

Proxemo – Manual

Index

Background.....	2
Emotions.....	2
To run or modify Proxemo:	3
To Exchange Emojis	3
Documentation of the Tizen project ,Proxemo'	4
data.c	4
view.c.....	5
proxemo.c.....	10







Background

- It is not trivial to evaluate affect and emotions in people with dementia when using interactive systems:
- Available methods to measure *User Experience* require communicating one's current emotional state and thus the ability to assess/reflect feelings and additionally to communicate them.
- Proxy ratings known to the context of dementia as *Quality-of-Life* methods mostly consider time intervals that are too long and their scales are not optimized for evaluating interactive systems.
- We developed a smartwatch app to document emotions by proxy. Initially we used the Emotions by Lawton (1999): pleasure, sadness, fear, anger, general alertness. After several studies we switched to the following set of emotions empirically based in observations of people with dementia interacting with technology.

Emotions

We found that differentiating negative affect was not important for formative evaluations of interactive technology and thus merged them into one general negative category. Positive emotions were more interesting, which is why we introduced a more differentiated view on pleasure:

Traits (c.f. Lawton) only serve as reference:

	Pleasure: Laughing, singing, smiling, kissing, stroking or gently touching other, reaching out warmly to other, responding to music, statements of pleasure;
	Interest: participating in a task, maintaining eye contact, eyes following object or person, looking around room, responding by moving or saying something, turning body or moving toward person or object;
	Pride: Similar to pleasure, additionally: autobiographically based; e.g. proud of home town, a certain experience/journey, children/grandchildren, own skills;
	Wistfulness: Resident rejoices over/tells about positive event in the past, knowing that this time is over now; looking back with a smile and a heavy heart;
	General negative affect, merged from: Depression/Sadness: Cry, frowning, eyes drooping, moaning, sighing, head in hand, eyes/head turned down and face expressionless, statements of sadness; Anxiety/Fear: shrieking, repetitive calling out, restlessness, wincing/grimacing, repeated or agitated movement, line between eyebrows, lines across forehead, hand wringing, tremor, leg jiggling, rapid breathing, eyes wide, tight facial muscles, statements of anxiety/fear; Anger: physical aggression, yelling, cursing, berating, shaking fist, drawing eyebrows together, clenching teeth, pursing lips, narrowing eyes, making distancing gesture, statements of anger;
	Agency/action ability (center-button): special ability, which cannot be naturally assumed, e.g. resident reads out a text, recognizes and names a person/animal

To run or modify Proxemo:

- Download and install Tizen studio (<http://www.tizenstudio.com/tizen-studio/download/>)
- Copy Proxemo folder/files to local workspace (on PC usually C:/user/workspace)
- Start Tizen and import Proxemo
 - NEVER USE WHITESPACE IN PROJECT- /FOLDER-NAME
 - File > import with parameters
 - Tizen project
 - Chose entire folder as path
 - Wearable in Version 3.0
- Follow instructions on Tizen website to deploy App on Watch:
 - Get Samsung-Certificate (registration of a Samsung account necessary)
 - Connect watch with Tizen
 - Run Proxemo on Gear S2/S3

To Exchange Emojis

- Exchange Emojis
 - Get new emojis as .png file and paste in edge/images
- Exchange user-pictures on smartwatch
 - Create round pictures of users and name them as it shall appear in the logfile (e.g. user34.png)
 - Move them via Tizen studio to the folder <file:///opt/usr/media/Images> OR send pictures via "Samsung-Gear" app; they will be stored in the correct folder automatically
 - Restart the Proxemo-App
 - Tip: if you want to use the app in several sessions without connecting it to a host-device in between, use the app *Filesmaster* or similar to store pictures in the *download* Folder until you need them in *images*
- Extracting log-files from the smartwatch
 - All logs are stored in <file:///opt/usr/media/documents/ProxemoLog.csv>
 - You can pull them via Tizen studio OR send them to a Bluetooth device using *Filesmaster*
- Reading log files
 - Log files are in the format CSV and contain log entries as <date><time><user><emotion-ID>
 - <user> is the name of the picture
 - The meaning of <emotion-ID> changes with the used emojis and was used in the original version to log
 - 0 = center-button = agency/action ability/<whatever meaning you assign to it>
 - 1 = 12-o'clock = pleasure
 - 2 = 2-o'clock = wistfulness
 - 3 = 5-o'clock = pride
 - 4 = 7-o'clock = NEGATIVE: anger/frustration/depression/sadness/fear
 - 5 = 10-o'clock = general-alertness/curiosity

Documentation of the Tizen project ,Proxemo'

data.c

Method name	Brief description
void data_initialize(void)	Standard „Initialization function for data module“ from Tizen
void checkForFolders(void)	checks if the Folders media/images/ProxemoEmotions, media/images/ProxemoUsers and media/documents/ProxemoLogs exist and if they don't <u>copys</u> the folders from the shared resources folder
void getCurrentUserName(char* currentUserName)	Returns the username of the currently active node
void loadAllPictures(void)	Loads all Pictures from STORAGE_DIRECTORY_IMAGES and saves them as LinkedList with start and endNode connected
void nextUser(void)	Switches the currently active user for the next one in the linked node structure
void previousUser(void)	Switches the currently active user for the previous one in the linked node structure
void data_finalize(void)	Standard „Finalization function for data module“ from Tizen
void data_get_full_path(const char *file_path, char *full_path, int path_max)	Standard „Get full path of resource @ <u>param</u> [in] file_path File path of target file @ <u>param</u> [out] full_path Full file path concatenated with resource path @ <u>param</u> [in] path_max Max length of full file path“ from Tizen
char *data_get_image_path(const char *part_name)	Get the path of the image file from STORAGE_DIRECTORY_IMAGES @ <u>param</u> [in] part_name Part name of the target image path
void data_write_file(const char* buffer)	Writes the CSV file to STORAGE_DIRECTORY_DOCUMENTS if it does not exist and appends the given char
int cp(const char *to, const char *from)	Copies the file from *from to *to

view.c

Method name	Brief description
void view_create(void)	Create Essential Object window, conformant and layout
Evas_Object * view_create_win(const char *pkg_name)	Standard creates view and adds settings from Tizen
Eina_Bool bezelRotationHandler(void *data, Eext_Rotary_Event_Info *ev)	Handles the BezelRotation. It calls the next or previous User
Evas_Object * view_create_conformant_without_indicator (Evas_Object *win)	Standard „Make a conformant without indicator for wearable app param[in] win The object to which you want to set this conformant Conformant is mandatory for base GUI to have proper size“ from Tizen
void view_dialer_create(const char *file_path)	Standard „Make essential object for this app , like conformant and layout param[in] file_path File path of EDJ file will be used“ from Tizen
static void _win_delete_request_cb(void *data, Evas_Object *obj, void *event_info)	Function will be operated when window is deleted and then closes the app
Evas_Object * view_create_layout (Evas_Object *parent, const char *file_path, const char *group_name, Eext_Event_Cb cb_function, void *user_data)	Standard „Make a layout to target parent object with edje file param[in] parent The object to which you want to add this layout param[in] file_path File path of EDJ file will be used param[in] group_name Name of group in EDJ you want to set to param[in] cb_function The function will be called when back event is detected param[in] user_data The user data to be passed to the callback functions“ from Tizen

Evas_Object *view_create_layout_for_conformant (Evas_Object *parent, const char *file_path, const char *group_name, Eext_Event_Cb cb_function, void *user_data)	Standard „Make and set a layout to <u>conformant</u> <u>param[in]</u> parent Target <u>conformant</u> object <u>param[in]</u> file_path File path of EDJ will be used <u>param[in]</u> group_name Group name in EDJ you want to set to layout <u>param[in]</u> cb_function The function will be called when the back event is detected <u>param[in]</u> user_data The user data to be passed to the callback functions“ from Tizen
Evas_Object *view_create_layout_by_theme (Evas_Object *parent, const char *class_name, const char *group_name, const char *style)	Standard „Make a layout with theme. <u>param[in]</u> parent Object to which you want to add this layout <u>param[in]</u> class_name The class of the group <u>param[in]</u> group_name Group name in EDJ that you want to set to layout <u>param[in]</u> style The style to use“ from Tizen
void view_destroy(void)	Destroy window and free important data to finish this application
void view_set_image (Evas_Object *parent, const char *part_name, const char *image_path)	Standard „Set image to given part <u>param[in]</u> parent Object has part to which you want to set this image <u>param[in]</u> part_name Part name to which you want to set this image <u>param[in]</u> image_path Path of the image file“ from Tizen
void view_set_text (Evas_Object *parent, const char *part_name, const char *text)	Standard „Set text to the part <u>param[in]</u> parent Object has part to which you want to set text <u>param[in]</u> part_name Part name to which you want to set the text <u>param[in]</u> text Text you want to set to the part“ from Tizen

void view_set_color (Evas_Object *parent, const char *part_name, int r, int g, int b, int a)	<p>Standard „Set color of the part</p> <p><u>param</u>[in] parent Object has part to which you want to set color</p> <p><u>param</u>[in] part_name Name of part to which you want to set color</p> <p><u>param</u>[in] r R of RGBA you want to set to the part</p> <p><u>param</u>[in] g G of RGBA you want to set to the part</p> <p><u>param</u>[in] b B of RGBA you want to set to the part</p> <p><u>param</u>[in] a A of RGBA you want to set to the part“ from Tizen</p>
void view_set_button (Evas_Object *parent, const char *part_name, const char *style, const char *image_path, const char *text, Evas_Object_Event_Cb down_cb, Evas_Object_Event_Cb up_cb, Evas_Smart_Cb clicked_cb, void *user_data)	<p>Standard „Make and set button.</p> <p><u>param</u>[in] parent Object to which you want to set the button</p> <p><u>param</u>[in] part_name Name of part to which you want to set the button</p> <p><u>param</u>[in] style Style of the button</p> <p><u>param</u>[in] image_path Path of image file will be used as button icon</p> <p><u>param</u>[in] text The text will be written on the button</p> <p><u>param</u>[in] down_cb Function will be operated when the button is pressed</p> <p><u>param</u>[in] up_cb Function will be operated when the button is released</p> <p><u>param</u>[in] clicked_cb Function will be operated when the button is clicked</p> <p><u>param</u>[in] user_data Data passed to the 'clicked_cb' function“ from Tizen</p>
Evas_Object * view_dialer_create_rectangle ()	<p>Standard „Make Rectangle Object to target window for hijacking touch event</p> <p>Add callback function will be operated when mouse down/up event is triggered“ from Tizen</p>
void view_dialer_set_entry (const char *part_name)	<p>Standard „brief Create entry object and keep the object for handling.</p> <p><u>param</u>[in] part_name Part of the layout which you want to locate Entry“ from Tizen</p>

<p>Evas_Object *view_create_entry(Evas_Object *parent, const char *part_name, Evas_Smart_Cb clicked_cb, void *user_data)</p>	<p>Standard „Make an Entry Object to target window</p> <p><u>param</u>[in] parent Object to which you want to set Entry</p> <p><u>param</u>[in] part_name Part of the layout which you want to locate Entry</p> <p><u>param</u>[in] clicked_cb The function will be called when the entry is clicked</p> <p><u>param</u>[in] user_data The user data passed to the callback function</p> <p>Add callback function will be operated when mouse clicked event is triggered“ from Tizen</p>
<p>int view_dialer_set_entry_text(int operation, const char *text)</p>	<p>Standard „Modify entry text of Entry object</p> <p>* @<u>param</u>[in] operation ENTRY_TEXT_CLEAR_ALL - clear all text, ENTRY_TEXT_ADD_TEXT - add specific text to current text, ENTRY_TEXT_BACKSPACE - delete one character by backspace</p> <p>* @<u>param</u>[in] text target text which will be added to current entry text“ from Tizen</p>
<p>static int _get_emotion(int x, int y)</p>	<p>Returns the corresponding emotion of user's touch event position by calculating distance and slope from center</p> <p><u>param</u>[in] evt_x X-coordinate value of event position</p> <p><u>param</u>[in] evt_y Y-coordinate value of event position</p>
<p>static void _rectangle_mouse_move_cb(void *data, Evas *e, Evas_Object *obj, void *event_info)</p>	<p>Function will be operated when mouse move event is triggered.</p> <p><u>param</u>[in] data The data to be passed to the callback function</p> <p><u>param</u>[in] e The handle to an <u>Evas</u> canvas to be passed to the callback function</p> <p><u>param</u>[in] <u>obj</u> The <u>Evas</u> object handle to be passed to the callback function</p> <p><u>param</u>[in] event_info The system event information</p>

static void _rectangle_mouse_down_cb(void *data, Evas *e, Evas_Object *obj, void *event_info)	<p>Function will be operated when mouse down event is triggered. The function gets the current location and its corresponding emotion and writes the result with the timestamp in the Log</p> <p><u>param[in]</u> data The data to be passed to the callback function</p> <p><u>param[in]</u> e The handle to an <u>Evas</u> canvas to be passed to the callback function</p> <p><u>param[in]</u> obj The <u>Evas</u> object handle to be passed to the callback function</p> <p><u>param[in]</u> event_info The system event information</p>
static void _rectangle_mouse_up_cb(void *data, Evas *e, Evas_Object *obj, void *event_info)	<p>Standard „Function will be operated when mouse up event is triggered</p> <p><u>param[in]</u> data The data to be passed to the callback function</p> <p><u>param[in]</u> e The handle to an <u>Evas</u> canvas to be passed to the callback function</p> <p><u>param[in]</u> obj The <u>Evas</u> object handle to be passed to the callback function</p> <p><u>param[in]</u> event_info The system event information“ from Tizen</p>
static void _dialer_layout_cb(void *data, Evas_Object *obj, void *event_info)	<p>Standard „Layout back key event callback function</p> <p><u>param[in]</u> data The data to be passed to the callback function</p> <p><u>param[in]</u> obj The <u>Evas</u> object handle to be passed to the callback function</p> <p><u>param[in]</u> event_info The system event information“ from Tizen</p>
static void _dialer_text_resize(Evas_Object *entry)	<p>Standard „Resize dialer entry text for long text</p> <p><u>param[in]</u> entry Entry object to be modified by this function“ from Tizen</p>
Evas_Object * view_dialer_get_layout_object(void)	<p>Standard „This function just return static layout object.“ from Tizen</p>
static void get_current_time(char* buffer)	<p>This function gets the current time and formats it to be used in the Log. It writes the time in the given buffer.</p>

proxemo.c

Method name	Brief description
static bool app_create (void *user_data)	Hook to take necessary actions before main event loop starts it creates the view Checks for necessary Folders and loads the User Pictures
static void app_control (app_control_h app_control, void *user_data)	Standard „This callback function is called when another application sends the launch request to the application <u>param[in]</u> app_control The handle to the app_control <u>param[in]</u> user_data The user data to be passed to the callback function“ from Tizen
static void app_pause (void *user_data)	Standard „This callback function is called each time the application is completely obscured by another application and becomes invisible to the user <u>param[in]</u> user_data The user data to be passed to the callback function “ from Tizen
static void app_resume (void *user_data)	Standard „This callback function is called each time the application becomes visible to the user <u>param[in]</u> user_data The user data to be passed to the callback function“ from Tizen
static void app_terminate (void *user_data)	This callback function is called once after the main loop of the application exits <u>param[in]</u> user_data The user data to be passed to the callback function
static void ui_app_lang_changed (app_event_info_h event_info, void *user_data)	Standard „This function will be called when the language is changed <u>param[in]</u> event_info The system event information <u>param[in]</u> user_data The user data to be passed to the callback function“ from Tizen
int main (int argc, char *argv[])	Standard „Main function of the application <u>param[in]</u> <u>argc</u> The argument count <u>param[in]</u> <u>argv</u> The argument vector“ from Tizen