

vFLOWer Toolkit User Guide

Table of Contents

1	Preface	3
2	Overview	3
3	Pre-requisites.....	3
4	Publishing vCO content to GitHub.....	4
4.1	Summary Steps	4
4.2	User Account Creation in GitHub	4
4.3	New Repository Creation in GitHub	4
4.4	Cloning New Repository to a Client Machine	6
4.5	vCO Package Creation	8
4.6	Exporting vCO Package to the Git Repository.....	8
4.7	Unpack and Convert vCO Package Into XML	10
4.8	Committing and Pushing Content to GitHub	12
5	Retrieving vCO content from GitHub	15
5.1	Summary Steps	15
5.2	Downloading vCO Content from GitHub.....	15
5.3	Build a vCO Package from XML Source Code.....	17
5.4	Importing vCO Package to the vCO Server	20

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 for publishable content. New repository can be created by cloning (forking) existing ByteLife's vFLOWer repository which includes all needed ANT scripts and folder structure.
3. New GitHub repository cloned to the client machine (git clone).
4. A vCO package created using vCO client to include all needed content.
5. The package exported to the client machine into specified input/output directory (part of repository folder structure) using vCO client.
6. ByteLife's ANT precommit script launched from the repository root directory to unpack/convert the vCO package into XML-based source code.
7. New content committed and pushed to the remote GitHub repository (git commit, git push).

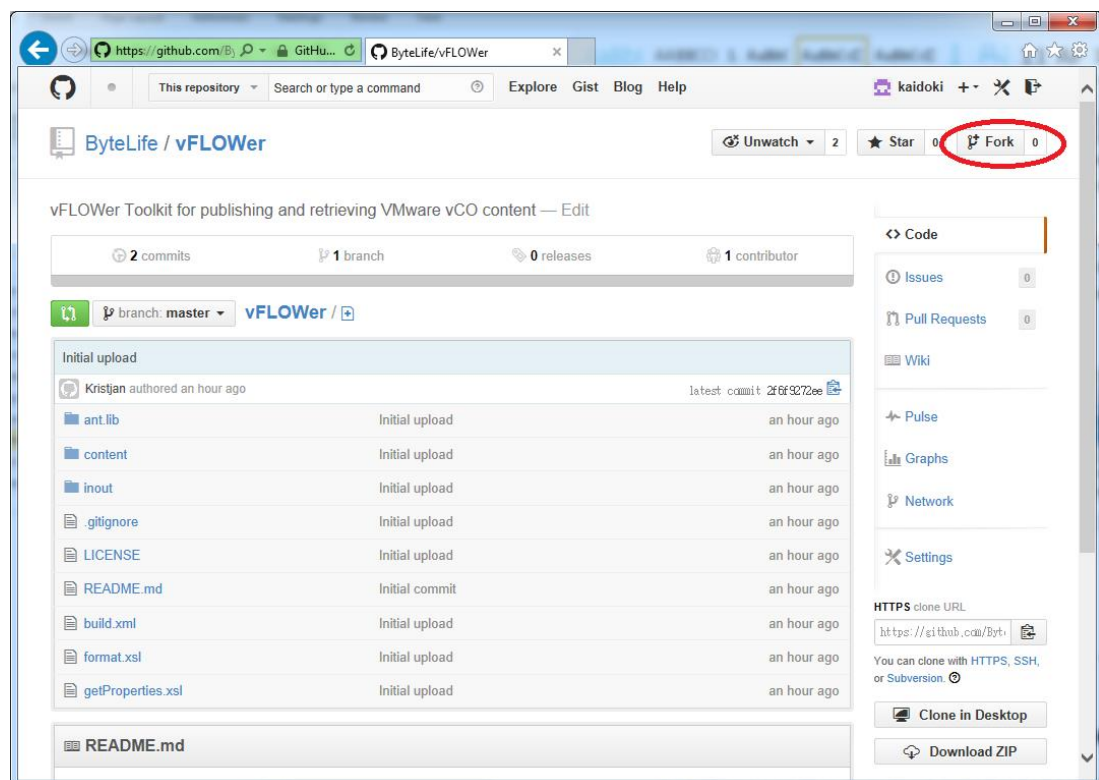
4.2 USER ACCOUNT CREATION IN GITHUB

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

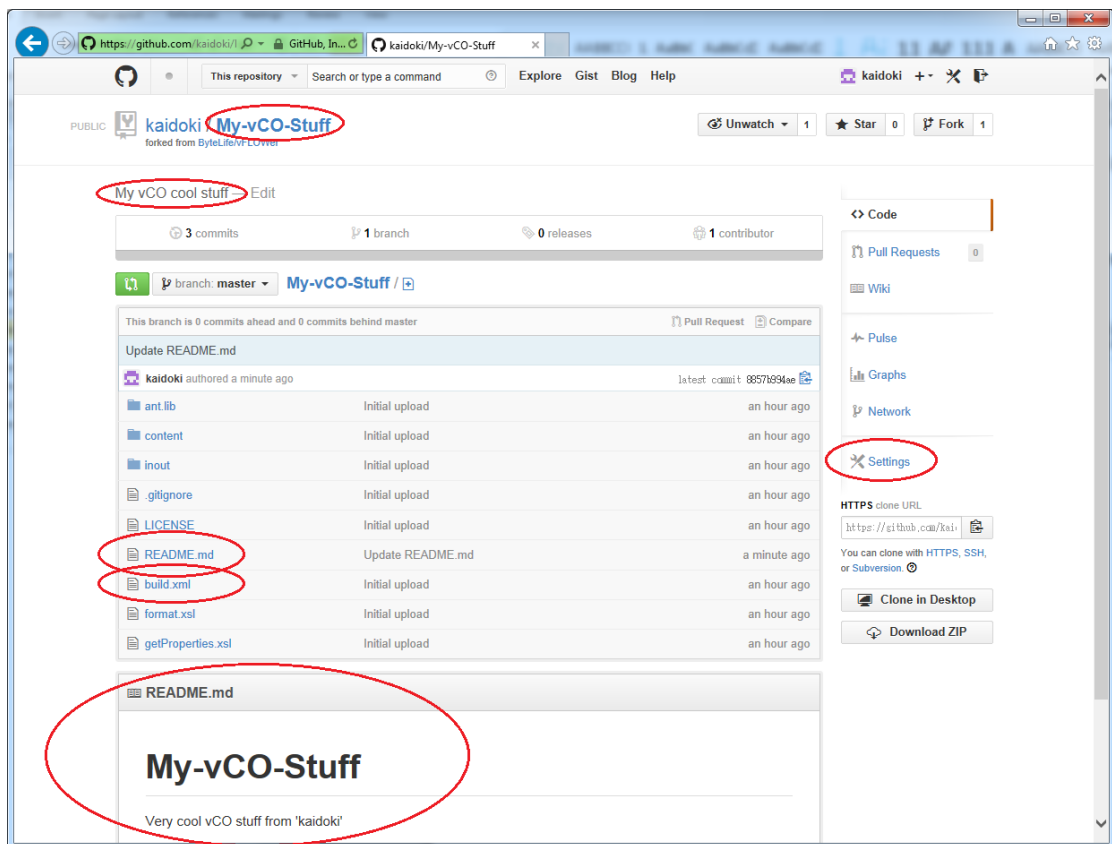
4.3 NEW REPOSITORY CREATION IN GITHUB

Easiest way to create new repository in GitHub having all needed content in place (folder structure and ANT scripts) is to fork existing ByteLife's repository called vFLOWer. Steps to accomplish this are the following:

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.



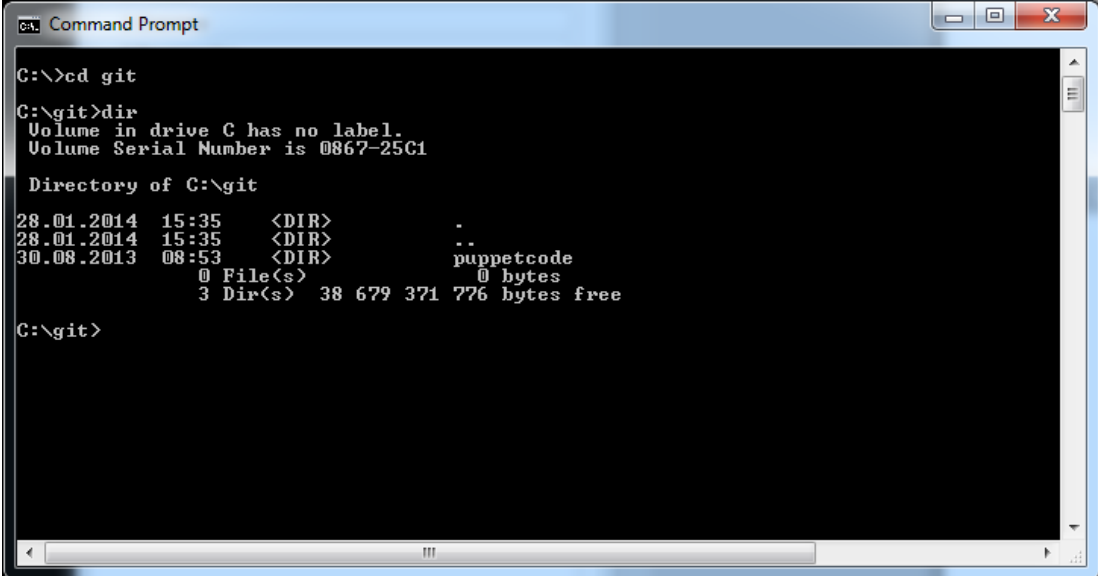
4. Select an account under which to create this new repository (you might have multiple accounts visible there).
5. Customize your new repository:
 - a. change repository name and description
 - b. change repository settings
 - c. edit README.md file to add your information
 - d. edit build.xml file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



4.4 CLONING NEW REPOSITORY TO A CLIENT MACHINE

To add your own content the newly created remote repository a local copy must be created to a client machine. Steps to accomplish this are the following:

1. Open a command prompt and change working directory to a local Git repositories root directory.



```
C:\>cd git
C:\git>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

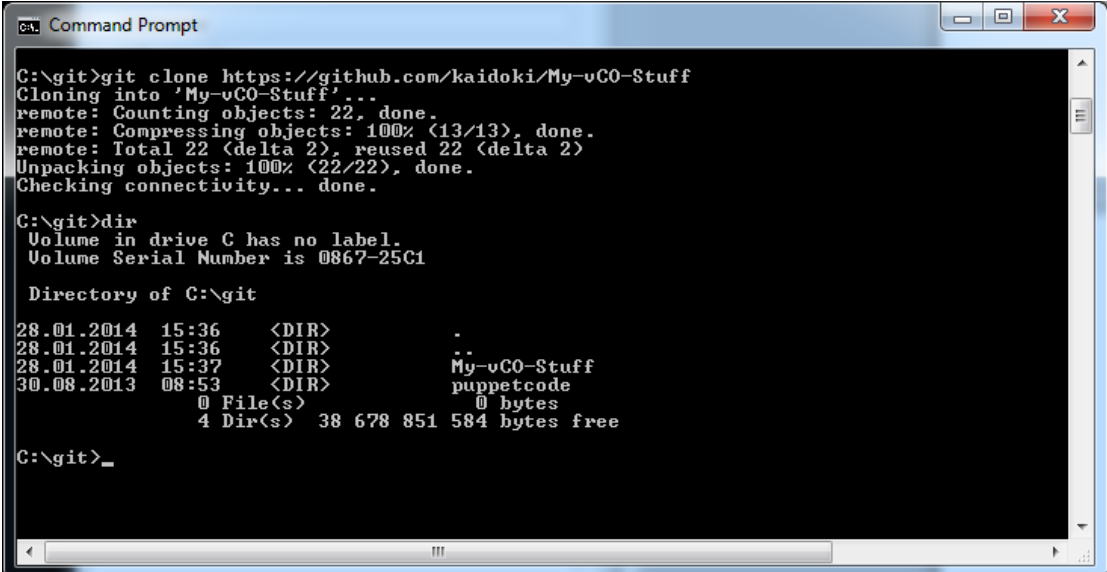
Directory of C:\git

28.01.2014  15:35    <DIR>          .
28.01.2014  15:35    <DIR>          ..
30.08.2013  08:53    <DIR>          puppetcode
               0 File(s)                0 bytes
               3 Dir(s)  38 679 371 776 bytes free

C:\git>
```

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

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



```
C:\git>git clone https://github.com/kaidoki/My-vCO-Stuff
Cloning into 'My-vCO-Stuff'...
remote: Counting objects: 22, done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 22 (delta 2), reused 22 (delta 2)
Unpacking objects: 100% (22/22), done.
Checking connectivity... done.

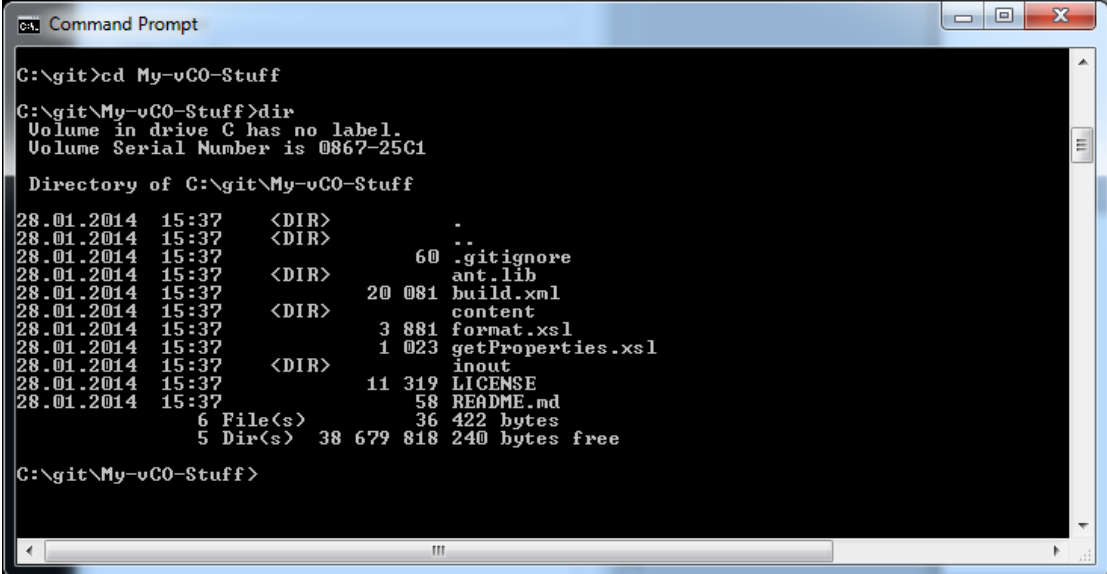
C:\git>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git

28.01.2014  15:36    <DIR>          .
28.01.2014  15:36    <DIR>          ..
28.01.2014  15:37    <DIR>          My-vCO-Stuff
30.08.2013  08:53    <DIR>          puppetcode
               0 File(s)                0 bytes
               4 Dir(s)  38 678 851 584 bytes free

C:\git>_
```

3. Verify your local repository's content.



```
C:\git>cd My-vCO-Stuff

C:\git\My-vCO-Stuff>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff

28.01.2014  15:37    <DIR>          .
28.01.2014  15:37    <DIR>          ..
                60 .gitignore
                ant.lib
28.01.2014  15:37    <DIR>          build.xml
                20 081 build.xml
28.01.2014  15:37    <DIR>          content
                3 881 format.xsl
28.01.2014  15:37    <DIR>          getProperties.xsl
                1 023 getProperties.xsl
28.01.2014  15:37    <DIR>          inout
                11 319 LICENSE
28.01.2014  15:37    <DIR>          README.md
                58 README.md
                6 File(s)          36 422 bytes
                5 Dir(s)        38 679 818 240 bytes free

C:\git\My-vCO-Stuff>
```

4.5 VCO PACKAGE CREATION

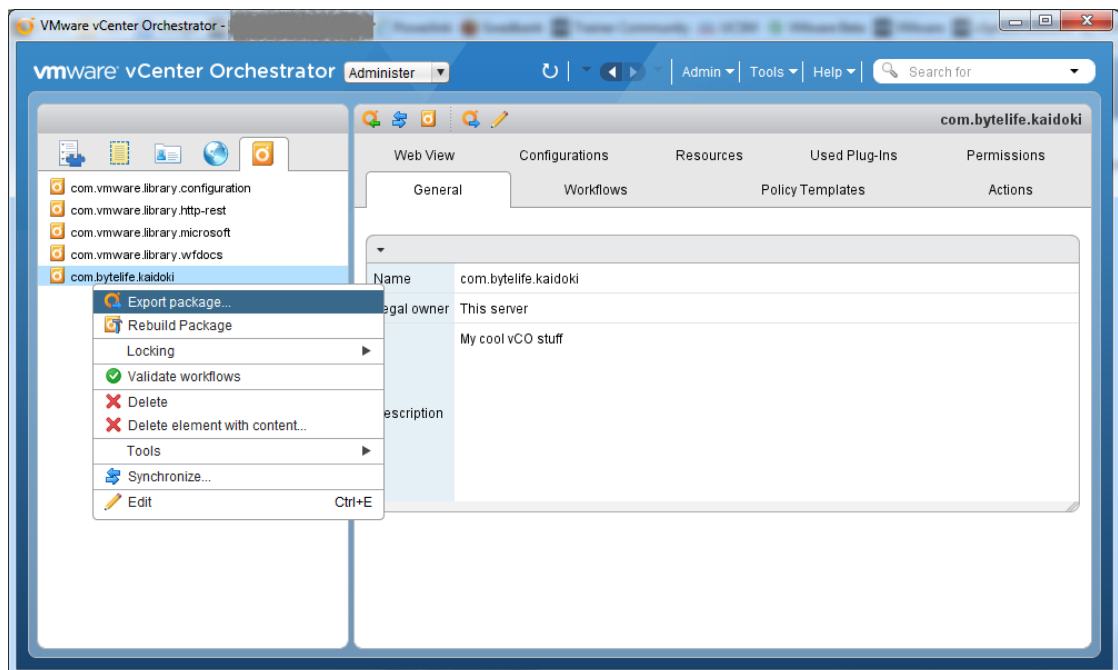
vCO package must be created manually using vCO Client. Package must contain all content needed by your vCO workflows. Follow the vCO documentation regarding package creation details available on VMware website:

https://www.vmware.com/support/pubs/orchestrator_pubs.html

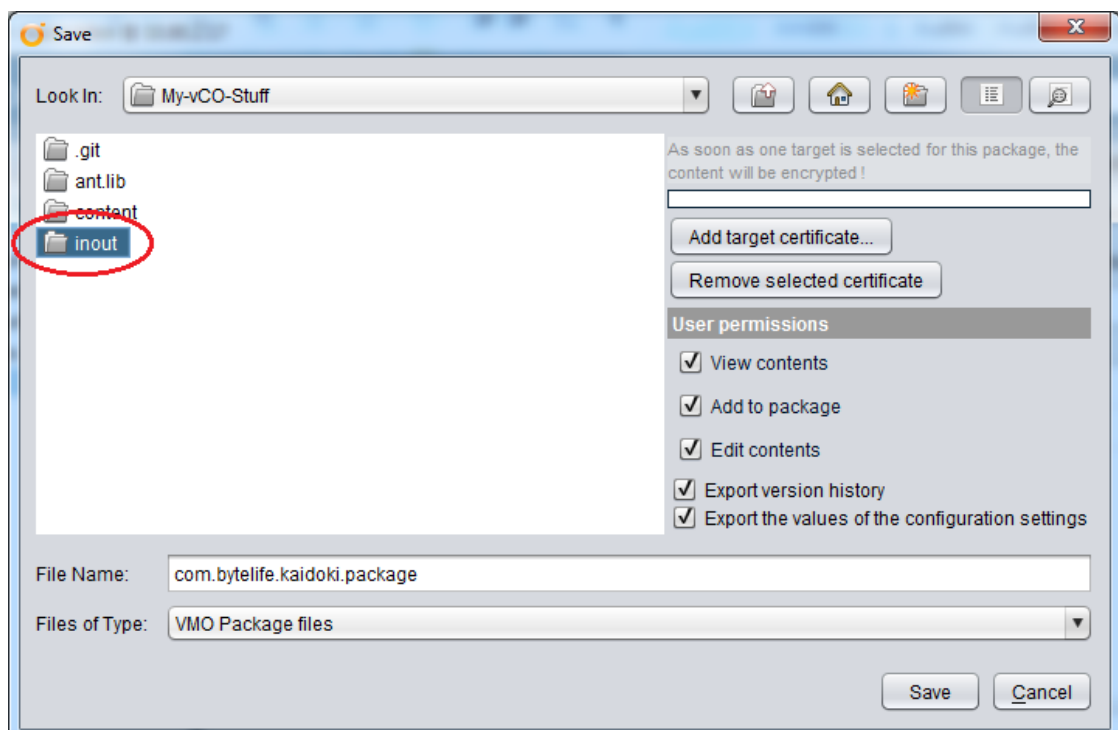
4.6 EXPORTING VCO PACKAGE TO THE GIT REPOSITORY

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...".



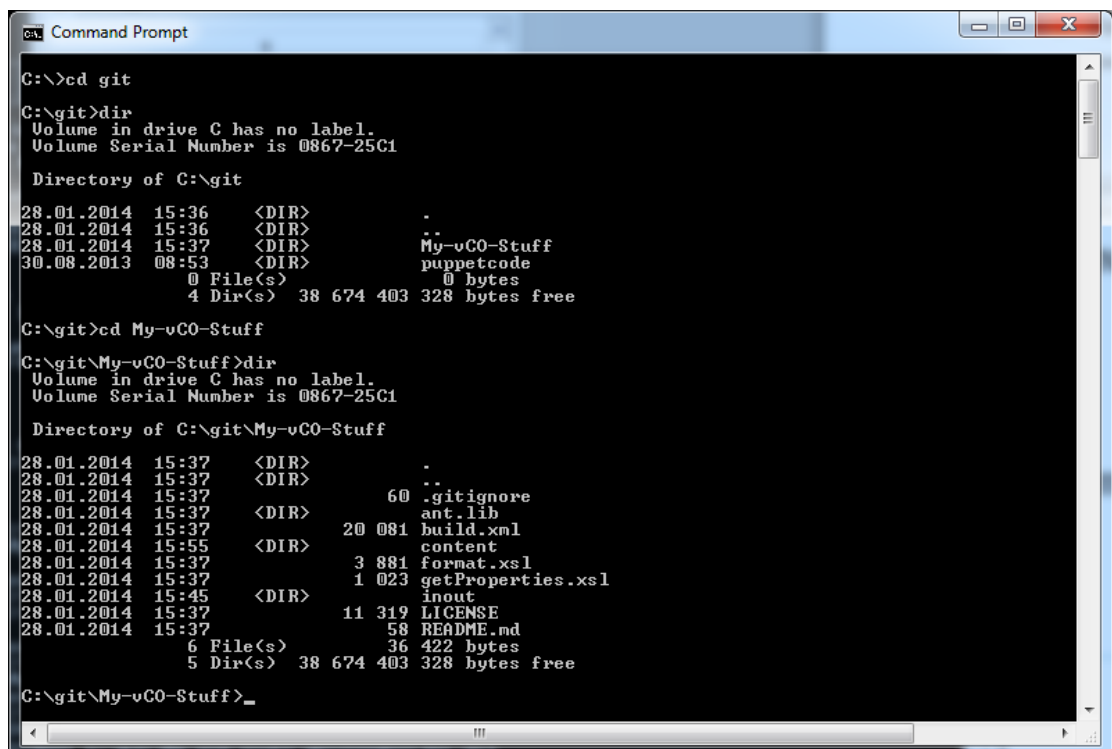
3. Save your package to local repository's input/output directory ("inout").



4.7 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).



```
C:\>cd git
C:\git>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git

28.01.2014  15:36    <DIR>        .
28.01.2014  15:36    <DIR>        ..
28.01.2014  15:37    <DIR>        My-vCO-Stuff
30.08.2013  08:53    <DIR>        puppetcode
               0 File(s)              0 bytes
               4 Dir(s)  38 674 403 328 bytes free

C:\git>cd My-vCO-Stuff
C:\git\My-vCO-Stuff>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff

28.01.2014  15:37    <DIR>        .
28.01.2014  15:37    <DIR>        ..
28.01.2014  15:37                60 .gitignore
28.01.2014  15:37                ant.lib
28.01.2014  15:37    <DIR>        20 081 build.xml
28.01.2014  15:37                content
28.01.2014  15:37                3 881 format.xml
28.01.2014  15:37                1 023 getProperties.xml
28.01.2014  15:37    <DIR>        inout
28.01.2014  15:37                11 319 LICENSE
28.01.2014  15:37                58 README.md
               6 File(s)              36 422 bytes
               5 Dir(s)  38 674 403 328 bytes free

C:\git\My-vCO-Stuff>
```

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

```
ant precommit
```

```

C:\git\My-vCO-Stuff>dir inout
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\inout

28.01.2014  15:45    <DIR>          .
28.01.2014  15:45    <DIR>          ..
28.01.2014  15:37                1 .gitignore
28.01.2014  15:54             10 046 com.bytelife.kaidoki.package
                2 File(s)              10 047 bytes
                2 Dir(s)  38 676 856 832 bytes free

C:\git\My-vCO-Stuff>ant precommit
Buildfile: C:\git\My-vCO-Stuff\build.xml

precommit:

unpack.package:
  [mkdir] Created dir: C:\git\My-vCO-Stuff\content\Packages\null144347482\null1637765576
  [unzip] Expanding: C:\git\My-vCO-Stuff\inout\com.bytelife.kaidoki.package into C:\git\My-vCO-Stuff\content\Packages\null144347482
unpack.package.element:
  [xslt] Processing C:\git\My-vCO-Stuff\content\Packages\null144347482\elements\574e9f2d-0509-4c8f-951d-3654a11f580f\info to C:\git\My-vCO-Stuff\content\Packages\null144347482\elements\574e9f2d-0509-4c8f-951d-3654a11f580f\null1893136602
  [xslt] Loading stylesheet C:\git\My-vCO-Stuff\getProperties.xsl
  [xslt] Processing C:\git\My-vCO-Stuff\content\Packages\null144347482\elements\574e9f2d-0509-4c8f-951d-3654a11f580f\categories to C:\git\My-vCO-Stuff\content\Packages\null144347482\elements\574e9f2d-0509-4c8f-951d-3654a11f580f\null1400742602
  [xslt] Loading stylesheet C:\git\My-vCO-Stuff\getProperties.xsl
  [xslt] Processing C:\git\My-vCO-Stuff\content\Packages\null144347482\elements\574e9f2d-0509-4c8f-951d-3654a11f580f\data to C:\git\My-vCO-Stuff\content\Workflows\kaidoki cool stuff\kaido test timer.workflow.xml
  [xslt] Loading stylesheet C:\git\My-vCO-Stuff\format.xsl
  [xslt] Processing C:\git\My-vCO-Stuff\content\Packages\null144347482\dunes-meta-inf to C:\git\My-vCO-Stuff\content\Packages\null144347482\null1287948934
  [xslt] Loading stylesheet C:\git\My-vCO-Stuff\getProperties.xsl
  [xslt] Processing C:\git\My-vCO-Stuff\content\Packages\null144347482\null1944795025 to C:\git\My-vCO-Stuff\content\Packages\com.bytelife.kaidoki.package.xml
  [xslt] Loading stylesheet C:\git\My-vCO-Stuff\format.xsl
  [delete] Deleting: C:\git\My-vCO-Stuff\inout\com.bytelife.kaidoki.package
  [delete] Deleting directory C:\git\My-vCO-Stuff\content\Packages\null144347482

BUILD SUCCESSFUL
Total time: 1 second
C:\git\My-vCO-Stuff>_

```

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

```

C:\git\My-vCO-Stuff>dir content
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:37       7 .gitignore
28.01.2014 15:55 <DIR>      Packages
28.01.2014 15:55 <DIR>      Workflows
                1 File(s)      7 bytes
                4 Dir(s)  38 674 518 016 bytes free

C:\git\My-vCO-Stuff>dir content\Packages
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content\Packages

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:55       349 com.hytelife.kaidoki.package.xml
                1 File(s)      349 bytes
                2 Dir(s)  38 674 518 016 bytes free

C:\git\My-vCO-Stuff>dir content\Workflows
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content\Workflows

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:55       kaidoki cool stuff
                0 File(s)      0 bytes
                3 Dir(s)  38 674 386 944 bytes free

C:\git\My-vCO-Stuff>dir "content\Workflows\kaidoki cool stuff"
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content\Workflows\kaidoki cool stuff

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:55       2 619 kaido test timer.workflow.xml
                1 File(s)      2 619 bytes
                2 Dir(s)  38 674 386 944 bytes free

C:\git\My-vCO-Stuff>_

```

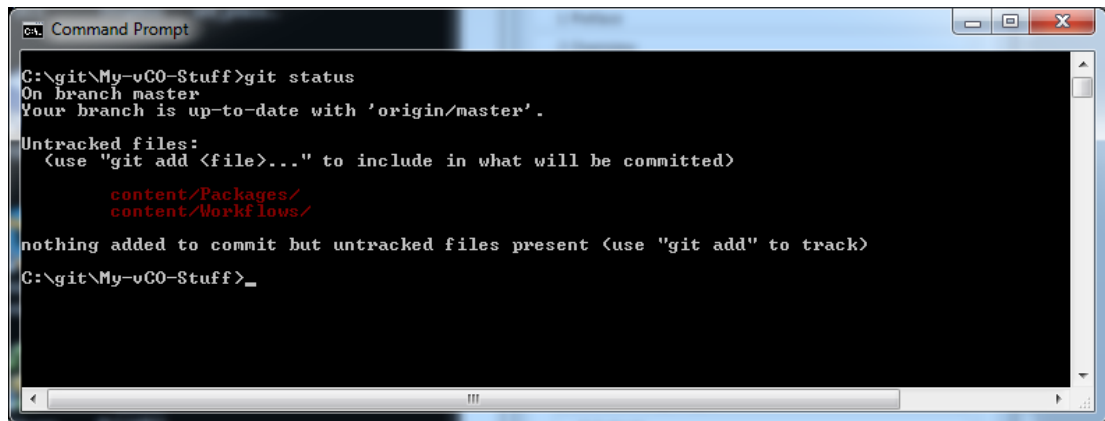
4.8 COMMITTING AND PUSHING CONTENT TO GITHUB

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.



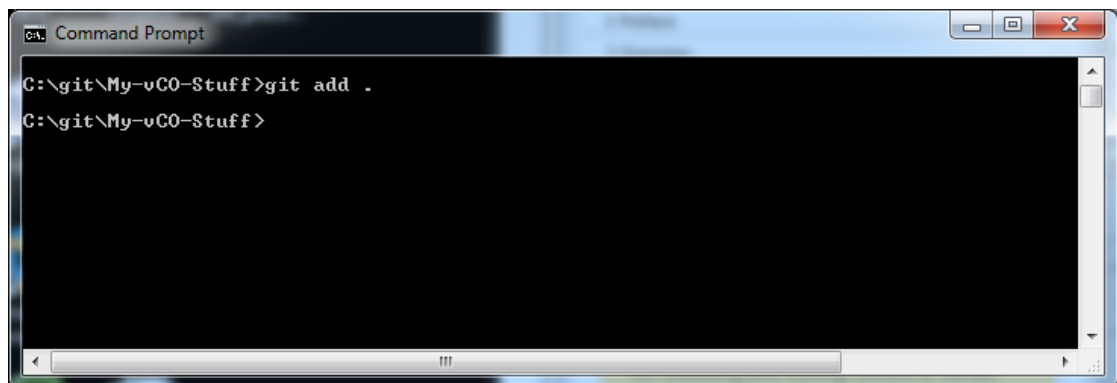
```
C:\git\My-vCO-Stuff>git status
On branch master
Your branch is up-to-date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        content/Packages/
        content/Workflows/

nothing added to commit but untracked files present (use "git add" to track)
C:\git\My-vCO-Stuff>
```

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

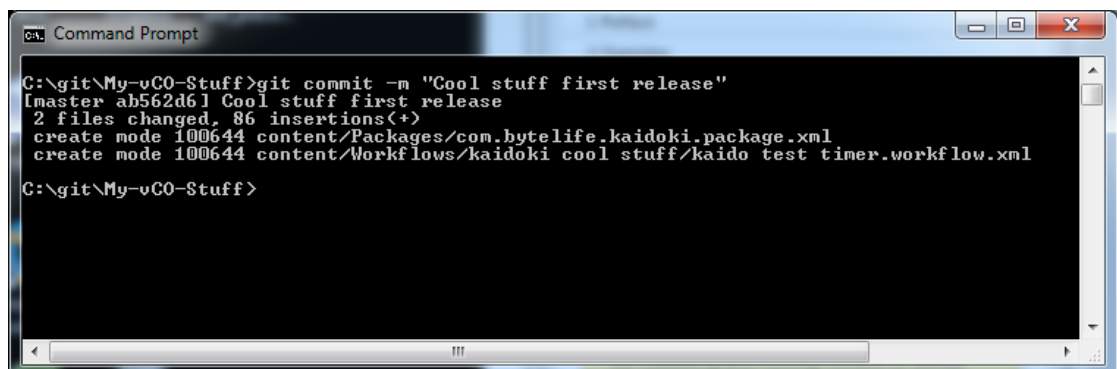
```
git add .
```



```
C:\git\My-vCO-Stuff>git add .
C:\git\My-vCO-Stuff>
```

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

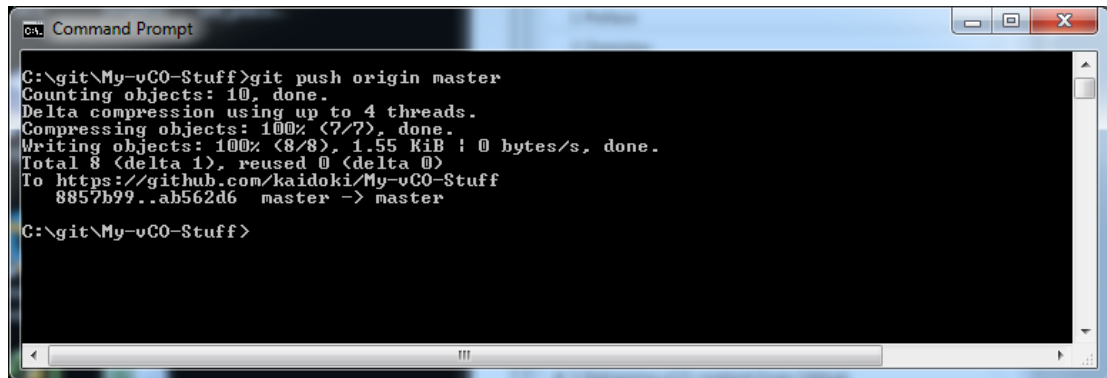
```
git commit -m <version comments>
```



```
C:\git\My-vCO-Stuff>git commit -m "Cool stuff first release"
[master ab562d61] Cool stuff first release
2 files changed, 86 insertions(+)
 create mode 100644 content/Packages/com.bytelife.kaidoki.package.xml
 create mode 100644 content/Workflows/kaidoki cool stuff/kaido test timer.workflow.xml
C:\git\My-vCO-Stuff>
```

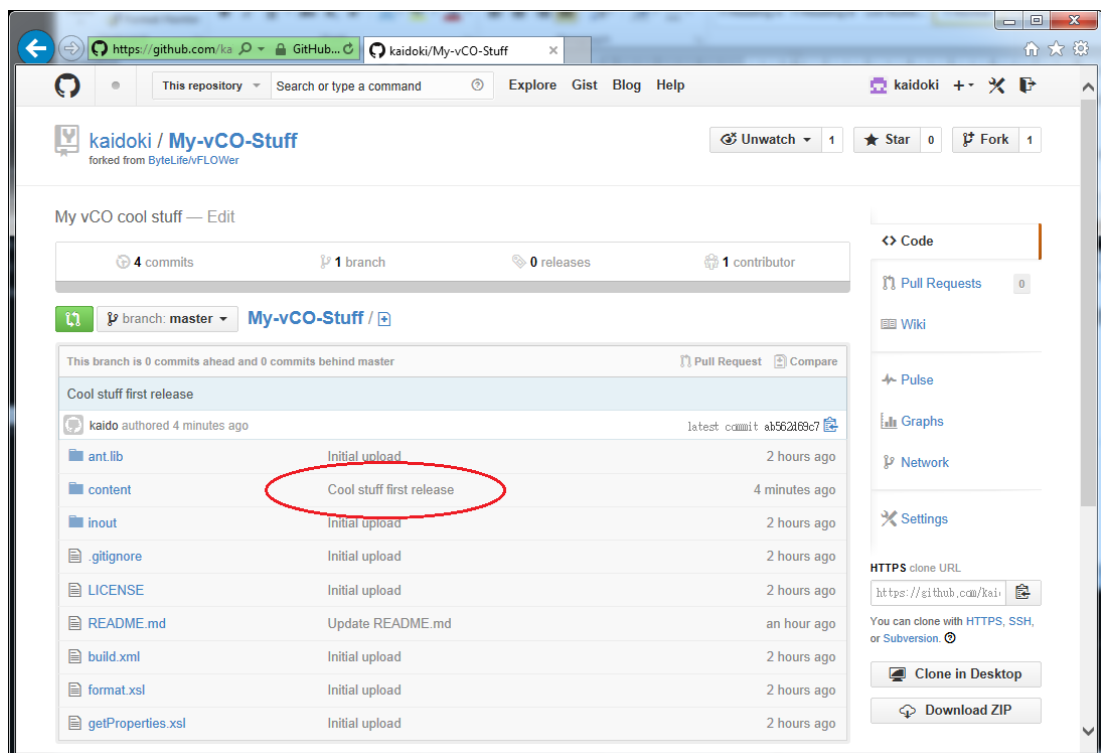
5. Push your changes to the GitHub repository using the following command:

```
git push origin master
```



```
C:\git\My-vCO-Stuff>git push origin master
Counting objects: 10, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 1.55 KiB | 0 bytes/s, done.
Total 8 (delta 1), reused 0 (delta 0)
To https://github.com/kaidoki/My-vCO-Stuff
8857b99..ab562d6 master -> master
C:\git\My-vCO-Stuff>
```

6. Verify your repository content in GitHub.



5 Retrieving vCO content from GitHub

5.1 SUMMARY STEPS

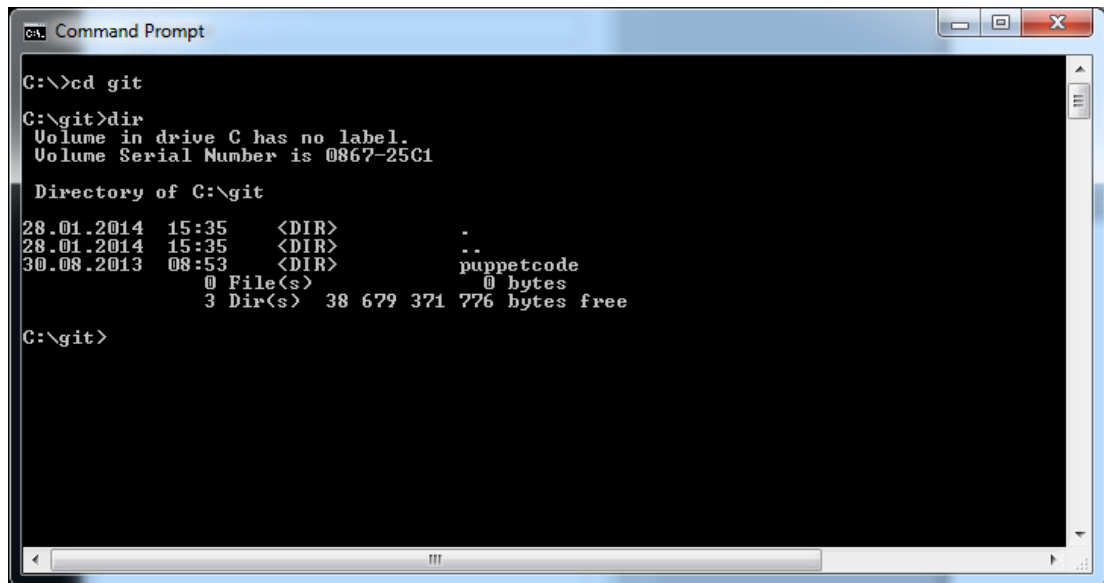
Multiple manual steps must be performed in order to retrieve vCO content from public GitHub repository. All steps can be performed from a client machine having all needed pre-requisite components installed.

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.



```
C:\>cd git
C:\git>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

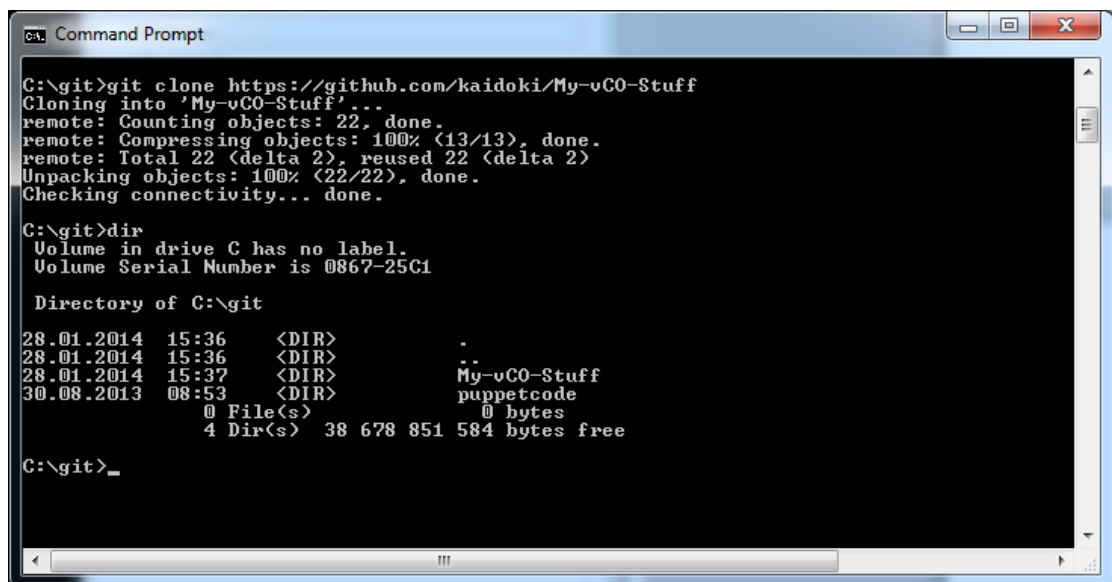
Directory of C:\git

28.01.2014  15:35    <DIR>          -
28.01.2014  15:35    <DIR>          ..
30.08.2013  08:53    <DIR>          puppetcode
               0 File(s)                0 bytes
               3 Dir(s)  38 679 371 776 bytes free

C:\git>
```

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

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



```
C:\git>git clone https://github.com/kaidoki/My-vCO-Stuff
Cloning into 'My-vCO-Stuff'...
remote: Counting objects: 22, done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 22 (delta 2), reused 22 (delta 2)
Unpacking objects: 100% (22/22), done.
Checking connectivity... done.

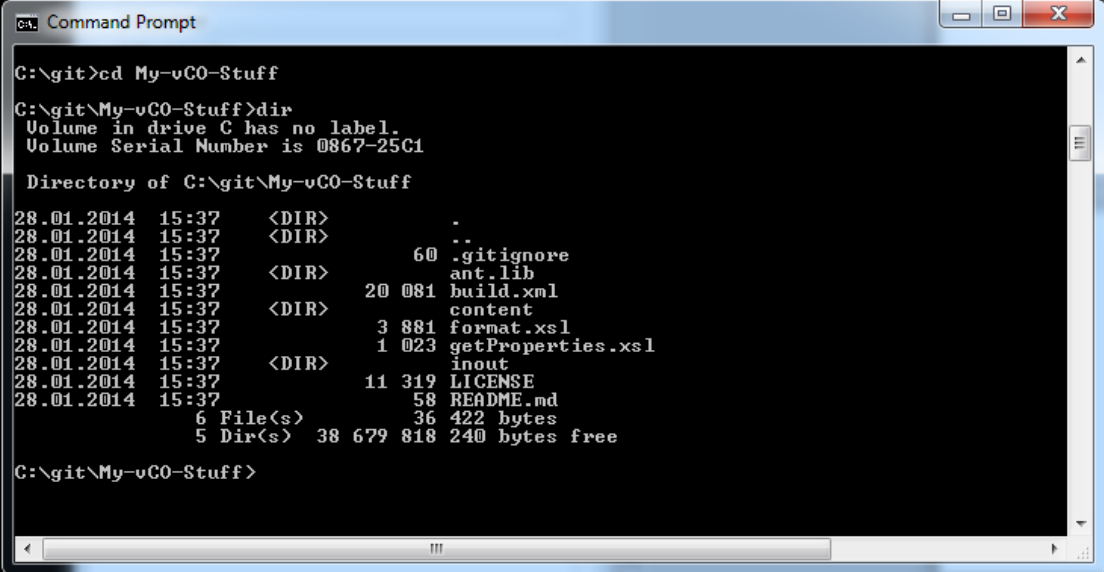
C:\git>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git

28.01.2014  15:36    <DIR>          -
28.01.2014  15:36    <DIR>          ..
28.01.2014  15:37    <DIR>          My-vCO-Stuff
30.08.2013  08:53    <DIR>          puppetcode
               0 File(s)                0 bytes
               4 Dir(s)  38 678 851 584 bytes free

C:\git>_
```

4. Verify your local repository's content.



```
C:\git>cd My-vCO-Stuff

C:\git\My-vCO-Stuff>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff

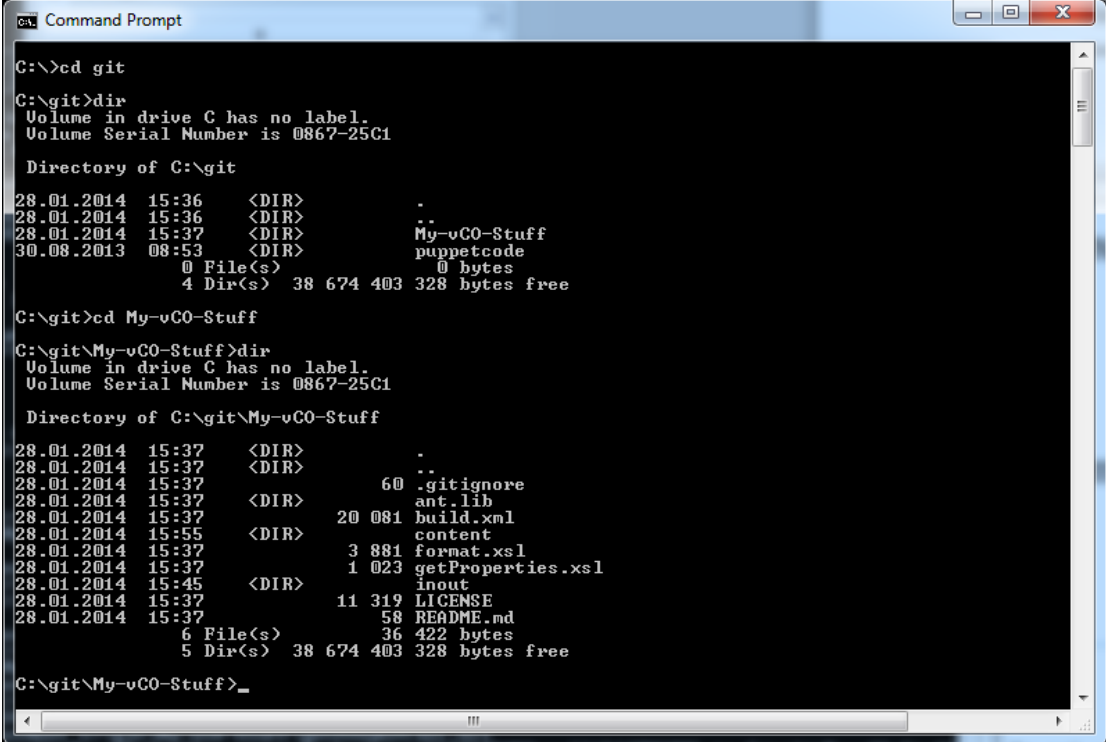
28.01.2014  15:37    <DIR>          .
28.01.2014  15:37    <DIR>          ..
28.01.2014  15:37             60 .gitignore
28.01.2014  15:37    <DIR>          ant.lib
28.01.2014  15:37       20 081 build.xml
28.01.2014  15:37    <DIR>          content
28.01.2014  15:37       3 881 format.xsl
28.01.2014  15:37       1 023 getProperties.xsl
28.01.2014  15:37    <DIR>          inout
28.01.2014  15:37       11 319 LICENSE
28.01.2014  15:37       58 58 README.md
                6 File(s)          36 422 bytes
                5 Dir(s)       38 679 818 240 bytes free

C:\git\My-vCO-Stuff>
```

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).



```
C:\>cd git
C:\git>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git
28.01.2014 15:36 <DIR>      .
28.01.2014 15:36 <DIR>      ..
28.01.2014 15:37 <DIR>      My-vCO-Stuff
30.08.2013 08:53 <DIR>      puppetcode
               0 File(s)      0 bytes
               4 Dir(s)  38 674 403 328 bytes free

C:\git>cd My-vCO-Stuff
C:\git\My-vCO-Stuff>dir
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff
28.01.2014 15:37 <DIR>      .
28.01.2014 15:37 <DIR>      ..
28.01.2014 15:37      60 .gitignore
28.01.2014 15:37 <DIR>      ant.lib
28.01.2014 15:37    20 081 build.xml
28.01.2014 15:55 <DIR>      content
28.01.2014 15:37    3 881 format.xsl
28.01.2014 15:37    1 023 getProperties.xsl
28.01.2014 15:45 <DIR>      inout
28.01.2014 15:37   11 319 LICENSE
28.01.2014 15:37    58 README.md
               6 File(s)      36 422 bytes
               5 Dir(s)  38 674 403 328 bytes free

C:\git\My-vCO-Stuff>_
```

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

```

C:\git\My-vCO-Stuff>dir content
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:37          7 .gitignore
28.01.2014 15:55 <DIR>      Packages
28.01.2014 15:55 <DIR>      Workflows
                1 File(s)          7 bytes
                4 Dir(s)  38 674 518 016 bytes free

C:\git\My-vCO-Stuff>dir content\Packages
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content\Packages

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:55          349 com.hytelife.kaidoki.package.xml
                1 File(s)        349 bytes
                2 Dir(s)  38 674 518 016 bytes free

C:\git\My-vCO-Stuff>dir content\Workflows
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content\Workflows

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:55          kaidoki cool stuff
                0 File(s)          0 bytes
                3 Dir(s)  38 674 386 944 bytes free

C:\git\My-vCO-Stuff>dir "content\Workflows\kaidoki cool stuff"
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\content\Workflows\kaidoki cool stuff

28.01.2014 15:55 <DIR>      .
28.01.2014 15:55 <DIR>      ..
28.01.2014 15:55          2 619 kaido test timer.workflow.xml
                1 File(s)        2 619 bytes
                2 Dir(s)  38 674 386 944 bytes free

C:\git\My-vCO-Stuff>_

```

3. Launch build script by running the following command:

```
ant build
```

```

C:\git\My-vCO-Stuff>ant build
Buildfile: C:\git\My-vCO-Stuff\build.xml

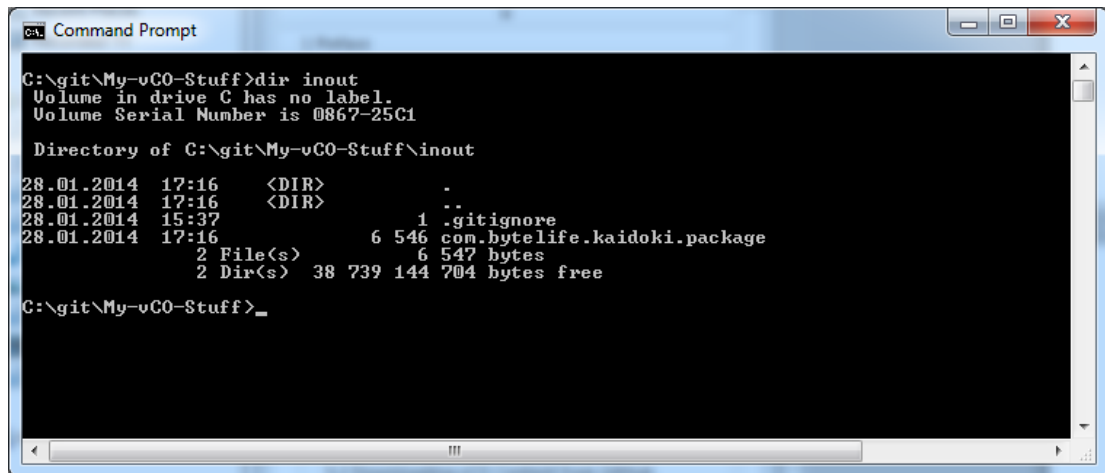
build:
build.package:
[mkdir] Created dir: C:\git\My-vCO-Stuff\inout\null1503761881\null1682146508\certificates
[exec] Loading 'screen' into random state - done
[exec] Generating a 2048 bit RSA private key
[exec] .....
[exec] .....+++
[exec] writing new private key to 'C:\git\My-vCO-Stuff\inout\null1503761881\private.key'
[exec] -----
pack.package.element:
[copy] Copying 1 file to C:\git\My-vCO-Stuff\inout\null1503761881\null1682146508\elements\574e9f2d-0509-4c
8f-951d-3654a1f580f
[copy] Copied 4 empty directories to 4 empty directories under C:\git\My-vCO-Stuff\inout\null1503761881\n
ull1682146508\signatures
[zip] Building zip: C:\git\My-vCO-Stuff\inout\com.hytelife.kaidoki.package
[delete] Deleting directory C:\git\My-vCO-Stuff\inout\null1503761881

BUILD SUCCESSFUL
Total time: 4 seconds

C:\git\My-vCO-Stuff>

```

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



```
Ca Command Prompt
C:\git\My-vCO-Stuff>dir inout
Volume in drive C has no label.
Volume Serial Number is 0867-25C1

Directory of C:\git\My-vCO-Stuff\inout

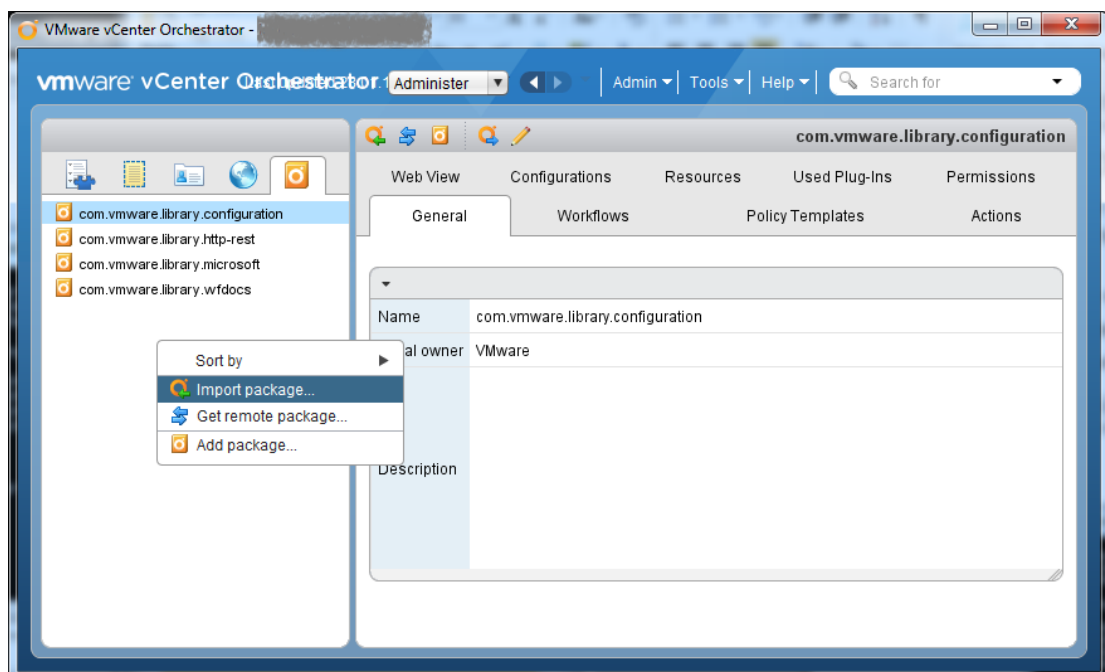
28.01.2014 17:16 <DIR>      .
28.01.2014 17:16 <DIR>      ..
28.01.2014 15:37             1 .gitignore
28.01.2014 17:16             6 546 com.bytelife.kaidoki.package
                2 File(s)          6 547 bytes
                2 Dir(s)      38 739 144 704 bytes free

C:\git\My-vCO-Stuff>
```

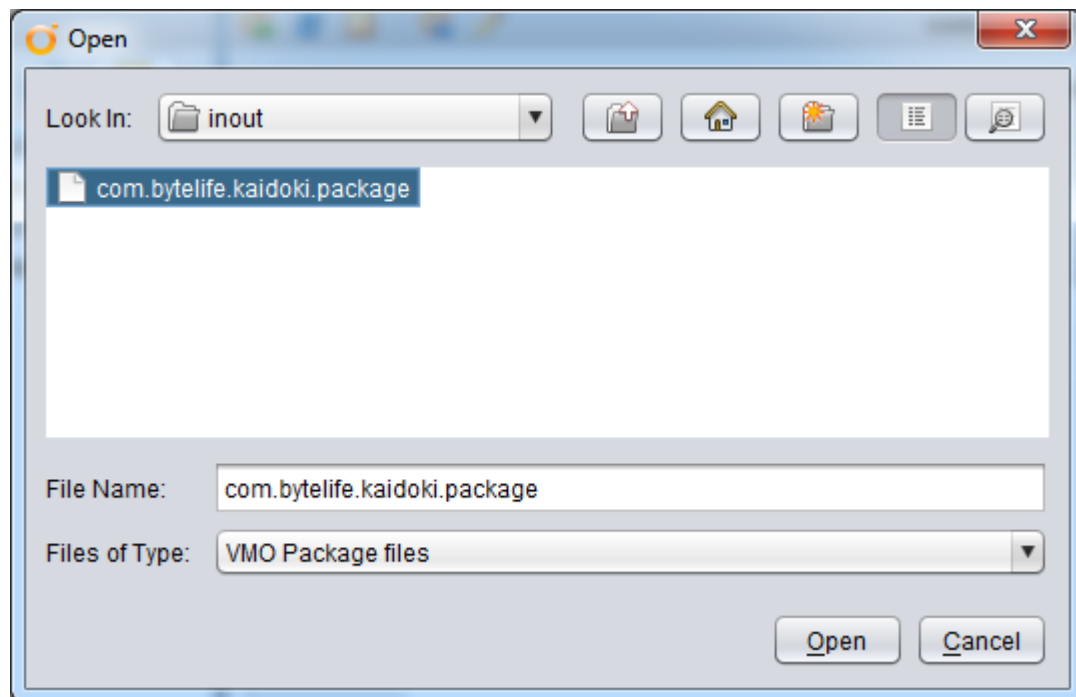
5.4 IMPORTING VCO PACKAGE TO THE VCO SERVER

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

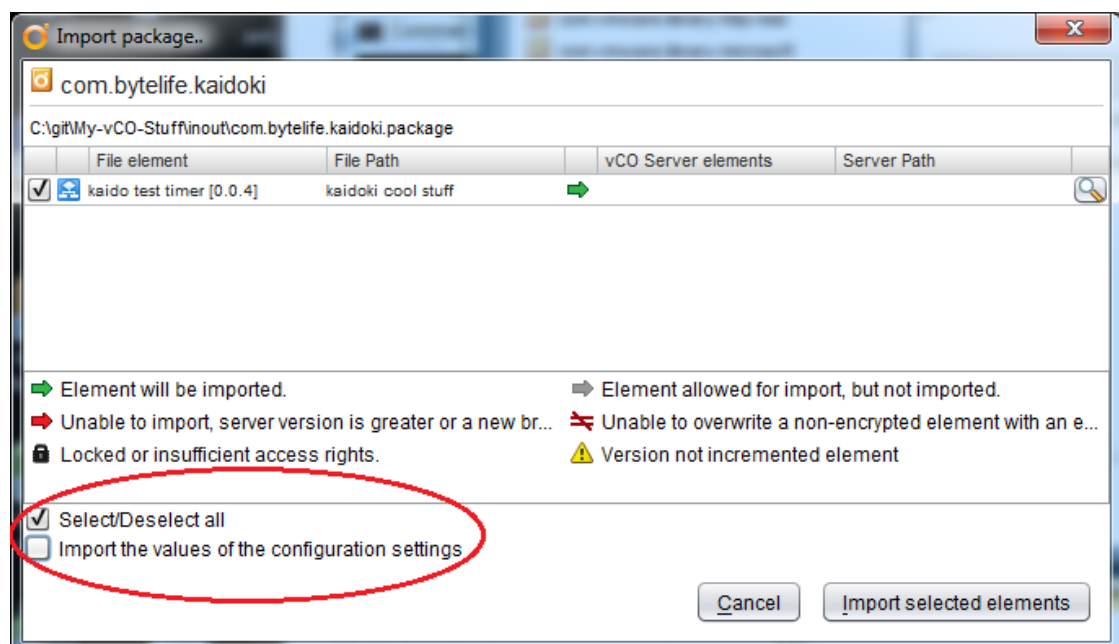
1. Open vCO Client and log into your vCO server and switch to "Administer" view.
2. Right-click on the empty space in the left panel and select "Import package..."



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.