Exercism

1 Visão geral do projeto

O Exercism é um software desenvolvido em Go com o objetivo de ensinar diversas linguagens de programação, atualmente são suportadas mais de 40 linguagens diferentes e milhares de problemas a serem resolvidos. Os exercícios são apropriados para programadores de diversos níveis, desde níveis bem iniciantes a experientes. O projeto já recebeu fork de 740 desenvolvedores, foi favoritado por 2351 desenvolvedores e cerca de 346 contribuidores.

2 Problema

Atualmente o processo de instalação do Exercism não é muito amigável em algumas plataformas. Para usuários de OS X o processo é simples graças a um pacote disponível no Homebrew e para usuários de Windows graças a um pacote por meio do Chocolatey, mas para usuários de Linux o processo não é trivial. Ao consultar os desenvolvedores do projeto a respeito do motivo para não haver pacotes para Debian e Ubuntu disponíveis a resposta foi que apenas nunca houve uma demanda por parte dos usuários e interesse por parte dos desenvolvedores. Por meio desse empacotamento acredito que o uso do software será muito mais atraente para novos usuários.

3 Escopo

A proposta para desenvolvimento durante o semestre 2.2016 durante a disciplina de Gerência de Configuração de Software consiste no empacotamento .deb do software Exercism.

4 Definições, Acrônimos e Abreviações

5 Papéis

Todo o projeto será desenvolvido por apenas uma pessoa, o aluno Eduardo de Oliveira Castro.

Para instalação do Command Line Client (CLI) do Exercism atualmente o usuário precisa fazer o download do arquivo compactado de acordo com seu sistema operacional e coloca-lo na pasta "~/bin" manualmente. Além desse processo, é necessário que o usuário insira sua API KEY para que possa haver comunicação entre a máquia do usuário e os servidores do Exercism.

1 tar -xzvf exercism-linux-64bit.tgz 2 mkdir ~/bin 3 mv exercism ~/bin/ 4 export PATH=$HOME/bin:$PATH 5 echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc 6 7 # Para inserir sua API KEY que pode ser encontrada na pagina de configuracoes da sua conta no Exercism 8 exercism configure --key=YOUR_API_KEY

6 Ambiente de desenvolvimento: Vagrant e Puppet

Afim de padronizar e facilitar o ambiente de desenvolvimento do Exercism fora construída um ambiente de desenvolvimento virtual (box) utilizando o Vagrant já com todas as dependências necessárias para a execução do projeto e a versão mais recente do código disponível no repositório oficial. Vale ressaltar que para que a seguinte box funcione corretamente é necessário que o computador do usuário possua os softwares Virtual Box, Vagrant e Puppet instalados.

6.1 Vagrantfile

-*- mode: ruby -*- # vi: set ft=ruby : # vagrantfile Vagrant.configure(2) do |config| config.vm.box = “puppetlabs/ubuntu-16.04-64-puppet” config.vm.provision :puppet do |puppet| puppet.environment = “development” puppet.environment_path = “environments” end config.vm.provision :shell, path: “scripts/initial_install.sh” end

6.2 initial_install.sh

#!/usr/bin/env bash sudo apt-get install -y git go get github.com/exercism/cli/exercism cd $GOPATH/src/github.com/exercism/cli go get -t ./... go install github.com/exercism/cli/exercism

7 Instalação do Exercism

7.1 Script de instalação manual

Para instalar por meio do seguinte script basta copiar o texto para um editor de textos qualquer, salva-lo como install.sh e executalo via terminal (./install.sh).

#!/bin/sh usage() { cat <<-USAGE Usage: ${0##*/} [<option>...] [<path>] Install exercism client to <path>. Default: * determined interactively if possible * /usr/local/bin if run as root * /usr/local/bin if it is writable * $HOME/bin otherwise Options: -v <version> Install client version <version>. Default: $DEFAULT_VERSION -o <operating system> Install client for <operating system>. Default: $DEFAULT_OS -a <architecture> Install client for <architecture>. Default: $DEFAULT_ARCH USAGE } warn() { echo "${0##*/}: $*" >&2 } fail() { warn "$*" exit 1 } setup_downloader() { if command -v curl >/dev/null; then DOWNLOADER=curl elif command -v wget >/dev/null; then DOWNLOADER=wget else fail “missing dependencies, please install one of: curl, wget” fi } setup_defaults() { DEFAULT_DIR="$(default_dir)" DEFAULT_OS="$(default_os)" DEFAULT_ARCH="$(default_arch)" DEFAULT_VERSION="$(latest_version)" || exit $? } default_dir() { local DIR=/usr/local/bin if ["$(id -u)" -eq 0]; then echo "$DIR” elif [-d "$DIR” -a -w "$DIR”]; then echo "$DIR” else echo "$HOME/bin” fi } default_os() { case "$(uname -s)" in Darwin) echo “mac";; *) echo “linux";; esac } default_arch() { echo "$(getconf LONG_BIT)bit” } latest_version() { follow https://github.com/exercism/cli/releases/latest | filter_version_tag || fail 'Failed to get latest version. Check your internet connection.' } follow() { case "$DOWNLOADER” in curl) curl --head --silent "$@";; wget) wget --server-response --quiet --output-document=/dev/null "$@" 2>&1;; esac | tr -d '\r' } filter_version_tag() { awk -v FS=/ '/Location:/\protect\char"007B\relaxprint $NF}' } parse_options() { while getopts o:a:v:h OPTNAME; do case $OPTNAME in o) OS=$OPTARG;; a) ARCH=$OPTARG;; v) VERSION=$OPTARG;; h) usage; exit;; *) usage >&2; exit 64;; esac done shift $(expr $OPTIND - 1) case $# in 0) ;; 1) DIR=$1;; *) usage >&2; exit 64;; esac } use_defaults() { : ${DIR:="$(read_directory)"} : ${DIR:=$DEFAULT_DIR} : ${OS:=$DEFAULT_OS} : ${ARCH:=$DEFAULT_ARCH} : ${VERSION:=$DEFAULT_VERSION} } read_directory() { if [-t 0]; then read -p “install client into: [$DEFAULT_DIR] " REPLY if [-n "$REPLY”]; then echo "$REPLY” else echo "$DEFAULT_DIR” fi fi } check_directory() { check_directory_exists check_directory_writable check_directory_in_path } check_directory_exists() { test -d "$DIR” || mkdir -p "$DIR” || fail “unable to create installation directory $DIR” } check_directory_writable() { test -w "$DIR” || fail “unable to write installation directory $DIR” } check_directory_in_path() { echo "$PATH” | tr : "\n” | grep -q "^DIR" || warn “installation directory $DIR not in PATH” } install() { EXERCISM_FILES="https://github.com/exercism/cli/releases/download/\protect\char"0024\relaxVERSION/exercism-\protect\char"0024\relaxOS-\protect\char"0024\relaxARCH.tgz" echo -e "\nDownloading $EXERCISM_FILES...\n” download $EXERCISM_FILES | extract "$DIR” [$? -eq 0] && echo “Installed exercism to $DIR” } download() { case "$DOWNLOADER” in curl) curl -s --location "$@";; wget) wget --output-document=- "$@";; esac || fail “failed to download $*" } extract() { tar xz -C "$1” exercism || fail “failed to extract exercism” } setup_downloader setup_defaults parse_options "$@" use_defaults check_directory install

7.2 Empacotamento .deb

Para o empacotamento .deb fora utilizado a biblioteca dh-make, como recomendado pelos próprios tutoriais do Debian e do Ubuntu. Para a instalação desse pacote foi utilizado o seguinte comando:

sudo apt-get install dh-make debhelper devscripts fakeroot

É criada uma pasta chamada “exercism-0.1”, seguindo a convenção de nomes exigidas para os pacotes, e colocado o arquivo .tar.gz com o source code do projeto, bem como o binário do mesmo e o seguinte Makefile:

prefix = /usr/local all: src/exercism-0.1 src/exercism-0.1: src/exercism-0.1 @echo “CFLAGS=$(CFLAGS)" | \ fold -s -w 70 | \ sed -e 's/^/# /' $(CC) $(CPPFLAGS) $(CFLAGS) $(LDCFLAGS) -o $@ $^ install: src/exercism-0.1 install -D src/exercism-0.1 \ $(DESTDIR)$(prefix)/bin/exercism-0.1 clean: -rm -f src/exercism-0.1 distclean: clean uninstall: -rm -f $(DESTDIR)$(prefix)/bin/exercism-0.1 .PHONY: all install clean distclean uninstall

Feito isso, dentro da pasta, é realizado o seguinte comando no terminal para que toda a estrutura de pastas de um pacote debian:

dh_make --native

debian/rules:

#!/usr/bin/make -f %: dh $@ override_dh_auto_install: $(MAKE) DESTDIR=$$(pwd)/debian/exercism prefix=/usr install

debian/control:

Source: exercism Maintainer: Eduardo Castro <edu-

ardocastro91@gmail.com> Section: misc Priority: optional Standards-Version: 3.9.2 Build-Depends: debhelper (>= 9) Package: exercism Architecture: any Depends: ${shlibs:Depends}, ${misc:Depends} Description: Software that helps people who wants to learn how to develop software. Exercism offers support to more than 40 languages, including all the most commons ones like Java, Ruby, Python, Swift, C, etc. The user receives a random challenge according to his capacity level and has to develop a software that makes all the tests (yes, Exercism is TDD based!) passes.

Feito isso basta usar o seguinte comando para que o arquivo .deb seja gerado:

debuild -us -uc

8 Link para o GitHub com os códigos do projeto

http://www.github.com/educastro/GCS-exercism

9 Referências

1. https://www.debian.org/doc/manuals/maint-guide/
2. http://packaging.ubuntu.com/html/
3. https://wiki.debian.org/Packaging
4. https://www.vagrantup.com/docs/
5. http://www.exercism.io/

10 Fontes dos textos e imagens, contribuidores e licenças

10.1 Texto

- **Exercism** *Fonte:* https://pt.wikiversity.org/wiki/Exercism?oldid=96393 *Contribuidores:* Eduardocastro91 e Anónimo: 3

10.2 Imagens

10.3 Licença