CONFIGURATION FILE

Referenced file: _eva_actions.json

Table of Contents

v1.1 12/3/2018	New value added to Jocr Action CLEAR_VARIABLES added
v1.2 29/8/2018	Jaction – waitForEventId param added
v1.3 31/8/2018	Jmain params added (version, createdDate, modifiedDate, statusEnabled, statusPosX, statusPosY) Jocr params added (hideArea, hideColor)
V1.4 14/3/2019	New Jdatabase, JhideArea, navigationBarEnabled (Zebra only), notificationPullDownEnabled (Zebra only) Various bug fixes
V1.5 23/4/2019	New Jscreen.important param New JCondition.orTextContains param New field “fromObject” inset_GLOBAL_VARIABLE
V1.6 16/5/2019	Changes in params: Jscreen.important, previousScreen, Jaction.action, Jaction.waitForObject, Jaction.checkObject, Jocr.retryWhenBlank
V1.7 17/5/2019	Jaction.action - STOP_EVA added

Defining variables

When defining variables the name *MUST* be in the following format

```
{{name}}
```

By not using the double braces you could receive unexpected values.

There are two types of variables, global and local. Global variables can be initialized in the beginning and be used throughout the whole lifetime of the service, whereas local variables are only available on the current screen.

If a local variable exists with the very same name as a global variable. The local variable will be used.
Prohibited names are "**{{lastAsr}}**", "**{{lastScan}}**". They cannot be changed nor cleared unless you are executing a script.
Name you can never change are "**{{currentScreen}}**", "**{{previousScreen}}**". These variables let you know from what trigger you came and which one currently active.

Types

int	Defaults to 0
Integer	Defaults to null
boolean	Defaults to false
Boolean	Defaults to null
String	Defaults to null (empty strings are treated as null)
double	Defaults to 0.0

It's very important to understand the difference between "int", "Integer", "boolean" and "Boolean"

By not defining an object in the JSON configuration file, that's declared internally in Java as "int", the variable will be initialized to a value of zero (0), whereas if it would have been declared as an Integer the value would be null.

This is important as you will see later in the "condition" part of the JSON. When making up a series of conditions there will be conditions that you do not wanna compare. So when the value is null, the comparison will not be made.

Example:

```
{
	"isVisible": null,
	"hasFocus": true
}
```

This would only check the value "hasFocus" but ignore the value of "isVisible", in fact, the "isVisible" doesn't even have to be declared here and could be left out completely.

_eva_actions.json

Main

description	String	A description about the project
position	int	All x/y positions are saved in an array, in that way you will have the possibility to have the same json file for various types of phones. Position is, in other words, the array index to be used when deciding what item to get.
homeEnabled	boolean	True, if to allow the user to press the **hardware** Home button. True for softkey HOME on Zebra devices with the correct EMDK version installed.
backEnabled	boolean	True, if to allow the user to press the **hardware** Back button
menuEnabled	boolean	True, if to allow the user to press the **hardware** Menu button. True for softkey RECENT on Zebra devices with the correct EMDK version installed.
navigationBarEnabled	boolean	True if to hide the navigation bar on Zebra devices with the correct EMDK version installed.
notificationPullDownEnabled	boolean	True if not to let user pull down the notification bar on Zebra devices with the correct EMDK version installed.
unlockEnabled	boolean	True, if to allow the user to lock/unlock the screen
statusEnabled	boolean	True if to show status of EVA Player. There are three different statuses: • Red – Eva Player is inactive (i.e. not searching for new screens) • Blue – Eva Player is searching for a new screen • Green – Eva Player is locked on a screen and waiting for user input. This option if not defined defaults to: true
statusPosX	Integer	X position on screen where to put status image. Defaults to 1.
statusPosY	Integer	Y position on screen where to put status image. Defaults to 67.

varAssigns	HashMap <String>,<String>	Initialize EVA Player with variables.
		
		Parameters: <name>, <value>
		Example:
		"varAssigns": {
		"{{support_num}}": "131517",
		"{{username}}": "hubone"
		}

		All variables defined here will be added as global variables.
databases	Jdatabase[]	Array of databases to use with scripts
screens	HashMap <String, Jscreen>	All available screens to pick up and make voice aware. Parameters: <screen name>, <Jscreen object> The name should not contain spaces and/or special characters.
version	String	Version of actions json
createdDate	String	Date when json was first created
modifiedDate	String	Date when json was last modified

Jdatabase

Load a two- column file into memory for quick access from a JavaScript.

name	String	Name of database. This is the name to use from a JavaScript
file	String	The name of the file. The file needs to be in the profile folder under a folder called “db”
type	String	What type of file it is. At the moment, the only valid type is “csv”
separator	String	How the columns separators are, usually by a “,”

Jscreen

The Jscreen object is where you define all the important stuff. This is where you define your conditions to match with the active screen on the device. If all conditions are matched a series of events will be triggered automatically. These events could be any from do everything needed to be done and continue to a new screen, or to wait for user input by voice and/or scanner.

description	String	Screen description
enabled	boolean	"true" if to enable this trigger
important	boolean	If set to true, this screen is allowed to be found even though EVA Player has another screen locked and waiting for user input. Be careful not to overuse this option as it could slow down your device. When a screen is found, it will be initialized and executed, thus canceling any other active screen. The “previousScreen” option will be ignored.
friendlyName	String	Friendly screen name
package	String	The complete package name to watch out for. The trigger will be ignored if "null" or empty.
eventIds	int[]	Multiple event ids are allowed for each screen.
previousScreen	String	This parameter uses the name as defined in the "screens" HashMap in the main part of the JSON. For this trigger to become valid, the value must correspond to the last accepted screen. Negative values are allowed by adding a exclamation mark (!) before the name. Then all previous screens except this one will be allowed to activate this trigger. Multiple values are allowed, separated with comma (,) You can prohibit that the screen is valid if “previousScreen” should be blank with a “!” character. This would be true when starting the service or the user unlocks/lock the screen.
conditions	Array of Jconditions	Array of conditions that has to be met in order to activate this trigger. The screen will be ignored if null or empty array.
initialization	Jinitialization	This object will initialize variables, lock/unlock screen, activate/deactivate scanner etc...
input	Jinput	This will trigger the ASR engine telling the user an action that needs to be done and wait for a response. It will only be triggered if there are no "autoActions" (see below) Null allowed. See Jinput for more information.
autoActions	Array of Jaction	If there is at least one action here, it will be executed and then stop and do nothing. "input" (see above) will be ignored completely.

exitHideArea	JhideArea	Hide an area from the user when exiting the screen. This is in case there is data on the screen that the user should not see, this would block it before Eva Player finds the next screen.

Jcondition

This object is used to find an element on the screen received from the OS. Not all data should be validated always, as this would slow down everything on slower devices (even faster devices shows problems).

id	int	A number. Used to identify the condition in the Logcat for debugging purpose
description	String	Condition description
class	String	Name of the class (i.e. android.widget.ImageView)
resource	String	The preferred way of searching for an object. The resource id is usually unique (not always though, be careful, lists can be tricky)
text	String	Must match this string exactly.
orText	String[]	Array of strings, if any of them is found this condition is considered fulfilled.
notText	String	Must NOT match this string. The match is exact. Used for example to make sure no TextView would contain a certain string, that could make the difference between two screens very much alike.
textContains	String	String must have this text in it.
orTextContains	String[]	Array of strings. If any of them is found, this condition is considered fulfilled.
posX	Array of Double	Center x position of element Make sure that there are as many positions available as defined in "position" of the main element
posY	Array of Double	Center y position of element Make sure that there are as many positions available as defined in "position" of the main element
fromOCR	Jocr	A Jocr object
isEditable	Boolean	Possible values: null, true or false
isEnabled	Boolean	Possible values: null, true or false
isPassword	Boolean	Possible values: null, true or false
isCheckable	Boolean	Possible values: null, true or false
isChecked	Boolean	Possible values: null, true or false
isFocused	Boolean	Possible values: null, true or false
isVisibleToUser	Boolean	Possible values: null, true or false
invalidObject	Boolean	If true and condition is met, condition failed If true and condition is NOT met, condition ok
doIf	String	Example "{{test}}=hello" This condition would compare the variable {{test}} with the String “hello”. The condition would only return true if the doIf is considered being correct. Numbers will be compared as numbers and strings as strings. Valid conditions: =, !=, >, <, >=, <=

Jaction

Do something.

id	int	A number. Used to identify the action in the Logcat for debugging purpose
description	String	Action description
delay	int	How many milliseconds to wait before executing this action. For example, you were to click two buttons, you could not do it too quickly, this delay makes sure to fire the action at the right moment.
waitToFinish	boolean	This variable is only used with the SPEECH_SAY action. If true it will wait for the text to be said, then continue with the next action in line, or you could have it say something while you are executing other actions simultaneously.
action	String	**SET_TEXT** Insert new text into an EditText object **SET_FOCUS** Change focus to this object **PERFORM_CLICK** Click this object (the preferred way if possible) **GESTURE_CLICK** This will be click on a certain position on the screen exactly as if it where the user pressing the screen with his/her finger. This causes problems with the lockScreen option (see Jinitialization for more information) Use fields “posX” and “posY” to define where to click. (Unlocks screen) **SPEECH_SAY** Say something using the TTS engine. **SPEECH_LISTEN** Send a <LISTEN> command to the VoiXtreme engine. **SWIPE_SCREEN_LEFT_TO_RIGHT** Change screen in a ViewPager (go left) (Unlocks screen) **SWIPE_SCREEN_RIGHT_TO_LEFT** Change screen in a ViewPager (go right) (Unlocks screen) **FINGER_GESTURE** Do a finger gesture (like a swipe for example) Use field “gesturePos”

		(Unlocks screen) **PRESS_HOME_BUTTON** Press the home button as if the user had done it. **PRESS_BACK_BUTTON** Press the back button as if the user had done it. **CLEAR_VARIABLES** In the field “text” you put your variable names (comma separated) you need cleared. **SET_GLOBAL_VARIABLE** DEPRICATED **SET_VARIABLE** field “text” = variable name field “value” = variable value field “isGlobalVariable” = true / false field “fromObject” = value from screen (fromObject is optional. If specified you need to add “%s” into the value field or the result will be lost) **START_OVER** Will force the engine to search for a screen field “disallowSameScreen” = true / false – Will allow EVA to find the current screen again or not. **SHUT_DOWN** Disable the Eva service, it will no longer check for screens and cleans up (it does not terminate) To restart you need to select a profile from EVA Selector. **STOP_EVA** This will unlock the terminal and EVA will not do anything more until the user presses the lock button. **UNLOCK_SCREEN** If you decide to stay on the screen but has terminated all other actions, this would unlock the screen for the user to do as he/she will. **RESET_CURRENT_SCREEN** Eva does not allow the finding of the same screen its currently at. This action will allow Eva Player engine to re-connect to the same screen. This is useful when only a part of the screen is changing, like switching article and the description changes.
text	String	Use this text with SET_TEXT and SPEECH_SAY. Can be null if "fromObject" or "inputConditions" are being used.

onObject	Jcondition	The target object for the following actions: Used with actions SET_TEXT, PERFOM_CLICK, GESTURE_CLICK and SET_FOCUS.
fromObject	Jcondition	SET_TEXT: Get the text from the following object. If "text" has any value, the fromObjects text will be appended at the end of the text string. This will change in future versions.
input	Array of Jcondition	SPEECH_SAY: Will concatenate all objects texts into one string. If "text" has a value, the concatenated strings will be appended to the end of the text. This will change in future versions.
doIf	String	Example "{{test}}=hello" This action would compare the variable {{test}} with the String “hello”. The action would only be executed if the condition returns true. Numbers will be compared as numbers and strings as strings. Valid conditions: =, !=, >, <, >=, <=
posX	Float[]	Used with GESTURE_CLICK
posY	Float[]	Used with GESTURE_CLICK
gesturePos	Float[][]	Used with FINGER_GESTURE
value	String	Used with SET_GLOBAL_VARIABLE
waitForEventId	Integer	Wait for an event before executing the event. This lets you do actions on screens that might come after the current one
waitForObject	Jcondition	Wait for an object to be present on the screen before executing the requested action.
checkObject	Jcondition	This will check if an object is available or not on the screen. Would be used normaly together with SET_VARIABLE. If the object is found, the string “%c%” will be replaced with a “1”, or a “0” if not found. This could later be used with doIf to determine what to do on the screen.

Jinitialization

- Initialize variables
- Perform any actions needed to be done before autoActions of Input
- Set objects that are allowed to be clicked on a locked screen
- Enable/disable scanner
- Lock/unlock screen
- Initialize the ASR command

varAssigns	Array of Jvariable	Set global or local variables (See Jvariable for more information)
script	String	Executes a JavaScript after “varAssigns” and before anything other.
bgScript	String	Run a javascript after variables that were set in a background thread have been initialized
preActions	Array of Jactions	If focus has to be set, or any other pre action is needed before you start the ASR engine, this is the place to add them. The actions here are to be executed before any "autoActions" or "input"
allowClick	Array of Jcondition	If you lock the screen but still want to allow the user to be able to make click on a certain button. The objects found meeting the conditions in this array will be clickable. NOTE: The reset (<RST>) command will be sent to the VoiXtreme engine when the button is clicked.
scanner	int	0 = Do nothing 1 = Activate scanner 2 = Deactivate scanner
multiScan	boolean	True if screen allows more than one scan
lockScreen	boolean	True or false to lock/unlock screen
asr	Jasr	"input" - Say something and wait for a user response. This will initialize the ASR engine.

Jasr

Initialize an ASR "input" command.

on	boolean	Enable/disable this ASR command

values	HashMap <String>,<String>	<command>, <value>
		The parameters to send to the ASR engine. The strings will be concatenated in the very same order as they appear in the JSON.
		
		The command will be converted to uppercase.
		
		Example:
		"values": {
		"asr": "X",
		"min": "0",
		"max": "0",
		"info": "Say hello {{user}}",
		"prompt": "Hello {{user}}"
		}
		
		Result:
		<ASR>X<MIN>0<MAX>0<INFO> etc...

Jvariable

Initialize a variable

name	String	The name of the variable. Cannot be null nor empty or it will be ignored. The name must use double braces to enclose it or strange results might be given. Example: {{username}}
value	String	Null is not allowed and variable will not be set. If fromObject is used, %s must be added somewhere in the string where it will be replace with the objects text. You could even set a variable using another variables value. Example: Hello {{username}}. Please take %s packages.
background	Boolean	Should this variable be read after the Vocal Engine has been engaged? This would save some time on pages with many OCR scans and create quicker access to screens
mainThread	Boolean	Ignored if background is used. Some screens don't do well with threads setting variable values. This forces the variable to be set on the main thread making sure it will always work
local	Boolean	If not set defaults to true.
strFormat	JstringFormat	How to parse the string before setting the variable
fromObject	Jcondition	Get the text from an object.
fromOCR	Jocr	A Jocr object

Jocr

Read text from the screen.

cropArea	Integer[][]	Crop the part of interest from the full screen image [[x, y, width, height]]
hideArea	JhideArea	A screenshot will be taken before executing this command so we are able to read the below text before hiding the area
fillAreaInverse	Integer[][]	Fill all with WHITE except within the rectangle specified. The rectangle within will be converted to black and white colors only. [[x, y, width, height]]
merge	Integer[][][]	Copy parts from an image and put them together (i.e. "R031 E 03 01" would become: "R031 E 03 01")
engineOCR	int	0 – Android Vision OCR Engine (default) 1 – Tesseract (to be implemented)
toBlackWhite	boolean	Convert image to black and white, some texts are difficult and need this in order to be interpreted correctly.
filterColor	String	Defaults to “#555555”. If specified, goes together with fillAreaInverse and toBlackWhite. Any color bigger than this color will be considered white and all below as black.
retryWhenBlank	int	If a value is blank, that could mean something went wrong (image or OCR engine fault). If this value is greater than 0, the image will be erased and a new one will be captured “requested” times. 10 times would correspond to 1 second.
enlargeBitmap	boolean	True if to double the size of the OCR bitmap. This helps the OCR engine to understand the text better in case of a difficult text.
enlargeWidth	float	How much to enlarge the width. (enlargeBitmap must be true)
enlargeHeight	float	How much to enlarge the height. (enlargeBitmap must be true)
createLocalVariable	String	This value is ONLY valid when inside a Jcondition object. When checking if the condition is true, we save the value as a local variable. If is only necessary if you would like to use this value later on without having to re-run the OCR engine to re-read the same value.

JhideArea

Hide an area from the screen. This is done by adding a window with a specific color on top of the specified area.

hideArea	Integer[][]	Draw over this area, thus hiding the text from the user
hideColor	String	Color to use when hiding an area. Default value set to #000000 (black)

JstringFormat

How to extract a value from a string. This is very simple for now, yet to be implemented would be trim leading zeros for example...

from	int	Copy from (start position is 0).
length	int	Copy this number of characters. If = 0 and fromEnd = 0 copy to end of string If = 0 and fromEnd <> 0 copy to beginning of string
fromEnd	int	Must be zero if to copy from beginning of string. If not zero, "from" will be completely ignored. If < 0 "length" will also be ignored. -2 would copy the last two characters of the string if > 0 Like "from" but from the end of string. "length" will be used to determine how many characters to copy.
trimLeadingZeroes	boolean	True if to remove leading zeroes from a number
regExReplace	String[]	Array with the fixed length of 2. First value = regEx Second value = what to replace it with,

Examples:
String: Hello World
{ "from": 0, "length": 5 }
Result: **Hello**

String: Hello World
{ "from": 6 }
Result: **World**

String: Hello World
{ "fromEnd": -5 }
Result: **World**

String: Hello World
{ "fromEnd": 6, "length": 4 }
Result: **ello**

Jinput

What to do when receiving an ASR or scanner event.

asr	HashMap <String>,<Array of Jaction>	What action/s to do when receiving user input via voice.
bc	HashMap <String>,<Array of Jaction>	What action/s to do when receiving scanned data.
scriptASR	String	Execute a JavaScript after receiving voice input from the VoiXtreme Engine
scriptBC	String	Execute a JavaScript after receiving data from a scanner.

Both "asr" and "bc" works in the same way. Each key element in the HashMap is a command or keyword. If other values than keywords is accepted, the last element should have an empty key, failing to put the empty key at the end will result in ignoring the keywords after it.

Example:

```
"asr": {
 "return": [{
    "action": "GESTURE_CLICK",
    "onObject": {
     "class": "android.widget.ImageView",
     "posX": [ 21.50, 21.50 ],
     "isEditable": false
    }}],
 "set_support": [{
    "action": "SET_TEXT",
    "text": "{{support_nr}}",
    "onObject": {
     "class": "android.widget.EditText"
    }}],
 "": [{
    "action": "SET_TEXT",
    "text": "{{lastAsr}}",
    "onObject": {
     "class": "android.widget.EditText"
    }}]
}
```

If the word "return" was found, we would do a GESTURE_CLICK on an ImageView, though if the user would have said "set support" we change the text in an EditText to a value from some variable, whereas if it isn't any keyword at all, we set the EditText to the value the user said.