Easily Apply a Version Status to all Component Versions of a Snapshot

This document fully describes the mechanism for easily applying a version status to all the right component versions identified by a snapshot. This capability is key to being able to reliably apply a version status to all the right component versions so that when you promote a snapshot to a new environment you can be assured that it will pass the environment gates used to control entry of a new application version into an environment.

The UrbanCode deploy GUI currently requires you to visit each component version in turn to apply a version status. This is both time consuming and error prone, especially for more complex applications with many components.

Other solutions can automatically apply the version status, perhaps during deployment but most often there is some period between deployment and promotion of a snapshot into a new environment. This period is usually when testing / validation of some sort occurs before the decision is made to ether promote or abandon a snapshot. It is only when the decision is made that the status should be applied, otherwise the gate to the next environment is opened prematurely giving rise to the possibility of accidental deployment of untested / unapproved code.

Solution Outline

The solution consists of 3 components:

a. An Application process that can be called from the environment GUI like any other application process to kick off the application (or removal) of a given version status.
b. A generic process which is the vehicle for the actual status application or removal
c. A plugin which does all the hard work.

Solution Assets

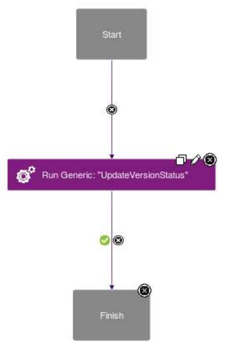
The solution is provided as a zip file which contains some scripts, a JSON file and another zip. This second zip is the UrbanCode plugin. You don't need to unzip this.

The outer zip also contains this document as a PDF.

Application Process

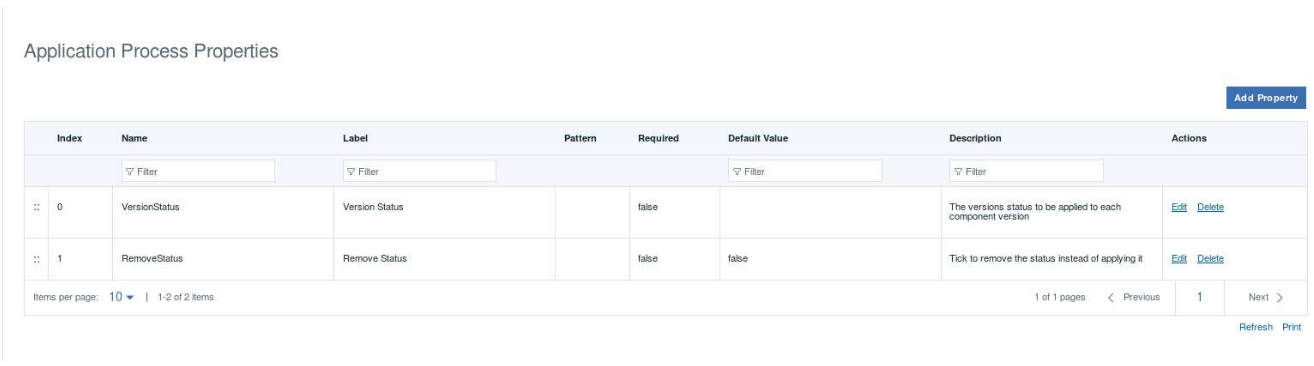
This is provided as an importable JSON file. It contains the necessary process to call the generic process and creates some Application Process properties to elicit information from the user. It can also establish a role to limit who is permitted to run the process.

The Application process is really all about collecting information on what we want to operate on. Much of the information required can be determined automatically from the context of the application process execution. This is the advantage of having an application process rather than running the generic process step directly.



The application process also needs some application process properties to help us request other information we need. The only step calls a generic process which in turn invokes the plugin step.

These are the properties that the process will have:


Application Process Properties

	Index	Name	Label	Pattern	Required	Default Value	Description	Actions
		Filter	Filter			Filter	Filter	
::	0	VersionStatus	Version Status		false		The versions status to be applied to each component version	Edit Delete
::	1	RemoveStatus	Remove Status		false	false	Tick to remove the status instead of applying it	Edit Delete

Items per page: 10 | 1-2 of 2 items
1 of 1 pages < Previous 1 Next >
Refresh Print

The VersionStatus property is an interesting one in that it uses an HTTP Select type field to enumerate the versions statuses available on the server. So, this field needs to know the server URL and also an AUTH Token to be able to make the API call. You may wish to restrict the scope of the token so that what you can do with it is limited.

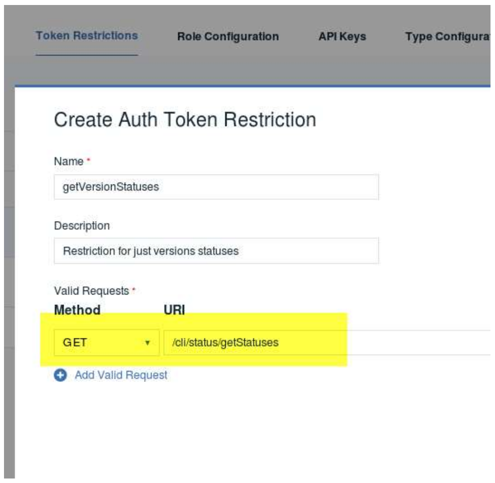
Auth Token

You will need an Auth Token that can be created from the UrbanCode **Settings > Tokens** menu. Its very straight forward. Just click Create Token, Pick the user the token will represent. Set and expiry date & time for way in the future – you don't want to have to keep renewing this. Give it a description of its purpose. So, something like ***Component VersionStatus Retrieval*** in this case***.*** If you are more security conscious then you could look at restricting IP's and /or adding a token restriction. See next section for Token Restrictions.

Once your token is created, you can't change any of this, you will need to create a new one. Also remember to copy the token before you close the final window, or you will have to create a new one anyway. You can't see the actual tokens generated after this point ever again.

Auth Token Restriction

As an optional step, you can also setup a restriction on the auth token you generate so that it can only be used to reach the Version Statuses endpoint. To create a token restriction, click the **Settings** tab then in the middle Security column, pick Token Restrictions. Now Click on Create Auth Token Restriction and complete the form like this:



Create Auth Token Restriction

Name *
getVersionStatuses

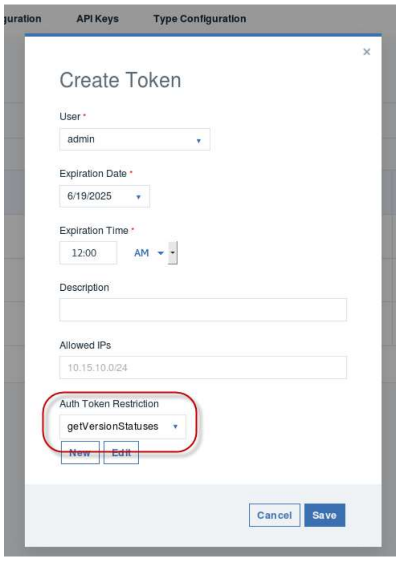
Description
Restriction for just versions statuses

Valid Requests *

Method	URI
GET	/cli/status/getStatuses

This will create a token restriction that will only allow a token associated with it to be used to access only the endpoints list in the URI column. In this case I've set the URI for getStatuses and I've also limited the method to GET.

So now when I create my Auth Token, I can link it to this restriction. So even in the unlikely event that somebody gets this token, they can only do this one thing with it.



Create Token

User *
admin

Expiration Date *: 6/19/2025

Expiration Time *: 12:00 AM

Description

Allowed IPs: 10.15.10.0/24

Auth Token Restriction: getVersionStatuses

Generic Process

The generic process required by the solution is automatically imported with the installation of the plugin and requires only a small modification. While it could be run directly, more information would need to be provided each time you want to apply version statuses.

UrbanCode Plugin

The plugin contains a step which allows the generic process to request that the given status is applied to (or removed from) the component versions referenced in the named snapshot (or if no snapshot is named, the most recent snapshot for that environment).

The plugin also contains the generic process and is loaded automatically with the plugin.

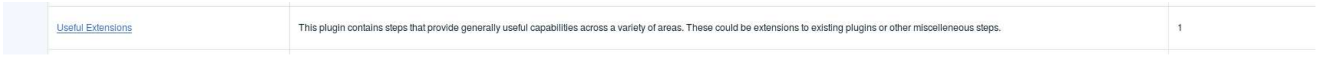
Installing the Solution

The installation is easily installed by firstly loading the associated plugin and then importing the application process into each application where you want to use the solution from. Simply unzip the combined archive which will give you some scripts (for both shell and PowerShell), a JSON file and the plugin zip file.

NOTE: The screen grabs that follow are from UCD 7.1 so they may not match your version of UCD as there have been some significant changes to screen layouts and workflow.

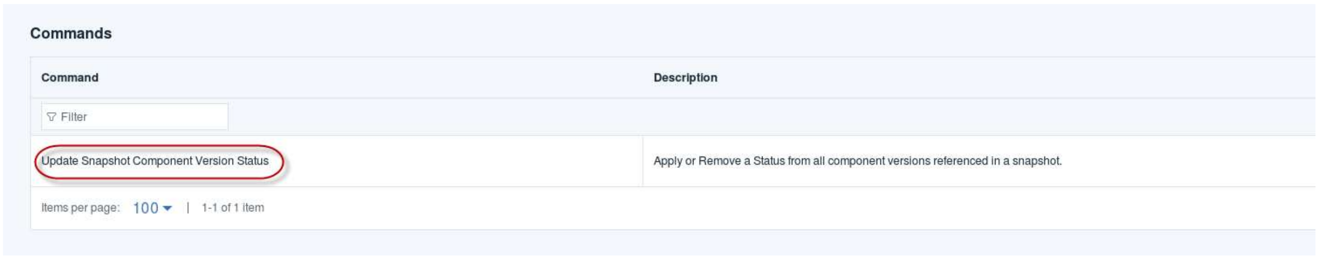
Loading the Plugin

Nothing special here, just go to the **Settings > Automation Plugins** and click on the **Load Plugin** Button. Browse to the plugin zip file and select it. The plugin will be imported into your server. You should now see



Useful Extensions	This plugin contains steps that provide generally useful capabilities across a variety of areas. These could be extensions to existing plugins or other miscelleneous steps.	1

Clicking on **Useful Extensions** you will see the list of steps available.



Commands

Command	Description
Update Snapshot Component Version Status	Apply or Remove a Status from all component versions referenced in a snapshot.

Additionally, if you go to the Processes top level tab you should see the generic process has been created.

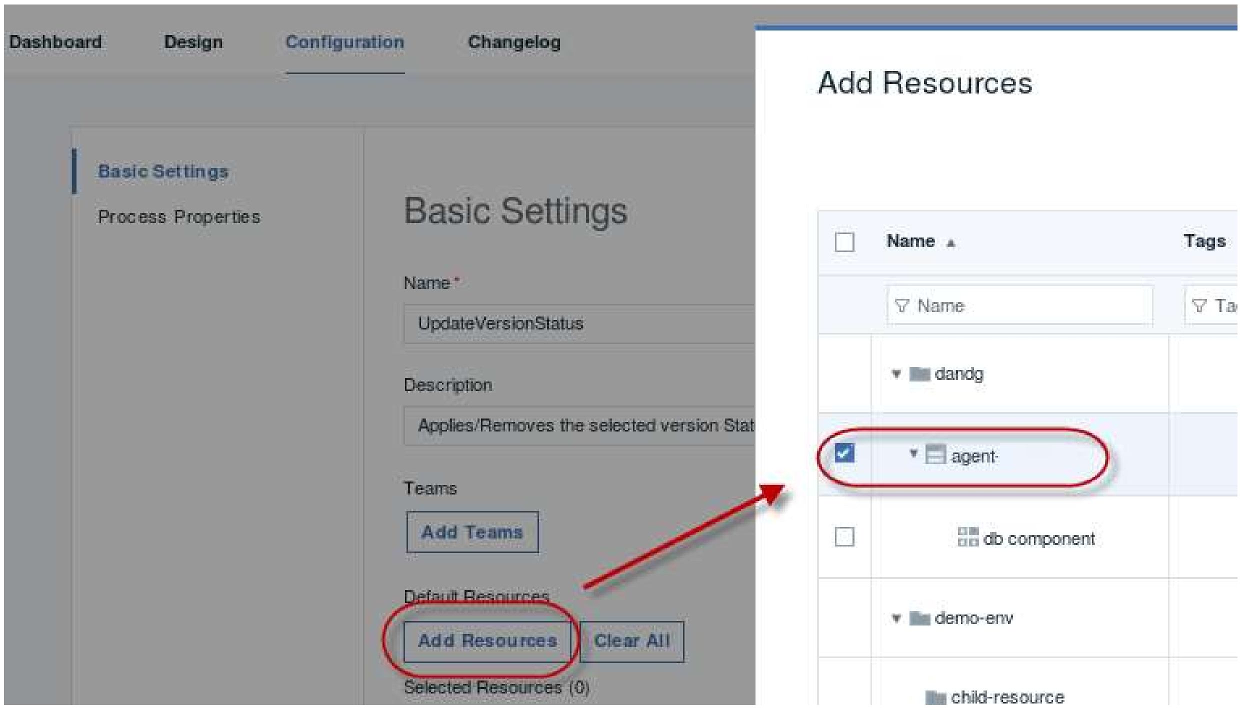
UpdateVersionStatus	Applies/Removes the selected version Status to/from all component versions specified in a snapshot	Run Edit Delete Copy Export

In the process designer you will find the step in the **Useful Utilities/Statuses** Drawer.

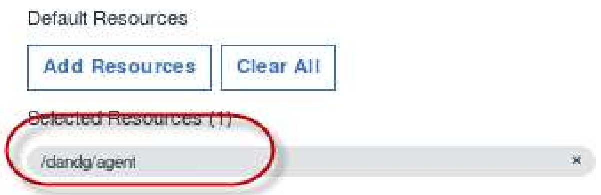
Setting the Generic Process Resource

The generic process needs to know what resource it will use to run on. So, you will need to select an agent that will be responsible for running these tasks. Since the process will be invoked relatively infrequently, it should not present a significant load on the agent chosen.

To set the resource, go to the processes tab and find the **UpdateVersionStatus** Generic process. Click on it and its Dashboard will open. Now Select the **Configuration** tab and select **Basic Settings**. Click the **Add Resources** Button and choose an agent – one that can be guaranteed to be online would be best.



The resource selected will now be displayed and will be the one used whenever this generic process is invoked.



You could choose an **Agent-Pool** as the resource if you wish to provide greater resilience in case the agent you might otherwise have chosen goes offline or is busy.

Click **Save** to save the Change. Remember to click the other **Save** button on the right-hand side of the screen so that the change is saved to the generic process settings.

Optionally, you could Add one or more Teams on this process to limit its usage.

NOTE: If the plugin is updated or re-imported, these changes will be lost and must be reapplied.

Application Process Import

The Application process contains several elements that must be edited prior to importing the application process. To make this easier, included is a shell script to edit the raw script and then import it into your UCD application. The only pre-requisites are that:

a. The location of the **udclient**[1] is included in your **PATH** variable.
b. The **updateStatusAppProcess.json** and the**-importAppProcess.sh (importAppProcess.ps1)** script are in your current directory
c. The **JAVA_HOME** variable is set to point at your local JRE. This should be a JRE compatible with the UCD toolset version you are using.
d. You have created an **AUTH Token** for the user you wish to be used to determine the available version statuses.

The script takes the following parameters:

a. The name of your UCD application; remember this is case sensitive
b. The name of the role you want to restrict this process to. This is case sensitive. Remember to enclose in quotes if there are spaces in the name e.g. “Automation Engineer”. Use "" for no role restriction.
c. The UCD AUTH Token that the imported process will use to determine the available versions statuses
d. The server port combination of your UCD server e.g. myucdHost:8443
e. The user ID of a UCD user that will be used to import the process e.g. admin
f. The password for the UCD user

Here is an example invocation of the script:

```
[root@localhost applyStatus]# ./importAppProcess.sh myUCDApp "Test Manager" ucd-token-string localhost:8443 admin admin
{
  "id": "17301435-c1f0-50c1-a722-ba7d911dc461",
  "name": "Update Component Version Status",
  "description": "Process to Apply\/Remove a Component Version status to\/from all component versions mentioned in a
napshot",
  "active": true,
  "inventoryManagementType": "AUTOMATIC",
  "offlineAgentHandling": "PRE_EXECUTION_CHECK",
  "versionCount": 1,
  "version": 1,
  "commit": 0,
  "path": "applications\/169d894b-d9e8-f02a-be61-64072bd98de2\/processes\/17301435-c1f0-50c1-a722-ba7d911dc461",
  "deleted": false,
  "metadataType": "applicationProcess"
}
```

The above will allow you to import the application process to each named application one at a time. This is good for initial testing to make sure that you’re happy with the solution. If you have many 10’s or more applications adding it to each one of them in turn will take a while.

To assist with this, there is another script called **importToAllApps.sh (importToAllApp.ps1)** that will import the process into all your existing applications. It will skip over ones that already have an application process of the same name and continue with the remaining applications.

NOTE: To run the **PowerShell** scripts you should invoke it like this:
`powershell.exe -ExecutionPolicy Bypass -File importAppProcess.ps1`

NOTE: The applications that can be updated will be determined by the teams that the user you supply belongs to. If you use an administrator ID it should be able to see all applications.

The **importToAllApps.sh** script takes the same parameters as **importAppProcess.sh** except for the omission of the UCD application name parameter.

[1] The **udclient** can be downloaded from the UrbanCode Deploy server from the tools sub menu of the Help menu at the top right of your screen. The Icon has changed in various releases but is ? in UCD 7.1.

NOTE: There is no documented API to remove an application process so removal would be a manual activity.

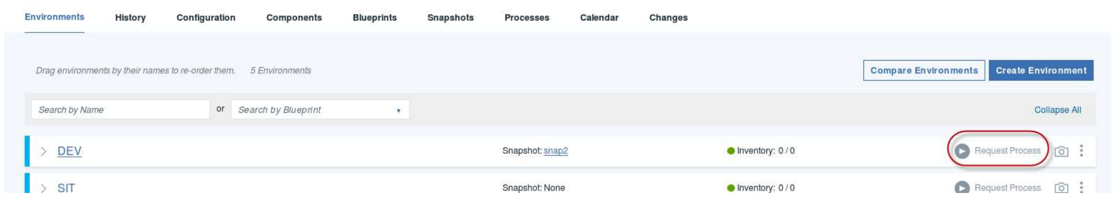
Running the Application Process

To run the process and apply a status to all the component versions referenced by a snapshot all you need are:

a. At least one version status defined
b. An application environment containing a snapshot with deployed component versions referenced in it.

Open your application and navigate to an environment with an available snapshot.

In this example we'll start at an applications environment screen.

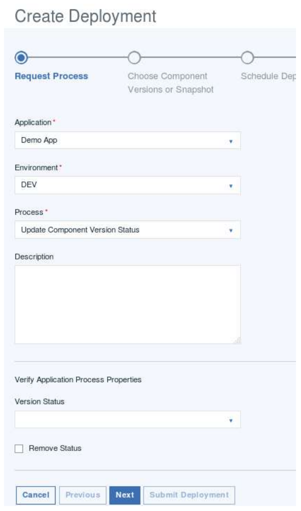


Drag environments by their names to re-order them. 5 Environments

DEV	Snapshot: snap2	● Inventory: 0 / 0	Request Process
SIT	Snapshot: None	● Inventory: 0 / 0	Request Process

Note that this **DEV** environment has a snapshot associated with it, so I'll default the snapshot to that one in the next step. Click the **Request Process** button indicated.

When we request the process (in UCD 7.1) we now see this:



Application*: Demo App

Environment*: DEV

Process*: Update Component Version Status

Description:

Version Status:

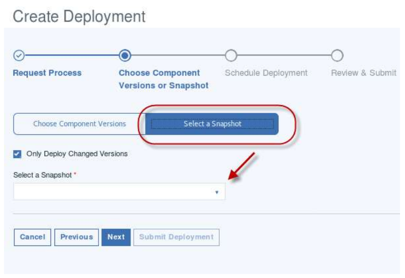
☐ Remove Status

The **Application** and **Environment** are automatically defaulted to the context you request the application process in. Pick the **Update Component Version Status** process.

In the **Version Status** field, I will pick the **Dev Approved** status from the dropdown list.

If you tick **Remove Status,** then the named status will be removed from all component versions instead of being applied.

The new UCD 7.1 deploy wizard has several steps, we can select the snapshot to use on the next step in the wizard, so click **Next**



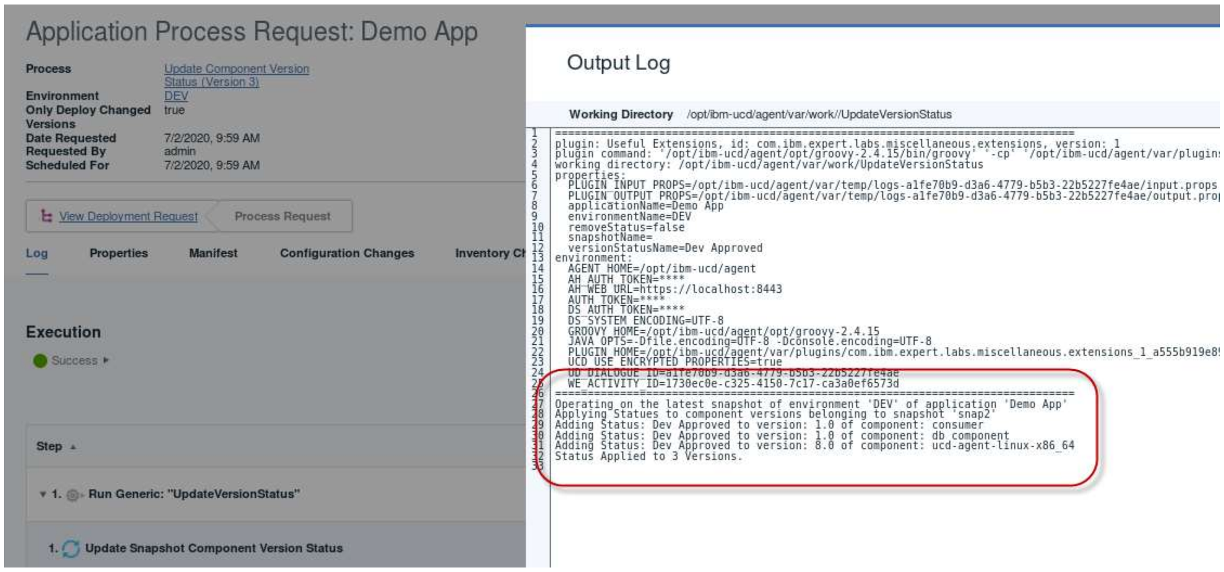
☑ Only Deploy Changed Versions

Select a Snapshot*:

If want you can supply a snapshot name, the component versions in that snapshot will be updated with the status you want. If you don't provide a snapshot name, then the process will use the most recent snapshot applied to or created in the environment you run the process in. If you aren't going to provide a snapshot name, you can just click **Next** on the step in the wizard.

In the remaining two tabs of the wizard there is nothing we need to do on those so just click **Next** 2 more times and then **Submit Deployment**

The process will now run and if we look at the step log, we will see something like this:


Application Process Request: Demo App

Process	Update Component Version Status (Version 3)
Environment	DEV
Only Deploy Changed Versions	true
Date Requested	7/2/2020, 9:59 AM
Requested By	admin
Scheduled For	7/2/2020, 9:59 AM

1. Run Generic: "UpdateVersionStatus"

1. Update Snapshot Component Version Status

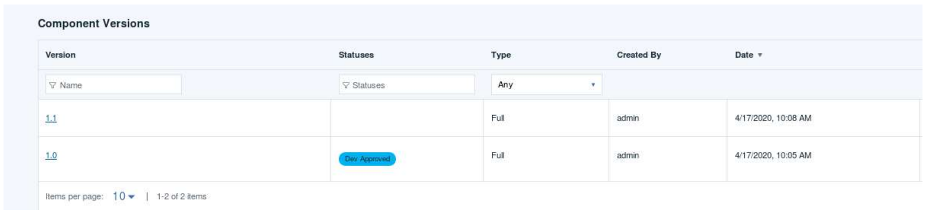
Output Log

Working Directory /opt/ibm-ucd/agent/var/work//UpdateVersionStatus

```
================================================================================
plugin: Useful Extensions, id: com.ibm.expert.labs.miscellaneous.extensions, version: 1
plugin command: '/opt/ibm-ucd/agent/opt/groovy-2.4.15/bin/groovy' '-cp' '/opt/ibm-ucd/agent/var/plugin
working directory: /opt/ibm-ucd/agent/var/work/UpdateVersionStatus
properties:
  PLUGIN_INPUT_PROPS=/opt/ibm-ucd/agent/var/temp/logs-a1fe70b9-d3a6-4779-b5b3-22b5227fe4ae/input.props
  PLUGIN_OUTPUT_PROPS=/opt/ibm-ucd/agent/var/temp/logs-a1fe70b9-d3a6-4779-b5b3-22b5227fe4ae/output.pro
  applicationName=Demo App
  environmentName=DEV
  removeStatus=false
  snapshotName=
  versionStatusName=Dev Approved
environment:
  AGENT_HOME=/opt/ibm-ucd/agent
  AH_AUTH_TOKEN=****
  AH_WEB_URL=https://localhost:8443
  AUTH_TOKEN=****
  DS_AUTH_TOKEN=****
  DS_SYSTEM_ENCODING=UTF-8
  GROOVY_HOME=/opt/ibm-ucd/agent/opt/groovy-2.4.15
  JAVA_OPTS=-Dfile.encoding=UTF-8 -Dconsole.encoding=UTF-8
  PLUGIN_HOME=/opt/ibm-ucd/agent/var/plugins/com.ibm.expert.labs.miscellaneous.extensions_1_a555b919e8
  UCD_USE_ENCRYPTED_PROPERTIES=true
  UD_DIALOGUE_ID=a1fe70b9-d3a6-4779-b5b3-22b5227fe4ae
  WE_ACTIVITY_ID=1730ec0e-c325-4150-7c17-ca3a0ef6573d
================================================================================
Operating on the latest snapshot of environment 'DEV' of application 'Demo App'
Applying Statues to component versions belonging to snapshot 'snap2'
Adding Status: Dev Approved to version: 1.0 of component: consumer
Adding Status: Dev Approved to version: 1.0 of component: db component
Adding Status: Dev Approved to version: 8.0 of component: ucd-agent-linux-x86_64
Status Applied to 3 Versions.
```

The log will show the status applied even if the status was already present. The same is true for removal, it will be listed even if it wasn't present.

And if we look at the versions of one of our components, we will see the applied status:


Component Versions

Version	Statuses	Type	Created By	Date ▾
▽ Name	▽ Statuses	Any ▾		
1.1		Full	admin	4/17/2020, 10:08 AM
1.0	Dev Approved	Full	admin	4/17/2020, 10:05 AM

NOTE: The user who invokes the process will possible need to be a member of a specific role if you set that up when you imported the application process and will of course also have to belong to at least one role that given them permission to apply a version status.

Enjoy !!

Alan Murphy
IBM Cloud Integration : Expert Labs
alan.murphy@uk.ibm.com
July 2020