

LiveCode 8.1.5 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
 - Send script form of send command (8.1.5-rc-3)
 - Add param that suppresses success message when building standalone (8.1.5-rc-1)
 - Type should work with accented characters (8.1.5-rc-1)
 - revAvailableHandlers and revAvailableVariables now in all engines (8.1.5-rc-1)
 - Specific engine bug fixes (8.1.5-rc-3)
 - Specific engine bug fixes (8.1.5-rc-2)
 - Specific engine bug fixes (8.1.5-rc-1)
- IDE changes
 - Create script only stack behavior (8.1.5-rc-1)
 - Drag and drop stackfiles (8.1.5-rc-1)
 - Specific IDE bug fixes (8.1.5-rc-3)
 - Specific IDE bug fixes (8.1.5-rc-1)
- LiveCode extension changes
 - Specific extension bug fixes (8.1.5-rc-1)
- 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
- 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	<code><x86 program files folder>/RunRev/LiveCode <version></code>
Linux	<code>/opt/livecode/livecode-<version></code>

The installations when installing for "This User" are:

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

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
<code>-allusers</code>	Install the IDE for "All Users". If not specified, LiveCode will be installed for the current user only.
<code>-desktopshortcut</code>	Place a shortcut on the Desktop (Windows-only)
<code>-startmenu</code>	Place shortcuts in the Start Menu (Windows-only)
<code>-location LOCATION</code>	The folder to install into. If not specified, the <code>LOCATION</code> defaults to those described in the "Installation" section above.
<code>-log LOGFILE</code>	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 (8.1.5-rc-3)

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.

Add param that suppresses success message when building standalone (8.1.5-rc-1)

By default `revSaveAsStandalone` displays the message `answer information "Standalone application saved successfully."` when it is done. You can turn off this message by setting the test environment to true but doing so suppresses all error messages and other feedback as well.

I am calling `revSaveAsStandalone` from my own scripts multiple times and want feedback and error reporting but not the success message. Adding an additional parameter to `revSaveAsStandalone` that suppress the success message would allow this.

Type should work with accented characters (8.1.5-rc-1)

The `type` command now handles Unicode characters in a manner consistent with normal keyboard entry. If a Unicode character is typed and it has a native mapping, then it is propagated as a keypress with the keycode being the code of the character. If it has no native mapping, it is propagated with keycode equal to the Unicode codepoint with bit 22 set to 1. In either case the string value of the keypress is the Unicode codepoint.

revAvailableHandlers and revAvailableVariables now in all engines (8.1.5-rc-1)

The 'the [effective] revAvailableHandlers' and 'the revAvailableVariables' properties are now available in both the IDE and Standalone engines.

Note: These properties are currently undocumented, and reserved for IDE related uses. You are free to use in your own code, but their semantics could change in any release.

Specific engine bug fixes (8.1.5-rc-3)

19983 Ensure `put URL tUrl` does not return empty when tUrl is invalid

Specific engine bug fixes (8.1.5-rc-2)

19861 Fix crash on startup on Linux when RGBA is not supported

19869 The effective revAvailableHandlers only includes private handlers of the target

Specific engine bug fixes (8.1.5-rc-1)

- 11146** Ensure that the initial orientation is not upside down on Android when "portrait" is selected.
- 13482** Document optional catch clause in try control structure
- 14266** Fixing crash when using "import eps"
- 15997** Implement player 'mirrored' property on Windows
- 16758** Fixed bug causing crash when setting 65535 points to a graphic
- 17540** Fix crash on Mac when displaying IDE usage message
- 17850** Fix inconsistent browser callbacks in browser widget on iOS
- 17969** Fixed bug preventing correct rendering of ovals when creating them
- 18407** Fixed bug preventing the use of "set the visible of line <> of field <>"
- 18670** Fixed bug preventing all table names being retrieved in MySQL db's
- 18689** Fixed bug causing erroneous cursor movement when using arrow keys
- 18939** Fix player image incorrectly scaled in edit mode
- 19069** Fixed bug causing crash when building standalone
- 19275** Fix Android crash when a stack is deleted shortly after switching to another stack.
- 19327** Fix incorrect browser widget location when stack has a menu on OSX
- 19361** Added missing parantheses for revIsSpeaking()

- 19509 Make sure Android Studio works with LiveCode
- 19520 Make sure mirrored property works correctly on Mac
- 19528 Allow 'relaunch' message to be in behavior of main stack
- 19529 Fix incorrect snapshot area when primary screen origin is not 0,0
- 19535 Fixed error causing blue rectangle to be drawn incorrectly when doing CMD-A on text
- 19538 Fixed bug preventing users from selecting text of length 65535
- 19541 Fix clipboard ownership checks on windows causing private clipboard data to be cleared
- 19581 Do not send pushNotificationReceived/localNotificationReceived message twice
- 19593 Type should work with accented characters
- 19613 Make sure setting the htmlText of a field does not remove superscripts
- 19615 Fixed references in the quit dictionary entry
- 19620 Update cursor when entering window on windows
- 19630 Make sure setting the iphoneSetAudioCategory is respected
- 19633 Modify tsNet libUrl wrapper to treat HTTP status codes > 400 as errors
- 19637 Errors building iOS standalones should only be reported once
- 19646 Make sure using mobile camera does not change the value of mobileLockIdleTimer
- 19649 Correctly parse multiple bytes escaped as hex in the format function
- 19650 Fix tsNet proxy support on Linux, OS X and Android
- 19666 Make sure you can set more than one javascriptHandlers on CEF Browser
- 19668 Fixed bug causing crash when using custom undo
- 19672 Prevent crash on throwing certain errors
- 19687 Preserve error from chunk-of-code send form
- 19688 Ensure 'put the objProp' causes a parse error
- 19693 Ensure closed players use no system resources
- 19699 Fixed bug that overrides previous matches in matchText
- 19742 Fix crash when deleting an object when a socket has a reference to a deleted object
- 19743 Fix crash when checking a watched variable on a deleted object
- 19775 Fix various error inconsistencies in selected object cut and delete
- 19796 Fix crash when SSL is unavailable on HTML5
- 19797 Implement put before msg in HTML5
- 19808 Notify property listener when text of control is changed via put cmd
- 19826 Fix iOS simulator deployment with Xcode 8.3.3
- 19830 Fixed regression introduced by bugfix-19535
- 19837 Fix crash due to deletion of object with pending message
- 19841 Message box does not find handlers in behaviors
- 19843 revAvailableHandlers and revAvailableVariables now in all engines

IDE changes

Create script only stack behavior (8.1.5-rc-1)

The menu for assigning a behavior to a control has two additional options:

- Create behavior from new script only stack
- Create behavior using control script and script only stack Either option will prompt you for a stack name and a location for the script only stack. The new stack will be saved, assigned as the behavior of the control, and then added to the stackfiles property of control's stack.

Drag and drop stackfiles (8.1.5-rc-1)

You can now drag and drop stack files onto the stackFiles field in the PI.

Specific IDE bug fixes (8.1.5-rc-3)

- 19967 Fix second parameter evaluation for script in message box
- 19985 Remove errant group from the script editor

Specific IDE bug fixes (8.1.5-rc-1)

- 18201 Make sure rulers can be hidden
- 18598 Prevent error on backspace in empty script editor
- 18685 Remember the last position of menubar on Windows and Linux
- 18878 Setting stackFiles in PI causes an error if you "cancel" the file dialog or select multiple files
- 19264 Ensure LCB errors display reasonably in script editor
- 19480 Ensure message box execution succeeds first time if no compile error
- 19547 Fixed bug preventing users from finding | and - in the script editor
- 19585 Improve rendering of Interactive Tutorial on Windows when screenPixelScale > 1
- 19589 Fix 'put globalVar' in msg box
- 19627 Clear deleted objects from project browser correctly
- 19629 Check for changes in filename when getting object row in project browser
- 19838 Add explicit 'select object' steps to the Interactive Tutorials to ensure the correct object is always selected.
- 5787 Drag and drop stackfiles

LiveCode extension changes

Specific extension bug fixes (8.1.5-rc-1)

- 16241 Runtime error when changing itemCount by more than one

Previous 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](#)