Julius Plugin

Usage Guide

Revision: 3
Created: February 16, 2017
Last updated: May 20, 2017
Author: Arsen Chaloyan
# Table of Contents

1 Overview .......................................................................................................................... 3
   1.1 Installation .................................................................................................................. 3
   1.2 Applicable Versions ................................................................................................. 3

2 Configuration Format ....................................................................................................... 4
   2.1 Document .................................................................................................................. 4
   2.2 Server Manager ........................................................................................................ 5
   2.3 Server Pool ................................................................................................................ 6
   2.4 Speech and DTMF Input Detector ............................................................................ 7
   2.5 Utterance Manager ................................................................................................... 9

3 Configuration Steps ......................................................................................................... 11
   3.1 Using Default Configuration .................................................................................... 11
   3.2 Specifying Models and Pools ................................................................................. 11
   3.3 Specifying Built-in Grammars .................................................................................. 12
   3.4 Specifying Speech/DTMF Input Detector .............................................................. 12
   3.5 Specifying Utterance Manager ................................................................................ 13

4 Supported Features ......................................................................................................... 15
   4.1 Supported MRCP Methods ....................................................................................... 15
   4.2 Supported MRCP Events ......................................................................................... 15
   4.3 Supported MRCP Header Fields .............................................................................. 15
   4.4 Supported Grammars ............................................................................................... 15
   4.5 Supported Results ...................................................................................................... 16

5 Usage Examples ................................................................................................................ 17
   5.1 Built-in Speech Grammar ......................................................................................... 17
   5.2 Built-in DTMF Grammar ........................................................................................ 18
   5.3 Speech and DTMF Grammars .................................................................................. 19

6 References ....................................................................................................................... 21
   6.1 Julius Tutorials .......................................................................................................... 21
   6.2 Specifications ............................................................................................................ 21
1 Overview

This guide describes how to configure and use the Julius plugin to the UniMRCP server. The document is intended for users having a certain knowledge of Julius and UniMRCP.

1.1 Installation

For installation instructions, use one of the guides below.

- RPM Package Installation (Red Hat / Cent OS)
- Deb Package Installation (Debian / Ubuntu)

1.2 Applicable Versions

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

<table>
<thead>
<tr>
<th>UniMRCP 1.4.0 and above</th>
</tr>
</thead>
<tbody>
<tr>
<td>UniMRCP Julius Plugin 1.0.0 and above</td>
</tr>
</tbody>
</table>
2 Configuration Format

The configuration file of the Julius plugin is located in /opt/unimrcp/conf/umsjulius.xml and the relevant data files are placed in the directory /opt/unimrcp/data/julius. The configuration file is written in XML.

2.1 Document

The root element of the XML document must be <umsjulius>.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>license-file</td>
<td>File path</td>
<td>Specifies the license file. File name may include patterns containing '*' sign. If multiple files match the pattern, the most recent one gets used.</td>
</tr>
</tbody>
</table>

Parent

None.

Children

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;jserver-manager&gt;</td>
<td>String</td>
<td>Specifies parameters of the Julius server manager.</td>
</tr>
<tr>
<td>&lt;speech-dtmf-input-detector&gt;</td>
<td>String</td>
<td>Specifies parameters of the speech and DTMF input detector.</td>
</tr>
<tr>
<td>&lt;utterance-manager&gt;</td>
<td>String</td>
<td>Specifies parameters of the utterance manager.</td>
</tr>
</tbody>
</table>

Example

This is an example of a bare document.

```
< umsjulius license-file="umsjulius_*.lic">
  ...
</ umsjulius>
```
2.2 Server Manager

This element specifies Julius server manager, which may contain several pools of Julius recognition servers.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>default-language</td>
<td>String</td>
<td>Specifies the default language to use, if not set by the client.</td>
</tr>
<tr>
<td>app</td>
<td>File path</td>
<td>Specifies the Julius server application to be launched. If no path specified, the directory /opt/unimrcp/data/julius is implied.</td>
</tr>
<tr>
<td>jconf</td>
<td>File path</td>
<td>Specifies the Julius server configuration file. If no path specified, the directory /opt/unimrcp/data/julius is implied.</td>
</tr>
<tr>
<td>maintenance</td>
<td>Boolean</td>
<td>Specifies whether the Julius server application needs to be implicitly launched and maintained by the plugin.</td>
</tr>
<tr>
<td>adin-port-base</td>
<td>Integer</td>
<td>Specifies a base port number used for audio streaming to Julius server.</td>
</tr>
<tr>
<td>ctrl-port-base</td>
<td>Integer</td>
<td>Specifies a base port number used for control messages sent to Julius server.</td>
</tr>
<tr>
<td>adin-mode</td>
<td>Integer</td>
<td>Specifies the audio input mode.</td>
</tr>
<tr>
<td>debug</td>
<td>Boolean</td>
<td>Sets the operation mode to debug, if enabled.</td>
</tr>
</tbody>
</table>

Parent

<ums:julius>

Children

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;jserver-pool&gt;</td>
<td>String</td>
<td>Specifies a pool of homogenous Julius server instances. One or more jserver-pool elements might be specified.</td>
</tr>
</tbody>
</table>

Example

This is an example of a bare server manager.
2.3 Server Pool

This element specifies a pool of homogenous Julius server instances.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>instance-count</td>
<td>Integer</td>
<td>Specifies the number of homogeneous Julius server instances included in the pool.</td>
</tr>
<tr>
<td>language</td>
<td>String</td>
<td>Specifies a language the model is made for.</td>
</tr>
<tr>
<td>sampling-rate</td>
<td>Integer</td>
<td>Specifies a sampling rate the model is made for.</td>
</tr>
<tr>
<td>acoustic-model-dir</td>
<td>Dir path</td>
<td>Specifies a directory containing the acoustic model data.</td>
</tr>
<tr>
<td>hmmdefs</td>
<td>File path</td>
<td>Specifies a file name for the HMM definitions.</td>
</tr>
<tr>
<td>hmmlist</td>
<td>File path</td>
<td>Specifies a file name for the HMM list.</td>
</tr>
<tr>
<td>grammar-dir</td>
<td>Dir path</td>
<td>Specifies a directory containing the pre-compiled speech grammar files.</td>
</tr>
<tr>
<td>grammar-names</td>
<td>String</td>
<td>This parameter specifies a comma-separated list of grammar names declared in the grammar directory.</td>
</tr>
<tr>
<td>grammar-prefix-file</td>
<td>File path</td>
<td>Specifies a name of the grammar prefix file.</td>
</tr>
</tbody>
</table>

Parent

< jserver-manager>
Children
None.

Example
The example below defines two en-US language models: one is for audio sampled at 8 kHz, the other – for 16 kHz.

```xml
<jserver-pool
  instance-count="2"
  language="en-US"
  sampling-rate="8000"
  acoustic-model-dir="acoustic-model-8kHz"
  hmmdefs="hmmdefs"
  hmmlist="tiedlist"
  grammar-dir="speech-grammar"
  grammar-names="command, digits, fruit"
  grammar-prefix-file=""
/><jserver-pool
  instance-count="2"
  language="en-US"
  sampling-rate="16000"
  acoustic-model-dir="acoustic-model-16kHz"
  hmmdefs="hmmdefs"
  hmmlist="tiedlist"
  grammar-dir="speech-grammar"
  grammar-names="command, digits, fruit"
  grammar-prefix-file=""
/>```

2.4 Speech and DTMF Input Detector

This element specifies parameters of the speech and DTMF input detector.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>speech-start-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait in transition mode before triggering a start of speech input event.</td>
</tr>
<tr>
<td>speech-complete-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait in transition mode before triggering an end of speech input event.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>noinput-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait before triggering a no-input event.</td>
</tr>
<tr>
<td>input-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait for input to complete.</td>
</tr>
<tr>
<td>dtmf-interdigit-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies a DTMF inter-digit timeout.</td>
</tr>
<tr>
<td>dtmf-term-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies a DTMF input termination timeout.</td>
</tr>
<tr>
<td>dtmf-term-char</td>
<td>Character</td>
<td>Specifies a DTMF input termination character.</td>
</tr>
<tr>
<td>normalize-input</td>
<td>Boolean</td>
<td>Specifies whether received spoken input stream should be normalized or not.</td>
</tr>
<tr>
<td>speech-leading-silence</td>
<td>Time interval [msec]</td>
<td>Specifies desired silence interval preceding spoken input. The parameter is used if normalize-input is set to true.</td>
</tr>
<tr>
<td>speech-trailing-silence</td>
<td>Time interval [msec]</td>
<td>Specifies desired silence interval following spoken input. The parameter is used if normalize-input is set to true.</td>
</tr>
<tr>
<td>speech-output-period</td>
<td>Time interval [msec]</td>
<td>Specifies an interval used to send speech frames to the Julius recognizer.</td>
</tr>
</tbody>
</table>

**Parent**

`<umsjulius>`

**Children**

None.

**Example**

The example below defines a typical speech and DTMF input detector having the default parameters set.

```xml
<speech-dtmf-input-detector
    speech-start-timeout="300"
    speech-complete-timeout="1000"
    noinput-timeout="5000"
    input-timeout="10000"
    dtmf-interdigit-timeout="5000"
    dtmf-term-timeout="10000"
    dtmf-term-char=""
    normalize-input="true"
>```
2.5 Utterance Manager

This element specifies parameters of the utterance manager.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>save-waveforms</td>
<td>Boolean</td>
<td>Specifies whether to save waveforms or not.</td>
</tr>
<tr>
<td>waveform-base-uri</td>
<td>String</td>
<td>Specifies the base URI used to compose an absolute waveform URI.</td>
</tr>
<tr>
<td>waveform-folder</td>
<td>Dir path</td>
<td>Specifies a folder the waveforms should be stored in.</td>
</tr>
<tr>
<td>expiration-time</td>
<td>Time interval [min]</td>
<td>Specifies a time interval after expiration of which waveforms are considered outdated.</td>
</tr>
<tr>
<td>purge-waveforms</td>
<td>Boolean</td>
<td>Specifies whether to delete outdated waveforms or not.</td>
</tr>
<tr>
<td>purge-interval</td>
<td>Time interval [min]</td>
<td>Specifies a time interval used to periodically check for outdated waveforms.</td>
</tr>
</tbody>
</table>

Parent

<umsjulius>

Children

None.

Example

The example below defines a typical utterance manager having the default parameters set.

```xml
<utterance-manager
    save-waveforms="false"
    waveform-base-uri="http://localhost/utterances/"
    waveform-folder=""
/>"
expiration-time="60"
purge-waveforms="true"
purge-interval="30"
/>
3 Configuration Steps

This section outlines common configuration steps.

3.1 Using Default Configuration

The default configuration and data files correspond to the en-US language and should be sufficient for the general use.

3.2 Specifying Models and Pools

While the default configuration and data files contain references to an en-US acoustic model and grammar files, which are getting installed with the package `unimrcp-julius-model-en-us`, other acoustic models and grammar files can also be used.

In order to add a new or modify the existing model, the following parameters must be specified:

- number of instances in the pool
- language the model is made for
- sampling rate the acoustic data corresponds to
- path to a directory containing acoustic model
- hmmlist and hmmdefs files

Note that, unless an absolute path is specified, the path is relative to the directory `/opt/unimrcp/data/julius/$language`.

The following example defines two server pools: one for en-US and the other for de-DE language. There are two instances of Julius server run in each pool.

```
<jserv-pool
    instance-count="2"
    language="en-US"
    sampling-rate="8000"
    acoustic-model-dir="acoustic-model-8kHz"
    hmmdirs="hmmdirs"
    hmmlist="tiedlist"
    grammar-dir="speech-grammar"
    grammar-names="command, digits, fruit"
    grammar-prefix-file=""
/>  
<jserv-pool
    instance-count="2"
```
3.3 Specifying Built-in Grammars

Built-in grammars are stored in the *fsg* format and can be referenced by the client by means of a built-in URI, such as:

```
builtin:speech/$name
```

where $name is the name of one of the grammars stored in the specified speech grammar directory for a particular server pool.

For instance, the package *unimrcp-julius-model-en-us* installs sample grammars called *digits*, *fruit*, *sample*, with the corresponding files located in the directory /opt/unimrcp/data/julius/en-US/speech-grammar. These sample grammars can be referenced by the client using one of the following built-in URIs:

```
builtin:speech/digits
builtin:speech/fruit
builtin:speech/sample
```

3.4 Specifying Speech/DTMF Input Detector

The default parameters specified for the speech and DTMF input detector are sufficient for the general use. However, various timeouts can be adjusted to better suite a particular requirement.

- **speech-start-timeout**
  
  This parameter is used to trigger a start of speech input. The shorter is the timeout, the sooner a START-OF-INPUT event is delivered to the client. However, a short timeout may also lead to a false positive.

- **speech-complete-timeout**
  
  This parameter is used to trigger an end of speech input. The shorter is the timeout, the shorter is the
response time. However, a short timeout may also lead to a false positive.

- **noinput-timeout**
  This parameter is used to trigger a NO-INPUT event. The parameter can be overridden per MRCP session by setting the header field NO-INPUT in SET-PARAMS and RECOGNIZE requests.

- **input-timeout**
  This parameter is used to limit input time. The parameter can be overridden per MRCP session by setting the header field RECOGNITION-TIMEOUT in SET-PARAMS and RECOGNIZE requests.

- **dtmf-interdigit-timeout**
  This parameter is used to set inter-digit timeout on DTMF input. The parameter can be overridden per MRCP session by setting the header field INTER-DIGIT-TIMEOUT in SET-PARAMS and RECOGNIZE requests.

- **dtmf-term-timeout**
  This parameter is used to set termination timeout on DTMF input and is in effect when dtmf-term-char is set and there is a match for an input grammar. The parameter can be overridden per MRCP session by setting the header field INTER-DIGIT-TIMEOUT in SET-PARAMS and RECOGNIZE requests.

- **dtmf-term-char**
  This parameter is used to set a character terminating DTMF input. The parameter can be overridden per MRCP session by setting the header field INTER-DIGIT-TIMEOUT in SET-PARAMS and RECOGNIZE requests.

### 3.5 Specifying Utterance Manager

The default parameters specified for the speech and DTMF input detector are sufficient for the general use. However, various timeouts can be adjusted to better suite a particular requirement.

- **save-waveforms**
  Utterances can optionally be recorded and stored if the configuration parameter save-waveforms is set to true. The parameter can be overridden per MRCP session by setting the header field SAVE-WAVEFORMS in SET-PARAMS and RECOGNIZE requests.

- **waveform-base-uri**
  This parameter specifies the base URI used to compose an absolute waveform URI returned in the header field WAVEFORM-URI in response to RECOGNIZE requests.

- **waveform-folder**
  This parameter specifies a path to the directory used to store waveforms in.

- **expiration-time**
  This parameter specifies a time interval in minutes after expiration of which waveforms are
considered outdated.

- **purge-waveforms**

  This parameter specifies whether to delete outdated waveforms or not.

- **purge-interval**

  This parameter specifies a time interval in minutes used to check for outdated waveforms if purge-waveforms is set to true.
4 Supported Features

4.1 Supported MRCP Methods
✓ RECOGNIZE
✓ START-INPUT-TIMERS
✓ SET-PARAMS
✓ GET-PARAMS

4.2 Supported MRCP Events
✓ RECOGNITION-COMPLETE
✓ START-OF-INPUT

4.3 Supported MRCP Header Fields
✓ Input-Type
✓ No-Input-Timeout
✓ Recognition-Timeout
✓ Waveform-URI
✓ Media-Type
✓ Completion-Cause
✓ Confidence-Threshold
✓ Start-Input-Timers
✓ DTMF-Interdigit-Timeout
✓ DTMF-Term-Timeout
✓ DTMF-Term-Char
✓ Save-Waveform
✓ Speech-Language
✓ Cancel-If-Queue

4.4 Supported Grammars
✓ Built-in/extendable FSG speech grammars
✓ Built-in/embedded DTMF grammar(s)
4.5 Supported Results

✓ NLSML
5 Usage Examples

5.1 Built-in Speech Grammar

This example demonstrates how to reference a built-in speech grammar in a RECOGNIZE request. The built-in speech grammar command is defined in the directory /opt/unimrcp/data/julius/en-US/speech-grammar.

C->S:

```
C->S: MRCP/2.0 333 RECOGNIZE 1
      Channel-Identifier: 0ab63ec7084a5444@speechrecog
      Content-Id: request1@form-level
      Content-Type: text/uri-list
      Cancel-If-Queue: false
      No-Input-Timeout: 5000
      Recognition-Timeout: 10000
      Start-Input-Timers: true
      Confidence-Threshold: 0.87
      Save-Waveform: true
      Content-Length: 22

      builtin:speech/command
```

S->C:

```
S->C: MRCP/2.0 83 1 200 IN-PROGRESS
      Channel-Identifier: 0ab63ec7084a5444@speechrecog
```

S->C:

```
S->C: MRCP/2.0 115 START-OF-INPUT 1 IN-PROGRESS
      Channel-Identifier: 0ab63ec7084a5444@speechrecog
      Input-Type: speech
```

S->C:

```
S->C: MRCP/2.0 540 RECOGNITION-COMPLETE 1 COMPLETE
      Channel-Identifier: 0ab63ec7084a5444@speechrecog
      Completion-Cause: 000 success
```
5.2 Built-in DTMF Grammar

This examples demonstrates how to reference a built-in DTMF grammar in a RECOGNIZE request.

C->S:

```
MRCP/2.0 266 RECOGNIZE 1
Channel-Identifier: d26bef74091a174c@speechrecog
Content-Type: text/uri-list
Cancel-If-Queue: false
Start-Input-Timers: true
Confidence-Threshold: 0.7
Speech-Language: en-US
Dtmf-Term-Char: #
Content-Length: 19

builtin:dtmf/digits
```

S->C:

```
MRCP/2.0 83 1 200 IN-PROGRESS
Channel-Identifier: d26bef74091a174c@speechrecog
```

S->C:

```
MRCP/2.0 113 START-OF-INPUT 1 IN-PROGRESS
Channel-Identifier: d26bef74091a174c@speechrecog
```
5.3 Speech and DTMF Grammars

This example demonstrates how to reference a built-in DTMF grammar and a speech grammar combined in a RECOGNIZE request. In this example, the user is expected to input a 4-digit pin.
S->C:

MRCP/2.0 115 START-OF-INPUT 2 IN-PROGRESS
Channel-Identifier: 6ae0f23e1b1e3d42@speechrecog
Input-Type: speech

S->C:

MRCP/2.0 399 RECOGNITION-COMPLETE 2 COMPLETE
Channel-Identifier: 6ae0f23e1b1e3d42@speechrecog
Completion-Cause: 000 success
Content-Type: application/x-nlsml
Content-Length: 214

<?xml version="1.0"?>
<result>
  <interpretation grammar="builtin:speech/pin" confidence="1.00">
    <instance>one two three four</instance>
    <input mode="speech">one two three four</input>
  </interpretation>
</result>
6 References

6.1 Julius Tutorials

- About Julius
- Language and Acoustic Models
- Grammar Files

6.2 Specifications

- Speech Recognizer Resource
- NLSML Results