Preparing for a staging release:

Make sure that all current work has been committed and staged to master. Make sure you are current on the master branch.

Make a new branch for the release: git checkout –b 20130218A0Release0\_1\_4.

Change the define in CLDefinitions from development cliff servers to the cloudstaging servers. (CloudApiPublic\Static\CLDefinitions: define STAGING\_BACKEND.

Change the BadgeCom, CloudSdkSyncSample, CloudApiPublic and CloudSetupSdkSyncSample, and CloudSetupSdkSyncSampleSupport Assembly and File Versions to the current release. E.g., 0.1.2.0. For CloudSetupSdkSyncSample, click Project Assistant, and then Application Information. Set the application version. For BadgeCom, edit the Resources\BadgeCom.rc Version field.

Test the application under Visual Studio to make sure it is working.

Push the release branch to remote. DO NOT CHECK-IN SIGNING FILES\* TO GITHUB!!!!!! Including .pfx files copied to the projects where you changed the settings. These should be ignored by gitignore, but remove them anyway.

\* By SIGNING FILES I do not mean the changes to the projects to turn on signing, but instead the certificates or keys

Staging release procedure:

Increment the version build number of any 3rd party DLL built from source if it was changed.

In Debug solution configuration, clean solution. Check that it succeeded.

In ReleaseSampleAppOnly solution configuration, clean solution. Check that it succeeded.

In Release64 solution configuration, clean solution. Check that it succeeded.

Stay in Release64 solution configuration

Build BadgeCOM project, check for build success

Delete BadgeCOM.dll from 3rdParty\bin\release

Switch to ReleaseSampleAppOnly solution configuration

Build BadgeCOM project, check for build success

Open a Visual Studio 2012 Developer Command Prompt “as Administrator”.

Change directory to ~\3rdParty\bin\Release in command prompt

Run these commands in command prompt (use copy/paste to copy copy the lines directly from below) :

* (Note: Change the versions in this line) tlbimp.exe BadgeCOM.dll /delaysign /publickey:C:\CertBackup\CloudSigning\CloudPlatformCodeSigning.pub /out:BadgeCOMLib.dll /company:Cloud.com /copyright:"Copyright (C ) Cloud.com. All rights reserved." /asmversion:0.1.6.0 /productversion:0.1.6.0
* sn.exe -R BadgeCOMLib.dll C:\CertBackup\CloudSigning\CloudPlatformCodeSigning.pfx
  + requires password to certificate
* copy /Y BadgeCOMLib.dll ..\..\..\CloudSdkSyncSample\bin\release\BadgeCOMLib.dll
* copy /Y BadgeCOMLib.dll ..\..\..\CloudSdkSyncSample\bin\debug\BadgeCOMLib.dll

Check the CloudApiPublic and Sample-Live-Sync references. Make sure that specific versions are selected, and that the references specify the proper versions.

In the CloudApiPublic References, change the BadgeCOMLib reference Specific Version to True. Press Ctrl+Shift+S to save all.

Build CloudApiPublic project, check for build success

Obfuscate CloudApiPublic binary in CloudApiPublic\bin\Release and copy from CloudApiPublic\bin\Release\Obfuscated to CloudApiPublic\bin\Release, CloudSdkSyncSample\bin\Release, and CloudSdkSyncSample\bin\Debug

Build CloudSetupSdkSyncSampleSupport project, check for build success

Switch to Debug solution configuration

Build CloudSdkSyncSample project, check for build success

Switch to ReleaseSampleAppOnly solution configuration

Build CloudSdkSyncSample project, check for build success

Build CloudSetupSdkSyncSample setup project, check for build success

Copy CloudSdkSetup.exe from the setup project output to a convenient location like C:\

Resource hack the copied CloudSdkSetup.exe to change the OriginalFilename field to an empty string in Version Info -> 1 -> 1033 and compile the script; then change the Icon Group -> 100 -> 0 by replacing resource with ~\Artwork\cloudForInstallShield.ico

Save changes as CloudSdkSetup.exe in the same “copied to” location

In the previous VS Developer Command Prompt windows, change directory to where you placed the “copied to and modified” CloudSdkSetup.exe.

Run these commands in command prompt (replace <password> with the certificate password):

* signtool remove /c CloudSdkSetup.exe
* signtool sign /f C:\CertBackup\CloudSigning\CloudPlatformCodeSigning.pfx /p <password> CloudSdkSetup.exe

Zip CloudSdkSetup.exe into a zip file with the naming convention CloudSDK-v0.1.2>.zip where the “0.1.2” is the “version.release.build”. The build should be incremented at each release. The others are incremented by management decision.

The zip file is the completed release. Copy the file to c:\Source\Projects\ArchivedCloudSdkReleases on the build machine. Then use search in win-client (or an automated tool) to gather the current .pdb files and move them into the ArchivedCloudSdkReleases as well.

Close the Visual Studio win-client solution.

Push the release branch to remote. DO NOT CHECK-IN SIGNING FILES\* TO GITHUB!!!!!! Including .pfx files copied to the projects where you changed the settings. These should be ignored by gitignore, but remove them anyway.

\* By SIGNING FILES I do not mean the changes to the projects to turn on signing, but instead the certificates or keys

Tag the last commit to master with this release number (e.g., Release0.1.2).

* Delete a local and remote tag named 12345:
  + git tag -d 12345
  + git push origin :refs/tags/12345
* Create a local tag named 12345 and push it to remote:
  + git tag 12345
  + git push –tags

Delete the .pfx files copied to the CloudApiPublic and CloudSetupSdkSyuncSampleSupport projects where you changed the settings.

* To do this: “Rebuild” the project “DeleteLicenseFilesOnly”.

When the release has been tested, merge the release branch back into master, but don’t merge the signing changes or the switch to the cloudstaging.us URL.

* git checkout master
* git merge --no-commit --no-ff Release0.1.5
* Use GIT Extensions to selectively merge modules. The CloudSetupSdkSyncSample.isl file must be manually merged to remove the signing changes, but leave the version number change.
* git push