A template LaTeX report

Author Authorsen

November 12, 2018

Abstract

This document demonstrates usage of LaTeX.

Contents

1 Introduction

This is a LaTeX report template. It demonstrates cross-references, citations, images, and code examples with minted.

2 Examples

2.1 Code

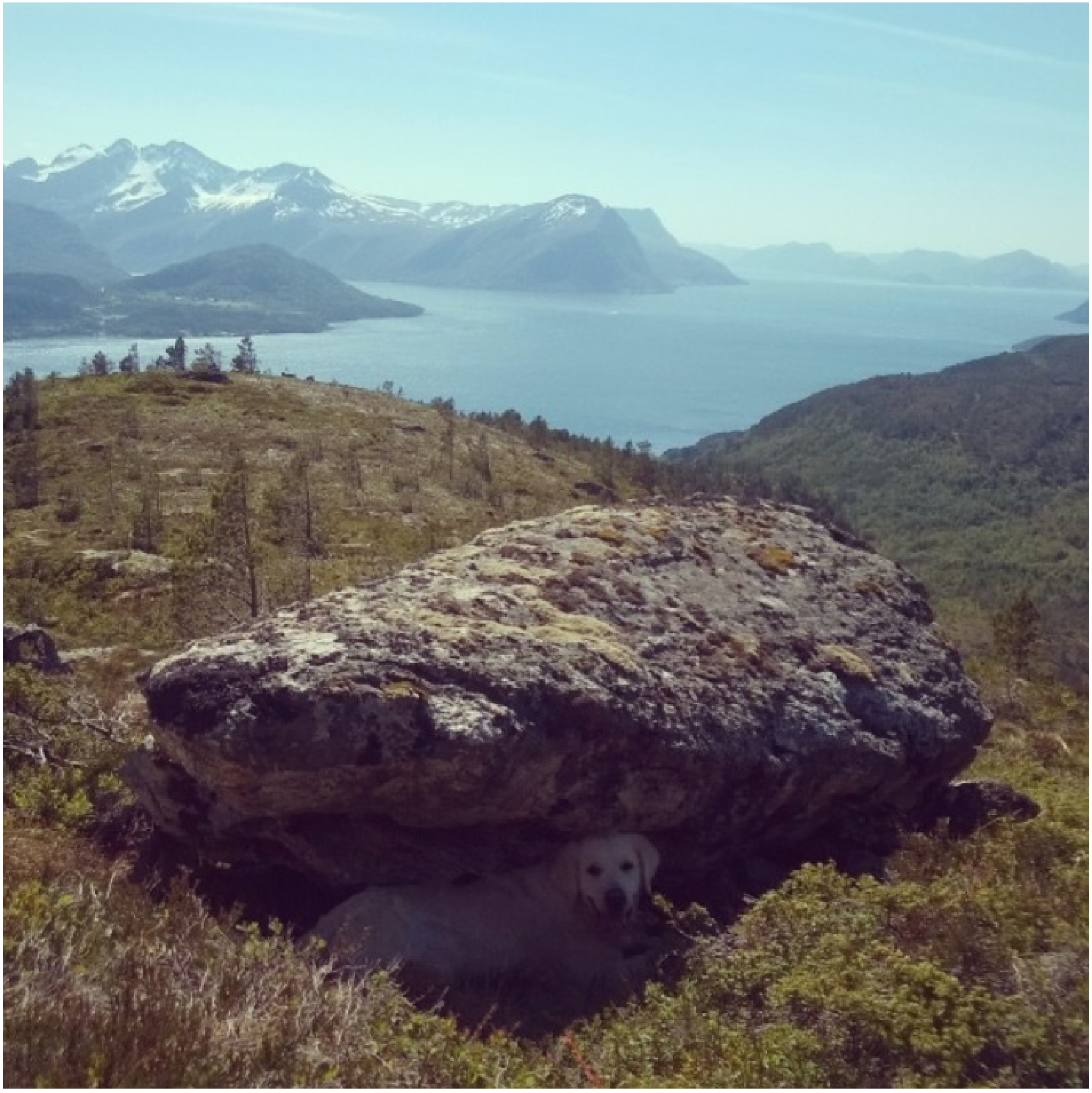
The following is a demonstration of how cross-references can be used to refer to appendix code. Citations are also used.

Beaker can work with different hypervisors by using plugins. If a plugin for a particular hypervisor does not exist, an alternative is to use Vagrant to manage the SUT's, and instead install and run Beaker as part of Vagrant's provisioning process. [2][1] An example of this is included in appendix A. This example would be used by running `vagrant up --verbose && vagrant destroy --force --verbose`.

2.2 Images

Figure 1 shows a picture of my dog. You're welcome. Because the ”H” option is set in the figure, like this: `\begin{figure}[H]`, LaTeX is forced to place it directly below this text. Because of the image's size LaTeX decides to put it on a new page. If you trust LaTeX to place your images, remove the ”H” option. The image will then appear wherever LaTeX can fit it in, and you'll have to rely on `\ref{}` and labels to refer to it.

Figure 1: My dog



3 Conclusion

This has been a demonstration of LaTeX in use.

References

[1] OpenStack documentation: Puppet Module Functional Testing. [Online; accessed 4-November-2018]. 1

[2] Puppet Module Functional Testing with Vagrant, OpenStack and Beaker. [Online; accessed 4-November-2018]. 1

A Beaker inside Vagrant example

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

require ’vagrant-openstack-provider’

#
# This is quite the minimal configuration necessary
# to start an OpenStack instance using Vagrant on
# an OpenStack with Keystone v3 API
#
# NOTE: this example is heavily
# inspired by http://my1.fr/blog/puppet-module-functional-testing-with-vagrant-openstack-and-beaker/
#
Vagrant.configure(’2’) do |config|

  config.ssh.username = ’ubuntu’

  config.vm.provider :openstack do |os, ov|
    os.server_name                         = ’vagrant_machine_in_openstack’
    os.security_groups                     = [ ’default’, ’linux’ ]
    os.identity_api_version                = ’3’
    os.openstack_auth_url                  = ’https://api.skyhigh.iik.ntnu.no:5000/v3’
    os.project_name                        = ’<PROJECTNAME>’
    os.user_domain_name                    = ’NTNU’
    os.project_domain_name                 = ’NTNU’
    os.username                            = ’<USERNAME>’
    os.password                            = ’<PASSWORD>’
    os.region                              = ’SkyHiGh’
    os.floating_ip_pool                    = ’ntnu-internal’
    os.floating_ip_pool_always_allocate = true
    os.flavor                              = ’m1.small’
    os.image                               = ’Ubuntu Server 16.04 LTS (Xenial Xerus) amd64’
    os.networks                            = [ ’<INTERNALNETID>’ ]

    ov.nfs.functional = false
  end

  # you could provision this machine using the same provisioning scripts used by
  # Heat, to create an exact duplicate
  config.vm.provision "shell", path: "bootscriptFromHeat.sh"


  # shell to install beaker, setup ssh, and run beaker tests.
  # written inline for sake of example
  config.vm.provision "shell", inline: <<-SHELL
    #!/bin/bash

    # install deps
    sudo apt-get update
    sudo apt-get install -y libxml2-dev libxslt-dev zlib1g-dev git ruby ruby-dev build-essential

    # prepare ssh
    echo "" | sudo tee -a /etc/ssh/sshd_config
    echo "Match address 127.0.0.1" | sudo tee -a /etc/ssh/sshd_config
    echo "    PermitRootLogin without-password" | sudo tee -a /etc/ssh/sshd_config
    echo "" | sudo tee -a /etc/ssh/sshd_config
    echo "Match address ::1" | sudo tee -a /etc/ssh/sshd_config
    echo "    PermitRootLogin without-password" | sudo tee -a /etc/ssh/sshd_config
    mkdir -p .ssh
    ssh-keygen -f ~/.ssh/id_rsa -b 2048 -C "beaker key" -P ""
    sudo mkdir -p /root/.ssh
    sudo rm /root/.ssh/authorized_keys
    cat ~/.ssh/id_rsa.pub | sudo tee -a /root/.ssh/authorized_keys
    sudo service ssh restart

    # prepare gems
    # this uses my gossinbacup module as an example, but it would be
    # possible to have the module as a parameter to this process
    git clone https://github.com/tholok97/gossinbackup
    cd gossinbackup
    sudo gem install bundler --no-rdoc --no-ri --verbose
    bundle install

    # run tests
    # this relies on SUT yaml definitions with hyporvisor set to "none",
    # like here: https://github.com/openstack/puppet-keystone/blob/master/spec/acceptance/nodesets/nodepool-xenial.yml
    export BEAKER_debug=yes
    bundle exec rspec spec/acceptance
SHELL

end
```