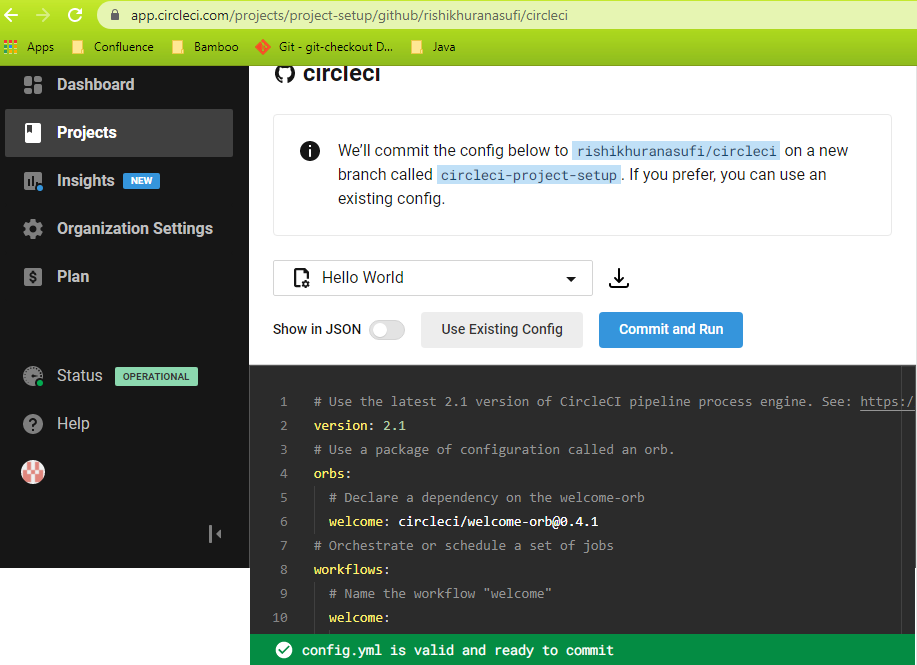
* 1. Enter repository name and click on Create Repository:



* 1. Newly created repository must be available in Circleci Project as highlighted below:



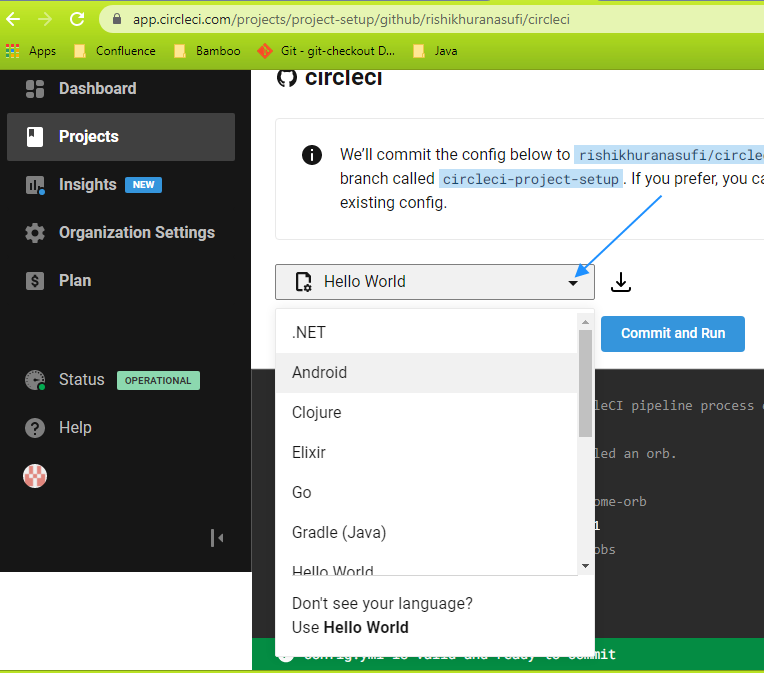
1. Click on Setup Project as highlighted below :
   1. 
2. Page will be redirected to settings of pipeline configuration as depicted below.



1. Pipeline configuration is done in a yaml or json file. This file needs to be committed to the core directory structure of the repository under directory (.circleci).

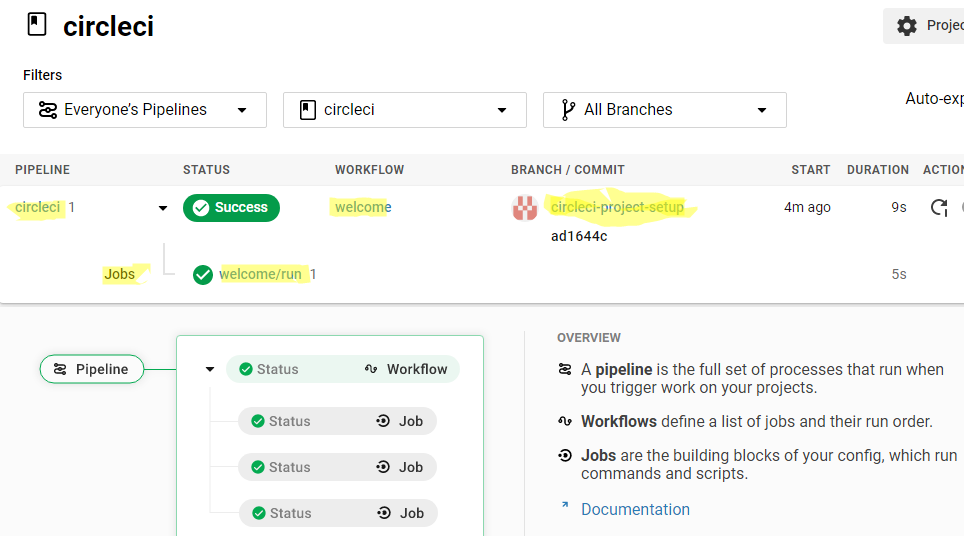
Detailed explanation of YAML / JSON will be provided in the next section.

1. Few default templates are available, click on drop down as shown below to see all available options :

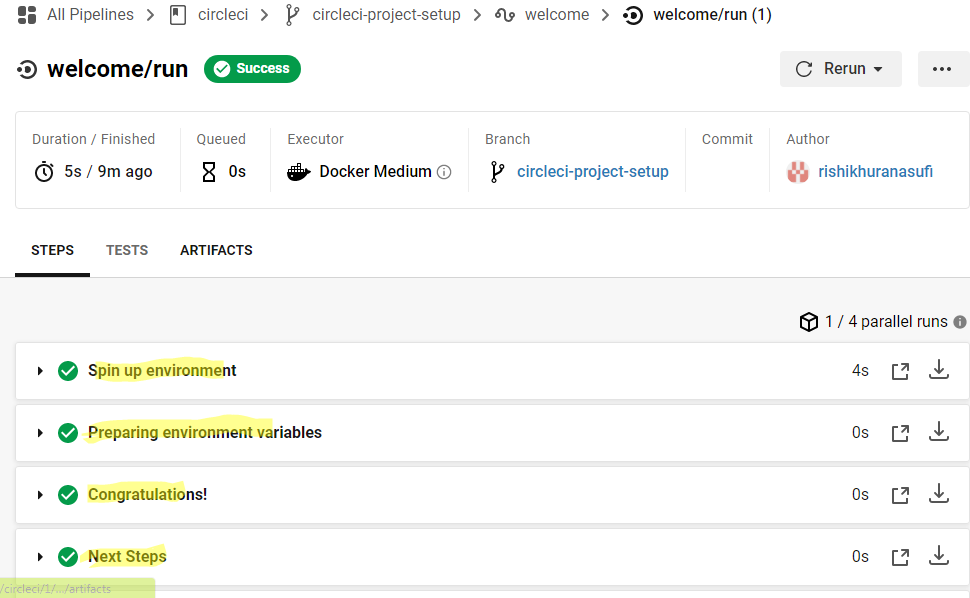


In this section to start with, use Hello-World.

1. Select Hello World from drop down and then click on Commit and Run button (Clicking on this button will create a new branch with name **circle-ci-project-setup**).
2. Pipeline will be created as per details specified in the config.yml file pushed in the last step.



1. Click on job Welcome/run to check detail in job of circleci pipeline.



It gives detailed steps, docker environment was spinned up to perform steps, environment variables are prepared and a step congratulating users.

1. So finally the first base version of pipeline is ready. Let's check out the next section on understanding config files (YAML & JSON) to configure pipelines.

# Configuring CircleCI - Pipeline as code using YAML

In this section, will discuss configuration keys that will be used in the config.yml file to configure the pipeline.

To Understand what YAML is, please refer to this [link](https://circleci.com/docs/2.0/writing-yaml/#overview).

The presence of a .circleci/config.yml file in your CircleCI-authorized repository branch indicates that you want to use the 2.x infrastructure(latest one).

Following yaml will be discussed one by one:

* [version](https://circleci.com/docs/2.0/configuration-reference/#version)
* [orbs (requires version: 2.1)](https://circleci.com/docs/2.0/configuration-reference/#orbs-requires-version-21)
* [commands (requires version: 2.1)](https://circleci.com/docs/2.0/configuration-reference/#commands-requires-version-21)
* [parameters (requires version: 2.1)](https://circleci.com/docs/2.0/configuration-reference/#parameters-requires-version-21)
* [executors (requires version: 2.1)](https://circleci.com/docs/2.0/configuration-reference/#executors-requires-version-21)

**Version :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **Required** | **Type** | **Description** |
| **version** | Yes | String | The version field is intended to be used in order to issue warnings for deprecation or breaking changes. |

**Orbs :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **Required** | **Type** | **Description** |
| **orbs** | No | Map | Orbs names can be defined and can be used as reference in YAML file, find below example |

The following example calls an Orb named hello-build that exists in the certified circleci namespace.

*version: 2.1*

*orbs:*

*technotrainer: circleci/hello-build@0.0.5*

*workflows:*

*"Hello Workflow":*

*jobs:*

*- technotrainer/hello-build*

In the above example, *technotrainer* is considered the orbs reference; whereas circleci/hello-build@0.0.5 is the fully-qualified orb reference.

**Commands :**

A command definition defines a sequence of steps as a map to be executed in a job, enabling you to reuse a single command definition across multiple jobs.

|  |  |  |  |
| --- | --- | --- | --- |
| **Key** | **Required** | **Type** | **Description** |
| steps | Yes | Sequence | A sequence of steps run inside the calling job of the command. |
| parameters | No | Map | A map of parameter keys. |
| description | No | String | A string that describes the purpose of the command. |

For Example :

*commands:*

*sayhello:*

*description: "A very simple command for demonstration purposes"*

*parameters:*

*to:*

*type: string*

*default: "Hello technotrainer participant"*

*steps:*

*- run: echo << parameters.to >>*