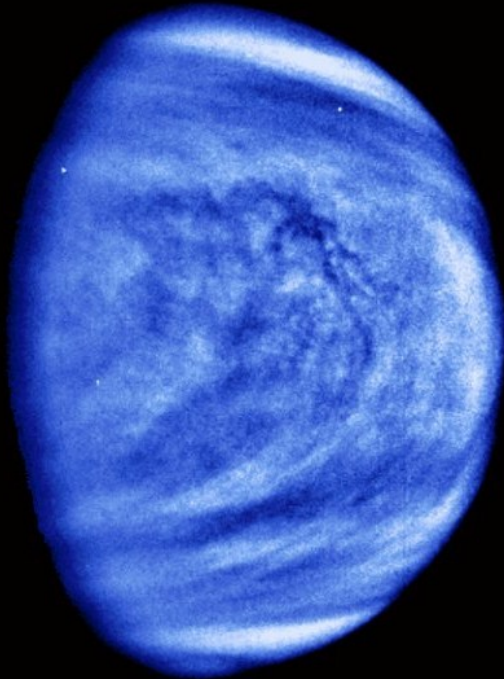


RunRevPlanet Carousel 3d

RunRevPlanet Carousel 3D adds an easy to use menu system to your desktop and mobile LiveCode applications. Users appreciate the immediate response and visual feedback. Add a popular contemporary menu style to your LiveCode applications for iOS, Linux, Mac or Windows.

Guide and Reference



RunRevPlanet Carousel 3d

Guide and Reference

Copyright (c) 2011 Scott McDonald PC Services
All rights reserved.

March 2011 Edition



Address: Scott McDonald PC Services
PO Box 139, Newtown, 2042
New South Wales, Australia
Facsimile: +612-9564-2347
Website: <http://www.runrevplanet.com>
Email: runrevplanet@smpcs.server101.com

Table of Contents

Introduction.....	3
Installation & Requirements.....	5
Conventions.....	6
Licensing Agreement.....	7
Making a Carousel Menu.....	9
Simple Carousel 3D Demo.....	13
Advanced Carousel 3D Demo.....	20
iOS Carousel 3D Demo.....	25
Carousel 3D Messages.....	28
Deploying Your Own Application.....	30
Handler Reference.....	32

About the Cover Image

The image on the cover of this RunRevPlanet Carousel 3D Guide and Reference is a public domain image from the NSSDC (National Space Science Data Center) Photo Gallery. Many thanks to NASA and the NSSDC for making this image, and many more images, available to the public.

This colorized picture of Venus was taken on February 14th, 1990, from a distance of almost 1.7 million miles, about 6 days after Galileo's closest approach to the planet. It has been colorized to a bluish hue to emphasize subtle contrasts in the cloud markings and to indicate that it was taken through a violet filter. Features in the sulfuric acid clouds near the top of the planet's atmosphere are most prominent in violet and ultraviolet light. The original image *PIA00072.jpg* can be found at:
http://nssdc.gsfc.nasa.gov/photo_gallery/photogallery-venus.html

About RunRevPlanet

The RunRevPlanet website is an initiative of Scott McDonald PC Services. Our goal is to make RunRevPlanet one of the top websites dedicated to LiveCode and an invaluable source of components, controls, tools and resources for LiveCode developers. Visit us at:
<http://www.runrevplanet.com>

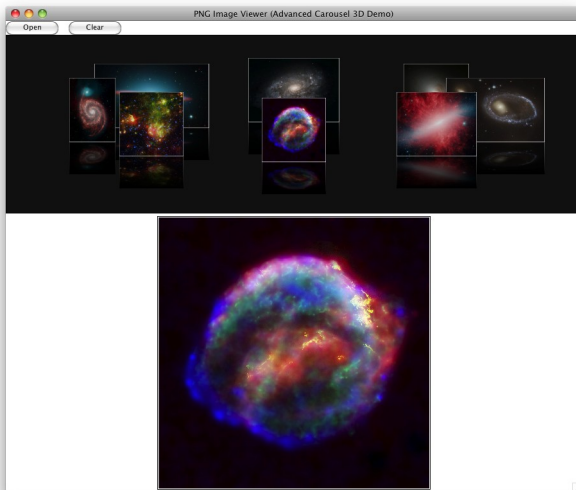
RunRevPlanet is not affiliated with RunRev Ltd.

Introduction

RunRevPlanet Carousel 3D is modern visual menu for your LiveCode applications in a style popular with contemporary users. The RRP Carousel 3D is written in 100% LiveCode script and does not use any platform specific libraries. This means you can add a modern menu to your cross platform application for iOS, Linux, Mac or Windows.

RRP Carousel 3D is simple to use and requires only a few lines of script to add attractive 3D menu to your LiveCode application. By using RRP Carousel 3D (RRPC3D) you can focus on the content of your application and not be concerned with the mechanics of making a modern menu on desktop and mobile platforms.

RRPC3D is easy to use, requiring only a few lines to create a fully working menu with a 3-D effects, smooth scrolling, an optional reflection and bounce animation effects, that your users will find fun and easy to use.



With a caching system, RRPC3D supports menus with unlimited numbers of entries. For example, RRPC3D, can be used to create a carousel style menu for viewing images in a folder. The number of images do not need to be known in advance and RRPC3D creates and loads the necessary menu items on demand.

RRPC3D can be used in image and document browsers, games, application front ends, any application where a simple to use visual menu with either mouse or touch based control will be familiar to your users and easy to use while looking attractive in modern.

Features of RRP Carousel 3D:

- Carousel menu with unlimited number of entries
- 3-D effect with smooth rotation animation
- LiveCode messages to indicate change of state
- Custom color background
- Optional menu item reflections, with glossy effect
- Control of animation speed
- Optional bounce animation on selection
- Supports touch interface on applicable platforms
- Does not require using front or back scripts
- Mobile platform support for iOS
- 100% LiveCode script



Installation & Requirements

To use RRPC3D you must have the following:

- Revolution 3.0, or newer, in the Studio or Enterprise edition.
- LiveCode 4.5, or newer.
- Using the iOS demo stack requires LiveCode 4.5.3, or newer.

RRPC3D may run with earlier versions of Revolution and LiveCode, but version 3.0 or higher is recommended. RRPC3D is distributed as a single ZIP archive containing these files:

```
rrpCarousel3D.rev  
rrpGrid.rev  
rrpPalette.rev  
rrpCarousel3D-Guide-Reference.pdf  
Readme.txt  
License.txt  
/Demos/Demo-Carousel3D-Advanced.rev  
/Demos/Demo-Carousel3D-iOS.rev  
/Demos/Demo-Carousel3D-Simple.rev  
/Demos/Images
```

Installation

To install RRPC3D, unzip the files above into a folder of your choice.

- * Do not unzip the files into your LiveCode program folder, instead put them into a convenient folder where you work in documents.

Conventions

The terms “rrpCarousel3D stack”, “rrpCarousel3D.rev” and “RRPC3D” may be used interchangeably in this guide depending on the context. Each of these names refers to RunRevPlanet Carousel3D.

The handlers in RRPC3D are all commands. Throughout this guide the term handler or command may be used interchangeably. In some, cases the commands are used like functions by using the result function.

The word LiveCode refers to the LiveCode IDE (Integrated Development Environment) and may be used interchangeably with the term IDE.

LiveCode script is shown in a fixed width font with the same formatting shown in the IDE Script Editor. An example of a script is shown below.

```
on rrpC3dSelectedM pID,pItemNumber
  lock screen
  call "rrpC3dSelectedItem 1125,pItemNumber" of ↵
    stack "rrpCarousel3D"
  put URL("binfile:" & the result) into image "Preview"
  set the pvWidth of image "Preview" to the width of image "Preview"
  set the pvHeight of image "Preview" to the height of ↵
    image "Preview"
  set the visible of image "Preview" to true
  ResizeImage
  unlock screen
end rrpC3dSelectedM
```

The ↵ symbol is used to indicate single lines of script that are too long to fit in the width here, and so are broken over two or more lines, but can be entered as a single line in the Script Editor. Scripts that are entered in the Message Box are also shown with a fixed width font but without any formatting such as below.

```
answer the cVersionNumber of stack "rrpCarousel3D"
```

- 🔍 A paragraph with this symbol highlights a point that could be unexpected behavior, or a potential problem that may not be obvious at first.

Licensing Agreement

The RRP Carousel 3D software and accompanying files and documentation are protected by Australian copyright law and also by international treaty provisions. Any use of this software in violation of copyright law or against the spirit of the terms of this agreement will be prosecuted.

RRP Carousel 3D is Copyright (c) 2011 Scott McDonald PC Services, all rights reserved.

Scott McDonald PC services authorizes you to make archival copies of the software for the sole purpose of backup and protecting your investment from loss. Under no circumstances may you copy the software and documentation for the purpose of distribution to others. Under no conditions may you remove the copyright notices from the software or documentation.

You may use an unlicensed copy of the RRP Carousel 3D to evaluate the RRP Carousel 3D for an unlimited time. You may not build or distribute a standalone application that uses the RRP Carousel 3D without first purchasing a License Key from Scott McDonald PC Services at the RunRevPlanet website, or without first purchasing an Unlock Code from an approved vendor.

You may distribute, without run-time fees or further licenses, your own executable applications based on the RRP Carousel 3D and the demonstration files after you have purchased a License Key or Unlock Code. You may not distribute applications that use the RRP Carousel 3D with your License Key or Unlock Code in an unencrypted stack file.

When distributing your own executable applications based on the RRP Carousel 3D you must encrypt with a secure password any stack that includes your License Name and Key or Unlock Code. You may not distribute your License Key or Unlock Code in a separate file with your application, or through other media such as Internet, email or print.

The previous restrictions do not prohibit you from distributing your own source code or stacks that depend on the RRP Carousel 3D. However others who receive your scripts need to download their own copy of the RRP Carousel 3D and purchase a License Key or Unlock Code in order to write programs that use your own code.

The RRP Carousel 3D may be used by one person on as many computer systems that person uses. We expect that group programming projects making use of the software will purchase a license for each member of the group. In such cases,

volume discounts may apply to site licensing agreements.

The RRP Carousel 3D will perform substantially in accordance with the accompanying documentation, and technical support by Scott McDonald PC Services will make commercially reasonable efforts to solve any problems associated with the RRP Carousel 3D. If you encounter a bug or deficiency, we will require a problem report detailed enough to allow us to find and fix the problem.

In no event will Scott McDonald PC Services or anyone else who has been involved in the creation, development, production, or delivery of the RRP Carousel 3D be liable for any direct, incidental or consequential damages, such as, but not limited to, loss of anticipated profits, benefits, use, or data resulting from the use of this software, or arising out of any breach of warranty.

By using this software you agree to the terms of this Licensing Agreement. If you do not agree, you should immediately erase all your copies of the RRP Carousel 3D and apply for a refund if you have purchased a Licensing Key or Unlock Code. Scott McDonald PC Services provides web-based and email support for the RRP Carousel 3D on an "as available" basis at no extra charge. When you send an email to technical support we will try to answer your support question within 48 hours.

Built with LiveCode. Portions (c)2000-2011 RunRev Ltd, All Rights Reserved Worldwide. You should review the License Agreement in your copy of LiveCode when building your own applications with LiveCode and the RRP Carousel 3D.

All names of products and companies used in this Licensing Agreement, the RRP Carousel 3D, or the documentation may be trademarks of their corresponding owners. Their use in this Licensing Agreement is intended to be in compliance with the respective guidelines and licenses.

Making a Carousel Menu

To make a carousel you need the RunRevPlanet Palette included with RRPC3D and then add some LiveCode to your application. A detailed description of the steps for coding are in the next three chapters about the Demos, but this chapter is a brief overview that outlines the basics.

Creating a Carousel

As a visual control, the carousel is created from the RunRevPlanet Palette. And unlike some controls from RunRevPlanet that require coding to initialize them, with RRPC3D all initialization is done automatically when pasting the control onto a card.

To create a carousel, the RunRevPlanet Palette must be open and the Edit (Pointer) tool active in the IDE.

Select the carousel by clicking on the background of the RunRevPlanet Palette above the top left corner of the carousel icon and click and drag a selection rectangle around it. Then right click (command click) in the middle of the selected carousel icon and from the contextual menu choose the Copy command.

Then in your stack, right click (command click) and choose the Paste Objects command from the contextual menu to create the carousel. By switching to the Run (Browse) tool you can immediately test the carousel with the sample menu items.

Clearing the sample items

After the carousel is made the sample items must be deleted so you can add your own menu items. To do this, call the `rrpC3dClear` command from your application, or you can use this line in the Message Box.

```
call "rrpC3dClear <id>" to stack "rrpCarousel3D"
```

Where `<id>` is the short ID of the carousel frame, which is the LiveCode graphic control that draws the carousel. You can find out the short ID of the carousel frame by looking for the control named "rrpC3dFrame" in the Application Browser, or by checking the Property Inspector for this control.

Adding Items to the Carousel

Menu items are added to the carousel using one of these three commands.

```
call "rrpC3dAddItemFromID C3dID,myID" of stack "rrpCarousel3D"
```

```
call "rrpC3dAddItemFromName C3dID,myName" of stack "rrpCarousel3D"
```

```
call "rrpC3dAddItemFromPath C3dID,myPath" of stack "rrpCarousel3D"
```

Where in this and the following examples, C3dID is a variable set to the short ID of the carousel frame control. The second parameter in each case is a variable that holds either the short ID, name or file path of the image that represents the menu item being added.

Depending on your application the images for the carousel menu items could be external files, or images stored in your application stack so use the appropriate command from the above in each case.

After adding items to the carousel, you can display the menu items with the added items by calling rrpC3dDraw.

```
call "rrpC3dDraw C3dID" of stack "rrpCarousel3D"
```

Customising the Look and behavior

RRPC3D has many commands that set the look of the carousel. For the complete list, refer to the Handle Reference chapter. All commands that include “Set” in the name are related to the look and behavior of the carousel.

For example, the following command sets the size of the carousel. While you can change the size of the carousel with a click and drag on the carousel frame, sometimes you need to change the size with code, for example, in a resizeStack handler.

```
call "rrpC3dSetSize C3dID,<bounds>" of stack "rrpCarousel3D"
```

And the next command sets the size of the image used for the menu items in the carousel.

```
call "rrpC3dSetImageSize C3dID,<width>,<height>" of stack "rrpCarousel3D"
```

🔍 In these examples, the angle brackets, are used to represent constants or variables that contain appropriate values for the setting.

Another command that has an important effect (and often needs to be experimented with) is `rrpC3dStepCount`. This sets the number of intermediate steps used when the carousel rotates from one menu item to another with an animation.

```
call "rrpC3dSetStepCount C3dID,<count>" of stack "rrpCarousel3D"
```

Small values for the `StepCount` and the animation will appear to jump as the carousel rotates, but too large a `StepCount` and the animation will be slow. The appropriate value varies depending on the size of the carousel and the speed of the machine that your application runs on. An appropriate starting point for this value is between 8 and 64.

The shape of the ellipse which is the path of the items in the carousel is changed with the `rrpC3dSetShape` command. More details for this command is in the next chapter and the `Handler Reference` chapter.

```
call "rrpC3dSetShape C3dID,percentWidth,percentHeight, ↵  
circleAngle"of stack "rrpCarousel3D"
```

Lastly, after calling any commands that change the appearance and behavior of the carousel, the following command must be called to update the internal settings of `RRPC3D`, Otherwise the effect of the changes will be unpredictable and not as intended.

```
call "rrpC3dUpdateLook C3dID" of stack "rrpCarousel3D"
```

Responding to Messages

`RRPC3D` uses messages sent to the current card (and stack) to allow you to respond to changes in the carousel. The generation of these messages is done automatically by `RRPC3D` as the user clicks or touches to interact with the carousel.

To make `RRPC3D` sends messages you must call this command to turn on the message mode. This can be done from within your application, or from the `Message Box` when developing your application.

```
call "rrpC3dSetMessageMode C3dID,true" of stack "rrpCarousel3D"
```

Once message mode is turned on, you can declare handlers such as the following in your card or stack scripts to act on these messages.

```
on rrpC3dPickM pID, pItemNumber
  -- perform an action when user clicks on the front item
end rrpC3dPickM
```

In this example the rrpC3dPickM message is handled to perform some action when the user clicks on the front item of the carousel.

Simple Carousel 3D Demo

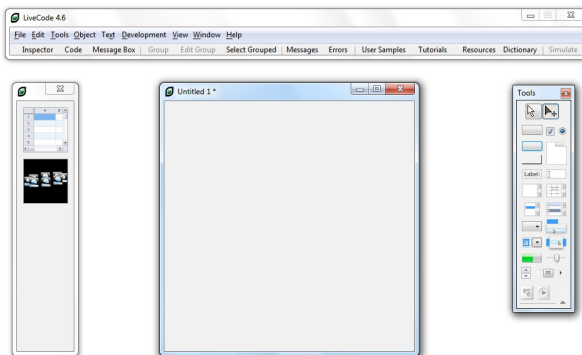
RRPC3D is a self-contained stack that you can use to add a modern and attractive 3D style menu to any LiveCode application. This chapter explains the process of adding a carousel menu to a simple stack. This demo is not a finished application, but shows the bare minimum to use RRPC3D. In the next chapter a more advanced demo that has a practical purpose is detailed.

Even though this demo is not a polished application, it is important to understand the basics of using RRPC3D before adding a carousel to your own applications. It only takes a few minutes to work through this demo, and there is a completed copy of the stack named Demo-Carousel-3D-Simple.rev in the Demos folder.

It is recommended that you carefully followed the steps the first time you use RRPC3D. Once you understand the process you are then ready to add carousels to your own applications.

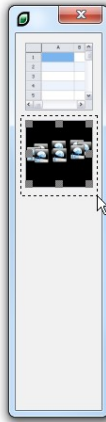
First you need an open stack to add the carousel to. Choose the New Mainstack command in the File menu. The carousel is a group of controls that you copy and paste onto your stack. To simplify this, the RunRevPlanet Palette is used as a source for the carousel control. Depending on the version of your RRPC3D, this Palette may include additional RunRevPlanet components.

Choose the Open Stack command in the File menu. The RunRevPlanet Palette stack is named rrpPalette.rev . For this tutorial it is assumed that both files rrpPalette.rev and the rrpCarousel3D.rev are in the same folder. Select rrpPalette.rev and choose the Open button.

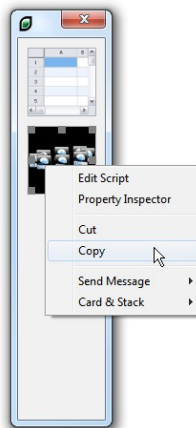


RRPC3D is the second object on the RunRevPlanet Palette. To copy the carousel, you must first select it. It is important for the Edit (Pointer) tool to be active to allow you to do this. First check that this is the case in the LiveCode Tools, or you can press control-0 (command-0) to select the Edit tool.

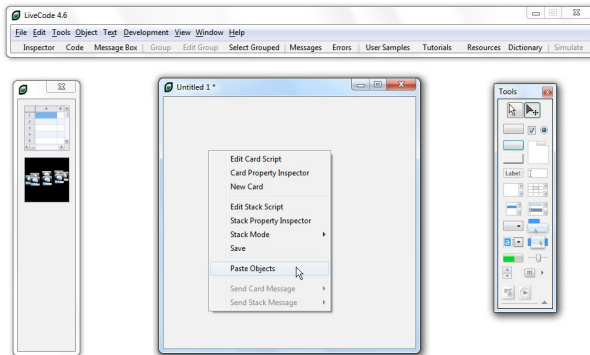
Select the carousel by clicking on the background of the RunRevPlanet Palette just above the top left corner of the carousel icon. Keep the mouse button pressed and click and drag to the bottom right corner of the carousel icon. A selection rectangle is drawn around the icon. When the selection handles appear you can release the mouse button.



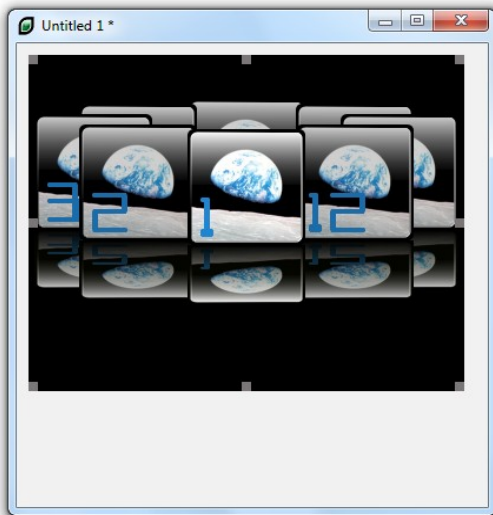
With the carousel icon selected you are ready to copy it to the clipboard. Right click (command click) in the middle of the selected icon and from the contextual menu choose the Copy command.



Then in the new empty main stack, right click and choose the Paste Objects command from the contextual menu.



This adds the carousel to your stack. The carousel is bounded by a rectangle which this documentation calls the carousel frame. This carousel frame is a LiveCode graphic control that holds all of the objects that make up the carousel.



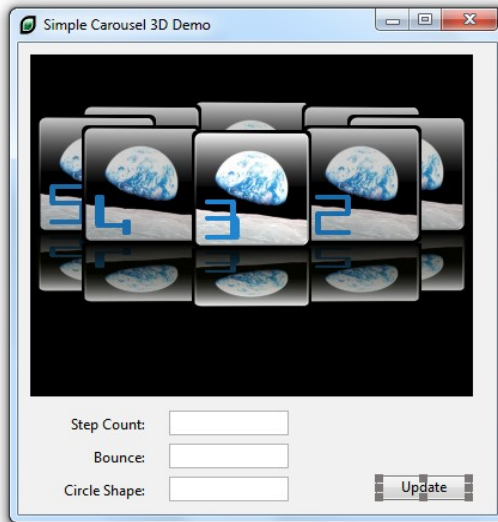
A new carousel is immediately filled with sample menu items to let you immediately test it. There are 12 items, of which only eight are visible at a time. By switching to be Run tool you can test the carousel by clicking on the items to watch the rotation animation.

- ☛ The speed of rotation is affected by several factors depending on the platform in use and the performance of the underlying hardware. If the rotation appears to slow or too fast this can be changed as explained later in this chapter.

The carousel frame allows you to move and resize the carousel. With the Edit tool selected, click inside the carousel frame (but not on a menu item) and selection handles appear so you can resize the frame. Similarly, the carousel is moved with a click and drag inside the selected carousel frame. When moving and resizing the frame, the carousel drawn in the new position after releasing the mouse.

Before proceeding, right click on a blank area of the new stack and choose the Property Inspector command in the contextual menu. Enter *My-Simple-Carousel-Demo* for the name and choose the Save command in the File menu. It is recommended that you navigate to the same folder as the *rrpPalette.rev* and *rrpCarousel3D.rev* files when saving.

Next, fields and a button are added to make it easy to change the appearance of the carousel. If the Run (browse) tool is selected, switch to the Edit (pointer) tool and drag three labels and three fields onto the stack below the carousel.



With the Property Inspector set the Content of the three labels to *Step count:*, *Bounce:* and *Circle shape:*. The three fields are then given appropriate names in the Property Inspector of *Count*, *Bounce* and *Shape*. Lastly add a button named *Update*.

The Update button is used in this demo to change the look of the carousel depending on the contents of the three fields. There are many other settings that can be changed to alter the look and behavior of the carousel, but these three are a starting point for this simple demo.

Right click (command click) on the Update button and from the contextual menu choose Edit Script. In the mouse up handler add the lines shown here:

```
on mouseUp
    local stepCount, bounce, percentWidth, percentHeight, circleAngle

    put field "Count" into stepCount
    put field "Bounce" into bounce
    put item 1 of field "Shape" into percentWidth
    put item 2 of field "Shape" into percentHeight
    put item 3 of field "Shape" into circleAngle

    -- 1003 is the short ID of the rrpC3dFrame graphic, in this case
    call "rrpC3dSetStepCount 1058,stepCount" of stack "rrpCarousel3D"
    call "rrpC3dSetBounce 1058,bounce" of stack "rrpCarousel3D"
    call "rrpC3dSetShape 1058,percentWidth,percentHeight, ↵
        circleAngle" of stack "rrpCarousel3D"

    call "rrpC3dUpdateLook 1058" of stack "rrpCarousel3D"
end mouseUp
```

Note how the call command is used. This is because RRPC3D does not need to be made a back script. In a typical application, the number of lines of LiveCode that refer to RRPC3D are small, and so it is not necessary to use up one of the slots available for putting a stack into the message path.

Here is a brief description of what the LiveCode does. Firstly the contents of the three fields are put into local variables. Then these variables are used in the RRPC3D commands.

The first command, rrpC3dSetStepCount sets the number of steps used in the animation between each menu item. The larger this number the smoother the animation, but the rotation is correspondingly slower. Depending on the type of images used for each menu item and the size of the carousel you will need to experiment with this number to get the best operation.

- ☛ The first parameter of `rrpC3dSetStepCount` and the other commands is the short ID number of the carousel frame. In this example it happens to be 1058, but the short ID will be different if you make your own demo. So you will need to check in the Property Inspector for the appropriate ID number when you add this code to your own copy of this stack.

The second command `rrpC3dSetBounce` sets whether there is a small “bounce” at the end of the animation. The bounce effect is when the rotation goes past the endpoint and springs back into position. The parameter for this command should be true or false.

The third command `rrpC3dSetShape` takes three parameters which determine the overall shape of the carousel. The path of the items in the carousel is an ellipse and this command sets the shape of the ellipse. The `percentWidth` value is the width of the ellipse in terms of the width of the carousel frame. Normally, this parameter is a number less than 100, so the ellipse is within the frame. The `percentHeight` is the height of the ellipse as a percentage of its width. Both these parameters are percentages which are expressed as whole numbers (without the % symbol) where 100 represents 100%.

The last parameter is the angle that determines the orientation of the ellipse. Normally the angle is zero so the ellipse looks like a circle viewed at an angle on the horizontal plane. Altering the angle, which is in degrees between 0 and 360, allow you to produce other orientations.

After these three calls are made, `rrpC3dUpdateLook` is called to update the appearance of the carousel. With this code added to the Update button, switch to the Run tool and enter appropriate values into the fields and then click Update to see how the appearance is changed.

Conclusion

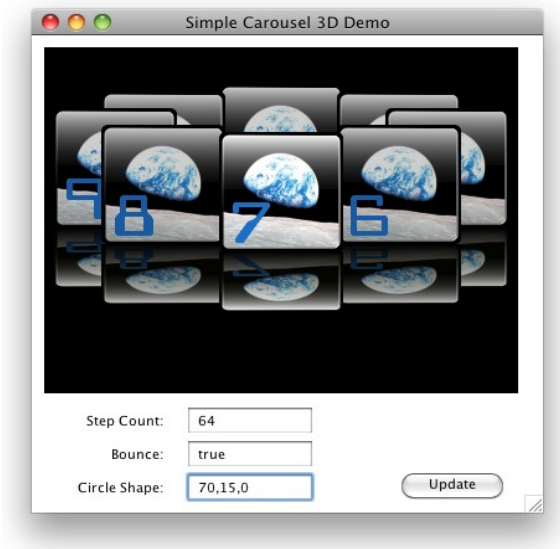
Here is a summary of the commands you have learned about in this chapter.

```
rrpC3dSetStepCount  
rrpC3dSetBounce  
rrpC3dSetShape  
rrpC3dUpdateLook
```

Here is a summary of what you have learned in this chapter:

- How to open the RunRevPlanet Palette
- How to select and copy the carousel control from the palette onto your own stack
- How to select, move and resize the carousel
- How and why the call command is used with RRPC3D
- What the carousel frame is and how to use the short ID of it
- The basic purpose of these commands: rrpC3dSetStepCount, rrpC3dSetBounce, rrpC3dSetShape, rrpC3dUpdateLook
- How the StepCount affects the speed of the carousel animation
- How to change the shape of the ellipse used for the animation

In the next chapter a more sophisticated demo is examined to learn more about using the carousel in a real application.

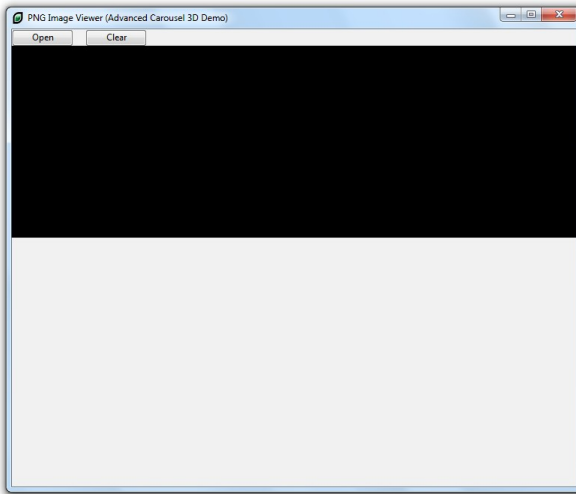


Advanced Carousel 3D Demo

The second demo stack is ready-made and the process of making it is not covered here. Instead this demo stack illustrates the use of the carousel in a real application.

The application is a PNG bitmap viewer that uses RRPC3D to allow scrolling through the images in a folder, with a larger preview of the currently selected image.

Open the Demo-Carousel-3D-Advanced.rev stack from the Demos folder. Before examining the LiveCode you can experiment with it by selecting the Run (Browse) tool in the IDE and then clicking on the Open button of the PNG Image Viewer to navigate to a folder with PNG bitmaps in it.



- If you do not have any suitable PNG files for testing this demo, a selection of files are in the Demos/Images folder for you to use.

After trying the demo, select the Edit (Pointer) tool and then right click in a blank area near the bottom of the Demo-Carousel-3D-Advanced stack and choose the Edit Stack Script command from the contextual menu. In the Script Editor you can examine all of the LiveCode handlers of the demo, which are all in the stack script.

The first handler is for the LiveCode openStack message.

```
on openStack
  CalcImageLoc
  ClearImages
  call "rrpC3dSetMessageMode 1125,true" of stack "rrpCarousel3D"
  call "rrpC3dSetImageSize 1125,100,100" of stack "rrpCarousel3D"
  call "rrpC3dSetReflectionPercent 1125,50" of stack "rrpCarousel3D"
  call "rrpC3dUpdateLook 1125" of stack "rrpCarousel3D"
end openStack
```

The CalcImageLoc command is defined elsewhere in the stack script and sets up the position of the preview image shown in the bottom half of the stack.

☛ Handlers specific to handling the bitmap images that are not directly relevant to how RRPC3D works are not detailed in this chapter, but you can examine the code to see how they work.

Then ClearImages is called and is examined later in the chapter. In summary it removes all the menu items from the Carousel and hides the preview image.

The call to rrpC3dSetMessageMode with the true parameter means RRPC3D sends messages when the user interacts with the carousel. The call to rrpC3dSetImageSize sets the width and height of the items in the carousel to 100 pixels each.

The call to rrpC3dSetReflectionPercent reduces the height of the “reflection” of the carousel items. The parameter of 50 means the height of the reflection is reduced to 50 percent of the original size. In this demo reducing the reflection height allows more room for the preview image, and gives it a more interesting look.

☛ These three calls do not need to be done every time the stack is opened, because settings to RRPC3D are persistent and are stored as properties of the carousel frame. But they are put here for convenience and simplicity.

Lastly, in openStack the rrpC3dUpdateLook command is called to update the carousel with the settings. RRPC3D does not automatically update the appearance of the carousel when such settings are changed, so rrpC3dUpdateLook must be called after you have change any settings that affect the look and behavior of the carousel.

The second handler is for the LiveCode `resizeStack` message.

```
on resizeStack
  local bounds
  lock screen
  CalcImageLoc
  ResizeImage
  put 0,22,the width of me,278 into bounds
  call "rrpC3dSetSize 1125,bounds" of stack "rrpCarousel3D"
  unlock screen
end resizeStack
```

This calls `CalcImageLoc` and `ResizeImage` to adjust the preview image and then calls `rrpC3dSetSize` with a the bounds that are the rectangle for the carousel. Calling `rrpC3dSetSize` does not require another call to `rrpC3dUpdateLook` because it does not change the look and behavior of the carousel, just the dimensions.

☛ The first parameter of `rrpC3dSetSize` and the other commands is the short ID of the carousel frame. In this example it happens to be 1125, but the short ID will be different when you make your own application so you will need to check in the Property Inspector for the appropriate ID number when you add code to your own stacks.

The next handler is called when the stack opens and when the Clear button is clicked to remove all of the menu items from the carousel.

```
command ClearImages
  call "rrpC3dClear 1125" of stack "rrpCarousel3D"
  set the visible of image "Preview" to false
end ClearImages
```

The next two handlers are for messages sent by `RRPC3D`.

```
on rrpC3dChangeStartM pID,pItemNumber
  set the visible of image "Preview" to false
end rrpC3dChangeStartM

on rrpC3dSelectedM pID,pItemNumber
  lock screen
  call "rrpC3dSelectedItem 1125,pItemNumber" of stack "rrpCarousel3D"
  put URL("binfile:" & the result) into image "Preview"
  set the pvWidth of image "Preview" to the width of image "Preview"
  set the pvHeight of image "Preview" to the height of image "Preview"
  set the visible of image "Preview" to true
  ResizeImage
  unlock screen
end rrpC3dSelectedM
```

The first handles the rrpC3dChangeStartM message which RRPC3D sends at the beginning of a carousel animation and is used here to hide the the preview image.

The rrpC3dSelectedM message is sent from RRPC3D when the image at the front of the carousel changes, once the animation is complete. The rrpC3dSelectedM message includes a second parameter which is the number of the menu item that is selected.

In the rrpC3dSelectedM handler the rrpC3dSelectedItem command is called which returns the “name” of the selected item in carousel, which in this case is the path of the carousel item image. The result of this command then used to load the image into the preview.

- ☛ All handlers in RRPC3D are commands, even those that return a value. So the LiveCode result function is used to get the returned information.

The final handler examined in this chapter is LoadPNGImages.

```
command LoadPNGImages pFolder
  get FileList(pFolder, "*.png")
  repeat for each line loopPath in it
    call "rrpC3dAddItemFromPath 1125,loopPath" of stack "rrpCarousel3D"
  end repeat
  call "rrpC3dDraw 1125" of stack "rrpCarousel3D"
end LoadPNGImages
```

This gets a list of all files in a folder with an extension of PNG. For the path of each file found, rrpC3dAddItemFromPath is called to add an item to the carousel using the image at the path. After adding all the images the rrpC3dDraw command is called to show the carousel with the images in it.

If there is more than a couple of items added to the carousel a small progress bar is shown in the middle of the carousel frame while rrpC3dDraw is executing to give an indication of how long the process will take. If there are many large images it may take more than a few seconds to load the images into the carousel for the first time.

- ☛ In this demo if the bitmap images are large there may be an initial “stuttering” during the carousel animation as the images are loaded into memory. After the all images are loaded the first time, the carousel then works smoothly.

There are a couple of other handlers in this demo that call RRPC3D commands, but these are related to do the display of the preview image and are beyond the scope of this chapter.

Conclusion

Here is a summary of the commands you have learned about in this chapter.

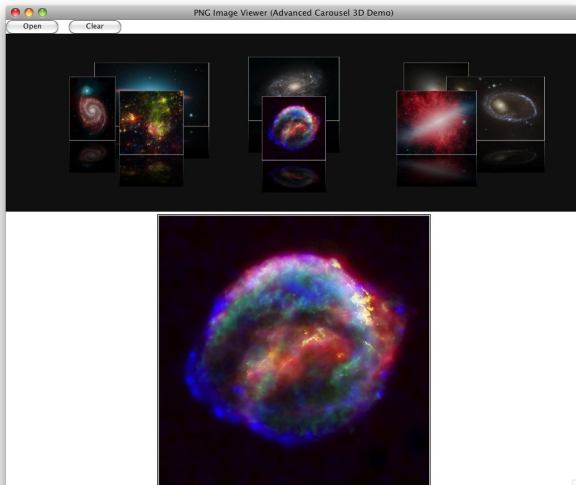
```
rrpC3dSetMessageMode  
rrpC3dSetImageSize  
rrpC3dSetReflectionPercent  
rrpC3dUpdateLook  
rrpC3dSetSize  
rrpC3dClear  
rrpC3dSelectedItem  
rrpC3dAddItemFromPath  
rrpC3dDraw
```

These two messages were also covered.

```
rrpC3dChangeStartM  
rrpC3dSelectedM
```

Here is a summary of what you have learned in this chapter.

- How to turn on the messaging mode of RRPC3D
- How to change the size of the carousel items and reflections
- How to add items to the carousel
- How to clear the carousel
- How to draw and update the carousel
- How to respond to messages from RRP3CD



iOS Carousel 3D Demo

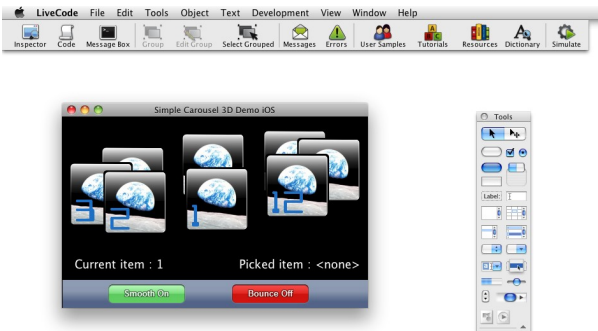
This chapter introduces a demo stack based on the simple demo stack. This third demo stack is ready-made and the process of making it is not covered here. Instead the Demo-Carousel-3D-iOS.rev stack illustrates use of RRPC3D in the mobile environment of the Apple iOS platform.

- ☛ To test this demo you must have LiveCode 4.5.3, or newer, with the Mobile Deployment addon for iOS. The Apple XCode and iOS SDK must also be installed to test this demo on the iPhone simulator.

The simple stack has been modified to suit the look of the iOS platform, and the carousel settings altered to optimised the animation for the lower powered platform.

- ☛ The reflections have been turned off to increase the speed of the animation, although depending on your requirements they could be turned back on.

The stack script has three handlers. There is more LiveCode in the script for the carousel frame, but it is automatically added to your stack when copying the carousel control from the RunRevPlanet Palette, and so is not considered here.



The first handler is for the mouseUp message. It handles the two buttons which use custom icons created with the revlet by Bernd Niggemann at: <http://berndniggemann.on-rev.com/iphonebtnsrevlet/>. The bitmaps of the icons are stored on the card named bitmaps in the stack. Clicking on either button sets whether the smooth and bounce settings of the carousel are turned on.

```

on mouseUp
  switch the short name of the target
  case "butnSmooth"
    call "rrpC3dSmooth 1046" of stack "rrpCarousel3D"
    if the result then
      set the icon of button "butnSmooth" to 1180
      set the hiliteicon of button "butnSmooth" to 1181
      call "rrpC3dSetSmooth 1046,false" of ↵
        stack "rrpCarousel3D"
    else
      set the icon of button "butnSmooth" to 1182
      set the hiliteicon of button "butnSmooth" to 1183
      call "rrpC3dSetSmooth 1046,true" of stack "rrpCarousel3D"
    end if
    break
  case "butnBounce"
    call "rrpC3dBounce 1046" of stack "rrpCarousel3D"
    if the result then
      set the icon of button "butnBounce" to 1178
      set the hiliteicon of button "butnBounce" to 1179
      call "rrpC3dSetBounce 1046,false" of ↵
        stack "rrpCarousel3D"
    else
      set the icon of button "butnBounce" to 1184
      set the hiliteicon of button "butnBounce" to 1185
      call "rrpC3dSetBounce 1046,true" of stack "rrpCarousel3D"
    end if
    break
  end switch
end mouseUp

```

- ☛ The first parameter of rrpC3dSetSmooth and rrpC3dSetBounce is the short ID of the carousel frame. In this demo it happens to be 1046, but the short ID will be different when you make your own application so you will need to check in the Property Inspector for the appropriate ID number when you add the code to your stack.

The next handler is for the rrpC3dChangedM message that is sent by RRPC3D when the front item changes. The second parameter is the number of the menu item currently at the front of the carousel. This message is sent repeatedly during the animation as the front item changes.

```

on rrpC3dChangedM pID, pItemNumber
  put "Current item : " & pItemNumber into field "Current"
  put "Picked item : <none>" into field "Picked"
end rrpC3dChangedM

```

Lastly, this handler indicates the item picked by the user by handling the `rrpC3dPickM` message sent by `RRPC3D`. The second parameter is the number of the menu item at the front of the carousel that has been clicked (touched).

```
on rrpC3dPickM pID, pItemNumber
  put " Picked item : " & pItemNumber into field "Picked"
end rrpC3dPickM
```

☛ Only the front item of the carousel can generate the `rrpC3dPickM` message when it is clicked. A click on any other menu item causes the carousel to animate to change the front item.

Running in the Simulator

To run this demo in the iOS Simulator, the `rrpCarousel3D` stack must be made a substack of `Demo-Carousel-3D-iOS.rev`.

Conclusion

Here is a summary of the commands you have learned about in this chapter.

```
rrpC3dSetSmooth
rrpC3dSetBounce
```

These two messages were also covered.

```
rrpC3dChangedM
rrpC3dPickM
```

Here is a summary of what you have learned in this chapter.

- How to turn on and off the smooth setting of the animation
- How to turn on and off the bounce setting of the animation
- How to respond to a change in the front item of the carousel
- How to respond to a click on the front item To click an item in the menu



Carousel 3D Messages

The RRPC3D optionally sends messages when the user interacts with the carousel. The messages can be intercepted by handlers in your stack. These are commonly named callback handlers. Defining a callback handler in your stack allows you to do actions in response to user interaction with the carousel.

- RRPC3D only sends these messages when the `rrpC3dMessageMode` command has been set to true. By default the `MessageMode` is false, so set it to true if your application defines callback functions.

The messages sent by RRPC3D and when they occur are:

- `rrpC3dChangedM` sent when the front item changes
- `rrpC3dChangeStartM` sent at the start of the animation that changes the front item
- `rrpC3dPickM` sent when the front item is clicked on
- `rrpC3dSelectedM` sent when the animation ends and there is a new front item

In each case the message ends with the letter M as a reminder that these are not handlers that you can call, but are instead messages that your own application can handle.

These messages are sent to the card and stack that contains the carousel.

The handlers that you can put in you stack to respond to the messages are not documented in the Handler Reference and instead listed here.

```
on rrpC3dChangedM pID,pItemNumber
  -- pID is the short Id of the carousel frame
  -- pItemNumber the number of the item, starting from 1
end rrpC3dChangedM
```

```
on rrpC3dChangeStartM pID,pItemNumber
  -- pID is the short Id of the carousel frame
  -- pItemNumber the number of the item, starting from 1
end rrpC3dChangeStartM
```

```
on rrpC3dPickM pID,pItemNumber
  -- pID is the short Id of the carousel frame
  -- pItemNumber the number of the item, starting from 1
end rrpC3dPickM
```

```
on rrpC3dSelectedM pID,pItemNumber
  -- pID is the short Id of the carousel frame
  -- pItemNumber the number of the item, starting from 1
end rrpC3dSelectedM
```

You only need to include in your script handlers for the messages that you specifically are interested in. In RRPC3D messages that are not handled are simply ignored.

Deploying Your Own Application

To deploy or distribute your own application with RRPC3D you must first purchase a License Key. This can be done from the RunRevPlanet.com website, or from other approved vendors. You can visit the RunRevPlanet website for full details, but in summary, a License Key can be purchased with a credit card and the key is emailed to you.

Setting the license name and key

If you have purchased a License Key direct from RunRevPlanet, for RRPC3D to function fully in a standalone application, two extra lines must be added your scripts. These lines are normally where you initialize your application and may be in the openStack handler. Below is a sample of these two lines.

```
set the cLicenseName of stack "rrpCarousel3D" to "Scott McDonald"  
set the cLicenseKey of stack "rrpCarousel3D" to ⚡  
"XXX-XXX-XXX-XXX-XXX-XXX"
```

Here the two licensing properties are set. The first, cLicenseName is the name that is registered when you purchase the License Key. The cLicenseKey property is a special character code that is unique and emailed to you after the purchase is completed.

With these two lines in your applications that have a carousel, you can make use of RRPC3D without any further payments or royalties.

☛ The License Key is for your use only and *must not* be made public or shared with others. This means that the script containing the key must be encrypted with a password. This requires setting the password property of the stack that sets these properties.

Setting these properties needs to be done only in the initialization of your application.

☛ When setting the cLicenceKey or cUnlockCode properties the code must be entered exactly as sent to you. For example, the hyphens are significant and must be included.

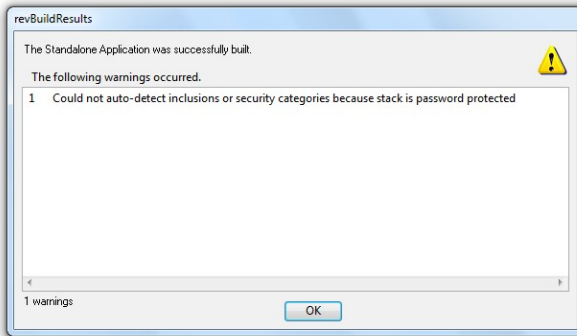
Acknowledgement

While not required, acknowledgement of the use of the “RunRevPlanet Carousel 3D” in the Readme file or the About box of your application is welcome.

Carousel 3D as a Substack

The RRPC3D stack can be distributed with your standalone application as a substack of your application. As a substack it becomes part of your mainstack file, but when making the standalone application you may get an error message.

When choosing the Save as Standalone Application command in the File menu the message shown below may appear. Depending on the version of LiveCode you are using, the wording or appearance of this warning may vary.



- ☛ To prevent this message in the General section of the Standalone Application Settings in the File menu, in the Advanced section the Select inclusions for the standalone application option must be selected. The stacks or script libraries required by your application must be selected manually.

Handler Reference

This chapter documents all the public handlers in the rrpCarousel3D stack. Each entry begins with the name of the handler and on the right whether it is a command or function. Next is the declaration of the handler as found in the rrpCarousel3D.rev stack.

This is followed by a short description of the action of the handler, and if there are any parameters a list of them. Lastly, general comments about the handler are included.

- ☛ Some of these handlers are indicated as being functions, but all are declared as commands. When the first line of the entry indicates a function, this means the command returns a value with the result function.

cLicenseKey

property

`cLicenseKey`

Set this property with a valid key to use RRPC3D in a standalone application.

Default: empty

Comment

The `cLicenseKey` property must be set to a valid key, otherwise the carousel will not work in a standalone application and an error message is shown. This property must be a key that is associated with the `cLicenseName` property. Refer to the *Deploying Your Own Application* chapter for more details.

cLicenseName**property**`cLicenseName`

Set this property with the name used when the License Key was purchased to use RRPC3D in a standalone application.

Default: empty

Comment

The `cLicenseName` property must be set to the name associated with the key in the `cLicenseKey` property, otherwise the carousel will not work in a standalone application and an error message is shown.

cVersionNumber**property**`cVersionNumber`

Read this property to find out the version number of `rrpCarousel3D`

Value: 1.0.0

Comment

The `cVersionNumber` property allows you to check the version of your copy of `rrpCarousel3D`. This value is needed when requesting technical support.

rrpC3dAddItemByID**function****command** rrpC3dAddItemByID pID,pImageID,pData

Adds an item to the carousel by the short ID of the image.

Returns

the number of the item in the carousel

Parameters

pID: the short ID of the carousel frame

pImageID: the short ID of the image object

pData: a string of data associated with the item

Comment

This command adds an item to the the carousel menu with an optional string of data. After adding an item to the carousel the command rrpC3dDraw can be called to the redraw the menu and show the added item, if it is visible.

rrpC3dAddItemByName**function****command** rrpC3dAddItemByName pID,pName,pData

Adds an item to the carousel by the name of the image.

Returns

the number of the item in the carousel

Parameters

pID: the short ID of the carousel frame

pName: the name of the image object

pData: a string of data associated with the item

Comment

This command adds an item to the the carousel menu with an optional string of data. After adding an item to the carousel the command rrpC3dDraw can be called to the redraw the menu and show the added item, if it is visible.

rrpC3dAddItemByPath**function****command** rrpC3dAddItemByPath pID,pPath,pData

Adds an item to the carousel by the file path of the bitmap.

Returns

the number of the item in the carousel

Parameters

pID: the short ID of the carousel frame

pPath: the path of the bitmap file of the image

pData: a string of data associated with the item

Comment

This command adds an item to the the carousel menu with an optional string of data. The pPath must be for a valid bitmap supported by LiveCode. After adding an item to the carousel the command rrpC3dDraw can be called to the redraw the menu and show the added item, if it is visible.

rrpC3dAnimationQuality**function****command** rrpC3dAnimationQuality pID**Returns**

the setting for the quality of the animation: normal, good, or best

Parameters

pID: the short ID of the carousel frame

Comment

RRPC3D can increase the speed of the animation by reducing the quality of the 3D effect. The associated rrpC3dSetAnimationQuality command is used to change this setting.

rrpC3dBackground**function****command** rrpC3dBackground pID**Returns**

the setting for the background: true or false

Parameters

pID: the short ID of the carousel frame

Comment

If true a background color is drawn inside the carousel frame. The associated rrpC3dSetBackground command is used to change this setting.

rrpC3dBackgroundColor**function****command** rrpC3dBackgroundColor pID**Returns**

the background color

Parameters

pID: the short ID of the carousel frame

Comment

The background color is drawn inside the carousel frame if the rrpC3dBackground function is true. The associated rrpC3dSetBackgroundColor command is used to change this setting.

rrpC3dBounce**function****command** rrpC3dBounce pID**Returns**

the setting for the bounce effect: true or false

Parameters

pID: the short ID of the carousel frame object

Comment

The associated rrpC3dSetBounce command is used to change this setting.

rrpC3dBounceStepCount**function****command** `rrpC3dBounceStepCount pID`**Returns**

the setting for the bounce step count: integer value

Parameters

pID: the short ID of the carousel frame object

Comment

The associated `rrpC3dSetBounceStepCount` command is used to change this setting. The `BounceStepCount` sets the amount of “overshoot” that occurs in the animation.

rrpC3dClear**command****command** `rrpC3dClear pID`

Clears the carousel by removing all items.

Parameters

pID: the short ID of the carousel frame object

Comment

This command removes all the items from the carousel by deleting them from memory. This command has an immediate effect and `rrpC3dDraw` does not need to be called.

rrpC3dDeleteItem**command****command** `rrpC3dDeleteItem pID, pIndex`

Deletes a menu item from the carousel.

ParameterspID: the short ID of the carousel frame object
pIndex: the index of the menu item**Comment**

This command deletes the menu item at the index and draws the carousel.

rrpC3dDraw**command****command** rrpC3dDraw pID

Draws the carousel.

Parameters

pID: the short ID of the carousel frame object

Comment

This command is called after adding items to the carousel.

rrpC3dFlush**command****command** rrpC3dFlush pID

Removes all menu items that are not currently visible from the carousel cache.

Parameters

pID: the short ID of the carousel frame object

Comment

This command removes all invisible image objects from the carousel cache by deleting them from memory. The menu items are still in the carousel, and the deleted images are recreated when the items are made visible. This command is useful for low memory situations.

rrpC3dHiliteFront**function****command** rrpC3dHiliteFront pID**Returns**

the setting for whether the front item appears brighter: true or false

Parameters

pID: the short ID of the carousel frame

Comment

If true all carousel items except the front item are shaded to appear darker and less prominent. The associated rrpC3dSetHiliteFront command is used to change this setting.

rrpC3dImageQuality**function****command** rrpC3dImageSize pID,pQuality**Returns**

the quality images of carousel menu items: normal, good, best

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetImageQuality command is used to change this setting.

rrpC3dImageSize**function****command** rrpC3dImageSize pID**Returns**

the size of the images of carousel menu items as a pair of items: width,height

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetImageSize command is used to change this setting.

rrpC3dItemCount**function****command** rrpC3dItemCount pID**Returns**

the number of items in the carousel: an integer

Parameters

pID: the short ID of the carousel frame

Comment

The ItemCount is the total number of items in the carousel.

rrpC3dLoadAllItems**command****command** rrpC3dLoadAllItems pID,pHideProgress

Loads all menu items into the carousel cache.

Parameters

pID: the short ID of the carousel frame object

pHideProgress: true if the progress bar is not shown

Comment

This command loads all image objects into the carousel cache. The menu items are normally only created when an item becomes visible in the carousel. This can cause “stuttering” when the carousel is rotated and image objects are added to the cache for the first time. By calling this command, the cache is filled with all objects.

rrpC3dMakeImageName**function****command** rrpC3dMakeImageName pImageName,pCardName,
pStackName**Returns**

a string that specifies the name of an image in a format suitable for the rrpC3dAddItemFromName command.

Parameters

pImageName: the short name of the image object

pCardName: the short name of the card with the image

pStackName: the short name of the stack with the image

Comment

When adding menu items from image controls on a stack, formatting the string to the format for rrpC3dAddItemFromName can be cumbersome with the need for quotes in appropriate places. This command simplifies the making of such a string.

rrpC3dMessageMode**function****command** rrpC3dMessageMode pID**Returns**

the setting for the message mode: true or false

Parameters

pID: the short ID of the carousel frame

Comment

If true RRPC3D sends messages when the user interacts with the carousel. The associated rrpC3dSetMessageMode command is used to change this setting.

rrpC3dMoveBack**command****command** rrpC3dMoveBack pID

Rotates the carousel 1 item from right to left.

Parameters

pID: the short ID of the carousel frame

Comment

This command is allows the rotation of the carousel without user interaction.

rrpC3dMoveForward**command****command** rrpC3dMoveForward pID

Rotates the carousel 1 item from left to right.

Parameters

pID: the short ID of the carousel frame

Comment

This command is allows the rotation of the carousel without user interaction.

command rrpC3dMoveToItem**command****command** `command rrpC3dMoveToItem pID,pItemNumber`

Rotates the carousel so the pItemNumber menu item is brought to the front.

Parameters

pID: the short ID of the carousel frame

pItemNumber: the number of the item, the first item is number 1

Comment

This command is allows the rotation of the carousel without user interaction.

rrpC3dProgressColor**function****command** `rrpC3dProgressColor pID`**Returns**

the color of the progress bar

Parameters

pID: the short ID of the carousel frame

Comment

The progress bar is shown during calls to rrpC3DDraw and rrpC3DLoadAllItems if there is a significant delay. The associated rrpC3dProgressColor command is used to change this setting.

rrpC3dReflection**function****command** `rrpC3dReflection pID`**Returns**

the setting for the reflection: true or false

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetReflection command is used to change this setting.

rrpC3dReflectionHeight**function****command** rrpC3dReflectionHeight pID**Returns**

the setting for the reflection height as a percent: an integer

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetReflectionHeight command is used to change this setting.

rrpC3dReflectionOffset**function****command** rrpC3dReflectionOffset pID**Returns**

the setting for the reflection offset: an integer

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetReflectionOffset command is used to change this setting.

rrpC3dSelectedData**function****command** rrpC3dSelectedData pID**Returns**

the data for the front (selected) menu item

Parameters

pID: the short ID of the carousel frame

Comment

When calling the rrpC3dAddItemBy commands, there is a optional parameter that stores an additional string associated with the menu item. This command returns that string.

rrpC3dSelectedIndex**function****command** rrpC3dSelectedIndex pID**Returns**

the index of the front (selected) menu item

Parameters

pID: the short ID of the carousel frame

Comment

The index is the number of the menu item which is numbered from 1 up to the number of items in the carousel.

rrpC3dSelectedItem**function****command** rrpC3dSelectedItem pID**Returns**

the name of the front (selected) menu item which is either a short ID, a name, of file path

Parameters

pID: the short ID of the carousel frame

Comment

The string returned by this calling the rrpC3dAddItemBy commands, there is a optional parameter that stores an additional string associated with the menu item. This command returns that string.

rrpC3dSelectedType**function****command** rrpC3dSelectedType pID**Returns**

the type of the front (selected) menu item which is either *I*, *N* or *P* depending on whether the item was added as a short ID, a name, of file path

Parameters

pID: the short ID of the carousel frame

Comment

The type is a string of length 1 that depends on the rrpC3dAddItemBy command used to add the item.

rrpC3dSetAnimationQuality**command****command** rrpC3dAnimationQuality pID,pQuality**Parameters**

pID: the short ID of the carousel frame

pQuality: the quality of the animation: normal, good, best

Comment

The default quality is *best* and should be used unless there are problems with the animation speed or smoothness. With the lower settings of *good* and *normal* RRPC3D reduces the processing load by skipping some of the intermediate steps during the animation resulting in menu items that momentarily may overlap incorrectly.

rrpC3dSetBackground**command****command** rrpC3dSetBackground pID,pEnable

Sets whether the carousel shows a background.

Parameters

pID: the short ID of the carousel frame

pEnable: true or false

Comment

If true a background color is drawn inside the carousel frame. The color of the background is set with the rrpC3dSetBackgroundColor command. When this command is set to false, the background of the underlying card is shown in the carousel.

rrpC3dSetBackgroundColor**command****command** rrpC3dSetBackgroundColor pID,pColor

Sets the background color of the carousel.

Parameters

pID: the short ID of the carousel frame

pColor: A valid LiveCode color

Comment

The background color is drawn inside the carousel frame if the rrpC3dBackground function is true.

rrpC3dSetBounce**command****command** rrpC3dSetBounce pID,pEnable

Sets whether that carousel has a bounce effect in the animation.

Parameters

pID: the short ID of the carousel frame object

pEnable: true or false

Comment

The bounce effect Is the amount of “overshoot” that occurs in the animation at the end of a rotation.

rrpC3dSetBounceStepCount**command****command** rrpC3dSetBounceStepCount pID,pCount

Sets the number of extra steps in the animation when the selected item reaches the front of the carousel.

Parameters

pID: the short ID of the carousel frame object

pCount: the number of extra steps taken for the bounce effect

Comment

The bounce effect Is the amount of “overshoot” that occurs in the animation at the end of a rotation. The default value is 4 but the visual effects will vary depending on the width of the carousel frame. A carousel frame that is relatively small may look best with a lower BounceStepCount, while a large carousel frame may require a higher BounceStepCount for the effect to be noticeable.

rrpC3dSetHiliteFront**command****command** rrpC3dSetHiliteFront pID,pEnable

Sets whether the carousel shows the front item brighter than the other visible menu items.

Parameters

pID: the short ID of the carousel frame

pEnable: true or false

Comment

By default the front item of the carousel is highlighted. This requires extra processing which on some platforms will slow the animation of the carousel. In such cases you can turn off the hilite with this command.

rrpC3dSetImageQuality**command****command** rrpC3dSetImageSize pID,pQuality

Sets the quality of the menu item images in the carousel

Parameters

pID: the short ID of the carousel frame

pQuality: the LiveCode image quality: normal, good or best

Comment

The default quality is *best* and should be used unless there are problems with the loading speed or smoothness.

rrpC3dSetImageSize**command****command** rrpC3dSetImageSize pID,pWidth,pHeight

Sets the size of the menu item images in the carousel

Parameters

pID: the short ID of the carousel frame

pWidth: the maximum width of a menu item image

pHeight: the maximum height of a menu item image

Comment

By default the width and height of the menu item images is 50 pixels. If an image is not a square and the height is larger than the size set with this command, then the image is resized and kept in proportion. This means in that case the width will not be equal to the value set here.

rrpC3dSetVisibleCount**command****command** rrpC3dSetVisibleMode pID,pCount

Sets the number of visible menu items in the carousel.

Parameters

pID: the short ID of the carousel frame

pCount: an integer

Comment

The VisibleCount is the maximum number of menu items shown on the ellipse of the carousel. This number must be a multiple of 4. The carousel can have more or less than this number of items. If there are more items in the carousel than can be shown, as the front menu item changes the newly visible items appear at the back.

rrpC3dSetMessageMode**command****command** rrpC3dSetMessageMode pID,pEnable

Sets whether the carousel sends messages when the menu items change.

Parameters

pID: the short ID of the carousel frame

pEnable: true or false

Comment

By default, the sending of messages is disabled in RRPC3D. Setting this command to true means that messages are sent with each change in the carousel.

rrpC3dSetProgressColor**command****command** rrpC3dSetProgressColor pID,pColor

Sets the color of the progress bar.

Parameters

pID: the short ID of the carousel frame

pColor: A valid LiveCode color

Comment

The progress bar is shown during calls to rrpC3DDraw and rrpC3DLoadAllItems if there is a significant delay. By default the color of the progress bar is a light grey, but if necessary can be changed to a different color with this command.

rrpC3dSetRectangle**command****command** rrpC3dSetRectangle pID,pRectangle

Sets the rectangle of the carousel frame.

Parameters

pID: the short ID of the carousel frame

pRectangle: four comma separated integers that defined the left, top, right and bottom edges of the carousel frame

Comment

This command can be used to resize or move the carousel frame. For example, this could be used in a resizeStack handler if the carousel size or position must be adjusted in response to such a message.

rrpC3dSetReflection**command****command** rrpC3dSetReflection pID,pEnable

Sets whether reflections are shown in the carousel.

Parameters

pID: the short ID of the carousel frame

pEnable: true or false

Comment

A reflection is a reversed mirror image of each menu item on the carousel. Having a reflection enhances the 3-D effect of the carousel, but does require extra processing which on some platforms will slow the animation of the carousel. In such cases you can turn off the reflection with this command.

rrpC3dSetReflectionOffset**command****command** rrpC3dSetReflectionOffset pID,pOffset

Sets the offset of the reflection from the default position.

Parameters

pID: the short ID of the carousel frame

pOffset: the number of pixels that the reflections are shifted up or down

Comment

By default the top edge of the reflection is aligned with the bottom edge of the menu item. By setting to a positive number the menu items of the carousel can appear to “float above” the carousel background.

rrpC3dSetReflectionHeight**command****command** rrpC3dSetReflectionHeight pID,pHeight

Sets the height of the reflections as a percentage of the full-size.

Parameters

pID: the short ID of the carousel frame

pHeight: the height of the reflection as a percentage where 100 means the reflection is the same height as image of the menu item

Comment

By default when reflections are shown, the image used for the reflection is the same height as the image of the menu item. In some cases, reducing the height of the reflection gives a more attractive appearance to the carousel.

rrpC3dSetShape**command****command** rrpC3dSetShape pID,pWidth,pHeight,pAngle

Sets the shape of the elliptical path used by the carousel.

Parameters

pID: the short ID of the carousel frame

pWidth: the width of the ellipse as a percentage of the width of the carousel frame

pHeight: the height of ellipse as a percentage of its width

pAngle: The angle of the ellipse where 0 Means the ellipse is oriented in the horizontal plane

Comment

Setting this command changes the shape of the carousel ellipse. Appropriate settings for the width and height depends on the size of the carousel frame and the size of the menu items in the carousel. After calling this command, rrpC3dUpdateLook must be called to affect the changes.

rrpC3dSetSmooth**command****command** rrpC3dSetSmooth pID,pEnable

Sets whether the rotation slows towards the end of the animation.

Parameters

pID: the short ID of the carousel frame

pEnable: true or false

Comment

Setting this command to true smooths the look of the rotation. With this setting set to false the animation stops instantly when the selected item is at the front. When set to true, the animation slows as the item reaches the front.

rrpC3dSetStepCount**command****command** rrpC3dSetStepCount pID,pCount

Sets the number of steps in the animation when the carousel rotates from one item to the next.

Parameters

pID: the short ID of the carousel frame

pCount: the number of intermediate steps that the animation users between each item

Comment

The larger the step count, the slower the animation. The step count also determines the smoothness of the animation. Two small a value for this setting and the menu items will appear to jump from each position to the next during the rotation. This value in conjunction with the StepDelay determines the speed of the animation.

rrpC3dSetStepDelay**command****command** rrpC3dSetStepDelay pID,pMilliseconds

Sets the delay in each step of the animation.

Parameters

pID: the short ID of the carousel frame

pMilliseconds: the number of milliseconds between each step

Comment

By default the delay between each step of the animation is zero milliseconds. If the rotation of the carousel is too fast, this command is used to increase the time taken for each step in the animation. This value in conjunction with the StepCount determines the speed of the animation.

rrpC3dSetVerticalOffset**command****command** rrpC3dSetVerticalOffset pID,pOffset

Moves the vertical position of the carousel within the carousel frame.

Parameters

pID: the short ID of the carousel frame

pOffset: A positive or negative integer to move the carousel down or up within the carousel frame.

Comment

The carousel is approximately vertically centred in the carousel frame, But depending on the height of the item images and whether the reflection is shown, the position of the carousel may need to be adjusted for the best look. The pOffset parameter moves to carousel as a percentage of the height of the carousel frame.

rrpC3dSmooth**function****command** rrpC3dSmooth pID**Returns**

the smooth setting: true or false

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetSmooth command is used to change this setting.

rrpC3dStepCount**function****command** rrpC3dStepCount pID**Returns**

the step count of the carousel: an integer

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetStepCount command is used to change this setting.

rrpC3dStepDelay**function****command** rrpC3dStepDelay pID**Returns**

the step delay of the carousel: an integer

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetStepDelay command is used to change this setting.

rrpC3dUpdateLook**command****command** rrpC3dUpdateLook pID

Updates the look of the carousel after its settings have been changed.

Parameters

pID: the short ID of the carousel frame

Comment

The commands that sets the attributes of the carousel do not immediately redraw the carousel with the new settings. This command is called after all settings have been done to update the look of the carousel.

rrpC3dUpdateSize**command****command** rrpC3dUpdateSize pID

Updates the internal state of the carousel after the rectangle of the carousel frame has been changed.

Parameters

pID: the short ID of the carousel frame

Comment

This command must be called if the size of the carousel frame has changed. This is necessary to ensure that the path of the animation is correct for the new size.

rrpC3dVerticalOffset**function****command** rrpC3dVerticalOffsetPercent pID**Returns**

the offset of the carousel within the carousel frame: an integer

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetVerticalOffset command is used to change this setting.

rrpC3dVisibleCount**function****command** rrpC3dVisibleCount pID**Returns**

the maximum number of menu items that can be visible in the carousel at the same time: an integer

Parameters

pID: the short ID of the carousel frame

Comment

The associated rrpC3dSetVisibleCount command is used to change this setting.

rrpC3dVisibleIndices**function****command** rrpC3dVisibleIndices pID**Returns**

a list of the item numbers of the visible menu items.

Parameters

pID: the short ID of the carousel frame

Comment

The item numbers are a comma delimited list on a single line.

