

LongRange Runtime Version 12 (RV12)

Contents

Contents.....	1
New Emphasis on Prototyping.....	1
Form View DEMOPROTO (RPG) and LRDMPROTO (LANSA) allows annotations.....	2
Shipped RPG, COBOL and DDS example code has a new look.....	2
Native Message Boxes	4
New EZI layer (LongRange for RPG only)	5
Signature capture is now a native feature.....	5
Annotation of Images.....	7
New image element LD (Landscape) and PT (Portrait) Sizing Properties	9
Focus Control	10
Automatic Notification of Schema Changes	10
Right Swiping to display the Menu has been removed (iOS).....	11
Improved Search Area commands - Value, MaxLength, DataType, HasFocus added	11
Improved control of how G-Rows and G-Cols that span others are sized.....	13

New Emphasis on Prototyping

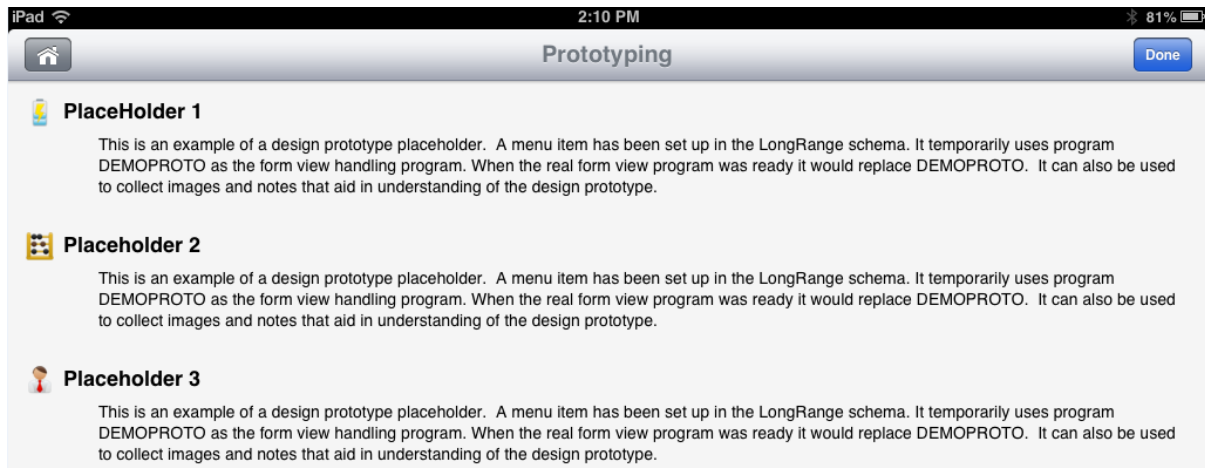
In latest RPG demo Programming Examples -> Prototyping has been added as a complete 'topic'.

This new emphasis is to foster the use of generic example form view DEMOPROTO and the creation of application prototypes.



Prototyping

Getting your mobile application design right is more important than coding it. One way to assess your design is by prototyping. To prototype a proposed form view use DEMOPROTO as your form view program. You can use DEMOPROTO in multiple places in your schema. It will display a different image for each location. Choose an image that shows what you propose to do on that form. Additional photos and notes may be used to help people viewing your prototype understand how it is going to behave.



[Form View DEMOPROTO \(RPG\) and LRDMPROTO \(LANSA\) allows annotations](#)
 DEMOPROTO's main image and attached images can all now be annotated.



[Shipped RPG, COBOL and DDS example code has a new look](#)

If you view shipped example RPG, DDS or COBOL source code you'll see a new look.



Some of the introductory examples include explanatory diagrams mixed in with the code

iPad 2:12 PM

Display the appropriate form view

```

Begsr DisplayFormView;
// Standard screen display logic
If (NOT IsNewForm);
  LRNG_SetSendChangesOnly();
Endif;
LRNG_Send();
/// Read and write the formview
Exfmt FORMVIEW;
//
LRNG_Receive();
LRNG_GetRequestedAction(RequestPROGRAM:RequestACTION);
Endsr;
/end-free
    
```

DDS format FORMVIEW defines a 'form view' - a grid layout base to build your App or

DDS Text Label

```

.....:10.....:20.....:30.....:40
Sex
Country
    
```

DDS field SEX

Uses the DDS HTML keyword '1' to define that it should appear as a *Dropdown*. The dropdown entries are directly defined in the DDS.

```

Sex 1 Not Specified
Country 1 Female
      1 Male
      1 Not Specified
    
```

DDS field COUNTRY

Uses the DDS HTML keyword '1' to define that it should appear as a *Dropdown*. The dropdown entries are directly defined in the DDS.

Endif;
Endsr;

Subroutine - Display this program's FORMVIEW to the mobile device user

```

Begsr DisplayFormView;
  // Write out and read back this program's 'Form View' layout (named F
  LRNG_Send();
  EXFMT FORMVIEW; // See below for layout details
  LRNG_Receive();
  // Get the requested action to be taken - and the program to handle i
  // This will either cause a loop around in this program or cause cont
  // to be returned back to the LongRange driver program LRNGDRIVER whi
  // will then call the requested program.
  LRNG_GetRequestedAction(RequestPROGRAM:RequestACTION);
Endsr;
/end-free
    
```

DDS format FORMVIEW defines a 'form view' - a grid layout base to build your App

The special &&F marker indicates fields starting in this column should expand to fill the device width

Text Literal

Input field ENTERTEXT

```

&&F
Type something in. Then touch the Submit command or use the keyboard's return key
&&F
&&F
&&F
    
```

Output field ECHOTEXT

The &&F marker and the use of grid layouts is covered in later examples. For the moment consider how even this very simple form view has to 'flex and flow' to accommodate a landscape mode iPad and a portrait mode iPhone:

You see the user interface 'flex and flow' model appearing everywhere now – from web sites you access via your PC's web browser to installed Apps that can function across the hugely different iPhones and iPads form factors. The days where a developer could just use a static WYSIWYG screen painter with absolutely position screen elements are now largely past.

Native Message Boxes

A new native message box capability has been introduced.

RequestACTION = "EXAMPLE_7"

Example 1 Example 2 Example 3 Example 4 Example 5 Example 7

Example 7

40 lines of text is extreme!

Example 4 line of text 2.

Example 4 text 3.

Example 4 line of text 4.

Example 4 text 5.

Example 4 line of text 6.

Example 4 text 7.

Example 4 line of text 8.

Example 4 text 9.

Example 4 line of text 10.

Example 4 text 11.

Example 4 line of text 12.

Option 1 Option 2

In LongRange for RPG see *Introductory Examples -> Message Boxes* (EXAM0091)

In LongRange for LANSAs see LREX0091

New EZI layer (LongRange for RPG only)

The EZI layer simplifies and abstracts commonly performed RPG and LongRange operations.

The EZI layer is presented by a service program named EZISERVICE in your LRNG_PROJ project library. The source code for EZISERVICE is included in LRNG_PROJ/QRPGLESRC source file.

The EZI layer is yours to use and modify as you see fit - the only provision is that you do not use it any way that competes with LongRange.

This initial introduction of the EZI layer is relatively low key. Expect the EZI layer to become more important and prominent in future versions.

Look out for the new “EZI” icon when looking at example RPG code

The image shows a screenshot of RPG code examples for EZI message boxes. The code is organized into four examples, each with a blue header and a red arrow pointing to the EZI icon in the code:

- Example 1 - Uses buttons defined as a menu in the application schema**: Shows code for `LRNG_SetProperties` with a menu name `EXAM0091_YES_NO_MENU`. A red arrow points to the EZI icon.
- EZI Example 2 - Uses the EZI simple OK message box**: Shows code for `EZI_OK_MessageBox`. A red arrow points to the EZI icon.
- EZI Example 3 - Uses the EZI 2 button message box with defaults**: Shows code for `EZI_TwoButton_MessageBox`. A red arrow points to the EZI icon.
- EZI Example 4 - Uses the EZI 2 button message box without defaults**: Shows code for `EZI_TwoButton_MessageBox`. A red arrow points to the EZI icon.

Two callout boxes on the right show the resulting dialog boxes:

- Example 2**: A simple dialog box with an "OK" button. Text: "This is the EZI_OK_MessageBox which can be set up and displayed by one RPG operation."
- Example 3**: A dialog box with "OK" and "Cancel" buttons. Text: "This is the EZI_TwoButton_MessageBox which can be set up and displayed by one RPG operation."

This example (EXAM0091) typifies the intention of the EZI layer.

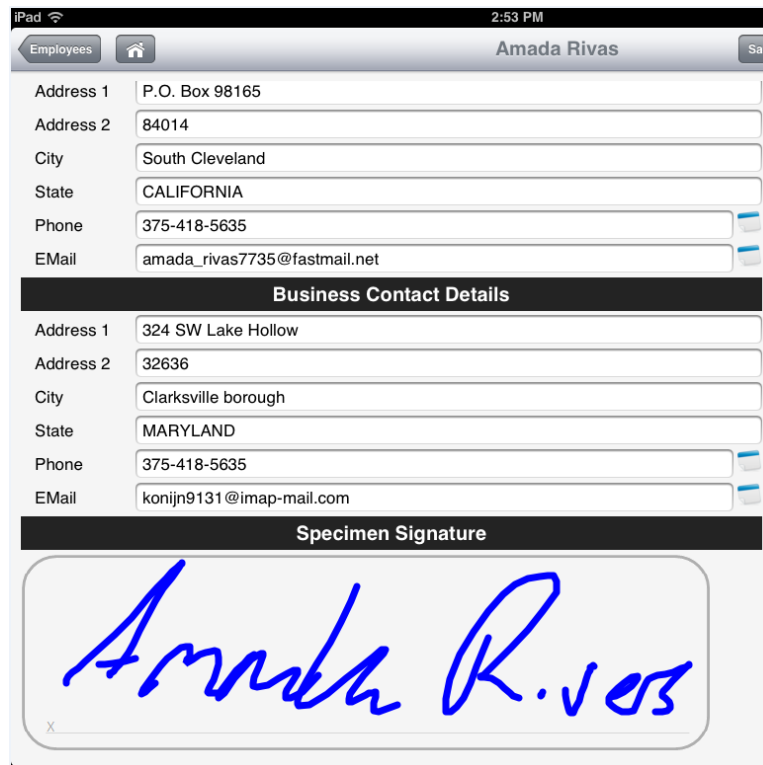
Example 1 is the 'low level' way of using the new message box capability.

Examples 2, 3 and 4 use the EZI layer to simplify and abstract the most common usages of the message box.

Signature capture is now a native feature.

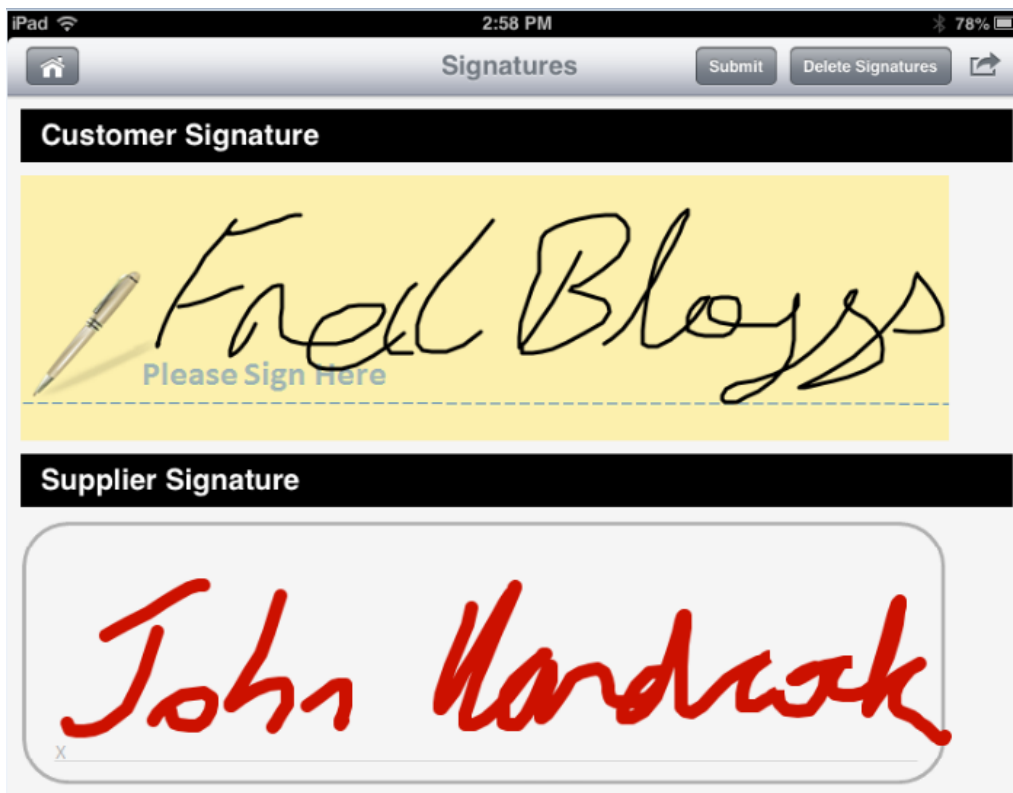
New properties added to the Image element allow signatures to now be captured natively.

In LongRange-RPG see the *Employees* demo ...



The screenshot shows an iPad interface for an application. At the top, it says "Employees" and "Amada Rivas". Below this are two sections of contact information. The first section, "Business Contact Details", includes fields for Address 1 (P.O. Box 98165), Address 2 (84014), City (South Cleveland), State (CALIFORNIA), Phone (375-418-5635), and EMail (amada_rivas7735@fastmail.net). The second section, "Specimen Signature", shows a handwritten signature in blue ink that reads "Amada Rivas".

And Use Case Examples -> Signatures (EXAM0093) ...



The screenshot shows an iPad interface for an application. At the top, it says "Signatures" and "78%". Below this are two sections of signature examples. The first section, "Customer Signature", shows a handwritten signature in black ink that reads "Fred Bloyss" on a yellow background with a pen icon and the text "Please Sign Here". The second section, "Supplier Signature", shows a handwritten signature in red ink that reads "John Handcock".

The existing RPG Advanced Web View Examples -> Signatures has now been deprecated ...

The screenshot shows an iPad application interface for 'Signatures'. At the top, the status bar displays 'iPad', signal strength, '3:02 PM', and '78%' battery. The app title 'Signatures' is centered, with 'Submit' and 'Next Set' buttons on the right. A red note reads: 'NOTE: The example has been superceded by "Use Case Examples" -> "Signatures" (EXAM0093). This example remains a good generic HTML <-> RPG example - but NOT as the best way to handle signatures.' Below this are two sections. The first section has input fields for 'Customer' (CUS4526193), 'Name' (ACME Engineering Plant 47), and 'Address' (47 Smith St, Sometown), followed by a yellow signature box with a pen icon and the text 'Please Sign Here'. The second section has input fields for 'Supplier' (SUP7363377), 'Name' (Warehouse Number 784), and 'Address' (784 Some St, Smithtown), followed by another identical yellow signature box.

In LongRange for LANSAs see the `Set_Image` method, the `Annotate_InPlaceEdit` and `Annotate_MergeWthImg` parameters and the `Save_Annotation` method for the same effect.

[Annotation of Images](#)

Photographs and other images may now be annotated.

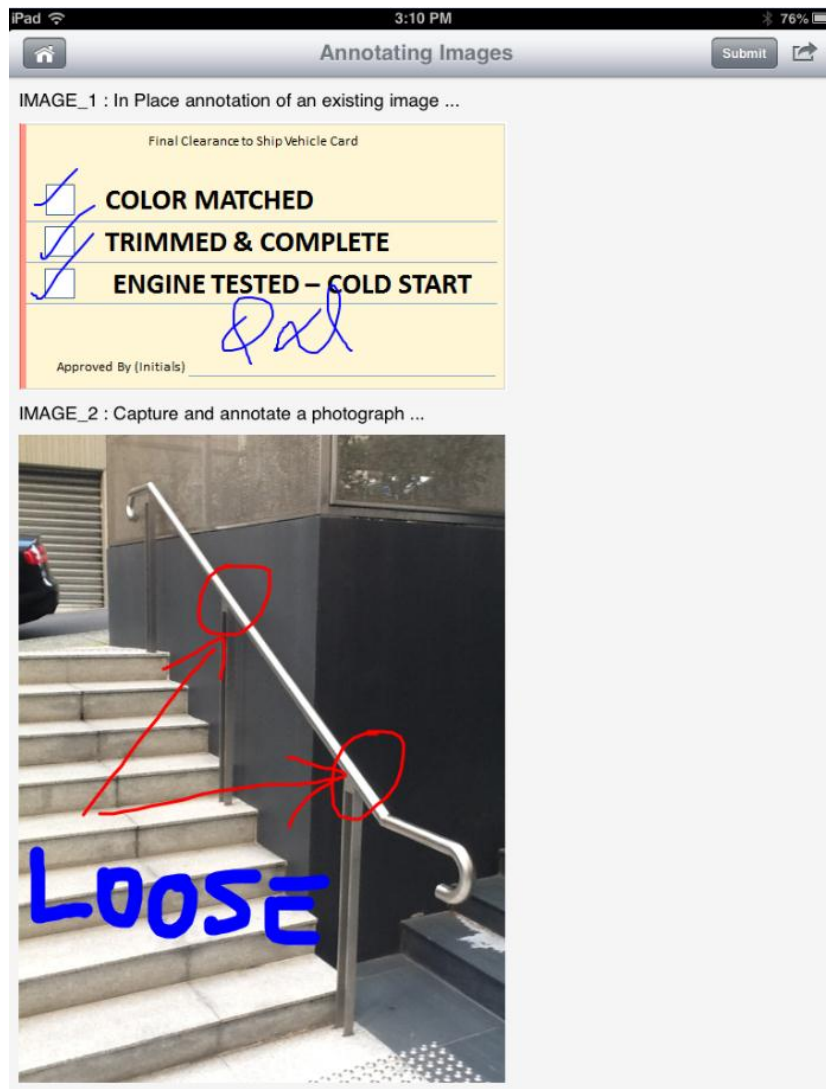


Image 1 shows an annotated image being used to replace a formerly manual vehicle sign off process.

Image 2 shows annotations making it quicker and easier for maintenance staff to locate and remove a workplace hazard.

The image annotation supports user or programmatic control of the annotation pen's nib width and the color of the ink used. See these LongRange-RPG examples for more on annotation:

- *Use Case Examples -> Annotating Images (EXAM0094)*
- *Advanced Examples -> Adding Value with Photos (EXAM0071)*
- Human Resources -> Incidents – Incident photos can now be annotated (eg: as above).
- Human Resources -> Assets – Asset Images can be annotated (eg: record asset damage).

See these LongRange for LANSAs examples for more on annotation:

- LRDMPROTO

New image element LD (Landscape) and PT (Portrait) Sizing Properties

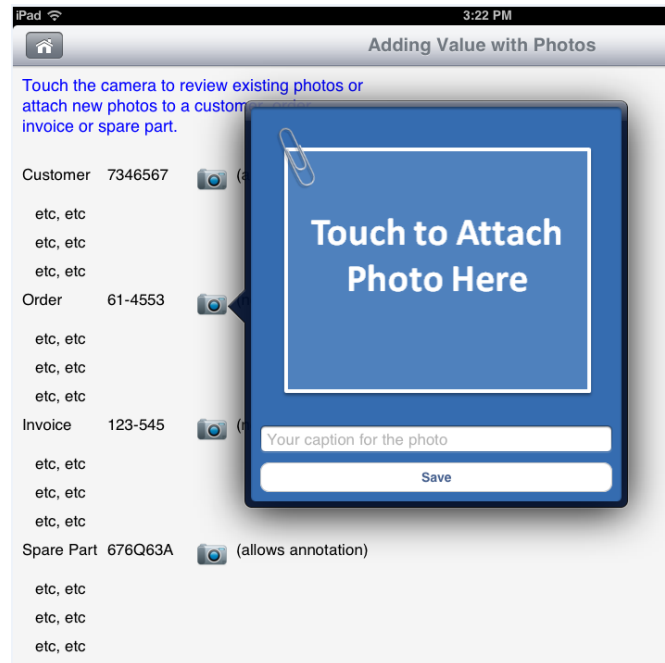
If you haven't encountered the landscape/portrait image sizing dilemma skip this section.

If you have - then you'll probably find these new LongRange for RPG LD and PT properties useful

<pre>NewImage.Resize.Width.LD NewImage.Resize.Width.PT</pre>	Use an alternative to NewImage.Resize.Width to specify the desired LD (Landscape) or PT (portrait) width only. Typically only an LD or PT pair are specified. When you specify an LD height/width pair the PT height/width pair are then defaulted as the reverse dimensions - ditto for an SD pair causing an LD pair to be defaulted. Requires client run time version 12 or later.
<pre>NewImage.Resize.Height.LD NewImage.Resize.Height.PT</pre>	Use an alternative to NewImage.Resize.Height to specify the desired LD (Landscape) or PT (portrait) height only. Typically only an LD or PT pair are specified. When you specify an LD height/width pair the PT height/width pair are then defaulted as the reverse dimensions - ditto for an SD pair causing an LD pair to be defaulted. Requires client run time version 12 or later.
<pre>NewImage.Thumbnail.Width</pre>	Specifies the width of the thumbnail to be created. The value can be an absolute value or a percentage value.
<pre>NewImage.Thumbnail.Width.LD NewImage.Thumbnail.Width.PT</pre>	Use as an alternative to NewImage.Thumbnail.Width for the same reason that NewImage.Resize.Width.LD & PT may be used. Requires client run time version 12 or later.
<pre>NewImage.Thumbnail.Height.LD NewImage.Thumbnail.Height.LD</pre>	Use as an alternative to NewImage.Thumbnail.Height for the same reason that NewImage.Resize.Height.LD & PT may be used. Requires client run time version 12 or later.

These properties have exact equivalents in LongRange for LANSA.


These new properties have been widely used across the shipped demonstration and example programs. For example, the RPG example EXAM0071 now no longer asks you to choose whether you are going to attach a portrait or landscape photo because it is using these properties



Focus Control

New properties support the better management of setting focus and the determination of what had the focus when a form view was submitted.

Text boxes, radio buttons, dates and drop downs now support a HasFocus property

<p>HasFocus</p> 	<p>Boolean. Readable and writeable.</p> <p>Textboxes, Dropdowns and Dates support HasFocus.</p> <p>Set to cause the element to receive the focus on the next display of the form view.</p> <p>Get to test whether the element had the focus at the last submission of the form view.</p> <p>Do not set the focus to multiple elements as this will cause undefined behaviour.</p> <p>To clear the focus on all elements set /Form.Focus to null. For example:</p> <pre>LRNG_AssignNullToProp('/Form.Focus'); // Clear all focus LRNG_SetBoolProperty('Form.Fields.NAME.HasFocus' : True); // Focus on NAME</pre> <p>Requires runtime version 12 or later.</p>
---	---

In LongRange for RPG see *Use Case Examples -> Managing Focus (EXAM0095)*

In LongRange for LANSA see *Use Case Examples -> Managing Focus (LREX0095)*

Automatic Notification of Schema Changes

If you update a LongRange schema (ie: an LXP file) on your server users should see this message soon after they perform their next action



Important Usage Notes:

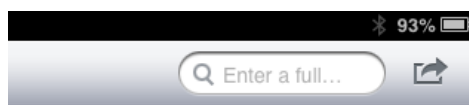
- If you backdate an LXP schema file it will not trigger this warning. Open and (re)save the backdated schema file. This is sometimes referred to as 'touching' the file to alter its timestamps.
- If the user replies Yes it is the equivalent of using the reload menu bar option.
- If the user replies No they will be prompted again in approximately 5 minutes – allowing them to complete any activity they may have been in the middle of.
- If you change images or html offline pages in the LongRange/Resource folder and need to make all clients load the new copy(s) use this mechanism simply touch the associate LXP file as previously described.

[Right Swiping to display the Menu has been removed \(iOS\)](#)

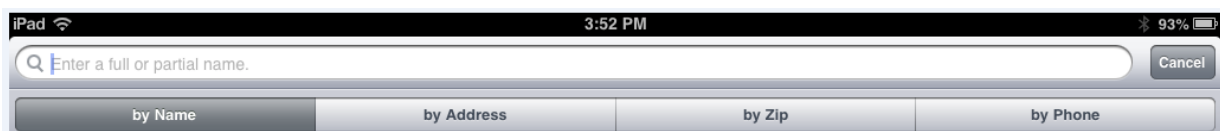
You can no longer right swipe to cause the menu to appear – you need to touch the home button. This produces consistent behaviour between iOS and Android and removes confusion over what swiping right will do or not do.

[Improved Search Area commands - Value, MaxLength, DataType, HasFocus added](#)

Search command menu items may be placed on the application title bar to facilitate application searching - like this:



Multiple search command menu items may be used – like this:



Search command menu items are defined by using LongRange Studio, where they are placed into the **Search Command Area** on the title bar:

Edit Menu Item Properties

Title:

Description:

By default, a menu item will use the text, icon, and description of the target object. You only need to provide the menu item properties that you want to change.

[Click here to see the text, description, and icon of the target object.](#)

Menu Item Symbolic Name

Symbolic Name:

[Show objects that are referencing this object](#)

Command Properties

Command Placement:

Display Style:

OK

Typically you give each search command menu item a unique **Symbolic Name**, and you indicate what **Action Name** the command sends to the server in the usual way :

Edit Operation

Operation Title, Icon & Description

Title:

Description:

Operation Type

Operation Type:

Operation Details

Action Name:

Operation Symbolic Name

Symbolic Name:

[Show objects that are referencing this object](#)

On iOS devices the **Description** specified acts as the placeholder text for the search field when it is empty.

If a search command menu item is symbolically named **EXAM0090_BY_NAME** (say) and it submits the requested action **SEARCH_BY_NAME** you find out when it is used like this

```
If (RequestACTION = 'SEARCH_BY_NAME');
```

And you retrieve what the user entered as the search value like this:

```
SearchValue = LRNG_GetPropAsStr('/CurrentOp.Param%1');
```

Programmatically you can also set these properties of **/Form.Menus.EXAM0090_BY_NAME.Search** :

Property	Description
MaxLength	Specifies the maximum allowable length of the search value.
DataType	Specifies the data type of the search value. This property supports the same values as the DataType property of the TextBox element.
Value	The value to be displayed in the search area. Not available on Android 2.3. Note that you can set the value of the search field using this property. However to get the value of the current search operation you use /CurrentOp.Parm%N property – most typically /CurrentOp.Parm%1.
HasFocus	Sets the focus to this search field. This is a write only property.

This feature requires client run time version 12 or later.

In LongRange for RPG see *Use Case Examples -> Searching* (EXAM0090) for more detailed search examples.

In LongRange for LANSA see LREX0090 and the Set_Menu methods parameters of the same names as above.

Improved control of how G-Rows and G-Cols that span others are sized

The new LongRange for RPG properties are:

Layout.DistributeMultiRow	<p>Specifies how the height of an element that uses Layout.Rows should be distributed amongst the rows it spans. Values are:</p> <ul style="list-style-type: none"> • NONE (default) - the element's height has no role in determining the height of the rows it spans. ie the spanned rows determine the height of the element. • FIRST - add excess height to first row spanned. • LAST - add excess height to last row spanned. • EVEN - distributed excess height evenly across all spanned rows. <p>For most situations LAST would be the most appropriate choice. Requires Runtime Version 12 or later</p>
Layout.Cols	<p>Specifies the number of columns to span.</p> <p>Use the special value -1 to indicate that the element should span across all the remaining columns in the row.</p>
Layout.DistributeMultiCol	<p>Specifies how the width of an element that uses Layout.Cols should be distributed amongst the columns it spans. Values are:</p> <ul style="list-style-type: none"> • NONE (default) - the element's width has no role in determining the width of the columns it spans. ie the spanned columns determine the width of the element. • FIRST - add excess width to the first column spanned. • LAST - add excess width to last column spanned. • EVEN - distributed excess width evenly across all spanned columns. <p>For most situations LAST would be the most appropriate choice. Requires Runtime Version 12 or later</p>

LongRange for LANSa has the parameters DistributeRows and DistributeCols available to all the element Set methods.