Jenkins and Blue Ocean

Lab 6

Create a Pipeline that executes Exercise 2 test automatically.

The pipeline must have at least the following stages:

1. Build
2. Flash LPCXpresso
3. RobotFramework tests

Note that in addition to returning required files to Oma you also need to show that your pipeline works to the instructor.

Setup

In this exercise we run Jenkins and Gogs (a Git server) in containers. On the first run we need to configure both Jenkins and Gogs. After the initial setup the container can be stopped and started at will. The settings are retained in named volumes so there is no need to reconfigure the services.

To keep the URLs consistent both in containers and on the host computer you need to edit /etc/hosts. On linux the file path is `/etc/hosts` (what a surprise!). On windows the file path is: `C:\Windows\System32\drivers\etc\hosts`. Note that you need administrator privileges to edit the file. Add the following lines to /etc/hosts:

```
# Reroute jenkins and gogs to localhost
127.0.0.1 jenkins
127.0.0.1 gogs
```

Windows version of Docker installs Docker compose automatically. If you use Linux you need to install Docker compose manually after installing Docker.

Copy Dockerfile and docker-compose.yml to the same directory. Open command prompt/terminal, go to the directory and execute:

```
docker-compose up
```

Now your containers are up and running and you can perform the initial configuration of Jenkins and Gogs. Following chapters explain how to setup the services.

When you want to stop your containers just execute:

```
docker-compose down
```

You can start the contained in detached mode by executing:

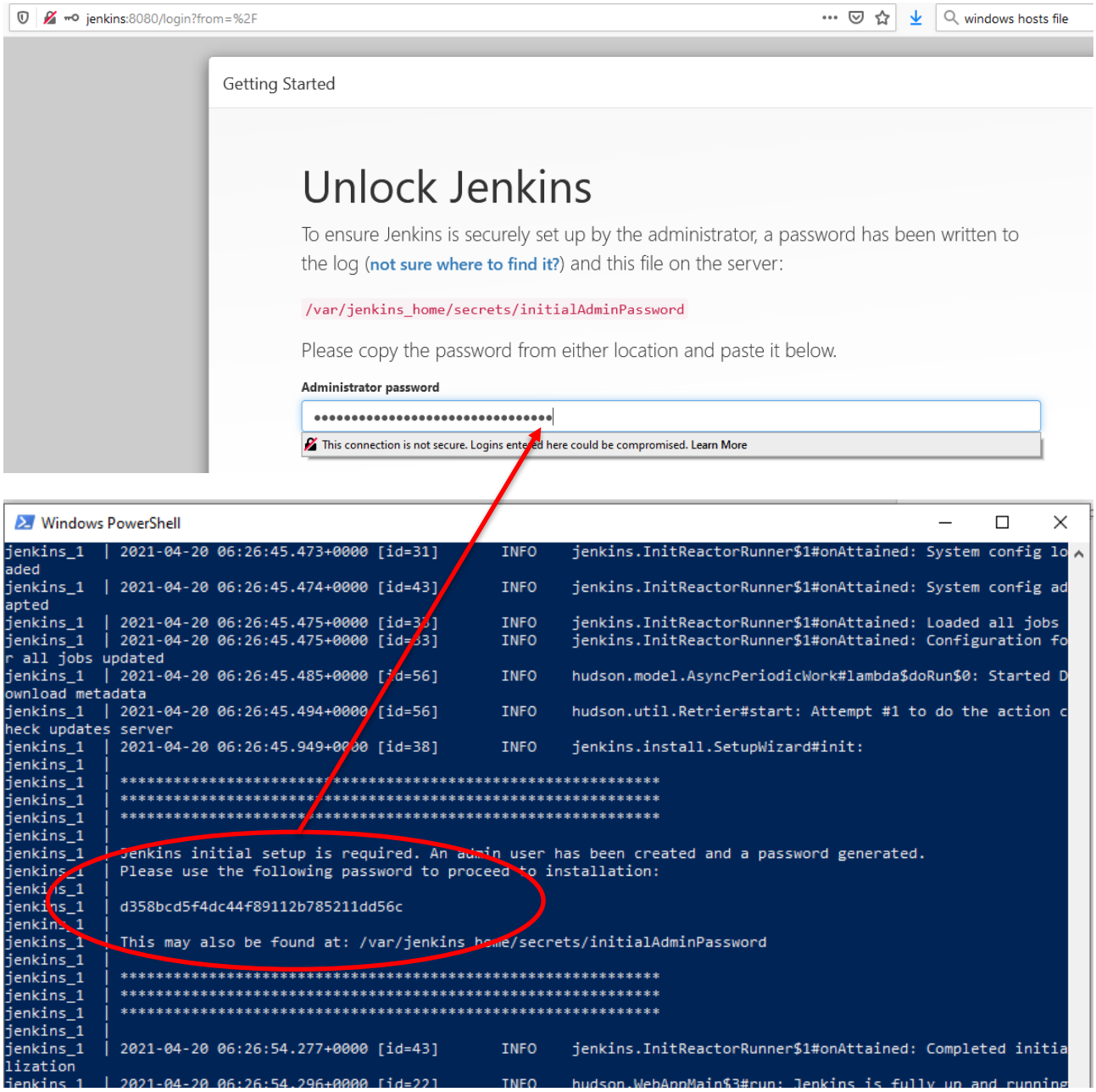
```
docker-compose up --detached
```

In detached mode you won't see the console messages. You don't really need the console messages after the initial setup so once your containers are setup start the containers in detached mode.

Jenkins setup

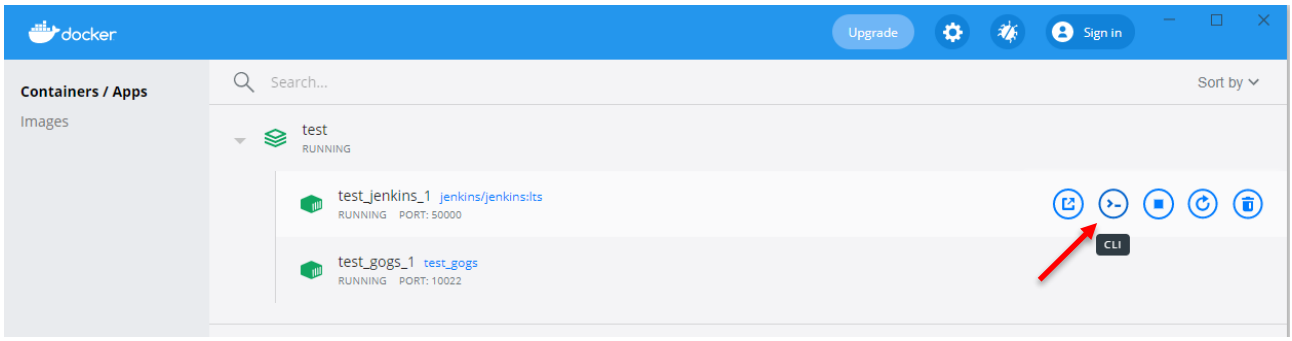
Navigate to `jenkins:8080` in your browser.

You can get the initial password from the logs that are printed on the console:



Note that the browser will complain about insecure connection. Just ignore the warning because we are actually connecting to localhost.

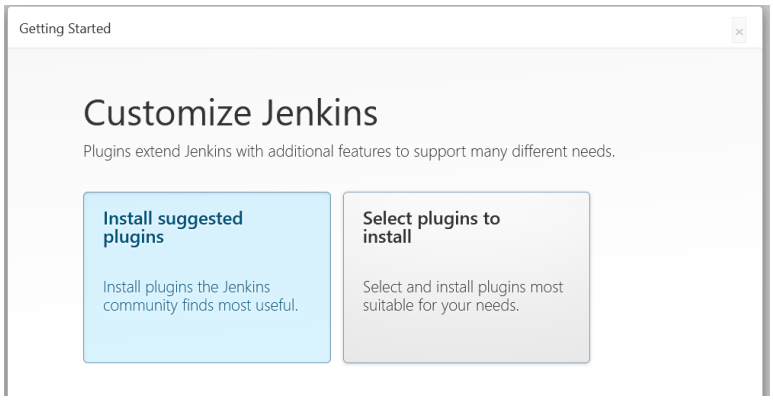
If you can’t find the initial admin password from the logs you can run a console in the container. See below.



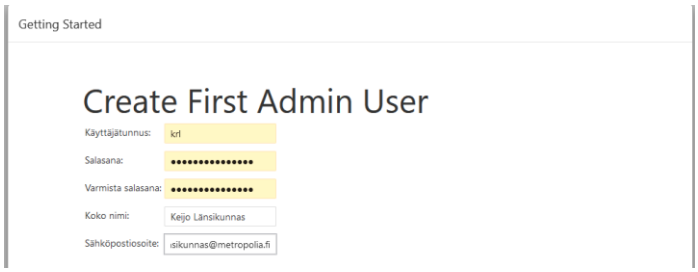
Type command in the console. Note that there is no tab-completion or command history in the Jenkins containers shell.

```
docker exec -it a2f9bd689bf1595d8bfe82bd1c73aa6428a170e6cdf5b64c190ae9c681eabc6c /bin/sh
$ cat /var/jenkins_home/secrets/initialAdminPassword
```

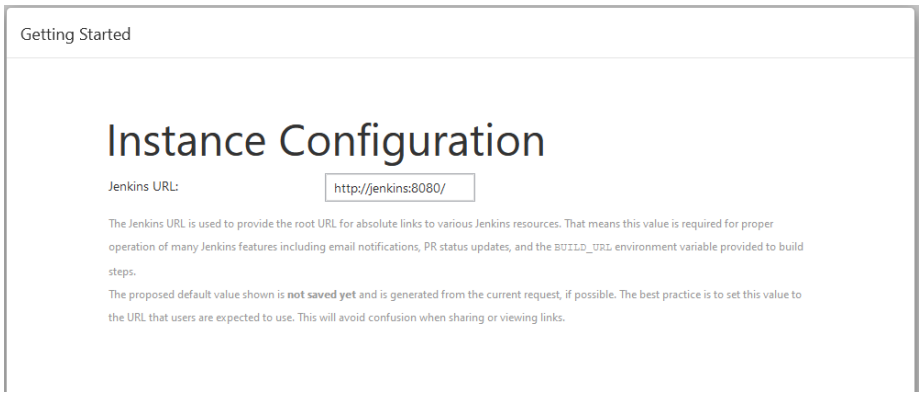
After entering the initial password, you get the following screen. Select install suggested plugins.

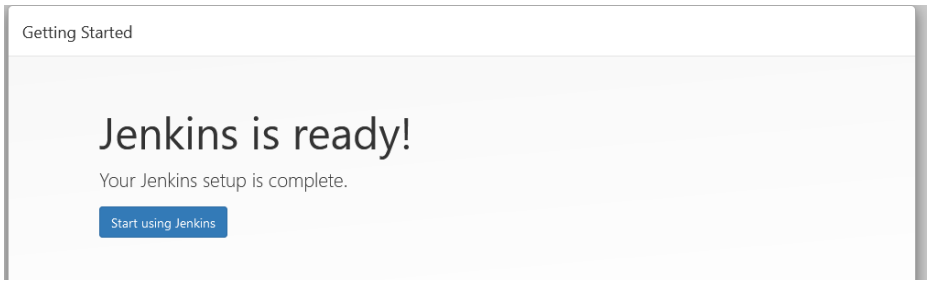


After the plugins have been installed you need to create the first admin user. It is important to keep this password safe because the initial password is a onetime password that will not work after this stage.

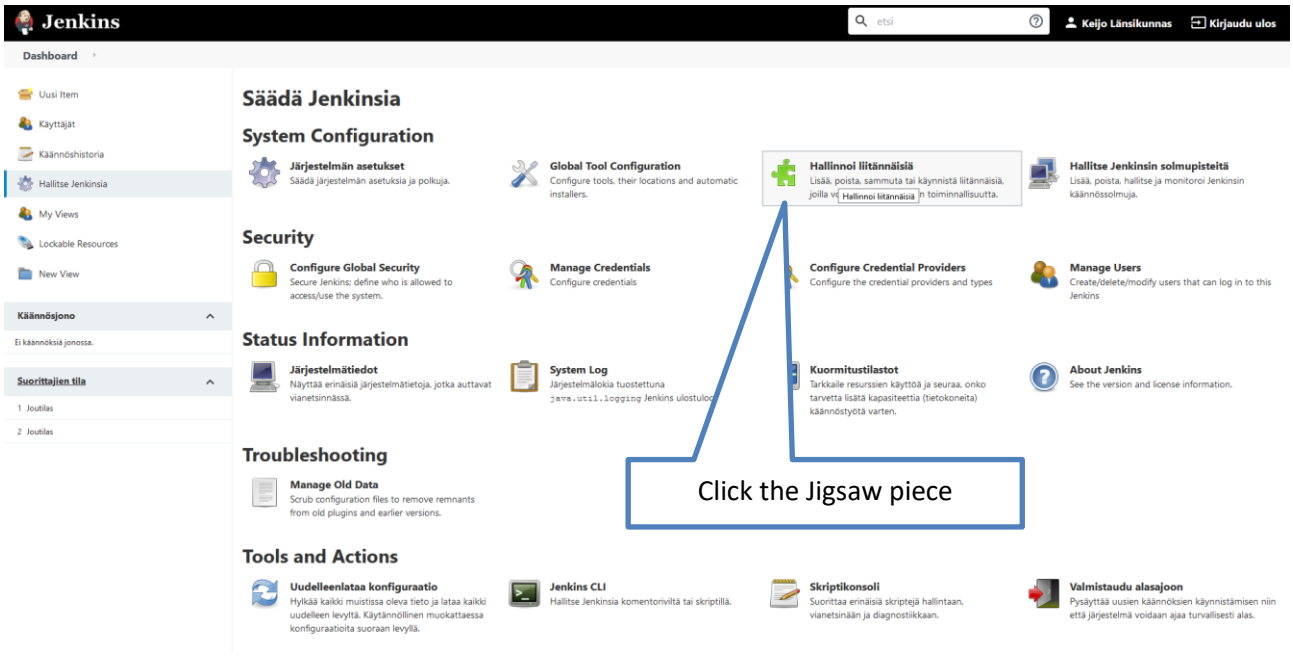


After creating Admin user, you need to configure the URL used to access Jenkins. Accept the default URL.

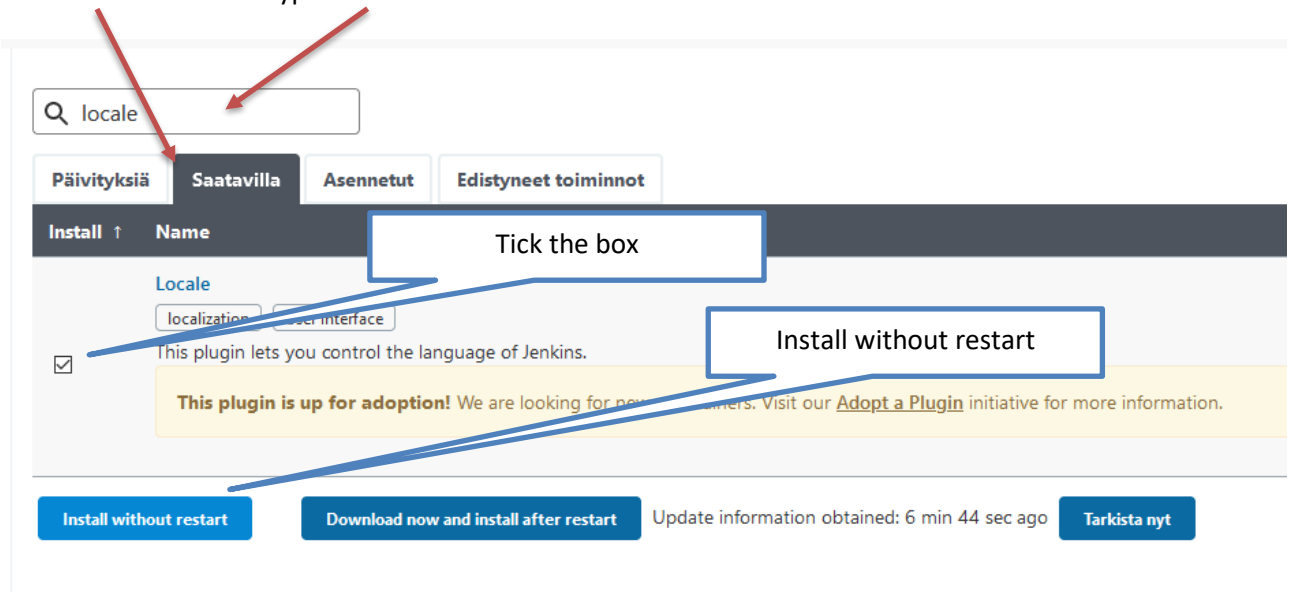




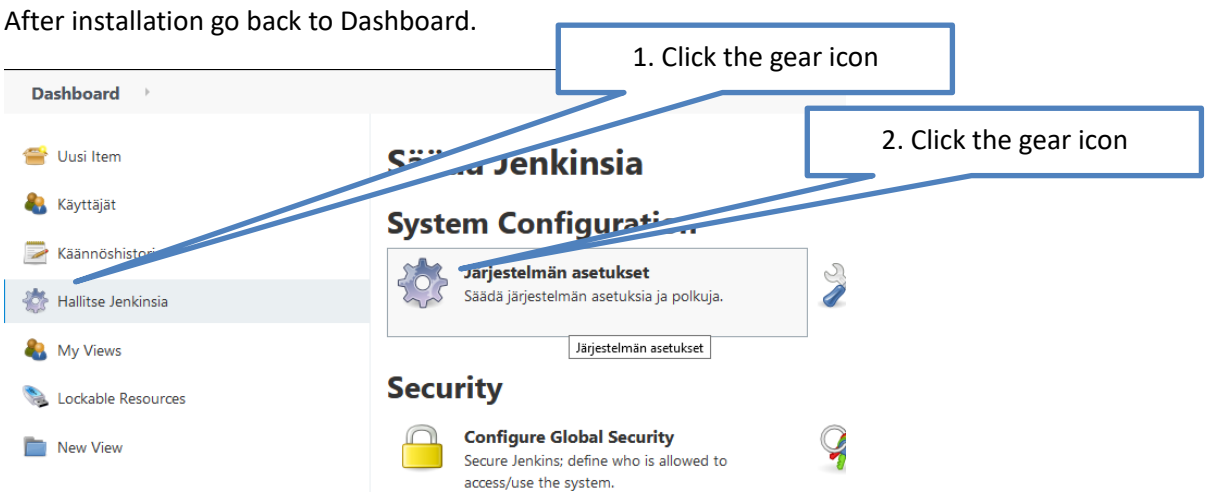
At this point Jenkins is NOT ready. Click `Start using Jenkins` and install some additional plugins according to the instructions below. Depending on your browser settings you may see part of Jenkins UI in English and part in some other language.



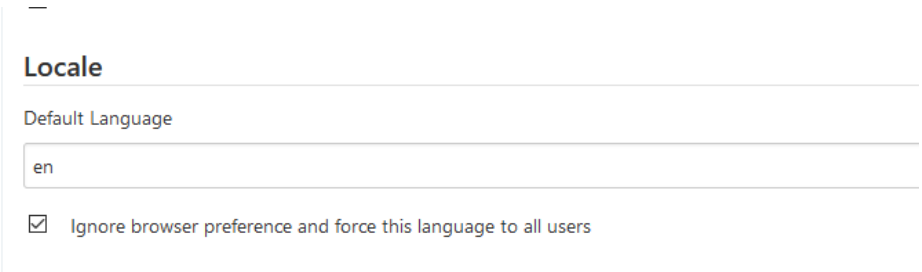
Select second tab and type `locale` in the search box.



After installation go back to Dashboard.

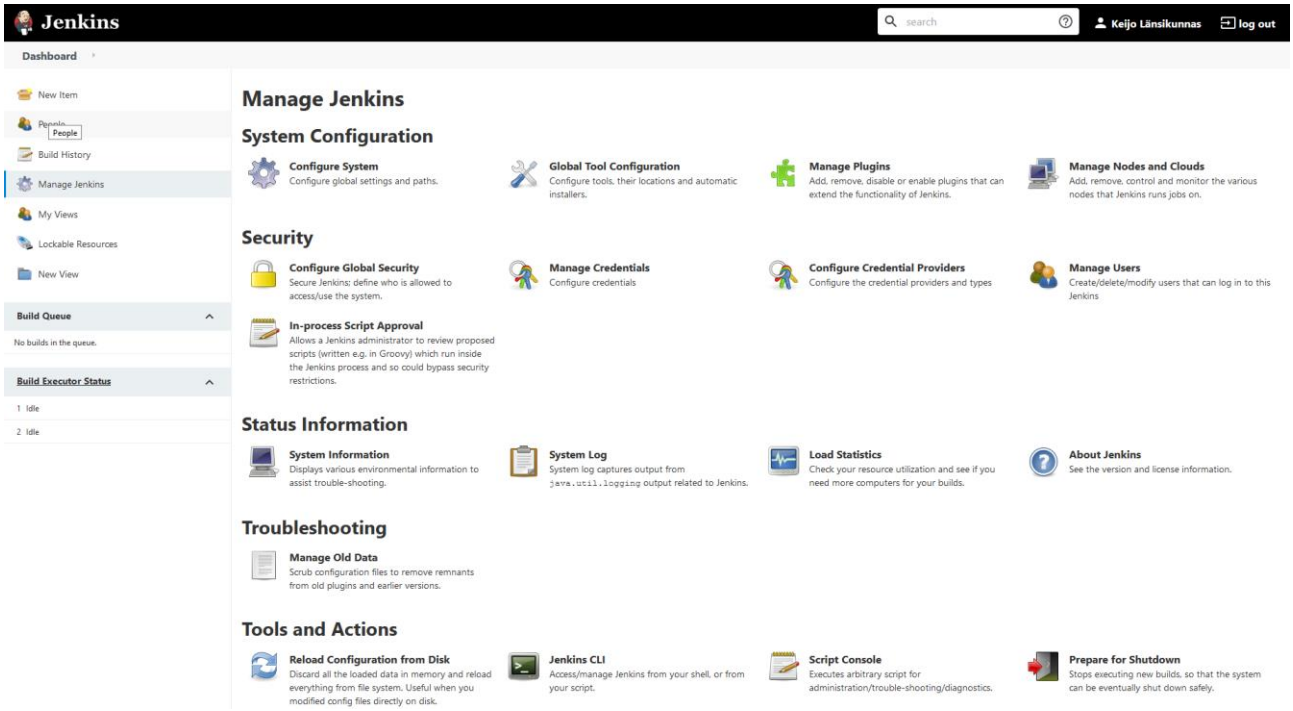


Scroll down to **Locale** and do the following changes:



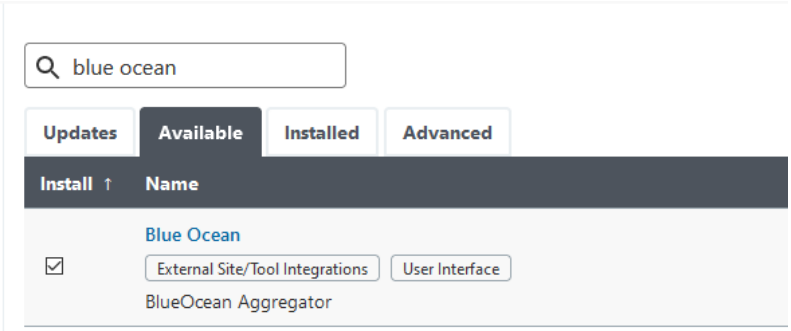
Click **Apply** at the bottom of the page.

Now your Jenkins UI is consistently in English.



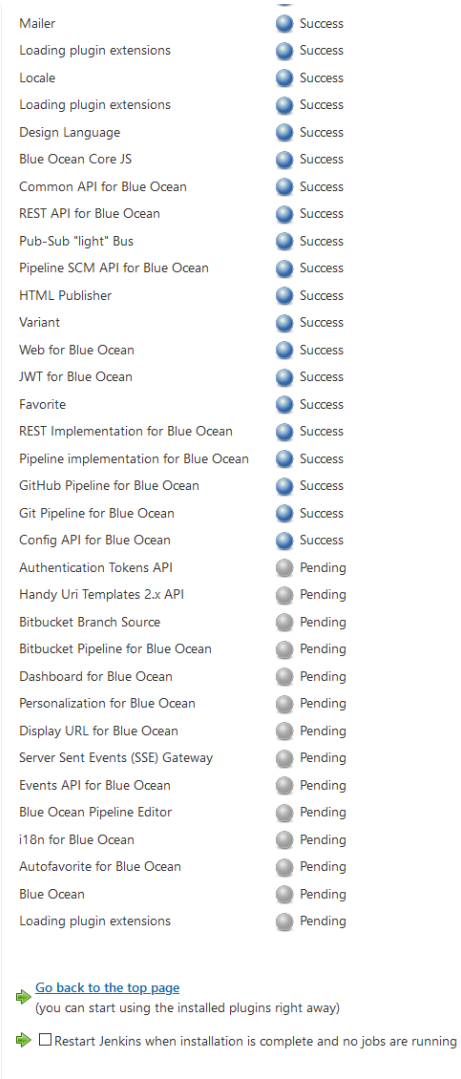
Go to Manage Plugins, select **available** and type `blue ocean` in the search box.

Select: `Blue ocean – BlueOcean Aggregator`



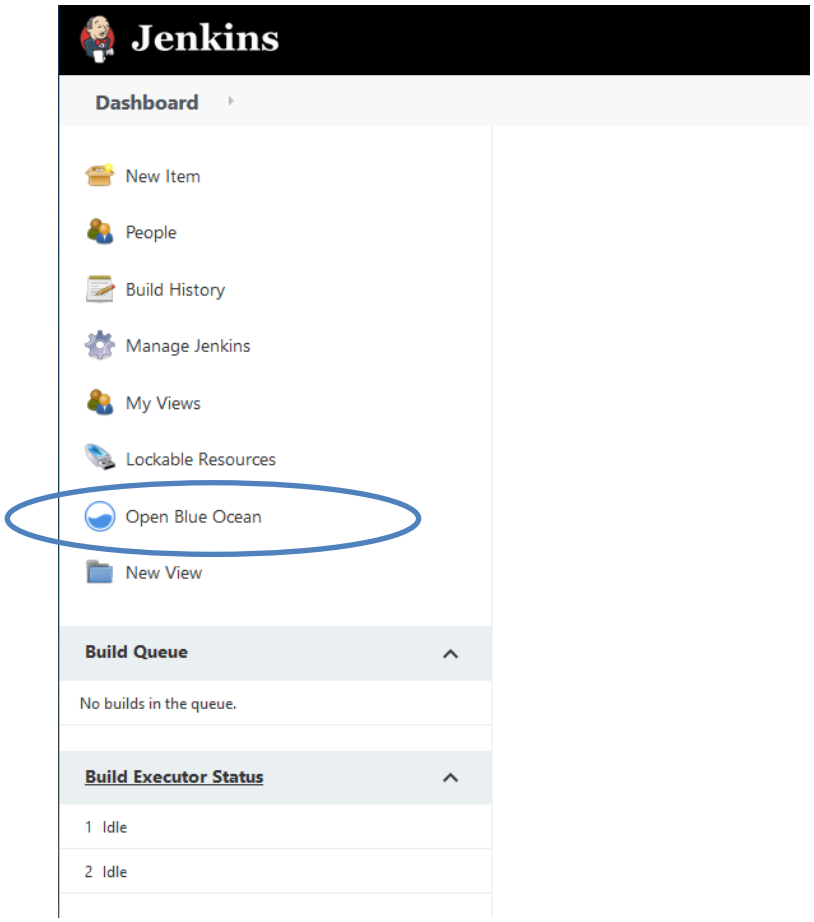
Click `Install without restart` at the bottom of the page.

Wait patiently, the aggregator installs quite a few plugins.



When the installation finishes, you can return to dashboard.

If installation was successful, you will see a new icon in the dashboard.



Jenkins needs a repository to connect to so the next step is to configure Gogs. Gogs is a light-weight Git server that is easy to manage.

Gogs setup

Initial setup is done on a single page. Settings should be as shown below. These settings can't be changed from GUI later! Open a new tab in browser and go to http://gogs:3000

Install Steps For First-time Run

If you're running Gogs inside Docker, please read Guidelines carefully before you change anything in this page!

Database Settings

Gogs requires MySQL, PostgreSQL, SQLite3 or TiDB.

Database Type * SQLite3

Path * data/gogs.db

The file path of SQLite3 or TiDB database.
Please use absolute path when you start as service.

Select SQLite3. Don't change the path

Application General Settings

Application Name * Gogs

Put your organisation name here huge and loud!

Repository Root Path * /data/git/gogs-repositories

All Git remote repositories will be saved to this directory.

Run User * git

The user must have access to Repository Root Path and run Gogs.

Domain * gogs

Set Domain to gogs

SSH Port * 10022

Set SSH Port to 10022

Port number which your SSH server is using, leave it empty to disable SSH feature.

☐ **Use Builtin SSH Server**

HTTP Port * 3000

Port number which application will listen on.

Application URL * http://gogs:3000/

Set URL to: `http://gogs:3000/`

This affects HTTP/HTTPS clone URL and somewhere in email.

Log Path * /app/gogs/log

Di

☐ **Enable Console Mode**

Optional Settings

Expand and ...

▸ Email Service Settings

▾ Server and Other Services Settings

☐ **Enable Offline Mode**

☐ **Disable Gravatar Service**

☐ **Enable Federated Avatars Lookup**

☐ **Disable Self-registration**

☐ **Enable Captcha**

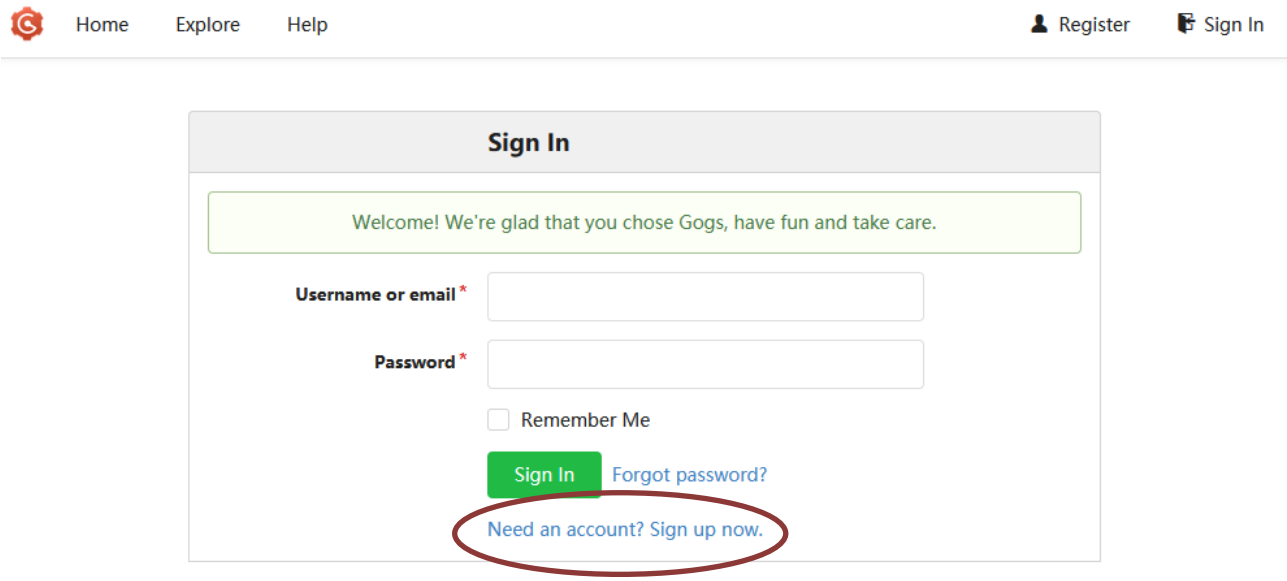
... untick Captcha

☐ **Enable Require Sign In to View Pages**

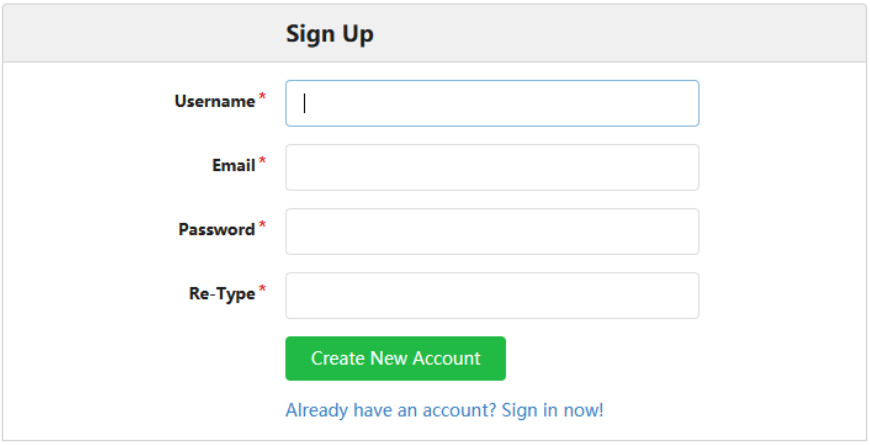
▸ Admin Account Settings

Install Gogs

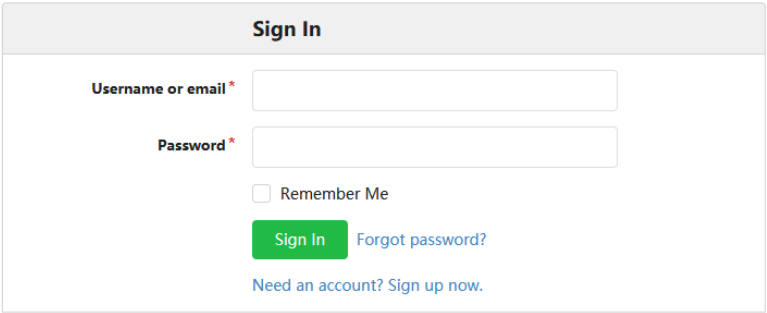
After installation you will be taken to sign in. At this point there is no user or admin account so you need to sign up first.



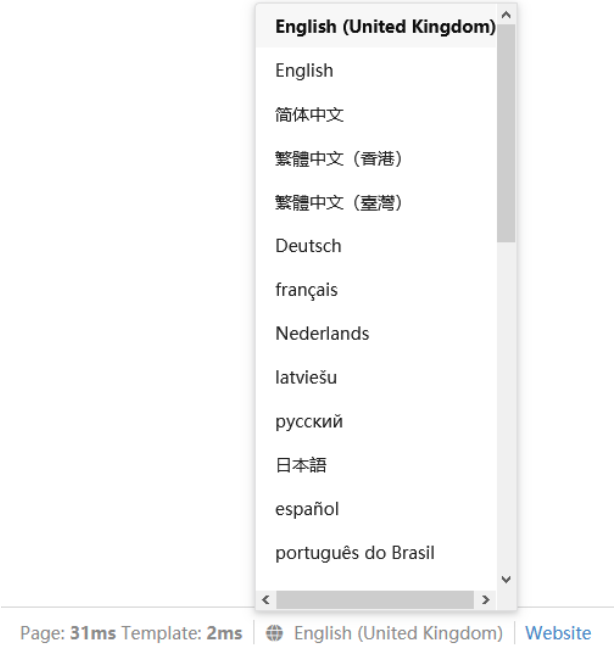
The first user account that you create will automatically get admin privileges.



When the account is created you can log in and start using Gogs.

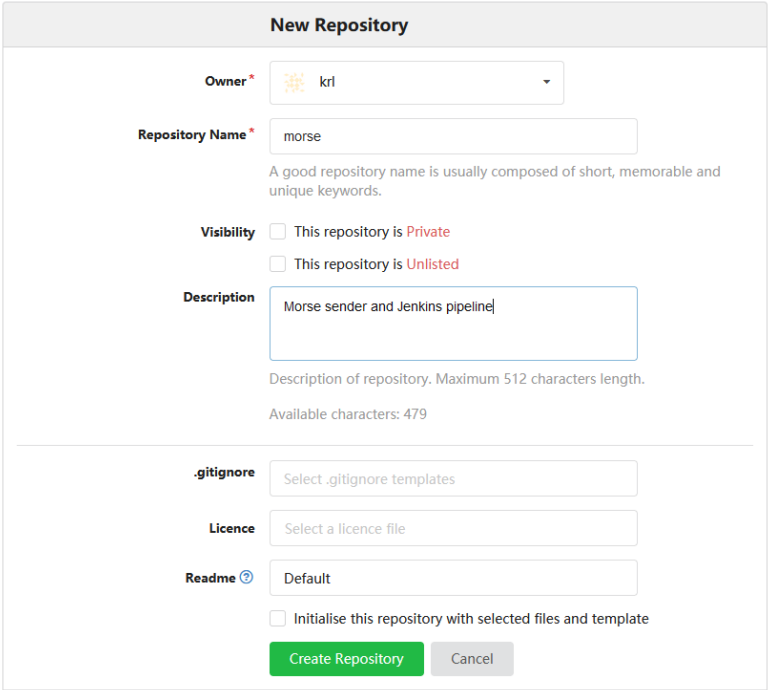


If your UI is not in the language you prefer you can change the language from the menu at the bottom of the page.

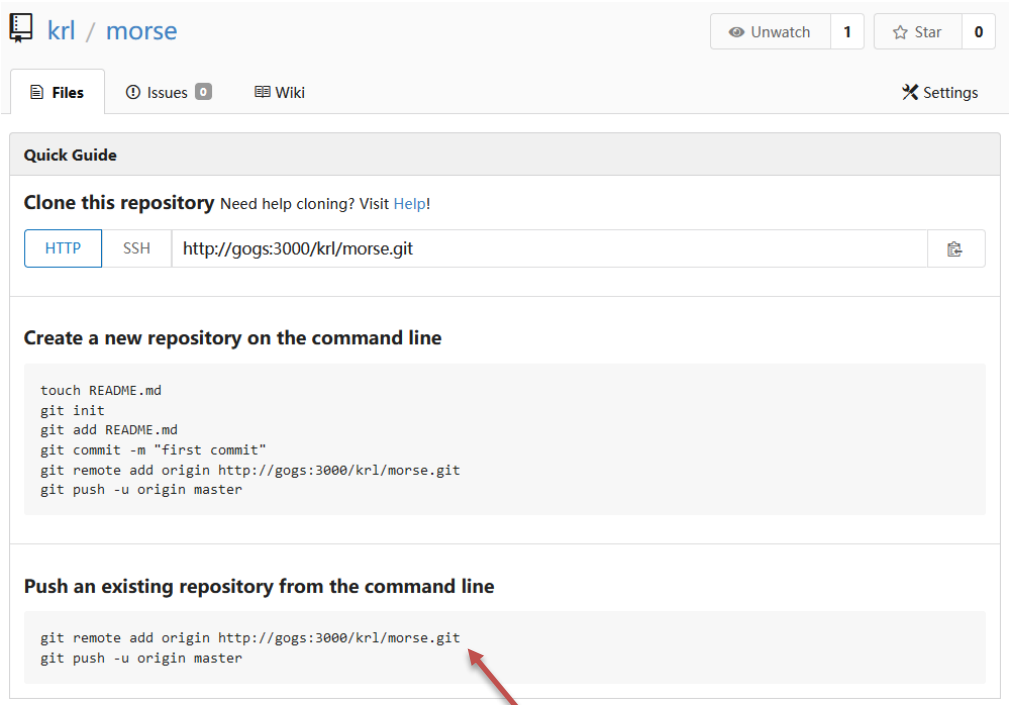


Create a new repository

The following shows an example how to create a repository in Gogs and how to push an existing repository to Gogs. When you create a new repository set the name and description of the repository, and leave the rest at their default values. This will create an empty repository.



After creating the repository, you get the quick guide of how to populate the repo with data. We will use the existing repository option with couple of additional actions.



We start by pulling a repository from Gitlab, or whatever service you use for yours, and then push that repo to Gogs.

Step 1: clone your existing repo

```
$ git clone https://gitlab.metropolia.fi/lansk/morsesender
```

Step 2: cd into the checked out folder and rename the origin. This preserves the link to the original repo in cases where you want to push changes to that repo also.

```
$ cd morsesender
$ git remote rename origin old-origin
```

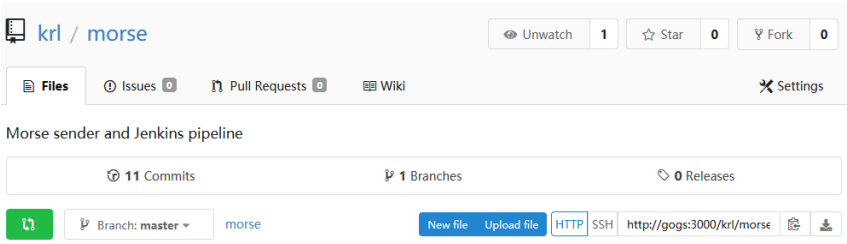
Step 3: set new origin (use steps shown in your repo's quick guide)

```
$ git remote add origin http://gogs:3000/krl/morse.git
```

Step 4: push repo to Gogs (use steps shown in your repo's quick guide)

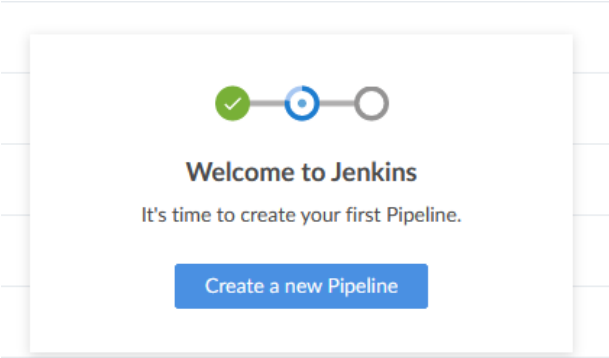
```
$ git push -u origin master
```

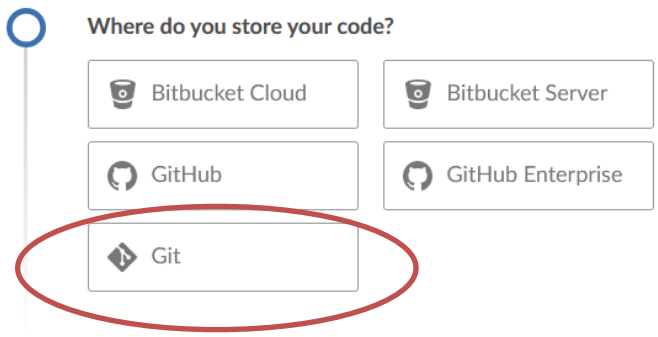
Now you have a repository in Gogs and you are ready start working with Jenkins.



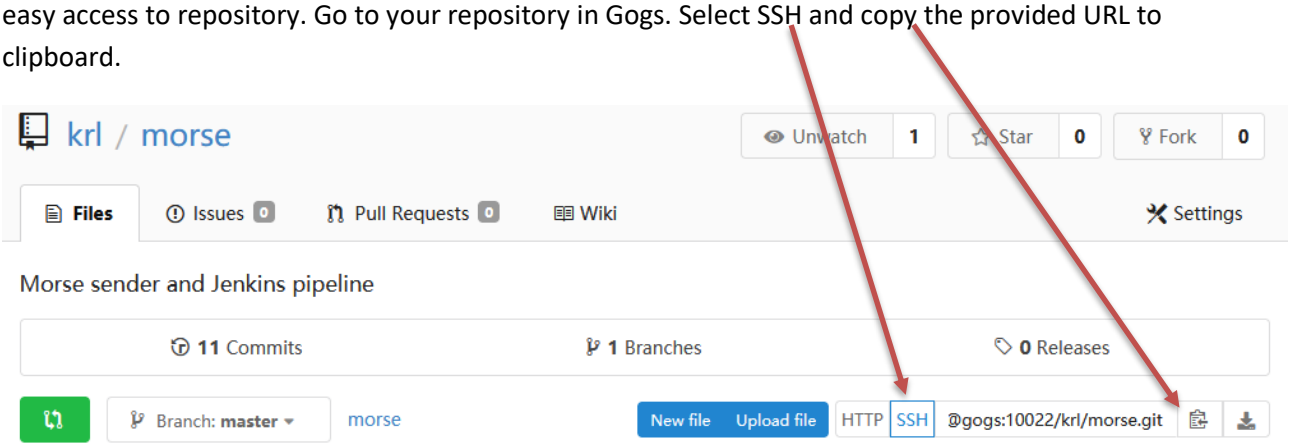
Create your first Pipeline

The following shows how to create a Pipeline with Blue Ocean plugin. In the dashboard click “open Ble Ocean”. On the first invocation you will be taken directly to Pipeline creation wizard.

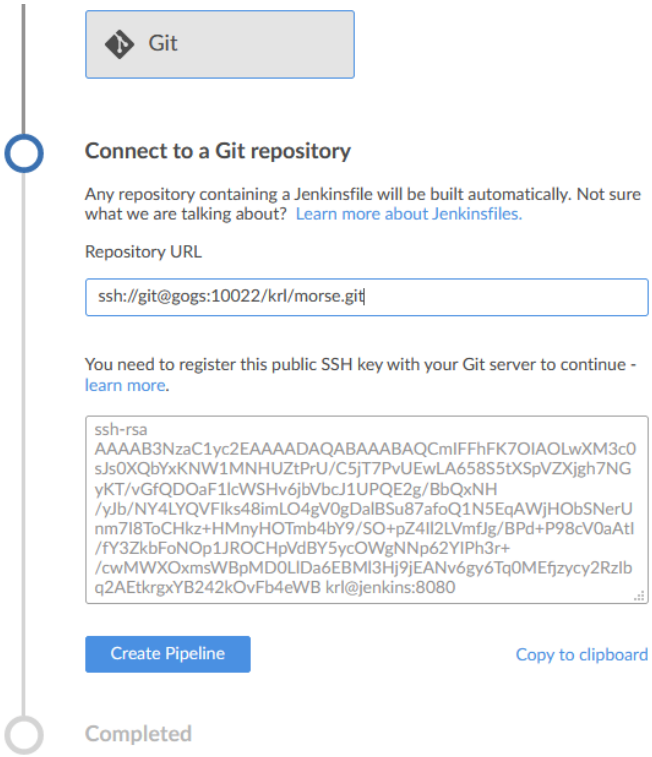




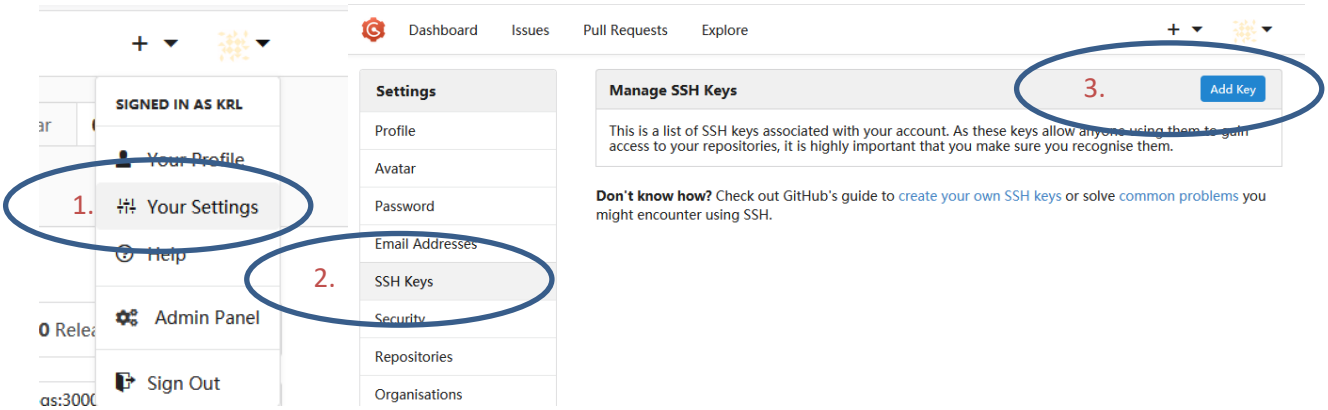
After selecting Git, you need to provide repository URL. Get the URL from Gogs. There are two types of URLs: http and ssh. For Jenkins we use ssh URL because with ssh we can setup public key authentication for easy access to repository. Go to your repository in Gogs. Select SSH and copy the provided URL to clipboard.



When you paste the URL to Repository URL box, Jenkins will automatically create a public key that needs to be installed in Gogs. You need to install the public SSH key to Gogs before you can continue.

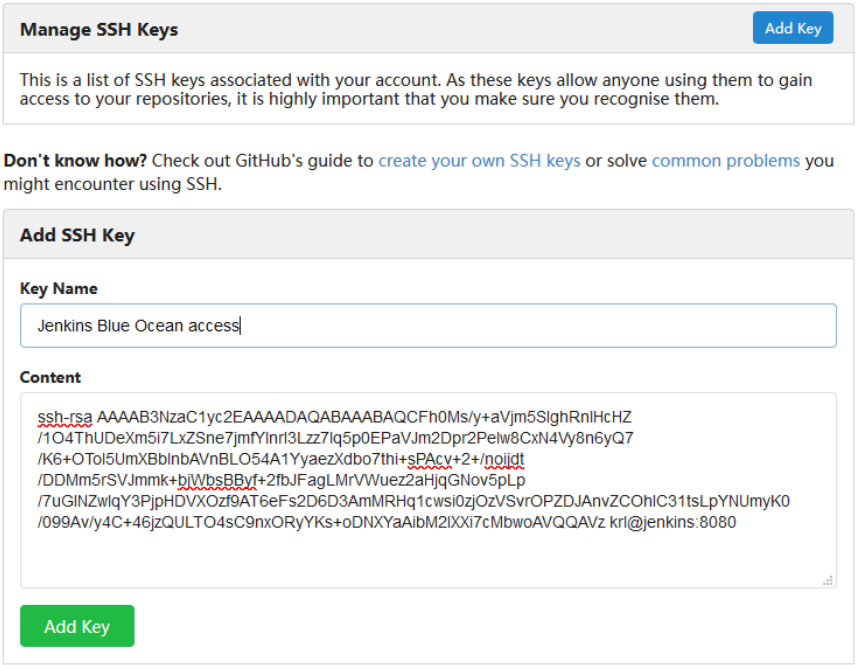


Copy the public key to the clipboard and switch to Gogs. Go to Your settings and select SSH Keys and click Add Key.

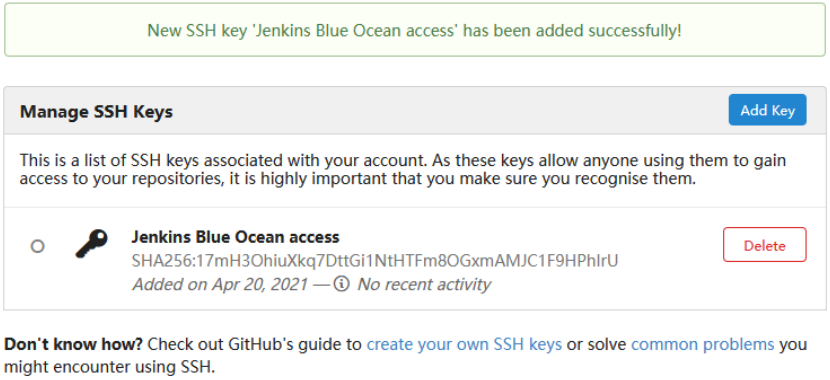


Note: Blue Ocean pipeline editor needs write access to the repository so we need to install the key in user settings.

Name the key and paste the key you copied from Jenkins to Content box.

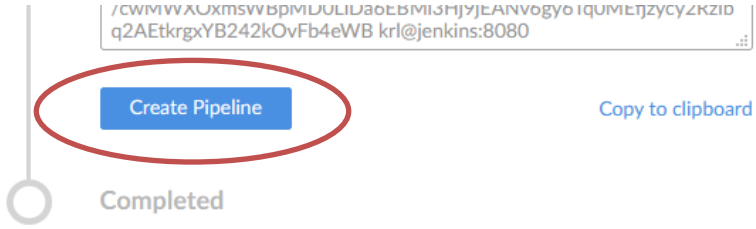


Then click Add Key.

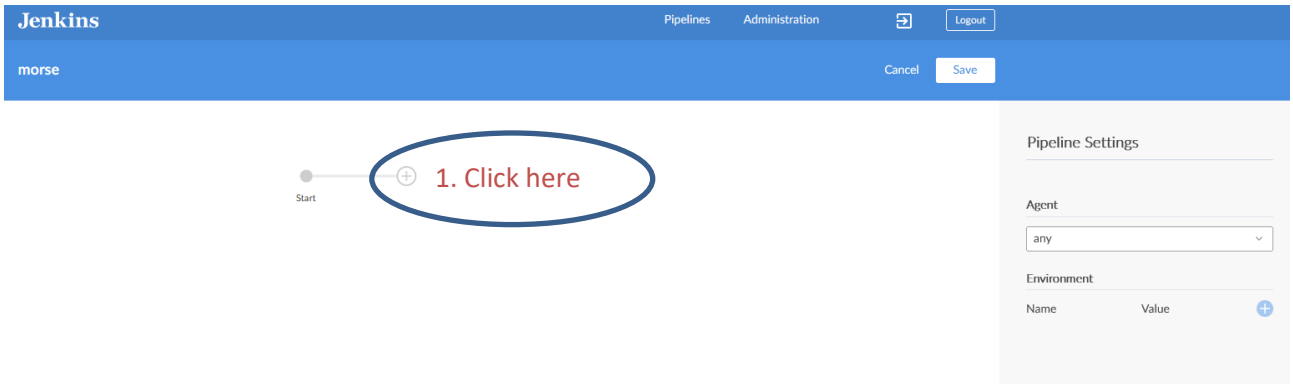


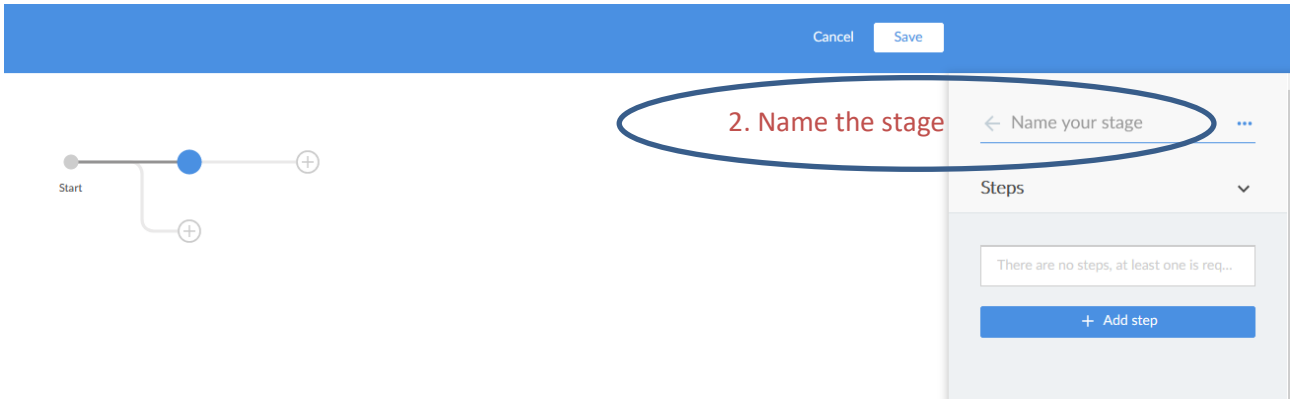
Now you are all set with installing the SSH key.

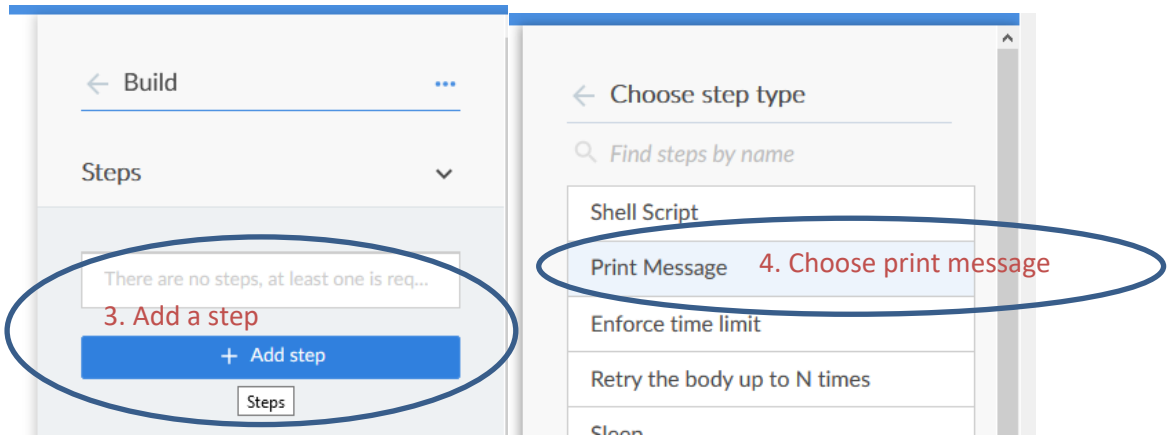
Switch back to Jenkins and click Create Pipeline.

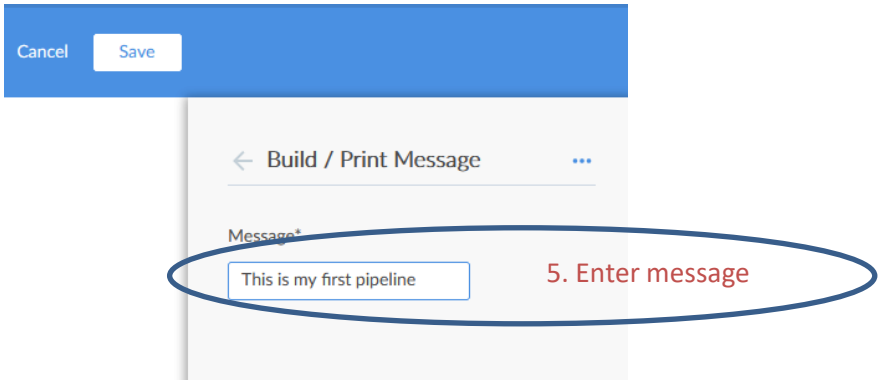


Jenkins verifies that it can access the repository and on success will open the pipeline editor.



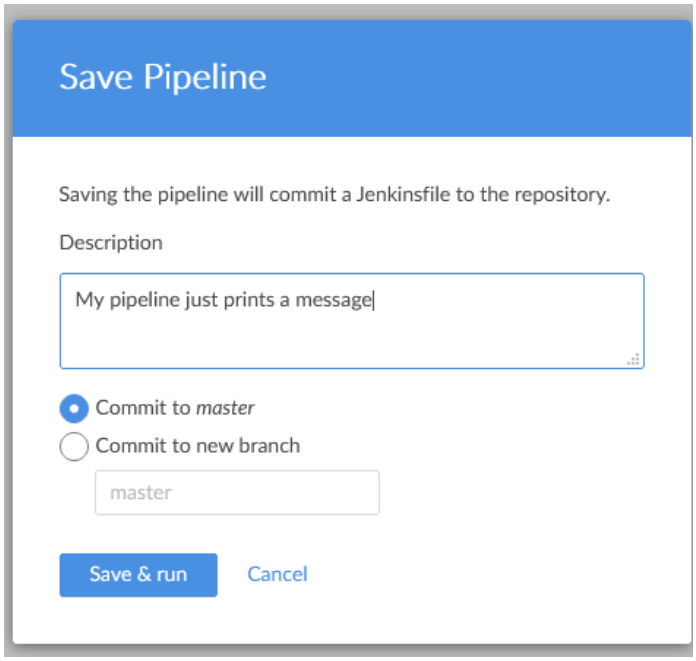




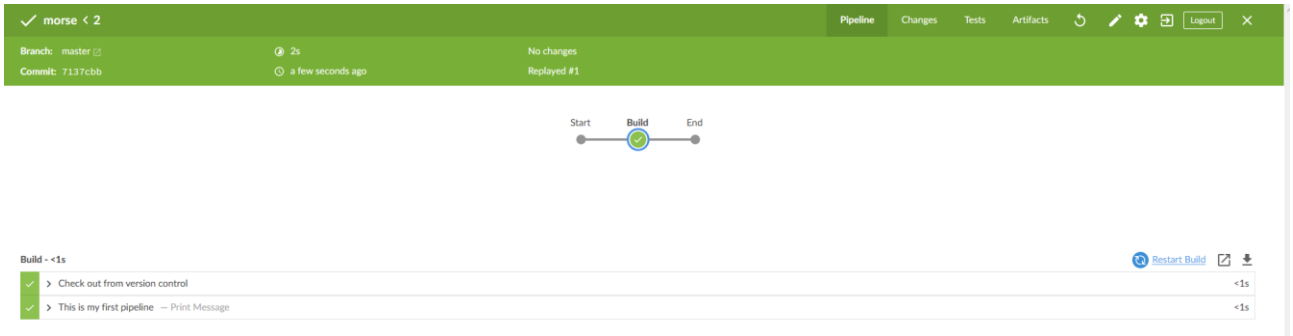


Finally click Save to save your new pipeline.

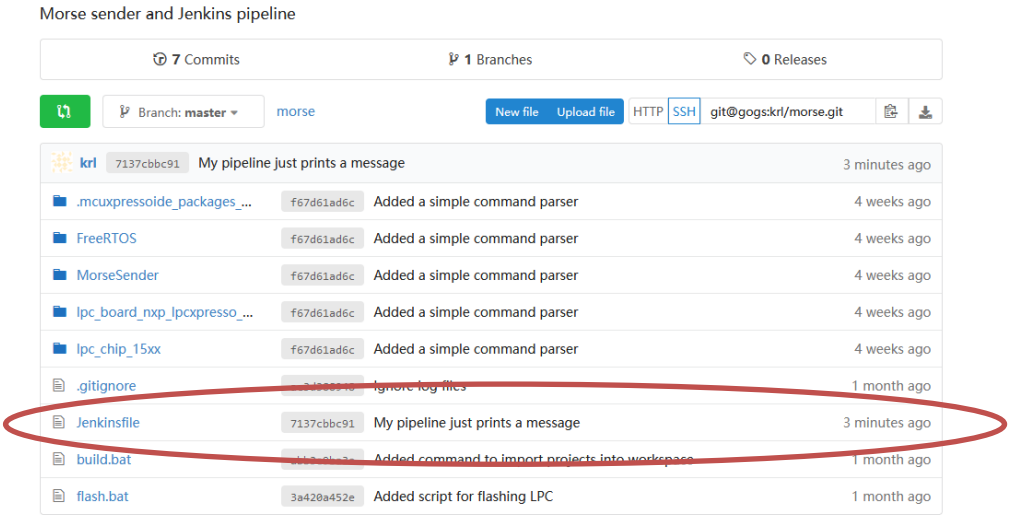
When you save the Pipeline the created (or modified) Jenkinsfile will be committed to your repository. You will be asked to enter commit message.



When you press Save & Run your Jenkinsfile will be saved and then the pipeline is run. Jenkins automatically checks out the repository and runs the Pipeline.



As you can see a new file was added to the repository.



Take a look at the Jenkinsfile

Naturally we can also edit the file manually.

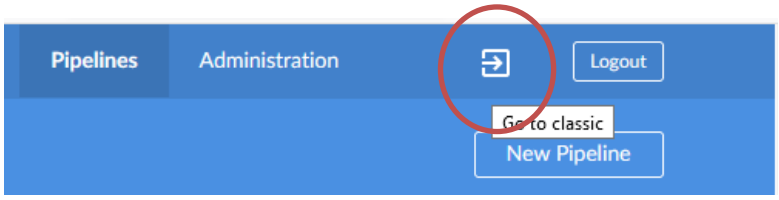
Pull Jenkinsfile from Gogs to local repository:

```
$ git pull

remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 1), reused 0 (delta 0)
Unpacking objects: 100% (3/3), 360 bytes | 22.00 KiB/s, done.
From http://gogs:3000/krl/morse
   f67d61a..7137cbb  master     -> origin/master
Updating f67d61a..7137cbb
Fast-forward
 Jenkinsfile | 11 +++++++++++
 1 file changed, 11 insertions(+)
 create mode 100644 Jenkinsfile
```

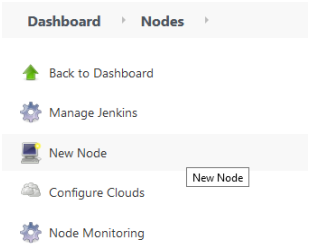
Create a node

If you are in Blue Ocean go first back to “classic” Jenkins.

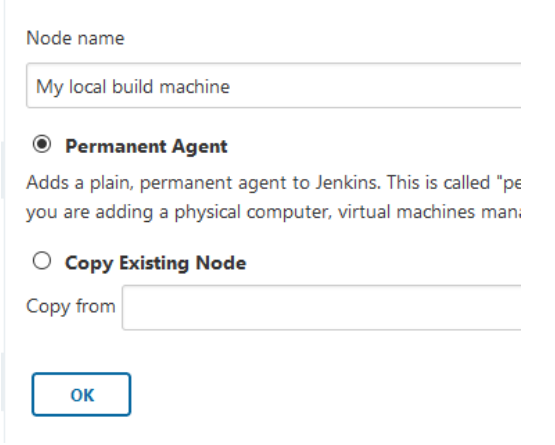


Go to Dashboard → Manage Jenkins → Manage Nodes and Clouds

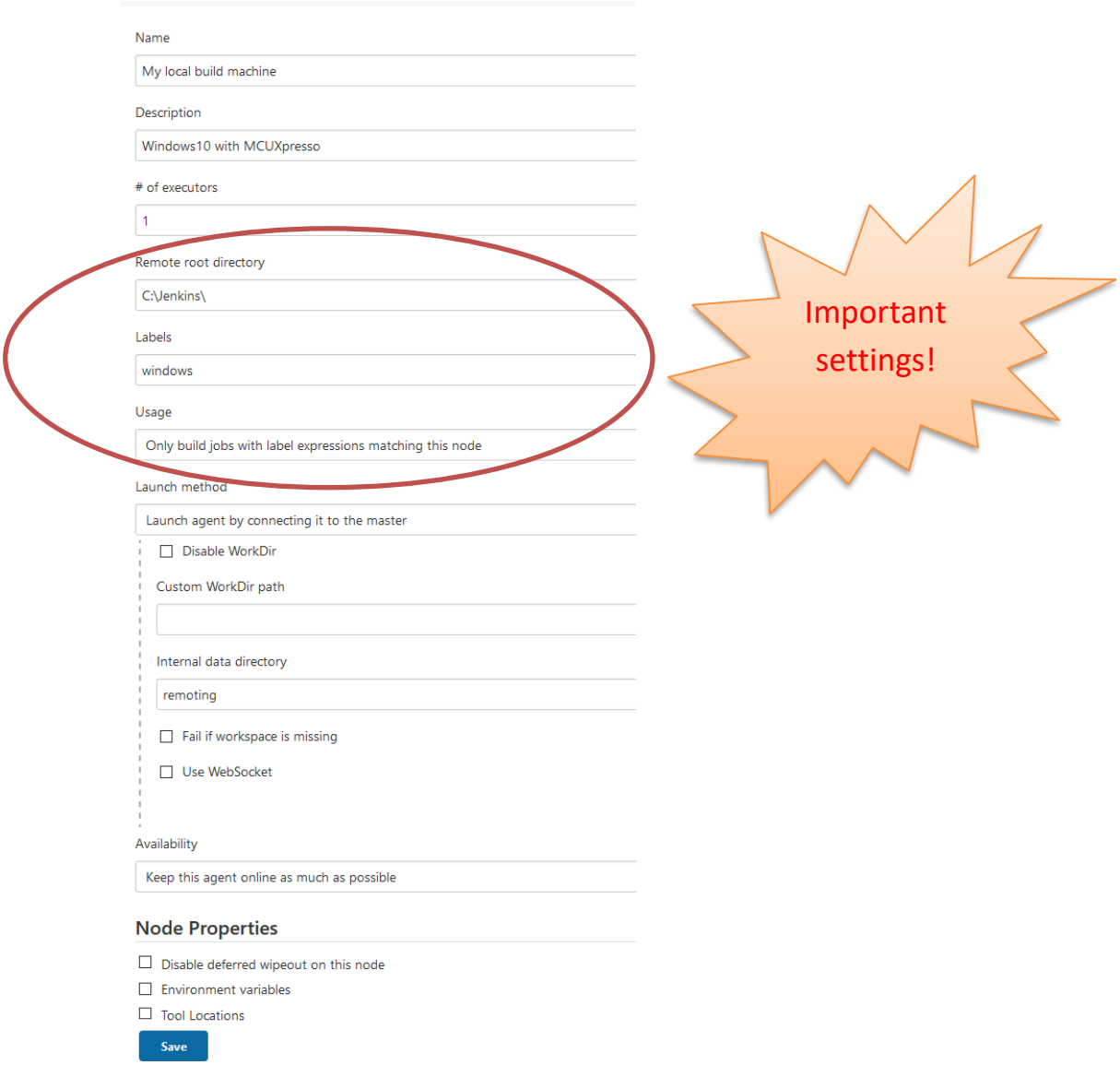
Click New node



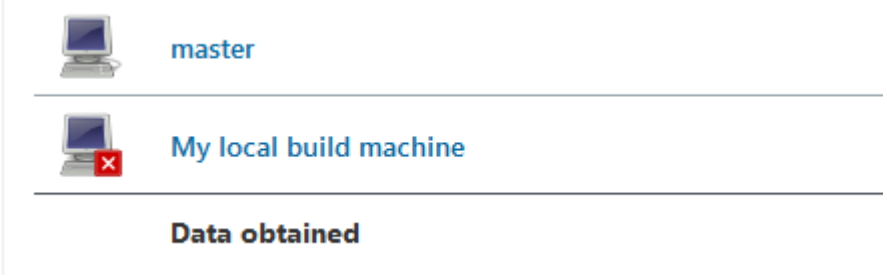
Name the node and select Permanent agent.



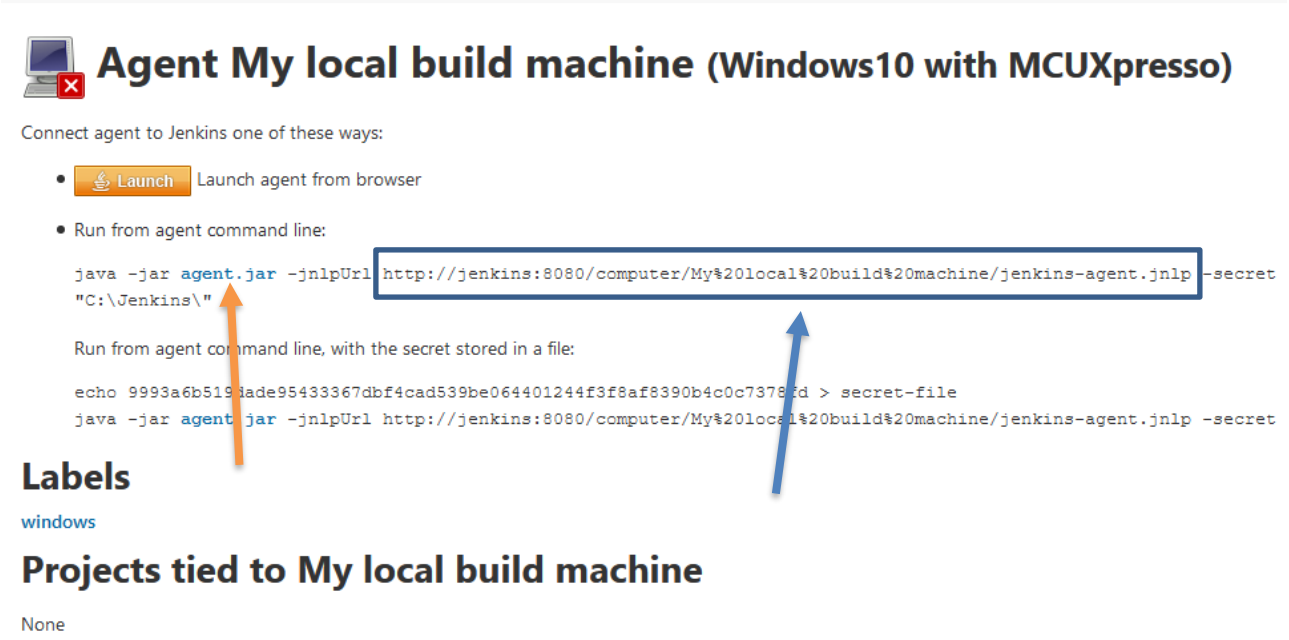
After pressing OK, you will get to configure the agent.



After saving the node, you will return node view and see that your agent is offline.



When you click your new node, you get to start the node. See the screen capture below.



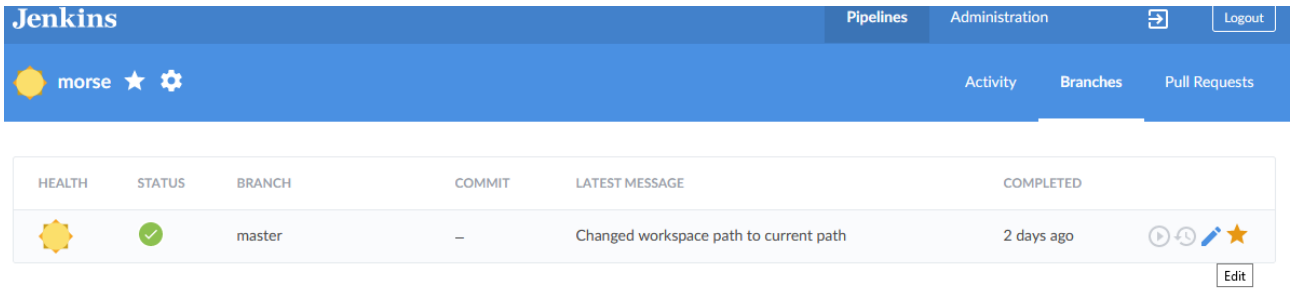
Pressing launch button download a jlnp-file. Firefox refuses to launch jlnp-files automatically so I saved the jlnp-file. Then I launched the jlnp to start the Agent. If you close the agent your node will be offline. When you click on the offline node you get this same start information again. If you save this information as jlnp-file or as a script you can easily start the agent without having to use Jenkins GUI.

In the newer versions of Jenkins the launch button is not shown. You have two options:

1. Copy **the link from the command line** and bookmark it in your browser. Then you can start the agent from your browser.
2. Click on **agent.jar** and save it to a directory, for example c:\Jenkins. Create a batch file with the whole command line and create a shortcut to the batch file on the desktop or what ever your favourite place is.

How to edit a Pipeline

1. Open blue ocean
2. Select Branches (should be default view)
3. Move cursor to end of line and click on pen icon to edit the pipeline.



How to automate build

The example project contains two batch files: build.bat and flash.bat.

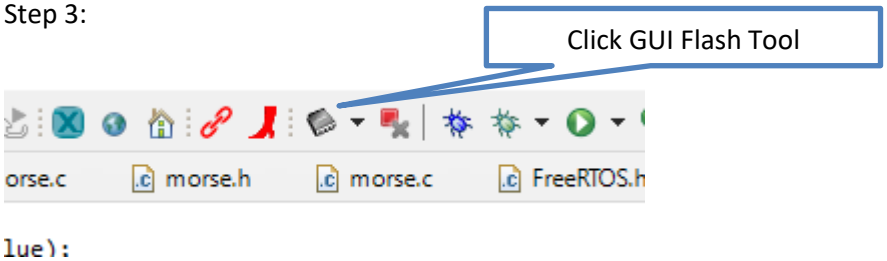
Build.bat performs a headless (no-GUI) build of the morse sender project. Note that you need to edit the batch file to match your current setup (install directory). To test the headless build without Jenkins do a fresh checkout to an empty test directory and run the batch in the checked-out directory.

Flash.bat contains a script to flash the built image to LPC-board. This is also installation specific. You can easily get script for your installation from the gui.

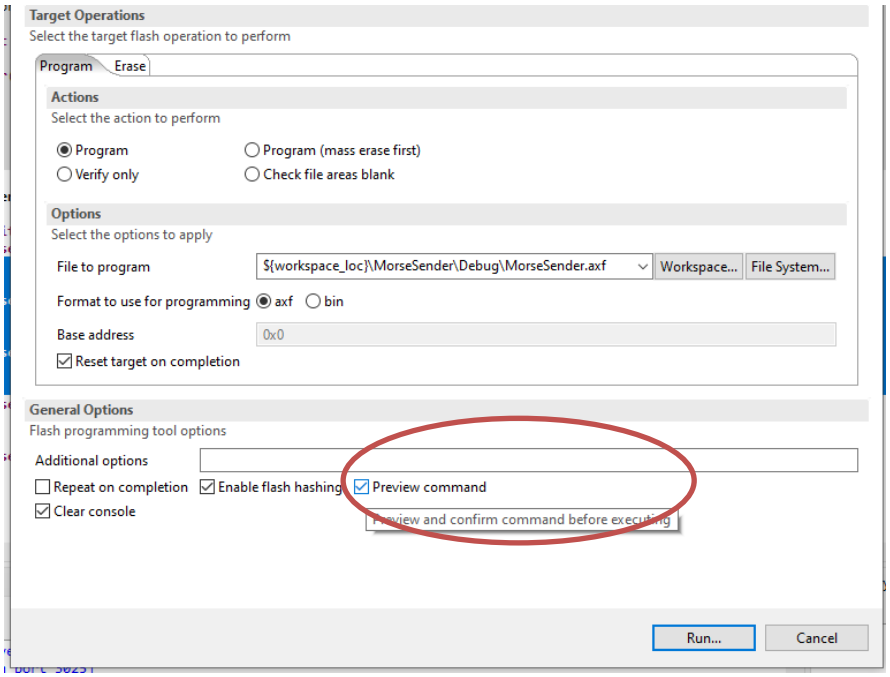
Step 1. Connect LPC to you computer

Step 2: Build morsesender in GUI

Step 3:



Step 4: Check “Preview command” and click “Run…”



Step 5: Select the type of script you want and copy the script to flash.bat.