

SpeechRecorder 6 Quick Start and User Manual

Christoph Draxler

draxler@phonetik.uni-muenchen.de

Klaus Jänsch

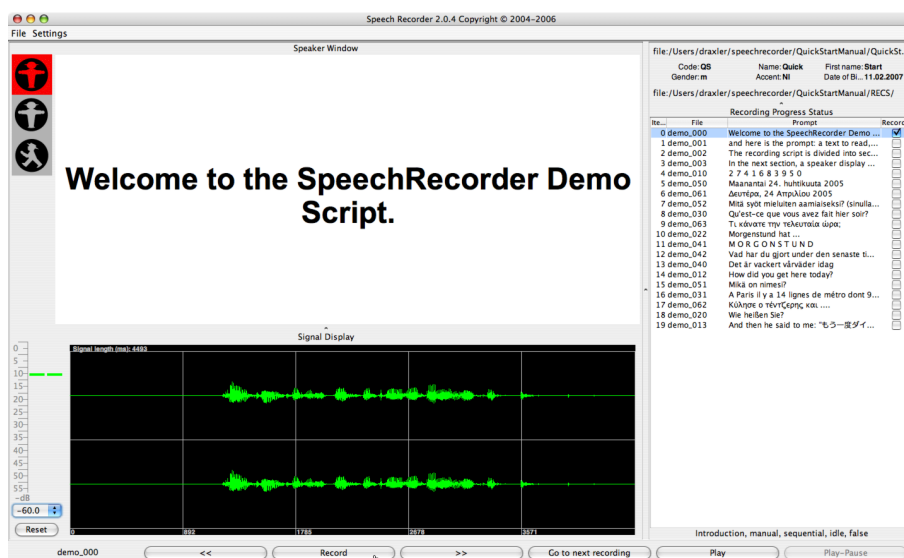
klausj@phonetik.uni-muenchen.de

Institut für Phonetik und Sprachverarbeitung

Universität München

June 8, 2021

June 8, 2021



SpeechRecorder is an application for script-driven speech, audio, and signal recordings. Its main features are

- platform independence (macOS, Windows and Ubuntu Linux)
- automatic and manual recording progress
- speaker and supervisor views on multiple screens
- full Unicode text, image and audio prompts

Quick Start

SpeechRecorder organizes recordings in projects. A *project* is a combination of a speaker database, a set of recording scripts, and a set of recording sessions. A recording session consists of an individual speaker, a recording script, the selected recording settings, and a directory into which the recorded files are written.

1. Download and install *SpeechRecorder* package from <http://www.speechrecorder.org>.
2. Select the command **Project > New** from the menu, give the project a name. The following items will now be created:
 - a workspace directory 'speechrecorder' in your home directory
 - a project directory in the workspace directory
 - a sample (or empty) recording script
 - an empty speaker database
 - a project configuration file

On the left side of the display, a small traffic light will show up. In the middle, the prompt area is displayed, and on the right side, the contents of the recording script are listed (see fig. 1 a)).

3. In the **Speakers** menu, select the option **Speaker...** and enter data for a speaker. Select the speaker in the table and close the dialog with the button **select**.
4. Click the button **Record** to start your recording session. Stop the current recording by clicking **Stop** or waiting until the recording timeout has been reached. After the recording has ended, the signal is displayed on the screen. Click on **Play** to listen to the recording.
5. Proceed to the next item by clicking **>>**. Start the next recording with the **Record** button.
6. After the final item has been recorded, *SpeechRecorder* displays a message. Click **Ok** to acknowledge the message.
7. You will find your recordings in the subdirectory **RECS** of the project directory. The menu item **Info** in the **Help** menu will help you to find the project file paths.

You're done – you've recorded your first session using *SpeechRecorder*!

Demo Script

The first section contains test items, recording progress is manual (i.e. you have to click to start and end a recording, and click again to proceed to the next item), and only the supervisor view is shown. The second section contains sample prompts in different languages and of different types. Recording progress is semi-automatic (i.e. after a recording is stopped, the script proceeds automatically to the next item), and the speaker view is displayed.

Multiple Displays

If you have two displays attached to your machine, the supervisor view will always be shown on the primary display, the speaker view on the secondary display(s) (see fig. 1 a) and b)).

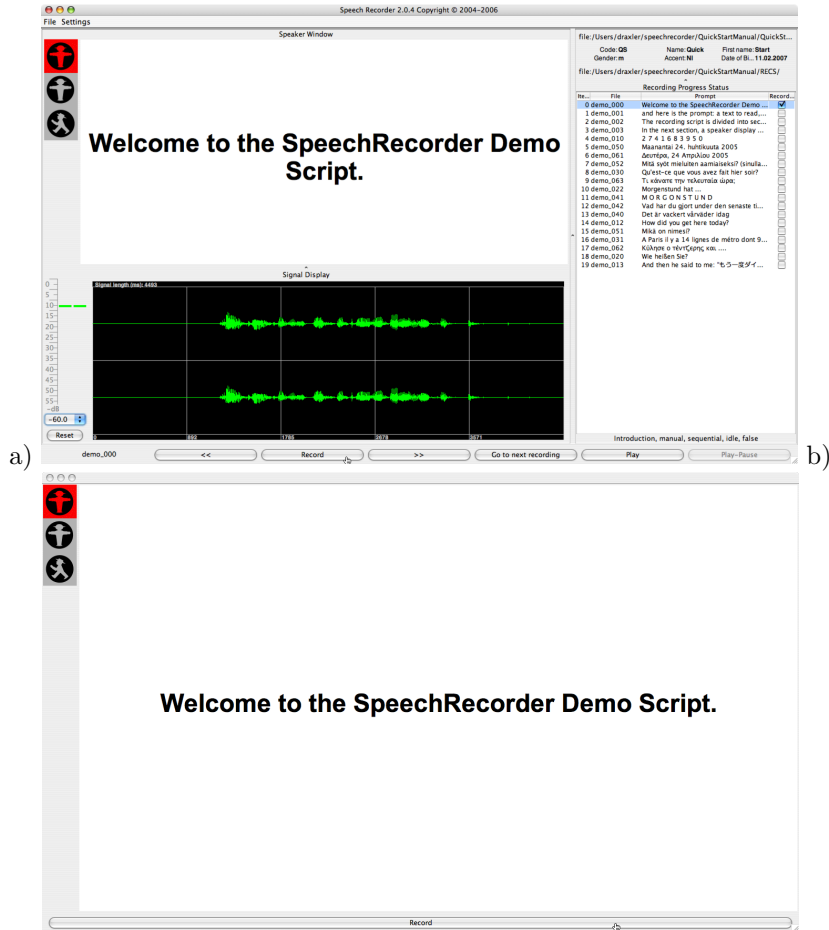


Figure 1: SpeechRecorder supervisor (a) and speaker (b) views

Citing SpeechRecorder

SpeechRecorder was originally presented at the International Conference on Language Resources and Evaluation in Lisbon in 2004 [DJ04]. Please use this reference to *SpeechRecorder* in your publications.

Contents

| | | |
|-----------|--|-----------|
| 1 | Recording Script | 5 |
| 1.1 | Edit recording script with internal script editor | 5 |
| 1.2 | Edit recording script with internal or external XML editor | 5 |
| 1.3 | The <recordingscript> element | 5 |
| 1.4 | The <section> element | 7 |
| 1.5 | Optional <group> element | 8 |
| 1.6 | The <recording> element | 10 |
| 1.7 | The <recprompt> element | 11 |
| 1.8 | The <mediaitem> element | 11 |
| 2 | Recording Phases | 16 |
| 3 | Menu File | 17 |
| 4 | Menu View | 17 |
| 5 | Menu Workspace | 17 |
| 6 | Menu Project | 17 |
| 6.1 | Menu item Export | 18 |
| 6.2 | Menu item Preferences | 18 |
| 7 | Menu Speaker | 21 |
| 8 | Menu Script | 21 |
| 8.1 | Edit script... | 21 |
| 8.2 | Edit script XML source | 21 |
| 8.3 | Script resources | 22 |
| 8.4 | Import text table | 22 |
| 8.5 | Export script... | 22 |
| 8.6 | Export script as text table file... | 22 |
| 9 | Menu Settings | 22 |
| 10 | Recordings via the Internet | 22 |
| A | Recording script DTD | 23 |
| B | Reserved keywords for recording scripts | 25 |
| C | Known issues | 25 |
| C.1 | Platform dependencies | 26 |
| D | Signal quality problems | 26 |
| E | Contacts and Copyright | 29 |
| F | Useful links | 29 |

1 Recording Script

A script specifies which items are to be recorded. A *script* consists of two parts, a header containing meta-data items, and the recording script proper. The *recording script* is divided into sections. A *section* is an organizational unit that specifies the presentation order, and progress mode for the recording items it contains.

A *recording item* or *recording* consists of the instructions, the prompt item, and a comment. Instructions and comment are optional. A prompt item consists of text, an image, or an audio clip. The text may be stored in the recording script, or fetched from an external file or URL. Images and audio clips must be loaded from external sources, e.g. a file or a URL.

A recording script is stored as an XML document. The current Document Type Definition (DTD) is given in Appendix A. It will also be copied to each new project as file *SpeechRecPrompts4.dtd*.

1.1 Edit recording script with internal script editor

Open the recording script editor with the menu item **Edit script...** in the **Script** menu. The editor has its own menu bar. The **Script** menu contains items to add new sections or recording items. In the left column you can edit recording script name, optional metadata and the list of sections. The middle column represents the selected section and displays the list of recording items. The right column represents the selected recording item. It has two tabs: In the **Prompt** tab you can edit the prompt which will be displayed to the subject. The **Control** tab defines how recording will be controlled during the recording session. Selected sections or items can be edited using cut,copy and paste commands shown in the edit menu or by key strokes. If you copy and paste a section or prompt items an active item code generator will apply new item codes to this items to avoid duplicate item codes.

1.2 Edit recording script with internal or external XML editor

If you have some experience using XML files you can edit the recording script with the internal editor (**Edit script source...** menu item in **Script**) or an external XML or text editor.

1.3 The <recordingscript> element

The <recordingscript> element contains the whole structure of the script. The DTD definition is:

```
<!ELEMENT recordingscript (virtualviewbox?,section*)>
```

```
<!ELEMENT virtualviewbox EMPTY>
```

```
<!ATTLIST virtualviewbox height CDATA #IMPLIED>
```

An optional <virtualviewbox> element can be used to scale the font size of all text prompts to a virtual height. The element is defined as:

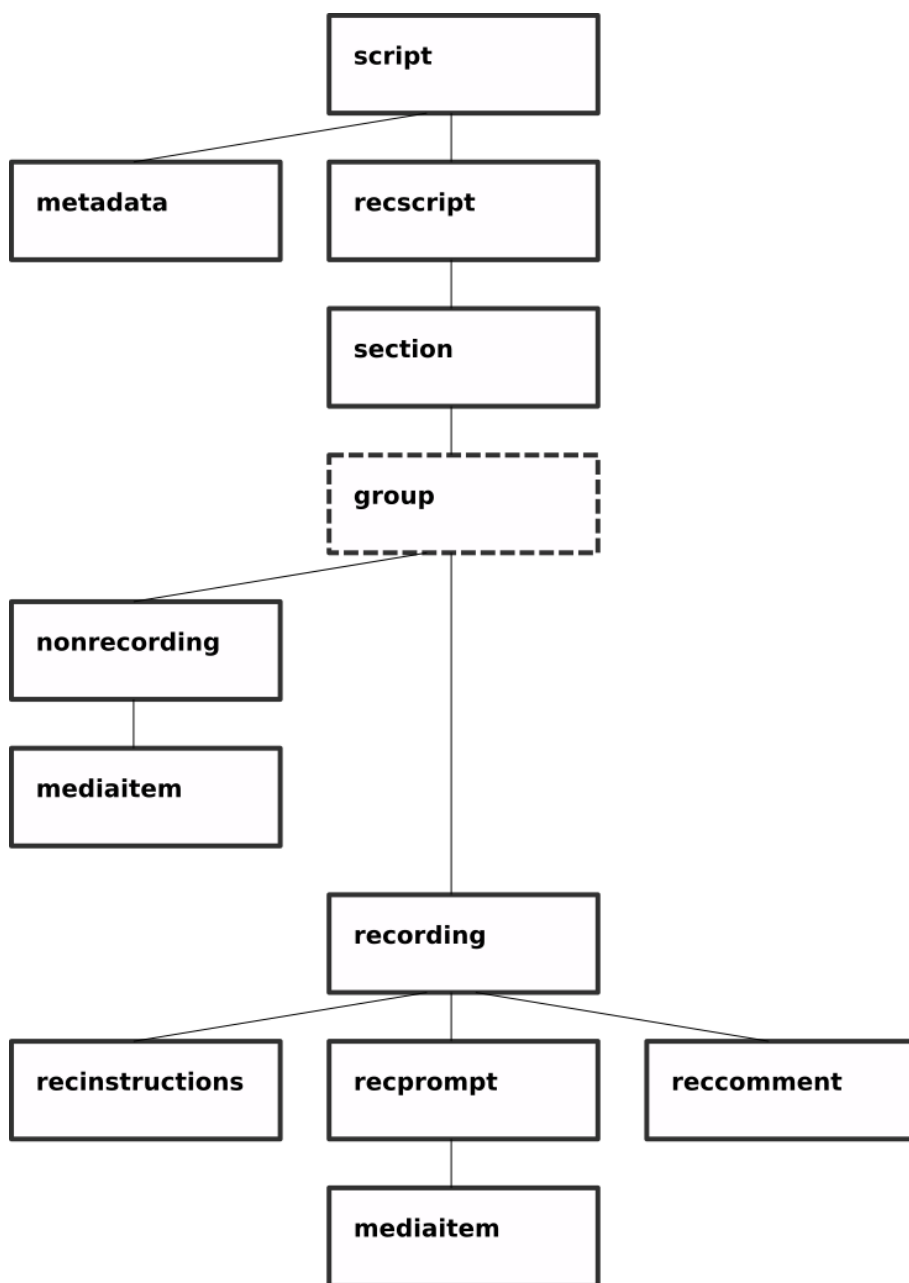


Figure 2: Structure of a *SpeechRecorder* recording script

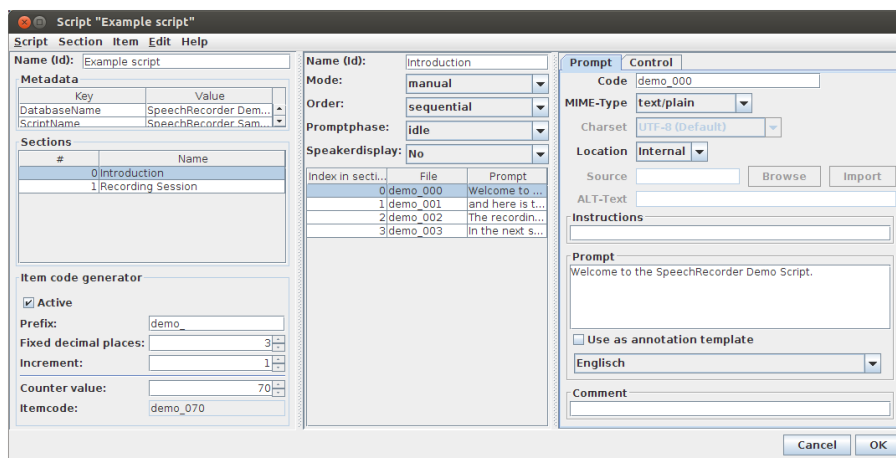


Figure 3: Recording script editor

```
<!ELEMENT virtualviewbox EMPTY>
<!ATTLIST virtualviewbox height CDATA #IMPLIED>
```

When this virtual height is set the font size of plain and formatted texts are not fixed anymore. The font size will then be scaled depending on size of the prompt display.

A good example value for the virtual height is 600 (pixels):

```
<virtualviewbox height="600"/>
```

1.4 The <section> element

A *section* groups together items that are presented and recorded in a similar manner.

In a recording script, the <section> tag is defined as follows:

```
<!ELEMENT section (nonrecording | recording)+ >

<!ATTLIST section name CDATA #IMPLIED
                  speakerdisplay CDATA #IMPLIED
                  order CDATA #IMPLIED
                  mode CDATA #IMPLIED
                  promptphase CDATA #IMPLIED >
```

All attributes are optional. *name* specifies the name of the section, e.g. *Introduction* or *Narrative*. *speakerdisplay* indicates whether the speaker view will be shown or not – allowed attribute values are *yes* and *no*.

order specifies the order in which the items or groups in this section will be presented. The allowed values are *sequential* or *random*.

mode controls the recording progress. The attribute value *manual* means that the user has to click once to advance to the next recording item, and again to start the recording. *autoprogess* means that the user clicks only once to advance

to and immediately start the next recording. *autorecording* finally means that the script proceeds to the next item and starts its recording without user action. However, the user may pause the script and resume recording later.

promptphase specifies when the prompt item is displayed. *idle* displays the item already before the actual recording, e.g. to give the user time for preparation. *recording* shows the prompt only during the recording phase (see section 2 for details)

Sample sections

```
<section name="Introduction" order="sequential"
      speakerdisplay="no" mode="manual"
      promptphase="idle">
...
</section>

<section name="Recording Session" order="random"
      speakerdisplay="yes" mode="autoprogess"
      promptphase="idle">
...
</section>
```

Section display

Information on the section is displayed below the table with the recording items in the supervisor view (fig. 4).

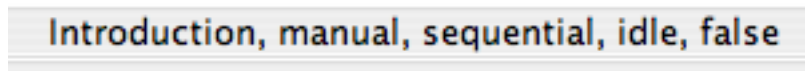


Figure 4: Section information display in the supervisor view

1.5 Optional <group> element

The **group** element allows you to group prompt-items inside a section. The main purpose of this element is randomization. For instance if you want to randomize sequences of two recordings. In this example the order of the recordings in the sequences (inside the **group** elements) will always be as given in the script, but the order of the sequences (the groups) will be randomized inside the section.

```
<section name="Recording Session" order="random">
<group>
<recording itemcode="demo_seq1_rec1">
    <recprompt>
        ...
    </recprompt>
</recording>
```



```

<recording itemcode="demo_seq1_rec2">
  <recprompt>
    ...
  </recprompt>
</recording>
</group>
<group>
<recording itemcode="demo_seq2_rec1">
  <recprompt>
    ...
  </recprompt>
</recording>
<recording itemcode="demo_seq2_rec2">
  <recprompt>
    ...
  </recprompt>
</recording>
</group>
</section>

```

Another example in which the order of the groups is fixed and the recording items inside the groups are randomized:

```

<section name="Recording Session">
<group order="random">
<recording itemcode="demo_trial1_rec1">
  <recprompt>
    ...
  </recprompt>
</recording>
<recording itemcode="demo_trial1_rec2">
  <recprompt>
    ...
  </recprompt>
</recording>
</group>
<group order="random">
<recording itemcode="demo_trial2_rec1">
  <recprompt>
    ...
  </recprompt>
</recording>
<recording itemcode="demo_trial2_rec2">
  <recprompt>
    ...
  </recprompt>
</recording>
</group>
</section>

```

1.6 The <recording> element

The <recording> element defines the id, contents, and timing of the current recording item. It consists of the optional <recinstructions> and <recomment> elements, and the mandatory <recprompt> element. <recinstructions> and <recomment> simply contain text – which is displayed to both the speaker and the supervisor, or the supervisor only, respectively.

```
<!ELEMENT recording (recinstructions?, recprompt, recomment?) >
```

```
<!ATTLIST recording itemcode CDATA #REQUIRED
recduration CDATA #IMPLIED
prerecdelay CDATA #IMPLIED
postrecdelay CDATA #IMPLIED
finalsilence CDATA #IMPLIED
beep CDATA #IMPLIED
rectype CDATA #IMPLIED
blocked CDATA #IMPLIED
>
```

<recinstructions> may have the attributes `mimetype` and `src` to allow instructions to be read in from an external source (see C for details).

The attribute `itemcode` of <recording> is mandatory. It uniquely identifies a recording item. `itemcode` can be an arbitrary string – however, because the `itemcode` becomes part of the audio file name, it may not contain characters that have a special meaning in the file system (see C for details). In the script editor GUI you can activate the media item generator, which generates unique item codes for you. The generator consists of a counter and a formatter. If a new recording is added the counter will be incremented by the value configured until the formatted item code value reaches a string value which does not exist in the script. The formatted value is concatenated by the given optional prefix (default "item") and the counter value as a string with fixed decimal places. The settings of the item code generator in the script editor are not stored to the project configuration. If you want to store this settings to the project configuration, open the corresponding form in the project configuration editor (Project -> Preferences ...-> Prompting -> Script).

`recduration` specifies the recording time in milliseconds. The default is to record infinite. `prerecdelay` and `postrecdelay` specify in milliseconds a time span during which recording is active, but the the prompt is still inactive (see 2 for details).

The `finalsilence` attribute defines the amount of milliseconds of speech silence until the recording will be automatically stopped. The GUI editor sets this to a fixed value of 4000 ms if silence detection is enabled for a recording item. The silence detector is currently implemented using fixed parameters.

The only `rectype` currently supported is *audio*. Default is *audio*.

The `blocked` attribute is of boolean type and determines if playback of a media prompt (i.e. a audio clip) blocks recording. If `blocked` is set, recording starts when playback of the prompt has finished. If not set playback of the prompt and recording start at the same time.

Recording sample

```
<recording prerecdelay="2000"
           recduration="20000"
           postrecdelay="500"
           itemcode="demo_001">
  <recprompt>
    ...
  </recprompt>
</recording>
```

1.7 The <recprompt> element

The <recprompt> element holds the prompt item. It holds one or two media items, which will be prompted to the user.

Example: Simple text prompt.

```
<recprompt>
  <mediaitem>Welcome to the SpeechRecorder Demo Script.</mediaitem>
</recprompt>
```

All text and image media types can be combined with audio prompts. Such a combined prompt will display the text or image and play an audio prompt simultaneously. In the script editor press **Item ; Add extra media item** to add a second media item. In the prompt item viewer a new tab 'Media 02' will appear. Apply the audio file to the new media item.

Examples for combined audio prompts in XML script source:

```
<recprompt>
  <mediaitem>Welcome to the SpeechRecorder Demo Script.</mediaitem>
  <mediaitem mimetype="audio/x-wave" src="resources/audio/welcome.wav"/>
</recprompt>

<recprompt>
  <mediaitem mimetype="image/svg+xml" src="resources/image/welcome.xml">
  <mediaitem mimetype="audio/x-wave" src="resources/audio/wav/welcome.wav"/>
</recprompt>
```

1.8 The <mediaitem> element

The <mediaitem> element describes a media item. It may be empty, or contain text which is displayed on the screen.

```
<!ELEMENT mediaitem (#PCDATA | promptdoc)*>
```

```
<!ATTLIST mediaitem mimetype CDATA #IMPLIED
charset CDATA #IMPLIED
src CDATA #IMPLIED
alt CDATA #IMPLIED
autoplay CDATA #IMPLIED
modal CDATA #IMPLIED
width CDATA #IMPLIED
```

```

height CDATA #IMPLIED
volume CDATA #IMPLIED
annotationTemplate CDATA #IMPLIED
languageISO639code CDATA #IMPLIED
countryISO3166code CDATA #IMPLIED
>

```

All attributes are optional.

mimetype specifies the type of prompt item. For image and audio prompts, this attribute provides a hint for displaying the prompt item – image items are drawn on the screen, audio is played via the system speakers or a headphone. Default MIME type is text/plain. **charset** Charset of external text prompt file. The encoding of prompt text embedded in the recording script is inherited from the encoding of the entire recording script which is by default UTF-8. **src** is a file name or a URL from which a prompt item is retrieved.

alt contains the text that is displayed if the item cannot be retrieved from the external source.

autoplay, **modal** and **volume** apply only to time-dependent prompt items, i.e. audio clips. If **autoplay** is set to *yes* the clip plays automatically as soon as the item is displayed, otherwise the user has to start playback explicitly. With **modal** set to *yes* item playback cannot be interrupted, and **volume** determines the audio volume for playback.

width and **height** specify the width and height in pixels of the image or video to display.

It is up to the recording script author to set the **mediaitem** attribute values to meaningful values. *SpeechRecorder* accepts the combinations given in table 1.8.

| mimetype | src | alt | autoplay | modal | width | height | volume |
|---------------|-----|-----|----------|-------|-------|--------|--------|
| text/plain | – | – | – | – | – | – | – |
| text/x-prompt | – | – | – | – | – | – | – |
| text/html | URL | – | – | – | – | – | – |
| text/rtf | URL | – | – | – | – | – | – |
| image/png | URL | + | – | – | + | + | – |
| image/jpeg | URL | + | – | – | + | + | – |
| image/svg+xml | URL | + | – | – | + | + | – |
| audio/x-wave | URL | + | + | + | – | – | + |

Table 1: Meaningful **<mediaitem>** attribute combinations.

An audio **<mediaitem>** element without contents displays a generic symbol for audio playback. An audio **<mediaitem>** element combined with text or image media displays the text or image contents and plays back the audio.

Formatted text prompts

The MIME type **text/x-prompt** is not an official MIME type. This type is only defined and usable in *SpeechRecorder* to display formatted text prompts.

The formatted text is described by a structure of some XML elements. The DTD definitions of these elements defined in DTD (since version 4) are:

```

<!ELEMENT promptdoc (body?)>

<!ELEMENT body (p)*>

<!ELEMENT p (text | font | br)*>

<!ELEMENT text (#PCDATA)>
<!--ATTLIST text decoration CDATA #IMPLIED-->
<!--ATTLIST text color CDATA #IMPLIED-->

<!ELEMENT font (text)>
<!--ATTLIST font size CDATA #IMPLIED-->
<!--ATTLIST font style CDATA #IMPLIED-->
<!--ATTLIST font weight CDATA #IMPLIED-->

<!ELEMENT br EMPTY>

```

The root element `<promptdoc>` of the prompt text document must be a child element of the `<mediaitem>` element. the `<body>` element contains the document structure: A paragraph is represented by the `<p>` element. Text must always be encapsulated by `<text>` elements. To use different font settings as the default settings defined by the project configuration the `<text>` elements can be enclosed by `` elements. The attributes `size`, `style` and `weight` define the font to use. The line break `
` element forces a new line.

Sample media items

The following `<mediaitem>` element displays a text prompt:

```

<mediaitem mimetype="text/plain">
    Welcome to the SpeechRecorder Demo Script.
</mediaitem>

```

This `<mediaitem>` element shows a formatted text loaded from a file:

```

<mediaitem mimetype="text/rtf"
    src="promptText.rtf" />

```

Note: RTF prompts only work on Windows operating systems.

This `<mediaitem>` element shows an image loaded from a relative (to the project directory) URL:

```

<mediaitem mimetype="image/jpeg"
    src="images/FelixWas.jpg"
    alt="Boy and washing machine" />

```

This `<mediaitem>` element shows an image loaded from a absolute URL:

```

<mediaitem mimetype="image/jpeg"
    src="http://www.speechrecorder.org/prompts/images/FelixWas.jpg"
    alt="Boy and washing machine" />

```

Formatted text prompts:

The following prompt should look almost the same as a plain text prompt:

```
<mediaitem mimetype="text/x-prompt">
  <promptdoc>
    <body>
      <p>
        <text>Welcome to SpeechRecorder Demo Script.</text>
      </p>
    </body>
  </promptdoc>
</mediaitem>
```

An often requested feature was the possibility to force a new line in the prompt. This can now be achieved with the line break `
` element:

```
<mediaitem mimetype="text/x-prompt">
  <promptdoc>
    <body>
      <p>
        <text>Welcome to SpeechRecorder Demo Script.</text>
        <br/>
        <text>How are you?</text>
      </p>
    </body>
  </promptdoc>
</mediaitem>
```

Often there is also the requirement to emphasize parts of the prompt text, in the following examples the word apple:

Underlined:

```
<mediaitem mimetype="text/x-prompt">
  <promptdoc>
    <body>
      <p>
        <text>She planted an </text>
        <text decoration="underline">apple</text>
        <text> tree in her garden.</text>
      </p>
    </body>
  </promptdoc>
</mediaitem>
```

Colored (green):

```
<mediaitem mimetype="text/x-prompt">
  <promptdoc>
    <body>
      <p>
        <text>She planted an </text>
        <text color="green">apple</text>
      </p>
    </body>
  </promptdoc>
</mediaitem>
```

```

        <text> tree in her garden.</text>
    </p>
</body>
</promptdoc>
</mediaitem>

```

Use font settings:
Increase font size (by 150%):

```

<mediaitem mimetype="text/x-prompt">
  <promptdoc>
    <body>
      <p>
        <text>She planted an </text>
        <font size="150.0%">
          <text>apple</text>
        </font>
        <text> tree in her garden.</text>
      </p>
    </body>
  </promptdoc>
</mediaitem>

```

Italic font:

```

<mediaitem mimetype="text/x-prompt">
  <promptdoc>
    <body>
      <p>
        <text>She planted an </text>
        <font style="italic">
          <text>apple</text>
        </font>
        <text> tree in her garden.</text>
      </p>
    </body>
  </promptdoc>
</mediaitem>

```

Combine underline decoration, green text, font size 150% and italic style:

```

<mediaitem mimetype="text/x-prompt">
  <promptdoc>
    <body>
      <p>
        <text>She planted an </text>
        <font size="150%" style="italic">
          <text color="green" decoration="underline">apple</text>
        </font>
        <text> tree in her garden.</text>
      </p>
    </body>
  </promptdoc>
</mediaitem>

```

1.8.1 Annotation templates

If you plan to use EMU-SDMS to annotate and analyze your recordings later you may mark a text media prompt item as annotation template. This means that the speaker is expected to read the prompt txt. If you export your project as an EMU-DB when the recordings are finished, the prompt text will appear in the TPL level of the exported database.

```
<recording itemcode="demo_031" postrecdelay="500" prerecdelay="2000" recduration="10000">
  <recinstructions mimetype="text/plain">Lisez la phrase</recinstructions>
  <recprompt>
    <mediaitem annotationTemplate="true" languageISO639code="fr">A Paris il y a 14 l
  </recprompt>
  <recomment> French sentence: In Paris there are 14 metro lines, 9 of which cross th
</recording>
```

If the expected spoken word(s) differ from the prompt a separate XML element 'annotationtemplate' for the annotation template is required. For example:

```
<recording itemcode="demo_022" postrecdelay="500" prerecdelay="2000" recduration="10000">
  <recinstructions mimetype="text/plain">Bitte ergnzen Sie</recinstructions>
  <recprompt>
    <mediaitem languageISO639code="de">Morgenstund hat ...</mediaitem>
  </recprompt>
  <annotationtemplate languageISO639code="de">Gold im Mund</annotationtemplate>
</recording>
```

Note: The optional attribute `languageISO639code` indicates the language of the template text as *ISO639* code.

2 Recording Phases

Each recording is performed as a sequence of phases. The sequence of phases is shown in fig. 5.

A modal prompt display means that the prompt item is shown, but marked as inactive, e.g. by using greyed-out text, low resolution images or a disabled audio button. The default setting is to have modal prompt display during the prerecording and postrecording phases, and an active prompt display during recording. The attribute `promptphase` of a `<section>` element determines the start of an active prompt display, and it overrides the default setting.

IDLE no recording, red light, prompt item is only displayed if the attribute `promptphase` is set to *idle*.

PLAYPROMPT the prompt is playing.

PRERECORDING recording, yellow light, modal prompt item display.

RECORDING recording, green light, active prompt item display.

POSTRECORDING recording, yellow light, modal prompt item display.

A prerecording phase is useful to either record environment noise prior to the main recording, or to give the speaker a precisely delimited time to prepare the utterance. A postrecording phase is most commonly used to avoid signal truncation due to clicking the `stop` button too early.

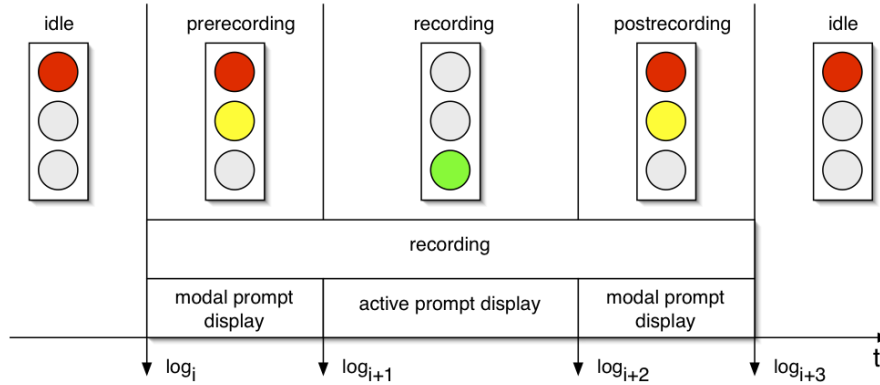


Figure 5: Recording phases

The timing for time-dependent prompts has to be set to appropriate values by the script author, e.g. to make sure that the recording time is sufficient for prompt playback and recording.

3 Menu File

Contains commands to save (configuration) files, to print a screenshot of current session and to quit.

4 Menu View

Contains commands to toggle the speaker display on and off and to customize the audio signal view.

5 Menu Workspace

The **Workspace** menu item opens an overview of the workspace. It shows the location of the workspace and a list of all projects. The context menu (right mouse click) of each project allows you to create a new, rename, duplicate, export to archive file or to EMU-DB, delete, open or close the project. (Some of these actions are only enabled if the project is closed.)

6 Menu Project

Contains commands to create new projects, open a project from the list of known projects or close a project. These projects must reside in the *SpeechRecorder*

directory in the user's home directory. A project can only be opened if no other project is open.

6.1 Menu item Export

The **Export** command packs a project archive in a zip-archive file. The **Import** command unpacks a project in a zip-archive file. The project will be deployed in the project directory.

6.2 Menu item Preferences

The **Preferences...** command opens a dialog window with the tabs **Project**, **Speakers**, **Recording**, **Playback**, **Control**, **Prompting**, **Annotation** and **Logging**.

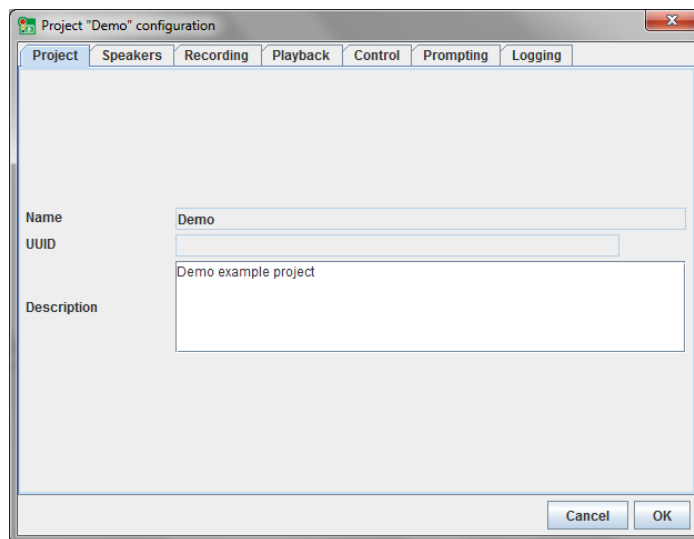


Figure 6: Settings > Project... command

6.2.1 Tab Project

The **Project** tab (fig. 6) presents the project name, optional UUID and optional description. The purpose of the UUID is to distinguish different projects with the same name in the future.

6.2.2 Tab Speakers

The **Speakers** tab contains the location of the speaker database. This database can be stored in the project directory, or any other accessible location in the local file system.

6.2.3 Tab Recording

The **Recording** tab allows the user to set the recording parameters. The tab has the subtabs **Device**, **Channel routing**, **Format** and **Options**. In the **Device**

tab (fig. 7) you can select the recording device SpeechRecorder should use for this project.

Tab Channel routing In the **Channel routing** tab it is possible to specify arbitrary connections between recording device channels and output file channels. This is only necessary for advanced recording setups.

Tab Format The sample rate, quantization, byte order, encoding, number of channels can be chosen in the **Format** tab.

Tab Options In the **Options** tab you can set the following parameters:

Audio capture scope If set to **ITEM** the capture device is opened and closed for each recording, if set to **SESSION** it is hold open during one recording session. **SESSION** is the default over **ITEM** for new projects starting with Speechrecorder version 3.0.x. We experienced recording problems with some capture devices, when they are openend for each recording: Some notebook audio chips switch the auto gain control on/off, which causes low frequency amplitudes at the begining of the recording. Some devices show a delay from opening the device to actually streaming audio data. And we have seen devices which do not deliver any data in rare cases.

Recording target **DIRECT** stores the captured audio data directly to the final audio (wav) file. **TEMP_RAW** stores first to a temporary file and if recording has stopped converts the audio data to the final audio file. **TEMP_RAW** is recommended if you store your recordings to a network file system if the throughput is not sufficient to store realtime audio data.

Overwrite The **Overwrite** control determines whether repeated recordings of an item overwrite previous ones or are stored as versions.

Overwrite warning If set the experimenter is prompted before recordings are overwritten.

Recording mode The **Recording mode** control sets the default recording progress mode. (Can be overridden by the recording script)

Seamless (auto) recording Using **Seamless (auto) recording** the recording files of an autorecording session will have no gaps. They could be concatenated later and maybe cutted at different timestamps. (Concatenation and cutting is not yet supported by Speechrecorder). Seamless recording is only supported if **Recording target** is set to **DIRECT**.

Autoprogress to next unrecorded item If checked the recording session will jump to the next unrecorded item in autoprogress or autorecording mode.

Reset peak at start of recording If checked the peak level hold value of the level meter is reset on each recording start. Otherwise the maximum peak level hold during the recording session. Default is to reset on start of each recording.

Default prerecording delay Default pre recording delay in milliseconds. May be overridden in recording script for each item.

Default postrecording delay Default post recording delay in milliseconds. May be overridden in recording script for each item.

Force post recording phase If checked a post recording phase will be executed even if the maximum recording length is reached.

Recording URL (directory) The URL where recordings will be stored. The default is the directory 'RECS' inside the project directory. Standalone Speechrecorder supports only the 'file:' protocol, therefore recordings can only be stored to the computers file system. You can select a directory by clicking the 'Browse...' button. If you select a directory on a network file system you might consider to change 'Recording target' to TEMP_RAW if you experience I/O errors. If you change the recording directory exiting recording data will not be moved to the new location. You have to do this manually.

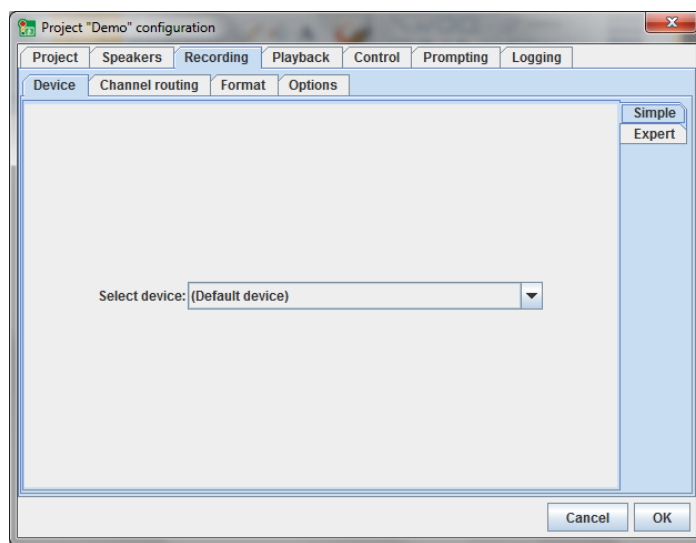


Figure 7: Settings > Project... > Recording > Device dialog window

6.2.4 Tab Prompting

The **Prompting** tab contains other tabs to edit the type of start stop signal (the traffic light), select fonts, their style and size for prompt, instructions and comment texts, the location of the recording script file, default value for prompt

autoplay (audio prompts), customizing the speaker window, project settings for the script item generator and playback device settings for audio prompts.

6.2.5 Tab Annotation

Speechrecoder is not an annotation tool, but it maybe helpful to store the prompt texts as annotation templates in common used annotation file formats.

Tab Persist Select the file formats here. Supported formats are:

- EMU Speech Database ManagementSystem [WJCH16]
 - <https://cran.r-project.org/web/packages/emuR/>
 - <https://github.com/IPS-LMU/emuR>
- Praat TextGrid [BW16] , <http://www.praat.org>
- Text file (single line)

Tab Auto annotation Select which levels/tiers to write to the annotation file:

- PRT: The prompt (text) itself
- TPL: Annotation template text. Either same as the prompt text if it is marked as annotation template in the recording script or a different text entered in the 'Annot. template' tab in the script editor.

Tab Logging Configure contents of log files and their verbosity and format here.

7 Menu Speaker

The command `Speakers...` opens the speaker database.

8 Menu Script

Contains commands to open the recording script with an editor or to edit the XML source of the script.

8.1 Edit script...

Edit your recording script using the graphical interface

8.2 Edit script XML source

Opens a simple integrated text editor to edit the XML source of your recording script.

8.3 Script resources

Opens the resource directory of your project with the file manager of your operating system. The resource directory is mainly intended for prompt media files (image, audio and video files) referenced by your recording script. If the resource directory does not yet exist for your project you will be asked to create it.

8.4 Import text table

Opens a dialog to import text table files (list of prompt texts) in your recording script.

8.5 Export script...

Opens a dialog to export your recording script as original XML file. (Among other things the script can be exported for an import to a WikiSpeech project.)

8.6 Export script as text table file...

Opens a dialog to export your recording script as a text table/list file. Please note that the text file will not contain all informations of the XML recording script.

9 Menu Settings

The `Skip...` command can be used to skip to a given recording item. The `Recording...` command opens an audio mixer GUI (but we recommend to use the audio mixer of the operating system).

10 Recordings via the Internet

One of the most interesting features of *SpeechRecorder* is its ability to transfer audio files to a remote server. This is achieved using the `http` (*hypertext transfer protocol*) in combination with the `post` method for sending data from the client to a server.

We host some recording projects on our `http://webapp.phonetik.uni-muenchen.de/wikispeech/` WikiSpeech database system [DJ08].

```
@InProceedings{ips_bibtex_4966,
author = {Draxler, Chr. and J\"ansch, K.},
title = {{W}iki{S}peech -- {A} {C}ontent {M}anagement {S}ystem for {S}peech {D}atabases},
booktitle = {Proc. of Interspeech},
year = {2008},
address = {Brisbane}
}
```

Please contact us, if you are interested in starting a WikiSpeech project.

A Recording script DTD

```
<!ELEMENT session (metadata?, recordingscript)>
<!ELEMENT script (metadata?, recordingscript)>

<!ATTLIST session id CDATA #REQUIRED>
<!ATTLIST script id CDATA #REQUIRED>

<!ELEMENT metadata (key, value)+>

<!ELEMENT properties (key, value)+>

<!ELEMENT key (#PCDATA)>

<!ELEMENT value (#PCDATA)*>


<!ELEMENT recordingscript (virtualviewbox?,section*)>

<!ELEMENT virtualviewbox EMPTY>
<!ATTLIST virtualviewbox height CDATA #IMPLIED>

<!ATTLIST section name CDATA #IMPLIED
               speakerdisplay CDATA #IMPLIED
               order CDATA #IMPLIED
               mode CDATA #IMPLIED
               promptphase CDATA #IMPLIED
>

<!ELEMENT section (group | nonrecording | recording)*>

<!ATTLIST group order CDATA #IMPLIED>

<!ELEMENT group (nonrecording | recording)*>

<!ELEMENT nonrecording (presenter?, mediaitem+)>

<!ATTLIST nonrecording duration CDATA #IMPLIED>

<!ELEMENT recording (recinstructions?, recprompt, recomment?,annotationtemplate?) >

<!ATTLIST recording itemcode CDATA #REQUIRED
recduration CDATA #IMPLIED
prerecdelay CDATA #IMPLIED
postrecdelay CDATA #IMPLIED
finalsilence CDATA #IMPLIED
beep CDATA #IMPLIED
rectype CDATA #IMPLIED
```

```

blocked CDATA #IMPLIED

>

<!ELEMENT recinstructions (#PCDATA) >

<!ATTLIST recinstructions mimetype CDATA #IMPLIED
charset CDATA #IMPLIED
src CDATA #IMPLIED
>

<!ELEMENT recprompt (presenter?, mediaitem+)>
<!ATTLIST recprompt target CDATA #IMPLIED>

<!ELEMENT presenter (properties?)>
<!ATTLIST presenter type CDATA #IMPLIED
classname CDATA #IMPLIED
>

<!ELEMENT recomment (#PCDATA)>

<!ELEMENT promptdoc (body?)>

<!ELEMENT body (p)*>

<!ELEMENT p (text | font | br)*>

<!ELEMENT text (#PCDATA)>
<!ATTLIST text decoration CDATA #IMPLIED>
<!ATTLIST text color CDATA #IMPLIED>

<!ELEMENT font (text)>
<!ATTLIST font size CDATA #IMPLIED>
<!ATTLIST font style CDATA #IMPLIED>
<!ATTLIST font weight CDATA #IMPLIED>
<!ELEMENT br EMPTY>

<!ELEMENT mediaitem (#PCDATA | promptdoc)*>

<!ATTLIST mediaitem mimetype CDATA #IMPLIED
charset CDATA #IMPLIED
src CDATA #IMPLIED
alt CDATA #IMPLIED
autoplay CDATA #IMPLIED
modal CDATA #IMPLIED
width CDATA #IMPLIED
height CDATA #IMPLIED
volume CDATA #IMPLIED

```



```

annotationTemplate CDATA #IMPLIED
languageISO639code CDATA #IMPLIED
countryISO3166code CDATA #IMPLIED

>

<!ELEMENT annotationtemplate (#PCDATA)>
<!ATTLIST annotationtemplate
languageISO639code CDATA #IMPLIED
countryISO3166code CDATA #IMPLIED

>

```

B Reserved keywords for recording scripts

A recording script may contain the following keywords for recording progress, presentation order, and recording type. Recording progress and presentation order are defined via attributes of the tag `<section>`, recording type via an attribute of `<recording>`, and mime-types via `<mediaitem>`.

recording progress: attribute `mode`, values *manual*, *autoprogess*, *autorecording*.

presentation order: attribute `order`, values *sequential*, *random*.

recording type: attribute `rectype`, values *audio*, *video*. Note: *video* as a recording type is not yet implemented.

mime-types: *text/plain* for text, *audio/x-wave*, *audio/x-aiff* for audio, *image/jpeg*, *image/gif* for images.

C Known issues

The following list contains some of the known problems of *SpeechRecorder*. If you find further bugs and errors, please contact draxler@phonetik.uni-muenchen.de.

- When recording with a laptop please disconnect the power adaptor and record using the battery only. External power adaptors may interfere with the audio signal, causing strong distortions. The effect may vary with the brand and type of laptop.
- If you want to use an external XML editor to edit the recording script:
 - Windows: Windows 7 Standard Editor, jEdit (jedit.sourceforge.net) or Atom Editor(atom.io)
 - Mac: we recommend to use the free editor TextWrangler (www.barebones.com) which allows you to import and export content in many encodings.
 - Linux: vim,gedit,jedit,Atom,...
- The following attributes are defined in the recording script DTD, but have not yet been implemented in *SpeechRecorder*:

- `mimetype` and `src` attributes for the `<recinstructions>` element are ignored.
- Silence detection to stop a recording is not yet implemented.
- Recording/prompting video is not yet implemented.
- The directory name into which the audio files are saved is named after the speaker number in the speaker database. Currently, this number is not visible in the speaker database.
- Attribute values for `itemcode` may be arbitrary strings, and the value becomes part of the audio file name. This may cause problems if the string contains characters that have a special meaning for the file system, e.g. `"/`, `":`, `","`, etc. It is thus recommended to use only the characters `a-z`, `A-Z`, `_` and `0-9` for the `itemcode` attribute.

C.1 Platform dependencies

- *Mac OS X*: Audio Interface CoreAudio only supports sample rates supported directly by the hardware. Use 'JavaSound' interface for other sample rates on Mac OS X later than 10.7.
- *Windows XP*: If the built-in USB audio driver is used, the sample rate will be set to 22.05 kHz when a Windows system beep is output via an M-Audio mobile pre USB device. *SpeechRecorder* does not detect this change of sample rate. All subsequent recordings will be made using the new sample rate. **Solution**: make sure to install and select the most recent M-Audio device driver from <http://de.m-audio.com/>.
- *Windows 7,8,10*: The operating system distinguishes audio devices of the same model by inserting a counter number in the device name. This only happens if you connect the same device to different (USB) ports. It is recommended to always connect to the same USB port. Another workaround is to use a regular expression for the audio device name which matches the numbered audio devices.

D Signal quality problems

If you encounter problems with the signal quality please check your setup.

- When recording with a laptop computer, please disconnect the power cord and run the computer on batteries only. The power supply of many laptops is not well shielded, resulting in strong interferences in the signal (see 8).

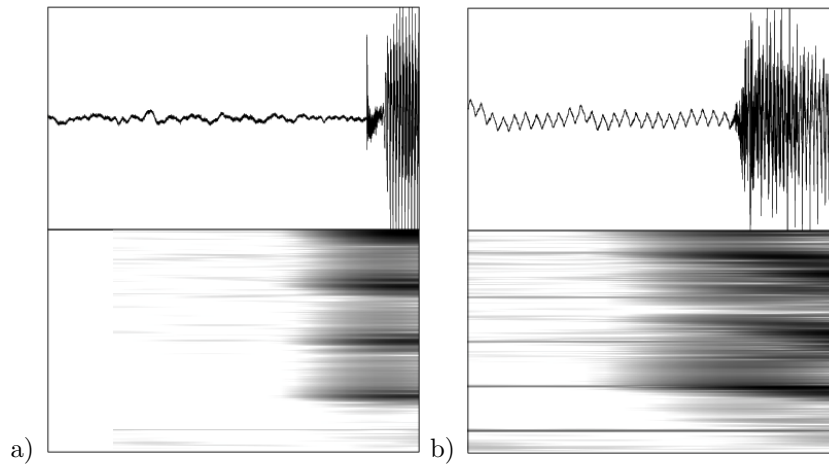


Figure 8: Oscillogram and sonogram of an audiosignal, recorded on a laptop running a) on batteries and b) connected to the power mains. The right oscillogram shows a marked sawtooth-pattern, and the sonogram shows thin horizontal lines at multiples of the mains frequency.

- Recordings have drop outs. If you use an external audio interface, please check if you can adjust the buffer size (latency) of the device to the highest level. Many devices come with special driver and control software to adjust values like levels and latency. The default configuration may be optimized for musicians, these need low latencies. But for reliable recordings it is better to have large buffers and therefore a high latency. Speechrecorder does some checks to detect buffer overruns and drop outs, but the capabilities of these checks are limited. If you are prompted about buffer overruns or that a buffer was read too late, your PC may be busy or blocked. This may for example be caused by automatic software update installations on Windows. Furthermore try to avoid that your PC switches to energy saving modes during a recording.
- Gain control

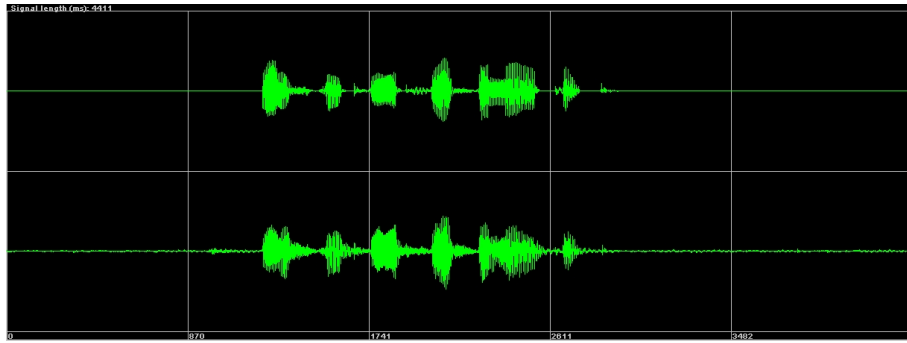


Figure 9: Optimal gain control: The signal covers the middle area between the top and bottom line. The signal should never reach the top and bottom lines but only fill about a third of the signal area.

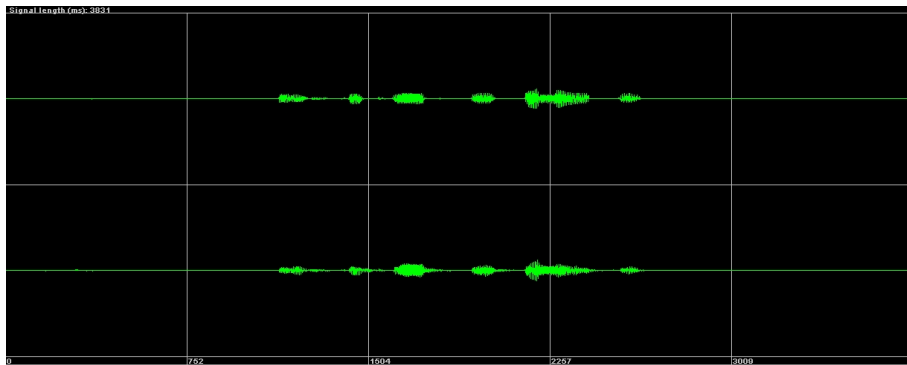


Figure 10: This signal is a little bit low, but still OK.

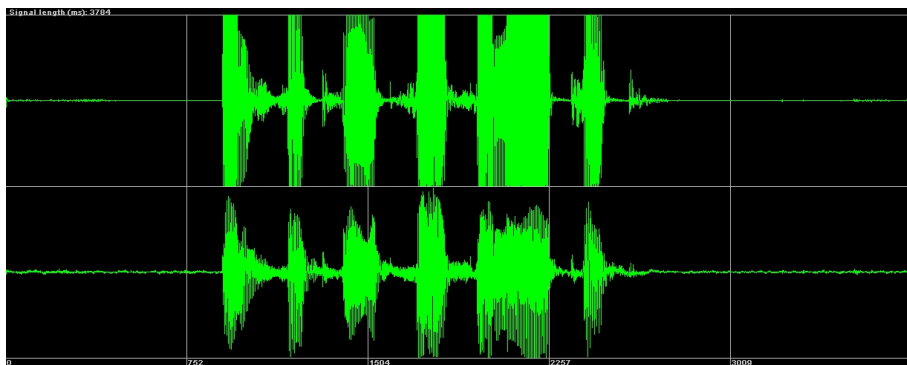


Figure 11: Overmodulated: the signal reaches or exceeds the top and bottom line! Unfortunately, these recordings are unusable!

E Contacts and Copyright

SpeechRecorder is being developed by the Institute of Phonetics and Speech Processing of Ludwig-Maximilian University in Munich, Germany. Its main authors are Christoph Draxler and Klaus Jänsch. Many people have contributed to the software by providing localized versions of the graphical user interface, or by suggesting improvements to the software.

The software is Copyright ©2004-2016 by Ludwig-Maximilian University of Munich, Germany. This program comes with ABSOLUTELY NO WARRANTY.

Please cite the original article describing *SpeechRecorder* if you refer to the software in your publications: [DJ04]. Alternatively, you may use the following BibTeX-formatted record:

```
@inproceedings{DraxlerJaensch2004,
  Author = {Chr. Draxler and K. J{"a"}nsch},
  Title = {{SpeechRecorder -- a Universal Platform Independent
           Multi-Channel Audio Recording Software}},
  Booktitle = {Proc. of LREC},
  Pages = {559-562},
  Address = {Lisbon},
  Year = {2004}}
```

You may use the software free of charge for academic, research and development, and commercial purposes. We particularly encourage the use of *SpeechRecorder* in university or school courses on speech recording.

You may distribute the software freely, provided that the packed `.jnlp` or `.jar` files are not altered.

The software is provided as is. The authors and Ludwig-Maximilians University cannot be held responsible for any damage caused by the use of the software.

F Useful links

- <http://www.phonetik.uni-muenchen.de/Bas/software>: BAS pages with links to software and updates

References

- [BW16] Paul Boersma and David Weenink. Praat: doing phonetics by computer [computer program]. version 6.0.15, 2016.
- [DJ04] Chr. Draxler and K. Jänsch. SpeechRecorder – a Universal Platform Independent Multi-Channel Audio Recording Software. In *Proc. of LREC*, pages 559–562, Lisbon, 2004.
- [DJ08] Chr. Draxler and K. Jänsch. WikiSpeech – A Content Management System for Speech Databases. In *Proc. of Interspeech*, Brisbane, 2008.
- [WJCH16] Raphael Winkelmann, Klaus Jaensch, Steve Cassidy, and Jonathan Harrington. *emuR: Main Package of the EMU Speech Database Management System*, 2016. R package version 0.1.7.