Powered by Universal Speech Solutions LLC

UniMRCP

Installation Manual

Developer Guide

Revision: 38

Last updated: May 20, 2017

Created by: Arsen Chaloyan

Table of Contents

1 Overview

This guide describes how to obtain and build UniMRCP from source. The document is intended for developers familiar with software programming and integrated development environments.

1.1 Applicable Versions

Instructions provided in this guide are applicable to the following versions.

UniMRCP 1.0.0 and above

1.2 Supported Platforms

UniMRCP natively supports Windows and Linux operating systems and can be compiled on and for a 32 or 64-bit platform. Other UNIX variants may be supported with no or minimal changes.

Operating System	32-bit	64-bit
Windows	✓	✓
Linux	✓	✓

2 Obtaining the Source Code

The source code can be obtained either by downloading one of officially released packages or by checking out a working copy from repository.

2.1 Downloading the Latest Release

The released source packages have the following naming convention *unimrcp-major.minor.patch* and are available in two formats: *zip* for Windows and *tar.gz* for Linux. The packages can be downloaded from the following location:

```
http://www.unimrcp.org/downloads/core
```

After downloading a package, open the archive and copy its content into a local directory.

2.2 Retrieving from the Repository

The source repository is hosted on GitHub. The code can be retrieved using a git client.

```
git clone https://github.com/unispeech/unimrcp.git
```

3 Obtaining the Dependencies

UniMRCP depends on a number of third-party tools and libraries which are released prepackaged and optimized for internal use. The released source packages of dependencies have the following naming convention *unimrcp-deps-major.minor.patch* and are also available in two formats: *zip* for Windows and *tar.gz* for Linux. The packages can be downloaded from the following location:

> http://www.unimrcp.org/downloads/dependencies

After downloading a dependencies package, open the archive and copy its content into the UniMRCP source directory. Note that the source directory of the dependencies doesn't actually matter for GNU build, since only installed header and library files of dependencies are being used.

4 Building the Project

4.1 The Windows Build

Prerequisites

UniMRCP provides solution and project files for Visual Studio 2005 and 2010, which can also be converted to and built with Visual Studio 2008, 2012 and 2013.

- ✓ Visual Studio 2005 or 2010 (2008, 2012 or 2013)

Procedure

First, build the dependencies by following the instructions below.

1. Open a solution file.
   - ❏ For VS2005 or VS2008, use unimrcpdeps.sln.
   - ❏ For VS2010, VS2012 or VS2013, use unimrcpdeps-2010.sln.
2. Choose a platform (Build -> Configuration Manager).
   - ❏ win32
   - ❏ x64
3. Choose a configuration (Build -> Configuration Manager).
   - ❏ Debug
   - ❏ Release
4. Build the solution (Build -> Build Solution).

Next, consider the same procedure for UniMRCP.

1. Open a solution file.
   - ❏ For VS2005 or VS2008, use unimrcp.sln.
   - ❏ For VS2010, VS2012 or VS2013, use unimrcp-2010.sln.
2. Choose a platform (Build -> Configuration Manager).
   - ❏ win32
   - ❏ x64
3. Choose a configuration (Build -> Configuration Manager).
   - ❏ Debug
   - ❏ Release
4. Build the solution (Build -> Build Solution).

Finalize output directory set-up by building the utility project *prepare*.

Build the utility project (Solution Explorer -> tools -> prepare).

This is a one-time operation which copies all the required dlls of dependencies as well as default configuration and data files to the corresponding output directory. As a result, the output directory will have the following structure:

- bin binaries (unimrcpserver, unimrcpclient, ...) and all the required dlls
- conf configuration files (unimrcpserver.xml, unimrcpclient.xml, ...)
- data data files
- lib libraries
- log log files
- plugin run-time loadable modules

4.2 The GNU Build

Prerequisites

UniMRCP requires the GNU toolchain to be installed.

- ✓ autoconf 2.59 or newer
- ✓ automake
- ✓ libtool 1.4 or newer
- ✓ gcc
- ✓ pkg-config

Procedure

First, build and install the dependencies by executing the following script.

```
./build-dep-libs.sh
```

Next, proceed with the UniMRCP build.
If the source has been checked out from repository, the *bootstrap* script must be executed first in order to generate the *configure* script and other required files.

```
./bootstrap
```

The usual sequence of commands *configure*, *make* and *make install* should follow in order to build and install the project from source.

```
./configure
make
make install
```

As a result, the project will be installed in the directory */usr/local/unimrcp* with the following structure:

- bin binaries (unimrcpserver, unimrcpclient, ...)
- conf configuration files (unimrcpserver.xml, unimrcpclient.xml, ...)
- data data files
- lib shared (convenience) libraries
- log log files
- plugin run-time loadable modules

5 Testing the Setup

Test your setup by using the sample UniMRCP client and server applications on the same host. The default configuration and data files should be sufficient for a basic test.

5.1 Launching the Applications

Windows

Launch the UniMRCP server application.

```
cd $(Configuration)\bin
unimrcpserver
```

Launch the sample UniMRCP client application.

```
cd $(Configuration)\bin
umc
```

Linux

Launch the UniMRCP server application.

```
cd /usr/local/unimrcp/bin
./unimrcpserver
```

Launch the sample UniMRCP client application.

```
cd /usr/local/unimrcp/bin
./umc
```

5.2 Running the MRCP Scenarios

Run typical speech synthesis, recognition, recorder, and verification scenarios by issuing corresponding commands from the console of the UniMRCP client application.

For speech synthesis, use:

```
run synth
```

For speech recognition, use:

```
run recog
```

For DTMF recognition, use:

```
run dtmf
```

For speech recorder, use:

```
run rec
```

For speech verification, use:

```
run verify
```

Visually inspect console output to check for any possible warnings or errors. Note that synthesized speech and recorded utterances are stored in the data directory. Also, be aware that the demo plugins loaded by default into the UniMRCP server are pure simulators. These plugins have been implemented for demonstration purposes only.