

Manual



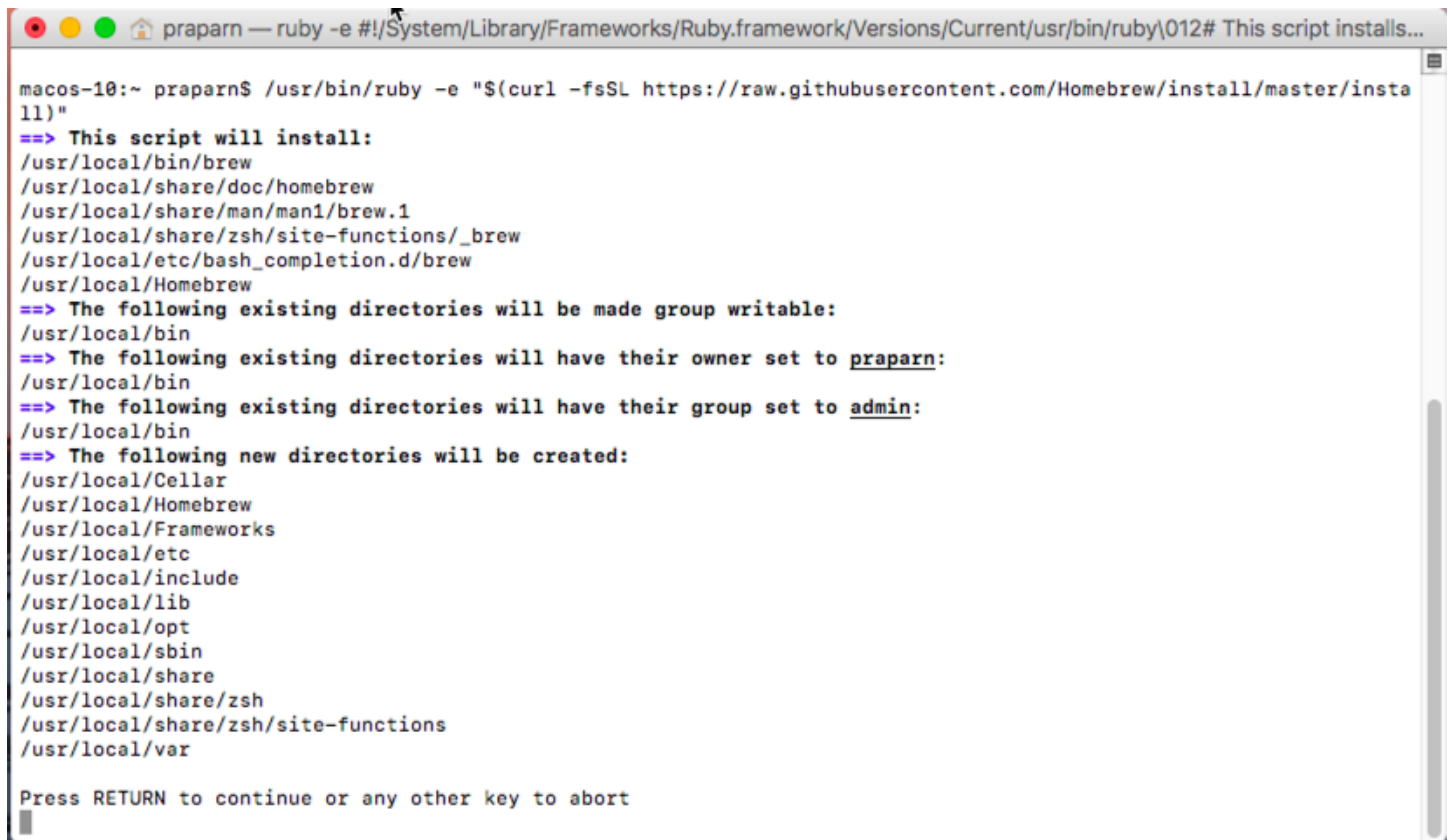
Install Minikube Software Set for
OSX Platform

Prerequisite

Install brew for MACOS

1. Install brew module by command:l.

`/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"` and enter key stroke for continue

A terminal window titled 'praparn — ruby -e #!/System/Library/Frameworks/Ruby.framework/Versions/Current/usr/bin/ruby\012# This script installs...' is shown. The terminal displays the output of the Homebrew installation script. It lists directories to be installed, existing directories to be modified for group writability and ownership, and new directories to be created. The script prompts the user to press RETURN to continue or any other key to abort.

```
macos-10:~ praparn$ /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
==> This script will install:
/usr/local/bin/brew
/usr/local/share/doc/homebrew
/usr/local/share/man/man1/brew.1
/usr/local/share/zsh/site-functions/_brew
/usr/local/etc/bash_completion.d/brew
/usr/local/Homebrew
==> The following existing directories will be made group writable:
/usr/local/bin
==> The following existing directories will have their owner set to praparn:
/usr/local/bin
==> The following existing directories will have their group set to admin:
/usr/local/bin
==> The following new directories will be created:
/usr/local/Cellar
/usr/local/Homebrew
/usr/local/Frameworks
/usr/local/etc
/usr/local/include
/usr/local/lib
/usr/local/opt
/usr/local/sbin
/usr/local/share
/usr/local/share/zsh
/usr/local/share/zsh/site-functions
/usr/local/var

Press RETURN to continue or any other key to abort
```

2. Input password for grant privilege for install and wait until all install process was done.

```
Press RETURN to continue or any other key to abort
==> /usr/bin/sudo /bin/chmod u+rw /usr/local/bin
Password:?
```

```
==> /usr/bin/sudo /bin/chmod g+rw /usr/local/bin
==> /usr/bin/sudo /usr/sbin/chown praparn /usr/local/bin
==> /usr/bin/sudo /usr/bin/chgrp admin /usr/local/bin
==> /usr/bin/sudo /bin/mkdir -p /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
==> /usr/bin/sudo /bin/chmod g+rw /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
==> /usr/bin/sudo /bin/chmod 755 /usr/local/share/zsh /usr/local/share/zsh/site-functions
==> /usr/bin/sudo /usr/sbin/chown praparn /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
==> /usr/bin/sudo /usr/bin/chgrp admin /usr/local/Cellar /usr/local/Homebrew /usr/local/Frameworks /usr/local/etc /usr/local/include /usr/local/lib /usr/local/opt /usr/local/sbin /usr/local/share /usr/local/share/zsh /usr/local/share/zsh/site-functions /usr/local/var
==> /usr/bin/sudo /bin/mkdir -p /Users/praparn/Library/Caches/Homebrew
==> /usr/bin/sudo /bin/chmod g+rw /Users/praparn/Library/Caches/Homebrew
==> /usr/bin/sudo /usr/sbin/chown praparn /Users/praparn/Library/Caches/Homebrew
==> /usr/bin/sudo /bin/mkdir -p /Library/Caches/Homebrew
==> /usr/bin/sudo /bin/chmod g+rw /Library/Caches/Homebrew
==> /usr/bin/sudo /usr/sbin/chown praparn /Library/Caches/Homebrew
==> Searching online for the Command Line Tools
==> /usr/bin/sudo /usr/bin/touch /tmp/.com.apple.dt.CommandLineTools.installondemand.in-progress
```

3. After install brew have finished install. Check brew by command: brew update

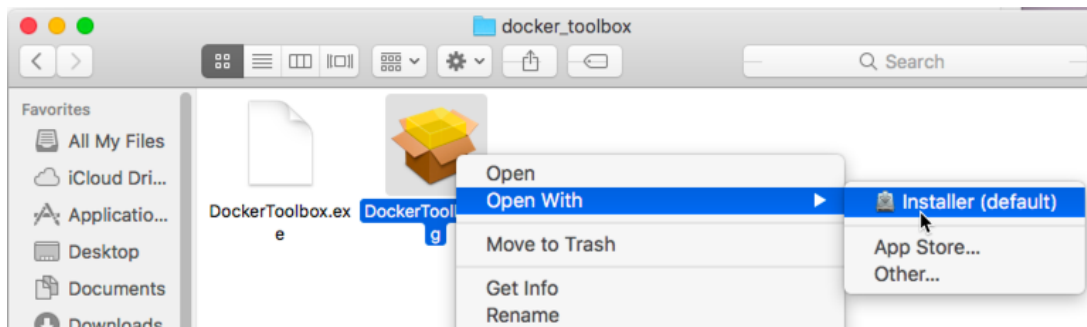
```
==> Tapping homebrew/core
Cloning into '/usr/local/Homebrew/Library/Taps/homebrew/homebrew-core'...
remote: Counting objects: 4449, done.
remote: Compressing objects: 100% (4250/4250), done.
remote: Total 4449 (delta 34), reused 462 (delta 13), pack-reused 0
Receiving objects: 100% (4449/4449), 3.53 MiB | 1.16 MiB/s, done.
Resolving deltas: 100% (34/34), done.
Tapped 4248 formulae (4,492 files, 11MB)
==> Cleaning up /Library/Caches/Homebrew...
==> Migrating /Library/Caches/Homebrew to /Users/praparn/Library/Caches/Homebrew...
==> Deleting /Library/Caches/Homebrew...
Already up-to-date.
==> Installation successful!

==> Homebrew has enabled anonymous aggregate user behaviour analytics.
Read the analytics documentation (and how to opt-out) here:
http://docs.brew.sh/Analytics.html

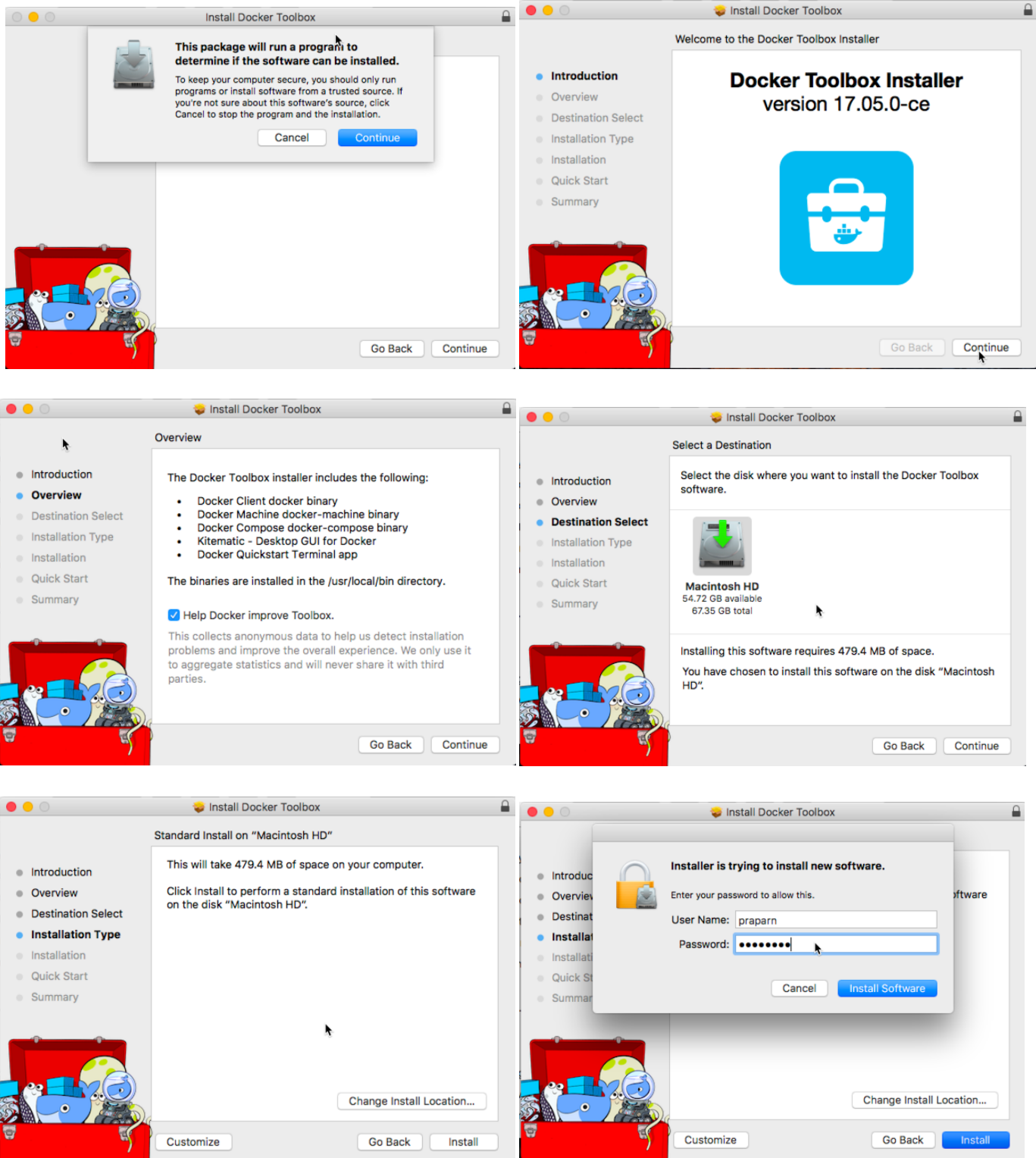
==> Next steps:
- Run `brew help` to get started
- Further documentation:
http://docs.brew.sh
[macos-10:~ praparn$ brew update
Already up-to-date.
macos-10:~ praparn$
```

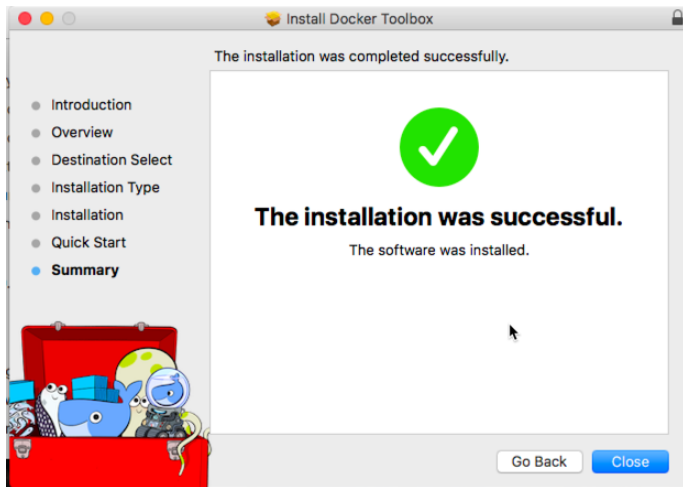
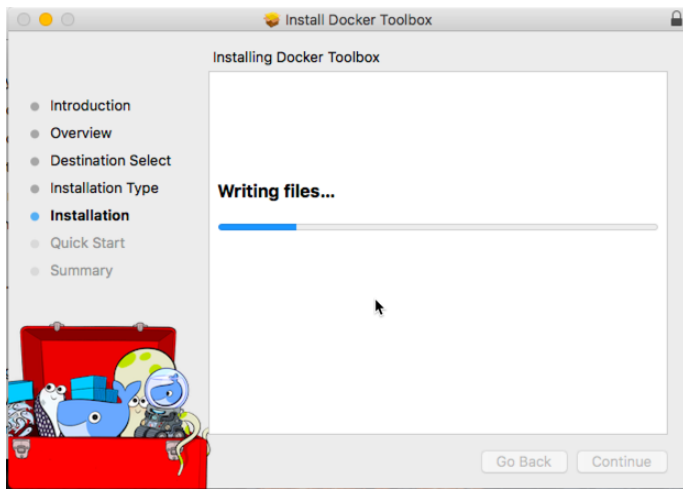
Install Docker Toolbox

1. Right Click on "DockerToolbox.dmg" and open with installer.

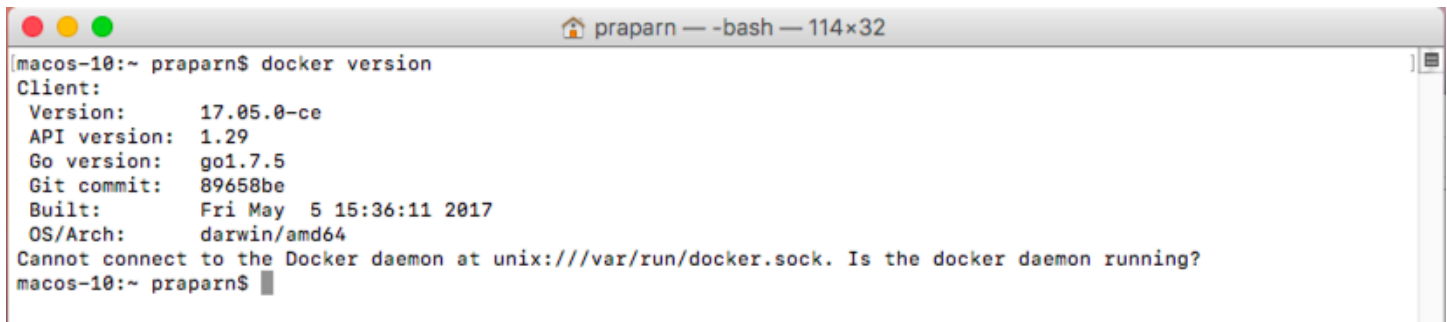


2. Following screen for setup





3. Check version of Docker Tools by command: `docker version`

A terminal window titled 'praparn — -bash — 114x32' with standard macOS window controls (red, yellow, green buttons). The terminal shows the command 'docker version' being executed. The output lists client information: Version 17.05.0-ce, API version 1.29, Go version go1.7.5, Git commit 89658be, Built Fri May 5 15:36:11 2017, and OS/Arch darwin/amd64. It then displays an error message: 'Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?'. The prompt returns to 'macos-10:~ praparn\$'.

```
macos-10:~ praparn$ docker version
Client:
 Version:      17.05.0-ce
 API version:  1.29
 Go version:   go1.7.5
 Git commit:   89658be
 Built:        Fri May  5 15:36:11 2017
 OS/Arch:      darwin/amd64
Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?
macos-10:~ praparn$
```

Install minikube / Initial minikube machine

1. Install minikube by command: brew cask install minikube

```
macos-10:~ praparn$ brew cask install minikube
==> Tapping caskroom/cask
Cloning into '/usr/local/Homebrew/Library/Taps/caskroom/homebrew-cask'...
remote: Counting objects: 3748, done.
remote: Compressing objects: 100% (3729/3729), done.
remote: Total 3748 (delta 34), reused 537 (delta 15), pack-reused 0
Receiving objects: 100% (3748/3748), 1.27 MiB | 1.18 MiB/s, done.
Resolving deltas: 100% (34/34), done.
Tapped 0 formulae (3,757 files, 4.0MB)
==> Creating Caskroom at /usr/local/Caskroom
==> We'll set permissions properly so we won't need sudo in the future
[Password:
==> Satisfying dependencies
==> Installing Formula dependencies from Homebrew
kubernetes-cli ... done
complete
==> Downloading https://storage.googleapis.com/minikube/releases/v0.20.0/minikube-darwin-amd64
##### 100.0%
==> Verifying checksum for Cask minikube
==> Installing Cask minikube
==> Linking Binary 'minikube-darwin-amd64' to '/usr/local/bin/minikube'.
🍺 minikube was successfully installed!
macos-10:~ praparn$
```

2. Check minikube interactive command

```
macos-10:~ praparn$ minikube
Minikube is a CLI tool that provisions and manages single-node Kubernetes clusters optimized for development workflows.

Usage:
  minikube [command]

Available Commands:
  addons          Modify minikube's kubernetes addons
  completion      Outputs minikube shell completion for the given shell (bash)
  config          Modify minikube config
  dashboard       Opens/displays the kubernetes dashboard URL for your local cluster
  delete          Deletes a local kubernetes cluster
  docker-env      Sets up docker env variables; similar to '$(docker-machine env)'
  get-k8s-versions Gets the list of available kubernetes versions available for minikube
  ip              Retrieves the IP address of the running cluster
  logs            Gets the logs of the running localkube instance, used for debugging minikube, not user code
  mount           Mounts the specified directory into minikube
  profile          Profile sets the current minikube profile
  service         Gets the kubernetes URL(s) for the specified service in your local cluster
  ssh             Log into or run a command on a machine with SSH; similar to 'docker-machine ssh'
  ssh-key         Retrieve the ssh identity key path of the specified cluster
  start           Starts a local kubernetes cluster
  status          Gets the status of a local kubernetes cluster
  stop            Stops a running local kubernetes cluster
  update-context  Verify the IP address of the running cluster in kubeconfig.
  version         Print the version of minikube
```


3. Install kubectl by command: brew install kubectl

```
praparn — -bash — 114x32
[macos-10:~ praparn$ brew install kubectl
Updating Homebrew...
^[[C==> Auto-updated Homebrew!
Updated 2 taps (caskroom/cask, homebrew/core).
==> Updated Formulae
camlp4      menhir      ocamlbuild  perl        subversion  vim@7.4
compcert    ocaml       ocamlsd1    rex
Warning: kubernetes-cli 1.6.6 is already installed
[macos-10:~ praparn$
```

4. Check kubectl interactive command

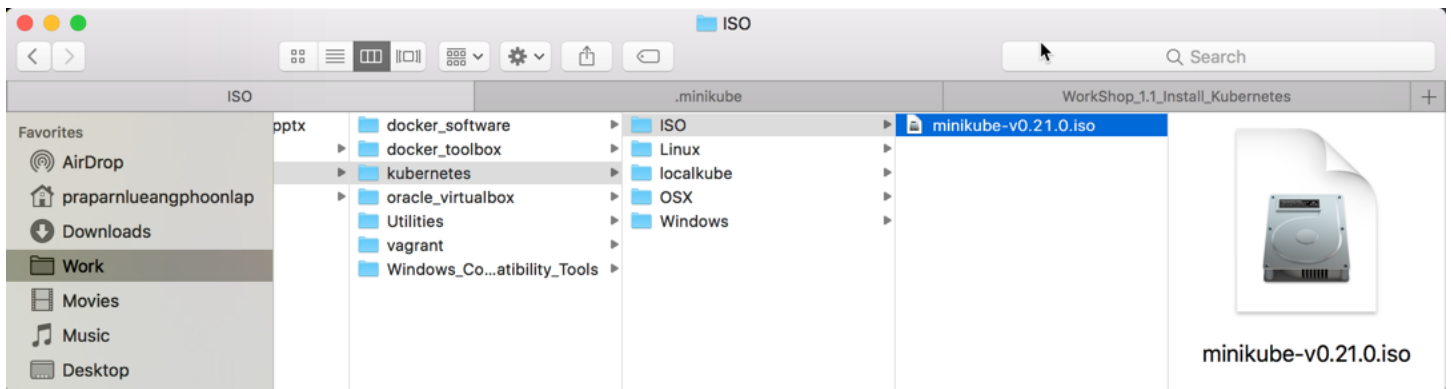
```
praparn — -bash — 114x32
[macos-10:~ praparn$ kubectl
kubectl controls the Kubernetes cluster manager.

Find more information at https://github.com/kubernetes/kubernetes.

Basic Commands (Beginner):
  create      Create a resource by filename or stdin
  expose      Take a replication controller, service, deployment or pod and expose it as a new
Kubernetes Service
  run         Run a particular image on the cluster
  set         Set specific features on objects

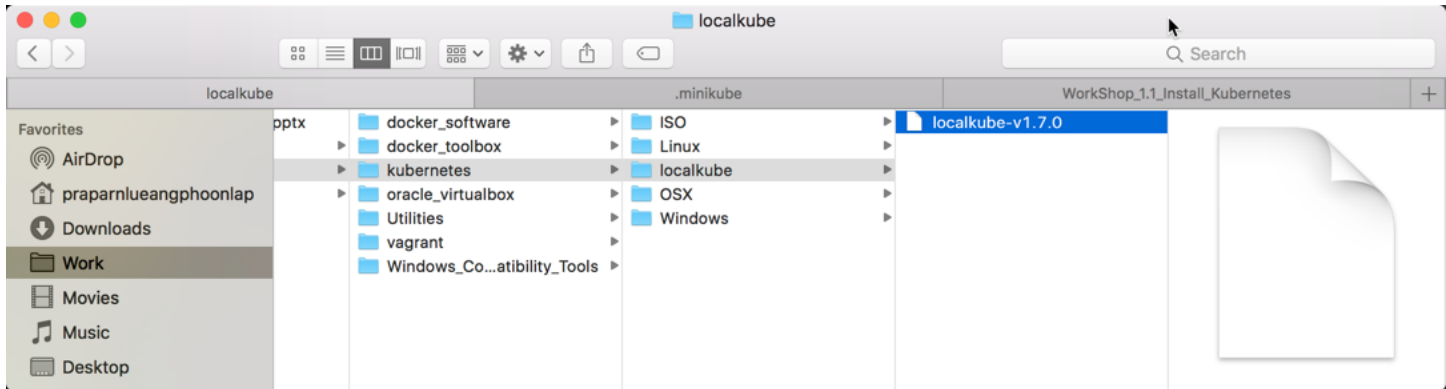
Basic Commands (Intermediate):
  get         Display one or many resources
  explain     Documentation of resources
  edit        Edit a resource on the server
  delete      Delete resources by filenames, stdin, resources and names, or by resources and
label selector
```

5. Copy file "minikube-v0.21.0.iso" from Software_Package/kubernetes/ISO/minikube-v0.21.0.iso to
/Users/<username>/.minikube/cache/iso



```
praparns-MacBook-Pro% cp minikube-v0.21.0.iso /Users/praparnlueangphoonlap/.minikube/cache/iso
praparns-MacBook-Pro%
```

6. Copy file "loalkube-v1.7.0" Software_Package/kubernetes/loalkube/loalkube-v.1.7.0" to
/User/<username>/.minikube/cache/



```
praparns-MacBook-Pro% ls
loalkube-v1.7.0          loalkube-v1.7.0-alpha.2
praparns-MacBook-Pro% cp loalkube-v1.7.0 /Users/praparnlueangphoonlap/.minikube/cache/loalkube
praparns-MacBook-Pro%
```

7. Configure minikube for use kubernetes version 1.7.0 by command: minikube config set kubernetes-version v1.7.0

```
praparns-MacBook-Pro% minikube config set kubernetes-version v1.7.0
```

8. Create minikube machine by command:

"minikube start --vm-driver=virtualbox profile=minikubelab1 --iso-url=https://storage.googleapis.com/minikube/iso/minikube-v0.21.0.iso"

```
praparns-MacBook-Pro% minikube start --vm-driver=virtualbox profile=minikubelab1 --iso-url=https://storage.googleapis.com/minikube/iso/minikube-v0.21.0.iso
Starting local Kubernetes v1.7.0 cluster...
Starting VM...
SSH-ing files into VM...
Setting up certs...
Starting cluster components...
Connecting to cluster...
Setting up kubeconfig...
Kubectl is now configured to use the cluster.
praparns-MacBook-Pro%
```

9. Check status of minikube's machine by command: "minikube status", "minikube ip"

```
praparns-MacBook-Pro:localkube praparn$ minikube status
minikubeVM: Running
localkube: Running
praparns-MacBook-Pro:localkube praparn$ minikube ip
192.168.99.104
praparns-MacBook-Pro:localkube praparn$
```

10. Test ssh to minikube's machine by command (user: docker, password: tcuser): minikube ssh

```
praparns-MacBook-Pro% minikube ssh
$ docker version
Client:
 Version:      1.12.6
 API version:  1.24
 Go version:   go1.6.4
 Git commit:   78d1802
 Built:        Wed Jan 11 00:23:16 2017
 OS/Arch:      linux/amd64

Server:
 Version:      1.12.6
 API version:  1.24
 Go version:   go1.6.4
 Git commit:   78d1802
 Built:        Wed Jan 11 00:23:16 2017
 OS/Arch:      linux/amd64
$
```

11. Check health of kubernetes cluster by command
 - a. `kubectl get nodes` → check node status
 - b. `kubectl get cs` → check cluster status

```
praparns-MacBook-Pro% kubectl get node
NAME          STATUS    AGE       VERSION
minikube      Ready     1m        v1.7.0
praparns-MacBook-Pro% kubectl get cs
NAME          STATUS    MESSAGE                     ERROR
scheduler     Healthy   ok                            
controller-manager Healthy   ok                            
etcd-0        Healthy   {"health": "true"}            
praparns-MacBook-Pro% 
```

12. Check status of kubernetes's elements by command
 - a. `kubectl get pods` → check pods element
 - b. `kubectl get deployment` → check deployment element
 - c. `kubectl get svc` → check service deploy on kubernetes
 - d. `kubectl describe svc` → check service description on kubernetes

```
praparns-MacBook-Pro:localkube praparn$ kubectl get pods
No resources found.
praparns-MacBook-Pro:localkube praparn$ kubectl get deployment
No resources found.
praparns-MacBook-Pro:localkube praparn$ kubectl get svc
NAME          CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
kubernetes    10.0.0.1     <none>        443/TCP    4m
praparns-MacBook-Pro:localkube praparn$ kubectl describe svc
Name:          kubernetes
Namespace:     default
Labels:        component=apiserver
               provider=kubernetes
Annotations:    <none>
Selector:      <none>
Type:          ClusterIP
IP:            10.0.0.1
Port:          https 443/TCP
Endpoints:     10.0.2.15:8443
Session Affinity: ClientIP
Events:        <none>
praparns-MacBook-Pro:localkube praparn$ 
```

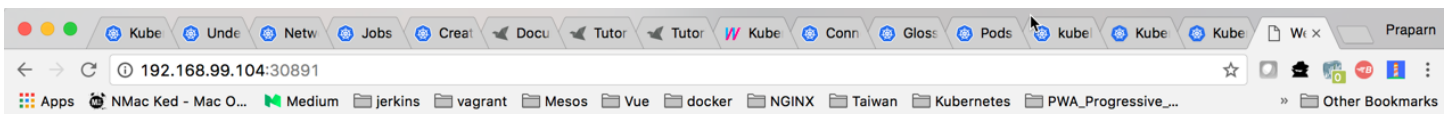
13. Test deployment "nginx" web server by command:
 - a. `kubectl run webtest --image=labdocker/nginx:latest --port=80` → deployment nginx (image: labdocker/nginx:latest) with port 80 service
 - b. `kubectl expose deployment webtest --target-port=80 --type=NodePort` → expose pods with service 80 (http)

```
praparns-MacBook-Pro:localkube praparn$ kubectl run webtest --image=labdocker/nginx:latest --port=80 deployment nginx
deployment "webtest" created
praparns-MacBook-Pro:localkube praparn$ kubectl expose deployment webtest --target-port=80 --type=NodePort
service "webtest" exposed
praparns-MacBook-Pro:localkube praparn$ 
```

14. Check port mapping for service with host by command:
 - a. `kubectl get svc webtest` → check mapping service
 - b. `kubectl describe svc webtest` → check description of service

```
praparns-MacBook-Pro:localkube praparn$ kubectl get svc
NAME          CLUSTER-IP   EXTERNAL-IP   PORT(S)        AGE
kubernetes    10.0.0.1     <none>        443/TCP        13m
webtest       10.0.0.83    <nodes>       80:30891/TCP   8s
praparns-MacBook-Pro:localkube praparn$ kubectl describe svc webtest
Name:         webtest
Namespace:    default
Labels:       run=webtest
Annotations:   <none>
Selector:     run=webtest
Type:         NodePort
IP:           10.0.0.83
Port:         <unset> 80/TCP
NodePort:     <unset> 30891/TCP
Endpoints:    172.17.0.4:80
Session Affinity: None
Events:       <none>
```

15. Test open webpage with port describe on command above (This example: <http://192.168.99.104:30891>)



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

16. Stop deployment by command and recheck again
 - a. `kubectl delete svc webtest`
 - b. `kubectl delete deployment webtest`

```
praparns-MacBook-Pro:localkube praparn$ kubectl delete svc webtest
service "webtest" deleted
praparns-MacBook-Pro:localkube praparn$ kubectl delete deployment webtest
deployment "webtest" deleted
praparns-MacBook-Pro:localkube praparn$
```