



Wavelink Speakeasy Reference Guide

Version 2.28

Revised 07/02/2012

Copyright © 2012 by Wavelink Corporation All rights reserved.

Wavelink Corporation
10808 South River Front Parkway, Suite 200
South Jordan, Utah 84095
Telephone: (801) 316-9000
Fax: (801) 316-9099
Email: customerservice@wavelink.com
Website: www.wavelink.com

No part of this publication may be reproduced or used in any form, or by any electrical or mechanical means, without permission in writing from Wavelink Corporation. This includes electronic or mechanical means, such as photocopying, recording, or information storage and retrieval systems. The material in this manual is subject to change without notice.

The software is provided strictly on an "as is" basis. All software, including firmware, furnished to the user is on a licensed basis. Wavelink grants to the user a non-transferable and non-exclusive license to use each software or firmware program delivered hereunder (licensed program). Except as noted below, such license may not be assigned, sublicensed, or otherwise transferred by the user without prior written consent of Wavelink. No right to copy a licensed program in whole or in part is granted, except as permitted under copyright law. The user shall not modify, merge, or incorporate any form or portion of a licensed program with other program material, create a derivative work from a licensed program, or use a licensed program in a network without written permission from Wavelink. The user agrees to maintain Wavelink's copyright notice on the licensed programs delivered hereunder, and to include the same on any authorized copies it makes, in whole or in part. The user agrees not to decompile, disassemble, decode, or reverse engineer any licensed program delivered to the user or any portion thereof.

Wavelink reserves the right to make changes to any software or product to improve reliability, function, or design.

The information in this document is bound by the terms of the end user license agreement.



Table of Contents

Chapter 1: Introduction	1
Speakeasy Features	1
Implementing Speakeasy	2
Document Assumptions and Conventions	2
Chapter 2: Using the Grammar File Manager	4
Launching the Grammar File Manager	5
Importing and Exporting Grammar Files	5
Editing a Grammar File	6
Grammar File Structure	7
Adding Terms to a Grammar File	8
Using Alternate Return Values	9
Enabling Voice Training for New Terms	10
Modifying User Prompts	11
Specifying Voice Profile Training Options	11
Chapter 3: Using Speakeasy with the Screen Reformatter	14
Adding Text-to-Speech in the Reformatter	14
Adding Speech-to-Text in the Reformatter	15
Adding Scripts to Reformatted Screens	21
Ordering Screen Actions	22
Deploying Reformatted Screens	22
Chapter 4: Speakeasy and Scripting	23
Creating Scripts	23
Activating Scripts	23
Select From Menu	24
On Key Combination	25
When Session Connects	25
On Barcode, MSR, or RFID Scan	25
On Screen Update	25
From Web Pages	25
From the Screen Reformatter	26
Chapter 5: Installation and Licensing	27
Installation Requirements	27
Installing Speakeasy Using Avalanche	28
Installing Speakeasy Using ActiveSync	29
Installing Speakeasy on a PC	30
Licensing	31
Uninstalling Speakeasy	31
Uninstalling Speakeasy from a Mobile Device	31
Uninstalling Speakeasy from a PC	32



Chapter 6: Using Speakeasy	33
Calibrating the Microphone.....	33
Configuring the Text-to-Speech Options.....	36
Creating a Voice Profile.....	37
Using the Speakeasy Listening Indicator.....	38
Chapter 7: Speakeasy Settings	39
Sample Speakeasy Scripts	43
Using a Voice Profile.....	43
Reading the Screen Aloud.....	43
Displaying Speech Results in a Dialog Box.....	44
Changing Text-to-Speech Modes.....	44
Speech Demo Script.....	46
Acquiring a Pick Quantity in the Industrial Browser.....	47
Acquiring a Pick Quantity Script.....	47
Acquiring Pick Quantity Grammar File.....	49
JavaScript Function clearFields.....	51
Wavelink Contact Information	52
Glossary	53



Chapter 1: Introduction

Speakeasy is a verbal communication system that facilitates real-time voice communication between the host computer and the mobile device user. It is available to use with the Wavelink Terminal Emulation (TE) Client 7.0 and newer versions. Speakeasy can be easily implemented from the Screen Reformatter or you can create scripts to customize how Speakeasy is used.

Speakeasy provides the ability to translate data from the host computer into spoken directions that the user is able to hear (text-to-speech). The user's spoken response can then be transcribed and sent back to the host computer (speech-to-text). You can install both text-to-speech and speech-to-text, or just the component that fits your company's needs. This introduction provides a list of Speakeasy features, guidelines for implementing Speakeasy, and document assumptions and conventions.

- [Speakeasy Features](#)
- [Implementing Speakeasy](#)
- [Document Assumptions and Conventions](#)

Speakeasy Features

Speakeasy has a variety of features that make using voice with terminal emulation easy and practical:

- **Voice profiles.** Speakeasy allows you to create voice profiles to increase correct results. Voice profiles allow each user to train on problematic words or phrases, increasing recognition and efficiency. Voice profiles are not necessary, but they can be useful in situations where pronunciation may vary from speaker to speaker.
- **Voice profile distribution with Avalanche.** When used with Wavelink Avalanche, Speakeasy can archive and distribute voice profiles easily and seamlessly. When a voice profile is created, the Avalanche Enabler uploads it to the mobile device server. When a user types in his username, Speakeasy checks if the voice profile exists on the device. If the profile doesn't exist on the device, a request is sent to the mobile device server, and the profile is downloaded and instantly ready for use.
- **Microphone calibration tools.** Speakeasy has built-in calibration tools that use Speakeasy settings to improve microphone performance. There is a quick calibration that uses a few verbal prompts and responses for calibration, or a full calibration that provides tools for the user to change settings manually.
- **Integration with the Screen Reformatter.** Speakeasy options are available in the Terminal Emulation Screen Reformatter, making it easy to add speech-to-text and text-to-speech to



reformatted screens. Use the Screen Reformatter to change Speakeasy settings, select the grammar files available, or perform specific functions for a speech-to-text result.

- **Range of available languages.** Speakeasy works with a variety of languages and dialects. For a current list of languages supported by Speakeasy, refer to the Wavelink Web site. To use Speakeasy with languages other than English, install the language (a.k.a. international) build of the TE Client.
- **Client-side processing.** All Speakeasy processing is performed on the mobile device, reducing bandwidth usage and eliminating the need for server modifications.

Implementing Speakeasy

Speakeasy is software installed on a mobile device that can read text aloud to a device user or accept voice commands as text entry. It is designed to be used with the Wavelink Terminal Emulation (TE) Client. Use the following steps to set up Speakeasy:

- Decide which emulation processes can be streamlined and made more efficient using either speech-to-text or text-to-speech.
- Ensure you have the appropriate hardware, such as headphones and voice-enabled devices.
- Obtain the desired licenses, speech-to-text packages, and text-to-speech packages from Wavelink.
- If you are using speech-to-text, create the list of words or phrases that the device users will need. Include the words and phrases in a grammar file. If you want to use voice training, specify in the grammar files which terms you want to train with.
- Use the screen reformatter and/or TE scripting to handle the speech elements. This may include having the screen read aloud to the user (text-to-speech) or using voice commands to return text, execute actions, and change settings (speech-to-text).
- Install the Speakeasy packages, grammar files, and scripts on the devices and distribute the licenses to the devices.
- If you implement voice training, users will be prompted to perform training when they start to use Speakeasy.

Document Assumptions and Conventions

This guide assumes that the reader has the following:

- Knowledge of wireless networks and wireless networking protocols.



- Knowledge of the terminals, operating systems, and mobile devices in your emulation environment.
- Knowledge of Wavelink Terminal Emulation and TE scripting.

This guide uses the following typographical conventions:

Courier New Any time you type specific information into a text box (such as a file name), that option appears in the *Courier New* text style. This text style is also used for any keyboard commands that you might need to press.

Examples:

Type `Enter` to continue.

Press `CTRL+ALT+DELETE`.

Bold Any time you interact with an option (such as a button or descriptions of different options in a dialog box), that option appears in the **Bold** text style.

Example:

Click **File > Open**.

Italics Any time this document refers to titles of dialog boxes, they appear in the *Italics* text style.

Example:

The *Screen Reformatter* dialog box appears.



Chapter 2: Using the Grammar File Manager

In order to use speech-to-text, you must configure the grammar files. Grammar files define what the Speakeasy engine will recognize. For example, the `digit.bnf` grammar file allows the engine to recognize when the speaker says any number zero through nine, “yes, please” or “no, thank you”. Speakeasy comes with several default grammar files that can be edited to fit your environment.

The Grammar File Manager allows you to manage the following speech-to-text options:

- **Specify which grammar files are installed on your device.** Speakeasy comes with several grammar files that you can use. You can use them as provided, edit them, or create your own grammar files. Speakeasy allows you to deploy as many grammar files as you want to the device, and you can use up to ten grammar files simultaneously (during a script or while on a specific screen that has been modified using the screen reformatter).
- **Specify the prompts Speakeasy uses during calibration and user training.** You can create different prompts for each language you use with Speakeasy. The prompts are used when calibrating the microphone, when the user is creating a voice profile, and when voice profiles are being archived or retrieved from the Avalanche server.
- **Specify which words or phrases each user should train on before they begin using Speakeasy.** If there are words that the engine has trouble recognizing, you can have each user create a voice profile. When a user creates a voice profile, he speaks the words specified and Speakeasy stores the results and uses them to increase positive results.

NOTE: After you make changes in the Grammar File Manager, you will need to deploy the changes to the mobile device. If you are using a Windows TE Client, you do not need to deploy the changes.

This section includes the following tasks for using the Grammar File Manager:

- [Launching the Grammar File Manager](#)
- [Importing and Exporting Grammar Files](#)
- [Editing a Grammar File](#)
- [Modifying User Prompts](#)
- [Specifying Voice Profile Training Options](#)

NOTE: In previous versions, grammar files were considered either engine mode or attribute mode. Starting with version 2.0.5 of Speakeasy, all grammar files are considered engine mode grammar files.



Launching the Grammar File Manager

If you are using Wavelink Avalanche, the Grammar File Manager can be accessed through the Avalanche Console after you have the speech-to-text package added to a profile. If you are using ActiveSync to install or if you are using a Windows TE Client, then you can access the Grammar File Manager through the Windows **Start** menu after you have installed Speakeasy on a PC.

Wavelink recommends you use the Windows TE Client for developing and testing grammar files, scripts, and reformatted screens.

To launch the Grammar File Manager from the Avalanche Console:

- 1 From the **Profiles** tab, select the software profile that has the speech-to-text base package.
- 2 Select the speech-to-text base package from the Software Packages list and click **Configure**.
- 3 The *Configure Software Package* dialog box appears. Select **Grammar File Manager** from the list and click **Launch**.

To launch the Grammar File Manager for the Windows TE Client:

- Click **Start > Programs > Wavelink SpeechToText Support > Grammar File Manager**.

To launch the Grammar File Manager for an ActiveSync installation:

- 1 Create an ActiveSync connection to the device.
- 2 Click **Start > Programs > Wavelink SpeechToText ActiveSync Support > Install to Device**.
- 3 The *Windows Speech-to-Text Installer* appears. Click **Grammar File Manager**.

Importing and Exporting Grammar Files

Grammar files must be imported/exported using the Grammar File Manager in order to be modified or copied to a different package. When you export a grammar file from the Grammar File Manager, it is exported as a text file with a `.bnf` extension. The `.bnf` file can be copied or edited using a text editor such as Notepad. The file is then imported back into the Grammar File Manager so that it can be available for download to the device.

To import a `.bnf` file into the Grammar File Manager:

- 1 Launch the Grammar File Manager.
- 2 Click **Import**.
- 3 The *Import Grammar* dialog box appears. Navigate to the location where the grammar file is saved, select it, and click **Open**.



- 4 The *Select Languages* dialog box appears. Select all the languages from the list that the grammar will be used with and click **OK**.
- 5 The *Edit Item* dialog box appears. Type a description for the grammar file in the **Description** text box and click **OK**.

NOTE: You can also edit the description for the grammar file after it has been added by selecting the file from the list in the Grammar File Manager and clicking **Edit**.

The grammar file is imported into the Grammar File Manager and you can select it to be downloaded to the device.

To export a grammar file from the Grammar File Manager:

- 1 Launch the Grammar File Manager.
- 2 Click **Export**.
- 3 The *Save As* dialog box appears. Navigate to the location where you want to save the file and click **Save**.

The file is saved as a text file with a `.bnf` extension and can be modified with a text editor or copied to a different Grammar File Manager.

Editing a Grammar File

Grammar files define which words and phrases the Speakeasy engine will recognize. To edit a grammar file, export it from the Grammar File Manager and open it using a text editor such as Notepad. When you are finished editing it, save your changes and import it back into the Grammar File Manager.

The Grammar File Manager comes with several grammars, but you can edit grammar files to contain the words you will use to perform tasks. For example, if you want to use a voice command to increase the speed of text-to-speech, add the term “faster” to a grammar file. Then use the screen reformatter to run a script for the speech-to-text result “faster” that increases the `tts_rate` setting. When the user says, “faster,” the text-to-speech speed increases.

To improve recognition, use longer and more distinct phrases. For example, rather than use the similar terms “faster” and “fastest,” replace them with the terms “go faster” and “warp speed”.

The following sections provide information on editing grammar files:

- [Grammar File Structure](#)
- [Adding Terms to a Grammar File](#)



- [Using Alternate Return Values](#)
- [Enabling Voice Training for New Terms](#)

Grammar File Structure

The following example is a basic grammar file (the grammar file `VoiceSpeed.bnf`) viewed in Notepad:

```

VoiceSpeed.bnf - Notepad
File Edit Format View Help
#BNF+EM V1.1;
/*****
GRAMMAR: VoiceSpeed.bnf

Description:
    This grammar file lets the user change the voice and voice speed, quit,
what Can I Say?
    You can increment or decrement the speeds or set to the normal rate of 1
    You can say three voices: Jill, Samantha & Tom.

*****/

!grammar voicespeed;
!start <speech>;

<speech>: slower
        faster
        normal
        address
        description
        repeat
        again
        jill
        Samantha
        Tom;

```

Example of a Grammar File

Each grammar file begins with a statement such as `#BNF+EM V1.1;` that identifies how Speakeasy should handle the grammar file. You do not need to modify this line.

To create comments — information ignored by the engine — you can begin a line with `//` or enclose text in `/*` and `*/` markers. You can edit the Description and the What Can I Say? sections to describe changes you make to the grammar file.

If you change the name of the grammar file, the command `!grammar` must use the new file name. For example, if the name of the file was changed to `VoicePick.bnf`, the command would read: `!grammar VoicePick;`

The `!start` command specifies which terms in the grammar will be available when the grammar is in use. In the above example, the engine will listen for all of the terms in the `<speech>` section. The words and phrases listed between `<speech>:` and the last `;` are



the terms the grammar will accept. Each word or phrase must be separated by | (an OR symbol). Terms may be listed all on the same line or aligned vertically for ease in editing.

In some grammar files, the main section is divided into subsections to make it easier to manage the terms. In the example fragment below, the terms are divided so that the `!slot` command will only apply to some of the terms:

```
!start <Speech>;
!slot <YesNo>;

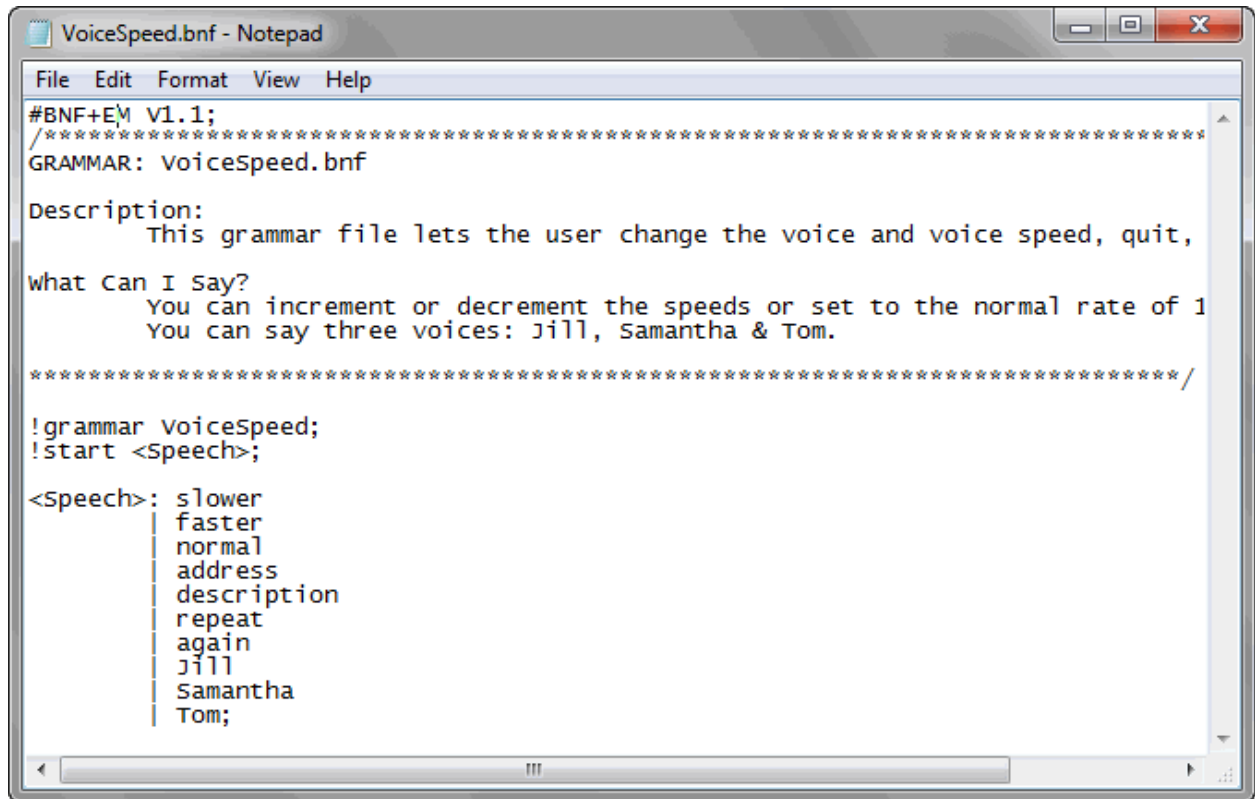
<Speech>: <YesNo> | <Maybe>;
<YesNo>: yes | no;
<Maybe>: maybe;
```

In this example, the `<Speech>` section includes the `<YesNo>` and the `<Maybe>` sections, so the engine will listen for all the terms in both the `<YesNo>` section and the `<Maybe>` section. However, only the terms in the `<YesNo>` group are included in the `!slot` command and used for voice training. (For more information about using the `!slot` command, see [Enabling Voice Training for New Terms](#) on page 10.)

Adding Terms to a Grammar File

A grammar file determines the words and phrases that the Speakeasy engine listens for. To add words or phrases to a grammar file, add it to an existing section (such as `<Speech>`) and separate it from the other terms with a | symbol (also known as an OR symbol or a pipe). The following example is the grammar file `VoiceSpeed.bnf` viewed in Notepad:





```

VoiceSpeed.bnf - Notepad
File Edit Format View Help
#BNF+E^M V1.1;
/*****
GRAMMAR: Voicespeed.bnf

Description:
    This grammar file lets the user change the voice and voice speed, quit,
what can I say?
    You can increment or decrement the speeds or set to the normal rate of 1
    You can say three voices: Jill, Samantha & Tom.

*****/

!grammar Voicespeed;
!start <speech>;

<speech>: s|lower
          |faster
          |normal
          |address
          |description
          |repeat
          |again
          |jill
          |Samantha
          |Tom;

```

Example of a Grammar File

In this example, the words and phrases listed between `<Speech>:` and the last `;` are the terms Speakeasy will recognize. Each word or phrase must be separated by an `|` symbol. Terms may be listed all on the same line or aligned vertically for ease in editing.

Once you have defined the words in a grammar file, use a script or the screen reformatter to define what happens when each word or phrase is recognized. Then deploy the grammar file and any other necessary files to the devices.

Using Alternate Return Values

You may want to have a grammar file return a value other than the exact phrase that the user says. This may be useful when you want to use phrases rather than words, or if you want different phrases to return the same result. Using phrases rather than single-syllable words improves recognition. For example, when the user says, "repeat prompts," the Speakeasy engine can act like it hears "repeat".

You can also use more than one word or phrase to return the same result. This may be helpful in a multilingual environment. For example, you could have an English grammar file that listens for "yes" and a Dutch grammar file that listens for "ja," but either grammar would return the result "yes." Then the script or reformatted screen would only have to be programmed for one result.



To use an alternate return value, list the word or phrase the engine should listen for and then `{@ = "result";}` where `result` is the alternate return value. If the result is empty (i.e., `@ = "";`), the engine will recognize the word or phrase but not return any results.

The following example is the grammar file `sapDeliveryEntryCmdAndCtrl.bnf` viewed in Notepad.

```
#BNF+EM V1.0;
// Accepts and returns a repeat, back, quit, and clear value.

!grammar sapDeliveryEntryCmdAndCtrl;
!language "American English";
!start <Speech>;

<Speech>: repeat prompts {@ = "repeat";} |
go back {@ = "back";} |
quit sap {@ = "quit";} |
clear fields {@ = "clear";} |
help list {@ = "help"};
```

Example of a Grammar File

In this example, if the user said "go back," Speakeasy would return the text "back". However, if the user said "back," it would not match the terms the engine is listening for. Add a separate entry for "back" if you want the engine to listen for it, too.

Enabling Voice Training for New Terms

When you add words or phrases to a grammar file, you can specify if they should be available for voice training. The `!slot` command in a grammar file specifies which words and phrases should be available for voice training.

While you are adding terms to a grammar file, there are two steps to perform in order to enable voice training for the new terms. First, group the terms that you want to use for training separate from the terms that you don't want to use for training. Assign each group a name, and then group the terms in the grammar file together under that name.

In the example below, the highlighted line shows that groups `YesNo` and `Cancel` are included in the main group, `Speech`. Then each term in the groups is defined on the last two lines.

```
!grammar YesNoCancel;
!start <Speech>;
!slot <YesNo>;
<Speech>: <YesNo> | <Cancel>;
<YesNo>: yes | no;
<Cancel>: cancel;
```



Notice that the `!start` command specifies the main group, `Speech`. Since `Speech` includes `YesNo` and `Cancel`, this grammar file means the engine will listen for all the terms in both those groups ("yes," "no," and "cancel").

After the terms have been grouped, use the `!slot` command on the line following the `!start` command. When you use the `!slot` command, specify the group of terms that you want to use for training. The example above will enable voice training for the terms in the group `YesNo`.

NOTE: The `!slot` command only works for groups of simple terms such as words, phrases, and alternate terms. It doesn't work for groups that contain other groups (such as `Speech` in the above example) or complex commands. You cannot use the `!slot` command for groups that contain `!repeat` or `#`.

Once you have put the terms in a group and specified the group using the `!slot` command, use the Grammar File Manager to select the terms for training before the package is deployed to the device. See [Specifying Voice Profile Training Options](#) on page 11 for information on selecting terms for user training.

Modifying User Prompts

Speech-to-text uses several prompts during calibration and voice profile training. These prompts default to English values, but you can modify the prompts for each language used with Speakeasy.

To modify the user prompts:

- 1 Launch the Grammar File Manager.
- 2 Click **User Prompts**.
- 3 The *User Prompts* dialog box appears. Select the language from the **Language** drop-down menu that the prompts will be available for, and modify the prompts as desired. If you are not using *Avalanche* to store and distribute voice profiles, you do not need to change the Training Data Archiving options.
- 4 Click **OK** to save your changes. The prompts will be used when you perform calibration or voice profile training.

Specifying Voice Profile Training Options

Speakeasy allows users to create personalized voice profiles, which can improve the positive results when you are using speech-to-text. When you create a voice profile, you speak a set of selected words to train the engine to recognize your voice and pronunciation. The words used for voice profile training come from the grammar files available on the device and are selected from the Grammar File Manager.

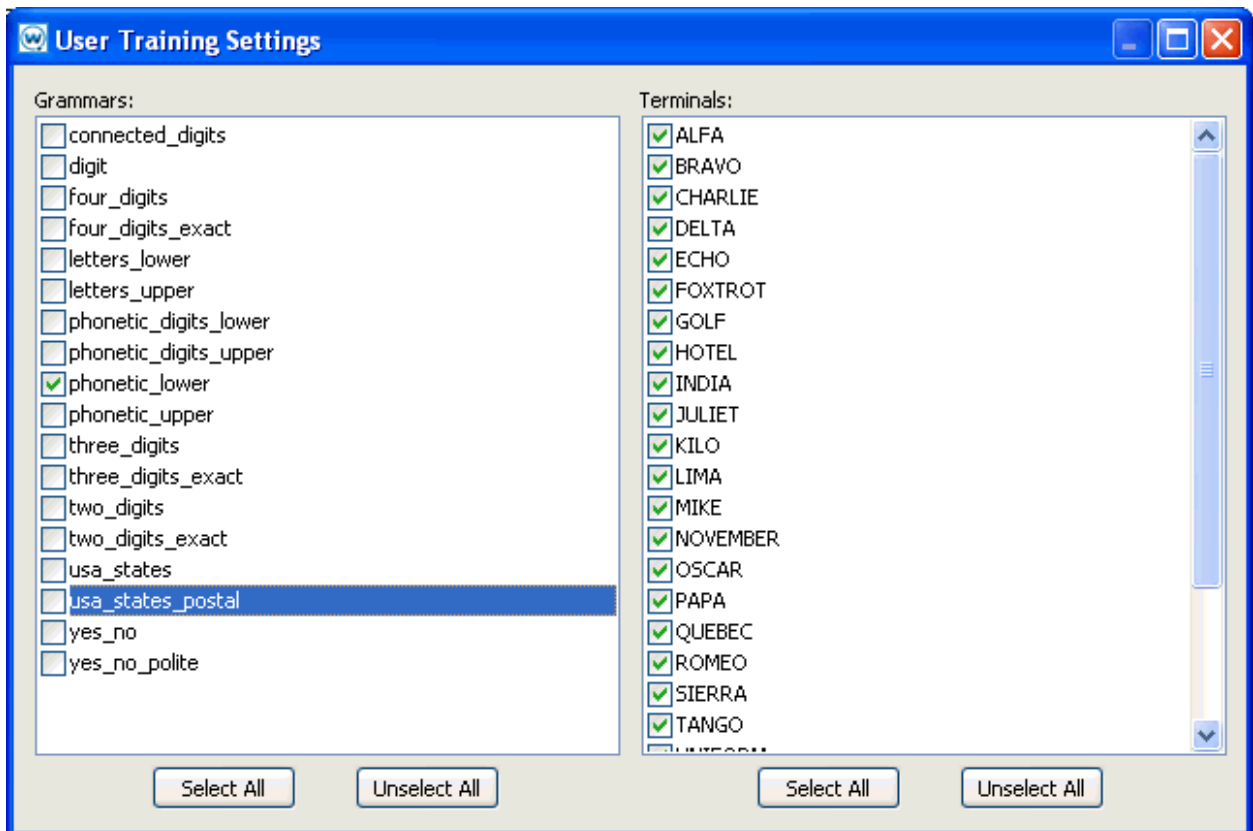


When you are selecting words and phrases to use for training, focus on terms that are short or that the engine may have trouble understanding. If you have two words that may sound similar or words that are only one syllable, you may want to include these in the training.

Before a word or phrase can be selected for voice training, you must define possible voice training words in the grammar file. For information on enabling voice training for words or phrases, see [Enabling Voice Training for New Terms](#) on page 10.

To specify the words used for voice profile training:

- 1 Launch the Grammar File Manager.
- 2 Click **Training Settings**.



User Training Settings dialog box

- 3 The *User Training Settings* dialog box appears. From the **Grammar** list, select at least one grammar that has words or phrases that you want to use for training.
- 4 When you have selected a grammar, the words and phrases associated with that grammar appear in the **Terminals** list. Enable the checkboxes for the terms that you want to use for training.



- 5 When you have selected all the terms you want to use for training, click the red **X** to close the dialog box. The terms will be available for training when a voice profile is created or when a grammar is used that a user has not yet trained for.

NOTE: For information on creating voice profiles, see [Creating a Voice Profile](#) on page 37.



Chapter 3: Using Speakeasy with the Screen Reformatter

The TE Client screen reformatter allows you to modify the appearance of Telnet emulation screens. You can create a screen layout that includes items you want the mobile device user to see, and excludes items that should not be visible to the user. The screen reformatter also allows you to add Speakeasy actions or scripts to a screen.

This section lists tasks for the screen reformatter that are used with Speakeasy. For a full description of options available from the screen reformatter, see the *Terminal Emulation Client User Guide*. This section provides information about the following:

- [Adding Text-to-Speech in the Reformatter](#)
- [Adding Speech-to-Text in the Reformatter](#)
- [Adding Scripts to Reformatted Screens](#)
- [Ordering Screen Actions](#)
- [Deploying Reformatted Screens](#)

NOTE: The Screen Reformatter does not work with Web emulation.

Adding Text-to-Speech in the Reformatter

You can add text to the modified screen that will be converted into speech and played back to the mobile device user. This can be text from the host screen or text added just for the modified screen. Multiple text-to-speech actions will be processed without a pause if they are adjacent. You can also use the screen reformatter to set text-to-speech settings.

To add text-to-speech from the screen reformatter:

- 1 Click **Edit > Add Text-to-Speech Text**.

-Or-

In the Initial Screen View, right-click and select **Add Text-to-Speech** from the context menu.

The *Text-to-Speech Text* dialog box appears.

- 2 Type your text in the **Text to be spoken** text box. This box can be left empty if you only want to change the text-to-speech settings.
- 3 Type the desired text-to-speech settings in the **Persistent Text-to-Speech Settings to use** and/or the **Temporary Text-to-Speech Settings to use** text boxes.



NOTE: Each setting must start with `tts_` and use the format `setting=value`. Multiple settings can be specified and should be separated by commas. The complete list of settings is available in the *Terminal Emulation Scripting Reference Guide*.

4 Click **OK**.

The *Text-to-Speech Text* dialog box closes and your text is added to the screen. It also appears in the Supporting Actions section of the Descriptive View.

To use existing text for text-to-speech:

1 In the screen reformatter, click and drag the mouse over the text you want to copy.

When you release the left mouse button, a context menu appears.

2 Select **Text-to-Speech Copy**.

The *Text-to-Speech Copy* dialog box appears.

3 Type the desired text-to-speech settings in the **Persistent Text-to-Speech Settings to use** and/or the **Temporary Text-to-Speech Settings to use** text boxes.

NOTE: Each setting must start with `tts_` and use the format `setting=value`. Multiple settings can be specified and should be separated by commas. The complete list of settings is available in the *Terminal Emulation Scripting Reference Guide*.

The Text-to-Speech action appears in the Supporting Actions section of the Descriptive View.

Adding Speech-to-Text in the Reformatter

You can add a speech-to-text action to the modified screen. This action converts the user's speech into text that will be processed according to the grammar(s) specified. It can print the text to the screen as keyboard data or perform an action associated with the command.

The **Local Actions** and **Global Actions** tabs in the *Speech-to-Text* dialog box allow you to assign special actions to Speech-to-Text results, instead of having those results treated as keyboard data. For each Speech-to-Text result received, the Local Actions are tested first. If no Local Actions match, then the Global Actions are tested. If no Global Actions match, the result will be treated as keyboard data and use the settings configured in the General tab.

Global Actions are shared among all the screens, so changing a Global Action for one Speech-to-Text support action will change the action for all the screens. Because the Local Actions take priority over Global Actions, you can override a Global Action by creating a Local Action for the same result value. For example, you could add a Global Action that lists available



commands when the user says “help”. If there is a screen where not all the commands are available, use a Local Action to override the Global Action with a list specific to that screen.

NOTE: The screen reformatter supports dynamic grammar generation. Instead of using an existing grammar file, use a list of words or phrases separated by | (pipe character) to generate an internal grammar. See [Using the Grammar File Manager](#) on page 4 for more information on grammar files.

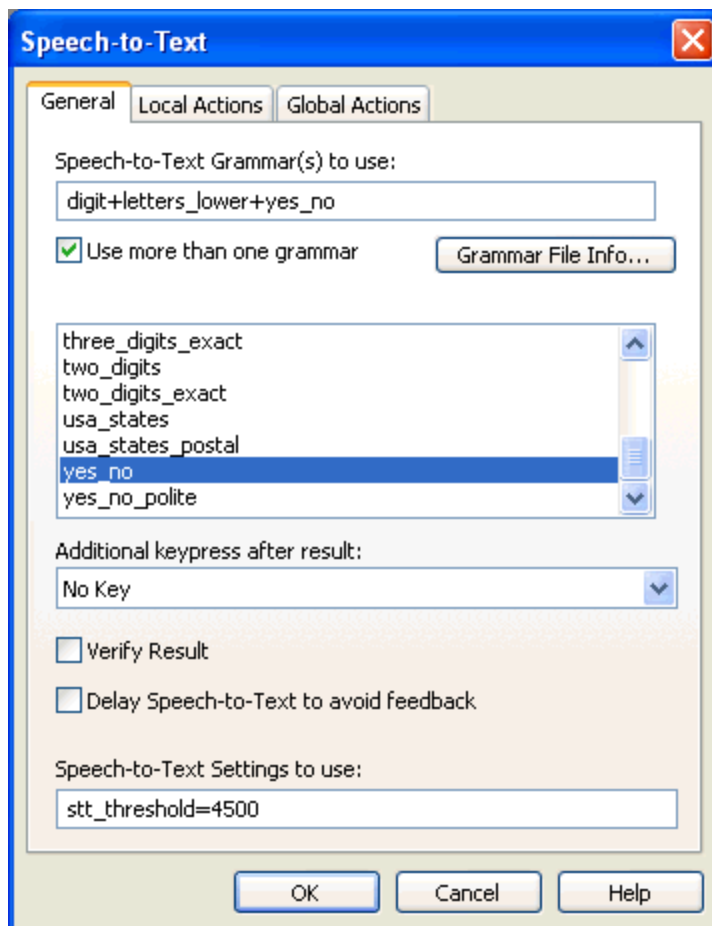
To add speech-to-text from the screen reformatter:

- 1 Click **Edit > Add Speech-to-Text**.

-Or-

In the Initial Screen View, right-click and select **Add Speech-to-Text** from the context menu.

The *Speech-to-Text* dialog box appears.



Speech-to-Text options

- 2 Select the **Speech-to-Text Grammar to use** from the drop-down menu.

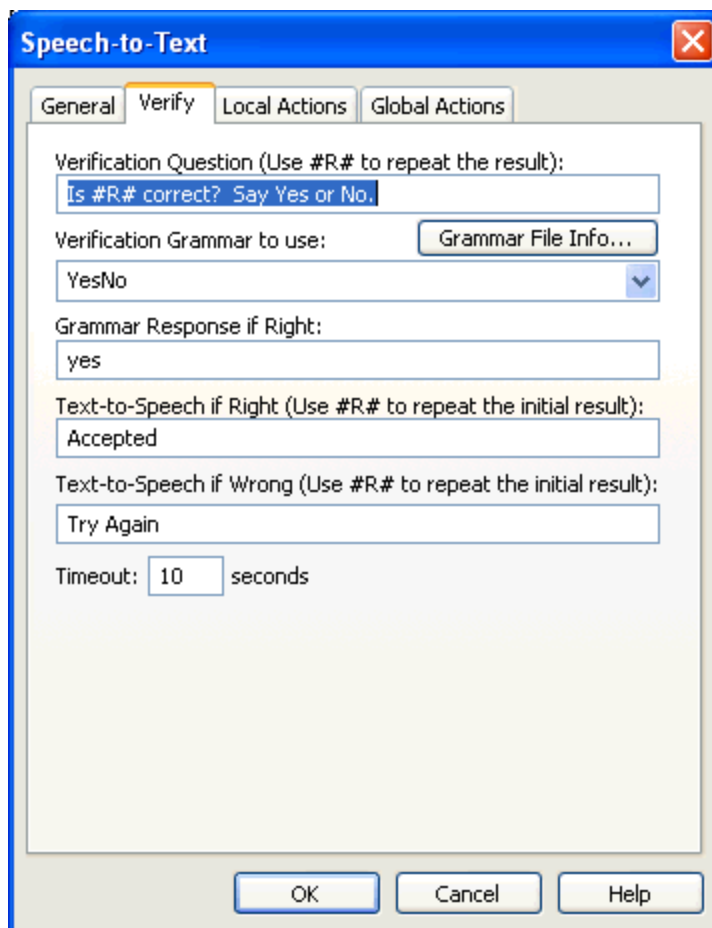


-Or-

Enable the **Use more than one grammar** option and choose the desired grammars from the list box.

- 3 If desired, select a key from the **Additional keypress after result** drop-down menu. The specified keypress will be processed after the speech-to-text action.
- 4 If you would like the Speech-to-Text engine to verify the text result, enable **Verify Result**.

This will make a **Verify** tab appear.



Verify Tab

- 5 Click the **Verify** tab and enter the any of the following information:
 - Enter a question in the **Verification Question** text box.
 - Enter a grammar name in the **Verification Grammar to use** text box.
 - Enter a response in the **Grammar Response if Right** text box.
 - Enter a response in the **Text-to-Speech if Right** text box.



- Enter a response in the **Text-to-Speech if Wrong** text box.
 - Enter a number (in seconds) in the **Timeout** text box. This is how long the screen reformatter will wait for verification that it repeated the correct result. If no verification is received, the result is discarded and no action is performed.
- 6 Return to the **General** tab.
 - 7 If you are not using a headset microphone, enable the **Delay Speech-to-Text to avoid feedback** checkbox. This will ensure that the microphone ignores input while text-to-speech actions are happening.
 - 8 If you want to use a speech setting in the Speech-to-Text action, enter the setting in the **Speech-to-Text Settings to use** text box.

NOTE: Each setting must start with `stt_` and use the format `setting=value`. Multiple settings can be specified and should be separated by commas. If a value is not a number, then the Speech-to-Text engine will use the value closest to the value text description provided. The complete list of settings is available in the *Terminal Emulation Scripting Reference Guide*. Once a setting has been changed, that value will be used for future Speech-to-Text actions until it is changed again.

- 9 If you want to configure local or global actions for the screen, select the appropriate tab and click **Add**.

The *Result/Action* dialog box appears.



Result/Action for Speech-to-Text

- 10 Type the speech-to-text result that you want to configure an action for in the text box and then select the action that will be performed when the user says that.

NOTE: The speech to text results must be an exact match for the action to be performed. Speech-to-text results are case-sensitive.

- **Replace the result with this text.** Replaces the speech-to-text result with the provided text. When this replacement is made, the **Additional keypress after result** value of the **General** tab is not used, so if you want to use an additional key press, you should include it here. For instructions on determining the value of a key press, see "Performing a Keyboard Test" in the TE Client User Guide.



Use `\` followed by the 4-digit hexadecimal number to specify a Telnet key press, and `\U` followed by the 4-digit hexadecimal number to specify a Unicode character. If you want to actually output a backslash character, use `\\`

For example, to replace the result “euro” with “€” followed by a VT `Enter` key press, use this value:

```
\U20ac\000d
```

- **Perform a key press.** Replaces the speech-to-text result with a key press. You can use the name of the key (such as `F3` or `Enter`) or the hexadecimal number.
- **Start a script.** Replaces the result with a script. Use an existing script or click **Edit Scripts** to launch the Script Editor. Assign initial values to the variables in the script by using a comma-delimited list inside of parentheses following the script name. For example, to run the script `DoItNow` and set the string variable **User** to **lucky** and the number variable **Position** to **17**, you would type this:

```
DoItNow(User="lucky",Position=17)
```
- **Pause Speech-to-Text while the script is running.** Disables speech-to-text for the screen reformatter while the script is running. Enable this option if the script is also using speech-to-text so that only one application is trying to use speech-to-text at a time.
- **Perform an IDA action.** Replaces the result with the a standard action. Many of these actions are also available other ways, such as by writing a script or using a key press value.
- **Set a Speech-to-Text or Text-to-Speech setting.** Replaces the result by changing the Speakeasy settings for the Client. Each setting must start with `stt_` or `tts_` and use the format `setting=value`. If a value is not a number, then the engine will use the value closest to the value text description provided. For the complete list of settings, see [Speakeasy Settings](#) on page 39.

You can change multiple settings with a comma-delimited list (no spaces). The list must contain only Speech-to-Text settings, or only Text-to-Speech settings. You cannot mix the two types of settings. If you need to do that, then create and call a script instead of using this option.

- **Standard Beep.** Replaces the result with a standard TE beep. If the beep has been modified in the emulation parameters, the modified beep will be used.
- **Error Beep.** Replaces the result with a TE error beep. If the beep has been modified in the emulation parameters, the modified beep will be used.



- **Restart the supporting actions for this screen.** Replaces the result with a restart of the supporting actions for the screen. Any text-to-speech actions and script actions will be repeated.
- **Do Nothing.** Ignores the result. This option is useful if a grammar is being used that can return results that don't apply to the current screen.

Click **OK** to save the changes to the Local/Global Action and return to the *Speech-to-Text* dialog box.

- 11 When you have configured speech-to-text for the modified screen, click **OK**.

The *Speech-to-Text* dialog box closes and the speech-to-text action is added to the modified screen.

If you need to modify the speech-to-text action, right-click the action in the Descriptive View and select **Edit Speech-to-Text**.

Adding Scripts to Reformatted Screens

You can also use the screen reformatter to launch Speakeasy scripts when the modified screen is first displayed. The script can provide additional functionality to the modified screen.

To add a script from the screen reformatter:

- 1 Click **Edit > Add Scripting Support**.

-Or-

In the Initial Screen View, right-click and select **Add Scripting Support** from the context menu.

The *Scripting Support* dialog box appears.

- 2 Select the desired script from the **Script to launch** drop-down menu.

NOTE: The **Script to launch** drop-down menu only displays scripts that have been saved in the Script Editor. For more information about Terminal Emulation scripting, refer to *Wavelink Terminal Emulation Scripting Reference Guide*.

- 3 Select how the screen reformatter will handle the script:
 - If you want to ensure that the script does not run multiple times, enable the **Don't launch the script if it is already running** checkbox.
 - If you want the script to abort when the modified screen is no longer in use, enable the **Stop the script when reformatting changes** option.



- If you want the screen reformatter to wait until the script has completed before it proceeds to the next action for the screen, enable the **Wait for the script to finish before performing the next supporting action**.

4 Click **OK**.

The *Scripting Support* dialog box closes and the script is added to the Supported Actions listed in the Descriptive View.

Ordering Screen Actions

When you have multiple actions on the modified screen, you can determine the order in which the actions occur. The actions are listed in the Descriptive View in the Supported Actions section. Supported screen actions are Speakeasy and scripting actions. You should list the text-to-speech actions before any speech-to-text actions.

To arrange actions in the desired order:

- 1 Select the desired action in the Modified Screen View.
- 2 Click the **Edit** menu.

-Or-

Right-click in the Initial Screen View.

A menu list appears.

- 3 Select one of the following options:
 - **Make First Action** to designate the current action as the first action to be performed.
 - **Make Next Action** to designate the current action as the next action in the list.
 - **Make Previous Action** to designate the current action as the previous action in the list.
 - **Make Last Action** to designate the current action as the last action to be performed.
 - **Delete Action** to remove the current action.

Deploying Reformatted Screens

Once you have finished modifying your screens, click **Save** to save your screens and exit the screen reformatter. The TE Client will update with the new screens the next time the device syncs.



Chapter 4: Speakeasy and Scripting

Use the Terminal Emulation Script Editor to create and execute scripts that automate Speakeasy processes, such as selecting the right voice profile or changing the speed of text-to-speech. A script can be started using the Screen Reformatter, by calling it from another script or a web page, or from the TE Client by clicking **Options > Scripting > Execute Script**.

NOTE: When using text-to-speech and scripting, you must convert an integer variable to a string variable in order for text-to-speech to read it.

This section provides overview information for the following topics:

- [Creating Scripts](#)
- [Activating Scripts](#)

For examples of Speakeasy scripts, see [Sample Speakeasy Scripts](#) on page 43.

Creating Scripts

The following steps provide an overview of how you manually create a Speakeasy script. For more detailed information about these steps or scripting actions, refer to *Wavelink Terminal Emulation Scripting Reference Guide*.

- 1 Name the script.
- 2 Select an activation method.
- 3 Build the script code. In the **Actions** tab, create the code, line-by-line, that describes how you want the script to perform.

NOTE: For actions specific to Speakeasy, see [Speakeasy Settings](#) on page 39.

- 4 Create any variables that you need for your script in the **Boolean Variables**, **Number Variables**, or **String Variables** tabs.
- 5 Assign host profiles that can perform the script.

Activating Scripts

When a script is created, it has an activation method assigned that specifies how it is activated. This section provides information about activating scripts using each of the available activation methods:

- [Select From Menu](#)



- [On Key Combination](#)
- [When Session Connects](#)
- [On Barcode, MSR, or RFID Scan](#)
- [On Screen Update](#)
- [From Web Pages](#)

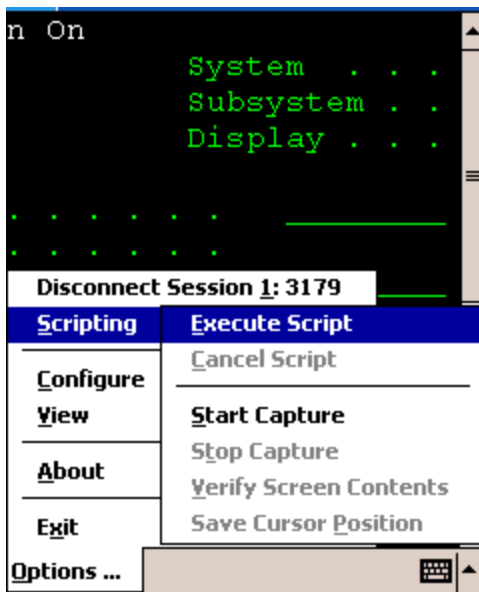
NOTE: If you plan to call a script from another script, from the screen reformatter, or from a web page, you do not need to select an activation method for the script in the Script Editor.

Select From Menu

With this activation method, you can activate the script from the **Options** or **Term** menu of the TE Client.

To activate a script from the TE Client menu:

- 1 Launch the TE Client on the mobile device.
- 2 From the **Options** or **Term** menu, select **Scripting > Execute Script**.



Executing Scripts from the Menu

- 3 If more than one script is available for the current host profile, select which script you want to use from the list.

NOTE: This option is not available if a script is running for the current session or if the session is not connected.



On Key Combination

With this activation method, the script activates when you press the specified key combination (as long as it is currently possible for script to run).

To activate a script on a specific key combination:

- 1 Launch the TE Client.
- 2 Enter the key combination you assigned to execute the script.

When Session Connects

With this activation method, the script activates when a session connects using the specified host profile.

To execute a script when the session connects:

- 1 Launch the TE Client.
- 2 From the **Options** or **Term** menu, select **Connect**.
- 3 Select the host to which you want to connect.
- 4 Click **OK**.

The script runs upon connection.

On Barcode, MSR, or RFID Scan

With this activation method, the script activates with each barcode, MSR, or RFID scan.

On Screen Update

With this activation method, the script activates (if activation is allowed) every time the text on the emulation screen changes. This includes updates from the host or when the user presses a key and the key value appears on the screen.

From Web Pages

A Wavelink script can be executed from a web page using the `wls` type, followed by the script name. If you plan to launch a script from a web page, do not select a script activation method when you create the script.

Executing Scripts from Web Pages Example 1

This example launches a script called `WebAuto` when the web page first loads.

```
<title>TE70 Test1 - Launch Telnet Scripts</title>  
<meta http-equiv="OnStartup" content="wls:WebAuto">
```



Executing Scripts from Web Pages Example 2

This example launches a script called `WebClick` when a user clicks the hyperlink here on the web page.

```
<p>  
Click <a href="wls:WebClick">here</a> to launch the WebClick script.  
</p>
```

From the Screen Reformatter

A Wavelink script can be launched for a specific screen using the screen reformatter. When a script is added for a screen in the reformatter, it is considered a screen action. When the correct screen appears on the device, screen actions (including speech-to-text and text-to-speech actions) are performed in the order they appear in the Descriptive View. For information on adding a script to a screen using the screen reformatter, see [Adding Scripts to Reformatted Screens](#) on page 21.



Chapter 5: Installation and Licensing

Speakeasy consists of multiple packages (in addition to the Terminal Emulation Client) that must be deployed to the mobile device. You can install Speakeasy packages on the mobile device using Wavelink Avalanche or Microsoft ActiveSync. You can also install Speakeasy packages for the Wavelink TE Client for Windows.

To improve speech recognition and positive results, Speakeasy speech-to-text uses grammar files to define expected input. If you are using speech-to-text, you should use the Grammar File Manager to modify and select the grammar files you plan to use before you install speech-to-text on the device. The Grammar File Manager is available whether you are installing via Avalanche or ActiveSync.

This section provides the following information on installing Speakeasy:

- [Installation Requirements](#)
- [Installing Speakeasy Using Avalanche](#)
- [Installing Speakeasy Using ActiveSync](#)
- [See "Installing Speakeasy on a PC"](#)
- [Licensing](#)
- [Uninstalling Speakeasy](#)

NOTE: Wavelink supports some third-party installation applications. For more information about supported installation options for your device, please see the Wavelink Web site. If you choose to use a third-party application to configure and install Speakeasy, please see the documentation for that application.

Installation Requirements

This section lists the hardware, software, and memory requirements that Speakeasy requires for best performance:

- Mobile device with headset microphone with a signal-to-noise ratio (SNR) better than 20 dBA
- 128 MB RAM; or 64 MB RAM with an SD card; or 64 MB RAM with 128 MB Flash Memory
- Wavelink Terminal Emulation Client version 7.0 or newer

In addition, if you want to be able to save your voice profiles and distribute them using Wavelink Avalanche, you will need:



- Avalanche Mobility Center or Avalanche Site Edition. You must use a mobile device server that is version 5.2 or newer.
- Wavelink Avalanche Enabler version 5.0-10 or newer.

Installing Speakeasy Using Avalanche

When you install Speakeasy on a device using Avalanche, each device that will be using Speakeasy (either speech-to-text or text-to-speech) must receive the Speech Registration package. If you are planning to use speech-to-text, you will also need the following packages:

- Speech-to-text Base package.
- Speech-to-text Language packages. Choose the package(s) specific to the language(s) you will be using.

If you are planning to use text-to-speech, you will need the following packages:

- Text-to-speech Base package.
- Text-to-speech Vocalizer packages. Choose the package(s) specific to the language(s) you will be using.

You should have the Terminal Emulation Client installed before installing Speakeasy on the device.

NOTE: The following instructions are for installing Speakeasy using the Avalanche Web Console. For instructions on installing using the Java Console, see the *Avalanche Java Console User Guide*.

To install Speakeasy using Avalanche:

- 1 Download the applicable packages from the Wavelink Web site.
- 2 If desired, use Avalanche to create a new software profile for the Speakeasy packages.
- 3 Add the Speakeasy packages to the software profile and navigate to the Software Profile Details page.
- 4 If you want to specify where on the device you want the Speakeasy packages installed, click **Configure** next to the name of the registration package. Select the TE Speech Configuration utility and click **Launch Config**. When the package configuration launches, provide the installation path so that the files install to the correct location. You should install the packages in a location with sufficient memory that is cold-boot persistent.
- 5 If you are using speech-to-text, click **Configure** next to the speech-to-text base package. Select the Grammar File Manager from the list and click **Launch Config**. Use the grammar



file manager to configure the grammars the device will need. The grammars define which words the speech-to-text engine will recognize. For more information on grammar files, see [Using the Grammar File Manager](#) on page 4.

- 6 Apply the software profile to the location(s) where you want the packages to be distributed and enable the profile.
- 7 Perform a deployment.
- 8 If desired, from the Enabler interface on the mobile device, click **File > Connect** to immediately connect to the mobile device server, download the packages, and install them.

NOTE: For more information on performing tasks from the Avalanche Console, refer to the [Avalanche User Guide](#).

Installing Speakeasy Using ActiveSync

Speakeasy can be installed on a mobile device using Microsoft ActiveSync. There are separate installation files for speech-to-text and text-to-speech.

If you are planning to use speech-to-text, you will need the following package:

- Speech-to-text ActiveSync installation package.

If you are planning to use text-to-speech, you will need the following package:

- Text-to-speech Vocalizer packages. Choose the package(s) specific to the language(s) you will be using.

You should have the Terminal Emulation Client installed before installing Speakeasy on the device.

[To install speech-to-text using ActiveSync:](#)

- 1 Download the applicable packages from the Wavelink Web site.
- 2 Establish an ActiveSync connection with the device.
- 3 Launch the ActiveSync installation speech-to-text package by double-clicking on the file.
- 4 The Setup Wizard appears. Click **Next**.
- 5 The License Agreement appears. Accept the agreement by clicking **I Agree**.
- 6 The Choose Installation Location screen appears. Click **Browse** to select a different installation location, or click **Install** to use the default location.

The Speech-to-text files are installed locally.



- 7 When the **Completing the Setup Wizard** screen appears, ensure that the **Run Wavelink Speech-to-Text ActiveSync Support** checkbox is enabled and click **Finish**. The wizard will close and the Speech-to-Text Installer will launch.
- 8 From the **Languages** list, select the languages you will be using.
- 9 To choose the grammars that will be installed on the device, click **Grammar File Manager**. For more information on managing grammar files, see [Using the Grammar File Manager](#) on page 4.
- 10 Click **Install** to install speech-to-text on the device. When the files are finished installing, the *Installation Successful* screen appears. Click **OK**.

To install text-to-speech using ActiveSync:

- 1 Download the applicable packages from the Wavelink Web site.
- 2 Establish an ActiveSync connection with the device.
- 3 Launch the ActiveSync installation text-to-speech package by double-clicking on the file.
- 4 The Setup Wizard appears. Click **Next**.
- 5 The License Agreement appears. Accept the agreement by clicking **I Agree**.
- 6 The Choose Installation Location screen appears. Click **Browse** to select a different installation location, or click **Install** to use the default location.

The text-to-speech files are installed locally.

- 7 When the **Completing the Setup Wizard** screen appears, ensure that the **Run Wavelink Text-to-Speech ActiveSync Support** checkbox is enabled and click **Finish**. The wizard will close and the Text-to-Speech Vocalizer Installer will launch.
- 8 From the **Languages** list, select the languages you will be using.
- 9 Click **Install** to install text-to-speech on the device. When the files are finished installing, the *Installation Successful* screen appears. Click **OK**.

Installing Speakeasy on a PC

Speakeasy can be installed on a Windows PC that has the Windows TE Client installed. Depending on your organization's needs, you may choose to install only speech-to-text, or only text-to-speech. Download the files from the Wavelink Web site and copy them to the computer where the TE Client is installed.

To install speech-to-text:

- 1 Double-click the Speech-to-Text executable.

The *Wavelink SpeechToText Support Setup Wizard* appears.



- 2 Follow the instructions in the Speech-to-Text Support Setup Wizard to complete the installation.
- 3 When the installation is complete, click **Finish** to exit the Wizard.

To install text-to-speech:

- 1 Double-click the Text-to-Speech Vocalizer executable.
The *Wavelink TextToSpeech Vocalizer Support Setup Wizard* appears.
- 2 Follow the instructions in the Text-to-Speech Vocalizer Support Setup Wizard to complete the installation.
- 3 When the installation is complete, click **Finish** to exit the Wizard.

Licensing

Speakeasy requires a separate license in addition to the standard Terminal Emulation licenses. You can use Speakeasy without a license, but you will be limited to the demo version. Speakeasy licenses can be distributed using the same method as other TE licenses. For information on licensing methods or distributing licenses, see the *Terminal Emulation Client User Guide*. To obtain licenses, please contact Wavelink Customer Service.

Uninstalling Speakeasy

Speakeasy can be uninstalled from the PC you used to install or from the mobile device. Uninstalling Speakeasy does not uninstall the TE Client.

- [Uninstalling Speakeasy from a Mobile Device](#)
- [Uninstalling Speakeasy from a PC](#)

Uninstalling Speakeasy from a Mobile Device

Speakeasy is installed on a mobile device using Avalanche or ActiveSync. To uninstall Speakeasy from a mobile device, use the same application that you used to install.

NOTE: The following instructions are for uninstalling Speakeasy from all devices. You can also uninstall Speakeasy from a specific device or region by changing how you want the Speakeasy software profile applied.

To uninstall Speakeasy using Avalanche:

- 1 From the Avalanche Console, navigate to the profile containing the Speakeasy packages.
- 2 Select the Speakeasy packages and click **Delete**.



- 3 Deploy your changes to the mobile device. When a package is deleted from the Avalanche Console, the package on the device is considered orphaned. You can delete orphaned packages from devices using a mobile device profile or by using the **Update Now** option on the Avalanche Console.

To uninstall Speakeasy using ActiveSync:

- 1 Create an ActiveSync connection to the device.
- 2 From the PC, click **Start > Programs > Wavelink SpeechToText ActiveSync Support > Install to Device** to uninstall the speech-to-text files.

-Or-

From the PC, click **Start > Programs > Wavelink TextToSpeech ActiveSync Support > Install to Device** to uninstall the text-to-speech files.

- 3 The Wavelink Installer appears. In the **Installation Location** drop-down menu, select the location where the files were originally installed.
- 4 Ensure that all the languages are deselected and click **Install**.
- 5 The files are removed from the device.

Uninstalling Speakeasy from a PC

Speakeasy can be used for a Windows TE Client or deployed from a PC to a mobile device via ActiveSync or Avalanche. If you used Avalanche to install, you do not need to uninstall from the PC.

To uninstall Speakeasy for a Windows TE Client from a PC:

- To uninstall speech-to-text support, click **Start > Programs > Wavelink SpeechToText Support > Uninstall SpeechToText Support**.

-Or-

- To uninstall text-to-speech support, click **Start > Programs > Wavelink TextToSpeech Vocalizer Support > Uninstall TextToSpeech Vocalizer Support**.

To uninstall Speakeasy ActiveSync Support from a PC:

- To uninstall speech-to-text ActiveSync support, click **Start > Programs > Wavelink SpeechToText ActiveSync Support > Uninstall SpeechToText ActiveSync Support**.

-Or-

- To uninstall text-to-speech ActiveSync support, click **Start > Programs > Wavelink TextToSpeech ActiveSync Support > Uninstall TextToSpeech ActiveSync Support**.



Chapter 6: Using Speakeasy

Once Speakeasy is installed on the device, the TE Client has tools to optimize Speakeasy. You can calibrate the microphone or text-to-speech options, create a voice profile, or enable the Speakeasy listening indicator to show when speech-to-text is listening.

When you create voice profiles, each person can create his own profile in order to optimize results. If you are using Speakeasy in conjunction with Avalanche, voice profiles can be stored on the mobile device server and distributed to devices as needed. Otherwise, each user will need to create a voice profile on each device he plans to use.

- [Calibrating the Microphone](#)
- [Configuring the Text-to-Speech Options](#)
- [Creating a Voice Profile](#)
- [Using the Speakeasy Listening Indicator](#)

Calibrating the Microphone

On some mobile devices, you can calibrate the microphone to optimize the speech detection in your current environment. The microphone settings that you select will become the default values for future speech-to-text processing on the mobile device. It is recommended that you calibrate the microphone before initial use.

You can perform a quick calibration that automatically sets values according to your speaking voice and the background noise, or you can perform a full calibration, manually setting the values using slider bars. Before you begin calibration, ensure that your headset is connected and working properly.

When the speech-to-text engine hears input, it tries to match it to the expected phrases in the available grammar files. The degree to which the spoken phrase and an expected response match is called the confidence value. A higher confidence value indicates a better match. If the confidence value is too low, the input is rejected.

To perform a quick calibration:

- 1 From the TE Client, click **Options > Configure > Microphone**.

The *Options* dialog box appears.

- 2 Click **Quick Calibration**.

The *Language* dialog box appears.

- 3 Select the language you want to use from the drop-down list and click **OK**.



The voice prompts will guide you through a prompt for speaking and a prompt for silence. When the calibration is complete, the new settings are applied and become the defaults for the device. For information on modifying the calibration prompts using the Grammar File Manager, see [Modifying User Prompts](#) on page 11.

To perform a full calibration:

- 1 From the Telnet **Options** menu, select **Configure > Microphone**.

The *Options* dialog box appears.

- 2 Click **Full Calibration**.

The *Test Settings* dialog box appears.

- 3 From the **Language** drop-down menu, select the language you will be using for speech-to-text conversion.

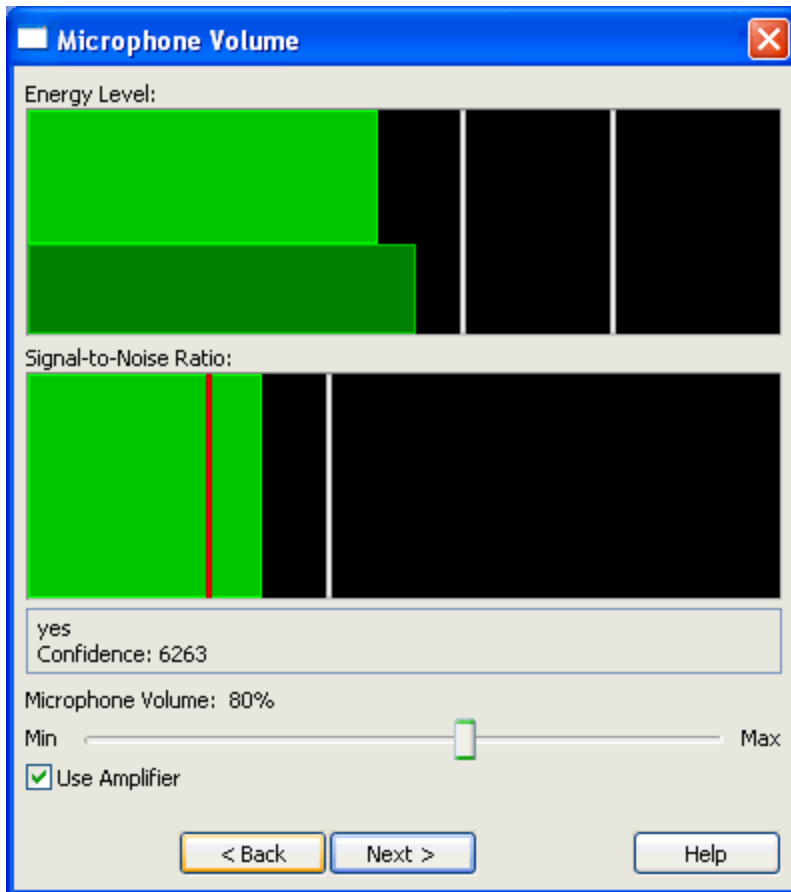
- 4 From the **Grammar** drop-down menu, select the grammar you will be using for speech-to-text conversion.

- 5 If you have a session currently connected, you can use the speech-to-text settings for that session instead of the default settings. Select the session from the **Session** drop-down menu.

- 6 Click **Next**.

The *Microphone Volume* dialog box appears.





Microphone Volume Dialog Box

- 7 Use the **Microphone Volume** slider to adjust the Energy Level.

The Energy Level indicates how much input the microphone is receiving. The current energy level indicated by the top, light-green bar should fall between the two white bars when the user is speaking. The darker bar underneath is the maximum level reached within the past 5 seconds.

NOTE: The signal-to-noise ratio will automatically adjust according to the noise level in your environment.


When you speak one of the words or phrases included in the grammar selected in step 4, the phrase appears in the box underneath the Signal-to-Noise Ratio. This box also displays the confidence value.

- 8 Click **Next**.

The *Other Settings* dialog box appears. The **Speech Detection State** indicates whether the speech engine detects the user's speech. When the user is not speaking, the state should be



red. When the speech engine detects something that may be speech, the state turns yellow. When the speech engine is certain it is detecting speech, the state turns green.

Speech Detection State: 

Speech Detection State

- 9 Use the **Absolute Threshold** and **Sensitivity** sliders to adjust the speech detection settings.

The **Absolute Threshold** value indicates the minimum amount of energy required to indicate when speech begins. Adjust this value so that the user's speech causes the state to turn yellow immediately, but any background noise causes the state to remain red.

The **Sensitivity** value determines when the speech engine begins detecting a user's speech. A higher sensitivity value means the speech engine will react more easily; a lower value means the speech engine will pick up less background noise.

- 10 Click **Next**.

The *Record/Playback* dialog box appears. If you want to check the input for your microphone, you can use the options on this dialog box to record and play back microphone input as a .wav file.

- 11 Click **Finish**.

The microphone calibration wizard disappears and the new microphone settings are applied. These settings will be the defaults for the device.

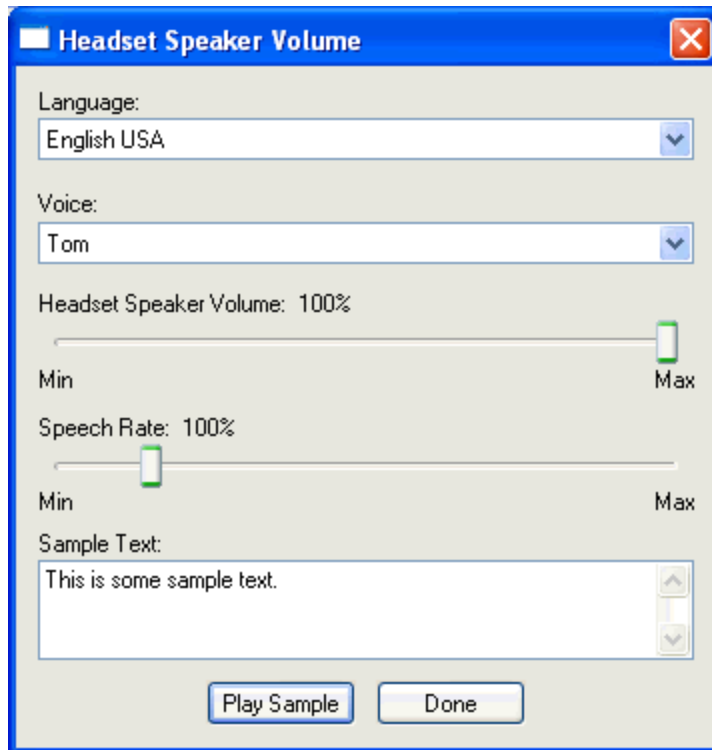
Configuring the Text-to-Speech Options

On some mobile devices, you can configure the speaker volume and speed settings. The speaker settings that you select will become the default values for future text-to-speech processing on the mobile device.

To configure text-to-speech options:

- 1 From the TE Client, click **Options > Configure > Speaker Volume**.





Speaker Settings

The *Headset Speaker Volume* dialog box appears.

- 2 From the **Language** and **Voice** drop-down menus, select the desired language and voice. The options that appear in the **Voice** drop-down menu depend on the language selected.
- 3 Use the sliders to set the volume and how fast the text-to-speech is pronounced.
- 4 If desired, type **Sample Text** in the box and click **Play Sample** to check your settings.
- 5 When you have configured the settings, click **Done**.

The speaker settings wizard disappears and the new speech settings are applied.

Creating a Voice Profile

A voice profile provides the Speakeasy engine with speech samples in order to get more positive results during speech-to-text. You can create a voice profile from the TE Client after Speakeasy is installed.

Before you create a voice profile, you should have the grammar files configured. When you configure grammar files, you select the specific words and phrases that users will train on to create voice profiles. For more information on selecting the terms for voice profiles, see [Specifying Voice Profile Training Options](#) on page 11.



NOTE: If you create a voice profile and later add a grammar file that requires additional training, you will be prompted to complete the training for that grammar file when you use it.

Once you have created a voice profile, use a script to implement the profile. For an example script to enter the username and use a voice profile, see [Using a Voice Profile](#) on page 43.

If you are using an Avalanche Enabler and mobile device server, voice profiles are archived on the mobile device server. If you enter the same username on a different device, the TE Client will automatically retrieve the voice profile from the mobile device server and use it.

To create a voice profile:

- 1 Ensure that your headset is connected.
- 2 From the TE Client, click **Options > Configure > Microphone**.
- 3 The *Options* dialog box appears. Click **User Training**.
- 4 The *Language* dialog box appears. Select the language you want to use from the drop-down menu and click **OK**.
- 5 The *User Name* dialog box appears. Type the user name in the text box and click **OK**.
- 6 Speakeasy will vocally prompt you to say the words and phrases selected for user training when the grammar files were configured. The user will be prompted to say each word or phrase twice. After all the selected terms have been recorded, the profile is saved.

Using the Speakeasy Listening Indicator

The Speakeasy listening indicator appears as a small icon on the TE Client screen whenever the speech-to-text engine is listening for input. The indicator is enabled or disabled from the emulation parameters for the TE Client.

To enable the Speakeasy listening indicator:

- 1 Access the Configuration Manager for the TE Client. For information on how to access the Configuration Manager, see the *Terminal Emulation Client User Guide*.
- 2 In the Indicators section, right-click the option **Speech-to-Text Listening Indicator** and select **Edit** from the context menu.
- 3 The *Speech-to-Text Indicator Light* dialog box appears. Select **Yes** from the drop-down menu and click **OK**.

When you use the Client to connect a session that uses speech-to-text, the listening indicator will appear whenever the Client is waiting for verbal input.



Chapter 7: Speakeasy Settings

This section lists settings supported by Speakeasy. These settings can be modified using the screen reformatter or a script; some of them are modified using the microphone calibration or text-to-speech configuration. For detailed descriptions and value ranges, see the Speakeasy Settings section of the *Terminal Emulation Scripting Reference Guide*.

Text-to-Speech Settings

Setting	Description
tts_calibrate	Opens the speaker volume calibration wizard.
tts_external_speaker_setting	Speaker setting for use on Motorola/Symbol mobile devices.
tts_frequency	Indicates the sampling frequency.
tts_language_long	Displays the full name of the language currently being used.
tts_language_short	Displays the three-letter abbreviation of the language currently being used.
tts_pitch	Indicates the pitch level of spoken text.
tts_rate	Indicates the speed level.
tts_readmode	Indicates how text should be separated.
tts_voice	Indicates the name of the voice that is currently selected.
tts_volume	Indicates the sound level.
tts_waitfactor	Indicates the length of the pause between messages.



Speech-to-Text Settings

Setting	Description
stt_accuracy	This value affects the trade-off between CPU load, memory requirements, and accuracy.
stt_adjust_gain	This feature allows the engine to automatically increase and decrease the microphone input volume.
stt_beep_threshold	If the confidence value for a result is below this value, then a negative acknowledgement beep will not be played.
stt_calibrate	Opens the microphone calibration wizard.
stt_calibration_silence	Sets how long the user is expected to remain silent during a quick microphone calibration.
stt_confidence	Indicates the minimum difference in confidence required between the top two speech-to-text results for the top result to be accepted.
stt_expanded	Use this to get the confidence value along with the speech-to-text result.
stt_fx_detect_start	Indicates the action the speech engine should take before attempting to determine what the user is saying.
stt_fx_microphone	Tells the speech engine the distance between the user and the microphone.
stt_fx_min_duration	Indicates the minimum duration (in ms) of speech before speech detection is activated.
stt_fx_sensitivity	Indicates the speech detection sensitivity.
stt_fx_silence	Indicates the milliseconds of silence used to indicate the user is done speaking.
stt_fx_threshold	Indicates the amount of energy the microphone input must have before the speech detection is activated.



Setting	Description
stt_idle_timeout	Indicates the total milliseconds for the engine to continue collecting results following the last result or timeout.
stt_language_long	Displays the full name of the language currently being used.
stt_language_short	Displays the three-letter abbreviation of the language currently being used.
stt_logging	Creates a Speech-to-Text log file in the root folder of the device.
stt_logging_audio	Sets the engine to log speech-to-text attempts as .wav files.
stt_logging_engine	If set to 1, the speech-to-text engine will create a log file in the root folder of the device.
stt_pool_size	Sets the number of terms the engine will examine closely for the best match.
stt_preserve	Causes the speech engine to save the current engine state for use later.
stt_priority	Determines how aggressively the microphone input is collected and speech analysis is performed.
stt_processing	Indicates the action the speech engine should take when returning a grammar result.
stt_reset	Modifies engine adaptation speed and/or saved engine information.
stt_reset_session_delay	Indicates the total milliseconds for the speech engine to wait for a valid response before reverting back to the last saved state.
stt_result_sound	Causes a sound to play for result recognition.
stt_save_increase	Increases the threshold for saving a new engine state as time progresses.



Setting	Description
stt_save_session_delay	Indicates the total milliseconds for the speech engine to wait before saving the next current state.
stt_save_threshold	Directs the speech engine to save the state if the result confidence is greater than the result confidence for <code>stt_threshold</code> and <code>stt_save_threshold</code> combined.
stt_server_timeout	When uploading or downloading user training data, the value for this setting is how long (in seconds) the Client will wait for a response from the Avalanche server.
stt_size	Displays the size of the speech-to-text engine being used.
stt_special_sounds	Indicates how the speech engine should interpret special sounds.
stt_threshold	Indicates the minimum amount of confidence for the most-likely result that will be accepted.
stt_timeout	Indicates the total milliseconds (ms) for the system to wait before responding to the speaker.
stt_use_jumpback	Sets a buffer to check if the engine is processing speech.
stt_use_word_ids	Enables support for Word IDs (the <code>!id</code> directive) in grammar files.
stt_volume	Indicates the current volume of the microphone input.



Sample Speakeasy Scripts

This section contains example scripts that perform various Speakeasy functions. You can use the Script Editor to modify and customize scripts as desired. For more information on scripting or the Script Editor, see the *Terminal Emulation Scripting Reference Guide*.

- [Using a Voice Profile](#)
- [Reading the Screen Aloud](#)
- [Displaying Speech Results in a Dialog Box](#)
- [Changing Text-to-Speech Modes](#)
- [Speech Demo Script](#)
- [Acquiring a Pick Quantity in the Industrial Browser](#)

Using a Voice Profile

The following example script asks the user for his username and then switches to the voice profile associated with that username.

```
Script( Set_Speech_User )
String( sSpeechToTextUserName, True )
Activate( Connection )
    Comment: If the persistent variable doesn't have a value, the default
is the username the Speech-to-Text engine is using.
    If( String_Empty( sSpeechToTextUserName ) )
        sSpeechToTextUserName = Speech_To_Text_Get_User_Name
    End_If

    Comment: Ask the user the username to use. The last username used will
be the default.
    sSpeechToTextUserName = Ask_String_Lowercase( "What is your Speech
Username?", "Speech", 1, 100, sSpeechToTextUserName )

    Comment: Tell the Speech-to-Text engine the username.
    Speech_To_Text_Change_User_Name( sSpeechToTextUserName )
Return
```

Reading the Screen Aloud

The following example script converts the current Terminal Emulation Client screen into speech that the user can hear.

```
nNumRows=Get_Screen_Rows
nCurrentRow=1
While( Number_Less_Than_Or_Equal( nCurrentRow, nNumRows ) )
    Speech_From_Text( Get_Screen_Text( nCurrentRow, 1 ), FALSE )
    nCurrentRow=Number_Plus( nCurrentRow, 1 )
End_While
Return
```



Displaying Speech Results in a Dialog Box

The following example script prompts the user for a number, converts the spoken number into text, and displays it in a dialog box on the mobile device. This script requires the `connected_digits` grammar file.

```
String(sResult)
Speech_From_Text("Say a number",FALSE)
Speech_To_Text(sResult, "connected_digits")
Ask_OK(sResult,"Number Returned")
Return
```

In this example the number is treated as a string because both the functions `Speech_To_Text` and `Ask_OK` require a string. However, it could be converted to a number if needed by using the following line:

```
nResult = String_To_Number_Decimal (sResult)
```

Changing Text-to-Speech Modes

When you use text-to-speech, the mode determines if it reads character strings as words or pronounce each letter separately. The text-to-speech mode is configured using the `tts_readmode` setting. The following example script demonstrates how to change the text-to-speech mode working in a voice-picking environment.

The following script also repeats picking information from the screen when the user requests it. This script is named `sapRepeatPrompts` and is designed to work with the script available in [Acquiring a Pick Quantity in the Industrial Browser](#) on page 47.

```
Script( sapRepeatPrompts )
String( strPromptBin )
String( strPromptBinValue )
String( strPromptMaterial )
String( strPromptMaterialValue )
String( strPromptBatch )
String( strPromptBatchValue )
String( strPromptQuantity )
String( strPromptQuantityValue )
String( strSearchString )
Number( nReadMode )
Number( nIndex )

    Comment: This function repeats the prompts when called from Command and
Control.
    Speech_To_Text_Cancel

    Comment: The Speech-to-Text and Text-to-Speech languages are specified.
    Speech_Change_Setting( "tts_language_short", Speech_Find_Setting_
Value("tts_language_short", "enu", FALSE))
```



```

    Speech_Change_Setting( "stt_language_short", Speech_Find_Setting_Value(
"stt_language_short", "enu", FALSE))

    Comment: Ensure read mode is in sentence mode.
    nReadMode = Speech_Find_Setting_Value( "tts_readmode", "sentence",
FALSE )
    Speech_Change_Setting( "tts_readmode", nReadMode )

    Comment: Annunciate Bin prompt.
    Speech_From_Text( strPromptBin, TRUE )

    Comment: Change read mode to character mode to annunciate bin number.
    nReadMode = Speech_Find_Setting_Value( "tts_readmode", "character",
FALSE )
    Speech_Change_Setting( "tts_readmode", nReadMode )

    Comment: Annuciate Bin number.
    Speech_From_Text( strPromptBinValue, TRUE )

    Comment: Change read mode back to sentence mode to annunciate material
prompt.
    nReadMode = Speech_Find_Setting_Value( "tts_readmode", "sentence",
FALSE )
    Speech_Change_Setting( "tts_readmode", nReadMode )

    Comment: Annunciate Material prompt.
    Speech_From_Text( strPromptMaterial, TRUE )

    Comment: Change to character mode to annunciate material.
    nReadMode = Speech_Find_Setting_Value( "tts_readmode", "character",
FALSE )
    Speech_Change_Setting( "tts_readmode", nReadMode )

    Comment: Annuciate Material.
    Speech_From_Text( strPromptMaterialValue, TRUE )

    Comment: Change to sentence mode to annunciate batch prompt.
    nReadMode = Speech_Find_Setting_Value( "tts_readmode", "sentence",
FALSE )
    Speech_Change_Setting( "tts_readmode", nReadMode )

    Comment: Annunciate Batch prompt.
    Speech_From_Text( strPromptBatch, TRUE )

    Comment: Change to character mode to annunciate Batch.
    nReadMode = Speech_Find_Setting_Value( "tts_readmode", "character",
FALSE )
    Speech_Change_Setting( "tts_readmode", nReadMode )

    Comment: Annuciate Batch.
    Speech_From_Text( strPromptBatchValue, TRUE )

    Comment: Change to sentence mode to annunciate pick quantity.
    nReadMode = Speech_Find_Setting_Value( "tts_readmode", "sentence",
FALSE )
    Speech_Change_Setting( "tts_readmode", nReadMode )

    Comment: Extract quantity value left of the period.
    strSearchString = "."
    nIndex = String_Find_First( strPromptQuantityValue, strSearchString,

```



```

TRUE )
    strPromptQuantityValue = String_Left( strPromptQuantityValue, nIndex )
    strPromptQuantityValue = String_Trim_Spaces_Start(
strPromptQuantityValue )

    Comment: Annunciate Pick Quantity prompt.
    Speech_From_Text( strPromptQuantity, TRUE )

    Comment: Annunciate Pick Quantity Value.
    Speech_From_Text( strPromptQuantityValue, TRUE )

```

Return

Speech Demo Script

The following example script creates the four buttons on the screen: **Digits, State, Play Screen, Done**. The **Digits** and **State** buttons allow the user to input a verbal response which is then displayed on the screen. The **Play Screen** button causes the mobile device to read back all the text on the screen, and the **Done** button allows the user to exit the script.

```

While_Not( bExit )
If_Not( bButtonsVisible )
    Button_Create_View( "Digits", 999, 1, 6, bGetDigits )
    Button_Create_View( "State", 999, 16, 5, bGetState )
    Button_Create_View( "PlayScreen", 1000, 1, 11, bPlayScreen )
    Button_Create_View( "Done", 1000, 13, 4, bExit )
End_If
Wait_For_Screen_Update
If( bPlayScreen )
    bPlayScreen=FALSE
    Button_Remove_All
    bButtonsVisible=FALSE
    Delay( 1 )

    numRows=Get_Screen_Rows
    nCurrentRow=1
    While( Number_Less_Than_Or_Equal
(nCurrentRow, numRows )
        Speech_From_Text( Get_Screen_Text( nCurrentRow, 1 ), FALSE )
        nCurrentRow=Number_Plus( nCurrentRow, 1 )
    End_While
End_If
If( bGetDigits )
    bGetDigits=FALSE
    Button_Remove_All
    bButtonsVisible=FALSE

    Message( "Say 1 or more digits...", 0 )
    szResult=""
    Speech_To_Text( szResult, "connected_digits" )
    Message_Clear
    szResult=String_Strip_Characters( szResult, "", FALSE )

```



```

        Keypress_String(szResult)
End_If
If(bGetState)
    bGetState=FALSE
    Button_Remove_All
    bButtonsVisible=FALSE

    Message("Say a USA state...",0)
    szResult=""
    Speech_To_Text(szResult,"usa_states")
    Message_Clear
    Keypress_String(szResult)
End_If
End_While
Button_Remove_All
Return

```

Acquiring a Pick Quantity in the Industrial Browser

The following example script is designed to work with the Wavelink Industrial Browser and allows the user to speak a number 0-9999 (referred to as the pick quantity), ask for help, go back, disconnect the session, or clear the field.

This script requires that text-to-speech, speech-to-text, and a custom grammar file named `sapqty_four_digits.bnf` is installed. The script also calls a JavaScript function: `clearFields`. The text of the grammar file and the JavaScript function are included below.

NOTE: This script calls another Wavelink script in order to repeat the information on the screen. The script `sapRepeatPrompts` is included in [Changing Text-to-Speech Modes](#) on page 44.

Acquiring a Pick Quantity Script

```

Comment: This function acquires the Pick Quantity via voice.
Comment: Ensure Speakeasy support has been installed.
If_Not( Speech_To_Text_Available )
    Ask_OK( "Speech-to-Text is not available.", "Error" )
    Return
End_If
If_Not( Speech_From_Text_Available )
    Ask_OK( "Text-to-Speech is not available.", "Error" )
    Return
End_If
Speech_To_Text_Cancel

Comment: The Speech-to-Text and Text-to-Speech languages are specified.
Speech_Change_Setting( "tts_language_short",
Speech_Find_Setting_Value("tts_language_short","enu",FALSE))
Speech_Change_Setting( "stt_language_short",
Speech_Find_Setting_Value("stt_language_short","enu",FALSE))

```



```

Comment: Acquire Pick Quantity via Speech-To-Text.
Comment: Initialize Speech-To-Text variables.
bSpeechStarted = FALSE
bSpeechDone = FALSE

Comment: Initialize confidence value. Speech uttered with confidence
values below this value will be rejected.
Speech_Change_Setting( "stt_threshold", 4500 )

Comment: Start Speech-To-Text if not already started.
Comment: This is needed so we start Speech_To_Text again if nothing was
stated before it times out.
If_Not( bSpeechStarted )
    Speech_From_Text( "Enter Quantity:", TRUE )

    Comment: With this Speech-To-Text function, the script waits for voice
input until stt_timeout value is reached if nothing is stated sooner.
    bSpeechStarted = Speech_To_Text_No_Wait( bSpeechDone,
    strSpeechResult, "sapqty_four_digits" )

    Comment: Wait_For_Screen_Update waits for speech as well.
    Wait_For_Screen_Update

End_If
If( bSpeechDone )
    Comment: If sSpeechResult is not empty it signifies that we received a
speech result.
    Comment: Command and Control is handled here as well.

    If_Not( String_Empty( strSpeechResult ) )

        If( String_Equal( strSpeechResult, "repeat", 0, TRUE ) )

            Comment: Set bRepeatPrompts so that the script sapRepeatPrompts will
be called.

            bRepeatPrompts = TRUE
            strSpeechResult = ""
            Speech_To_Text_Cancel
            Return

        End_If

        If( String_Equal( strSpeechResult, "back", 0, TRUE ) )

            Comment: Press F2 key.
            Comment: Keypress_Key("VT220", "F2")
            Ask_OK( "F2 Key Pressed", "Back Function" )
            bBackFunction = TRUE
            strSpeechResult = ""
            Speech_To_Text_Cancel
            Return

        End_If

        If( String_Equal( strSpeechResult, "clear", 0, TRUE ) )
            Comment: Call clearFields javascript function
            Web_Scripting( "javascript:clearFields();" )
            strSpeechResult = ""
            Return

        End_If

        If( String_Equal( strSpeechResult, "quit", 0, TRUE ) )
            Comment: Disconnect session.

```



```

        Speech_To_Text_Cancel
        Disconnect
        Return
    End_If
    If( String_Equal( strSpeechResult, "help", 0, TRUE ) )
        Comment: Annunciate help list.
        Speech_To_Text_Cancel
        Speech_From_Text("Help list annunciates this list.", TRUE)
        Speech_From_Text("Repeat prompts repeats prompts and data.",TRUE)
        Speech_From_Text("Go back moves to the previous screen.",TRUE)
        Speech_From_Text( "Clear fields clears target bin, material, batch,
and pick quantity fields.", TRUE )
        Speech_From_Text("Quit SAP disconnects the session.", TRUE)
        strSpeechResult = ""
        Return
    End_If
End_If
Speech_To_Text_Cancel
Return

```

Acquiring Pick Quantity Grammar File

The following is an example of a grammar file that will recognize a four-digit number. This grammar file is used with the Acquiring a Pick Quantity script example.

```

#BNF+AM V1.0;
/*****
GRAMMAR: sapqty_four_digits.bnf
Description: This is a grammar that recognizes a four-digit number, 0 to
9999. It also recognizes the commands repeat, back, quit, and clear.
What Can I Say? You can say any number from 0 to 9999. You can also
speak two, three, or four separate digits.
*****/
!grammar sapqty_four_digits;
!start <Speech>;
<Speech>: (edit | change quantity) <FourDigits> {@ = #2;} | <Command>;
<NonZeroDigit>: 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9;
<Digit>:
    0 {@ = "0";} |
    OH {@ = "0";} |
    <NonZeroDigit>;
<Teens>: 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19;
<Tens>:
    TEN {@ = "10";} |
    TWENTY {@ = "20";} |

```



```

TWENY {@ = "20";} |
THIRTY {@ = "30";} |
THIRDY {@ = "30";} |
FORTY {@ = "40";} |
FORDY {@ = "40";} |
FIFTY {@ = "50";} |
FIFDY {@ = "50";} |
SIXTY {@ = "60";} |
SIXDY {@ = "60";} |
SEVENTY {@ = "70";} |
SEVENDY {@ = "70";} |
EIGHTY {@ = "80";} |
EIGHDY {@ = "80";} |
NINETY {@ = "90";} |
NINEDY {@ = "90";} ;

```

<CombinedTens>:

```

TWENTY {@ = "2";} |
TWENY {@ = "2";} |
THIRTY {@ = "3";} |
THIRDY {@ = "3";} |
FORTY {@ = "4";} |
FORDY {@ = "4";} |
FIFTY {@ = "5";} |
FIFDY {@ = "5";} |
SIXTY {@ = "6";} |
SIXDY {@ = "6";} |
SEVENTY {@ = "7";} |
SEVENDY {@ = "7";} |
EIGHTY {@ = "8";} |
EIGHDY {@ = "8";} |
NINETY {@ = "9";} |
NINEDY {@ = "9";} ;

```

<ExactThreeDigits>:

```

<NonZeroDigit> HUNDRED [AND] ( <Teens> | <CombinedTens> <NonZeroDigit>
{@ = #1+#2;} | <Tens> | <NonZeroDigit> {@ = "0"+#1;} ) {@ = #1+#4;} |
<NonZeroDigit> HUNDRED {@ = #1+"00";} |
<NonZeroDigit> ( <Teens> | <CombinedTens> <NonZeroDigit> {@ = #1+#2;} |
<Tens> ) {@ = #1+#2;} |
<Teens> {@ = "0"+#1;} | <CombinedTens> <NonZeroDigit> {@ = "0"+#1+#2;} |
<Tens> {@ = "0"+#1;} |
<Digit> {@ = "00"+#1;} ;

```

<FourDigits>:

```

<NonZeroDigit> THOUSAND [AND] <ExactThreeDigits> {@ = #1+#4;} |
<NonZeroDigit> THOUSAND {@ = #1+"000";} |
<NonZeroDigit> HUNDRED [AND] ( <Teens> | <CombinedTens> <NonZeroDigit>
{@ = #1+#2;} | <Tens> | <NonZeroDigit> {@ = "0"+#1;} ) {@ = #1+#4;} |
<NonZeroDigit> HUNDRED {@ = #1+"00";} |

```



```

<NonZeroDigit> ( <Teens> | <CombinedTens> <NonZeroDigit> {@ = #1+#2;} |
<Tens> ) {@ = #1+#2;} |
<Teens> |<CombinedTens> <NonZeroDigit> {@ = #1+#2;} | <Tens> |
!repeat(<Digit> {@ = previous.@ + #1;},1,4);
<Command>: repeat prompts {@ = "repeat";} |
    go back {@ = "back";} | quit sap {@ = "quit";} |
    clear fields {@ = "clear";} | help list {@ = "help";}

```

JavaScript Function clearFields

The following JavaScript function is an example of a function that will clear the fields in the Wavelink Industrial Browser. This is used with the Acquiring a Pick Quantity script example.

```

/*****
This function is called to clear the fields when the clear command and
control command is spoken.

There are no inputs provided when this function is called.
*****/

function clearFields()
{
    //Clear target Bin field
    var strFieldNameClear = "s4000_binp[1]";
    var strFieldValueClear = "";
    document.getElementsByName(strFieldNameClear).value = strFieldValueClear;
    var arrayFieldNameClear = document.getElementsByName(strFieldNameClear);
    arrayFieldNameClear[0].value=strFieldValueClear;

    //Clear target Batch field
    var strFieldNameClear = "s4000_chargp[1]";
    var strFieldValueClear = "";
    document.getElementsByName(strFieldNameClear).value = strFieldValueClear;
    var arrayFieldNameClear = document.getElementsByName(strFieldNameClear);
    arrayFieldNameClear[0].value=strFieldValueClear;

    //Clear Pick qty field
    var strFieldNameClear = "s4000_qty[1]";
    var strFieldValueClear = "";
    document.getElementsByName(strFieldNameClear).value = strFieldValueClear;
    var arrayFieldNameClear = document.getElementsByName(strFieldNameClear);
    arrayFieldNameClear[0].value=strFieldValueClear;
}

```



Wavelink Contact Information

If you have comments or questions regarding this product, please contact Wavelink Customer Service.

E-mail Wavelink Customer Support at: CustomerService@wavelink.com

For customers within North America and Canada, call the Wavelink Technical Support line at 801-316-9000 (option 2) or 888-699-9283.

For international customers, call the international Wavelink Technical Support line at +800 9283 5465.

For Europe, Middle East, and Africa, hours are 9 AM - 5 PM GMT.

For all other customers, hours are 7 AM - 7 PM MST.



Glossary

action

A single step taken by the Screen Reformatter.

Avalanche Console

Wavelink's management application that allows you to centrally configure and manage mobile devices throughout your network.

Avalanche Enabler

A software component installed on mobile devices which allows you to configure and manage the device using the Avalanche Console. The Enabler facilitates communication between the mobile device and an Avalanche server.

Avalanche Server

A software component that facilitates communication between the Avalanche Console and Avalanche Enablers.

Avalanche software package

A specially bundled piece of software (e.g., a firmware update or an application) that you can download to a device using Avalanche.

Avalanche Update

A download (or configuration) that is available to a device through Avalanche. Examples of updates include software packages and network profiles. The deletion of orphaned packages from a device is another type of update.

character mode

A text-to-speech mode where the Vocalizer pronounces each character rather than trying to pronounce words.

ConnectPro

The ConnectPro server is an optional component of Terminal Emulation that handles session persistence. ConnectPro acts as a proxy between the mobile device and the emulation host. If the device loses connectivity or goes to sleep, the

ConnectPro server maintains the session until the device reconnects. ConnectPro is free with Terminal Emulation but is installed separately.

Descriptive View

The top right section of the Screen Reformatter. This view displays information about the Initial Screen and Modified Screen.

emulation parameters

Configurations for the TE Client that allow you to set terminal emulation-related behavior on a mobile device. These parameters can be global or host-specific.

Enabler

See Avalanche Enabler.

grammar file

A file that specifies what terms the speech-to-text engine will recognize.

Grammar File Manager

An application that manages the grammar files that are distributed to the device, the terms used for voice profiles, and the prompts used during training and calibration.

host

A server or workstation that hosts a specific software or network service.

host profile

A configuration for the TE Client that allows you to save host information (such as IP address and Telnet port) on mobile devices.

HTTP/HTTPS

HyperText Transfer Protocol/Secure HyperText Transfer Protocol. Protocols used for WEB emulation. HTTP is an insecure protocol. HTTPS is based on TLS and is more secure than HTTP.



IDA command

A special value used to invoke a device action, program action, or emulator action within the TE Client Industrial Browser.

Industrial Browser

A browser interface that gives you the ability to access web-based applications from a mobile device. You can develop your own web pages using META tags and IDA commands to enable specific functionality in the Industrial Browser.

initial field value

A value supplied to a field whenever the modified screen is used.

Initial Screen View

The bottom section of the Screen Reformatter. This view displays the screen that was captured, before modifications.

License Server

The License Server is an optional component of Terminal Emulation that handles licensing for TE Clients. It distributes licenses wirelessly and tracks licenses that haven't been used recently if you need to redistribute your licenses. ConnectPro is free with Terminal Emulation but is installed separately.

listening indicator

An icon that is displayed on the TE Client when the speech-to-text engine is listening for input.

localization

A service of the Telnet CE Client that allows you to configure the Client to display in a specific language.

MAC address

Media Access Control address. The hard-coded address of a device, a 12-digit hexadecimal number. The first 6 hexadecimal characters identify the manufacturer. The last 6 hexadecimal numbers are unique for each network device

produced by the manufacturer. Also called the hardware address.

master file

A file that contains several modified screens along with original screen captures. Master files are identified by a .wlrmf file extension.

META tag

Tags that allow specific functionality in the Industrial Browser for web pages.

Modified Screen View

The the top left section of the Screen Reformatter. This view displays the modified version of the captured screen.

network profile

A set of pre-configured network parameters (SSID, IP address, etc.) that can be downloaded to a device using Avalanche.

orphaned package

A software package that has been deployed to an Enabler through Avalanche, but has since been disabled or is not recognized by the server.

real-time statistics

Details about the mobile device that are sent by the Enabler to the Avalanche Console, where they can be viewed. The real-time statistics feature is only available for Avalanche-deployed TE Clients.

Resource Editor

A TE feature that allows you to import and deploy sound or picture files to your mobile devices.

result

The return value of a function.

return

A statement that ends the processing of the current function and returns control to the calling function, with or without a return value.



screen capture file

In order to use the screen reformatter, you record the screens using the TE Client. The resulting screen capture file is imported into the screen reformatter and modified. Screen capture files are identified by a .wltsc file extension.

screen reformatter

The screen reformatter allows you to redesign how the emulation screen is displayed on the mobile device. Include only the text or options you want to be available to the user. You can also add other text or scripting and Speakeasy actions for each screen.

sentence mode

A text-to-speech mode where the Vocalizer pronounces each group of characters as a word. See also character mode.

Session Monitor

An Avalanche-integrated component of the TE Client that allows a user at the Avalanche Console to monitor or control the TE Client. This is available for Avalanche-deployed TE Clients only.

software package

A specially bundled piece of software (e.g., a firmware update to a radio card or an application) that you can download to a client using Avalanche.

speech rate

The speed at which the text-to-speech is read.

speech-to-text

The component of Speakeasy that takes sound input and produces text.

SSH

Secure Shell. A protocol encapsulating Telnet that uses a secure channel to send encrypted information across a network.

SSL/TLS

Secure Sockets Layer/Transport Layer Security. Protocols that encrypt information using handshakes and ciphers. TLS is based on SSL and is more advanced.

string

An ordered sequence of symbols chosen from a predetermined set.

TELNET

A TCP/IP protocol that allows a Client to connect and interact with a remote host system.

TermProxy

See ConnectPro.

text-to-speech

The component of Speakeasy that takes text input and reads it aloud. Also called the Vocalizer.

TLS

See SSL/TLS.

verification item

An item on the original Telnet screen that must be verified before the modified screen will be displayed.

Vocalizer

The text-to-speech component of Speakeasy.

voice profile

Speech samples that increase positive results during speech-to-text. Each user can create his own voice profile. If you are using Wavelink Avalanche, voice profiles can be easily archived and re-distributed on an as-needed basis.

Windows TE Client

A TE Client designed to work on a desktop PC running Windows 2000, XP, or Vista.

