LiveCode 5.0.1 Release Notes

Table of Contents

Overview

LiveCode 5.0.1 is an update to 5.0, introducing the OpenGL compositor for Android and several bug fixes.

This document describes all the changes that have been made – including bug fixes and new syntax.

For information about improvements made to the iOS port of the engine, please see the iOS Release Notes PDF.

For information about improvements made to the Android port of the engine, please see the Android Release Notes PDF.

At the time of writing, this information has yet to been integrated into the dictionary or User's Guide.

Known issues

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

Platform support

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

Windows

The engine supports the following Windows OSes:

- Windows 2000 SP4
- 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

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

Linux

The linux engine requires the following:

- 32-bit installation, or a 64-bit linux distribution that has a 32-bit compatibility layer
- 2.4.x or later kernel
- X11R5 capable Xserver running locally on a 24-bit display
- glibc 2.3.2 or later

- gtk/gdk/glib (optional – required for native theme support)
- pango/xft (optional – required for pdf printing, anti-aliased text and unicode font support)
- lcms (optional – required for color profile support in JPEGs and PNGs)
- gksu (optional – required for elevate process support)

Note: *The optional requirements (except for gksu and lcms) are also required by Firefox and Chrome, so if your linux distribution runs one of those, it will run the engine.*

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

Note: *LiveCode and standalones it builds may work on remote Xservers and in other bit-depths, however this mode of operation is not currently supported.*

Mac

The Mac engine supports:

- 10.4.11 (Tiger) on Intel and PowerPC
- 10.5.8 and later (Leopard) on Intel and PowerPC
- 10.6.x (Snow Leopard) on Intel
- 10.7.x (Lion) on Intel

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

Proposed changes

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

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

Setup

Installation

The structure of the IDE install has changed significantly in this release.

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 5.0.1
- Linux: /opt/runrev/livecode-5.0.1
- Mac: /Applications/ LiveCode 5.0.1.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 5.0.1
- Linux: ~/.runrev/components/livecode-5.0.1
- Mac: ~/Applications/ LiveCode 5.0.1.app

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

Uninstallation

On Windows, the installer hooks into the standard Windows uninstall mechanism. This is accessible from the appropriate pane in the control panel.

On Mac, simply drag the app bundle to the Trash.

On Linux, the situation is currently less than ideal:

- open a terminal
- *cd* to the folder containing your rev install. e.g.
  ```
  cd /opt/runrev/livecode-5.0.1
  ```
- execute the *.setup.x86* file. i.e.
  ```
  ./.setup.x86
  ```
- follow the on-screen instructions.

Reporting installer issues

If you find that the installer fails to work for you then please file a bug report in the RQCC or email support@runrev.com so we can look into the problem.

In the case of failed install it is vitally important that you include the following information:

- Your platform and operating system version
- The location of your home/user folder
- The type of user account you are using (guest, restricted, admin etc.)
- The installer log file located as follows:
    - **Windows 2000/XP**: <documents and settings folder>/<user>/Local Settings/
    - **Windows Vista/7**: <users folder>/<user>/AppData/Local/RunRev/Logs
    - **Linux**: <home>/.runrev/logs
    - **Mac**: <home>/Library/Application Support/Logs/RunRev

Activation

The license system has been replaced in this release.

The new 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 now 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 now possible to activate an installation of LiveCode for all-users of that machine by using the command-line. When invoked in this fashion, no GUI will be displayed, activation being controlled by any arguments passed.

On both platforms, the command is of the form:

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

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

This action can be undone using the following command:

<exe> deactivate

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

In what follows <livecodeexe> should be replaced with the path to the installed LiveCode executable or app that has been previously installed.

On Windows, you need to do:

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

start /wait <livecodeexe> deactivate

On Mac, you need to do:

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

“<livecodeexe>/Contents/MacOS/LiveCode” deactivate

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

Engine Changes

Pixel Perfect Intersect (5.0.0)

The **intersect** function has been updated to take an extra optional parameter:

intersect(object, object, [threshold])

The new method parameter can be one of the following:

- alpha value - An integer between 0 and 255 which specifies a threshold that the alpha value of each pixel must be greater than or equal to in order to be counted during calculation of the intersect.
- 'bounds': Specifies that the intersect should be calculated using the rects of the two objects. This is the default method.
- 'pixels': Specifies the intersect should be calculated using the visible pixels of the object.

Graphics Architecture Improvements (5.0.0 – Updated 5.0.1)

For 5.0.0, the graphics architecture has undergone a significant overhaul. In particular, the point at which the engine redraws has been cleaned up and a new accelerated mode has been introduced.

Redraw Changes

The method which the engine uses to track and apply redraws has been improved. If no **lock screen** is in effect, then any redraws will be flushed after each and every command has finished executing.

If **lock screen** is in effect, then redraws will be flushed after the screen is completely unlocked - either through a final **unlock screen** or when control returns to the event dispatch loop (which ever happens first).

Additionally, redraws will be flushed after engine-triggered updates (e.g. moves resulting from the move command, or gif animation).

Visual Effect Changes

If you wish to apply a visual effect at the point of an 'unlock screen' you must use:

lock screen for visual effect [in rect**]**

You must now tell the engine you intend to use a visual effect to display the changes.

When you **lock screen for visual effect** the engine renders an offscreen copy of the current defaultStack which it will use to transition from if/when a visual effect next occurs. Note that if you **lock screen for visual effect** and the screen is already locked, no effect will happen as no snapshot can be taken.

If you **lock screen for visual effect** you don't have to use a visual effect. The snapshot the engine takes will be discarded at the next redraw point.

The reason for this change is to allow the engine to use better/faster methods of rendering window content. In particular, it doesn't have to worry about using an approach that will allow it to fetch back window content at any point.

You can also specify an optional rectangle if you only want the visual effect to occur in a particular area of the screen.

Accelerated Rendering

A new 'accelerated rendering' feature is now available and can be used on a per-stack basis. This optional rendering approach caches the visual appearance of objects, reducing the act of rendering to compositing images together.

The accelerated rendering feature is controlled with the following stack properties:

- the compositorType
- the compositorCacheLimit
- the compositorTileSize

The compositor properties should be configured in **preOpenStack** or **preOpenCard** - they are not persistent (not saved into the stackfile).

compositorType (Updated 5.0.1)

set the compositorType of stack "my stack" **to** "none|software|coregraphics|opengl"

The **compositorType** determines what back-end to use for compositing the cached content. It can be one of the following:

- empty (or 'none') - do not use accelerated rendering (the default).
- 'software' - use the built-in engine blending functions to composite (the same the engine uses for inks)
- 'coregraphics' - use CoreGraphics to composite (Mac / iOS only)
- 'opengl' - use OpenGL to composite (iOS / Android only)

As of 5.0.1-dp-1 some optimizations have been made to the OpenGL compositor based on feedback using the OpenGL analyser in “Instruments”. As a result, some rendering glitches may have been introduced.

compositorCacheLimit

set the compositorCacheLimit of stack "my stack" **to** 16 * 1024 * 1024

The **compositorCacheLimit** determines the maximum number of bytes the engine should use to cache content for the stack. If the limit is not big enough to render the stack, accelerated rendering will not be used and the default rendering mode will be used.

compositorTileSize

set the compositorTileSize of stack "my stack" **to** 16|32|64|128|256

The **compositorTileSize** determines how big the fragments of object images it caches should be. It must be a power-of-two between 16 and 256 inclusive (i.e. 16, 32, 64, 128 or 256).

layerMode

The engine determines what to cache based on the value of the per-object **layerMode** property.

> **set the layerMode of group** "my group" **to** "static|dynamic|scrolling"

This property is (currently) one of:

- 'static' : the object will not be cached individually and only as part of the background whenever possible.
- 'dynamic' : the object will be cached independently from all others.
- 'scrolling': applies to groups only and is used where the group contains contents to scroll.

In general, any objects which do not move/resize/change or do so very rarely should be set to be 'static'; and any object which moves/changes/resizes frequently (i.e. every frame) should be set to 'dynamic'. In particular, the only cost involved in moving a dynamic object is that of re-compositing the scene - which is vastly cheaper (computationally) than redrawing.

The 'scrolling' **layerMode** can be used to good effect to cache scrolling group content. If a group is completely unadorned (no border, no scrollbars), then the engine will take the bounds of the group to be that of its contained objects and clip the rendering of those objects to the rect. In particular, it will be the unclipped content that is cached - thus eliminating redraw overhead for scrolling.

Notes

The accelerated rendering mode only looks at top-level objects to determine what to cache - objects within groups are always rendered as part of their (top-level) ancestor group.

Accelerated rendering comes with some restrictions:

- Bitmap effects which use the multiply and color dodge blend modes will not work correctly on top-level objects.
- Only 'blendSrcOver' and the image processing blend inks will work on top-level objects.
- 'opengl' mode does not support any inks apart from blendSrcOver on top-level objects.
- Using an ink other than 'blendSrcOver' on a top-level object will implicitly make it use 'dynamic' layerMode

In general, to ensure the fastest possible redraw speed, note the following:

- Minimize top-level objects. The engine processes all top-level objects each redraw, therefore if you have invisible objects and resources that are never directly displayed place them all in an invisible group on the card.
- Only change object properties when necessary. Changes to many object properties will result in the object requiring an update, therefore minimize such changes.
- Update all objects together. The screen is flushed after every command that caused an update to be requested therefore try to update all objects on a card simultaneously within a lock/unlock screen pair.
- Use opaque unadorned groups to aggregate objects. The engine is able to eliminate a good proportion of content that is displayed behind opaque unadorned groups, so if you have a number of controls which (together) cover a rectangular region, place them in an opaque

group.

The accelerated rendering feature has the following known issues:

- Creating/deleting/moving objects within dynamic unadorned groups (i.e. scrolling layers) will cause redraw problems. For now, make sure you turn **layerMode** to static while modifying the contents of such groups.
- The OpenGL compositor used on iOS is still quite inefficient this will be improved in a subsequent dp. For now, a bigger **compositorTileSize** generally improves performance significantly (32 is a reasonable size for non-Retain iPhones, a minimum of 64 is better for Retina iPhones and iPads).
- Inks that are not strictly supported by accelerated rendering will cause redraw artefacts. The engine will implicitly coerce such inks to blendSrcOver in a subsequent dp.
- The **layerMode** property is not currently persistent (it will not save into the stackfile).
- Top-level objects that inherit a **backPattern** from the card or stack will render incorrectly in 'dynamic' **layerMode** when moved.
- There are various rendering glitches when using accelerated rendering on Linux.

Important: If you turn on accelerated rendering and have no dynamic layers then you will likely see a drop in performance as then any changes will always be redirected through the cache which is more costly than just rendering the changes directly.

WindowShape Changes

On Mac, both sharp (1-bit masked) and soft (alpha masked) window shapes are now treated uniformly - by modifying the alpha-channel of the Window.

On Windows and Linux, they continue to be treated separately since those platforms do not necessarily support alpha blended window shapes or do so at significant cost.

Mac Menubar Scroll Improvements

A number of issues related to the 'automatic' stack scrolling performed when a Mac menubar is set have been resolved. These are 2271, 3396, 3643, 5268. In particular, stacks will always be saved with the correct (non scrolled) height and the long standing issue of standalones being incorrectly sized should no longer happen.

Deprecated Features

All the bitwise and arithmetic inks are now deprecated.

New applications should use only the porter-duff and imaging blend modes:

- blendClear
- blendSrc
- blendDst
- blendSrcOver
- blendDstOver
- blendSrcIn

- blendDstIn
- blendSrcOut
- blendDstOut
- blendSrcAtop
- blendDstAtop
- blendXor
- blendPlus
- blendMultiply
- blendScreen
- blendOverlay
- blendDarken
- blendLighten
- blendDodge
- blendBurn
- blendHardLight
- blendSoftLight
- blendDifference
- blendExclusion

Pattern Origin Updates

Prior to 5.0.0-dp-2, the origin of a pattern (where it starts tiling from) is the top left of the object from which the pattern is inherited. For example, the means that the origin of a pattern of a graphic inherited from the card will be the top left of the card, rather than the top left of the graphic.

AS of 5.0.0-dp-2, this functionality has been altered for objects with 'dynamic' **layerMode** (and for objects within a group with 'dynamic' **layerMode**). The origin of the patterns of such objects is now the top left of the object itself, irrespective of where the pattern is set.

Optimizing Button Redraws

It is often useful to use buttons to display images and sprites, using an unadorned button and setting its icon. Optimizations have been made in order to improve the rendering speed of such buttons. In this case, make sure the following properties have been set:

- opaque off
- showBorder off
- showName off
- shadow off
- hiliteBorder off
- armBorder off
- showFocusBorder off
- no hiliteIcon
- no hoverIcon

This will minimise the number of redraws of the button (buttons get redrawn when their state changes, with the above properties off, there are no implicit state changes that occur that require a redraw).

Known Issues

There are an assortment of known issues that are unrelated to the accelerated rendering mode. Theses issues will be addressed in upcoming dps.

- Switching from fullscreen movies on iOS causes redraw issues.
- Rotation effect on iOS causes redraw issues.
- Text can disappear sometimes on Linux when blended layers are displayed.
- Player objects on Mac will not display correctly when initially switching from selected to non-selected in pointer tool mode.

Mac – revCapture video capture library (4.6.1 – experimental)

The revCapture library is begin developed as a replacement for revVideograbber – utilising the most up to date APIs on each platform, and eliminating many issues the previous library had (including problems introduced by Apple with iSight video-capture in 10.6.5 onwards).

At this time, the revCapture library is Mac-only, and is integrated as part of the revVideograbber external; it will be moved into its own external and become cross-platform over time.

While largely similar in function to revVideograbber, the revCapture library does have a number of differences:

- There are no configuration dialogs – instead, available inputs and codecs can be queried and configured through script.
- Previewing and recording can be started and stopped independently.
- Previewing takes place direct into a (configurable) image object, allowing the preview to be rendered with effects, blending, inks and interleave with other objects.
- Only supports output to QuickTime MOV files with a limited collection of preconfigured codecs.

The available handlers are described in the following sections.

Note: *As it stands the revCapture library has only been tested with built-in iSight hardware, although there is no reason it shouldn't work with any hardware that QTKit supports.*

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.

Session management

All capture operations take place within a capture session, which is managed using:

revCaptureBeginSession

revCaptureEndSession

Call **revCaptureBeginSession** to initialize the capture session, allowing the rest of the capture library to be used. When the capture session is done with, call **revCaptureEndSession**. This will release any hardware, and cancel any recording and previewing operations.

Note: *Calling any of the revCapture functions without calling* ***revCaptureBeginSession*** *at some point before hand will cause a no session error to be thrown.*

Input configuration

Unlike revVideograbber, configuration of the inputs to use for video/audio capture is done entirely through script rather than invoking standard dialogs.

Querying available inputs

To query the currently available inputs capable of capturing audio or video data use:

revCaptureListAudioInputs()

revCaptureListVideoInputs()

These both return a newline-delimited list of the available inputs of the given type.

Configuring the audio input

To configure the audio input to capture from in the current session, use:

revCaptureGetAudioInput()

revCaptureSetAudioInput *audioInput*

Where *audioInput* is one of:

- the name of an input returned by **revCaptureListAudioInputs()**
- *default*, indicating the default audio input device on the system should be used
- *none*, indicating that no audio should be captured

If setting an audio input fails because the new device could not be accessed, a *could not open input* error is thrown. If the current session could not connect to the new device, a *could not connect to input* error is thrown.

Configuring the video input

To configure the video input to capture in the current session, use:

revCaptureGetVideoInput()

revCaptureSetVideoInput *videoInput*

Where *videoInput* is one of:

- the name of an input returned by **revCaptureListVideoInputs()**
- *default*, indicating the default video input device on the system should be used
- *none*, indicating that no video should be captured

If setting an audio input fails because the new device could not be accessed, a *could not open input* error is thrown. If the current session could not connect to the new device, a *could not connect to input* error is thrown.

Compression configuration

Similar to input configuration, choosing the codec to use when recording audio and video streams is done entirely through script.

Querying available codecs

To query the currently available codecs for audio or video recording use:

revCaptureListAudioCodecs()

revCaptureListVideoCodecs()

These both return a newline-delimited list of the available codecs of the given type.

Configuring the audio codec

To configure the audio codec to use when recording, use:

revCaptureGetAudioCodec()

revCaptureSetAudioCodec *audioCodec*

Where *audioCodec* is either *none* to indicate that no audio compression should be performed, or the name of a codec returned by **revCaptureListAudioCodecs()**.

If the given codec name is unrecognized, or is not suitable for audio compression then an *invalid codec for media type* error is thrown. If recording is currently happening, a *recording in progress* error is thrown.

Configuring the video codec

To configure the video codec to use when recording, use:

revCaptureGetVideoCodec()

revCaptureSetVideoCodec video*Codec*

Where *videoCodec* is either *none* to indicate that no video compression should be performed, or the name of a codec returned by **revCaptureListVideoCodecs()**.

If the given codec name is unrecognized, or is not suitable for video compression then an *invalid codec for media type* error is thrown. If recording is currently happening, a *recording in progress* error is thrown.

Preview management

Configuring the video preview image

In order to be able to see a video preview of a capture session, the long id of an image object to target needs to be provided. To manage this use:

revCaptureGetPreviewImage()

revCaptureSetPreviewImage *imageLongId*

Where *imageLongId* is the long id of an existing empty image object to use; or empty to turn off the preview image.

Setting the preview image will lock the corresponding object meaning it can't be resized or have it's content modified by anything apart from the capture session. If the image cannot be found, locked, or an offscreen buffer initialized a suitable error will be thrown.

Configuring the audio preview volume

To configure the volume of the audio preview of a capture session use:

revCaptureGetPreviewVolume()

revCaptureSetPreviewVolume *newVolume*

Where *newVolume* is an integer between 0 (no audio preview) and 100 (audio preview at maximum volume).

Previewing a session

To start or stop a preview from running use:

revCaptureStartPreviewing

revCaptureStopPreviewing

The status of the preview is independent to that of recording, meaning that you can preview without recording, and record without previewing.

If a preview of the session could not be initialized when calling **revCaptureStartPreviewing** a *could not connect to output* error is thrown.

Note: *Starting to record may have an effect on the video preview's aspect and frame size as the video capture pipeline is optimized by the system based on the needs of the recorded output.*

Record management

Configuring the output file

To configure the filename to use when recording a capture session use:

revCaptureGetRecordOutput()

revCaptureSetRecordOutput *recordFilename*

Where *recordFilename* is an (absolute) path to the file to use for recording.

If recording is currently in progress, a *recording in progress* error is thrown.

Configuring the maximum frame size

To configure the maximum size of a frame in the recorded file use:

revCaptureGetRecordFrameSize()

revCaptureSetRecordFrameSize *maxWidth, maxHeight*

Where *maxWidth* and *maxHeight* specify the maximum size of a frame (in pixels) that should be generated in the (compressed) output. To use the default (optimal) frame size for the current settings, specify both width and height as 0.

Note: *These act as a hint to the pipeline, and the actual size of the frames in the output file may be smaller than this, or have a different aspect ratio.*

Configuring the maximum frame rate

To configure the maximum frame rate to aim for in the recorded file use:

> **revCaptureGetRecordFrameRate()**
>
> **revCaptureSetRecordFrameRate** *maxFrameRate*

Where *maxFrameRate* is the number of frames per second to aim for. To use the default (optimal) frame rate for the current settings, specify the rate as 0.

Note: *This acts as a hint to the pipeline as to the rate to aim for. The resulting frame rate in the recorded file may be less than, or more than this setting.*

Starting recording

To start recording the current session use:

> **revCaptureStartRecording**

This configures the session with previously selected record options and codecs, deletes the currently specified output file (if present) and starts the recording process. It has no effect if the session is already being recorded.

If recording starts successfully, **the result** is empty.

If an error occurs when the output file is being prepared for recording, **the result** will contain *recording failed*.

If the session could not be configured for recording, a *could not connect to output* error is thrown.

Stopping recording

To stop recording the current session use:

> **revCaptureStopRecording**

This finishes outputting any pending samples to the output file and stops recording. It has no effect if the session is not currently being recorded.

If recording finished successfully, **the result** is empty.

If an error occurs while trying to finish writing the output file, **the result** will contain *recording failed.*

XML text node access (4.6 – experimental)

Support has been added to some revXML functions for manipulating text nodes. Consider the following XML fragment:

```
<summary>
        Removes a <keyword>message</keyword> that was queued with the
<command>send</command> command and is waiting to be sent.
<summary>
```

This has the following structure as an XML tree:

```
ELEMENT(summary)
```

```
        TEXT(Removes a)
        ELEMENT(keyword)
            TEXT(message)
        TEXT( that was queued with the )
        ELEMENT(command)
            TEXT(send)
        TEXT( command and is waiting to be sent.)
```

Notice that the named XML nodes, are interspersed with (essentially unnamed) nodes containing text content. These (previously unaccessible) nodes can now be accessed (with some functions) by using a path of the form:

```
    <parent>/[<index>]
```

Here this references the <n>th text node under <parent>. For example, the above tree has the following accessible nodes:

```
    summary/[1]
    summary/keyword
    summary/keyword/[1]
    summary/[2]
    summary/command
    summary/command/[1]
    summary[3]
```

To access text content of a node simply use the *revXMLNodeContents* function with this extended path format. Note that the previous behavior is preserved – if you specify a node with a '/[n]' suffix, the text content of the node will be returned (if it is a text node). (i.e summary/keyword and summary/keyword/[1] are the same thing to *revXMLNodeContents*).

The following functions have been augmented with an additional (optional) parameter *incText*:

```
    revXMLFirstChild(docId, [ incText ])
    revXMLNextSibling(docId, nodePath, [ incText ])
    revXMLPrevSibling(docId, nodePath, [ incText ])
    revXMLChildNames(docId, nodePath, delimiter, filter, incCounts, [ incText ])
```

Here if *incText* is specified and is *true*, the functions will include text nodes in their processing.

For example, this allows you to loop over all nodes *including text nodes* using something like:

```
    local tCurrentNode
    put revXMLFirstChild(tDocId, tParentNode, true) into tCurrentNode
    repeat while tCurrentNode is not empty
        ... use tCurrentNode ...
        put revXMLNextSibling(tDocId, tCurrentNode, true) into tCurrentNode
    end repeat
```

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.

Ordered and unordered lists (4.6 – experimental)

Properties

Experimental support has been added to the field for simple, single level, ordered and unordered lists. To make a paragraph display as an element in a list use the *listStyle* property:

set the listStyle of line *lineIndex* **of** *field* **to** *style*

Where *style* is one of:

- *disc* – the paragraph is rendered as an element in an unordered list with the standard bullet character as marker (U+2022).
- *circle* – the paragraph is rendered as an element in an unordered list using the (unicode) character U+25E6 as marker.
- *square* – the paragraph is rendered as an element in an unordered list using the (unicode) character U+25AA as marker.
- *decimal* – the paragraph is rendered as an element in an ordered list, the label using standard (Arabic) decimal numerals. i.e. 1, 2, 3, 4, etc.
- *lower latin* – the paragraph is rendered as an element in an ordered list, the label using lowercase (Latin) letters. i.e. a, b, c, …, aa, ab, etc.
- *upper latin* – the paragraph is rendered as an element in an ordered list, the label using uppercase (Latin) letters. i.e. A, B, C, …, AA, AB, etc.
- *lower roman* – the paragraph is rendered as an element in an ordered list, the label using lowercase Roman numerals. i.e. i, ii, iii, iv, …, etc.
- *upper roman* – the paragraph is rendered as an element in an ordered list, the label using lowercase Roman numerals. i.e. I, II, III, IV, …, etc.

For both ordered and unordered list the marker is placed at the first tab-stop, and the content of the paragraph is placed (and wraps) at the second tab-stop.

For ordered lists the index of the item is determined by the number of preceding paragraphs with the **same** *listStyle* property.

Setting the *listStyle* property of a paragraph to empty causes it to revert to a normal (non-list item) paragraph.

htmlText format

Paragraphs with a non-empty *listStyle* present themselves in *htmlText* wrapped with <*LI*> tags. Sequences of such paragraphs with the same *listStyle* are bracketed by <UL> or <OL> tags. These tags take a *type* attribute, matching the (legacy) HTML attribute of the same name:

- *disc → disc*
- *circle → circle*
- *square → square*
- *1 → decimal*
- *a → lower latin*
- *A → upper latin*
- *i → lower roman*
- *I → upper roman*

For example:

1. Numbered Item 1
2. Numbered Item 2

- Bulleted Item 1
- Bulleted Item 2

```
<ol type='1' >
<li><p>Numbered Item 1</p></li>
<li><p>Numbered Item 2</p></li>
</ol>
<ul type='disc'>
<li><p>Bulleted Item 1</p></li>
<li><p>Bulleted Item 2</p></li>
</ul>
```

rtfText format

Paragraphs with a non-empty *listStyle* are appropriately marked in *rtfText* using both the (legacy) *pn* family of paragraph numbering tags and also with the new *listtable* tags.

By using both sets of tags a reasonable degree of interoperability is achieved with both TextEdit (and other Cocoa applications) on Mac, and Word and WordPad on Windows.

***Note:** Unfortunately, OpenOffice does not have particularly good rtf import / export capabilities (it doesn't even round-trip correctly through itself!) and thus copying / pasting of lists between LiveCode and OpenOffice will not work reliably or correctly.*

plainText format

Paragraphs with a non-empty *listStyle* are appropriately exported in plain text form when using the *plainText*, *unicodePlainText*, *formattedText* and *unicodeFormattedText* properties.

For example, the above example would be rendered (for plainText) as:

<tab>1.<tab>Numbered Item 1<return>

<tab>2.<tab>Numbered Item 2<return>

<tab>•<tab>Bulleted Item 1<return>

<tab>•<tab>Bulleted Item 2<return>

Persistence

As it stands the *listStyle* property does not save into the stack-file and will not do so until version 5.0 when the file format is revised (for various technical reasons, it is not possible to add this as a saveable property with the current stackfile format).

It is recommended that htmlText be used to save the content of fields in custom properties, for restoration on reload.

Important: This feature is currently highly experimental. This means that it may not be complete, fail in some circumstances that you would expect it to work, or change considerably before becoming final. Additionally, it might have problems or gotchas that make it significantly harder to use than other LiveCode features at this time.

Runtime execution stack configuration (4.5.1 – experimental)

In order to be able to more reliably control the maximum level of recursion, a new global property **stackLimit** has been introduced.

This property allows a script to set (in bytes) the maximum size of the (runtime) stack the engine uses for recursive computation. A change in the setting will only take effect when all currently executing handlers complete, and at this time the stack size limit will be reconfigured to the given limit, or the nearest amount to it depending on available memory.

The stackLimit currently in effect can be fetched using **the effective stackLimit**.

The recursionLimit property is now bounded by the stackLimit – attempts to set the recursionLimit greater than the stackLimit will see it downwardly adjusted to the maximum current size allowed.

Note: *The changes to the recursionLimit property and the new stackLimit property are only implemented on Windows at present.*

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.

revBrowser windowId property (4.5.1 – experimental)

There is a long standing issue with revBrowser that causes browser instances to be lost whenever the stack it is attached to has its window re-created. Previously, cases where this would occur had to be avoided when a browser was present on a stack.

To resolve this problem a new property has been added to browser instances – windowId. The windowId property allows the stack to which a browser instance is attached to be changed after it has been created.

If the windowId is set to 0, the browser instance is temporary hidden. If the windowId is set to a valid stack windowId, the browser instance will move to that stack.

For example, to toggle the resizable property of a stack hosting a browser use the following code:

```
revBrowserSet pBrowserId, "windowId", 0
set the resizable of stack pBrowserStack to pNewResizeableValue
revBrowserSet pBrowserId, "windowId", the windowId of stack pBrowserStack
```

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.

Improved image export (4.5)

Raw data export (experimental)

It is possible to export raw image data using the following forms:

> **export** *target* **as raw with palette** *colors*
>
> **export** *target* **as raw with (standard | optimized) palette**
>
> **export** *target* **as raw with** *count* **color optimized palette**
>
> **export** *target* **as raw [argb | bgra | abgr | rgab]**

The first three of these operate in the same way as for the other formats as described above except that instead of formatted image data you get the raw palette indices packed appropriately depending on the size of the palette:

> <= 2 colors will be 1 bpp
>
> <= 4 colors will be 2 bpp
>
> <= 16 colors will be 4 bpp
>
> <= 256 colors will be 8 bpp

The final form allows export of the full 32-bit data of the image with 8 bits per component. In this case, the components are not pre-multiplied with any alpha channel, and appear ordered in memory in increasing bytes.

e.g. The argb form will give you:

> byte 0 = alpha
>
> byte 1 = red
>
> byte 2 = green
>
> byte 3 = blue

Important: This feature (raw data export) 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.

Elevated process support (4.5 – experimental)

Sometimes it is necessary to perform operations on the local machine as an administrator, and a typical pattern for a GUI application doing this is for it to prompt for authentication at certain points.

Modern operating systems do not permit a process to elevate itself, nor grant itself increased privilege. Instead, they only allow a running process to launch another process with increased privilege. Therefore, in order to support this, a new form of the **open process** command has been introduced that can launch a slave process with elevated permissions:

> **open elevated process** *process* [**for [text | binary] (read | write | update | neither)]**

This form operates identically to the normal version, except that engine will ask the system to launch the given process with admin/root privileges.

The standard way for a GUI application that needs to perform privileged operations to be structured is to split the application into two parts: a GUI front-end that interacts with the user, and a command-line back-end that is run with elevated permissions. These two parts can then talk to each other using a standard master-slave approach, or some other form of IPC such as sockets.

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.

Status icon support (4.5 – experimental)

Windows, Linux and Mac all have an area where so-called 'status icons' can be displayed. On Windows this is the system tray on the bottom right of the start bar, on Linux this is typically the right of the panel at the top of the screen, and on Mac this is on the menubar.

The engine has support for adding a single status icon, and it can be configured using **the statusIcon**, **the statusIconTooltip** and **the statusIconMenu**:

> **set the statusIcon to** *imageId*
>
> **set the statusIconMenu to** *iconMenuSpec*
>
> **set the statusIconToolTip to** *toolTip*

Here *imageId* is the id of the image you wish to use as the icon. It will be scaled down automatically to the appropriate size for the platform and then set. The *toolTip* specifies what message appears when the user hovers over the status icon.

The *iconMenuSpec* allows you to configure a menu that will appear when the user does a 'menu' click on the icon. This string uses a subset of the standard engine menu specification:

> [<tab> *] ['('] <label> ['|' <tag>]

Here the number of tabs determines the depth of the menu (i.e. use this to create sub-menus). The optional tag is used when calling the **statusIconMenuPick** message.

Before the engine displays the status icon menu, it will send a **statusIconMenuOpening** menu to the current card of the defaultStack. You can use this opportunity to change the icon menu before it is displayed, this is an analog to handling *mouseDown* in a menu button.

When the user selects an item from the dock menu, the engine will send an **statusIconMenuPick** message to the current card of the default stack:

> **statusIconMenuPick** *which*

Here *which* will be a list of labels or tags (if specified) separated by '|' which determines which item was selected.

In addition, the engine will send the following message in response to clicks on the icon:

> **statusIconClick** *button*
>
> **statusIconDoubleClick** *button*

You can use these to perform an appropriate action.

***Note:** If you wish to display a menu from the status icon you must use the statusIconMenu property, attempting to open a normal popup menu in response to one of the click messages is not guaranteed to work.*

Note: *This syntax is only implemented on Windows at the moment and replaces the previously unsupported use of* ***the icon*** *and* ***the iconMenu*** *for this purpose. The properties specified above will have no effect on Mac and Linux at this time.*

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.

Listing sub-keys in the registry (4.5 – experimental)

To get a list of sub-keys in the Windows registry use the following function:

listRegistry(*parentKey*)

This will return a return-delimited list of sub-keys, i.e. those keys which are direct children of the given *parentKey*. The specified key should be in the same format as the other registry functions.

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.

HTTPS – automatic root certificate discovery (4.5 – experimental)

In previous versions it was necessary to set the *sslCertificates* property to the root certificates that HTTPS connections should be verified against. Support has now been added to locate and load the root certificates installed (and kept up to date) as part of the OS.

This uses the standard root certificate keychain on Mac, the standard root certificate store on Windows and uses a number of heuristics to locate this information on Linux.

You can easily find out if the system-installed root certificates are being found by running the following command in the message box:

get url "https://www.google.com"

put the result & **return** & **it**

If this results in an error about verification failure then it is likely that root certificates have not been found. Please let us know (particularly on Linux) if you find this simple test fails, making sure you give us full details of your system (e.g. Linux distribution and version).

Note: *Unfortunately this feature does not currently work correctly on Mac 10.6.x. For now, we advise including an appropriate root certificates collection with your application, as was previously necessary, and setting the* ***sslCertificates*** *property appropriately.*

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.

Mac – dock icon support (4.5 – experimental)

Previously unsupported syntax for manipulating the dock icon on Mac is now experimental.

Choosing an image

The current dock icon image can be set by using the global **icon** property:

set the icon to *imageId*

The engine will attempt to find an image with the given id, resize it to 128x128 and then set it as the dock icon for the application.

This property has no effect on other platforms.

Note: *The image is only guaranteed to persist while the application runs, although in some cases the OS does appear to cache it beyond this.*

Configuring the dock icon menu

In addition to changing the dock icon image, you can also configure the menu that appears when the user clicks on it.

To set the dock icon menu use the global **iconMenu** property:

set the iconMenu to *iconMenuSpec*

Here, *iconMenuSpec* is a string describing the menu. This uses a subset of the standard menu specification syntax. The string should be a return-delimited list of items specified as follows:

[<tab> *] ['('] <label> ['|' <tag>]

Here the number of tabs determines the depth of the menu (i.e. use this to create sub-menus). The optional tag is used when calling the **iconMenuPick** message.

Before the engine displays the icon menu, it will send a **iconMenuOpening** menu to the current card of the defaultStack. You can use this opportunity to change the icon menu before it is displayed, this is an analog to handling *mouseDown* in a menu button.

When the user selects an item from the dock menu, the engine will send an **iconMenuPick** message to the current card of the default stack:

iconMenuPick *which*

Here *which* will be a list of labels or tags (if specified) separated by '|' which determines which item was selected.

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.

Specific bug fixes (5.0.1)

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

	Focus not revoked when object hidden using 'hide ... with visual effect'.
9798	Optimized visual effect rendering on Mac in stacks with no windowshape (eliminates jerkiness introduced in 5.0).
9800	Redraw issues with QT players on Mac.
9808	New objects don't redraw correctly when created.
9810	Toggling 'show invisibles' fails to redraw bottom-most stack.

9813 Focus border is not included in redraw when hiding field.
9822 Menubar and visual effects don't work well together.
9824 Phantom images appear in stack on Mac desktop when running visual effects.
9826 Saving a stack in 4.6.4+ causes empty card names to become the empty string in 4.6.3 and before.
9843 Posting to HTTPS URLs fails.
9846 Showing with visual effect fails with front scripts
9852 Unable to set blendLevel of stacks with a windowShape that contains alpha.
9863 Crash displaying animated image from filesystem in field

Specific bug fixes (5.0.0)

Memory leak in scripts using intersect().
Crash when sprites overlap edges of the card on Windows intermittently in compositor mode.
Crash on Windows when calling flushEvents or wait and using compositor mode.
Unlock screen when screen not locked causes screen to become locked
Wrong color used to draw on Windows if new color has same red and blue when inverted as old color.
Pointer tool selection rectangle not drawn in accelerated rendering mode.
Ovals < 5 pixels in width or height not selectable with pointer tool.
Selection handles don't appear correctly in accelerated rendering mode.
Non-opaque constant color tiles render incorrectly in CoreGraphics mode due to using pre-multipled color.
Controls within a scrolling group can't be moved or resized
Removed size limit on scrolling groups
First core image effect doesn't work properly
Go stack in window doesn't play visual effects on mobile
Go stack (to different stack) doesn't play visual effects on mobile
Occasional graphic glitches in Linux when compositorType in use
Text in Linux not rendering correctly in some cases when in groups (which have blendlevel, ink set)
Updating invisible objects which layerMode dynamic/scrolling does not work correctly
Intersect gives incorrect result when button icon bounds is not the same as the bounds of the button.
2271 Building a standalone cuts off a stack
3396 Adding a menubar to a stack with transparent window shape has odd results
3643 Renaming the menubar results in positioning error when stack is re-opened
4198 Setting stack's windowShape resets stack's shadow.
5268 Wrong window size if stack is hidden when opened
9734 "command line" now returned in standalone -ui mode.
9739 Resizing image objects doesn't redraw correctly if on dynamic layer
9741 Setting inheritable props doesn't invalidate the viewport and cache
9744 Multiple stacked effects crash mobile
9762 Player width/height swapped on Mac
9769 Core image effects have inverted colors
9777 Changing image contents doesn't redraw the screen properly
9792 Make sure the whole of the window is updated when the window shape is changed.
9797 Twiddle controls don't hit-test correctly.

IDE Changes

Getting folder locations within the IDE

If you write plugins, or have code that relies on the location of IDE files then please ensure you use the following access functions to locate them:

revEnvironmentToolsPath()	The location containing the main IDE files.
revEnvironmentToolsetPath()	The location of the main IDE stacks.
revEnvironmentExternalsPath()	The location of the externals that come with the IDE.
revEnvironmentPluginsPath()	The location of the plugins that come with the IDE.
revEnvironmentRuntimePath()	The location of the standalones that come with the IDE.
revEnvironmentDocumentationPath()	The location of the documentation files.
revEnvironmentResourcesPath()	The location of the resources that come with the IDE.
revEnvironmentCustomizationPath()	The location of the IDE customization folder.
revEnvironmentUserCachePath()	The location of the folder to use for caching files.
revEnvironmentUserPreferencesPath()	The location of the folder to use for preference files.
revEnvironmentUserExternalsPath()	The location of the folder to use for additional externals.
revEnvironmentUserPluginsPath()	The location of the folder to use for additional plugins.
revEnvironmentUserResourcesPath()	The location of the folder to use for additional resources.

Important: Third-party IDE extensions must avoid placing any files inside the application bundle or under revEnvironmentToolsPath() (not least because you will probably not have privileges to do so!). Instead, they should use the user-externals and user-plugins paths as provided. These paths are determined by the user's customization path setting, configurable in the preferences.

Start Center Updates (5.0.0)

The start centre has been reorganized to link to the latest resources with the view to providing better assistance to those new to LiveCode.

Auto-Updater (5.0.0)

The auto-updater has returned in 5.0.0. On launch, if applicable, the updater will provide a list of updates that are available to the user based on the current version and license type. If selected, free updates will be downloaded and installed, paid updates will redirect the user to the appropriate account page.

Standalone builder

Windows – UAC Manifest

It is now possible to specify what action UAC should take on Windows Vista and higher when the standalone is launched. You can choose one of the following options:

Default	No UAC option is provided in the manifest.
Save as Invoker	The application will run with the same privileges as the process that invoked it.

Highest available	The application will be elevated to the highest privilege level the current user is allowed.
Require administrator	The application will be run as administrator after prompting the user for appropriate login/elevation rights.

Web

Using the 5.0.0 standalone builder to build for Web will create revlets. These are compatible with the existing revWeb plug-in available from revweb.runrev.com.

An updated version of the IDE for producing LiveCode Applets and the associated LiveCode player will be made available in due course.

Note: *As the currently available revWeb plug-in uses the 4.0 engine, you must be careful to only use features that are present in that version.*

Datagrid

The datagrid is currently at version 1.0.2 (build 14).

1.0.2 (build 12)	Added specific scrollbar width for linux platform.
	DeleteFieldEditorAndOpenNext now continues looking for a column to edit if EditValue is passed in the behavior script. This allows the developer to skip a column for editing. Previously this would only occur if EditValue was not handled.
	DeleteFieldEditorAndOpenNext now skips invisible columns.
	The control that DeleteFieldEditorAndOpenNext was targeting would sometimes be changed behind the scenes. This would cause CloseFieldEditor to be sent to the wrong control.
1.0.2 (build 13)	ResetList code is now in dgResetList. This allows the developer to intercept
	ResetList and manually call dgResetList while adding their own logic.
1.0.2 (build 14)	SortDataByKey specifically uses the first line of data in the column. Previously if there were multiple lines in a value it would cause problems.
	Fixed some issues in the auto scroll code used in drag/drop operations.
	When setting the "scrollbar width" property the new value was not being passed into the function that sets the scrollbar width (bug 9684).

Specific bug fixes (5.0.1)

(bug fixes specific to the current build are highlighted in bold)

Specific bug fixes (5.0.0)

9757	Behave like background property not working.

Resource and Documentation Changes

User Guide

No changes.

Dictionary

Dictionary entries have been created and updated for the appropriate new and updated features.

Mobile Examples

No changes.

Previous Release Notes

4.5.0 Release Notes	http://www.runrev.com/downloads/livecode/4_5_0/LiveCodeNotes-4_5_0.pdf
4.5.1 Release Notes	http://www.runrev.com/downloads/livecode/4_5_1/LiveCodeNotes-4_5_1.pdf
4.5.2 Release Notes	http://www.runrev.com/downloads/livecode/4_5_2/LiveCodeNotes-4_5_2.pdf
4.5.3 Release Notes	http://www.runrev.com/downloads/livecode/4_5_3/LiveCodeNotes-4_5_3.pdf
4.6.0 Release Notes	http://www.runrev.com/downloads/livecode/4_6_0/LiveCodeNotes-4_6_0.pdf
4.6.1 Release Notes	http://www.runrev.com/downloads/livecode/4_6_1/LiveCodeNotes-4_6_1.pdf
4.6.2 Release Notes	http://www.runrev.com/downloads/livecode/4_6_2/LiveCodeNotes-4_6_2.pdf
4.6.3 Release Notes	http://www.runrev.com/downloads/livecode/4_6_3/LiveCodeNotes-4_6_3.pdf
4.6.4 Release Notes	http://www.runrev.com/downloads/livecode/4_6_4/LiveCodeNotes-4_6_4.pdf
5.0.0 Release Notes	http://www.runrev.com/downloads/livecode/5_0_0/LiveCodeNotes-5_0_0.pdf

Revisions

Revision 1	MM	Document created for 5.0.0-dp-1.
Revision 2	MM	Updated engine bug fix list for 5.0.0-dp-2. Updated section "Status icon support" (bug 9733). Added section "Pixel Perfect Intersect". Added scrolling layerMode to "layerMode" section. Added section "Pattern Origin Updates"
Revision 3	MM	Updated engine bug fix list for 5.0.0-dp-3. Updated IDE bug fix list for 5.0.0-dp-3. Updated section "Pixel Perfect Intersect" Updated section "Visual Effect Changes" Added section "Auto Updater"
Revision 4	MM	Added section "Optimizing button redraws" Updated engine bug fix list for 5.0.0-dp-4.
Revision 5	MM	Updated engine bug fix list for 5.0.0-rc-1.
Revision 6	MM	Updated engine bug fix list for 5.0.0-rc-2.
Revision 7	MM	Updated engine bug fix list for 5.0.0-gm-1. Added section "Start Centre Updates".
Revision 8	MM	Updated section "Overview". Updated section "compositorType". Updated engine bug fix list for 5.0.1-dp-1. Added link to previous release notes.
Revision 9	MM	Updated engine bug fix list for 5.0.1-dp-2.
Revision 10	MM	Updated engine bug fix list for 5.0.1-dp-3.
Revision 11	MM	Updated engine bug fix list for 5.0.1-rc-1.
Revision 12	MM	No changes.
Revision 13	MM	No changes.