LiveCode 8.0.0-dp-5 Release Notes

Table of contents

Overview

LiveCode 8.0 is the most exciting release in the history of the technology. It provides a simple way to extend the functionality or control set of LiveCode.

Our focus in LiveCode 8.0 is extensibility. You can now build and share widgets (custom controls) and libraries that are treated by LiveCode as engine level elements.

LiveCode 8.0 can be thought of as a version 7.0 with a new module allowing extensions to be plugged into the engine. As a result, 8.0 should be as functional and stable as LiveCode 7.0.

I don't want to build extensions. What's in it for me?

Many love LiveCode because of the productivity benefits and donâ€™t have time to build extensions. If that is the case just kick back and start using LiveCode 8 and keep an eye on the extensions portal. You can start using new controls and libraries as they are built by other community members.

LiveCode Script vs LiveCode Builder

To make it possible to create extensions and plug them into the LiveCode engine we've created a new flavour of our language called . LiveCode Builder looks a lot like LiveCode Script so should feel familiar for any seasoned LiveCode developer. There is lots of new syntax which exposes parts of the LiveCode engine that were only previously available to those who were skilled c/c++ developers.

LiveCode Builder is a new language and is therefore highly experimental and should be considered an early prototype. It will take some getting used to but we know you'll love it once you see how powerful it is. The best way to get started is to read the "Extending LiveCode" guide which can be found in the dictionary under the "Guide" tab.

Warning

It is important to stress that **no aspect of this release should be considered final**. **Every piece of syntax in LiveCode Builder is subject to change**

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.
- revBrowser for 32-bit Linux fails to run (causing the dictionary to be blank).

Platform support

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

Windows

The engine supports the following Windows OSes:

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

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

Linux

The linux engine requires the following:

- Supported architectures:

32-bit or 64-bit Intel/AMD or compatible processor
32-bit ARMv6 with hardware floating-point (e.g. RaspberryPi)

- Common requirements for GUI functionality:

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

- Requirements for 32-bit Intel/AMD:

glibc 2.3.6 or later

- *Requirements for 64-bit Intel/AMD:*

glibc 2.15 or later

- Requirements for ARMv6:

glibc 2.7 or later
Note: The GUI requirements are also required by Firefox and Chrome, so if your Linux distritution runes one of those, it will run the engine.
Note: If the optional requirements are not present then the engine will still run but the specified features will be disabled.
Note: It may be possible to compile and run LiveCode Community 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*

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

Setup

Installation

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

- Windows: <x86 program files folder>/RunRev/LiveCode 8.0.0-dp-5
- Linux: /opt/runrev/livecode-8.0.0-dp-5
- Mac: /Applications/LiveCode 8.0.0-dp-5.app

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

- Windows: <user roaming app data folder>/RunRev/Components/LiveCode 8.0.0-dp-5
- Linux: ~/.runrev/components/livecode-8.0.0-dp-5
- Mac: ~/Applications/LiveCode 8.0.0-dp-5.app

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

Uninstallation

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

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

```
cd /opt/runrev/livecode-8.0.0-dp-5
```

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

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

- follow the on-screen instructions.

Reporting installer issues

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

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

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

Activation

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

Multi-user and network install support (4.5.3)

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

Command-line installation

It is possible to invoke the installer from the command-line on both Mac and Windows. When invoked in this fashion, no GUI will be displayed, configuration being supplied by arguments passed to the installer.
On both platforms, the command is of the following form:
<exe> install noui *options*
Here *options* is optional and consists of one or more of the following:

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

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

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

Command-line activation

In a similar vein to installation, it is possible to activate an installation of LiveCode for all-users of that machine by using the command-line. When invoked in this fashion, no GUI will be displayed, activation being controlled by any arguments passed.
On both platforms, the command is of the form:
<exe> activate -file *license* -passphrase *phrase*
This command will load the manual activation file from *license*, decrypt it using the given *passphrase* and then install a license file for all users of the computer. Manual activation files can be downloaded from the 'My Products' section of the LiveCode customer accounts area.
This action can be undone using the following command:
<exe> deactivate
Again, as the LiveCode executable is actually a GUI application it needs to be run slightly differently from other command-line programs.
In what follows <livecodeexe> should be replaced with the path to the installed LiveCode executable or app that has been previously installed.
On Windows, you need to do:
start /wait <livecodeexe> activate -file *license* -passphrase *phrase*
start /wait <livecodeexe> deactivate
On Mac, you need to do:
“<livecodeexe>/Contents/MacOS/LiveCode” activate -file *license* -passphrase *phrase*
“<livecodeexe>/Contents/MacOS/LiveCode” deactivate
On both platforms, the result of the activation will be written to the console.

Engine changes

LiveCode Builder Host Library (8.0.0-dp-5)

Simplified Canvas Effect Constructor

You can now create a new canvas effect object without setting up an array of properties.
Default values will be assumed for unspecified properties:

- size: 5
- spread: 0
- distance: 5
- angle: 60

Example:
variable tEffect as Effect
put outer shadow effect into tEffect

Ability to set the dontUseQT property for a player object (Windows and OSX) (8.0.0-dp-5)

It is now possible to set the dontUseQT property for a player object. On Windows, the default value of the dontUseQt (global) property is false. This means that any player object created will use the QuickTime API for multimedia playback. With this new feature, you can set the dontUseQT property of a player to true, without changing the value of the global dontUseQt property. In that way you can have both QuickTime and non-QuickTime players playing at the same time.

On OSX, the default value of the dontUseQT (global) property is true if the OSX version is greater or equal to 10.8. 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. In that way you can have both QuickTime and AVFoundation players playing at the same time. This can be particular useful for supporting some media formats or codecs that are not supported by the default AVFoundation player
(for example .midi files, Sorenson Video 3, H.261 codecs etc)

mobileLaunchData function (8.0.0-dp-5)

This new function is available on Android and returns an array containing information from the Intent object used to launch the currently running app.

Available information

- `action` - The general action the app was launched to perform.
- `data` - The data to operate on.
- `type` - The MIME type of the data provided.
- `categories` - Additional information about the action to perform.
- `extras` - An array of action-specific data set by the calling activity.

New "mirrored" property for the OSX player. Applies to both QTKit and AVFoundation

player. (8.0.0-dp-5)

The default value of this property is "false". Setting this property to "true" makes the video frames to be displayed mirrored.

Syntax: set the mirrored of player "myPlayer" to true

New 'readyForDisplay' ios player property (8.0.0-dp-5)

We added a new readyForDisplay property to the ios player. This maps to the native readyForDisplay property, and is read-only. For more details see the dictionary entry of mobileControlGet function.

Script Only Stack Property (8.0.0-dp-5)

There is a new boolean stack property **scriptOnly** which specifies whether the stack should be saved as script only. A script only stack does not retain any objects or custom properties.

Xcode 7.0 (8.0.0-dp-5)

iOS 9.0 SDK is now part of the valid SDKs that can be used for iOS standalone application deployment, from Mac OSX 10.10.4.

If you installed the new version of Xcode at another location than the previous one, we invite you to update the Xcode in Preferences > Mobile Preferences.

All the previously existing versions of iOS supported are still supported.

LCS-Widget: Add 'popup widget' command (8.0.0-dp-5)

Syntax: `popup widget <kind> [ at <location> ] [ with properties <propertyArray> ]`

Summary: Opens a widget within a popup window.

Example:

```
local tProps
// Set the size of the popup
put "0,0,120,50" into tProps["rect"]
// Set the initial color value
put "1,1,0.5" into tProps["initialColor"]

// Show the widget in a popup window
popup widget "com.example.mycolorpicker" at the mouseloc with properties
tProps
```

revCapture - revCaptureListVideoCodecs() results in crash (8.0.0-dp-5)

To palliate this problem, some getters in the library revCapture must return possibly UTF-8 encoded names (such as the codecs) to allow the script writer to set them.
In the same idea, some setters can be given UTF-8 encoded strings.

Affected getters:
- revCaptureListAudioInputs

- revCaptureListVideoInputs
- revCaptureGetAudioInput
- revCaptureGetVideoInput
- revCaptureGetPreviewImage
- revCaptureListAudioCodecs
- revCaptureListVideoCodecs
- revCaptureGetAudioCodec
- revCaptureGetVideoCodec
- revCaptureGetRecordOutput

Affected setters:
- revCaptureSetAudioInput
- revCaptureSetVideoInput
- revCaptureSetPreviewImage
- revCaptureSetAudioCodec
- revCaptureSetVideoCodec
- revCaptureSetRecordOutput

Support for the web platform using HTML5 (8.0.0-dp-5 - experimental)

The LiveCode engine will now run in web browsers that support HTML5. This means that you can now deploy simple LiveCode apps to users without any installation required.

To deploy a stack as an HTML5 application, enable the "Build for HTML5" checkbox on the "HTML5" page of the standalone settings window, and then generate the standalone in the normal way.

For more information on HTML5 deployment, including options for embedding LiveCode standalones in web pages, please see the "HTML5 Deployment" guide in the IDE dictionary.

Important: This feature is currently experimental. This means that it may not be complete, or may fail in some circumstances that you would expect it to work. Please do not be afraid to try it out as we need feedback to develop it further.

the *commandName* and the *commandArguments* (8.0.0-dp-3)

12018 have been longstanding issues: *$0* is not the command name, but the first command argument (which does not follow the way Bash works).

To solve this issue and avoid breaking any script that uses the current way command arguments are retrieved wth LiveCode (*$0* being the first commandline argument instead of the command name), we introduced two functions to allow the users to access the commandline name and arguments.

- the `commandName` returns the command that has been used to start the executable
- the `commandArguments` returns a 1-based, numeric array of the commandline arguments if no index is given. Returns the arguments at this index otherwise (or empty if the index is < 1 or > number of parameters)

These functions are only implemented for desktop standalone applications and server scripts. They will return *empty* on mobile platforms and in the IDE.

Image metadata (8.0.0-dp-3)

A new read only image property has been added to access the metadata in the image file. The returned array is in the same format as that used for the export command. If no metadata is found then the property returns empty rather than an array with empty elements. Currently the only metadata key that is

implemented is `density` which can be used to determine pixel density in pixels per inch. Metadata is currently only parsed from JPEG and PNG file formats.

For example:

```
put the metadata of image 1 into metadataArray

set the width of image 1 to the width of image 1 div (medatadaArray["density"] / 72)
set the height of image 1 to the height of image 1 div (medatadaArray["density"] / 72)
```

LiveCode Builder (8.0.0-dp-3)

LiveCode Builder Language

LiveCode Builder is a variant of the current LiveCode scripting language (LiveCode Script) which has been designed for 'systems' building. It is statically compiled with optional static typing and direct foreign code interconnect (allowing easy access to APIs written in other languages). The compiled bytecode can then be packaged together with any required resources (icons, documentation, images, etc) into a .lce extension package.

Unlike most languages, LiveCode Builder (LCB) has been designed around the idea of extensible syntax. Indeed, the core language is very small - comprising declarations and control structures - with the majority of the language syntax and functionality being defined in modules.

Note: It is an eventual aim that control structures will also be extensible, however this is not the case in the current incarnation).

The syntax will be familiar to anyone who has coded with LiveCode Script, however LiveCode Builder is a great deal more strict - the reason being it is intended that it will eventually be compilable to machine code with the performance and efficiency you'd expect from any 'traditional' programming language. Indeed, over time we hope to move the majority of implementation of the whole LiveCode system over to being written in LiveCode Builder.

Note: One of the principal differences is that type conversion is strict - there is no automatic conversion between different types such as between number and string. Such conversion must be explicitly specified using syntax (currently this is done using syntax like *... parsed as number* and *... formatted as string*.

Extensions

There are two types of extensions which can be written in LCB: widgets and libraries. All installed extensions appear in the new Extension Manager stack, which can be opened from the Tools menu.

An LCB library is a new way of adding functions to the LiveCode message path. Public handlers in loaded LCB libraries are available to call from LiveCode Script.

A widget is a new type of custom control which, once compiled and packaged, can be loaded into the IDE. Using the widget is no different from any of the classic LiveCode controls you've been used to. Simply drag it onto a stack and start interacting with it as you would any another control.

You can reference the widget in script as a control:

```
set the name of the last control to "clock"
```

Or more specifically as a widget:

```
set the tooltip of widget 1 to "This is my nice new clock widget"
```

Getting Started

To get started with LiveCode Builder, click on the "Dictionary" icon in the IDE toolbar, select the "Guide" tab and then "Extending LiveCode" from the drop-down menu. This will show you the user-guide on getting started with writing widgets and libraries in LCB. Alternatively, you can start by looking at some of the extensions shipped with LiveCode 8 - the source and other resources for these are located in the "extensions" sub-folder of your LiveCode installation directory (source files are named

Mobile Sockets (8.0.0-dp-3)

Socket support has been added to the mobile platforms. The following syntax has been added to the iOS and Android engines.

Commands:

- accept
- open socket
- close socket
- read from socket
- write to socket
- secure socket

Functions:

- openSockets

Messages:

- socketClosed
- socketError
- socketTimeout

Properties:

- socketTimeoutInterval

If you are secure sockets, the SSL library must be included in your standalone. To do this for iOS, make sure the "Encryption" checkbox of "Basic Application Settings" section on the iOS screen of the Standalone Application Settings window is selected. To do this for Android, make sure the "SSL & Encryption" checkbox of "Basic Application Settings" section on the Android screen of the Standalone Application Settings window is selected.

Packaged extensions naming consistency (8.0.0-dp-3)

Earlier versions of the widgets and libraries which are bundled with the IDE were named inconsistently.

Now all LiveCode extensions are named either com.livecode.widget.<widget name> or com.livecode.library.<library name>.

Note: This change will break some stacks that have widgets saved on them, or scripts which refer to a widget by its kind.

Feature: Popup Widgets (8.0.0-dp-3)

Added the ability to use widgets within popup dialog windows.

New Syntax:

- popup widget <Kind> at <Position> [with properties <Properties>]
- Launch the named widget as a popup. The popup can return a value in the result.
- currently popped up
- test if this widget is part of a popup
- close popup [returning <Result>]
- set the result of the calling popup statement to <Result>

Various bugs with navigation bar widget (8.0.0-dp-3)

- Selecting "names" as the itemStyle does not work
- Changing the navIcons via script / property inspector does not work
- The navSelectedIcons property is missing.
- editMode should default to false
- add 'navigate' message to widget docs

'sort cards of background ...' crashes (8.0.0-dp-3)

You cannot 'sort cards of background' so this now throws an error, rather than crashing.

revZipOpenArchive can fail on 64-bit Linux (8.0.0-dp-3)

The revZipOpenArchive command can fail to open a valid zip archive on 64-bit versions of Linux. This was due to a 64-bit cleanliness problem in the libzip library which has now been fixed.

Objects are only deleted on idle (8.0.0-dp-3)

The engine will now flush any recently deleted objects after each command as long as they were created during the current event handling loop.

If an object is created during one event handling loop, and then deleted during another nested event handling loop it won't be flushed until control returns to the original event handling loop.

The upshot is that in tight loops, creating and deleting objects will result in objects being flushed immediately, reducing memory usage and making it easier to write object processing code which creates and deleted many objects.

Specific bug fixes (8.0.0-dp-5)

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

15934 **Holding down the SHIFT key while pressing the Backspace key does nothing**

15908 **LCB: calling libffi closures causes Android crash**

15897 **LiveCode crashes when trying to clone a group containing a widget object whose kind is not installed**

15895 **Using put with only the first half of a surrogate pair in the message box locks up IDE**

15870 **HTML5: embedded images are colour-swapped**

15868 **HTML5: 'the mouseloc' always returns 0**

15866 **line breaks do not survive parsing by documentation reader**

15848 Crash on repeating revXMLMatchingNode twice on the same tree
15846 MCWidgetEvalIsPointWithinRect(pPoint
15845 Solid Paint With Color API entry incorrect
15844 mobilePickDate "time"
15836 Widgets: OnOpen / OnClosed messages sent when widget is relayered.
15822 watchedvariables and breakpoints property parsing too strict
15816 MobilePick does not display Cancel Done if index is filled in
15814 Can't read a file using UNC path in Windows
15808 Widgets: SVG Icon widget not documented
15805 Docs: Second Syntax element overwrites first
15798 Array property references in variables not resolved properly
15764 Android mobileControlCreate "player" not working
15763 Can't select the last control by chunk expression
15756 various codepoint and codeunit functionality broken in standalones
15750 Documentation updated for URL("file:") and Unicode files
15738 The "My LiveCode/Plugins" folder isn't created by the IDE when updating the BAF
15684 "extras" key in array returned with mobileGetLaunchData is empty
15676 Unicode chars in a script can make the Script Editor crash on script opening
15649 cursor property not reset to empty at cursor unlocking in LiveCode 7
15606 systemVersion reports incorrect value from Windows 8.1
15345 Rendering issues
15273 setting imagesource in tabbed field hides text and gives wrong width
15208 LiveCode crashes when using System Character Viewer
15129 LCS-Widget: Add 'popup widget' command
14970 Dictionary: Tables not correctly specified in htmltext entry
14837 tan entry in API dictionary contains typographical error. function end on line 40 was cosInDegrees instead of tanInDegrees
14447 [[Docs]] QTVersion of QuickTime return 0.0
14423 revCapture - revCaptureListVideoCodecs() results in crash
13820 Player object cannot play a midi file on Yosemite.
13754 Changing the scalefactor might hide the stack off screen
13694 Impose a limit of 1024 form fields in an HTTP POST request.
9942 The "system time" is always retured in long format on Linux
9744 iOS video playback starts with black frames
1751 command "flip the selobj vertical" returns errors

Specific bug fixes (8.0.0-dp-4)

15752 Large variable cause the IDE to hang

Specific bug fixes (8.0.0-dp-3)

15723 Incorrect wording in Business edition activation screen
15719 An error in a preOpenStack script aborted openStack
15700 iOS sockets only accept a single connection
15692 [Docs] Dictionary entry for "flip" command is missing text
15681 Occasional issue parsing SVG data in LCB

15675 IDE crashes on startup if DataTree is installed
15648 Exporting text to RTF text generates invalid output if there is the backgroundcolor has been changed
15646 Memory leak when using 'put into of '
15633 ceiling (*ceil()* synonym) is missing from the dictionary
15630 get property tVar of my script object not working in develop branch
15623 machine() returns "unknown" in LiveCode 7.x on OSX
15620 Check if m_rep is nil in MCImage::GetMetadataProperty
15618 Codeunit and delimited chunk offsets probably broken in 8.0
15605 Property inspector does not update when graphics are being created using tools
15602 Recursionlimit property cannot be set by script > 65535
15597 LiveCode 7 fail to export valid RTF text if lists are used
15592 memory leak in shell() on Mac
15577 Execution error caused when trying to set the "visited" property of a text link
15571 textEncode crashes with encoding UTF-32
15569 Referenced jpg crashes application on Linux
15568 Crash if SSL lib can't load properly - ARTS
15556 Mouse focus not synced when object is created
15516 Don't crash when attempting to get a child of a non-group control
15514 Blocking socket syntax now works in LiveCode Server.
15509 Condition " in " does not throw parse error for 'case' or 'repeat until/while'
15508 Allow debug mode standalone engines to load script-only TEST_STACKs
15494 mobileControlget ID
15482 Fix Linux drag-and-drop issues between LiveCode controls
15474 Max interger literal number limited to 2^32 on 64-bit Linux
15472 Invisible player appears and causes LC to hang
15467 Memory leak in MCPlayer::resolveplayerfilename
15457 Repeat for each element subtly different in 7.0
15455 custom property gets truncated if it contains a NULL byte
15439 enabledTracks "can't set this property" in LC 7
15424 LC 7 not playing file from server in Windows
15422 The number of trueWords returns the number of words as per old definition
15405 LCS: Can't create a widget in a group
15403 mobileAddContact doesn't work
15386 Failure to incorporate code with 'include' on 7.0 servers
15383 global allowabledragactions property broken
15379 Regular Expression with binary input fails in LC 7
15378 backColor doesn't work properly for graphic on closed stack
15371 mobileGetContactData fails when there are multiple labels of the same property
15370 mobileFindContact crashes LiveCode when contact not found
15368 put before line x of URL broken on LC 7 & 8
15359 Export image with list broken in LC 7.0
15358 LC 8 has a very noisy startup
15352 Set label field contents to empty in Property Inspector causes crash
15327 offset(tosearch
15321 "record sound" is not creating a file in LC7 (Windows)

15315 All text properties of a field's text are deleted when putting text after the last line
15314 "secure socket" crashes livecode
15309 token chunk broken in complex chunk expressions
15305 Entering keystroke option-e twice in a field crashes LiveCode
15296 mobileExportImageToAlbum fails on Android
15286 Palette Actions: Nav items need tooltips
15267 byte x to - 1 of string fails in 7.0.4
15264 Crash in Stack with Player Object when selecting "Tracks" in the player property inspector
15258 Custom properties lose unicode text when copied
15243 Checking tabStops property over line range with differing tabStops set causes crash
15241 Crash when setting clipboardData to wrong type of value
15224 Various bugs with navigation bar widget
15214 IDE-Widgets: Icon picker does odd things when resized
15200 Default cursor doesn't reset when set to empty
15196 Setting Filename of Player in Substack to Empty Causes Crash
15193 export snapshot leaks memory and eventually crashes LC 7.0.4 rc3
15191 Windows - Hiding a player in Run mode does not update the screen
15175 tabAlign field property is not saved in the stack file
15168 Setting the beep sound crashes iOS simulator in LC 7.0.x
15164 Externals do not load if their path has a Unicode character
15161 WeekDayNames on Linux Ubuntu miss accented characters
15156 Putting value into item of empty variable hangs LiveCode
15151 libJson not working on Android sim or Device
15140 String list parsing for properties incorrect
15139 Putting string data into complex byte chunk crashes LiveCode
15124 mobilegetcontactdata crashes app on ios 8.2
15123 Crash when deleting a control in edit group mode
15105 Appleevent handler crashes 7.0 standalone
15096 Escape key does not dismiss the 'answer color' dialog
15090 App quits with setting accelerated rendering on opening in iphone 5S and 6
15062 Standalones saved in a folder with Unicode char will not launch on Mac
15056 Read from file for (x | x chars | x bytes) returns empty
15045 Parse issue with nested char chunk expression
15040 alt+0244 text entry field does not work on Windows
15034 import snapshot does not work correctly on 64-bit Linux.
15030 the effective textstyle crashes LiveCode
14996 LCB-Canvas: polyline path
14994 Clicking in field with listBehavior true and lockText false crashes
14965 setting cursor to id of non-existing image crashes LC 7.0.4 rc1
14961 Gradient - Quality set to "good" makes LC crash
14960 LC8 DP1 crash while importing JPG image
14885 'sort cards of background ...' crashes
14855 mobileSensorReading(sensor_name
14835 Fix word wrapping in fields containing tabs
14833 Format strings of the form "%0s" do not work correctly.

14814 print cardâ€¦ into pageRect fails in LC7.0.3
14806 LCB-Canvas: curve through examples are incorrect in docs
14792 MobileUpdateContact fails on iOS/Android in LC 7.0.x
14766 iOs standalone encryption not working in LiveCode 7.0
14717 Points properties of graphic are too strict - they should except either '
14712 Save As converts names to lower-case on Linux
14700 revZipOpenArchive can fail on 64-bit Linux
14696 Open File Refuses To Work With 1GB+ Files on Windows
14661 LiveCode 7 IDE Doesn't Use Standard Windows Font
14612 read from file COM1: --- port doesn't work ---
14611 answer file/folder .. with .. does not work anymore with backslash delimiter
14438 mobileGetContactData does not work on Android in LC 7.0
14413 revZipOpenArchive fails in 7.0.1 with Unicode characters
14326 revZip externals does not work on iOS
14287 openSockets returns an extra char to an open port
14160 LiveCode 7.0 silently crashes if the Print Spooler is turned off
14139 ChartsEngine 1.2.1 can't be registered with LiveCode 7
13923 formattedHeight of a button is incorrect if button has an icon
13575 Dictionary - iphoneSetKeyboardReturnKey examples
12108 Better argument handling for livecode server scripts
7217 selectionChanged not sent on arrow navigation
6791 Objects are only deleted on idle
3537 When inputting non-latin data into a field

Specific bug fixes (8.0.0-dp-1)

14851 Popup won't stop displaying when displayed in mouseDown of button widget
14602 URLEncode crashes LiveCode
14599 LCB: Text sort is inconsistent with string comparison
14538 bool formatted as string does not work

IDE changes

Point editor (8.0.0-dp-5)

A point editor (com.livecode.pi.point) has been added to manipulate appropriate properties in the property inspector (eg hotspot, loc)

Menu bar (8.0.0-dp-3)

The menubar has been made a script-only stack to facilitate bugfixes and community contributions. Users should not notice much difference in terms of its appearance. Some of the menu items have been changed, however:

The 'New Mainstack' item now has a submenu with a range of size choices, as well as the option to create a script-only stack. Selecting script-only stack will prompt a choice of name, and subsequently open the stack in the script editor.

We have centralised the building and handling of contextual menus in the menubar script, thereby making per-object contextual menus display and behave consistently throughout the IDE.

The Object > New Control submenu is now generated based on the property information present for each object type, and the newly added Object > New Widget submenu is generated based on the currently loaded widget extensions.

Property Inspector

A number of changes have been made to property editors in the property inspector:
- The color editors now use a color swatch widget to display the chosen color
- Numeric editors have a slider if the property has an associated min/max, and an increment/decrement twiddle if it has a step value.
- The navbar widget now uses a version of itself as an editor for its properties (com.livecode.pi.navbar)
- A graphic effects property editor has been added (com.livecode.pi.graphiceffect)
- A gradient property editor has been added (com.livecode.pi.gradientramp)
- A script property editor has been added, which contains a button to edit the selected script (com.livecode.pi.script)
- A time zone property editor has been added, which contains a drop-down list of time zones (com.livecode.pi.timezone)

Widget metadata and the IDE

Widget metadata now controls a number of additional features with respect to how the widget interacts with the IDE.
Firstly, the preferredsize attribute controls the initial size of the widget when dragged out from the tools palette.
For example, the navbar widget now has

metadata preferredSize is "320,49"

so that when dragged out, it is created at the correct size for an original iPhone screen.

Secondly, the uservisible attribute controls whether the widget appears at all in the tools palette of the IDE. A number of widgets have been declared user invisible for this release, either because they are not meant to be
draggable objects at all (eg the icon picker widget, which is designed to be popped up) or are not quite refined
to the point where they are suitable for user stacks, but are included because they are being used in the IDE (for example the tree view widget).

Finally if present, the svgicon attribute will be used to display an icon for the widget in the tools palette, taking precedence over the included icon resources. All of the widgets included by default in the tools palette now use
svg icon paths.

Standalone Settings

A field has been added to the Copy Files tab of the standalone settings which is populated with the list of currently
installed extensions. All selected extensions from this list are included in standalones and loaded when the standalone
is launched. 'Use' dependencies are automatically calculated and included along with the top-level widget.

Property Inspector (8.0.0-dp-3)

The property inspector has been rewritten to allow properties of widgets to be inspected and edited. It has been implemented with flexibility and extensibility in mind, since it must be able to control the values of widget properties in any way required by the widget developer. Each property now has a number of attributes which affect how it appears in the inspector.

Property Attributes

The following is the list of property attributes:

default

The default value of the property. If there is no default value (for example the 'loc' property does not have one), the string "no_default" can be used. The property inspector pops up a contextual menu when editors are right-clicked allowing the user to set the property back to a default value.

editor

The editor that will be used to display the value of the property and allow it to be edited. See the dedicated section below for details on property inspector editors.

group

Properties are grouped by themselves in the inspector by default. If a particular group name is specified for a set of properties, their editors are placed next to each other in the inspector.

label

The label to use for this property.

options

For properties whose value is a choice from a set of options, that set should be specified as a comma delimited list for the options attribute. Default editors are provided for 'enum' type properties (choice of exactly one from a set) and 'set' type properties (choice of zero or more from a set).

Lists of options can be generated using LiveCode Script for the inspector at run-time, by using the 'execute' syntax - for example the options for the textFont property are generated using

```
execute: get the fontNames; sort it
```

Whatever remains in the 'it' variable after executing the specified script is used as the list of options.

section

The section attribute controls which tab of the property inspector contains the property in question. Currently this is required to be one of the following

- Basic
- Data Grid
- Custom
- Table

- Colors
- Effects
- Icons
- Position
- Text

But in the future it may be possible to specify custom sections.

user_visible

Properties are visible in the property inspector by default. Set the user_visible attribute to false to hide a given property from the user.

read_only

Read only properties will be displayed in the property inspector but the corresponding editor will have its "editorEnabled" property set to false. See the Editors section below for more details on enabled/disabled editors.

Widget Properties

Widget metadata is used to control the display of widget properties in the inspector. Items of metadata which determine property attributes are of the form:

metadata <property>.<attribute> is "<value>"

These are stored as property data for the widget at load time. The <attribute> can be any of those specified in the Property Attributes section above. If the attributes are not specified, their values are as follows:

- default - "no_default"
- editor - "com.livecode.pi.number" for Integer/Real properties, "com.livecode.pi.<type>" for properties of type <type>.
- group - the name of the property
- label - the name of the property
- options - empty
- section - "Basic"
- user_visible - true
- read_only - true if there is no specified 'set' handler or variable for the property, false otherwise.

Script Object Properties

Script-level properties of objects (including widgets) are specified in files in the Toolset/resources/supporting_files/property_definitions folder. The propertyInfo.txt file specifies the default values for all the property attributes. Each object type then has a specification of which properties should be displayed in the inspector when it is the selected object, and any options/default/group values which override the defaults.

Editors

Currently an editor must be a stack consisting of a group named "template" and a button named "behavior". The property inspector looks up the specified editor for a given property, clones the template group, and sets its behavior to the long id of the button.

The behavior script must at a minimum implement the following three handlers:

on editorInitialize
on editorUpdate
on editorResize

There are a number of properties available to any editor:

- editorMinWidth
- editorMaxWidth
- editorEnabled
- editorEffective
- editorValue

These should be set or got appropriately. For example, if an editor consists of a text field, the editorUpdate handler should update the value of the field with 'the editorValue of me'. Similarly, if the content of the field changes, the field should call a function in the behavior which sets 'the editorValue of me' to the content of the field.

The editorEnabled and editorEffective properties are set by the generic behavior depending on the property info and the values of the properties. The editorEffective is true if the value of the property in question is empty but there is an effective value in play. The editor should alter the display of its value accordingly.

Editors can specify their min and max width if required.

The following editors are built-in, and available to use for widget properties with common types:

- com.livecode.pi.array - a Tree View widget
- com.livecode.pi.boolean - a check box
- com.livecode.pi.color - a color swatch and dialog
- com.livecode.pi.colorwithalpha - a color swatch and dialog, and alpha value slider
- com.livecode.pi.enum - an option menu
- com.livecode.pi.file - a file selector
- com.livecode.pi.number - a single-line field with increment/decrement twiddle
- com.livecode.pi.pattern - a pattern selector
- com.livecode.pi.set - a field with multi-select list behavior
- com.livecode.pi.string - a single-line field
- com.livecode.pi.text - a multi-line field

There are also some bespoke editors for particular object properties:

- com.livecode.pi.customprops
- com.livecode.pi.datagrid
- com.livecode.pi.textalign
- com.livecode.pi.textstyle

It is our intention that ultimately a widget alone will be able to function as a property editor, however currently this feature is not available.

IDE stackfiles named with version. (8.0.0-dp-1)

When a binary stackfile is rewritten in the IDE for a new version, it should have a (major) version in the filename to prevent unwanted IDE merging between versions. This can also be used to ensure incompatible stacks are not loaded if present - the IDE will only load stacks with a version less than or equal to its version.

For example, from 8.0 onwards, revTools has filename /Toolset/revTools.8.rev.

Specific bug fixes (8.0.0-dp-5)

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

15926 **Add ability in the script editor variables pane to filter list of variables and show/hide global/environment variables**
15893 **Message box emptied when closing and opening**
15882 **Suppress messages in IDE can't be turned off**
15854 **Stop Editing Group menu item doesn't work in LC8**
15847 **Datagrid PI needs min width**
15843 **Right click context menu stops working after Send Message from Card context menu**
15842 **Lock Location group property cannot be set or disabled from context menu**
15841 **GUI preferences not changing sort order of objects in project browser**
15838 **Changing size of dictionary headers crashes LiveCode**
15832 **Single-line message box doesn't like ';'**
15827 **Default iPhone 6 Plus screen should be 414 X 736**
15824 **Dictionary: Script error**
15801 **The tool palette was empty (except for the run/design choices)**
15794 **Tab panel icon in the tools palette is cropped**
15791 **typo in dialog title on opening an extension file**
15770 **Can't place group on card from menubar**
15769 **Add new card option to stack contextual menu**
15759 **Autocomplete does not recognise property names and sentences starting with "the"**
15757 **Message box does not allow editing of msg variable**
15755 **Building a standalone with "Answer Dialog" checked in standalone settings causes error at launch**
15745 **PI can open up (close to) off screen**
15744 **mainStack menu in PI only lists open stacks**
15723 **Incorrect wording in Business edition activation screen**
14465 **Rulers don't stay fixed to the stack when moving the stack**
14224 **Combo box will not allow typing into an "Ask" dialog.**
13709 **some message box output in LC when aligning text using menu Text->Align->...**
13646 **graphic bug in standalone application settings window**
11872 **impossible to change the case from all caps etc**
3962 **Delete key completely clear Ask dialog**

Specific bug fixes (8.0.0-dp-4)

15803 Open file doesn't recognise .mc as valid livecode stack extension
15796 Selecting a checkbox adds a border
15795 Progress bar icon missing from tools palette
15793 Tools palette has incorrect title
15787 "widget builder in plugins" is actually "extension builder" under "Tools"
15786 Datagrid PI script error
15776 Tools should be ordered as they are in previous releases
15739 Inconsistent letter case in the my_livecode folder

Specific bug fixes (8.0.0-dp-3)

15601	traversalOn property missing from widget property inspector
15600	Array and enum return values not displayed correctly in dictionary
15542	Typo in stack property inspector in LC 8.0
15464	Browse tool selected when launching LC8
15430	PI doesn't show foregroundColor for legacy graphic control
15427	Cut
15323	Style property should be an enum
15293	Behavior property inspector control should have way to edit behavior script or open stack/card that has behavior
15285	IDE: Property inspector string value should change property when clicking outside the field.
15247	Default Field name should be "Field"
15235	Simulators not listed in Development > Test Target menu
15227	card 'Single Line' Message Box script doesn't pass openCard and resizestack
15220	Widgets Tab of extension manager is empty when reopening
15216	IDE: Infinite loop when resolving load order
15180	Can't put values of debug variables from the message box
15001	Extension Builder: need to set the hideConsoleWindows to true before executing shell commands
14971	Debugger Break Point not met on right click "Send Card/Stack message"
14890	Control icons missing from tools palette
14873	Close and remove from memory does nothing from File menu
14852	PI color editor doesn't react well to colors with alpha value
14831	Open script-only stacks in script editor when they are opened
14822	Edited status of stack not being set
14738	Inspect menu missing from property inspector
14561	Widgets are not ordered in the tools palette or extension manager

Specific bug fixes (8.0.0-dp-1)

14627	
13475	in the openstack handler dispatching a mouseUp to a btn does not work correctly
13447	Project Browser control layer display
13417	IDE systemVersion comparison no longer works with Yosemite
13398	Sample - Book Library.livecode edit and delete features broken
13362	Script editor opens revmenubar script when no other stack is open
13343	Cannot install Android standalone on some devices
13215	Can't type in output field of message box
13191	FIX: flip graphic horizontally and vertically for complex graphics
13159	Palettes not observing decorations under certain circumstances
12880	File->Exit should be File->Quit
11755	flip graphic gives erroneous results with complex graphics

LiveCode Builder changes

LiveCode Builder Tools

lc-compile

Warnings

- A new warning has been added for identifiers that may conflict with syntax keywords.
- `metadata` definitions that occur before module imports no longer

trigger a warning.

Command-line interface

- A new `--verbose` command line flag has been added. If it is specified, **lc-compile** will output additional debugging information.

Compiler generates an error if integer literal too big

The compiler will generate an error if an integer literal is too big to fit into the (current) unsigned 32-bit integer representation.

LiveCode Builder Language

Core types

- The following deprecated core type names have been removed:
    - `boolean` (replaced by `Boolean`)
    - `integer` (replaced by `Integer`)
    - `real` (replaced by `Real`)
    - `number` (replaced by `Number`)
    - `string` (replaced by `String`)
    - `data` (replaced by `Data`)
    - `array` (replaced by `Array`)
    - `list` (replaced by `List`)

Sort using arbitrary comparison handler

The ability to sort a list using an arbitrary comparison handler has been added. The syntax is

sort <List> using handler <Handler>

A public handler type SortCompare has been added to the sort module.
The handler used for sort comparison must be of type SortCompare, i.e. be of the form

MyComparisonHandler(in pLeft as any, in pRight as any) returns Integer

Identifiers

- Identifiers are now expected to match `[A-Z0-9_.]`.

Syntax

- Syntax keywords are no longer permitted to match `[A-Z0-9_.]`.
- `metadata` definitions may now occur anywhere a module's top-level

context.

- `use` declarations may now occur anywhere in a module's top-level

context.

Change to handler return type syntax.

- The syntax for declaring the return type for a handler, or handler type has been changed to `[ 'returns' 'nothing' | 'returns' <Type> ]`.
- The previous syntax `as <Type>` or `as undefined` will continue to be supported until dp-4 at which point it will be removed along with the `undefined` type keyword.
- The compiler will emit warnings for the use of the deprecated syntax.

Case-Sensitivity

- All identifiers are now case-insensitive - i.e. a handler Main can be called as mAin, MAIN and main.

Replace concept of 'undefined' with 'nothing'

- The use of the keyword 'undefined' is now deprecated, 'nothing' should be used instead.
    - Use 'returns nothing' to indicate 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', 'is not undefined' syntax is now deprecated, 'is' and 'is not' should be used with 'nothing' instead
    - Use `<expr> is nothing` and `<expr> is not nothing` to test whether an expression has a value or not
    - The phrase `<left> is <right>` will now return true if `<left>` and `<right>` are both nothing
    - The phrase `<left> is not <right>` will now return true if one of `<left>` or `<right>` are nothing (but not both).

Foreign handler definitions require explicit typing.

- A foreign handler definition must declare an explicit return type.
- Each parameter in a foreign handler definition must declare an explicit type.

Foreign Handler Types

It is now possible declare foreign handler types:

```
foreign handler type MyCallback(in pContext as optional pointer, in pValue as any) as CBool
```

When used in the context of a foreign handler definition, a foreign handler type will cause automatic bridging of the LCB handler to a C function pointer which can be called directly by the native code.
The function pointers created in this fashion have lifetime equivalent to that of the calling context. In particular, for widgets they will last as long as the widget does, for all other module types they will last as long as the module is loaded.

IntSize Type

- There is now an IntSize foreign type, mapping ssize_t.

LiveCode Builder Host Library

Determining if a widget is enabled

- It is now possible to determine the enabled state of a widget from within its script.
    - The `my enabled` property returns true if the widget is currently enabled
    - The `my disabled` property returns true if the widget is currently disabled
    - If script changes the enabled (or disabled) property of the widget then an OnParentPropChanged message will be sent.

Ability to display a popup menu

- New syntax has been added to popup a menu constructed from a provided menu text.
    - `popup menu <MenuText> at <Point>`

Ability to access a widget's effective font

- The textFont, textSize and textStyle properties have been reserved to the host.
- New syntax `my font` has been added which returns a Canvas.Font matching the current effective values of the text properties that have been set on the widget.

Detecting successive clicks

- The OnClick event is sent every time a mouseDown/mouseUp sequence is detected by the engine on a widget.
- Use 'the click count' syntax to fetch the number of successive clicks which happened close together and within a certain time of each other.

Widget Printing

- Widgets now print along with other controls.
    - Widgets will be rasterized at screen resolution and then printed as an image.
    - Higher-fidelity printing of widgets will be implemented at a later date.

Composed widgets

The ability to compose widget objects has been added. Widgets can either be 'host' widgets, created when a widget is directly embedded in a stack, or 'child' widgets which are created when a widget is used as a child widget within another widget.

Syntax

A `Widget` type has been added, so that variables can contain references to child widget objects. A variable to hold a widget reference can be defined in the usual way, e.g.

variable tWidget as Widget

New widget syntax has been added to create, place, unplace and manipulate child widgets.

- a new widget <kind> - Creates a widget object of the specified kind.
- place <widget> [at (bottom|top) | (below|above) <other widget>] - Adds a child widget to the parent on the specified layer.
- unplace <widget> - Removes a child widget from the parent.
- the target - Returns the child widget that started the current execution.
- my children - Returns a list of the currently placed child widgets of this widget.
- property <property> of <widget> - Enables manipulation of a property implemented by a child widget.
- the rectangle of <widget> - Enables manipulation of the rectangle property of a child widget.
- the width of <widget> - Enables manipulation of the width property of a child widget.
- the height of <widget> - Enables manipulation of the height property of a child widget.
- the location of <widget> - Enables manipulation of the location property of a child widget.
- the enabled of <widget> - Enables manipulation of the enabled property of a child widget.
- the disabled of <widget> - Enables manipulation of the disabled property of a child widget.
- annotation <name> of <widget> - Enables tagging of child widgets with named values.

Events

Events triggered on child widgets (such as OnMouseUp) are automatically passed up to the parent, as long as the child's event handler returns nothing. If any event handler returns something, the event is considered handled and is not passed to the parent.

Messages

Messages posted by the child widget can be handled by the parent in an `On<message name>` handler. For example, if the child has the code

post "dataChanged" with [mDataArray],

this can be handled in the parent by adding

public handler OnDataChanged(in pArray as Array).

Posted messages can only be handled by a direct parent, and a widget's script object will only receive messages posted by host widget, i.e. the topmost parent.

Example

See https://github.com/runrev/livecode/blob/develop/extensions/widgets/simplecomposed/simplecomposed.lcb

for an example of how the host/child relationship can be used.

Native Code Access

LiveCode extensions can now contain native code libraries which LCB will use to resolve foreign handler references.
The foreign handler binding string should be of the form `libname>function` to use this feature. In this case, the engine will look for a library `libname` on a per-platform basis when the foreign handler needs to be resolved.
Native code libraries should be present inside the `resources` folder inside the extension archive. The engine derives the appropriate path from the requested library name and current platform. The structure is as follows:

```
<extension>/
    resources/
        code/
            mac/
                <library>.dylib
            linux-x86/
                <library>.so
            linux-x86_64/
                <library>.so
            win-x86/
                <library>.dll
```

Note: At present, only the desktop platforms are supported.
Note: The above structure is likely to change in a future release. In particular the `code` folder will sit at the same level as resources rather than within it.

LiveCode Builder Standard Library

Mathematical functions

- Several mathematical functions now throw "domain errors" when applied to values that the function is not defined for, including

`log10()`, `ln()`, `asin()` and `acos()`, and `x ^ y`.

Foreign function interface

- The following deprecated foreign type names have been removed from the `com.livecode.foreign` module:
    - `pointer` (replaced by `Pointer`)
    - `bool` (replaced by `CBool`)

    - `uint` (replaced by `UInt32` or `CUInt`)
    - `int` (replaced by `Int32` or `CInt`)
    - `float` (replaced by `Float32` or `CFloat`)
    - `double` (replaced by `Float64` or `CDouble`)
    - `NativeCString` (replaced by `ZStringNative`)

Sequence operations

- New syntax has been added for searching partial contents of sequence types (`List`, `String` and `Data`) based on the `offset` operation.
    - `the offset of <Needle> before <Position> in <Haystack>`
    - `the offset of <Needle> after <Position> in <Haystack>`
    - Equivalent syntax has been added for the `index` operation.

LiveCode Extension changes

Tree View Widget

Sorting Options

The tree view widget now has the ability to sort the keys of its arrayData in different ways.
Two properties have been added to achieve this:

- sortOrder: either "ascending" or "descending"
- sortType: either "text" or "numeric".

The default sort order of the widget is ascending numeric.

Header Widget

Documentation

The messages posted by the header bar widget have been documented.

Bugs fixed

[15815] headerAction, leftAction and searchAction messages not documented.

Navigation Bar Widget

Widget Theme

The theme of the navigation bar widget can now be set to iOS, Android(Dark) or Android(Light).
The following property has been added:

- widgetTheme: either "iOS", "Android(Dark)" or "Android(Light)"

The following property has been modified:

- opaqueBackground --> backgroundOpacity: either "Opaque", "Translucent" or "Transparent"

The default theme is iOS and the default background is opaque.

JSON Library

JSON Library Added

An LCB library, **com.livecode.library.json**, has been written to provide support for generating and parsing JavaScript Object Notation (JSON) data. See also <http://json.org>.

Functions

The library has two public handlers, `JsonImport` and `JsonExport`. `JsonImport` takes a string containing JSON-formatted text and parses it into a LiveCode value. `JsonExport` takes a LiveCode value and returns the equivalent value as a string in JSON format.

Using the library

The library is automatically loaded into the IDE, and the `JsonImport` and `JsonExport` handlers placed in the message path where they are available to call from any object.
In LiveCode Script, `JsonExport` takes any value and converts it to a string representing a JSON encoded value.
To use the library from a LiveCode Builder widget or library, simply add it to the list of use clauses:

```
use com.livecode.library.json
```

When using the `JsonExport` handler in LCB, an error is thrown if the value passed is not of one of the following types:
- String
- Number
- List
- Array
- Boolean
- nothing

Examples

From LiveCode Script:

```
local tData, tJSON
put "a,b,c,d" into tData
split tData by comma
put JsonExport(tData) into tJSON -- contains {"1": "a","2": "b","3":
"c","4": "d"}
```

From LiveCode Builder:

```
variable tJSON as String
put "[1,1,2,3,5,8]" into tJSON
variable tData as List
put JsonImport(tJSON) into tData -- contains [1,1,2,3,5,8]
```

Color Swatch Widget

Bugs fixed

[15851] error getting swatchColor when it is empty

Dictionary additions

- **create widget** (*command*) has been added to the dictionary.
- **export widget** (*command*) has been added to the dictionary.
- **import widget** (*command*) has been added to the dictionary.
- **load extension** (*command*) has been added to the dictionary.
- **popup** (*command*) has been added to the dictionary.
- **unload extension** (*command*) has been added to the dictionary.
- **loadedExtensions** (*function*) has been added to the dictionary.
- **newWidget** (*message*) has been added to the dictionary.
- **widget** (*object*) has been added to the dictionary.
- **is not really** (*operator*) has been added to the dictionary.
- **is really** (*operator*) has been added to the dictionary.
- **kind** (*property*) has been added to the dictionary.
- **scriptOnly** (*property*) has been added to the dictionary.

Dictionary changes

- The entry for **accept** (*command*) has been updated.
- The entry for **close socket** (*command*) has been updated.
- The entry for **create stack** (*command*) has been updated.
- The entry for **export with palette** (*command*) has been updated.
- The entry for **get** (*command*) has been updated.
- The entry for **launch url** (*command*) has been updated.
- The entry for **mobileAddContact** (*command*) has been updated.
- The entry for **mobileComposeHtmlMail** (*command*) has been updated.
- The entry for **mobileComposeMail** (*command*) has been updated.
- The entry for **mobileComposeUnicodeMail** (*command*) has been updated.
- The entry for **mobileUpdateContact** (*command*) has been updated.
- The entry for **open socket** (*command*) has been updated.
- The entry for **play video** (*command*) has been updated.
- The entry for **print link** (*command*) has been updated.
- The entry for **put cookie** (*command*) has been updated.
- The entry for **read from socket** (*command*) has been updated.
- The entry for **resolve image** (*command*) has been updated.
- The entry for **revBrowserAddJavaScriptHandler** (*function*) has been updated.
- The entry for **revBrowserNavigate** (*command*) has been updated.
- The entry for **revBrowserSet** (*command*) has been updated.
- The entry for **revMail** (*command*) has been updated.
- The entry for **revXMLRPC_SetHost** (*command*) has been updated.
- The entry for **secure socket** (*command*) has been updated.
- The entry for **write to socket** (*command*) has been updated.
- The entry for **clickLoc** (*function*) has been updated.
- The entry for **clickV** (*function*) has been updated.
- The entry for **commandArguments** (*function*) has been updated.
- The entry for **commandName** (*function*) has been updated.
- The entry for **mobileGetLaunchData** (*function*) has been updated.
- The entry for **mobileStorePurchaseError** (*function*) has been updated.

- The entry for **mouseH** (*function*) has been updated.
- The entry for **mouseV** (*function*) has been updated.
- The entry for **openSockets** (*function*) has been updated.
- The entry for **revBrowserOpen** (*function*) has been updated.
- The entry for **revBrowserOpenCef** (*function*) has been updated.
- The entry for **specialFolderPath** (*function*) has been updated.
- The entry for **tan** (*function*) has been updated.
- The entry for **time** (*function*) has been updated.
- The entry for **URL** (*keyword*) has been updated.
- The entry for **codepoint** (*keyword*) has been updated.
- The entry for **codepoints** (*keyword*) has been updated.
- The entry for **dateItems** (*keyword*) has been updated.
- The entry for **trueWord** (*keyword*) has been updated.
- The entry for **trueWords** (*keyword*) has been updated.
- The entry for **purchaseStateUpdate** (*message*) has been updated.
- The entry for **remoteControlReceived** (*function*) has been updated.
- The entry for **socketClosed** (*message*) has been updated.
- The entry for **socketError** (*message*) has been updated.
- The entry for **socketTimeout** (*message*) has been updated.
- The entry for **AndroidBrowser** (*object*) has been updated.
- The entry for **iosBrowser** (*object*) has been updated.
- The entry for **HTMLText** (*property*) has been updated.
- The entry for **activatePalettes** (*property*) has been updated.
- The entry for **clipboardData** (*property*) has been updated.
- The entry for **innerGlow** (*property*) has been updated.
- The entry for **innerGlow** (*property*) has been updated.
- The entry for **password** (*property*) has been updated.
- The entry for **serialControlString** (*property*) has been updated.
- The entry for **socketTimeoutInterval** (*property*) has been updated.
- The entry for **sslCertificates** (*property*) has been updated.
- The entry for **sslCertificates** (*property*) has been updated.
- The entry for **stackFiles** (*property*) has been updated.

Previous Release Notes

7.0.6 Release Notes	http://downloads.livecode.com/livecode/7_0_6/LiveCodeNotes-7_0_6.pdf
7.0.4 Release Notes	http://downloads.livecode.com/livecode/7_0_4/LiveCodeNotes-7_0_4.pdf
7.0.3 Release Notes	http://downloads.livecode.com/livecode/7_0_3/LiveCodeNotes-7_0_3.pdf
7.0.1 Release Notes	http://downloads.livecode.com/livecode/7_0_1/LiveCodeNotes-7_0_1.pdf
7.0.0 Release Notes	http://downloads.livecode.com/livecode/7_0_0/LiveCodeNotes-7_0_0.pdf
6.7.6 Release Notes	http://downloads.livecode.com/livecode/6_7_6/LiveCodeNotes-6_7_6.pdf
6.7.4 Release Notes	http://downloads.livecode.com/livecode/6_7_4/LiveCodeNotes-6_7_4.pdf
6.7.2 Release Notes	http://downloads.livecode.com/livecode/6_7_2/LiveCodeNotes-6_7_2.pdf
6.7.1 Release Notes	http://downloads.livecode.com/livecode/6_7_1/LiveCodeNotes-6_7_1.pdf
6.7.0 Release Notes	http://downloads.livecode.com/livecode/6_7_0/LiveCodeNotes-6_7_0.pdf
6.6.2 Release Notes	http://downloads.livecode.com/livecode/6_6_2/LiveCodeNotes-6_6_2.pdf
6.6.1 Release Notes	http://downloads.livecode.com/livecode/6_6_1/LiveCodeNotes-6_6_1.pdf
6.6.0 Release Notes	http://downloads.livecode.com/livecode/6_6_0/LiveCodeNotes-6_6_0.pdf
6.5.2 Release Notes	http://downloads.livecode.com/livecode/6_5_2/LiveCodeNotes-6_5_2.pdf
6.5.1 Release Notes	http://downloads.livecode.com/livecode/6_5_1/LiveCodeNotes-6_5_1.pdf
6.5.0 Release Notes	http://downloads.livecode.com/livecode/6_5_0/LiveCodeNotes-6_5_0.pdf
6.1.3 Release Notes	http://downloads.livecode.com/livecode/6_1_3/LiveCodeNotes-6_1_3.pdf
6.1.2 Release Notes	http://downloads.livecode.com/livecode/6_1_2/LiveCodeNotes-6_1_2.pdf
6.1.1 Release Notes	http://downloads.livecode.com/livecode/6_1_1/LiveCodeNotes-6_1_1.pdf
6.1.0 Release Notes	http://downloads.livecode.com/livecode/6_1_0/LiveCodeNotes-6_1_0.pdf