```c
/*
 * different addons for AKFAvatar
 * Copyright (c) 2007,2008,2009,2010,2011,2012,2013
 * Andreas K. Foerster <info@akfoerster.de>
 *
 * This file is part of AKFAvatar
 *
 * AKFAvatar is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * AKFAvatar is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, see <http://www.gnu.org/licenses/>.
 */

/*
 * Not all parts of this library are available on all platforms!
 */

#ifndef AVTADDONS_H
#define AVTADDONS_H

#include "akfavatar.h"
#include <wchar.h>
#include <stdlib.h>
#include <stdarg.h>
#include <unistd.h>
#include <string.h>
#include <fcntl.h>

/* some of these functions are used as parameters */
#ifdef __cplusplus
extern "C" {
#endif

#define AVT_ADDON  extern

/*****************************************************************
 * Section: character encodings
 * support for single byte charsets via tables
 *****************************************************************/

/*
 * ISO-8859
 * -2 (Latin-2) For eastern European languages
 * -3 (Latin-3) For southern European languages
 * -4 (Latin-4) For northern European languages
 * -5 (Cyrillic) see also KOI8-R and KOI8-U, which are more widely used
 * -7 (Greek) For modern Greek
 * -8 (Hebrew) For modern Hebrew/Yiddish (use with care!)
 * -9 (Latin-5) For Turkish
 * -10 (Latin-6) For Nordic languages
 * -11 (Thai)
 * -13 (Latin-7) For Baltic languages
 * -14 (Latin-8) For Celtic languages
 * -15 (Latin-9) For western European languages (with Euro)
 * -16 (Latin-10) For south-eastern European languages
 */
AVT_ADDON const struct avt_charenc *avt_iso8859_2 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_3 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_4 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_5 (void);
```

```
AVT_ADDON const struct avt_charenc *avt_iso8859_7 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_8 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_9 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_10 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_11 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_13 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_14 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_15 (void);
AVT_ADDON const struct avt_charenc *avt_iso8859_16 (void);

/*
 * KOI8-R, KOI8-U
 * Cyrillic alphabet for Russian and Ukrainian
 * KOI stands for "Đ\232Đ¾Đ´ Đ\236Đ±Đ¼ĐµĐ½Đ° Đ\230Đ½Ñ\204Đ¾Ñ\200Đ¼Đ°Ñ\206Đ¸ĐµĐ¹, 8 Đ±Đ¸Ñ
\202"
 *
 * see also RFC 1489, RFC 2319
 */
AVT_ADDON const struct avt_charenc *avt_koi8r (void);
AVT_ADDON const struct avt_charenc *avt_koi8u (void);

/*
 * Vendor specific encodings
 */

/*
 * Code page 437 (IBM437)
 * This is the native charset of IBM-PC compatibles.
 * It is the built in charset for the textconsole.
 * It is used for .nfo files.
 *
 * implemented for comatibility with textconsole programs
 *
 * The characters 0-31 are interpreted as control characters
 */
AVT_ADDON const struct avt_charenc *avt_cp437 (void);

/*
 * Code page 850 (IBM850, DOS-Latin-1)
 * This has all printable characters of ISO Latin-1,
 * but in different positions.
 *
 * used for textconsole programs under Windows in western Europe
 *
 * The characters 0-31 are interpreted as control characters
 */
AVT_ADDON const struct avt_charenc *avt_cp850 (void);

/*
 * Maps for Microsoft Windows codepages
 * 1250 (central an eastern European)
 * 1251 (Cyrillic)
 * 1252 (Western languages)
 *
 * Microsoft calls them "ANSI", though they are _not_ ANSI standards!
 *
 * implemented, because they are widely used
 */
AVT_ADDON const struct avt_charenc *avt_cp1250 (void);
AVT_ADDON const struct avt_charenc *avt_cp1251 (void);
AVT_ADDON const struct avt_charenc *avt_cp1252 (void);

/*
 * use the systems charset as encoding
 * depends on the locale for LC_CTYPE
 *
 * use with care!
 * just use it for filenames and messages from the system
```

```
 * on some systems it is badly supported
 */
AVT_ADDON const struct avt_charenc *avt_systemencoding (void);

/*
 * the following definitions are only for internal use
 * for charmap definitions
 */

struct avt_char_map
{
  unsigned short int start, end;
  unsigned short int table[];         // limited to the BMP!
};

AVT_ADDON size_t map_to_unicode (const struct avt_charenc *self,
                                 avt_char *dest,
                                 const char *src);

AVT_ADDON size_t map_from_unicode (const struct avt_charenc *self,
                                   char *dest, size_t size,
                                   avt_char src);

/*****************************************************************
 * Section: avtccio
 * C-specific functions for input/output
 *
 * the calling program must have used avt_initialize and
 * avt_mb_encoding before calling any of these functions.
 *****************************************************************/

AVT_ADDON int avt_printf (const char *format, ...);
AVT_ADDON int avt_putchar_char (int c);
AVT_ADDON int avt_puts (const char *s);
AVT_ADDON int avt_scanf (const char *format, ...);
AVT_ADDON int avt_vprintf (const char *format, va_list ap);
AVT_ADDON int avt_vscanf (const char *format, va_list ap);


/*****************************************************************
 * Section: avtcwio
 * C-specific functions for wide characters input/output
 *
 * the calling program must have used avt_initialize before calling
 * any of these functions.
 *****************************************************************/

AVT_ADDON int avt_wprintf (const wchar_t *format, ...);
AVT_ADDON wint_t avt_putwchar (wchar_t c);
AVT_ADDON int avt_putws (const wchar_t *s);
AVT_ADDON int avt_wscanf (const wchar_t *format, ...);
AVT_ADDON int avt_vwprintf (const wchar_t *format, va_list ap);
AVT_ADDON int avt_vwscanf (const wchar_t *format, va_list ap);


/*****************************************************************
 * Section: colorchooser
 * color-chooser for AKFAvatar
 *****************************************************************/

AVT_ADDON const char *avt_color_selection (void);


/*****************************************************************
 * Section: filechooser
 * file-chooser for AKFAvatar
 *****************************************************************/
```

```
typedef bool (*avt_filter) (const char *filename, void *data);

/*
 * file selection
 * starts in current working directory - changes the working directory
 * if filter is NULL it shows all files (but no hidden files)
 * data is passed to the filter (may be NULL)
 * returns 0 on success or -1 on error
 */
AVT_ADDON int avt_file_selection (char *filename, int filename_size,
                                  avt_filter filter, void *data);


/*********************************************************************
 * Section: avtreadfile
 * for reading a file into a buffer
 * the buffer is allocated with malloc and must be freed by the caller
 *********************************************************************/

/*
 * read a text file
 * if file_name is NULL read from stdin
 * the string will be termitated by four zero-bytes
 * returns length of the string without the terminating zeros,
 * or -1 on error
 */
AVT_ADDON int avt_read_textfile (const char *file_name, char **buffer);

/*
 * read a data file
 * returns the size of the buffer in bytes, or -1 on error
 */
AVT_ADDON int avt_read_datafile (const char *file_name, void **buffer);

/*
 * get the output of the command (not for interactive programs)
 * the string will be termitated by four zero-bytes
 * returns length of the string without the terminating zeros,
 * or -1 on error
 */
AVT_ADDON int avt_read_command (const char *command, char **buffer);

/*
 * run pager on file with current encoding and balloon-size
 * returns 0 on success or -1 on error
 */
AVT_ADDON int avt_pager_file (const char *file_name, int startline);

/*
 * run pager on output of command with current encoding and
 * balloon-size
 * returns 0 on success or -1 on error
 */
AVT_ADDON int avt_pager_command (const char *command, int startline);


/*********************************************************************
 * Section: avtvorbis
 * Ogg Vorbis support for AKFAvatar based on stb_vorbis
 *********************************************************************/

AVT_ADDON avt_audio *avt_load_vorbis_file (char *filename, int playmode);

AVT_ADDON avt_audio *avt_load_vorbis_data (void *data,
                                           int datasize,
                                           int playmode);

/*
```

```
 * read from a stream at current position size bytes maximum
 * if size is 0 then get the rest of the file
 * on error it restores the file position and returns NULL
 */
AVT_ADDON avt_audio *avt_load_vorbis_stream (avt_stream *stream,
                                             size_t size,
                                             bool autoclose,
                                             int playmode);


/*************************************************************************
 * Section: language
 * getting a language code for messages
 *************************************************************************/

/*
 * returns lowercase language code
 * (should be ISO 639-1 or ISO 639-2)
 * or NULL on error
 */
AVT_ADDON const char *avt_get_language (void);

/*************************************************************************
 * Section: base directory
 * get the base directory of executable
 * currently for GNU/Linux and Windows only
 *************************************************************************/

/*
 * sets name to the base directory
 * name should be a buffer of the given size
 * returns 0 on success and -1 on error
 */
AVT_ADDON int avt_base_directory (char *name, size_t size);

/*************************************************************************
 * Section: arch
 * functions for handling ar archives
 *************************************************************************/

/*
 * return file descriptor, if it's an archive
 * or -1 on error
 */
AVT_ADDON int avt_arch_open (const char *archive);

/*
 * finds a member in the archive
 * and leaves the fileposition at its start
 * the member name may not be longer than 15 characters
 * returns size of the member, or 0 if not found
 */
AVT_ADDON size_t avt_arch_find_member (int fd, const char *member);

/*
 * finds first archive member
 * and leaves the fileposition at its start
 * if member is not NULL it will get the name of the member
 * member must have at least 16 bytes
 * returns size of first member
 */
AVT_ADDON size_t avt_arch_first_member (int fd, char *member);

/*
 * read in whole member
 * the result is allocated with malloc and must be freed by the caller
 * the result gets some binary zeros added, so it can be used as string
 * returns NULL on error
```

```
 */
AVT_ADDON char *avt_arch_get_member (int fd, size_t size);

/*
 * read in whole member of a named archive
 * the result is allocated with malloc and must be freed by the caller
 * the result gets some binary zeros added, so it can be used as string
 * if size is not NULL it gets the size of the member without the terminator
 * returns NULL on error
 */
AVT_ADDON char *avt_arch_get_data (const char *archive, const char *member,
                                   size_t *size);


/*****************************************************************
 * Section: avtterm
 * terminal emulator for AKFAvatar
 * (not available for MinGW)
 *****************************************************************/

typedef int (*avt_term_apc_cmd) (wchar_t*);

/*
 * execute a subprocess, visible in the balloon
 * if prg_argv == NULL, start a shell
 * returns file-descriptor for output of the process
 * or -1 on error (also if not supported at all)
 */
AVT_ADDON int avt_term_start (const char *working_dir, char *prg_argv[]);

AVT_ADDON void avt_term_run (int fd);

AVT_ADDON void avt_term_nocolor (bool nocolor);


/* register handler for APC commands (optional) */
AVT_ADDON void avt_term_register_apc (avt_term_apc_cmd command);

/*
 * the following functions are only meant to be called
 * from the APC handler
 */

/* APC: (de)activate slowprint mode */
AVT_ADDON void avt_term_slowprint (bool on);

/* APC: send data to stdin of the running program */
AVT_ADDON void avt_term_send (const char *buf, size_t count);

/* APC: send string literal to stdin of the running program */
#define avt_term_send_literal(l)  avt_term_send("" l, sizeof(l)-1)

/*
 * APC: update size of textarea
 * call this after you have changed the size of the balloon
 */
AVT_ADDON void avt_term_update_size (void);

#ifdef __cplusplus
}
#endif

#endif /* AVTADDONS_H */
```