DataMax Software Group Inc.
1101 Investment Blvd. , Suite 250
El Dorado Hills, CA 95762
USA

RFgen Users Guide

All Editions
RFgen 6.0.1

Table of Contents

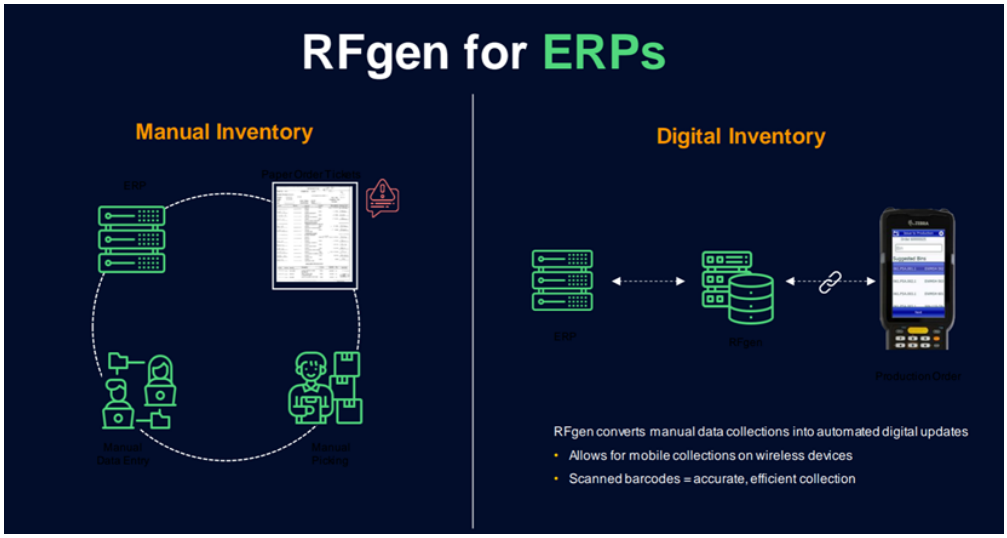
Table of Contents

Welcome to RFgen

RFgen helps collect barcoded data from mobile devices, convey the digitized data to your ERP system and print barcode labels to label printers.

The basic workflow consists of mobile devices running applications used to scan barcodes and conduct warehouse tasks. When a user completes a task, the data in the transaction is submitted by the user to the RFgen server which in turns conveys the data in the transaction to the Enterprise Resource Planning (ERP) system. If desired, the customer can also use the new RFgen Secure Label feature to design labels, merge data from a specific source of data (database) and mass print barcode label to label printers in the customer's environment. Secure Label requires additional licensing from RFgen.

The major components in the RFgen solution consists of the **RFgen server**, **RFgen client software**, the application database (collection of warehouse apps that are installed to the user's devices), and a data-source (i.e. inventory data used in a Cycle Count app, shipping data for a picking app), and connections to the ERP system.



The **Mobile Unity Platform Console** (RFgen Server) features: Tools for configuring and administrating RFgen services.

The ***RFgen Mobile Development Studio*** features the tools to design, test, and create profiles (the apps and dependent resources) that deploy to RFgen clients in addition to configuring the RFgen server.

The **RFgen Client software** enables communicate with the RFgen server. Once the connection is made between the client and server, profiles are downloaded to the client. A **Thin Client profile** enables users to work in near realtime when the client is connected to the RFgen server. A licensed Offline client (also called Batch,Mobile or Fat client) profile enable users to work offline and convey updates to the ERP when the user reconnects with the RFgen server.

The **User Management Console** is used for assigning menus to groups of users or individuals and linking applications to specific menu items.

The **Mobile Enterprise Dashboard** provides supervisors/admins the ability monitor connected client sessions and if desired, if needed log into a session to provide help.

The **Transaction Management Dashboard** lets supervisors/admins view queued, successful and failed submissions to the ERP. If needed it also provides tools to edit and repost failed transactions for reprocessing or view historical submissions.

Additional help resources

- RFgen Install and Upgrade Guide
- RFgen Mobile Client Install Guide for Android
- RFgen Mobile Client Install Guide for iOS
- RFgen Mobile Client Install Guide for Windows CE/Mobile
- RFgen Mobile Client Install Guide for Windows Desktop Client

Basic Administration Steps

To get started, your RFgen server must first be authorized to run. The authorization requires a certification which is installed via your services console.

The three ways to authorize your server are:

- Email RFgen Support
- Phone Support
- RFgen Web

Next, you'll need to configure the data source that will be used in your mobile transactions. Depending on how your warehouse environment is seutp, this can require a connection to an ERP and/or a separate database or, no connection if the transactions are through an internet connection.

Before you configure your **Mobile Development Studio** or the **Mobile Unity Platform Services Console**, it is necessary to understand the basic concepts of your Data Base Management System (DBMS).

- Knowledge of data structures is of particular importance for database applications since it is necessary to understand the basic concepts of 'tables', 'fields/columns', and 'data types' prior to creating applications.
- Understanding Structured Query Language (SQL) 'syntax' is also helpful with database applications.
- Experience with the Microsoft Visual Basic/VBA programming language is helpful in the development of advanced data collection applications, for use with both SQL databases and legacy host-based applications.
- Once you have installed the Mobile Unity Platform Services (RFgen Server) you will need to configuration your server services and other environment features as well as set up data source connections if they have not be setup already.

  For information on configuring the RFgen server, see TextHint.

After you have configured application data source(s), and added new Enterprise connections, the following steps are generally implemented for ODBC or SQL-compliant databases:

1. Transaction tables specific to your application are designed via your main database system (i.e., data to be captured is defined in your database). Typically, an individual 'transaction table' with appropriate field definitions is established for each data entry process.

2. The Database transaction table field definitions are downloaded to RFgen. (See Enterprise Connections > Download Enterprise Objects.

   These definitions contain all the necessary information concerning the transaction data to be written to your database. (No data is downloaded from the database.) Table definitions may be partially edited (inside Mobile Development Studio), if required. Similarly, ERP business rules and functions may be accessed and downloaded, if using the RFgen ERP integration suites.

3. Data entry Applications are made for each transaction table. Pre-programmed data properties such as 'defaults', 'validations/edits', 'table validations', etc. are added, as required. This is located under Configuration > System Properties > User Defined System Properties.

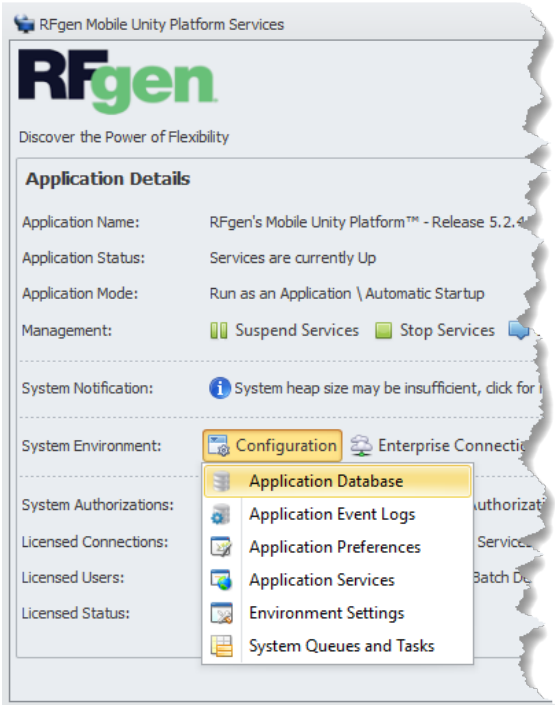
4. (Optional) Once your configuration and resources are setup, you can, if desired, use the Mobile Development Studio to design, modify, script, test and deploy mobile applications.
5. There are a variety of methods to control which apps are made available to individual or groups of users. If desired, user access to specific applications can be managed by adding users to the RFgen server, creating application menus, and assigning who can to the menu(s) in the **Mobile Development Studio > Solution Explorer >** Users and in the Menus and Roles, or by managing users and their menus in the User Management Console.
6. The applications, menus, device configurations and other resources needed to run on the client are packaged by the RFgen developer in a Profile in Dev Studio. The profile installs to the client once the RFgen client and server establish a communication session.

Client Connection

7. In order for your mobile devices/clients to communicate with the RFgen server the RFgen Mobile Client software installed. Upon startup, the user on the client side will need to enter the RFgen server information. Once the communication session is established, the client profile (from step 6) is deployed to the client. Depending on the information in the profile, the client then functions as a Thin and or Mobile client.

Upon completion, the user on the client should see the RFgen login and/or a menu of applications.

Configuration Overview



The Configuration menu is used to setup the RFgen server services, associate it with the specific application database to be used for your environment (i.e. Development, Test, Production environment), and set your preferences and defaults in the Mobile Dev Studio and/or Mobile Unity Platform Services Console (RFgen Services console).

If your Dev Studio and RFgen Services Console are on the same system, whether you configure from Dev Studio or Services Console, the results will be the same.

RFgen Server Configuration (accessible from the Mobile Unity Platform Console or Mobile Dev Studio):

- The Application Database configures the database storing all the solution objects.
- The Application Preferences is for Service Console/Dev Studio user interface and language settings, and Mobile Application Designer/Testing and scripting settings.
- The Application Services is primarily for service communication and security. It includes the server name, port configurations, setup for servers used in load balancing/custers, service run mode, NAT Firewall setup, and Administrative and Encryption configurations.
- The Environment Settings contains the system options, timeout values, Scanner Options, Google Maps Integration (License Key), device menu strip SideBar Options and Menu Options, and a way to assign actions to device function keys.

Configurations Available Only in Dev Studio

- The Scripting Environment settings to allow direct access to Active Directory Objects (ADO) and XML language extension parameters and have them globally loaded into BAS files.
- The Source Control Options allow developers to use a third-party source control product if its plug-in is supported.
- The System Properties for adding user-defined system properties.

- The User Access Control console allows you to authenticate connections between the RFgen Server and the *Mobile Platform Unity Management Console*, and *User Management Console*.

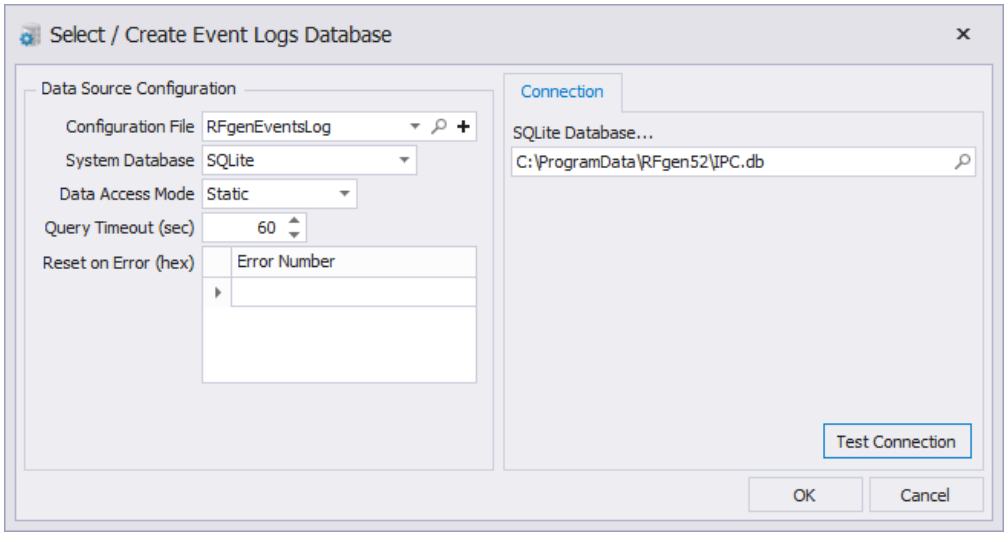
Connections Available Only in Dev Studio

- The Download Enterprise Objects option enables you to download all or specific types of objects from a connected database.
- The View Enterprise Objects option enables you to select and display previously downloaded objects.

To Configure Event Logging or Create an Event Database

By default RFgen has event logging turned on, and the output captured in Dev Studio or the Mobile Unity Platform Console, click on **Reports > System Event Logs / Application Logs**. The factory provided events and performance logging is in the IPC.db file which is located in the **ProgramData\RFgen 52** directory.

Use the Create Event Logs Database screen to create your own database and connect to it from RFgen.

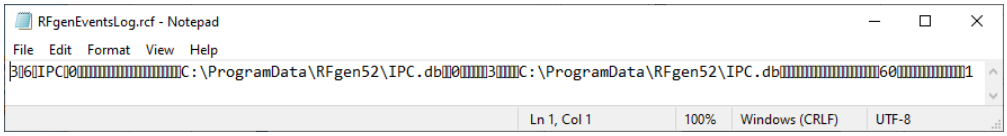


You can create your own database by copying the IPC.db (SQLite), or create your database from Access or Oracle, or connect to an existing database hosted by another server (i.e. SQL Server).

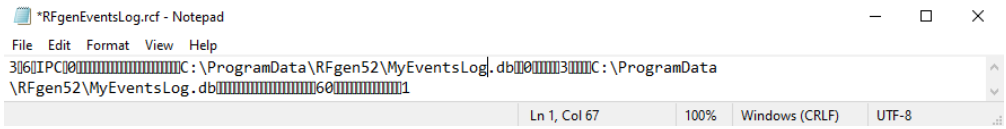
Create database in SQLite

1. From your Windows system, copy the IPC.db file and rename it. This file is typically located in your **ProgramData\RFgen 52** directory. For example "MyEventsLog.db" and save it in the same location as the IPC file.
2. From your Windows system, create another .rcf file, rename it.
3. Open the file in Notepad or a similar editor, and change its path to the location of the database created in step 1.
   For example change the "IPC.db" to "MyEventsLog.db".

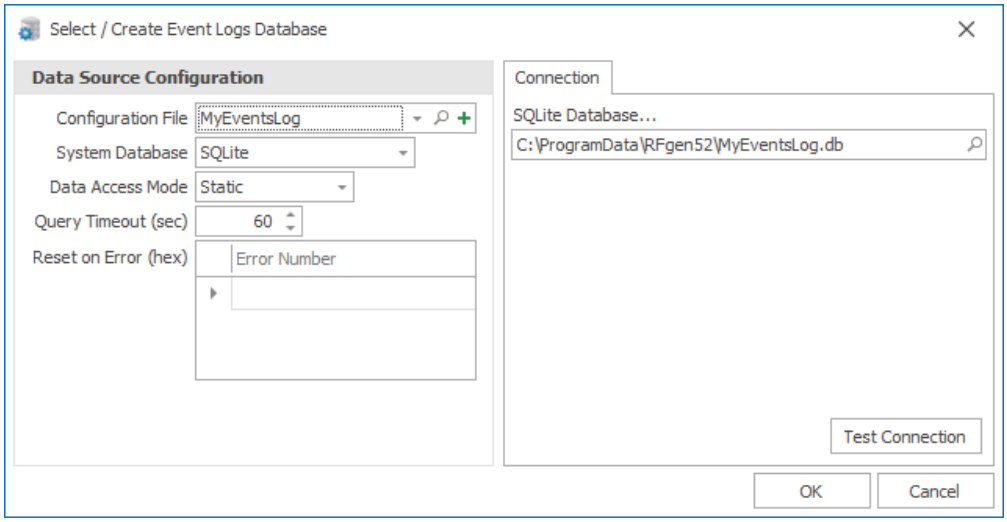
   Before:



After:



4. Save the rcf file and make sure the extension is .rcf (not notepad.)
5. Click on the **Mobile Development Studio > Connections > Application Event Log** screen, or the **Mobile Unity Platform Console > Configuration > Application Event Logs** screen.
6. In the **Configuration File** field, click the search icon, and locate the new rcf file. In this example, select "MyEventsLog".
7. The Connection tab should populate with the database created from step 1.



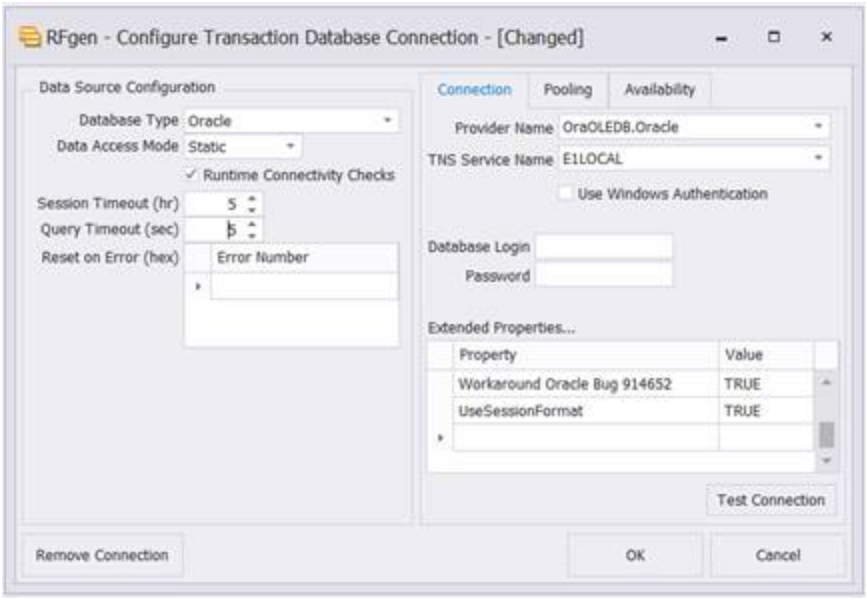
8. Click on **Test Connection**. A Connection Successful message should display.
9. Click **OK**.
10. View the output from **Reports > System Event Logs**.

Related Topics

For more details viewing and filtering log files, see the System Event Log topic.

Configuring Transaction Management DB Connection

To configure transaction connections, select **Enterprise Connections > Transaction Management Database**.



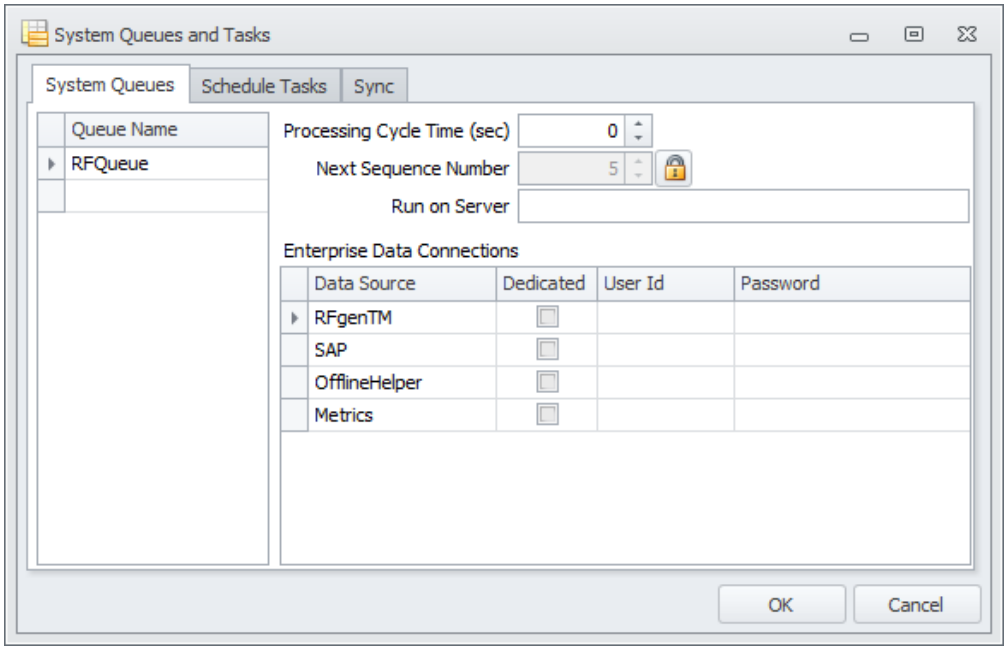
Configuring a Transaction Management database connection is the same as any other database connection. All the same fields apply. This connection is simply dedicated for the queuing process. See the section "Configuring Database Connections" for details.

There is one property that is unique: **Reset on Error (hex)** when turned on will reset the data connection under the following conditions. This should not be used unless deemed necessary.

If you submit a SQL statement to the DB, and an error code is returned, RFgen will look for the code in the *Reset on Error (hex) table*.

- If the error code is found, RFgen will try to reset the data connector. (RFgen won't resend the submitted query that produced the error code.)
- If RFgen cannot find the error code in the *Reset on Error (hex) table*, RFgen won't reset the data connector.

Configuring System Queues and Tasks



System Queues Tab

This table of queues allows for several queue processes to take place at the same time. The name RFQueue is the default name for the first thread that will process transactions.

The **Processing Cycle Time** number is the number of seconds that will pass before the system checks this queue for transactions and if one or more are found the entire queue is processed.

The **Next Sequence Number** option will allow the user to change the sequence number used when queuing or making entries in the logs. To change the number, click the lock icon then change the value. A sequence number helps identify the specific transaction for reference purposes.

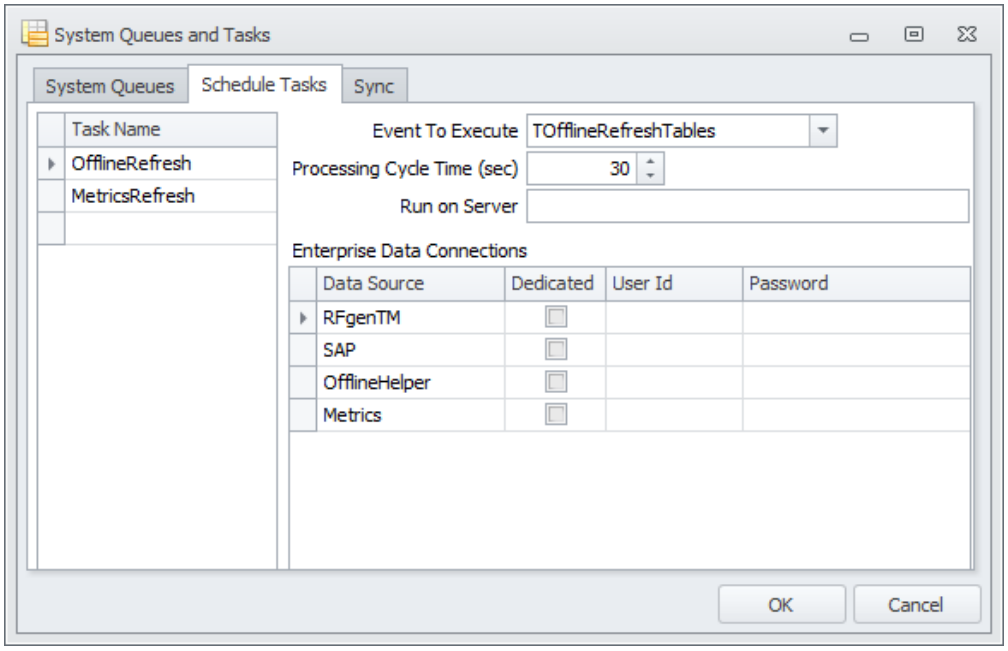
The **Run on Server** is used to specify a server to prevent your queued transactions and events from running on all servers that are actively used for load balancing. If this field is left blank, RFgen will not check for any server changes and run all processes on all server(s) connected to the RFgen database. If the server IP address or "Local Host" is present, then RFgen checks which server to use and runs processes against the assigned server.

Enterprise Data Connectors

Each data source can be configured separately for each queue meaning that each queue can have either a dedicated connection to a specific data connector or it can share a limited pool of handles to that data connection. This is the **Dedicated** check box option. In either case a user and password could be specified for the queue to use when it communicates with that connection.

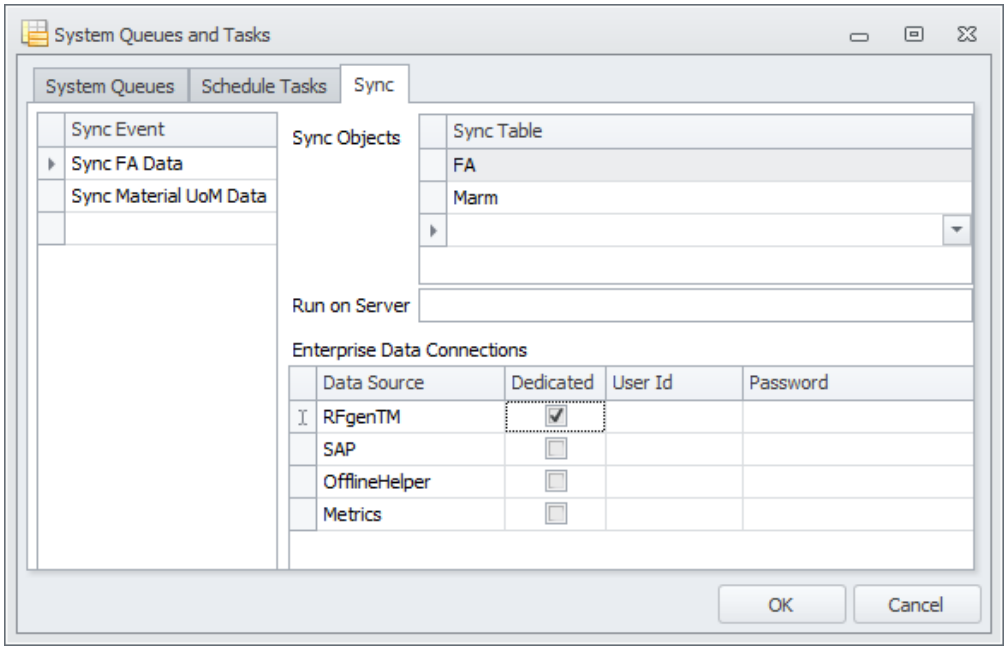
Note: Each queue process uses one client license because it is, in essence, an automated user performing tasks against the server and the backend systems. For example, a 10-client licensed system with three separate queue processes will only allow up to seven concurrent devices.

Schedule Tasks



The **Schedule Tasks** tab allows the user to specify a task name and then assign it to a Timed Event macro chosen from the **Event to Execute** drop down. The **Processing Cycle Time** is in seconds and determines how often the server will run the Timed Event macro. Just as with the queues each data source can be configured separately as well as taking a client license. See the above section for more details.

Sync tab



The purpose of the Sync (synchronization) tab is to list the tables that must exist in order for synchronizing data to occur between the data source and target. This tab is used to ensure the tables exist.

While the synchronization is designed to run in the background once the sync commands have be scripted in your solution, you use the **Sync Event**, Sync Table and selected Enterprise Data Connections to "precheck" these structures .

Note

Before a client can process data offline, it must first have a database of the tables with the data to be updated. These kinds of tables that are provisioned through a client profile are called "Sync Tables." The format of the table, its schema, and datatypes etc need to be identical in form, fit and function on the client side as it was from the source (the master database which is considered the source / source of truth.

Add New Enterprise Connections

To add a new database connection to a data source such as SQLite, DB2, ODBC, OleDb, Oracle, SQL Server, or MS Access, see To Add an New Data Source Connection.

To add a new Enterprise Resource Platform (ERP) such as Oracle EBS, see To Add/Connect to JD Edwards, or for SAP see Configuration for SAP.

To add a new Screen Mapping connection from the RFGen server to host systems (for legacy systems), see Connect to a New Host.

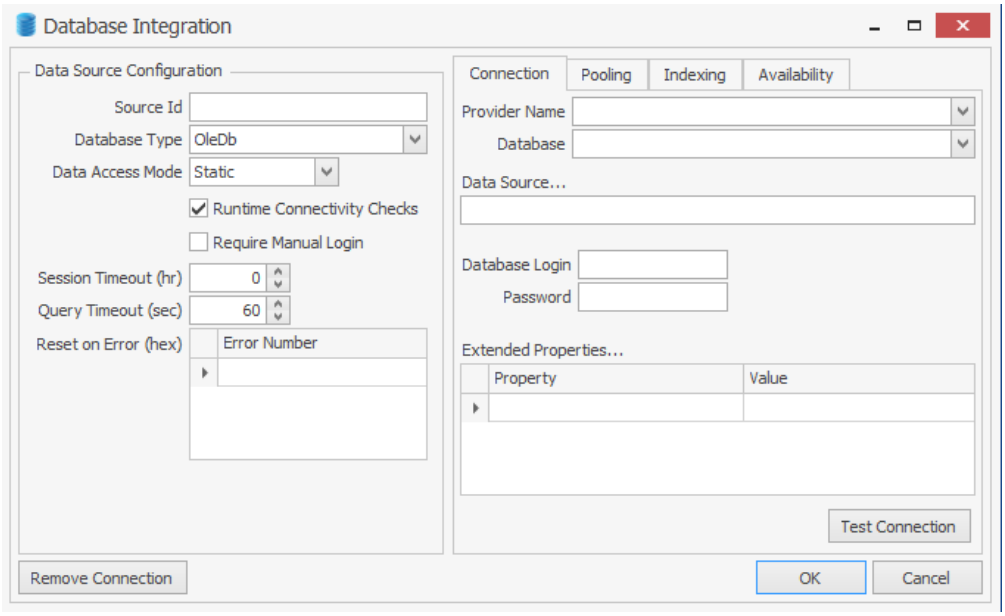
Note: The Add a New Web Services Connection screen is in the process of deprecation.

Adding a New DataSource Connection

To add a connection to a new data source, from the **Mobile Development Studio** or **Mobile Unity Platform Services Console**, click on **Enterprise Connections > Add New Database Connection.**

RFgen supports connection to home-grown or commercial databases that are SQL-compliant.

Use the **Data Integration > Data Source Configuration screen** to configure how RFgen will access the data hosted by a commercial or home-grown database and set the timeouts should a non-optimal event occur.



The **Source Id** is the name of data connection. Spaces and extended characters are not recommended for this field.

The **Database Type** drop down field selects which type of database is to be used to host the solution objects. Changing this value changes the content of the Connection screen, to show database specific configuration fields.

Data Access Mode sets the cursor to either Static or Dynamic when retrieving data from the database. Usually, Static is best because it is fast and safe.

However, if you have a database that will make a copy of the data from the database system to the RFgen system you can change this option from Static to Dynamic, so performance will not suffer.

Internally, this sets the cursor option to either adoOpenStatic or adoOpenDynamic.

The **Runtime Connectivity Checks** will turn on verification if the connection between RFgen and source is still there.

The **Runtime Connectivity Check** is a process that will verify whether the datasource is still connected. the user doesn't have the permission to access a table, the connection will reset (via a select count process) and go into a loop.

The **Require Manual Login** forces a user to login before that user can connect to RFgen which in turn, manages connection to a datasource on the behalf of the user. Whether you should check this box depends on the type of users in your environment and how you want to manage their connectivity to the RFgen which in turn requests login to the ERP. If you check this box, you will also need to enter the **Database Login** and **Password** information so the user can manually login and use this connection through RFgen.

For example, with JD Edwards, if **Require Manual Login** is checked, when the user logs in to process a transaction, his/her login user ID is checked against the Named User list.

If the Named User list is not setup correctly in RFgen (or JD Edwards), or, if the user isn't on the Name User list, a database error may result.

If **Require Manual Login** is unchecked, the client will be allowed to connect with the database automatically.

The **Session Timeout** value will disconnect and reconnect to the database at the specified interval. This may be required if the database is configured to not allow a connection that never times out. This value is in hours.

The **Query Timeout** specifies how long the server should wait before giving up on the database driver to come back with a response.

Reset on Error (hex) is a list of hexidecimal values that if returned by the ODBC driver will cause a reset of the connection. The process for adding a value is to first get the error number from the error log.

For example, the error log shows -21456327. Use the Windows calculator in Programmer mode, select Dec and Dword options, enter the number and if you need the negative sign use the ± button to change its sign. Then click the Hex option.

You should get: FEB89A39. Enter this value into the box with a "0x" prefix like: 0xFEB89A39

Connection, Pooling, Indexing, and Availability Tabs

See the Connections Tab topic for details on creating a connection to the source provider.

See the Pooling Tab topic for information on setting up users in a pool.

See the Indexing Tab topic for information on configuring Discovery and Indexing schemes to optimize performance when extracting data from the source.

See the Availability Tab for information on scheduling downtime so RFgen doesn't attempt to connect if the source is not available.

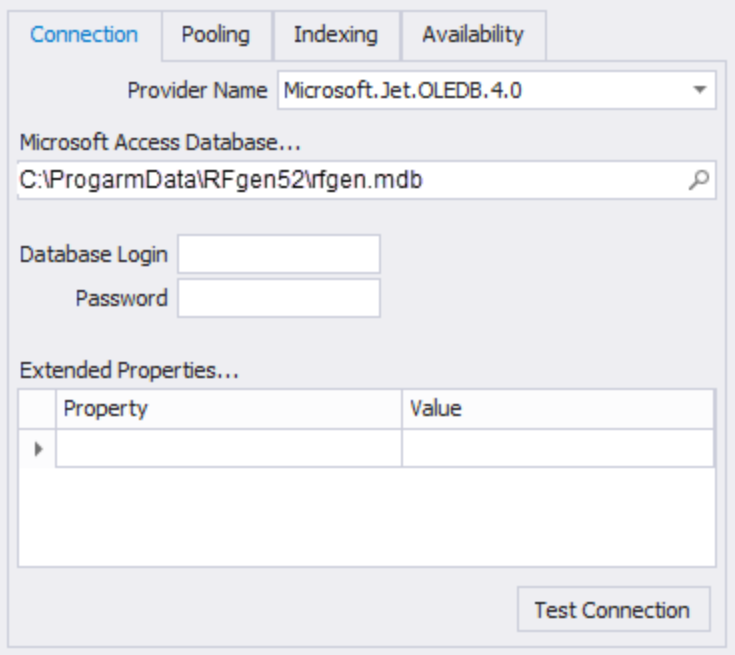
Connection Status Indicator

When the server connects to a database, it will display a connection indicator at the bottom of the Mobile Development Studio window.

If a red circle appears in the indicator, a valid icon has not been made.

To troubleshoot an invalid database connection, click on the Mobile Development Studio Reports menu bar selection, then check your Event Logs to see if a message has been generated. Most likely, a problem was encountered with your Data Source entry.

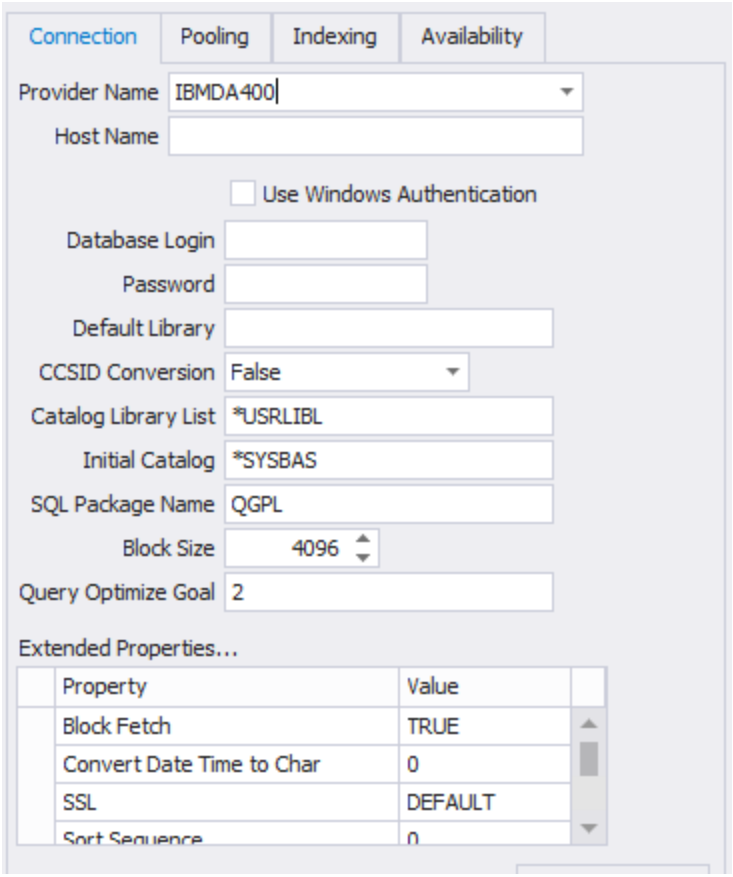
Connection Tab



The **Provider Name** selection will depend on the type of database you want to use. Note that these providers are not necessarily installed. All provider options must already exist on the server to be used.

For an Access database select the appropriate Provider Name for the type of system (32 bit or 64 bit).

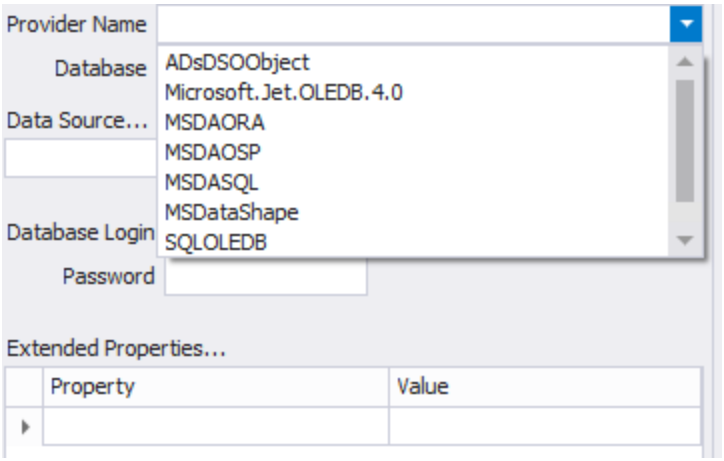
The path, login, password and extended properties are then used to make the connection. In the case of Access most of these fields are not necessary.



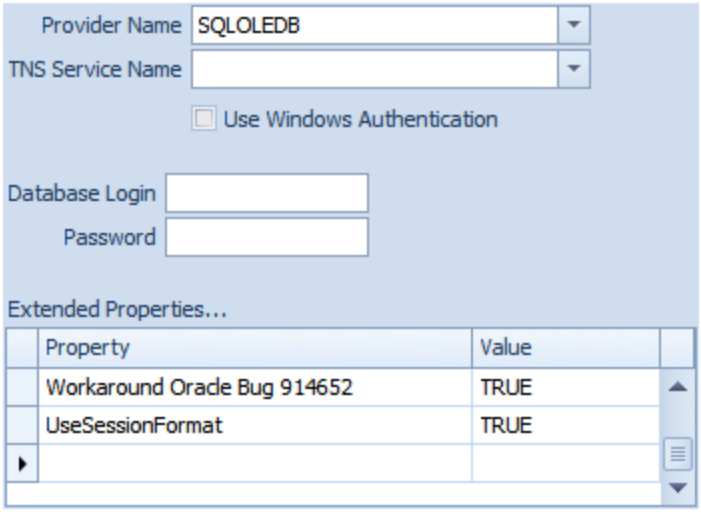
In the case of DB2, you can specify all the same settings that would normally go in the ODBC DSN entry for the iSeries Access driver.

For ODBC DSN entries that come from Control Panel / Administrative Tools, this option assumes that the connection has already been established the server will just reference what is setup in Control Panel. This option should be used if other programs also rely on the same database connection and it is easier to maintain the settings in one place rather than many.

The OleDb option is the most generic method of connecting to a database. The Provider Name shows many options, most of which need to be manually installed (acquired by the manufacturer) before the server can take advantage of them.

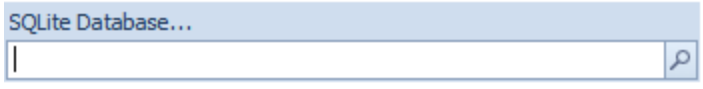


In the case of Oracle, ODBC is not used but the TNS Server Name points to the Oracle server. Also specify the Provider Name and review the **Extended Properties** for accuracy. The Use Windows Authentication option will take advantage of the Active Directory when connecting to the database.



For SQL Server specify the Provider Name, Server Name and Database Name. The Use Windows Authentication option will take advantage of the Active Directory when connecting to the database. If you want to connect directly to the MDF file itself, specify the Attach Database File option and locate the database file directly. The Logical Name is typically the filename without a file extension and should not be necessary.

The **Extended Properties** are usually not required.

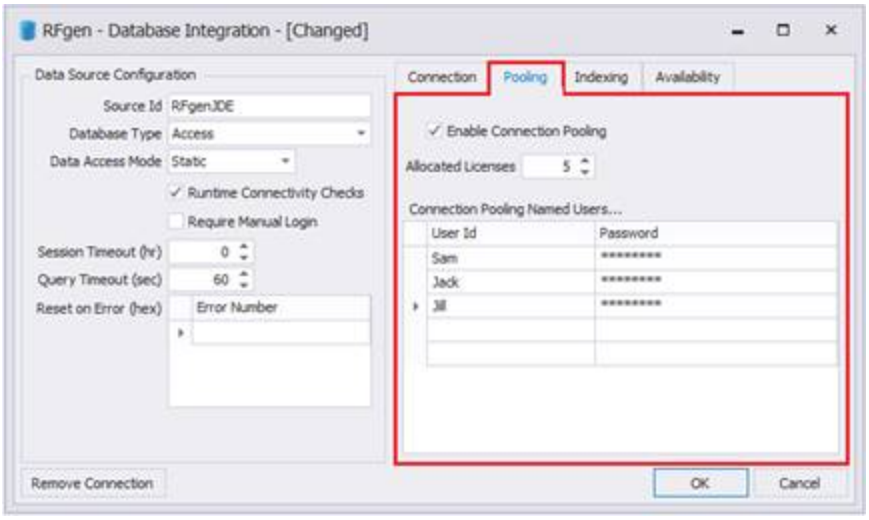


For SQLite database connections just specify the DB file itself. There are no other settings. You can specify a location and name that does not exist and clicking the Test Connection button will create the database for you.

Finally click on the **Test Connection** button to verify connectivity. If the database has not already been setup to support the solution tables they will be created at this time. Clicking the Save Changes button will also create what is necessary but won't test the connection. Either button will also notice if the database came from an older release and ask if you want it upgraded.

Pooling

Pooling refers to configuring multiple database connectors into one licensed connection for license conservation purposes. Pooled sessions can setup for Named Users if the pooled connection is for Oracle JD Edwards.

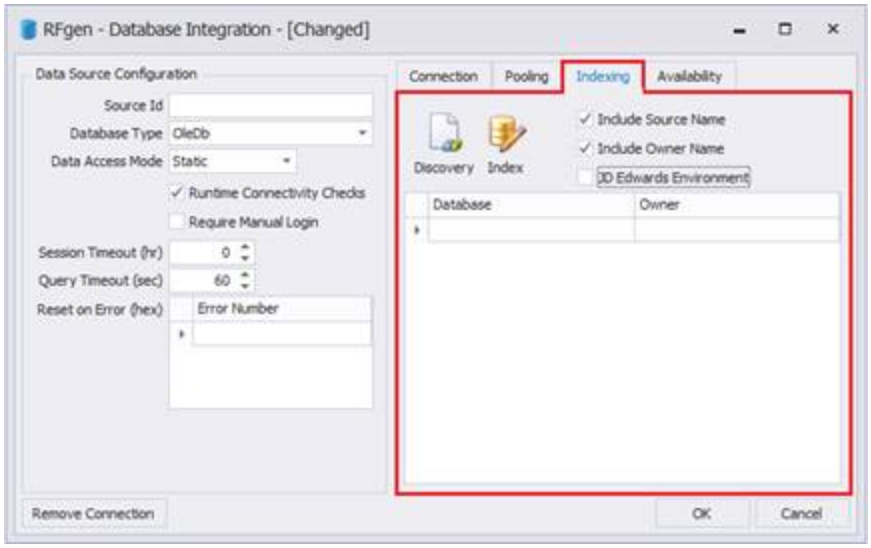


The **Enable Connection Pooling** checkbox turns on Pooling.

As **Allocated Licenses** are incremented, a Pool(n) session appears for each. Enter a User ID and Password for each in the corresponding boxes (if different from the defaults).

The connection pooling User ID and Password fields contained in this window are for allowing users to log in under non-default settings. As each session is taken from the pool (when simultaneous access is required) the next pool's settings will be used. For example, if your system only allowed two connections with a particular User ID, Pool (3)'s User ID and Password could be specified and the first two will be taken from the default information.

Indexing tab



This has advanced features for connecting to the database.

The **Discovery** option will attempt to fill in the grid automatically with all the databases and owners ultimately indexing everything in the database.

The **Index** icon means the server performs the indexing when the user saves the connection. The server will index tables within those additional databases so they can be referenced by name only. For example, F0005 is a control table in the database. Using this indexing it may be accessed simply as F0005 and the server will qualify it with database.owner.F0005 internally before SQL execution.

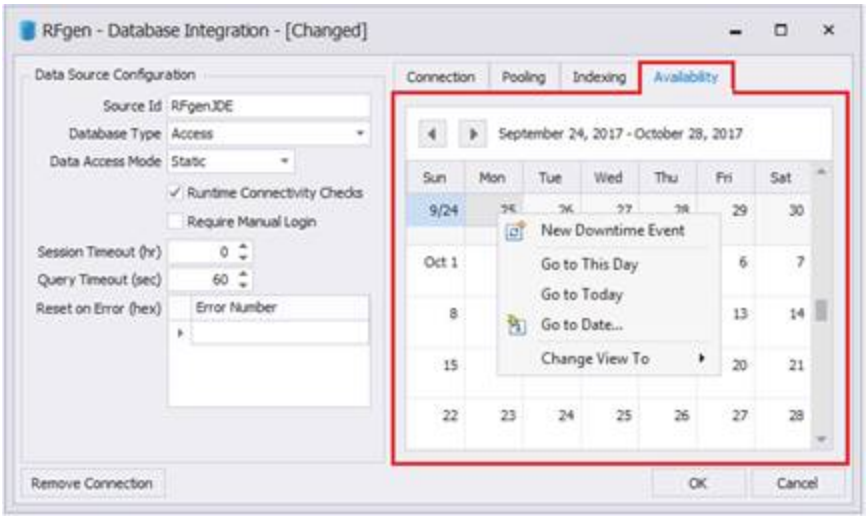
The **Source or Owner Name** information are options to more uniquely qualify the tables or database structure in case other connected databases have tables that are named the same when downloading the tables. For example, your main database may be SQL Server or Oracle, but you have a need to connect to tables contained in other databases or entirely different ERP or legacy systems. Databases may be different in type, as long as they are SQL compliant.

The **Database** and **Owner** grid allows the user to restrict which tables are indexed for a specific data connection. In this grid, specify the list databases and / or a list of owners of the tables that are necessary for the data collection application.

The **JD Edwards Database Indexing** option is a JDE specific option that will query certain JDE tables to determine how the system is actively configured for the selected environment and then index specific JDE tables.

Availability Tab

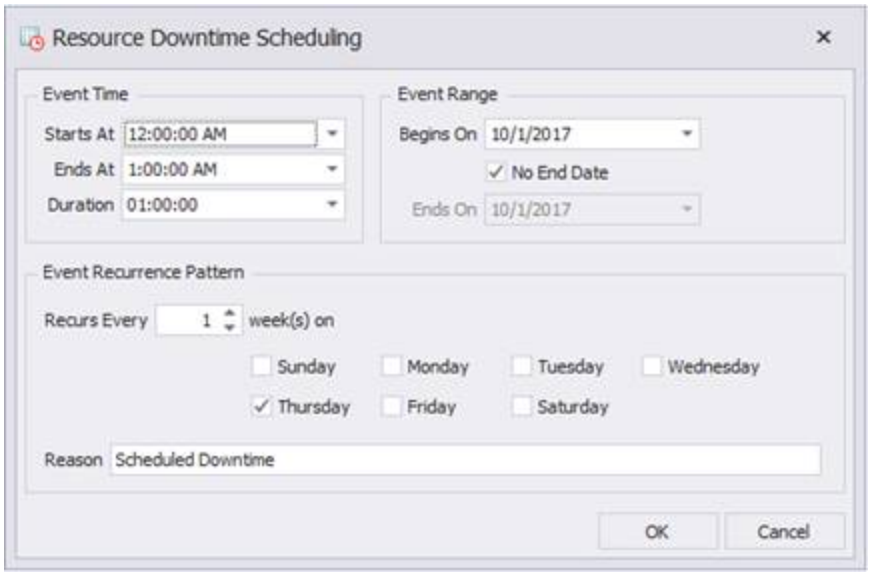
The **Availability** option is used to schedule down time for maintenance purposes.



To schedule downtime, right-click on the date or days in the calendar and select the appropriate item from the menu.

In this example, the **New Downtime Event** was selected.

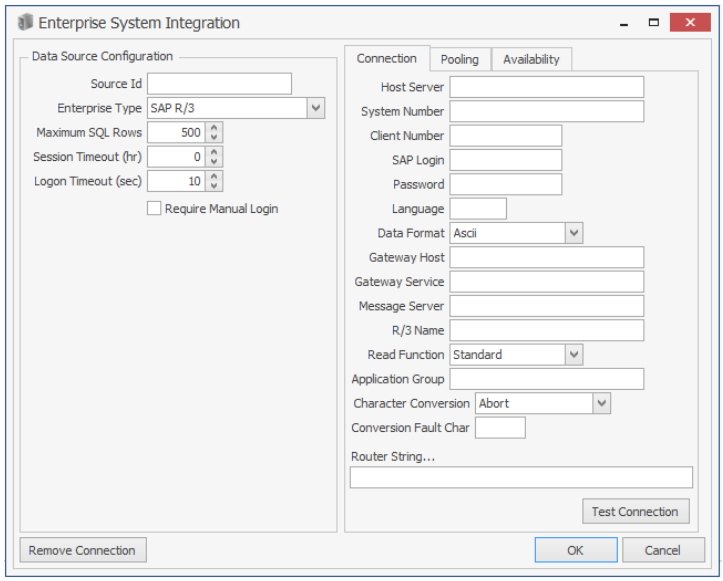
In the Event Time box, the connection will be unavailable for 30 minutes, every Thursday, between 12 AM and 1 AM beginning Oct 1, 2017 and will reoccur until an End Date is supplied.

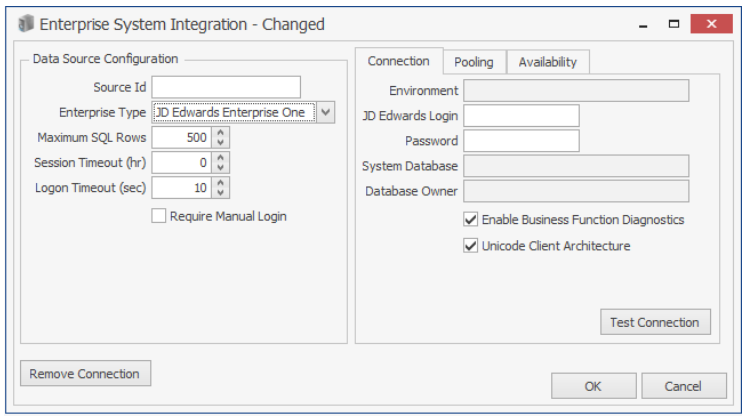


Add an New ERP System Connection

To add an **SAP** or **JDE** ERP connection from the Mobile Development Studio or from the RFgen Mobile Unity Platform Services Console select **Connection or Enterprise Connection > Add New Enterprise Connection > Add New ERP System Connection**. The Enterprise System Integration screen displays.

Depending on your Enterprise Type, your screen will look different than the examples shown below.





If you are connecting to Oracle JD Edwards, see Configuring for Oracle JD Edwards.

If you are connecting to SAP, see Configuring for SAP.

Related Topics

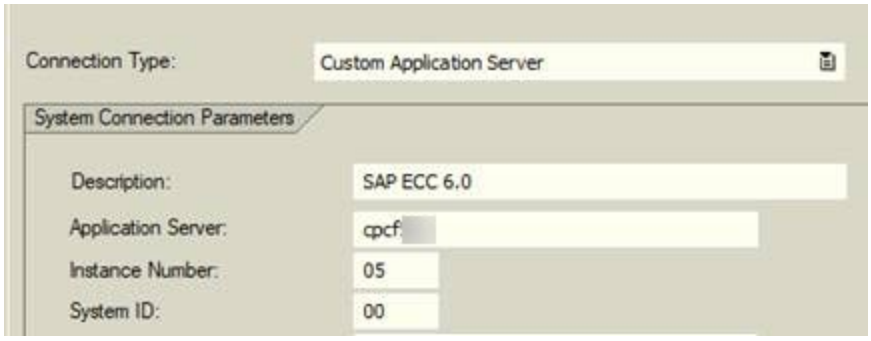
See Pooling Tab for information for setups that have pooled users as connected clients.

See Availability Tab for information on the scheduling downtime.

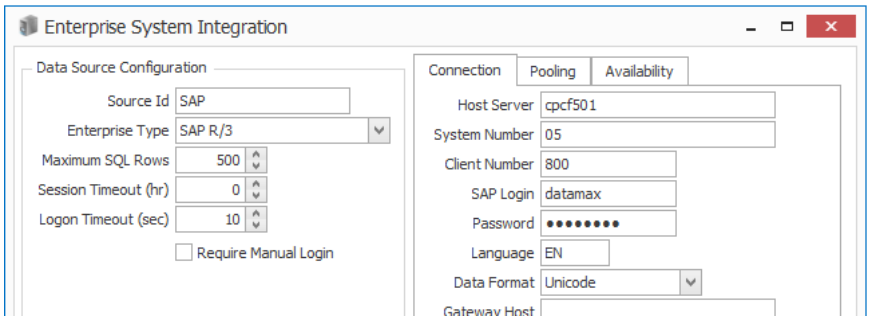
Configuring RFgen for SAP

RFgen is a zero-footprint integration to SAP, meaning there is no custom coding or additional software to install on your SAP server.

RFgen makes the connection to SAP using the SAP Connection Properties found in your SAP Logon screen.



Example of an SAP Connection Parameters Screen



Example of Corresponding RFgen Data Source Screen

From that connection, RFgen can download the schema for any remotely enabled function module/BAPI.

Setup Requirements

RFgen requires a 'User ID' to make the connection to the SAP server. A System ID user is recommended for this purpose. For example, you can enter "RFgenConnect" or even "RFgen".

Requires a non-expiring password

Must be assigned authorization objects S_RFC and S_TABU_DIS

To configure an SAP connection

Select **Enterprise Connections > Add New Enterprise Connection > Add New ERP System Connection**.

First enter a **Source ID** name and then change the **Enterprise Type** to *SAP*.

Next, enter a value for the **Max SQL Rows**. The **Max SQL Rows** prevents a 'lock-up' scenario in case a query was bringing back too many records by accident. ERP systems can typically have millions of records and this could prevent a frozen client. The command ERP.ReadData can perform SQL statements against the SAP connector.

The **Session Timeout** value (in hours) will disconnect and reconnect to the ERP at the specified interval. This may be required if the ERP is configured to not allow a connection that never times out.

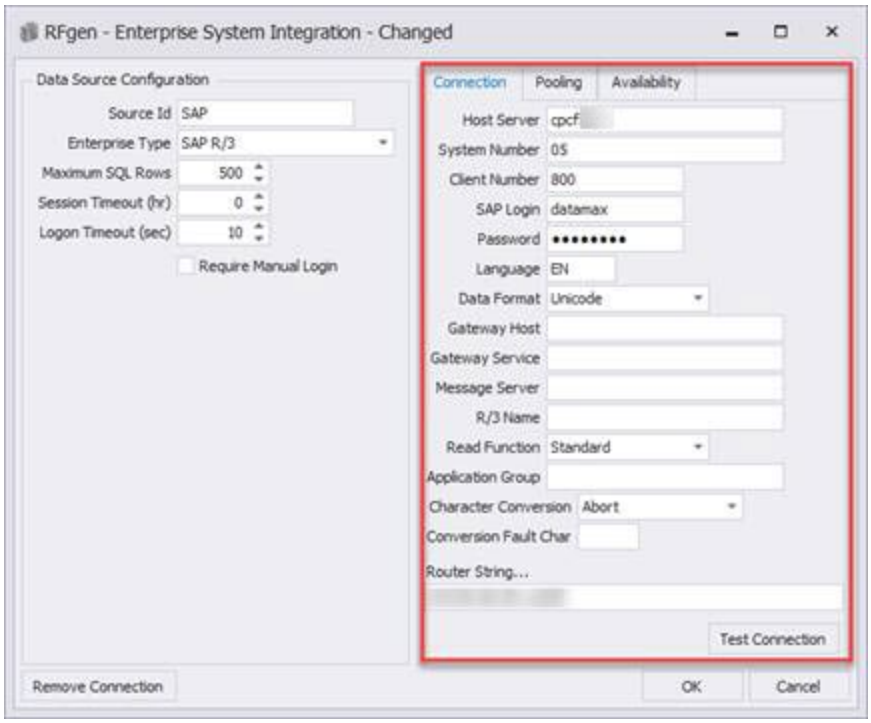
The **Logon Timeout** (seconds) will return with a failure in the error log if a request to log on to SAP never comes back.

You can select a **Manual Login** if desired.

Connection Tab

See Connection Tab for SAP information on completing the Connection tab fields.

Connection Tab - SAP



For SAP, the **Host Server** is the application server. Enter the **System Number**, **Client Number**, **SAP Login, Password,** and **Language** with the same data as stored in the SAP GUI Logon Pad application. The System Number also dictates the port number being used to communicate with SAP. For instance a 01 entry would possibly set the port to 3301 where a 05 entry would set the port to 3305.

The **Data Format** option tells RFgen how to interact with the system, either by using Unicode-formatted communication or non-Unicode (ASCII) formatted data.

The **Gateway Host, Gateway Service**, **Message Server**, **R/3 Name**, **Application Group** and **Router String** are optional parameters. If your Logon Pad requires these settings, then add them here for RFgen.

Read Functions has two options: *Standard* and *BBP*. The Standard option executes SQL queries using the RFC_READ_TABLE function module; the BBP option executes SQL queries using the BBP_RFC_READ_ TABLE function module. Only used with SAP HANA.

For SAP load balancing, configure the following fields: **System Number**, **Client Number**, **User ID**, **Password**, **Message Server**, **R/3 Name,** and **Application Group**. Leave the **Host Server** blank since the Application Group setting will distribute requests to multiple host servers. In some cases, you must leave the System Number blank as well.

The **Character Conversion** and **Conversion Fault Char** properties are designed to handle problems when SAP sends data in a different code page than what RFgen is configured to display. If the text does not have a translation, RFgen can be configured to abort the conversion, copy the bad character, or replace the character with the character entered in the Conversion Fault Char property.

The **Enable SNC** (secure network connection) checkbox is used for the encryption of SAP data. For more details, see SAP Data Encryption.

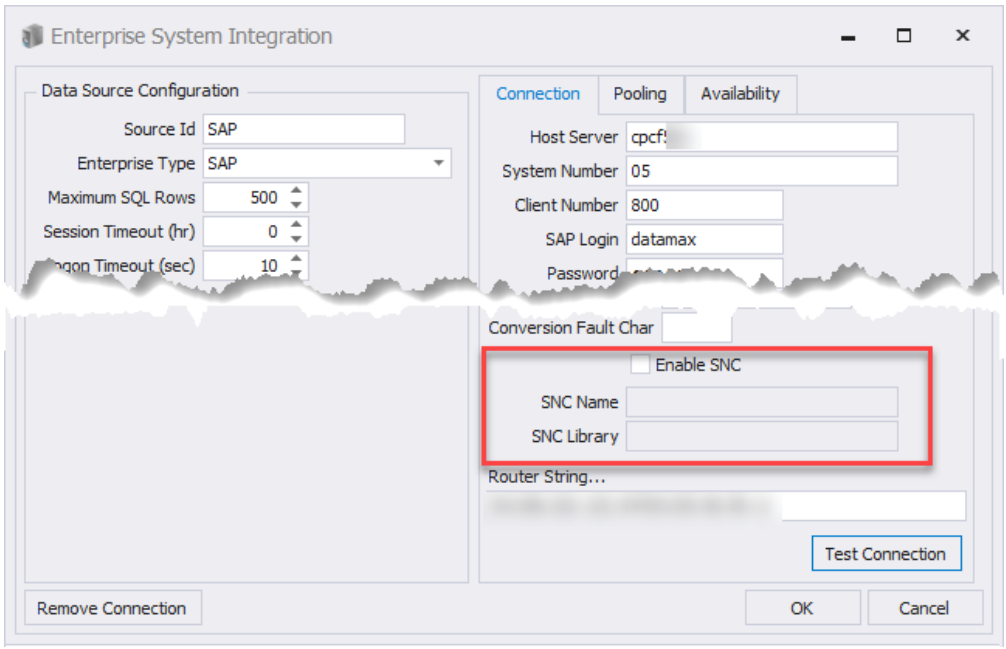
The **Router String** is an optional entry that can be used to setup connections between intermediate stations. For more details, refer to your SAP documentation on SAProuters.

Related Topics

See Pooling Tab for information for setups that have pooled users as connected clients.

See Availability Tab for information on the scheduling downtime.

SAP Data Encryption



If you have users (clients) who collect and process data that is processed at sites outside your firewalls, you may want your data encrypted for security purposes. For example, a salesperson who processes sales orders offsite; A warehouse work who inventories lumber at an outdoor storage facility or workers who are connected but through the cloud.

RFgen supports the SAP Secured Network Communications protocol which secures the communication paths between various SAP system client server components. With SAP SNC, you receive application-level, end-to-end protection for data communicated between two SNC-protected components.

The SNC function is provided through the SAP Cryptographic Library which delivers encryption functions in SAP systems. For more information on the use of or implementation of the SAP Cryptographic Library, see SAP documentation.

To enable a Secure Network Connection (SNC) for SAP

1. Click on **Connections** > [your SAP Enterprise Connection] from the Mobile Development Studio. Or click on **Enterprise Connections** > [your SAP Enterprise Connection] from the Mobile Unity Platform Console.

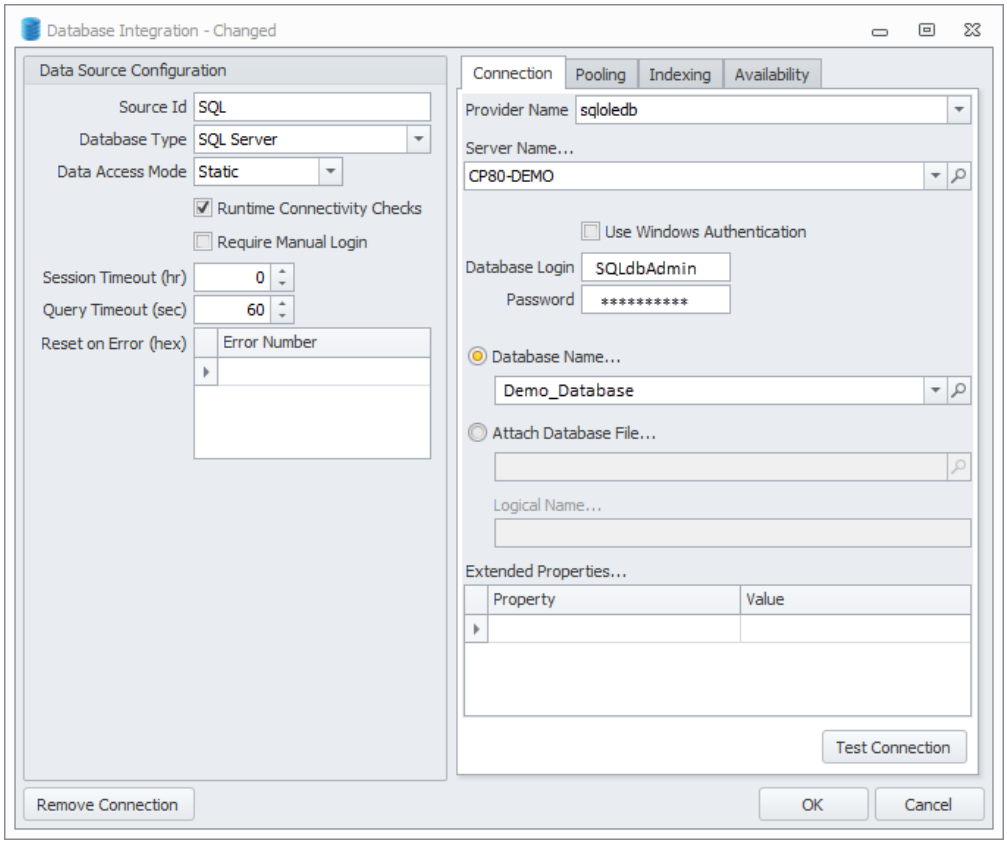
   Note: If you are creating an Enterprise connection for the first time, refer to the online help for Adding a new enterprise connection. Then return to this process.

2. The Enterprise System Integration screen displays.
3. In the Connections tab, check the Enable SNC box.
4. Enter the SNC name.
5. Enter the path to the SAP Cryptographic Library (where SNC dll is stored).
6. Click on **Test Connection** .
7. If your test is successful, click **OK**.

Configuring for SQL Server

If your data is formatted in the sequence query language (SQL) and stored on an SQL server, then the connection to SQL data must be added to RFgen. Complete the left panel, **Data Source Configuration** as you would any of your other data sources except that the database type will be set to SQL Server.

Note: Use the **Reset on Error (hex)** table to force the RFgen server to automatically reset its connection with the SQL Server if the SQL server stops responding or drops its connection and sends the exact error code / error number listed in the table. The code can be in decimal format or hexidecimal format. A common error code to use for the Microsoft SQL server is 0x80004005.



In the **Connections** tab (right pane), complete the **Provider Name** and **Server Name**.

The **Use Windows Authentication** if checked enables authentication of the user login through Windows Active Directory when the user logs into the database.

If the **Use Windows Authentication** is left unchecked, the database login information is passed directly to the server listed above the Database Login/Password fields.

If you want to connect directly to the SQL Server Database MDF file itself, specify the **Attach Database File** option and locate the database file directly.

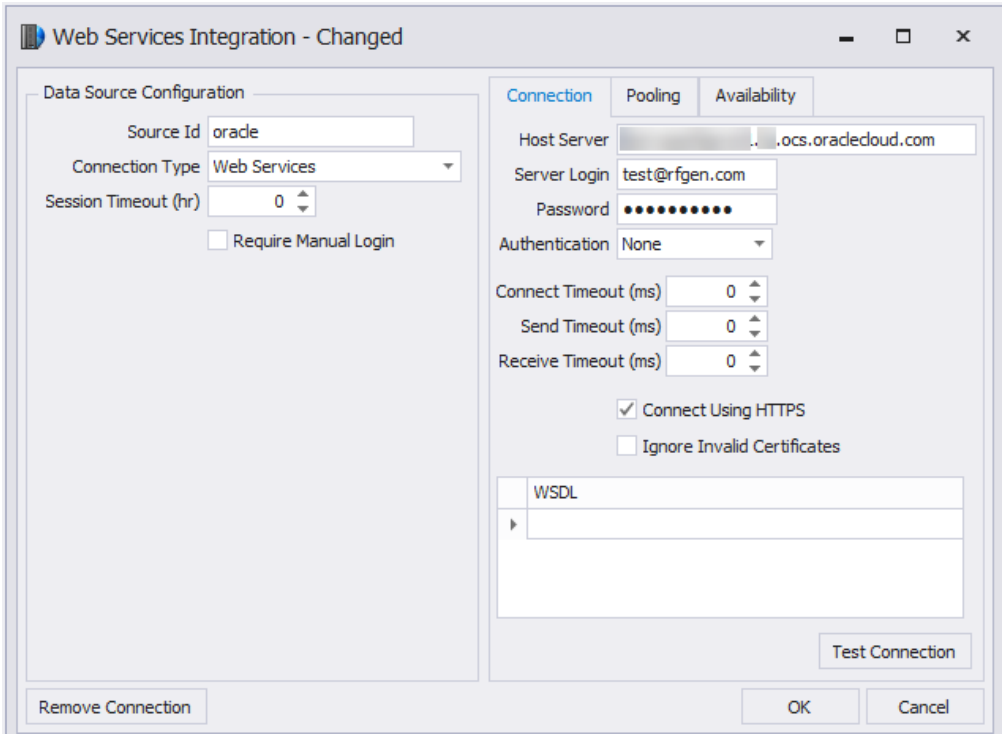
The **Logical Name** is typically the filename without a file extension and should not be necessary.

The **Extended Properties** are usually not required.

Click on **Test Connection** to verify the connection to the data source. If the database has not already been setup to support the solution tables, they will be created at this time.

Clicking on **OK** will also create what is necessary, but won't test the connection. Either button will recognize if the database came from an older release of RFgen, and ask if you want it upgraded.

Adding A New Web Services Connection

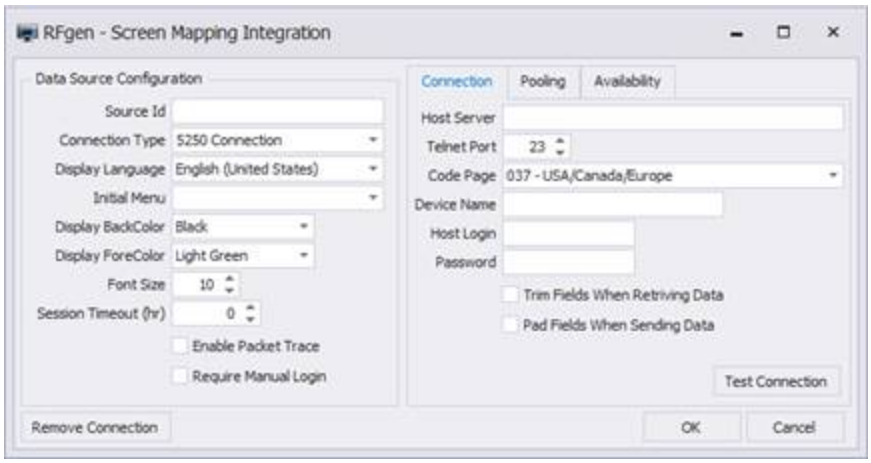


The web service integration screen can be used to connect with the Oracle SCM web service provider.

For other ERPs, the web service is not supported.

Configuring the Host Connection

In the Mobile Development Studio, click on **Enterprise Connections > Add New Enterprise Connection > Add New Screen Mapping Connection**. The following window will appear.



The first entry is the **Source Id** used to reference the data connection only. This can have any value but spaces and extended characters are not recommended.

Choose the **Connection Type** (VT220, TN5250 or TN3270); i.e., the protocol used to communicate with your host system. Notice that there is an additional option called Console Application. This type is designed to launch a console application rather than use a telnet server and then pass that display through the server to the device using the HostScreen prompt control. One example would be the SAP console application (SAPCNSL.EXE) running on the server and being displayed and allowing interaction with the user on a mobile device. Simply specify a process or executable name to run and any passing parameters necessary.

The preferred option is UTF-8 but if a legacy system's output is language specific then the **Display Language** field should be changed to make the screen render correctly. The Language field can be left as (Default) if a code page is specified or if UTF-8 is used.

Preferences for the emulation screen include the **Back Color**, **Fore Color** (the color of the font) and **Font Size**. These are only for development since the screens are hidden during production.

The **Session Timeout** value (in hours) will disconnect and reconnect to the legacy server at the specified interval. This may be required if the legacy server is configured to not allow a connection that never times out.

In the case of communication errors the **Enable Packet Trace** option can be set and a trace log of the communication will be captured. This is used by support staff to diagnose issues on behalf of the customer. Please contact support if this switch is necessary.

If the **Require Manual Login** is checked, a connection request is created between the user and the ERP system. If this box is unchecked, the user login uses the ERP connection between RFgen and the ERP system.

Connection Tab

Next, type in the **Host Server** name or IP address. The **Telnet Port** is the port that the server uses to communicate with your host. The default for a telnet server is port 23.

If TN5250 or TN3270 are selected, you may enter a **Code Page** for specifying the language being used in the protocol and an IBM **Device Name** for the host system. Code pages were selected for loading when you loaded the screen mapping software. These fields are hidden in the VT setup.

For VT220 the **Data Stream** field can be set to either Standard or UTF-8 to accommodate the type of packet data coming from the host system.

When using the connection type 3270 or 5250, the **Device Name** field is designed to make each connected device appear unique to the host system. Leaving it blank, the host system will not distinguish between the connecting clients. Fill this field in with a name and the server will automatically add a three digit, zero padded number to each client so the host system will see each connecting session as a unique device.

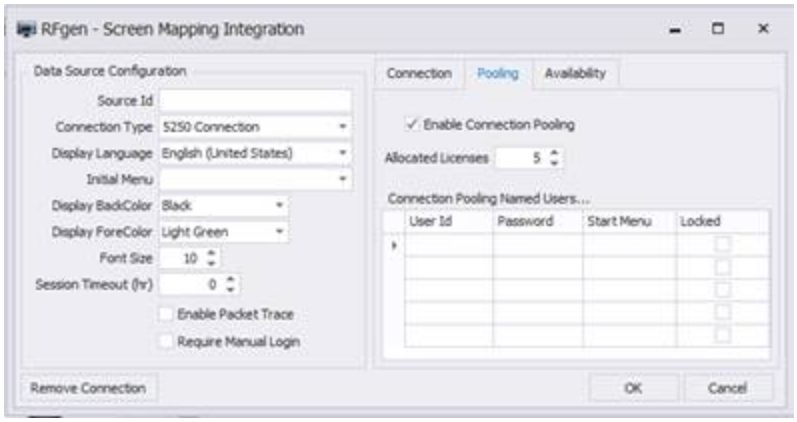
The **Host Login** and **Password** fields are used only if SSH is used when connecting to the host system. Under the VT220 options, if Connect via SSH is checked then the Host Login and Password are required.

Trim Fields When Retrieving Data set to enabled will auto trim spaces from the host output fields. If a variable is defined for a section of the host screen (like where error messages are displayed), this feature

will trim the text for easier use in message boxes, for example.

The **Pad Fields When Sending Data** option when enabled will use spaces to pad any input. A variable defined for a region of the host screen where input will take place also has a length property assigned at the time the field was defined. If the data is 3 characters, but is placed in a host screen field designed for 10 maximum characters, the server can pad the input data to fill up the host screen input field.

There are some additional properties for the VT220 mode only. **Echo Characters Locally** means that the server will print the typed characters on the telnet screen because the host is accepting the keystrokes but not showing them to the user. **Wrap Text at End of Line** will force the server to place the additional text on the next available line if it doesn't fit in the current field. Most host system will do this automatically. **Destructive Backspace** means that the server will receive a backspace command and apply it to the screen as a command that removes the last character. Some systems would move the cursor but not remove the character. **Send Whole Key Packets** forces the server to submit keystrokes in one packet instead of two in some cases. Most host systems already support keystrokes coming in as one or more packets. **Send Return + Line Feed** will add a carriage return plus a line feed to the Enter keystroke when communicating with the host. **Connect via SSH** will establish an SSH (secure) connection to the host from the server. If this option is turned on then the SSH **User Name** and **Password** fields will be required.



Pooling Tab

Connection Pooling can be enabled and the maximum connections allowed in the pool can be selected. This selection will determine how the server and its clients will interact with your host system.
The options for the **Pooling Status** are:

Disabled – Setting connection pooling to disabled will cause the server to spawn a connection to the host system for each active mobile device. Each connection will be linked to a particular device on a one-to-one basis, and will be shut down when that device disconnects. Note: there is no limitation on the number of connections allowed.

Enabled – Setting connection pooling to Enabled will cause the server to spawn a single connection to your host system. As each device requires access to the host system, they will go to the pool and retrieve one of the available connections. When they are finished, the device will release the connection back to the pool. If no connections are available, the server will start a new connection (up to the specified maximum) and add it to the pool. After 10 minutes of non-use, an opened pooled connection will be terminated releasing resources on the server and potentially licenses on the host system. Keep in mind that unless the

SM.BeginTrans and SM.CommitTrans commands are used, it would be possible for one user to position the screen in one place while another user also uses that pooled connection to perform their tasks causing both users to get failures.

The **Connection Pooling Named Users** grid dictates how each host session is started. You may also override the default settings by configuring a specific pooled session separately.

Session - Each of the individual pooled connections are listed separately. This provides for specific settings for each connection.

User Id- If the host system requires that unique names be used or creating multiple logins with the same user is prevented, each pooled connection can have its own user ID. Session, user, and password information can be obtained at runtime with the commands SM.SessionUser, SM.SessionPwd, and SM.SessionID.

Password- This is the corresponding password used for each unique user ID.

Start Menu- Each session can have its own main menu. When a session is requested and no main menu is specifically assigned or the "(Default)" value is used, the next available session will execute the requested main menu based on the scripts and chosen transaction. If a session is requested and the next available session does have a main menu assigned, and it is not the required one, other sessions will be evaluated for a matching main menu. If one is found and available, it will be used.

Locked- The ability to lock a session means that the session can ONLY be used with the specified main menu and will not allow other main menus, even if all other available sessions are in use. For example, there are 10 pooled sessions, five locked on main menu A and five locked on main menu B. If a session with main menu A is requested and all five sessions for main menu A are currently used, the server will look to the sessions assigned to main menu B. If they are not locked, the server will take one of them. Since they are locked into main menu B, in use or otherwise, the server will wait for one of the first five to be released.

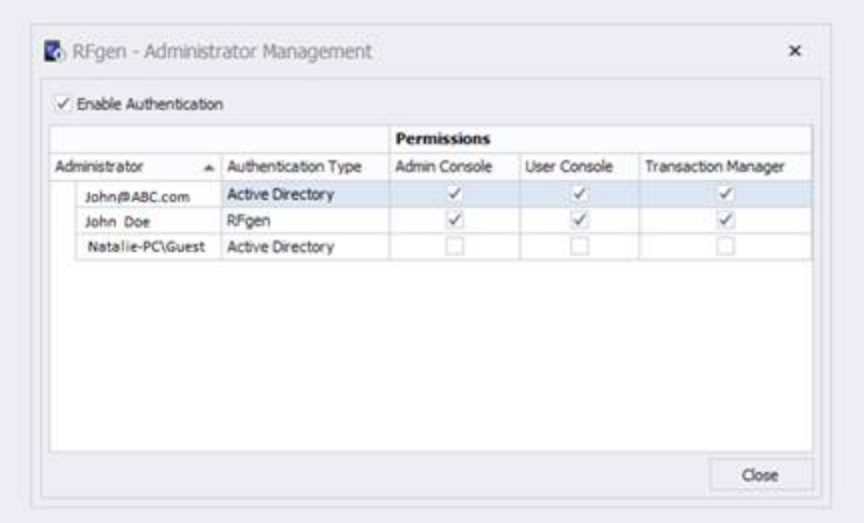
The purpose of locking a set number of sessions to a specific main menu is to ensure that there is always some bandwidth available for certain transactions. Not locking them means that they will be marked with a preference for a type of transaction (the use of a specific main menu), but will switch to another main menu when necessary.

For example, there are 10 pooled sessions available and the first five have one main menu assigned and the last five have a second main menu assigned. When a session with the second main menu is requested, the 6^{th} session handle will be used. This is only significant because of the Locked property.

The **Test Connection** button will verify all settings before saving the connection. This is not required.

The **Save** button will save changes but will not test and verify settings.

Configuring User Access Control



The **User Access Control** feature restricts administrative access to RFgen consoles and dashboards if they are configured to authenticate the user via the credentials saved in the RFgen server, or Microsoft Active Directory .

To prevent unauthorized access, the dashboard / console must be connected to the same application database that was used to configure User Access Control in the RFGen server.

For example, if you had two application databases -- one called Test, the other Production -- and User Access Control was only setup in the RFgen server when it was connected to the Production database, if the console / dashboard was connected to the Test database, the user will not be prompted to enter his/her credentials. But, it the console / dashboard was changed to use the Production application database, the user would be prompted.

To Enable Authentication

This feature is used for restricting access to the:

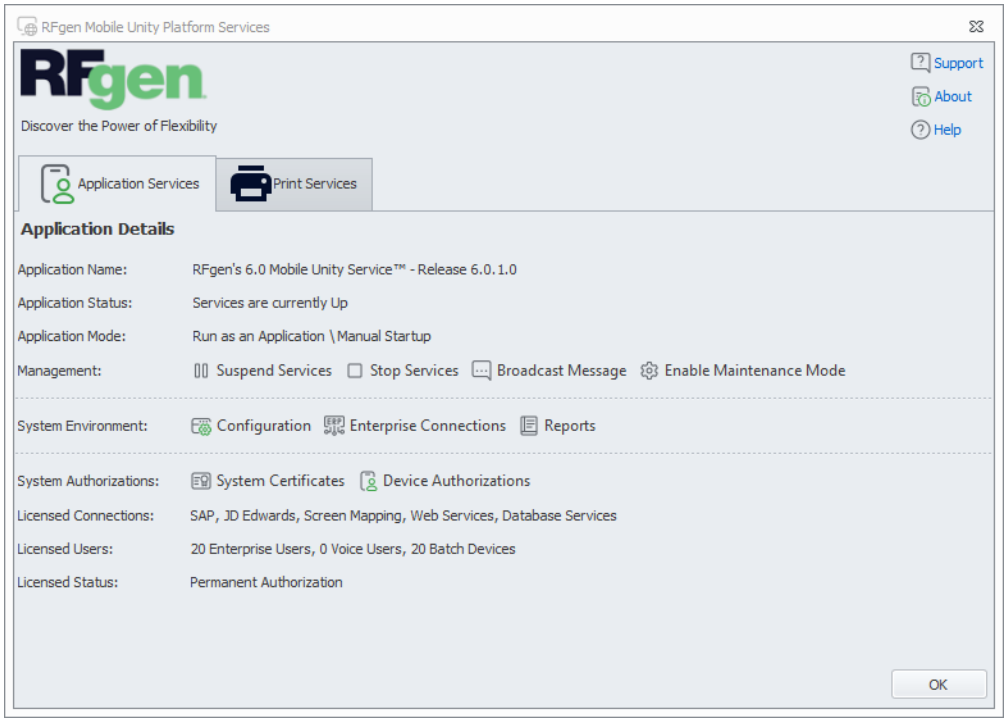
- *Admin Console* (the Mobile Enterprise Dashboard)
- User Console (User Management Console)
- Transaction Manager (Transaction Management Dashboard)

1. In the **Mobile Dev Studio**, click on **Configuration > User Access Control**.
2. Click on the **Enable Authentication** box.
3. Right-click on any of the column headings (or white space).
4. Select the desired action to add, edit, or removed an administrator.
5. If adding, enter the user's information, then click **OK**.
6. Check the box for one of the following that the administrator will be allowed to access.
7. Click **OK** to save your changes.

Update the Target Dashboard / Console

1. Launch the dashboard / console.
2. In the Configuration or Connection menu, select Application Database.
3. In the Data Source Configuration > Configuration File, click on the down arrow.
4. Select the application database that has the User Access Control settings.
5. (Optional) Click Test Connection. Click OK.
6. The application database should display on the lower left corner.
7. Click **OK**.

Services Overview



The **Mobile Unity Platform Console** (Services console) is a graphical interface for administration of your server services at a glance. The Services console also allows you to:

- View Printer Services (available in RFgen 6.0.1 and higher)
- Identify the version of RFgen installed on the server.
- Check the running status of server services. It tells you if your services are up or down, and whether the server is in Maintenance Mode.
- Stop or start the server service and allow or disallow users to connect (Maintenance Mode), and broadcast a message to all connected users.
- Configure the server and setup connections to database systems and enterprise (ERP systems).
- View reports (application event logs) and export them to Excel.

- View server authorization status, number of licensed uses. If you server is not yet activated, it can be activated through here. (See Web Authorization.)
- Under Device Authorizations, view which mobile devices are connecting with your server and controls which ones are authorized to connect.

When started, the Server enables multiple communication sessions between your server and your remote devices (up to the number of authorized users.

If you have load balancing configured, you will also see a Load Balance Status panel near the bottom of the screen.

System Environment > Configuration > Application Services may be used to specify a different port. The service will administer to all types of clients (GUI-based devices, XML, Vocollect, etc.) simultaneously.

Related Topics

Configure Your RFgen Server or Mobile Development Studio.

Print Server Console Overview

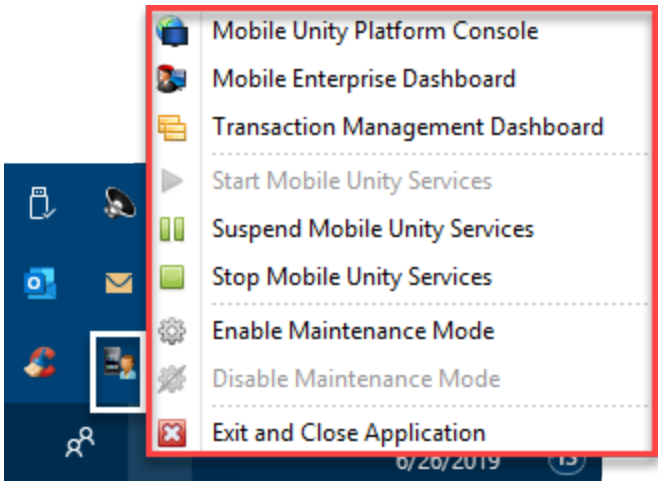
For the Client - Server connection process, see Basic Implementation Steps.

Accessing the RFgen Server Console (Mobile Unity Platform Console) and Services

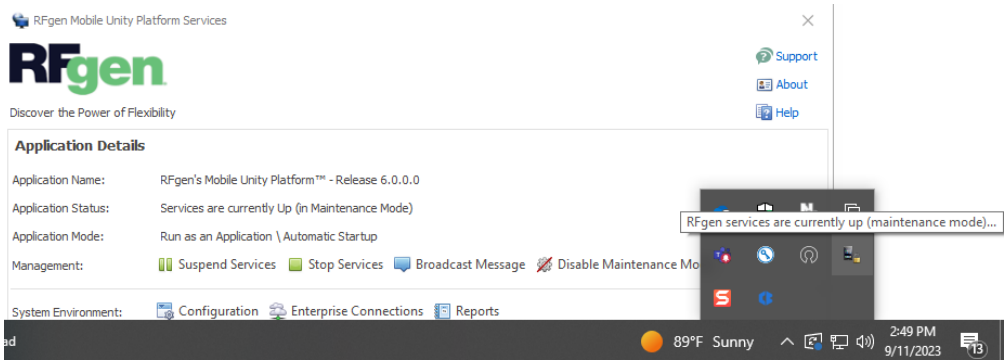
If you want to display the RFgen server console from your Microsoft Windows System Tray, follow these steps.

For this procedure, first add your Mobile Unity Platform console icon to the Windows System Tray. For details refer to your Microsoft Windows System Help.

1. Click on the Mobile Services Manager icon  in your Windows system tray.
2. The Mobile Services Manager Menu icon displays.



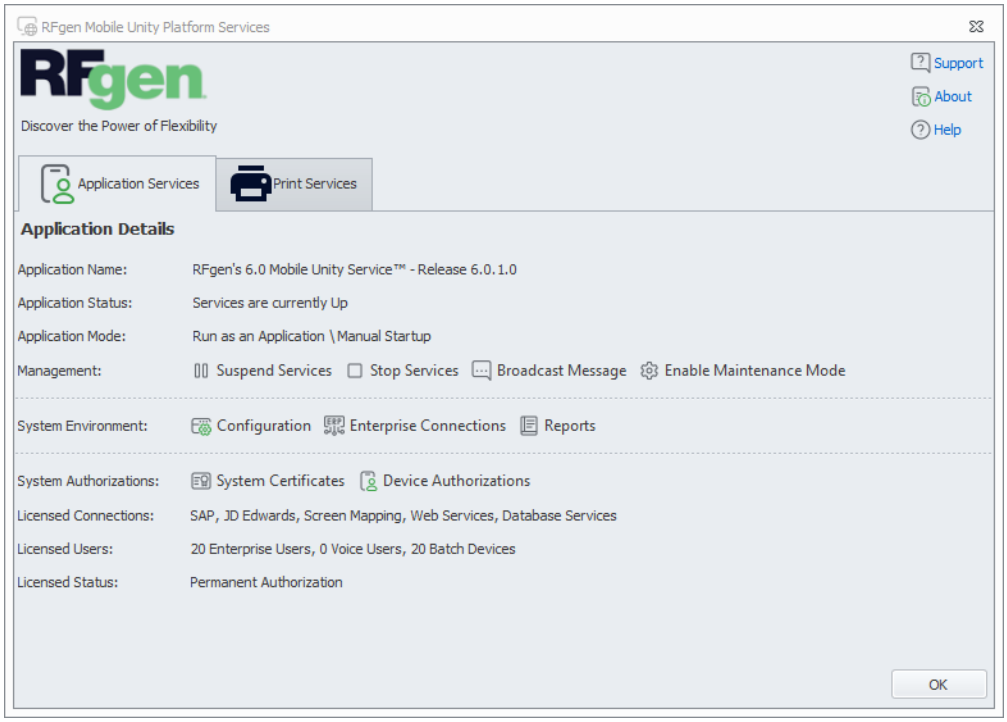
3. Click on the Mobile Unity Platform Console.
4. The RFgen Mobile Unity Platform Services Console displays.



Management of RFgen Server Services

The Mobile Unity Platform Services console provides a summary of your server's status and functions to change its operational status.

To run the application services status is shown as currently down or suspended, it may need to be license and authorized.

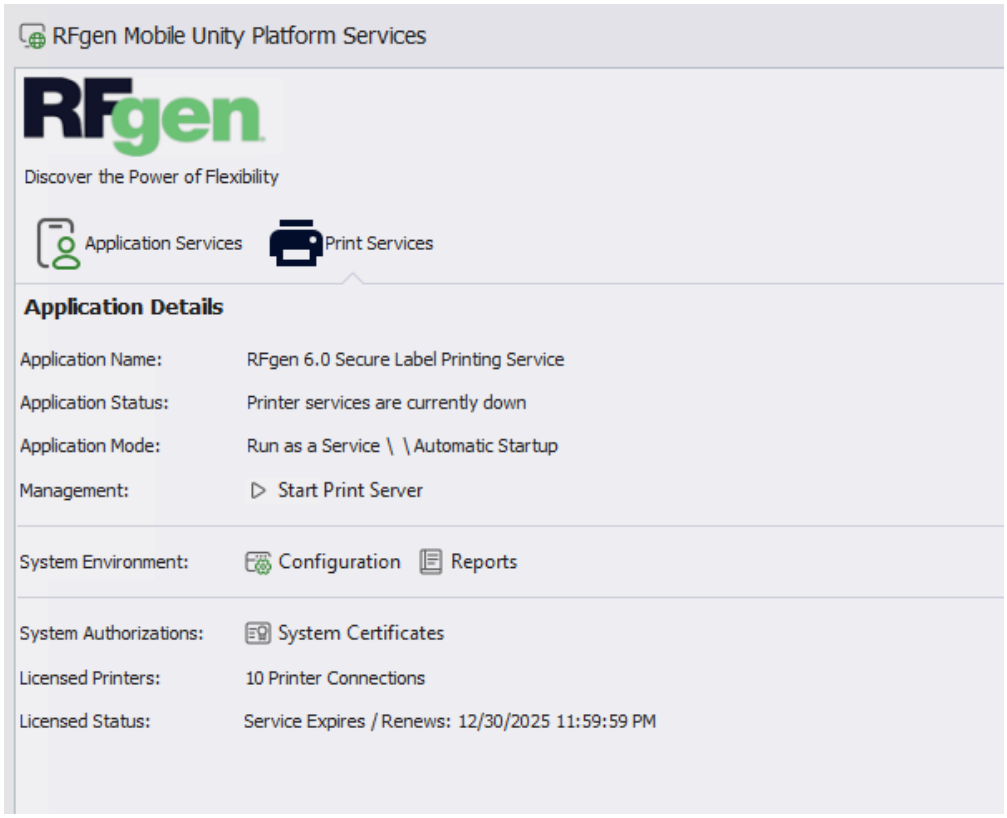


Suspend/Stop/Start/Resume Services If you need to stop or suspend your services, simply click on the appropriate button. The different between a suspension and a stop is that if you Stop Services, your server goes through a longer process to restart the service than if its simply paused. You turn on your services via the Windows Task Bar menu.

Note: You can also use commands to start or stop RFgen server services.

Management of Print Services

The Mobile Unity Platform Services console includes the **Print Services** tab for RFgen versions 6.0.1 and higher.



The Secure Label Print Services tab lists:

- The version of RFgen installed on the server.
- The status of the printer server service. (Automatic Startup is the only mode supported at this time.)
- The print service Start/Stop button.
- The Configuration and Reports menu (common to the Application Services). Use this to set Printer Services under Environment Settings.
- The print server System Certificate authorization status
- The number of licensed printers this server is connected to.

To run the print services status the connected printer must be licensed and authorized.

Stop/Start Services If you need to stop or start your services, simply click on the appropriate button. You turn on your services via the Windows Task Bar menu.

You can also use these commands from a Command Prompt where the RFgen files are installed (i.e. C:\Program Files\RFgen60:

RFSLS600 -Start

Mobile Unity Platform Console: Reports Menu

The **Reports Menu** provides views to performance and application/system events that occurred in the time set at the top of the screen.

While events are turned on by default, to view performance metrics, you will need to enable the values in Environment Settings.

- The **Application Logs (Logfiles)** displays system errors including connection errors, changes to the RFgen system environment, RFgen client user login/logoff events, and changes that occur to specific modules in the RFgen server or Solution Explorer. This information can be export to Excel, and filtered using SQL statements.
- The **Performance Monitoring** log lists events generated by the execution of scripts that exceeded a threshold value. (i.e. Flagged events that exceed processing time thresholds.) It includes Display Options and an Export to Excel tool.

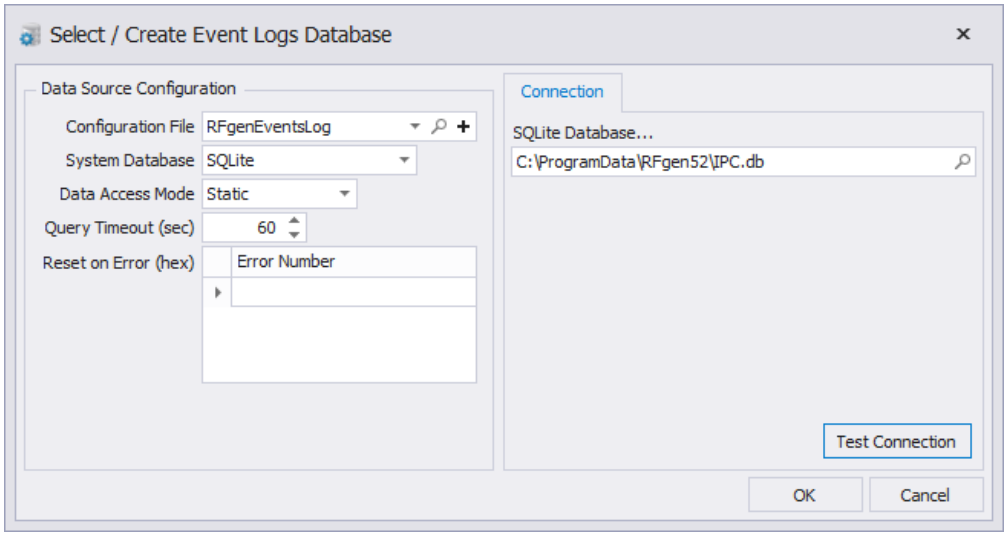
For more details on configuring and/or connecting to the source files, see:
* To configure an Event database

* To configure Performance Monitoring

To Configure Event Logging or Create an Event Database

By default RFgen has event logging turned on, and the output captured in Dev Studio or the Mobile Unity Platform Console, click on **Reports > System Event Logs / Application Logs**. The factory provided events and performance logging is in the IPC.db file which is located in the **ProgramData\RFgen 52** directory.

Use the Create Event Logs Database screen to create your own database and connect to it from RFgen.

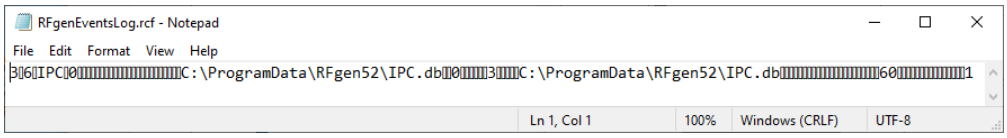


You can create your own database by copying the IPC.db (SQLite), or create your database from Access or Oracle, or connect to an existing database hosted by another server (i.e. SQL Server).

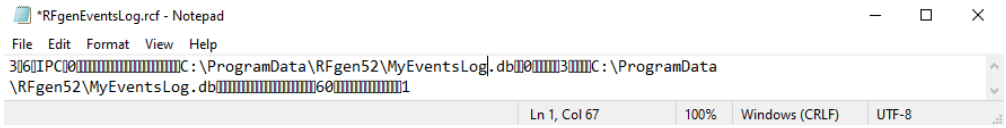
Create database in SQLite

1. From your Windows system, copy the IPC.db file and rename it. This file is typically located in your **ProgramData\RFgen 52** directory. For example "MyEventsLog.db" and save it in the same location as the IPC file.
2. From your Windows system, create another .rcf file, rename it.
3. Open the file in Notepad or a similar editor, and change its path to the location of the database created in step 1.
   For example change the "IPC.db" to "MyEventsLog.db".

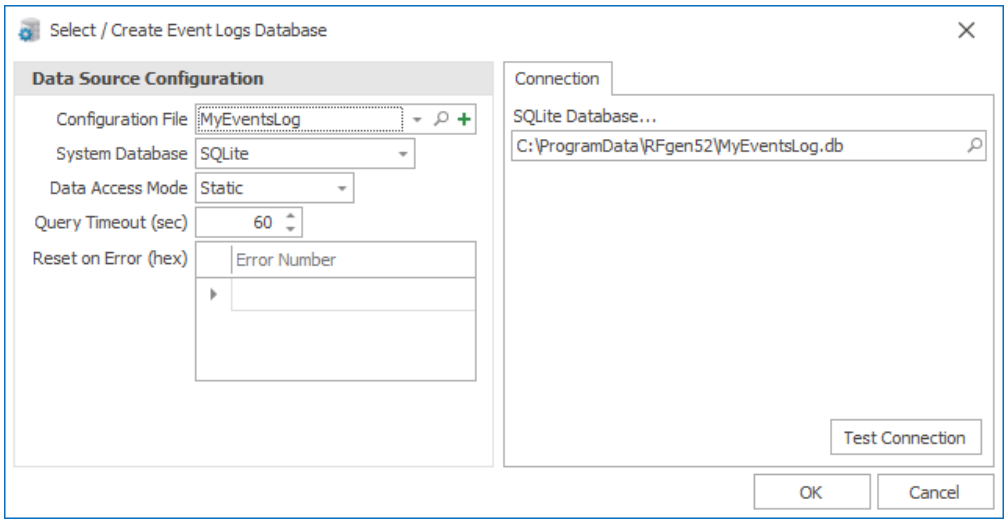
   Before:



   After:



4. Save the rcf file and make sure the extension is .rcf (not notepad.)
5. Click on the **Mobile Development Studio > Connections > Application Event Log** screen, or the **Mobile Unity Platform Console > Configuration > Application Event Logs** screen.
6. In the **Configuration File** field, click the search icon, and locate the new rcf file. In this example, select "MyEventsLog".
7. The Connection tab should populate with the database created from step 1.



8. Click on **Test Connection**. A Connection Successful message should display.
9. Click **OK**.
10. View the output from **Reports > System Event Logs**.

Related Topics

For more details viewing and filtering log files, see the System Event Log topic.

To Setup Performance Monitoring

To view data in the Dev Studio Reports Performance Monitoring screen, the following items will need to be setup:

1. Setup a connection to a transaction database

2. Setup connection to the events database

3. Set threshold values so to trigger data to be captured in the events and database monitors

4. Run test application/data so to verify the monitors are receiving the data.

Before you start

- Will need a database application. RFgen supports Access, DB2, ODBC, OleDb, Oracle, SQL Server and SQLite.
  (SQLite is the easiest and most common application to use.)
- You will need the path to your RFgen Program Data file so to connect to the IPC.db and the RFgenTM. These are typically located in the Program Data\RFgen 5.1\IPC.db file or the Program Data\RFgen 5.1 file.
- Your own test data to verify the output in the Performance Monitor. (i.e. Use one of your applications.)

The IPC.db file contains performance data and events.

Setup Connection to Transaction Database

In this example we will setup a connection to a Transaction database using SQLite. This database will be used to track the transactions if the transaction executions trigger one of the thresholds set in Environment Settings > Performance Monitoring.

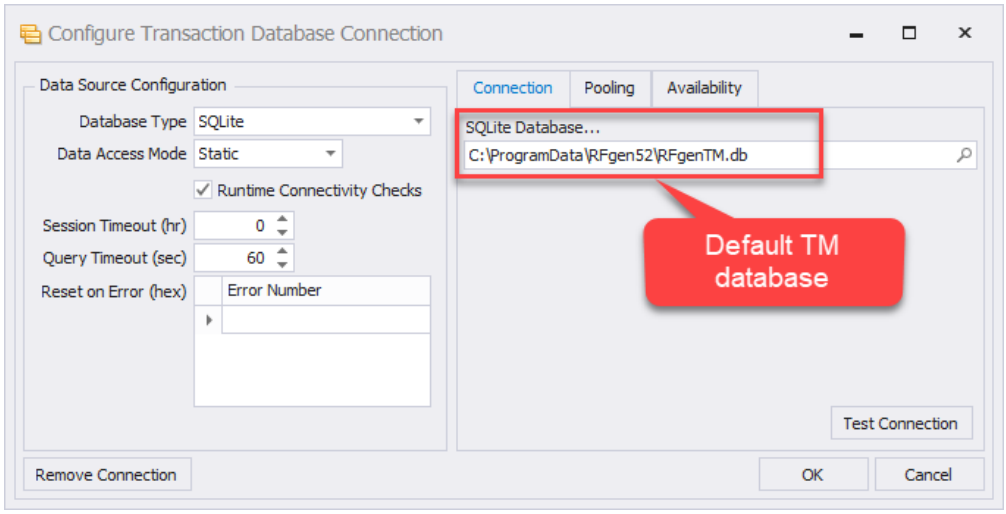
These steps can be performed in the Mobile Development Studio or in the Mobile Unity Platform Console.

You can use any standard database; for the purposes of this example, we used Microsoft SQLite.

1. Open **Connections** > **Transaction Management Database**.

2. In the **Configure Transaction Database Connection**, **Database Type** menu select **SQLite**.

3. In the SQLite Database... search, enter the path or search and select **RFgen TM.db**



4. Leave the other values the same for now. Click on **Test Connection** and verify you have a good connection, then click **OK**.

5. A small RFgenTM icon should display in the lower left corner of your screen.

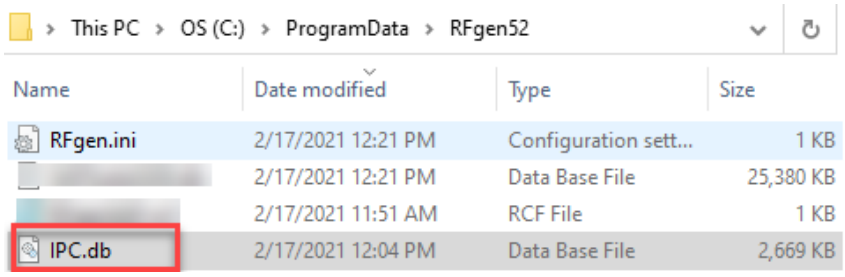
Setup Your Events Database

1. Open **Connections** > **Application Events Log**.

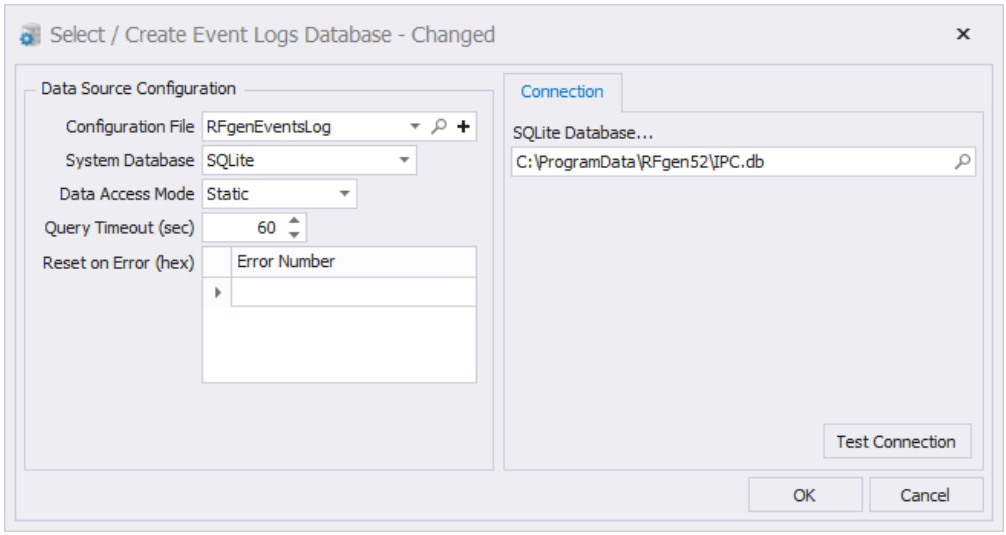
2. In the **Select / Create Event Logs Database**, **Configure File** field click on the **+** sign. In the pop-up Window, enter a configuration file name. The configuration file can be any name as long as there are no spaces in the name. For example, "RFgenEvent". RFgen will automatically default the extension to .rcf. Do NOT use an rcf file that is purposed for your rfgen applications as the format of the database would be different than the format used for the rfgen applications database.

3. In the **Sytem Database** drop down list, select **SQLite**.

4. In the SQLite Datase field select the Search button and select **IPC.db** then click **Open**.



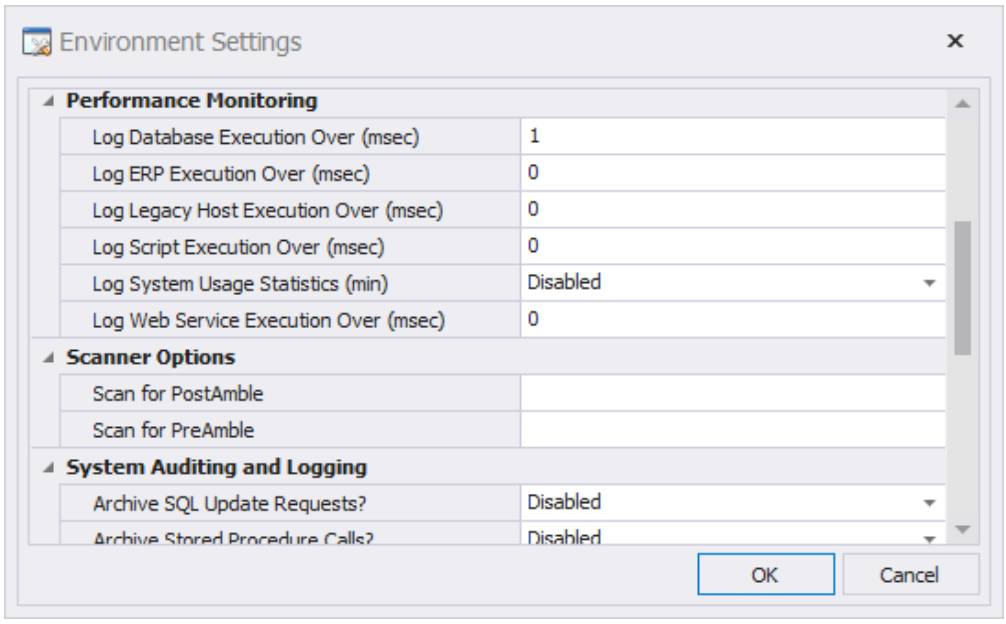
5. The screen will look similar to this screen below.



6. Click on **Test Connection**. A Good Connection or Connection Successful message should display. Click **OK** to close.

Set Performance Monitoring Thresholds

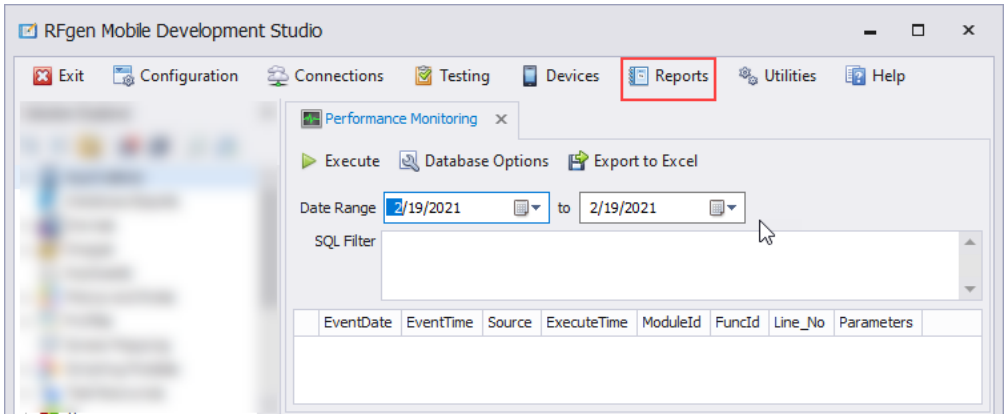
1. In the **Configuration > Environment Settings > Performance Monitoring** table enter a 1 in the Log Database Execution Over (msec) table.



2. Click **OK** to exit.

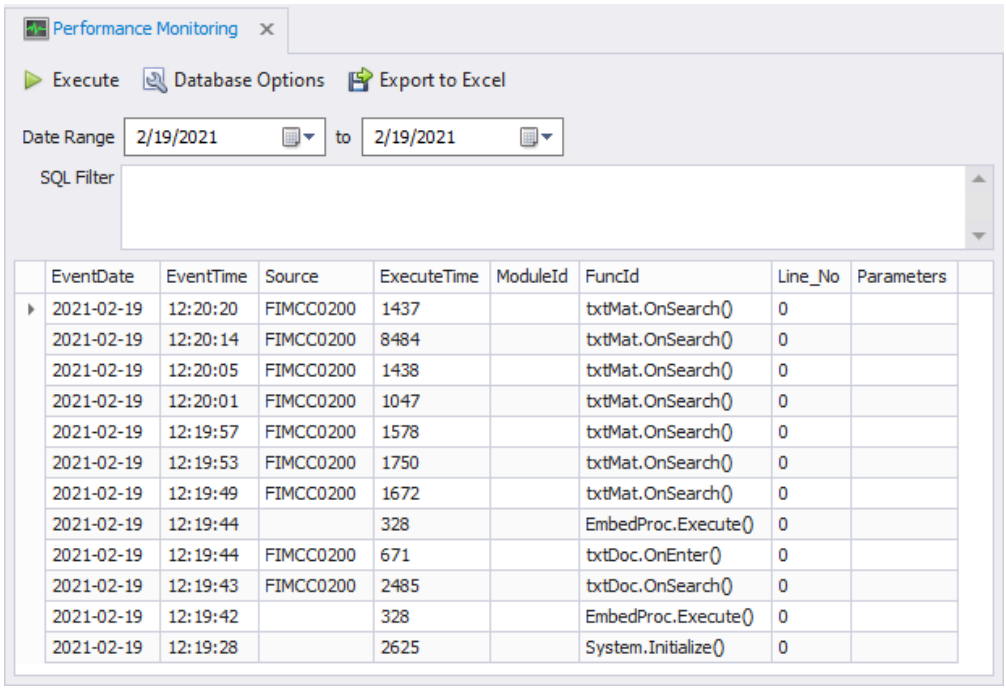
Remember -- In real environments, these thresholds would be set for a specific purpose and value. Arbitrary values would otherwise cause constant logging and added traffic to your environment.

3. Open your **Reports > Performance Monitoring Screen**.



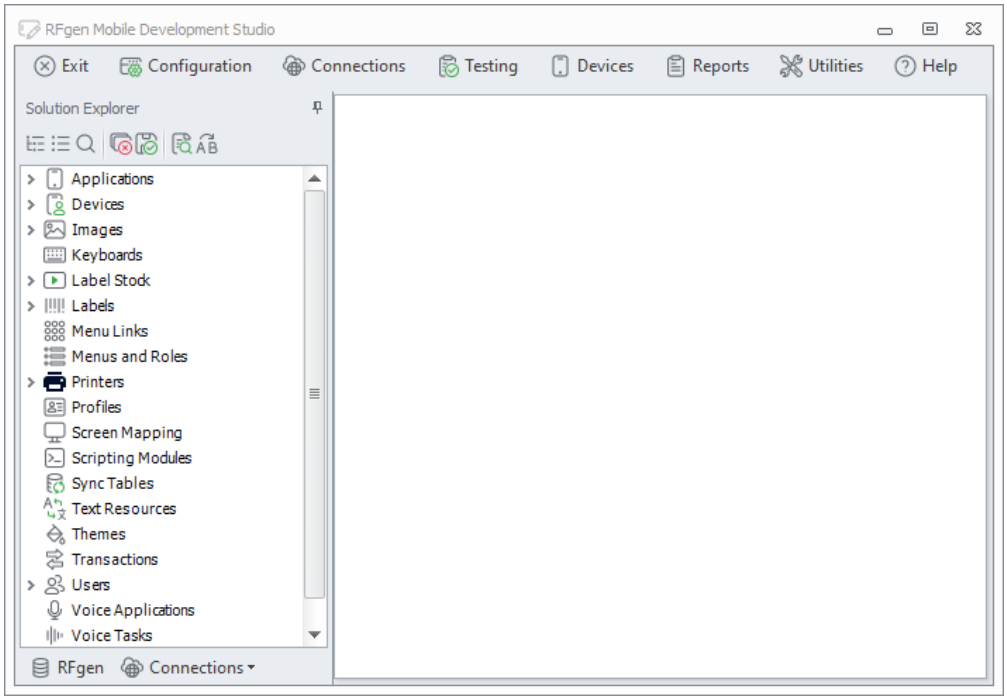
4. If you have been running transactions since you connected to the two databases above, click on the Execute button to view your data.

The screen may look similar to this one below. However, if you do not have a transaction to generate data, see the next process To Setup Test Data.



EventDate	EventTime	Source	ExecuteTime	ModuleId	FuncId	Line_No	Parameters
2021-02-19	12:20:20	FIMCC0200	1437		txtMat.OnSearch()	0	
2021-02-19	12:20:14	FIMCC0200	8484		txtMat.OnSearch()	0	
2021-02-19	12:20:05	FIMCC0200	1438		txtMat.OnSearch()	0	
2021-02-19	12:20:01	FIMCC0200	1047		txtMat.OnSearch()	0	
2021-02-19	12:19:57	FIMCC0200	1578		txtMat.OnSearch()	0	
2021-02-19	12:19:53	FIMCC0200	1750		txtMat.OnSearch()	0	
2021-02-19	12:19:49	FIMCC0200	1672		txtMat.OnSearch()	0	
2021-02-19	12:19:44		328		EmbedProc.Execute()	0	
2021-02-19	12:19:44	FIMCC0200	671		txtDoc.OnEnter()	0	
2021-02-19	12:19:43	FIMCC0200	2485		txtDoc.OnSearch()	0	
2021-02-19	12:19:42		328		EmbedProc.Execute()	0	
2021-02-19	12:19:28		2625		System.Initialize()	0	

Mobile Development Studio Overview



The **Mobile Development Studio** provides all the tools you need to create a mobile solution, test your solution, and manage the client after the RFgen client software has been installed. Below is a quick view of the ribbon menu items.

The top menu enable you to: a) Configure the RFgen software; b) Setup Data Source connections; c) Configure as applicable ERP or host or web object connections; d) Configure your Transaction Database connections; e) test your apps and sessions; f) manage devices (i.e. connection sessions to the RFgen server); and g) report on performance.

The **Solution Designer** panel provides the tools to:

- Design and create mobile applications for different methods of data collection.
- Design and create barcode labels and applications to print labels using the Secure Label feature.
- Design and view how your screen looks in a target device.
- Script the application.
- Define users and menus to access the application.
- Build installation files for specific client platforms.
- Store and maintain objects for reuse or reference when building applications. (i.e. Globally accessible scripting modules, data transactions, translated terms/phrases, images, and soft keyboards).

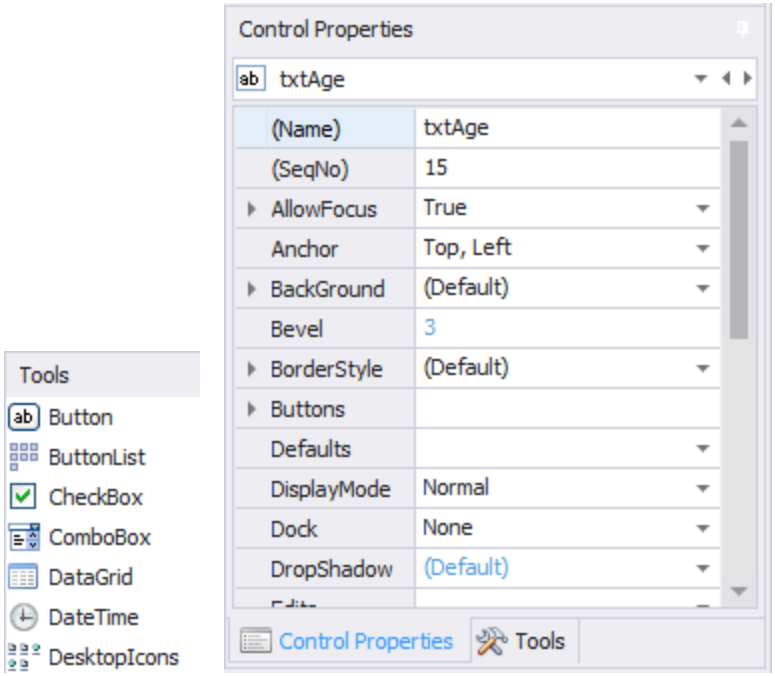
If you have special voice-driven hardware equipment, the Solution Designer can be used to create voice applications that are executed via verbal commands.

Application Tools and Controls

Applications are composed of graphical objects (Forms, Pages, and Graphical Controls), and code (scripts, functions, etc).

When you create a new application in the Solution Explorer, it adds a new **Form** and **Control Properties** for the form. The form provides multiple functions: it allows you to visualize the layout of the objects on your form, provide style elements that are consistent across multiple pages of an application, and contains the various controls which are dragged to the form.

The **Tools** panel contains graphical controls (also called "prompts") which are used to layout the user interface for your application. The Tools panel includes the Button, ButtonList, CheckBox, DataGrid, DateTime, Frame, Image, ImageList, Label, Layout, Line, ListBox, Map, Memo, Menu, Panel, PanelList, RadioButton, Signature, SpinEdit, TabControl, TextBox, RadioButton, Signature, SpinEdit, TabControl, TextBox, and TreeView Control.



The **Control Properties** panel contains the tool/control attributes (background color, size, etc). For more details, see the Control Properties Tab topic. All control property definitions are listed in the Graphical Control Properties.

Note that some controls that have unique properties not shared by other controls.

Forms and Pages. When you create a new application, it adds a form. The form's primary function is to add application displays (screen areas), set the general look and feel for the application, create a display for each locale by country if desired, and add pages which can be used to hold multiple controls (labels, textboxes, and other graphical objects). For more details, see Form Properties.

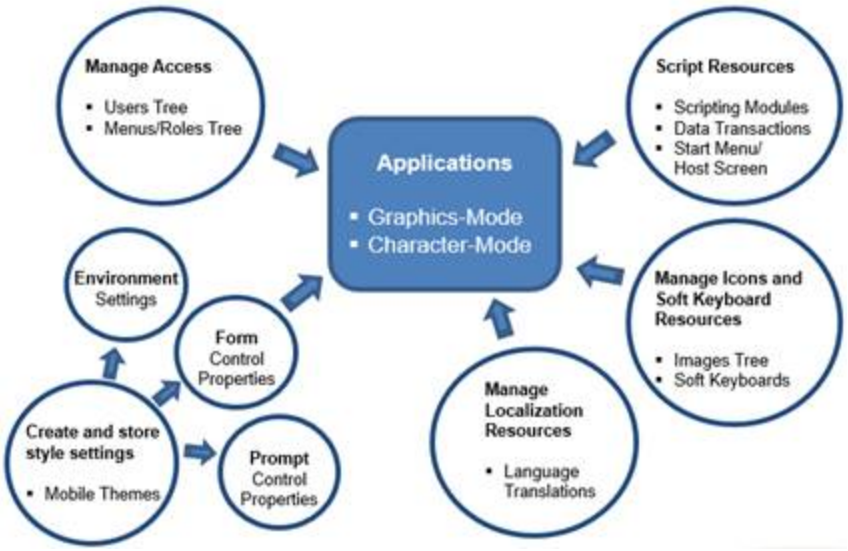
Page and Controls. By default, controls are owned by the page or form they were dragged to.
If you have multiple pages, but only want a control to appear on a specific page, drag the control on to the

page and it will only be available on that page. Controls that are on a form and made visible (or invisible) regardless of which page a user is on.

Container Controls. While most controls may be the parent of another control, RFgen also provides graphical Container whose purpose is to maintain the layout order and position of a child control. For example, the Frame, Layout, Panel, and PanelList controls are container controls. Some will display an outline (i.e. Layout control) but won't appear functional until runtime. **Note:** Some controls such as the Layout controls have unique properties when its on a Form instead of a Page.

Solution Explorer/Resources Overview

The illustration below shows the various resouces and functions that can be used to create applications and organize how its accessed.



The **Applications** node is used to design the visual aspects of your mobile application and basic validation for the data collection transaction. >

The **Data Transactions** node is used to manage host (data entry) transaction macros. See the RFgen Screen Mapping documentation or the Transactions tree section below for more information.

The **Device Tables** node allows you to setup various tables for storage on mobile devices (iOS, Android and Windows CE). These are used as a resource in Mobile Profile. This allows the user of the device to make changes to the table that is installed to the device.

The **Image** node is a collection of images that can be used with your mobile application, menus or any other interface that may require an icon or picture

The **Language Translations** node lists your translated/customizable strings that may be referenced from the code and enables you create new text translations on display forms and in the menu headers.

The **Menu and Roles** node is used to organize applications and link a menu item to an application. If

desired the RFgen Menu, can be imported from the RFgen system, and used to customize the appearance of your menus.

The **Mobile Profiles** node is used to set the values and communication information that be "packaged" and deployed to the device when the device successfully connects to the server. This tool also is used to set other function such as whether the client in the future will connect as a Thin client or Batch (Mobile Client). For more details on other optional setting, see Mobile Profiles.

The **Mobile Themes** tree lists customizable Android, Apple and Standard theme configuration settings for the mobile devices. It also serves as a centralized location for managing the defaults for common styling of controls. (i.e. BackColor, Fonts, Bezels etc.)

The **Screen Mapping** tree is used to add new hosts and capture screen (navigation) macros.

The **Scripting Modules** tree is used to manage global scripts that can be used by any application, timed event, transaction macro, etc.

The **Soft Keyboards** tree is used to create new soft keyboards for mobile applications as a supplemental means of entering data.

The **Target Devices** tree is used to contain factory-provided images of devices or add new device images. These images are also called "device skins" as you can then use RFgen to visualize how your application will appear in the target device.

The **Users** tree is used to provide user names, passwords, and a primary menu for each user.

The **Voice Application/VoiceTasks** tree contains pre-defined scripts that execute on Vocollect hardware and interact with RFgen to provide the voice solution together with the backend data solution. Voice applications are unique in that they can only be developed with the scripts designed to execute tasks. The use of other resources such as the prompts/controls from the Tools tab, images, soft keyboards, etc. cannot be used to develop a voice application.

Database Exports

The **Solution Explorer > Database Exports** feature (previously called "Device Tables) allows you to preconfigure data tables for export to a Profile where they can be downloaded and used when the client is in batch or offline mode. For example, if the user is collecting data on an oil rig in the middle of the Pacific ocean, and the client has no connection to the company's network, the application can still access and list the specific ERP data that's needed to process a business task if the client's profile contains the database and tables required for the task.

You can setup an empty database or a database with the desired table(s), and also specify how it will be provisioned on the device.

For more details, see To add a database for export to client.

The **Table Name** is the name or identification of the table to be exported.

The **Description** is a required field describing your table.

Schema Mode sets how the table will be created. There are three options: *Data Only*, *Create Always* (default) and *Create on Change*.

- *Data Only* means that RFgen will not create the table at all. It must exist in the database already. The Data Mode property will determine how the data is filled in.
- *Create Always* means if the table already exists, it will be deleted and recreated on the mobile database.
- *Create on Change* means that the existing table schema on the mobile database will be compared to the equivalent table schema on the server. If there is a difference, the existing table schema on the mobile database will be replaced with the updated table schema from the server.

Data Provisioning specifies how the data will be replaced when the client profile is reprovisioned. The options are: Clear and Copy (default), Copy Only and Schema Only.

- *Clear and Copy* will delete the contents of the table and re-populate it with the new data from the server.
- *Copy Only* will place the data from the server on top of the existing data. Where it is the same, there is no change. Where it is different and depending on the keys, either it will overwrite data or insert new records.
- *Schema Only* will always send an empty table to the mobile database.

Data Normalization removes leading or trailing white spaces. The options are: Trim Left, Trim Right, Both, or None.

Source Table lists the table in the database (or data spource) referenced through the **Solution Explorer > Connections** menu. This field is a drop down list of connected databases which will display the fields (columns/rows) in the Database Fields below it. For more information on displaying the list, see the topic To add a database for export to clients.

Database Fields

Use this area -- Field Name, Index, Normalize to list the fields (columns and/or rows) to be included in the table that will be exported to the client. Specifying the field(s) reduces the volume of data that will be installed to the client.

- To add a field, click on the drop down list.
- To delete a field, right click on the row and tap the Delete button from your keyboard.

Index Checkbox - If this is checked, RFgen indexes this field and makes data retrieval more efficient on the client. If unchecked, the field is not indexed.

Normalization Checkbox - If this is checked, RFgen will trim the data in the field according to the value specified above. All checked items will follow the same rule specified under Data Normalization.

Data Type - This column is for display-only purposes, and will show the field's data type.

Provisioning SQL Statements

To further enhance performance, you can enter a SQL query statement here. For example, if the table is going to be exported to client where the user collects data at a specific location you can enter your query here so the value is readily available.

Transactions

The transaction designer enables you to create and edit Timed Event macros, Host Transaction and Data Transaction macros.

The modules listed under this group are stored in the RFgen Application database but can be customized via this screen.

Note that you can also use this editor to create transactions that will run on a Batch (offline client) if you set the Procedure type to "Offline".

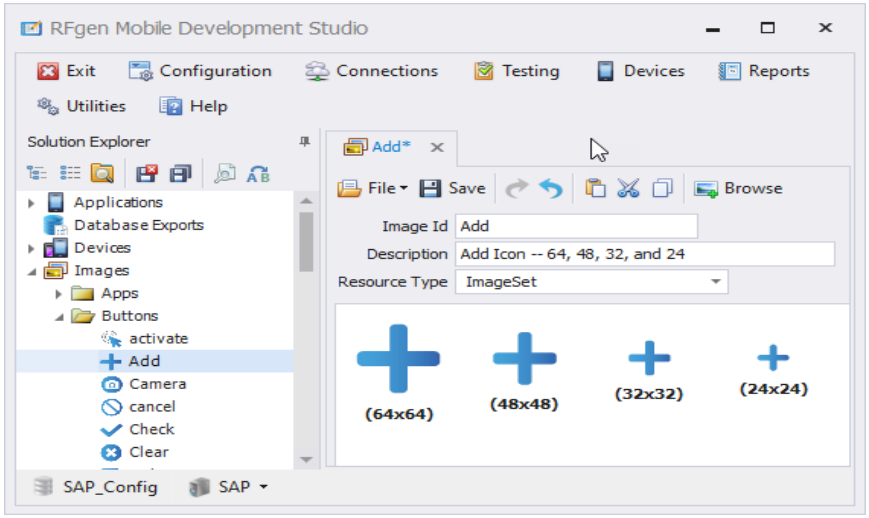
A Timed Event macro is a macro that runs on a timer configured under the Configuration / Transaction Management / Transaction Management Events dialog. If you want some script to run without a user being present, create a Timed Event macro and enable the Transaction Management capabilities. There are no passing parameters for Timed Event macros.

A Data Transaction macro is a script that can accept parameters passed in and out and can execute in a queued or non-queued manner. You would create a Data Transaction macro: if multiple applications could take advantage of the same process, you need a history of the data being processed kept by the Transaction Manager, applications are run in a Mobile environment and later uploaded to the server for processing or queuing is implemented for all applications.

For more details, see To Create a Data Transaction Macro.

Note: This feature was previously called "Data Transactions".

Images



Use **Images** folder to store the images (icons, background images, company logos). You can use the Browse button to upload any image and size. While RFgen does not set a limit on the image size, its best to

upload an image that will be closely scaled to the final viewing size within an application. RFgen will store your original size, and scale the image (i.e. menu icon) automatically or in accordance to the values manually set in the application.

The images can be referenced by Image prompts at design-time or runtime or used as part of the configuration for mobile device backgrounds. This window allows the user to drag and drop an image for quick selection.

The **Image ID** is referenced from the GUI properties or from the code to extract this image from the application database.

The **Description** is an optional field used to describe the image.

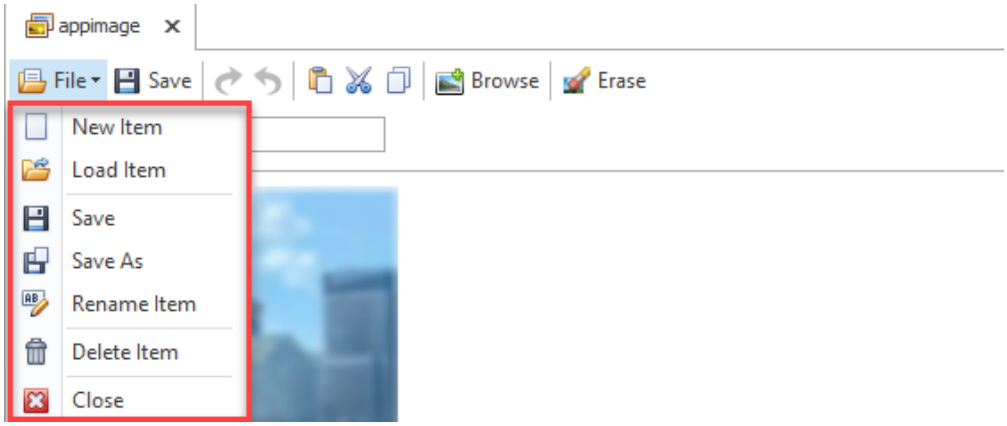
The **Resource Type** categorizes the image as an ImageSet or Animation.

The Resource Type - **ImageSets** option store a group of images and displays their size. The Animation option allows the storage of images setup for animation (movement).

The Resource Type - **Animation** option stores a series of images setup for animation (movement). The Frame Rate is how long you want the image to run in milliseconds. (100 milliseconds is the minimum.) The Frames field is the number of frames in your image. When selecting the image, set the Icon property to (Custom) and enter the Name of the image that was uploaded to your Images folder.

Use the **Browse** button to upload images.

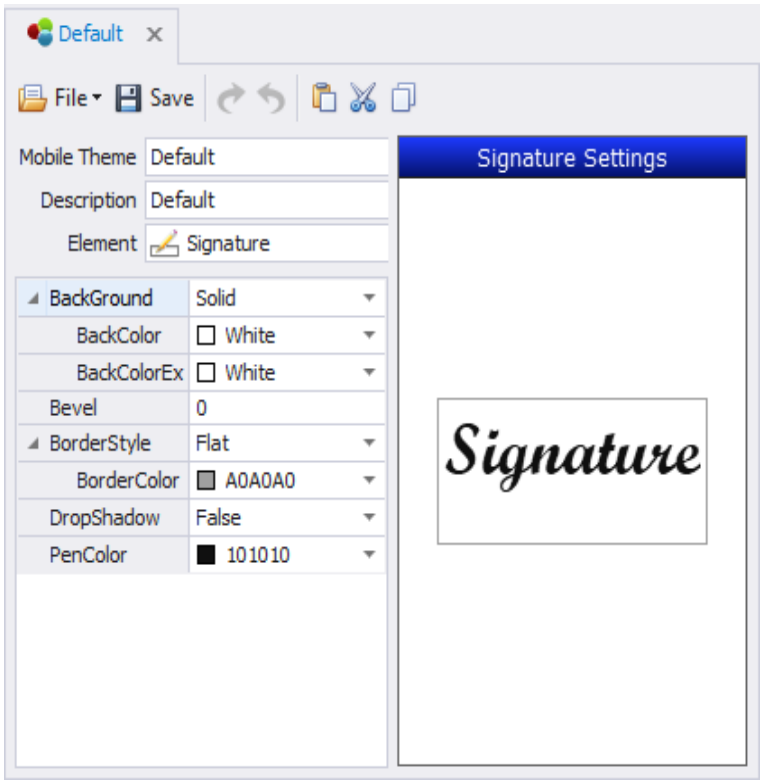
Image Menu



The **File** menu enables you to perform standard file operations.

The **Browse** icon is used to import images from your computer or other locations to the Images tree. Supported formats are jpg, gif, bmp, png and jpeg.

Themes: Signature



This sets the style for the Signature control in the Solution Explorer and is typically used in applications where authorization sign off is required.

For more details, see Graphical Control Properties.

Keyboards-Customized

The **Keyboards** feature enables you to customize soft keyboard interface for entries that cannot be scanned and may need to be localized (translated) into another language. For example, if you added a new keyboard node (object), under the type "FullAlpha," this keyboard would be displayed if you selected "FullAlpha" in the Keyboard property of a Memo control or TextBox control. You can also associate system functions and actions to a character on a keyboard. For example, to use a key to launch a mic (Microphone) feature from the Android system, this can be done by setting a keyboard's (Style) to "Action" and the Action to "Mic". If you wanted to use the AutoCorrect feature for text boxes in an app, you can set the (Style) to Action and the Action to "AutoCorrect" and these features will be picked up from the Android or iOS device.

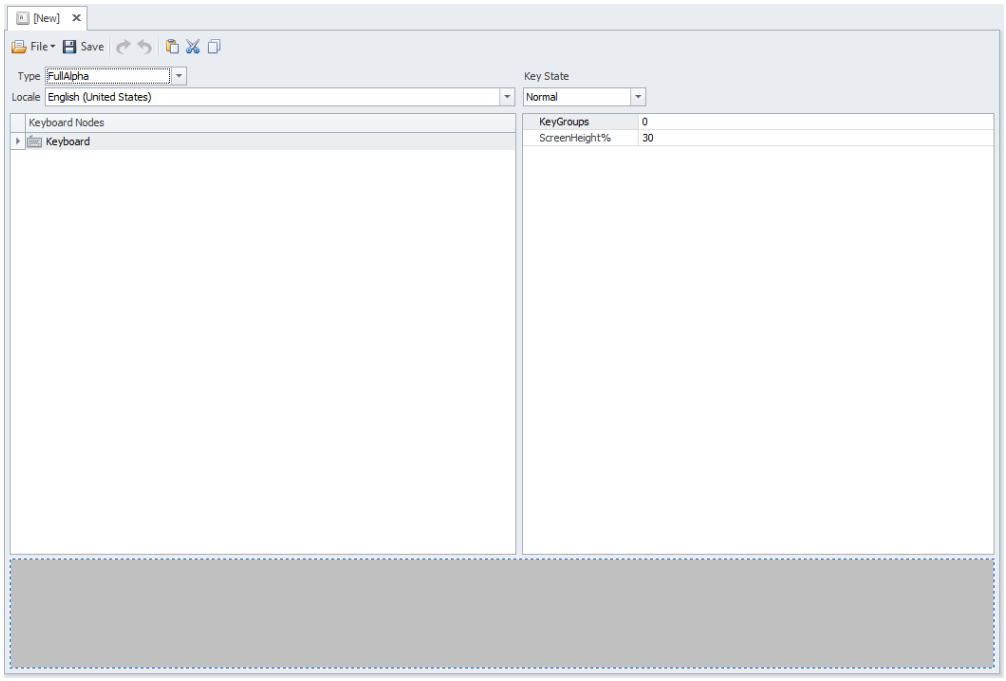
To add and customize a keyboard

Before you start -- decide what type of keyboard you need, and if you need to customize a keyboard, or use a pre-configured file.

To use a preconfigured file, see To import a keyboard file. To create your own keyboard see To Add and Customize a Keyboard.

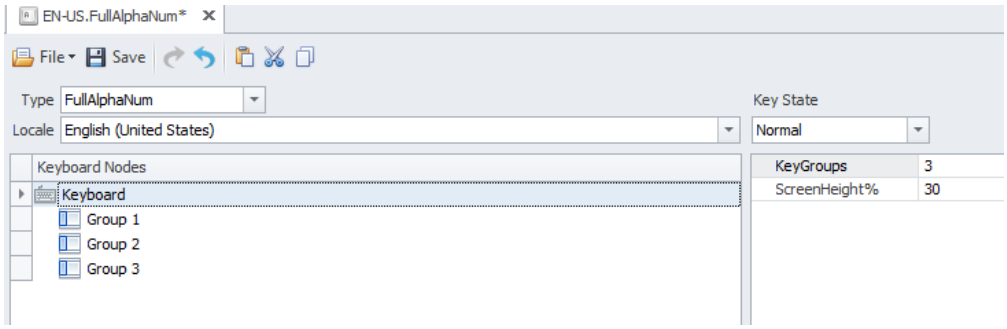
1. In the Solution Explorer, navigate to the Keyboards node.

2. Right-click on Keyboards and select **Add New Keyboard** to add a keyboard, or select an existing one under the Keyboards node. Depending on your selection, a screen similar to the one below will display.



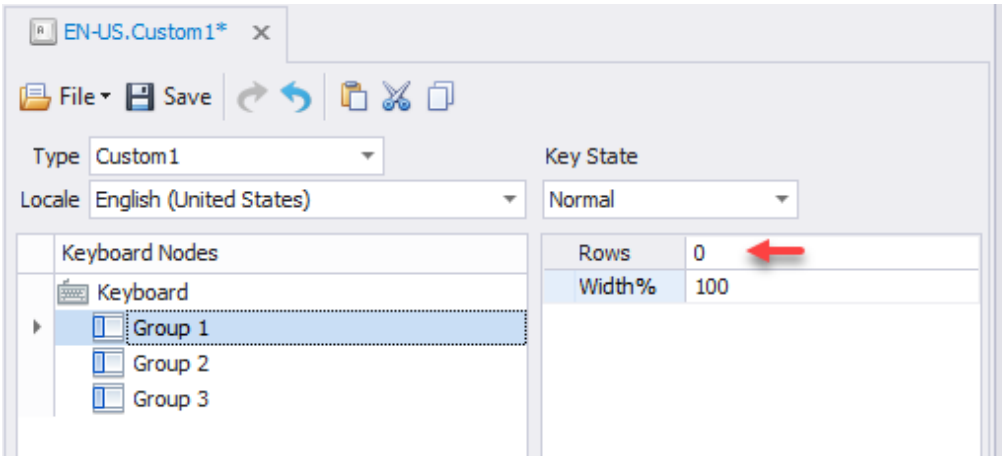
3. Select a Type, Locale, and Key State from the drop down list. Change the default of 30% of your target screen height (ScreenHeight%) if needed.

The Keyboard is organized by **KeyGroups** of common character types. For example, one for function keys, the second for alphabetical keys, and the third for numbers.
4. Enter the number of groups you want for KeyGroups. In the example, we created 3 groups

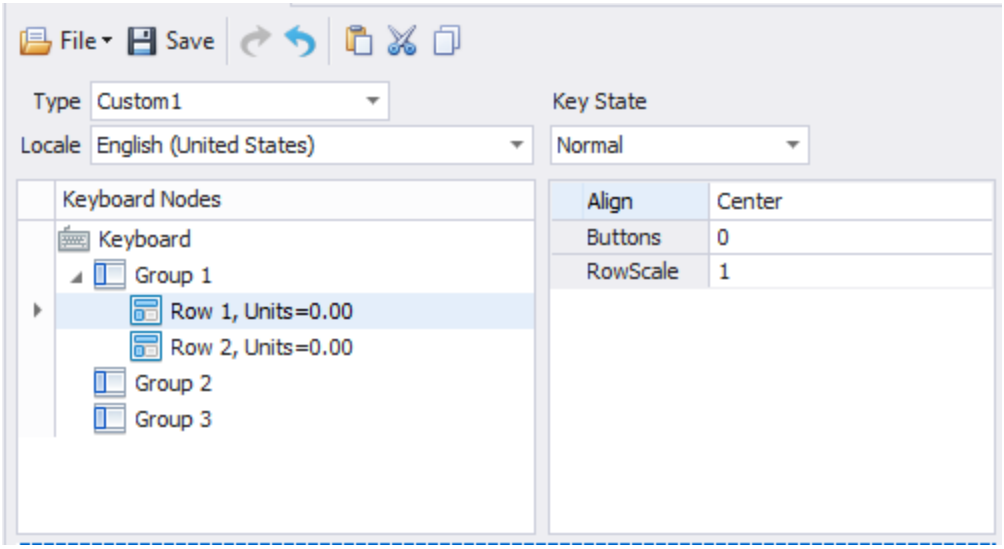


5. A container for each group is added under Keyboards.

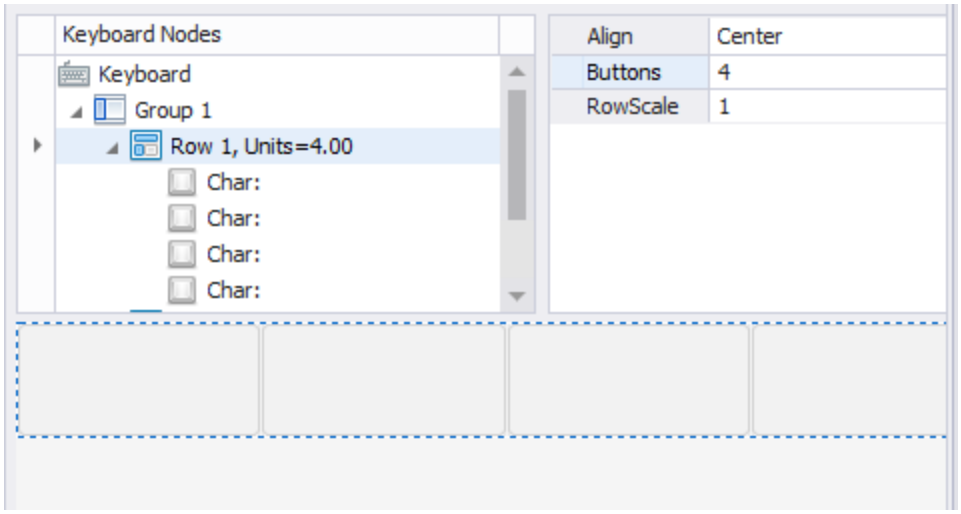
6. In the Keyboard pane, select a group, then enter the number of rows desired for each group. Unless needed, keep the width as 100 percentage for groups width for each row.



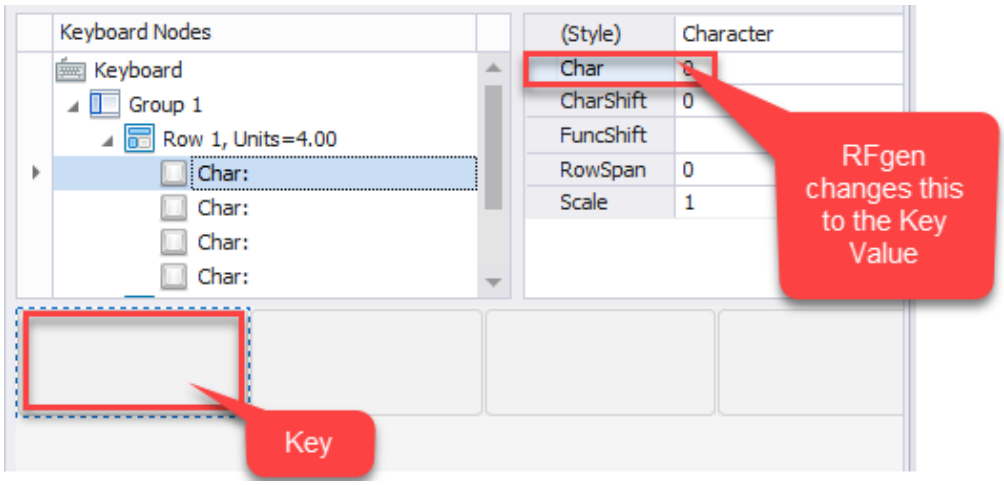
The rows are added under each group. Select a row and enter the number of Buttons. Keep the RowScale as 1 for now.

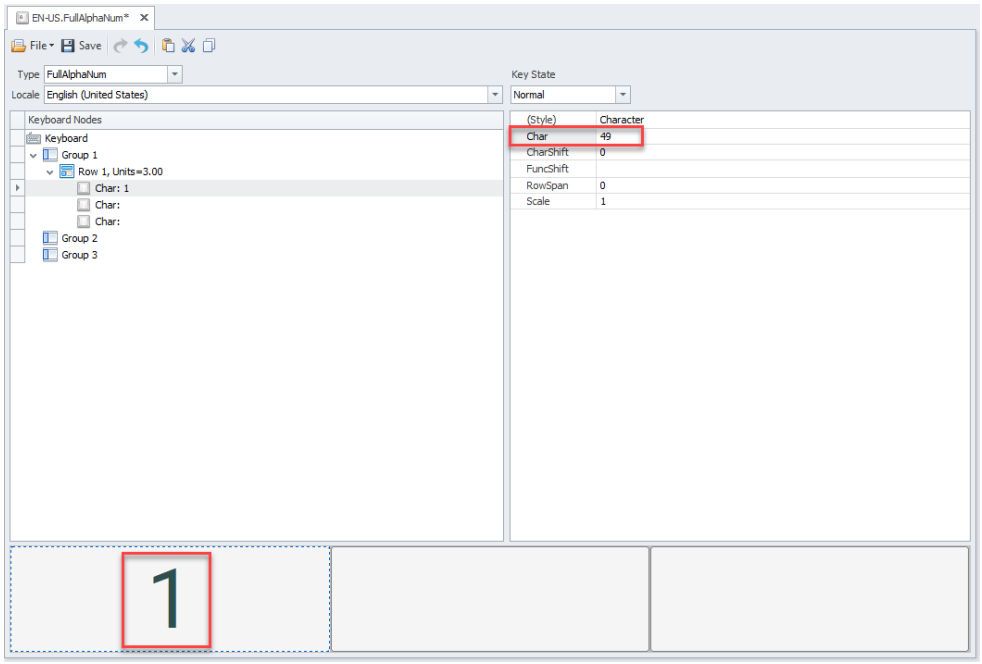


After you enter the number of buttons/row, the row units will equal the number of buttons. A list of characters are added under the row and in the display area, an layout of the row.



7. Select a character under the row, and then enter the values you want display on the key of the keyboard. For example, if you want the first char in the row (far left) to be the number "1", enter "1" for the Char value (upper right). RFgen will automatically change the Character to the Key Code Value for the local of the keyboard. For example if you enter "1" for the Char value, RFgen changes this to key code value 49 and displays a 1 as the key.





Repeat assignment of the values for all characters in each row for each group.

To Stylize a Keyboard

The look and feel of the keyboard is set from Solution Explorer > Themes > [name of the theme] > Element = Keyboard.

Keyboard Designer Features and Options

Bevel – Shapes the outer edges of keyboard.

Caption – Enables you to enter text for buttons that may have a function other than a character or numeric key value. (i.e. Tab, Shift, Delete.) **Caption** displays if **Key State = CapsLock** and **(Style) = Character + Text** or **KeyDown + Text**.

KeyGroups. Groups the sections of the keyboard. For example, the left section is alphabetical keys and the right section, numeric keys.

Keyboard ID – Enter a text or numeric identifier.

Key State – Associates character or function of the key based on one of these three states: Normal, CapsLock, and FuncLock

For blank space buttons, set the (Style) to *BlankSpace*.

If the **Key State** is *Normal*, you can assign a button two values: the lower-case letter (Char) and its upper case (CharShift).

If **Key State** is *FuncLock*, you can only assign one value, the Fkey to the button. Leave *Char* and *CarShift* to 0.

Type – Allows presets settings for all alphabetical characters, alpha-numeric characters, or all numeric characters.

Locale – The Local/language that the characters/labels of each key in the keyboard.

Margin – Sets the spacing between buttons.

RowSpan – Sets the button height based on the number of rows there are above and/or belowthe button.

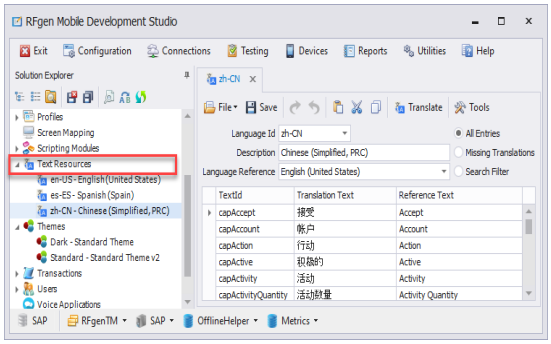
ScreenHeights% – Is a percentage of the screen size.

Units – The number of units a row is divided into. (I.e. number of equally sized buttons in a row.) The most granular number of units will be the base set of units for all rows. This way, a button can be allowed to span 3 or more units.

The profile automatically sets the keyboard theme (if a keyboard is used), but wanted to verify if this is the case.

Text Resources (Language Translations)

The **Solution Explorer >Text Resources** translates and stores the text strings so you can easily create and distribute mobile applications in the language that is best for the end user. This is especially helpful for companies that have plants in different countries but want to use the same application/transactions in their warehouses/plants.



Translation Methods

You can either have a language:

- Translated automatically (Uses a service provided by Google Translate when you click the Translation button).
- Translated manually(i.e. Imports the translated strings from Tools > Import From Excel.)

How this works

Translation requires at least two language files: One file will serve as your source for translations (called the *Reference Language*), and the other file or files will contain your translated text.

Each language file has a table composed of three columns: TextId, Translation Text, and Reference Text.

- The TextId column is a unique id assigned to a text string in the Translation Text column. This links the translated text to a control's property such as Caption, Error Message, NullText, etc.
- The Translation Text column contains the text strings/characters (i.e. Chinese characters, Spanish text etc) in the locale of the language file.
- The Reference Text column contains the text strings that serves as a source for the text that gets translated, and will appear in the Translation Text column of a language file. Any of the language files can be the "base source" for another language as long as the source language file is different from the destination language file. This is how you can have one set of text ids and strings and have multiple languages generated from this one set.

* If you have a database of resource strings and translated text, these items can be imported into the a Dev Studio Text Resources file.

* If you need to save or share your translations, you can export the text Ids and translated text to Excel.

Translate button

The Translate button will translate the text from the language that has Text Id, Translation Text, and Reference Text columns filled in, to another language where the Translation text is not filled in.

Tools Menu

You can also use the **Tools** menu to:

- **Import System Entries** - used to import factory-provided text ids for common actions such as Abort, Backup, Cancel, Main Menu, OK etc.
- **Export From Excel** - used to export language with text ids and translated text into Excel.
- **Import From Excel** - used if you have customized text ids and translated text.
- **Find Active Translations** - identifies unused translations in dark red and list missing translations in blue.
- **Replace Translation Ids** - will replace the value assigned in a control's Caption (via its TextID link) with the value that is assigned to the TextID in Text Resources in all application forms.

Other Actions

Table entries can be grouped, filtered, and searched. Language columns can also be moved, but only single entries may be deleted at a time. (You cannot remove the entire Language column.)

Translate button + All Entries option. This will translate the text from the Reference column into the Translation Text column. The translation will be based on the select set in the Language Id above the table.

Translate button + Missing Translations option. The Translator will only translate text from the Reference Text column if the Reference text's corresponding Translation Text field is blank. If the Translation Text field has text, it won't be translated.

Search Filter option. Use this feature to enter a string and find the entry in the table.

The **File** icon is used to create a new item (text resource), reload a resource, save an existing text resource with a different file name, delete, a text resource, and close a text resource.

The **Reload** and **SaveAs** features will allow you to create a new file from an existing file.

The **Save** button will save the text resource object name using the Language ID that selected from the Language ID menu list.

For example, if you selected "de-DE," "en-US - English," or "fr-FR" the Text Resource objects are named this as well, and the description is also automatically applied for the resource id (for example "fr-FR - French (France)").

Related Topics:

- Create new text resources and translate text, see To create a text resource.
- Import text ids and text from Excel, see To import translated text from Excel.
- Export text ids and text from Excel, see To export translated text from Excel.
- Translate text sourced from another language file, see To source text from a Reference file
- Apply text resource ids globally, see To apply text ids across all applications.
- To learn about the Apply Translation tool for updating or previewing all application translation values/textIds or parse script, see Apply Translations.

To Translate Strings into Multiple Locales

This topic explains how to translate strings from a base language into other languages (i.e. Spanish (Spain), Chinese(Simplified) etc).

Before You Start

* Do you have translated list of text strings you want to use? If so, see To import translated text.

* Are you working with a RFgen application database that already had translated strings, and the application database was upgraded from 5.1 to 5.2? If so, see the topic To apply translated text across all applications.

* Do you simply want to translate a text string through the RFgen Google Translation tool? If yes, then see How Translations Work or continue to Step A.

Overview of how the translator works

RFgen uses Google Translator to perform translations. Translations performed by Google are not context-aware. This means the term Google translated may not be the correct term for how that term is used in context in the country's business culture. For example, in the United States, the term "Plant" is commonly used for warehouse, but in other countries, the literal translation is a green plant that grows in your yard and does not have the contextual meaning of a warehouse or manufacturing facility.

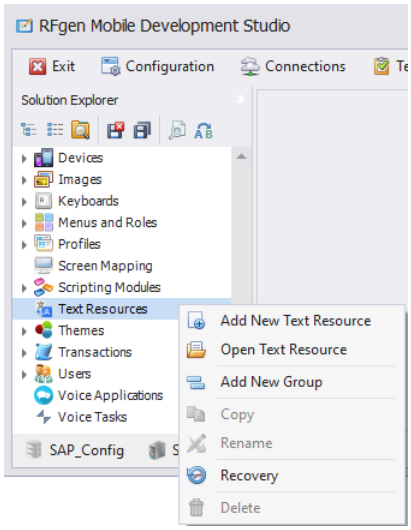
It is recommended that a native speaker review the translations to avoid any translations that are out of context.

For an overview of this feature, see Text Resources - How this Works.

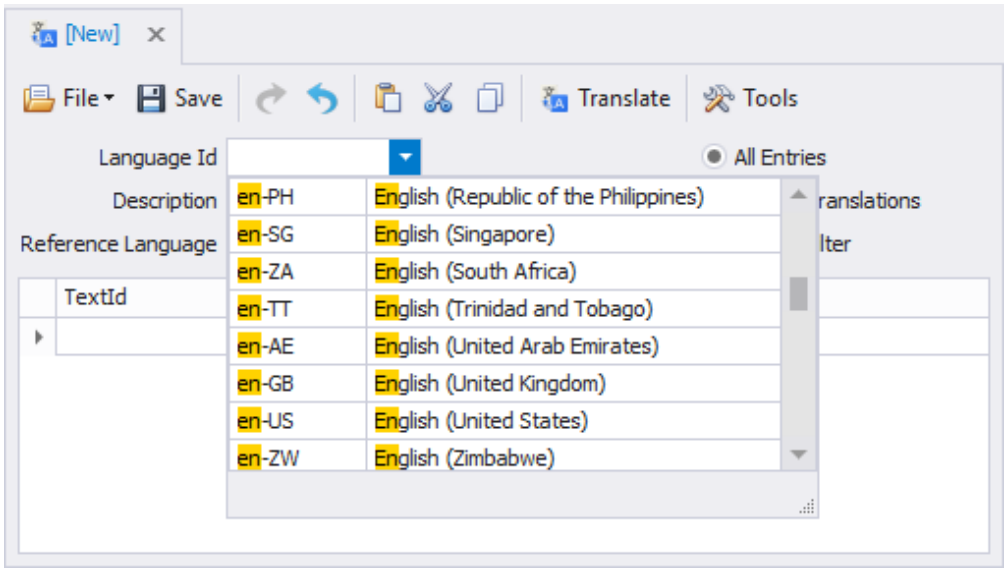
Step A. Add Your Primary Text Resource

1. First create your primary list of text strings (Text Resource table) in your native language. This text resource object will be your primary starting point/base for the strings to be used all the other languages. It will also serve as the source that provides additions and deletions to the other Text Resource objects in the other languages.

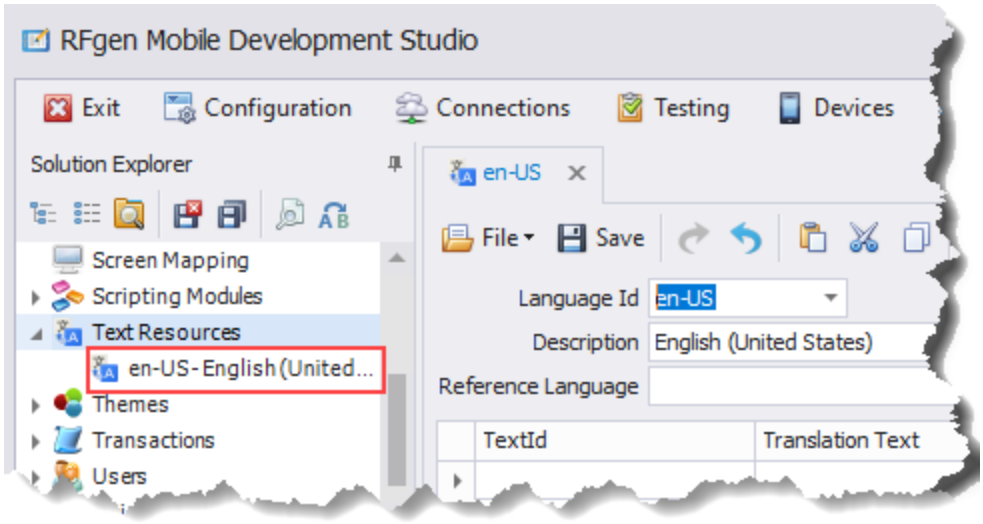
Navigate to **Solution Explorer > Text Resource,** right-click on the **Text Resources** icon and select **Add New Text Resource.**



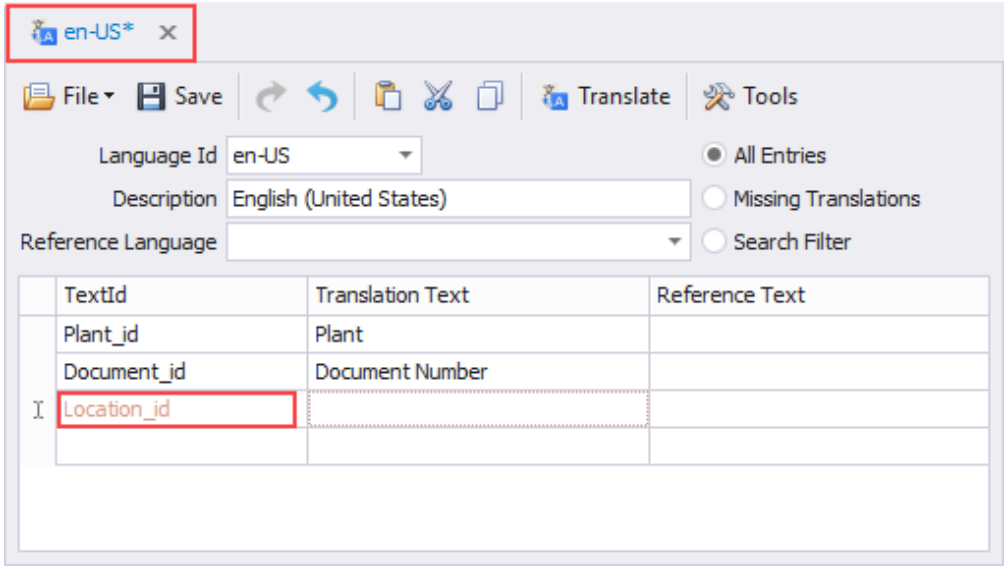
2. In the popup screen [New], select your language from the **Language Id** menu list. Scroll to or enter the first two letters of the language you want while in the drop down menu. In our case, we entered "en-US."



3. **Save** your selection. This adds a new file under the Text Resource folder using the language it identifies with. In this example, the new resource is en-US-English(United States).



In the tab, leave the **Reference Language** blank.



4. Enter your text string identifiers in the **TextId** column.

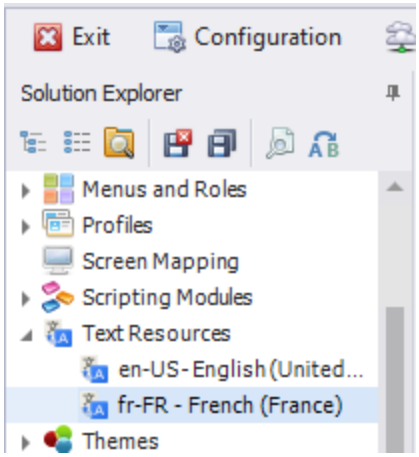
5. Enter the meaning of the id in the **Translation Text** column. The text in the **Translation Text** column is the text that will display in your application at runtime when its linked properly to the element in the application.

For example "Plant_id" and "Plant." The Translation Text can accept any alpha-numeric characters and symbol.

If the text string is colored, (Location_id) this means the entry is incomplete and a value is missing in one of the fields. In this case, the string "Location" was missing from Translation Text. Once the missing text is entered, the color changes back.

Keep the **All Entries** button selected.

6. Click **Save** to save changes.

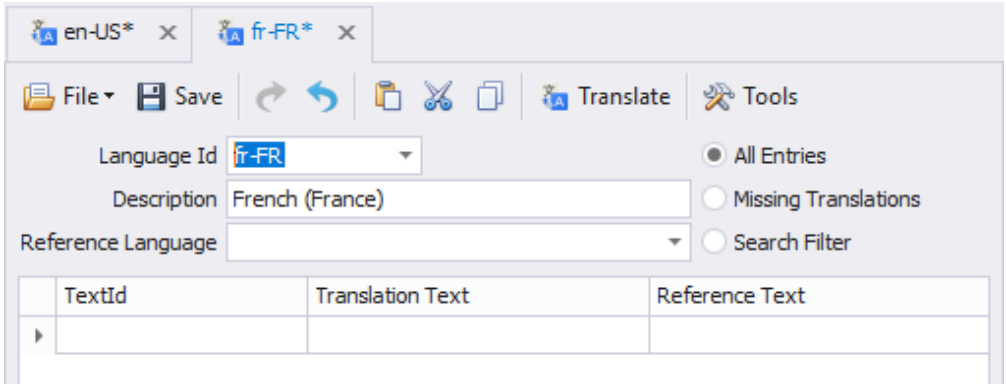


Step B. Add Other Languages and Populate from Reference Language

Once your primary Text Resource is created, you can now create the other tables that will store your translated strings in the locale of desired language. In this example, the new table will be used to store strings in French.

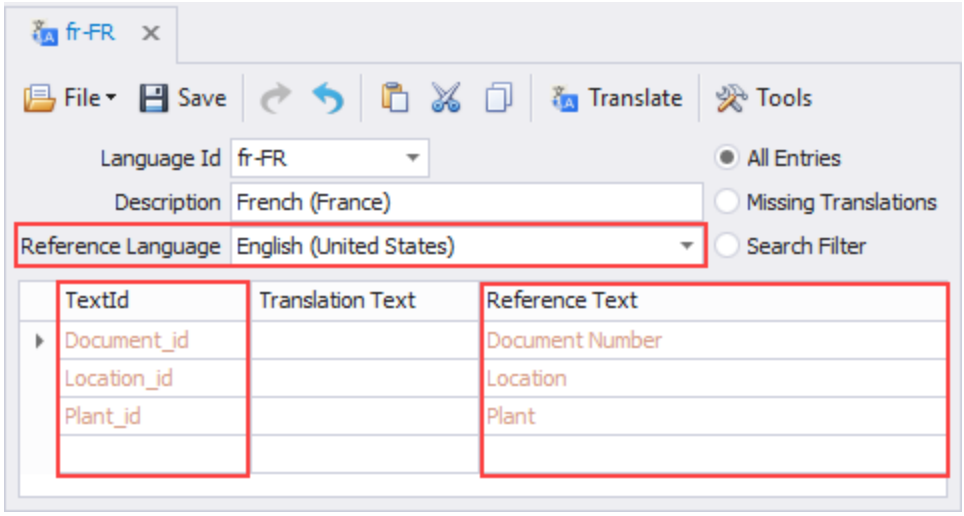
1. Right-click on the **Solution Explorer > Text Resources** icon and select **Add New Text Resource.**

2. In the Language Id menu, select the desired language. For example, enter "fr" to create a French Text Resource table.



3. Populate the fr-FR table's **TextId** and **Reference Text** from Step A above by selecting your primary resource language from the the **Reference Language** menu list.

In this case, we'll select "English (United States)".



4. The fr-FR TextId and Reference columns are filled in using the TextIds and Translation Text from the primary resource (En-US).
The colored text indicates the Translation Text is missing.

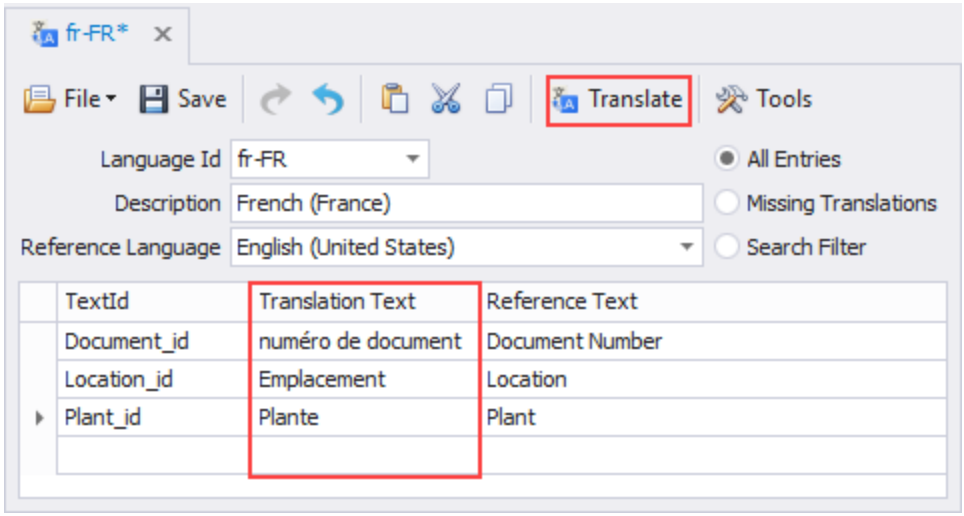
5. Click **Save**. The fr-FR file is added under the Text Resources folder.

Step C. Translate Text

1. To translate the missing values click on the **Translate** icon.

2. Click **Yes**on the pop-up message:
Translate missing entries using Google Translation Services?

3. The Translated Text is now filled in with the text strings in French.



4. Click **Save**. This adds the object under the Text Resources node.

5. Repeat steps B and C to create additional translations.

To Import Translated Text

This topic explains how to import translated strings from an Excel spreadsheet. Use this method to retain text strings that were from other databases and to retain text strings that YOU had translated (as opposed to translations generated by the RFgen translator -- which is based Google technology).

Text strings and assigned resource ids (TextIds) that were imported will NOT be retranslated when the RFgen Translate button is clicked.

Before you start

A. Setup the Excel template to match the table heading in the Text Resources.

- Import only one translated language at a time. For example, if you have an Excel spreadsheet with five translations, RFgen will only import columns A and B (the first two columns).

- Set Excel Column A heading as "TextId" and your text resource ids to be imported here.

- Set Excel Column B heading as "Translation Text" and list your text strings here.

- The remaining columns will be ignored when you import so no need to setup the Reference Text column.

* Store the Excel file in a folder.

B. In the Dev Studio > Text Resources tree, create a language file to hold your import, but do not save it until after you completed the import. For example, if English will be your base or common language you'll use for all the resource assignments, begin with this language first.

Import Process

1. On the tab for the language file just created in step b above, ensure the Language Id is set to the language to be imported.

For example, Language Id = en-US.

2. In the **Tools** menu, select **Import From Excel**.

3. In the pop-up screen select the Excel file to be imported and click **Open**.

4. The table should be filled with the TextId and Translation Text from the Excel file. At this point, you can now choose to setup this language to be the language that will be used for other translations or start assigning the text ids to control captions, headings etc so that they will appear translated.

5. Click **Save**.

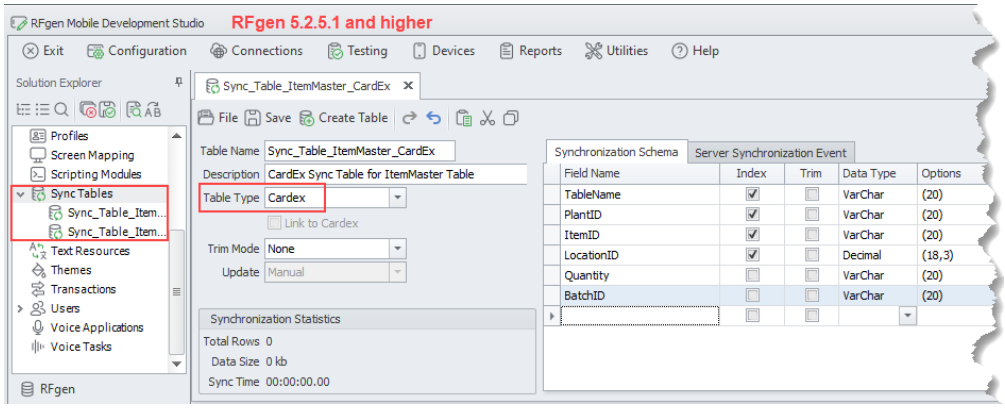
For more details, see Assigning TextIds to Controls, Translating to other languages in Testing, and Translation Overview.

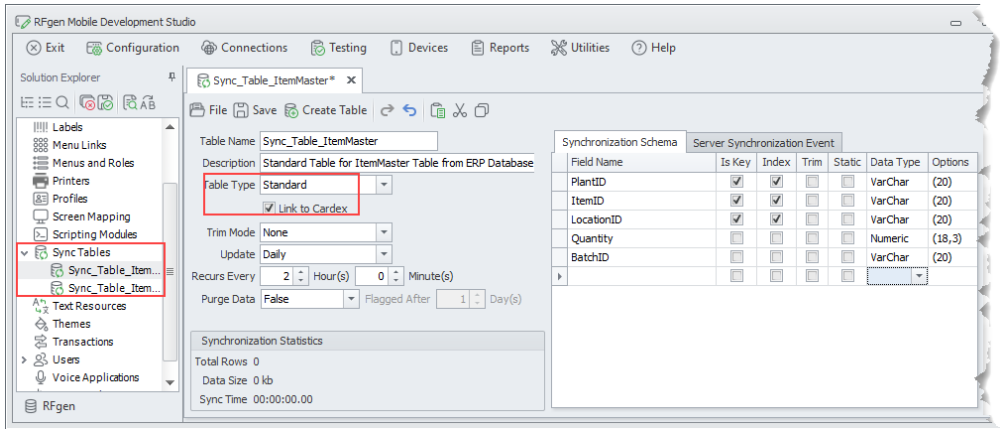
Sync Tables

Sync Tables replaced **Database Exports**. When a user is making inventory changes on a RFgen client while disconnected from the network, there needs to be a way to store and track the changes on the client.

When the client reconnects, the its data changes are submitted to the RFgen server which in turn synchronizes the data with the instance owner of the data (ie. ERP).

When you create SyncTables, you are creating a *Synchronization Schema* which is a listing of the fields that will be looked up during the sync process for a standard tabel. You also create a Cardex table which will reside on the client and on the server. The Cardex (like a registrar) keeps track of the looks ups provided by the standard tables. (So you would typically only have one CardEx table one or more Standard tables.)





Use the **Synchronization Schema** tab to set which fields are the primary keys, and how those fields shall be looked up before the data gets synchronized.

RFgen will run synchronization as a background process, but only if you enable it by adding commands like **Sync.Start**(X number of minutes/internal). The scripting event and parameters for synchronization (i.e. which ERP) are executed from the **SyncTable** > [name of your sync table] > **Server Synchronization Event** tab.

Table Name is the name or identification of the table to be exported.

Description is a required field describing your table.

Table Type sets the type of table that will be created. The options are: *Standard* (default) and *Cardex*.

Link to Cardex can be check only if the Table Type is Standard. This is checked if a CardEx Table Type exists.

Update specifies how often the table on the RFgen server should be update the ERP. The options are: Manual, Monthly, Weekly or Daily. If not using Manual, then you can set the period when updates reoccur and set whether the data is to purged. Note that the updates are one way -- from the RFgen server to the ERP. What gets updated from the ERP data source needs to be specified / scripted in the TextHint tab.

Synchronization Schema tab

Field Names (also referred to as Database Fields) specifies the names of the columns to be included in the table that will installed or created on the RFgen Client (mobile device). The data from the ERP is populated into these fields. Specifying the field(s) reduces the volume of data that will be installed to the client.

- To add a field, click on the drop down list.
- To delete a field, right-cick on the row and tap the Delete button from your keyboard.

Is Key - If checked, indicates that this field is a Primary key. Unchecked means it is not.

Index - If this is checked, RFgen indexes this field. Indexing makes data retrieval more efficient on the client. If unchecked, the field is not indexed.

Trim - If this is checked, RFgen will trim the data in the field according to the value specified under Trim Mode.

Data Type - Is used to specify the data type that will be populated by the ERP.

Options - Sets the number of characters to be contained in the field for the datatype being used. The value should be set in parenthesis. For example, if the datatype was VarChar, you might enter `(50)` in the Options column.

Server Synchronization Event tab

Use this tab to script the details of synchronization. For example, what is the ERP source for synchronization updates? What databases are used?

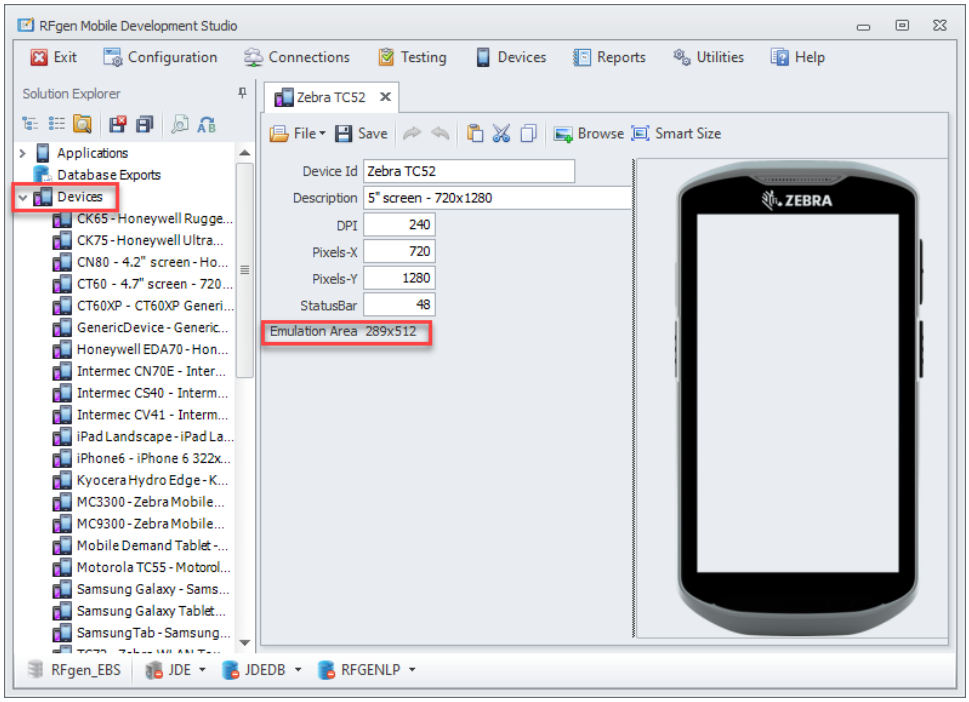
For more details on synchronization language extensions, see Sync Commands in Sync Extensions.

For details on how you control the connections, see Connections > System Queues and Tasks and review the Sync tab.

Devices

The Devices node is used to store images of a device (handheld reader, tablet, printer, etc.) and set the areas where your application will be displayed so yon can visualize how it will look if you are developing applications. Once a device is added here, its image and information is populated to other areas in Dev Studio.

Note: RFgen provides ready-to-use images of popular hand held devices for quick and easier design layout of applications. You can also add your own image by downloading it from another source, then using the Browse button to upload it to the Devices folder.



If the device image is larger than the display area for the device, use the **Smart Size** button to scale the large image so it will fit the display area.

For more details on setting the DPI and dimensions for desktop, handheld, tablet, and phone images, see To Add a Device Skin.

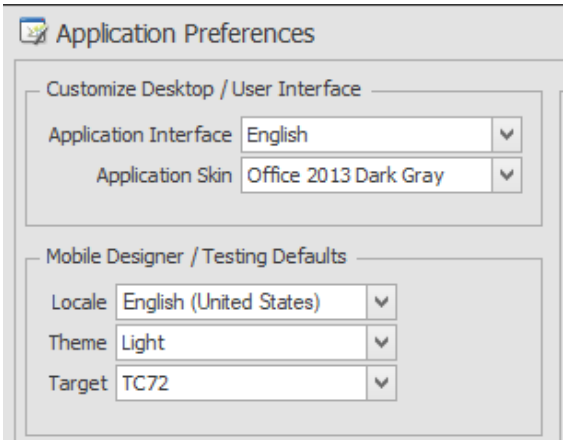
If you are adding a printer:

Interface refers to which language the printer uses to control print operations. ZPL II is the Zebra Programming Language. ESC/Label is used by Epson Label printers. Windows for the interface then this connection type is automatically selected and not editable. Both ZPL II and ESC/Labels can be used to support barcode and RFID labels.

DPI is dots/inch when a label is printed. This is not the DPI of the printer image you are storing.

RFID If checked, displays the RFID Retry Count, and when that model is selected for a printer form it takes that setting as well and is used for that printer queue's setting when printing to the RFid label.

If you have a favorite device image you want used for all your applications and in testing, you can set the image in **Configuration > Application Preferences > Mobile Designer/Testing Defaults, Target**.



To Add a Device

The **Solution Explorer > Devices** node is used to add devices and am image of the device.

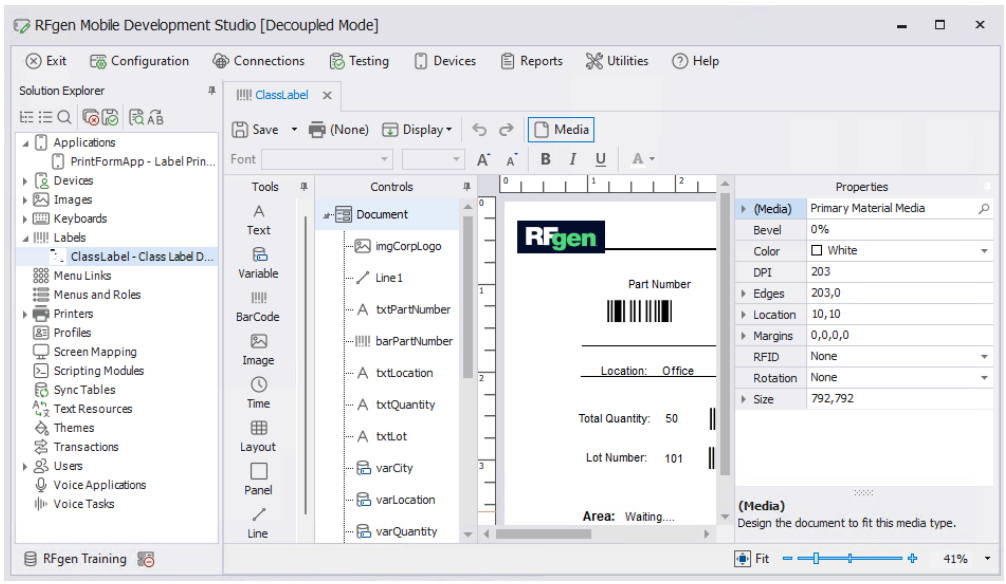
For details, see To Add a Device Skin to serve as a resources for the user interface to ensure the application fits into the device's display area.

If you have **Secure Label** enabled, use Solution Explorer > Devices to add the image of the barcode label printer. Image uploads are optional. You will need to retrieve or create the image from manufacturer before it can be uploaded into Devices.

The data/image entered in Solution Explorer > Devices for the Barcode/Label Printer will also be added to the Solution Explorer > Printers node. To add a printer, see To Add A Printing Device.

Labels

The **Solution Explorer > Labels** node is used to design barcode labels for a selected media loaded on your label printer. You can design different labels for a specific media (i.e. 3 inch by 4 inch label stock media) or for different media sizes.



Each label is saved as a Document with a unique name/ID and description so that you can have various templates for different types of barcode labels and layout formats.

The Tools panel supplies the graphical objects for laying out your barcodes, images, and text (captions). You can also stylize each object through their Property settings.

As you select a specific control, the ribbon menu will display additional property settings that are specific to the control. For example, if you are adding an image, the image icon on the ribbon menu appears so you can upload the image to be displayed. If you added a TextBox, the ribbon menu will display a binding property to enable binding for the control the textbox value is bound to.

Lastly, you can test print your label document so to verify if the output looks and fills in variables as expected.

Labels Tools Panel

Although the Label interface looks similar to the Application designer its purpose is to provide a static template for a printed label. There are no "Tab Numbers" for sequencing of events of dynamic interactions with an end users. The **Document** is the parent container for the graphical objects.

Some controls have a bind property that sets which data source will be used to populate the variable objects.

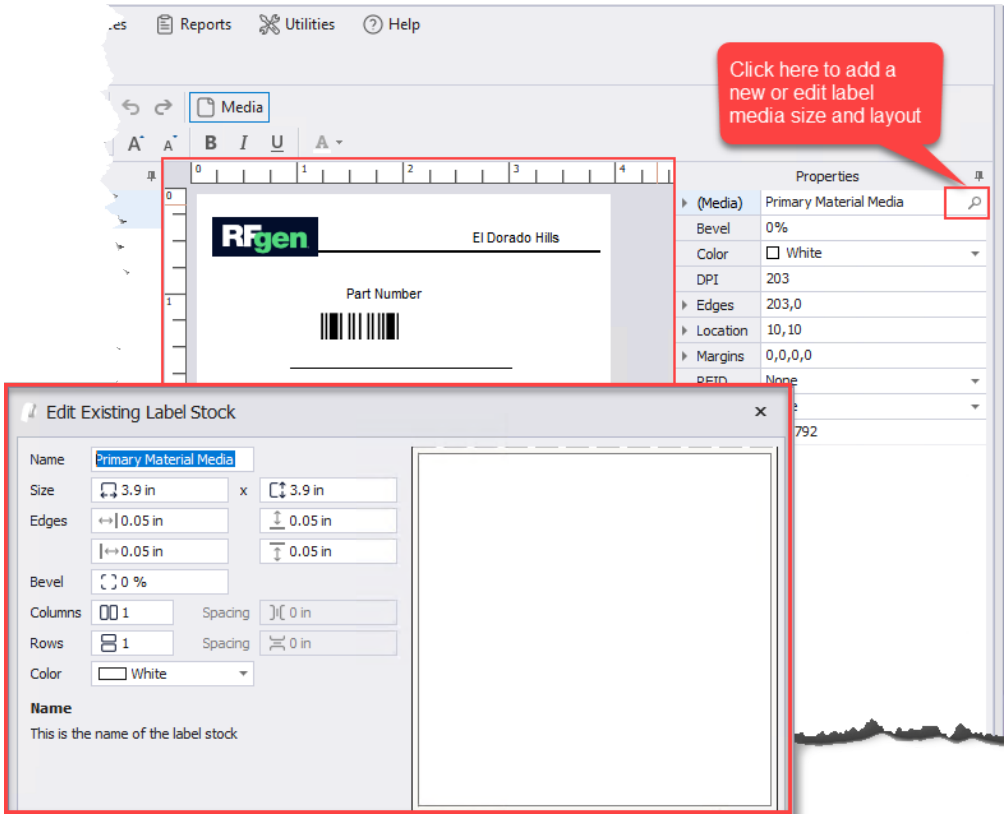
The Tools panel includes the Text, Variable, Barcode, Image, Time, Layout, Panel, and Line controls. Click here for more details about the Image, Time, Layout or Panel control. The description of the control's properties appear at the bottom of the panel.

Ruler Units - The units are set from Environment Settings > Studio Options Ruler Units and supports metric or standard (Inches).

Properties panel - The right-most panel displays the properties for a selected control. Note that a description of the property displays at the bottom of the panel. A few important properties are listed below.

Document control: Is a parent object.

The **Media** property lists the dimensions of the physical Label Media that was added to the Solution Explorer > Label Stock node.



Text control: Is a child object used to contain a caption. If desired, the **BindInput** and **BindTo** properties can be used to link the caption text to another control's input. (For example, the caption can be sourced from a BarCode control.)

Variable: Is the container for each datafield. (I.e. If printing a shipping label, you can have a Var-Name, Var-Address to contain your customer's name and shipping address.)

Barcode: DataMAX supports just about all barcode formats-- One dimensional (1D), Two dimensional (2D), Quick Response (QR codes), Portable Data File (PDF), Datamatrix (used as it has a smaller footprint and is easy to read). Note that we do not provide a description for each type here as these can be easily looked up on the Internet and our list will continue to grow over time.

Image: Can be used to add a company's logo, or other icons or background image.

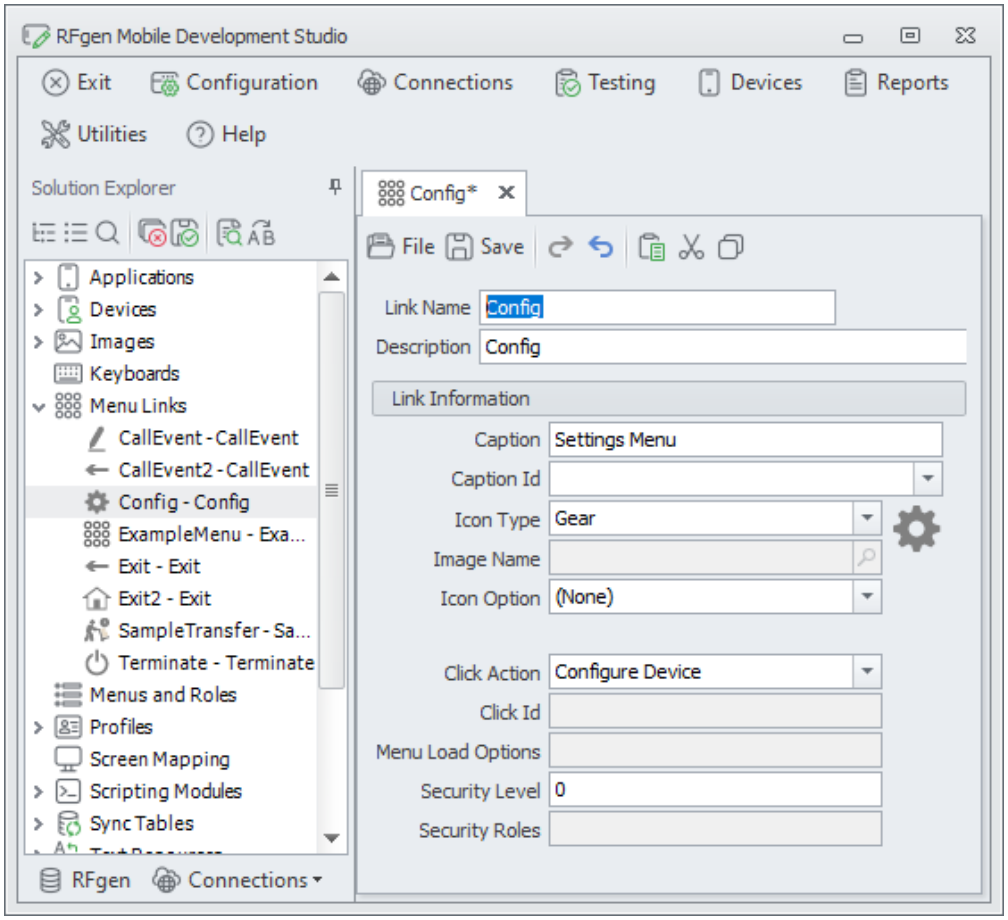
Time: Can be used to print the time the label was printed. Supports most custom data and time format strings. For example MMMM dd, yyyy hh:mm TT gives you September 03, 2025 05 05:42pm. The local of the time is based on the time setting of the Dev Studio server.

Layout: This is a parent control that adjusts the widths of columns and heights of rows using the space available inside the control to calculate the alignment of objects. RFgen allocates the available space using these factors:

- The number of columns and rows present
- The contents of each cell
- The property (or methods) assigned to each column or row

Panel: The Panel Object is a parent control that enables you to easily perform group actions such as deleting, hiding or moving all child controls at once by deleting, hiding or moving the parent Panel control. It also makes it easier to apply common grouping values such as the background.

Menu Links



The **Menu Link** objects created here will display in the **Solution Explorer** > **Menus and Roles** > [*name of the Menu item*] > **Menu Items**, **Link Names** drop down list. Or, if you click on the **Add Items** icons, your list will display in the in the **Select MenuLinks** screen.

Note: In RFgen version 5.2, Menus Links was named "Icons."

Each link is associated with an icon and the icon is associated with an action. The following describe the settings for Menu Links.

Link Name is the name of your object. For example, the name used for the link to your menu or application. **Description** is your description of the object name. Both fields are required.

Link Information contains the settings unique to your Menu Link object.

Caption is the heading that displays on your menu.

Caption ID is used for localization of the Caption if you have the Text Resource setup for it.

Icon Type is the icon you want to representation your menu (or submenu item). You can select the icon for this link from one of the RFgen-factory-provided icons.
If you don't like the RFgen-factory-provided icons, select **Custom**. This enables the **Image Name** drop down list. You can then select an icon you created in the Solution Explorer > Images folder.

Icon Option lets you set the orientation of the icon.

Icon Groups is used to select the theme color for the icons in a menu.

Display Mode changes the menu list style so you can visualize the menu items in List mode or in the Default mode defined under themes. (Default formats are listed the Menu > ListStyle in Themes.)

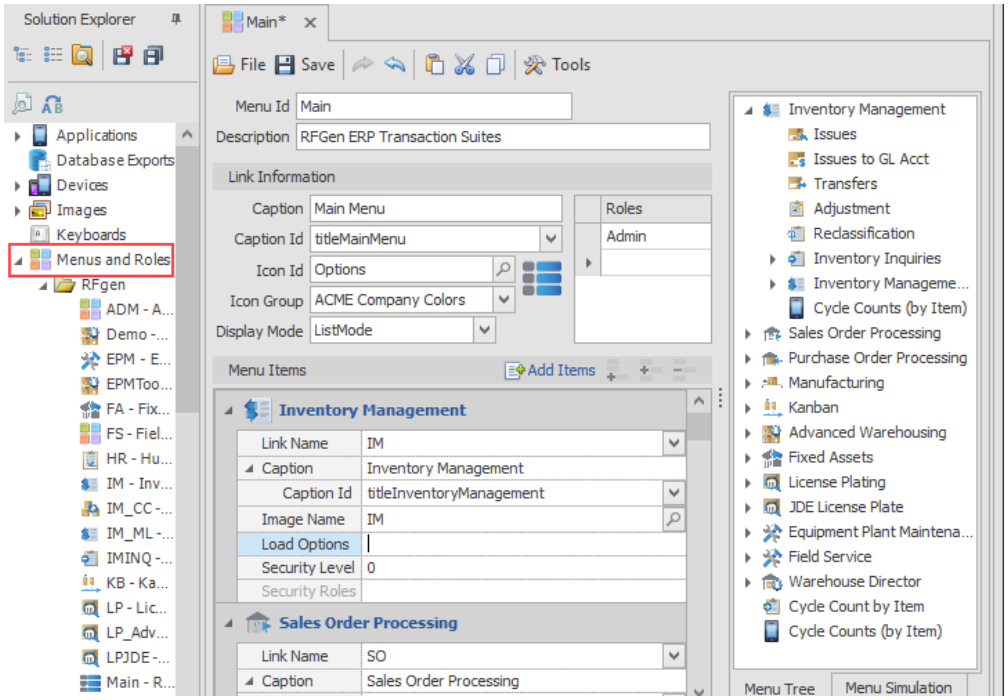
Click Action selects the action that will occur when the end-user taps the icon. RFgen by default provides these list of actions: Advance, Backup, Call Event, Call Form, Call Menu, Clear Input, Configure Device, Exit, Scan, Search, Show/Hide SideBar, Show/Hide SIP, ShutDown, Sign Out (also known as Sign Off or LogOff), and Submit.
Note: Some actions may depend on the functions available in the mobile device.

Menu Load Options is an optional entry for executing scripts or default values. For example, if the application has a value defined as "MoveType", the values related to "MoveType" will be loaded with the application when the user selects that application from the menu.

Security Level works off a range of values and must be coordinated with the Security Level setting for the User. the security levels are customizable and works as follows. If you setup security levels 0 to 3 for all menus with X, Y, Z apps, and assign Sam the user a security level 0 to 3, Sam will have access to the menus with apps X, Y, Z. But if you assigned Mary the user security level of 4 or higher, she will not have access to any menus with security levels 3 or lower.

Security Roles is an Optional field and is read only. If a security role (or roles) was setup in an application, then the Security Role value will display in this field. For more details, see the topic To Limit User Access to Menus.

Menus and Roles



This screen is used to create a menu and assign applications to the menu. The Menu is then assigned to users in the Users screen so the use can accessed the desired application.

Link Information is a container for linking from a source or acting as a source for other functions.

Caption is the heading for the menu, and *Caption ID* is used for localization of the Caption if you have the Text Resource setup for it.

Icon Id can be left blank or used to assign an image that will display with the caption text.

Icon Groups is used to select the theme color for the icons in a menu.

Display Mode changes the menu list style so you can visualize the menu items in List mode or in the Default mode defined under themes. (Default formats are listed the Menu > ListStyle in Themes.)

Use the **Add Items** (top of menu bar) provides utilities to perform global searches, replacement and import/export tasks. For more details, see Menu Tools.

Menu Items

Link Name is the name of the application menu name.

Caption is the name that will show on the menu. **Caption ID** is the text resource id if you want the caption to be localized.

Image Name allows you to source the icon for your menu or application from the Images group in the Mobile Development Studio > Images.

Load Options is an optional entry for executing scripts or default values. For example, if the application has a value defined as "MoveType", the values related to "MoveType" will be loaded with the application when the user selects that application from the menu.

Security Level works off a range of values and must be coordinated with the Security Level setting for the User. the security levels are customizable and works as follows. If you setup security levels 0 to 3 for all menus with X, Y, Z apps, and assign Sam the user a security level 0 to 3, Sam will have access to the menus with apps X, Y, Z. But if you assigned Mary the user security level of 4 or higher, she will not have access to any menus with security levels 3 or lower.

Security Roles is an Optional field and is read only. If a security role (or roles) was setup in an application, then the Security Role value will display in this field. For more details, see the topic To Limit User Access to Menus.

The **Menu Tree** displays all your menus, submenus and applications linked to each menu.

The **Menu Simulation** view shows how your menus would look on screen or inside a device. For more details, see Menu Simulation.

Assign menus to user or user groups

Once the menus, submenus and applications are setup, you can assign the menus to user accounts in the **Solution Explorer > Users** tree. This helps control which Mobile Apps a user will can access on their mobile device or Windows desktop system.

For more details, see User Overview.

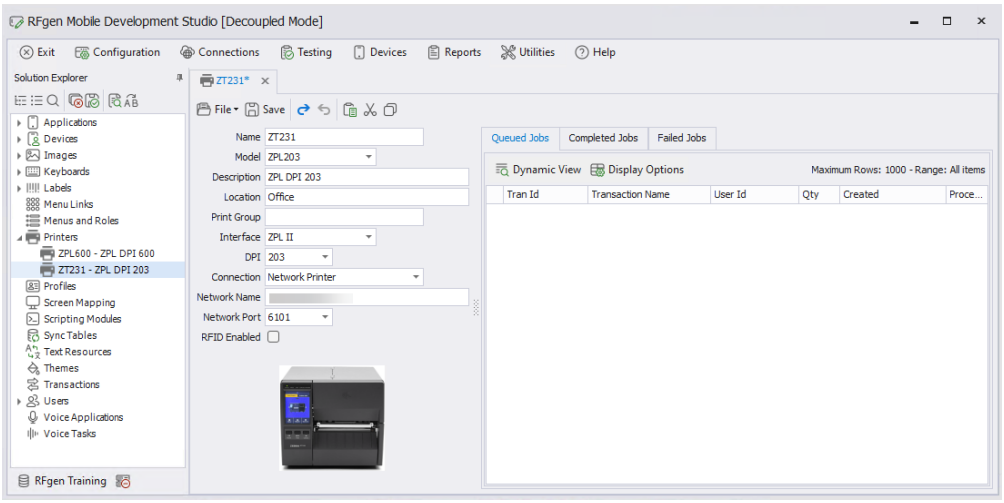
When testing your solution, the process generally begins with the Login app, then the menu and then the application selected in testing.

If you plan on testing using this path, a default user (i.e. Sam), menu and app (i.e. Login), and menu of other apps should be setup before you begin testing.

NOTE: The *User Management Console* also provides features that allows Console users to setup Menus and Users without providing Console users full access to everything on the server. This enables database updates from the server or the console by different users. For more details, refer to the *User Management Console* section in this guide.

Printers

The **Solution Explorer > Printers** nodes is used to configure which physical label printer that you want your barcode label to be print to. If you added images of your printer to **Solution Explorer > Devices** and selected the **Interface** type, **DPI**, and **RFID Enabled** in Devices, it will also populate the Printers screen if you select the same model in Devices.



To add a new printer, right click on the printer node and select Add New Printer.

Name: Enter the name or identifier for your printer. Spaces and special characters like dashes and underscores are not allowed. This field is required.

Model: You can either enter a model # or select one from the list.

Description: Enter the description of your printer. This field is required.

Location - If desired describe the physical location of the printer. This entry is optional.

Print Group – If desired describe the group your printer belongs to. This entry is optional.

Interface - This refers to the language the printer uses. Most use the Zebra Programming Language **ZPLII**. **ESC/Label** is for Epson Label printers. **IPL** is for handheld Intermec devices that use the Intermec Programming Language. **Windows** is used to process print requests through Windows and if you had the printer's driver installed on a Windows system. If can be used with some printers but is limited to the languages supported by the installed driver.

DPI: The value here is dependent on the output DPI of the selected printer. TIP: Confirm the DPI setting for the printer based on the manufacturer's printer's manual.

Connection Properties The printer connection types are: Bluetooth, Driver, Network, and Weblink.

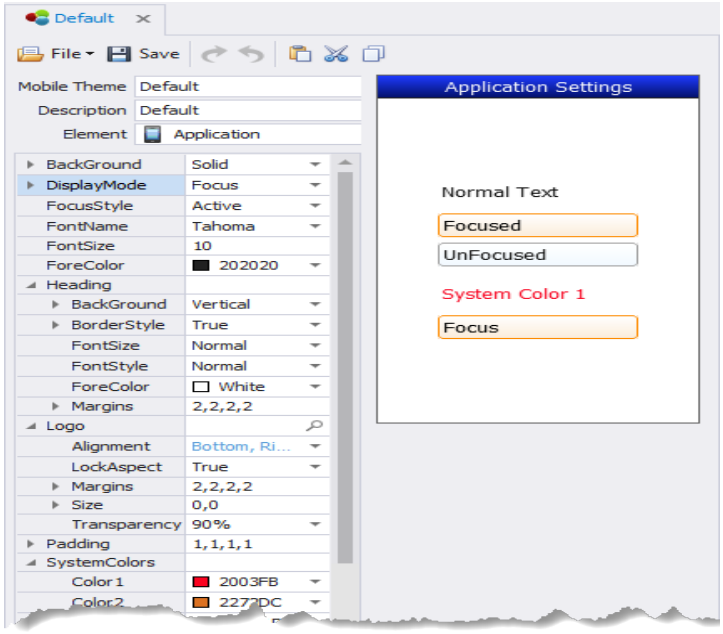
- Weblink and Bluetooth look up the name of the Device entered and send over the print job over that device.
- Weblink - For Zebra's Weblink connections between the print server and any printers connecting using that instead of TCP which the usual Network setting for the printer uses.
- Bluetooth - For mobile devices that have a paired bluetooth connection with the label printer. Print requests from the app uses the Device.SetBluetooth and Device.SendBluetooth extensions.
- Driver Printer - Relies on the functions supported from the driver to make connections and communicate with the printer. If you had set "Windows" as you interface, then the Driver Printer is automatically selected. Is also used for USB connections to the printer.
- Network Printer - Uses the TCP protocol. If you select Network Printer, you also need to set the Network Name/IP Address and Network port of the printer.

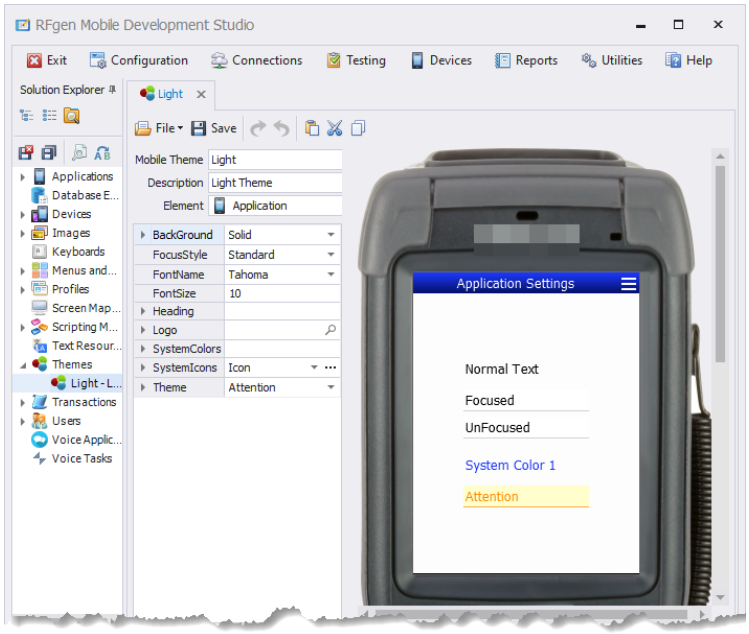
Note: Work with your IT admin or network provider to ensure that the IP address of the printer is accessible to the server that the Secure Label service is running from.

RFID Enabled. Check if you are printing RFID labels. This allows you to set the number of times the Secure Label printer will retry the printout of a label. For example, when a printer model is selected for a printer form, it takes that setting as well and is used for that printer queue's setting when printing to the rfid label

Once your printer configurations are saved, a new printer form is added to the Solution Explorer. If you also have a TM Dashboard connection, this will create a printer queue entry to the Transaction Management Database.

Themes Overview





The **Theme** (also called "Mobile Themes") is a collection of control property values and event/focus settings that are used to create a common color theme for an application and common look and feel when a user interacts with the application. For example, you can create a theme that uses your company's colors and company's logo, and highlights a selection if bright yellow. Or, you can have another theme that uses colors which work best in low light conditions and also provides a WARNING state color themes.

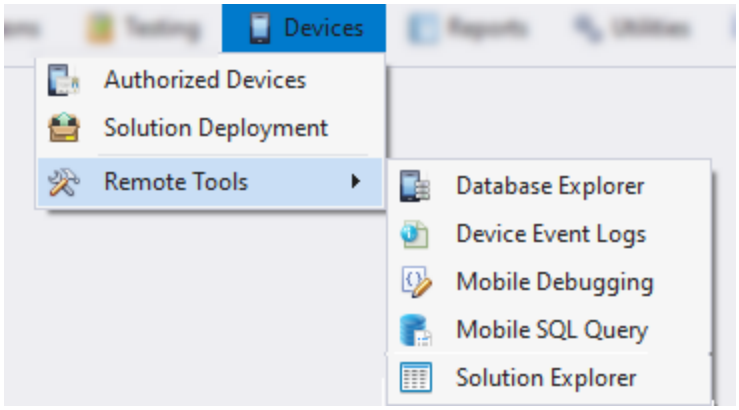
Themes are structured by **Element** (Application, Button Control, Label Control etc) and serves as the parent for the corresponding element in the **Solution Explorer > Applications** designer. For example, if you created a theme called "Light Grey" and set your Button Control's BackGround property to light gray, then all buttons added to a form will automatically have a gray background.

- To add, remove, rename, copy or recover a theme, use the Mobile Themes Right-Click Menu.
- To view a theme element in a device, see To Enable Device Emulation.

Related Topics:

- For Application property definitions see Theme Applications.
- For property definitions on a graphical control, see Graphical Control Properties.
- For element descriptions and details on element properties, see Theme Elements.

Devices Menu Overview



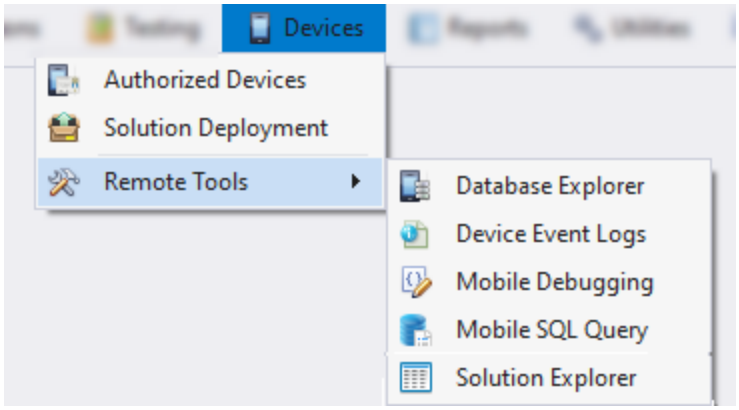
Authorized Devices - tracks authorized devices and enables administrators to: a) manually authorize mobile clients, and; b) manually authorize Thin Clients (if the server is configured to Restrict Online Access), or c) view automatically authorized Thin Clients. This feature is used for authorizing all types of client connections. For more information on authorization of devices, and setting restrictions see Device Authorizations.

Solution Deployment - used to send updated or additional components to Windows CE/Mobile devices that have the RFgen CNC (Client Network Control) software installed on the remote client. It also provides various ways to transfer CAB or CNC files to Windows CE\Windows Mobile devices. The server can then update mobile applications, menus, users etc. as part of as needed for data collection. If the server is upgraded, and new client files need to be distributed, this function can help perform this task. Full profiles can also be sent to the device. This feature is not used on Android or iOS devices.

Remote Tools - Contains additional tools you can use for troubleshooting or managing items remotely. These are designed to work with Windows devices and include the following options:

- **Database Explorer** - shows the data on Windows CE\Windows Mobile device's database.
- **Device Event Logs** - are used to request the Application Errors or Performance Data log files from a remote device (Windows CE or Windows Desktop mobile client).
- **Mobile Debugging** - opens a remote session with Windows CE devices for debugging purposes.
- **SQL Query Window** - is used to inquire or update data stored in the mobile device's database. Thin client solutions do not have users, menu, applications, etc. and therefore do not contain databases.
- **Solution Explorer** - is used to connect with a remote device and upload solution files to the remote Windows Desktop Client device and Windows CE/Mobile.

Devices Menu Overview



Authorized Devices - tracks authorized devices and enables administrators to: a) manually authorize mobile clients, and; b) manually authorize Thin Clients (if the server is configured to Restrict Online Access), or c) view automatically authorized Thin Clients. This feature is used for authorizing all types of client connections. For more information on authorization of devices, and setting restrictions see Device Authorizations.

Solution Deployment - used to send updated or additional components to Windows CE/Mobile devices that have the RFgen CNC (Client Network Control) software installed on the remote client. It also provides various ways to transfer CAB or CNC files to Windows CE\Windows Mobile devices. The server can then update mobile applications, menus, users etc. as part of as needed for data collection. If the server is upgraded, and new client files need to be distributed, this function can help perform this task. Full profiles can also be sent to the device. This feature is not used on Android or iOS devices.

Remote Tools - Contains additional tools you can use for troubleshooting or managing items remotely. These are designed to work with Windows devices and include the following options:

- **Database Explorer** - shows the data on Windows CE\Windows Mobile device's database.
- **Device Event Logs** - are used to request the Application Errors or Performance Data log files from a remote device (Windows CE or Windows Desktop mobile client).
- **Mobile Debugging** - opens a remote session with Windows CE devices for debugging purposes.
- **SQL Query Window** - is used to inquire or update data stored in the mobile device's database. Thin client solutions do not have users, menu, applications, etc. and therefore do not contain databases.
- **Solution Explorer** - is used to connect with a remote device and upload solution files to the remote Windows Desktop Client device and Windows CE/Mobile.

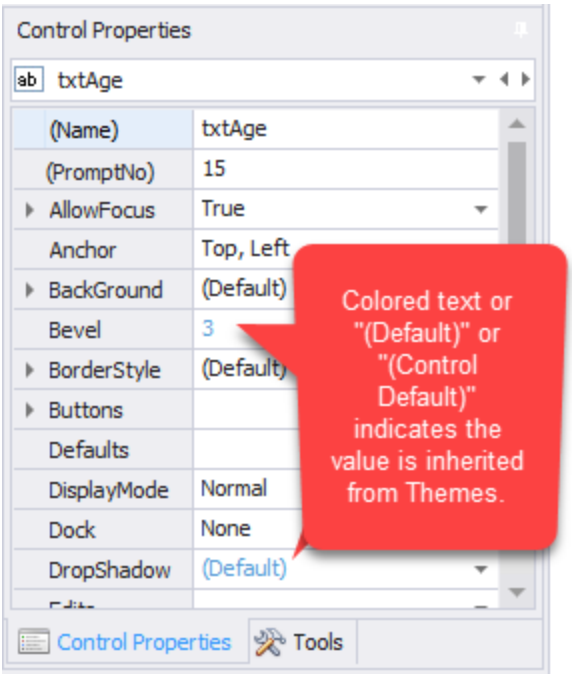
Graphical Control Properties

The **(Name)** property is a standard property for all controls except the Form.

The **(PromptNo)** property is the Prompt sequence value RFgen assigns to objects placed on a form or a page. If you move the location of a control or page, the PromptNo also updates, or you can edited it directly in the Application Designer Control Panel.

If your script uses App.PromptNo VBA extension, it should return the PromptNo value for the specified control.

HOW TO TELL IF YOUR THEMES ARE BEING APPLIED



If the value for a property appears in colored text or the value shows (Default) or Control Default this means the property value is using the corresponding value set in Themes.

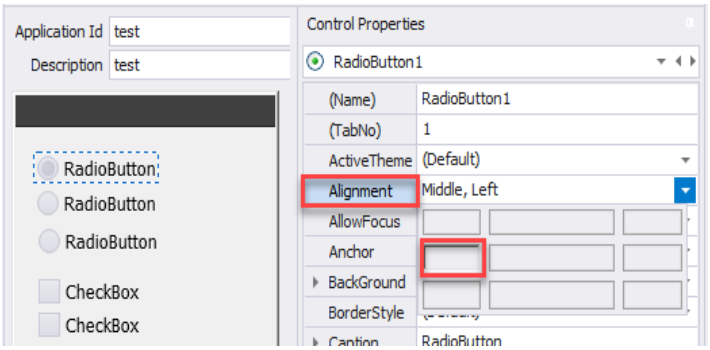
Control Property Descriptions

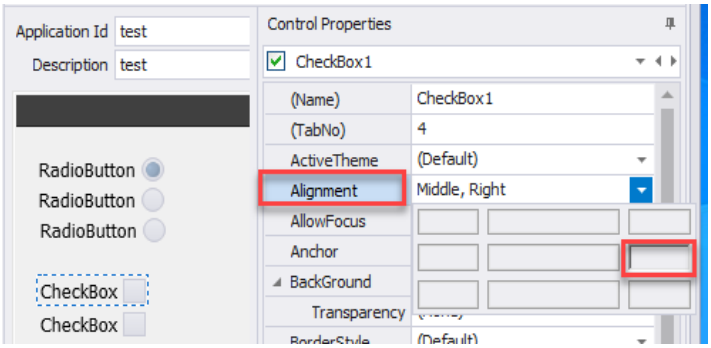
Descriptions for all the control properties are listed alphabetically on this page. Tap **Ctrl + F** keys to **Search** for a property definition on this page.

Not all properties appear for all control types. For example Image controls, CheckBox, RadioButton, Page controls and others will have their own unique properties.

The **Action** property under the **Button** property provides built-in operations for buttons in controls like the TextBox. For example, TextBox1> Buttons> Action = Search, and the TextBox1 > Button > Image = Search icon, and Buttons > Visible = Ture will add the OnSearch event in your Script View. You can also set TextBox1 > Buttons > Action = Event, Buttons > Visible = True, and set TextBox1 > Events > [select an Event like OnScan=True, and this will add the OnScan Procedure to your script.

The **Alignment** property places text and images relative to another object within a control. For the CheckBox and Radio Button controls, the images are shifted relative to the labels when you select a position in the Alignment drop down menu. (See example below)





For the Button control, Alignment positions the Caption text and the image within the button space. To separate the image from the Caption text, use the Image > Alignment to position the image relative to the Caption text. For the Image control, the image is positioned relative the edges of the image box.

The **AllowFocus** if set to True, enables the Page object or a prompt to receive focus and then select the actions that the user can take once the object receives focus. "False" prevents the control from receiving focus, removes a TabNo assignment or prevents one from being assigned. You can enable these options when the object is able to receive focus:

- *AllowBackup*- to the previous control;
- *AutoSelectText* - selects all the text in the edit where the prompt lands if set to true; places cursor at the end of edit if false;
- *EraseOnBackup* - will erase the text if user backs up;
- *FocusOnClick* - This;
- *OnEnter - Advance* - dropdown option for: **Exit** the form, **Hold** the focus, **Reset** the form (i.e. Clear entries), or **Submit** when the user taps the Enter key.

The **AllowRoll** property is used on controls that contain long lists of items and helps you "jump" to items faster. For example if you press the up arrow while on the first line in a long list of items, the last item from the list will display. Or, if you are at the bottom of a list, and press a down arrow, you are taken to the top of the list.

The **AlternateItem** property group sets the background color, font color (ForeColor), visibility of the colors for alternate rows in a list (ComboBox, DataGrid, ListBox, PanelList etc.). To display the alternate color, set Visible to True.

The **Anchor** property sets a child object's position and dimensions relative to where its anchored to the parent. For example if a TextBox control was anchored to a Form on all four sides, the TextBox would be

proportionally resized if the Form was displayed on a different mobile device. But if the TextBox was Anchor equaled "Bottom", "Right", then its size won't change, and its position will stay relatively the same. If no other anchors except the Bottom is set, then RFgen automatically centers the child object inside the parent. If the parent is resized, the chile object's size is not changed even if the parent's size changes.

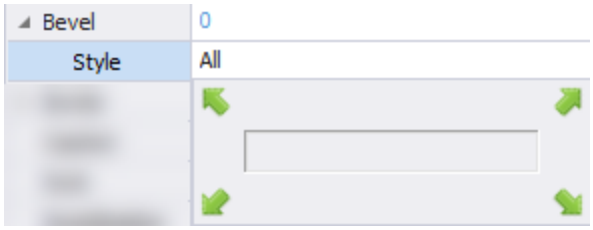
The **AutoSize** property will stretch the object's background to the lowest and right-most portion of the screen, depending on the option selected. This property is available for the Button, DateTime, Frame, Label, Layout, and Panel controls. The values are: (None), Content, Height, and Width. If AutoSize is set to "(None)" then you can drag-n-resize the control. But if the control is in AutoSize mode, the ability to drag-n-resize the control is disabled and the control is sized by its content, width or height. For example, if you have a Panel control that was empty (no controls inside it), and AutoSize was set to "Contents," "Height," or "Width," then the Panel will automatically resize to 0 pixels. If there was an image in the Panel, and AutoSize was set to "Contents" then it would resize the Panel around the image relative using the right-most and bottom portion of the panel as the start location. See the Size property for the size in pixels. See Manage Paged Collection for details on AutoSize Property for Pages.

The **BackColor and BackColorAlt** (previously called **BackColor(1)** and **BackColor(2)**) properties are used to create either solid backgrounds or gradients depending on the option chosen in the Background Fill property. The color can be set from using Custom Color tab or by a 6-character hex value (which gives you 16 million colors to choose from).

The **BackGradient** This property was replaced by the **BackGround** property.

The **BackGround** property has three categories for setting the background color of a control. If set to Solid, RFgen applies the value from Color 1. If a directional values such as Diagonal Right, Diagonal Left, Vertical etc. is selected, RFgen applies the values from **Color 1** and **Color 2** to create a gradient background. If Transparent is selected, the next-closest background color behind the control will be used. For example, if your Mobile Theme Application BackGround = Red, and your Mobile Theme Label BackGround=Transparent, and your Label Caption color = White, then your Label text will appear as white on a red background in Mobile Themes. In the application designer, the Label text will also appear as white text on a red background if the application's Form Active Theme = (Default) and the Label Active Theme = (Default). But, if your Mobile Theme Label BackGround = Solid and the colors 1 and 2 = Pink, then in the solution designer, the label's background will appear as white text on pink because uses the Label's solid background color blocks the background color from the Application Form.

The **Bevel** (graphical mode only) property sets the curvature of a square's and rectangle's corner edge where 0 is no bevel and 100 turns the object into a circle or oval shape. The **Bevel Corners**



The **BindToColumn** is only used with data-centric controls on PanelRows in the PanelList control. Its used to bind a data-centric controls such as a Textbox or Label to a specific column. For example, if your first column is 1, then the **BindToColum**n value should be "1". The list values can be ordered to start with 0 or

1; You can force the ordered list to start with "1" by checking the box under Configuration > Environment Properties > Environment > List Items Collection is One Based.

The **BorderStyle** property sets the style of an object's border. Border style options are: (Default), Flat, None, Raised, Sunken, Thick, Thickline, ToolTip, and Underline. If ToolTip is selected, use Themes > [element name] > BorderStyle: ToolTip: Slope Property to shape the tail of the ToolTip.

The **Brush** property is used to select a color that will override a Theme color.

The **Button** property group stylizes the search icon button that displays inside the TextBox when the OnSearch event is set to True.

The **Button Pressed** has been obsoleted in 5.2. This property was used to set the color values for this control when its selected. In 5.2 all press states styles (colors) are now calculated from the colors assigned to the button so there is no need to configure them.

The **ButtonSize** property was removed in version 5.1 of RFgen. See the **Size** property.

The **ButtonStyle** property was obsoleted in 5.2.

The **Button Pressed** property was obsoleted in 5.2. All press states are now automatically calculated to apply a selected appearance using the BackGround coloring of the button.

The **Caption** property is used to hold text that is static or dynamic.
Note that the **TextId** property under Caption is used to populate the contents of the Label or TextBox at runtime, depending on the value that's used. For more details, see the TextId property description.

Caption - LinkTo property - used to mirror the text between the textbox and one or more labels after the user taps Enter. This is accomplished by using the same ID in both controls. To enable/disable the **DataLink** property must be set to True.

The **CellMargins** property sets the distance between the contents of a cell and the cell border in pixels, and is used to make it easier to view multiple lines of text in a control.

The **Checked** property sets the status of a CheckBox prompt.

The **CheckBox** property group is used to stylize the elements in the CheckBox, ComboBox, DataGrid, ListBox, and TreeView controls. To enable the checkbox as a subfeature in controls used for listing text and values in columns or grids, you must select the "CheckBox" value in the control's Column > (Style) property. The CheckBox property group includes the BackGround, BorderStyle, ForeColor, Margins, and Size (for sizing the checkbox).

The **Colorize** property, if set to True, converts simple grey-colored images to the color selected in the **Brush** property. For example, the **Colorized Brush** can change the grey in a grey-colored Chevron icon to  to red . This tool is intended for the system icons that are provided by RFgen in the InlineButton property.

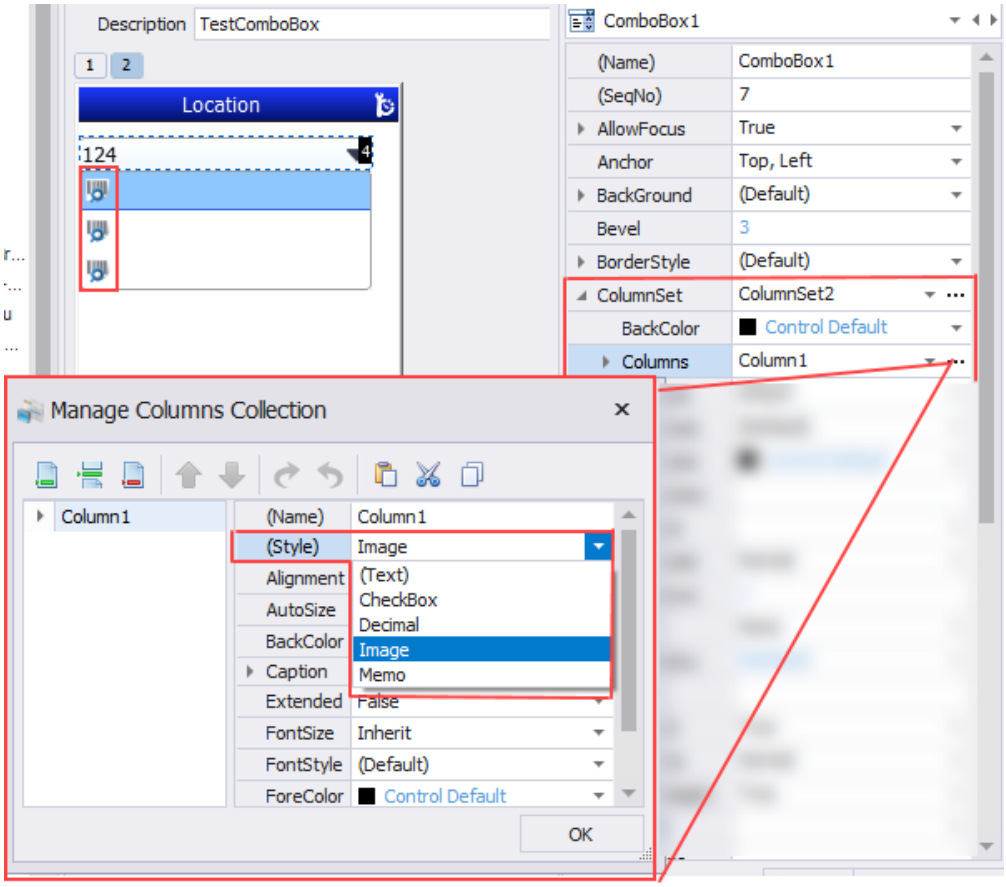
The Colorize Brush is NOT intended for changing the color of images or icons comprised of multiple colors as it will apply the same color to entire image.

The **ColumnSet** property group (available in the ComboBox) is used to set the (template) of a column and the stylizing of data each column created within a column set.
a) If you click on the down arrow, a lists of your column sets displays so you can select the one you want to

edit;
b) If you click on the elipsis (...) it opens the **Manage ColumnSet Collection** window and is used to add, insert, delete column sets and set the colors and fonts for each one.
Once your columns have been added and stylized, the subproperties and values will be listed under columns. For more details, see *Columns property below*.

The **Columns** subproperty group (under ColumnSet property) is used to customize and control the presentation of the data within a specific column. For example, if you can set column 1 to contain CheckBoxes with white text on a black background, column 2 to contain text only with a grey background and red text, and restrict data via TrimSpaces property.

The **Columns (Name)** property sets the unique that for the column for reference purposes.



The **Column (Style)** lists factory-provided styles: *(Text)* for display of read-only values, *CheckBox* for selection of data items, *Decimal* for display of numeric text, *Image* (icon), *Memo* for data entry by the user, or the *SpinEdit* button to add/decrease a value. **TrimSpaces** will remove leading and trailing spaces from the data so column alignment will be smaller. The **Width** property can either be set to a specific size or -1 to indicate that the column should stretch to the right taking up any available space.

The **Color 1** and **Color 2** properties are used in combination to create gradient colors in properties such as the Background property. These values can be set in the Custom, Named Color, or Theme tab of the color palette tool.

DataLink property enables the mirroring of data from a source object (i.e. TextBox) to the destination label. The source is set in the **Caption > LinkTo** property. False disables the link between the source and destination label.

The **Defaults** property and DefaultList property sets any number of built-in values or custom values as the initial value of the prompt. For more details, see To Set Text Defaults in a Control , or refer to the Developers Reference Guide. The Defaults property appears for property groups such ad EditText which are typically in the TextBox and Memo control.

The **Display** property. See the Form Display property.

The **DisplayMode** property provides a list of factory-provided styles (from Themes) that can be used to standardize the appearance of objects.

The **Dock** property is used to automate the placement and scale of sibling objects under a parent object. While the Anchor property is also used to position and set sizing relative to whether its anchored, the docking property forces siblings to respect the space of neighboring objects so you can dock items in a series.

For example, if a Label and TextBox were placed above a Button control and all three were on the same page, and their dock properties = Top and Fill, RFgen would first align the Label and TextBox to the top of the screen and space them out to fill the screen's width. Then RFgen would position the Button control to fill in the remaining area of the screen.

DropShadow places a dark border around the bottom and right sides of the control for a 3D effect.

The **Edits** property sets any number of built-in values as the requirement for the entered data. For example, it can be used to validate the data entered manually or data that was scanned to this field. This property is usually a member of the TextBox control or Memo control EditText property group. For a list of Edit Property Options used to validate text or perform other checks against a source, see Edit Property Options in the RFgen Developers Reference Guide.

The **EditText** property sets the color of the text (i.e. text inside a memo) that a user can edit at runtime. EditText may contain these subproperties used to stylize text at runtime: Alignment, FontSize, FontStyle, and Margins. It may also include the Edits property for inserting values that perform validations against the data entered in the prompt.

The **EntryRequired** property, set to True, forces users to enter data into the prompt, while setting it to False, allows users to skip the field. If the prompt never gets the focus, this property will not get used.

The **ErrorMessage** property is text displayed as an App.MsgBox when the data entered fails to meet the criteria in the Edits property. You can link this to a text resource if you enter the **TextId** which is stored under Solution Explorer > Text Resources.

The **ExtendedColumn** property specifies which column will be stretched to the right edge of the control. The default is the last column designated by -1 but specifying 1, 2, or 3 as examples would use the remainder of the width by stretching a middle column.

The **Events** section contains a list of Visual Basic for Applications (VBA) scripting events. The default values are set to False. If you select True, the script view for your application displays and the selected event object's script module is added. For a description of an Event, see the VBA Events topic or enter the Event's name in the Search field above. If Events is hidden, click on the Show Scripting Events in the Options Menu.

The **FocusStyle** property specifies how an object shows it is the active/selected object at run time. The default is Standard. Other options are: Active Border or Visible w/Focus.

The **FontSize** property (graphical mode only) defaults to a baseline value called "Normal" which is set in **Mobile Themes > Application > FontSize**. You increase or decrease the font size (in points) if you want to override the default value. For example, if your Normal = 11 points but you want your Labels > Caption text = 14 points, you would set your Labels > Caption > FontSize to +3. In controls where the FontSize property is nested into subproperties, you may see "Inherit" used instead of "Normal".

The **FontStyle** property (graphical mode only) sets the prompt's data field display to a particular style. The default is Normal. Other options are combinations of Bold, Italic and Underline.

The **ForeColor** property (graphical mode only) allows the user to select from a color pallet or enter a 6 character hex value (for 16 million colors) to set the fore color of the label caption of the prompt. For the controls such as the Button control, the ForeColor property will also apply to the icon, if the Icon: Colorize value is set to True.

The **Format** property is an extension of the VBA Format command and pre-formats the entered data to the mask entered here. See the VBA Format command for examples. The double quotes are not necessary as they are in the VBA Format command.

The **FrameStyle** property lets the user create rectangles, vertical or horizontal lines for the frame control only.

The **GenerateMember** property was removed in RFgen 5.2. This property helps improve an application's performance by telling RFgen what it should or shouldn't generate. The values are: *Non*e, *Member Only*, *Event Only* and *All* (which is Member + Event*)*.

None will NOT generate a member variable; *Member Only* generates a member variables with no events; *Events Only* generates object events only, but no member variables; and *All* generates both member variables and object events. The default for prompts like the Textbox and Labels is "Member Only," whereas prompts like Buttons, DataGrid, CheckBoxes, ComboBoxes, Maps, MenuList and Signature will default to "All."

The **Heading** property group consists of additional properties that are used to stylize the header of a control. You can turn on/off the display of the header (Visible = True), change its BackGround, BorderStyle, add text/caption to the Heading field, and set the FontSize, FontStyle, FontColor, margins, and associate the text/caption in the Heading field with a text resource ID for translation purposes.

The **Heading Caption** property group includes: Color - sets the caption text color and color of any characters used in the form header. FontSize and FontStyles set the caption text font style and size. The Margins (left, right, top and bottom) are in pixels and set the distance between the caption edge of the header. For example a margin of 20 pixels would make the header bigger.

Heading Icon Property Group - is used to add icon buttons and stylize the icon in the header. Icons are added for navigation or menu access purposes. The position of the icon(s) is based on factors such as how many icons were added, icon size, margin and padding allocated to the icon, and positioning (alignment setting) of the header text.

The **Image** property group is available in the ButtonList control and Menu control. For the **Image control**, the Image property group is used to select an image resource, select its alignment within its container on a

form, page, or inside another control, and set how its proportioned when its resized. For more details, see the Image Control topic. For all the ButtonList, ImageList, and Menu controls, the Image property group sets the size and margin for the image(s) listed in these controls but does not allow selection of an image resource. For the **Button control**, the Image property group is used to select an image resource, set the alignment, margins, and size within a button.

ImageID has been deprecated in 5.2. In 5.1 it was used to select an image resource.

ImageMode has been deprecated in 5.2. In 5.1 it was used to position the image (i.e. Top-Left, Top-Cener, Top-Right, Disabled, Stretch, or Tile)

ImagePath has been deprecated in 5.1. In 5.0 this property set the file path to an image located on the hard drive instead of the Resources > Images folder.

ImageSize has been deprecated in 5.1. In 5.0, it set the width and height of the graphic itself regardless of the size of the control. Images can be displayed a number of ways and this property sets the image size for graphical lists, button or desktop menu lists.

InLineButton property group - Is present only for the ComboBox, Map, SpinEdit, and TextBox controls. In the TextBox, this property group enables you to add a customized button that is associated with an event and stylized how the icon looks. For the other controls, it simply stylizes the buttons that come with the controls.

InputState puts the container/prompt in a state where its "ReadOnly" (no data can be inputted to this control). "Disabled" also prevents the control from receiving inputs. "Normal" allows inputs (i.e. write data) to the control. For more details on the VBA Extension "InputState" or see this topic in the Developer's Reference Guide.

The **IntegralHeight** property dynamically sets the height relative to the fontsize it contains and prevents manual changes to the height of the control if the IntegralHeight Property is set to True. For example, the TextBox height will change relative to the TextOptions FontSize if you change the FontSize from Normal to +32. You can change the TextBox's location and width it if you want to.

If you IntegralHeight to False, the TextBox size does not size automatically when you change the TextOptions FontSize, but you can manually make the TextBox taller and change its location and width it if you want to.

The **Items** property is used with the ButtonList, ComboBox, Menu, and other controls that list items. It provides a set of subproperties for styling the elements in a group -- some of which may not be used. These subproperties are: BackGround, BoarderStyle, ScaleText ("True" scales with the size of the item; False uses the Text FontSize), Separation (distance in pixels between items), and Size (height and width in pixels). See also "Selected Item."

The **Keyboard** The Keyboard property is listed in controls where user interaction is possible (i.e. select a radio button, enter text etc). If a virtual or specialized keyboard is needed, this property provides a drop-down list of keyboards defined from the Solution Explorer > Keyboard resource folder. **Possible Values:***None* means the keyboard type is not defined, and nothing will display at runtime. *System* is the device's native touchscreen / soft keyboard. If a type if defined, the **Visibility** property is enabled, and the options *Never*, *OnClick*, or *OnFocus* display. *Never* hides the keyboard, but can be still be shown via the sidebar menu option "Show/Hide SIP"). *OnClick* means that once the control has focus, if you click on the control it will then display the keyboard; This is not a toggle, it simply displays the keyboard. To hide it, you

would need to use scripting or change the focus to another control. *OnFocus* means when the control gets focus, it will display its keyboard.

The **KeyField** property is for linked textboxes only and designates which prompts will be used as key fields when attempting to perform an internal SQL Update statement for the linked application. This property is automatically filled in when the user downloads a table or view structure and links the application to that structure.

The **Layout** property is added to a child control when the child control becomes a member of the Layout control. The subproperties are: Col, ColSpan, DockingMode, Row, and RowSpan.

The **LineColor** property selects the color of the lines between rows or columns in a control that supports multiple rows or columns.

The **LineSize** property sets the thickness of a line in the LineControl.

The **LineStyle** property is used for list type controls that also use the Columns property. As the name implies, it makes horizontal and/or vertical visible or keeps them hidden.

The **LinkLabel** property links TextBox entries to be the specified label so that the entries are mirrored to the label at runtime.

The **LinkToPages** property is available under the Columns property and Rows property of the Layout control when the Layout control is on a form. If the Layout control is on a page, this property is not present. It creates a link between the objects in the column or row of the Layout control and the pages of a form. For example, if you linked Row 1 to pages 1 through 3, you would see the contents of Row 1 on pages 1 through 3. A common use of this property is to provide breadcrumbs of the values that were entered where each prompt was on a page and the label on that page is reflected on the top of the page progressively. The values can be the page number of the page, the name of the page, or series where the values are separated by a comma. For example for page number use: 1,2,3. For a series and a specific page use:

1-3,5. For reference by the page's name: pgSerial, pgBatch, pgQty.

If you want to link multiple pages within a row, you can also enter a series formatted as

The **ListData** property is for list boxes, combo boxes and list views only and contains a collection of values to be assigned to the prompt when the application loads.

The **ListHeading** property allows the code environment to overwrite the caption of the prompt with formatted data from a database lookup using the Prompt.List.SetColumn method.

The **ListHeight** property is for combo boxes only and sets the number of rows the control will use when displaying a list of possible values.

The **ListOptions** property group is typically included in controls that list items organized in groups or individual rows.

The **ListOptions, FilterMode** enables filtering using SQL filtering by what the search criteria contains or starts with.

The **ListOptions, Images** sets the presentation of an image used in the list.

The **ListOptions, LineStyle** sets the presentation of the lines used to separate rows or columns in the list.

The **ListOptions, MultiSelect** property is used in list controls, and will enable users to select more than one area (i.e. cell or row etc) if set to True.

The **ListOptions, MultiSelect, UpdateColumn** property specifies whether the which columns to update when the multiselect is enabled in a list control.
For other controls, the row styles are similar but may have different properties.

The **ListOptions, PersistData** if set to True will cause the data to remain the same unless its specifically changed. False allows the data to be changed (i.e. Lost data in a form reload).

The **ListSorted** property is for list boxes, combo boxes and list views only and keeps the contents of the list sorted.

The **ListStyle** property changes the presentation of the data displayed between a Standard text list, an Image List that uses images next to the text description, Buttons or Desktop style like a Windows desktop. This is the control used on the internal RFMenu form.

The **Location** property sets the position of the control in pixels for graphical applications. The location is relative to the parent container and may have different values for different controls. For example the Layout Control location is identified by the number of pixels from the top-left of a form. But if a button was inside the Layout control, the button's location , whereas a and in rows and columns for fixed-length character applications.

The **Logo Property Group** is used primarily in Mobile Themes - Applications.

The **LockAspect** property is used to lock the x and y ratios of an image so that they don't become distorted if stretched or resized when the application is displayed on a larger/smaller screen or if the device is rotated.

The **LockOrientation** property. This is a member of the Form Display group. For details, see Form Properties.

The **Manage Collection** property. See ColumnSet Property.

The **Manage ColumnSet Collection** is available for the **ColumnSet** property in the ComboBox, DataGrid, Layout, ListBox, and TreeView controls. Its used to add, insert, and remove columns. In controls used to list text and values, the Collection helps you design and stylize columns. You can stylize text and the control's background color, set your alignments, set the caption, enable or disable "TrimSpaces," and allow the columns to be extended, formatted, and visible (or hidden). In the ComboBox or Layout controls , these properties are not present.
In the Layout control (used for containing other controls) the Manage Columns Collection is designed to add, insert, and remove columns and assign column names and SizeModes (values that set how the object in the cell of the Layout control will be sized), and whether the column is hidden or visible. For more details, see *How to use the Layout Control*.

The **Manage Displays Collection** property is used to set the values a specific display in a Form control. This way, your form can be linked to a display for different languages but still be used for the same app.

The **Manage Icons Collection** property group is used to add or remove icons, and if in a list, set the list order.

The **Manage Rows Collection** property. See Manage Columns Collection.

The **Margins** property is used to pad the spacing between the rows or images of the displayed data.

The **MaskInput** property (available only in the TextBox control) is used to mask the input with asterisks if the value is True. The default is set to False.

The **MaxWidth** property is used to set the maximum space allowed for the control. For example, the Radio Button maximum width is the widest space allowed for the button and its text label.

The **MenuIcon** property on the Form is used to provide a background image.

The **Multiline** property was used in older versions for long strings in a control under the "TextOptions" property. It has been deprecated in 5.2.3.

The **(Name)** property is the internal name / identifier of a control/prompt. Tip: As a best practice, follow the Hungarian notation where textboxes are named 'txtPart' and list boxes are named 'lstParts' as examples. This way, when referring to them in the script, there is an inherent understanding of what types of data will be used for the prompt.

The **NormalizeText** property will trim the spaces from both sides of the displayed data or captions of the buttons or desktop icons.

The **NullText property** - was replaced by TextHints. It was used to help users know what should be entered, but is not retained as a data value for transactions.

The **Overflow** property specifies which way the remaining items will be displayed. If there are more items than will fit on the device's screen this option can be set to horizontal or vertical which means the user can swipe bottom-to-top or right-to-left to access the remaining data.

The **Padding** property sets in pixels, the amount of space between the left, top, right, and bottom edge of a control and other objects. The spacing remains constant even if the size of the control changes.

The **(PageNo)** property lists the order sequence of a page on a form.

The **PageMode** property is only available in the PanelList control. When set to True, it enables a user to swipe the screen horizontally to view more rows of data.

The **Password** property, for the data field portion of the prompt, sets the display of the text equal to asterisks (*) instead of clear text.

The **PenColor** property sets the color of a signature in the Signature control. In Mobile Themes Signature element, the pen color inherits the value from the ForeColor property.

The **Pressed property** changes the style and color of a control (i.e. Button, Button List, ComoboBox, DataGrid, Tab) when its pressed. This property is available for all pressable controls and list theme colors and a palette of configurable colors at the control and theme level.

The **PromptNo** property is the Prompt sequence value RFgen assigns to objects placed on a form or a page. If you move the location of a control or page, the PromptNo also updates, or you can edited it directly in the Application Designer Control Panel. If your script uses App.PromptNo VBA extension, it should return the PromptNo value for the specified control.

The **Required** property is used in prompts such as the Memo control where input is required before the user can continue to the next page, prompt or task in the application.

The **RowAltColor** and **RowSelector** (TreeView Control only) properties sets every other row to the color selected and enables users to select the row (True) as opposed to just viewing the content in each row.

The **ScaleMode** property is a Form Display property that specifies how the form and form elements will scale when the application is deployed the actual device. For more details, see Form Properties > Display: ScaleMode .

The **ScaleToFit** property increases or decreases the icon captions to fit inside buttons or desktop icons when the overall size of the button or desktop tile size is changed. This feature is enabled when its True and is disabled if set to False.

The **ScaleText** property automatically scales the text front to fit the space in a control and is available for these controls: Button, Checkbox, DateTime, Label, and RadioButton. True enables this feature, and False disables it. **ScaleText** property settings on individual controls take precedence over the **Configuration > Environment > Auto-Scale Text to Fit Controls** settings. If the **ScaleText** value is set to " (Default)" then the value which is set in Themes will be used for the control.

The **Scrollbars** can be enabled for horizontal scrolling, vertical scrolling, both, or none. If set to Automatic, RFgen calculates and displays the scrollbars for you. This property is for select controls only. See Mobile Themes > Scrollbar for more details.

The **Selection** property group is used stylize a selected item from a group of items or list in a parent control such as the ButtonList, CompboBox, DesktopIcons, ImageList, ListBox, or PanelList. The subproperties include BackGround, Border, Text, Transparency and Visible. To compare against the unselected item properties, see Items property topic.

The **ShowBorder** property will hide or show the border of an element. The values are True, False, or (Default). *(Default)* uses the property value set for the control (element) in Mobile Themes. *True* will display the border; *False* will hide it.

The **ShowInForm** is a child property of the Menu: Heading property. If the Heading property is set to True, then ShowInForm can be set (True/False) to suppress the menu's heading caption. **True** hides (suppresses) the Menu's own heading, and the Form's heading is displayed as the Menu's heading. **False** will enable the menu's heading and remain visible if the Menu control object is not obscuring the Form's header. If the menu object was set to the same dimensions of the form (and overlayed the header), then Form's heading will be used as the Menu's heading even though ShownInForm is set to false.

The **ShowLines** property will hide or show the lines between rows and columns. The options are (Default) which uses the theme properties, None for hiding all the lines, Horizontal for showing only the lines between rows, Vertical for showing only the lines between columns, and Both for showing lines between rows and columns.

The **SelColor** refers to the color of the selection bar shown in controls like the combo box or list box. The highlighted value is what will be chosen when the user presses the enter key.

The **Size** is the background height and width of the control in pixels. If AutoSize is used, it can influence the height and width values. In the ButtonList control, this sizes the buttonlist container-- not the items in the list.

The **Sorted** This was replaced by SortMode in RFgen 5.2.

The **SortMode** property is used in list controls and will sort content in accordance to the value selected. The values are "(Default)", None, Ascending, or Descending.

The **Source** property (the HostScreen control only) selects an executable to be emulated within the Host Screen control.

The **StretchImage** is used to either shape an image to the size of the control or allow the image to be its natural size whether it fits in the control or not.

The **Style** property in the ButtonList control specifies whether the items in the ButtonList are to be presented as a squarish buttons or icons (text and icon but no button as a container). In the ComboBox control, the Style property provides an option to display the drop down list as full screen or display the drop down menu in its standard size.

The **SystemIcons** property group is used to associate RFgen-supplied actions (Call Event) with a customized icon or a RFgen-supplied icon. For example, if you wanted to include a Cancel operation in your application, use the functions/properties in this group to select the Cancel operation and also associate it with an icon. When you are done adding icons, a value in brackets { } shows the total number of icons associated with the control. For specific details on how to link a customized icon to a Action, see To link SystemIcons with system operations.

The **TabNo** property is cursor/prompt sequence number for controls that can accept the focus. This property is not the same as the SeqNo property. For example, a Label control cannot accept focus but have a SeqNo but will not a TabNo. Since a TextBox can have focus and have a SeqNo and TabNo. You can edit a TabNo so to force where the cursor goes after a specific object loses focus.

To view the list of controls/objects' tab numbers, click on the  Sort icon or see Sort Controls.

The **TagLine** properties are used to stylize the text that appears above a ComboBox, TextBox, Memo, or SpinEdit control when a **TextHint** is used as a visual clue on what type of data needs to be entered into the prompt/field. (For example, you can have "3000" appear as a tagline above TextBox.) The TagLine properties include the FontSize, ForeColor, and Separation -- the number of pixels of space between the text and the top of the box.

The **Text** property group is used in the ButtonList Control and specifies the color of the font, size of the font, style of the font, and position via the margin values. Whether the values here are overridden by other property settings in the control will depend on how deep this property is nested.

The **TextHint: DisplayMode** property group is used to set the position of the TextHint. The optons are: (Default), Both, External, Internal, Toggle and WaterMark. External - adds a tag line above an empty contro t then disappears after the entry is make. Internal displays a tagline inside the prompt but leaves space for the user to enter the prompt then goes away after a value has been entered. Watermark shows your texthint inside the prompt, and goes away when the user overwrites an entry; if no entry is make it stays there. "Both" will add the tiny TextHint above and inside the prompt.

After an entry is made in the control, you can use the **TextHint** properties to: a) Set a tagline to display above the control after the user entered data over a watermark; b) Set a tagline to continue to display above a control after an entry was made to a blank control; c) Have the tag line disappear after an entry was made; or d) not use a tag line at all (nothing displays before or after entry.)

The **TextID** property links a text (word or statement that is unique to the control) to a resource listed in the **Solution Explorer > Text Resources** table so that the value will be localized (translated) in the application at runtime. If the identifier is the TextId from the table in the Solution Explorer > Text Resources folder, the corresponding string in Translate Text column will display as the text string for the Caption, Heading, Error Message, or NullText. If the value is preceeded with a "%", it links user inputs from the source TextBox to a destination Label at runtime.

For more details see Text Resources or To apply translations across all applications.

The **TextOptions** property sets whether the text will be on a Single Line or Multiple Lines. Expand the property group to set how the text will wrap if Multi-Line is selected. In 5.2.4.x and higher, length of the Multi-Line text will be set under the Size > Width property. When the Muli-Line property is checked, and the width is used to set the wrapped text's width, the Auto Size is disabled.

The **TextOptions** property group in the ButtonList, IconList, ImageList, and Menu. In this group, TextOption sets the values for the FontSize and FontStyle. The sizing of the icon names (captions) and icons in a button/tile are affected by the margin, **ScalePoint**, and **ScaleText**. The **Margin** is the distance between text and the edge of the icon container. **ScalePoints** reduces the size of the text by the factor supplied. For example **ScalePoint** 0 or 1 sets the text default size; 2 makes the text 2 times smaller then the icon size. **ScaleText**, if set to True enables scaling of text; False prevents it from being scaled (reduced).

The **Theme** property changes the border of the title bar area to one of several hardcoded styles.

The [**Title**] property in Themes > Dialog is used stylize in the title of a Dialog box (pop up messages dialog box).

The **ToolTip** property is a value that can be selected from any control that supports a BorderStyle.

The **Transparency** property sets the level of transparency of a child control and bases its coloring on the parent (i.e.Form, Page1, Panel). "(Default)" uses the level of transparency set in the corresponding Mobile Theme element. "None" will use the item's BackGround color. A percentage value (5 %, 10% etc), sets the transparency level.

The **UpdateColumn** property. See ListOptions > MultiSelect > UpdateColumn.

The **UseMenuTheme** property will override the local properties and apply the default theme properties for the menu control.

The **ValidationTable** property presents a list of downloaded tables that can be used to verify that the data entered already exists in this table and the Validation Field. The two properties must be used together.

The **ValidationField** property presents a list of table fields specified by the Validation Table property. This is the reference field to determine if the data entered in the prompt already exists. If it does not, the Error Message property will be used to warn the user.

The **Visible** property, set to True, makes a prompt or element visible, while setting it to False makes it invisible. Even though the prompt may be invisible, the GotFocus, OnEnter and Lost Focus events will still be executed for this prompt if the focus automatically shifts from a prompt before this prompt to one after this prompt.

The **WaterMark** property group is used with the **TextHint** property group. The WaterMark is a type of caption that displays temporarily inside a ComboBox, TextBox, Memo and SpinEdit control so the user has a

clue what to enter into the prompt/field. Once the user enters data, the watermark caption is overwritten. The WaterMark properties include the ForeColor of the watermark text, and Transparency (darkness or opacity of the text).

The **ZOrder** was obsoleted in 5.1.

The **Design Mode** (which used to be under Display tab) has to **Configuration > Desktop Preferences** in version 5.1.

Last updated Oct 23, 2023

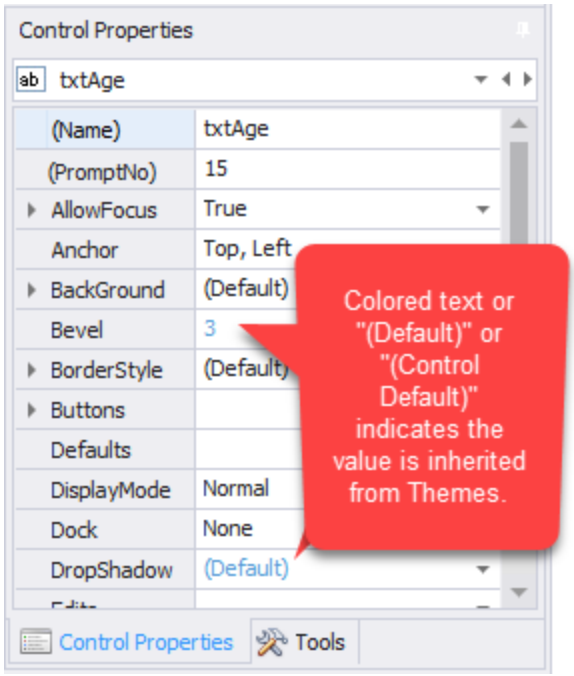
Graphical Control Properties

The **(Name)** property is a standard property for all controls except the Form.

The **(PromptNo)** property is the Prompt sequence value RFgen assigns to objects placed on a form or a page. If you move the location of a control or page, the PromptNo also updates, or you can edited it directly in the Application Designer Control Panel.

If your script uses App.PromptNo VBA extension, it should return the PromptNo value for the specified control.

HOW TO TELL IF YOUR THEMES ARE BEING APPLIED



If the value for a property appears in colored text or the value shows (Default) or Control Default this means the property value is using the corresponding value set in Themes.

Control Property Descriptions

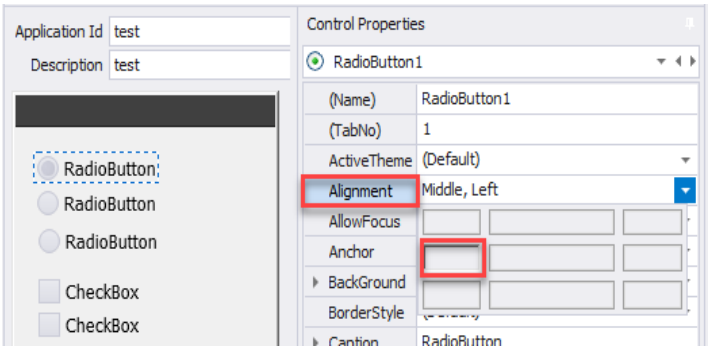
Descriptions for all the control properties are listed alphabetically on this page. Tap **Ctrl + F** keys to **Search** for a property definition on this page.

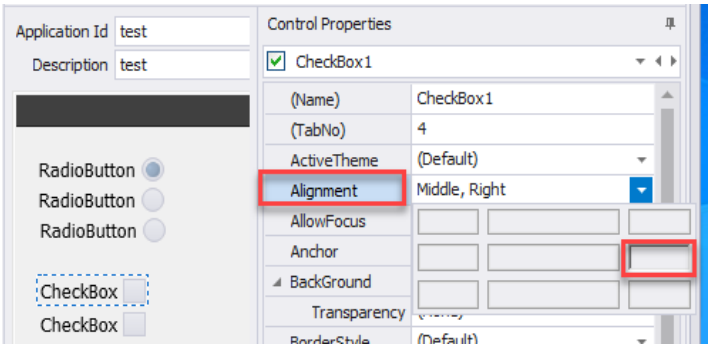
Not all properties appear for all control types. For example Image controls, CheckBox, RadioButton, Page controls and others will have their own unique properties.

The **Action** property under the **Button** property provides built-in operations for buttons in controls like the TextBox. For example, TextBox1> Buttons> Action = Search, and the TextBox1 > Button > Image = Search icon, and Buttons > Visible = Ture will add the OnSearch event in your Script View. You can also set

TextBox1 > Buttons > Action = Event, Buttons > Visible = True, and set TextBox1 > Events > [select an Event like OnScan=True, and this will add the OnScan Procedure to your script.

The **Alignment** property places text and images relative to another object within a control. For the Check-Box and Radio Button controls, the images are shifted relative to the labels when you select a position in the Alignment drop down menu. (See example below)





For the Button control, Alignment positions the Caption text and the image within the button space. To separate the image from the Caption text, use the Image > Alignment to position the image relative to the Caption text. For the Image control, the image is positioned relative the edges of the image box.

The **AllowFocus** if set to True, enables the Page object or a prompt to receive focus and then select the actions that the user can take once the object receives focus. "False" prevents the control from receiving focus, removes a TabNo assignment or prevents one from being assigned. You can enable these options when the object is able to receive focus:

- *AllowBackup*- to the previous control;
- *AutoSelectText* - selects all the text in the edit where the prompt lands if set to true; places cursor at the end of edit if false;
- *EraseOnBackup* - will erase the text if user backs up;
- *FocusOnClick* - This;
- *OnEnter - Advance* - dropdown option for: **Exit** the form, **Hold** the focus, **Reset** the form (i.e. Clear entries), or **Submit** when the user taps the Enter key.

The **AllowRoll** property is used on controls that contain long lists of items and helps you "jump" to items faster. For example if you press the up arrow while on the first line in a long list of items, the last item from the list will display. Or, if you are at the bottom of a list, and press a down arrow, you are taken to the top of the list.

The **AlternateItem** property group sets the background color, font color (ForeColor), visibility of the colors for alternate rows in a list (ComboBox, DataGrid, ListBox, PanelList etc.). To display the alternate color, set Visible to True.

The **Anchor** property sets a child object's position and dimensions relative to where its anchored to the parent. For example if a TextBox control was anchored to a Form on all four sides, the TextBox would be proportionally resized if the Form was displayed on a different mobile device. But if the TextBox was Anchor equaled "Bottom", "Right", then its size won't change, and its position will stay relatively the same. If no other anchors except the Bottom is set, then RFgen automatically centers the child object inside the parent. If the parent is resized, the chile object's size is not changed even if the parent's size changes.

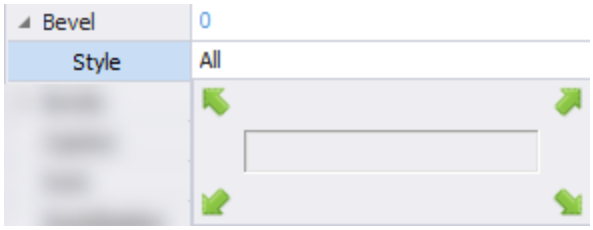
The **AutoSize** property will stretch the object's background to the lowest and right-most portion of the screen, depending on the option selected. This property is available for the Button, DateTime, Frame, Label, Layout, and Panel controls. The values are: (None), Content, Height, and Width. If AutoSize is set to "(None)" then you can drag-n-resize the control. But if the control is in AutoSize mode, the ability to drag-n-resize the control is disabled and the control is sized by its content, width or height. For example, if you have a Panel control that was empty (no controls inside it), and AutoSize was set to "Contents," "Height," or "Width," then the Panel will automatically resize to 0 pixels. If there was an image in the Panel, and AutoSize was set to "Contents" then it would resize the Panel around the image relative using the right-most and bottom portion of the panel as the start location. See the Size property for the size in pixels. See Manage Paged Collection for details on AutoSize Property for Pages.

The **BackColor and BackColorAlt** (previously called **BackColor(1)** and **BackColor(2)**) properties are used to create either solid backgrounds or gradients depending on the option chosen in the Background Fill property. The color can be set from using Custom Color tab or by a 6-character hex value (which gives you 16 million colors to choose from).

The **BackGradient** This property was replaced by the **BackGround** property.

The **BackGround** property has three categories for setting the background color of a control. If set to Solid, RFgen applies the value from Color 1. If a directional values such as Diagonal Right, Diagonal Left, Vertical etc. is selected, RFgen applies the values from **Color 1** and **Color 2** to create a gradient background. If Transparent is selected, the next-closest background color behind the control will be used. For example, if your Mobile Theme Application BackGround = Red, and your Mobile Theme Label BackGround=Transparent, and your Label Caption color = White, then your Label text will appear as white on a red background in Mobile Themes. In the application designer, the Label text will also appear as white text on a red background if the application's Form Active Theme = (Default) and the Label Active Theme = (Default). But, if your Mobile Theme Label BackGround = Solid and the colors 1 and 2 = Pink, then in the solution designer, the label's background will appear as white text on pink because uses the Label's solid background color blocks the background color from the Application Form.

The **Bevel** (graphical mode only) property sets the curvature of a square's and rectangle's corner edge where 0 is no bevel and 100 turns the object into a circle or oval shape. The **Bevel Corners**



The **BindToColumn** is only used with data-centric controls on PanelRows in the PanelList control. Its used to bind a data-centric controls such as a Textbox or Label to a specific column. For example, if your first column is 1, then the **BindToColumn** value should be "1". The list values can be ordered to start with 0 or 1; You can force the ordered list to start with "1" by checking the box under Configuration > Environment Properties > Environment > List Items Collection is One Based.

The **BorderStyle** property sets the style of an object's border. Border style options are: (Default), Flat, None, Raised, Sunken, Thick, Thickline, ToolTip, and Underline. If ToolTip is selected, use Themes > [element name] > BorderStyle: ToolTip: Slope Property to shape the tail of the ToolTip.

The **Brush** property is used to select a color that will override a Theme color.

The **Button** property group stylizes the search icon button that displays inside the TextBox when the OnSearch event is set to True.

The **Button Pressed** has been obsoleted in 5.2. This property was used to set the color values for this control when its selected. In 5.2 all press states styles (colors) are now calculated from the colors assigned to the button so there is no need to configure them.

The **ButtonSize** property was removed in version 5.1 of RFgen. See the **Size** property.

The **ButtonStyle** property was obsoleted in 5.2.

The **Button Pressed** property was obsoleted in 5.2. All press states are now automatically calculated to apply a selected appearance using the BackGround coloring of the button.

The **Caption** property is used to hold text that is static or dynamic.
Note that the **TextId** property under Caption is used to populate the contents of the Label or TextBox at runtime, depending on the value that's used. For more details, see the TextId property description.

Caption - LinkTo property - used to mirror the text between the textbox and one or more labels after the user taps Enter. This is accomplished by using the same ID in both controls. To enable/disable the **DataLink** property must be set to True.

The **CellMargins** property sets the distance between the contents of a cell and the cell border in pixels, and is used to make it easier to view multiple lines of text in a control.

The **Checked** property sets the status of a CheckBox prompt.

The **CheckBox** property group is used to stylize the elements in the CheckBox, ComboBox, DataGrid, ListBox, and TreeView controls. To enable the checkbox as a subfeature in controls used for listing text and values in columns or grids, you must select the "CheckBox" value in the control's Column > (Style) property. The CheckBox property group includes the BackGround, BorderStyle, ForeColor, Margins, and Size (for sizing the checkbox).

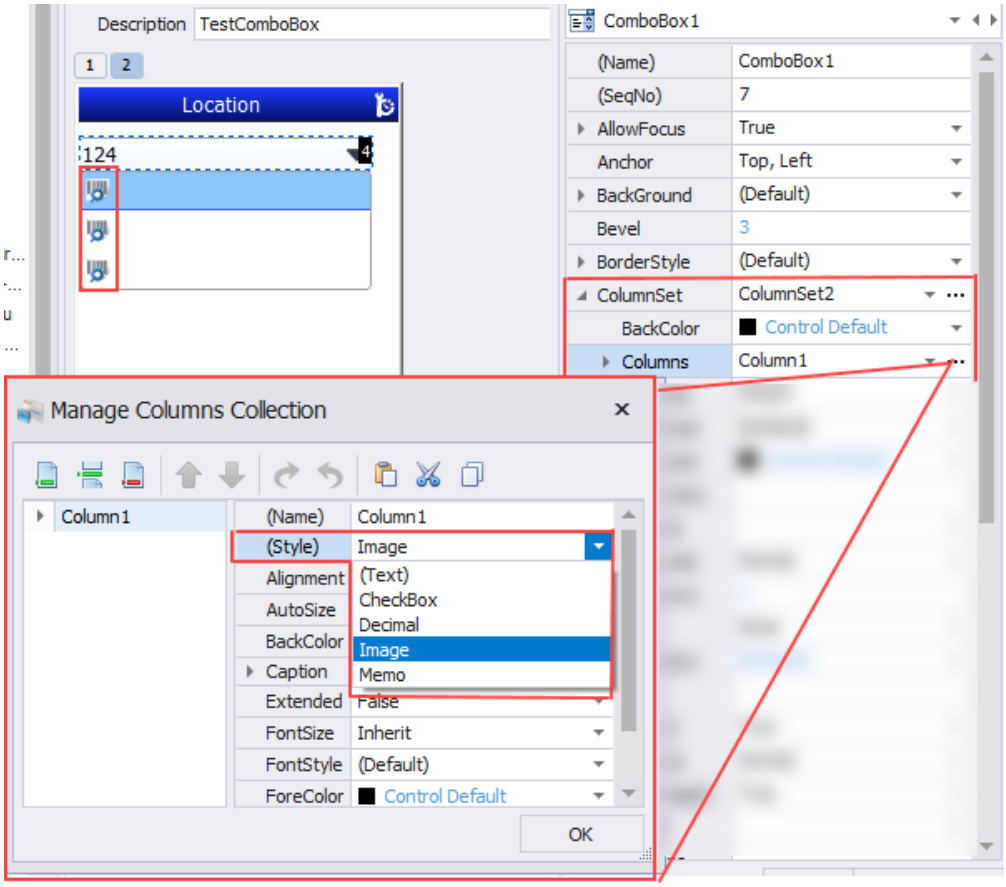
The **Colorize** property, if set to True, converts simple grey-colored images to the color selected in the **Brush** property. For example, the **Colorized Brush** can change the grey in a grey-colored Chevron icon to « to red «. This tool is intended for the system icons that are provided by RFgen in the InlineButton property.

The Colorize Brush is NOT intended for changing the color of images or icons comprised of multiple colors as it will apply the same color to entire image.

The **ColumnSet** property group (available in the ComboBox) is used to set the (template) of a column and the stylizing of data each column created within a column set.
a) If you click on the down arrow, a lists of your column sets displays so you can select the one you want to edit;
b) If you click on the elipsis (...) it opens the **Manage ColumnSet Collection** window and is used to add, insert, delete column sets and set the colors and fonts for each one.
Once your columns have been added and stylized, the subproperties and values will be listed under columns. For more details, see *Columns property below*.

The **Columns** subproperty group (under ColumnSet property) is used to customize and control the presentation of the data within a specific column. For example, if you can set column 1 to contain CheckBoxes with white text on a black background, column 2 to contain text only with a grey background and red text, and restrict data via TrimSpaces property.

The **Columns (Name)** property sets the unique that for the column for reference purposes.



The **Column (Style)** lists factory-provided styles: *(Text)* for display of read-only values, *CheckBox* for selection of data items, *Decimal* for display of numeric text, *Image* (icon), *Memo* for data entry by the user, or the *SpinEdit* button to add/decrease a value. **TrimSpaces** will remove leading and trailing spaces from the data so column alignment will be smaller. The **Width** property can either be set to a specific size or -1 to indicate that the column should stretch to the right taking up any available space.

The **Color 1** and **Color 2** properties are used in combination to create gradient colors in properties such as the Background property. These values can be set in the Custom, Named Color, or Theme tab of the color palette tool.

DataLink property enables the mirroring of data from a source object (i.e. TextBox) to the destination label. The source is set in the **Caption > LinkTo** property. False disables the link between the source and destination label.

The **Defaults** property and DefaultList property sets any number of built-in values or custom values as the initial value of the prompt. For more details, see To Set Text Defaults in a Control , or refer to the Developers Reference Guide. The Defaults property appears for property groups such ad EditText which are typically in the TextBox and Memo control.

The **Display** property. See the Form Display property.

The **DisplayMode** property provides a list of factory-provided styles (from Themes) that can be used to standardize the appearance of objects.

The **Dock** property is used to automate the placement and scale of sibling objects under a parent object. While the Anchor property is also used to position and set sizing relative to whether its anchored, the docking property forces siblings to respect the space of neighboring objects so you can dock items in a series.

For example, if a Label and TextBox were placed above a Button control and all three were on the same page, and their dock properties = Top and Fill, RFgen would first align the Label and TextBox to the top of the screen and space them out to fill the screen's width. Then RFgen would position the Button control to fill in the remaining area of the screen.

DropShadow places a dark border around the bottom and right sides of the control for a 3D effect.

The **Edits** property sets any number of built-in values as the requirement for the entered data. For example, it can be used to validate the data entered manually or data that was scanned to this field. This property is usually a member of the TextBox control or Memo control EditText property group. For a list of Edit Property Options used to validate text or perform other checks against a source, see Edit Property Options in the RFgen Developers Reference Guide.

The **EditText** property sets the color of the text (i.e. text inside a memo) that a user can edit at runtime. EditText may contain these subproperties used to stylize text at runtime: Alignment, FontSize, FontStyle, and Margins. It may also include the Edits property for inserting values that perform validations against the data entered in the prompt.

The **EntryRequired** property, set to True, forces users to enter data into the prompt, while setting it to False, allows users to skip the field. If the prompt never gets the focus, this property will not get used.

The **ErrorMessage** property is text displayed as an App.MsgBox when the data entered fails to meet the criteria in the Edits property. You can link this to a text resource if you enter the **TextId** which is stored under Solution Explorer > Text Resources.

The **ExtendedColumn** property specifies which column will be stretched to the right edge of the control. The default is the last column designated by -1 but specifying 1, 2, or 3 as examples would use the remainder of the width by stretching a middle column.

The **Events** section contains a list of Visual Basic for Applications (VBA) scripting events. The default values are set to False. If you select True, the script view for your application displays and the selected event object's script module is added. For a description of an Event, see the VBA Events topic or enter the Event's name in the Search field above. If Events is hidden, click on the Show Scripting Events in the Options Menu.

The **FocusStyle** property specifies how an object shows it is the active/selected object at run time. The default is Standard. Other options are: Active Border or Visible w/Focus.

The **FontSize** property (graphical mode only) defaults to a baseline value called "Normal" which is set in **Mobile Themes > Application > FontSize**. You increase or decrease the font size (in points) if you want to override the default value. For example, if your Normal = 11 points but you want your Labels > Caption text = 14 points, you would set your Labels > Caption > FontSize to +3. In controls where the FontSize property is nested into subproperties, you may see "Inherit" used instead of "Normal".

The **FontStyle** property (graphical mode only) sets the prompt's data field display to a particular style. The default is Normal. Other options are combinations of Bold, Italic and Underline.

The **ForeColor** property (graphical mode only) allows the user to select from a color pallet or enter a 6 character hex value (for 16 million colors) to set the fore color of the label caption of the prompt. For the controls such as the Button control, the ForeColor property will also apply to the icon, if the Icon: Colorize value is set to True.

The **Format** property is an extension of the VBA Format command and pre-formats the entered data to the mask entered here. See the VBA Format command for examples. The double quotes are not necessary as they are in the VBA Format command.

The **FrameStyle** property lets the user create rectangles, vertical or horizontal lines for the frame control only.

The **GenerateMember** property was removed in RFgen 5.2. This property helps improve an application's performance by telling RFgen what it should or shouldn't generate. The values are: *None*, *Member Only*, *Event Only* and *All* (which is Member + Event*)*.

None will NOT generate a member variable; *Member Only* generates a member variables with no events; *Events Only* generates object events only, but no member variables; and *All* generates both member variables and object events. The default for prompts like the Textbox and Labels is "Member Only," whereas prompts like Buttons, DataGrid, CheckBoxes, ComboBoxes, Maps, MenuList and Signature will default to "All."

The **Heading** property group consists of additional properties that are used to stylize the header of a control. You can turn on/off the display of the header (Visible = True), change its BackGround, BorderStyle, add text/caption to the Heading field, and set the FontSize, FontStyle, FontColor, margins, and associate the text/caption in the Heading field with a text resource ID for translation purposes.

The **Heading Caption** property group includes: Color - sets the caption text color and color of any characters used in the form header. FontSize and FontStyles set the caption text font style and size. The Margins (left, right, top and bottom) are in pixels and set the distance between the caption edge of the header. For example a margin of 20 pixels would make the header bigger.

Heading Icon Property Group - is used to add icon buttons and stylize the icon in the header. Icons are added for navigation or menu access purposes. The position of the icon(s) is based on factors such as how many icons were added, icon size, margin and padding allocated to the icon, and positioning (alignment setting) of the header text.

The **Image** property group is available in the ButtonList control and Menu control. For the **Image control**, the Image property group is used to select an image resource, select its alignment within its container on a form, page, or inside another control, and set how its proportioned when its resized. For more details, see the Image Control topic. For all the ButtonList, ImageList, and Menu controls, the Image property group sets the size and margin for the image(s) listed in these controls but does not allow selection of an image resource. For the **Button control**, the Image property group is used to select an image resource, set the alignment, margins, and size within a button.

ImageID has been deprecated in 5.2. In 5.1 it was used to select an image resource.

ImageMode has been deprecated in 5.2. In 5.1 it was used to position the image (i.e. Top-Left, Top-Cener, Top-Right, Disabled, Stretch, or Tile)

ImagePath has been deprecated in 5.1. In 5.0 this property set the file path to an image located on the hard drive instead of the Resources > Images folder.

ImageSize has been deprecated in 5.1. In 5.0, it set the width and height of the graphic itself regardless of the size of the control. Images can be displayed a number of ways and this property sets the image size for graphical lists, button or desktop menu lists.

InLineButton property group - Is present only for the ComboBox, Map, SpinEdit, and TextBox controls. In the TextBox, this property group enables you to add a customized button that is associated with an event and stylized how the icon looks. For the other controls, it simply stylizes the buttons that come with the controls.

InputState puts the container/prompt in a state where its "ReadOnly" (no data can be inputted to this control). "Disabled" also prevents the control from receiving inputs. "Normal" allows inputs (i.e. write data) to the control. For more details on the VBA Extension "InputState" or see this topic in the Developer's Reference Guide.

The **IntegralHeight** property dynamically sets the height relative to the fontsize it contains and prevents manual changes to the height of the control if the IntegralHeight Property is set to True. For example, the TextBox height will change relative to the TextOptions FontSize if you change the FontSize from Normal to +32. You can change the TextBox's location and width it if you want to.

If you IntegralHeight to False, the TextBox size does not size automatically when you change the TextOptions FontSize, but you can manually make the TextBox taller and change its location and width it if you want to.

The **Items** property is used with the ButtonList, ComboBox, Menu, and other controls that list items. It provides a set of subproperties for styling the elements in a group -- some of which may not be used. These

subproperties are: BackGround, BoarderStyle, ScaleText ("True" scales with the size of the item; False uses the Text FontSize), Separation (distance in pixels between items), and Size (height and width in pixels). See also "Selected Item."

The **Keyboard** The Keyboard property is listed in controls where user interaction is possible (i.e. select a radio button, enter text etc). If a virtual or specialized keyboard is needed, this property provides a drop-down list of keyboards defined from the Solution Explorer > Keyboard resource folder. **Possible Values:***None* means the keyboard type is not defined, and nothing will display at runtime. *System* is the device's native touchscreen / soft keyboard. If a type if defined, the **Visibility** property is enabled, and the options *Never*, *OnClick*, or *OnFocus* display. *Never* hides the keyboard, but can be still be shown via the side-bar menu option "Show/Hide SIP"). *OnClick* means that once the control has focus, if you click on the control it will then display the keyboard; This is not a toggle, it simply displays the keyboard. To hide it, you would need to use scripting or change the focus to another control. *OnFocus* means when the control gets focus, it will display its keyboard.

The **KeyField** property is for linked textboxes only and designates which prompts will be used as key fields when attempting to perform an internal SQL Update statement for the linked application. This property is automatically filled in when the user downloads a table or view structure and links the application to that structure.

The **Layout** property is added to a child control when the child control becomes a member of the Layout control. The subproperties are: Col, ColSpan, DockingMode, Row, and RowSpan.

The **LineColor** property selects the color of the lines between rows or columns in a control that supports multiple rows or columns.

The **LineSize** property sets the thickness of a line in the LineControl.

The **LineStyle** property is used for list type controls that also use the Columns property. As the name implies, it makes horizontal and/or vertical visible or keeps them hidden.

The **LinkLabel** property links TextBox entries to be the specified label so that the entries are mirrored to the label at runtime.

The **LinkToPages** property is available under the Columns property and Rows property of the Layout control when the Layout control is on a form. If the Layout control is on a page, this property is not present. It creates a link between the objects in the column or row of the Layout control and the pages of a form. For example, if you linked Row 1 to pages 1 through 3, you would see the contents of Row 1 on pages 1 through 3. A common use of this property is to provide breadcrumbs of the values that were entered where each prompt was on a page and the label on that page is reflected on the top of the page progressively. The values can be the page number of the page, the name of the page, or series where the values are separated by a comma. For example for page number use: 1,2,3. For a series and a specific page use:

1-3,5. For reference by the page's name: pgSerial, pgBatch, pgQty.

If you want to link multiple pages within a row, you can also enter a series formatted as

The **ListData** property is for list boxes, combo boxes and list views only and contains a collection of values to be assigned to the prompt when the application loads.

The **ListHeading** property allows the code environment to overwrite the caption of the prompt with formatted data from a database lookup using the Prompt.List.SetColumn method.

The **ListHeight** property is for combo boxes only and sets the number of rows the control will use when displaying a list of possible values.

The **ListOptions** property group is typically included in controls that list items organized in groups or individual rows.

The **ListOptions, FilterMode** enables filtering using SQL filtering by what the search criteria contains or starts with.

The **ListOptions, Images** sets the presentation of an image used in the list.

The **ListOptions, LineStyle** sets the presentation of the lines used to separate rows or columns in the list.

The **ListOptions, MultiSelect** property is used in list controls, and will enable users to select more than one area (i.e. cell or row etc) if set to True.

The **ListOptions, MultiSelect, UpdateColumn** property specifies whether the which columns to update when the multiselect is enabled in a list control.
For other controls, the row styles are similar but may have different properties.

The **ListOptions, PersistData** if set to True will cause the data to remain the same unless its specifically changed. False allows the data to be changed (i.e. Lost data in a form reload).

The **ListSorted** property is for list boxes, combo boxes and list views only and keeps the contents of the list sorted.

The **ListStyle** property changes the presentation of the data displayed between a Standard text list, an Image List that uses images next to the text description, Buttons or Desktop style like a Windows desktop. This is the control used on the internal RFMenu form.

The **Location** property sets the position of the control in pixels for graphical applications. The location is relative to the parent container and may have different values for different controls. For example the Layout Control location is identified by the number of pixels from the top-left of a form. But if a button was inside the Layout control, the button's location , whereas a and in rows and columns for fixed-length character applications.

The **Logo Property Group** is used primarily in Mobile Themes - Applications.

The **LockAspect** property is used to lock the x and y ratios of an image so that they don't become distorted if stretched or resized when the application is displayed on a larger/smaller screen or if the device is rotated.

The **LockOrientation** property. This is a member of the Form Display group. For details, see Form Properties.

The **Manage Collection** property. See ColumnSet Property.

The **Manage ColumnSet Collection** is available for the **ColumnSet** property in the ComboBox, DataGrid, Layout, ListBox, and TreeView controls. Its used to add, insert, and remove columns. In controls used to list text and values, the Collection helps you design and stylize columns. You can stylize text and the control's background color, set your alignments, set the caption, enable or disable "TrimSpaces," and allow

the columns to be extended, formatted, and visible (or hidden). In the ComboBox or Layout controls , these properties are not present.
In the Layout control (used for containing other controls) the Manage Columns Collection is designed to add, insert, and remove columns and assign column names and SizeModes (values that set how the object in the cell of the Layout control will be sized), and whether the column is hidden or visible. For more details, see *How to use the Layout Control*.

The **Manage Displays Collection** property is used to set the values a specific display in a Form control. This way, your form can be linked to a display for different languages but still be used for the same app.

The **Manage Icons Collection** property group is used to add or remove icons, and if in a list, set the list order.

The **Manage Rows Collection** property. See Manage Columns Collection.

The **Margins** property is used to pad the spacing between the rows or images of the displayed data.

The **MaskInput** property (available only in the TextBox control) is used to mask the input with asterisks if the value is True. The default is set to False.

The **MaxWidth** property is used to set the maximum space allowed for the control. For example, the Radio Button maximum width is the widest space allowed for the button and its text label.

The **MenuIcon** property on the Form is used to provide a background image.

The **Multiline** property was used in older versions for long strings in a control under the "TextOptions" property. It has been deprecated in 5.2.3.

The **(Name)** property is the internal name / identifier of a control/prompt. Tip: As a best practice, follow the Hungarian notation where textboxes are named 'txtPart' and list boxes are named 'lstParts' as examples. This way, when referring to them in the script, there is an inherent understanding of what types of data will be used for the prompt.

The **NormalizeText** property will trim the spaces from both sides of the displayed data or captions of the buttons or desktop icons.

The **NullText property** - was replaced by TextHints. It was used to help users know what should be entered, but is not retained as a data value for transactions.

The **Overflow** property specifies which way the remaining items will be displayed. If there are more items than will fit on the device's screen this option can be set to horizontal or vertical which means the user can swipe bottom-to-top or right-to-left to access the remaining data.

The **Padding** property sets in pixels, the amount of space between the left, top, right, and bottom edge of a control and other objects. The spacing remains constant even if the size of the control changes.

The **(PageNo)** property lists the order sequence of a page on a form.

The **PageMode** property is only available in the PanelList control. When set to True, it enables a user to swipe the screen horizontally to view more rows of data.

The **Password** property, for the data field portion of the prompt, sets the display of the text equal to asterisks (*) instead of clear text.

The **PenColor** property sets the color of a signature in the Signature control. In Mobile Themes Signature element, the pen color inherits the value from the ForeColor property.

The **Pressed property** changes the style and color of a control (i.e. Button, Button List, ComoboBox, DataGrid, Tab) when its pressed. This property is available for all pressable controls and list theme colors and a palette of configurable colors at the control and theme level.

The **PromptNo** property is the Prompt sequence value RFgen assigns to objects placed on a form or a page. If you move the location of a control or page, the PromptNo also updates, or you can edited it directly in the Application Designer Control Panel. If your script uses App.PromptNo VBA extension, it should return the PromptNo value for the specified control.

The **Required** property is used in prompts such as the Memo control where input is required before the user can continue to the next page, prompt or task in the application.

The **RowAltColor** and **RowSelector** (TreeView Control only) properties sets every other row to the color selected and enables users to select the row (True) as opposed to just viewing the content in each row.

The **ScaleMode** property is a Form Display property that specifies how the form and form elements will scale when the application is deployed the actual device. For more details, see Form Properties > Display: ScaleMode .

The **ScaleToFit** property increases or decreases the icon captions to fit inside buttons or desktop icons when the overall size of the button or desktop tile size is changed. This feature is enabled when its True and is disabled if set to False.

The **ScaleText** property automatically scales the text front to fit the space in a control and is available for these controls: Button, Checkbox, DateTime, Label, and RadioButton. True enables this feature, and False disables it. **ScaleText** property settings on individual controls take precedence over the **Configuration > Environment > Auto-Scale Text to Fit Controls** settings. If the **ScaleText** value is set to " (Default)" then the value which is set in Themes will be used for the control.

The **Scrollbars** can be enabled for horizontal scrolling, vertical scrolling, both, or none. If set to Automatic, RFgen calculates and displays the scrollbars for you. This property is for select controls only. See Mobile Themes > Scrollbar for more details.

The **Selection** property group is used stylize a selected item from a group of items or list in a parent control such as the ButtonList, CompboBox, DesktopIcons, ImageList, ListBox, or PanelList. The subproperties include BackGround, Border, Text, Transparency and Visible. To compare against the unselected item properties, see Items property topic.

The **ShowBorder** property will hide or show the border of an element. The values are True, False, or (Default). *(Default)* uses the property value set for the control (element) in Mobile Themes. *True* will display the border; *False* will hide it.

The **ShowInForm** is a child property of the Menu: Heading property. If the Heading property is set to True, then ShowInForm can be set (True/False) to suppress the menu's heading caption. **True** hides (suppresses) the Menu's own heading, and the Form's heading is displayed as the Menu's heading. **False** will enable the menu's heading and remain visible if the Menu control object is not obscuring the Form's header. If the menu object was set to the same dimensions of the form (and overlayed the header), then Form's heading will be used as the Menu's heading even though ShownInForm is set to false.

The **ShowLines** property will hide or show the lines between rows and columns. The options are (Default) which uses the theme properties, None for hiding all the lines, Horizontal for showing only the lines between rows, Vertical for showing only the lines between columns, and Both for showing lines between rows and columns.

The **SelColor** refers to the color of the selection bar shown in controls like the combo box or list box. The highlighted value is what will be chosen when the user presses the enter key.

The **Size** is the background height and width of the control in pixels. If AutoSize is used, it can influence the height and width values. In the ButtonList control, this sizes the buttonlist container-- not the items in the list.

The **Sorted** This was replaced by SortMode in RFgen 5.2.

The **SortMode** property is used in list controls and will sort content in accordance to the value selected. The values are "(Default)", None, Ascending, or Descending.

The **Source** property (the HostScreen control only) selects an executable to be emulated within the Host Screen control.

The **StretchImage** is used to either shape an image to the size of the control or allow the image to be its natural size whether it fits in the control or not.

The **Style** property in the ButtonList control specifies whether the items in the ButtonList are to be presented as a squarish buttons or icons (text and icon but no button as a container). In the ComboBox control, the Style property provides an option to display the drop down list as full screen or display the drop down menu in its standard size.

The **SystemIcons** property group is used to associate RFgen-supplied actions (Call Event) with a customized icon or a RFgen-supplied icon. For example, if you wanted to include a Cancel operation in your application, use the functions/properties in this group to select the Cancel operation and also associate it with an icon. When you are done adding icons, a value in brackets { } shows the total number of icons associated with the control. For specific details on how to link a customized icon to a Action, see To link SystemIcons with system operations.

The **TabNo** property is cursor/prompt sequence number for controls that can accept the focus. This property is not the same as the SeqNo property. For example, a Label control cannot accept focus but have a SeqNo but will not a TabNo. Since a TextBox can have focus and have a SeqNo and TabNo. You can edit a TabNo so to force where the cursor goes after a specific object loses focus.

To view the list of controls/objects' tab numbers, click on the  Sort icon or see Sort Controls.

The **TagLine** properties are used to stylize the text that appears above a ComboBox, TextBox, Memo, or SpinEdit control when a **TextHint** is used as a visual clue on what type of data needs to be entered into the prompt/field. (For example, you can have "3000" appear as a tagline above TextBox.) The TagLine properties include the FontSize, ForeColor, and Separation -- the number of pixels of space between the text and the top of the box.

The **Text** property group is used in the ButtonList Control and specifies the color of the font, size of the font, style of the font, and position via the margin values. Whether the values here are overridden by other property settings in the control will depend on how deep this property is nested.

The **TextHint: DisplayMode** property group is used to set the position of the TextHint. The optons are: (Default), Both, External, Internal, Toggle and WaterMark. External - adds a tag line above an empty contro t then disappears after the entry is make. Internal displays a tagline inside the prompt but leaves space for the user to enter the prompt then goes away after a value has been entered. Watermark shows your texth-int inside the prompt, and goes away when the user overwrites an entry; if no entry is make it stays there. "Both" will add the tiny TextHint above and inside the prompt.

After an entry is made in the control, you can use the **TextHint** properties to: a) Set a tagline to display above the control after the user entered data over a watermark; b) Set a tagline to continue to display above a control after an entry was made to a blank control; c) Have the tag line disappear after an entry was made; or d) not use a tag line at all (nothing displays before or after entry.)

The **TextID** property links a text (word or statement that is unique to the control) to a resource listed in the **Solution Explorer > Text Resources** table so that the value will be localized (translated) in the applic-ation at runtime. If the identifier is the TextId from the table in the Solution Explorer > Text Resources folder, the corresponding string in Translate Text column will display as the text string for the Caption, Head-ing, Error Message, or NullText. If the value is preceeded with a "%", it links user inputs from the source TextBox to a destination Label at runtime.

For more details see Text Resources or To apply translations across all applications.

The **TextOptions** property sets whether the text will be on a Single Line or Multiple Lines. Expand the prop-erty group to set how the text will wrap if Multi-Line is selected. In 5.2.4.x and higher, length of the Multi-Line text will be set under the Size > Width property. When the Muli-Line property is checked, and the width is used to set the wrapped text's width, the Auto Size is disabled.

The **TextOptions** property group in the ButtonList, IconList, ImageList, and Menu. In this group, Tex-tOption sets the values for the FontSize and FontStyle. The sizing of the icon names (captions) and icons in a button/tile are affected by the margin, **ScalePoint**, and **ScaleText**. The **Margin** is the distance between text and the edge of the icon container. **ScalePoints** reduces the size of the text by the factor supplied. For example **ScalePoint** 0 or 1 sets the text default size; 2 makes the text 2 times smaller then the icon size. **ScaleText**, if set to True enables scaling of text; False prevents it from being scaled (reduced).

The **Theme** property changes the border of the title bar area to one of several hardcoded styles.

The [**Title**] property in Themes > Dialog is used stylize in the title of a Dialog box (pop up messages dialog box).

The **ToolTip** property is a value that can be selected from any control that supports a BorderStyle.

The **Transparency** property sets the level of transparency of a child control and bases its coloring on the parent (i.e.Form, Page1, Panel). "(Default)" uses the level of transparency set in the corresponding Mobile Theme element. "None" will use the item's BackGround color. A percentage value (5 %, 10% etc), sets the transparency level.

The **UpdateColumn** property. See ListOptions > MultiSelect > UpdateColumn.

The **UseMenuTheme** property will override the local properties and apply the default theme properties for the menu control.

The **ValidationTable** property presents a list of downloaded tables that can be used to verify that the data entered already exists in this table and the Validation Field. The two properties must be used together.

The **ValidationField** property presents a list of table fields specified by the Validation Table property. This is the reference field to determine if the data entered in the prompt already exists. If it does not, the Error Message property will be used to warn the user.

The **Visible** property, set to True, makes a prompt or element visible, while setting it to False makes it invisible. Even though the prompt may be invisible, the GotFocus, OnEnter and Lost Focus events will still be executed for this prompt if the focus automatically shifts from a prompt before this prompt to one after this prompt.

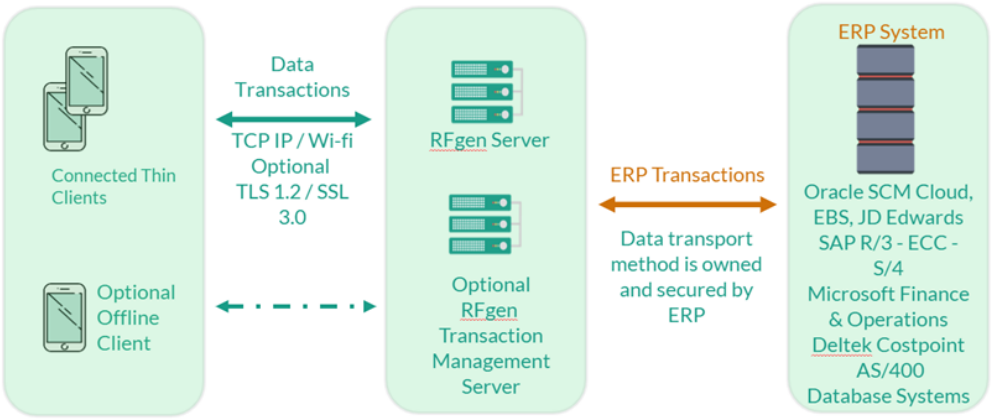
The **WaterMark** property group is used with the **TextHint** property group. The WaterMark is a type of caption that displays temporarily inside a ComboBox, TextBox, Memo and SpinEdit control so the user has a clue what to enter into the prompt/field. Once the user enters data, the watermark caption is overwritten. The WaterMark properties include the ForeColor of the watermark text, and Transparency (darkness or opacity of the text).

The **ZOrder** was obsoleted in 5.1.

The **Design Mode** (which used to be under Display tab) has to **Configuration > Desktop Preferences** in version 5.1.

RFgen Client Software

How it works



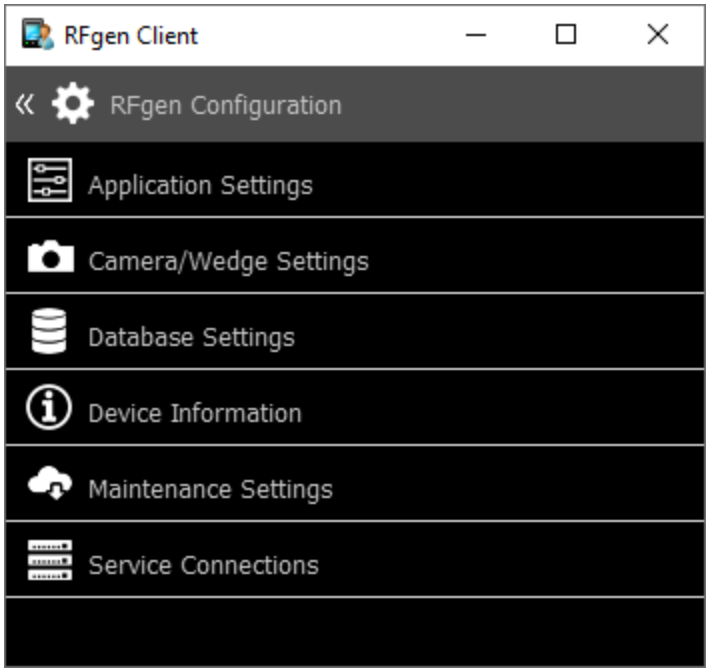
RFgen provides the RFgen client software to enable mobile devices to communicate with the RFgen server. The client software is available from the RFgen Download site, from Google Play, or the Apple App store.

Once the communication between the server and client is established, the RFgen server will deploy the mobile applications that enables your mobile device to perform warehouse tasks.

With the RFgen client software installed, the mobile device is typically set to operate as a **thin client**. With a thin client, changes are conveyed from the device to the RFgen server and from the RFgen server to the ERP system. For example, if you were changing the location of an item from bin 0001 to bin 4560, the change is conveyed through the application running on the device which is then passed to the RFgen server and from the RFgen server to the ERP.

The other operational mode is offline where you can use the device to conduct tasks while disconnected from the network. This allows you to work offline, but the changes are stored on the device until it reconnects with the RFgen server which in turn conveys the changes to the ERP. This mode is also called "Batch" because its taking batches of jobs (changes within a type of workflow) and saving them up until the information can be passed back to the ERP.

Client Configuration Settings



RFgen Configuration is a collection of status and settings that are used to change how a mobile client (Android, iOS, Windows Desktop, or Windows Mobile/CE) starts up, receives updates, and displays your application screens.

For more information, see the specific topics on Application Settings, Camera Settings, Database Settings, Device Information, Maintenance Settings, or Service Connections.

Thin Client Overview

While in thin client mode, the user interacts with a session running on the server. Since all the processing takes place on the server, the mobile device cannot be a point of failure or lose data.

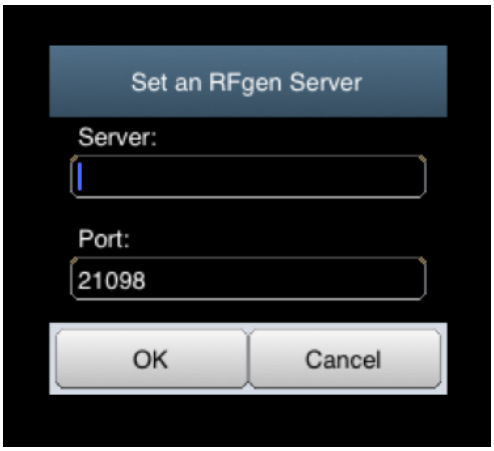
If the wireless device goes out of range of the network, the mobile device screen will appear to stop since the server cannot order the screen to refresh. The client on the mobile device will continue attempts at reconnecting and will then resend the last piece of data entered. RFgen has added a "Guaranteed Packet Delivery" system to the protocol to ensure no loss of data and an always-synced application.

The advantage to the thin client is real time updates to backend systems as well as complete validation data available to ensure the collected data is as accurate as possible. The disadvantage is the need for a wired, wireless, or cellular connection available while collecting data. This client does not require any authorization process.

Connect and Deploy to RFgen Clients

Once you have created your mobile profiles in the Mobile Development Studio, and installed RFgen software to target devices, you enter the RFgen Server hostname or IP address in the RFgen screen. This will request

the server to communicate with the client and download the mobile profile of your choosing.



Note: If you do not want to download from the server over a network and your device is Windows Desktop, CE or Android, click here for alternate methods of installing a mobile profile.

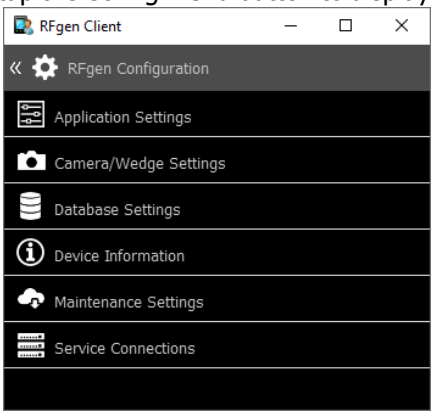
To install a Batch Client license

If you are using a mobile client in Batch (offline mode or batch mode), a batch license must be installed to the device and then activated/authorized. This process only applies to devices that will be running in batch/offline mode.

In the process the server will download a batch licenses (certificate) to the client. Before proceeding, ensure the server has licenses available to distribute to a batch client.

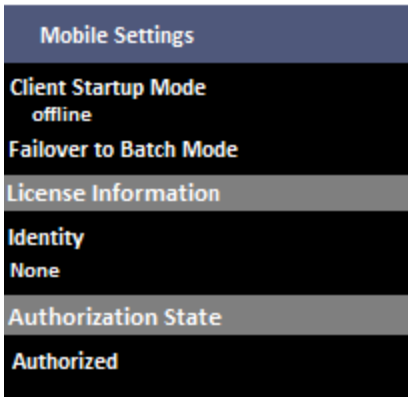
Android or iOS: Once the mobile profile has been deployed / installed to the client, you activate the license via the "Activate" command from the menu strip.

Steps

1. Connect to the RFgen server.
2. If this is the first time the RFgen client is communicating with the server (i.e. this is a new install of the RFgen mobile client software) enter the server name or its designation.
3. When your mobile application Login displays,
   tap the Config menu button to display the RFgen Configuration menu.
4. 

5. Tap **Maintenance Settings**, then tap **Activate Batch License**. This button is only present when a certificate is present for a download.

You can also verify if the device is authorized by reviewing the device's RFgen Configuration > Mobile Settings > License Information: Authorization State.



The connection may require the RFgen Administrator to authorize the device connection from the server.

Installing the RFgen Windows CE Client

The **RFgen Mobile Client** executable (**CE Client.exe**) installs software enables communicate between a Windows CE device and the RFgen server. This software is required before you can deploy client profiles (mobile apps, menus, resources etc) to the Win CE device from the server.

Profiles are created in the Mobile Development Studio and transferred to the client inside a CAB file. (In other words the CAB file is the format used to package a Profile created in RFgen.)

Installing to the RFgen Server

The **RFgen Client** software (which you download from the RFgen Portal) installs CE-version specific software that enables the client to communicate with the server, and it also provides files used by the Mobile Development Studio to build CAB files.

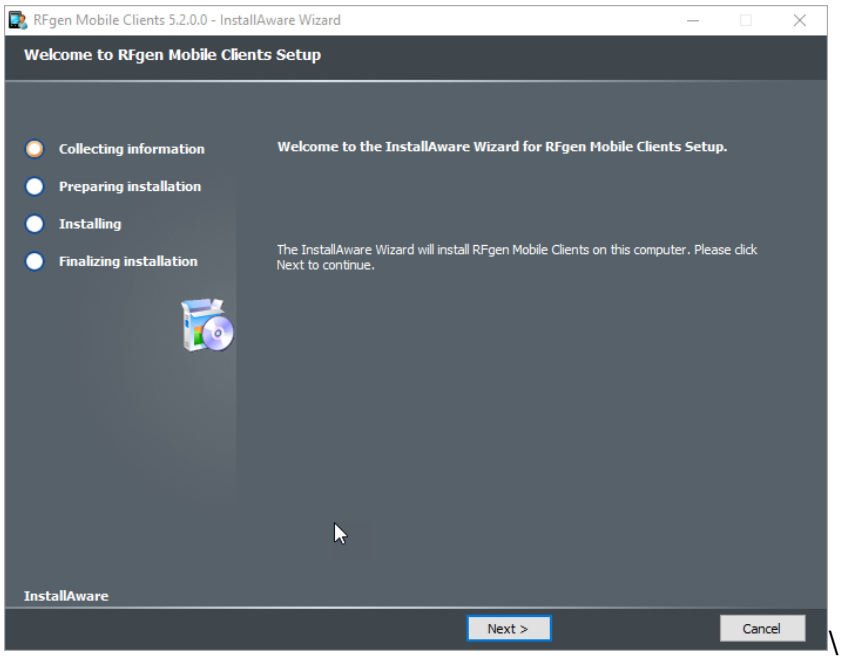
Therefore, before you can deploy any solution to a Windows CE device, you'll need to install the RFgen Client software on the same server where your **Mobile Development Studio** resides.

Note: By default, this package installs to c:\Program Files (x86)\RFgen52 folder. **If you installed the 64-bit version of the RFgen 5.2 Mobile Development Studio or installed the 64-bit version of the RFgen 5.2 Mobile Unity Platform server, make sure the RFgen Client installs to the same folder (i.e C:\Program Files\RFgen52).** By having them in the same RFgen folder location, the Mobile Development Studio will be able to locate the files needed when you are ready to build or deploy files to the client.

RFgen Client Install Process

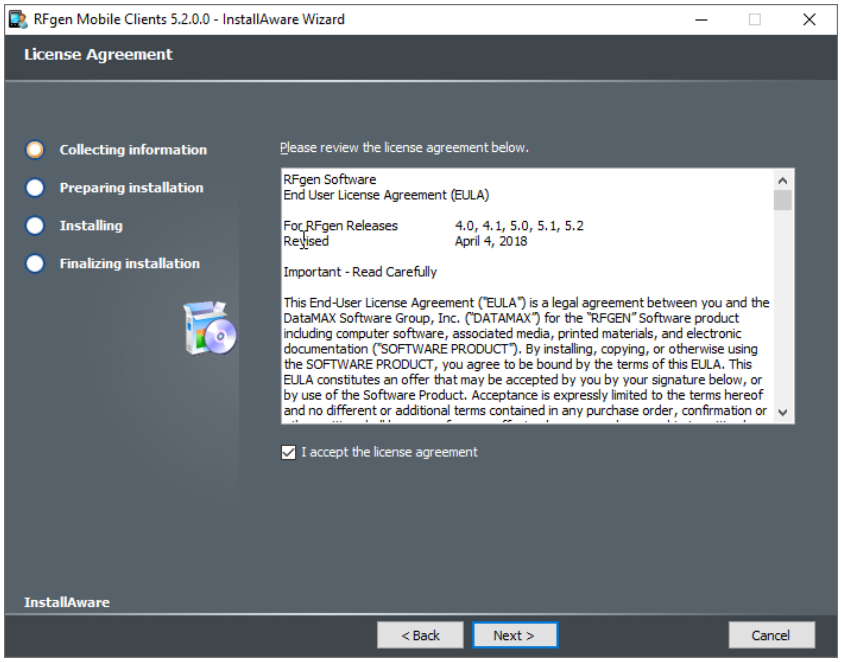
1. After you downloaded the RFgen Client package, launch it as you would any other Windows application.

The **Welcome** screen displays.



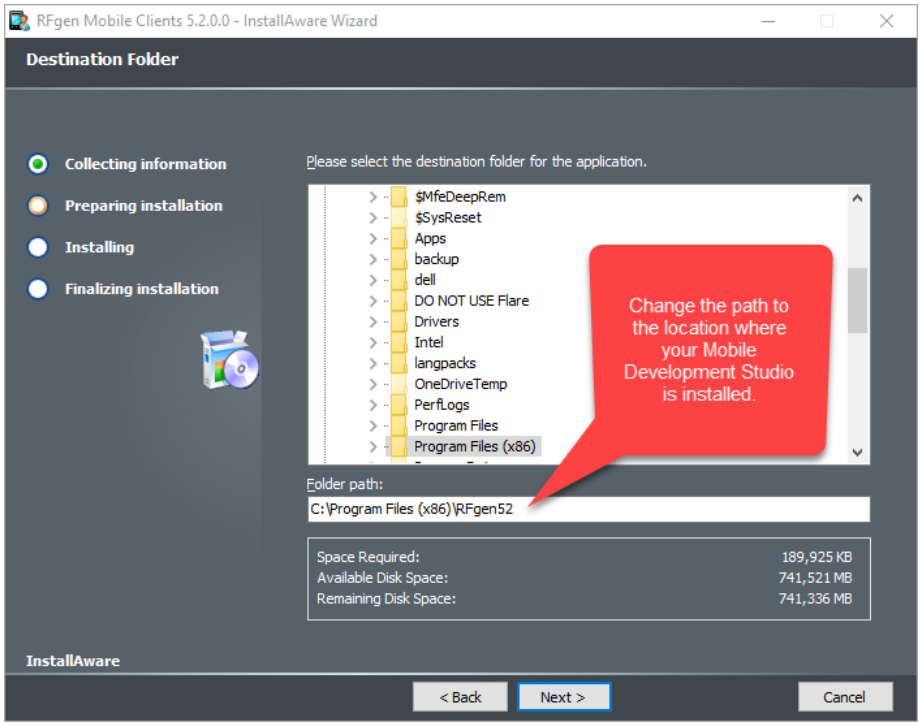
2. If you need to exit the process, click Cancel. Otherwise, click **Next** to continue the install process.

3. The **License Agreement** screen displays.



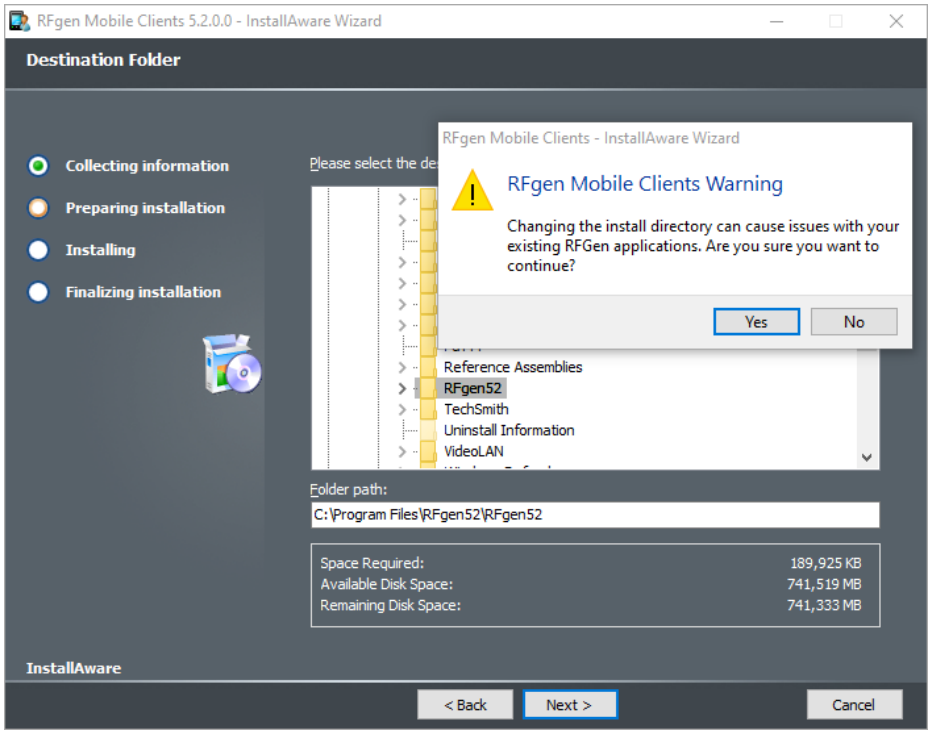
4. Click the checkbox and then click **Next** to continue.

5. The **Destination Folder** screen displays.



6. Change the path to the location where your Mobile Development Studio resides. For example, if you installed the 64-bit version of the Mobile Unity Platform server, its default path is `C:\Program Files\RF-gen52`.

7. The **RFgen Mobile Clients Warning** screen displays. Click Yes to continue; No if you want to go back and change the path again.

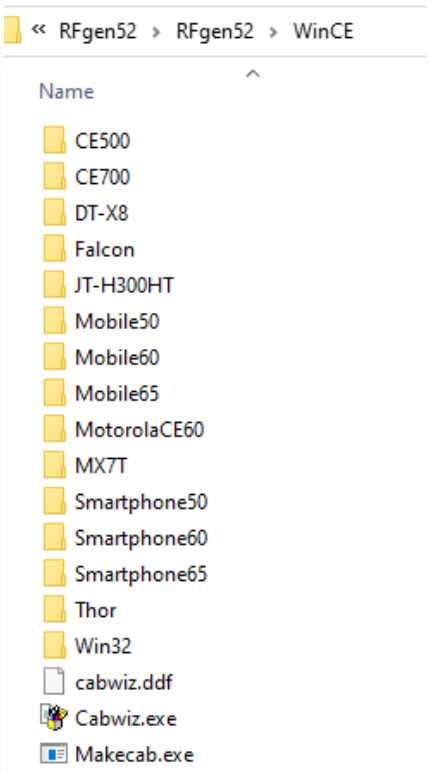


8. Click **Next** to continue.

9. The **Ready to Install** screen displays.

10. Click **Next** to continue. When its done, click the **Finish** button.

11. Note that the RFgen 5.2 folder under the parent RFgen 52 folder contains the various Windows CE device files and Cabwiz.exe and Makcab.exe files.



12. Now you are ready to prepare your CAB files for Windows CE. For more details, see Solution Deployment and To Create CAB files.

Activating Mobile (Batch) Clients

On Mobile Clients, the RFgen administrator can deploy authorization certificates to RFgen clients on Android or iOS systems through the server **Device Management > Device Authorizations** feature.

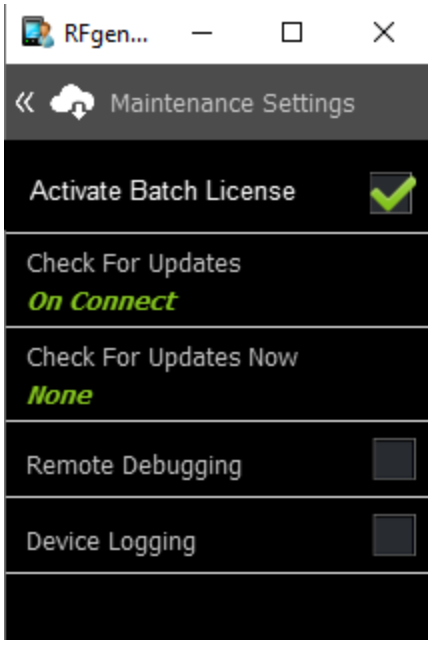
Deployment of the certificate occurs after the device has received a Mobile Profile and the RFgen Administrator has setup the certificate to be deployed to a specific device (from the server).

To activate / authorize the license from the device, launch the device **Menu Strip > Options** and tap the **Activate** button should display.

EXAMPLE

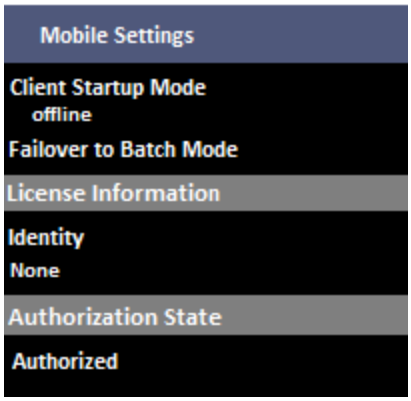
This is an example of how you can activate the device. Your own application and menus may be different as these can be customized by the developer.

1. Connect to the RFgen server.
2. When your mobile application Login displays, click on the configuration menu button to display the RFgen Configuration menu.
3. Tap the Maintenance Settings and check the box for Activate Batch License.This button is only present when the certificate is present for a download.



4. After you check the box and

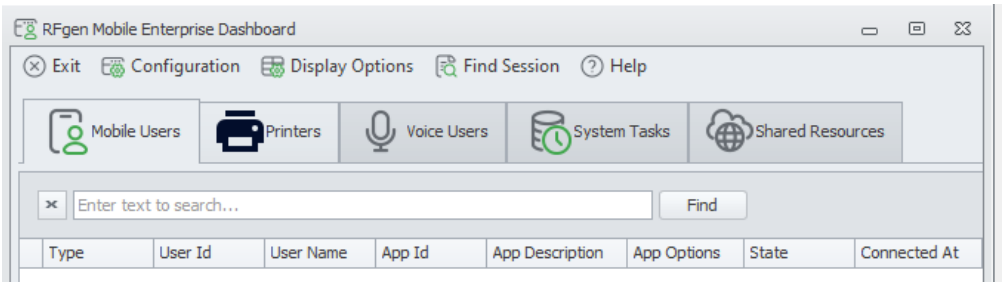
You can also verify if the device is authorized by reviewing the device's **RFgen Configuration > Mobile Settings > License** Information: Authorization State.



Mobile Enterprise Dashboard

RFgen provides a Windows-based dashboard to view and manage mobile device sessions when the client is connected. This dashboard automatically displays for the server its installed to.

However, if you are working with more than one RFgen server or the print server, you can select which server group you want to monitor activities from.



The **Mobile Users** tab allows managers/admins to monitor the activities of users on connected RFgen Client(s). You can login on the user's device sessions, see what they are seeing to help them out.

The **Printers** tab displays the status of your connected label printers in this dashboard if you have the **Secure Label** enabled (licensed) and your server setup to accept connections from the print server.

The following tabs are for legacy use. **Voice Users** tab, **System Tasks** , **Shared Resources**.

Mobile User Features

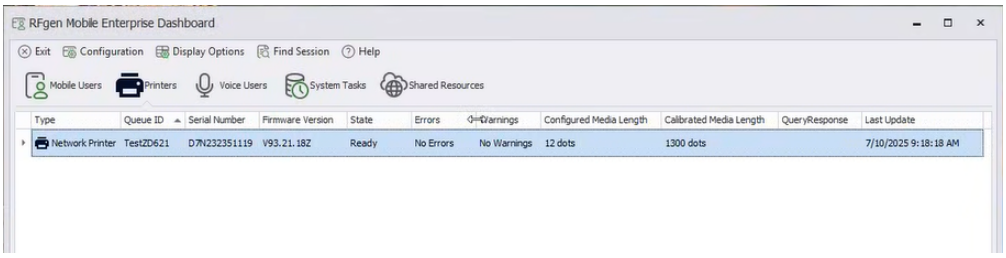
As devices log in, they are displayed in the dashboard.

This capability may be used as a system resource. For example, a hardwired (networked) user who connects to the system will receive the same screen that appears on the screens of remote devices.

For more information on this tab, see To monitor and interact with an active client.

If you have the Secure Label feature enabled (licensed) you can also view your connected printers in this dashboard.

Printer Features



For Secure Label users, this displays a list of the printers connected to your RFgen server/print server. This information is updated automatically every 30 seconds.

If you do not see your printer, it could mean the printer had not yet connected to the server. If the printer was connected but then disconnected, this status would be logged.

The following information displays for:

Type - Lists as the Network Printer, Bluetooth, or WebLink.

Queue ID - Which queue is in use.

Serial Number - The serial number of the printer if its available.

Firmware Version - The Firmware Version of the printer if its available.

State - Ready means its ready to print. The other states can be Disconnect or Offline. The information here may vary by the print model/maker.

Errors - Describes what errors were detected. The information here may vary by the print model/maker.

Warnings - List if any Warnings were detected. The information here may vary by the print model/maker.

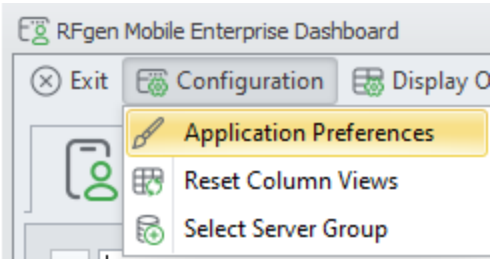
QueryResponse - This field is currently reserved.

Configured Media Length - The length of the media entered in the Mobile Dev Studio > Stock Media node. This measurement is in DPIs.

Last Update - The data and time of the last update.

Calibrated Media Length - The self-assessed length of the media detected by the printer in DPIs. The printer uses this to help align the label so it starts printing where it needs to start the roll.

To Configure Application Preferences

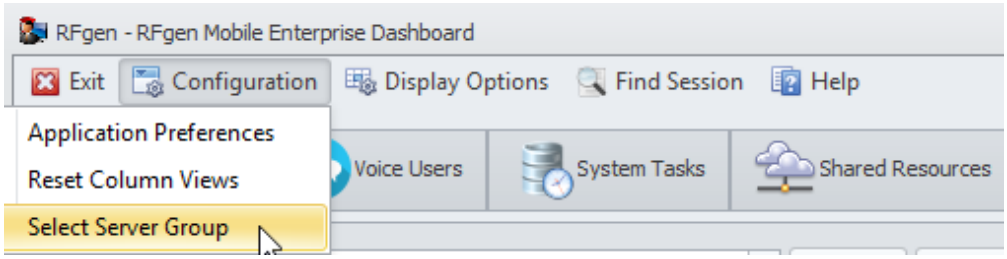


You can change the language, appearance, and default Local of the Mobile Applications:

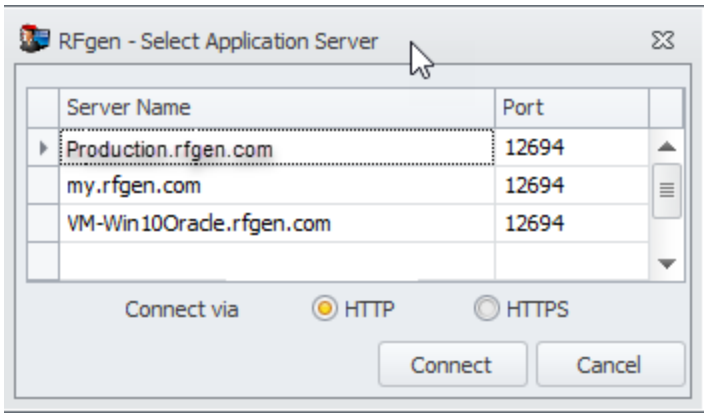
1. From the RFgen Mobile Enterprise Dashboard, click on Configuration > Application Preferences.
2. Select the Application Theme from the drop down menu.
3. Click **OK**. The screen changes immediately to the chosen theme.

To Configure a Server Group

1. From the RFgen Mobile Enterprise Dashboard, select: **Configuration > Select Server Group**.



2. The **Select Application Server** screen displays.

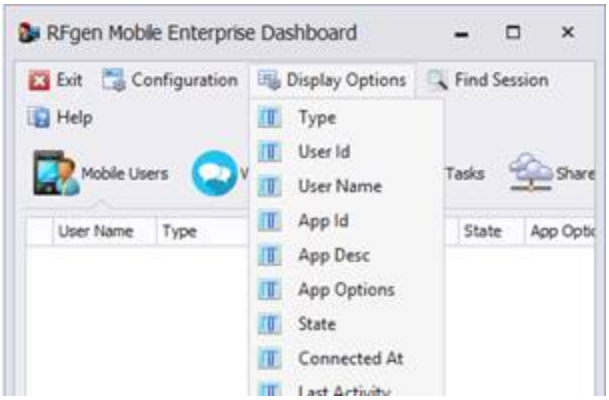


Select the server as the source for viewing information in the dashboard.

3. Select the connection type (HTTP versus HTTPS).
4. Enter your credentials if required to access the server.
   Press Connect.
5. The server/server group you connected to displays in the lower left corner of the dashboard.

Display Options

The Display Options menu lists the column headings that are used in the Mobile Users, Voice Users, System Tasks, and Shared Resources tabs. To change the column headings, select **Display Options** menu and then right-click on the column heading you want to add or change. The list below describes each header type.



The Display Options details are:

Type – Type of connection. For example, a Windows Desktop connection will show up as a Mobile Device.

User ID – the user ID or operator who logged in.

User Name – The full name of the logged in user

App Id – shows the menu or application screen name currently being viewed by the user.

App Desc – the description of the current form.

App Options – any passed in parameters to the current form from the menu

State – shows either Disabled or Active depending on the suspend status of the client connection.

Connected At – shows when the connection was established.

Last Activity – shows when the very last keystroke was made by the user.

Server Name – Is the name you assign the RFgen server or its IP.

Task Id – is the process identifier of the client session executable that can be located in the processes list of the Task Manager.

IP Address – assigned IP address of the device.

GUID – This will be a GUID identifying graphical devices since in some environments the IP address alone is not enough to uniquely identify a client session.

Platform – Describes the platform of the RFgen Client, which can be: Windows Desktop, Windows CE or Mobile, Android, or iOS client.

Size – the size of the screen display used by the client's application.

Theme – the mobile theme that is used by the client's application.

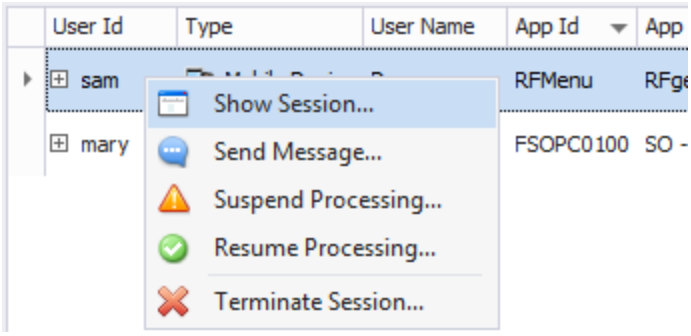
Locale – The Microsoft Locale ID value used by the client's application. For example, 1033 is English - United States.

RTL – Right-to-Left (versus Left-to-Right) setting – which is the orientation of the application for the locale of the client. For example, English is read from right to left.

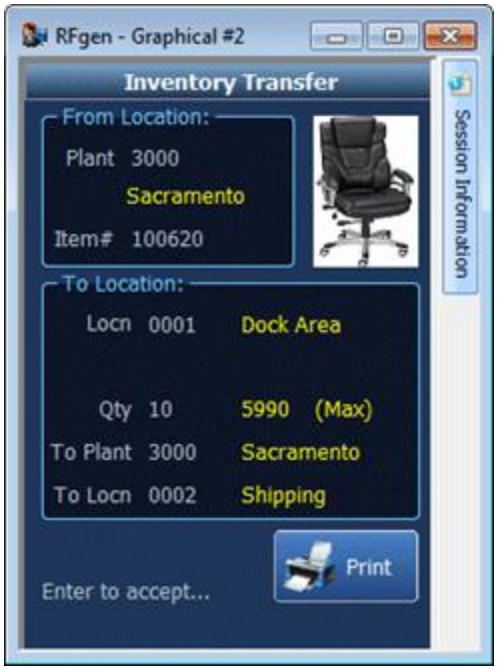
To Monitor and Interact with an Active Client

From the RFgen Mobile Enterprise Dashboard, select the **Mobile Users** tab to oversee and shadow (view the user's session on your system).

1. To monitor or interact with an active client session, simply right-click on the row of the device show session you want to monitor. A selection menu will appear.



2. Click on **Show Session**… The selected client device screen window will appear on your screen.

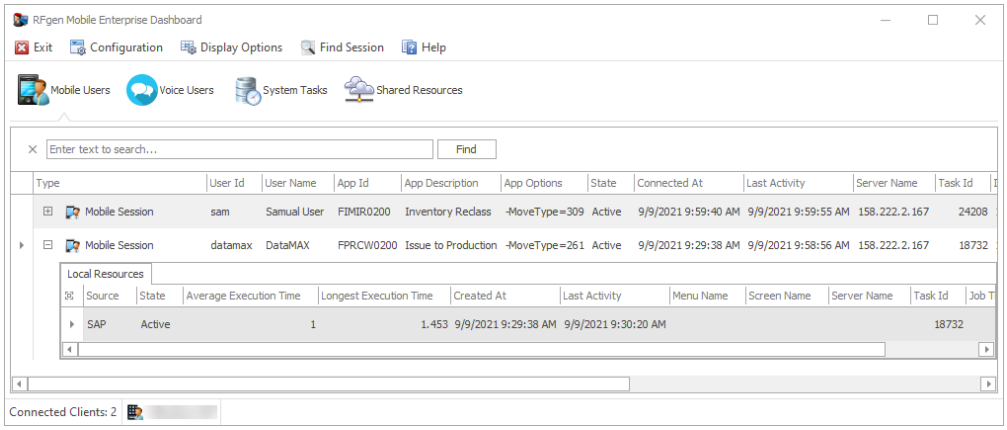


3. While the session window is open, all activity for the device will appear in the window.

   To take control of the session, click inside the screen display area and interact with the prompts. The same screen will appear on your screen. At this point the user on the client will see your actions.

   To end your session, click on the "X" in upper right corner to end your remote session. The client session will continue to run unless you used the "Suspend Processing…" or "Terminate Session" commands from the right click menu.

Mobile Enterprise Dashboard Menu



Configuration has two settings: *Application Preferences* and *Select Server Group*. The *Application Preferences* allows you to change the Windows Theme for the dashboard.

Select Server Group stores discovered RFgen servers. Once a connection is established, the selected server will display at the bottom of the Dashboard. If the connection is invalid, a red (-) i con will appear.

Connected Clients- The total number clients connected to the server displays when you have clients connected. If no clients are connected the total is 0.

Display Options lists the column headers used to view user connection information in the Dashboard area.

Find Session allows you to find a specific session when you many sessions going on in the dashboard. For a description of each item in this menu, refer to **To Configure Your Views**.

Help menu allows you to access the topics from the RFgen Manual, obtain information on how to access Support, and view version and platform information about the Dashboard.

To Configure Your Views

Select the icon for which you want to design a view. For example, select the **Mobile Users** icon.

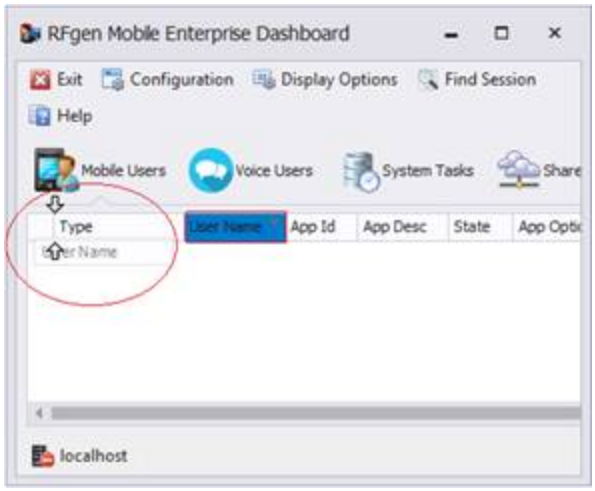
In the display panel area, you can choose to:

a. Rearrange the order of columns – by selecting then dragging it to its new location

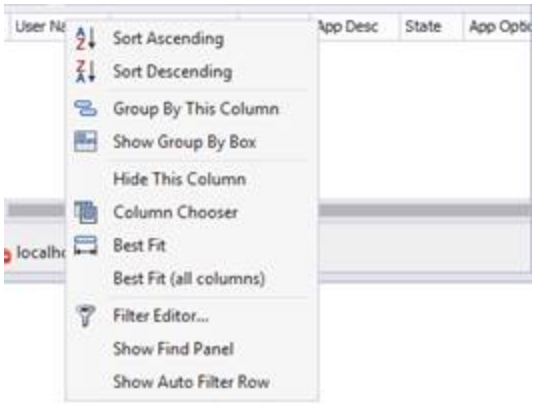
b. Hide a column – by selecting this from the Right-Click Edit menu

c. Add a column from the **Display Options** menu

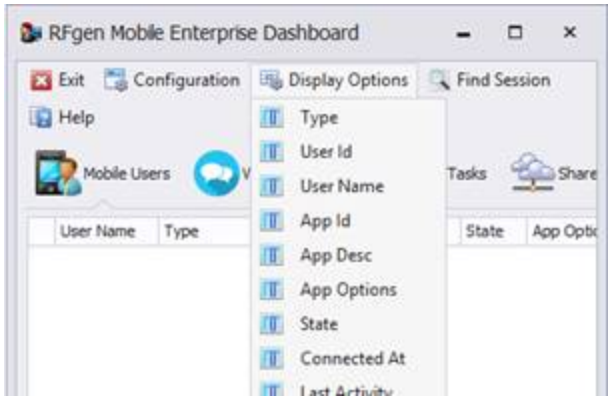
d. Sort and filter columns using the options from the Right-click menu.



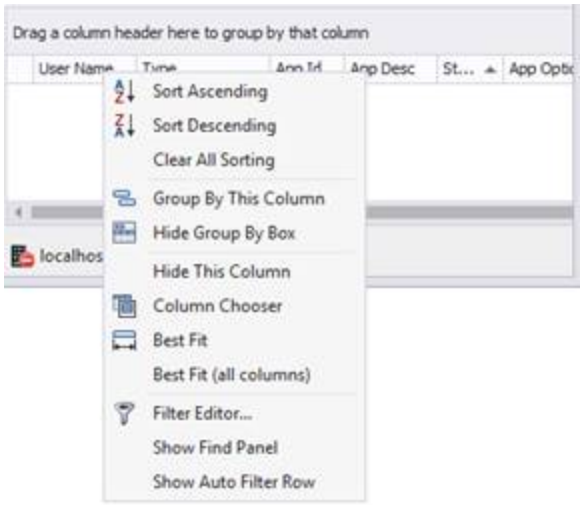
Example a. Rearranging columns



Example b: Hide or Show a Column

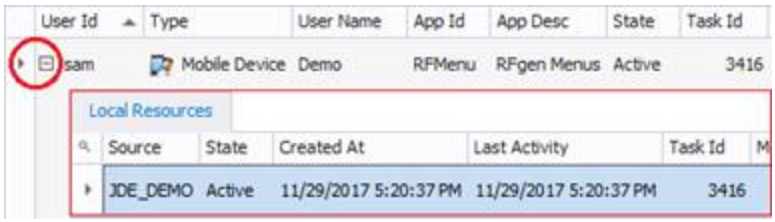


Example c: Add a column from Display Options menu



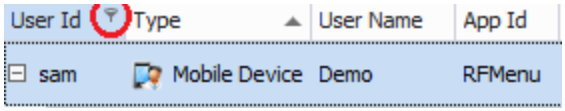
Queue sessions are shown for each queue that is setup. Graphical and Character sessions are displayed for each connected user and represent the type of device they are using.

This view appears when the dashboard is first started. As data entry devices log in, each appears in its own row.

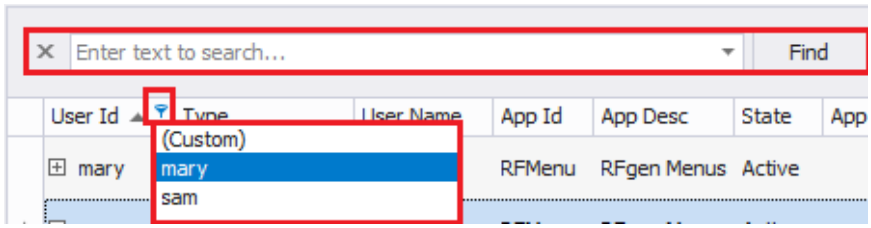


The pointer indicates which row is selected.

The "+" icon allows you to expand the details for the selected row (logged-in device). The "-" will hide the details for the selected row.



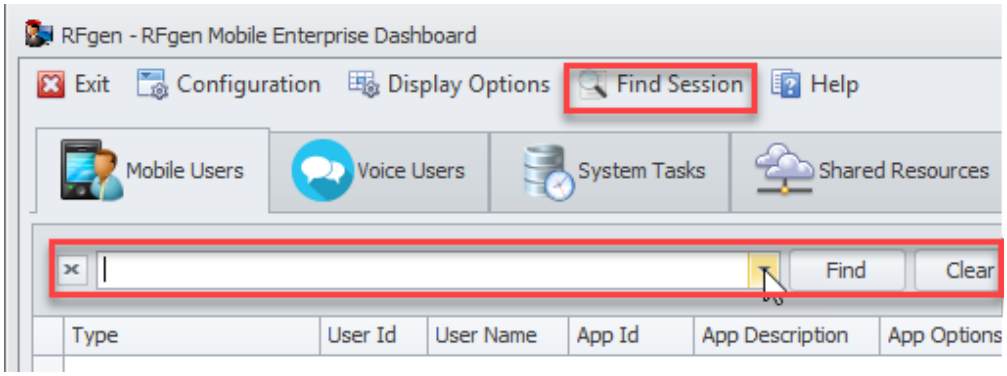
To bring up the filter icon, click on the background of a column header.



You can also view a summary of the values for a given column by clicking on the filter icon at the top of a column heading.

See Display Options for more details on these headings.

To Find a Session



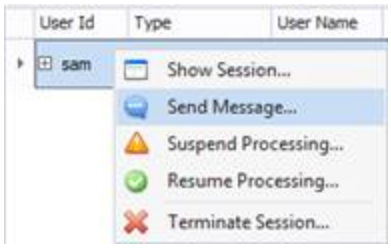
If you have hundreds of sessions running, use the Find Sessions tool to bring up the session.

1. From the RFgen Mobile Enterprise Dashboard, click on **Find Session**.
2. Enter or select the session from the search box.
3. Click **Find**.

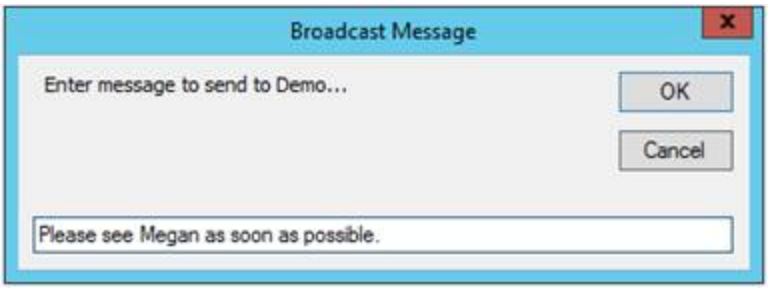
To Broadcast a Message

Send Message allows you to send a message to the device user from the Mobile Enterprise Dashboard.

1. To send a message to a specific client, right-click the row of a device show session and select **Send Message...** from the menu.

2. 

   A message box will display. Enter your message and click **OK**.

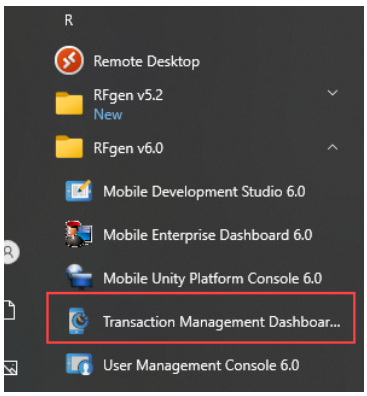
3. 

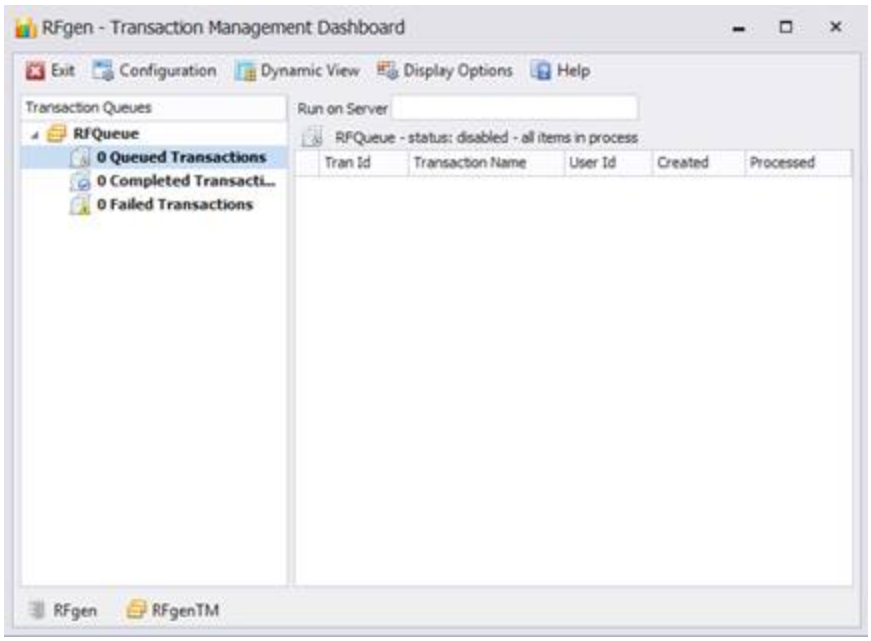
The client will get a pop-up message on their screen.

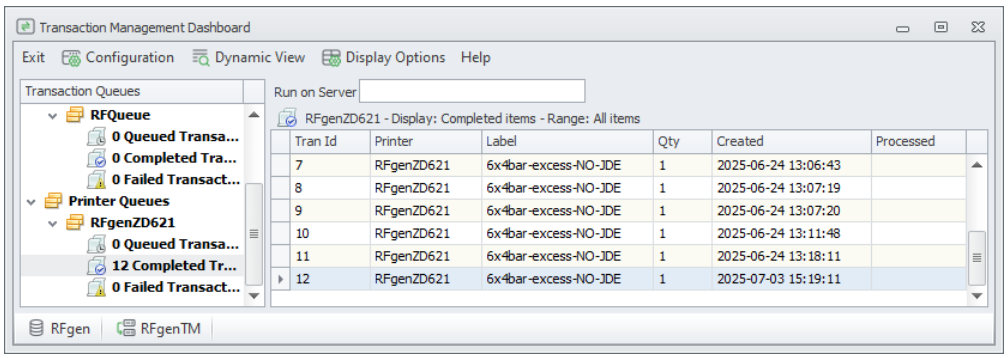
Transaction Management Server and Dashboard

RFgen provides a dashboard to view and manage the RFgen Transactions if you have this feature configured.

If you have Secure Label enabled, the TMD can also be used to track and manage printer queues.







The RFgen Transaction Management (TM) server is used to manage queues and queue the processing of transactions.

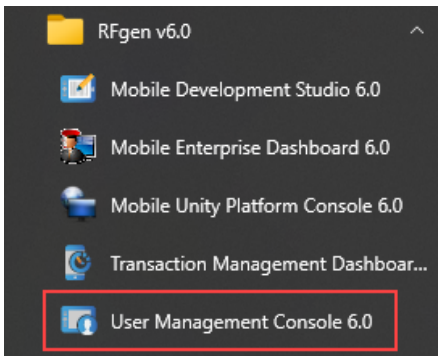
You use the TM Dashboard to: View the queues, control how individual queues are started or stopped, Set how queues are completed. If a transactions fails, you can use the Dashboard to edit and resubmit the queued transaction.

These three types of logs are available: "In Process" transactions are data collection entries waiting to be posted to the host application (typically because the host is offline or not available, or the batch client is off-line, and cannot connect and sync up with the host server); "Completed" transactions and "Rejected" transactions may also be displayed. Transactions may be edited, reposted, marked as completed or deleted by means of the right-click menu option from the desired record.

Before you can view data in the TM dashboard, the Transaction Management database must be configured.

For more deails, refer to the topic "Configuring Transaction Management DB Connection".

User Management Console

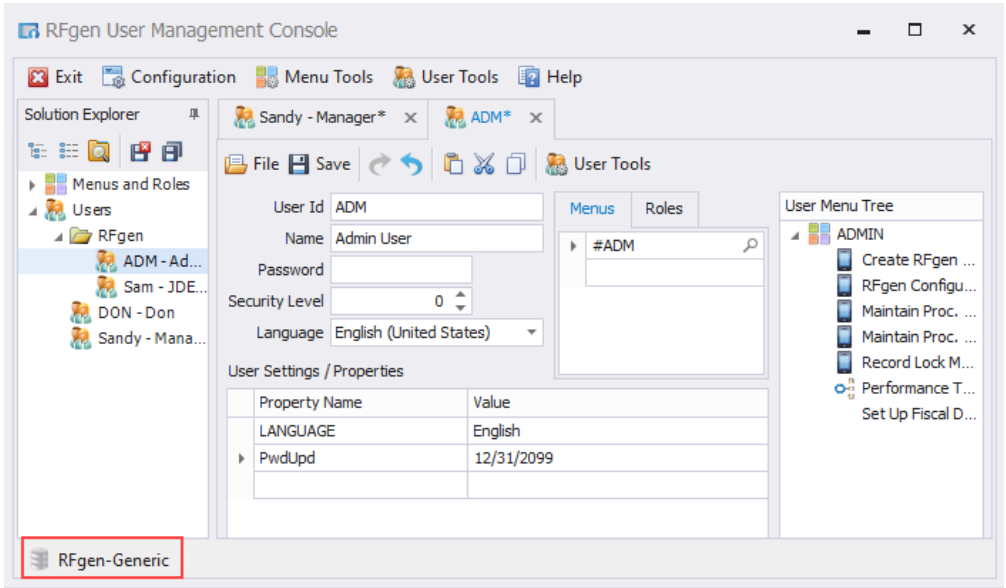


RFgen provides two types of User Management Consoles -- the Windows Console and an Web Console.

This topic provides details on the User Management Windows Console. For details on the browser User Management Console, refer to the Web Console.

Note that users existing RF users do not have access to the Web console. The default users Web Admin has access to the Web Console and ability to grant access to other users so they can access the Web Console.

The **User Management Console** (UMC) is a separate Windows interface that enables warehouse managers to add, remove and manage Mobile Application users to/from the RFgen Mobile Unity Platform (RFgen Server). To launch the console, click on the User Management Console program from your RFgen v5.2 or v6.0 Application Programs folder.



This console can also be used to assign menus and specific applications to each user without having to ask the RFgen Administrator for help. Changes made in the User Management Console are reflected in the Mobile Development Studio Users tree.

- If additional changes are required to applications, code, macros or resources, these changes can be performed by the RFgen Administrator through the **RFgen Mobile Development Studio**.
- Before you start, make sure your UMC is connected to the correct application database which provides the applications, menus and users that are configured in this console. In the image above the active database is RFgen-Generic. For information on Configuring the UMC database (or the connection to the source), see Configuring the User Management Console App DB (data source).

For an overview of how applications, menus/roles, and users are setup so to allow multiple users to have access to all menus/applications, or just a subset of menus/applications that is distributed to a group of users, see User Overview.

To view how the RFgen manager/developer can restrict access to specific apps, see To Limit User Access.

Index

I

J

L

M

P

R

S

T

U

V

W