FieldCall

80003

User's Guide
And
Installation/Setup

301 – 1555 Fir Street
White Rock BC V4B 4B6
Canada
http://www.mpowered.biz

What's New for version 80003

- New iOS app, and the Android app is completely updated.
- Android app follows the Webapp style introduced with iOS. Be sure to read through the setup section to get up-to-speed on changes if you have Android users.
- Added new grouping on main screen to hold daily COMPLETED or CANCELLED Calls.
- Added picture-holding capability, and on-hold pictures can now be viewed to allow visual confirmation before upload.
- Added support for 3 image sizes for photos: 800, 1200 and 1600. A new Setting "Image Size" defaults to 1200.
- Added new links to FieldBlade to make collaboration between these apps super-easy. Use FieldCall to get your daily list of Calls, and use FieldBlade to give you an expanded universe of possibilities. (Requires a FieldBlade licence.)
- Email notifications on Call count changes w/ new Setting "Calls count" set to Notify. Requires some server setup, see setup section in this document titled "Email notifications".
- Many other adds, improvements and glitch fixes: including improving AD auth to allow per-user.

Upgrading from a previous version? See the Upgrade Notes at the end of this document for helpful advice.

1. Using FieldCall

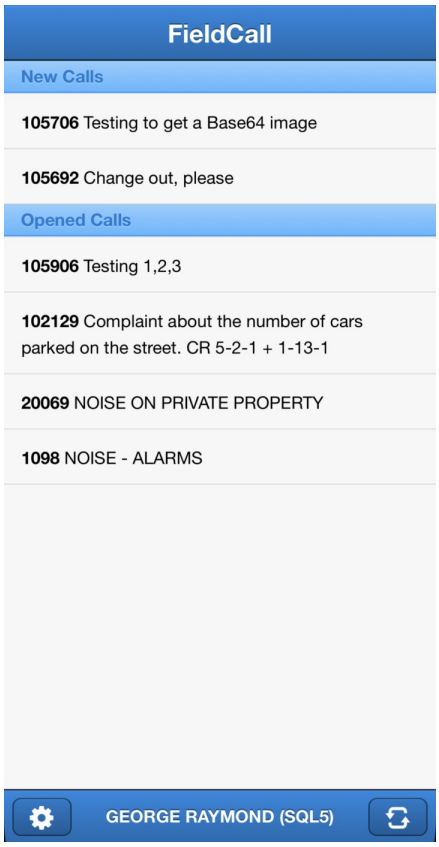
Note: This section assumes that you have successfully installed/upgraded and configured FieldCall. If you have not yet done so, follow the instructions in section 2, Installation/Upgrading.

Before we begin, let's lay a foundation for what FieldCall is designed to do, and what it is not meant to do. FieldCall is designed to replace the function of printing out workorders from Tempest and handwriting notes on them to be re-keyed into Tempest workflow later in the office. FieldCall allows you to do everything that you would do with your printed workorders, only it does it wirelessly – from anywhere you have a data transmission-capable signal on your device. FieldCall is NOT designed to be a complete replacement for Tempest Calls. For example, there is no functionality to create or re-issue Calls (you may want to look at Mpowered's FieldBlade product if you need full Calls functionality.)

There are three main screens in FieldCall - the main Calls screen, the Call Details screen, and the Call Workflow screen. That's it! Getting access to your assigned Calls, and adding workflow uses a simple navigation method used by most modern handheld devices, so you should be on your way in no time.

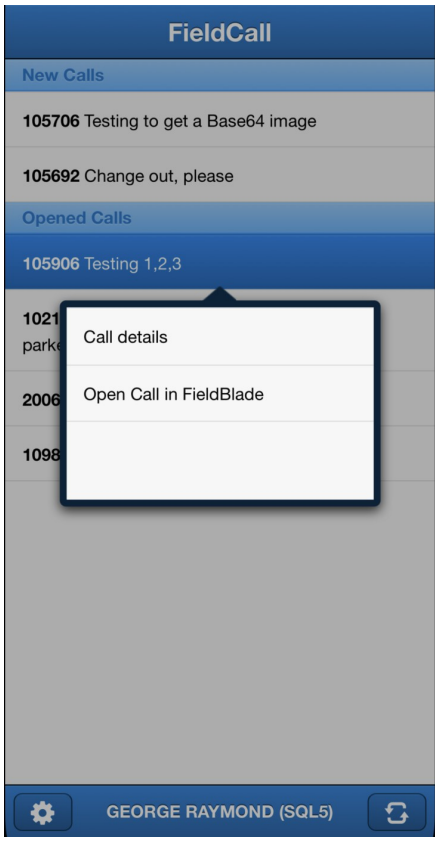
The main Calls screen

Below is the main Calls screen:



The main screen shows you a list of your assigned Calls. There are heading rows for “New Calls” and “Opened Calls” to divide up your Calls. Rows with Calls in them show the Call number and as much of the description as will fit on the screen.

To view a Call's details, you tap on a list item and then tap on "Call details" in the menu:

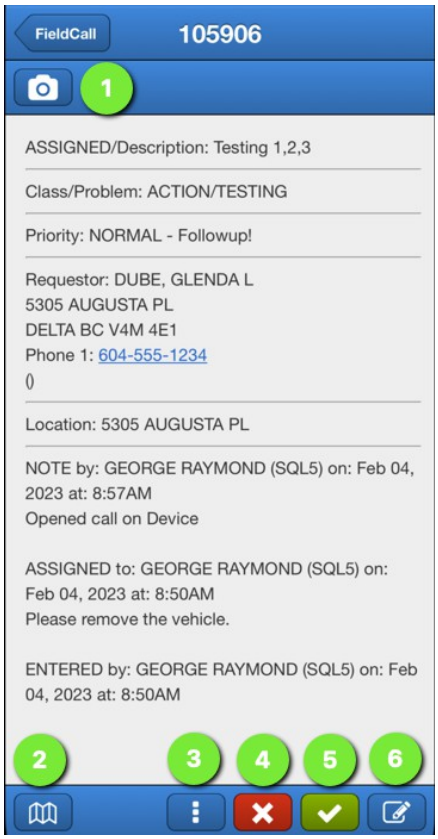


As a shortcut, you can double-tap the list item which will also load up the Call details.

If your City has a FieldBlade licence (and the server settings exist – see the setup section below), you can also use "Open Call in FieldBlade". This will open the Call in FieldBlade in a new Google Chrome (which must be installed on an iOS device) web browser page, and from there you can access all the functionality you know and love in FieldBlade. Using the FieldCall/FieldBlade link in this way exponentially improves your FieldBlade usefulness.

The Details screen

Selecting the “Call details” menu item will load a screen showing you all of the pertinent details for the Call:



You can scroll the Details screen up and down by swiping to view all of the information. The Call workflow (up to 25 of the most recent) is shown at the bottom of this screen in reverse chronological order. Viewing calls details on a Call which you have not previously opened on the device will automatically enter a NOTE workflow with the text “Opened call on Device”.

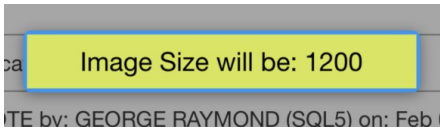
1. Allows you to take a photo (or select from the photo roll) that you want to upload as an attachment to the Call in Tempest.
2. The Map button shows if the Call is on a property, and opens a Google maps web page at that address.
3. Displays an action sheet.
4. Lets you to cancel the Call, adding CANCELLED workflow.
5. Lets you to complete the Call, adding COMPLETED workflow.
6. Lets you add a note to the Call, adding NOTE workflow.

Buttons 1, 3, 4, 5, and 6 are described in detail below.

1. The Photo button

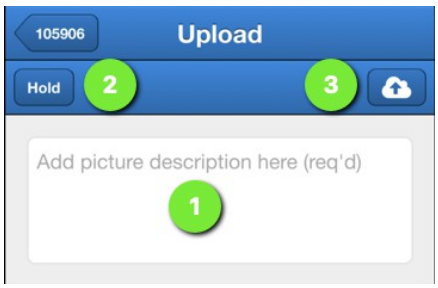
The Photo button allows you to take a photo (or select from the photo roll) that you want to upload as an attachment to the Call in Tempest.

Tapping the button, displays a quick toast displaying the current Image Size, for example:

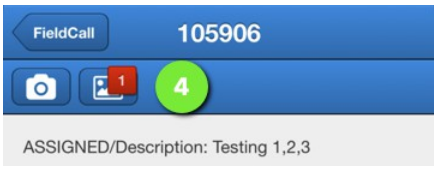


The default image size is 1200, but this can be edited in Preferences to be either 800 (small), 1200 (medium) or 1600 (large) as the maximum long side (in pixels) stored for Uploaded images.

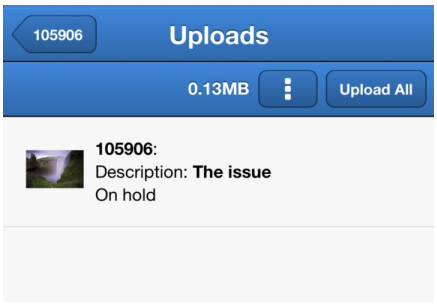
Now, a menu will allow you to choose either Camera or Photo library. Once the photo is taken (or photo roll/library item selected), the Upload screen is displayed and you must 1. add a description, and then choose either 2. Hold or 3. Upload immediately:



Choosing Hold saves the image in an Uploads wait list on your device, and sends you back to the Call screen with 4. the “Uploads” button (indicating there are held pictures waiting to be uploaded):

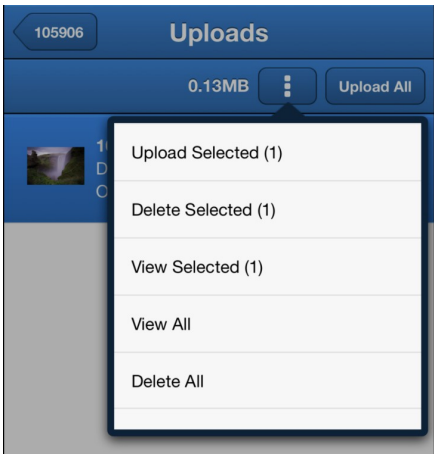


The Uploads button has a badge on it, displaying the current number of held pictures, and tapping the button displays the Uploads wait list:



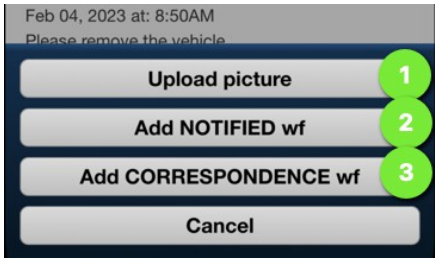
This is the list of all the Uploads waiting to be uploaded. Tapping Upload All will immediately begin uploading all the items on the the list. On Android, you can potentially interrupt the upload process, if desired, by hitting the Back button; but on iOS the upload process will continue until done.

Selecting an item (or items), and tapping the ... button displays a menu, allowing you to do various things with the selection, including deleting or viewing the selected items:



Button 3. - the action sheet

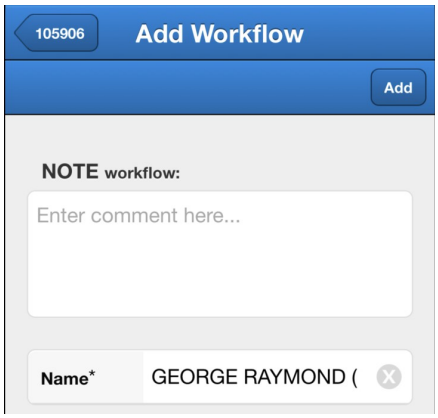
This button displays an action sheet at the bottom of the screen:



Upload picture (1.) does the same thing as tapping the Photo button as described above. Buttons 2. and 3. allow you to add (perhaps less-used) NOTIFIED and CORRESPONDENCE workflow to the Call.

Buttons 4. 5. and 6. - Add CANCELLED, COMPLETED, NOTE Workflow

These buttons display a similar screen allowing you to add that workflow type, as well as a comment to go with the workflow:

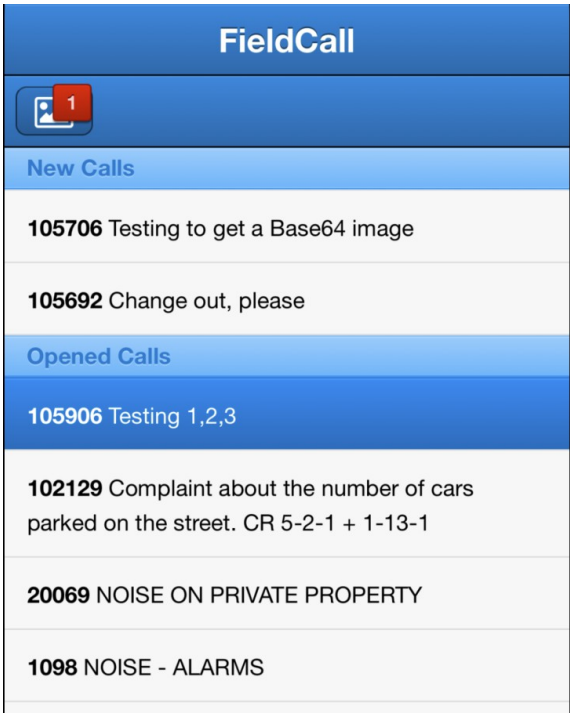


Your name is pre-entered for you in the Name field, so all you need to do is tap in the comment area, and enter your comment. When you are done, tap the Add button to add the workflow to the Call. You are returned to the Call details screen, and the new workflow is displayed in the workflow area at the bottom of the screen.

COMPLETED and CANCELLED will also set the Call status appropriately, and completing a Call that has a repeat frequency set (e.g. ANNUALLY) will request that you re-issue the Call in Tempest or FieldBlade as re-issuing is out of the scope of the FieldCall app.

Returning to the main Calls screen

If there are any Uploads in the wait list, the Uploads button is displayed top left, and tapping the button displays the same Uploads wait list screen as described above – allowing you to process the Uploads wait list from the main screen as well.



Additionally, if the Call we tapped into was in the upper New Calls section when we started, it will now be displayed in the lower Opened Calls section.

2. Installation/Upgrading

Note: If you are upgrading from a previous version, see the notes near the end of this section regarding upgrading. Then read through the rest of this section for further information.

System Requirements

Device:

Both iOS and Android phone and tablet devices are supported.

Because device versions are generally a fast-moving target it makes little sense to add a specific device list here. If you are purchasing new devices, just make sure you purchase late models that support Bluetooth SPP (every major device we know of lately has this, so it is not too much of a worry).

For Android, QA testing occurs on Samsung devices, and so these are generally recommended. Please contact support@mpowered.biz if you have other questions.

Web Services:

IIS servers must be capable of creating an Application Pool with a .NET CLR Version of at least v4.0.30319
The current targeted .NET version is 4.7.2

Tempest Licences:

Calls for Service
Web Customer

Technical Specialist Tasks

Because these next item(s) may take some time, these are shown out of context here right away so the appropriate technical specialist can get them set up, hopefully by the time everything else in this document is ready to go.

Firewall – allow http traffic - External to/from Internal

The FieldCall webservices security model uses an External/Internal model – where the External webservices do nothing but proxy requests from the Webapp through to the Internal webservices using http (usually, although you can use https as well). This model isolates the exposure of crucial database connection settings to only the Internal webservices machine.

```
Task: Create a firewall from/to rule for http (or https)  back
and forth from the External/Internal machines.
```

Firewall – allow http traffic – to http://www.mpowered.biz

The FieldCall webservices check for updates on this server, and if not enabled will cause about a 2 minute delay on each webservices request until a timeout condition is achieved.

```
Task: Create a firewall to rule for http://www.mpowered.biz
```

Network Service account with read/write permissions to the Calls Attachments Directory (see G. in the Technical Overview below)

The MpoweredApps Application Pool (that will be created in IIS in an installation step below) on the Internal machine will need an account with Read/Write access to the Calls attachments folders (as well as having local administrator permissions so it can run the webservices).

```
Task: Create* a new (high-powered) network service account
(e.g. mpowered-webservices) for the MpoweredApps app pool (to
be created below) that has local administrator permissions AND
also has read/write permissions for the Tempest Calls
attachments directory(s) as defined in Tempest > Tools >
Workgroups > {workgroup} > Attachments Directory.
```

* If you have already created this account for another Mpowered app, you could just add the read/write permissions to that account for the Tempest Calls attachments directory(s) as defined in Tempest > Tools > Workgroups > {workgroup} > Attachments Directory.

Create database user MpoweredWeb

Create a database user named MpoweredWeb in each Tempest database (usually LIVE and TEST) that you wish to access.

Grant database user MpoweredWeb database access permissions

Grant the permissions to MpoweredWeb as found in \Docs\dbgrants.txt which contains a list of permissions specific to each FieldCall version or update. As each new FieldCall version or update is released, this list may change, so stay up-to-date as a missed permission will cause problems for users.

IMPORTANT!: After performing a Tempest update (or group of updates), you will need to re-grant the permissions, because unfortunately the Tempest update process does not retain 3rd-party permissions.

Download the Install package

Go to www.mpowered.biz and click on Downloads. Here you will find links to various packages that match recent versions of Tempest. For example, for Tempest version 80000, you would download the highest FieldCall install package starting with 800, in this case FieldCall ver. 80003. This will download the ZIP package which you can then extract into a working directory on your internal and external webservers.

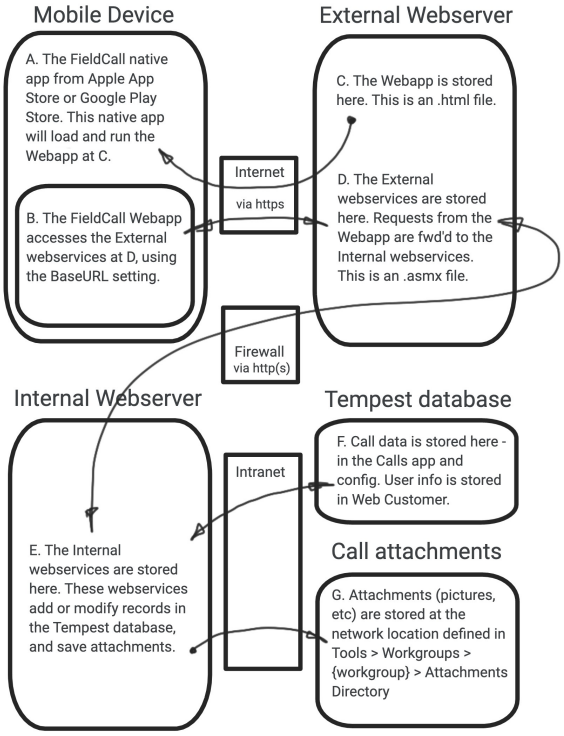
Contents of the ZIP package

Once the ZIP package is extracted, you will find this structure:

\Docs
\Dotnet
\Webapp

Technical overview of the FieldCall structure:

Let's take a look at how everything works conceptually.



A. The device (iOS or Android) downloads and runs the "Mpowered FieldCall" native app from the App or Play Store. When the app is run, it will ask for a Webapp URL, which is located at C. When the "Go FieldCall" button is tapped, the native app loads and runs the Webapp (B).

B. The Webapp is what Users interact with to process Calls – the top portion is shown here:

One of the main configurations within the Webapp is the Base URL setting when it needs to get or set data in Tempest by sending requests over the Internet (via https) to the External webservices located at D.

D. The External webservices forward requests (through the firewall via http) from the Webapp (B) to the Internal webservices (E). The Web.config file for D is very simple, containing only a base location for the Internal webservices (E).

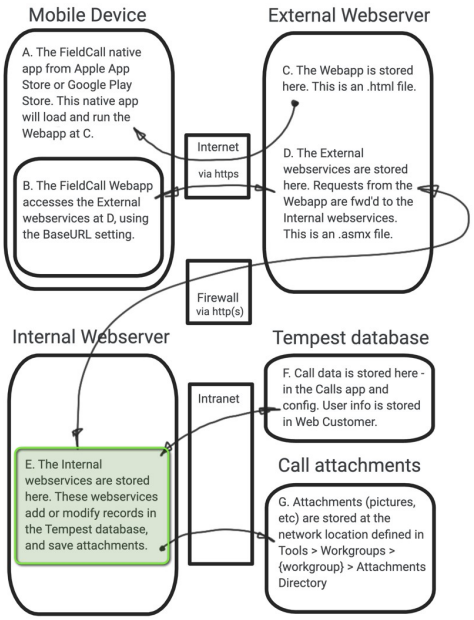
E. The Internal webservices process requests from the External webservices (D) – usually getting or setting data in the Tempest database (F) and/or attachments (G), and then responding back (through the firewall) to the External webservices (D) which then responds back to the Webapp (B). The Web.config file for E contains a connection string to the Tempest database, therefore it can be encrypted - if that level of security is desired.

NOTE: C (the Webapp) & D (the External webservices) must exist on the same https webserver.

Installation order

We are going to install each piece of the puzzle, making sure each part works correctly before moving on to the next part. We'll start with installing the Internal Web Services (E), then move on to installing the External Web Services (D) and Webapp (C), do any configuration needed in Tempest (F), and then finish by installing and running FieldCall on the device (A + B).

E. Internal Web Server - Installing the Web Services



On your <u>internal</u> (behind the firewall) web server, create a home directory for the Mpowered .NET webservices if you don't already have one… something like:

```
C:\inetpub\wwwroot\Mpowered\FieldCall-80003WS
```

Option 1: If your back-end database is **SQL Server**: copy the entire \Dotnet**Redmond*** directory contents from the download here. Now on your internal web server, you should have this structure:

```
…\wwwroot\Mpowered\FieldCall-80003WS\
      bin\
            FC80003.dll
            FieldCall.asmx
            Web.config.internal.txt
            Web.config.external.txt    * delete this file
```

Delete the Web.config.**external**.txt file

Only use one of these options! The Redmond dll will not work with Oracle, and similarly, the Oracle dll will not work with SQL Server.

Option 2: If your back-end database is **Oracle**: copy the entire \Dotnet**Oracle*** directory contents from the download here. Now on your internal web server, you should have this structure:

```
…\wwwroot\Mpowered\FieldCall-80003WS\
      bin\
            FC80003.dll
            Oracle.ManagedDataAccess.dll
            Oracle.ManagedDataAccessDTC.dll
            FieldCall.asmx
            Web.config.internal.txt
```

Now edit the Web.config.internal.txt file and look for a section with the tag <connectionStrings> near the bottom. Here you will see a sample connection string for SQL Server named "MpoweredSQL", and one for Oracle named "MpoweredORA". You can completely remove the line that doesn't apply to your site. DON'T change the first part of the connection string name, i.e. "MpoweredSQL" or "MpoweredORA".

With the connection string you will use, edit it so that YOURHOST becomes the server name where the Tempest database lives, and INSTANCE becomes the name of the database instance. Also, change the Password= to the MpoweredWeb password you created earlier. (NOTE: the password is entered in clear text here – this file should be secured (or encrypted) so that only people with proper permissions can view this file.) If you don't know the server name or password values, you may have to talk with your Database Administrator.

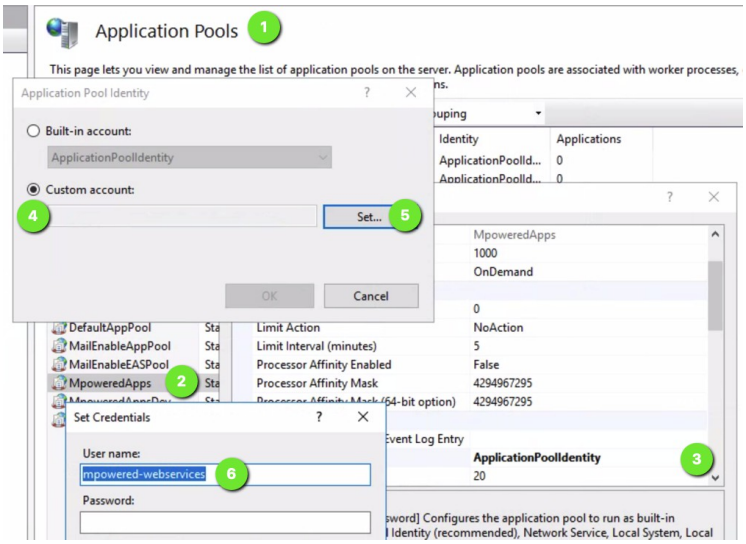
Note: you can have multiple connection strings in this file, for example you could have an MpoweredSQLProd and an MpoweredSQLTest connection string each pointing to the Production and Test Tempest databases. When you enter the Authentication settings on the mobile device, you choose which DSN (connection string) to use.

Very important to rename this file!

Save and exit. Rename the Web.config.internal.txt file to **Web.config**

Now we need to fire up IIS Manager on the internal web server. Browse into Application Pools, and right-click and choose Add Application Pool. Create a new pool named "MpoweredApps" using .NET CLR Version v4.0.30319 (if you do not have this version, you will need to install MS .NET Framework 4.5 on this machine), Integrated, Start application pool immediately ON. Click on the newly created pool, and browse to Advanced Settings on the right side menu. Make sure that Enable 32-Bit Applications is set to True.

This is also where we need to change out the default Identity (account) that IIS creates for this App Pool.



Still in the Application Pools > MpoweredApps > Advanced Settings, scroll down to the Identity item (3), and click on the ... beside ApplicationPoolIdentity. Click on Custom account (4), Set... (5), and add the credentials (6) for the account that was created in the "Technical Specialist Activities > Network Service account" task above. Click OK a number of times till you are done with all dialog boxes.

Now on the left tree, browse down to Sites > Default Web Site > Mpowered and right-click on FieldCall-80003WS. Choose Convert to Application. Keep the Alias as FieldCall-80003WS, but select Application pool MpoweredApps, and click OK. This should change the icon in the tree to:  .

Now right-click on FieldCall-80003WS again, and choose Manage Application > Browse. The default browser should appear with the FieldCall .NET services listing, containing links for AA_ServiceInfo, AB_ServiceTest, AC_DatabaseTest, etc. Click on AC_DatabaseTest. If you are a SQL Server site, you can just hit Invoke; otherwise you will need to enter something like <root><dsn>MpoweredORA</dsn></root> into the postedGET field and hit Invoke. You should get an XML page that says "SUCCESS: Found nnnn rows in the calls_calls table". This means that the DSN was set up correctly, and we are getting a connection to the Tempest database.

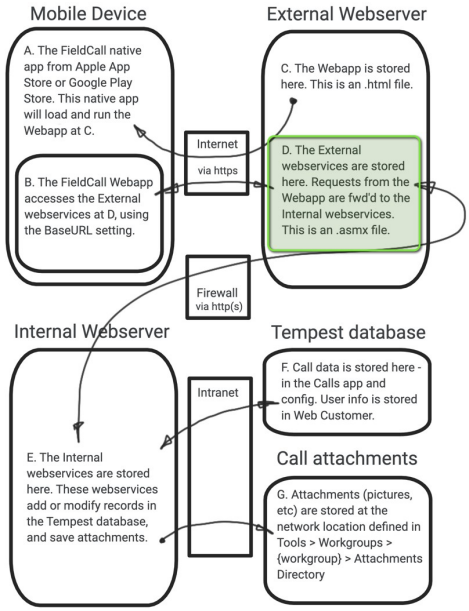
For SQL Server, if you get the message "Timeout expired. The timeout period elapsed prior to completion of the operation or the server is

not responding." you may be able to solve the issue by running "exec sp_updatestats" on the database.

That completes the set-up of the internal web server.

Note that if you are using a web application firewall (for example, Barracuda's Web Application Firewall - WAF), you probably will not need to set up the external webserver as described in the remaining part of this section below - the WAF will manage the external/internal forwarding.

D. External Web Server - Installing the Web Services



(Before starting, read through the section "Creating a virtual application for the External webservices" below as it may affect how you name directories here.) The set up of the external webserver is almost identical to the internal webserver setup. On your external (outside the firewall) webserver, create a home directory for the Mpowered .NET webservices… something like:

```
C:\inetpub\wwwroot\Mpowered\FieldCall-80003WS
```

Copy the entire \Dotnet\Redmond directory from the download here. Now on your external webserver, you should have this structure:

The Redmond directory is used on the external server for both SQL Server and Oracle.

```
…\wwwroot\Mpowered\FieldCall-80003WS\
      bin\
            FC80003.dll
            FieldCall.asmx
            Web.config.internal.txt    * delete this file
            Web.config.external.txt
```

Delete the Web.config.**internal**.txt file

Now edit the Web.config.external.txt file and look for a section with the tag <appSettings> near the bottom. Here you will see a "requestForwardTo" key. It is the value that you must edit to point to the webservices location on the internal webserver (through the firewall). You may need to get your firewall expert to help you figure this one out. In most cases you will simply need to change {ip} to the ip address of the internal webserver (as seen from outside the firewall).

Very important to rename this file!

Save and exit. Rename the Web.config.internal.txt file to **Web.config**

Now we need to fire up IIS Manager on the external webserver. Browse into Application Pools, and right-click and choose Add Application Pool. Create a new pool named "MpoweredApps" using .NET CLR Version v4.0.30319 (if you do not have this version, you will need to install MS .NET Framework 4.5 on this machine), Integrated, Start application pool immediately ON. Click on the newly created pool, and browse to Advanced Settings on the right side menu. Make sure that Enable 32-Bit Applications is set to True, and click OK.

Now on the left tree, browse down to Sites > Default Web Site > Mpowered and right-click on FieldCall-80003WS. Choose Convert to Application. Keep the Alias as FieldCall-80003WS, but select Application pool MpoweredApps, and click OK. This should change the icon in the tree to:  .

Now right-click on FieldCall-80003WS again, and choose Manage Application > Browse. The default browser should appear with the FieldCall .NET services listing, containing links for AA_ServiceInfo, AB_ServiceTest, AC_DatabaseTest, etc.

Click on AB_ServiceTest and hit Invoke. You should get an XML page that says "SUCCESS". This means that the "requestForwardTo" key was set up correctly, and we have a connection to the internal web service.

Close the browser, and right-click on FieldCall-80003WS (in IIS Manager) again, and choose Manage Application > Browse. This time click on AC_DatabaseTest. If you are a SQL Server site, you can just hit Invoke; otherwise you will need to enter something like `<root><dsn>MpoweredORA</dsn></root>` into the postedGET field and hit Invoke. You should get an XML page that says "SUCCESS: Found nnnn rows in the calls_calls table". This means that the external/internal connection is working, and the internal webservices are proxying correctly to the Tempest database.

Creating a virtual application will save you time in the future, and allows you to easily deploy updates to devices.

<u>Creating a virtual application for the External webservices</u>
If you are interested in saving time and staging updates as they are released by Mpowered, you could name the above directory:

```
C:\inetpub\wwwroot\Mpowered\FieldCall-80003WS-START
```

Then, creating a virtual application in IIS at the \Mpowered level called “fcws” and pointing it to FieldCall-80003WS-START will allow you to use a Base URL that looks like this:

https://yourserver/mpowered/fcws/FieldCall.asmx/

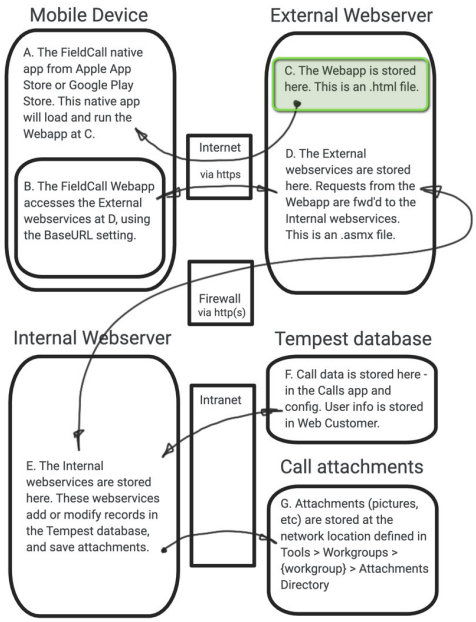
Using a virtual application can save you time when its time to upgrade. So, for example, when and if there is an update, and UPDATE01 is released, you could create a new directory:

```
C:\inetpub\wwwroot\Mpowered\FieldCall-80003WS-UPD01
```

and install the updated webservices there. When you have finished testing the update and are ready to release to your users, simply change the Physical Path in the virtual application fcws in IIS to point to the new directory, and all users with devices that have a Base URL of: https://yourserver/mpowered/fcws/FieldCall.asmx/
will automatically start using the new webservices without changing anything on their device the next time they start the app.

It's still a good idea to let the users know that an update is happening! Schedule the update for a time after everyone has finished for the day.

C. External Web Server - Installing the Webapp



(Before starting, read through the section "Creating a virtual directory for the Webapp" below as it may affect how you name directories here.) On your external (outside the firewall) webserver, create a versioned directory for the Webapp… something like:

```
C:\inetpub\wwwroot\Mpowered\FieldCall-80003WA
```

Copy the entire \WebApp directory from the download here. Now on your external webserver, you should have this structure:

```
…\Mpowered\FieldCall-80003WA\
      FieldWorksX\
      lib\
      resources\
      app_nnnnnnnnnnnnnnn.js
      favicon.ico
      index.html
```

Creating a virtual directory will save you time in the future, and allows you to easily deploy updates to devices.

Creating a virtual directory for the Webapp

If you are interested in saving time and staging updates as they are released by Mpowered, you could name the above directory:

```
C:\inetpub\wwwroot\Mpowered\FieldCall-80003WA-START
```

Then, creating a virtual directory in IIS at the \Mpowered level called "fcwa" and pointing it to the FieldCall-80003WA-START will allow you

to use a Webapp URL on the device app that looks like this:

https://yourserver/mpowered/fcwa/index.html
Using a virtual directory can save you time when its time to upgrade. So, for example, when and if the Webapp gets an update, and UPDATE01 is released, you could create a new directory:

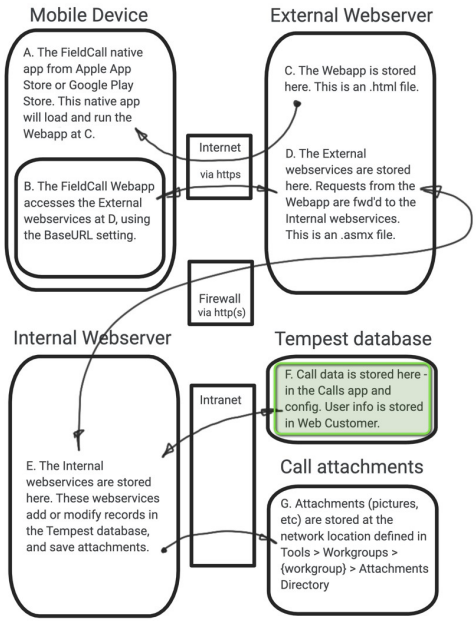
```
C:\inetpub\wwwroot\Mpowered\FieldCall-80003WA-UPD01
```

and install the updated Webapp there. When you have finished testing the update and are ready to release to your users, simply change the virtual directory fcwa in IIS to point to the new directory, and all users with devices that have a Webapp URL of:
https://yourserver/mpowered/fcwa/index.html
will automatically start using the new Webapp without changing anything on their device the next time they start the app.

It's still a good idea to let the users know that an update is happening! Schedule the update for a time after everyone has finished for the day, and get everyone to force-close the FieldCall app when they're done for the day.

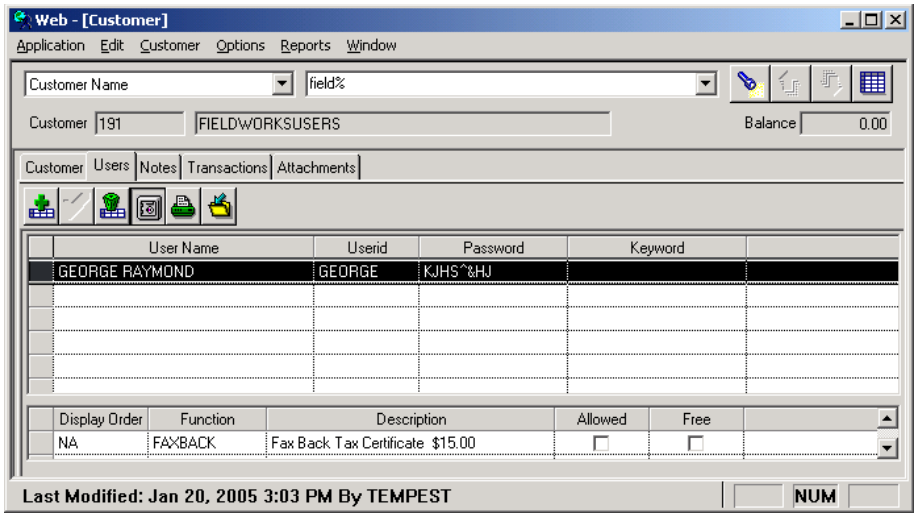
That completes the set-up of the external webserver.

F. Tempest Configuration



Create the FIELDWORKSUSERS customer in Web Customer

In Tempest Web Customer, create the FIELDWORKSUSERS customer. Make the user an INTERNAL type. Then, create the Calls user in the FIELDWORKSUSERS users whose UserId corresponds to the user's actual database UserID:



In this example, we have created user GEORGE RAYMOND. George is a Calls user, and logs into Tempest with the UserID GEORGE. You also need to set a password for GEORGE. This will be the password GEORGE will need to enter on the device in the next steps. In our example, GEORGE's password is KJHS^&HJ

The users created in the FIELDWORKSUSERS customer should not have any functions turned on.

You will need to create a user in the FIELDWORKSUSERS customer for each user of FieldCall.

Enabling Active Directory network passwords

As a more secure (and user-friendly) option to using the Web Customer password (as shown above) is to enable network password authentication via MS Active Directory. To use network password authentication, your Tempest User Ids must match the Active Directory name.

1. Activating for individual FieldCall users: Add these settings in the Internal Web.config in the <appSettings> branch as follows:

```
<add key="domain" value="CORP" />
<add key="domainmodel" value="WCKEYWRD" />
```

changing CORP to the name of your corporate active directory name. Then, for each user that you want to switch from Web Customer password to network password, change their keyword in Web Customer to "AD". That FieldCall user should be advised to Authenticate with their network password.

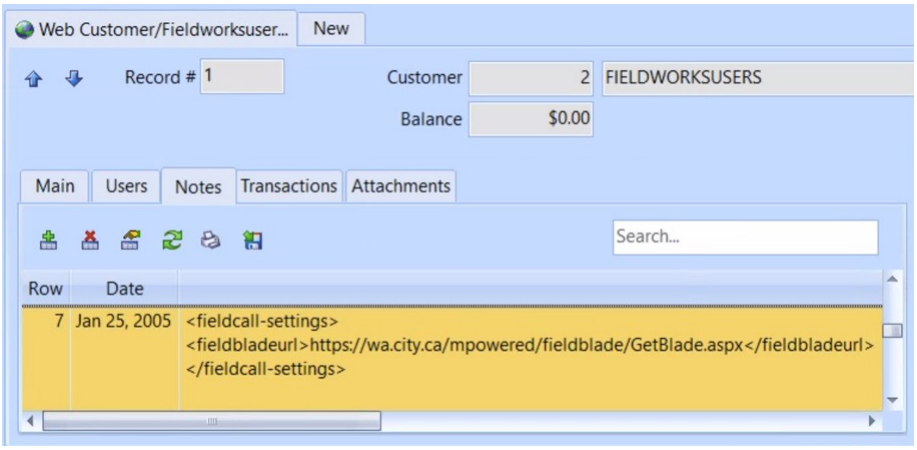
2. Activating for all FieldCall users all at once (usually for new customers): If you are switching password models, it may be better to use option 1, because activating with this option will instantly apply to all FieldCall users, and if they are not aware of the change, they will be asked to Authenticate, and be stuck trying to use their Web Customer password which will not work anymore! To activate network passwords for all FieldCall users, add a setting in the Internal Web.config in the <appSettings> branch as follows:

```
<add key="domain" value="CORP" />
```

changing CORP to the name of your corporate active directory name. This should be a planned change, and FieldCall users should then be advised to Authenticate with their network password.

Control Settings

FieldCall control settings are stored as an XML string in the Notes tab of Web Customer for the FIELDWORKSUSERS, in a note dated Jan 25, 2005:

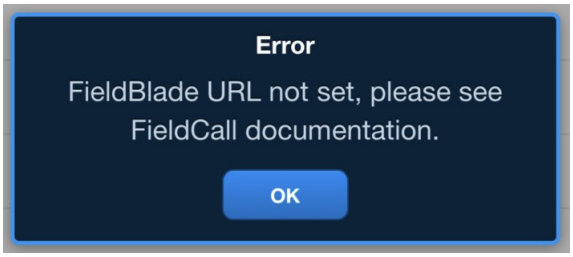


The XML string entered into the note must be formatted correctly for FieldCall to interpret the settings correctly (see XML notes). An example of this setting is:

<fieldcall-settings>
<fieldbladeurl>https://wa.city.ca/mpowered/fieldblade/GetBlade.aspx</fieldbladeurl>
</fieldcall-settings>

With this XML string we are telling FieldCall where our FieldBlade Base URL is located. On our external web server we have a virtual directory mpowered/fieldblade which points to our latest version of FB so that we don't have to change this setting each time FB is updated.

This is currently the only control setting in FieldCall, and if a setting is not provided in the XML string, FieldCall will display this error message if you choose the “Open Call in FieldBlade” option:



Email Notifications

To enable notifications for FieldCall, the Internal server will require some additional setup, and you must make sure that you have run the latest database grants found in \Docs\dbgrants.txt on the database(s). The internal Web.config will need some additional settings to define your email server, and timeframes to allow emailing. See the released Web.config.internal.txt in the <appSettings > section. Add this section to your existing Internal server's Web.config, and then review all settings, and change to suit your configuration and operational timeframes. Make sure you use an online Web.config validator such as: https://elmah.io/tools/config-validator/ to ensure your Web.config XML is valid. The web service includes a method named ba_notifications which is the notifications runner. You will probably want to run the ba_notifications method at a frequency (15 min, for example). Follow the instructions in the released Web.config.internal.txt file for creating a Powerscript file, that can be run using Task Scheduler.

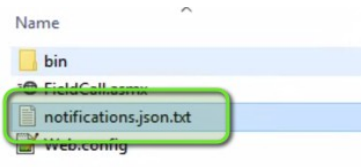
Then, to enable notifications for a FieldCall user, they must:
1. turn Setting "Calls count" to Notify in FieldCall, and
2. have a current email address in Tempest Resources.
Then, after a Refresh in FieldCall, if the user's active Calls count changes, the system will send an email when this process runs. Notifications data is held in-memory on the webserver, and can optionally be persisted to disk (see below).

If you have done the setup as described above, and nothing seems to be happening, the Web.config.internal.txt has instuctions on getting debug output. Mpowered is here to help, as always, if needed.

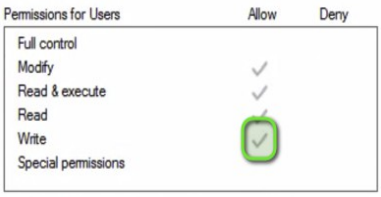
Persisting the notifications database

By default FieldCall stores and retrieves notifications data from a dictionary held in the FieldCall service's memory. This works great, until the power goes out, or you need to restart the web server or FieldCall web service, etc. At that point, any notifications data held in memory is lost, and begins to rebuild once the service is running again, and users perform refreshes. Persisting the notifications data allows FieldCall to store this data on the server and survive service shutdowns. It is a simple step to enable persisting, but it requires setting a write permission on a file on the Internal web server, which may not be for everyone. And note that the default in-memory option is a completely useable option, and that persisting the notifications data with the steps below is definitely not required.

If you wish to proceed, create a file in the FieldCall web service directory named notifications.json.txt:

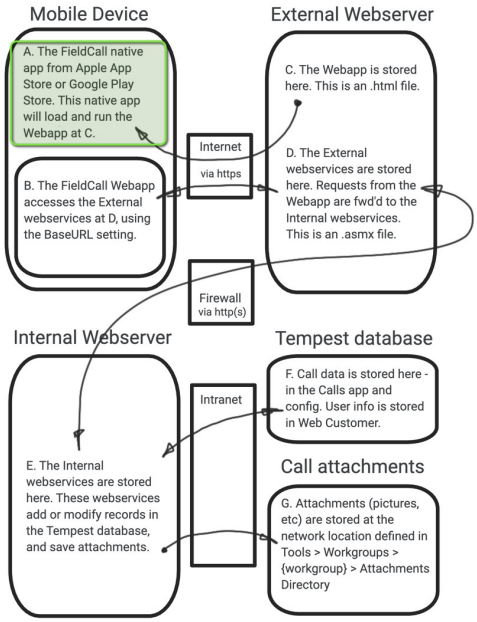


Then, grant the “Users” object Write permission to the file:



That's it. Now, the persisted notifications data stored in this file will be used any time the notifications system needs to read or change it.

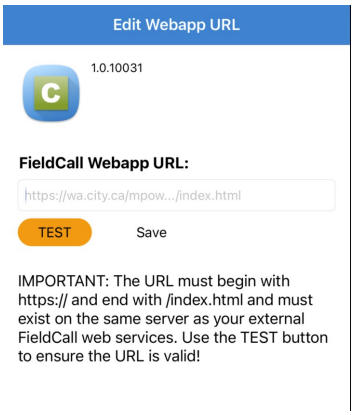
A. Load the FieldCall app to the device



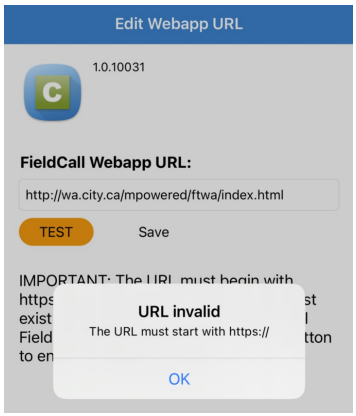
On the App or Play Store, search for **Mpowered FieldCall**. The app's icon looks like this:

Tap the Install button to install the app on your device.

When you first open the FieldCall native app on the device, it looks like this:



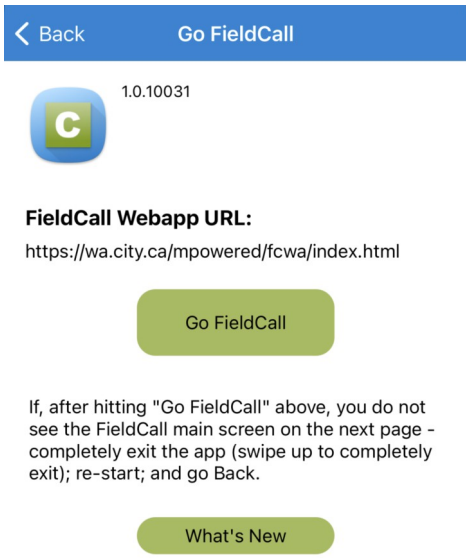
In the field, enter the URL to the virtual directory you created in step C above, and press TEST. NOTE: the TEST button can only check certain formatting of the URL, not whether it actually exists, so BE CAREFUL about what you are typing in here. For example, here we missed the s in https:



(Devices will often have an auto-correct feature which will change what you enter here mostly without you knowing, and mess things up. It's best to temporarily turn off auto-correct when entering the Webapp URL.)

Once you are sure about what you have entered, press the Save button.

Now, you will have the Go FieldCall screen:

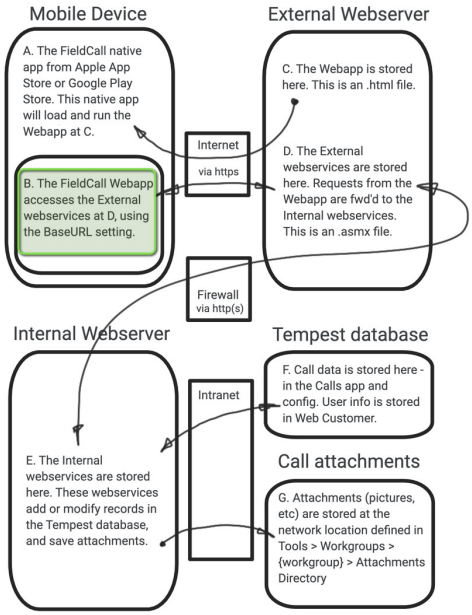


In the future, FieldCall will always start with this screen, now that you have entered a Webapp URL. Tap Go FieldCall, and in a few seconds, the Webapp will appear.

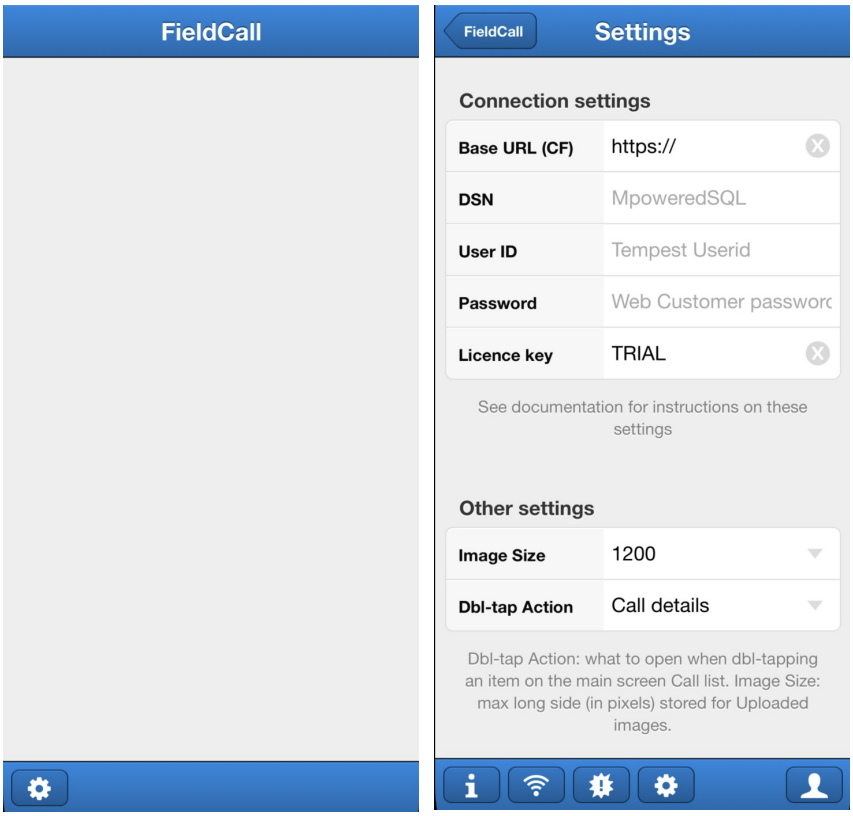
(That is, if everything was entered correctly, the Webapp will appear. If, after the spinner stops, you have a pure white screen, or you get a message like “The specified URL cannot be found.”, then its probable that something wrong in the Webapp URL entry. Swipe-close (i.e. completely close) the app, run it again, and review the Webapp URL shown. If it's wrong, then you can use the Back button (upper left) to go back and edit the Webapp URL and try again.)

If all went well, you should see the FieldCall app appear with the gear icon in the lower left corner – as shown in the next section, and you can carry on with setting up the FieldCall Webapp.

B. FieldCall Webapp setup



Tap the Gear button in the lower left corner to get to the Settings screen:



Connection settings

Connection settings are required as this establishes the location (on the Internet) of the webservices allowing FieldCall to communicate with Tempest data, as well as establishing your credentials.

Change the defaults to your site's specific values, for example:

Base URL: the location of your webservices root on the external server, e.g.:
https://wa.city.ca/mpowered/fcws/FieldCall.asmx/

After entering the Base URL, you can test whether FieldCall can reach it by tapping the Radar button. The messages shown will indicate if your Base URL is reachable.

DSN: the .NET (e.g. MpoweredSQL) Data Source Name.

User ID: the Web Customer ID as created above in the step "Create the FIELDWORKSUSERS users in Web Customer".

Password: the Password as created above in the step "Create the FIELDWORKSUSERS users in Web Customer".

Licence key: the 5 character licence key supplied to you by Mpowered. Licence keys can be purchased from Mpowered.

Other settings

Image Size: defaults to 1200, but can be set to 800 (normal res), 1200 (medium res) and 1600 (high res). The number chosen will be the maximum long side (in pixels) stored for Uploaded images.

Dbl-tap Action: defaults to Call details, but can be set to Open Call in FieldBlade. This setting determines what to open when dbl-tapping an item on the main screen Call list.

Debug/support settings

These settings should be changed when working with Mpowered support.

Bottom tool bar

i button: Opens a screen showing contact and technical information, and may be used by Mpowered support during the course of a support call.

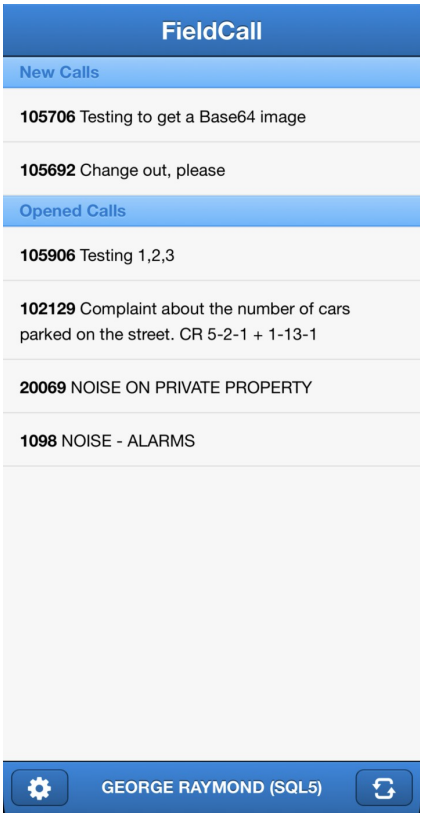
Radar button: Useful to determine if you are able to connect to the entered Base URL.

! button: This screen shows information logged by the system, and may be used by Mpowered support during the course of a support call.

Gear button: allows storage of various Settings configurations, usually used by your IT staff or Mpowered developers.

Authenticate button

Once you have entered the connection settings, tap the Authenticate button (silhouette). This will validate the information you entered, and bring you to the main screen. The user's name (from Tempest) will be displayed in the bottom toolbar on the main screen, and any Calls assigned to the user will automatically be displayed:



The refresh button (lower right) can be used to refresh the Assigned Calls list at any point in the day, and should be done at minimum at the beginning of every day.

Upgrading from a previous version

1. If you wish to test this new release, please review the "Testing releases/upgrades" section below. Otherwise, continue with these steps.
2. Copy/Install the 80003 webapp and webservices to your webservers as per the instructions in the Setup section. The webapp is required for Android users as of this version. Also note that webservices are now targeting .NET 4.7.2, so you should (but still optionally) update your Web.config (internal and external) targetFramework attributes to use 4.7.2 rather than 4.5. See the released Web.config files for examples.
3. Run all the grants in the Docs\dbgrants.txt file.

General upgrade notes

All webapps and webservices releases and patches are cumulative and include all fixes and upgrades from previous updates. FieldCall is integrated with Tempest, and may or may not require maintenance as described below.

Major releases

A major release of FieldCall (FC) will coincide with a major Tempest release, that is, when any of the first 3 digits of a release change, e.g. 72000 to 80000. You *must* (and can only) upgrade FC when you have upgraded the underlying database in order to continue using FC. All major releases are full (i.e. cumulative), i.e. webapps and webservices are released as a full package, and will usually require upgrading the webservers with the new versions. After every major release, run all the grants in the Docs\dbgrants.txt file. (Note that since Tempest 8, major releases seem to have stopped.)

Patch releases

When any of the last 2 digits of a FieldCall release change, e.g. 80002 to 80003, this is an Mpowered patch release. Mpowered *does not* synchronize these patches with Tempest. Therefore, when Tempest releases a patch, there will not necessarily be a corresponding patch release by Mpowered. Mpowered releases patches in order to fix bugs and/or introduce new features. All patch releases are full (i.e. cumulative), i.e. webapps and webservices are released as a full package, and will usually require upgrading the webservers with the new versions. After every patch release, run all the grants in the Docs\dbgrants.txt file.

Testing releases/upgrades

To test releases before going into production, install the new webservices as explained in the Setup section (ensuring that you are using different directory names for the webapp and webservices than production/live). On a test device, run FieldCall and point its Webapp URL to the **new fully specified webapp directory (not the virtual directory)*** and then inside the webapp point the Base URL to the **new fully specified webservices directory (not the virtual application directory)***, and DSN to the **TEST** database DSN. Once testing is complete and then to upgrade all users, simply edit the virtual directories* in IIS for both the webapp and webservices used for production/live to point to the new assets and all devices will magically update! iOS devices may need to do a swipe-close and wait 15 minutes in order to reload the new webapp, hence it is best to do the virtual directory change after working hours. For more information contact Mpowered.

* see the setup section for detailed descriptions of managing updates using virtual directories in IIS.