Honeywell

Guided Work Solutions

Scraping Plug-In 1.4

Product Description

Disclaimer

Honeywell International Inc. ("HII") reserves the right to make changes in specifications and other information contained in this document without prior notice, and the reader should in all cases consult HII to determine whether any such changes have been made. HII makes no representation or warranties regarding the information provided in this publication.

HII shall not be liable for technical or editorial errors or omissions contained herein; nor for incidental or consequential damages resulting from the furnishing, performance, or use of this material.

Web Address: automation.honeywell.com

Patents

For patent information refer to hsmpats.com

TABLE OF CONTENTS

CHAPTER

1 INTRODUCTION

This document is intended to familiarize the user with the basic functionality of the Guided Work Solutions (GWS) Scraping Plug-In.

With GWS Scraping Plug-In, developers can create powerful voice workflows over their existing emulation solutions just by creating plain code files. Emulated screens are defined in GWS Scraping Plug-In as different Scraping States. Each code file contains the parametrization of the emulated screen via the implementation of the most fitting State.

Available Scraping States are:

State Type	Description
Login	A state where the login password is asked to the user. The username is retrieved automatically from the device or VoiceConsole if associated. An additional message can be spoken to the user before asking for password.
Logout	A state where logout confirmation is asked to the user. A logout reason can be asked to the user if configured.
Menu	A state to define a generic Menu screen where the user can select one of multiple options presented for the screen. Menu options are spoken to the user one by one.
Message	A state where a message is spoken to the user. After the message is spoken a defined action is executed on the emulated screen. An additional message can be spoken to the user after executing the defined command if it is enabled.

State Type	Description
Selection	A state where the user is asked to scan one or more items. A message is spoken to the user to indicate what should be scanned. An additional message can be spoken to the user after scanning the item if it is enabled.
WorkInformation	A state where the initial work information is retrieved from the user: • Initial location of the user. • Vehicle type being used. • Area where working.
HandlingUnitConfirmation	A state where the required handling unit is spoken to the user.
PickingStart	A state where initial information of the assigned Picking operative is spoken to the user: • Order number assigned. • Total number of picks of the assigned order. • Total volume for the goods to be picked in the assigned order.
PickingLine	A state where the user is directed through a single picking line. In this state the user is asked to confirm: • Location by scanning barcode, spelling check digits or simple voice confirmation. • Source carrier from which the goods are picked (optional). • Product ID that is being picked (optional). • Quantity picked by spelling the number or simple voice confirmation. • Carrier where leaving the product (optional).

State Type	Description
PickingLineCancel	A state where the user is asked to confirm the picking line cancellation reason. The user can be asked to confirm whether there is an error to be corrected in the location or not.
PlaceInDestination	A state where the user is asked to leave the carrier in deposit location. The user can scan the location barcode or spell check digits. The user can be asked to confirm the printed carrier label by spelling a defined number of digits of the code.
State	A state where interaction with the user is not required. When triggered, the state executes predefined commands on host server. All states defined above inherit this state to add an “Execute” method that allows executing extra commands after all fields are filled.

GWS Scraping Plug-In provides the user with a series of tools to model emulated screens and shape the voice workflow:

- **State properties**: Each State offers predefined properties that help to model the screen behavior (for instance: whether product validation is required or not).
- **Configuration settings (Scraping Settings)**: Each State has a series of related settings that help to shape further the voice workflow (for instance: whether product validation is made automatically, or check digits are asked of the user).

GWS Scraping Plug-In solution can be deployed in two ways. This document covers both scenarios.

- Using the GWS Service.
- Embedded in the GWS App without using the GWS Service.

The current integration mechanism is based on Telnet Terminal emulation. All Telnet communication integration is processed through Rebex.TerminalEmulation third-party library (https://www.rebex.net/terminal-emulation.net/).

Getting Help

Additional Documentation

Additional documentation may be found in your product package and on online partner portals.

Find most Honeywell Voice technical documentation at help.honeywellaidc.com.

Honeywell Voice Reseller Services

If you purchased equipment or services through a Honeywell Voice reseller, please contact your reseller first for support or to purchase a support plan.

Honeywell Voice Technical Support

Submit incidents or questions to honeywell.custhelp.com or contact Honeywell Technical Support Center:

- **Americas**
  Email: VoiceTechnicalSupport@Honeywell.com
  Phone: +1(866) 862-7877
- **Europe, Middle East, Africa**
  Email: VoiceTechnicalSupport@Honeywell.com
  Phone: +44 (0) 1344-65-6123
- **Rest of World**
  Email: VoiceTechnicalSupport@Honeywell.com
  Phone: +1 (412) 376-9384

To report support incidents or ask technical questions for other Honeywell devices, visit honeywell.com/PSStechnicalsupport.

Honeywell Voice Customer Service

For order placement or customer service inquiries:

- **North America, Latin America**
  Email: VoiceCustomerServiceAmericas@Honeywell.com
  Phone: +1(866)862-6553
- **Europe, Middle East, Africa, Turkey**
  Email: voicecustomerserviceEMEA@honeywell.com
  Phone: +44 (0) 1698-915777
- **Japan**
  Email: csjapan.pss@honeywell.com

Phone: +81-3-6730-7344

- **Brazil**
  Email: ACSHSMCentraldepedidos@honeywell.com
  Phone: +55 (31) 2391-5600
- **Asia Pacific**
  Email: VoiceCustomerServiceAPAC@honeywell.com
  Phone: +44 16989 15777

Honeywell Voice Hardware Repair

For returns or to check the status of a Return Material Authorization (RMA) for Voice hardware products:

- **Americas**
  Email: VoiceRMA@Honeywell.com
  Phone: +1 (866) 417-6988
- **Europe, Middle East, Africa**
  Email: VoiceEMEARMA@honeywell.com
  Phone: +1 (866) 417-6988
- **Rest of World**
  Email: VoiceRMA@Honeywell.com

For returns or to check the status of an RMA for other Honeywell hardware products, visit the SPS RMA portal: sps-support.honeywell.com/s/pss/pss-rma

CHAPTER

2

AVAILABLE FEATURES SUMMARY

- Mapping of emulation screens to voice dialogs: Define each screen GWS Scraping State and shape the defined voice workflow through State properties and configuration settings.
- Support for custom scripting:
    - Parametrized screen behavior through State properties (for instance: detect if product validation is required or not)
    - Input validation through Validation Scripts
    - Screen processing
- Telnet & SSH communication support through Rebex.TerminalEmulation third-party library.
- Support for different emulation types:
    - VT52
    - ANSI
    - VT100 and successors
    - Xterm terminal
    - IBM 5250
- Support for most common server authentication methods.
- Data validation on Server side.

Migrating from VoiceExpress

As the GWS Scraping Plug-In is a whole new approach to Telnet interpretation, there is no direct migration path for existing VoiceExpress implementations. Refer to the Getting Started section of this document for a high-level description of the implementation process and to the *Guided Work Solutions Scraping Plug-In Developer Guide* for a more detailed description on how to create a new Context targeting the existing emulation system.

CHAPTER

3

REQUIREMENTS

Developing for GWS Scraping requires the following:

- An IDE capable of creating .NET Framework / .NET Standard libraries.
- Honeywell Recommends Microsoft Visual Studio 2022 17.5 or later to start with the offered project templates.

Voice Workflow is already developed for every State, so Honeywell VoiceInterface Objects knowledge is not required.

For further information about Voice Workflow and GWS Scraping Context implementation, please refer to *Guided Work Solutions Scraping Plug-In Developer Guide*.

CHAPTER

4

GETTING STARTED

Step 1: Gather Information

The first step when creating a GWS Scraping implementation is to make sure that you have all the relevant information about the workflow to be modeled.

The following is some of the information to gather. It is not necessarily a complete list. The more information you gather, the more reliable your implementation is.

- The inputs and labels that appear on the screen,
- How the user interacts with those,
- What validations are performed over the input data,
- Variations of a screen,
- Messages that are shown to the user, etc.

Step 2: Select States

The next step is to select the most fitting *State* for each screen. This depends on the desired behavior. Selecting a *State* provides the required properties to be fulfilled by the developer. For instance, implementing a *ILogin* State requires *Operator* and *Password* properties to be defined.

Step 3: Create Code Project

The next step is to create a new code project to implement the *Context* of the GWS Scraping Plug-In. This can be done by using the provided *Honeywell GWS Connector Scraping Context* template. The template creates a project with base code structure, including implementation examples of:

- Connection Management (Connect / Reconnect / Disconnect).
- Emulation current State identification.
- State implementation.
- Property implementation.

Step 4: Modify Context

Lastly, modify Context as required by your terminal emulation environment and implement all selected States and its properties. Implementing each State property means to specify how the interaction is done with the emulation session to read or write its value. To ease the development, GWS Scraping provides a series of Code helpers for each integration mechanism that allow to auto-generate interaction code just by decorating the properties with one of the available Attributes.

CHAPTER

5

SYSTEM LOGGING

Once you configure and execute GWS App with the Scraping Plug-In, you can see different types of logs as you use the application.

When using GWS Scraping Plug-In three types of logs regarding the terminal emulation are generated:

- Monitor: Information related to *Monitor* is included in a single file named "TelnetMonitor.txt". This file includes the current screen being processed.
- Session: In these files you are all interactions between the client (GWS Scraping Plug-In) and the server (Host Server). These files can be found inside the Logs folder as “TelnetSession_yyMMdd_hhmmss.txt”.
- Validation: These files contain a “screen shot” of the *Monitor* when a validation or any terminal emulation action related error occurs.

When working with GWS Service, the Logs path is relative to the installation path: GWS Connector Service/Logs/yyyy/MM/dd/DeviceName.

When working in Embedded mode, the Logs path is relative to the device's Documents folder: Documents/Logs/yyyy/MM/dd.

CHAPTER

6

GWS SCRAPING SAMPLE

A sample implementation is included to test and enhance the understanding of Contexts in the GWS Scraping Plug-In. This context, represented by the GWS-Scraping.Sample project, is used to integrate with Telnet terminal screen emulation. Additionally, it includes a Mock Server to emulate Telnet messages that respond to this implementation. This is explained in more detail below.

Using GWS-Scraping with the Sample Context

Refer to the information below to configure the GWS Scraping Plug-In with the Sample context for deployment with either GWS Service (Server Mode) or GWS App (Embedded Mode). Choose the appropriate option depending on implementation.

Configure the GWS Service

1. Copy:
    - GWS-Scraping/Service/<version>contents to the folder where GWS Service is located.
    - GWS-Scraping.Sample/Service/GWS-Scraping.Sample-<version>-Service to the same location.
    - GWS-Scraping.Sample/Config files to the same location.
2. Configure:
    - Replace service configuration with GWS-Scraping.Sample/Service/Honeywell.GWS.Connector.exe.config file (or use it as reference to complete your current config file)

Android devices

Install the GWS App

1. Install the GWS App using the provided APK file.
2. Copy the license file into the Downloads folder on the Android device.

3. Copy the set of GWExternalResources.resx files matching the plug-in to be used into the Documents folder on the Android device. Notice that if there are other Honeywell Guided Work applications installed on the device, you can enable coexistence by creating a folder called ‘com.honeywell.GWSApp’ in the Documents folder and placing all the files and folders there rather than directly in the Documents folder.
4. Run the application and grant all required permissions. Verify that the application has been licensed.

Target GWS Service

Go to the **Settings** > **GWS Connector** settings screen and configure settings to target to the GWS Service installed in the previous section. Ensure the Server workflow is selected and provide the required information:

- **Backward compatibility**: disable as GWS Service is fully compatible.
- **Server Secure Connections**: disable if not using certificates on GWS Service.
- **LUT / ODR / Error hosts and ports**: provide the information for the host address and ports configured in the Entry Point of the GWS Service.
- **Encoding**: only for compatibility with older devices
- **Connection Timeout**: this should not be changed from default value

Use Embedded Mode

1. Copy:
    - GWS-Scraping/App/<version> to Documents/plugins. If a previous version of this folder exists, delete the contents first.
    - GWS-Scraping.Sample/App/GWS-Scraping.Sample-<version>-App to the same location.
    - GWS-Scraping.Sample/Config files to Documents folder.
    - Copy GWS-Scraping/App/GWExternalResources contents to the Documents folder on the device.
2. Configure:
    - Run the app, go to settings, change the workflow to run plug-ins and select the right plug-in.
    - Make sure you set the 'Server' property correctly: "user:password@<ip_address>:23", which refers to the server and the port on which the Mock Server runs.
    - Configure Scraping:ContextType as: Honeywell.GWS.Connector.Library.Scraping.Sample.Context, Honeywell.GWS.Connector.Library.Scraping.Sample
    - Configure Scraping:SettingsFileName as: scraping.yaml.

- Alternatively, edit GWS-Scraping.Sample/OneTimeStartupSettings.config file and place it in the Download folder on the device.

A700x Devices

In Talkman devices, you must use the ToolVad utility to create a new VAD package. This VAD includes all the necessary files to the original VAD.

For GWS Service, the tool is used to add the resource files to the original VAD.

in contrat, in Embedded mode, the tool is used to add the plug-in, along with the corresponding resource files.

The followin guide covers both Service and Embedded modes. It explains how to generate the VAD with all required files and outlines the steps to create the task and the task package.

Setting up ToolVad

1. If you haven't done it yet, create a new folder (for example c:\ToolVad) and place the latest version of the ToolVad utility in that folder.
2. Place the provided original GWS App VAD task file in the same folder.

Refer to the *GWS ToolVad User Guide* for more information about this tool.

Target GWS Service:

1. Create a new folder with the name of the plug-in, where we put all required files.
2. Copy the Resources folder from the plug-in into that new folder. The resulting file structure looks like this:

   /ToolVad/<plugin-folder>

   |-- **resources**

   | |-- GWExternalResources.resx

   | |-- GWExternalResources.es.resx

   | └ - ...

   |-- GWS_App-X.Y.vad

   └ - ToolVad.exe
3. Once you have copied all the elements, open a terminal in the ToolVad folder and run the following command:

   .\ToolVad.exe -v GWS_App-X.Y.vad -r <plugin-folder>\resources -o GWS_App-X.Y-<plugin-name>.vad
4. The utility creates a new VAD file containing the embedded resources.

Import Task and Create Task Package

1. Open VoiceConsole and import the new task file by selecting **VoiceConsole** > Operator Management > **Tasks** > **Import task** and choose the VAD file created with ToolVad.
2. Create a task package by selecting **VoiceConsole** > **Operator Management** > **Task packages** > **Create new task** package and select the task from the previous step.
3. Configure the Task Settings tab as follows:
    - Secure Connections: Unchecked (disabled) unless the GWS Service has been configured to use secure Entry points
    - LUTHost
    - LUTPort
    - ODRHost
    - ODRPort
    - ErrorHost
    - ErrorPort
    - EncodingName
    - ExecutionPlatformInformation
    - ConnectionTimeout
    - Other fields can be left with their default valueSelect the Device Settings tab and use the Advanced Settings and make the following changes:
4. Save the task package.

To modify any phonetic substitution after the task package is created:

1. Open the task package and select 'Edit this task package' from the left pane.
2. Select the Phonetic Sub tab and configure the values for the VCOMMANDs the plug-in is using.
3. Save the task package.

Use Embedded Mode

1. Create a new folder with the name of the plug-in (<plugin-folder>), where we put all required files.
2. Copy the resources folder from the plug-in into that new folder.
3. Copy GWS-Scraping.Sample/Config folder contents inside resources folder.
4. Inside <plugin-folder> create a new folder where we put the plug-in binaries (<plugin-binaries-folder>).
5. Copy GWS-Scraping/App/<version> folder contents inside <plugin-binaries-folder> folder.

6. Copy GWS-Scraping.Sample/App/GWS-Scraping.Sample-<version>-App folder contents to the same location. The resulting file stricture looks like this:

```
/ToolVad/<plugin-folder>
 |-- resources
 |  |-- customers
 |  |-- scraping.yaml
 |  |-- locations
 |  |-- GWExternalResources.resx
 |  |-- GWExternalResources.es.resx
 |  └- ...
 |-- <plugin-binaries-folder>
 |-- GWS_App-X.Y.vad
 └- ToolVad.exe
```

7. Once you have copied all the elements into the folder, open a terminal in the ToolVad folder and run the following command:

   .\ToolVad.exe -v GWS_App-X.Y.vad -r <working-folder>\resources -p <working-folder>\<plugin-binaries-folder> -o GWS_App-X.Y-<plugin-name>.vad

8. The utility creates a new VAD file containing the embedded plug-ins.

Import the Task and Create a Task Package

1. Open VoiceConsole and import the new task file:
   VoiceConsole > **Operator Management** > **Tasks** > **Import task** and choose the vad file created with ToolVad.
2. Create a Task Package:
   VoiceConsole > **Operator Management** > **Task Packages** > **Create new task packag**e and select the task from the previous step.
3. Configure the 'Task Settings' tab as follows:
   - Workflow: Change the text from 'Server' to 'Plugin'.
   - Plug-In: Type 'Scraping (Server)' in this field.
4. Configure the 'Phonetic Substitution' values for VCOMMANDS that the plug-in is using. The key must be 'VCOMMANDXX', where the X is the number of the command.
5. Configure the 'Advance Settings'
   - Disable voice commands not being used (if not using Pick Up and Go recognizer) by adding 'DisableVocabWord_VCOMMANDXX=1' where X is the number of the command (from 1 to 99).
   - Add the plug-in related parameters by using the setting key preceded by 'AdditionalParam_' prefix.

- o AdditionalParam_Server=<user>:<password>@<ipadress>:<port>
- o AdditionalParam_ScrapingSettingsFileName=scraping.yaml
- o AdditionalParam_ ScrapingContextType=Honeywell.GWS.Connector.Library.Scraping.Sample.Context, Honeywell.GWS.Connector.Library.Scraping.Sample

6. Complete Create Task Package wizard.

> **IMPORTANT**
> Depending on the version of VoiceConsole and how VoiceConsole is configured, special characters such as '\', '/', ':', '=', '.'used for the server parameter may need to be encoded per the table below. The equal sign (=) can only be used after the parameter name. Any additional uses of = in the value must be encoded.
>
> To enable special character entry, edit **System Configuration** within VoiceConsole. Under **Other Configuration**, set **Task Package - Advanced Settings Type** to **GWS Connector Embedded Mode**.

Example:

AdditionalParam_Server=Server=100.100.100.100

would be encoded as

AdditionalParam_Server=Server_chr06100_chr046100_chr046100_chr046100

Value	Symbol	Description
_chr032	<space>	Space
_chr044	,	Comma
_chr045	-	Dash
_chr046	.	Point
_chr047	/	Slash or divide
_chr058	:	Colon
_chr061	=	Equals
_chr064	@	At sign
_chr092	\	Backslash

Values are taken from ASCII code character list using decimal value at the third position (i.e., _ chrXXX). You can check any reference of ASCII table “printable” chars from 032 to 125 (126 is a reserved character for internal use). For more information, see https://www.ascii-code.com/characters/printable-characters.

Sample Advanced Settings:

- Using encoded characters

Advanced Settings

```
DownLoadAllTemplates=1
DisableVocabWord_stop=1
DisableVocabWord_select=1
AdditionalParam_Server=http_chr058_chr047_chr047100.100.100.100_chr0588002
AdditionalParam_DeviceLogEnabled=true
```

- Without using encoded characters

Advanced Settings

```
DownLoadAllTemplates=1
DisableVocabWord_stop=1
DisableVocabWord_select=1
AdditionalParam_Server=http:\\100.100.100.100:8002
AdditionalParam_DeviceLogEnabled=true
```

Mock Server: Simulating Telnet Messages

A GWS-Scraping Sample Mock Server is a project designed to simulate Telnet messages. It is used for testing the GWS Scraping solution with the Sample context. In this section, we explain how to run the server and highlight its limited functionality.

Getting Started

Requirements

Before you can run the Mock Server application, ensure that your system meets the following requirements:

1. **Operating System Compatibility**: The Mock Server binary provided is meant to be run on Windows, but as source code is provided, it is possible to target the Linux operating systems by building it from the source. See Building the Mock Server.
2. **.NET 7 Runtime**: The application is developed using .NET 7. To check if you have the .NET 7 Runtime installed, open a terminal or command prompt, and run the following command:
   dotnet --version
   If the command returns a version number 7.x.x or higher, you already have the .NET 7 Runtime installed, and you can proceed to use the application. If the .NET 7 Runtime is not installed, you can download and install it from the following URL:
   https://dotnet.microsoft.com/en-us/download/dotnet/7.0

3. **Configuration**: Ensure you have edited the configuration file as described in the Using GWS-Scraping with the Sample Context section above, specifying the server connection value.
4. **Firewall**: Make sure that Firewall configuration allows connections to the Mock Server application on TCP port 23.
5. **Executable**: You should have the Mock Server executable file ready for launch. You can obtain the executable from the official distribution package (GWS-Scraping-<version.\GWS-Scraping.Sample\MockServer\GWS-Scraping.Sample-<version>-MockServer) or build it from the source code. See Building the Mock Server.

Once you've met these requirements, you can proceed to launch and use the Mock Server.

Running the Mock Server

To start the Mock Server, locate the executable and run it by double-clicking, or use a terminal or command prompt and run the following command (please replace <PATH> with the actual file path where the executable is located on your system):

```
<PATH>\Honeywell.GWS.Connector.Library.Scraping.Sample.MockServer.exe
```

To show raw data communication, use this alternative command:

```
<PATH>\Honeywell.GWS.Connector.Library.Scraping.Sample.MockServer.exe –show-rawdata-communication
```

The server then listens for incoming Telnet connections on a predefined port (port 23) and responds to client requests with predefined messages.

Building the Mock Server

If you need to generate the Mock Server from the source code, follow these steps:

1. **Open the Project with a Code Editor:** Use a code editor, such as Visual Studio, Visual Studio Code, or any other of your choice, to open the GWS-Scraping.Sample solution.
2. **Compile the Project:** Within the code editor, compile the GWS-Scraping.Sample.MockServer project. This generates the executable from the source code.
3. **Obtain the Executable:** Once the compilation is successfully completed, find the server executable in the project folder or in a specific location configured in your development environment.

With these steps, you have generated the server executable from the source code and are ready to run it.

Limitations

The Mock Server has some limitations.

No Support for Incorrect Logins

The server does not support the concept of incorrect logins. Regardless of the username and password provided by the client, the server always responds as if the login is successful. This limitation is essential to keep in mind when testing authentication-related scenarios.

Fixed Responses

The server provides fixed responses regardless of the type of vehicle or any other parameter the client may choose. This means that the server's behavior is not dynamic and does not reflect real-world variability. The server always replies with the same two picking lines and destination location.

Lack of Support for Certain Commands

The server does not support the following commands that involve communication with the Telnet server:

- Leave: The server does not handle the "leave" command, which is used to leave the current order assignation.
- Cancel: The server does not support the "cancel" command, which is used to cancel the current line.
- Skip line: The server does not handle the “skip line" command, which is used to skip the current line.
- Close carrier: The server does not handle the “close carrier” command, which is used to close the carrier.

The End session command is the only command supported.

Lack of Logout Reasoning

When the user terminates the session with the “End Session” command, the server does not provide a menu to select the reason for ending the session. The session simply ends without any additional information.

No Control Over Session duration

The application does not enforce any limitations on session duration. The client can remain connected for as long as they wish, and the server does not implement any timeout or session management features.

Server Logs

The mock server does not generate any log files. Instead, all the information can be viewed on the terminal where it is running. If we have started the server without any arguments, it only displays the following information:

- Telnet client connections and disconnections
- Password: Password inserted by the user
- Vehicle: Vehicle selected
- Picked: Quantity picked
- Carrier: Value inserted for the carrier
- Location: Value inserted for the destination location

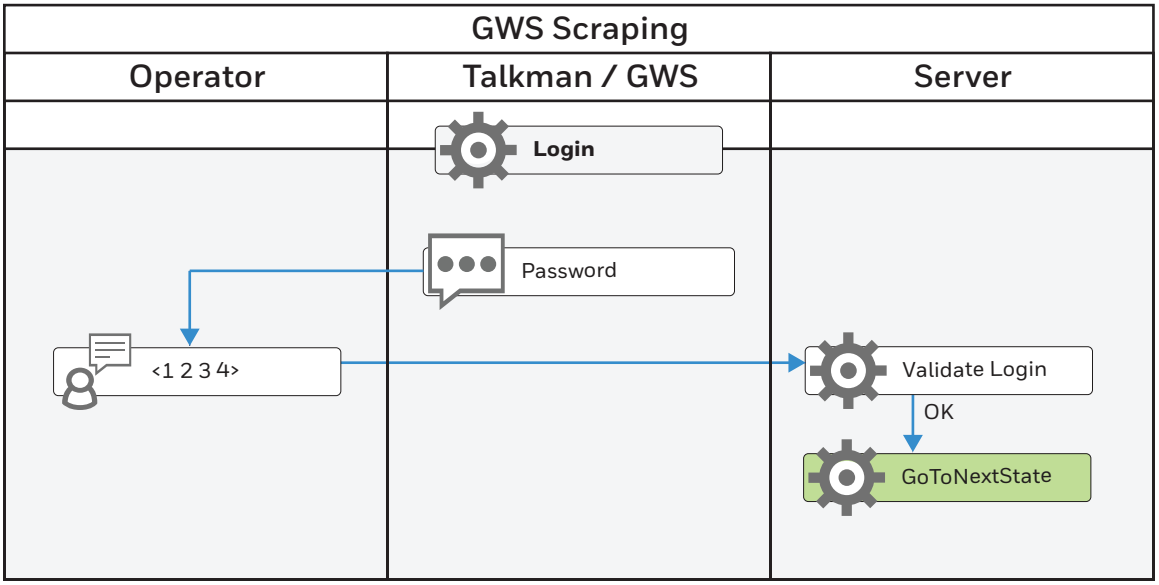
In case you have added the argument "--show-dataraw-communication" when launching the server executable, you also can see the exchanged data between the Plug-In and the server in hexadecimal format.

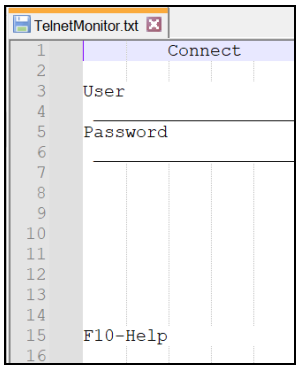
Sample Walkthrough

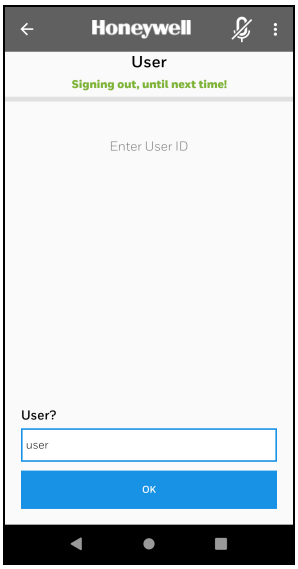
The Sample walkthrough provides a step-by-step guide to the screens and actions that the user can perform while navigating through the server demo, considering its limitations and the lack of authentication in the system.

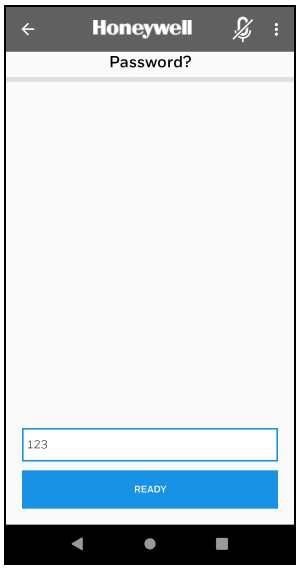
Login (Login State)

In this screen, the user can enter any username and password since the server does not perform authentication checks.

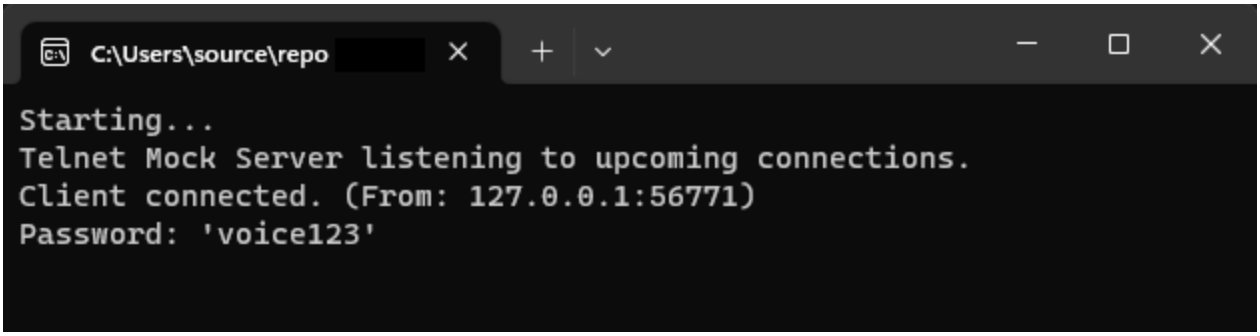






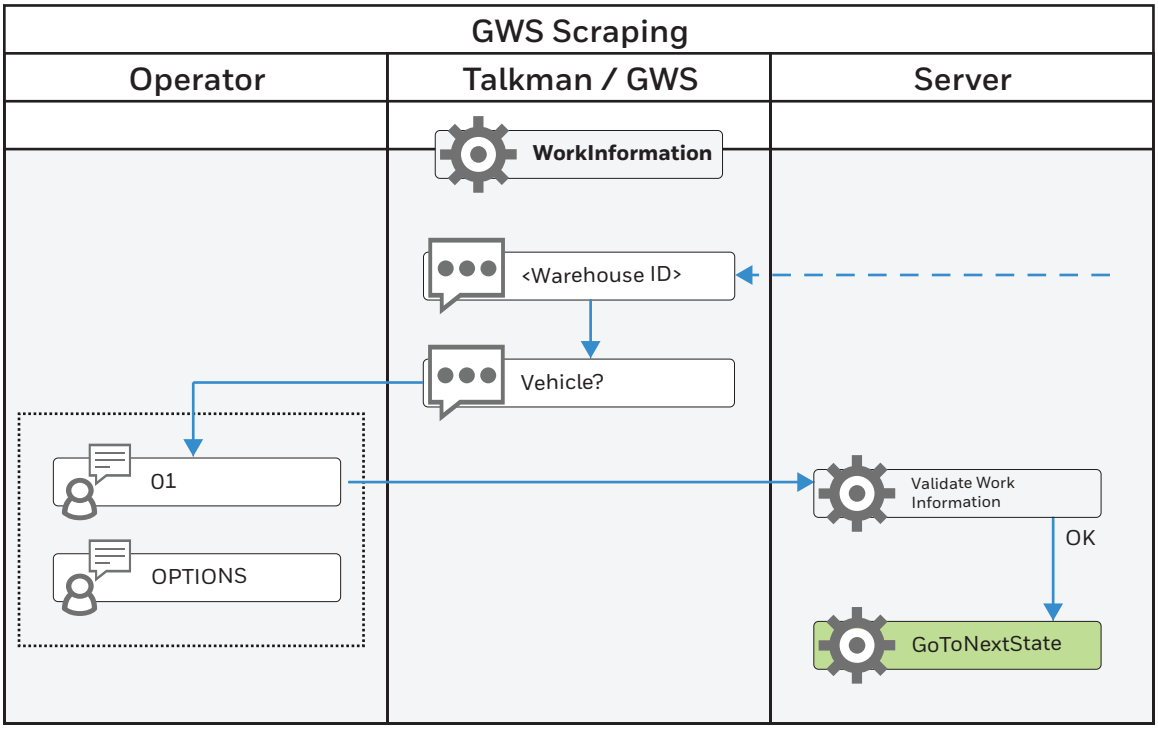


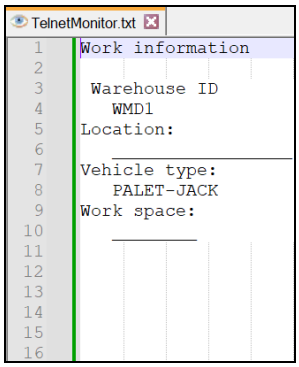
The server receives the password along with the prefix defined in the scraping.yaml file:

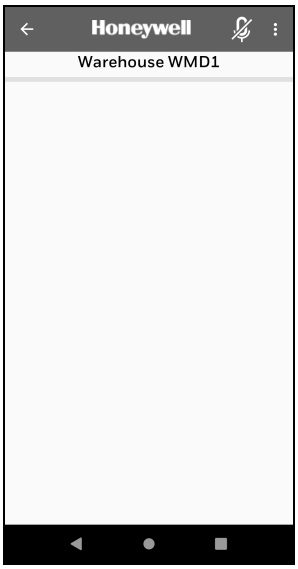


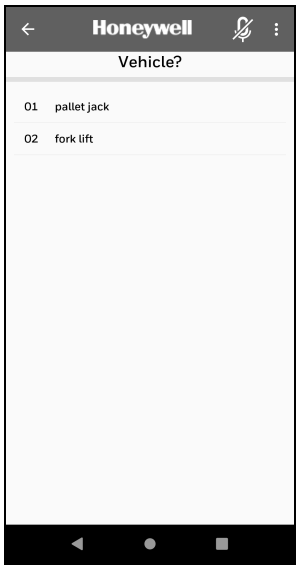
Work Information (WorkInformation State)

After logging in, the information about the current warehouse is displayed and the user is asked to select a vehicle type. Area and Location information is sent automatically.





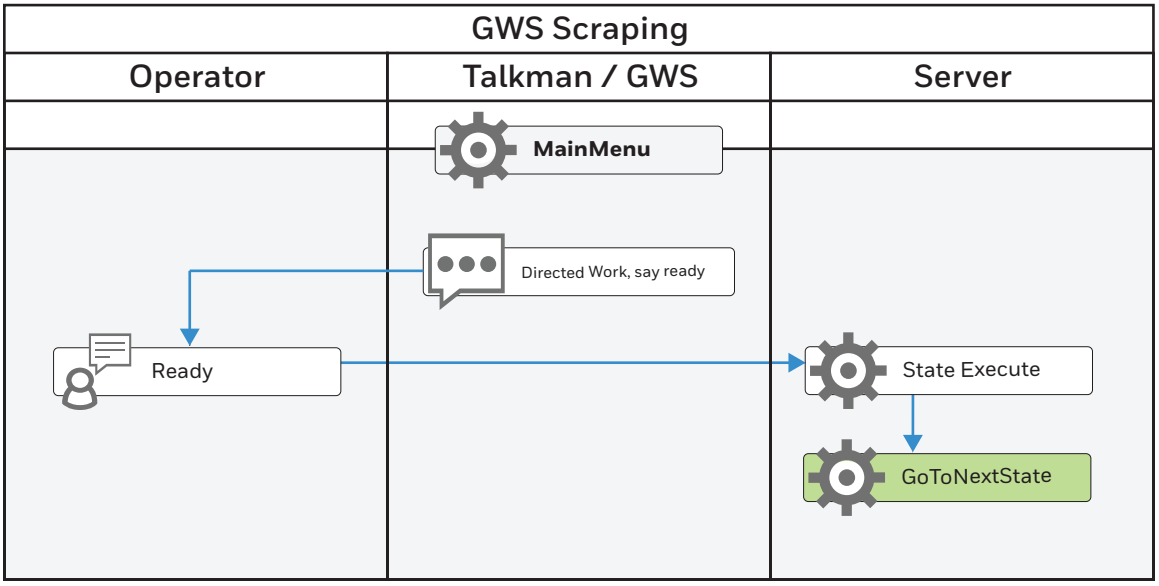


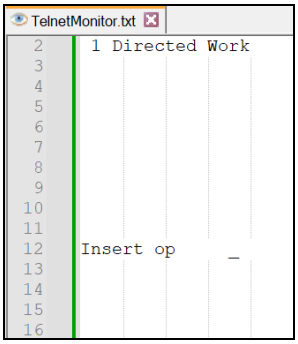


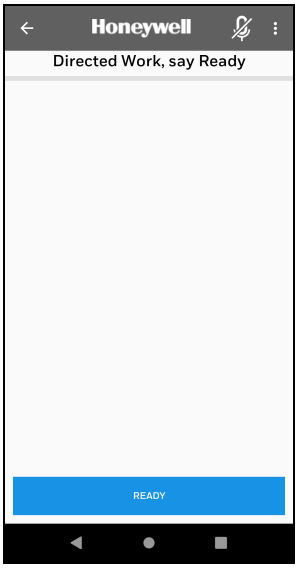
Main Menu (Message state)

The user is asked to select the work type to be assigned.

The terminal emulation screen provided by the server is a Menu screen. It is modeled as *IMessage* State as an example of a case where always the same (**Directed Work**) option must be selected so it does not require a full menu parametrization. In this case, a message is presented to the user with the selected option information.

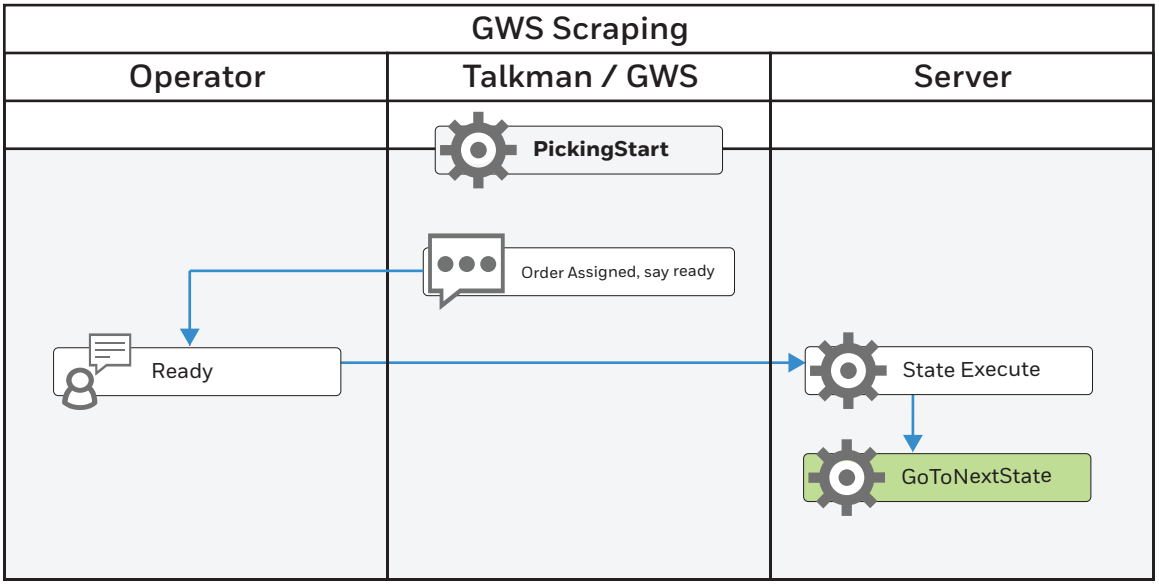


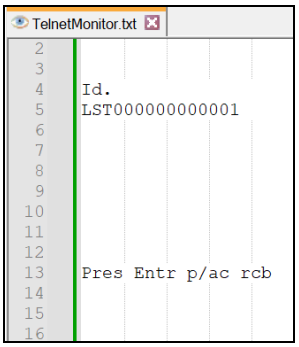


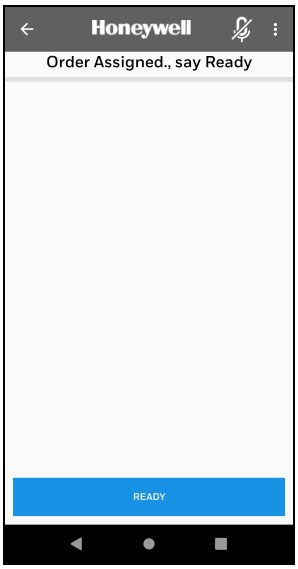


Picking Start (Message State)

Next, a message is displayed indicating that order has been assigned to the user.

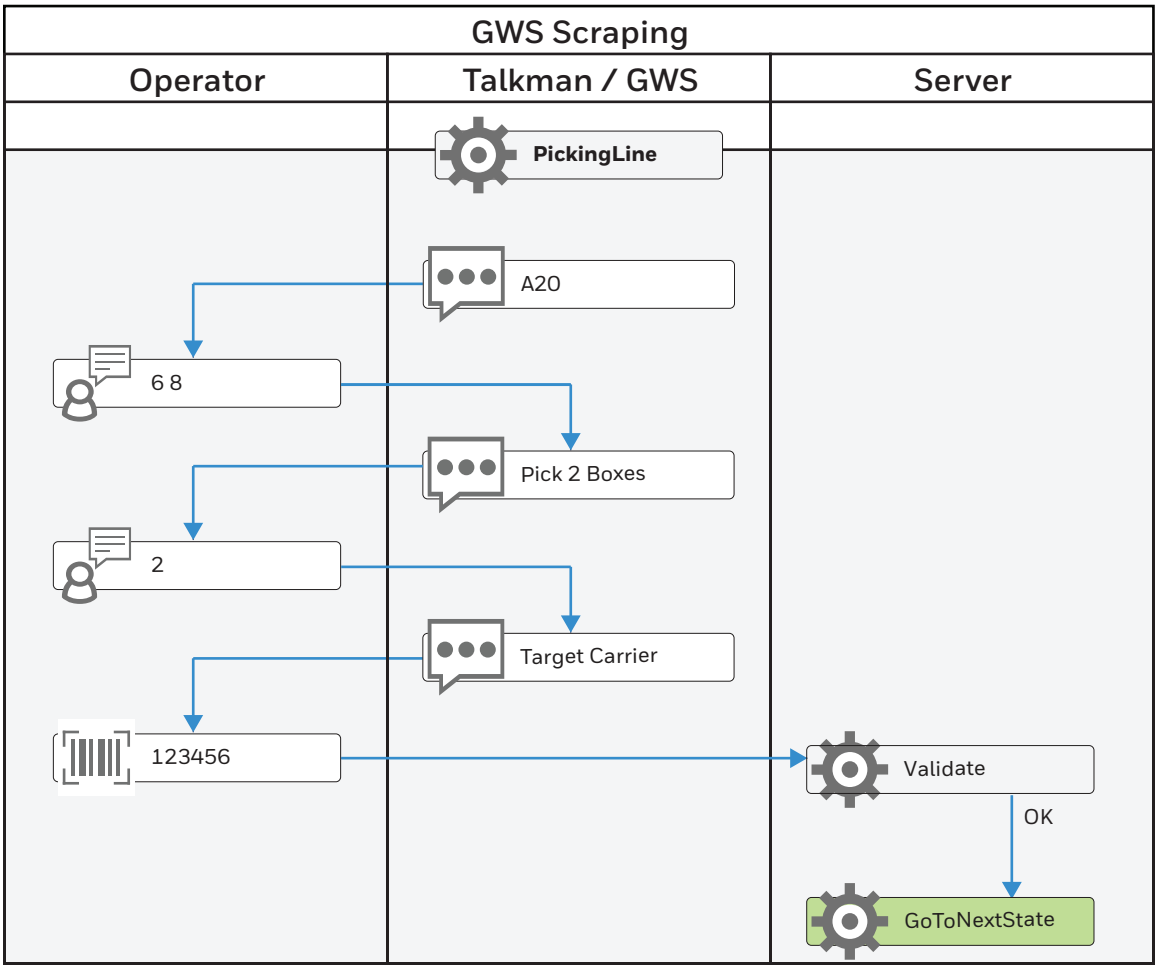


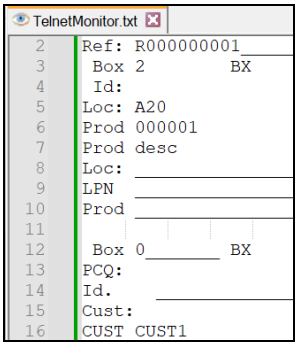


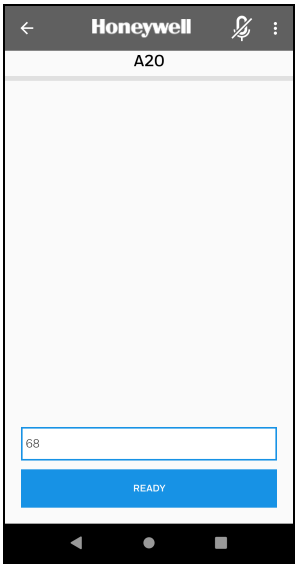


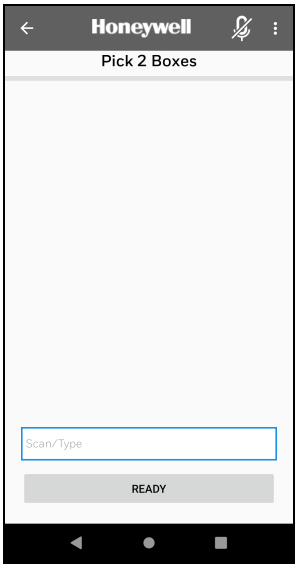
First Picking Line (PickingLine State)

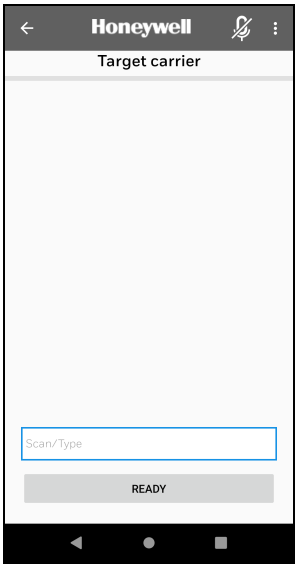
In this screen, the user must validate the location using a control digit (68). The user is then asked to specify the quantity of items to pick. Since the server does not enforce quantity controls, the user can enter any value. After that the user is asked to scan the target carrier.


GWS Scraping
Operator
Talkman / GWS
Server
PickingLine
A20
6 8
Pick 2 Boxes
2
Target Carrier
123456
Validate
OK
GoToNextState


TelnetMonitor.txt
2 Ref: R000000001
3 Box 2 BX
4 Id:
5 Loc: A20
6 Prod 000001
7 Prod desc
8 Loc:
9 LPN
10 Prod
11
12 Box 0 BX
13 PCQ:
14 Id.
15 Cust:
16 CUST CUST1

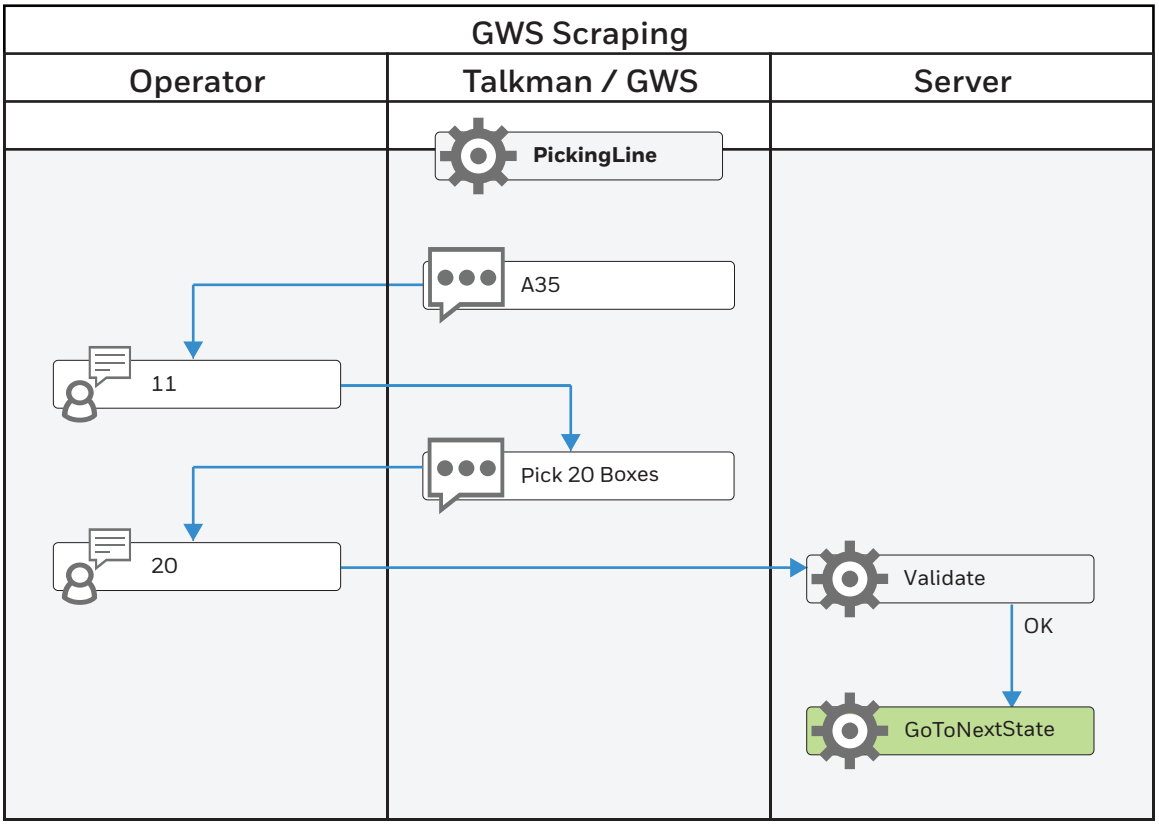

Honeywell
A20
68
READY

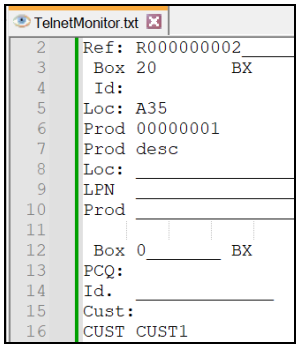

Honeywell
Pick 2 Boxes
Scan/Type
READY

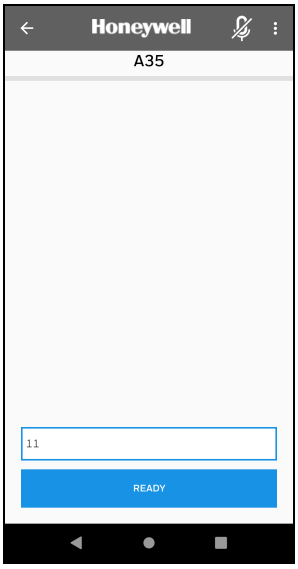

Honeywell
Target carrier
Scan/Type
READY

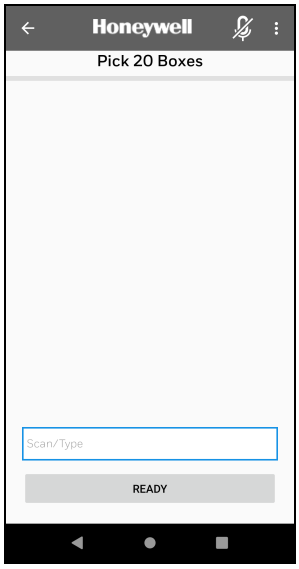
Second Picking Line (PickingLine State)

On this screen, the user must validate another location (11) and specify the quantity of items. Since a carrier was scanned previously, there is no need to rescan it, as it is assumed to be validated automatically.



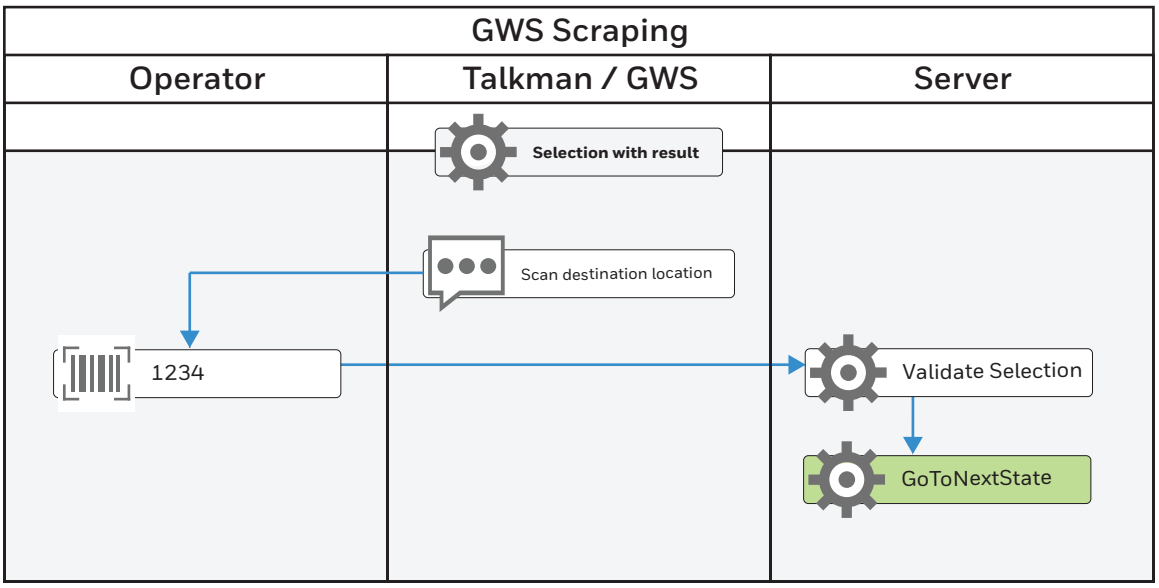


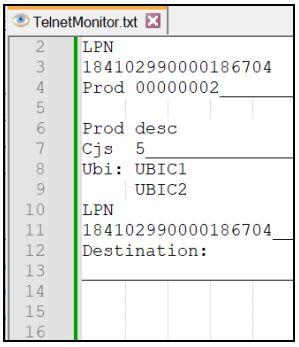


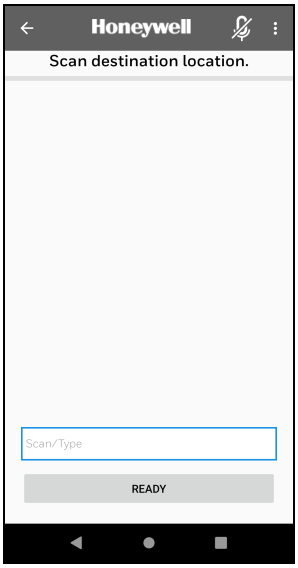


Depositing in Pick & Place Location (Selection State)

After picking the items, the user is instructed to deposit them in a "pick & place" location and validate it by scanning location code.

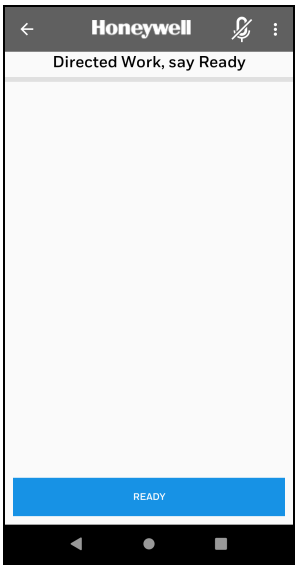


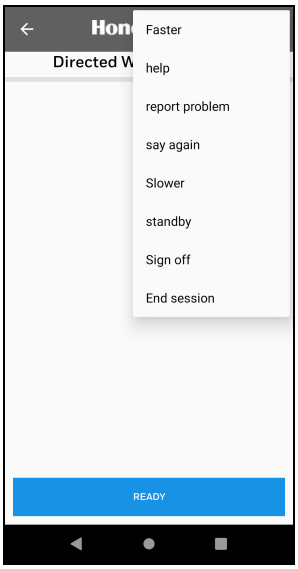




Main Menu (Message State)

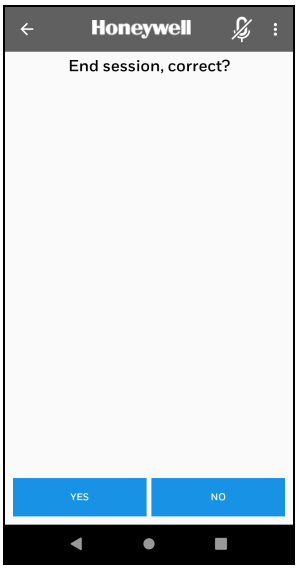
After completing a task, the user returns to the Main Menu screen, where they can choose to end the session by activating “End Session” command.

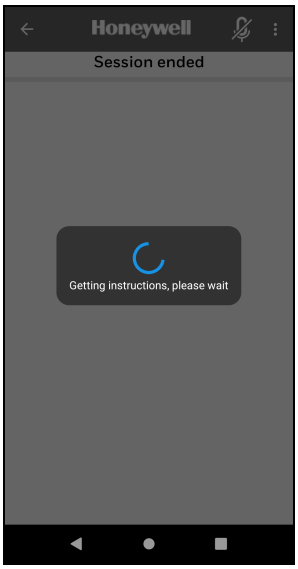




End of session

If the user decides to end the session, a closure reason screen is displayed. Please note that in the demo, this reason is sent automatically without requiring a specific selection from the user.





APPENDIX

A VOICE COMMANDS

GWS App supports up to 99 voice commands that can be enabled or disabled during the voice workflow. Voice commands are defined in the GWExternalResources.resx file set, where the en_US file is the main one and the satelite files provide translations for other languages.

Voice Command Definition

Each voice command is composed of three entries defined in the resource files.

1. **Command Key (VCOMMANDxx)**: The unique identifier of the command. This key determines how the command is used throughout the entire application and how the vocabulary template is stored in VoiceConsole. It should only be defined in the main GWExternalResources.resx file.
2. **Display Value (VCOMMANDxx_Display)**: The text that is shown on screen, during voice training and in the commands list.
3. **Spoken Value (VCOMMANDxx_Spoken)**: The phonetic form of the word or phrase used during voice recognition and text-to-speech. This value must always be provided in lower-case (as upper-case letters are spelled).

```
<!-- Command Key -->
<data name="VCOMMAND01" xml:space="preserve">
    <value>leave</value>
    <comment>This is command 01</comment>
</data>
<!-- Display -->
<data name="VCOMMAND01_Display" xml:space="preserve">
    <value>Leave</value>
    <comment>Displayed during voice training of command 01</comment>
</data>
<!-- Spoken -->
<data name="VCOMMAND01_Spoken" xml:space="preserve">
    <value>leave</value>
    <comment>The phonetic word for command 01</comment>
</data>
```

Both the **Display** and **Spoken** values must be defined in every language-specific resource files. If not, default English definitions are used.

```
<data name="VCOMMAND01_Display" xml:space="preserve">
    <value>Abandonar</value>
</data>
<data name="VCOMMAND01_Spoken" xml:space="preserve">
    <value>abandonar</value>
</data>
```

Command	Display Value	Spoken Value	Description
VCOMMAND01	Leave	leave	Leave the current screen
VCOMMAND02	Cancel	cancel	Cancels the current operation
VCOMMAND03	Skip line	skip line	Skip the current line
VCOMMAND04	Close carrier	close carrier	Close the current carrier
VCOMMAND05	Change format	change format	Change the unit of measure for the pick
VCOMMAND06	Order	order	Informational command for order
VCOMMAND07	Customer	customer	Informational command for customer
VCOMMAND08	Location	location	Informational command for location
VCOMMAND09	Product	product	Informational command for product
VCOMMAND10	Complete	complete	Complete the selection (when multiple options are enabled)
VCOMMAND11	Carrier	carrier	Informational command for carrier
VCOMMAND30	End session	end session	Ends the session with the remote server

Adding a voice command

If you need to add a new voice command as part of your customization project, remember that you must include the **command key**, **display value**, and **spoken value** in the main **GWExternalResources.resx** file. After that, you need to add the corresponding translations for the **display** and **spoken** entries in each translated resource file.

Customizing a voice command name

This mechanism lets you change the name of a voice command. For example, if you want to rename a command from 'Pass assignment' to 'Pass', you'll need to update the command key (optional), the display value, and the spoken value. Notice that if you change the command key entry, the updated voice command must be retrained by all users.

<!-- Command Key -->

<data name="VCOMMAND01" xml:space="preserve">

<value>pass</value>

<comment>This is command 01</comment>

</data>

<!-- Display -->

<data name="VCOMMAND01_Display" xml:space="preserve">

<value>Pass</value>

<comment>Displayed during voice training of command 01</comment>

</data>

<!-- Spoken -->

<data name="VCOMMAND01_Spoken" xml:space="preserve">

<value>pass</value>

<comment>The phonetic word for command 01</comment>

</data>

Honeywell
855 S Mint St
Charlotte, NC 28202

automation.honeywell.com

GWS Scraping Plug-In Product Description

May 2026