LiveCode 8.1.7-rc-1 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
- 10.12.x (Sierra) 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 8.2 on MacOS X 10.11
- Xcode 8.3 on MacOS 10.12

It is also possible to set other versions of Xcode, to allow testing on a wider range of iOS simulators. For instance, on OS X 10.10 (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 deployment for the following versions of iOS:

- 6.1 [simulator]
- 7.1 [simulator]
- 8.2 [simulator]
- 9.2
- 10.2
- 10.3

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)
- 7.0 (Nougat)

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>`

Platform	Path
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.
`-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

Send script form of send command (8.1.7-rc-1)

The syntax

```
send script <script> to <obj>
```

has been added to allow a chunk of script to be executed in the context of an object without any attempted evaluation of parameters that occurs with the original form of the send command.

For example, suppose there is a stack named "Stack" with script

```
on doAnswer pParam
   answer pParam
end doAnswer

function myName
   return the short name of me
end myName
```

and a button on the stack named "Button" with script

```
on mouseUp
   send "doAnswer myName()" to this stack
   send script "doAnswer myName()" to this stack
end mouseUp

function myName
   return the short name of me
end myName
```

clicking the button would result in an answer dialog first saying "Button" as the `myName` function would be evaluated in the button context, then "Stack" as using the `script` form would result in the `myName` function being evaluated in the stack context.

Synthesize an MS Paint compatible clipboard format for images (8.1.7-rc-1)

The engine will (once again) synthesize a DIBV5 format when an image is copied to the clipboard. This will be a 32-bit RGBA DIB. Windows then automatically synthesizes a 24-bit RGB DIB format.

Fix treatment of NUL containing arguments in ask dialogs (8.1.7-rc-1)

Prior to 7, any arguments passed to LiveCode provided ask dialogs (e.g. ask question) containing NUL would be truncated at the NUL. After 7, any such arguments would cause incorrect calling of

the ask dialog. The pre-7 behavior has been resurrected, meaning that trailing NUL bytes in arguments passed to ask dialogs will be ignored.

Improve export/import snapshot from screen on iOS (8.1.7-rc-1)

The from screen form of export/import snapshot has been changed to use a different API on iOS7+, which allows a greater variety of native layers to be captured.

Specific engine bug fixes (8.1.7-rc-1)

15302 **Fix common misspelling of occurred**
16131 **Ensure backdrop is sized to fill the screen on Linux**
17323 **Ensure backdrop window is behind all other windows on Linux**
17639 **Fix vertical placement of caret on long wrapped lines**
17657 **Make sure modifier keys are recognised in keysDown()**
18369 **Add explicit instruction to DMG images**
18447 **Moved misplaced text in iconGravity dictionary entry**
18526 **Allow command key shortcuts to work in color dialog**
18955 **Fix crash when using HTML file input dialog in browser widget**
19035 **Fix crash when using tsNet with OS X 10.6 and 10.7**
19080 **Do not show linking warnings when building iOS standalones**
19420 **Fix crash on startup when resuming android app after quit**
19599 **Ensure correct source rect is used for 'print card from lt to rb'**
19701 **Try HTTP basic authentication if a HTTP server responds with 401 without supplying the WWW-Authenticate header**
19713 **Synthesize an MS Paint compatible clipboard format for images**
19766 **Fix FTPS connection support on LC server under Linux**
19820 **Given the dictionary entries for remove and place references to each other.**
19857 **Make sure `clipsToRect` is included in the group properties**
19861 **Fix crash on startup on Linux when RGBA is not supported**
19869 **The effective revAvailableHandlers only includes private handlers of the target**
19891 **Ensure player controller thumb shows within the allowed range**
19893 **Ensure player respects startTime in reverse playback**
19929 **Fix crash when using click command with invalid mouse stack or click stack**
19935 **Made various improvements to the show and hide dictionary entries**
19936 **Ensure window masks with no transparency still work on Mac**
19950 **Fix crash due to invalid object in event queue**
19965 **Fix error when building a standalone if the added stackfiles have substacks**
19972 **Make sure setting playRate to negative does have an effect when player has reached the end of movie**

19983	Ensure `put URL tUrl` does not return empty when tUrl is invalid
19994	Ensure startup stack substacks are cleanly removed from memory
20014	Fixed several errors in the start/stop using font dictionary entries.
20019	Ensure `GetPixelHeightOfCanvas()` returns the height of the canvas
20030	Ensure the S/B always uses a valid certificate when codesigning iOS standalones
20061	Ensure data is not lost when opening and saving a stack with a widget that is not loaded
20080	Fix malformed documentation of the `the universal time`
20144	Corrected hiliteChanged dictionary entry for the switch button extension.
20167	Fix capitalization of menu item "Hide Others" on Mac
20201	Ensure `the controlNames` does not return numbers instead of names for controls in groups
20206	Mention in `revBrowserOpenCEF` dictionary entry that it is no longer supported on Mac in LC 8+
20209	Generate correct RGB values in rtftext
20212	Ensure setting the enabledTracks of a player is reliable
20231	Fix crash on Windows when exiting with taskbar hidden.
20232	Ensure dragdata["files"] returns a Unix path on all platforms
20239	Ensure answer folder shows the prompt on OSX 10.11 and above
20259	Fix crash when inserting large binary data to SQLite databases that aren't opened with the "binary" option
20282	Ensure 'the engine folder' returns a LiveCode path on Windows
20285	Fix crash when dispatching to an object and the defaultStack has been deleted
20298	Ensure a diamond checkmark is used when requested on Mac
20308	Fix iOS 64-bit Mach-O structure
20310	Make pasting from MS Paint work
20321	Fix treatment of NUL containing arguments in ask dialogs
20324	Convert dropped file paths correctly on Windows
9092	Fixed typo in the revPrintField dictionary entry.
9992	Improve export/import snapshot from screen on iOS

Specific engine bug fixes (8.1.7-rc-1)

17098	Ensure cursor moves to end of last tab in line.

IDE changes

Default handlers (8.1.7-rc-1)

Objects no longer have default scripts that appear in the script editor when their empty scripts are edited. Instead, all the associated message handlers for the object type now appear in a list underneath the list of handlers that are present in the script. When clicked, these lines add the selected default handler to the end of the current script.

If there is a default script for this handler and object type
in the appropriate location (Toolset/resources/supporting_files/default_scripts/ for 'classic' objects, /support/ for widgets), the content of the handler is obtained from that script (including preceding comments)

If there is no default script for this handler and object type, the handler is constructed using information from the documentation, namely the Summary element is used as a preceding comment to describe the handler, and then the handler declared with all the specified parameters.

First run backdrop (8.1.7-rc-1)

The IDE now has a backdrop by default on first-run. This can be turned off as usual via the view menu. Users with existing preferences should be unaffected.

ideScriptEdited message (8.1.7-rc-1)

A new IDE message has been added:

```
ideScriptEdited pScript, pObj
```

This message is sent when the script of an object as displayed in the script editor is changed. pScript contains the current contents of the script editor field for pObj, which, until applied, is not necessarily the same as `the script of pObj`.

Script editor handler menu (8.1.7-rc-1)

The Handler menu of the script editor menubar has been modified in accordance with the default handler changes to the script editor handler list. It now has the following structure:

Go to handler... -> list of extant handlers Add default handler... -> list of default handlers Show default handlers

The show default handlers menu item toggles the script editor preference to show the default handler list, which defaults to true.

Interactive Tutorial syntax (8.1.7-rc-1)

The syntax `load stack <FileName>` has been added to interactive tutorials. This allows prepared stacks to be imported as operating stacks in the current tutorial.

The prepared stack will be loaded from the internal resources folder of the tutorial (i.e. from `_resources/<FileName>`). Any `cTutorialTag` custom property of objects on the stack will be

converted to tags for objects which can subsequently be used in the current tutorial.

Specific IDE bug fixes (8.1.7-rc-1)

17485	Allow accessing Image Library and Object Library from LiveCode menubar
18915	Allow a 'set the name of stack' step in interactive tutorials
19511	Move "User Guide" higher in Help menu
19888	Fix some minor typos and errors in interactive tutorials
19889	Allow tutorial instruction window to be moved
19940	Always use Return key in tutorial instructions instead of Enter key.
19942	Fix typos in BMI tutorial
19943	Fix ambiguity in a couple of BMI tutorial instructions
19967	Fix second parameter evaluation for script in message box
19978	Ensure default script is editable from within the Extensions Builder
19984	Ensure custom props value comparison is case-sensitive
19985	Remove errant group from the script editor
20039	Position tutorial controls in better location when using 'Do It For Me'
20040	Use smaller images of todo list on smaller screens
20041	Prevent tutorials breaking when stack name is changed
20044	Detect specific errors in user scripts in tutorial
20046	Ensure there are separate scripting / apply steps in tutorial
20071	Make default handler name text grey
20072	Add space above default handler list and before each name
20074	Prevent removal of initial P from default handler name
20077	Default to 3-column tools palette
20102	Don't shortcut 'is changed' property steps
20103	Clear highlights before epilogue of interactive tutorial
20112	Unlock cursor on tab-command-alt key
20117	Don't override existing users' backdrop setting
20133	Ensure cloning stacks or cards from the Project Browser works correctly
20170	Fixed incorrect name of PI template stack
20171	Make sure LiveCode 8+ launches correctly if only an old (livecode.rev) Preferences file is present
6289	Ensure navigation with arrow keys works in the LiveCode Preferences window

LiveCode extension changes

Specific extension bug fixes (8.1.7-rc-1)

19907	Prevent LCB error when deleting key at path with no selection

Dictionary additions

- **deleteAudioclip** (*message*) has been added to the dictionary.

Previous release notes

- LiveCode 8.1.6 Release Notes
- LiveCode 8.1.5 Release Notes
- LiveCode 8.1.4 Release Notes
- LiveCode 8.1.3 Release Notes
- LiveCode 8.1.2 Release Notes
- LiveCode 8.1.1 Release Notes
- LiveCode 8.1.0 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