LiveCode 6.7.0 Release Notes

Table of contents

Overview

This document describes all the changes that have been made for LiveCode 6.7.0, including bug fixes and new syntax.

Known issues

- The installer will currently fail if you run it from a network share on Windows. Please copy the installer to a local disk before launching on this platform.

Platform support

The engine supports a variety of operating systems and versions. This section describes the platforms that we ensure the engine runs on without issue (although in some cases with reduced functionality).

Windows

The engine supports the following Windows OSes:

- Windows XP SP2 and above
- Windows Server 2003
- Windows Vista SP1 and above (both 32-bit and 64-bit)
- Windows 7 (both 32-bit and 64-bit)
- Windows Server 2008
- Windows 8.x (Desktop)

Note: *On 64-bit platforms the engine still runs as a 32-bit application through the WoW layer.*

Linux

The linux engine requires the following:

- 32-bit installation, or a 64-bit linux distribution that has a 32-bit compatibility layer
- 2.4.x or later kernel
- X11R5 capable Xserver running locally on a 24-bit display
- glibc 2.3.2 or later
- gtk/gdk/glib (optional – required for native theme support)
- pango/xft (optional – required for pdf printing, anti-aliased text and unicode font support)
- lcms (optional – required for color profile support in JPEGs and PNGs)
- gksu (optional – required for elevate process support)

Note: *The optional requirements (except for gksu and lcms) are also required by Firefox and Chrome, so if your linux distribution runs one of those, it will run the engine.*
Note: *If the optional requirements are not present then the engine will still run but the specified features will be disabled.*
Note: *LiveCode and standalones it builds may work on remote Xservers and in other bit-depths, however this mode of operation is not currently supported.*

Mac

The Mac engine supports:

- 10.6.x (Snow Leopard) on Intel
- 10.7.x (Lion) on Intel
- 10.8.x (Mountain Lion) on Intel
- 10.9.x (Mavericks) on Intel

Note: *The engine runs as a 32-bit application regardless of the capabilities of the underlying processor.*

Setup

Installation

Each distinct version has its own complete folder – multiple versions will no longer install side-by-side: on Windows (and Linux), each distinct version will gain its own start menu (application menu) entry; on Mac, each distinct version will have its own app bundle.
The default location for the install on the different platforms when installing for 'all users' are:

- Windows: <x86 program files folder>/RunRev/ LiveCode 6.7.0
- Linux: /opt/runrev/livecode-6.7.0
- Mac: /Applications/ LiveCode 6.7.0.app

The default location for the install on the different platforms when installing for 'this user' are:

- Windows: <user roaming app data folder>/RunRev/Components/LiveCode 6.7.0
- Linux: ~/.runrev/components/livecode-6.7.0
- Mac: ~/Applications/ LiveCode 6.7.0.app

Note: *If your linux distribution does not have the necessary support for authentication (gksu) then the installer will run without admin privileges so you will have to manually run it from an admin account to install into a privileged location.*

Uninstallation

On Windows, the installer hooks into the standard Windows uninstall mechanism. This is accessible from the appropriate pane in the control panel.
On Mac, simply drag the app bundle to the Trash.
On Linux, the situation is currently less than ideal:

- open a terminal
- *cd* to the folder containing your rev install. e.g.

```
cd /opt/runrev/livecode-6.7.0
```

- execute the *.setup.x86* file. i.e.

```
./.setup.x86
```

- follow the on-screen instructions.

Reporting installer issues

If you find that the installer fails to work for you then please file a bug report in the RQCC or email support@runrev.com so we can look into the problem.
In the case of failed install it is vitally important that you include the following information:

- Your platform and operating system version
- The location of your home/user folder
- The type of user account you are using (guest, restricted, admin etc.)
- The installer log file located as follows:
- **Windows 2000/XP:** <documents and settings folder>/<user>/Local Settings/

- **Windows Vista/7:** <users folder>/<user>/AppData/Local/RunRev/Logs
- **Linux:** <home>/.runrev/logs
- **Mac:** <home>/Library/Application Support/Logs/RunRev

Activation

The licensing system ties your product licenses to a customer account system, meaning that you no longer have to worry about finding a license key after installing a new copy of LiveCode. Instead, you simply have to enter your email address and password that has been registered with our customer account system and your license key will be retrieved automatically.
Alternatively it is possible to activate the product via the use of a specially encrypted license file. These will be available for download from the customer center after logging into your account. This method will allow the product to be installed on machines that do not have access to the internet.

Multi-user and network install support (4.5.3)

In order to better support institutions needing to both deploy the IDE to many machines and to license them for all users on a given machine, a number of facilities have been added which are accessible by using the command-line.
***Note:** These features are intended for use by IT administrators for the purposes of deploying LiveCode in multi-user situations. They are not supported for general use.*

Command-line installation

It is possible to invoke the installer from the command-line on both Mac and Windows. When invoked in this fashion, no GUI will be displayed, configuration being supplied by arguments passed to the installer.
On both platforms, the command is of the following form:

```
<exe> install noui options
```

Here *options* is optional and consists of one or more of the following:

-allusers	Install the IDE for all users. If not specified, the install will be done for the current user only.
-desktopshortcut	Place a shortcut on the Desktop (Windows-only)
-startmenu	Place shortcuts in the Start Menu (Windows-only)
-location *location*	The location to install into. If not specified, the location defaults to those described in the *Layout* section above.
-log *logfile*	A file to place a log of all actions in. If not specified, no log is generated.

Note that the command-line variant of the installer does not do any authentication. Thus, if you wish to install to an admin-only location you will need to be running as administrator before executing the command.
As the installer is actually a GUI application, it needs to be run slightly differently from other command-line programs.
In what follows <installerexe> should be replaced with the path of the installer executable or app (inside the DMG) that has been downloaded.
On Windows, you need to do:

```
start /wait <installerexe> install noui options
```

On Mac, you need to do:

```
“<installerexe>/Contents/MacOS/installer” install noui options
```

On both platforms, the result of the installation will be written to the console.

Command-line activation

In a similar vein to installation, it is possible to activate an installation of LiveCode for all-users of that machine by using the command-line. When invoked in this fashion, no GUI will be displayed, activation being controlled by any arguments passed.
On both platforms, the command is of the form:

```
<exe> activate -file license -passphrase phrase
```

This command will load the manual activation file from *license*, decrypt it using the given *passphrase* and then install a license file for all users of the computer. Manual activation files can be downloaded from the 'My Products' section of the RunRev customer accounts area.
This action can be undone using the following command:

```
<exe> deactivate
```

Again, as the LiveCode executable is actually a GUI application it needs to be run slightly differently from other command-line programs.
In what follows <livecodeexe> should be replaced with the path to the installed LiveCode executable or app that has been previously installed.
On Windows, you need to do:

```
start /wait <livecodeexe> activate -file license -passphrase phrase
start /wait <livecodeexe> deactivate
```

On Mac, you need to do:

```
“<livecodeexe>/Contents/MacOS/LiveCode” activate -file license -passphrase phrase
“<livecodeexe>/Contents/MacOS/LiveCode” deactivate
```

On both platforms, the result of the activation will be written to the console.

Proposed changes

The following changes are likely to occur in the next or subsequent non-maintenance release:

- The engine (both IDE and standalone) **will require** gtk, gdk, glib, pango and xft on Linux

Engine changes

Cocoa Support (6.7.0)

With 6.7 we have replaced the majority of Carbon API usage with Cocoa. The goals of this work are three-fold:

- Allow embedding of native 'NSViews' into LiveCode windows (in particular, browser controls).
- Enable submission of LiveCode apps to the Mac AppStore.
- Enable eventual building of 64-bit versions of LiveCode for Mac.

We have achieved the first two of these goals in 6.7.

The instability issues caused by the AppStore sandbox when using mixed Cocoa and Carbon APIs has been resolved - LiveCode apps built with 6.7 can be successfully sandboxed and thus submitted to the AppStore.

The dontUseQT property is now true by default on Mac. This means that, by default, the AVKit implementation of the player will be used on 10.8 and above. Note that, as it stands, when dontUseQT is true neither QT visual effects nor sound recording will work.

The final goal (64-bit support) will be gradually worked towards over the next few LiveCode versions as the engine gets 'decarbonated' (usage of Carbon APIs which do not have 64-bit equivalents removed).

An important internal change which will affect maintainers of Mac externals that use the windowId is that this property now returns the 'global window number' (which is the unique ID the Window Server uses to identify windows). To turn this into a Cocoa NSWindow pointer use [NSApp windowWithWindowNumber: t_window_id]. Note that it is no longer possible to get a Carbon WindowRef, nor should this be attempted as trying to mix Carbon and Cocoa in this manner will cause instability inside the sandbox environment required by the Mac AppStore.

An important script visible change that has occurred due to the move to Cocoa is screen updating. Previously (when using Carbon) the OS would 'coalesce' successive requests to update the screen - the window buffer would be updated, but the window buffer would only be flushed when the OS decided to. In Cocoa, after a screen update the window buffer is *always* flushed. Outside of 'lock screen', the engine applies any screen updates after each command execution therefore in 6.7+ make sure you use lock screen around blocks of code that make many screen updates - unless you want each update to be visible. It should be noted that the behavior in 6.7 is now the same as on Windows and Linux however the OS takes longer to flush window updates to the screen on Mac than on the other platforms meaning that using lock screen is important.

Note: QTVR movies are no longer supported as they are not supported by QTKit nor AVKit.

Note: Drawers no longer work on Mac, they will appear as normal stacks.

Location Services Disabled with LC 6.6.4 (rc1) (6.7.0)

A new function **mobileLocationAuthorizationStatus** (or **iphoneLocationAuthorizationStatus**) has been added. This returns the current location authorization status of the calling application. The status can be one of the following:

- **notDetermined**: User has not yet made a choice with regards to this application
- **restricted**: The application is not authorized to use location service
- **denied**: User has explicitly denied authorization for this application, or location services are disabled in Settings.

- **authorizedAlways**: User has granted authorization to use their location at any time, including monitoring for regions, visits, or significant location changes.
- **authorizedWhenInUse**: User has granted authorization to use their location only when the app is visible to them (it will be made visible to them if you continue to receive location updates while in the background). Authorization to use launch APIs has not been granted.

We have also changed the flow of the messages being sent to the user when using Location Services in iOS 8:

- In the standalone application settings tab, the developer can choose the type of the authorization request for their app.
The two available options are either "always" or "when in use". Selecting "always" means that the app will prompt the user to grant authorization to use their location
at *any* time, including monitoring for regions, visits, or significant location changes. The app then has access to the user's location even when the app is in the
background. On the contrary, if "when in use" is selected, the app will prompt the user to grant authorization to use their location only when the app is visible on screen. You can choose only one type, not both. This means that if you go to Settings -> Privacy -> Location, you will see only two choices available ("Never" and either "Always" or "While using the app") for this app, keeping it consistent with other iOS apps.

- When the app is installed (on device or simulator) for the very first time, a dialog will pop up asking the user to authorize the app to use their location
"always" or "when in use", depending on what was previously chosen in the standalone application settings.

- Every time the app is launched, it remembers the user's preference. No other popup dialogs will appear.

- The user can at any time change their preferences in Settings -> Privacy -> Location -> ..

- In that way, you need not modify your existing scripts that used Location Services, in order to add iOS 8 support.

Multimedia on MacOS with AVFoundation (6.7.0-rc-3)

What has changed?

The player object until now used QuickTime/QTKit APIs for audio and video playback. Since both QuickTime and QTKit have been deprecated by Apple, we have updated the player to use the new AVFoundation API. AVFoundation does not provide a controller for multimedia playback until OSX 10.9 and their new control bar is also missing some of the features provided by the QTKIt controller, which required us to implement our own controller to ensure backward compatibility.
We have added two new properties to the player object enabling you to customise the appearance of the controller:

- The **hilitecolor** of a player is the color of the played area, the colour of the volume area, as well as the background color of a controller button when it is pressed.

- The **forecolor** of a player is the color of the selected area. The selected area is the area between the selection handles.

We have also added support for getting information about the download progress of a remote multimedia file:

- The **loadedtime** of a player is the time up to which the movie can be played. The download progress is also displayed on the controller well.

You can also query the **status** property of the player. This property can take either of the values:
- **loading** (for remote multimedia files)

- **playing**
- **paused**

A new message is added to the player:
- The **playRateChanged** message is sent to the player when the rate is changed by the rate scrollbar controller. To enable the rate scrollbar controller, hold shift + click on scrubForward/scrubBack buttons of the player controller.

Note AVFoundation player is supported in OSX 10.8 and above. On systems running OSX 10.6 and 10.7, LiveCode continues to provide player functionality using the QTKit API.

iOS 8 Support (6.7.0-rc-3)

Support for iOS 8 device and simulator builds has been added to 6.6.4-rc-1 for OS 10.9 users. This means that if you are using OS 10.9 you must now have Xcode 6 installed in order to perform device builds. The requirements for all previous OS X versions will remain the same.

Bugs relating to orientation, push notifications and screen sizes on iOS 8 have been resolved in addition to standalone builder updates allowing for the specification of new iPhone 6 splash screens.

Copy files do not work with the iOS 8 simulator (6.7.0-rc-3)

This fix has been tweaked for 6.6.4-rc-3. If, when attempting to deploy to the iOS 8 simulator you get the error "Unable to start simulation: Unable to run app in Simulator", delete any previous version of the app installed on the simulator and redeploy.

Fix OSX specialFolderPath("asup") (6.7.0-rc-3)

Mark the installer as retina-capable (6.7.0-rc-3)

Fix post-install launching on Linux (6.7.0-rc-3)

Use correct pixel order for OSX PPC (6.7.0-rc-3)

Non-executable file redirection on Mac (6.7.0-rc-2)

Mac AppStore rules require that only executables (including bundles and apps) are present within the Contents/MacOS folder in the application bundle.
However, historically (for cross-platform purposes), LiveCode applications traditional place resources relative to the engine executable, resulting in non-executable files to be present in the Contents/MacOS folder which violates AppStore signing policy.
To remedy this situation without requiring users to change scripts, a simple redirection facility has been implemented in the engine:
If an attempt is made to open a file for read which falls within Contents/MacOS and does not exist, the engine will attempt to open the same path but under Contents/Resources/_MacOS instead.
If an attempt is made to list files in a folder which falls within Contents/MacOS, the engine will list files in that folder and concatenate them will files within the same folder under Contents/Resources/_MacOS.
Additionally the standalone builder has had an extra processing step added on Mac:
After the Mac bundle has been built, the S/B recurses through Contents/MacOS and creates an identical folder structure based at Contents/Resources/_MacOS. All non-executable files in any folders under Contents/MacOS are moved to the same folder under Contents/Resources/_MacOS whereas any Mach-O executable files are left where they are.

The result of this is that after building a standalone, from a script's point of view nothing has changed; but the app bundle will conform to the rules required for signing for the Mac AppStore.

Fix a button focus issue (6.7.0-rc-2)

Determining the edition of the running engine (6.7.0-dp-9)

There is a new global property **the editionType**. This property can be used to determine what edition of the engine is currently running.
It returns *community* if the currently running engine is the LiveCode Community IDE or a standalone built with LiveCode Community.
It returns *commercial* if the currently running engine is the LiveCode Commercial IDE or a standalone built with LiveCode Commercial.

Sound recording returns error message (6.7.0-dp-9)

The Quicktime implementation of sound recording has been updated to use the SGAudioMediaType in place of the deprecated SoundMediaType, to fix various errors with sound recording. Moreover a pausing feature has been added to the sound recorder, via the syntax **record pause** and **record resume**.

Threaded Rendering (6.7.0-dp-8)

In an effort to boost graphic rendering performance, when LiveCode is run on a machine with multiple cores, all stacks with accelerated rendering turned off will be split into multiple tiles, with each tile being rendered individually (and ideally concurrently). The number of tiles the stack is split into depends upon the machine the stack is running on.

So, for example, on a machine with 4 or more cores, a stack will be split up into 4 rectangular tiles - top left, top right, bottom left, bottom right. (On dual core machines, we use a top half bottom half split). Each tile will be rendered individually, meaning that in an ideal situation, we have 4 concurrent drawing operations providing a 4 fold speed increase. This is the ideal, but in reality, in the above case, we see closer to a 2 fold improvement. This will obviously vary with the number of concurrent operations the given machine can perform.

Note that this feature is currently disabled on Windows due to threading conflicts caused by the player object.

Effective points of graphics (6.7.0-dp-7)

You can now use 'the effective points' and 'the effective relativePoints' properties of a graphic object to fetch a polygon representation of rectangle, round rectangle, oval and regular polygon graphic objects.

Player messages aren't sent correctly. (6.7.0-dp-7)

The occurrence of playStarted, playPaused and playStopped messages has been cleaned up.

The playStarted message will only be sent when the rate of the movie changes from zero to non-zero - whether via clicking the play button, setting the playRate or by using play start / play resume via script.

The playPaused message will only be sent when the rate of the movie changes from non-zero to zero - whether via clicking the pause button, setting the playRate or by using play pause / play stop.

The playStopped message will only be sent when the movie reaches the end of playback.

These are the only cases in which the messages will be sent - in particular, setting the filename will no longer send any messages and you will not get multiple messages of the same type in succession.

Showing a modal dialog confuses mouse state. (6.7.0-dp-6)

When a modal dialog is shown, the engine will now immediate reset the mouse state to ensure the context of events is the new dialog. In particular, mouseRelease will be sent if the mouse is down and mouseLeave will be sent if the mouse is within the previous window.

Nine-way stretch for images (6.7.0-dp-5)

You can now set 'the centerRect' property of an image. This property should be a rectangle, with co-ordinates relative to the formattedRect of the image.
The property specifies the area of the image that should be stretched when the image is scaled.
For example, if the centerRect of an image which is 16x16 is set to 4,4,12,12 then:

- The 4x4 corner portions of the image will not be stretched
- The top 4x8 and 8x4 side portions of the image will be stretched horizontally or vertically (depending on orientation)
- The middle 8x8 area will stretch to fill the middle.

This property is useful for using images as backgrounds to buttons and interface elements - allowing a non-stretched border with stretched interior to be specified.

Updated text rendering for iOS and OS X (6.7.0-dp-5)

In order to improve performance, the text rendering routines for iOS and OS X and been updated to use the latest APIs. This has had a significant improvement in the text rendering speed, particularly on OS X.

It's worth noting that the previous OS X routines used synthesised font styles. That is, bold and italic styles were emulated (by slanting or thickening) if the font being rendered was not bold or italic. This is not the case for the new routines. If there is not a font present on the system with the given style, the plain alternative will be used.

The is the case with the default font - Lucida Grande. By default, systems only come with the bold variant. Thus, if you have a field with the default font and italic style, it will be rendered in plain style.

As part of LiveCodes progression toward unicode, the use of older symbol fonts is no longer fully supported. In order to ensure symbol fonts are drawn correctly, the font must be unicode encoded. The fonts that come with the latest versions of OS X are all unicode encoded.

Export snapshot with metadata (6.7.0-dp-5)

An optional

```
with metadata <metadata array>
```

clause has been added to the

```
export snapshot
```

command. Currently the only metadata key that is implemented is

```
density
```

which can be used to include pixel density metadata in pixels per inch.

For example:

put 144 into theMetadataA["density"]

> export snapshot of group 1 at size the width of group 1 *2, the height of group 1* 2 with metadata theMetadataA

New variant of open and secure socket (6.7.0-dp-5)

New variants of open and secure socket have been added:

open secure socket *socket* **with verification for host** *host*

secure socket *socket* **with verification for host** *host*

The new host parameter allows the user to specify the host name the connection should be verified against. This is particularly useful if server your socket is directly connected to is not the end host you are talking to. For example when tunnelling through a proxy to connect to a HTTPS URL.

Multiple density image support for patterns. (6.7.0-dp-5)

This extends the existing image resolution independence features to any pattern using that image.
When an object pattern is set to a multiple density image, that pattern will automatically use the best source image for the density at which it is drawn.

arrayDecode crashes on linux with certain input. (6.7.0-dp-5)

After dragging onto a field when LiveCode is in the background, focus doesn't work properly until reset. (6.7.0-dp-5)

QT-related features don't work. (6.7.0-dp-5)

QT effects and sound recording will now work as long as 'dontUseQT' is set to false. In this case, the player will default to using QTKit.
If you are submitting an app to the Mac AppStore, or wish to use AVFoundation player on 10.8 and above, ensure that dontUseQT is set to true in your startup handler, or before any code or stack which uses QT is run.

Objects which are adjacent don't necessary appear so at non integral scale factors. (6.7.0-dp-5)

At non-integral scale factors (such as 150% Hi-DPI mode on Windows), objects which should appear next to each other can have a visible channel.
This issue isn't completely fixable due to the nature of approximations used when compositing to the screen. However, this problem has been mitigated in a couple of ways - firstly antialiasing is forced on whenever the scale factor is non-integral; secondly clipping rectangles always fall on device pixel boundaries.

Standalone engine still links to QTKit / QuickTime. (6.7.0-dp-5)

Queuing too many pending messages causes slowdown and random crashes. (6.7.0-dp-5)

A limit on the number of user-defined pending messages (those created with 'send in time') has been imposed. If there are more than 64k messages in the pending message queue, 'send in time' will now throw an error when attempting to queue another one.
This limit has been imposed to prevent engine lock up and eventual instability due to memory exhaustion in the case that pending message loops cause rapid increases in the number of pending messages.

Inconsistencies in behavior when doing 'delete the selectedChunk'. (6.7.0-dp-5)

The following should all operate the same way after selecting a line in a field by doing 'triple-click', or just selected the whole line without the paragraph break:

- pressing backspace
- executing 'delete the selectedChunk'
- executing 'get the selectedChunk; delete it'

Previously, 'delete the selectedChunk' would cause paragraph styles not to be set correctly on the resulting paragraph; or the paragraph break to be included when it should not be - this is no longer the case.
Previously, 'get / delete it' would only work correctly the first time the command was executed - this is no longer the case.

In-App Purchasing (6.7.0-dp-3)

Why has the API changed?

The LiveCode engine until now supported in-app purchasing for apps distributed through the Google Play store (formerly Android Market), as well as the Apple AppStore. This support is now extended so that apps distributed through other avenues (the Amazon & Samsung app stores) can make use of the in-app purchase features provided. For this reason, new LiveCode commands have been added, and some of the old ones have slightly changed. However, all of the old commands are still supported (for the Google Play Store and the Apple AppStore). In order the existing scripts users have written to continue to work, all it needs is to add one or two extra lines, depending on the store. More details on this later. Moreover, the new API allows the user to query specific product information (such as price, description etc) before they make a purchase, and supports purchasing of subscription items for all available stores. Furthermore, for the Google Play Store, the new API uses the newest version of Google In-App Billing API (v3), that offers synchronous purchase flow, and purchase information is available immediately after it completes. This information of in-app purchases is maintained within the Google Play system until the purchase is *consumed*. More on the consumption of purchased items later.

Note that our in-app purchasing implementation for the Amazon Appstore is unfortunately restricted to commercial license holders only. This is due to the Amazon PML license being incompatible with the GPL. If you wish to use the Amazon in-app purchasing features of LiveCode you will need to be a valid commercial licence holder.

What has changed?

To start with, the main changes are the following:

- Each item has an extra property, the *itemType*, that has to be specified before making a purchase. This is done using the **mobileStoreSetProductType** command. The *itemType* can either be *subs*, for subscription items, or *inapp* for consumable and non-consumable items.

- Due to a restriction of the newest version of Google In-App Billing API, you cannot buy consumable items more than once, unless you consume them. This is done using the **mobileStoreConsumePurchase** command. Note that this command is actually only used when interacting with the Google Play Store API. What it does is sending a consumption request to Google Play, so that you will be able to buy this product again. You would typically implement consumption for items that can be purchased multiple times (i.e. for consumable products, such as in-game currency, fuel etc). Note that in case you call mobileStoreConsumePurchase on a non consumable product, then you no longer own this item.

- The new purchase flow has become simpler.

Instead of

- creating a purchase request (**mobilePurchaseCreate** productID)

- store the new purchase request ID (put the result into tPurchaseID),

- setting properties such as quantity and developer payload (**mobilePurchaseSet** tPurchaseID, "quantity", pQuantity)

- sending a purchase request to the store (**mobilePurchaseSendRequest** tPurchaseID)

now all it needs is just

- set the product type (**mobileStoreSetProductType** productID, itemType)

- make a purchase (**mobileStoreMakePurchase** productID, quantity, developerPayload)

- The **purchaseStateUpdate** message that the store sends in response to **mobileStoreMakePurchase**, contains not only the purchase identifier and the state of the purchase, but also the product identifier of the requested item:

purchaseStateUpdate *purchaseID*, *productID*, *state*

- So you can query a purchased product property using the product identifier, instead of the purchase identifier:

mobileStoreProductProperty *productID*, *propertyName*

Note that the old function **mobilePurchaseGet** *purchaseID*, *propertyName* will still work.

- You can get information on a specific item (such as product identifier, product type, price etc), using the **mobileStoreRequestProductDetails** command. The store responds:

In case the request is successful, a **productDetailsReceived** message is sent by the store.

In case of failure, a **productRequestError** message is sent by the store.

- You can get a list of all known completed purchases using **mobileStorePurchasedProducts** function. This returns a list of product identifiers of restored or newly bought purchases.

What needs to change in existing scripts?

It is recommended that scripts which were written using previous versions of LiveCode (and thus use the old LiveCode API for in-app purchasing), should be used to run on these versions. However, it is still possible to run an existing script (that makes use of in-app purchasing feature) on LiveCode 6.7, only by changing a few things:

- **purchaseStateUpdate** message is now called with 3 parameters, (purchaseID, productID, state), instead of two (purchaseID, state). This applies to apps built for both the Google Play Store and the Apple AppStore.

- before sending a **mobilePurchaseSendRequest**, you have to specify the type (*subs* or *inapp*) of the item using **mobileStoreSetProductType** *productID*, *type* command (Google Play Store only).

- if you want to buy more than one consumable item, you have to consume it first. This can be done by using the **mobileStoreConsumePurchase** *productID* command (Google Play Store only).

If you want to build apps for Amazon and/or Samsung Store, you have to use the newest LiveCode API. Â

How to use the new API?

Setup

Before you can use IAP, you must set up products in each vendor's developer portal. In brief, you have to:

- Create each product you want to sell, giving it a unique identifier. Note that for the Samsung Seller Office, the developer cannot choose the product identifier. This is assigned by the store.
- Submit the items for approval to the appropriate store. Some stores may require additional metadata, such as screenshots of your for sale items.
- Set up unique test accounts. The user is not charged when making a purchase using the test account details. This applies to Apple and Google. Amazon and Samsung have different methods for testing.

For more detailed store-specific information, you can have a look at the links below:

Apple AppStore

Google Play Store

Amazon Appstore

Samsung Apps Store and more specifically click here

Purchase Types

There are three classes of products users can purchase:

1. One-time purchases that get "consumed". Typically, these items are called *consumables*. The user can buy as many times as they want (virtual coins/bullets in a game), except in apps built for the Google Play Store, where the user has to consume the purchased item first, and then buy (one) more.
2. One-time purchases that last forever, such as unlocking extra features, downloading new content once. These items are usually called *non-consumables*.
3. Subscriptions where the app user pays a periodical fee to receive some ongoing service. Subscriptions can either be auto-renewable or non-renewable.

Each vendor uses different terminology for these purchases :

	Apple	**Google**	**Amazon**	**Samsung**
one-time, gets consumed	consumable	unmanaged	consumable	consumable
one-time, lasts forever	non-consumable	managed	entitlement	non-consumable
	auto-renewable non-	auto-	auto-	non-

subscriptions	auto-renewable , non-renewable	auto-renewable	auto-renewable	non-renewable

Testing

Again, each store uses a different method of testing.

For the Apple AppStore, you can create test accounts. More details here.

For the Google Play Store, you can create test accounts as well as test using static responses. More details here. Note that you cannot test subscriptions using the test account. This means that the test user will be charged when purchasing a subscription item. A possible workaround to this, is to log into the Google Wallet Service as a seller, using your Google Developer account details, and "refund" and then "cancel" the order of the subscription item that the test user had just purchased.

For the Amazon Appstore, you can test your app using SDK Tester. This is a developer tool that allows users of the Amazon Mobile App SDK to test their implementation in a production-like environment before submitting it to Amazon for publication. More details here.

For the Samsung Apps Store, Samsung IAP API offers three modes to test the service under various conditions : *Production Mode, Test Mode Success, Test Mode Fail*. During development period, you can select the mode in the Standalone Application Settings window. Before releasing your application, you must change to Production Mode. If you release your application in Test Mode, actual payments will not occur. More details on page 6 and 7 here.

Note that in Production Mode, your app can only interact with item groups with *sales* status. This information exists in the Samsung Seller Office. However, item groups are only given sales status after the app has been certified. In other words, you can test your app in Production Mode only after it has been certified by Samsung.

Syntax

Implementing in-app purchasing requires two way communication between your LiveCode app and the vendor's store. Here is the basic process:

- Your app sends a request to purchase a specific in-app purchase to the store
- The store verifies this and attempts to take payment
- If payment is successful the store notifies your app
- Your app unlocks features or downloads new content / fulfils the in-app purchase
- Your app tells the store that all actions associated with the purchase have been completed
- Store logs that in-app purchase has been completed

Commands, Functions and Messages

To determine if in-app purchasing is available use:

mobileStoreCanMakePurchase()

Returns *true* if in-app purchases can be made, *false* if not.

Throughout the purchase process, the store sends **purchaseStateUpdate** messages to your app which report any changes in the status of active purchases. The receipt of these messages can be switched on and off using:

mobileStoreEnablePurchaseUpdates
mobileStoreDisablePurchaseUpdates

If you want to get information on a specific item (such as product identifier, product type, price etc), you can use:

mobileStoreRequestProductDetails *productID*

The *productID* is the identifier of the item you are interested. Then, the store sends a *productDetailsReceived* message, in case the request is successful, otherwise it sends a *productRequestError* message:

productDetailsReceived *productID*, *details*

The *productID* is the identifier of the item, and *details* is an array with the following keys - that are different depending on the store:

For Android stores (Google, Amazon, Samsung), the keys are:

- *productID* : identifier of the requested product
- *price* : price of the requested product
- *description* : description of the requested product
- *title* : title of the requested product
- *itemType* : type of the requested product
- *itemImageUrl* : URL where the image (if any) of the requested product is stored
- *itemDownloadUrl* : URL to download the requested product
- *subscriptionDurationUnit* : subscription duration unit of the requested product
- *subscriptionDurationMultiplier* : subscription duration multiplier of the requested product

Note that some Android stores do not provide values for all the above keys. In this case, the value for the corresponding key will be empty.

For iTunes Connect store (Apple), the keys of *details* array are the following:

- *price* : price of the requested product
- *description* : description of the requested product
- *title* : title of the requested product
- *currency code* : price currency code of the requested product
- *currency symbol* : currency symbol of the requested product
- *unicode description* : unicode description of the requested product
- *unicode title* : unicode title of the requested product
- *unicode currency symbol* : unicode currency symbol of the requested product

If **mobileStoreRequestProductDetails** is not successful, then a *productRequestError* message is sent :

productRequestError *productID*, *error*

The *productID* is the identifier of the item, and *error* is a string that describes the error.

Before sending a purchase request for a particular item, you have to specify the type of this item. To do this, use :

mobileStoreSetProductType *itemType*

The *itemType* can either be *subs* or *inapp*.

To create and send a request for a new purchase use:

mobileStoreMakePurchase *productID*, *quantity*, *developerPayload*

The *productID* is the identifier of the in-app purchase you created in the vendor's developer portal and wish to purchase. The *quantity* specifies the quantity of the in-app purchase to buy (iOS only - always "1" in

Android) . The *developerPayload* is a string of less than 256 characters that will be returned with the purchase details once complete. Can be used to later identify a purchase response to a specific request (Android only).

To get a list of all known completed purchases use:

mobileStorePurchasedProducts()

It returns a return-separated list of product identifiers, of restored or newly bought purchases which are confirmed as complete. Note that in iOS, consumable products as well as non-renewable subscriptions will not be contained in this list.

Once a purchase is complete, you can retrieve the properties of the purchased product, using:

mobileStoreProductProperty *(productID, property)*

The parameters are as follows:

- *productID* : identifier of the requested product
- *property* : name of the purchase request property to get

Properties which can be queried can differ depending on the store:

For the Samsung Apps Store (Android), you can query the properties:

- *title* : title of the purchased product
- *productId* : identifier of the purchased product
- *price* : price of the purchased product
- *currencyUnit* : currency unit of the product price
- *description* : description of the product as specified in the Samsung Seller Office
- *itemImageUrl* : URL where the image of the purchased product is stored
- *itemDownLoadUrl* : URL to download the purchased product
- *paymentId* : payment identifier of the purchased product
- *purchaseId* : purchase identifier of the purchased product
- *purchaseDate* : purchase date, in milliseconds
- *verifyUrl* : IAP server URL for checking if the purchase is valid for the IAP server, using the *purchaseId* value

For the Google Play Store (Android), you can query the properties:

- *productId* : identifier of the purchased product
- *packageName* : application package from which the purchase originated
- *orderId* : unique order identifier for the transaction. This corresponds to the Google Wallet Order ID
- *purchaseTime* : time the product was purchased, in milliseconds
- *developerPayload* : developer-specified string that contains supplemental information about an order. You can specify a value for this in **mobileStoreMakePurchase**
- *purchaseToken* : token that uniquely identifies a purchase for a given item and user pair.
- *itemType* : type of the purchased item, *inapp* or *subs*
- *signature* : string containing the signature of the purchase data that was signed with the private key of the developer. The data signature uses the RSASSA-PKCS1-v1_5 scheme

For the Amazon Appstore (Android), you can query the properties:

- *productId* : identifier of the purchased product
- *itemType* : type of the purchased product. This can be *CONSUMABLE*, *ENTITLED* or *SUBSCRIPTION*
- *subscriptionPeriod* : string indicating the start and end date for subscription (for subscription products only)
- *purchaseToken* : purchase token that can be used from an external server to validate purchase

For Apple AppStore (iOS), you can query the properties:

- *quantity* : amount of item purchased. You can specify a value for this in **mobileStoreMakePurchase**
- *productId* : identifier of the purchased product
- *receipt* : block of data that can be used to confirm the purchase from a remote server with the iTunes Connect store
- *purchaseDate* : date the purchase / restoration request was sent
- *transactionIdentifier* : unique identifier for a successful purchase / restoration request
- *originalPurchaseDate* : date of the original purchase, for restored purchases
- *originalTransactionIdentifier* : the transaction identifier of the original purchase, for restored purchases
- *originalReceipt* : the receipt for the original purchase, for restored purchases

Once you have sent your purchase request and it has been confirmed, you can then unlock or download new content to fulfil the requirements of the in-app purchase. You must inform the store once you have completely fulfiled the purchase using:

mobileStoreConfirmPurchase *productID*

Here, *productID* is the identifier of the product requested for purchase.

mobileStoreConfirmPurchase should only be called on a purchase request in the *paymentReceived* or *restored* state (more on the states of the purchase later). If you don't send this confirmation before the app is closed, **purchaseStateUpdate** messages for the purchase will be sent to your app the next time updates are enabled by calling the **mobileStoreEnablePurchaseUpdates** command.

To consume a purchased product use:

mobileStoreConsumePurchase *productID*

Here, *productID* is the identifier of the product requested for consumption. Note that this command is actually only used when interacting with the Google Play Store API. This is because the Google Play Store API has a restriction that ensures a consumable product is consumed before another instance is purchased. *Consume* means that the purchase is removed from the user's inventory of purchased items, allowing the user buy that product again.

Note that **mobileStoreConsumePurchase** must only be called on consumable products. If you call **mobileStoreConsumePurchase** on a non-consumable product, then you no longer own this product.

To instruct the store to re-send notifications of previously completed purchases use:

mobileStoreRestorePurchases

This would typically be called the first time an app is run after installation on a new device to restore any items bought through the app.

To get more detailed information about errors in the purchase request use:

mobileStorePurchaseError *(purchaseID)*

The store sends **purchaseStateUpdate** messages to notifies your app of any changes in state to the purchase request. These messages continue until you notify the store that the purchase is complete or it is cancelled.

purchaseStateUpdate *purchaseID, productID, state*

The state can be any one of the following:

- *sendingRequest* : the purchase request is being sent to the store / marketplace
- *paymentReceived* : the requested item has been paid for. The item should now be delivered to the user and confirmed via the mobileStoreConfirmPurchase command
- *alreadyEntitled* : the requested item is already owned, and cannot be purchased again
- *invalidSKU* : the requested item does not exist in the store listing
- *complete* : the purchase has now been paid for and delivered
- *restored* : the purchase has been restored after a call to mobileStoreRestorePurchases. The purchase should now be delivered to the user and confirmed via the mobileStoreConfirmPurchase command
- *cancelled* : the purchase was cancelled by the user before payment was received
- *error* : An error occurred during the payment request. More detailed information is available from the mobileStorePurchaseError function

Clipboard data 'styledText' array accessor. (6.7.0-dp-1)

A new clipboard format has been added 'styledText'. This format returns (or sets) the clipboard to a styled text array - the same format as the 'styledText' property of field chunks. All text formats can convert to and from the 'styledText' key.
For example, you can now do:
set the clipboardData["styledText"] to the styledText of line 5 of field 3
set the styledText of line 6 of field 3 to the clipboardData["styledText"]

Note that the dragData can now also be used with this new format in exactly the same way.

Improved revBrowser external (6.7.0-dp-1)

The revBrowser external has been updated to support Cocoa on OSX, and now embeds the browser control properly within the window.
In addition a new browser component based on CEF (Chromium Embedded Framework) has been added.

This new browser allows for a consistent appearance across all platforms with a modern, well supported feature set.

To use the new CEF browser use the *revBrowserOpenCef* command in place of *revBrowserOpen*. This will create a CEF browser instance which can be used with the existing revBrowser commands and functions in exactly the same way as before.

JavaScript integration

The new chrome browser allows us to add the ability to call LiveCode handlers from within the browser using JavaScript. To make a LiveCode handler visible to JavaScript, use the *revBrowserAddJavaScriptHandler* command, and to remove it use the *revBrowserRemoveJavaScriptHandler* command. LiveCode handlers are added as functions with the same name attached to a global 'liveCode' object. When called, these functions will result in the corresponding LiveCode handler message being sent to the browser card with the browser instance ID and any function arguments as parameters.

Example:

With the handler "myJSHandler" registered using *revBrowserAddJavaScriptHandler*, it can be called from the browser like so:

liveCode.myJSHandler(tFieldContents, tAction);

the LiveCode handler would then be called with the following parameters:

- pBrowserInstance (the browser instance id, as returned from the *revOpenBrowserCef* function)
- pFieldContents (the first argument of the JavaScript function call)
- pAction (the second argument of the JavaScript function call)

OS 10.5 (Leopard) Support (6.7.0-dp-1)

As of version 6.7-dp-1, Mac OS 10.5 (Leopard) support has been dropped from LiveCode. This is primarily for technical reasons: In order to support the latest OS X features (e.g. Cocoa) dropping 10.5 support was required.

As Leopard was the last Mac version to support PPC, support for the PPC architecture has also been dropped and the Universal and PPC options have been removed from the Standalone Builder.

Users wishing to produce 10.5 compatible executables can still do so using LiveCode version 6.6.x (and earlier).

Setting the label of an option or combo-box does not update the menuHistory. (6.7.0-dp-1)

Previously, setting the label of an option or combo-box control would not update the menuHistory property. Now, setting the label of such a control will search through the list of items in the control and set the menuHistory to the first item that matches (taking into account the setting of the caseSensitive local property).
Note: Unlike setting the menuHistory property direct, this does not cause a menuPick message to be sent.

pixelScaling not enabled on Windows Commercial edition (6.7.0-dp-1)

Specific bug fixes (6.7.0)

(bug fixes specific to the current build are highlighted in bold, reverted bug fixes are stricken through)

13753	**Project Browser reports incorrect control for behavior of a card**
13752	**Double-Clicking On a Player Doesn't Show Inspector**
13750	**Picker broken on iPhone 4 iOS 7.1**
13746	**the shape property of stacks is broken**
13721	**Externals using 'LCObjectPost' don't always cause the action to trigger - particularly on Yosemite.**
13711	**Player plays audio but not video**
13710	**[[Player]] video image not shown under some circumstances**
13708	**mobilepickphoto in landscape orientation causes an orientation change**
13707	**[[iOS 8]] Denying access to location services when the app is launched for the very first time causes the app to freeze**
13699	**iOS 8 Keyboard is invisible if privacy set to "While Using the App"**
13677	**iOS Picker appears under the keyboard on iOS 8**
13675	**Scrollbar for the font selection in the script editor preferences doesn't work**
13665	**Ask/answer calls in (pre)openstack cause iOS 8 apps to hang**
13622	**Make sure PATH variable passes through to shell() properly on Yosemite.**
13590	**Location Services Disabled with LC 6.6.4 (rc1)**
13510	**Shutdownrequest message sent twice when triggered from quit in menu or Cmd-Q on Mac.**
13493	**Scroll is being reset in 6.7 when it is not in 6.5.2**

13403	**Text wraps when field width set to formattedWidth**
13360	**LiveCode application takes up 98% of processor**
13351	**printing a field with listbehaviour set to true makes gray background**

Specific bug fixes (6.7.0-rc-3)

13662	OS X standalone can't be run out of app bundle
13659	When Voice Over is turned on
13656	iOS 8 ask and answer dialogs do not handle rotation correctly
13650	Crash when opening stack
13644	wait loop not being broken
13639	mobilepickcontact works under ios 7 but not under ios 8
13634	screenshots taken in landscape view are rotated by 90 degrees on iOS 8
13621	mobileFindContact fails silently on iOS 7.1 and higher
13584	Simulator launches with incorrect version
13583	Copy files do not work with the iOS 8 simulator
13569	changes to [[Player]] in preOpenCard are visible to user
13568	Extra undo of paint tools crashes live code
13555	keydown event not getting passed to mainstack in modal dialogs
13540	[[Player]] Shift + click in controller sets showSelection to true
13539	menuPick not triggered under certain conditions
13535	Threaded rendering crash
13526	Stack location reported incorrectly if mouse released while dragging window
13523	Fix OSX specialFolderPath("asup")
13522	pull down menus do not work properly in modal dialogs
13516	if an error is encountered after a drag-and-drop
13512	4 inch iPhone apps do not use the full screen
13503	PDF printing does not work correctly on iOS 8.
13501	Referenced image fails to load in 6.7.0 RC2 and 7.0 RC1
13485	Manifest file not needed in standalone bundle
13484	mobilePick and mobilePickDate do not work in iOS 8
13462	revPrintField clips document under some circumstances
13451	RGB imageData values (charToNum) are different on Mac / Windows
13349	Go stack in window displays new stack before before preopenStack/preopenCard messages are triggered
13317	Mark the installer as retina-capable
13236	mobilePickPhoto camera view is rotated on iPad when in landscape or in portraitUpsideDown
13208	Image file color profiles don't seem to be handled correctly
12876	Fix post-install launching on Linux
12464	The effective screenrect returns incorrect values when hiding/showing keyboard on iOS
12142	mobileSensorReading("location"
11968	Use correct pixel order for OSX PPC
11817	major speed degradation between 7.x

Specific bug fixes (6.7.0-rc-2)

13458	Memory leak when setting the text of an image

13455	Non-executable file redirection on Mac
13447	Project Browser control layer display
13445	Crash due to multicore rendering.
13442	rawKeyDown for arrowKey up and down fires twice when holding altKey down
13429	Fix crash when using magnifier tool.
13423	Dead char followed by an invalid character sends an invalid RawKeyDown message
13417	IDE systemVersion comparison no longer works with Yosemite
13398	Sample - Book Library.livecode edit and delete features broken
13393	playloudness should be rounded to nearest integer rather than down
13389	[[Player]] Can't make left to right selection
13374	LiveCode hangs when setting the text of a field with HGRID and a hidden last line
13372	Control-Tab key combos not being passed to rawKeyDown
13362	Script editor opens revmenubar script when no other stack is open
13354	App store no longer allows use of deprecated MPMoviePlayerContentPreloadDidFinishNotification
13343	Cannot install Android standalone on some devices
13342	[[Player]] - Shift + Space Bar whilst focus is on player should create selection
13328	Setting menuhistory with a stack panel sends invalid menuPick second parameter
13310	[[Player]] add new property loadedTime to the dictionary
13309	[[Player]] setting playLoudness by script does not update speaker icon
13306	[[Player]] loadedTime initially has the value of last movie
13295	Image distortion when setting imageData to the imageData.
13286	[[CEF browser]] Closing a stack without closing the browser
13278	Fix a button focus issue
13222	Moving graphic while editing gradient causes artifacts
13119	[[Player]] new controller does not have playRate scrollbar control
13092	[[Player]] Status property requested for player object
13048	Touch on Movie does not work
13006	[[Player]] Make controller scrollbar slightly wider
12830	[[Player]] Movie not updated instantly when using QT player and alwaysBuffer is true
10703	import snapshot from object is broken

Specific bug fixes (6.7.0-rc-1)

13284	Mouse is still inside the window even when resizing
13279	rawKeyDown passes the wrong keycode if Ctrl is pressed.
13273	templateImage framecount is not accurate for animated GIF
13272	Setting the scrollbars property of a CEF browser to false has no effect.
13270	SSL doesn't work with MySQL driver on Android and iOS.
13268	[[Player]] empty player shows image of last video when loading new video
13267	Thumb does not update properly when movie is playing in some circumstances.
13264	App crashes when showing referenced images on Android devices
13261	Visual effect push problem
13256	htmlText with many nested styles can cause a crash.
13250	Crash when rendering Mac themed scrollbars
13243	[[Player]] Player object retains callbacks even though callbacks set to empty

13240	Test System crashes reliably
13230	Polygon markers draw incorrectly
13221	Artifacts can appear in bitmap effects when multicore rendering is used.
13220	Polyline with same starting point as ending point draws as degenerate dot in PDF printing.
13215	Can't type in output field of message box
13201	textFont in Text Formatting of inspector cannot scroll by a mouse
13196	Hirigana input source causes LiveCode to hang when entering 'h' then 'a'.
13193	[[Player]] LC hangs when you open a stack with a player with filename that does not exist
13191	FIX: flip graphic horizontally and vertically for complex graphics
13190	iOS standalone building fails with "cannot find valid identity"
13187	[[Player]] Printing players doesn't work
13174	Text is clipped when printing to PDF from OS X
13159	Palettes not observing decorations under certain circumstances
13137	Setting currentTime of a player in response to a currentTimeChanged message can cause a hang.
13121	[[Player]] progress of movie downloaded/playable not indicated in controller well
13118	Add Hi-DPI support option to Windows standalone settings dialog.
12880	File->Exit should be File->Quit
12361	
12327	Toggle usePixelScale property
11755	flip graphic gives erroneous results with complex graphics

Specific bug fixes (6.7.0-dp-9)

13172	Auto detecting WPAD on OS X added
13171	WPAD URLs attempted to be used as proxies on OS X
13170	Auto config pac processing broken on OS X
13167	Crash when stack loses focus.
13166	Tab key doesn't insert tabs into fields that should accept them.
13163	showAll and LetterBox fullscreenmodes break on iOS
13161	Setting the playloudness of the templateAudioClip adjusts the system volume
13160	playloudness does not update when the user presses the mute button on keyboard
13156	[[player]] Step back button and Callback Problems
13155	Crash setting iconGravity - only one line needed
13154	Crash when attempting to decode malformed arrayEncoded value.
13149	Crash when resizing fields containing tabbed text
13140	Quitting from the dock when app is in background waits until app is foreground to actually terminate.
13127	centerRect property has no effect when image resizeQuality is "best"
13120	[[Player]] cmd key + click on step forward and step backward button for rewind
13109	crash dragging player to an empty stack
13099	[Player] Playback is locked when alwaysBuffer is true and video is playing
13091	[[player]] callback feature doesn't work when user moves scrubber
13064	[[Player]] playSelection not working correctly
13063	[[Player]] selection by shift + click Play button delayed start and stop.
13016	Erratic behavior whilst moving a window along with excessive WindowServer CPU usage.
12966	revVideoGrabber - revVideoGrabDialog "audio" does not bring up dialog

12885	Sound recording returns error message
12867	Gradient colours display incorrectly on android
12798	libURL doesn't report actual redirect error
2902	No error indicated if malformed URL used in url expression

Specific bug fixes (6.7.0-dp-8)

13132	put the engine folder crashes in OSX 10.6
13117	multi-line button label shows incorrectly - backslash n not working
13116	Crash while drawing Mac themed controls
13114	Cmd+Ctrl+ doesn't generate keyDown messages and causes a system beep.
13113	The extents don't work if there are negative indicies.
13112	Crash while resampling images
13105	[[Player]] CurrentTimeChanged message not sent by 'step forward' or 'step backward' buttons
13104	[[Player]] updating of playRate by controller buttons inconsistent
13102	Crash while resizing field
13101	Crash when resizing fields
13098	Composition sequences can result in incorrect output if they are invalid (for example alt-e g would give gg).
13096	Answer files with relative path
13073	Initial appleEvents not handled correctly.
13072	Focusing on nothing inside focusIn / enterField will cause a crash.
13065	[[Player]] making selection with shift + click player thumb inconsistent results
13059	menupick message not sent from popup stacks
13058	Wrong mouseLeave messages if the stack is pixelScaled or scaled in any fullscreen mode.
13038	uuid: not enough randomness available
13021	high dpi images don't fill image rect if scaled images don't have the correct size
12852	Setting the label of a button after the unicodeLabel can cause an infinite loop in some cases.
9941	Accented characters in string passed to revSpeak cause nothing to be spoken.

Specific bug fixes (6.7.0-dp-7)

13046	Player doesn't wait to make sure currentTime has updated to where it has been set to.
13026	put the engine folder behaves inconsistently
13025	Linux GTK spinboxes were inverted
13011	currentTimeChanged not sent when dragging controller thumb.
13010	If an object is in a group then mouseStillDown will not fire if it is in a behavior of the object.
13009	ImageData copying to offscreen images is broken
13007	Native encoded characters > 127 don't appear in menuPick string.
13005	Modifying selection by shift + click in the controller doesn't work correctly.
13004	Controller not updated when setting currentTime by script.
13003	selectionChanged message not sent when player selection changes via controller.
13002	Empty selection results in selection of full movie.
13001	Crash when changing focus inside openField handler.
13000	Image colours not showing correctly on Android device
12997	Changing windowShape leaves artifacts behind.
12990	Popup menus always highlight first item on Mac

12989	Player incorrectly reports timeScale and duration when using AVFoundation.
12979	Setting the rect of a player object doesn't work correctly if the stack is pixelScaled or scaled in any fullscreen mode.
12978	Development -> Plugins opens the wrong plugin
12968	ctrl z and edit -> undo are not working
12963	Player messages aren't sent correctly.
12962	Player won't play video in certain circumstances.
12961	Print dialogs do not keep new settings if displayed as sheet.
12946	play stop command does not work.
12944	Text does not align at correct tabStops in some cases.
12923	Setting the text style in the Text menu does not update correctly
12865	LiveCode Server process does not end when run as CGI
12864	The contents of a folder added to the Copy Files section of the Standalone Application Settings are not copied during a standalone build when it's a subfolder of the defaultFolder and the defaultFolder contains the main stack file.
12844	Attempted to arrayDecode non-arrayEncoded data can cause a crash.
12822	LiveCode server fails to read from https URLs on Fedora
12819	revAvailableHandlers works on password protected stacks.
12804	Clipping incorrect when printing to PDF.
12594	NUL bytes in fields will cause printing to pdf to fail silently.
12429	Setting the defaultNetworkInterface more than once causes instability.
12298	Autocomplete text in Message Box does not line up with text you are typing
12250	File view in file dialog doesn't update if you change the filtered types.
11828	Use 64-bit file APIs on Linux
11797	scriptLimits no longer enforced
9405	Clear previous highlight before drawing Linux GTK button highlight

Specific bug fixes (6.7.0-dp-6)

12939	Closing a modal dialog causes a pause until a system event occurs.
12932	Stacks can receive mouseDown before resumeStack.
12930	Engine crashes when moving stack after closing its parent
12928	IDE: Development > Rev Online != menubar > user samples
12902	iOS standalones won't build due to missing template files.
12898	Showing a modal dialog confuses mouse state.
12897	Popup Menu crashes LiveCode 6.7.0 DP5 standalones
12894	centerRect and iconGravity need to be added to the properties
12892	Certain images have red & blue swapped when deployed to iOS
12886	the enabledTracks of a player is formatted incorrectly.
12882	Wrong button sent to mouseDown / mouseUp when targetting a card.
12879	Inspector opening at odd location
12875	Shift-click on play button doesn't create selection.
12871	Clicking in the well of a player with a selection should clear the selection.
12870	Make sure controller is redrawn after setting selection by script.
12869	Behavior of shift-click in well on player not correct.
12863	RevVideo Grabber preview in wrong location after resizing stack
12860	Android stack drawn at wrong scale until forced redraw of object

12842	copying imageData between images can lead to unpredictable results
12829	6.7.0-dp-5 Mac installer fails if system security permissions are set
12825	playStopped sent for player with no filename
12764	setting the effective rect of a stack with vscroll > 0 doesn't work properly
12720	Focus gets confused if focus changes in response to a suspendStack message on Mac
12426	iOS font selection does not work correctly.
12296	Browser: Closing revBrowserCef when downloading makes LiveCode unstable
12131	Tool Palette location forgotten between launches
11620	pixelScale global property not documented
11375	In Dictionary the search box is only apparently selected when you come from Code Editor

Specific bug fixes (6.7.0-dp-5)

12841	Crash when switching to Chinese input method on Mac.
12835	Player: scaling a player down causes controller to get confused
12833	Player: hilite handles do not use fill length of the bar
12831	arrayDecode crashes on linux with certain input.
12826	answer file with type doesn't work correctly if only one type is specified.
12824	Windows position in the wrong place when constrained by the windowBoundingRect.
12821	Player callbacks not operating correctly.
12818	[[Player]] Selection thumbs should not be visible when selection duration is 0
12817	[[Player]] Selection not created when clicking shift and dragging player thumb
12816	[[Player]] SelectionStart and SelectionFinish handles too large and can obscure player thumb
12815	[[Player]] Selection indicator does not align with the selection thumbs
12812	[[Player]] loop goes to beginning of movie not selection start time when playSelection is true
12810	[[Player]] controller icons not updated when keyboard shortcuts used to control playback
12809	[[Player]] put the tracks of player 1 does not work properly
12806	message box is slow
12800	Go stack in window [windowId] doesn't work.
12799	On Mac
12794	The centerRect image property doesn't handle hi-res images correctly
12793	Plugins don't load in revBrowserCEF on OSX
12780	IDE stacks white rather than grey
12773	After dragging onto a field when LiveCode is in the background
12769	setting dragData[files] does not work
12765	The effective rect of a stack with vscroll > 0 is incorrect
12763	Player: Setting player size to < 132 width breaks some controller elements
12761	Player: dragging the in selection hilite marker moves out marker also
12760	Player: setting the filename to a local file that doesn't exist crashes LC
12759	Player: Setting the "in" marker for selection playback beyond the start point of player sets marker to unexpected value
12758	Player: setting the filename to a URL that isn't a video crashes LC
12757	[[Player]] Selecting "track" from a players property inspector
12756	Player: Can't select any audio files
12753	Player: Click outside of a selection allows video to be played outside selection

12751	QT-related features don't work.
12750	Player: Progress circle and end hilite don't light up
12747	Shortcuts: the uncomment script shortcut cmd _ does not work
12746	Player: First frame of video is not loaded immediately when filename is set
12745	Player: File chooser doesn't filter all available video formats
12737	Player: Can't drag out or create a player in script
12731	Player: Hiding and showing resized player changes player size to original size
12722	Unable to use edit mode when video is playing with new player object
12719	zipalign tool not found during standalone build after update to Android SDK tools
12715	Incomplete stack drawing when opening stack with acceleratedRendering on retina display
12709	Project Explorer not updating after stack was changed to substack
12708	Submenus of popups don't send menuPick on selection.
12705	Fix sending of mouseRelease messages with new platform layer
12702	Editing image then switching card and saving causes stack corruption
12701	CEF browser crashes if htmltext is set to empty
12699	Images don't appear or are clipped when printed to PDF
12690	Some fonts have accents cut off on capital letters on Mac.
12688	Blocking socket calls always timeout.
12687	Can't deselect MacOS build
12686	File and folder dialogs incorrectly use the topStack to sheet against
12671	CEF browser pauses frequently when there is no other activity on the stack
12670	Extra mouseMove with incorrect co-ordinates sent after mouseEnter when changing windows.
12669	WebAuthenticationPanel class in OSX revbrowser conflicts with same class in WebKit library
12668	File handle leak on Mac
12648	Shell command does not accept spaces despite being quoted (Windows)
12647	Multiple moves created whilst lock moves in effect fail to be synchronized.
12646	Crash when fetching the alphadata of a resized image
12636	Entries in the Project Browser won't expand
12634	Cursor does not change correctly when over a revBrowser[CEF] instance.
12632	minHeight setting on Mac includes title bar height when it shouldn't.
12631	CEF browser returns incorrect values for rect property
12628	Instability when using revAppendXML and revCopy/MoveRemoteXMLNode.
12612	Use sub-pixel positioning for laying out text within fields
12602	revBrowser placed incorrectly when dpi scaling enabled on Windows
12599	Redraw slowdown in 6.7 (regression)
12593	setting effective rect to working screenrect fails
12590	Screen updates occur during 'menu update' mouseDown message causing pauses when updating menus on first click.
12589	Pasting text into a field can sometimes cause strange selection behavior.
12578	'listIndent' attribute does not round-trip through htmlText
12567	Connecting to an HTTPS URL via a proxy fails if libURLSetVerification is true
12566	Tunnelled proxies do not authenticate correctly
12557	Objects which are adjacent don't necessary appear so at non integral scale factors.
12556	The rtfText does not represent 'metadata' tags correctly.
12549	Hiding / deleting a stack doesn't update the mouseStack when it should.
12543	Standalone engine still links to QTKit / QuickTime.

12529	LC 6.7 dp4 plays an imported wav only once
12528	Project Browser does not scroll down to show everything
12524	Hiding player controller stretches movie image vertically
12523	[[Player]] Setting playRate of player has no effect on playRate
12512	player currentTimeChanged message does not include time parameter
12506	Instability with manipulating QTKit players.
12501	Setting callbacks in player causes crash
12495	[[Bug 12495]] Animating windowShape does not work properly on Mac.
12481	Various actions on players (such as hiding and showing) prevent it from working properly.
12479	Maximum number of paragraphs which can be set with styledText is 64k
12470	Terminal window appears when accessing User Samples on Windows
12468	Middle button paste doesn't work correctly in other apps when LiveCode has the selection on Linux.
12467	Changing decorations causes no cursor to appear over a stack.
12463	Queuing too many pending messages causes slowdown and random crashes.
12462	Maximize button in Windows title bar doesn't use full screen under high-dpi Windows
12458	Crash when reading invalid image data
12451	Popup windows and combo-box menus don't disappear when they should.
12437	Cursor changes incorrectly for top pixel of borderless windows on Mac.
12436	import snapshot does not always display crosshair
12404	When using import/export snapshot from screen in non-interactive mode
12401	Browser: revBrowserSnapshot issues in 6.7.0 DP3
12388	Drag-drop does not work if 'private' data type is used on Mac.
12370	Key codes are mapped differently resulting keyboard shortcuts not acting correctly with non-English keyboard layouts.
12363	Cmd-Z shortcut missing from "Undo" menu item
12354	AcceleratedRendering causes double-sized stack controls on Retina displays
12351	Crash on write then read until EOF on driver
12339	mouseRelease message sent after selection from popup menu.
12321	On Windows 7 Fullscreen set to false does not return to previous size
12312	VideoGrabber doesn't work on Mac.
12303	Setting the text of a field chunk should not clear the paragraph styles of an empty line.
12297	Windows opened in popup mode have decorations.
12225	Menubar in application makse revBrowser misaligned
12156	Single line message box field is too large
12055	mobileVibrate not vibrating when passed a variable
12014	Stack with iOS deployment checked(mac) when opened on windows system does not allow you to disable this deployment option. This in turn breaks Android APK generation
11928	Inconsistencies in behavior when doing 'delete the selectedChunk'.
11878	Pasting with the middle mouse button on linux doesn't work correctly.
11839	Menus in Script Editor have testSize too small
11809	Ensure that spaceAbove area is redrawn when hilite changes.
11503	Dictation is unstable on Mac.
11493	Buttons in Ask
11383	Help menu: duplicate name
11346	Command click in project browser

11240	Duplicate Menu and Replicate dialog lock messages
10942	Project Browser disappears in dual screen
10593	When setting the styledText of a range
4001	ask dialog icon is a button with its autohilite set to true

Specific bug fixes (6.7.0-dp-4)

12384	revBrowserOpenCEF doesn't send browserDocumentComplete when setting htmltext
12367	Standard menu items with tags (Cut/Copy/Paste etc.) do not enabled/disable correctly.
12364	Crash when calling revBrowserClose on revCEFBrowser

Specific bug fixes (6.7.0-dp-3)

12348	Scrollbars don't get keyboard input when focused if no field is on the card on Mac.
12329	Menu items are all disabled in modal dialogs on Mac.
12328	Make sure invisible stacks still have a valid windowId on Mac.
12324	New CEF-based revBrowser has no authentication dialog
12304	Pasting LiveCode controls from other copies of LiveCode doesn't work
12299	formattedWidth/Height of player objects is incorrect
12284	Command key shortcuts don't work if revBrowser has focus on Mac.
12270	Stack location is always constrained to screen on Mac.
12267	Resize stack cursor not set sometimes.
12264	resizable and maximize decorations not honoured correctly on Mac.
12259	Answer dialogs do not respond to mouse input if opened from a menu on Mac.
12255	dragDrop sometimes doesn't work when dropping from other applications (on Mac).
12254	dragImageOffset is incorrect horizontally on Mac.
12253	Crash after releasing mouse on stack after drag-drop operation on Mac.
12252	File > Exit menu item not hidden on Mac.
12249	Player object does not respond properly to edit tool.
12230	Accelerated rendering mode doesn't work correctly on Mac if using coregraphics mode.
12226	revBrowser doesn't grab mouse focus on Mac.
12216	the qtVersion returns empty
12163	Beach-ball shows in tight loops on Mac.
12130	Resize cursor does not always display in window edges on Mac.
12116	NullPointerException on device logcat when app was launched and closed if in-app purchasing box is not ticked
12115	Import / export snapshot from rect of window ignores rect on OSX
12109	"Export snapshot ... from window ..." wrongly scales snapshot rect
12089	Setting a player filename to a url causes hang.
12087	moveStack messages are not sent continously on Mac.
12086	rawKeyDown sends incorrect code for shifted keys on Mac.
12085	"export snapshot of window ..." locks up on OSX
12082	Changing an applied pattern causes the IDE to crash
12080	Setting hidePalettes to false does not work on Mac.
12073	Make sure 'unlock cursor' takes immediate effect
12068	No default menubar in standalones on Mac.
12056	Clicking on a cell of a table field causes the IDE to crash

12047	Arrow keys don't work in CEF browser on Mac.
11922	Players do not render at correct size on Retina Macs.
10767	Non-CEF browser not working in modal dialogs on Mac.
5545	The mouse cursor does not hide when typing on Mac.

Specific bug fixes (6.7.0-dp-1)

11917	Setting the label of an option or combo-box does not update the menuHistory.
11808	pixelScaling not enabled on Windows Commercial edition

Dictionary changes

- The entry for **mobileLocationAuthorizationStatus** (*function*) has been updated.

Previous Release Notes

6.6.2 Release Notes	http://downloads.livecode.com/livecode/6_6_2/LiveCodeNotes-6_6_2.pdf
6.6.1 Release Notes	http://downloads.livecode.com/livecode/6_6_1/LiveCodeNotes-6_6_1.pdf
6.6.0 Release Notes	http://downloads.livecode.com/livecode/6_6_0/LiveCodeNotes-6_6_0.pdf
6.5.2 Release Notes	http://downloads.livecode.com/livecode/6_5_2/LiveCodeNotes-6_5_2.pdf
6.5.1 Release Notes	http://downloads.livecode.com/livecode/6_5_1/LiveCodeNotes-6_5_1.pdf
6.5.0 Release Notes	http://downloads.livecode.com/livecode/6_5_0/LiveCodeNotes-6_5_0.pdf
6.1.3 Release Notes	http://downloads.livecode.com/livecode/6_1_3/LiveCodeNotes-6_1_3.pdf
6.1.2 Release Notes	http://downloads.livecode.com/livecode/6_1_2/LiveCodeNotes-6_1_2.pdf
6.1.1 Release Notes	http://downloads.livecode.com/livecode/6_1_1/LiveCodeNotes-6_1_1.pdf
6.1.0 Release Notes	http://downloads.livecode.com/livecode/6_1_0/LiveCodeNotes-6_1_0.pdf
6.0.2 Release Notes	http://downloads.livecode.com/livecode/6_0_2/LiveCodeNotes-6_0_2.pdf
6.0.1 Release Notes	http://downloads.livecode.com/livecode/6_0_1/LiveCodeNotes-6_0_1.pdf
6.0.0 Release Notes	http://downloads.livecode.com/livecode/6_0_0/LiveCodeNotes-6_0_0.pdf