RunRevPlanet HTMLToolbar Stack

The RunRevPlanet HTMLToolbar Stack adds a formatting toolbar to any field in your Revolution application. The stack is written in 100% Revolution script and does not use any platform specific libraries. Add extended editing to your cross platform application for Linux, Mac or Windows.

Guide and Reference

RunRevPlanet HTMLToolbar

Guide and Reference

July 2009 Edition

Address: Scott McDonald PC Services
PO Box 139, Newtown 2042
New South Wales, Australia
Facsimile: +612-8580-5726
Website: http://www.runrevplanet.com
Email: runrevplanet@smpcsupport.com

Table of Contents

About the Cover Image

The image on the cover this RunRevPlanet HTMLToolbar 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 close up image of Hall crater on Phobos was obtained by the Viking 1 Orbiter on Feb. 28, 1977, when the spacecraft was only 606 kilometers from the Martian moon . The crater, at the top left, is 6 km in diameter. Just to the upper left of the crater is Phobos' south pole. The whole image shows an area about 16 km across. The original image vo1_252a63.tiff can be found at:
http://nssdc.gsfc.nasa.gov/imgcat/html/object_page/vo1_252a63.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

Introduction

The RunRevPlanet HTMLToolbar adds a basic editing toolbar to any field in your Revolution application. The RRP HTMLToolbar is written in 100% Revolution script and does not use any platform specific libraries. This means you can add basic editing to your cross platform application for Linux, Mac or Windows. The RRP HTMLToolbar (RRPHT) is simple to use and requires only a one line of script to implement a more extensive HTML subset than a regular field object, and add a toolbar for easier editing in your Revolution applications. By using RRPHT you are free to focus on the unique features of your application and not be concerned with the low-level details of implementing an editing toolbar.

The RRPHT provides an toolbar to control the attributes of the text in a field. The toolbar is active and shows the attributes of the text at the cursor position in the field. The text can be read from the field in a HTML compatible format. All this is added with a single line of script in your application.

Features of the RRPHT are:

- Automatic editing toolbar
- Support for bold, italic and underline attributes
- Support for heading tags.
- Support for unordered lists (bullet list)
- Support for ordered lists (numbered lists)
- Support for hyperlinks
- HTML compatible supporting the following tags: <p>, <b>, <i>, <u>, <ul>, <ol>, <li>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, <font>, <a>
- Fast operation
- Find button for text searching
- 100% Revolution script

Installation & Requirements

To use the RRPHT you must have the following:

- Revolution 3.0, or newer, in the Studio or Enterprise edition.

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

```
rrpHTMLToolbar.rev
rrpHTMLToolbarFind.rev
rrpHTMLToolbarLink.rev
rrpHTMLToolbar-Guide-Reference.pdf
Readme.txt
License.txt
/Demos/Demo-HTMLToolbar.rev
```

Also included are three folders:

```
/Demo-HTMLToolbar/Linux
/Demo-HTMLToolbar/MacOSX
/Demo-HTMLToolbar/Windows
```

These contain executable bundles of the demo application, which illustrates the use of the RRPHT in a simple way.

Conventions

The terms "rrpHTMLToolbar stack", "rrpHTMLToolbar.rev" and "RRPHT" may be used interchangeably in this guide depending on the context. Each of these names refers to the RunRevPlanet HTMLToolbar.

Handlers in Revolution come in different types. There are event handlers, commands, functions and property handlers. For the sake of brevity where the specific type is not important, throughout this guide these terms may be used interchangeably. The generic term handler may refer to commands, functions and property handlers.

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

Revolution 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 rawKeyDown
   -- event handler added by the rrpHTMLToolbar stack
   call "RawKeyDownPress memoText" of stack "rrpHTMLToolbar"
   pass rawKeyDown
end rawKeyDown
```

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 must be entered as a single line in the Script Editor.

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

In this guide the term Object Inspector is used to refer to the Property Inspector as it known in the contextual menus of the Revolution IDE.

Licensing Agreement

The RRP HTMLToolbar 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 HTMLToolbar is Copyright (c) 2009 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 HTMLToolbar to evaluate the RRP HTMLToolbar for an unlimited time. You may not build or distribute a standalone application that uses the RRP HTMLToolbar 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 HTMLToolbar and the demonstration files after you have purchased a License Key or Unlock Code. You may not distribute applications that use the RRP HTMLToolbar with your License Key in an unencrypted stack file.

When distributing your own executable applications based on the RRP HTMLToolbar 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 HTMLToolbar. However others who receive your scripts need to download their own copy of the RRP HTMLToolbar and purchase a License Key or Unlock Code in order to write programs that use your own code.

The RRP HTMLToolbar 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 HTMLToolbar 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 HTMLToolbar. 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 HTMLToolbar 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 HTMLToolbar 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 HTMLToolbar 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)2000-2010 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 Toolbar.

All names of products and companies used in this Licensing Agreement, the RRP HTMLToolbar, 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.

Using the RRP HTMLToolbar

The RRPHT is a self contained stack that you can add as a substack to Revolution application to add an editing toolbar to any field. Understanding how the stack works is not necessary to use it, just add the required scripts in the necessary event handlers and you have an instant editing toolbar.

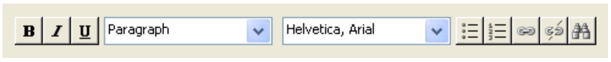
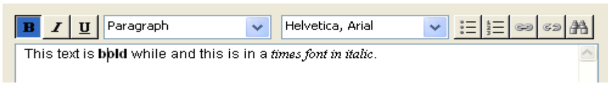


Figure 1: The HTMLToolbar toolbar

The toolbar is added to a field in your application with a single line of script and is immediately active. The buttons in the illustration above from left to right are for:

- Bold style
- Italic style
- Underline style
- Paragraph/Heading style
- Font style
- Bullet (unordered) list
- Numbered (ordered) ist
- Insert hyperlink
- Remove hyperlink
- Find

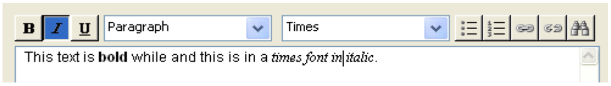
The toolbar is active, in other words it also shows the style of the text at the cursor, as shown in Figure 2 below where the current word is bold.



This text is **bold** while and this is in a *times font in italic*.

Figure 2: The toolbar with bold text

Moving the cursor across to the italicised Times text the toolbar shows this.



This text is **bold** while and this is in a *times font in italic*.

Figure 3: The toolbar with Times in italic

In summary, the toolbar works as you and your users would expect. The next two chapters detail explain the basic operation of the RRPHT.

Scripting an Editor

To use the RRPHT you need to add a short script to your application. A detailed description of the steps required for these scripts is in the next Tutorial chapter, but this chapter is a brief overview that outlines the basics.

Initialize

Firstly you need to initialize the stack. This is done with the following statement.

```
call "Initialize memoText" of stack "rrpHTMLToolbar"
```

Where memoText is the name of the field object to be given an editing toolbar. Here the field is named memoText, but this is just an example, it can be the name of any field in your application that needs the toolbar. Initialization needs to be called only twice, once during design time and then during the initialization of your application.

💣 The above script must be executed during the design time of your application, so the toolbar is created before your application is saved as a standalone. This is necessary because the initialization process adds scripts to your stack, which may exceed the scriptLimits of the Revolution runtime.

At design time from the IDE you can execute this script in the Message Box to create the toolbar. When the Initialize call is made the toolbar is immediately added above the field.

💣 If the position of the field object results in a awkwardly positioned toolbar, you can delete the toolbar objects from the stack and re-position the field before calling Initialize again.

During the initialization of your application, a convenient place call it is in the openStack handler of the card or stack.

```
on openStack
   call "Initialize memoText" of stack "rrpHTMLToolbar"
end openStack
```

Using the editor

After the stack is initialized the toolbar is created and is ready to use, there are no additional scripts to add to your application.

💣 Further scripts are added to the field object, but these are automatically inserted by RRPHT and you do not need to manually add them to your code.

Putting and getting text from the Editor

While the toolbar handles the editing and formatting of the text in the field object, special formatting characters are used internally to support the features of the editor. What this means is that the text and HTMLText properties should not be used to set or get the text in the field with the toolbar.

Instead the cHTMLSubset property must be used to set and get the text in the field. This is a property of the field, not the rrpHTMLToolbar stack. For example, continuing with the script above, this line puts the formatted HTML text from the memoText field into myVariable.

```
put the cHTMLSubset of field "memoText" into myVariable
```

The HTML from cHTMLSubset is a minimal subset that is required to support all the features of the toolbar. The Supported HTML chapter describes the tags that are supported by RRPHT.

💣 When setting the text in the field with cHTMLSubset, any tags that are not supported by RRPHT are stripped from the property and will not be present when the property is read.

Start using

Some library stacks for Revolution use the start using command to put the scripts of the stack in the message path. With the RRPHT this is not recommended and the name of stack should be explicitly put into your scripts when referencing the initialize handler.

HTMLToolbar Demo

In this chapter, you can follow the steps required to make the demo-HTMLToolbar.rev stack. Here it is assumed that you have a basic knowledge of using the Revolution IDE, but if you are new to Revolution the details here should be sufficient for you to make the stack. Alternately, you can open the completed stack in the Demos folder and examine it instead of creating the stack yourself.

In this tutorial, a simple HTMLToolbar Stack with a single field large enough for multiple lines of text and two buttons to initialize and reset the toolbar will be added.

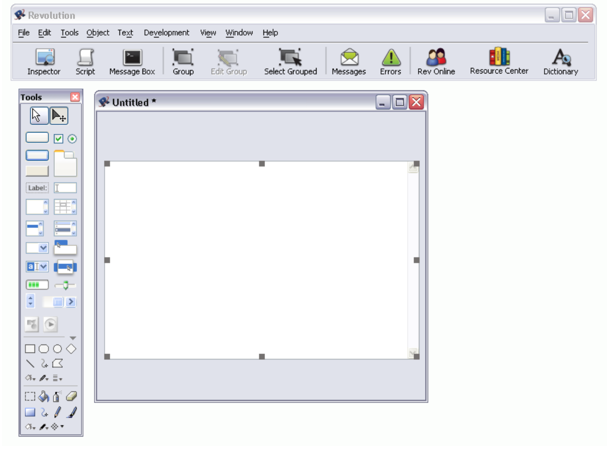


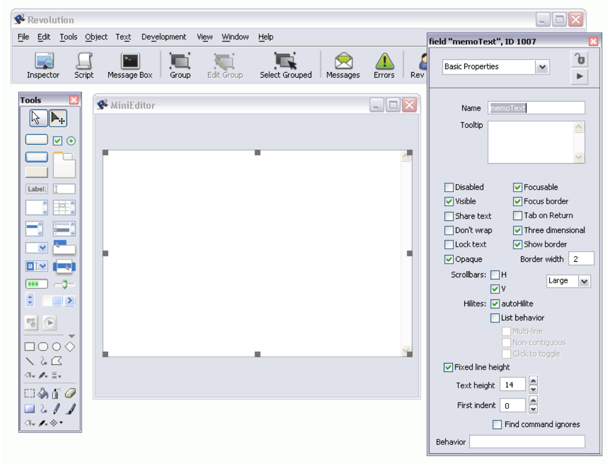
Figure 4: The Scrolling Field object in the new stack

First select the New Mainstack command in the File menu. Then click and drag a Scrolling Field object onto the stack window. Resize the field so that it fills most of the window, but leave enough free space at the top for the toolbar and at the bottom for two buttons as shown in Figure 4.

Next click on the field object in the stack window and select the Inspector on the IDE toolbar. Then in the Object Inspector change the name to memoText as shown in Figure 5. When naming objects, the convention used here is to have up to 4 lowercase letters indicating more about the type of object, which is then followed (without a space) by a capitalized word to describe what the object will contain or its action.

💣 The Figures in this chapter show objects that are sized to fit neatly on these pages, but you can be more generous with the sizes of your objects when working though this chapter.

Click anywhere in the blank background of the stack window with a single click and in the Object Inspector set the Name and Title of the stack to Demo-HTMLToolbar and Mini Editor. A double click on the background selects the card, which is not what you want. You can check that it is the stack that is selected by making sure the title bar of the Object Inspector begins with the word stack.



field "memoText", ID 1007

Basic Properties

Name: memoText
Tooltip:

[ ] Disabled [✓] Focusable
[✓] Visible [✓] Focus border
[ ] Share text [ ] Tab on Return
[ ] Don't wrap [✓] Three dimensional
[ ] Lock text [✓] Show border
[✓] Opaque Border width 2

Scrollbars: [ ] H [✓] V Large

Hilites: [✓] autoHilite
[ ] List behavior
[ ] Multi-line
[ ] Non-contiguous
[ ] Click to toggle

[✓] Fixed line height
Text height 14
First indent 0
[ ] Find command ignores

Behavior

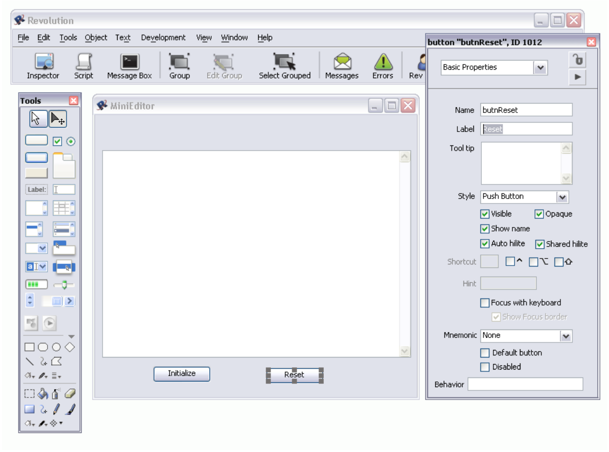
Figure 5: Renaming the field object to memoText

At this point it is worthwhile saving the stack by choosing the Save command in the File menu, and since this is the first time the stack is saved you will be prompted for the filename and folder that you want to use.

Now, click and drag two Push Buttons onto the stack below the field object. With these two buttons in position, the Object Inspector is used to give them the required properties. In this tutorial, the different IDE windows have been positioned to for clear images in this chapter, but when you do it the windows may appear in different positions and you may find additional IDE windows appearing.

In both these cases, the difference between what you see on your screen and what is shown here is unimportant. You can close unnecessary windows, such as the Application Browser, when it is not needed.

In the Object Inspector the Name of the first button is set to butnInitialize, and the Label to Initialize. Set the same properties of the second button to butnReset and Reset respectively as in Figure 6.


Revolution
File Edit Tools Object Text Development View Window Help
Inspector Script Message Box Group Edit Group Select Grouped Messages Errors Rev
Tools
MiniEditor
Initialize
Reset

button "butnReset", ID 1012

Basic Properties

Name butnReset
Label Reset
Tool tip
Style Push Button
✓ Visible ✓ Opaque
✓ Show name
✓ Auto hilite ✓ Shared hilite
Shortcut
Hint
☐ Focus with keyboard
✓ Show Focus border
Mnemonic None
☐ Default button
☐ Disabled
Behavior

Figure 6: The two buttons Initialize and Reset

Now you are ready to add the scripts to the buttons to make the Mini Editor work.

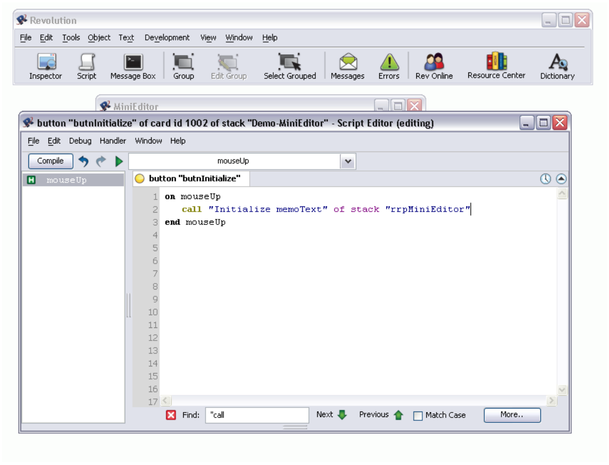
The first script is added to the Initialize button. One way to do this is to select the button in the Mini Editor window and then right click (control click with a single button mouse) and choose the Edit Script command from the contextual menu. When you choose this command the Script Editor window opens, and here it is resized to fit on this page. By default when you first choose the Edit Script command for a button, an empty mouseUp handler.

This is the required handler, so the script is added between the first and last lines. Add this line:

```
call "Initialize memoText" of stack "rrpHTMLToolbar"
```

Initialize is the name of the handler being called in the rrpHTMLToolbar stack, and memoText is the name of the field that is have the editor toolbar. This script is shown in Figure 7.

Press the control-s (command-s) keyboard shortcut and the Script Editor saves the handler and checks for errors in the script, which there should be none. If there are errors, please check the line in the mouseUp handler for typing mistakes.


Revolution
File Edit Tools Object Text Development View Window Help
Inspector Script Message Box Group Edit Group Select Grouped Messages Errors Rev Online Resource Center Dictionary
MiniEditor
button "butnInitialize" of card id 1002 of stack "Demo-MiniEditor" - Script Editor (editing)
File Edit Debug Handler Window Help
Compile
mouseUp
button "butnInitialize"
Find: "call
Next Previous Match Case More..

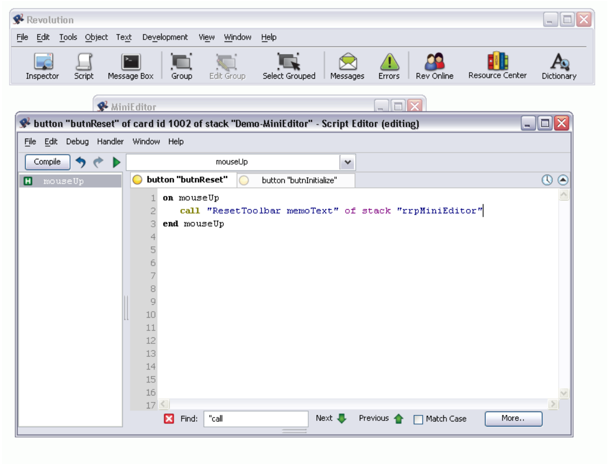
```
on mouseUp
   call "Initialize memoText" of stack "rrpMiniEditor"
end mouseUp
```

Figure 7: Initializing the HTMLToolbar stack

Then in a similar way add this script to the mouseUp handler of the Reset button.

```
call "ResetToolbar memoText" of stack "rrpHTMLToolbar"
```

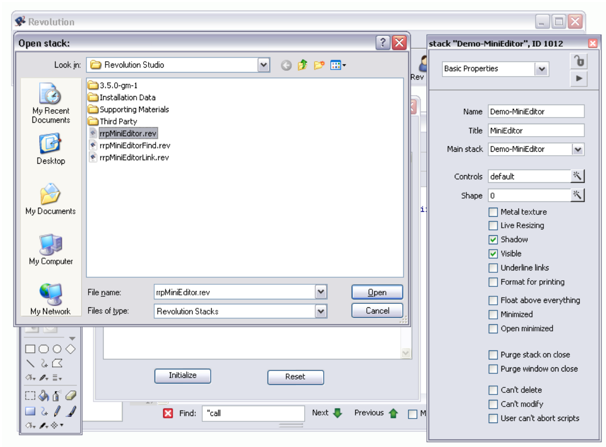
The script is shown in the Script Editor in Figure 8.


Revolution
File Edit Tools Object Text Development View Window Help
Inspector Script Message Box Group Edit Group Select Grouped Messages Errors Rev Online Resource Center Dictionary
MiniEditor
button "butnReset" of card id 1002 of stack "Demo-MiniEditor" - Script Editor (editing)
File Edit Debug Handler Window Help
Compile
mouseUp
button "butnReset"
button "butnInitialize"
Find: "call
Next Previous Match Case More..

```
on mouseUp
   call "ResetToolbar memoText" of stack "rrpMiniEditor"
end mouseUp
```

Figure 8: The script for the Reset button

After saving again, the RRP HTMLToolbar is added to the application. To do this, open the rrpHTMLToolbar.rev stack with the Open Stack command in the File menu. You may need to navigate to the folder where the rrpHTMLToolbar.rev stack is installed if it is in another folder, but if you first put it into the same folder as your Demo-HTMLToolbar.rev stack it will be easy to find. In Figure 9, the stack is in the Revolution folder, but it may be elsewhere on your system.


Open stack:

Look in: Revolution Studio

- 3.5.0-gm-1
- Installation Data
- Supporting Materials
- Third Party
- rrpMiniEditor.rev
- rrpMiniEditorFind.rev
- rrpMiniEditorLink.rev

File name: rrpMiniEditor.rev

Files of type: Revolution Stacks

stack "Demo-MiniEditor", ID 1012

Basic Properties

Name: Demo-MiniEditor

Title: MiniEditor

Main stack: Demo-MiniEditor

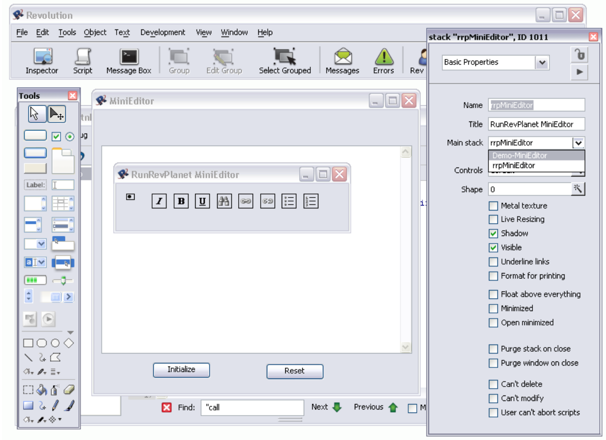
Controls: default

Shape: 0

- [ ] Metal texture
- [ ] Live Resizing
- [x] Shadow
- [x] Visible
- [ ] Underline links
- [ ] Format for printing
- [ ] Float above everything
- [ ] Minimized
- [ ] Open minimized
- [ ] Purge stack on close
- [ ] Purge window on close
- [ ] Can't delete
- [ ] Can't modify
- [ ] User can't abort scripts

Figure 9: Opening the rrpHTMLToolbar stack

Once open, the rrpHTMLToolbar.rev stack is visible by the window that has the title RunRevPlanet HTMLToolbar. A few details of the stack can be seen in the Object Inspector. This stack contains the library of handlers that combined are the HTMLToolbar. The stack window is also the container for the icons used in the toolbar.


stack "rrpMiniEditor", ID 1011

Basic Properties

Name: rrpMiniEditor

Title: RunRevPlanet MiniEditor

Main stack: rrpMiniEditor (Demo-MiniEditor, rrpMiniEditor)

Shape: 0

- [ ] Metal texture
- [ ] Live Resizing
- [x] Shadow
- [x] Visible
- [ ] Underline links
- [ ] Format for printing
- [ ] Float above everything
- [ ] Minimized
- [ ] Open minimized
- [ ] Purge stack on close
- [ ] Purge window on close
- [ ] Can't delete
- [ ] Can't modify
- [ ] User can't abort scripts

Figure 10: Making rrpHTMLToolbar a substack of the demo

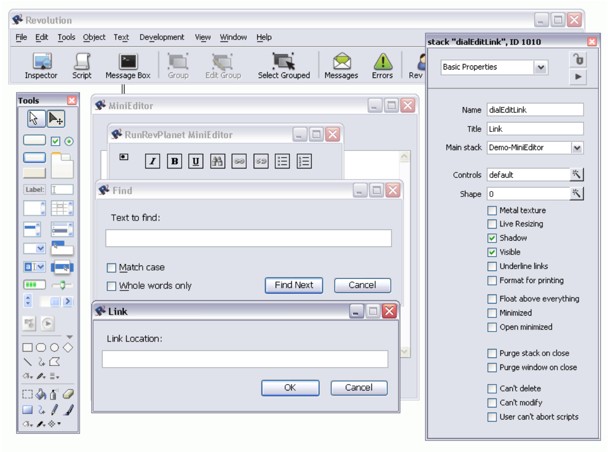
For the convenience of this tutorial the HTMLToolbar stack can be made a substack of the mainstack. To do this, first select the rrpHTMLToolbar.rev stack with a single click anywhere on the blank background of the RunRevPlanet HTMLToolbar window. Then in the Object Inspector in the Main stack drop-down list select Demo-HTMLToolbar as shown in Figure 10.

The rrpHTMLToolbar stack does not need to be made a substack in a standalone application, but this is done in this tutorial so the demo can be saved as a single stack file for easy distribution.

There are two more stacks that are part of the HTMLToolbar. They are the dialogue boxes for the Link and Find buttons on the toolbar. Again, it is not necessary for them to be substacks, but for convenience in this tutorial they are both made substacks. To do this, repeat the same steps used to make rrpHTMLToolbar.rev stack a substack. The two stacks rrpHTMLToolbarFind.rev and rrpHTMLToolbarLink.rev should be found in the same folder as rrpHTMLToolbar.rev as shown back in Figure 9.

After opening each of the stacks as in Figure 11, select each one and make it a substack in the Object Inspector in the Main stack drop-down list select Demo-HTMLToolbar.

When using these two dialogues in your own applications you can customize their appearance to match your user interface style.


Revolution
File Edit Tools Object Text Development View Window Help
Inspector Script Message Box Group Edit Group Select Grouped Messages Errors Rev
Tools
MiniEditor
RunRevPlanet MiniEditor
Find
Text to find:
Match case
Whole words only
Find Next
Cancel
Link
Link Location:
OK
Cancel

stack "dialEditLink", ID 1010

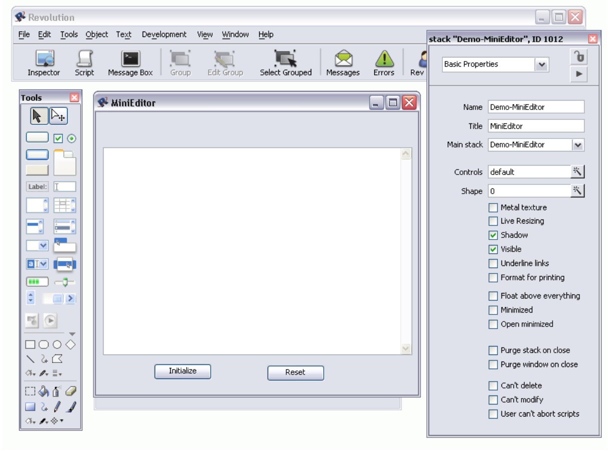
Basic Properties

Name: dialEditLink
Title: Link
Main stack: Demo-MiniEditor
Controls: default
Shape: 0

☐ Metal texture
☐ Live Resizing
☑ Shadow
☑ Visible
☐ Underline links
☐ Format for printing
☐ Float above everything
☐ Minimized
☐ Open minimized
☐ Purge stack on close
☐ Purge window on close
☐ Can't delete
☐ Can't modify
☐ User can't abort scripts

Figure 11: The Find and Link stacks

You are done! With only these objects and two scripts you have made a simple, but effective editor application. Before testing the stack, save it again. Then choose the Run (browse) tool to make the Demo-HTMLToolbar stack active as in Figure 12.


Revolution
File Edit Tools Object Text Development View Window Help
Inspector Script Message Box Group Edit Group Select Grouped Messages Errors Rev
Tools
MiniEditor
Initialize
Reset

stack "Demo-MiniEditor", ID 1012

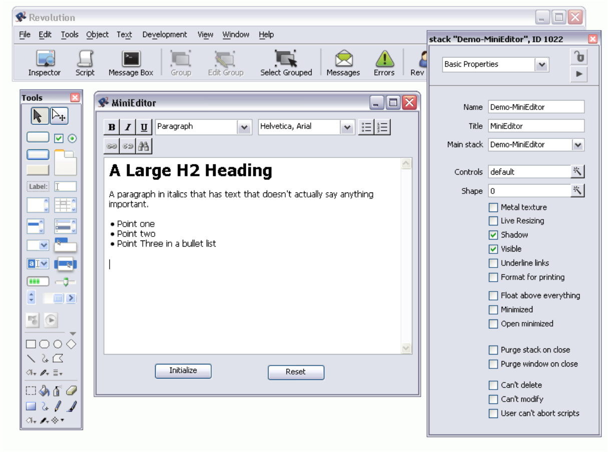
Basic Properties

Name: Demo-MiniEditor
Title: MiniEditor
Main stack: Demo-MiniEditor
Controls: default
Shape: 0

- [ ] Metal texture
- [ ] Live Resizing
- [x] Shadow
- [x] Visible
- [ ] Underline links
- [ ] Format for printing
- [ ] Float above everything
- [ ] Minimized
- [ ] Open minimized
- [ ] Purge stack on close
- [ ] Purge window on close
- [ ] Can't delete
- [ ] Can't modify
- [ ] User can't abort scripts

Figure 12: Choose the Run tool to get ready to edit

Then click on the Initialize button. This creates the toolbar for the HTMLToolbar and you can start entering text and testing the buttons on the toolbar like in Figure 13.


Revolution
File Edit Tools Object Text Development View Window Help
Inspector Script Message Box Group Edit Group Select Grouped Messages Errors Rev
Tools
MiniEditor
B I U
Paragraph
Helvetica, Arial
Initialize
Reset

A Large H2 Heading

A paragraph in italics that has text that doesn't actually say anything important.

- Point one
- Point two
- Point Three in a bullet list

stack "Demo-MiniEditor", ID 1022

Basic Properties

Name: Demo-MiniEditor
Title: MiniEditor
Main stack: Demo-MiniEditor
Controls: default
Shape: 0

- [ ] Metal texture
- [ ] Live Resizing
- [x] Shadow
- [x] Visible
- [ ] Underline links
- [ ] Format for printing
- [ ] Float above everything
- [ ] Minimized
- [ ] Open minimized
- [ ] Purge stack on close
- [ ] Purge window on close
- [ ] Can't delete
- [ ] Can't modify
- [ ] User can't abort scripts

Figure 13: Using the HTMLToolbar toolbar

💣 The toolbar buttons may overlap if there is not enough free space above the field object. If this happens, click on the Reset button to remove the toolbar.

Then choose the Edit (pointer) tool to resize the memoText field and try again.

In a finished application, even though the Initialize command is called at design time, here by clicking on a button, it must also be part of the initialization of your application.

The best place to initialize the rrpHTMLToolbar stack may vary, but the openStack handler on the card that has the field object is normally a convenient place. To do this you could right click (control click) anywhere on the blank background of the Demo-HTMLToolbar stack and choose the Edit Card Script command from the contextual menu.

When opening the Script Editor on a card for the first time there is no default handler, so the complete handler needs to be entered with these lines.

```
on openStack
   call "Initialize memoText" of stack "rrpHTMLToolbar"
end openStack
```

This is the same script that was added to the mouseUp event of the Initialize button in this tutorial.

💣 If the memoText field script is open in the Script Editor, you may get a warning about "Scripts externally modified" when clicking on the Initialize or Reset buttons. When that occurs you should click on Reload, provided your own changes have already been saved.

Deploying Your Own Application

To deploy or distribute your own application with the RRPHT you must first purchase a License Key or Unlock Code. This can be done from the RunRevPlanet 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.

If purchasing from another vendor, the process may be slightly different and you may have an Unlock Code but the end result is the same, you can deploy or distribute your own application with RRPHT.

💣 Without a valid key you are not entitled to distribute standalone applications and an application without a valid key will not have a fully working editing toolbar.

Depending on whether you have a license name and key or an unlock code, follow the appropriate section next to find out how to use them in your standalone applications.

Setting the license name and key

For the RRPHT to function fully in a standalone application, two extra lines must be added your scripts. These lines should be immediately after the call to initialize the stack. Below is a sample of these two lines.

```
call "Initialize memoText" of stack "rrpHTMLToolbar"
set the cLicenseName of stack "rrpHTMLToolbar" to "Scott McDonald"
set the cLicenseKey of stack "rrpHTMLToolbar" ↩
   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 all your applications that you want to distribute with the HTMLToolbar, you can make use of the RRPHT 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 contains the initialization script for the RRPHT.

Setting these two properties needs to be done only in the initialization of your application and must be done after calling the Initialize command.

Setting the unlock code

If you have purchased a developer's license from another vendor you will have a single unlock code that allows RRPHT to function fully in a standalone application, one extra line must be added your scripts. This line should be immediately after the call to initialize the stack. Below is a sample of the line.

```
call "Initialize memoText" of stack "rrpHTMLToolbar"
set the cUnlockCode of stack "rrpSpellCheck" ↵
   to "XXX-XXX-XXX-XXX-XXX-XXX-XXX-XXX"
```

Here the unlock property is set. The cUnlockCode property is set to a special character code that is unique and made available to you after the purchase is complete.

With this extra line in all applications you want to distribute with the HTMLToolbar, you can make use of the RRPHT without any further payments or royalties.

💣 The Unlock Code is for your use only and *must not* be made public or shared with others. This means that the script containing the code must be encrypted with a password. This requires setting the password property of the stack that contains the initialization script for the RRPSCS.

Setting this property needs to be done only in the initialization of your application and must be done after calling the Initialize command.

💣 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 HTMLToolbar" in the Readme file or the About box of your application is welcome.

HTMLToolbar as a Substack

The RRPHT can be distributed with your standalone application as a separate stack, rrpHTMLToolbar.rev or as a substack in your application. If included as a substack it becomes part of your mainstack file which is can be convenient, 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 in Figure 14 may appear.

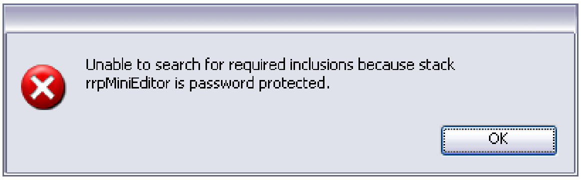

Unable to search for required inclusions because stack rrpMiniEditor is password protected.

Figure 14: Password protected warning

💣 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.

Dialogue Stacks

You must also distribute the two stacks rrpHTMLToolbarFind.rev and rrpHTMLToolbarLink.rev with your application when the Link and Find buttons are included in the toolbar, which is the default.

These stacks are not password protected which means you can customize their appearance to match your application.

💣 If you modify the rrpHTMLToolbarFind.rev and rrpHTMLToolbarLink.rev stacks, do not change the names of any of the objects or the scripts in those objects.

Supported HTML

RRPHT does not support full HTML. It provides an editing toolbar to enhance the editing experience when using a Scrolling Field in a Revolution application. A property is added to the Scrolling Field object to let you access the text in the field as valid HTML.

The features of the Scrolling Field object are not extended by RRPHT, so the supported HTML subset is still significantly more limited than a full HTML editor. Instead of replacing the Scrolling Field with a different text object, RRPHT enhances what the Scrolling Field can already do. Understanding this is important when using the cHTMLSubset property.

These are the supported tags.

<p>, </p>
<b>, </b>
<i>, </i>
<u>, </u>
<ul>, </ul>
<ol>, </ol>
<li>, </li>
<h1>, </h1>
<h2>, </h2>
<h3>, </h3>
<h4>, </h4>
<h5>, </h5>
<h6>, </h6>
<font>, </font>
<a>, </a>

One way to get an understanding of how these tags are supported, is to enter and format text using the toolbar, and then with the cHTMLSubset property get the HTML. For example, this script puts the HTML in the Message Box.

```
put the cHTMLSubset of field "memoText" into msg
```

Where memoText is the name of the field object with the HTMLToolbar. Or you could copy it to the clipboard so you can paste it into a suitable editor for inspection with this script

```
set the clipboardData["text"] to the cHTMLSubset of field "memoText"
```

Handler Reference

This chapters documents all the handlers in the rrpHTMLToolbar stack. Each entry begins with the name of the handler and on the right whether it is a command, function or property. Next is the declaration of the handler as found in the rrpHTMLToolbar.rev stack. Properties do not show the declaration, only the name.

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. Handlers that are only used internally by the rrpHTMLToolbar stack are not listed here.

---

cButtonGroupPadding property

```
cButtonGroupPadding
```

Sets the number of pixels used to space groups of related buttons on the edit toolbar.

Value: a positive integer
Default: 4

Comment
The cButtonGroupPadding property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not affect the spacing of the buttons.

---

cButtonLeftOffset property

```
cButtonLeftOffset
```

Sets the number of pixels that the edit toolbar is positioned to the left or right of the left edge of the field.

Value: an integer number
Default: 0

Comment
The cButtonLeftOffset property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not affect the position of the toobar.

cButtonSize property

```
cButtonSize
```

Sets the size in pixels of the buttons on the toolbar.

Value: a positive integer
Default: 23

Comment
The cButtonSize property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not affect the size of the buttons.

cButtonStyle property

```
cButtonStyle
```

Sets the visual style of the buttons on the toolbar

Value: a valid style for a button object
Default: rectangle

Comment
The cButtonStyle property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not affect the style of the buttons.

cButtonTopOffset property

```
cButtonTopOffset
```

Sets the number of pixels that the edit toolbar is positioned above the top edge of the field.

Value: a negtive integer
Default: -4

Comment
The cButtonTopOffset property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not affect the positioning of the toobar. If this property is 0 the toolbar is adjacent to the top of the field with no space between them. Negative numbers increases the space between the toolbar and the field.

cHTMLSubset property

```
cHTMLSubset
```

Get this property to read the field as HTML. This is a special property of the field that the edit toolbar is attached to.

Value: valid subset of HTML
Default: empty

Comment
The cHTMLSubset property is created in the Initialize command and so cannot be read or written to before calling Initialize. This property is not part of the rrpHTMLToolbar stack, but is added to the field object as a virtual property.

cLicenseKey property

```
cLicenseKey
```

Set this property with a valid key to use the rrpHTMLToolbar stack in a standalone application.

Default: empty

Comment
The cLicenseKey property must be set to a valid key, otherwise the toolbar 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.

cLicenseName property

```
cLicenseName
```

Set this property with the name used when the License Key was purchased to use the rrpHTMLToolbar stack 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 toolbar will not work in a standalone application and an error message is shown.

cMenuWidthFactor property

```
cMenuWidthFactor
```

Sets the how many times wider the style and font menus are when compared to the buttons on the toolbar.

Value: a positive integer
Default: 6

Comment
The cMenuWidthFactor property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not affect the width of the menus on the toobar. By default the menus are 6 times the width of the buttons.

cShowBold property

```
cShowBold
```

Set this property to false to not include the Bold button on the toolbar.

Value: true or false
Default: true

Comment
The cShowBold property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cShowBulletList property

```
cShowBulletList
```

Set this property to false to not include the Bullet List (unordered list) button on the toolbar.

Value: true or false
Default: true

Comment
The cShowBulletList property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cShowFind property

```
cShowFind
```

Set this property to false to not include the Find button on the toolbar.

Value: true or false
Default: true

Comment
The cShowFind property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cShowFont property

```
cShowFont
```

Set this property to false to not include the Font menu on the toolbar.

Value: true or false
Default: true

Comment
The cShowFont property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the menu.

cShowItalic property

```
cShowItalic
```

Set this property to false to not include the Italic button on the toolbar.

Value: true or false
Default: true

Comment
The cShowItalic property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cShowLink property

```
cShowLink
```

Set this property to false to not include the Link button on the toolbar.

Value: true or false
Default: true

Comment
The cShowLink property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cShowNumberList property

```
cShowNumberList
```

Set this property to false to not include the Number List (ordered list) button on the toolbar.

Value: true or false
Default: true

Comment
The cShowNumberList property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cShowStyle property

`cShowStyle`

Set this property to false to not include the Style menu on the toolbar.

Value: true or false
Default: true

Comment
The cShowStyle property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the menu.

cShowUnderline property

`cShowUnderline`

Set this property to false to not include the Underline button on the toolbar.

Value: true or false
Default: true

Comment
The cShowUnderline property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cShowUnlink property

`cShowUnlink`

Set this property to false to not include the Unlink button on the toolbar.

Value: true or false
Default: true

Comment
The cShowUnlink property is only read during the call to Initialize and so should be set before calling Initialize if you want to change the default. Changing this property after the toolbar is created does not change the visibility of the button.

cVersionNumber property

```
cVersionNumber
```

Value: the version number of the RRP HTMLToolbar.
Default: 1

Comment
The cVersionNumber property is read-only and cannot be set. The version number will change with each new release of the stack.

Initialize command

```
command Initialize pField
```

Creates the toolbar at design time and initializes the internal state of the rrpHTMLToolbar stack at runtime.

Parameters
pField: the name of the field object that is the editor.

Comment
The Initialize command is called while in the IDE to create the toolbar. This can be done from the Message Box, or with a button as done in the Tutorial chapter. Then after the toolbar is created, the Initialize command must still be called during the startup of your application to initialize the runtime behavior of the HTMLToolbar.

RemoveToolbar command

```
command RemoveToolbar pField
```

Removes the toolbar from the field object.

Parameters
pField: the name of the field object that has the toolbar.

Comment
The RemoveToolbar command removes the toolbar from the stack but does not change any of the property settings or the scripts added to the field object.

ResetToolbar command

```
command ResetToolbar pField
```

Removes the toolbar from the field object and resets all the properties.

Parameters
pField: the name of the field object that has the toolbar.

Comment
The ResetToolbar command removes the toolbar from the stack and sets all the properties to the default settings. Scripts added to the field object to support the toolbar are also removed.

cUnlockCode property

```
cUnlockCode
```

Set this property with a valid code to use the rrpSpellCheck stack in a standalone application.

Default: empty

Comment
The cUnlockCode property must be set to a valid key, otherwise the toolbar will not work in a standalone application and an error message is shown. Refer to the Deploying Your Own Application chapter for more details. This property is an alternative to using the cLicenseKey and cLicenseName properties.

■■■■