Contents

- 1 Description
- 2 Usage
- 3 Examples
- 4 say_string
- 5 Talking Clock
- 6 See Also

Description

The **say** application will use the pre-recorded sound files to read or say various things like dates, times, digits, etc. The say application can read digits and numbers as well as dollar amounts, date/time values and IP addresses. It can also spell out alpha-numeric text, including punctuation marks. There's a transcript of the pre-recorded files in the sources under docs/phrase/phrase_en.xml

Usage

```
say <module_name>[:<lang>] <say_type> <say_method> [gender] <text>
```

- Module name is usually the channel language, e.g. "en" or "es"

- Say type is one of the following

```
NUMBER
ITEMS
PERSONS
MESSAGES
CURRENCY
TIME_MEASUREMENT
CURRENT_DATE
CURRENT_TIME
CURRENT_DATE_TIME
TELEPHONE_NUMBER
TELEPHONE_EXTENSION
URL
IP_ADDRESS
EMAIL_ADDRESS
POSTAL_ADDRESS
ACCOUNT_NUMBER
NAME_SPELLED
NAME_PHONETIC
SHORT_DATE_TIME
```

- Say method is one of the following (for example, passing a value of "42"):

- pronounced - e.g. "forty two"
- iterated - e.g. "four two"
- counted - e.g. "forty second"

- Say gender is one of the following (For languages with gender-specific grammar, like French and German)

```
FEMININE
MASCULINE
NEUTER
```

Examples

```
<!-- use in a phrase macro, different from using in the dialplan-->
<action function="say" data="733#" method="iterated" type="name_spelled"/>

<!-- spelled out alpha-numeric + ascii code, like "a-b-seven-three-three-pound", ASCII code 32-
<action application="say" data="en name_spelled iterated ab733#"/>

<!-- spell out called number, like "one-two-three-four-five" -->
<action application="say" data="en name_spelled iterated ${destination_number}"/>

<!-- read number, like "twelve thousand, three hundred forty-five" -->
<action application="say" data="en number pronounced 12345"/>

<!-- read IP address, like "one two dot three four dot five six dot seven eight" -->
<action application="say" data="en ip_address iterated 12.34.56.78"/>

<!-- read IP address, like "twelve dot thirty-four dot fifty-six dot seventy-eight" -->
<action application="say" data="en ip_address pronounced 12.34.56.78"/>

<!-- pronounce a dollar value, even works for negatives, like "negative one dollar and ninety-s
<action application="say" data="en CURRENCY PRONOUNCED -1.96"/>

<!-- pronounce a short date/time (like "Wednesday, 2:23PM") -- substitute [timestamp]
     with a UTC timestamp -->
<action application="say" data="en short_date_time pronounced [timestamp]"/>
```

Note: "say <language> number" works up to 999999999 (9 digits) in current languages.

say_string

say_string is the new method of doing say and use it in mod_say_en and mod_say_ru . We can do more to centralize the say things and go back and apply it to other languages. Using this method you can set the desired file extension as well.

```
say_string <module_name>[.<ext>] <lang>[.<ext>] <say_type> <say_method> [<say_gender>] <text>
```

From CLI:

```
eval ${say_string en.gsm en current_date_time pronounced ${strepoch()}}
```

Usage Example:

```
<action application="playback" data="${say_string en.wav en current_date_time pronounced ${strepo
```

Talking Clock

```
<extension name="Talking Clock Time" ><!--e.g. 10:56pm-->
   <condition field="destination_number" expression="9190">
       <action application="say" data="en CURRENT_TIME pronounced ${strepoch()}"/>
       <action application="hangup"/>
   </condition>
 </extension>

 <extension name="Talking Clock Date" ><!--e.g. March 8, 2011-->
   <condition field="destination_number" expression="9191">
       <action application="say" data="en CURRENT_DATE pronounced ${strepoch()}"/>
       <action application="hangup"/>
   </condition>
 </extension>

 <extension name="Talking Clock Date and Time" ><!--e.g. March 8, 2011 10:56pm-->
   <condition field="destination_number" expression="9192">
       <action application="say" data="en CURRENT_DATE_TIME pronounced ${strepoch()}"/>
       <action application="hangup"/>
   </condition>
 </extension>
```

See Also

- Dialplan
- Dialplan Application