Android Development

Android development is supported by the Barcoders ABS Wavelink replacement.

This consists of 2 classes to embed in the application

- ABSConstants.bas
- ABSStudio.cls

The class ABSStudio.cls creates every object and method that would have been created by WavelinkOLE.tlb.

Therefore, whenever these objects are instantiated, the code must now instantiate as ABSStudio rather than the separate Wavelink classes, for example:

Original code:

```
Private termIO As New RFTerminal
Private rfMainMenu As New RFMenu
Private rfConsole As New RFIO
```

Becomes:

```
Private termIO As New ABSStudio
Private rfMainMenu As New ABSStudio
Private rfConsole As New ABSStudio
```

The RDTMenu1 Project has been modified to include a number of new compile switches, which are accessible through VersionMaker:

- ANDROID - part of the Simul group. This controls swapping Wavelink, Debug and Android includes when building.
- ABS_LOGGING - additional logging for every call to an ABSSyudio method, including the call parameters.
- TALKIE - controls whether the app makes use of ABSSpeak to speak text labels. This is a tri-state flag:
    - 0 - disabled
    - 1 - Enabled for RFError and DialogBox text
    - 2 - Enabled for text labels when entering an InputField as part of an InputScreen

All changes to the code in order to support ABSStudio are wrapped in the ANDROID compile switch. Search for ANDROID to find the changes.

Most changes are in the RDT_IO module (RDTInput.bas).

ABSStudio does not support line-drawing characters, so ensure that the system is set to Block characters to ensure that this works as expected.

ABSStudio does not return errors like Wavelink, and so this code has been removed for Android (RDT_IO.ErrorAction)

ABSStudio does not support hiding the cursor "off-screen" and therefore this code is removed for Android (RDT_IO.HideCursor)

ABSStudio does not support or need Flush Output and so this code is removed for Android (RDT_IO.FlushOutput)

ABSStudio supports only 1 menu object, which is automatically reused for every menu displayed. As Wavelink supports multiple named menus, the code has been modified fairly extensively to control rebuilding of the main menu if any sub-menu has been displayed(RDT_IO.ResetMainMenu and MainProcess.Main and ResetMainMenu)

ABSStudio does not support menu coordinates and so this code is removed for Android (RDT_IO.MenuSetCoordinates)

ABSStudio does not support screen timeouts and so this code is removed for Android (RDT_IO.Setup)

ABSStudio returns semi-colon instead of CR or NL on get event and so this code is changed for Android (RDT_IO.GetEvent)

Testing

The changes for Android are at the same level as Wavelink and Wavelinkesque and can be tested as such - new modulear functionality should not require specific Android device testing after the product is bedded in.

Releasing

Releasing should be through the same VersionMaker build and release process that is use today - build the Android version fo the app to "RDTMenu1_Android.Exe". Modify your VesionMaker configuration file for this, as per the example below):

```
[Id]
Id{T}Site
[Major Vars]
Simul{O}RDT(WaveLinkOLE.tlb[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\TypeLib\{08105893-8012-11D0-82BE-444553540
ESQUE(WLesque.dll[HKEY_LOCAL_MACHINE\SOFTWARE\Classes\TypeLib\{3692ACB7-79BB-4112-8FE3-AEE2EB8A2F2D}\3.0\0\
ANDROID(=RDTMenu1_Android.exe)
[Minor Vars]
Logging{C}LOGGING,LOGGING_WINDOW,LATE_BACKOUT
Android-Specific{C}ABS_LOGGING,TALKIE
```

Note that the release script has been modified to support the application being build in this way with this naming convention. This script will also require updating to any new installations of WCS, and to any existing installations that will start to use ABSStudio.

Note: Upgrades to our software requiring upgrade to Barcodes ABS Server is an exceptional process.

The install to both the Android ABS Studio Server and Android client must be rolled in or back manually.

Never upgrade the ABS Server or Android app unless required by an upgrade.

Contents

1 Android Support Guide

1.1 Android Support Guide

References:

- Installing Barcoders Android Server
- Android Usage Guide

Filesystem:

- "C:\Program Files\Barcoders" - the ABSStudio files
- "C:\logs" - contains:
  - barcoders_upgrade.log - the upgrade log
  - ACDebugLog.txt - the server log
- "C:\data"

Shortcuts should be created on the desktop:

- "Start Barcoders Server" - the server itself - should already be running
- "Start Barcoders Admin" - similar to Wavelink Administrator - shows the connected devices and licenses.
- "Start Barcoders Updater" - for updating the version.

The ABSStudio Server runs in a command window that must remain open at all times. Do not close this window as this will stop all Android functionality for that customer.

When Android devices connect, the processes will be running under the "Start Barcoders Server" - it is from here that they can be killed with Task Manager.

The Android version of the RDT program is installed in exactly the same place as the existing RDT program RDTMenu1.exe and is called RDTMeni1_Android.exe.

1.1.1 Debug Settings

In "C:\Program Files\Barcoders\AndroidStudio" there are two configuration files:

- appsettings.json - application log levels
- config.json - the applications being run by ABSStudio

To configure applications that can be run by the system, add entries to the config.json file, example below:

```
{
        "ProgramTasks": [
                {       "ExecutablePath": "C:/WCSTEST/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (TEST)",
                        "UsePipes": "True",
                },
                {       "ExecutablePath": "C:/WCSLIVE/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (LIVE)",
                        "UsePipes": "True",
                },
        ]
}
```

To change logging for the application, modify the Logging setting of the appsettings.json file, example below:

```
  "Logging": {
    "LogLevel": {
      "Default": "Info",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Error"
    }
```


```
},
```

Values can be changed to

- Info
- Warning
- Error
- Debug

Logging is exported to the command window of the application.

Logging is also sent to the server log.

Debug mode can also be enabled on the shortcut when running the ABSServer, e.g.

```
"C:\Program Files\Barcoders\AndroidStudio\Android Connector.exe" --debug
```

Once debug has been enabled, the log files may be searched for the terminal ID.

Note: We can clear (delete) the log file and it will recreate with no issues whilst still running.

Note: A single copy of the log file is added to by the server - it does not change or rotate every day or hour.

A script is being investigated to copy and to cleardown old scripts - details will be added here and in the installation guide when available.

Tracing terminal ID through Debug logs

- Find the terminal ID in WCS Maintenance, using the Connected Users screen - copy the terminal ID (large characters with hyphens for Android devices)
- Use this to filter the log file for everything for that terminal ID.

1.1.2 Versions

When running the system, the Android application and the Android Server connector must be the compatible versions.

To see the Android app version, get the user to:

- Go to Settings
- Apps
- Find Warehouse Terminal
- Check version

To check the Server version, check the latest updated version number in the upgrade log.

Note: Upgrades to our software requiring upgrade to Barcodes ABS Server is an exceptional process.

The install to both the Android ABS Studio Server and Android client must be rolled in or back manually.

Never upgrade the ABS Server or Android app unless required by an upgrade.

1.2 Troubleshooting Guide

1.2.1 Android device can't connect

- Check can other Wavelink devices connect
    - ♦ Proves system up and running

- Check error on device
    - When using the app on a wireless device, the application being used is subject to the connectivity of the device to the network and to the application. If there are any issues with the network connectivity or the app cannot be accessed, the application will display the main screen, showing "Error Connecting". Ensure that there is a network connection, then advise hit any key on any physical or popup keyboard to reconnect.
    - If displays selection of system then does not show WCS splash screen:
        - Check Android app set up properly in JSON config settings.
    - If system selected then splash screen displays, then Error connecting to WCS
        - Check WCS running - basic WCS issue

- Check Android Connector Window
    - If not already running, use "Start Barcoders Admin" shortcut on Desktop, or "C:\Program Files\Barcoders\AndroidConnectorWindow\TermProtoTestWindow.exe"

- Check command window open
    - Command window will be named "Start Barcoders Server", or "C:\Program Files\Barcoders\AndroidStudio\Android Connector.exe"
    - Check Task Manager that it is running
    - Start if not

- Check license
    - Look at license.xml file
    - Shows users and expiration.
    - Note: This file cannot be modified as it is checked. Modifications will fail the check and the system will be useless.
    - Compare total number of users in the Barcoders Admin window - if there are connected devices up to the maximum, no more users can connect.

- Check for versions
    - Confirm version of Android connector version in server logs or in Upgrade log.
    - Confirm Android version on device
    - Confirm validity and compatibility of versions - check local project documentation.
    - Advise update device for compatibility

- Check for errors in the server.
    - Check the command window.
    - Check the server log.

1.2.2 Every Android Device cannot connect

Perform checks above to ensure that there are no other issues. If no other issues are found, the ABS Studio Server likely has failed and must be restarted.

Stop the barcode Server command window - this will disconnect EVERY android user.

Warning: This will interrupt any users that are connected, to ANY site. It is the equivalent of stopping the Wavelink server/service on the machine. Use with caution and with the permission of the operation.

Start the barcode command window again.

Users can now reconnect.

2 C-TMS Automotive RF Process

Automotive RF Process

2.1 Messages

Message	From > To	Description
404	TMS > WCS	Order Preadvice Message which creates the TMS_Preadvice_Header and TMS_Preadvice_Detail records.
804	WCS > TMS	Order Creation module sends these as each item is scanned onto the order.
441	TMS > WCS	Item/Load Creation message. Should get one of these for each 804 message. I think this creates the TMS_Load* records.
442	TMS > WCS	Trip ready to load message which I presume updates the TMS_Load* records to ?Ready to Load? status (!).
841	WCS > TMS	Load Confirmation
431	TMS > WCS	Unload/Receipt message (only if x-docked)
432	TMS > WCS	Trip ready to unload.
831	WCS > TMS	Unload Confirmation.
441	TMS > WCS	Item/Load Creation message again. The process loops around again to the 3rd message above.
851	WCS > TMS	Asset (Item) Enquiry
451	TMS > WCS	Asset (Item) Enquiry Response
471	TMS > WCS	DU (Media) Type Updates
472	TMS > WCS	New Depot

2.2 Standing Data

Whenever DU Type information is changed in C-TMS then 471 messages are sent to WCS for all of the depots requiring WCS RF Cross-Docking containing this information for WCS to maintain its own Media Types against the corresponding depots.

Whenever a depot is marked as requiring WCS RF Cross-Docking functionality then a 472 messages is sent from C-TMS to create this new depot in WCS.

A button (Re-Send ALL) can then be pressed to send all of the DU Types to WCS for all of the appropriate depots including the newly marked one.

2.3 Order Creation

404 messages arrive at around 00:10 each night, showing the next 24 hours of orders to be fulfilled. There should be one per destination location.

Order Creation starts. This process is used to associate pallets/cages (i.e. containers or assets) to an order.

First, a label printer is requested.

Then a media type:

If re-usable Asset, the process requests the user to scan the Asset code (barcode on the cage). This is validated as being the right media type by passing a 851 message to CTMS and getting a 451 message from C-TMS in response.

If non-reusable, ID is generated by WCS.

RDT prompts for destination - WCS matches this to a destination and order on the Preadvice tables.

If OK, a label is printed (even if it?s a reusable Asset with a barcode), a Preadvice Item record is created, 804 message is sent.

Note: In CTMS, this associates the item to the order. If this is an asset, and Asset History record is NOT created at this stage, so there?s no audit that the Asset was ever put on the order until Loading.

Note: *If this is a reusable Asset, the item is added by with a sequence (generated from the number of times the asset has been used) to keep the item unique.*

2.4 Loading

Once the order reached a certain status in CTMS **Warning**: INFO REQUIRED, the order is planned on a trip and that trip is allocated a marshalling location (outbound bay) in CTMS, the system generates loading messages (441 or 442 messages - **Warning**: INFO REQUIRED.

This creates Load* records for the orders, trips and items at status C (Load Checked, or Ready For Loading, if you prefer).

Note: This refers to a previous process that was designed into Dunelm for Load Checking before Loading proper - this is no longer used.

User blind enters the trip ID.

User scans each item on the trip.

Note: *For Assets: Although the barcode scanned contains only the asset code without the unique sequence, and the messages sent to WCS contain this sequence, the RDT identifies these as checking for an asset on the trip that matches what we have of the ID. There should only be one, but if not, the system sorts them in reverse sequence and selects the top 1.*

An 841 message sent for each item scanned (*for Assets: the whole ID plus sequence is sent back*).

Note: There is some functionality to Unload an item on RDT - **Warning**: INFO REQUIRED

Once complete, trip is marked as En Route.

Note: For Assets, at scanning, an Asset History record is created showing the Asset as In Transit at this stage.

2.5 Delivery

At this stage, CTMS generates messages to Microlise.

For Assets: Only the Asset ID (not the sequence) is sent to Microlise for delivery.

Microlise scans each item at destination and sends back a message to CTMS upon completion.

Order is marked as Complete.

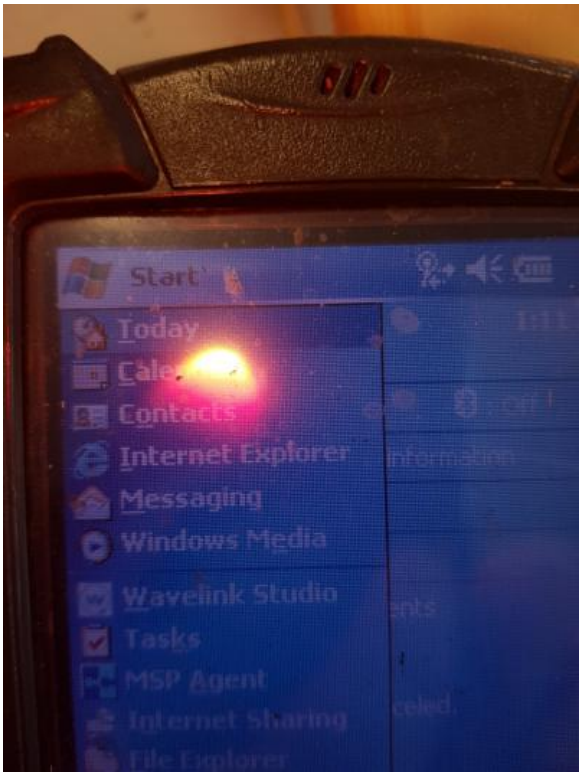
2.6 Unloading

At that depot, Receipt could now start:

Warning: INFO REQUIRED

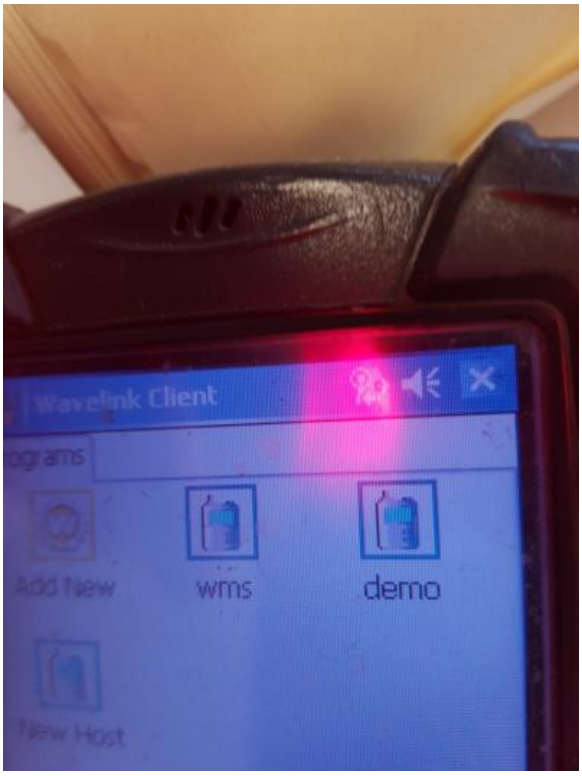
3 Changing the IP of an RF Unit

1. From the ?Start? menu, open the ?Wavelink Studio? option



2. When you click on ?Wavelink Studio? this will display the RF application icons

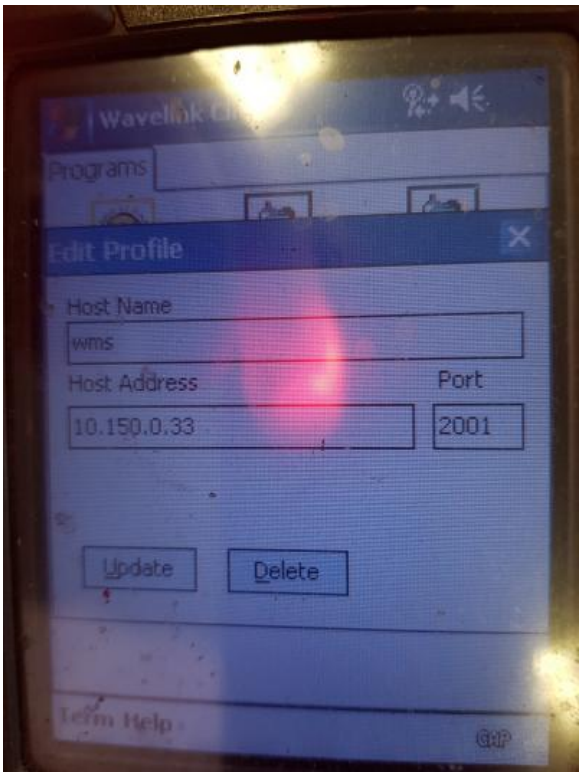
3. Press and hold (long hold) on the required RF icon (*the name is part of the configuration so will be different to the name displayed in the screen shot displayed below*)



4. The configuration options for the chosen icon will be displayed.

5. Enter the updated I.P. address in the ?Host Address? box and click the ?Update? button

6. This will save the configuration change to the selected icon



4 Installing Barcoders Android Server

Note: This is an incomplete guide at present

4.1 Installing the server

Install on Windows Server.

This server must have the following configured by TechHub:

- Port 50061 for TCP
- Access to internet for license validation and application download.

Install on the RDT server:

- Show hidden files and files with known extensions
- Create the following directories:
  - c:\logs
  - c:\data
- Set the PC power settings to not turn off.
- Open the firewall port.
  - Add inbound exception to port 50061 for TCP, named "Barcoders Android"
- Navigate to the "Android RFServer Files" folder.
  - Cut and Paste the "Barcoders" folder to the c:\Program Files directory.
  - Cut and Paste the "Barcoders.com" folder to the c:\ProgramData directory.
- Find your License folder and copy your two license files (userid.dat and license.xml) into the c:\ProgramData\Barcoders.com folder.
- Update the server to the latest version (see section #Updating the server

- Create some shortcuts on the desktop, running from C:\Program Files\Barcoders:
  - "Start Barcoders Updater", from "C:\Program Files\Barcoders\BarcodersUpdater.exe", set as Run as Administrator
  - "Start Barcoders Server", from "C:\Program Files\Barcoders\AndroidStudio\Android Connector.exe"
  - "Start Barcoders Admin", from "C:\Program Files\Barcoders\AndroidConnectorWindow\TermProtoTestWindow.exe"

4.2 Updating the server

- Ensure that the server is stopped (close the Admin and Server windows).
- Run BarcodersUpdater.exe as administrator (or use the "Start Barcoders Updater" shortcut.
- A window will display - click the button to update to that version.
- When complete, close the window.

4.3 Updating to a specific version

To revert to a prior version of the server

- Run CMD (DOS level) as administrator.
- Navigate to the c:\program files\barcoders directory.
- Type the following at the c:\ prompt

```
c:\BarcodersUpdater --license-id versions/{vers}
```

- Where {vers} is the required version number

e.g.

```
WIN-R
cmd
CTRL-SHIFT-ENTER
Confirm Administrator rights
cd c:\program files\barcoders
BarcodersUpdater --license-id versions/23.52
```

4.4 Configuring the Applications

In "C:\Program Files\Barcoders\AndroidStudio" there are two configuration files:

- appsettings.json - application log levels
- config.json - the applications being run by ABSStudio

To configure applications that can be run by the system, add entries to the config.json file, example below:

```
{
        "ProgramTasks": [
                {       "ExecutablePath": "C:/WCSTEST/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (TEST)",
                        "UsePipes": "True",
                },
                {       "ExecutablePath": "C:/WCSLIVE/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (LIVE)",
                        "UsePipes": "True",
                },
        ]
}
```

To change logging for the application, modify the Logging setting of the appsettings.json file, example below:

```
  "Logging": {
    "LogLevel": {
      "Default": "Info",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Error"
    }
  },
```

Values can be changed to

- Info
- Warning
- Error
- Debug

Note: We can clear (delete) the log file and it will recreate with no issues whilst still running.

Note: A single copy of the log file is added to by the server - it does not change or rotate every day or hour.

A script is being investigated to copy and to cleardown old scripts - details will be added here and in the support guide when available.

4.5 Updating Licenses

To update the license manually:

- Contact Barcoders
- Get updated license files.
- Drop into ProgramData
- Stop and restart ABS server.

To renew the license automatically:

- Contact Barcoders
- Barcoders renew license on server
- Stop and restart ABS server.

Note: Confirmation on some processes is being sought:

- Who in Aptean is in charge of renewals of Android licenses?
  - We want to be changing and updating licenses BEFORE expiration.
  - We want to plan downtime for this.
  - Customer Product Manager should handle this.
- Confirmation of the license renewal process with Barcoders requested.

4.6 Downloading the Android App

Side-load the Barcoders Android application on your device.

- Enter URL in Google Search or Chrome, or scan the barcode in above, or scan using Zebra Xing scanner app.

```
https://www.barcoders.com/app-release.apk
```

- Download APK (confirm if asked)
- Confirm name if asked.
- Open Downloaded APK (from open/install button or from notification drag down from top of screen or from Apps/Downloads or from Apps/Files).
- Confirm installing.
- If prompted confirm settings to allow installing unsafe apps, then return to * above.
- Install - confirm install location if requested (accept default)

The application will be named "Warehouse Terminal" and have a green Barcoders icon for its graphic. Drag to the home screen for easy access.

4.7 Configuring the Android App

- When you start the APK for the first time, it will display an IP address pointing to our RFServer here. You?ll need to change it to point to the IP address of your Android RFServer. To do this navigate to https://www.barcoders.com/make-server-barcode.php and generate a barcode PNG for the IP address of your Android RFServer. Print it out and scan it at the step above. You may have to exit and get back in for it to recognize the new IP address.
- When the menu come up, select the bottom option and test.

4.8 Zebra Enterprise Keyboard

Zebra provide a freely-downloadable enterprise keyboard that can be configured for function key presses.

https://www.zebra.com/gb/en/products/software/mobile-computers/enterprise-keyboard.html

https://www.zebra.com/us/en/support-downloads/software/productivity-apps/enterprise-keyboard.html

https://techdocs.zebra.com/enterprise-keyboard/4-1/guide/about/

https://www.zebra.com/us/en/support-downloads/software/productivity-apps/enterprise-keyboard-designer.html

There are two versions:

- Version 1.6 does not allow function keys. Only works on Android 5 below
- Version 4 allows function and special keys to be set up. Only works on Android 6 above.

Versions should be downloaded from the links above. Alternatively, these can be supplied by Aptean for sideloading.

4.8.1 General Configuration

After installation, the keyboard must be enabled.

- Go to Settings/System/Languages/On-screen Keyboard/Manage On Screen Keyboard.
- Enable Enterprise Keyboard.

On devices with a physical keyboard:

- Exit and return to any app that allows the popup keyboard and display it.
- Click the keyboard icon in the bottom right.
- Enable "Show Virtual Keyboard".

4.8.2 Zebra EKB v1.6

Guide: https://techdocs.zebra.com/enterprise-keyboard/2-0/guide/settings/

Function keys can be set up on this standard keyboard (on this or the later version).

- Go to Settings/System/Languages/On-screen Keyboard/Manage On Screen Keyboard.
- Click Enterprise Keyboard.
- Click Remapping
- Pick the key to be remapped on which layout - there are 4 on Numeric, 1 on Alpha and one on Symbol that can be remapped.
- Remap the key to output Unicode (U+24BB)

The remapped key will then display as an F on the keyboard.

You can then hit the F key - an F will appear, and then you can enter the number keys 1-9 and hit ENTER - this will enter the selected function key e.g. the remapped key, plus 8 plus ENTER will send F8.

4.8.3 Zebra EKB v4.x Configuration

Creation of full keyboard layouts can be completed through the Zebra EKB Designer. This tool allows creation of multiple keyboard layouts.

- Guide:
  - https://techdocs.zebra.com/enterprise-keyboard/latest/guide/settings/
  - https://techdocs.zebra.com/ekd/1-9/guide/about/
- Keycodes: https://techdocs.zebra.com/mx/keymappingmgr/
- Download:

https://www.zebra.com/us/en/support-downloads/software/productivity-apps/enterprise-keyboard-designer.html

You will need to deploy to each device. This can be through:

- The EKD application itself, device by device.
- Sideload, through ADB, device by device.
- Your chosen MDN software if it supports it.

The blow guide focusses on deploying a pre-created custom keyboard layout through the EKD software. You will need:

- The EKD software
- The keyboard layout
- A direct connection between the device and the PC running the software.

The sample SAP project works perfectly on supported devices - download from here: https://techdocs.zebra.com/ekd/latest/samples/.

- Get the keyboard layout project from URL or pre-created (MC3x-FKeys)
- Start EKD
- Load the project
- Select a layout
- Connect the device to your computer
- Send to the device

Alternatively, this designer keyboard can be sideloaded onto the device into the following area: /enterprise/device/settings/ekb/config

Once the keyboard is loaded, the layout must be enabled.

- Got to DataWedge
- Select the profile associated to your app (in this case BC-AndroidTerminal-v2)
- Click the Enterprise Keyboard section
- Ensure Enabled is checked.
- Click Select Layout
- Choose "qwertylayout" as the default under the loaded EKD layout project (MC3x-FKeys)

Now the standard keyboard will be the selected custom keyboard layout.

The keyboard supports

- Alpha uppercase and lowercase layouts
- Accessible numbers and symbols when long-pressing keys
- Symbols layout
- Numeric Only layout
- Functions layout, supporting
    - Numbers
    - Function Keys
    - ESC key
    - Cursor Keys
    - and more.

When displayed, the keyboard will start on alpha layout.

- Use the Caps or ABC key to switch between Alpha Uppercase and Lowercase.
- Use the 123 key to switch to numeric
- Use the FUNC key to switch to function and control keys (including cursor keys)
- Use the /*?# key to switch to symbol keys.

5 UG WCS Android Support Guide

aptean

Aptean

WCS Android Support Guide

WCS USER GUIDE

16/01/24 - 2.00
Reference: UG-WCS_AND

6 Installation Guide

Note: This is an incomplete guide at present

6.1 Installing the server

Install on Windows Server.

This server must have the following configured by TechHub:

- Port 50061 for TCP
- Access to internet for license validation and application download.

Install on the RDT server:

- Show hidden files and files with known extensions
- Create the following directories:
  - c:\logs
  - c:\data
- Set the PC power settings to not turn off.
- Open the firewall port.
  - Add inbound exception to port 50061 for TCP, named "Barcoders Android"
- Navigate to the "Android RFServer Files" folder.
  - Cut and Paste the "Barcoders" folder to the c:\Program Files directory.
  - Cut and Paste the "Barcoders.com" folder to the c:\ProgramData directory.
- Find your License folder and copy your two license files (userid.dat and license.xml) into the c:\ProgramData\Barcoders.com folder.
- Update the server to the latest version (see section #Updating the server

- Create some shortcuts on the desktop, running from C:\Program Files\Barcoders:
  - "Start Barcoders Updater", from "C:\Program Files\Barcoders\BarcodersUpdater.exe", set as Run as Administrator
  - "Start Barcoders Server", from "C:\Program Files\Barcoders\AndroidStudio\Android Connector.exe"
  - "Start Barcoders Admin", from "C:\Program Files\Barcoders\AndroidConnectorWindow\TermProtoTestWindow.exe"

6.2 Updating the server

- Ensure that the server is stopped (close the Admin and Server windows).
- Run BarcodersUpdater.exe as administrator (or use the "Start Barcoders Updater" shortcut.
- A window will display - click the button to update to that version.
- When complete, close the window.

6.3 Updating to a specific version

To revert to a prior version of the server

- Run CMD (DOS level) as administrator.
- Navigate to the c:\program files\barcoders directory.
- Type the following at the c:\ prompt

```
c:\BarcodersUpdater --license-id versions/{vers}
```

- Where {vers} is the required version number

e.g.

```
WIN-R
cmd
CTRL-SHIFT-ENTER
Confirm Administrator rights
cd c:\program files\barcoders
BarcodersUpdater --license-id versions/23.52
```

6.4 Configuring the Applications

In "C:\Program Files\Barcoders\AndroidStudio" there are two configuration files:

- appsettings.json - application log levels
- config.json - the applications being run by ABSStudio

To configure applications that can be run by the system, add entries to the config.json file, example below:

```
{
        "ProgramTasks": [
                {       "ExecutablePath": "C:/WCSTEST/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (TEST)",
                        "UsePipes": "True",
                },
                {       "ExecutablePath": "C:/WCSLIVE/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (LIVE)",
                        "UsePipes": "True",
                },
        ]
}
```

To change logging for the application, modify the Logging setting of the appsettings.json file, example below:

```
  "Logging": {
    "LogLevel": {
      "Default": "Info",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Error"
    }
  },
```

Values can be changed to

- Info
- Warning
- Error
- Debug

Note: We can clear (delete) the log file and it will recreate with no issues whilst still running.

Note: A single copy of the log file is added to by the server - it does not change or rotate every day or hour.

A script is being investigated to copy and to cleardown old scripts - details will be added here and in the support guide when available.

6.5 Updating Licenses

To update the license manually:

- Contact Barcoders
- Get updated license files.
- Drop into ProgramData
- Stop and restart ABS server.

To renew the license automatically:

- Contact Barcoders
- Barcoders renew license on server
- Stop and restart ABS server.

Note: Confirmation on some processes is being sought:

- Who in Aptean is in charge of renewals of Android licenses?
  - We want to be changing and updating licenses BEFORE expiration.
  - We want to plan downtime for this.
  - Customer Product Manager should handle this.
- Confirmation of the license renewal process with Barcoders requested.

6.6 Downloading the Android App

Side-load the Barcoders Android application on your device.

- Enter URL in Google Search or Chrome, or scan the barcode in above, or scan using Zebra Xing scanner app.

```
https://www.barcoders.com/app-release.apk
```

- Download APK (confirm if asked)
- Confirm name if asked.
- Open Downloaded APK (from open/install button or from notification drag down from top of screen or from Apps/Downloads or from Apps/Files).
- Confirm installing.
- If prompted confirm settings to allow installing unsafe apps, then return to * above.
- Install - confirm install location if requested (accept default)

The application will be named "Warehouse Terminal" and have a green Barcoders icon for its graphic. Drag to the home screen for easy access.

6.7 Configuring the Android App

- When you start the APK for the first time, it will display an IP address pointing to our RFServer here. You?ll need to change it to point to the IP address of your Android RFServer. To do this navigate to https://www.barcoders.com/make-server-barcode.php and generate a barcode PNG for the IP address of your Android RFServer. Print it out and scan it at the step above. You may have to exit and get back in for it to recognize the new IP address.
- When the menu come up, select the bottom option and test.

6.8 Zebra Enterprise Keyboard

Zebra provide a freely-downloadable enterprise keyboard that can be configured for function key presses.

https://www.zebra.com/gb/en/products/software/mobile-computers/enterprise-keyboard.html

https://www.zebra.com/us/en/support-downloads/software/productivity-apps/enterprise-keyboard.html

https://techdocs.zebra.com/enterprise-keyboard/4-1/guide/about/

https://www.zebra.com/us/en/support-downloads/software/productivity-apps/enterprise-keyboard-designer.html

There are two versions:

- Version 1.6 does not allow function keys. Only works on Android 5 below
- Version 4 allows function and special keys to be set up. Only works on Android 6 above.

Versions should be downloaded from the links above. Alternatively, these can be supplied by Aptean for sideloading.

6.8.1 General Configuration

After installation, the keyboard must be enabled.

- Go to Settings/System/Languages/On-screen Keyboard/Manage On Screen Keyboard.
- Enable Enterprise Keyboard.

On devices with a physical keyboard:

- Exit and return to any app that allows the popup keyboard and display it.
- Click the keyboard icon in the bottom right.
- Enable "Show Virtual Keyboard".

6.8.2 Zebra EKB v1.6

Guide: https://techdocs.zebra.com/enterprise-keyboard/2-0/guide/settings/

Function keys can be set up on this standard keyboard (on this or the later version).

- Go to Settings/System/Languages/On-screen Keyboard/Manage On Screen Keyboard.
- Click Enterprise Keyboard.
- Click Remapping
- Pick the key to be remapped on which layout - there are 4 on Numeric, 1 on Alpha and one on Symbol that can be remapped.
- Remap the key to output Unicode (U+24BB)

The remapped key will then display as an F on the keyboard.

You can then hit the F key - an F will appear, and then you can enter the number keys 1-9 and hit ENTER - this will enter the selected function key e.g. the remapped key, plus 8 plus ENTER will send F8.

6.8.3 Zebra EKB v4.x Configuration

Creation of full keyboard layouts can be completed through the Zebra EKB Designer. This tool allows creation of multiple keyboard layouts.

- Guide:
  - https://techdocs.zebra.com/enterprise-keyboard/latest/guide/settings/
  - https://techdocs.zebra.com/ekd/1-9/guide/about/
- Keycodes: https://techdocs.zebra.com/mx/keymappingmgr/
- Download:

https://www.zebra.com/us/en/support-downloads/software/productivity-apps/enterprise-keyboard-designer.html

You will need to deploy to each device. This can be through:

- The EKD application itself, device by device.
- Sideload, through ADB, device by device.
- Your chosen MDN software if it supports it.

The blow guide focusses on deploying a pre-created custom keyboard layout through the EKD software. You will need:

- The EKD software
- The keyboard layout
- A direct connection between the device and the PC running the software.

The sample SAP project works perfectly on supported devices - download from here: https://techdocs.zebra.com/ekd/latest/samples/.

- Get the keyboard layout project from URL or pre-created (MC3x-FKeys)
- Start EKD
- Load the project
- Select a layout
- Connect the device to your computer
- Send to the device

Alternatively, this designer keyboard can be sideloaded onto the device into the following area: /enterprise/device/settings/ekb/config

Once the keyboard is loaded, the layout must be enabled.

- Got to DataWedge
- Select the profile associated to your app (in this case BC-AndroidTerminal-v2)
- Click the Enterprise Keyboard section
- Ensure Enabled is checked.
- Click Select Layout
- Choose "qwertylayout" as the default under the loaded EKD layout project (MC3x-FKeys)

Now the standard keyboard will be the selected custom keyboard layout.

The keyboard supports

- Alpha uppercase and lowercase layouts
- Accessible numbers and symbols when long-pressing keys
- Symbols layout
- Numeric Only layout
- Functions layout, supporting
    - Numbers
    - Function Keys
    - ESC key
    - Cursor Keys
    - and more.

When displayed, the keyboard will start on alpha layout.

- Use the Caps or ABC key to switch between Alpha Uppercase and Lowercase.
- Use the 123 key to switch to numeric
- Use the FUNC key to switch to function and control keys (including cursor keys)
- Use the /*?# key to switch to symbol keys.

7 Android Usage Guide

7.1 Downloading the Android App

Side-load the Barcoders Android application on your device.

- Enter URL in Google Search or Chrome, or scan the barcode in above, or scan using Zebra Xing scanner app.

```
https://www.barcoders.com/app-release.apk
```

- Download APK (confirm if asked)
- Confirm name if asked.
- Open Downloaded APK (from open/install button or from notification drag down from top of screen or from Apps/Downloads or from Apps/Files).
- Confirm installing.
- If prompted confirm settings to allow installing unsafe apps, then return to * above.
- Install - confirm install location if requested (accept default)

The application will be named "Warehouse Terminal" and have a green Barcoders icon for its graphic. Drag to the home screen for easy access.

7.2 Configuring the Android App

- When you start the APK for the first time, it will display an IP address pointing to our RFServer here. You?ll need to change it to point to the IP address of your Android RFServer. To do this navigate to https://www.barcoders.com/make-server-barcode.php and generate a barcode PNG for the IP address of your Android RFServer. Print it out and scan it at the step above. You may have to exit and get back in for it to recognize the new IP address.
- When the menu come up, select the bottom option and test.

7.3 Using the app

7.3.1 Main Screen

When the application starts, you will be presented with an initial connection screen, which will then display all installed systems, usually as follows:

- (Customer) TEST system
- (Customer) LIVE system
- (Customer) LIVE system - site 2

You can click the Cancel button to exit the menu, and the back button to exit the app completely.

7.3.2 Settings

Application settings can be accessed by long-pressing anywhere on the screen.

This will show you a menu allowing you to modify the media volume or access the settings.

The following settings are supported (also indicating recommended values):

- **Voice** settings:
  - *Enable Voice* - default Disabled. Can be enabled to read all dialogue and error messages.
  - *Reading speed* - default 5. Recommended: 6
- **Display** settings
  - *Enable Image Popups* - default Disabled. Recommended: Disabled.
  - *Image Popup Duration (seconds)* - default 2. Recommended: 2.
- **Network** settings
  - *Connection Timeout (seconds)* - default 10. Recommended: 10.
- **Monitoring** settings
  - *Log Outbound Data* - default to Off, can set to On.

7.3.3 Keyboard

You can use the device's keyboard and scanner to enter data. You can also call up a pop-up keyboard by pressing anywhere on the screen.

The application will ensure that you can see the data being entered on the screen.

Clicking the Android Back button, pressing anywhere on the screen or entering test with the tick or Enter key on the keyboard will hide the keyboard.

The application will use the configured Android system keyboard. Note that the default Android keyboard does not support the following:

- Function keys
- CLR/ESC button
- Cursor buttons.

Zebra provide a freely-downloadable enterprise keyboard that can be configured for function, control and cursor key presses (depending on your Android version). This may be used on any device, with or without a physical keyboard.

Where the application uses function keys, the function keys may be used on any physical or popup keyboard.

The Android Back button (either physical or on the screen) is used as the CLR or ESC button.

Note: On some devices, the physical cursor UP and DOWN keys do not work as expected. In this case, use the Zebra Enterprise Keyboard for the cursor keys.

7.4 Scanner

The scanner on your device will be enabled by default for all fields.

Long-pull the trigger to scan barcodes - the scanner will stop once it has red the barcode.

If Voice is enabled, a short-pull of the trigger will repeat the last phrase that the device spoke.

7.5 Troubleshooting

When using the app on a wireless device, the application being used is subject to the connectivity of the device to the network and to the application. If there are any issues with the network connectivity or the app cannot be accessed, the application will display the main screen, showing "Error Connecting". Ensure that you have a network connection, then hit any key on any physical or popup keyboard to reconnect

8 Android Support Guide

8.1 Android Support Guide

References:

- Installing Barcoders Android Server
- Android Usage Guide

Filesystem:

- "C:\Program Files\Barcoders" - the ABSStudio files
- "C:\logs" - contains:
  - barcoders_upgrade.log - the upgrade log
  - ACDebugLog.txt - the server log
- "C:\data"

Shortcuts should be created on the desktop:

- "Start Barcoders Server" - the server itself - should already be running
- "Start Barcoders Admin" - similar to Wavelink Administrator - shows the connected devices and licenses.
- "Start Barcoders Updater" - for updating the version.

The ABSStudio Server runs in a command window that must remain open at all times. Do not close this window as this will stop all Android functionality for that customer.

When Android devices connect, the processes will be running under the "Start Barcoders Server" - it is from here that they can be killed with Task Manager.

The Android version of the RDT program is installed in exactly the same place as the existing RDT program RDTMenu1.exe and is called RDTMeni1_Android.exe.

8.1.1 Debug Settings

In "C:\Program Files\Barcoders\AndroidStudio" there are two configuration files:

- appsettings.json - application log levels
- config.json - the applications being run by ABSStudio

To configure applications that can be run by the system, add entries to the config.json file, example below:

```
{
        "ProgramTasks": [
                {       "ExecutablePath": "C:/WCSTEST/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (TEST)",
                        "UsePipes": "True",
                },
                {       "ExecutablePath": "C:/WCSLIVE/RDTMenu1_Android.exe",
                        "Name": "RDTMenu1 (LIVE)",
                        "UsePipes": "True",
                },
        ]
}
```

To change logging for the application, modify the Logging setting of the appsettings.json file, example below:

```
  "Logging": {
    "LogLevel": {
      "Default": "Info",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Error"
    }
```


```
},
```

Values can be changed to

- Info
- Warning
- Error
- Debug

Logging is exported to the command window of the application.

Logging is also sent to the server log.

Debug mode can also be enabled on the shortcut when running the ABSServer, e.g.

```
"C:\Program Files\Barcoders\AndroidStudio\Android Connector.exe" --debug
```

Once debug has been enabled, the log files may be searched for the terminal ID.

Note: We can clear (delete) the log file and it will recreate with no issues whilst still running.

Note: A single copy of the log file is added to by the server - it does not change or rotate every day or hour.

A script is being investigated to copy and to cleardown old scripts - details will be added here and in the installation guide when available.

Tracing terminal ID through Debug logs

- Find the terminal ID in WCS Maintenance, using the Connected Users screen - copy the terminal ID (large characters with hyphens for Android devices)
- Use this to filter the log file for everything for that terminal ID.

8.1.2 Versions

When running the system, the Android application and the Android Server connector must be the compatible versions.

To see the Android app version, get the user to:

- Go to Settings
- Apps
- Find Warehouse Terminal
- Check version

To check the Server version, check the latest updated version number in the upgrade log.

Note: Upgrades to our software requiring upgrade to Barcodes ABS Server is an exceptional process.

The install to both the Android ABS Studio Server and Android client must be rolled in or back manually.

Never upgrade the ABS Server or Android app unless required by an upgrade.

8.2 Troubleshooting Guide

8.2.1 Android device can't connect

- Check can other Wavelink devices connect
    - ♦ Proves system up and running

- Check error on device
    - When using the app on a wireless device, the application being used is subject to the connectivity of the device to the network and to the application. If there are any issues with the network connectivity or the app cannot be accessed, the application will display the main screen, showing "Error Connecting". Ensure that there is a network connection, then advise hit any key on any physical or popup keyboard to reconnect.
    - If displays selection of system then does not show WCS splash screen:
        - Check Android app set up properly in JSON config settings.
    - If system selected then splash screen displays, then Error connecting to WCS
        - Check WCS running - basic WCS issue

- Check Android Connector Window
    - If not already running, use "Start Barcoders Admin" shortcut on Desktop, or "C:\Program Files\Barcoders\AndroidConnectorWindow\TermProtoTestWindow.exe"

- Check command window open
    - Command window will be named "Start Barcoders Server", or "C:\Program Files\Barcoders\AndroidStudio\Android Connector.exe"
    - Check Task Manager that it is running
    - Start if not

- Check license
    - Look at license.xml file
    - Shows users and expiration.
    - Note: This file cannot be modified as it is checked. Modifications will fail the check and the system will be useless.
    - Compare total number of users in the Barcoders Admin window - if there are connected devices up to the maximum, no more users can connect.

- Check for versions
    - Confirm version of Android connector version in server logs or in Upgrade log.
    - Confirm Android version on device
    - Confirm validity and compatibility of versions - check local project documentation.
    - Advise update device for compatibility

- Check for errors in the server.
    - Check the command window.
    - Check the server log.

8.2.2 Every Android Device cannot connect

Perform checks above to ensure that there are no other issues. If no other issues are found, the ABS Studio Server likely has failed and must be restarted.

Stop the barcode Server command window - this will disconnect EVERY android user.

Warning: This will interrupt any users that are connected, to ANY site. It is the equivalent of stopping the Wavelink server/service on the machine. Use with caution and with the permission of the operation.

Start the barcode command window again.

Users can now reconnect.

9 WCS Labels Changes

The page is intended to provide a guideline on the construction and layout of the WCS Label formats.

10 Latest versions

```
<template>

<parameters version=?2?>

Carton  : <item name="Carton" format="trim" />

Packing Date: <item name="SysDate" type=?date? format="DD/MMM/YYYY" />

Order   : <item name="OrderNo" len="15" />

Customer: <item name="Customer" format="trim,ucase" />

<item name="AddrLine1" />

<item name="AddrLine2" />

<item name="Town" />

<item name="County" />

<item name="PostCode" />

<item name="Country" />

--------------------------------

Contents:

Stock            Batch       Qty

<group name="details" repeat="20">

<item name="Stock" len="15" /> <item name="Batch" len="8" /> <item name="Qty"
type=?numeric? format="####0" />

</group>

Label <item name="LabelNo" type=?numeric? format="####0" /> of <item name="BoxCount"
type=?numeric? format="####0" />

</template>
```

Version on TEMPLATE tag determines whether we are using old method with curly brackets or new method.

10.1 Groups

There is a requirement to use Deconsolidation for some promotional materials.

The process fits what they want to do for these items exactly, but this is only a small percentage of the stock being picked.

Need to check that Despatch Confirm cleans up Picking Container and Deconsolidation records for orders effectively, when using Final Media functionality.

Would require ?labels? produced from this process, which are more like mini packing lists:

```
Carton  : XXXXXXXXXXXXXXXXXXXX

Packing Date: DD/MMM/YYYY
```


```
Order   : XXXXXXXXXXXXXXXXXXX

Customer: XXXXXXXXXXXXXXXXXXX

Addr1

Addr2

Town

County

Country

Postcode

--------------------------------

Contents:

Stock            Batch       Qty

XXXXXXXXXXXXXXXX XXXXXXXXXXX XXXXX

<repeating up to X times>

Label X of Y
```

So we would produce as many labels as required to show the contents of the box, for the order.

Would need to extend label picking class so that you could specify sub groups, and how many times they could be printed.

```
<template>

<parameters version=?3?>

<text>Carton  : </text><item name="Carton" format="trim" />

<text>Packing Date: </text><item name="SysDate" type=?date? format="DD/MMM/YYYY" />

<text>Order   : </text><item name="OrderNo" len="15" />

<text>Customer: </text><item name="Customer" format="trim,ucase" />

<item name="AddrLine1" />

<item name="AddrLine2" />

<item name="Town" />

<item name="County" />

<item name="PostCode" />

<item name="Country" />

<text>--------------------------------</text>

<text>Contents:</text>

<text>Stock            Batch       Qty</text>
```


```
<group name="details" repeat="20">

<item name="Stock" len="15" /> <item name="Batch" len="8" /> <item name="Qty"
type=?numeric? format="####0" />

</group>

<text>Label </text><item name="LabelNo" type=?numeric? format="####0" /><text> of
</text><item name="BoxCount" type=?numeric? format="####0" />

</template>
```

Need to design the tags fully, to decide what formatting details we can use.

Need MaxLen, Len, Format (Dates, Text and Numbers), mid and trim functions, etc.

Note: Have extended this to full XML, even the plain text bit.

Problem with this: if this isn?t plain text, but printer language, each item print location is defined in the plain text. So:

```
^FO16,780^A0,30,26^FDProduct Code^FS

^FO310,780^A0,30,26^FDLot No^FS

^FO560,780^A0,30,26^FDCtns^FS

^FO656,780^A0,30,26^FDPces^FS

^FO16,830^A0,30,26^FD1234567890123456789O^FS

^FO310,830^A0,30,26^FD12345678901234^FS

^FO560,830^A0,30,26^FD00010^FS

^FO656,830^A0,30,26^FD00100^FS
```

This is ZPL to print some label headers, then print out the 4 items. The ?830? bit in the FO statement shows the y coordinate, where the line is printed. In our group, we need to specify the start (830) and the ?height? to be added to each line (lets say 80):

```
<template>

<parameters version=?3?>

<text>

^FO16,780^A0,30,26^FDProduct Code^FS

^FO310,780^A0,30,26^FDLot No^FS

^FO560,780^A0,30,26^FDCtns^FS

^FO656,780^A0,30,26^FDPces^FS

</text>

<group name="details" repeat="5" ystart=?830? yspacing=?80?>

<text>^FO16,</text><item name=?ypos? /><text>^A0,30,26^FD</text><item name=?StockCode?
/><text>^FS</text>
```


```
<text>^FO310,</text><item name=?ypos? /><text>^A0,30,26^FD</text><item name=?Batch?
/><text>^FS</text>

<text>^FO560,</text><item name=?ypos? /><text>^A0,30,26^FD</text><item name=?Cases?
/><text>^FS</text>

<text>^FO656,</text><item name=?ypos? /><text>^A0,30,26^FD</text><item name=?Units?
/><text>^FS</text>

</group>

</template>
```

So, ystart defines the start y coordinate, yspacing defines how much to add on though each group, ypos defines the dictionary item which will be set by the previous 2 parameters.

This could also be expanded to xstart, xspacing and xpos, for producing lists. We would maybe also have to define the directionality - horizontal or vertical.

10.2 1.3 Barcodes

Need to include some information regarding barcode types. Most are just plain text producers e.g. CODE-39, but CODE-128 and EAN-128 allow changes based on content. For example:

12345678 can be codes as 128C

SSN12345678 can be coded as 128A or B, but the barcode is twice the length.

The barcode can be set to be in 128C, but the alpha portions can be set to be in 128A or B, with control sequences.

The language property on the parameters tag could control the characters used to change code type:

```
<parameters version=?3? language = ?ZPL?>
```

For example, a UCC barcode in zebra format:

```
^FD>;>8021234567890123415010203l0>6ROT45678>8>5370224^FS
```

So the format could be:

```
<text>^FD>;>802</text>

<item name=?StockCode? />

<text>15</text>

<item name=?SellBy? />

<text>10>6</text>

<item name=?CustBatch? />

<text>>8>537</text>

<item name=?Qty? fomat=?9999? />

<text>^FS</text>
```

That could cause problems to the parser though, so it might be better if:

```
<text>^FD</text>
```


```
<text type=?EAN128C?>02</text>

<item type=?EAN128C? name=?StockCode? />

<text>15</text>

<item name=?SellBy? />

<text>10</text>

<item type=?EAN128C? name=?CustBatch? />

<text>37</text>

<item name=?Qty? fomat=?9999? />

<text>^FS</text>
```

The property ?EAN128C? or ?CODE128C? on text tags will start the barcode ?>;? and set the code type C ?>8?

The property ?EAN128C? or ?CODE128C? on item tags will check the element for alphabetic characters and encapsulate the alpha text with ?>6? at the start and ?>8? at the end.

Still need some mechanism of terminating variable-length items though. Possibly send AI through as property, as follows:

```
<text>^FD</text>

<text type=?EAN128C? />

<item type=?EAN128C? ai=?02? name=?StockCode? />

<item type=?EAN128C? ai=?15? name=?SellBy? />

<item type=?EAN128C? ai=?10? name=?CustBatch? />

<item type=?EAN128C? ai=?37? name=?Qty? format=?9999? />

<text>^FS</text>
```

AI property would control putting the AI at the start of the printed field.

Could do same when checking for lowercase characters as well (change from B to A, etc).

Possibly would need a similar label printed from WMS (Pick Container Enquiry screen).

10.3 1.4 Optional Sections

I can see a need to modify this so that sections or items are only printed if certain criteria are met.

So:

```
<template version="3" language="ZPL">

<text>Carton  : </text><item name="Carton" format="trim" />

<text>Packing Date: </text><item name="SysDate" type=?date? format="DD/MMM/YYYY" />

<text>Order   : </text><item name="OrderNo" len="15" />

<text presentif=?customer?>Customer: </text>
```


```
<item name="Customer" format="trim,ucase" presentif=?customer? />
<item name="AddrLine1" />
<item name="AddrLine2" />
<item name="Town" />
<item name="County" />
<item name="PostCode" />
<item name="Country" />
<text>---------------------------------</text>
<text>Contents:</text>
```

etc

or

```
<template version="3" language="ZPL">
<text>Carton  : </text><item name="Carton" format="trim" />
<text>Packing Date: </text><item name="SysDate" type=?date? format="DD/MMM/YYYY" />
<text>Order   : </text><item name="OrderNo" len="15" />
<group name=?address? presentif=?customer?>
<text>Customer: </text>
<item name="Customer" format="trim,ucase" />
<item name="AddrLine1" />
<item name="AddrLine2" />
<item name="Town" />
<item name="County" />
<item name="PostCode" />
<item name="Country" />
</group>
<text>---------------------------------</text>
<text>Contents:</text>
```

etc

The ?presentif? parameter controls whether the item or group contents should be displayed at all. Specifying a field name simply checks if there is a value in that string (or zero). Or you could specify a condition (?qty<>0?).