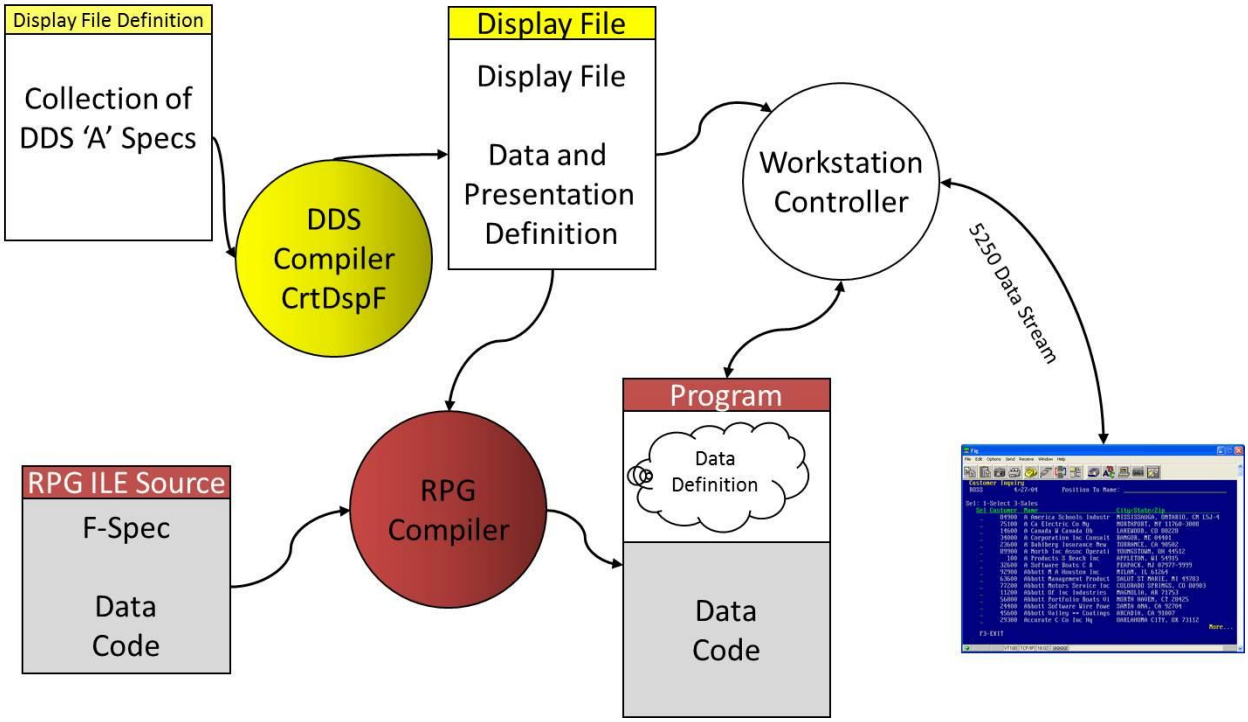
ASNA Mobile RPG
Mobile Display File Overview

Mobile Display File Overview

How an RPG Program Relates to a Traditional Display File

The Big Picture



To begin the process of showing a traditional Display File, the CRTDSPF command 'compiles' the DDS specs into a display file object. When the RPG compiler gets invoked, it takes the definitions of fields and formats from an existing display file and embeds them into the program alongside with the code found the RPG source.

At run time, the RPG program input/output operations direct data to/from a workstation controller (the Workstation function manager). The controller merges the program data with the presentation instructions of the display file to generate a 5250 data stream; green screen terminals or emulators can display the 5250 stream directly, or a screen scraper can manipulate the stream to provide alternate representations of the screens.

Traditional Display Files: Defining Both Data and Presentation

A Display File defines the format of the information to be presented on a display station, and how that information is processed by the system on its way to and from the display station. Data Description Specifications (DDS) describe the data referred to by a Display File.

A Display File is an object on the system; an object is a named storage space that consists of a set of characteristics that describe it and, in the case of a Display File, the data.

Before an application program can work with a display station, a Display File must be opened to allow data to flow between the program and the display station.

Since a Display File does not have a set of data uniquely associated with it, the relationship between the data and the Display File is established when the Display File is opens and ends when it is closes.

The file description, which is created at the same time as the Display File, describes the characteristics of the display file and determines how the Display File does the following:

- Controls the display station
- Formats output data from the program for presentation at the display station
- Formats input data from the display station for presentation to the program

A file description describes data at three different levels:

- File level
- Record level
- Field level

File level — Describes the set of records that make up the file. A record is an ordered set of one or more fields. A record-level description allows you to tell the system what a particular record looks like, or its *record format*.

Record level — Identifies which fields make up the record format and the order of the fields within the record format. The system can then perform separate operations on each field described with a field-level description in the record-level description. For example, one field can be highlighted while another is not.

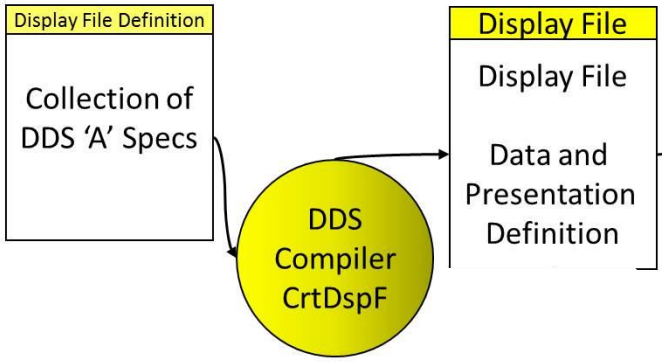
Since records are used to transfer data between the system and the application program, a record-level description is required for display files.

Field level — Fields are the smallest unit of data recognized and handled by the system. A field-level description allows you to give the system detailed characteristics of a field, such as:

- Where on the screen the field appears
- What type of data is valid for the field
- Whether the field should be highlighted in some way
- How it will be presented from the program to the system on output and from the system to the program on input
- Where each field is relative to the start of a record
- What the characteristics of each field will be while in the system
- Where the data for each field should be acquired from for output
- Where and how input from the display station should be placed so the program can use it
- Whether the field is an input-capable field or output-capable field only

Creating a Traditional Display File

All Display Files are created using the Create Display File (CRTDSPF) command. The input for the CRTDSPF command is a DDS source file. DDS can be created by hand in an RPG editor or with a tool (such as the built-in Screen Design Aid [SDA]).

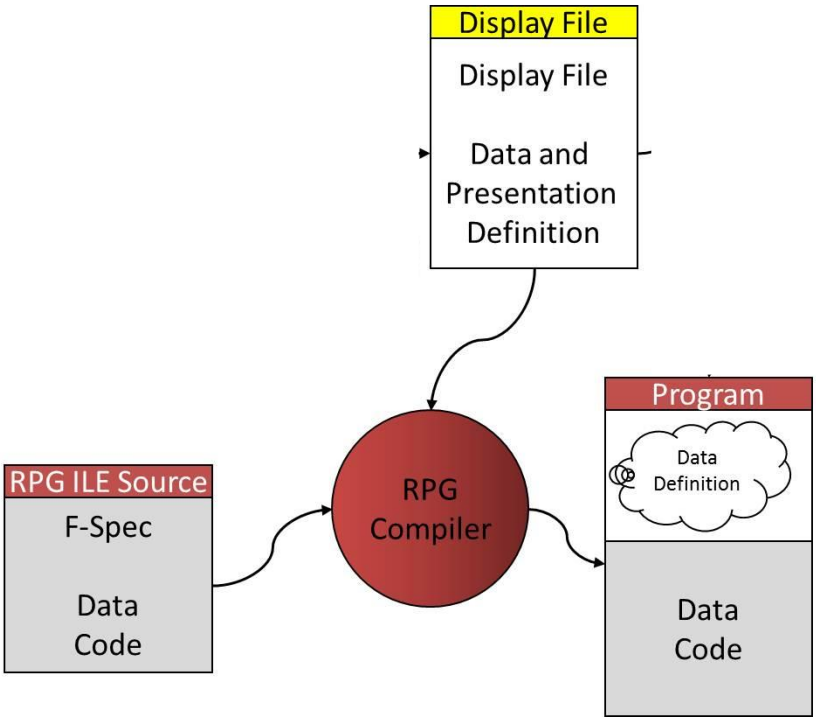


The CRTDSPF creates the display file in the library specified in the command.

Externally Described Display Files

Externally described data exists independent of any program that uses the file. Using externally described data, you can produce a detailed and standard description of both the Display File and any data that can be processed through the display file.

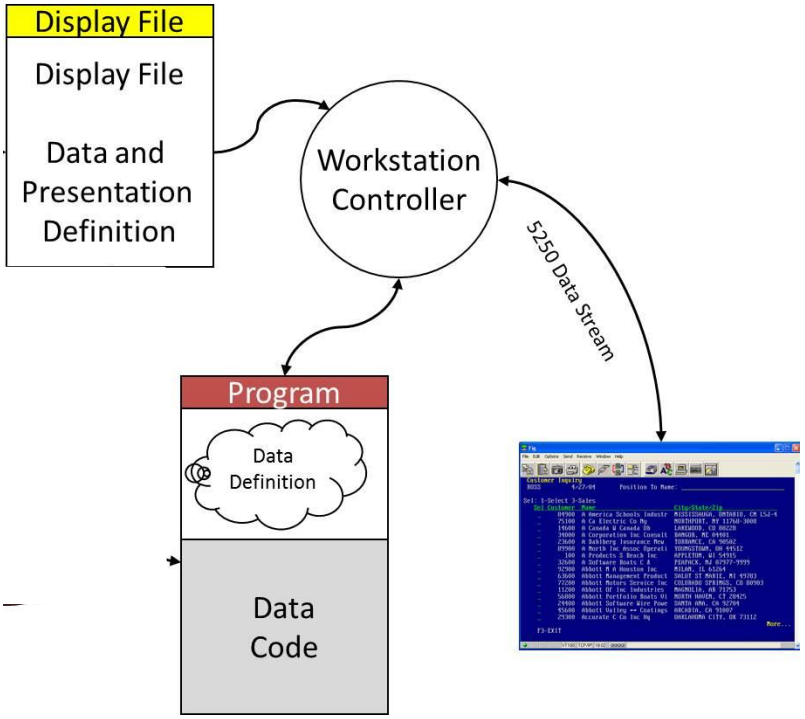
To use externally described data, you must declare that the Display File is to be used as an externally described file. The RPG compiler will then extract the file description from the display and incorporate it into the program.



As part of the compilation process, a hash value is inserted into the program code for each record format used in the program. Each hash value is like a signature summarizing the record format details significant to the program, mainly the names, types and sequence for each field found in the format. These hash values are known as Format-Levels.

Running the Program

At run time, the RPG program input/output operations direct data to/from a workstation controller (the Workstation function manager), the controller merges the program data with the presentation instructions of the display file to generate a 5250 data stream; green screen terminals or emulators can display the 5250 stream directly or a screen scraper can manipulate the stream to provide alternate representations of the screens.



Format-Level Checking

Level checking is an automatic method used when the program is run. It determines if the file description has changed since the program was last compiled. Depending on the type of change, the program may only need to be recompiled without modification.

Example — assume you compiled a program two months ago and, at that time, the Display File the program uses was defined as having three fields in a particular record. Last week another programmer added a new field to the record format, so that now the screen would show four fields. When the old program runs and tries to open the file, the system notifies your program that a significant change occurred to the file's definition since the last time the program was compiled.

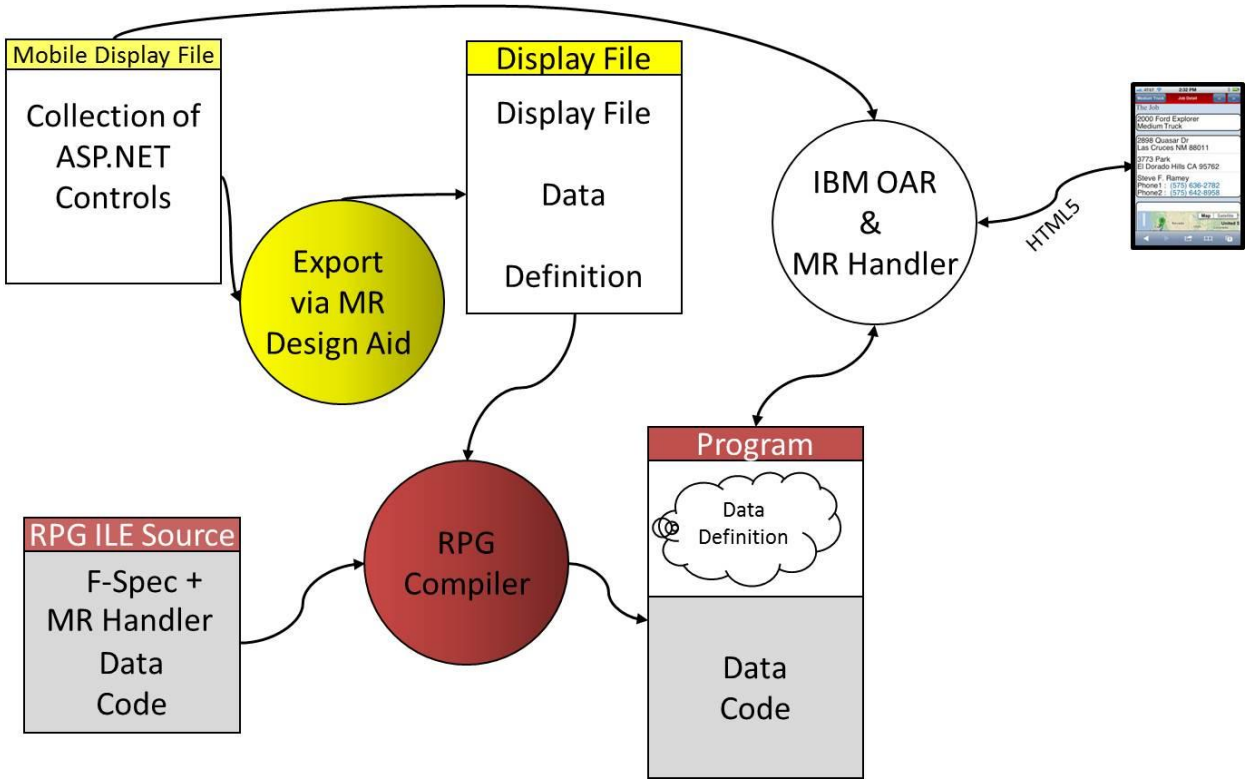
This notification is known as a **record format level check**.

How MR Changes the Picture

The New Picture

In version 7.1 (and PTFed back to 6.1) IBM i systems gained the exciting feature of being able to provide an RPG program an alternate way of handling the I/O request issued to a Workstation File. This feature was packaged under the name IBM Rational Open Access RPG Edition (OAR for short).

OAR formalized a mechanism to specify a "handler" program name in the F-Spec for a Workstation file that will be invoked at execution time in place of the workstation controller. A display file is still used at compilation time, but not necessarily at run time. The following diagram shows this new setup when using Mobile RPG.



The main differences introduced to the source of an application are: First, the RPG code needs to be told, via a keyword on the file declaration F-spec (in the lighter maroon block on the left), the name of the program or procedure (ASNAMOBILE) that will handle the I/O requests at run time. Second, the display file is only used to obtain the data definitions at compile time, any presentation instructions are unused. Instead, most of the presentation instructions come directly from the Mobile Display File.

The handler, in effect, uses the ASP.NET controls from the Mobile Display File and the data from the program to generate HTML5 (as well as other code that Web Browsers can interpret, such as CSS and Javascript). While a modern Web Browser is required to interpret the new output, the web app can easily be configured as an accessible icon or tile on the mobile device's homescreen. It's also a straightforward matter to configure the app to claim the full screen by default; doing so frees up space that would otherwise be taken up by an address bar at the top or a status bar at the bottom.

MR Display File Role: Defining Data and Presentation

The RPG compiler is not aware that the Mobile Display File exists and it has no way of connecting the Workstation file, defined in an F-spec, with the Mobile Display File defined in the Windows system. So in order for the compiler to know about the record formats and fields present in the user interface, a traditional Display File must exist in the system.

Mobile RPG provides a facility to create a traditional Display File on the IBM i system that defines all the information the compiler needs to know about the data formats of records and fields from a Mobile Display File. Since this Display File is only used to satisfy the external description of the Workstation file in the RPG program, it contains the minimum required information, namely format and field names and types.

Creating an MR Display File

Mobile Display Files are created in the Mobile RPG Design Aid, an extension of Microsoft Visual Studio 2012. The MR Design Aid provides facilities to create Mobile websites and reflect (or export) any record format or field modification back to the IBM i so that the RPG ILE compiler can process the modified field or format. This step is performed on a Windows development machine, rather than directly on a Mobile Device.

Drag/Drop Some Controls to an .ASPX Page

The Visual Studio Toolbox holds a long list of "Controls." Each Control is an object that can be added to the Mobile Display File. Doing so is as simple as clicking on the desired control and dragging it into the main pane of the Mobile RPG Design Aid.

Invoke the Export Facility

After a Mobile Display File is constructed, it must be 'exported' back to the IBM i to create the traditional Display File that will be used as the external description of the Workstation File defined in an F-spec. To create the traditional Display File, a user has to make use of the Mobile Design Aid's Export facility. This can be accomplished by right clicking on the file and selecting **Export to Display File** from the context menu.

Location of New File (On the IBM i)

The new Display File (created once the Mobile Display File is exported back to the IBM i) will be stored in a library that the user selects during the Export process.

Format-Level Checking

As described previously, the IBM i provides a facility called Level checking to automatically determine, when the program is run, if the file description has changed since the program was last compiled. Depending on the type of change, the program may be allowed to run as is, it may only need to be recompiled without modification, or — if a more significant change has occurred, for example a record or field being removed — the program may need to be modified to adapt to the change before execution is permitted.

Cosmetic changes, like changing the color of a field or adding a constant label, are insignificant to the program and require no alteration of the traditional Display File. Changes that affect the size or position of a field within the record will prevent the program from running, in this case the Mobile Display File should be exported to the IBM i to create a new version of the traditional Display File, and the RPG program should be recompiled. If a more significant change is made (such as adding or removing complex components like charts or maps) the RPG program should be modified to take advantage of the new elements introduced in the display file and avoid using any that have been removed, in addition to exporting the Mobile Display File, before recompiling it.

MR Display File

In a Mobile RPG application, the IBM i Display File is essentially replaced with an ASPX (web) page that contains the new, mobile-ready display information. This ASPX page is known as a Mobile Display File; it is composed of numerous components (some of which handle content, some of which handle presentation), all of which come together to make a coherent, mobile-friendly user interface.

There are three aspects to a Mobile Display File:

- The definition of the file using some programming source
- The visual representation on a user's screen
- The data formats accessible to the RPG program via the Workstation File

The interaction between the RPG program and the Mobile Display File also happens in three areas:

- Data interchange between the RPG fields and the Display File fields
- The Display File components affect the setting of the RPG indicators and vice versa, the indicators of the RPG program are affected by some components of the Display File. In DDS, the former indicators were conditional indicators while the later ones were response indicators.
- The RPG program can supply an informational data structure via the INFDS keyword in the Workstation File's F-spec. This data structure is populated with data about the result of every operation on the file, after each operation completes.

MR Display File Is Composed of a Set of Record Formats

Each MR Display File contains a set of Record Formats. Like in DDS, there are three kinds of records: Subfile, Subfile Control, and plain records.

Typically, a mobile application will present more than one screen to the user. In general, each screen for the Mobile device will be composed of a single plain record or a subfile.

Each Record Format serves as a container for Controls: The basic unit of the Mobile RPG design experience. Controls are used to define fields and other user interface components.

Defining a Control

A "**Control**" is an ASPX construct that represents a single "item" in the MR Display File, or an ASPX page in general. Anything that a designer puts in the MR Display File will be placed there by adding a control.

A Control typically corresponds to a user interface element that displays on the screen. Controls can be represented on the screen as text, images, buttons or some other widget, depending on the type of control.

Some Controls contribute fields that are present in the data record format and are accessible to the RPG program. For instance, the DdsCharField defines a character field on the data portion of the record format and is typically represented visually as text on the screen, although it may be presented on the screen as a checkbox or a dropdown box.

Controls Have Properties

Every control has "Properties." The Properties of a control define much of how a control looks (height, width, color, visibility) and what it contains. Essentially, anything a designer does to a control that isn't removing it entirely, they do by modifying Properties.

Some properties can be set programmatically, and most of the ASNA Controls allow programs on the IBM i to set some of their properties; other properties are reserved to be established at design time.

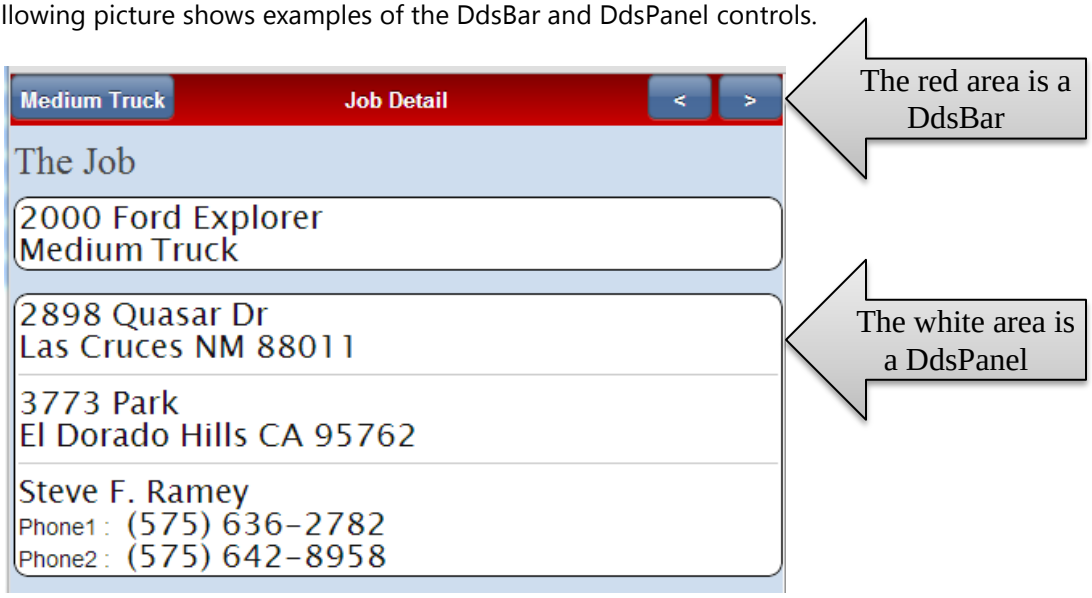
The MR Design Aid provides a window where property values can be established. Below is a sample properties window showing some properties for an ASNA DdsImage control:

Some Controls Contain Other Controls

Certain controls can contain other controls. For example, the DdsBar control is used to present to the user a navigation bar at the top of the screen. This Control can contain buttons, character fields, or both as the designer chooses.

Another example is the panel (DdsPanel) control which is used to group a set of fields and other controls into an area of the screen. This control has a property that can let the developer control, with RPG indicators, whether the panel, with all of its contained controls, is to be shown on the screen or not.

The following picture shows examples of the DdsBar and DdsPanel controls.



ASNA Controls for MR

ASNA provides several controls that designers can use to create the MR Display File. Some of these are closely analogous to those found in traditional Display Files, while others introduce the functionality of a fully-featured Mobile App.

Analogous to DDS

Several Controls are very similar to portions of traditional Display Files. There are controls that are akin to record formats, fields of different types like character and numeric. There is a control that allows the developer to specify which function keys are available throughout the display file. Like in traditional Display File, constants can be added to the display file. Many of the controls have properties (like visibility, protection, and color) which can be conditioned with indicators provided by the RPG program, and some properties can affect the resulting indicators passed back to RPG.

Record Formats

We have already mentioned that Mobile RPG provides these three kinds of record formats as controls:

- Plain Record Format (DdsRecord)
- Subfile Control Record Format (DdsSubfileControl)
- Subfile Record Format (DdsSubfile)

There are several characteristics that are common to all three of these record formats. Regardless of kind, each format:

- Has a name that identifies it uniquely to RPG
- May contain other controls

In addition, Plain and Subfile Control record formats may:

- Define which function, or attention, keys are enabled and the indicator they turn on if the user activates them. Each function key can be conditioned with an Indicator expression.
- Provide hidden fields to receive the name of the field that was active when the user submitted its data back to the application.
- Establish a navigation bar and a footer bar (see DdsBar below).

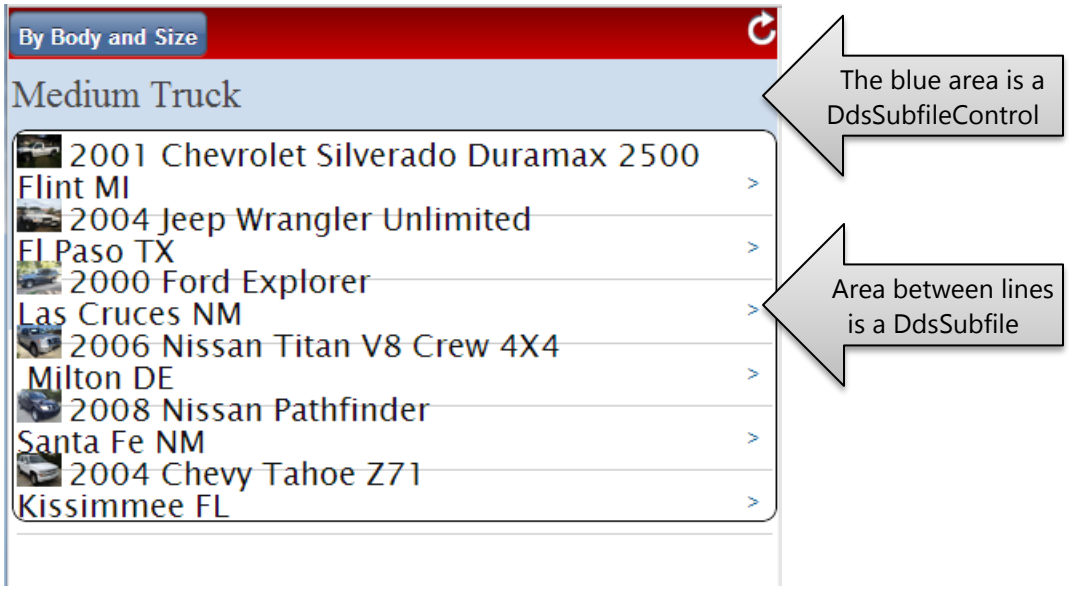
Subfiles

A subfile is a group of records that have the same record format and are read from and written to a display station in one operation. Subfiles are useful when multiple similar records that are related must be displayed.

Each subfile you describe in your Mobile Display File requires adding two controls:

- A subfile record format (**DdsSubfile** control).
  The subfile record format control serves as a container for the fields in one row of the subfile. The RPG program uses the subfile record format to read a subfile, write new records to a subfile, and update the subfile. Operations on the subfile record format are performed between the subfile and RPG program; the mobile device screen is not changed by operations on a subfile record format.
- A subfile control record format (**DdsSubfileControl** control).
  The subfile control record format is a container for the subfile. Additionally, it contains heading information and controls subfile functions such as size, initialization, and clearing. The RPG program performs operations on the subfile control record format to write the subfile to the mobile device screen and to read the subfile from the screen. Some of the most important properties of the subfile control are:
  - **SubfileSize** — specifies the size of the subfile.
  - **SubfilePage** — specifies the number of records in the subfile to be displayed simultaneously.
  - **DisplayRecords** — specifies when to begin displaying records in a subfile.
  - **ClearRecords** — specifies when the subfile will be clear of all records before new records are to be written.
  - **DisplayFields** — specifies if the fields in the Subfile Control, not the actual subfile records are to be shown on the screen.

The following picture shows a subfile control and its corresponding subfile:



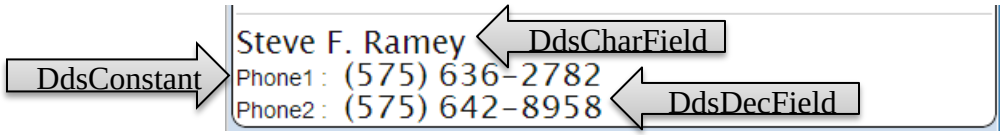
Fields

Mobile RPG provides a set of controls that represent traditional fields in a display file. These are:

Control	Field Type	Type/Format properties	Notes
DdsCharField	Character	Length DBCS type and length	Can be displayed as a textbox, a checkbox, or a dropdown box.
DdsDateField	Date	DateFormat: ISO, USA …	User can enter a date via the Device's calendar widget
DdsDecField	Decimal	Type: Binary, Packed , Zoned Length, Decimals	Edit codes and edit words can control presentation format
DdsDecDateField	Decimal	Type: Binary, Packed , Zoned Length, Decimals DateFormat: ISO, USA …	The field is presented to RPG as a decimal field, but to the user as a date widget
DdsTimeField	Time	TimeFormat: ISO, USA …	
DdsTimestampField	Timestamp		

All fields have also a set of properties that control their behavior and look; here are some of the most significant:

Property	Notes
Alias	Provides the field name to be used by the RPG compiler when defining this field to the program
ChangeInd	Specifies which indicator to set on when this field changes
Color	Specifies the color of a field
ErrorMessage **ErorMessageId**	Identify a message to be associated with this field when the field is in error
MessageId	Identify at run time the messages identifier to be used with this field
PositionCursor	Gives focus to this field when the screen is presented to the user
Protect	Protects the contents of field from user input
Usage	Specifies if this field is: output-only, input-only, input/output (both), hidden, or program-to-system
VirtualRowCol	The Row and Column reported to the RPG program if this field has the focus when the user submits his data to the program
VisibleCondition	Provide a set of conditional indicators determining whether the field is visible and rendered



Constants

In addition to field controls, Mobile RPG provides the DdsConstant control to add labels to a record format. Labels are always shown as text to the user and have no effect on the RPG program. The program can affect some of the characteristics of the constants by setting indicator values that are used in properties of the constant. Some of the more common properties of constants are:

Property	Notes
Color	Specifies the color of a field
Text	The text to be shown on the screen
VisibleCondition	Provide a set of conditional indicators determining whether the text is visible and rendered

Containers and Controllers

Now let's look at some controls provided by Mobile RPG that start to depart somewhat from what is available to the traditional Display Files.

Button

The DdsButton control is rendered to the user as a button with text, as a link, or as a clickable image. When the user touches the button, the screen's data is submitted back to the RPG program. Button properties determine where the 'cursor' was assumed to be at the moment of the submission and which function key is reported to have been activated.

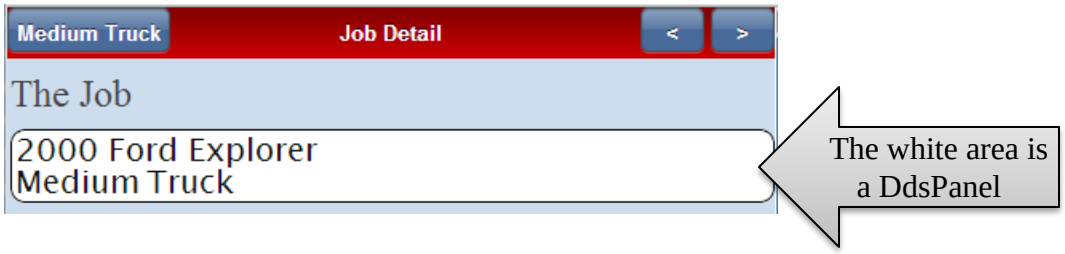
A button may define, via the TextField and TextFieldLength, a character field which is included in the record format presented to RPG. It is through this field that a program can set the text to be presented to the user. Other interesting properties are:

Property	Notes
AidKey	Attention Identifier (Function) Key to report back to RPG, via indicators or through the feedback information data structure (INFDS).
ButtonStyle	It can be one of: Button, Link, Image.
FieldName	The name of the field to be reported back to RPG as having 'focus' when the screen was submitted.
Image	The path to the image used as the button. This path points to a location on the Web server file system.
Text	Define the text to be shown on the screen. This text is a constant established at design time.
TextField & TextFieldLength	These properties define a character field whose runtime value is to be shown on the screen. If these properties are given, the **Text** property is ignored.
VirtualRowCol	The Row and Column reported to the RPG program when this button is touched and the screen is submitted.

Panel

A DdsPanel control is used to visually group a set of controls in an area of the screen. A panel may define, via the GroupingTextField and GroupingTextFieldLength, a character field which is included in the record format presented to RPG. It is through this field that a program can set the title of the panel to be presented to the user.

The other interesting property of a panel (shared with some other controls) is VisibleCondition which allows the developer to provide a logical expression, involving indicators, controlling whether the panel and all of its contained controls is visible and rendered to the user screen.



Navigation Bar with its Segments

The DdsBar control is provided by Mobile RPG to facilitate the creation of navigation bars. These navigation bars are typically shown on the top or bottom of a screen and help the user move through the application.

A DdsBar is divided horizontally into segments of equal size. Each segment in turn can contain controls. Controls in the segments are typically buttons, links, text constants, and fields.

The controls of a segment can be aligned to the left, right, or center. For example, the bar shown below is divided into three segments.

The first segment is aligned left; it contains a single DdsButton showing text drawn from a field. The second segment aligns its contents to the center; it holds the DdsConstant 'Job Detail'. Finally the third segment aligns its contents to the right; it contains two DdsButtons with images pointing left and right.

The DdsRecord and the DdsSubfileControl controls have optional properties that point to a DdsBar to be used as a navigation bar or as a footer bar. The navigation bar is always rendered at the top of the screen and the footer at the bottom of the screen. When the user scrolls the contents of the screen, the navigation and footer bars remained 'attached' to their edge of the screen. Notice how on the picture below the contents of the format have scrolled 'under' the DdsBar.

Controls with Multiple Fields and Record Formats

So far we have looked at controls that have a one-to-one relationship with a single record format (DdsRecord, DdsSubfileControl, and DdsSubfile) in the RPG program or with a single field. There is a group of controls which define more than one control in the record format presented to the RPG program or that may even define the two record formats used by a subfile.

Link

By default, the DdsLink control is rendered as a hyperlink widget on the user's screen. Hyperlinks require two character fields to operate: first, the text shown to the user (in many applications this text is shown underlined, and with a distinct color); second, the location of the page to be shown when the user touches the hyperlink. This second value is given as a URL (Uniform Resource Locator).

For example, the text to be shown to the user could be 'Search Engine' and the URL could be 'www.google.com'. When the user touches the text, Mobile RPG opens the default internet browser to the specified address.

The DdsLink can also be used to provide links to phone numbers, email accounts, texting accounts, and other "profiles" such as Facebook or Skype accounts, interfacing with the apps installed on the phone.

Here are the DdsLink properties used to establish the two core values and other interesting properties:

Property	Notes
ApplicationType	This property defines the type of application that the link is intended to call when tapped: Browser, Mail, Telephone, Text, or Explicit (a type of application defined within the property, such as Facebook or Reddit apps).
LinkStyle	The property defines whether the DdsLink will appear as a traditional

	hyperlink or as an Image.
Image	This property defines the path to the image used for the link.
TextFieldName & TextFieldLength	These properties define the field whose value is to be shown on the screen.
UrlFieldName & UrlFieldLength	These properties define the field whose value is to be used as the target URL of the hyperlink.
VisibleCondition	Provide a set of conditional indicators determining whether the text is visible and rendered.

Image

One of the popular capabilities of mobile devices is their ability to take and show pictures. There are some applications that take advantage of these capabilities to help businesses improve their processes. Mobile RPG provides the DdsImage to facilitate the manipulation of images within the application.

The DdsImage makes it very easy to send to the mobile device an image stored in the IFS of the IBM i where the application is running. The control offers the below properties to point to the IFS file location and name:

Property	Notes
Directory	The path to the Directory in the IFS. This is a constant value established at design time.
DirectoryField & DirectoryFieldLength	These properties define a character field whose runtime value is used as the path to the directory holding the image file. If these properties are set, the **Directory** property is ignored.
Name	The name of the image file. This is a constant value established at design time.
NameField & NameFieldLength	These properties define a character field whose runtime value is used to get the name of the image file. If these properties are set, the **Name** property is ignored.
ImageLocation	This property sets the location of the image to HostIFS, ThisWebSite, or ExternalSource.
MaxImages	This property sets the maximum number of images that can be displayed in this DdsImage control (useful for controlling bandwidth).

For example, an inventory processing application could rely on pictures stored in the IFS on the directory '/Items/Pictures', each item could have its image stored in a .JPG file named as the Item's ID. If the developer wanted to show the user the item's image, he could add a DdsImage to the record format and set the following properties like this:

Property	Value
Directory	'/Items/Pictures'
NameField	Item_Image
NameFieldLength	20

This would define a character field of length 20 called Item_Image. Before issuing the EXFMT to the record, the developer would move the file name to the Item_Image field, something like this:

```
*...1....+....2....+....3....+....4....+....5....+....6....+....7...+....
```

```
CL0N01Factor1+++++++Opcode(E)+Factor2+++++++Result++++++++Len++D+HiLoEq....
 *
C      ITEMID          CAT(P)      '.JPG':0       ITEM_IMAGE
C                      EXFMT       MYREC
```

The name of the file may contain the wild characters '*' and '?'. If a wild carded name is provided, Mobile RPG will search the directory for all image files that satisfy the name pattern. If more than one file is found, then the DdsImage will render as a group of images. For instance, the picture below used the following values:

Property	Value
Directory	'/Vehicles/Pictures'
NameField	ImgName
NameFieldLength	32

```
*...1....+....2....+....3....+....4....+....5....+....6....+....7...+....
CL0N01Factor1+++++++Opcode&ExtExtended-factor2++++++++++++++++++++++++++++++
 *
C                      EVAL        ImgName = %trimr(VHID) + '-????.jpg'
C                      EXFMT       MYREC
```

All JPG files whose names were formed by concatenating the vehicle Id (VHID field) with a dash and 4 characters would appear on the screen. If the vehicle id was 'ABC', files named ABC-0001.jpg, ABC-0002.jpg, ABC-WXYZ.jpg, or ABC-A1B2.jpg would be selected (if present in the 'Vehicles/Pictures' directory).

DdsImage also allows the option of giving users the power to upload new images. This is accomplished with the following properties:

Property	Notes
UploadCondition	An indicator expression enabling the upload capability of the control.

UploadCaption	Text to be shown prompting the user to upload an image.
UploadKey	Function Key reported to the RPG program when the user touches the Upload button.
UploadedNameField & UploadedNameFieldLength	These properties define a character field where the name of the uploaded file will be made available to the RPG program.

When the control is rendered, the user will see a set of widgets on the screen that will allow him to select an image from the gallery of this device or to take a picture if the device has an integrated camera. When the user is satisfied with the chosen image, he will be able to touch a button to start the upload process.

The name given to the file is the one specified in the **Name/NameField** properties, if the value given contains wildcards, Mobile RPG will generate a new name following the pattern of the wildcard name with random numbers used in place of the wild cards. The name used by Mobile RPG will be reported back in the field defined in the **UploadedNameField** property. The location of the new file will be the one specified in the **Directory/DirectoryField** properties.

Following the example above, if the user loaded a new vehicle picture, the name of the new file could be ABC-1287.jpg (remember that the number is randomly generated).

HostFile

Just like the files holding images in the IFS can be sent to the device with DdsImage, other kinds of files in the IFS can be sent with the DdsHostFile control. For example, a PDF file holding a copy of an invoice could be made available to the app's user.

The DisplayType property of DdsHostFile controls how the file sent to the device is shown to the user with the following options:

- Show a hyperlink to display the file in a separate instance of the browser
- Show a hyperlink to display the file in the current window
- Embedded in a frame within the page

These are the DdsHostFile significant properties:

Property	Notes
Path	Full path to the file in the IFS. This is a constant value established at design time. The path includes all directories and file name.
PathField & PathFieldLength	These properties define a character field whose runtime value is to be used as the full path to the IFS file. If these properties are set, the **Path** property is ignored.
Text	A constant used as the hyperlink's text
TextField & TextFileLength	These properties define a character field whose runtime value is to be used as the text for the hyperlink. If these properties are set, the **Text** property is ignored.

Map

The ability to display maps on a mobile device has opened up new opportunities for applications to serve data in a more meaningful way to its users. Google has created a service to dispense map images for almost any region of the world. The Mobile RPG DdsGMap control facilitates the creation of

applications that take advantage of Google mapping services. Depending on the type of application being created, it may be necessary to obtain a service contract with Google in order to use their service.

The DdsGMap presents to the user a Google Map with optional markers, routes, and terrain views. To the RPG program, the DdsGMap presents itself as a Subfile. Each record of the subfile represents an address to be included in the map.

For example, a subfile with the following four records could display a driving route between the address contained in the subfile as shown next:

RRN	Location
1	2898 Quasar Dr., Las Cruces, NM 88011, USA
2	Rio Grande Gorge Bridge, Taos, NM, USA
3	Hoover Dam Bypass, Boulder City, NV, USA
4	3773 Park Dr., El Dorado Hills, CA 95762, USA

Some of the more interesting properties of the DdsGMap that interact with the RPG program are:

Property	Notes
SubfileName	The name of the subfile record format used to for the map.
SubfileControlName	The name of the subfile control record format used to for the map.
ClearIndicator	The number of the indicator to be used to clear the records in the subfile.
AddressField & AddressFieldLength	These properties define the subfile character field where the program will provide the address to be included in the map.

These properties are used to control the way the map is rendered:

Property	Notes
MapType	The type of map to be shown can be: Roadmap, Satellite, Hybrid or Terrain
MarkTypes	The map can show a Pin for each address or it can connect the addresses with a Route.

Chart

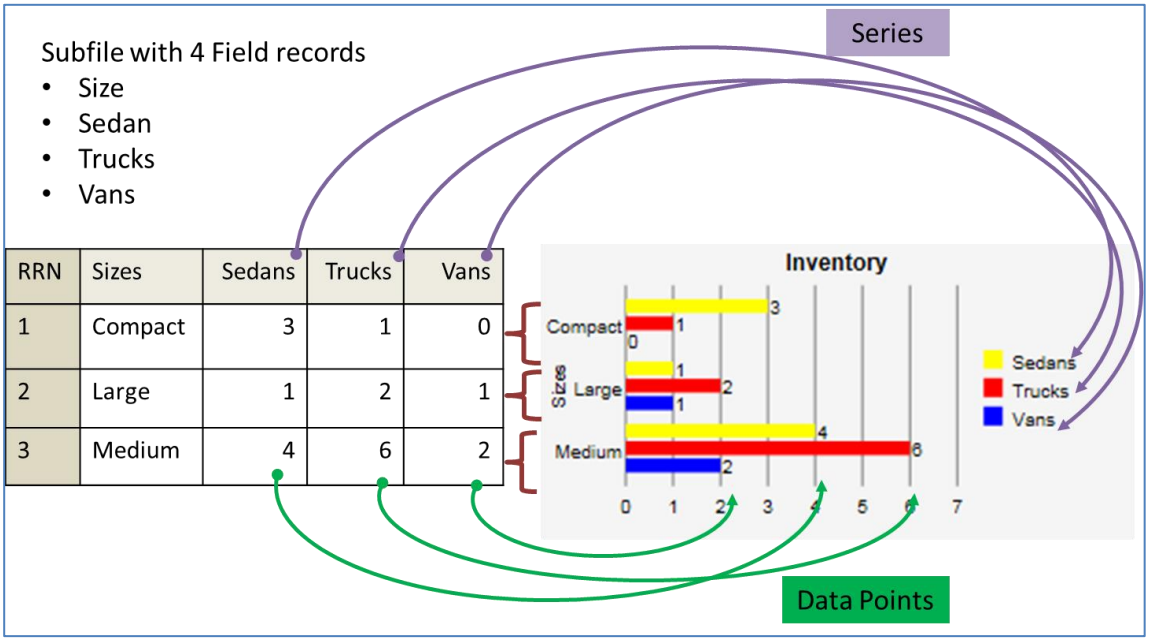
For certain situations, data displayed in a chart is much easier to read than data shown in a table. The DdsChart control facilitates the task of turning tabular data from a subfile into a chart.

Take for example the data in the following subfile with information regarding vehicles:

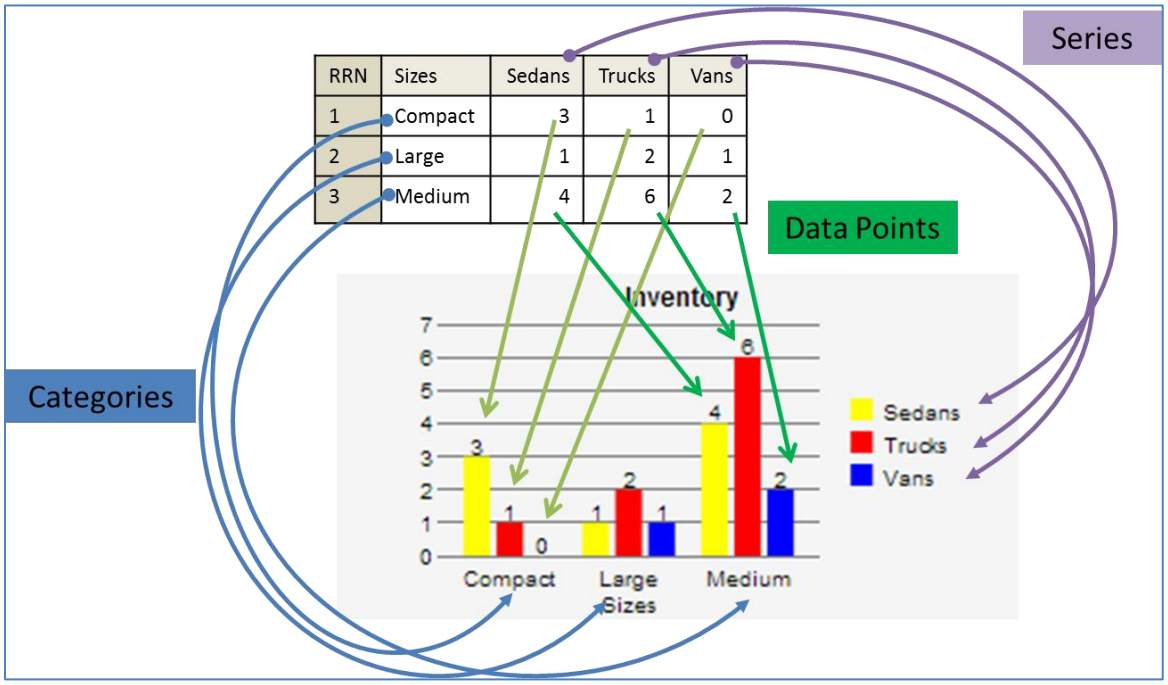
RRN	Sizes	Sedans	Trucks	Vans
1	Compact	3	1	0
2	Large	1	2	1
3	Medium	4	6	2

These are three series containing counts for types of vehicle: Sedans, Trucks, and Vans. Each of these is categorized according to the vehicle size: Compact, Large, and Medium. There is a subfile field for the names of the category entries called Size and one field for each one of the series. The category field type is 'character' while the series fields are 'numeric'.

The information on the subfile could be shown as a bar chart as follows:

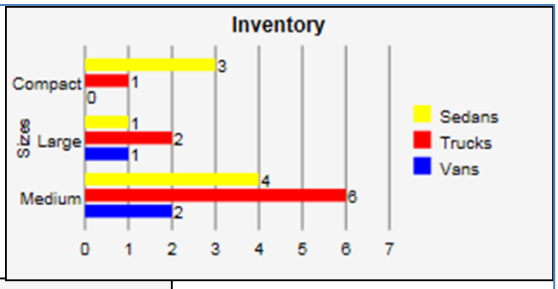


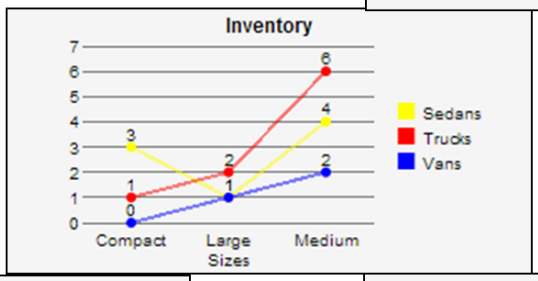
Or the same information could be shown as a column chart like this:

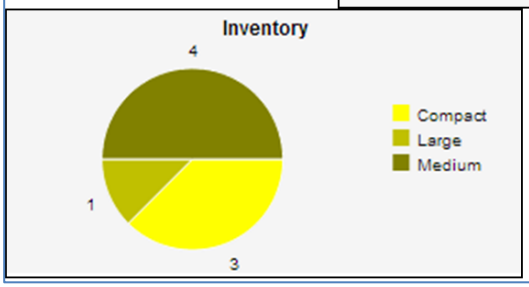


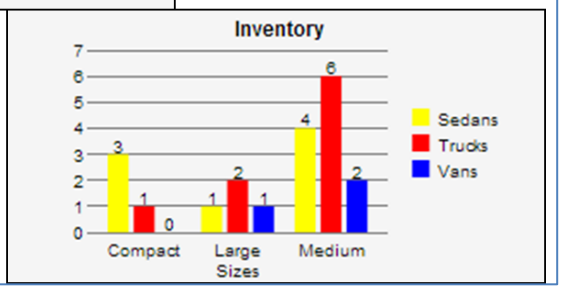
The kind of chart is controlled by the DdsChart **ChartType** property which can be one of: Column, Bar, Line, or Pie. The actual chart type is irrelevant to RPG; the program still sees all of the data as a subfile.

RRN	Sizes	Sedans	Trucks	Vans
1	Compact	3	1	0
2	Large	1	2	1
3	Medium	4	6	2









Here is a group of interesting properties for the DdsChart:

<table>
<tr><th>Property</th><th>Notes</th></tr>
<tr><td>SubfileName</td><td>The name of the subfile record format used to for the chart data.</td></tr>
<tr><td>SubfileControlName</td><td>The name of the subfile control record format used to for the chart data.</td></tr>
<tr><td>ClearIndicator</td><td>The number of the indicator to be used to clear the records in the subfile.</td></tr>
<tr><td>CategoryField & CategoryFieldLength</td><td>Define the subfile character field where the program will provide the labels for each category.</td></tr>
<tr><td>Series</td><td>A collection of series definitions. Each series has its own properties as follows:
<table>
<tr><th>Series Property</th><th>Notes</th></tr>
<tr><td>DataField</td><td>The name of the subfile field where the RPG program will provide data for this series. In a sense, this is the series name.</td></tr>
<tr><td>Length & Decimals</td><td>The subfile field will be of decimal type packed; these properties specify the length and number of decimals for the packed number.</td></tr>
<tr><td>Color or</td><td>A constant color name, alternatively the ColorField property can be set.</td></tr>
<tr><td>ColorField</td><td>The name of a character field to be defined in the subfile control record format where the RPG program can provide at runtime the name of the color to be used for the series.</td></tr>
<tr><td>Legend</td><td>A constant label to be used as the legend for this series. The user sees this as the series name. Alternatively the LegendField property can be set.</td></tr>
<tr><td>LegendField & LegendFieldLength</td><td>The name of a character field to be defined in the subfile control record format where the RPG program can provide the series' legend at runtime.</td></tr>
</table></td></tr>
</table>

There are also properties to set, via a constant or defined character fields, the following titles: Category Title, Value Title, and Chart Title. If fields are provided for these labels, Mobile RPG defines them as character fields in the subfile control record format.

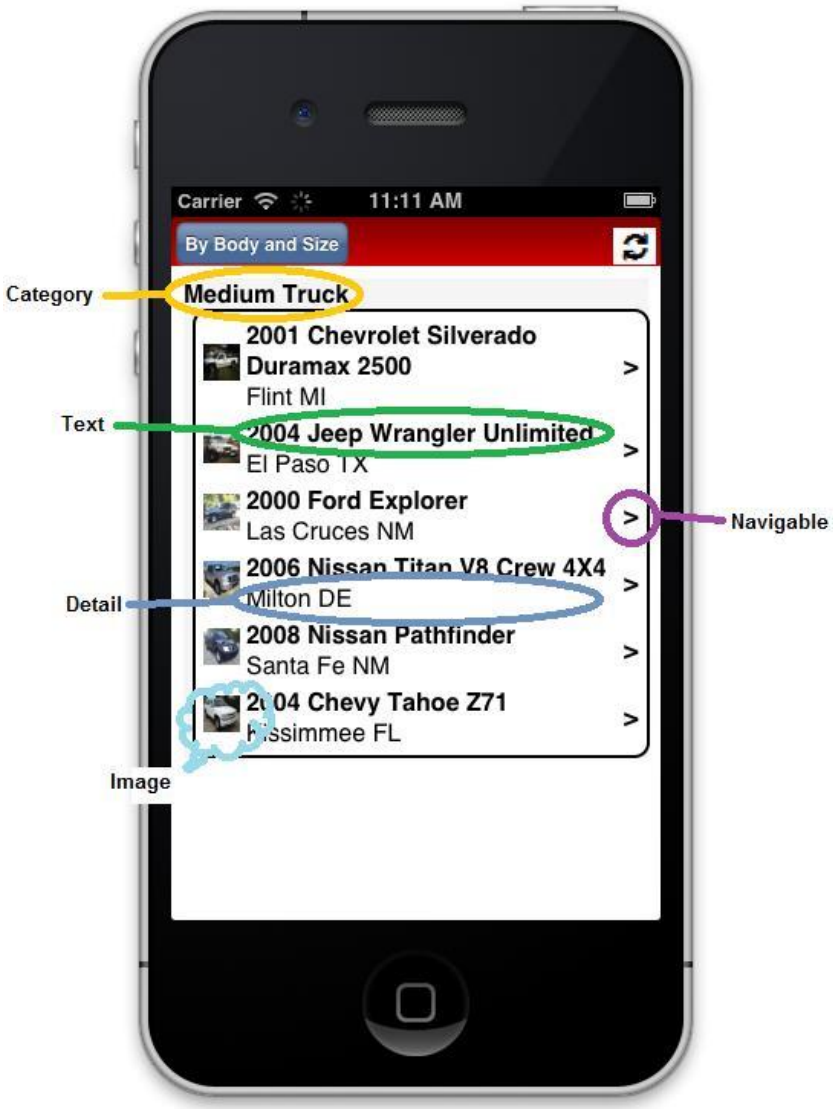
The developer can also set up additional properties to react if the user touches various parts of the chart. One property can define a field in the subfile control record format where the DdsChart can report to the program that the user has touched the chart's title or a series' legend. Another property can define a field in the subfile record format where the DdsChart can report back to the program that the user has touched an individual series' data point or the category title. These properties enable the program to react and possibly drill down when the user selects any one of these chart areas.

List

Often, Mobile applications express their data and options in the form of lists with checkboxes, thumbnail images, radio buttons, subtext, and various other embellishments that wouldn't be possible in a Traditional Subfile. To create these enhanced lists in Mobile RPG, one can use the DdsList Control.

The overall style of the list set at design time via the **ListType** property, which presents the options of a checkboxes, a dropdown, a listbox, a list with radio buttons, and a classic list.

Additionally, the control lets designers define the Category (for the list overall), as well as a Subfile and Subfile controller to associate with the list. Further, up to four visible features can be added to each list entry: main **Text**, **Detail** text, an **Image**, and an interactive right-pointing chevron that we call a **Navigable**. The default locations of these parts are detailed on the image of a classic list below:



The content and appearance of each of the highlighted parts is determined by the following properties:

Property	Notes
CategoryField & CategoryFieldLength	These properties define a character field whose runtime value is used to get the text content of the Category.
TextField & TextFieldLength	These properties define a character field whose runtime value is used to get the text content of the Text area.

DetailField & DetailFieldLength	These properties define a character field whose runtime value is used to get the text content of the Detail area.
DetailAlignmnet	Sets the Alignment of the Detail within the list.
ImageDirectoryField & ImageDirectoryFieldLength	These properties define a character field whose runtime value is used as the path to the directory holding the image file.
ImageNameField & ImageNameFieldLength	These properties define a character field whose runtime value is used to get the name of the image file.
ImageLocation	Sets the location of the image within the list.
NavigableField & ImageNameFieldLength	These properties define a character field whose runtime value is used as the address to navigate to if the chevron or list item are touched.

Other interesting properties of the DdsList Include:

Property	Notes
ListType	This property sets the type of the list from the following options: Checkbox, Dropdown, RadioButton, and Classic.
ValueField	This property can be used to create a "hidden field" with data accessible to the program but not the user.
Categorized	This property sets the list as either one with multiple categories, or a single uncategorized list.
SortCategory	This property determines how the categories of the list will be sorted: Ascending, Descending, or None
SortItem	This property determines how the items of the list will be sorted: Ascending, Descending, or None

Geolocation

Many mobile Apps take advantage of the built-in ability of mobile devices to pinpoint themselves on the Global Positioning System. MR Apps can do the same through the DdsGeolocation Property. It allows the program to access the devices' built-in GPS API, and therefore determine the location of the end-user.

This particular property uses a number of "packed" fields, each of which three properties. These properties follow the schema: *Name*Field, *Name*FieldLength, *Name*FieldDecimals. The runtime values of these three properties define IBM i fields that the program can use to pinpoint the end user's geolocation.

Property for Field *Name*	Notes
Latitude	Expressed in Decimal Degrees. Bounded by +/- 90°. Positive latitudes are north of the equator. Negative latitudes are south of the equator.
Longitude	Expressed in Decimal Degrees. Bounded by +/- 90°. Positive longitudes are east of the Prime Meridian (Greenwich, England). Negative longitudes are west of the Prime Meridian.
Accuracy	This property determines the accuracy to which the Latitude and Longitude are reported. Expressed in Meters.
Altitude	Expressed in Meters. This reports the altitude of the device relative to the WGS84 ellipsoid (roughly sea level),
AltitudeAccuracy	Expressed in Meters.

This Control is potentially very useful if you wish to use navigation software other than the GoogleMaps API.