ByteLife

vFLOWer Toolkit User Guide

PREPARED BY: Kaido Kibin

www.bytelife.com

Table of Contents

1 Preface

VMware vCenter Orchestrator content (workflows, actions, configuration items, etc.) can be exported/imported as packages which are stored in binary compressed format. Due to this binary nature it has been difficult to publish vCO content to remote versioning repositories (such as GitHub) in a reasonable way. Current document describes a solution developed in ByteLife Solutions called vFLOWer Toolkit to overcome this challenge allowing public vCO workflow market to become a reality.

Note. Current user guide is focusing to the GitHub as a most common public repository. Any repository server (remote or local) could be used instead.

2 Overview

vFLOWer Toolkit along with its pre-requisite components allows vCO administrators and/or developers to perform the following actions:

- Retrieve vCO content source code in XML format from remote version control repositories and build a binary vCO package to be imported into vCO.
- Publish vCO content source code in XML format to remote version control repositories by unpacking/converting exported binary vCO packages.

Current version of this integration tool handles only vCO packages. For publishing, vCO packages must be created first including all needed content.

3 Pre-requisites

To work properly vFLOWer Toolkit needs multiple additional publicly available software components:

- Apache Ant™. Tested with version 1.9.3. http://ant.apache.org.
- Oracle Java™ SE Development Kit. Tested with JDK™ 1.7.0.45 for Windows. http://www.oracle.com/technetwork/java/javase/downloads/index.html.
- OpenSSL. Needed for vCO packages retrieval only. Tested with OpenSSL 1.0.1e for Windows. http://www.openssl.org.
- Git command-line client. Tested with Git 1.8.5.2 for Windows. http://git-scm.com/downloads.

- VMware vCenter Orchestrator client.

Notes

- This product includes software developed by the Ant-Contrib project (http://sourceforge.net/projects/ant-contrib).
- When using Windows as a client platform, Git client must be set up with UNIX-style file formats. vCO is expecting that.
- On a client machine ANT_HOME and JAVA_HOME environment variables must be set. In addition, PATH environment variable must be updated to include ANT, Java, Git client and OpenSSL binary directories.

4 Publishing vCO content to GitHub

4.1 SUMMARY STEPS

Multiple manual steps must be performed in order to publish vCO content to GitHub repository. All steps can be performed from a client machine having all needed pre-requisite components installed.

1. User account created in GitHub (if not existing).
2. New repository created in GitHub and in client machine.
3. A vCO package created using vCO client to include all needed content.
4. The package exported to the client machine into specified input/output directory (part of repository folder structure) using vCO client.
5. ByteLife's ANT precommit script launched from the repository root directory to unpack/convert the vCO package into XML-based source code.
6. New content commited and pushed to the remote GitHub repository (git commit, git push).

4.2 USER ACCOUNT CREATION IN GITHUB

Follow the "Sign up for GitHub" process in https://github.com.

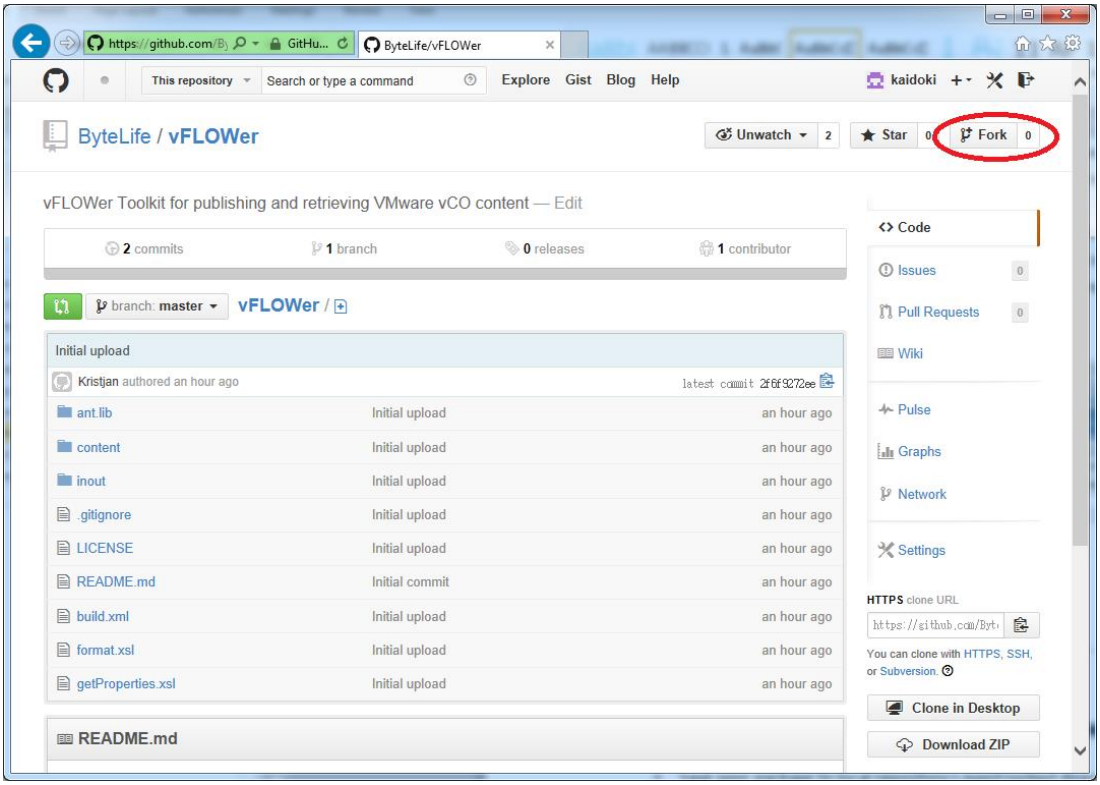
4.3 NEW REPOSITORY CREATION

To publish vCO content new repository must be created. To make it usable by vFLOWer Toolkit new repository must contain all needed folders and ANT components. There are two ways to accomplish this:

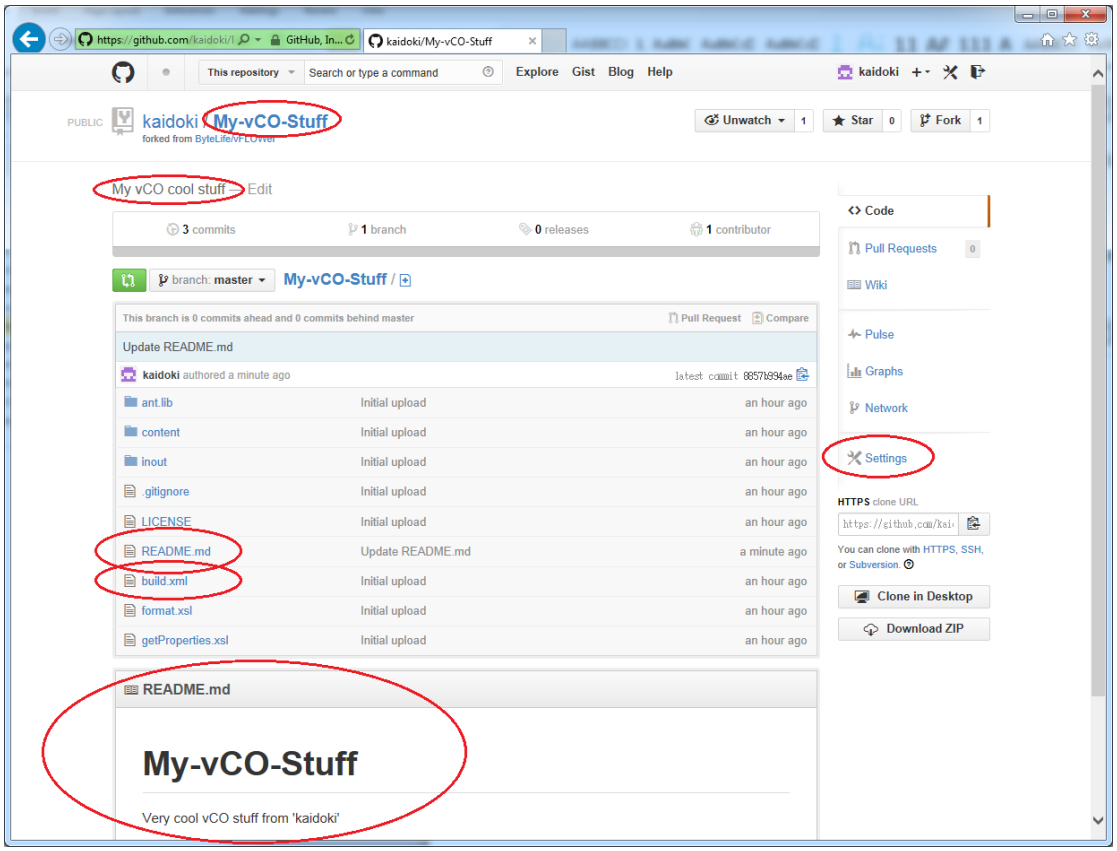
1. Fork ByteLife's vFLOWer repository in GitHub. Forking creates exact copy of source repository and leaves it logically connected to the original one. This allows later changes in vFLOWer Toolkit to be easily merged into the target repository. Downside is that as target repository is bound to the source repository GitHub does not show it in search results. Potential users must know exact location of that repository.
2. Creating new empty repository in GitHub and copying vFLOWer Toolkit content into it. This can be accomplished by cloning ByteLife's vFLOWer repository locally to a client machine, changing its origin (remote master) to the empty GitHub repository and pushing content up to GitHub. This way new repository is totally independent, thus making it visible through GitHub search functionality. Downside is that future changes in vFLOWer Toolkit must be manually merged (copied) into target repository.

4.3.1 Forking ByteLife's repository

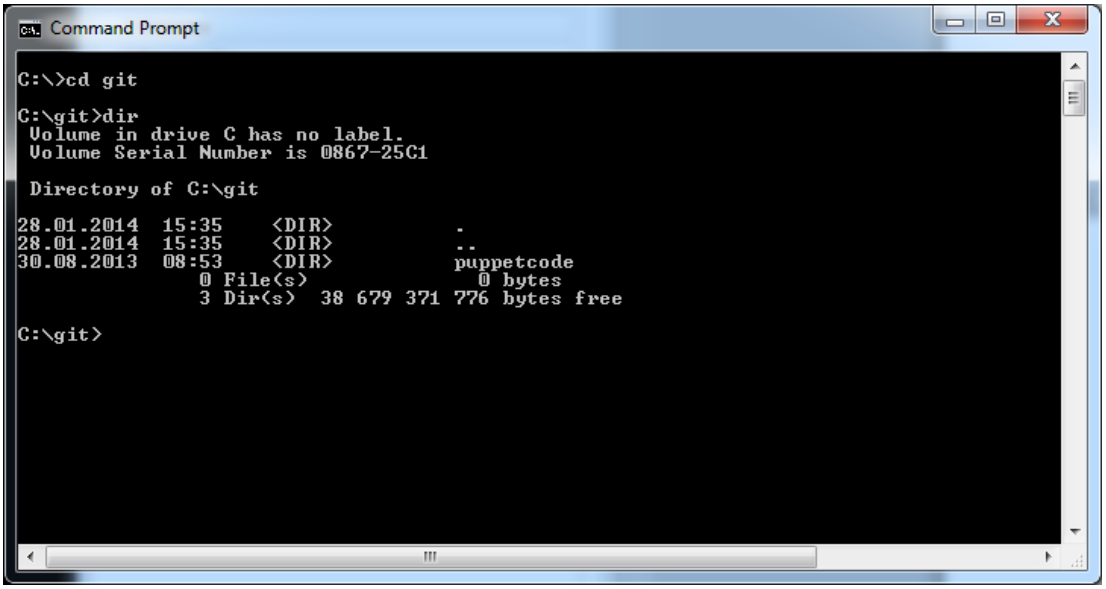
1. Log into GitHub using your user account.
2. Locate ByteLife's vFLOWer repository by typing keyword "flower" to the search field.
3. Click "Fork" button on the upper right corner.



4. Select an account under which to create this new repository (you might have multiple accounts visible there).
5. Customize your new repository:
    a. change repository name and description
    b. change repository settings
    c. edit README.md file to add your information
    d. edit build.xml file to change project name and self-signed certificate information used to digitally sign vCO packages during build

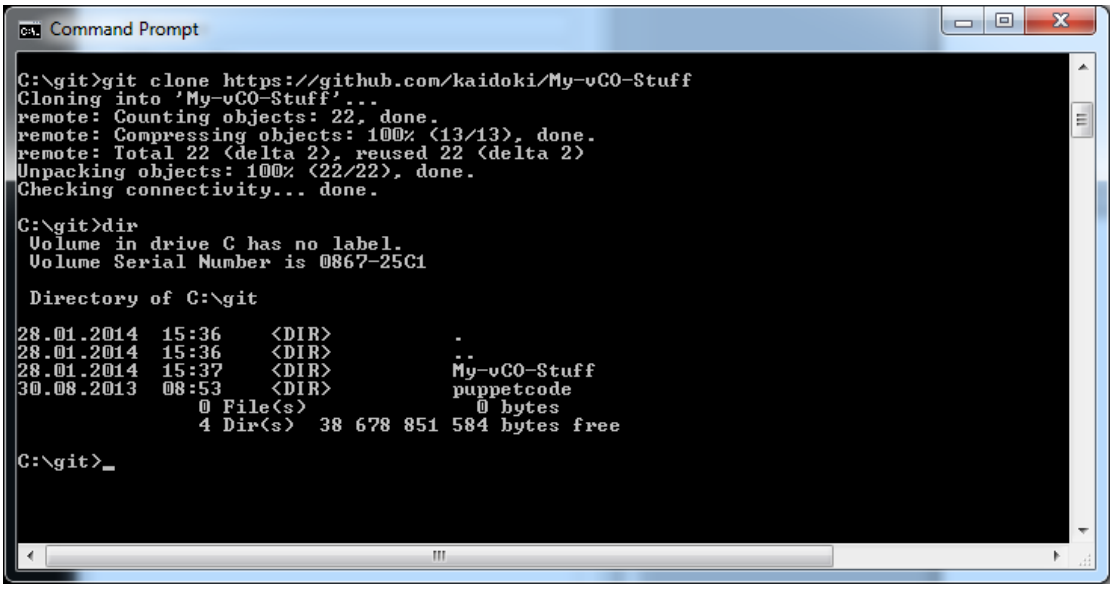


6. Clone remote repository to your local machine:
    a. Open a command prompt and change working directory to a local Git repositories root directory.

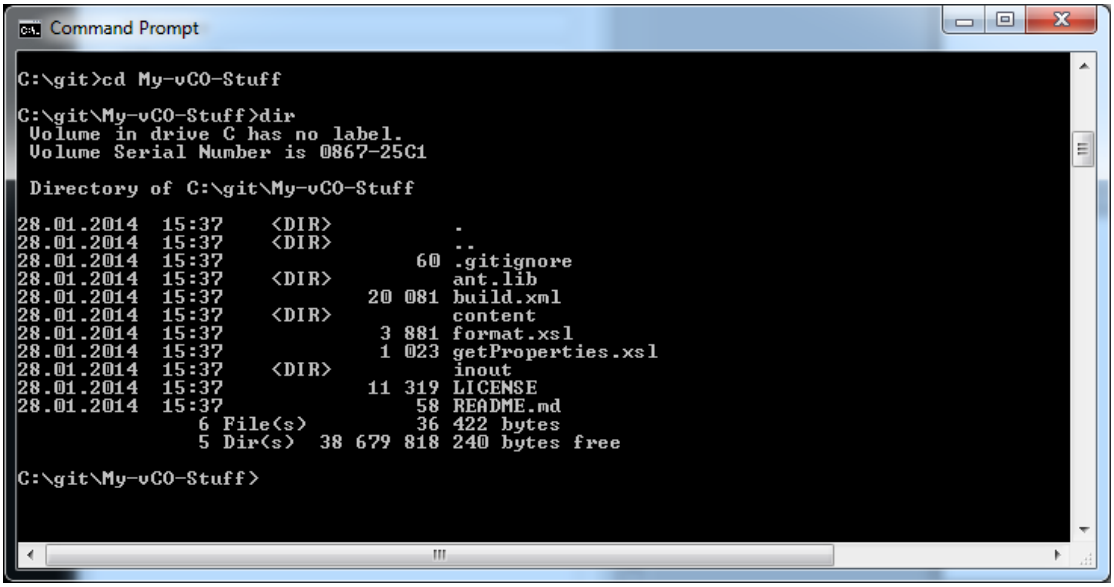


b. Clone the remote repository to your client machine using the following command:

```
git clone <your GitHub repository URL>
```



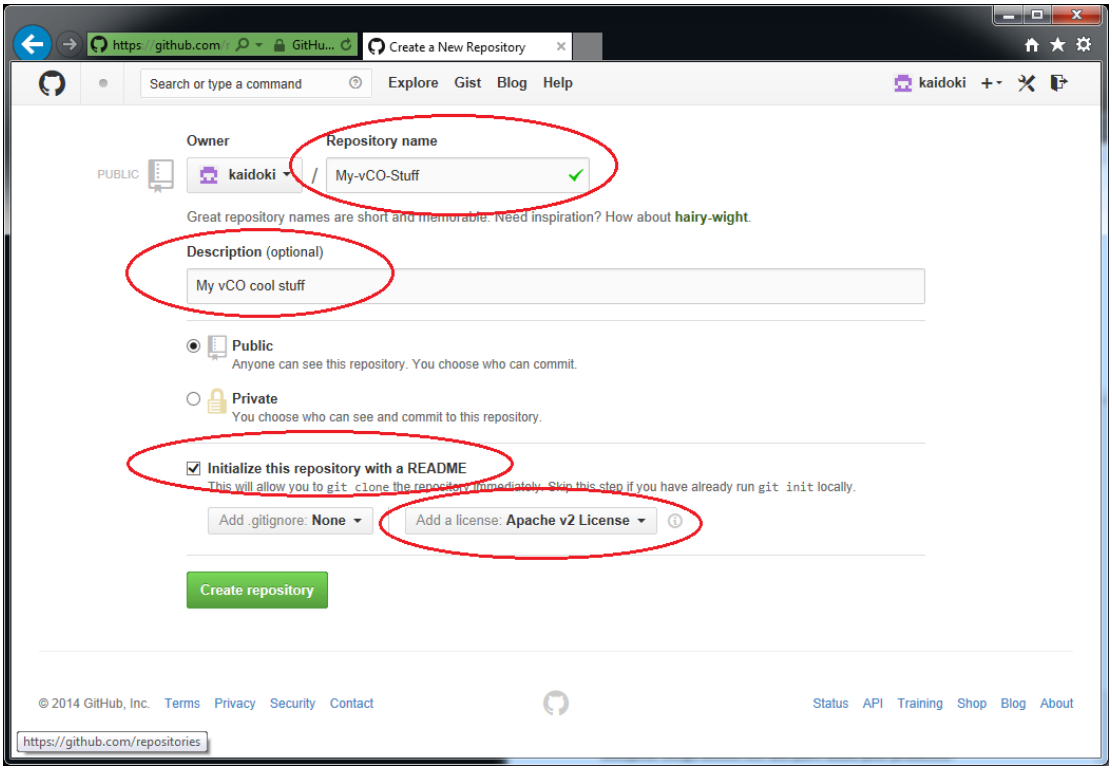
c. Verify your local repository's content.



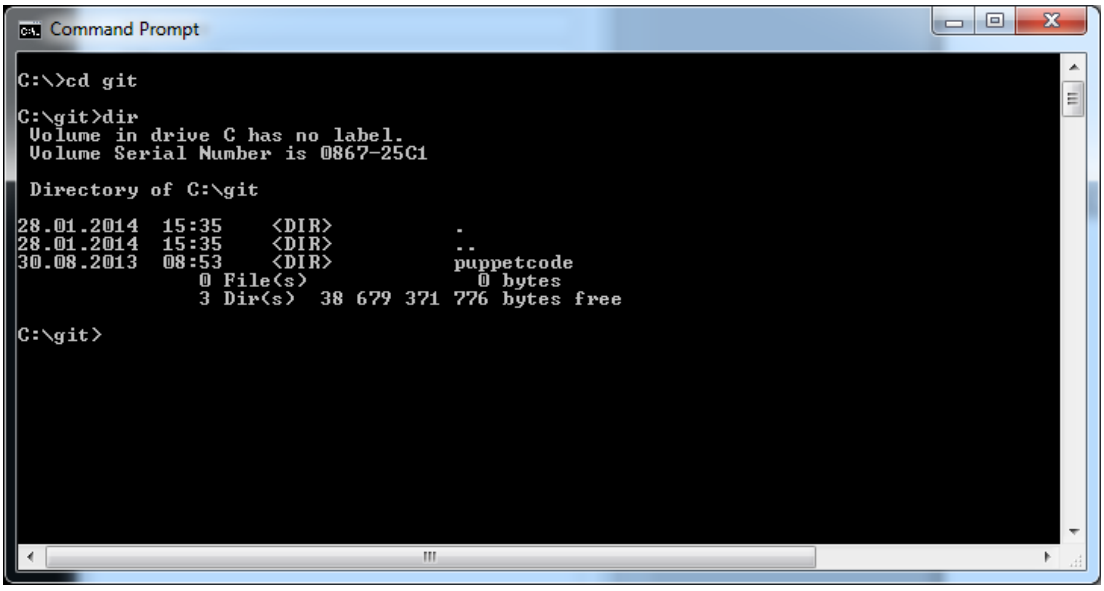
7. Continue with vCO Package Creation step.

4.3.2 Creating new repository in GitHub

1. Log into GitHub using your user account.
2. Click “New repository” button.
3. Fill in the repository creation form:
    a. Give name to your repository
    b. Add description for your repository
    c. Optionally add README to you repository
    d. Optionally add proper license to your repository (“Apache v2 License” should be suitable in most cases)
4. Confirm your changes by clicking “Create repository” button.

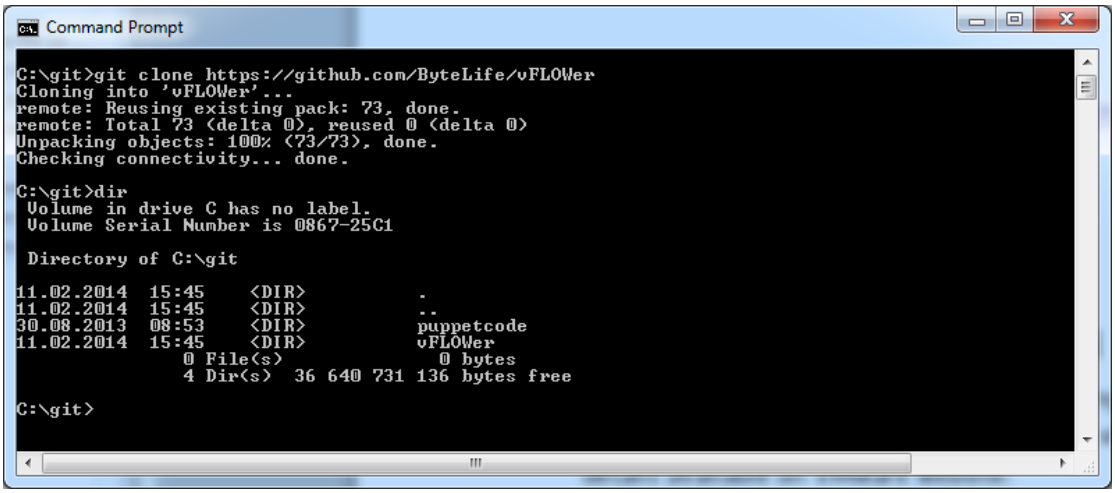


5. Clone ByteLife's vFLOWer repository to your local machine:
    a. Open a command prompt and change working directory to a local Git repositories root directory.



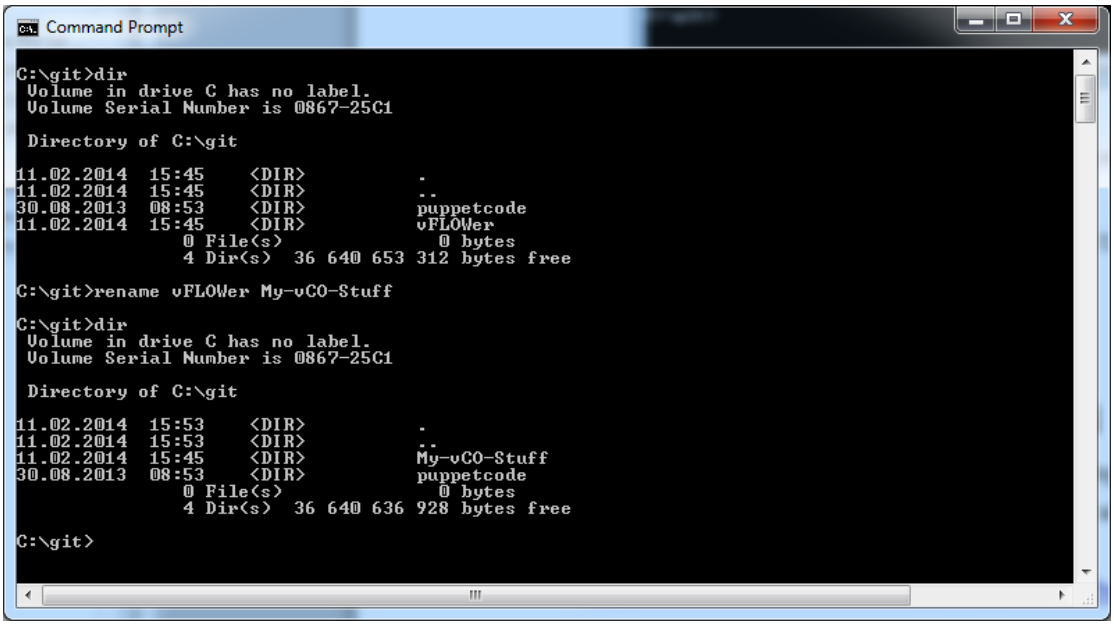
    b. Clone the ByteLife's repository to your client machine using the following command:

```
git clone https://github.com/ByteLife/vFLOWer
```

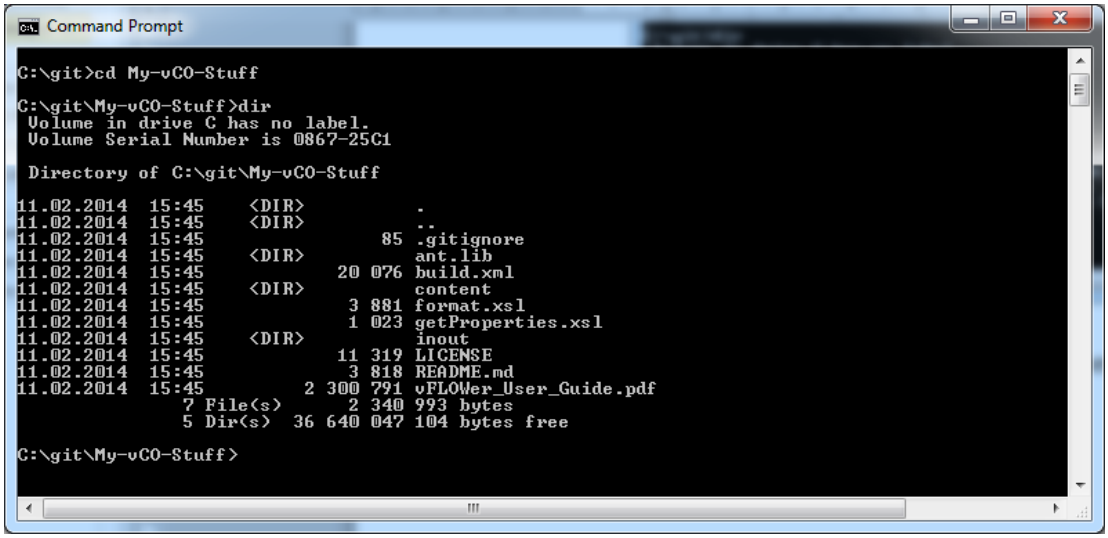


6. Rename local vFLOWer repository to match your new empty GitHub repository:

```
rename vFLOWer <Your new GitHub repository name>
```

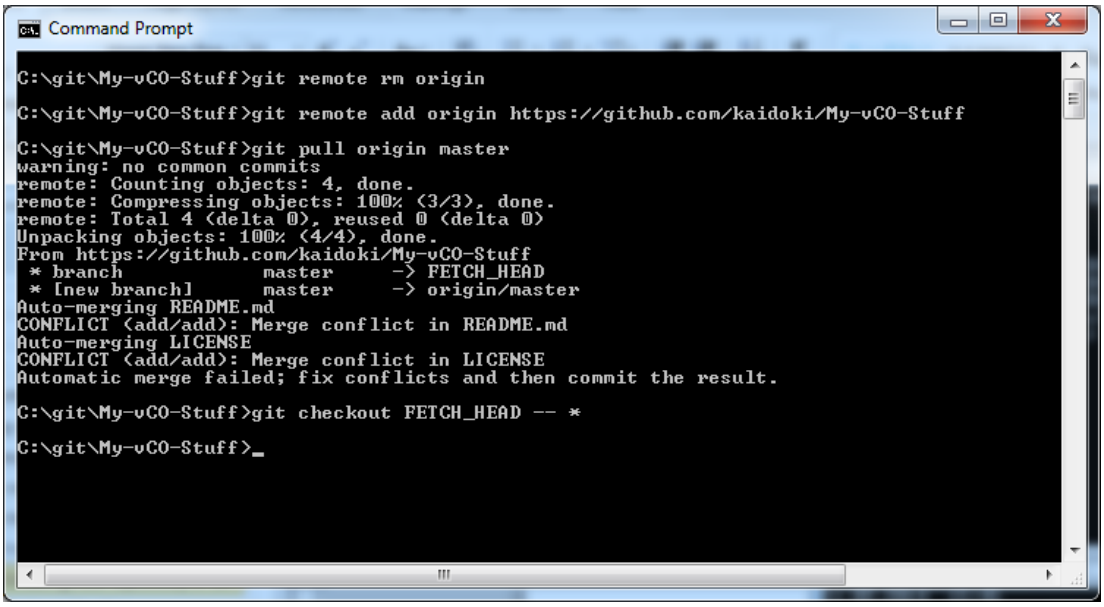


8. Change your working directory into new local repository.



9. Change local repository's master origin to unlink it from ByteLife's vFLOWer repository and link it to your new remote repository:

```
git remote rm origin
git remote add origin <your GitHub repository URL>
git pull origin master
git checkout FETCH_HEAD -- *
```



First command removes current remote repository information (which points to ByteLife's repository) from local repository. Second command adds your new GitHub repository as remote repository to the local copy. Third command syncs remote copy

with local copy. There will probably be two merge conflicts for license file and readme because they existed in both repositories (in your new repository and also in ByteLife's vFLOWer repository). This generates two version branches locally. Fourth command specifies that version branch from your repository will overwrite ByteLife's version.

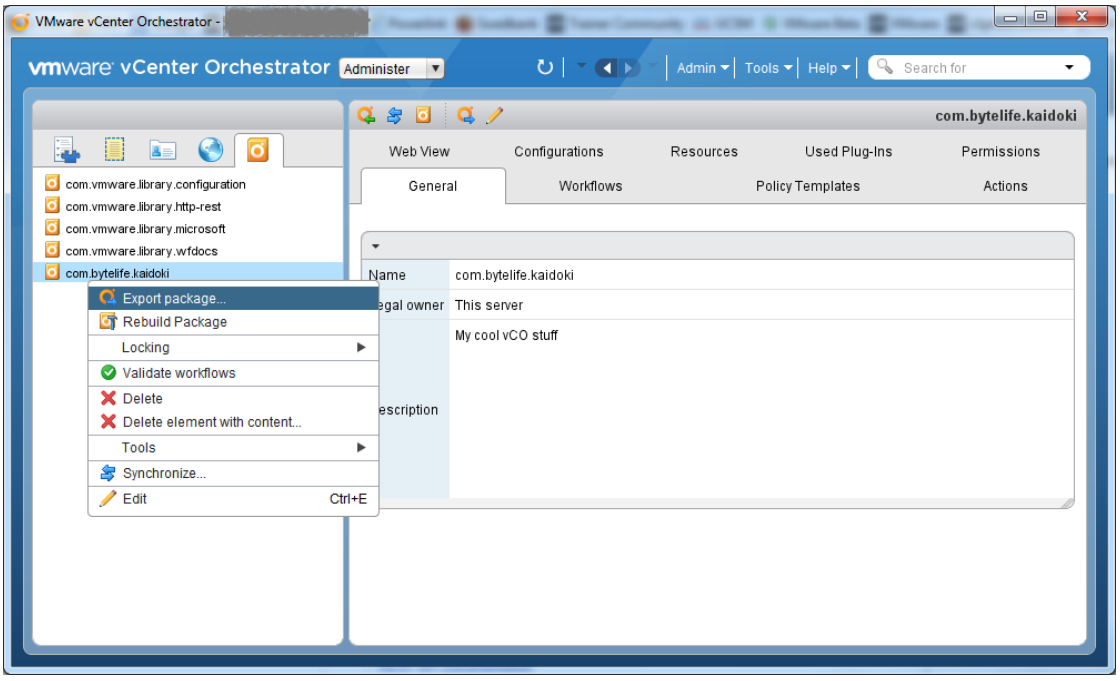
4.4 VCO PACKAGE CREATION

vCO package must be created manually using vCO Client. Package must contain all content needed by your vCO workflows. Follow the vCO documentation regarding package creation details available on VMware website:
https://www.vmware.com/support/pubs/orchestrator_pubs.html

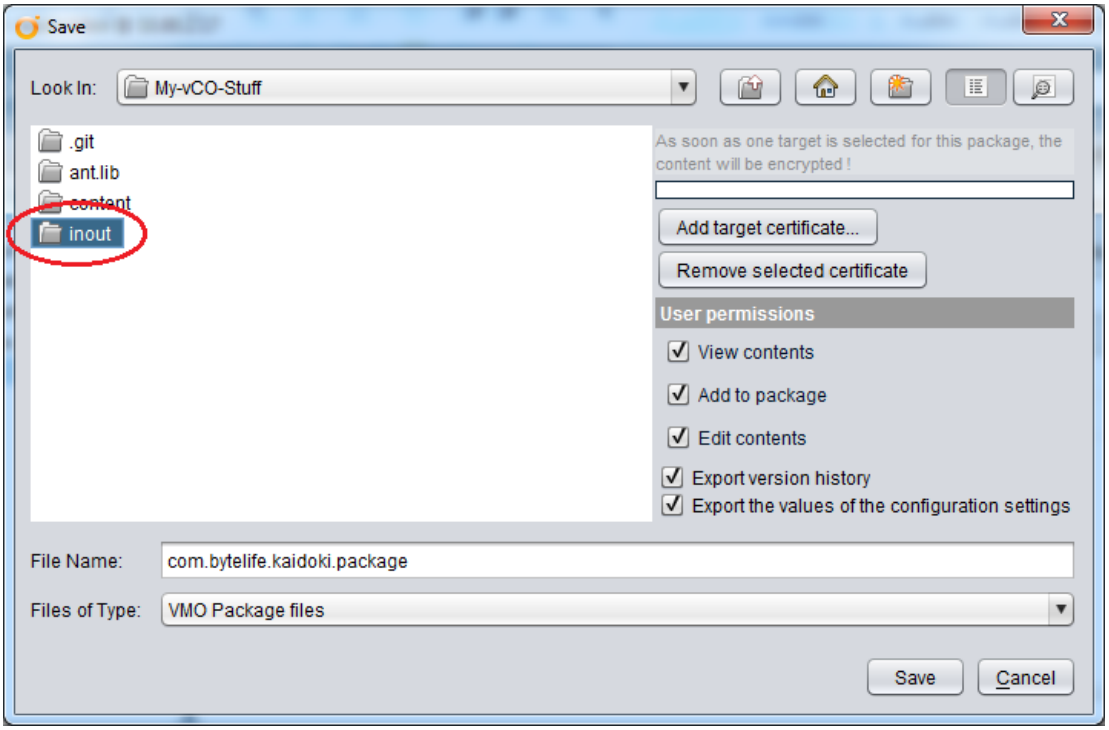
4.5 EXPORTING VCO PACKAGE TO THE GIT REPOSITORY

vCO package must be exported to the local repository input/output directory using vCO Client. Steps to accomplish this are the following:

1. Open vCO Client and log into your vCO server and switch to "Administer" view.
2. Locate your package under Packages tab in left panel, right-click on it and select "Export package...".



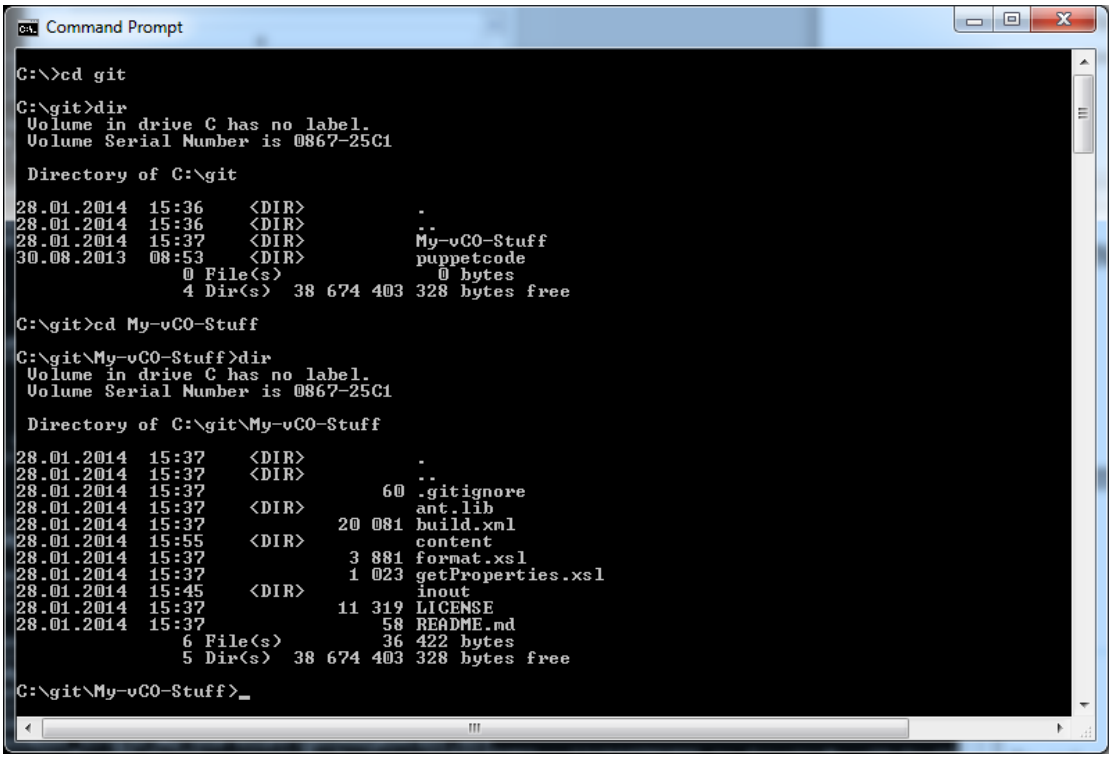
3. Save your package to local repository's input/output directory ("inout").


Save
Look In:
My-vCO-Stuff
.git
ant.lib
content
inout
As soon as one target is selected for this package, the content will be encrypted !
Add target certificate...
Remove selected certificate
User permissions
View contents
Add to package
Edit contents
Export version history
Export the values of the configuration settings
File Name:
com.bytelife.kaidoki.package
Files of Type:
VMO Package files
Save
Cancel

4.6 UNPACK AND CONVERT VCO PACKAGE INTO XML

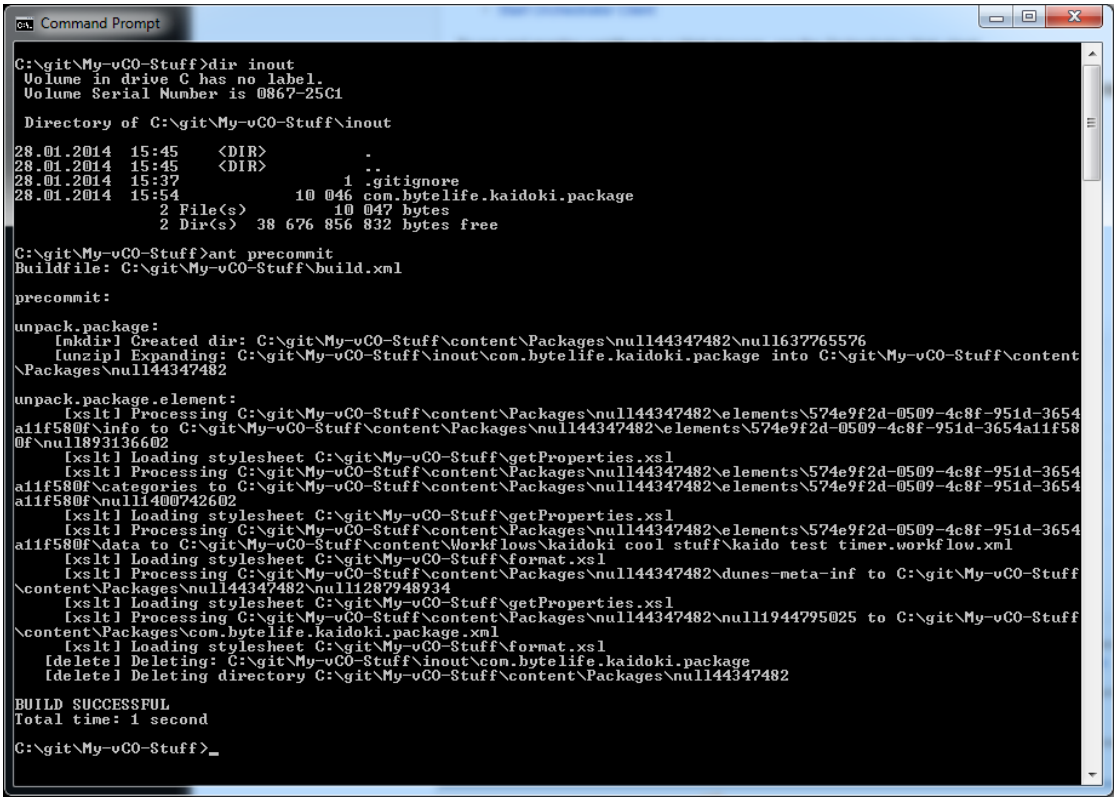
To get a versionable source code out of the binary vCO package a custom ANT script (provided by vFLOWer Toolkit) must be used to unpack and convert vCO package into XML format. Steps to accomplish this are the following:

1. Open a command prompt and change to your local repository's root directory (location of the `build.xml` file).

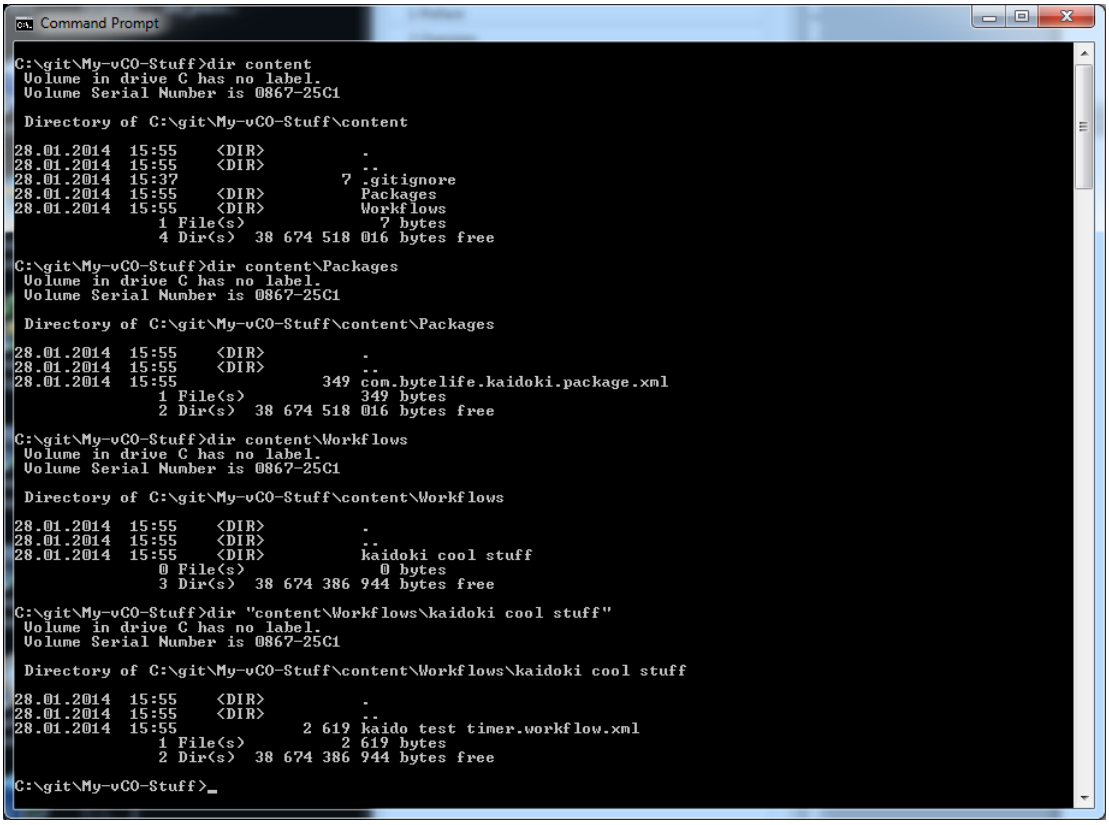


2. Verify that vCO package exists in your repository's "inout" directory.
3. Launch unpacking/conversion script by running the following command:

```
ant precommit
```



4. Verify the content of the repository's "content" directory. There must be new subfolders created containing XML source files.



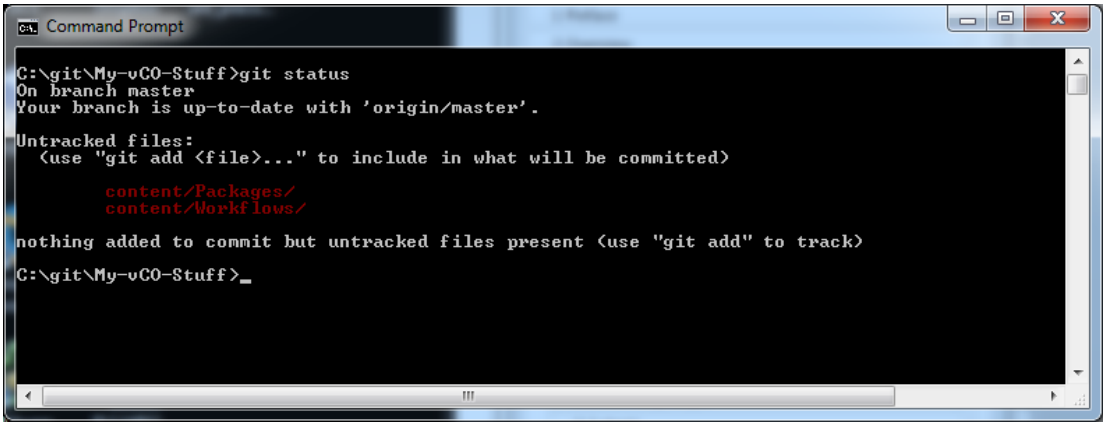
4.7 COMMITING AND PUSHING CONTENT TO GITHUB

Now that you have your vCO package converted to XLM-based source code files it can be published to the GitHub. Steps to accomplish this are the following:

1. Open a command prompt and change to your local repository's root directory (location of the `build.xml` file).
2. Verify changes in the local repository using the following command:

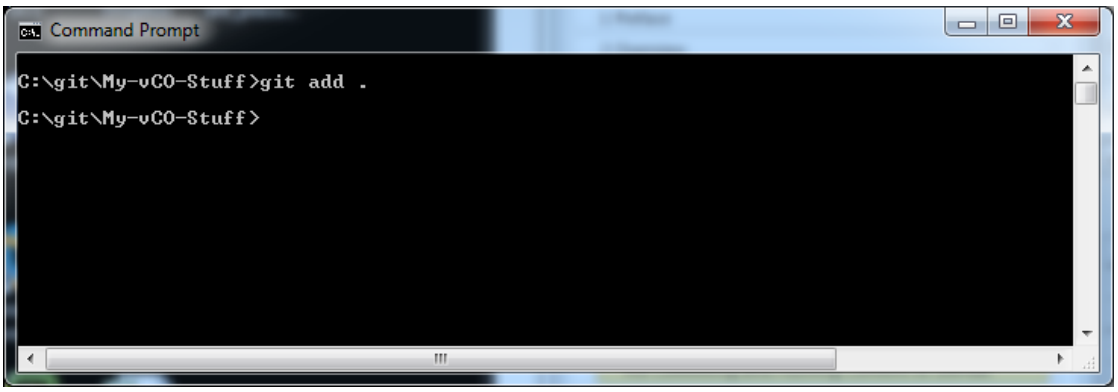
   ```
   git status
   ```

   New content of the "content" directory should be listed in the output.



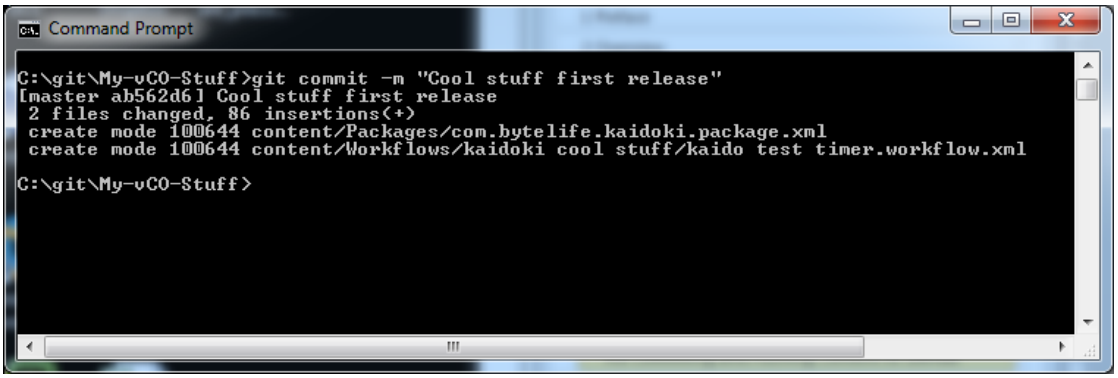
3. Add all changes starting from the current directory to the versioning snapshot using the following command:

```
git add .
```



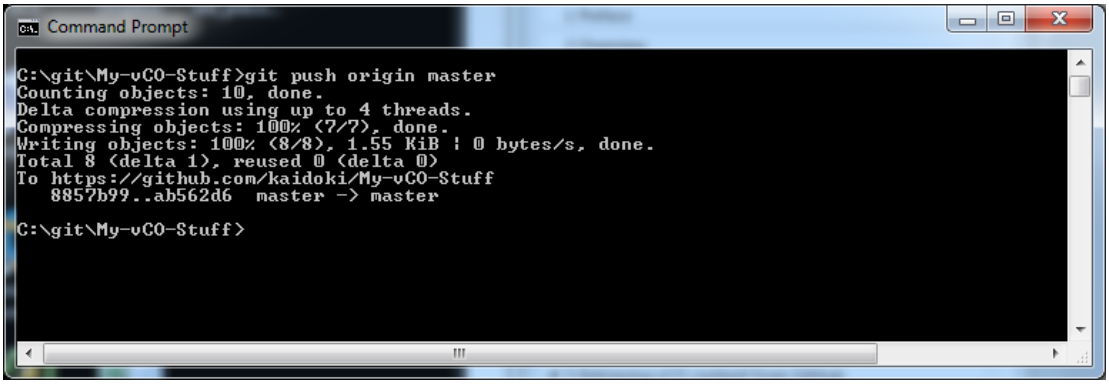
4. Commit changes in the versioning snapshot using the following command:

```
git commit -m <version comments>
```

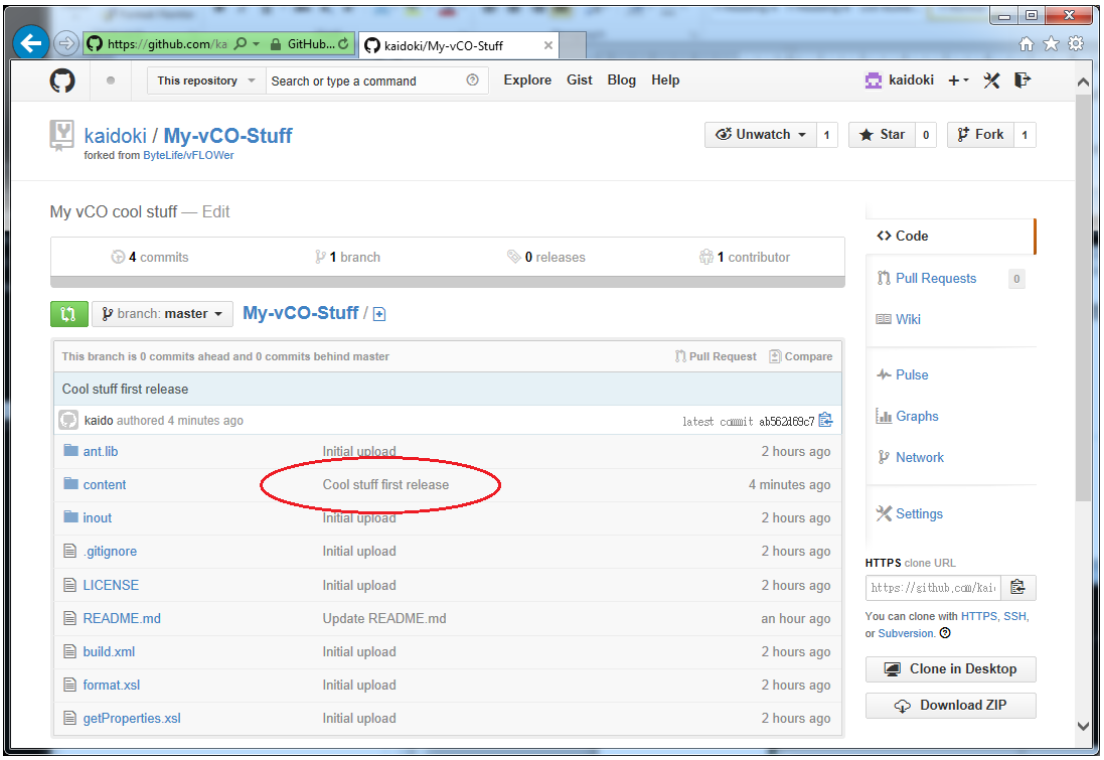


5. Push your changes to the GitHub repository using the following command:

```
git push origin master
```



6. Verify your repository content in GitHub.



5 Retrieving vCO content from GitHub

5.1 SUMMARY STEPS

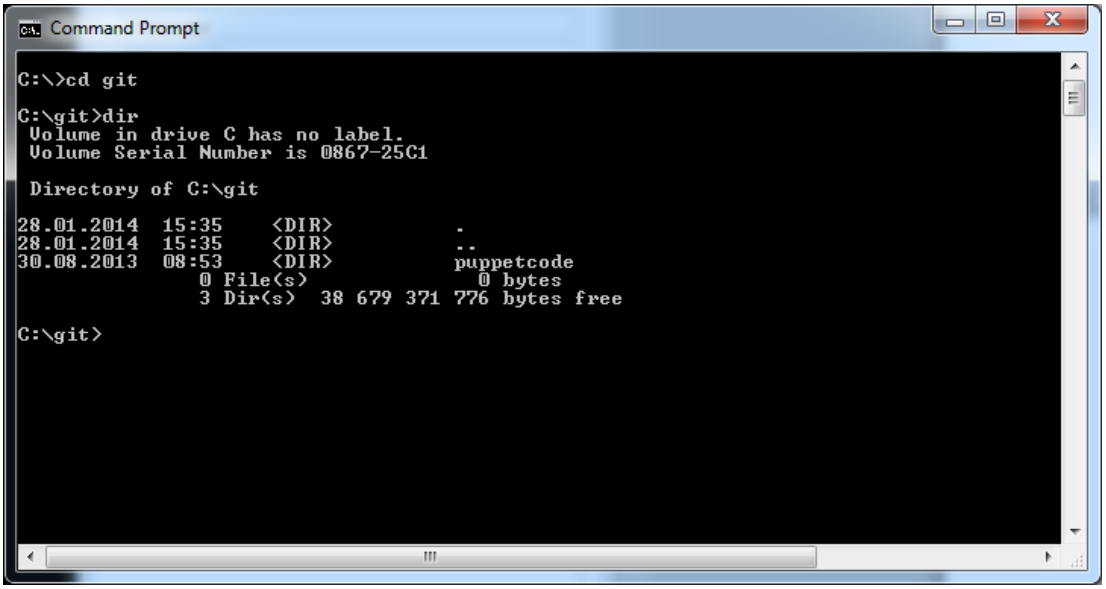
Multiple manual steps must be performed in order to retrieve vCO content from public GitHub repository. All steps can be performed from a client machine having all needed pre-requisite components installed.

1. vCO content source code downloaded from a public GitHub repository. Content can be downloaded by cloning remote repository (git clone) to the client machine which includes all needed ANT scripts and folder structure.
2. ByteLife's ANT build script launched from the repository root directory to build a vCO package from downloaded XML-based source code.
3. The package imported from specified input/output directory to the vCO using vCO client.

5.2 DOWNLOADING VCO CONTENT FROM GITHUB

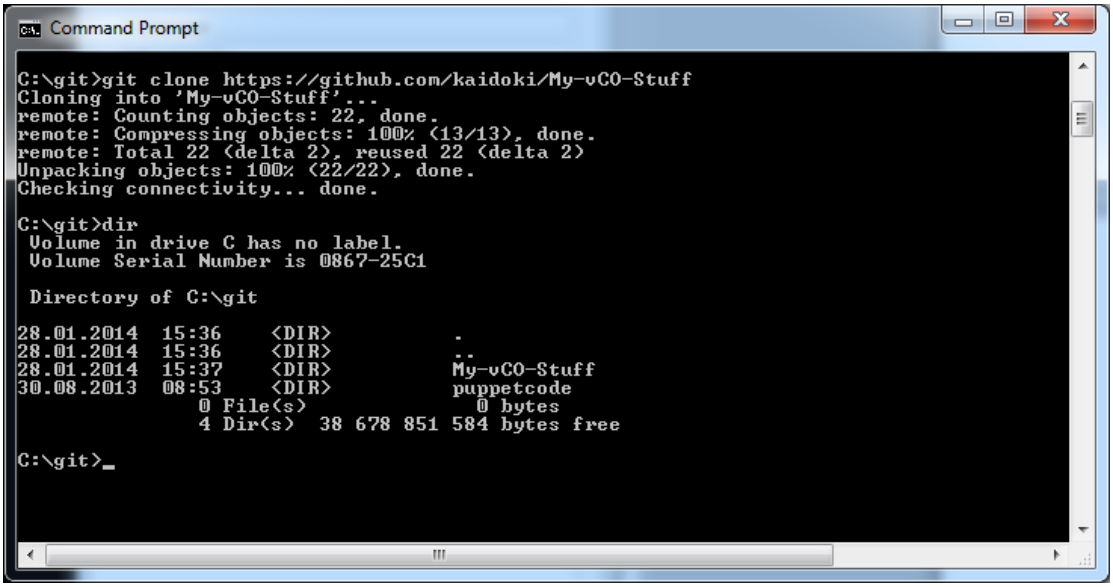
Easiest way to retrieve vCO workflows from the GitHub is to clone suitable remote repository locally to your client machine. Selected remote source repository must be created using vFLOWer Toolkit, otherwise it might not contain necessary folder structure nor required ANT scripts. Steps to retrieve vCO package source code from GitHub are the following:

1. Verify the content and the URL of the remote repository.
2. Open a command prompt and change working directory to a local Git repositories root directory.

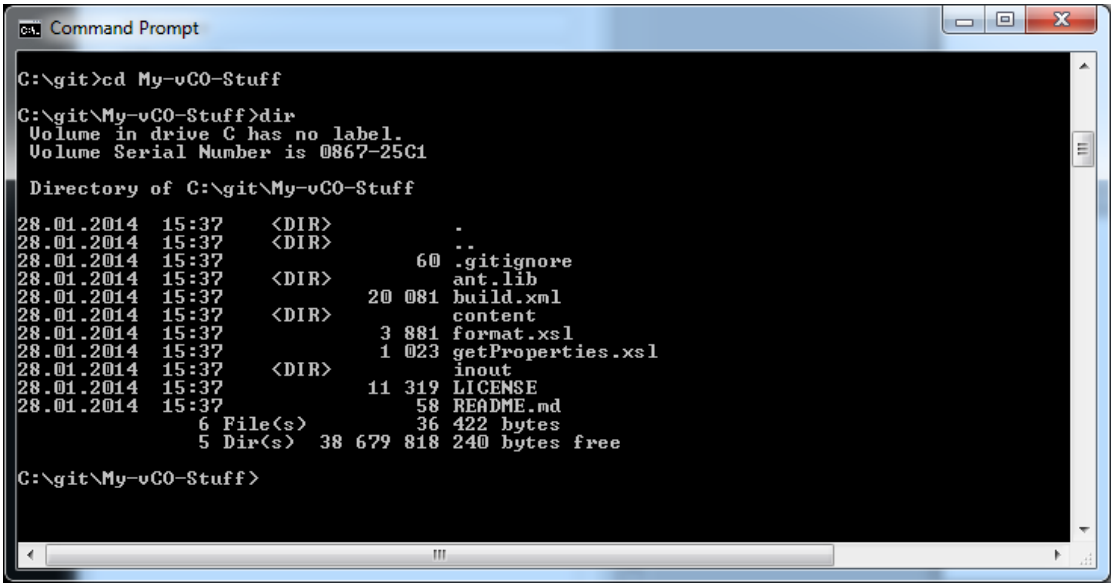


3. Clone the remote repository to a client machine using the following command:

```
git clone <GitHub repository URL>
```



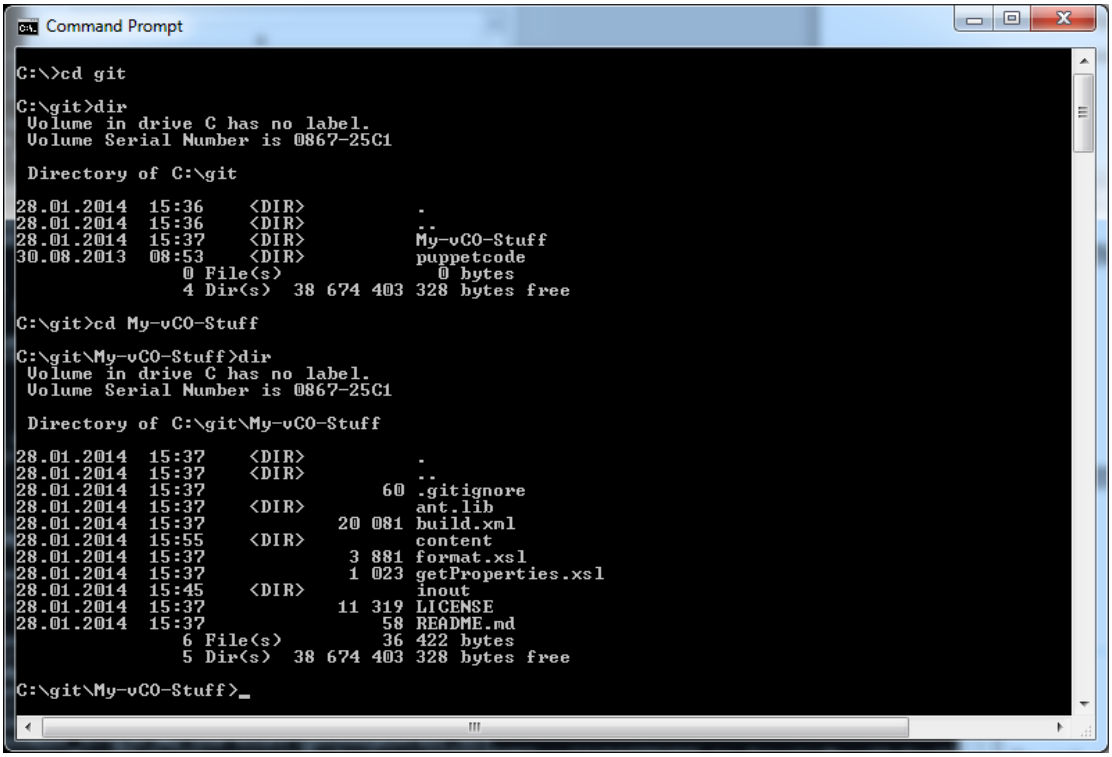
4. Verify your local repository's content.



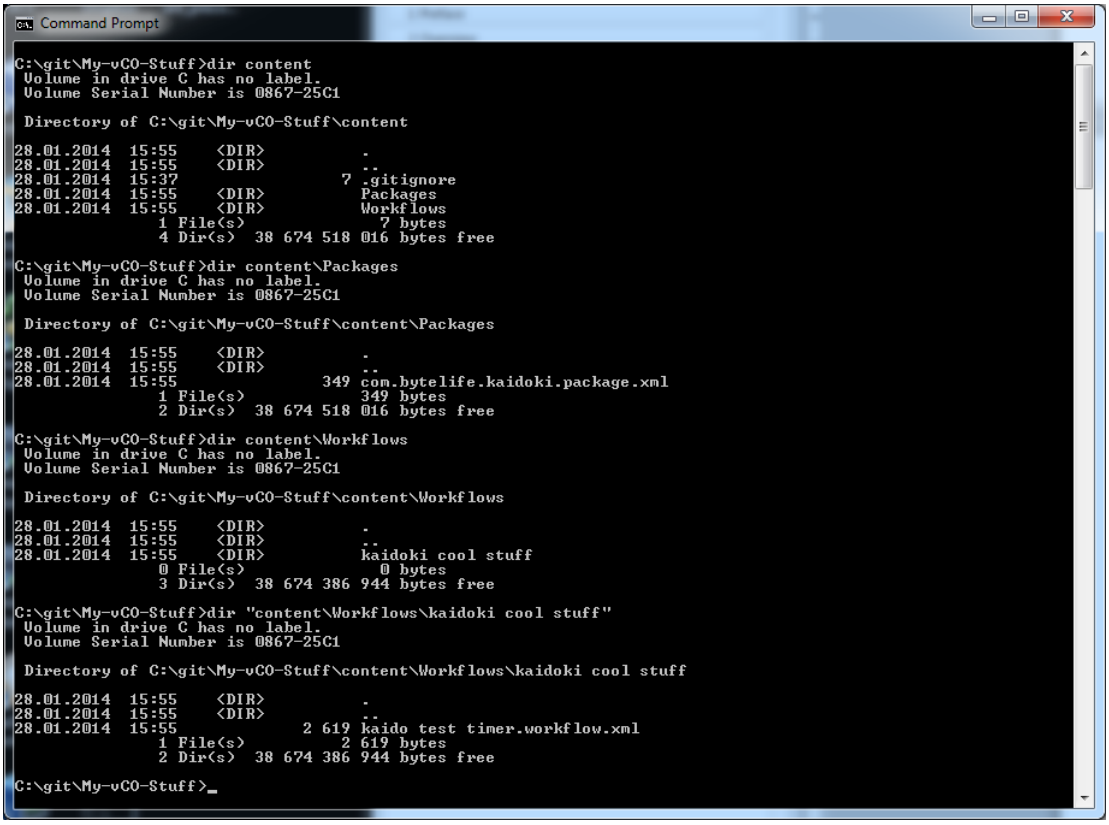
5.3 BUILD A VCO PACKAGE FROM XML SOURCE CODE

To build a binary vCO package from the downloaded XML-based source code a custom ANT script (provided by vFLOWer Toolkit) must be used. Steps to accomplish this are the following:

1. Open a command prompt and change to your local repository's root directory (location of the `build.xml` file).

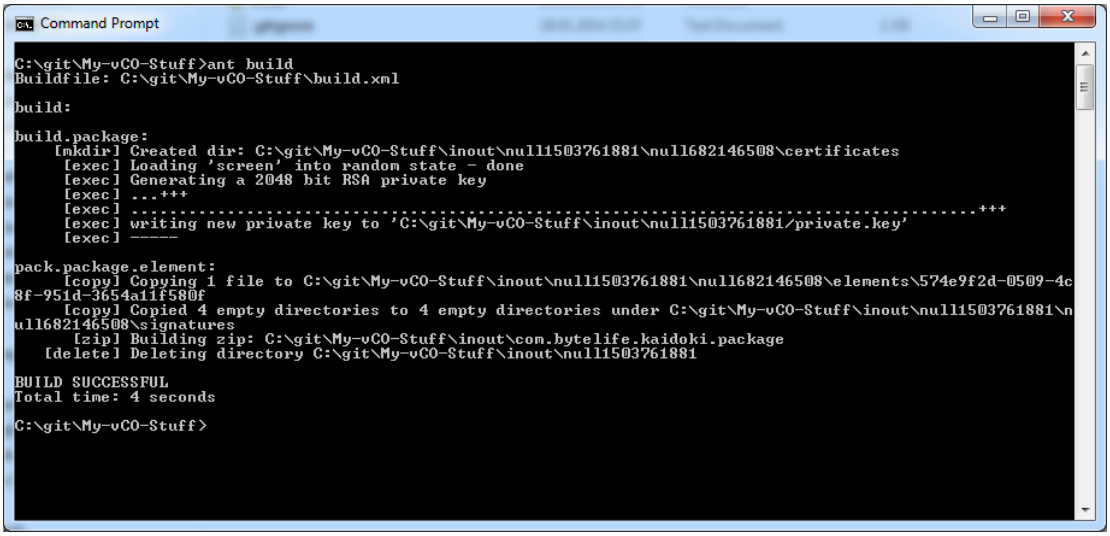


2. Verify the content of the repository's "content" directory. There must be new subfolders containing XML source files.

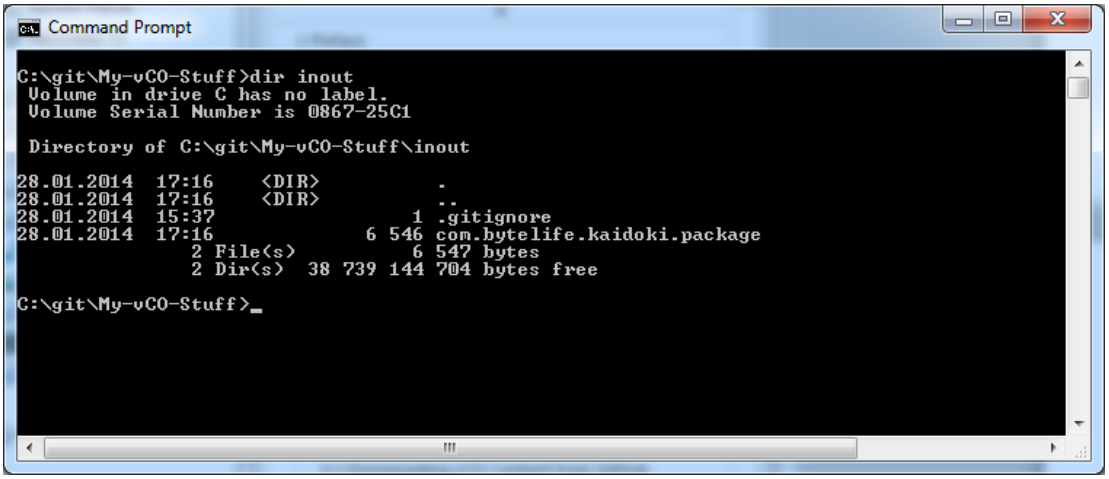


3. Launch build script by running the following command:

```
ant build
```



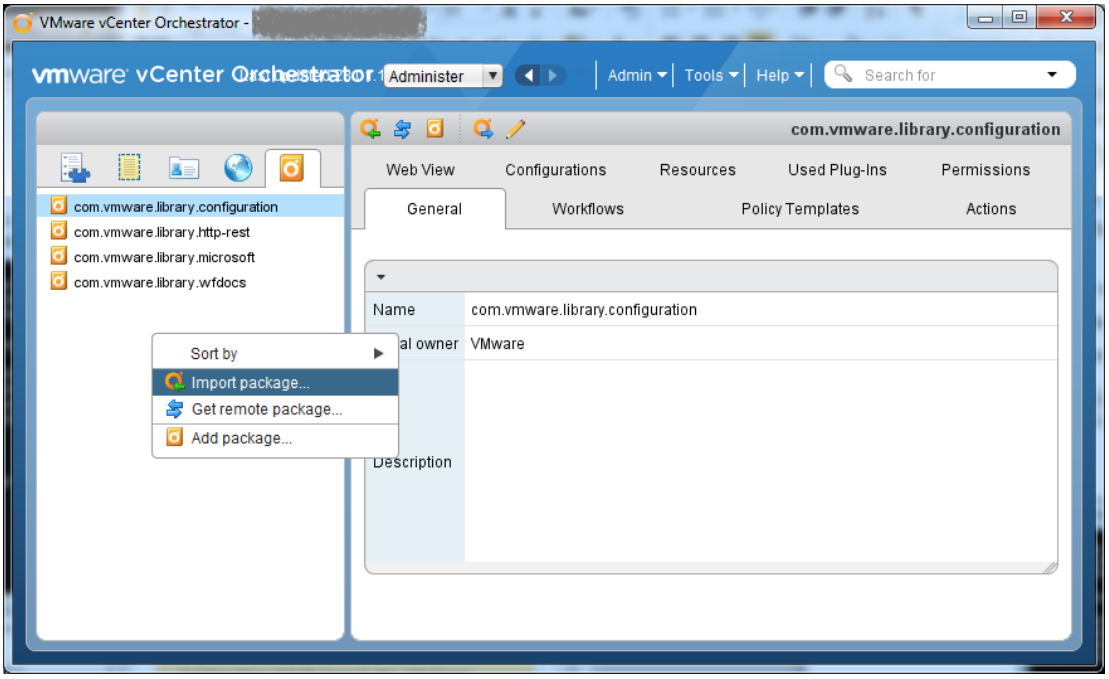
4. Verify that new vCO package was created to your repository's "inout" directory.



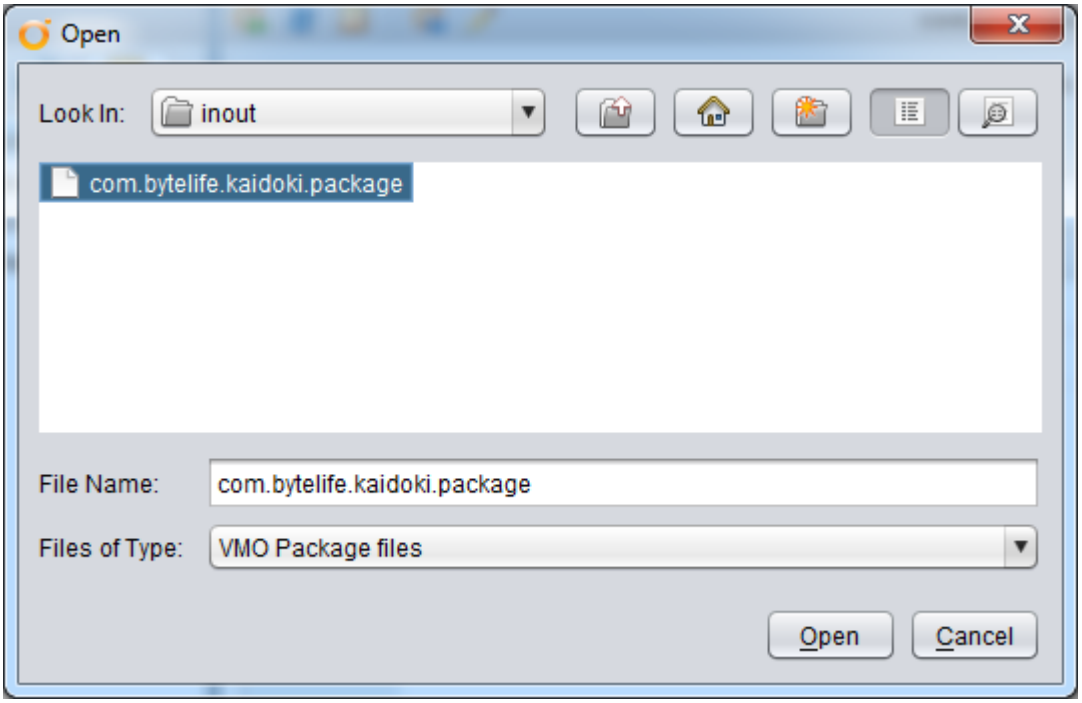
5.4 IMPORTING VCO PACKAGE TO THE VCO SERVER

Newly built vCO package must be imported using vCO Client. Steps to accomplish this are the following:

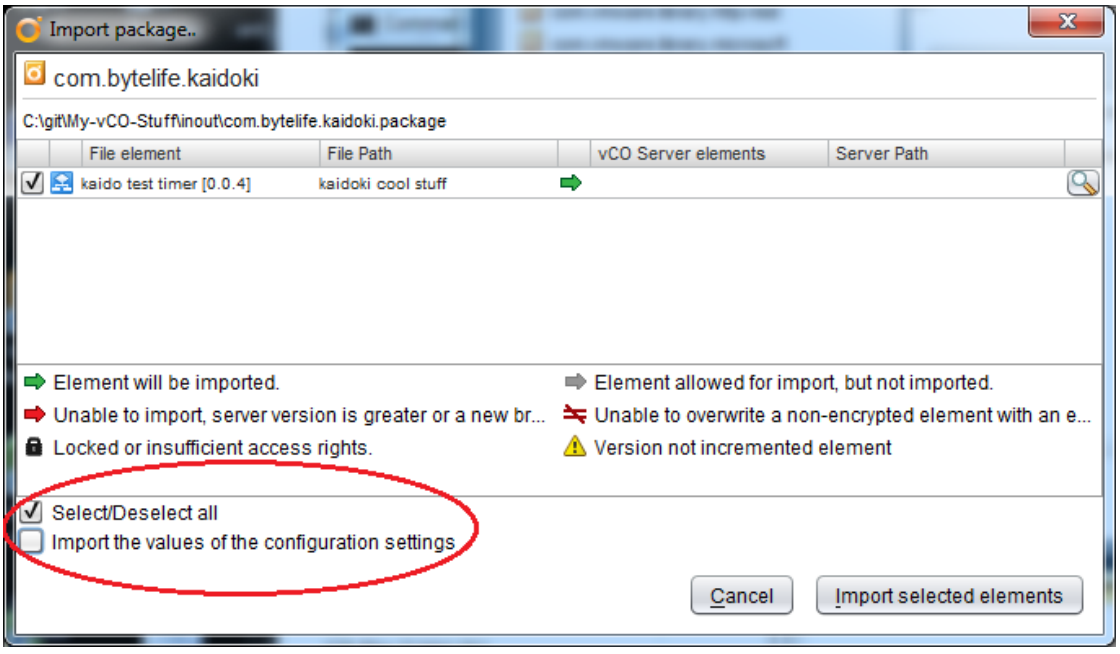
1. Open vCO Client and log into your vCO server and switch to “Administer” view.
2. Right-click on the empty space in the left panel and select “Import package…”.



3. Locate your vCO package from the repository’s input/output directory (“inout”).



4. Confirm the digital signature of the vCO package.
5. Select the package elements to be imported. Verify that "Select/Deselect All" checkbox is selected and "Import the values of the configuration settings" is de-selected.



6. Verify that package was successfully imported and start using workflows.