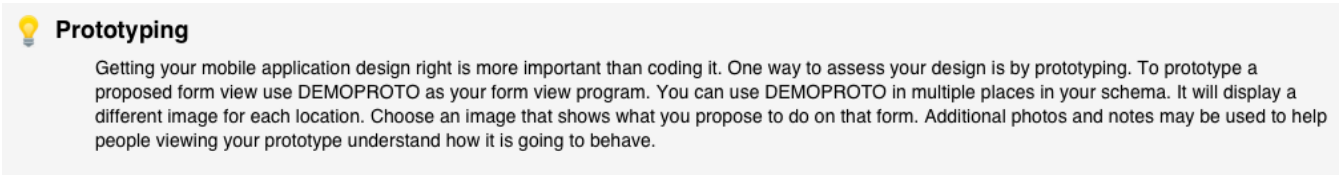
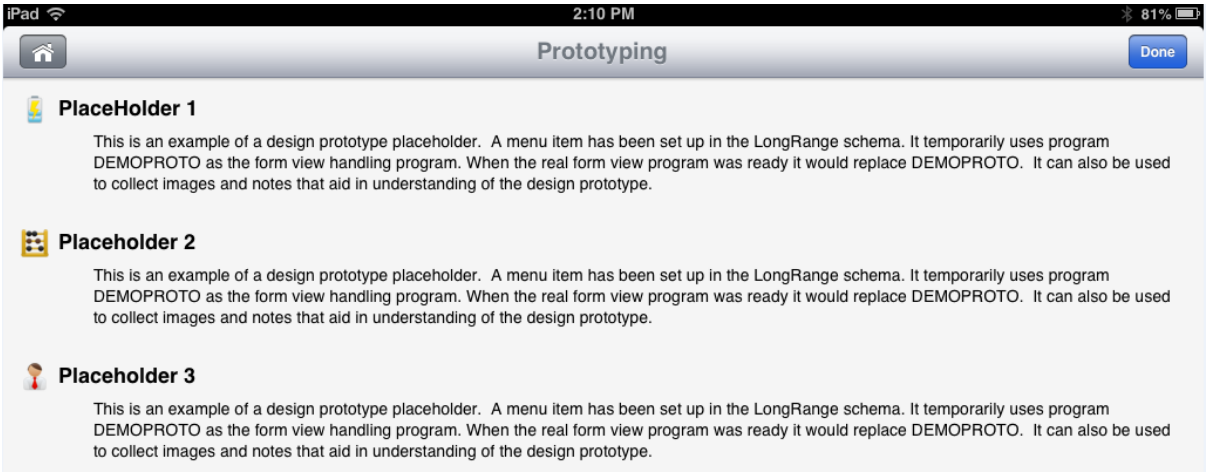
LongRange Runtime Version 12 (RV12)

Contents

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.





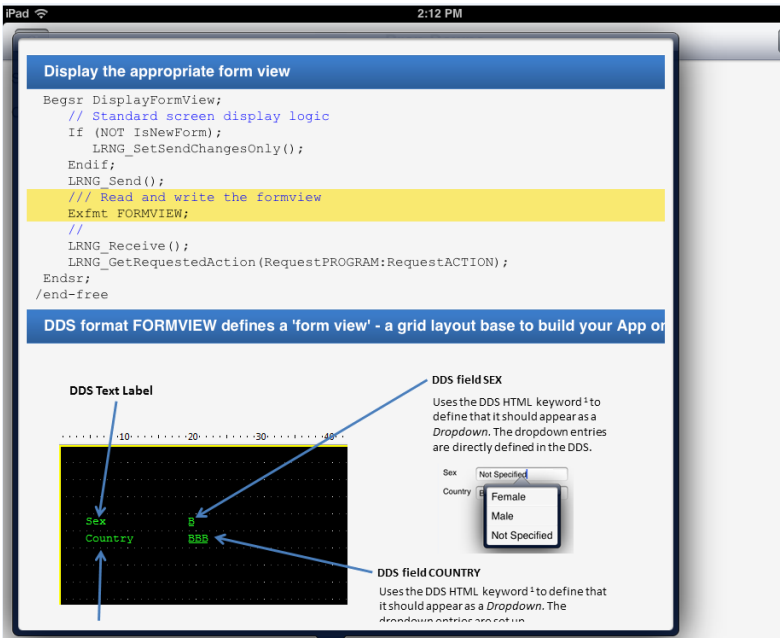
Form View DEMOPROTO (RPG) and LRDMPROTO (LANSA) allows annotations

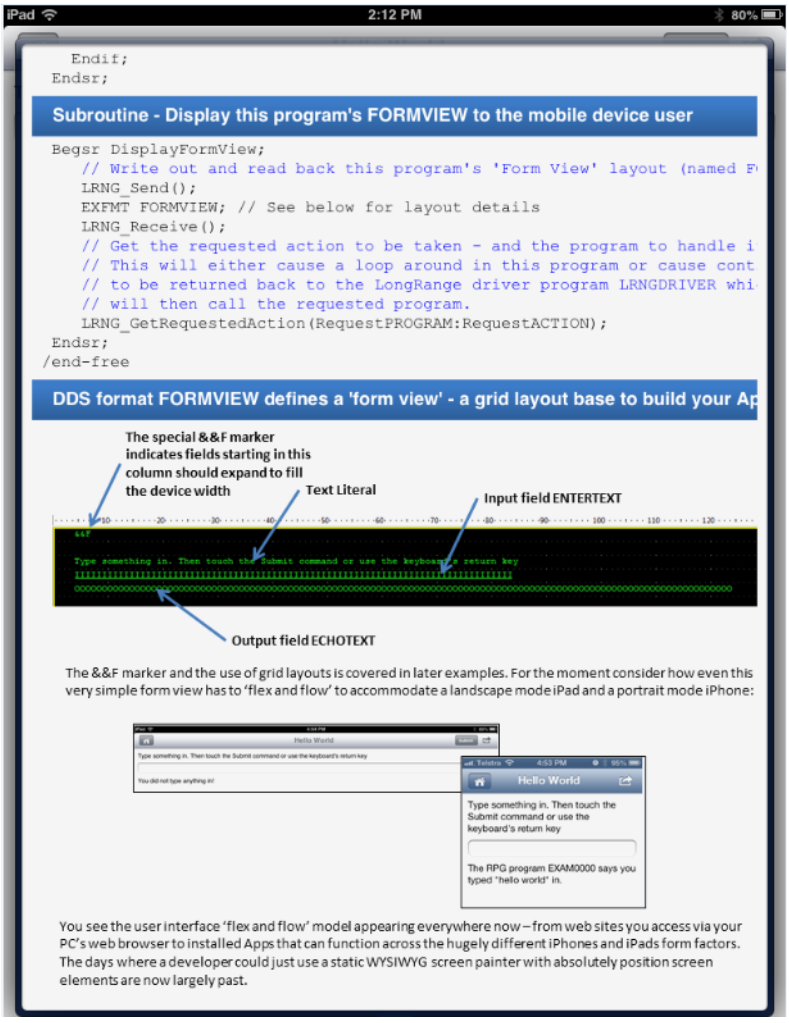
DEMOPROTO's main image and attached images can all now be annotated.

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.

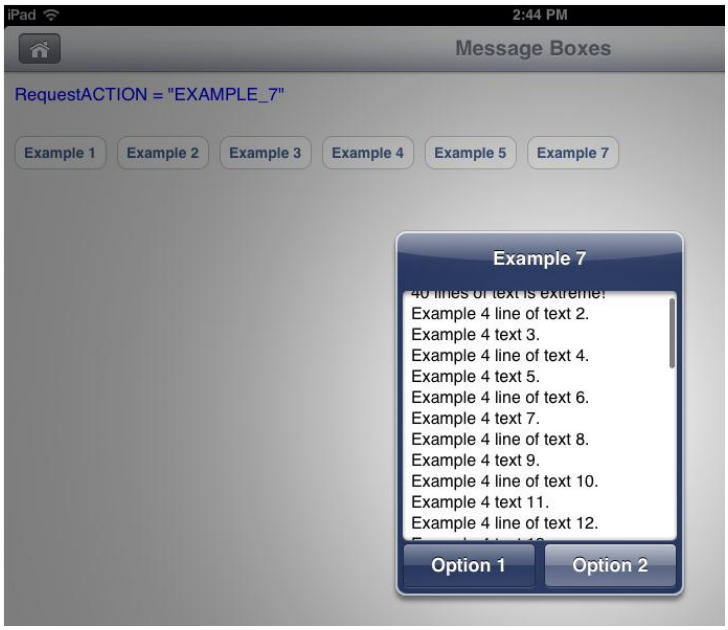
Some of the introductory examples include explanatory diagrams mixed in with the code …….





Native Message Boxes

A new native message box capability has been introduced.



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

In LongRange for LANSA see LREX0091

New EZI layer (LongRange for RPG only)

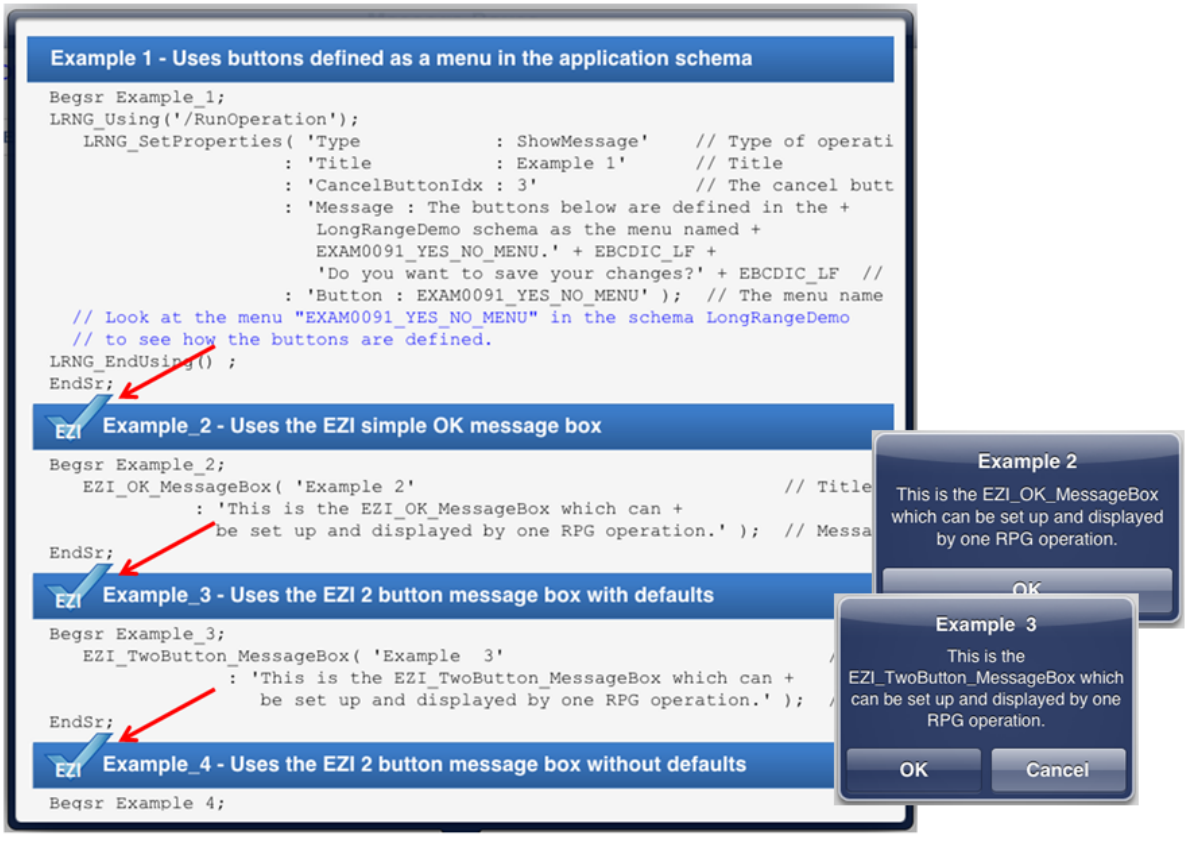
The EZI layer simplifies and abstracts commonly performed RPG and LongRange operations.

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 ……..



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

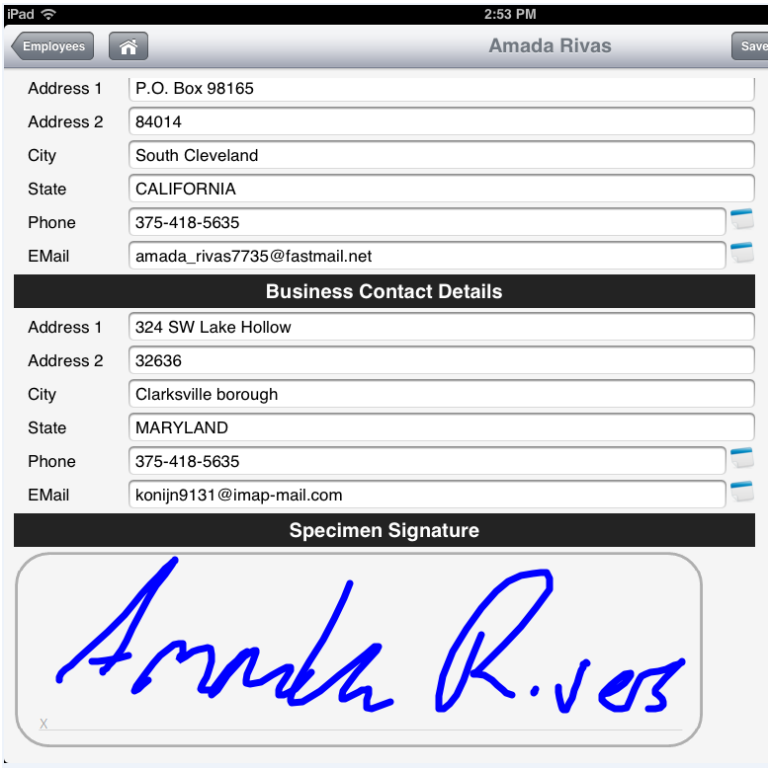
Example 1 is the 'low level' way of using the new message box capability.

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.

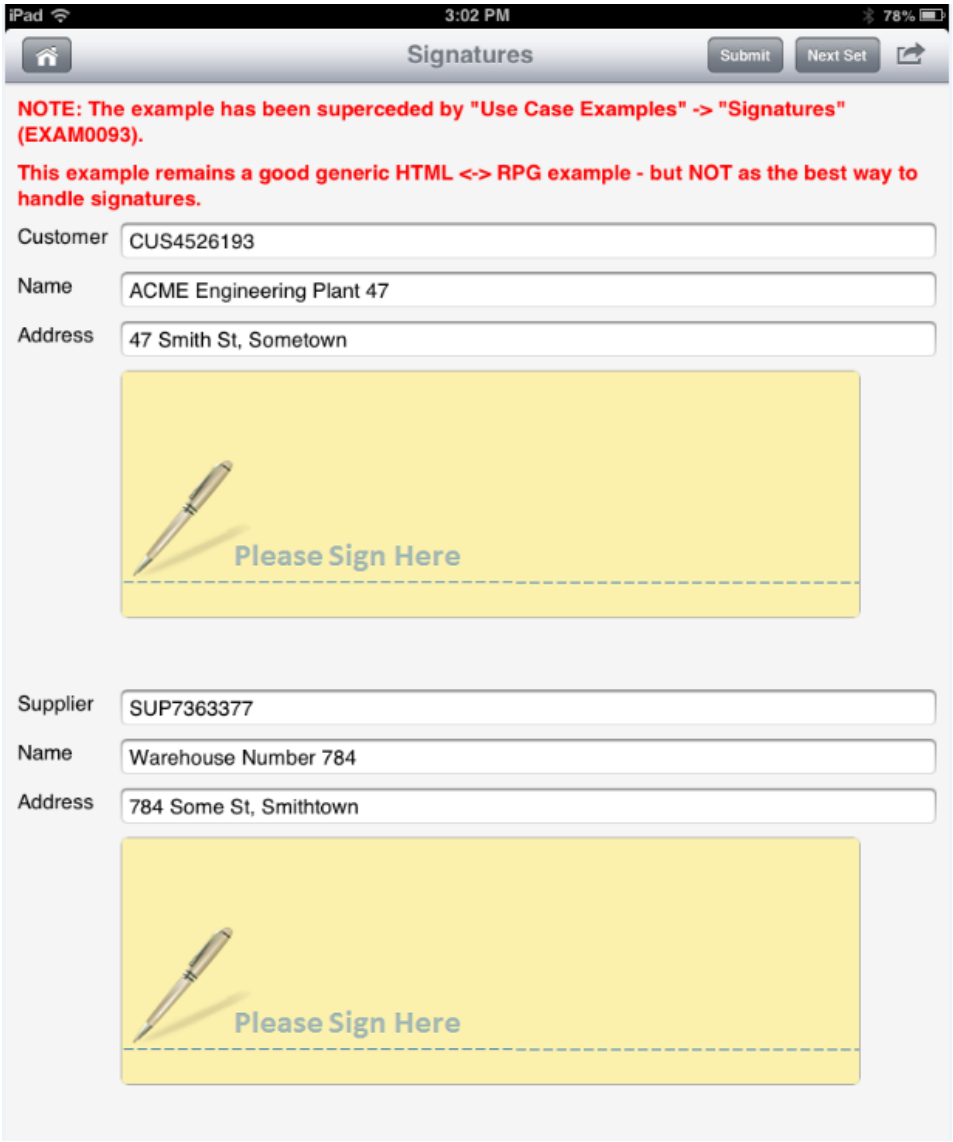
New properties added to the Image element allow signatures to now be captured natively.

In LongRange-RPG see the *Employees* demo …



And *Use Case Examples -> Signatures* (EXAM0093) …

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



In LongRange for LANSA 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.

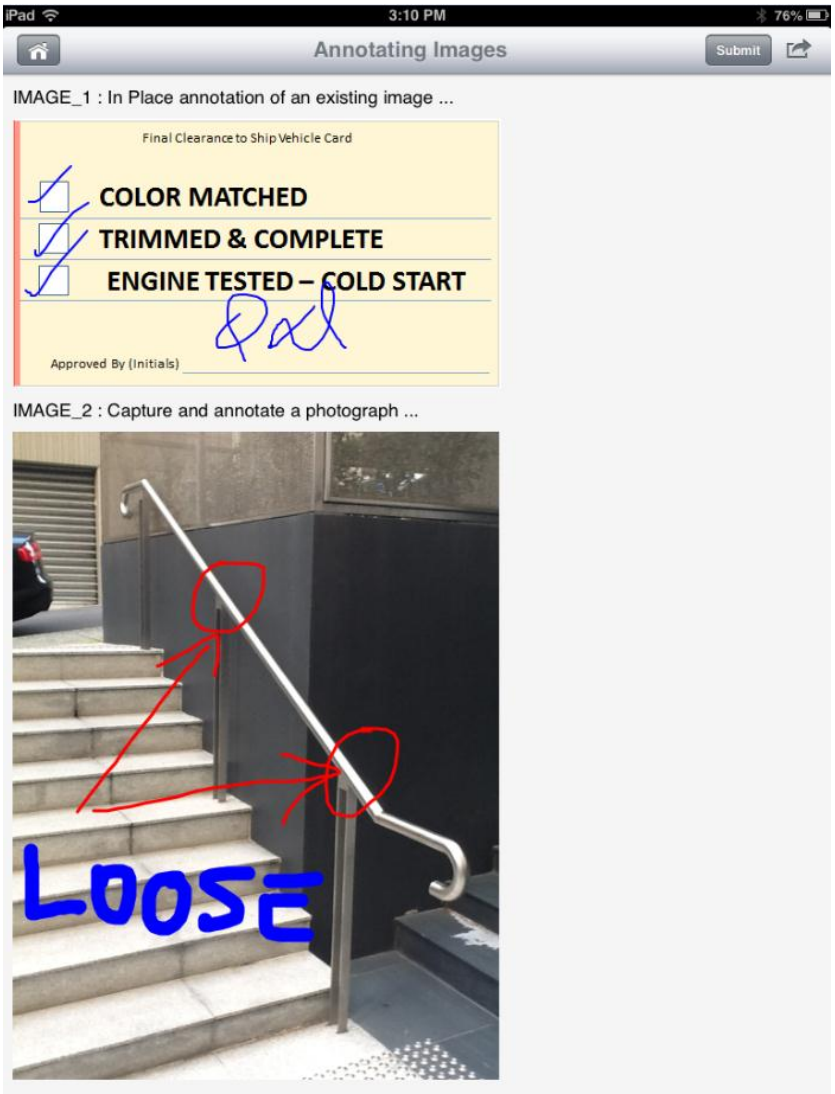


Image 1 shows an annotated image being used to replace a formerly manual vehicle sign off process.

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 LANSA 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 ….

`NewImage.Resize.Width.LD` `NewImage.Resize.Width.PT`	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.
`NewImage.Resize.Height.LD` `NewImage.Resize.Height.PT`	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.
`NewImage.Thumbnail.Width`	Specifies the width of the thumbnail to be created. The value can be an absolute value or a percentage value.
`NewImage.Thumbnail.Width.LD` `NewImage.Thumbnail.Width.PT`	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.
`NewImage.Thumbnail.Height.LD` `NewImage.Thumbnail.Height.LD`	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 dtermination of what had the focus when a form view was submitted.

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

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

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:

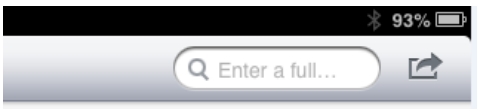
- If you backdate an LXP schema file it will not trigger this warning. Open and (re)save the backdated schema file. This is sometimes referred to as 'touching' the file to alter its timestamps.
- If the user replies Yes it is the equivalent of using the reload menu bar option.
- If the user replies No 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)

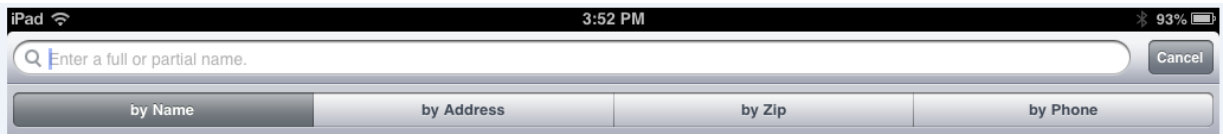
You can no longer right swipe to cause the menu to appear – you need to touch the home button. This produces consistent behaviour between iOS and Android and removes confusion over what swiping right will do or not do.

Improved Search Area commands - Value, MaxLength, DataType, HasFocus added

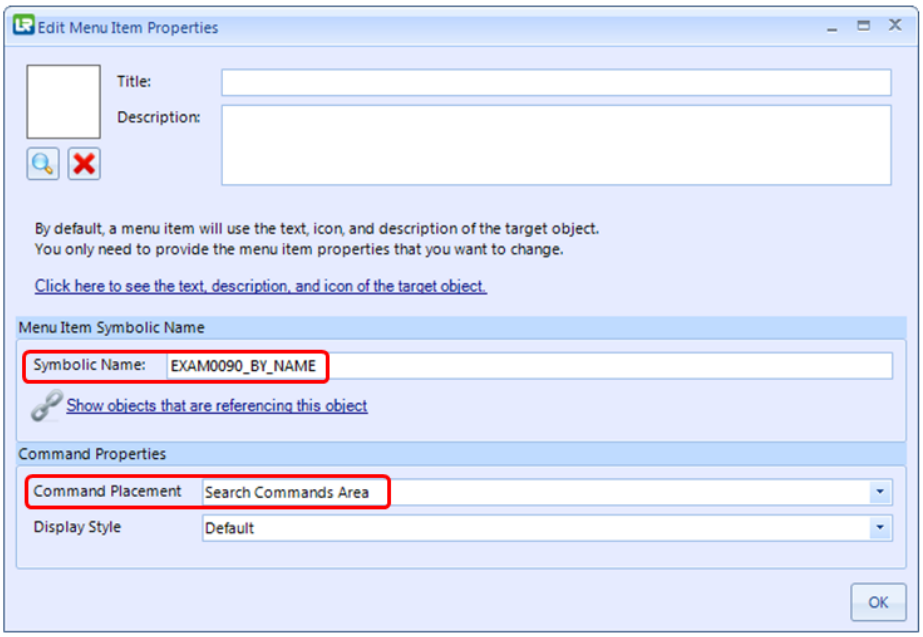
Search command menu items may be placed on the application title bar to facilitate application searching - like this:



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:



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 :

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 retreive what the user entered as the search value like this:

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

Programtically 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	Specifies how the height of an element that uses Layout.Rows should be distributed amongst the rows it spans. Values are: • **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** - distribute excess height evenly across all spanned rows. For most situations LAST would be the most appropriate choice. Requires Runtime Version 12 or later
Layout.Cols	Specifies the number of columns to span. Use the special value -1 to indicate that the element should span across all the remaining columns in the row.
Layout.DistributeMultiCol	Specifies how the width of an element that uses Layout.Cols should be distributed amongst the columns it spans. Values are: • **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** - distribute excess width evenly across all spanned columns. For most situations LAST would be the most appropriate choice. Requires Runtime Version 12 or later

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