Roku

Channel Packaging and Publishing Guide

Roku Streaming Player Version 4.1 12/21/2011

Table of Contents

1.0 Introduction to Channel Packaging and Publishing

The purpose of this document is to provide a guide for how channels are packaged, uploaded, and published to the Roku Channel Store. Developers will upload their channels for distribution to Roku streaming players via https://owner.roku.com/Developer/Home/

When a Channel is created, a developer has an immutable choice between a public or private channel. There will be no opportunity to change your mind later. Public channels can be made available to end users through the Roku Channel Store after being approved by Roku. Private channels will only be made available to end users via a channel code, but do not require approval by Roku.

Channel packages are initially uploaded to the Channel Store as unpublished channel versions. Developers can do beta-testing by accessing these unpublished packages via Roku assigned channel codes.

When satisfied with the testing results, developers can directly publish private packages. This has the effect of making the channel available to end users via a “Vanity” channel code that the developer has chosen for their channel. There is no approval process for publishing private channels. Private channels are used by developers who wish to manage the distribution of their channels to a select group of end-users. The Channel Store now supports URLs that will add a channel via the channel code to a user’s Roku box. Developers can publish these URLs on their website to make it easy for their user’s to add their channel.

Public channels cannot be directly published. They will be submitted for approval by Roku. Assuming you have followed all the Design Guidelines, this approval process usually takes about a week. Once approved by Roku, your channel package will be published and available to end users in the Channel Store.

Since Firmware version 2.6:

With the v2.6 release of the SDK, we have also added versioning support to the Channel Store. This versioning support enables the developer to specify the minimum firmware and hardware dependencies for each package version submitted. The Channel Store versioning support will only display channels to end users’ Roku boxes that meet the developer specified minimum firmware version and hardware dependencies. Because the Roku box always selects the highest package version that is available for a channel that also meets the minimum firmware and hardware dependencies of the currently running Roku box, an end user never has to make any package version decisions for a particular channel.

Instead of end users making channel version decisions, we require developers to specify firmware and hardware dependencies when loading package versions for a channel. Currently, the only hardware dependency is the availability of a USB port. Firmware dependency specification is a somewhat complex topic for developers and we will cover it in greater depth with usage patterns and FAQs later in this document.

Since Firmware version 2.9:

With version 2.9, we've introduced the ability to charge for your channel within the channel store. By upgrading your account to a "Premium Developer" account, you can begin charging for your channel. You can upgrade you account by selecting "Enroll in Premium Developer Program" on your account's "developer site" home page. You will need to supply your TaxID and agree to the Premium Developer terms before completing your enrollment.

2.0 Getting Started

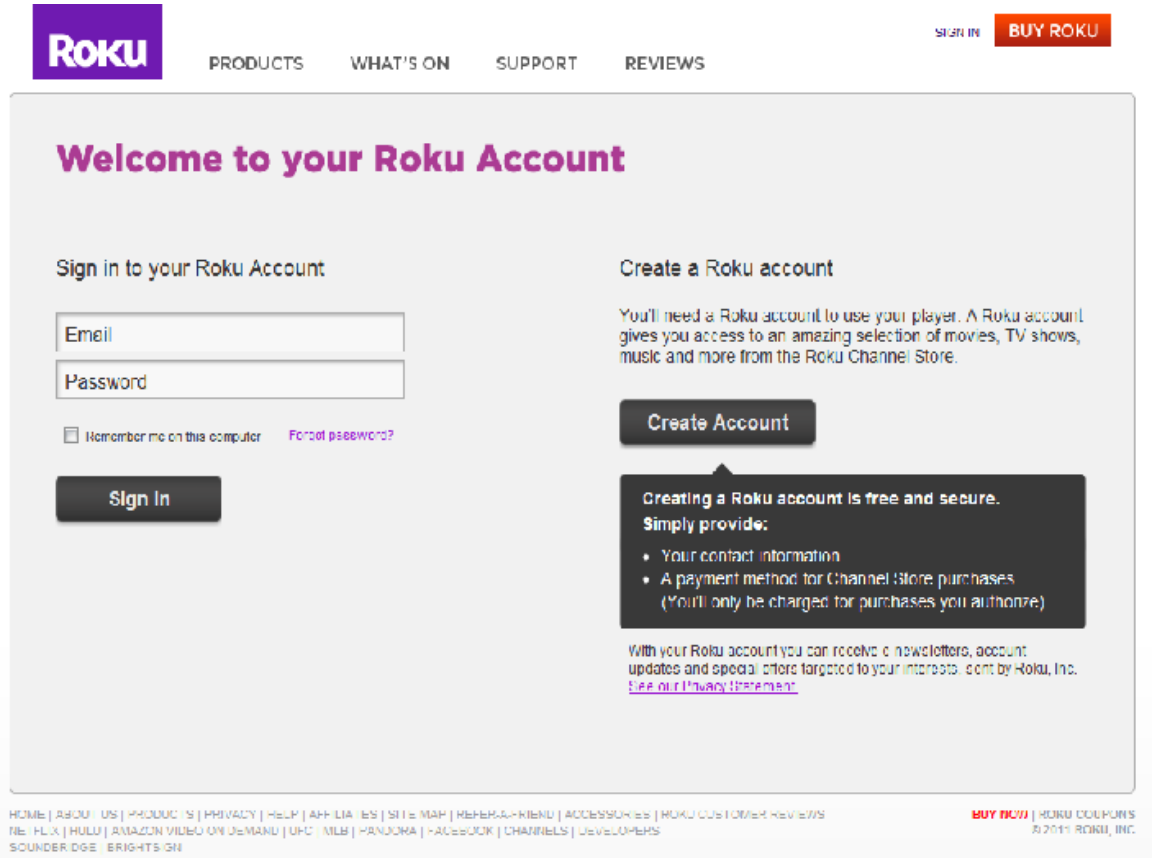
The following section will walk you through creating a developer account and linking your box to that account.

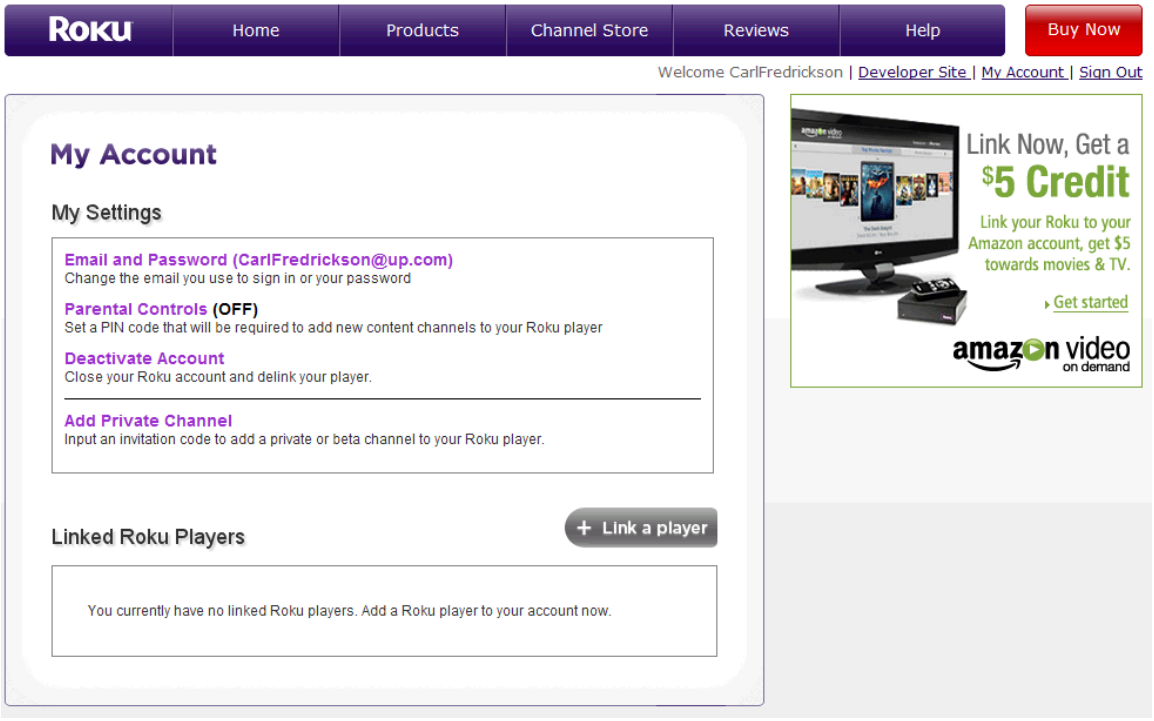
2.1 Software Updates

These instructions assume that you have an up-to-date version of the Roku Streaming Player firmware. Please make sure that your firmware is updated before starting this process.

2.2 Creating a Roku Account

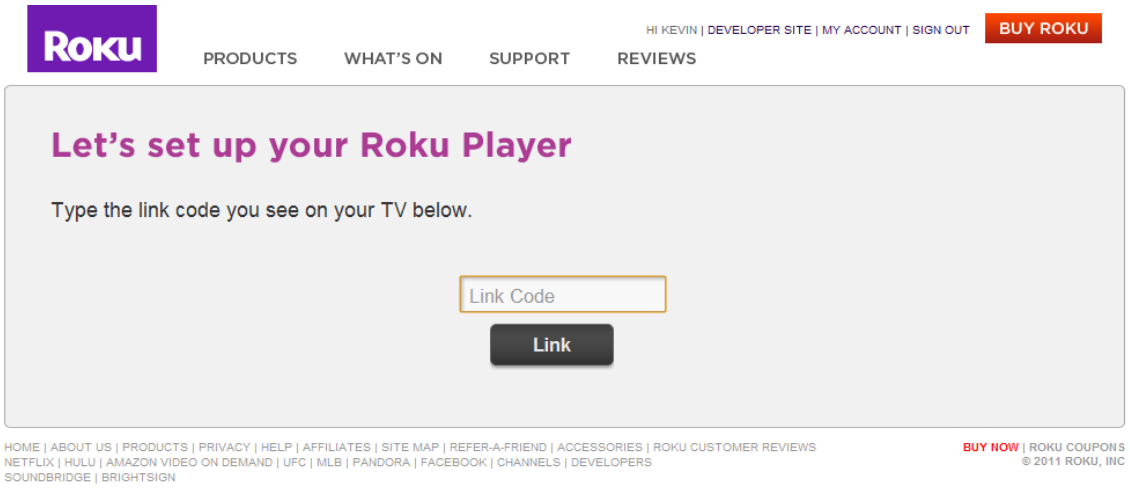
Account creation is done at https://owner.roku.com website. To get started, login or create a new account.





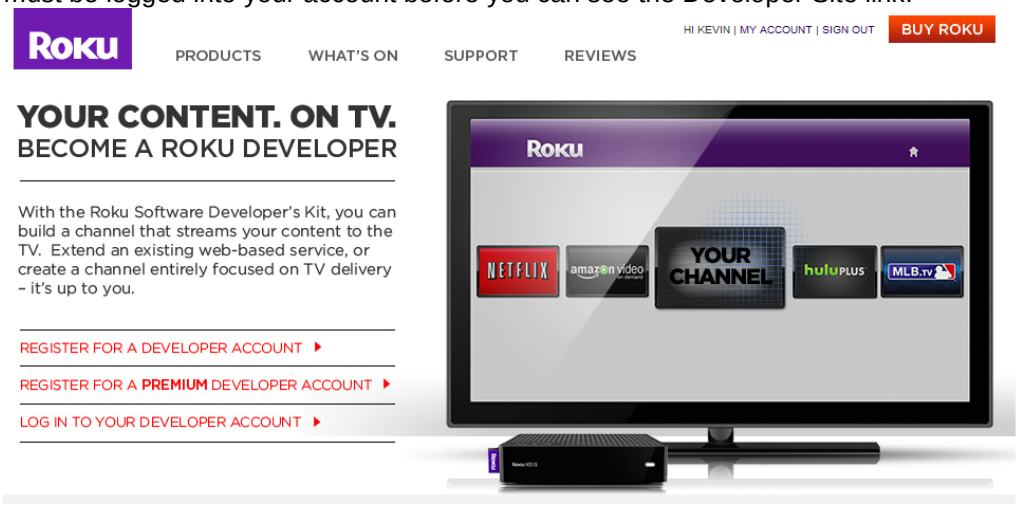
2.3 Linking Your Roku Streaming Player

You must link your player to your Roku account to access the Channel Store. This process uses the rendezvous style of linking and involves getting the link code from your Roku streaming player and entering it into the link page on the website. The device will present the account linking page to the user when they enter the Channel Store on the device for the first time.



2.4 Signing up for the Developer Program

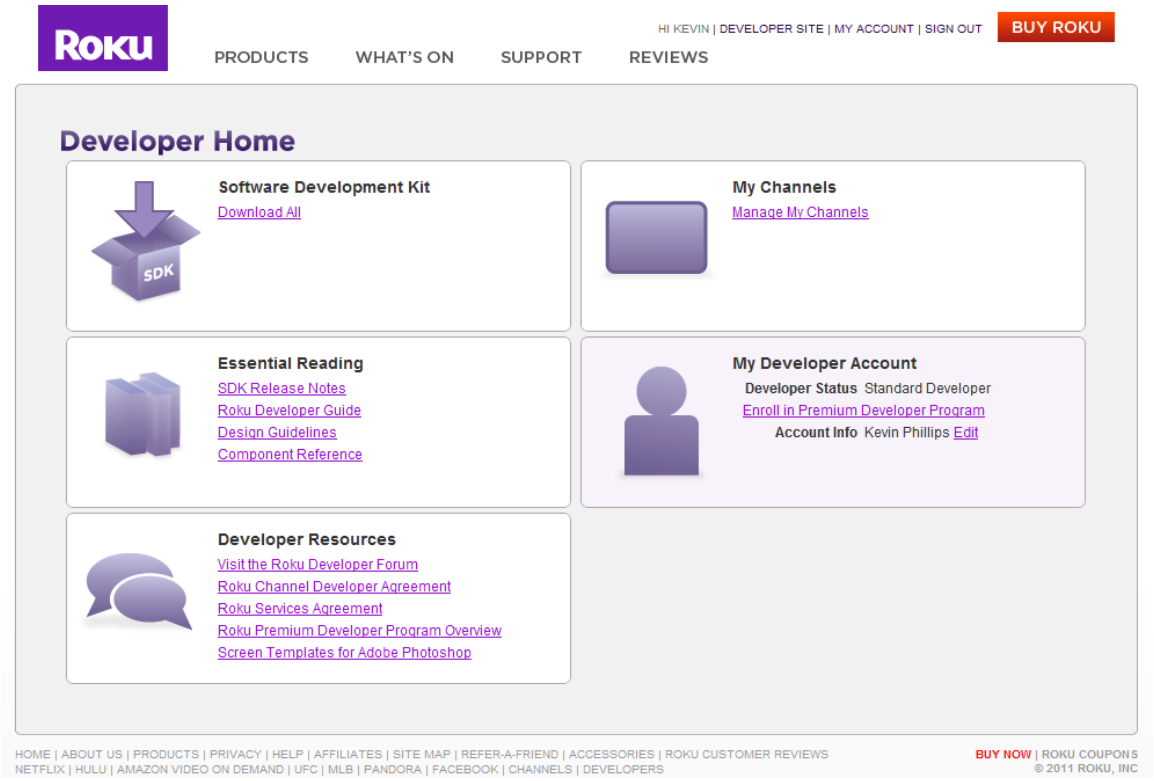
In order to upload or publish a channel, you need to become a registered developer. Sign up for the Developer Program by selecting the **Developer Site** link on the Roku account page. You must be logged into your account before you can see the Developer Site link.



When you haven't yet enabled your developer account, you get the developer home page above. You may click on "Register for a Developer Account" or "Register for a Premium Developer Account". With a Premium developer account, you can charge for your channels within the Roku Channel store. Roku will handle the billing and share revenue with you according to the Roku Services Agreement.

After applying for a Premuim Developer Account, a Roku representative will get back to you to complete your enrollment.

If you start with a regular Developer Account, you may enroll in the Premium Developer Program at any later time. You would click on “Enroll in Premium Developer Program” on your developer home page below:



3.0 Packaging Applications

There are a few simple steps required to package your application before uploading to the Roku Channel Store. This process creates a secure version of your application suitable for general release. The packaging process uses cryptographic hardware on the player during the process, so it requires you to have a player set up for development access.

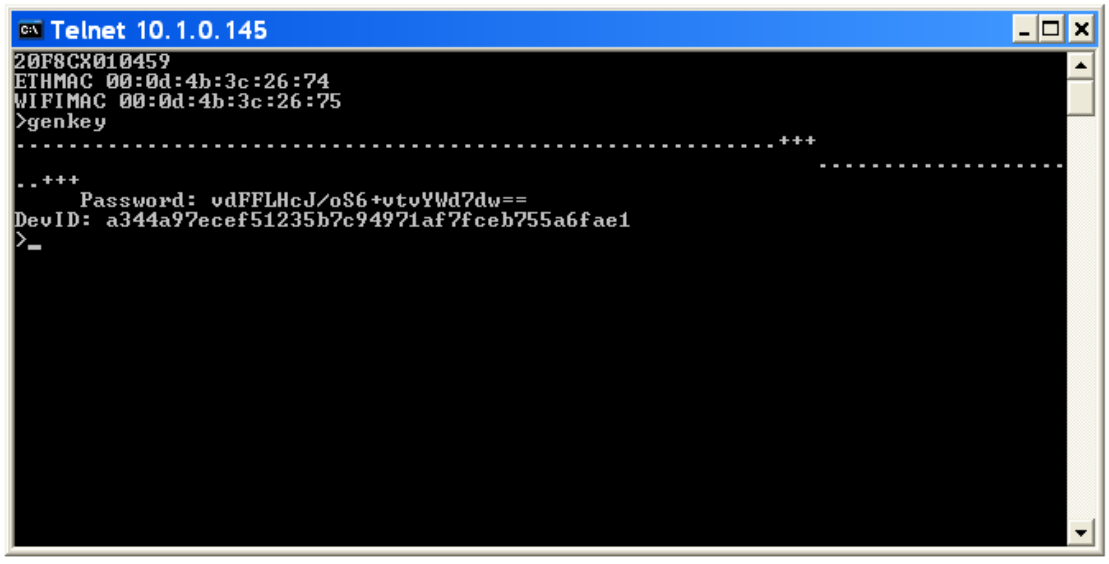
3.1 Genkey

The Roku Streaming Player provides a simple console utility that generates the RSA keys required for packaging. The "**genkey**" utility will generate a set of **Keys**, **Password** and **DevID** necessary for encrypting your application.

The keys generated during the process will be reused every time you package your application, so it's important to keep this information safe. The generated keys are encrypted for security and can only be used if you have the password. The DevID is not confidential and indicates which set of keys was used for signing and encrypting the application.

To access the console, locate the IP address of your device on the "player info" screen in settings and type the following from a terminal application:

> telnet <rokuPlayer-ip-address> 8080
For example:
> telnet 192.168.1.12 8080



3.2 Package Utility

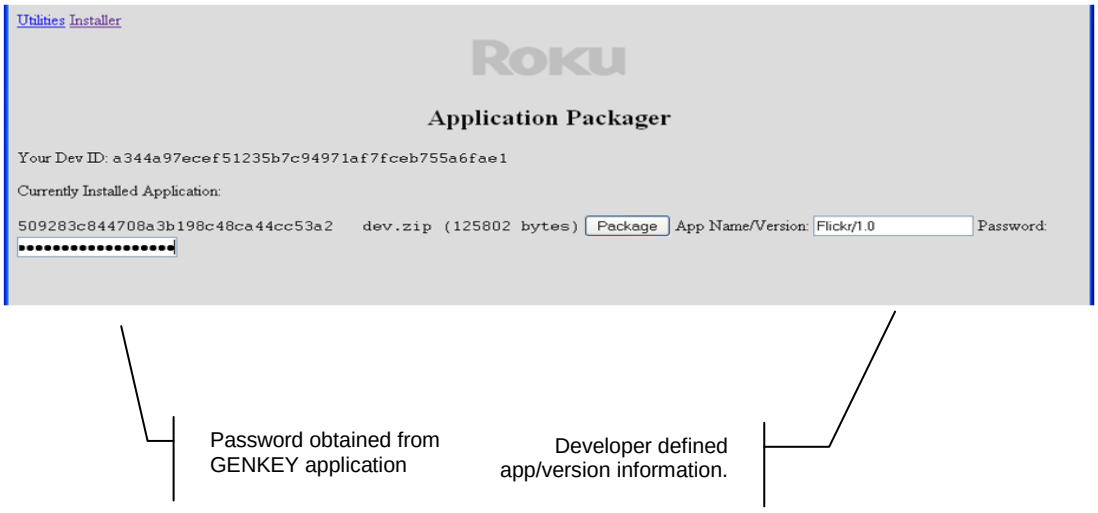
The Developer Web Page on your Roku Streaming Player is used to package your application.

The Developer Web Page is disabled by default. It is enabled using a special sequence of remote key presses. The sequence is: 3 Home, 2 Up, Right, Left, Right, Left, Right. This sequence should bring up a screen that asks whether you want to enable the installer. Click the “enable installer” button and answer yes to the pop-up dialog question. The Roku Streaming Player will reboot.
Now open a browser to your box using the IP address of your device. For example: http://192.168.1.23

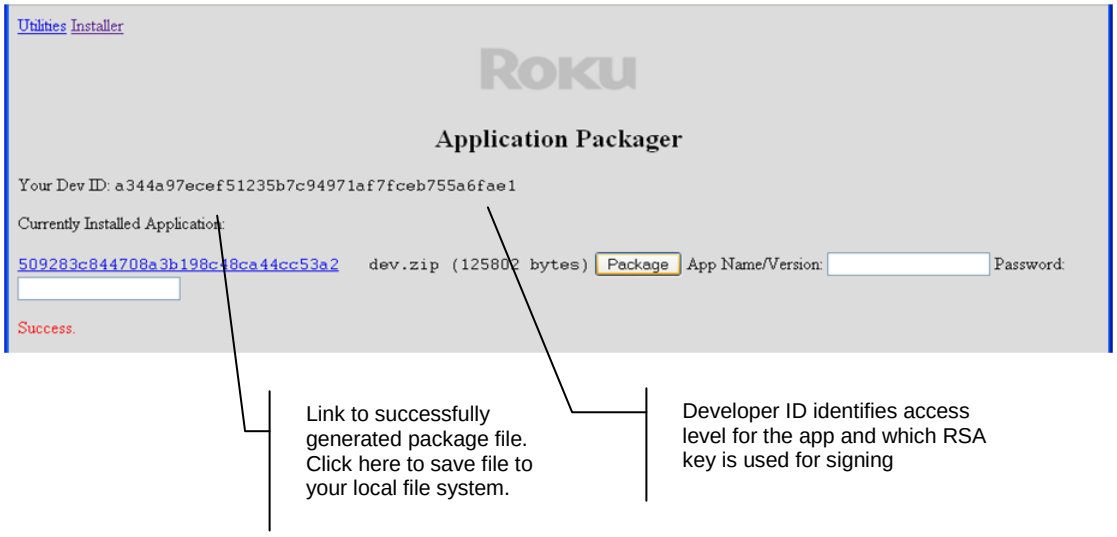
Install your application as you would normally, by selecting your application zip file using the “Browse…” button and then pressing “Install”. The box now has a copy of your application installed, plus the keys are still resident on the device from when you performed the “genkey” operation.

Important: If there are no keys installed on your device, the Packager option will not be available. You may either generate a new set of keys or use the ReKey option on the utilities page to install keys on your device.

Click on the “Packager” link at the top of the page. This screen allows you to enter an app name/version string to embed in your package and the password to allow access to the keys on the device. Pressing the “Package” button will take the application currently installed on your box, encrypt it and sign it with your developer specific keys. When complete, it will generate a finished .pkg file ready for deployment. The .pkg file can be downloaded to your PC from the link displayed on the page when packaging is complete.



Packaging process complete



The Utilities page allows you to reinstall a set of keys for packaging your application and to inspect the contents of a packaged application. These features require that you enter the corresponding password for those keys.

Each application package contains an encrypted version of the keys that were used to initially package that app. The "ReKey" button allows you to install a set of keys from an existing applications and use those keys to package a new application. This allows you to use the keys resident in the previous version of the application to sign the new version of the application or enable multiple boxes to sign your application. To do this you ***MUST*** have the password that was created when the keys were originally generated. It is highly recommended that you safely store the keys and password for future use. You may create a NULL application package which can be used as a container for storing a set of keys for future use. You may create a NULL application package which can be used as a container for storing a set of keys for future use.

3.3 Brief Overview of the Package Security Model

The Roku Streaming Player is designed to be a general purpose streaming video player capable of supporting multiple content providers and applications through the Channel Store. The Security Model has provisions in place to prevent unauthorized access to content, services or any other application specific data. The Roku firmware was designed to provide safe, "sandboxed" access to platform services through a virtual machine environment. The "sandbox" provides:

Restricted access to platform services

- No access to extend or modify binaries
- No access to audio/video data

Restricted access to other developers application data

- No access to data from applications signed with a different KprivDev
- No access to log, console or other output
- No access to source code for other applications

Each application has access to a secure system registry for storage of application specific data. The registry is encrypted for safe storage of secrets and can only be read by applications signed with the same key. The registry is unique to a specific developer context. In general, each application maintains its own unique registry, but if a family of applications needed to share registration information this is also possible.

When a developer performs a genkey on their Roku Streaming Player, they generate an RSA key pair. When they package their application, they sign and encrypt the application with the private key genkey produced. The package signature also creates an association to a secure registry. The firmware enforces access to the registry based on the key that signed the package.

The developer can create multiple applications that share a common registry by signing all of the applications with the same key. If the private key (or its password) is lost, new versions of the application will be unable to access the registry. Since the private key is stored in the package, but protected with AES encryption via the password, it can always be recovered from a previous package as long as the password is remembered. So it is of utmost importance to safely and securely store that password and package containing the private key. With the password and a package containing the corresponding keyset from a genkey, any Roku Streaming Player can be "Rekeyed" to enable signing of packages with the same key that was stored in a previous package.

3.4 Consequences of the Package Security Model

- Applications can share data via registry entries if and only if the applications are packaged with same key
- New versions of applications will have to re-link if they are packaged with a different key
- Be careful about creating applications that are a "suite" and share the same signing key. When apps are part of the same suite, they should share the same linking information in the registry because this linking information will not be removed from the registry until all channels in the suite are removed.
- It is critical to save the password and keep track of the key that signed an application. If the password is lost, you will not be able to sign any new packages (including new versions) with the same key.
- Multiple developers can use the same key to sign their applications by "Rekeying" the Roku Streaming Player with a package containing the key and entering the corresponding saved password.

3.5 Content Security Model

The SDK supports a content security model using time-expiring random urls and a meta-data feed accessed with SSL mutual authentication. The success of the security model requires a proper implementation by the developer. Roku is not responsible or liable for any losses resulting from the use of this SDK.

The general idea is that a developer could protect the meta-data feed with mutual SSL authentication. Each Roku box has a unique client certificate that is signed by Roku as the Certificate Authority. The firmware also supports a special header, x-roku-reserved-dev-id, that always sends the developer id of the currently running application. Since this is authoritative and can't be forged, the developer's web server can only accept connections from a Roku Streaming Player running their application. Likewise, the Roku Streaming Player can enforce that it is talking to the developer's web server by including the CA certificate of the authority that issued their web server certificate in their channel application. The implementation details of this mutual SSL authentication method are included in section 4.25 of the Component Reference Guide.

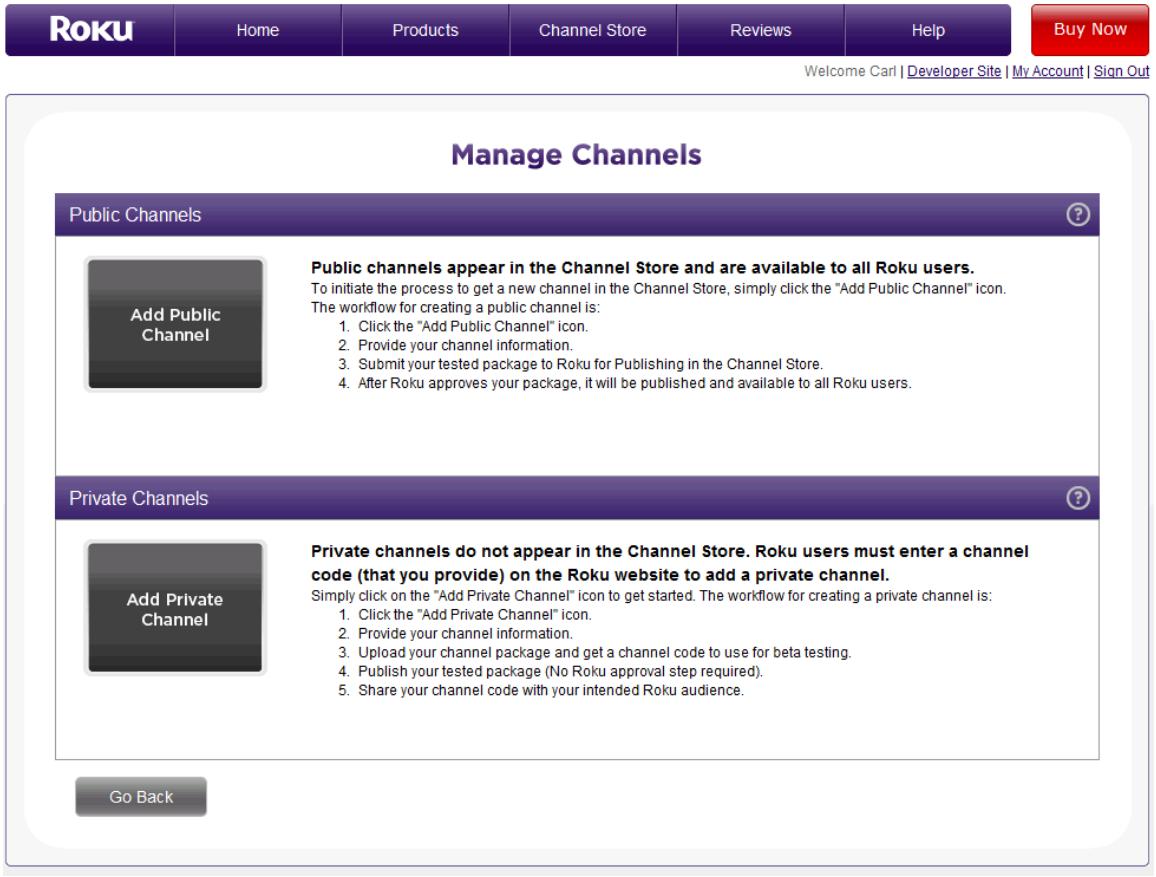
In the secure meta-data feed, the developer can send the URLs to his content. Since these URLs are long random strings, it would be very difficult for a hacker to guess the URL. Since they are time-expiring, even if they guessed correctly, the damage would be limited as they will disappear when they expire.

Playback of content over SSL-encrypted HTTPS connections ensures that the content is encrypted while en route to the Roky player. The decrypted content is only available for playback and never stored on the Roku Streaming Player. The output ports are also protected: The HDMI port features HDCP and the component video outputs support both CGMS-A and Macrovision. .The USB is not for general purpose use.

4.0 Managing Channels

4.1 Overview

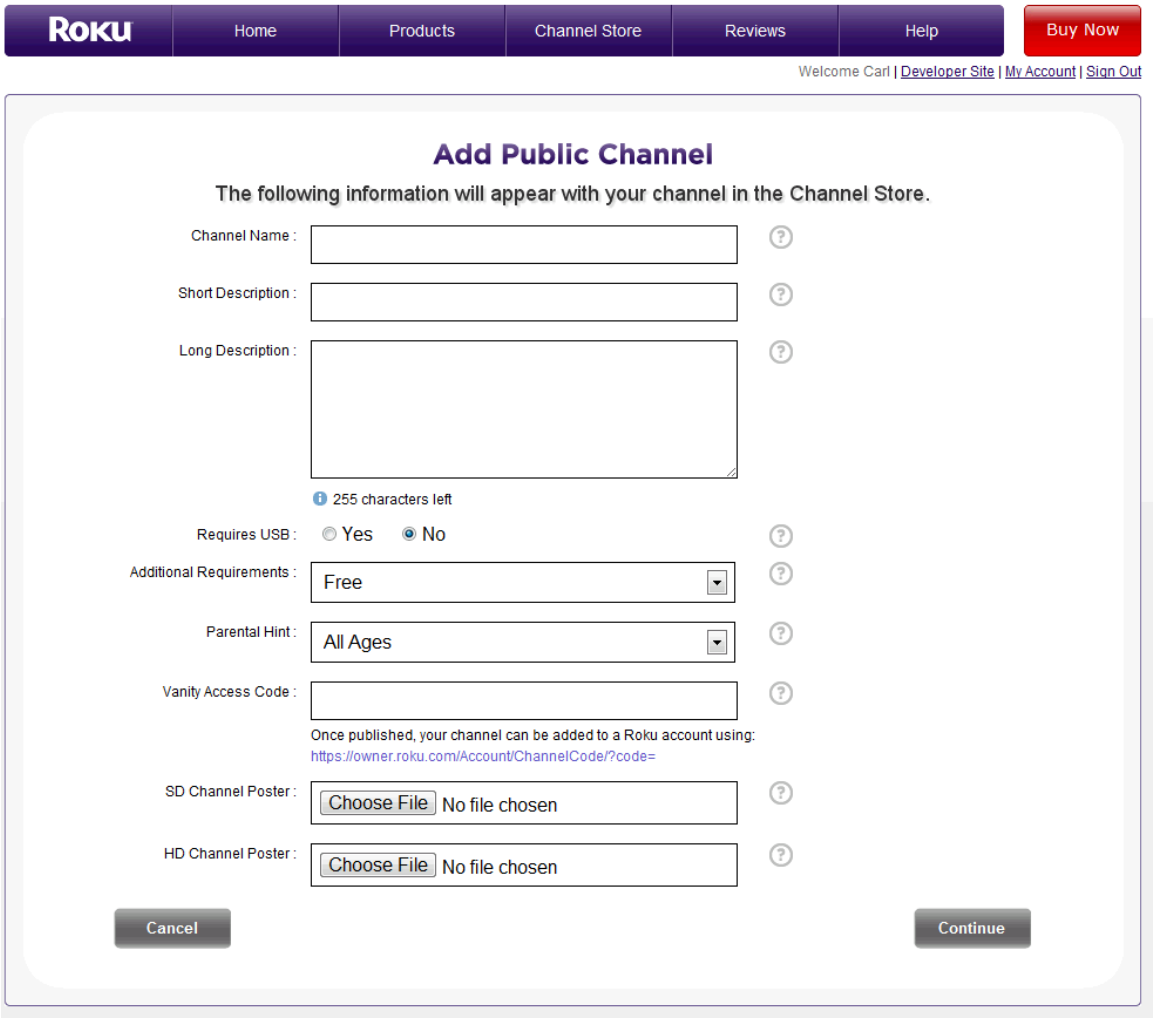
When the developer first clicks on “Manage Channels” on the Developer Site home page, he will get the following empty Manage Channels page:



At this point, the developer is confronted with the choice of creating a public or private channel. It is important to consider the decision carefully as there is no way to reverse the decision later. (If you did change your mind, you would have to create a new channel and delete the old one). Public channels will show up in the Channel Store and will be selectable for install from every user’s Roku box. Private channels will not appear in the channel store and can only be installed via a channel code on the Roku website. With the v2.6 release of the SDK, these channel codes can be specified by the developer and permanent URLs can be distributed by the developer to ease the end user’s install of channels via channel codes.

4.2 Adding a Public Channel to the Roku Channel Store

1. **Select the "Add Public Channel" icon on the Manage Channels page. This brings up the Add Public Channel page:**

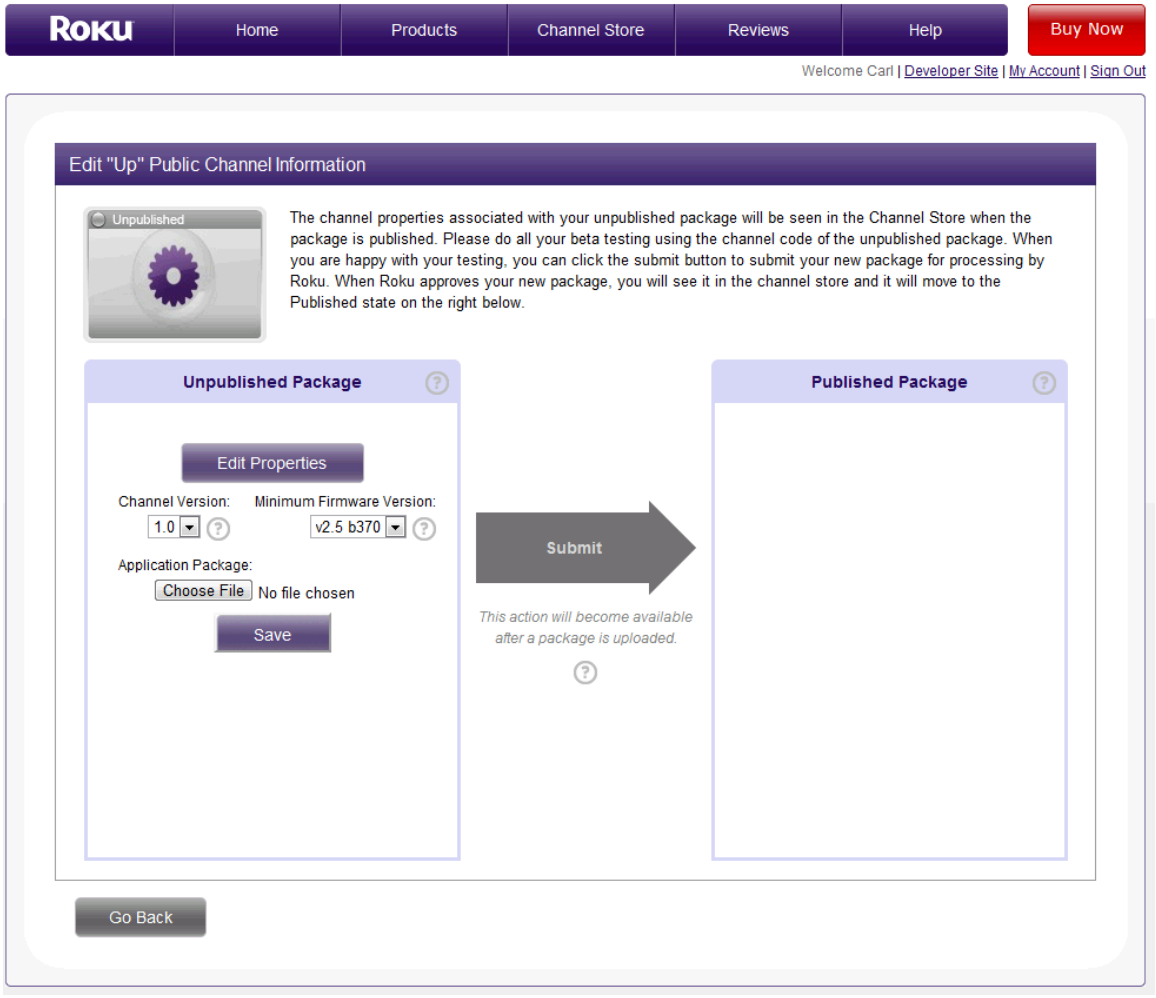


On this page, you will enter initial metadata about your channel. In v2.6 of the SDK Channel Store release, we've added a couple of new parameters. First, you'll notice a "Requires USB" field. This field allows the developer to specify a hardware dependency on the Roku Box. Clicking "Yes" requires that the user's Roku box has a USB port in order to view or install this channel. Second, there is a new ability to specify a vanity channel access code that is applied to package versions that you publish. In this way, the same vanity channel code will install the corresponding published version of the channel. Note that the Vanity Access code will not function until the channel, private or public, is published. You can copy the link below the Vanity Access Code box and distribute it to your users as a way to add your channel to their Roku accounts.

Note that users can always add channels to their account via a specified vanity channel access code, but whether that channel shows up on a Roku box linked to the user's account also depends on if the Roku box meets the "Requires USB" specification for the channel. Later, we'll also see how to specify minimum firmware revisions for particular packages. The minimum

firmware revision and requires USB are package dependencies that limit the scope of which Roku boxes will have access to the package.

2. **When you've completed the Add Public Channel page and click "Continue", you get the Channel Information Page:**



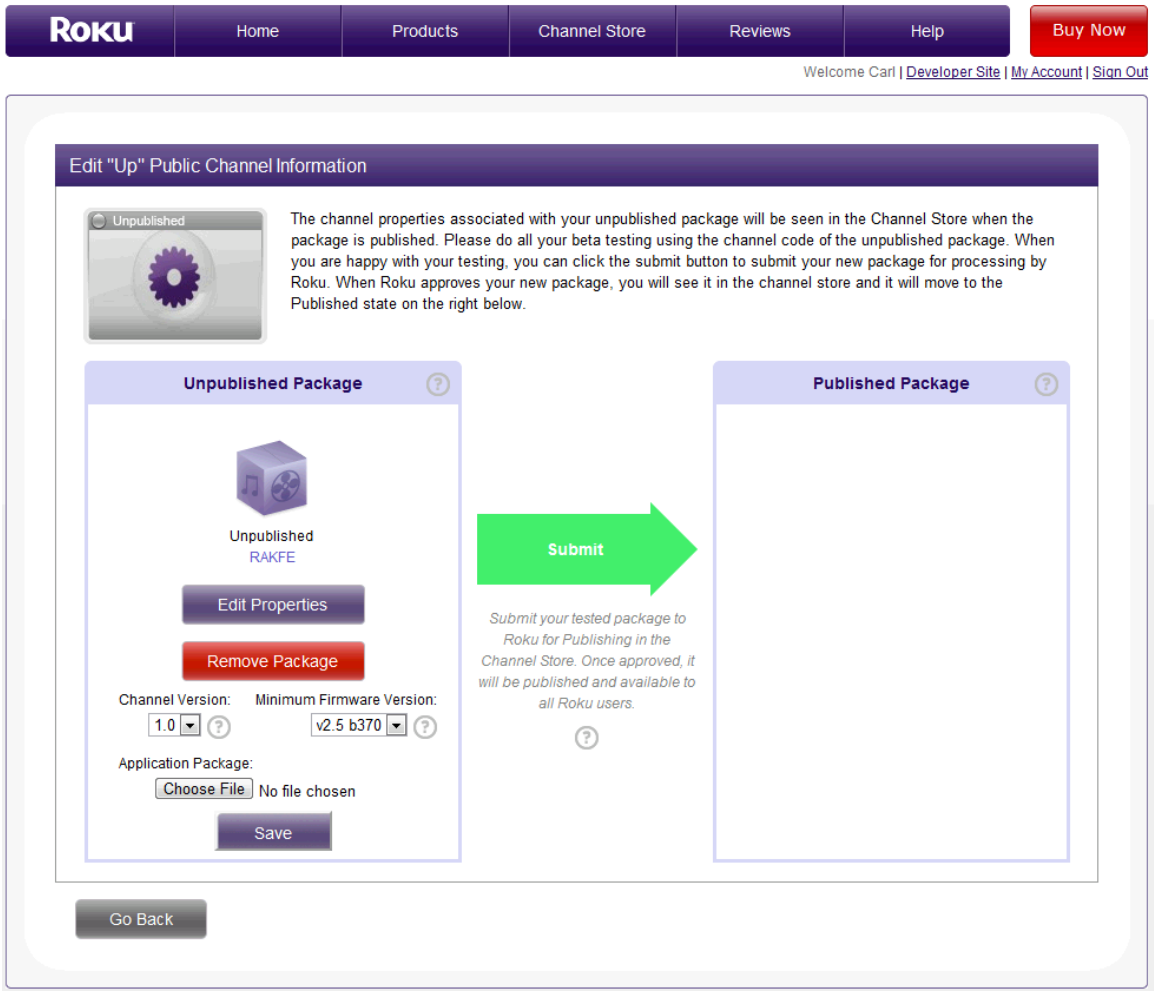
On this page, you will upload a package file by clicking on the "Choose File" button. Then you will enter a channel version number and minimum firmware dependency to associate with the chosen package file. The "Submit" button remains grey until a package file is uploaded and the associated channel version and minimum firmware version have been saved by clicking on the "Save" button. Note that the "Edit Properties" button allows you to go back and edit the meta-data parameters associated with this package. The package specific properties can be changed on a per package version basis. The initial values were entered on the previous Add Public Channel page.

Over time, you may end up publishing several versions of packages for a particular channel. If a package channel version has the same Minimum Firmware Version dependency as a previous package channel version it effectively replaces that package in the Channel Store. Roku boxes will always download the latest channel version package that meets the Minimum Firmware Version and hardware requirements.

If you have published packages with different Minimum Firmware Versions, it's possible that a particular Roku box may get an older package channel version than the one you just published. For example, if you publish packages:
1.0 Min Firmware v2.4
2.0 Min Firmware v2.6

Then Roku boxes running firmware v2.4 or firmware v2.5 will download and run package channel version 1.0 of your channel. Roku boxes running firmware v2.6 or later will be running package channel version 2.0 of your channel. When Roku may later update the firmware on the v2.4 box, the channels are automatically updated to the newer version.

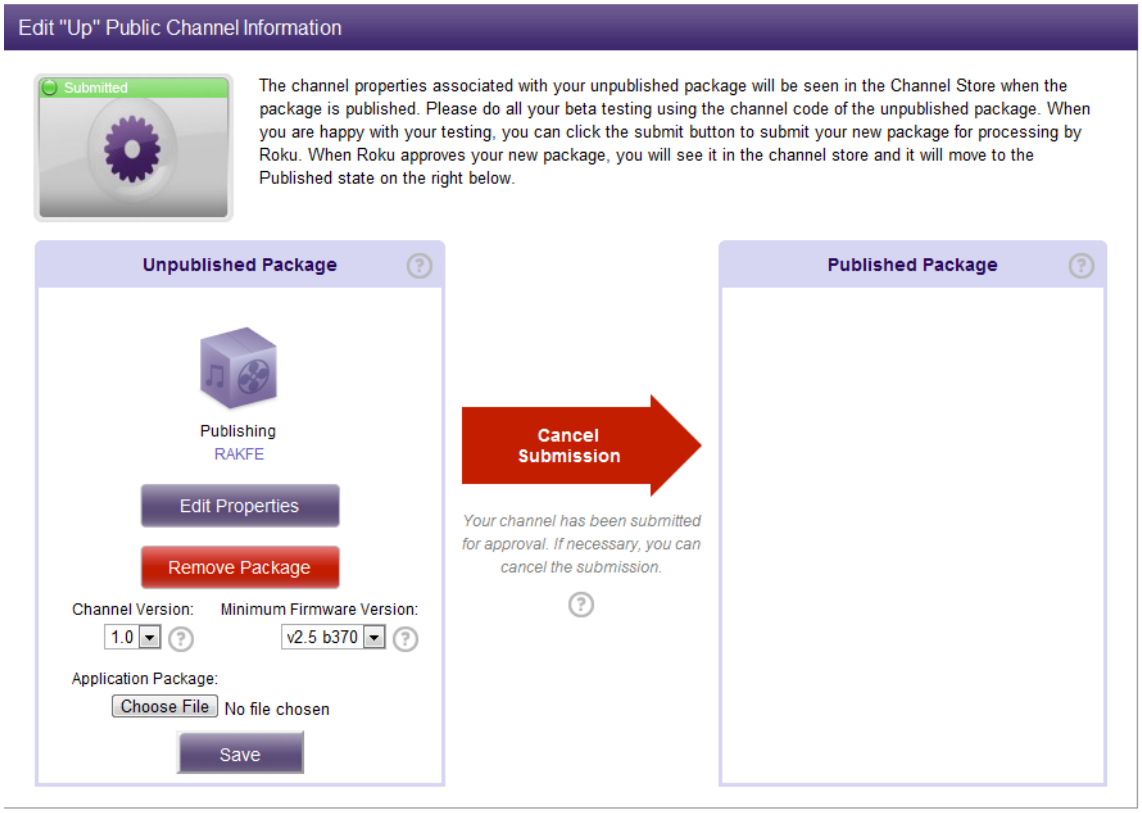
3. **After clicking "Save" to associate the package with a Channel Version number and a Minimum Firmware Version dependency, The Submit button turns green and becomes clickable.**



At this point, you can use the Roku assigned unpublished channel code ("RAKFE" in the example above) to have your beta testers add the unpublished package to their Roku boxes and test this version out. Be sure that you have reviewed the "Before Publishing" checklist in

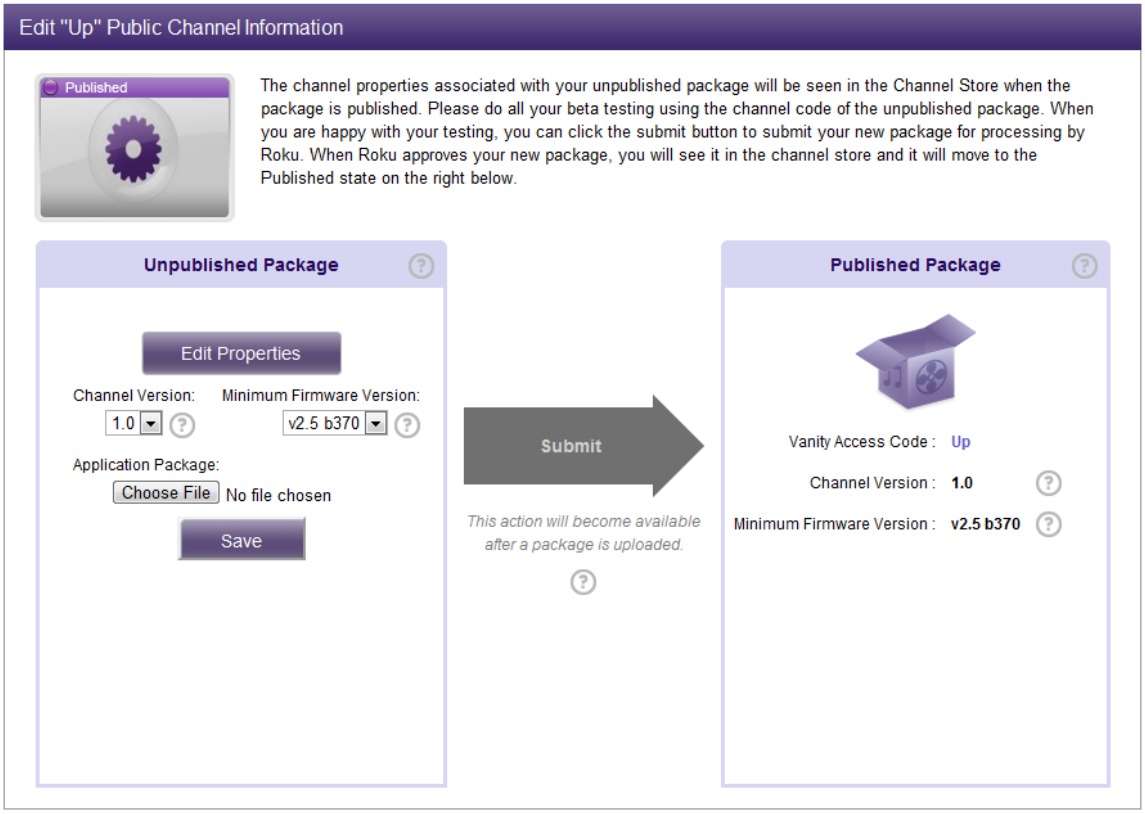
the Developer Guide. Your application should also follow the conventions documented in the Design Guidelines. When you are satisfied with the results, click the “Submit” button to submit your package to Roku for approval and publishing.

4. **After clicking “Submit” to request Roku approval, the state of the package becomes “Submitted” and the stripe above your channel image turns green. The green arrow has also turned into a red arrow that is now labeled “Cancel Submission”.**



In this step, your unpublished package has been submitted to Roku for approval. As soon as Roku approves your channel, it will be published and available to end users in the Channel Store.

5. **After a few days, Roku will either approve your channel or contact you via email about some changes we would like you to make to meet our Design Guidelines.**



When your channel is approved, the channel vanity access code is assigned to the package and it now appears in the “Published Package” box. The “Unpublished Package” box is empty once again and ready for you to upload the next package version of your channel. Note that the stripe above your channel is now purple and the status is “Published”.

You can always come back to this page to update your channel with new package channel versions. Any new packages you upload are always entered as an “Unpublished Package” on the left, and then eventually Submitted or Published to become a “Published Package” on the right.

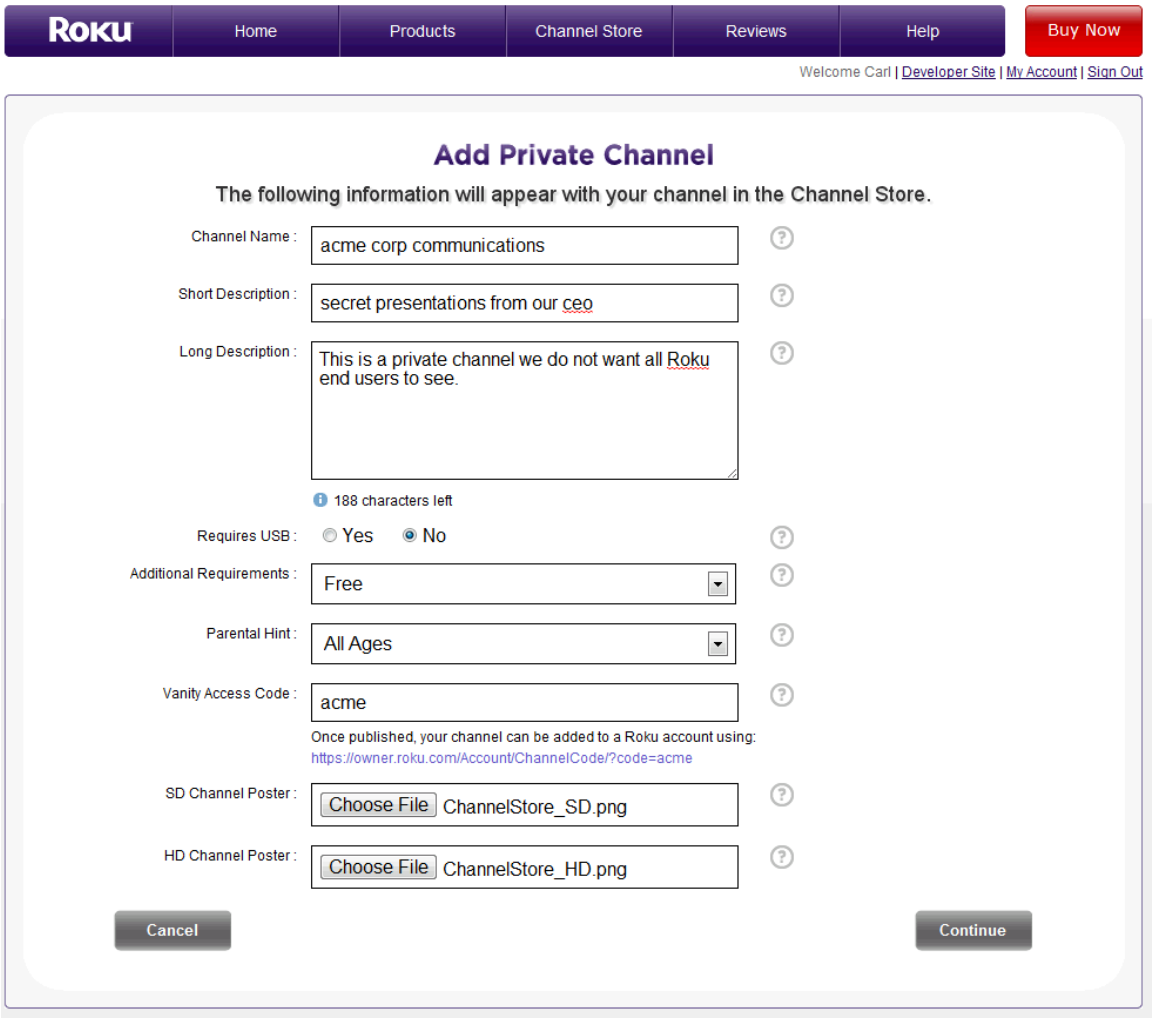
6. **Now that your package is published, it is available to Roku end users. It should roll-out to all end users within a day or two. Users can force an immediate update to the package you just published by entering the Channel Store on their Roku box.**

Entering the Channel Store on a Roku box synchronizes and updates all the channels associated with the corresponding Roku account. A particular Roku box will select the appropriate channel package version by choosing the highest package channel version that meets the “Requires USB” and “Minimum Firmware Version” package dependencies. If there

are no package channel versions that meet the dependency criteria, the channel is not displayed in the Channel Store nor installed on the Roku box.

4.3 Adding a Private Channel to your Developer Account

1. **Select the "Add Private Channel" icon on the Manage Channels page. This brings up the Add Private Channel page:**



This page is just like the Public Channel page. There are two major differences between the public and private channels. Private channels will not appear in the "Store" on user's Roku boxes and private channels can be published immediately without approval by Roku.

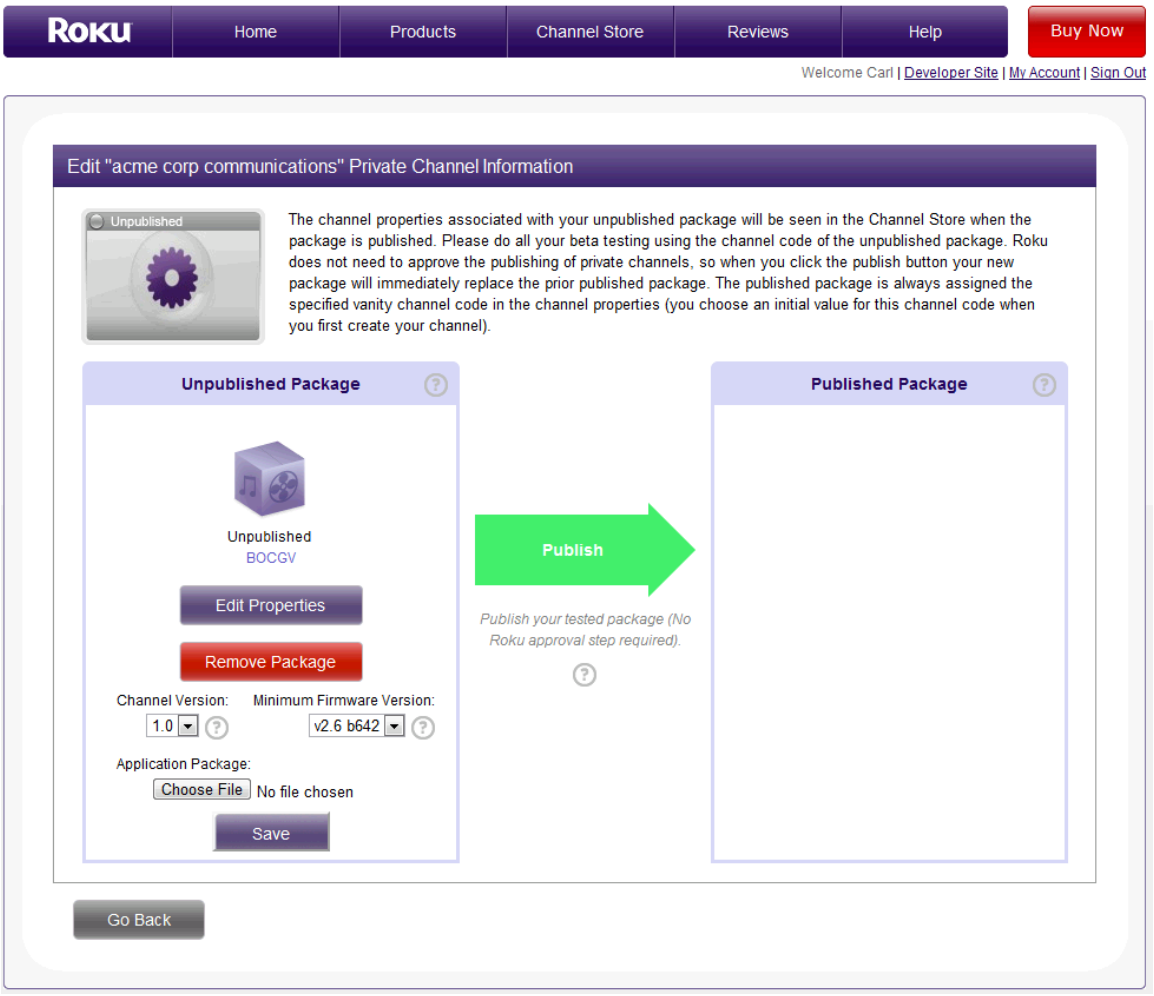
Users must use a channel code to associate a private channel with their Roku account. Roku accounts can have multiple boxes linked to them, and when a box synchronizes its channels the private channels are added to the boxes. Private channels are useful when you want to create a channel that is available only to a special population and you don't want your channel to be known to all Roku customers. One popular use of this scenario is for corporate communications. With the v2.6 SDK release, the Channel Store also supports vanity channel access codes for published private channels.

The vanity channel access code is highly recommended because it is the only way to assure that private channels added to a user's account continue to receive channel updates. The codes assigned by Roku are package specific, whereas the Vanity Code applies to all published package channel versions.

2. **When you've completed the Add Private Channel page and click "Continue", you get the Channel Information Page:**

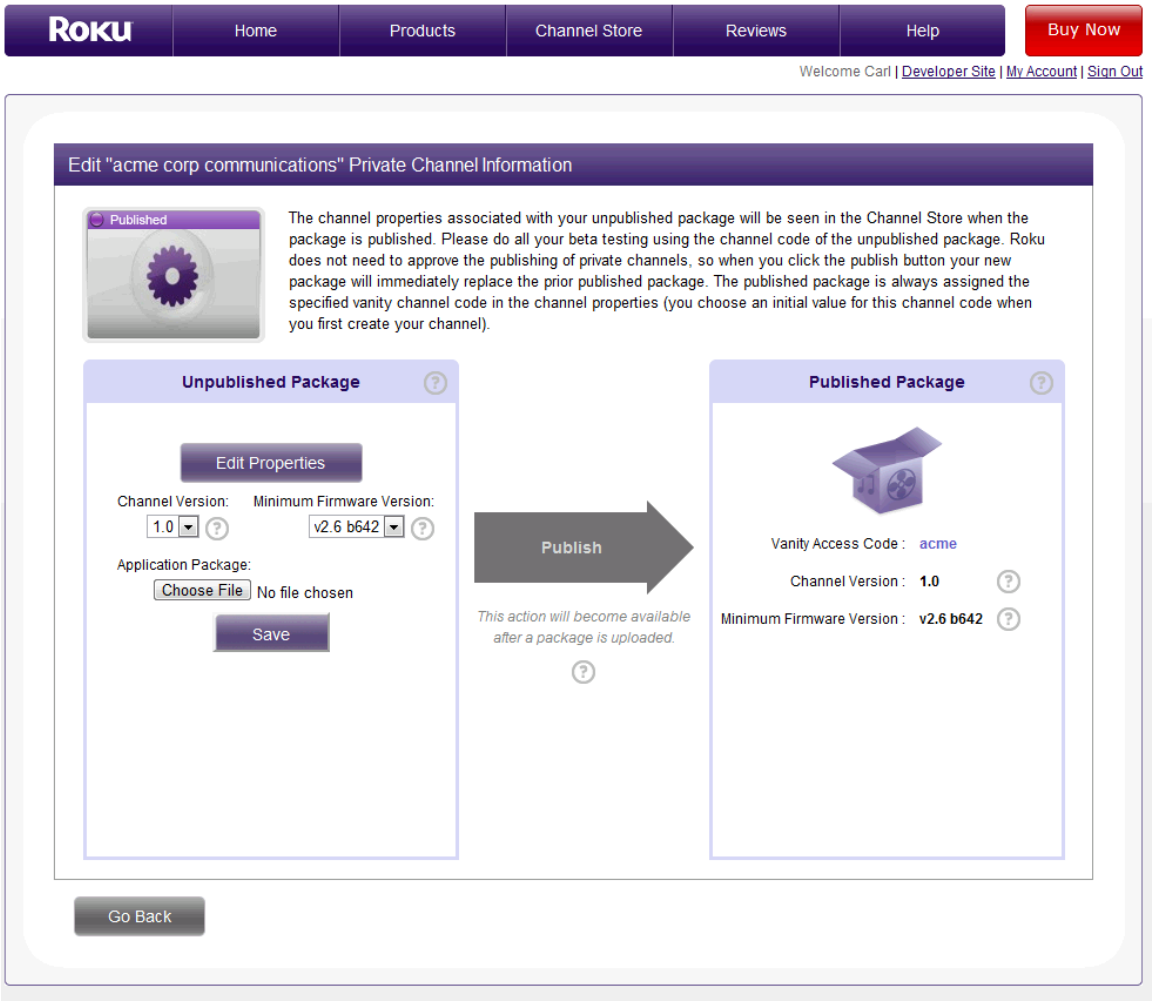
Please review this step in the 'Add Public Channel' for a discussion about package channel version selection on the Roku box. It works the same for public and private channels.

3. **After clicking "Save" to associate the package with a Channel Version Number and a Minimum Firmware Version dependency, The "Publish" button turns green and becomes clickable.**



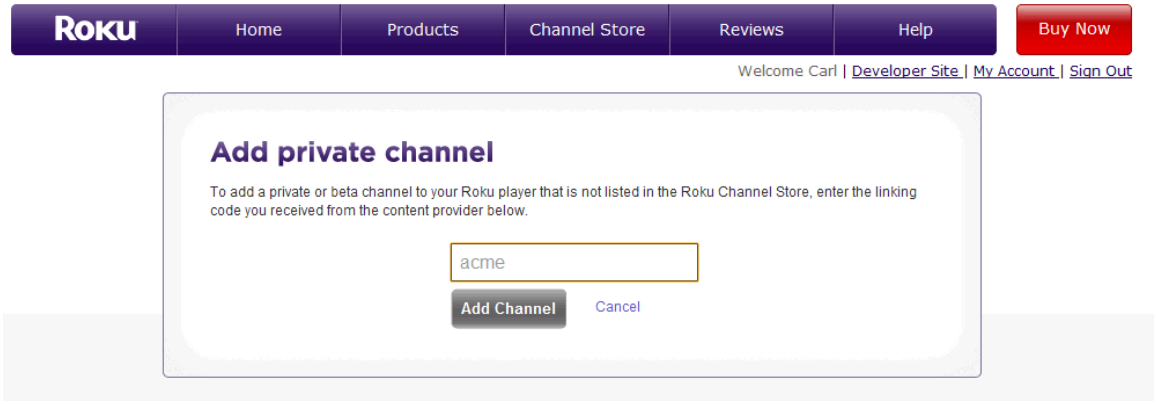
Notice that private channels have a “Publish” button rather than a “Submit” button at this step. Private channels may be instantly published by developers and do not require Roku approval.

4. **After clicking “Publish” to associate the package, the state of the package becomes “Published” and the stripe above your channel image turns purple to indicate the “Published” state. The green arrow has also turned grey and the “Unpublished Package” information has been moved to the “Published Package” box.**



When your channel is published, the channel vanity channel access code is assigned to the package and it now appears in the “Published Package” box. The “Unpublished Package” box is empty once again and ready for you to upload the next package version of your channel. Note that the stripe above your channel is now purple and the status is “Published”.

5. **Users can now install the channel via the Vanity Channel Access Code on the "Add Channel" page of their Roku account. You may also share a link with your user base that will take users directly to the Add Channel page with your channel code.**

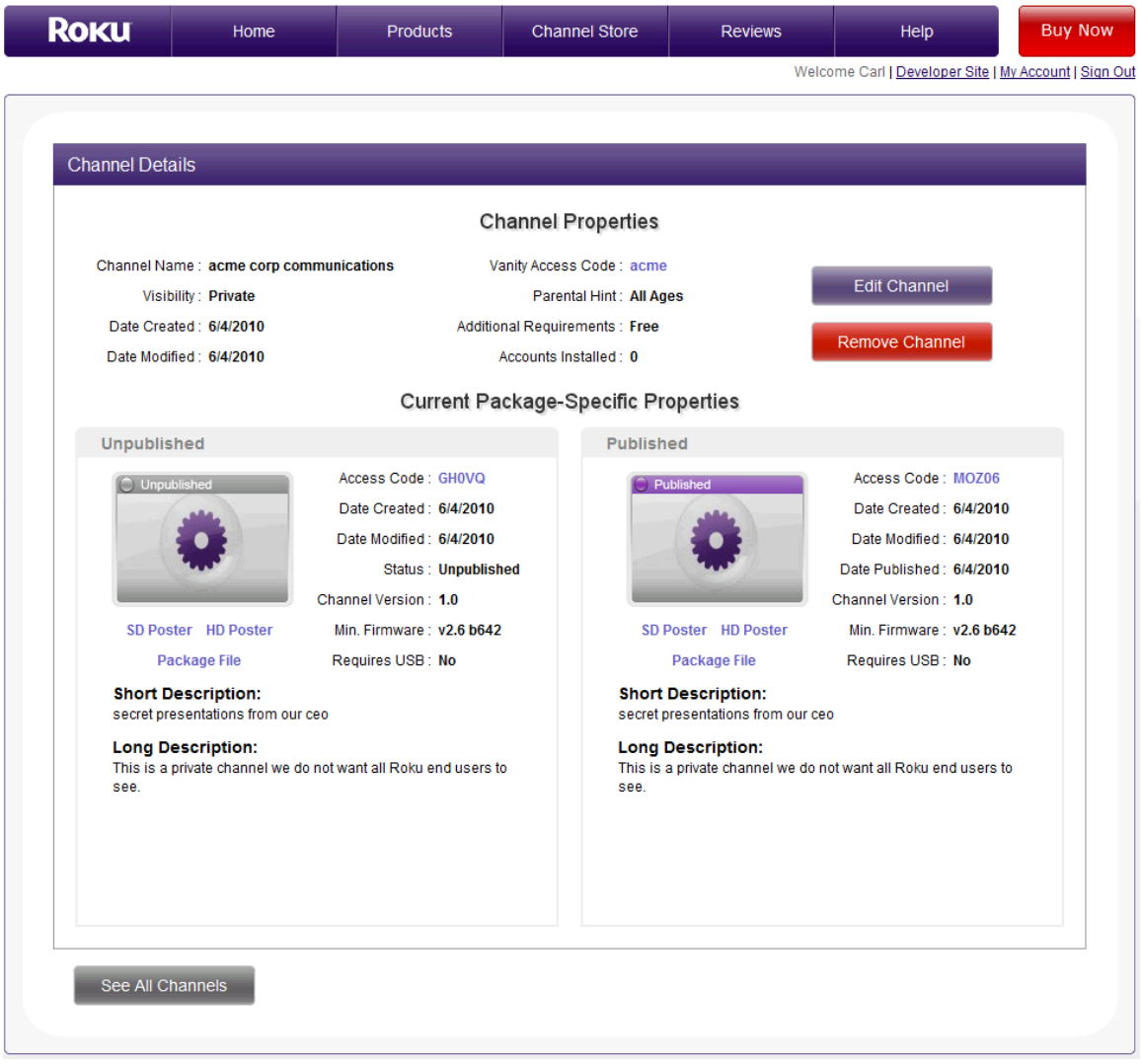


In this example, users just had to enter the "acme" channel code on the "Add Channel" page in their Roku account.

4.4 Removing a Channel from the Roku Channel Store

It is possible to remove a channel from a developer account so that the channel is not available in the Channel Store and cannot be added to a Roku Streaming Player. For users who have previously installed the channel, the channel will not be removed from the player. This is a drastic action as all versions of your channel will be removed from the channel store. So you will have to confirm the action with another click on a confirmation page.

Choose Remove Channel on the Channel page:



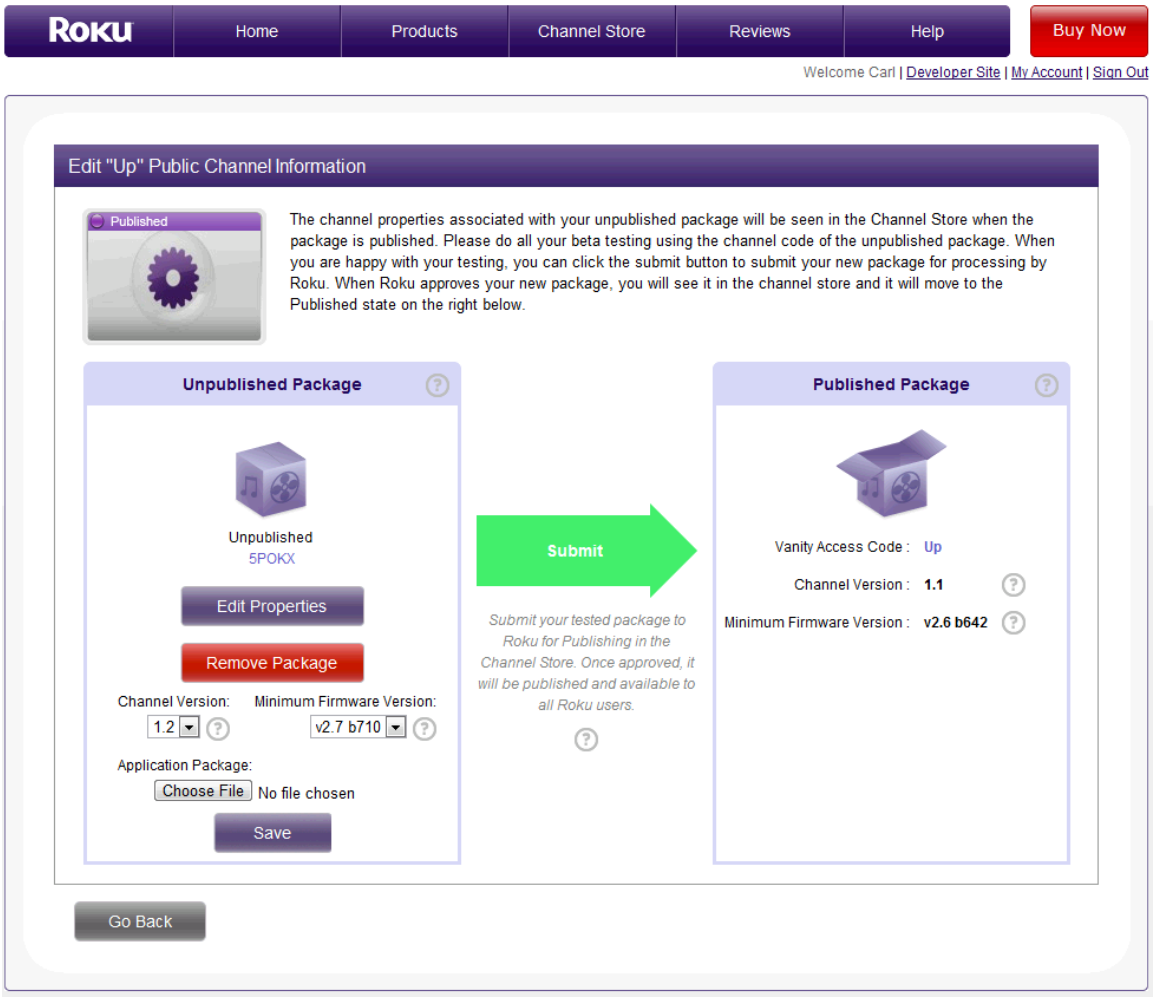
4.4 Updating Published Channels

You may “Submit” a public channel for publishing or “Publish” a private channel again at any time. Once the unpublished channel becomes the new published channel, the old published channel version is no longer accessible on the developer site. However, if it is the highest channel version that meets the Minimum Firmware Version dependency and Requires USB dependency for a particular Roku box, it may still be the chosen version to be installed by that box.

Note that the Minimum Firmware Version selections are monotonically increasing, so that as you enter new channel package versions with higher Minimum Firmware Version dependencies, you can no longer choose to publish packages with a lower Minimum Firmware Version dependency.

Warning:
One of the consequences of this versioning model is that you will no longer have the ability to change the historic published packages that have dependencies on older Minimum Firmware Versions. Yet, they could still be installed by any Roku boxes running that older firmware version.

Roku does control the upgrades of boxes, so the population of boxes running firmware that is two or more minor versions behind is very limited.



5.0 Common Scenarios

5.1 New channel version requires new firmware; leave old channel version on old firmware and only upgrade channel on new firmware boxes.

You may want to use some of the exciting new components in a new firmware version, like the new Image Canvas that is only available in firmware v2.6. In this scenario, users that are still running v2.4 firmware would continue to run the current package channel version you have uploaded to the channel store. Only Roku boxes running the new v2.6 firmware will run the new package channel version. As you code your new version, there is no need to do a version check or worry about the different behaviors of the different firmware versions. This application version will only run on Roku boxes that meet the Minimum Firmware Version requirement specified on the Channel Information page.

To do this, you will specify a Minimum Firmware Version of v2.6 when saving your new unpublished channel and then publish it.

This scenario eases the coding burden as you can rely on the minimum firmware version dependency and do not have to check this explicitly in your application code.

This should be the default usage pattern.

5.2 New channel version requires new firmware; force users to upgrade firmware in order to run

You may not want any channel version to run on v2.4 boxes, and instead just display a dialog box informing the user that they must upgrade their firmware to v2.6 in order to run your new channel version.

To make this happen, you would first code your package to check the firmware version that the app is running on. The app would then either display a paragraph screen informing the user that they must upgrade to firmware v2.6 or if already running v2.6 firmware continue executing your channel knowing that new components like the Image Canvas are valid to use. You will choose a Minimum Firmware Version of v2.4 when saving your new unpublished channel and then publish it.

That way the Channel Store does not filter this version from running on Roku boxes with v2.4 firmware. Instead your application code does the firmware version check and takes different actions based on the firmware it is running on.

The usbplayer sample application has this firmware version check and usage pattern.

5.3 New channel version has different behavior throughout your application based on the detected firmware it is running on.

As in scenario 5.2 above you will do a firmware version check in your application, but instead of displaying an upgrade screen and exiting your application you adjust the functionality of your application for the older firmware.

For example, before displaying a button whose action is to display an Image Canvas you could check if the firmware your application is running on is v2.6 or greater.

The usbplayer sample application has a function you can use to add these version checks to your code.

5.4 New channel version only runs on USB hardware

By specifying “Requires USB” on the “Channel Properties” page, you assure that your channel version will only be seen and delivered to Roku boxes that have a USB port. If you are developing a channel that is for a device that will plug into a USB port and export a filesystem you can specify this hardware dependency so that only boxes that have a USB port ever display or install your channel version.

5.5 New channel version only functions properly on USB hardware and informs the user to buy a Roku model with USB support.

You can leave the “Requires USB” setting on the “Channel Properties” page unset, and instead check the model your application is running on in your code. Depending if the currently running model supports USB or not you will either run your USB code or display a paragraph screen informing the user of the model dependency and encourage them to get a Roku model that has a USB port.

5.6 New channel has reduced functionality on non-USB hardware and enhanced functionality on USB hardware

Your channel may try to play local media feeds that it can find in local repositories. It may also try to play media off a USB drive if it is available. In this case, you would leave the “Requires USB” setting off on the “Channel Properties” page. It is fine to have code that adds media to posterScreens when it receives an isStorageDeviceAdded() event. You will never receive this event on models that don’t have a USB port, so the Model check in your code is unnecessary in this case.