LiveCode 8.1.0 Release Notes

- Overview
- Known issues
- Platform support

    - Windows
    - Linux
    - Mac
    - iOS
    - Android
    - HTML5

- Setup

    - Installation
    - Uninstallation
    - Reporting installer issues
    - Activating LiveCode Indy or Business edition
    - Command-line installation
    - Command-line uninstallation
    - Command-line activation for LiveCode Indy or Business edition

- Engine changes

    - Change in empty is among the items of `1,2,3,` behavior (8.1.0-rc-2)
    - Filtering array keys and elements has been added to the filter command (8.1.0-rc-1)
    - minStackFileVersion property (8.1.0-rc-1)
    - A new scriptStatus of object property has been implemented (8.1.0-rc-1)
    - Ensure non-BMP characters roundtrip through htmlText (8.1.0-rc-1)
    - Group command does not return new group id in it (8.1.0-rc-1)
    - List folders other than the default folder (8.1.0-dp-3)
    - Fix truncation when saving field paragraphs where length exceeds 32767 characters. (8.1.0-dp-3)
    - Changes to the dontUseQT property of a player object (Windows and OSX) (8.1.0-dp-2)
    - Add several IDE messages (8.1.0-dp-2)
    - Improved return command (8.1.0-dp-2)
    - Revert a target stack (8.1.0-dp-2)
    - New vectorDotProduct function (8.1.0-dp-2)
    - Add functions for getting synchronous modifier key state (8.1.0-dp-2)
    - Curly brace subscripts are now a syntax error (8.1.0-dp-2)
    - Improve Android timestamp accuracy for GPS and sensors (8.1.0-dp-2)
    - Improved GPS support on Android and iOS (8.1.0-dp-2)
    - Automatic LCB extension inclusion in standalones (8.1.0-dp-1)
    - Standalone 'Search for inclusions' for mobile deployment (8.1.0-dp-1)
    - Standalone script library inclusions for mobile deployment (8.1.0-dp-1)
    - Windows DirectShow Player Control (8.1.0-dp-1)
    - Add support for custom entitlements for iOS (8.1.0-dp-1)
    - Specific engine bug fixes (8.1.0)
    - Specific engine bug fixes (8.1.0-rc-2)

- Specific engine bug fixes (8.1.0-rc-1)
- Specific engine bug fixes (8.1.0-dp-3)
- Specific engine bug fixes (8.1.0-dp-2)
- Specific engine bug fixes (8.1.0-dp-1)

- IDE changes

  - Script compilation errors are now indicated on the project browser (8.1.0-rc-1)
  - Tutorial stack browser display (8.1.0-rc-1)
  - Search for user tutorials (8.1.0-rc-1)
  - Keyboard navigation has been added to the project browser (8.1.0-rc-1)
  - ideControlMoved message (8.1.0-dp-3)
  - Ignore moveStack and resizeStack execution errors in script debug mode (8.1.0-dp-3)
  - Deprecated syntax in the dictionary (8.1.0-dp-2)
  - Dictionary UI Improvements (8.1.0-dp-2)
  - Autofocus on message box when typing (8.1.0-dp-2)
  - Standalone Inclusions Interface (8.1.0-dp-1)
  - Specific IDE bug fixes (8.1.0-rc-2)
  - Specific IDE bug fixes (8.1.0-rc-1)
  - Specific IDE bug fixes (8.1.0-dp-3)
  - Specific IDE bug fixes (8.1.0-dp-2)
  - Specific IDE bug fixes (8.1.0-dp-1)

- LiveCode Builder changes

  - LiveCode Builder Standard Library
  - LiveCode Builder Language
  - Specific LCB bug fixes (8.1.0-rc-1)
  - Specific LCB bug fixes (8.1.0-dp-3)
  - Specific LCB bug fixes (8.1.0-dp-2)
  - Specific LCB bug fixes (8.1.0-dp-1)

- LiveCode extension changes

  - Line Graph widget
  - SVG Icon widget
  - Specific extension bug fixes (8.1.0-dp-3)

- Dictionary additions
- Previous release notes

Overview

LiveCode 8.1 provides important improvements for delivering high-quality cross-platform applications!

- LiveCode Indy and Business editions now come with the tsNet external, which supercharges LiveCode's Internet features and performance. LiveCode 8.1 also introduces mergHealthKit, for accessing activity, sport and health data on iOS devices.

- The standalone builder now has a greatly-improved user experience for including externals,

script libraries and LiveCode Builder extensions in your cross-platform application. Usually, it'll now do the right thing automatically, but you can still select the specific inclusions you need.

- The IDE has lots of other upgrades, too: a keyboard-navigable Project Browser that highlights any scripts that failed to compile, an improved dictionary user interface, and access to the message box just by starting to type.

- The player control can be used in Windows application without any need for users to install any additional libraries or dependencies, thanks to a brand new player implementation based on DirectShow. For most apps, it should now be unnecessary to install or use QuickTime at all.

- The LiveCode Builder programming language has had some enhancements as part of the Infinite LiveCode project. Variables now get initialised by default, `unsafe` blocks and handlers can be used to flag sections of code that do dangerous things, and you can even include raw `bytecode` if necessary.

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.

- The browser widget does not work on 32-bit Linux.

- 64-bit standalones for Mac OS X do not have support for audio recording or the revVideoGrabber external.

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

LiveCode supports the following versions of Windows:

- 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)
- Windows 10

Note: On 64-bit Windows installations, LiveCode runs as a 32-bit application through the WoW layer.

Linux

LiveCode supports the following Linux distributions, on 32-bit or 64-bit Intel/AMD or compatible processors:

- Ubuntu 14.04 and 16.04
- Fedora 23 & 24
- Debian 7 (Wheezy) and 8 (Jessie) [server]
- CentOS 7 [server]

LiveCode may also run on Linux installations which meet the following requirements:

- Required dependencies for core functionality:

    - glibc 2.13 or later
    - GLib 2.0 or later

- Optional requirements for GUI functionality:

    - GTK/GDK 2.24 or later
    - Pango with Xft support
    - esd (optional, needed for audio output)
    - mplayer (optional, needed for media player functionality)
    - lcms (optional, required for color profile support in images)
    - gksu (optional, required for privilege elevation support)

Note: If the optional requirements are not present then LiveCode will still run but the specified features will be disabled.

Note: The requirements for GUI functionality are also required by Firefox and Chrome, so if your Linux distribution runs one of those, it will run LiveCode.

Note: It may be possible to compile and run LiveCode Community for Linux on other architectures but this is not officially 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
- 10.10.x (Yosemite) on Intel
- 10.11.x (El Capitan) on Intel

iOS

iOS deployment is possible when running LiveCode IDE on a Mac, and provided Xcode is installed and has been set in LiveCode *Preferences* (in the *Mobile Support* pane).

Currently, the supported versions of Xcode are:

- Xcode 4.6 on MacOS X 10.7
- Xcode 5.1 on MacOS X 10.8
- Xcode 6.2 on MacOS X 10.9
- Xcode 6.2 and 7.2 on Mac OS X 10.10
- Xcode 7.3 on MacOS X 10.11

It is also possible to set other versions of Xcode, to allow testing on a wider range of iOS simulators. For instance, on Yosemite, you can add *Xcode 5.1* in the *Mobile Support* preferences, to let you test your stack on the *iOS Simulator 7.1*.

We currently support the following iOS Simulators:

- 6.1
- 7.1
- 8.2
- 9.2
- 9.3

Android

LiveCode allows you to save your stack as an Android application, and also to deploy it on an Android device or simulator from the IDE.

Android deployment is possible from Windows, Linux and Mac OSX.

The Android engine supports devices using ARMv6, ARMv7 or ARMv8 processors. It will run on the following versions of Android:

- 2.3.3-2.3.7 (Gingerbread)
- 4.0 (Ice Cream Sandwich)
- 4.1-4.3 (Jelly Bean)
- 4.4 (KitKat)
- 5.0-5.1 (Lollipop)
- 6.0 (Marshmallow)

To enable deployment to Android devices, you need to download the Android SDK, and then use the 'Android SDK Manager' to install:

- the latest "Android SDK Tools"
- the latest "Android SDK Platform Tools"

You also need to install the Java Development Kit (JDK). On Linux, this usually packaged as "openjdk". LiveCode requires JDK version 1.6 or later.

Once you have set the path of your Android SDK in the "Mobile Support" section of the LiveCode IDE's preferences, you can deploy your stack to Android devices.

Some users have reported successful Android Watch deployment, but it is not officially supported.

HTML5

LiveCode applications can be deployed to run in a web browser, by running the LiveCode engine in

JavaScript and using modern HTML5 JavaScript APIs.

HTML5 deployment does not require any additional development tools to be installed.

LiveCode HTML5 standalone applications are currently supported for running in recent versions of Mozilla Firefox, Google Chrome or Safari. For more information, please see the "HTML5 Deployment" guide in the LiveCode IDE.

Setup

Installation

Each version of LiveCode installs can be installed to its own, separate folder. This allow multiple versions of LiveCode to be installed side-by-side. On Windows (and Linux), each version of LiveCode has its own Start Menu (or application menu) entry. On Mac OS X, each version has its own app bundle.

On Mac OS X, install LiveCode by mounting the `.dmg` file and dragging the app bundle to the `Applications` folder (or any other suitable location).

For Windows and Linux, the default installation locations when installing for "All Users" are:

Platform	Path
Windows	`<x86 program files folder>/RunRev/LiveCode <version>`
Linux	`/opt/livecode/livecode-<version>`

The installations when installing for "This User" are:

Platform	Path
Windows	`<user roaming app data folder>/RunRev/Components/LiveCode <version>`
Linux	`~/.runrev/components/livecode-<version>`

Note: If installing for "All Users" on Linux, either the **gksu** tool must be available, or you must manually run the LiveCode installer executable as root (e.g. using **sudo** or **su**).

Uninstallation

On Windows, the installer hooks into the standard Windows uninstall mechanism. This is accessible from the "Add or Remove Programs" applet in the windows Control Panel.

On Mac OS X, drag the app bundle to the Trash.

On Linux, LiveCode can be removed using the `setup.x86` or `setup.x86_64` program located in LiveCode's installation directory.

Reporting installer issues

If you find that the installer fails to work for you then please report it using the LiveCode Quality Control Centre or by emailing support@livecode.com.

Please include the following information in your report:

- Your platform and operating system version
- The location of your home or user folder
- The type of user account you are using (guest, restricted, admin etc.)
- The installer log file.

The installer log file can be located as follows:

Platform	Path
Windows 2000/XP	`<documents and settings folder>/<user>/Local Settings/`
Windows Vista/7	`<users folder>/<user>/AppData/Local/RunRev/Logs`
Linux	`<home>/.runrev/logs`

Activating LiveCode Indy or Business edition

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.

Command-line installation

It is possible to invoke the installer from the command-line on Linux and Windows. When doing command-line installation, no GUI will be displayed. The installation process is controlled by arguments passed to the installer.

Run the installer using a command in the form:

```
<installer> install noui [OPTION ...]
```

where `<installer>` should be replaced with the path of the installer executable or app (inside the DMG) that has been downloaded. The result of the installation operation will be written to the console.

The installer understands any of the following `OPTION`s:

Option	Description
`-allusers`	Install the IDE for "All Users". If not specified, LiveCode will be installed for the current user only.

- Option	Description
`desktopshortcut`	Place a shortcut on the Desktop (Windows-only)
`-startmenu`	Place shortcuts in the Start Menu (Windows-only)
`-location LOCATION`	The folder to install into. If not specified, the `LOCATION` defaults to those described in the "Installation" section above.
`-log LOGFILE`	The file to which to log installation actions. If not specified, no log is generated.

Note: the command-line installer does not do any authentication. When installing for "All Users", you will need to run the installer command as an administrator.

As the installer is actually a GUI application, it needs to be run slightly differently from other command-line programs.

On Windows, the command is:

```
start /wait <installer> install noui [OPTION ...]
```

Command-line uninstallation

It is possible to uninstall LiveCode from the command-line on Windows and Linux. When doing command-line uninstallation, no GUI will be displayed.

Run the uninstaller using a command of the form:

```
<uninstaller> uninstall noui
```

Where is *.setup.exe* on Windows, and *.setup.x86* on Linux. This executable, for both of the platforms, is located in the folder where LiveCode is installed.

The result of the uninstallation operation will be written to the console.

Note: the command-line uninstaller does not do any authentication. When removing a version of LiveCode installed for "All Users", you will need to run the uninstaller command as an administrator.

Command-line activation for LiveCode Indy or Business edition

It is possible to activate an installation of LiveCode for all users by using the command-line. When performing command-line activation, no GUI is displayed. Activation is controlled by passing command-line arguments to LiveCode.

Activate LiveCode using a command of the form:

```
<livecode> activate -file LICENSEFILE -passphrase SECRET
```

where `<livecode>` should be replaced with the path to the LiveCode executable or app that has been previously installed.

This loads license information from the manual activation file `LICENSEFILE`, decrypts it using the given `SECRET` passphrase, and installs a license file for all users of the computer. Manual activation files can be downloaded from the My Products page in the LiveCode account management site.

It is also possible to deactivate LiveCode with:

```
<livecode> deactivate
```

Since LiveCode is actually a GUI application, it needs to be run slightly differently from other command-line programs.

On Windows, the command is:

```
start /wait <livecode> activate -file LICENSE -passphrase SECRET
start /wait <livecode> deactivate
```

On Mac OS X, you need to do:

```
<livecode>/Contents/MacOS/LiveCode activate -file LICENSE -passphrase SECRET
<livecode>/Contents/MacOS/LiveCode deactivate
```

Engine changes

Change in empty is among the items of `1,2,3,` behavior (8.1.0-rc-2)

In previous versions of LiveCode `is among the items of` behaved incorrectly when a trailing delimiter was present. Trailing delimiters are generally ignored in the LiveCode engine and legacy behaviors where they are not are documented anomalies that are maintained for backwards compatibility reasons. This change resolves the anomaly detailed in bug report 16297

For example:

```
put the number of items of "1,2,3," is 3 --> puts true in all engines
put empty is among the items of "1,2,3," --> puts true in LiveCode 6
put empty is among the items of "1,2,3," --> puts false in LiveCode 8
```

> **Warning** If your stacks depend on the legacy anomalous behavior they should be checked to ensure all trailing empty items are wrapped by an item delimiter

Filtering array keys and elements has been added to the filter command (8.1.0-rc-1)

The filter command now supports the filtering of arrays by matching keys or elements.

Example:

```
local tArray
put true into tArray["foo"]
put false into tArray["bar"]
filter keys of tArray with "f*"
put the keys of tArray is "foo"
```

minStackFileVersion property (8.1.0-rc-1)

A new stack property has been added to determine the minimum stack file version that can be safely used to save a stack without data loss. For example, if a widget is on the stack the `minStackFileVersion` will be 8.0 and saving with a lower stack file version will result in the loss of the widget from the stack. The minimum `minStackFileVersion` reported by the property is 7.0.

A new scriptStatus of object property has been implemented (8.1.0-rc-1)

Use `the scriptStatus of <object>` to determine the status of the last time the script property was set or the script was compiled when the stack was opened.

Ensure non-BMP characters roundtrip through htmlText (8.1.0-rc-1)

Previously, unicode characters outside of the basic multilingual plane (i.e. those with codepoint < 65536) would fail to roundtrip through the htmlText property of fields. This has now been fixed.

In addition, fixing this issue also means that unicode characters (of any codepoint) can now appear in the metadata attribute of 'p' and 'span' tags.

Finally, the imageSource property can now span multiple characters. This is required to allow it to apply to surrogate pairs (i.e. characters with codepoint

65535) and unicode character sequences which are considered a single 'char' (i.e. human readable character / grapheme).

Group command does not return new group id in it (8.1.0-rc-1)

The group command has been changed so that it returns the long id of the newly created group in the 'it' variable.

If no group is created (as a result of using 'group' with an empty selection) the 'it' variable will be set to empty.

List folders other than the default folder (8.1.0-dp-3)

When called as a function, the **files** and **folders** functions now take an optional argument specifying which directory to list. This makes writing filesystem code a lot easier, since code that looked like:

```
local tOldFolder, tFilesList
put the defaultFolder into tOldFolder
set the defaultFolder to "/path/to/target/directory"
put the files into tFilesList
set the defaultFolder to tOldFolder
return tFilesList
```

can be replaced with:

```
return files("/path/to/target/directory")
```

Fix truncation when saving field paragraphs where length exceeds 32767 characters. (8.1.0-dp-3)

Note: This fix introduces a new stack file format version (8.1) which is required to preserve the paragraph text. Saving with a legacy stack file version will result in loss of data for field text affected by this bug.

Changes to the dontUseQT property of a player object (Windows and OSX) (8.1.0-dp-2)

It is now possible to set the **dontUseQT** property for a player object.

On Windows, the default value of the global **dontUseQt** and **dontUseQtEffects** properties has changed from true to false. This means that by default players created on Windows will use the DirectShow API for multimedia playback.

On OSX, QuickTime is unable to be supported in 64 bit builds the default value of the global **dontUseQT** and **dontUseQTEffects** properties changed in version 6.7; it is true on OS X version 10.8 and up, or on all versions of OS X if the engine is 64 bit. This means that any player object created will use the AVFoundation API for multimedia playback.

With this new feature, you can set the **dontUseQT** property of a player to false, without changing the value of the global **dontUseQt** property. If you do this, you can have both QuickTime and AVFoundation players playing at the same time, which can be particular useful for supporting some media formats or codecs that are not supported by the default AVFoundation or DirectShow player (for example .midi files, Sorenson Video 3, H.261 codecs etc)

Warning: QuickTime has not been maintained or supported by Apple for quite some time. You are encouraged to check your applications for any dependence on QuickTime, and remove it if found.

Add several IDE messages (8.1.0-dp-2)

Messages are now sent when audio clip and video clip controls are created or deleted.

- **newAudioclip**: sent when an audio clip is created
- **deleteAudioclip**: sent when an audio clip is deleted
- **newVideoclip**: sent when a video clip is created
- **deleteVideoclip**: sent when a video clip is deleted

Improved return command (8.1.0-dp-2)

The 'return' command has had two new forms added:

```
return <value> for value
return <value> for error
```

When running in a command handler, the 'return ... for value' form will cause execution of the handler to halt, and control to return to the calling handler. At this point the 'it' variable in the calling handler will be set to 'value' and 'the result' will be set to empty. In contrast, the 'return ... for error' form will cause the 'it' variable in the calling handler to be set to empty and 'the result' to be set to 'value'.

When running in a function handler, the 'return ... for value' form will cause execution of the handler to halt, and control to return to the calling handler. At this point the return value of the function call will be 'value', and 'the result' will be set to empty. In contrast, the 'return ... for error' form will cause the return value of the function call to be empty, and 'the result' will be set to 'value'.

These forms of return are designed to be used by script library functions to allow them to have the same ability as built-in engine functions and commands - namely the ability to return a value (in it for commands, or return value for functions) *or* return a status (in the result).

Revert a target stack (8.1.0-dp-2)

It is now possible to revert a stack that is not the topStack, using

```
revert <stack reference>
```

New vectorDotProduct function (8.1.0-dp-2)

A new **vectorDotProduct** function has been added. It computes the vector dot product of two single-dimensional arrays with identical keys.

More specifically:

```
vectorDotProduct(tArray1, tArray2)
```

Will compute:

```
put 0.0 into tSum
repeat for each key tKey in tArray1
    add tArray1[tKey] * tArray2[tKey] to tSum
end repeat
return tSum
```

If the two arrays do not have the same set of keys, then an error is thrown.

Add functions for getting synchronous modifier key state (8.1.0-dp-2)

LiveCode currently provides functions for checking the state of so-called "modifier" keys: Caps Lock, Control, Command, Shift, Alt/Option. These functions return either "up" or "down", reflecting the state of the key at the time the function was called. However, it is often desireable to check the state of the key at the time the event was generated and this is not possible using these functions.

New functions called "eventAltKey", "eventShiftKey", etc have been added; these return the state of the key at the time the event began processing. This is useful in keyDown and rawKeyDown handlers to check whether a modifier was pressed at the time the key the event relates to was pressed (if the non-event forms are used instead, there is a chance the modifier key has been released and the wrong result will be generated).

Note that the "eventXXXKey" functions should *not* be called after a wait; their value is undefined after any form of wait has occurred.

Curly brace subscripts are now a syntax error (8.1.0-dp-2)

Previously, it was possible to use curly brackets or braces `{}` instead of square brackets `[]` in array and custom property syntax. This was an undocumented and unknown feature to most users.

Using `{}` to subscript arrays is now a script syntax error. Curly brackets have been reserved for future use.

Improve Android timestamp accuracy for GPS and sensors (8.1.0-dp-2)

Timestamps for sensors on Android were previously passed in a low-precision format, resulting in "sticky" timestamps that did not change more than a few times a minute. This has now been resolved and timestamps are now reported to microsecond resolution (though the accuracy is

unlikely to be at the microsecond level).

In addition to this change, the timestamps are now reported in "monotonic" time rather than "wall-clock" time ("wall-clock" time is the time you see reported as the current time). This means that the timestamps are now independent of changes to the device clock as a result of adjustments or daylight savings changes. If you want to match the readings to the device time instead, get the current time when receiving the location update rather than using the timestamp in the update.

Improved GPS support on Android and iOS (8.1.0-dp-2)

GPS behavior is now identical on Android and iOS. On both platforms, the location reading returned by the `mobileSensorReading` function is that which was sent with the last system `locationChanged` event. (This brings iOS behavior inline with that of Android).

Additionally three new handlers have been implemented:

```
mobileGetLocationHistory
mobileSetLocationHistoryLimit
mobileGetLocationHistoryLimit
```

Whenever a system `locationChanged` event occurs, the location reading is pushed onto the front of a list. The list is capped at the length set by the location history limit, dropping any old samples over this length.

The `mobileGetLocationHistory` function returns a numerically keyed array of all accumulated samples since the last time it was called with lower indices being older samples. Calling the function clears the internal history.

Each element in the array is the same format as the detailed location array as returned from the `mobileSensorReading` function.

If an application wants historical access to all samples, then it should set the location history limit to the maximum number of samples it ever wants to record, or 0 to record the entire history (between calls to `mobileGetLocationHistory`).

The best way to use the history is to fetch the list in `locationChanged` and process each sample in turn, rather than the sample provided with the `locationChanged` event (which will always be the last sample in the history). e.g.

```
on locationChanged
   local tHistory
   put mobileGetLocationHistory() into tHistory
   repeat for each element tSample in tHistory
      processLocationChanged tSample
   end repeat
end locationChanged
```

The default history limit is 1 meaning that only one sample is ever kept at a time.

Automatic LCB extension inclusion in standalones (8.1.0-dp-1)

When a standalone is built, the modules required for the widgets that are on the stack (or any of its substacks) are now included in the application automatically, regardless of whether the 'Search for required inclusions...' option is selected in the standalone settings.

If 'Search for required inclusions' is enabled, the scripts of the application will be searched for uses of the public handlers of any available LCB libraries, and any uses of the 'kind' of available widgets to determine whether the relevant modules are included. For example, if the script contains:

```
create widget as "com.livecode.widget.svgpath"
```

then the 'SVG Path' widget and all its dependencies will be included.

Standalone 'Search for inclusions' for mobile deployment (8.1.0-dp-1)

The standalone builder 'Search for required inclusions...' option now supports mobile deployment, both to device and simulator.

Standalone script library inclusions for mobile deployment (8.1.0-dp-1)

Script libraries can now be included in mobile applications in the same way as for desktop applications, via the 'Inclusions' pane of the standalone builder.

Windows DirectShow Player Control (8.1.0-dp-1)

Due to the recent decision by Apple to end support for QuickTime on Windows, the player implementation on that platform has been replaced with one based on DirectShow. This is a multimedia API that is available by default on all versions of Windows supported by LiveCode.

The new implementation should function as a drop-in replacement for the old one, though some properties are not yet implemented.

Property Changes

On Windows, the behaviour of some properties of the player control have changed.

- The **loadedTime** property previously did not work on Windows, but now does.
- The **alwaysBuffer**, **enabledTracks**, **mediaTypes**, **mirrored**, **trackCount** and **tracks** properties do not currently work, but will be re-enabled in a subsequent release.

On all platforms, the following player control properties, which are specific to QuickTime and QTVR, have been deprecated: **constraints**, **currentNode**, **movieControllerId**, **nodes**, **pan**, **tilt**, and **zoom**.

Supported File Formats

Media format support in the new Windows player control depends on which codecs are installed.

A list of the file formats and compression types available as standard.aspx) on Windows is available in the MSDN documentation

Add support for custom entitlements for iOS (8.1.0-dp-1)

Custom entitlements can now be added to an iOS app by including one or more `.xcent` files in the copy files section of the standalone builder containing an XML snippet of key/value pairs. For example, if you wanted to add the entitlement for HomeKit to your app you might create a file named `HomeKit.xcent` with the following content:

```
<key>com.apple.developer.homekit</key>
<true/>
```

Specific engine bug fixes (8.1.0)

18327	**Calculate global MCmajorosversion in a more robust way**
18364	**savingStandalone and standaloneSaved should only be sent once when building for multiple platforms**

Specific engine bug fixes (8.1.0-rc-2)

16397	Change in empty is among the items of `1,2,3,` behavior
18205	Fixed tsNet.dll "fails to initialise" error on Windows Server 2008/2012 Standard
18247	For external libUrl drivers, call the external function lvExtIsBlocked() to check for existing blocking connections
18261	Fixed crash on 64-bit iOS simulator when tsNet/Internet lib is included in the standalone
18282	Fix a crash after deleting a shared background group
9571	Document menu item tags

Specific engine bug fixes (8.1.0-rc-1)

17646	Ignore trailing whitespace when parsing SVG paths
17764	Fix widget drawing when attempting to draw empty strings
17963	Allow building multiple Linux architectures at the same time
17972	Fix missing fullscreenMode enum documentation
18031	Clean up empty folders after moving `Copy files` to resources folder in Mac standalones
18042	Ensure guide PDF escapes special characters correctly.

18063   Fix color-swap when exporting image with more than 256 colors to GIF on Linux

18068   Ensure non-BMP characters roundtrip through htmlText

18100   Fix browser widget remaining visible after going to another stack on mobile

18103   Pass settings parameter to HTML5 builder correctly

18109   [Dict] mobileControlGet(myPlayer, "duration") is supported on Android

18123   Windows: fix crash when opening player from hidden stack

18124   Allow libUrlFtpUpload and libUrlFtpUploadFile to support SFTP URLs

18129   Ensure matchChunk returns the correct char positions

18136   Make sure arithmetic commands throw errors for bad inputs

18138   Silence documentation parser warnings on non-doc block comments

18148   Fix inline example of using imageData to get color channels

18152   Fix browser position on Mac when shown in resized palette stack

18153   Fix browser widget disappearing when stack decorations changed

18162   Ensure Mac universal externals are found correctly

18179   Fix malformed dictionary entry for the "go" command.

18196   Add 8.1 file format to stack version glossary entry

18207   Fix Windows player playback of url streams

18210   Fix error message when copying uninstaller fails

18220   Ensure all relevant LCB interface files are present in IDE

18222   Fix widgets not being saved when on substack but not on main stack

18226   Make sure database drivers are included in the standalone

18248   Display appropriate error when trying to build script only standalones

18251   Fix video of player not automatically resized when setting filename

6530   Group command does not return new group id in it

7214   Examples in dictionary entries referring to file paths should be updated

7344   Correct named shades of gray (Gray1-100) to match the X11 color names

Specific engine bug fixes (8.1.0-dp-3)

17149   Ensure iconGravity property is preserved when copying/cloning buttons.

17392   Ensure line endings of utf-8 docs are converted

17633   Fix long delay when launching standalone displaying a browser widget

17690   Fix truncation when saving field paragraphs where length exceeds 32767 characters.

17782   Fix incorrect measurement and placement of rotated text on Windows

17868   Only save standalone stackfile for current standalone target

17890   Fix player callbacks still calling after being cleared

17922   Check standalone stack for widgets before resolving dependencies

17937   Make sure the action of 'return for' is local to the caller.

17957   Ensure V1 externals can set it when handlers called from top-level in server.

17959   Fix a crash on iOS 9 when rendering subviews

17961   Fix error in layerMode dictionary entry

17965   Do something sensible when GDK reports a maximum cursor size of 0

17971   Fix formatting of resizeStack message docs entry

18019   Ensure cards with objects on can be deleted

18041   Ensure codeunitOffset and codepointOffset return 1 indexed offset

18045   Make sure cookie filePath is set correctly

18065   Allow check for private among the keys of the fullDragData

Specific engine bug fixes (8.1.0-dp-2)

15183   Ensure dependencies of built-in inclusions are included

17180   Ensure deleted objects executing scripts can not be deleted

17275   Add functions for getting synchronous modifier key state

17317   Deprecate liveResizing and metal stack properties

17469   Curly brace subscripts are now a syntax error

17523   Fix LCB docs builder handling of string-like property names

17553   Paint Tools Not Working in IDE

17609   Return an empty item instead of a random value if altitude reading is not available on iOS

17637   Update docs to reflect changes to standalone builder inclusions

17661   Improve Android timestamp accuracy for GPS and sensors

17662   Improved GPS support on Android and iOS

17697   Fix player view occasionally not showing on Mac

17698   Fix Windows player frame seeking

17700   Fix Windows player not pausing when in edit mode

17701   Fix incorrect Windows player rect on opening

17708   Fix incorrect player currenttime value for videos longer than 7m 15s

17747   Make sure widgets get mouseUp in popup stacks

17754   Ensure external code blobs are included in standalones

17776   Parse deprecated LCB syntax properly

17781   Fix OSX mouse event errors when using QTKit player

17797   Enable playback of MP3 and other audio files in Windows player

17815   Fix native layer of player not showing when stack opened

17828   Fix player slider moving outside its track boundary

17834   Prevent possible assertion failure related to revMessageBoxRedirect

17842   Ensure read from socket documentation matches engine behavior

17844   Allow the dropChunk function to parse as a chunk reference

17856   Deal with pdfPrinter inclusion setting correctly

17891   Reset the m_was_licensed instance variable to true before calls to an external's handler

Specific engine bug fixes (8.1.0-dp-1)

17515   Add support for custom entitlements for iOS

17573   Don't retain other platforms' temporary standalone settings data

IDE changes

Script compilation errors are now indicated on the project browser (8.1.0-rc-1)

The Project Browser will now indicate objects that have scripts that were not able to be compiled. The script line number indicator will appear red for scripts with errors.

Tutorial stack browser display (8.1.0-rc-1)

The tutorial system can now display a browser widget set to a url given in its specification. In order to do so, simply include the url as a section of the step in question. For example:

```
step "Instructions With Browser"
    Please watch the following video with detailed instructions on how
    to proceed.
url
    <url of video>
action
    interlude
    go to step "Create New Mainstack"
end step
```

Search for user tutorials (8.1.0-rc-1)

The `revIDEStartTutorial` command will now search both in the IDE and in the user's `My LiveCode/Interactive Tutorials/` folder for tutorials to start.

So for example, an interactive tutorial placed at

```
My LiveCode/Interactive Tutorials/courses/Database
Course/tutorials/SQLite/lessons/1. Introduction.txt
```

can be run using

```
revIDEStartTutorial "Database Course", "SQLite", "1. Introduction"
```

Keyboard navigation has been added to the project browser (8.1.0-rc-1)

The project browser now supports the following keyboard actions:

- up and down arrowKey - move up and down the object list
- left and right arrowKey - contract and expand child objects
- returnKey - open object
- shift + returnKey - open script

- shift + option + returnKey - open behavior script
- command + returnKey - open inspector

ideControlMoved message (8.1.0-dp-3)

`ideControlMoved` has been added to the set of IDE messages to which other objects can subscribe. `ideControlMoved pTarget` is dispatched to all subscribers after the control `pTarget` is moved with the pointer tool (i.e. in edit mode) in the IDE.

Ignore moveStack and resizeStack execution errors in script debug mode (8.1.0-dp-3)

Previously when in script debug mode an execution error in the context of a moveStack or resizeStack handler would cause the IDE hang and occasionally crash.

Deprecated syntax in the dictionary (8.1.0-dp-2)

Deprecated syntax is now be prefixed in the dictionary entry list by a warning icon.

Dictionary UI Improvements (8.1.0-dp-2)

The dictionary stack UI has been updated to incorporate a few user interface improvements:

- The panel containing list of entries that match the current filter and search terms is now resizable, and contains three columns - the name of the entry, the type, and the syntax. The list can be sorted by any of these three.
- The history breadcrumb has been condensed into back and forward buttons, plus a history dropdown menu button.
- The filters pane now has associations and platforms filter categories, and is now scrollable if the content is too large for the pane. The numbers associated with each filter have been removed. The list of filters is laid out in two columns, sorted in alphabetical order.

Autofocus on message box when typing (8.1.0-dp-2)

If the message box is open, it will now automatically gain focus and receive keystrokes when you start typing with no field focused. This makes it much faster and more convenient to quickly run message box commands without hunting for the message box.

Standalone Inclusions Interface (8.1.0-dp-1)

The standalone settings user interface has been reworked to unify the notion of app inclusion. There is now an 'Inclusions' pane which allows the user to select from a complete list of available inclusions. The list contains information about which platforms are supported.

The 'Inclusions' pane significantly improves the cross-platform development experience provided by LiveCode (since the iOS and Android panes no longer have separate check boxes for the

various built-in externals such as revxml), as well as paving the way for much better extensibility in the future.

Specific IDE bug fixes (8.1.0-rc-2)

18265	Ensure message box is colorized correctly when the script editor hasn't been opened
18270	Ensure preferences are changed when LiveCode quits

Specific IDE bug fixes (8.1.0-rc-1)

14217	Keyboard navigation has been added to the project browser
15708	Don't offer to make substacks of script-only stacks
17520	Fix IDE error on View > Palettes menu
17660	Ensure IDE sends ideSelectedObjectChanged message when selecting objects
17663	Fixed typo in file association message displayed at startup when launching new version of LC
17693	Correctly identify the stack when building a standalone from the project browser
17696	Player control's default video doesn't work in Windows player
17863	Improve message box autocompletion and error handling
17870	Improved speed of formatting scripts
17902	Ensure apply button on variable visualiser enables correctly
17925	Ensure LCB modules are recompiled when lc-compile is updated
17991	Make Windows systemVersion handling in revDeleteFolder more robust
18026	Maintain scroll when focussing in on inclusions card in standalone build
18047	Fix object grouping menu items from contextual menus
18053	Use appropriate cursors for selection handles
18134	Colorize the message box with script colorization settings
18140	Move text properties to text pane of PI
18158	[Project Browser] Fix distorted view with nested groups and object_type_indicator == name
18166	Windows menu now shows stack names while the LC script window is on top
18169	[Extension Builder] Make sure scrollbar is visible when console_content > field_height
18170	[Extension Builder] Make console content selectable and thus enable copying
18174	Sort Script Editor handler list alphabetically per each different type
18175	Present a warning dialog if preserving stack file version will result in data loss
18183	Give the focus to the search box in the API Reference when it opens
18208	Make sure that file selector for field contents allows "All Files"
18214	Typo in LiveCode Builder doc
18249	Removed double extension of script only stacks on Linux and Windows
4695	Fixed runtime error when saving as legacy

Specific IDE bug fixes (8.1.0-dp-3)

11834	Don't encode/decode iOS app name in standalone settings

17409	List all loaded stacks in the behavior picker
17461	Ensure toolbar can not cover fullscreen window widget on Mac
17521	Output array-style parameter descriptions correctly
17586	Make "value" field of Custom Properties Inspector scrollable
17594	Fix backspace handling in project browser field editor
17600	Fix Refresh Data Grid button in PI
17690	Add 8.0 file format as an option in "Save as.." dialog
17718	Ensure property inspector is visible
17740	Improved formatting of line continuations in the script editor
17755	Ensure modal dialogs aren't moved to the main screen by script
17765	Fix typo in BMI Interactive Tutorial text
17768	Fix swapped labels in Project Browser footer
17852	Make sure multiple object text style/align props don't change on their own
17895	Documentation for setting dgText [TRUE] is ambiguous
17929	Prevent names in Dictionary overlap other columns
17932	Don't resolve angle brackets as links in code in descriptions
17948	Ensure the showInvisibles of script editors is false
18081	Ensure widget docs are generated on Windows
2867	Ignore moveStack and resizeStack execution errors in script debug mode
7157	Ensure IDE stacks are not listed in recentCards

Specific IDE bug fixes (8.1.0-dp-2)

16167	Remove reference to Windows XP from preferences stack
16325	Ensure preferences reset correctly
17146	Ensure that the Script Editor "Find" UI resizes correctly
17405	Display virtual font names correctly in the script editor font preference menu
17467	Fix recent files list update issue
17739	Make object list in project browser flat when sorting by name
17759	show alpha values of gradients dynamically
17826	Make sure "move" command results in smooth movement when executed from the msg box
296	Autofocus on message box when typing
8985	Clicking type disambiguation in Docs pane of Script Editor should work

Specific IDE bug fixes (8.1.0-dp-1)

17189	Replace Revolution text with LiveCode in Dialogs

LiveCode Builder changes

LiveCode Builder Standard Library

List operations

- The `&` operator can now be used to concatenate two `List`s. For example,
  `[1, 2] & [3, 4]` evaluates to `[1, 2, 3, 4]`.

LiveCode Builder Language

Core types

- The use of the keyword `undefined`, which was deprecated in LiveCode 8.0.0, has been removed. Use `nothing` instead.

  - Use `returns nothing` when defining a handler which returns no value.
  - Use `nothing` to indicate no value when manipulating optionally type variables

- The `is defined`, `is undefined`, `is not defined`, and `is not undefined` syntax, which was deprecated in LiveCode 8.0.0, has been removed. Use `is` and `is not` with `nothing` instead.

  - Use `<expr> is nothing` and `<expr> is not nothing` to test whether an expression has a value or not
  - The expression `<left> is <right>` will now evaluate to `true` if `<left>` and `<right>` are both nothing
  - The expression `<left> is not <right>` will now evaluate to `true` if one of `<left>` or `<right>` are nothing (but not both).

Variables

- Out parameters are now initialized by default to a suitable empty value at the start of the handler. For example:

```
public handler GetMyValue(out rValue as Integer)
end handler
```

  will result in `rValue` being set to 0.

- Handler local variables are now have lexical scope. This means variables are accessible from the point of definition to the end of the block they are defined in. Note that:

  - each `repeat` control structure is considered a single block.

  - each separate block in an `if/else if/else` control structure is considered a single block.

- Variables are now reset (either to their default value, or unassigned) at the point of the variable definition. In particular, any variables defined within a `repeat` block are reset on each iteration:

```
repeat 5 times
   variable tVar as optional String
   -- tVar is reset to "nothing" every time the loop runs
end repeat
```

- Variables in an inner block can now shadow those in an outer block. For example, the following is valid:

```
variable tX as Array
repeat 5 times
   variable tX as Number
   repeat 4 times
       variable tX as String
   end repeat
end repeat
```

- Typed variables are now initialised by default to a suitable empty value. For example:

```
variable tList as List
push "element" onto tList
```

Untyped and `optional` variables are initialised to `nothing`.

Bytecode Blocks

- bytecode can now be directly written in handlers using a bytecode block:

```
bytecode
   register tTemp
   assign_constant tTemp, 1
end bytecode
```

- for more details on what bytecode operations can be used see the LiveCode Builder Bytecode Reference

- bytecode blocks are not for general use and the current set of bytecode operations are subject to change

Unsafe Attributes

- The compiler now understands the idea of 'safety' of handlers and blocks of code.

- Handlers can be marked as being 'unsafe', e.g.

```
unsafe handler Foo()
    ... do unsafe things ...
end handler
```

- Blocks of statements can be marked as being 'unsafe', e.g.

```
unsafe
    ... do unsafe things ...
end unsafe
```

- All foreign handlers are considered to be 'unsafe'.

- All bytecode blocks are considered to be 'unsafe'.

- Calls to foreign handlers and unsafe handlers can only be made within unsafe handlers or unsafe statement blocks.

- Usage of bytecode blocks can only be made within unsafe handlers or unsafe statement blocks.

Specific LCB bug fixes (8.1.0-rc-1)

15181 Implement the `&` concatenation operator for `List`
18194 Return value of correct type when using 'the width of widget'

Specific LCB bug fixes (8.1.0-dp-3)

18022 Multiline block comment should terminate initial line.

Specific LCB bug fixes (8.1.0-dp-2)

16212 Escape XML reserved characters in manifest files
17526 Make variables lexically scoped in statement blocks.
17767 Process escape sequences in all string literals.
17809 Initialize out parameters to default.

Specific LCB bug fixes (8.1.0-dp-1)

14659 Initialise variables with a default value

LiveCode extension changes

Line Graph widget

Show and hide the graph lines

Setting the showLines of the graph to false will draw the point markers but not the connecting lines turning the graph into a scatter plot.

Marker styles

The markerStyles of the line graph can now be set to a return delimited list of:

- circle
- filled circle
- square
- filled square
- diamond
- filled diamond

Setting the markerStyles to empty will cause the lines of the chart to draw without markers and set the showLines of the chart to true.

SVG Icon widget

Properties

- New **scaledWidth** and **scaledHeight** properties have been added. These are read-only properties that expose the effective size of the rendered SVG path, independent of the widget's size.

  When **maintainAspectRatio** is false, these values are equal to the width and height of the widget.

Specific extension bug fixes (8.1.0-dp-3)

18054   Ensure that appearance reflects loaded state

Dictionary additions

- **deleteAudioclip** (*message*) has been added to the dictionary.
- **deleteVideoclip** (*message*) has been added to the dictionary.
- **eventAltKey** (*function*) has been added to the dictionary.
- **eventCapsLockKey** (*function*) has been added to the dictionary.
- **eventCommandKey** (*function*) has been added to the dictionary.
- **eventControlKey** (*function*) has been added to the dictionary.
- **eventOptionKey** (*function*) has been added to the dictionary.
- **eventShiftKey** (*function*) has been added to the dictionary.
- **libURLSetDriver** (*command*) has been added to the dictionary.
- **minStackFileVersion** (*property*) has been added to the dictionary.
- **mobileGetLocationHistory** (*function*) has been added to the dictionary.

- **mobileGetLocationHistoryLimit** (*function*) has been added to the dictionary.
- **mobileSetLocationHistoryLimit** (*command*) has been added to the dictionary.
- **newAudioclip** (*message*) has been added to the dictionary.
- **newVideoclip** (*message*) has been added to the dictionary.
- **scriptStatus** (*property*) has been added to the dictionary.
- **vectorDotProduct** (*function*) has been added to the dictionary.

Previous release notes

- LiveCode 8.0.2 Release Notes
- LiveCode 8.0.1 Release Notes
- LiveCode 8.0.0 Release Notes
- LiveCode 7.1.4 Release Notes
- LiveCode 7.1.3 Release Notes
- LiveCode 7.1.2 Release Notes
- LiveCode 7.1.1 Release Notes
- LiveCode 7.1.0 Release Notes
- LiveCode 7.0.6 Release Notes
- LiveCode 7.0.4 Release Notes
- LiveCode 7.0.3 Release Notes
- LiveCode 7.0.1 Release Notes
- LiveCode 7.0.0 Release Notes
- LiveCode 6.7.9 Release Notes
- LiveCode 6.7.8 Release Notes
- LiveCode 6.7.7 Release Notes
- LiveCode 6.7.6 Release Notes
- LiveCode 6.7.4 Release Notes
- LiveCode 6.7.2 Release Notes
- LiveCode 6.7.11 Release Notes
- LiveCode 6.7.10 Release Notes
- LiveCode 6.7.1 Release Notes
- LiveCode 6.7.0 Release Notes
- LiveCode 6.6.2 Release Notes
- LiveCode 6.6.1 Release Notes
- LiveCode 6.6.0 Release Notes
- LiveCode 6.5.2 Release Notes
- LiveCode 6.5.1 Release Notes
- LiveCode 6.5.0 Release Notes
- LiveCode 6.1.3 Release Notes
- LiveCode 6.1.2 Release Notes
- LiveCode 6.1.1 Release Notes
- LiveCode 6.1.0 Release Notes
- LiveCode 6.0.2 Release Notes
- LiveCode 6.0.1 Release Notes
- LiveCode 6.0.0 Release Notes