

# UVersionControl Quick Guide

This document contains all the basic information needed in order to use Unity Version Control.

This project is open source and is maintained here:

<https://bitbucket.org/Kjems/uversioncontrol/overview>

## Features

- An open source Version Control(VC) integration into Unity using SVN as backend
- GUI locks to avoid accidentally modifying a resource without VC lock
- Overview window showing current status on relevant files in relation to VC
- Interactive icons on all assets in project- and hierarchy-view
- SceneView gui to quickly get the lock on the current open scene
- The concept of .meta files are abstracted away from the user

## Important Note

When using UVersionControl without Unity's Team License it is highly recommended to use SVN v1.7 on both Windows and Mac.

### *Windows*

On windows it is recommended to install Tortoise SVN 1.7+ with commandline enabled during the installation.

### *Mac / OSX*

It is recommended to install [MacPorts](#). With MacPorts installed do 'sudo port install subversion'. In Unity select the Settings item in UVC menu. Add `/opt/local/bin/` to the environment path.

## Requirements

- Commandline svn client. A version 1.7 client is recommended.
- An already existing SVN checkout with credentials cached
- Use following Unity guide on setting up version control:  
<http://unity3d.com/support/documentation/Manual/ExternalVersionControlSystemSupport>
- If you have a 'Unity Team License' then see 'Team Features' section at the bottom.

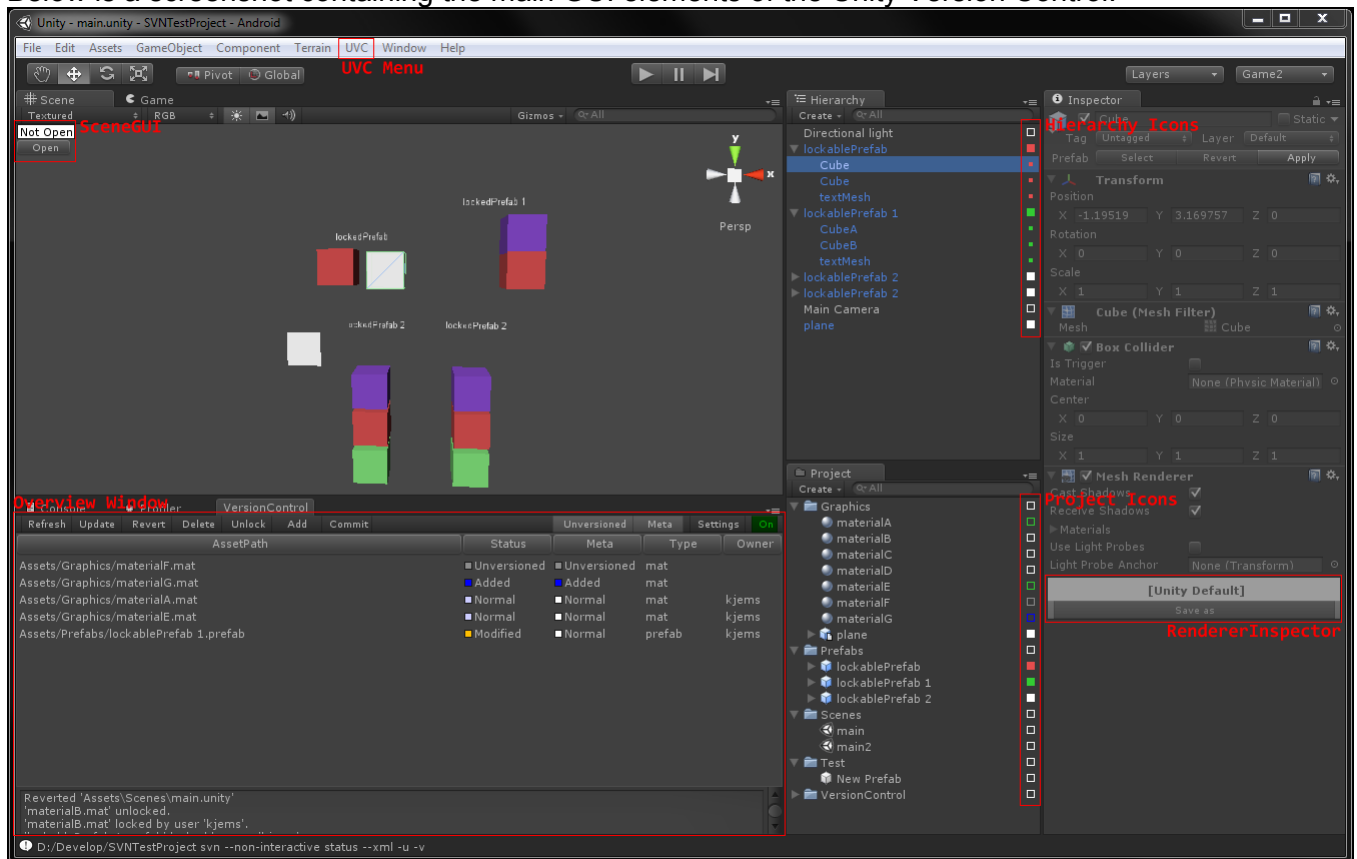
## Terminology

- **Open** : Try and get exclusive write access to a file
- **Open Here** : Having exclusive write access to a file
- **Open Local** : From this state it is only possible to 'Revert' or 'Open'
- **Opened by** : Someone else has exclusive access to the file
- **Refresh** : Refresh state from the server
- **Commit** : Upload the selected files to the server
- **Add** : Add files locally for a later 'Commit'
- **Revert** : Revert the file content to the state of the server
- **Update** : Get the newest version from the server

- **Unlock** : Release exclusive write access to a file
- **Delete** : Delete a file locally and when committed also on the server
- **Not in version control** : Have not yet been added and committed to the server
- **Diff** : Open a window showing the difference between a file locally and on the server

## GUI Overview

Below is a screenshot containing the main GUI elements of the Unity Version Control.



## SceneGUI

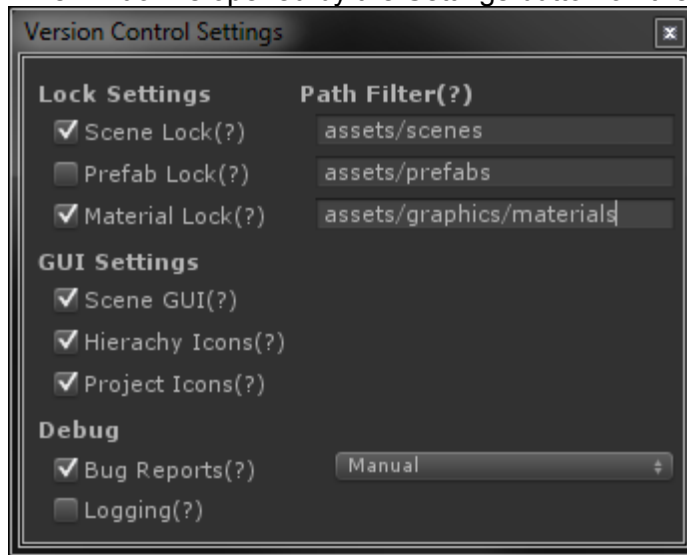
SceneGUI is an easy access to status and interactivity on the current open scene. The GUI is located in the upper left corner of the SceneView.

## Overview Window

The Overview Window provide a list of interesting files. The top menu bar have a set of commands which can be executed on the selected files in the listview. The top menu bar also contain a set of filters which can hide Unversioned files or files with only a modification in the associated .meta file. An On / Off button is placed on the right side of the top menu bar. In the bottom of the overview window is a live output from the version control backend as it performs the requested actions.

### *The Settings Window*

This window is opened by the Settings button on the top-right side of the overview window













In the settings window it can be selected which types of assets should be made read-only by the GUI depending on whether the user have exclusive access to the file or not. An assetpath filter can be added to each of the asset types, Scenes, Prefabs and Materials so only files containing the filter in the path is made read-only in the gui. The three GUI components SceneGUI, Hierarchy Icons and Project Icons can be toggled on and off separately.




### *Hierarchy- and Project Icons*

The hierarchy and project icons show the state of a file but when clicked also provide actions on the clicked file. The actions available are contextual so only actions which makes sense in the file current state is available.

### *Icon Colors*

-  Normal file status, as in not-modified but under version control
-  Unversioned which means not a part of the version control yet
-  Added but not yet committed to version control
-  Open, which means having exclusive write access
-  Open by someone else. Mouse over icon to see who
-  Open local, which means temporary access locally only
-  Missing on disk but still under version control
-  Deleted but not yet committed to version control
-  Changed on the server but still not downloaded locally
-  Conflict between the local file and the file on the version control server

### *Icon Shapes*

-  Hollow icon means a scene object
-  Filled icon means a prefab root
-  Small filled icon means a prefab child

### *Renderer Inspectors*

The GUI in the renderer is a shortcut to version control actions on the attached materials of the given renderer. As it varies from different projects whether it is a good idea to provide this GUI for the user and because it requires to override OnInspector for all renderer components it has been put in a separate assembly (RendererInspectors.dll) which can be deleted if the feature is unwanted. If used it is possible to add additional GUI logic to the inspectors by using the `RendererInspectorManager.AddInspector` static function.

### *UVC Menu*

The available Menu item are Refresh, Update, Overview Window, Report Bug and About.

### *Team features*

Unity provide special callbacks for users with a “Unity Team License” which this integration makes use of but they will only work if Team License is available. The only callback used is when a file is moved in project view. If Unity Team License is not owned an exception will be thrown by Unity when a file is moved. All Team License features has been put in a separate assembly (TeamFeatures.dll) and can simply be deleted if team license is not available. The version of UversionControl available from the asset store will have the TeamFeatures.dll renamed to TeamFeatures.dl\_ so the few with Team License can enable the features by renaming .dl\_ back to to .dll.