Chinese IME Tutorial

Year 2012

Version 1.00

Internet@TV

1. Overview 3
2. Creating the Basic Widget 5
3. Basic Text Entry using the IME 8
4. Register Anykey callback function 11
5. Register keypad changed callback function 12
6. Register Enter and other keypress event callback function 13
7. Register other callback function 14
8. Register Control functions 15
9. IME Function and Value Reference 16

1. Overview

This tutorial will show the steps needed to add text input capability to a widget, using the IME featu re of the widget manager. This will be done by creating a simple widget with a text input box and a password input box, and enabling user input with a virtual keyboard display. This widget will also d emonstrate the optional notifications available from the IME feature, by displaying various status m essages.

Picture 1-1 (12key keypad)

Picture 1-2 (qwerty keypad)

2. Creating the Basic Widget

Start the SDK for Samsung TV widgets:

Add index.html in IMETotorial project, with contents as follows:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>IME Test widget</title>
<style type="text/css">
body{
	margin:0;
}
#item {
	position: absolute;
	left: 20px;
	top: 210px;
	padding:0 6px;
	font-size: 20pt;
	text-decoration: none;
	background:#809FFF;
}

#item2 {
	position:absolute;
	left:20px;
	top:440px;
	background:#809FFF;
}

#search {
	position: absolute;
	padding:5px;
	left: 40px;
	top: 350px;
	width: 375px;
	height: 70px;
	background:#E1E1E1;

}

#search div{
	float: left;
	font-size: 15pt;
	text-align:right;
	line-height:135%;
	border-color :skyblur;
	border-width: 1px;
	border-style:solid;
}

#searchText1, #searchText2{
	font-size: 12pt;

}

#resultBox{
	position: absolute;
	left: 20px;
	width:100px;
	height:60px;
	background:#FF99CC;
```

```
        top:470px;
}

</style>
<script type="text/javascript" src="$MANAGER_WIDGET/Common/OpenSrc/jquery-
1.4.2.min.js"></script>
<script type="text/javascript"
src="$MANAGER_WIDGET/Common/Plugin/Define.js"></script>
<script type="text/javascript"
src="$MANAGER_WIDGET/Common/API/TVKeyValue.js"></script>
<script type="text/javascript" src="$MANAGER_WIDGET/Common/API/Widget.js"></script>
<script type="text/javascript" src="$MANAGER_WIDGET/Common/API/Plugin.js"></script>
</head>
<body onLoad="func_onLoad()">

<script type="text/javascript" src="ime_sample.js"></script>
<script type="text/javascript" src="$MANAGER_WIDGET/Common/IME_cn/ime.js"></script>

        <a id='item' href="javascript:" onkeydown=normalobjKeyFunc(this);
onfocus=itemFocus(this); onblur=itemBlur(this);>
                Refresh
        </a>

        <div id="search">
                <table width="100%" height="100%" border="0px" cellpadding="0px"
cellspacing="0px">
                        <tr>
                                <td>Search:</td>
                                <td>
                                        <input size=30 type="text" id="searchText1">
                                        </td>

                        </tr>
                        <tr>
                                <td>Password:</td>
                                <td>
                                        <input size=30 type="password"
id="searchText2">
                                </td>
                        </tr>
                </table>
        </div>


        <a id='item2' href="javascript:" onkeydown=normalobjKeyFunc(this);
onfocus=itemFocus(this); onblur=itemBlur(this); onclick=normalobjKeyFunc(this)>
        SetString Test
        </a>
        <div id="resultBox"></div>

</body>
</html>
```

If you have unzipped the provided files, the widget should already have an HTML index page.

Add ime_sample.js in IMETotorial project, with contents as follows

```
/**
 * Sample to use IME module for Samsung TV Widget
 */
```

```
//Variable to save an IME object
var ime = null;
var ime2 = null;

/**
 * On load function
 */

function func_onLoad(){
	alert("func_onLoad begin...");

	keyc = new Common.API.TVKeyValue();
	widgetAPI = new Common.API.Widget();
	widgetAPI.sendReadyEvent();
	var pluginAPI = new Common.API.Plugin();
	pluginAPI.registAllKey();

	//Only support in 2012th TV
	//_g_ime.pluginMouse_use_YN = true;
	//_g_ime.dimm_use_YN = false;
	//_g_ime.Recog_use_YN = true;
```

Now, start the SDK emulator. If you see the message 'alert() : func_onLoad begin…' in the log manager, that means you have successfully created the widget. You should be able to see the provided layout on the screen, consisting of several text boxes.

You should also be able to view this on the TV with the User Widget upload feature.

Important, sendReadyEvent(), registAllkey() functions,

This functions registering for the widget to event and handle all keys used by the IME

```
1.widgetAPI.sendReadyEvent(); //event send function for widget
2.pluginAPI.registAllKey();  //key registe funciton for widget
```

3. Basic Text Entry using the IME

In this section we will add the IME feature to the widget, which will enable us to enter text in the input boxes.

Add the following class definition to ime_sample.js:

```
function func_onLoad(){
        alert("func_onLoad begin...");

        keyc = new Common.API.TVKeyValue();
        widgetAPI = new Common.API.Widget();
        widgetAPI.sendReadyEvent();
        var pluginAPI = new Common.API.Plugin();
//      pluginAPI.registIMEKey();
        pluginAPI.registAllKey();

        //Only support in 2012th TV
        //_g_ime.pluginMouse_use_YN = true;
        //_g_ime.dimm_use_YN = false;
        //_g_ime.Recog_use_YN = true;

        ime = new IMECNShell("searchText1", ime_init_text,this);
        if(!ime){
                alert("object for IMECNShell create failed", 3);
        }

        ime2 = new IMECNShell("searchText2", ime_init_passwd,this);
        if(!ime2){
                alert("object for IMEShell create failed", 3);
        }

}
```

This class creates an IMECNShell object for the specified HTML input element.

```
ime = new IMECNShell (‘inputbox id’,callback function, this);

parameters : inputbox id,
             callback function,
              this.content (option)
```

Note that the IME feature supports HTML <input> elements only; other kinds of element are not supported. The callback function ime_init_text is called when the IME object has been fully created. Creation of an IME object is asynchronous, and no IME methods should be called until the callback has been received. In the callback, we notify the index.html that this HTML input element is ready.

Add the following to the definition of callback function for searchText1

```
 function ime_init_text(imeobj){

        //if you need input object, use "getInputObj" function
        var inputobj = imeobj.getInputObj();

        alert("start initializing : "+inputobj.id);

        //option : set function that keypad/qwerty select on IME start, default is
'qwerty'
        //imeobj.setKeySetFunc('qwerty');
        //imeobj.setKeySetFunc('12key');
```

```
        //option : keypad x,y position, IME XT9 default right alignment
        //imeobj.setKeypadPos(1355, 159); //1080p
        //imeobj.setQWERTYPos(860, 159);

        imeobj.setKeypadPos(780, 150); //720p
        imeobj.setQWERTYPos(570, 150);

        //imeobj.setKeypadPos(427, 80); //540p
        //imeobj.setQWERTYPos(430, 80);              //IME XT9, new function
        //imeobj.setWordBoxPos(150,15);      //IME XT9, this function do not use more

        //option : set function that runs after all key input (pass array as arguments)
        //when IME runs the function, first argument is Keycode, second argument will be
the first element of the array
        //imeobj.setAnyKeyFunc(testf2);

        // option : set function that callwhen left key pressed on cursor position is at
the beginning of input box
//      imeobj.setPrevEventOnLeftFunc(testf1);

        // option : set function that called when letter is completed
        imeobj.setOnCompleteFunc(onC);

        // option : set function that called when enter key is pressed.
//      imeobj.setEnterFunc(sampleEnter);

        // option : set function that called when input reached maxlength
//      imeobj.setMaxLengthFunc(sampleMaxLength);

        //option : 키코드에 새로운 함수를 연결하고 싶을때.. ex) red key , blue key
        //if fuctions is registered, keycode will be a parameter when the function is
called.
        //if the function returns true, IME continues
        //if the function returns false, IME returns
        imeobj.setKeyFunc(88, textobjKeyFunc);
        imeobj.setKeyFunc(keyc.KEY_UP, textobjKeyFunc);
        imeobj.setKeyFunc(keyc.KEY_DOWN, textobjKeyFunc);
        imeobj.setKeyFunc(keyc.KEY_RETURN, closeIME);
//      imeobj.setBlockSpace(true);
//      imeobj.setKeyFunc(keyc.KEY_5, testf1);


//      imeobj.setKeyFunc(keyc.KEY_YELLOW, function(){});
//      imeobj.setKeyFunc(keyc.KEY_RIGHT, function(){});

        // to unregister key Function assign null function to keycode
//      imeobj.setKeyFunc(5, function(){});

        //option : when keypad/qwerty changed event callback function,for input object
x,y position change
        imeobj.setKeypadChangeFunc('12key',move12KeyPosition);
        imeobj.setKeypadChangeFunc('qwerty',moveqwertyPosition);

        //option : when inputbox highlight event callback function,for input object focus
        imeobj.setInputHighlightFunc(inputHighlight);

        //get IME version : return string is 'XT9', 'T9'
        alert("[IME] ========================================
imeobj.IsSupportXT9:"+imeobj.IsSupportXT9);

        //get IME mode : return string is '12key', 'qwerty'
        var strKeySet = imeobj.getKeySet();
        alert("[IME] ========================================== strKeySet:"+strKeySet);
```

```
        //document.getElementById("item").focus();
        //document.getElementById("item2").focus();

        if(_g_ime_cn.pluginMouse_use_YN){
                imeobj._focus();
                document.getElementById('searchText1').focus();
        }
        else{
                document.getElementById('searchText1').focus();
        }

        /*
        var ret;
        if(ret = imeobj.setMode("_num")){
                alert("ret : true "+ret);
        }
        */

        alert("ime_init end...")
```

Now we will create the Input objects, and register to handle the keys needed for the IME to work correctly.

Add the following to the definition of callback function for searchText2, with password type.

```
function ime_init_passwd(imeobj){

        var inputobj = imeobj.getInputObj();

        alert("start initializing : "+inputobj.id);

        //imeobj.setKeypadPos(370,40);

        imeobj.setKeyFunc(keyc.KEY_UP, passwdobjKeyFunc);
        imeobj.setKeyFunc(keyc.KEY_DOWN, passwdobjKeyFunc);

        //document.getElementById("searchText2").focus();

        alert("ime_init_passwd end...")
```

4. Register Anykey callback function

And add the following code to complete ime_ sample.js:

```
//2nd dummy function for testing
function testf2(a,b,c) {
        alert("------------- test2 ----------------");
        alert("-------- args1: "+ a +" ---------");
        alert("-------------------------------------");
```

1. setAnyKeyFunc – the specified callback function will be called each time a key is pressed, no matter what the key.

Parameter : key code of key press event.

5. Register keypad changed callback function

And add the following code to complete ime_sample.js:

```
function move12KeyPosition(arg){
	//define 12key/qwerty keypad basic position : right alignment
	alert("12key position change:"+arg);
	if (arg == '12key') {
		if (curWidget.height == '720') {
			$("#search").css("right", 384); //1280-896
			$("#search").css("top", 106);
		}
		else {
			$("#search").css("right", 286); //960-676
			$("#search").css("top", 82);
		}
	}
	else
		if (arg == 'qwerty') {

			if (curWidget.height == '720') {
				$("#search").css("right", 714); //1280-566
				$("#search").css("top", 106);
			}
			else {
				$("#search").css("right", 534); //960-426
				$("#search").css("top", 82);
			}

		}
}

function moveqwertyPosition(arg){
	alert("qwerty position change:"+arg);

	if (curWidget.height == '720') {
		$("#search").css("right",714);		//1280-566
		$("#search").css("top",106);
	}
	else{
		$("#search").css("right",534);		//960-426
		$("#search").css("top",82);
	}

}
```

1. numeric or qwerty keypad changed event callback function. IME keypad align left.

  Parameter : keyset string, ‘12key’, ‘qwerty

```
function move12KeyPosition(arg){}
function moveqwertyPosition(arg){}
```

2.this is widget display resolution, height size (‘540’, ‘720’)

```
curWidget.height == '720';
```

6. Register Enter and other keypress event callback function

And add the following code to complete ime_sample.js:

```
// Function for enter (argument : enterd string)
function sampleEnter(str) {
        alert("Enter kye pressed : "+str);
}
function textobjKeyFunc(keyCode){
        switch(keyCode) {
                case(29460) : // Up Key
                        alert("[IME] ============================ Up Key!");
                        if (_g_ime.pluginMouse_use_YN) {
                                ime._blur();
                        }
                        else{

                        }
                        document.getElementById('item').focus();
                          break;

                 case (29461) : // Down Key
                        if(_g_ime.pluginMouse_use_YN){
                                ime._blur();
                                ime2._focus();
                        }
                        else{
                                document.getElementById('searchText2').focus();
                        }
                        break;

                 case (88) : // return Key
                        alert("============================ here is it!");
                        return false;
                         break;

                 case (keyc.KEY_RED) : // RED Key

                         break;

        }
        return false;
}
```

1.set enter key callback function

```
function sampleEnter(str)
```

2. Set a callback function that will be called when when a specific key is pressed.
ex) red key, blue key,
when return false of call back function, exit and do not more key event.

Parameter : keycode of keypress event

```
function textobjKeyFunc(keyCode){}
   return false :
```

7. Register other callback function

- setOnFocusFunc - the specified callback function will be called when the input object is focused.
- setEnterFunc - the specified callback function will be called when the enter key is pressed on the input box. We will display a message on the HTML page when this callback is triggered.
- setKeyFunc - the specified callback function will be called when the specified key is pressed. We used this callback in the previous section to change the focus. Here we have added a new callback for the Info key, as an example of how key handling can be customised for the requirements of a particular widget. We will respond to this callback by displaying a message. We return true as we don't want to prevent the IME from taking action if it needs to.
- setInputHighlight - - the specified callback function will be called when input box highlight. in‘12key’ mode is aways highlight and in ‘qwerty’ mode is move highlight with TV remocon

8. Register Control functions

- setBlockSpace – Block space input.
- getKeySet – return current keyset mode. ‘12key or ‘qwerty’ string
- setInputMode –If you just want to use one keyboard mode.（‘12key or ‘qwerty’）

9. IME Function and Value Reference

The following functions are all member functions of the IME object.

Function IMECNShell	
Create IME object	
Syntax	IMECNShell(inputObjId, callbackFunc, context)
Parameter	• inputObjId: id of the input object • callbackFunc: called when the IME object has been fully created • context:
Return Value	IME object
Remarks	None
Example	imeObj = IMECNShell('input_obj_id', callbackFunc, this);

Function setInputMode	
Set only one keyboard mode	
Syntax	setInputMode (imeObj,strkeySet)
Parameter	• imeObj • strkeySet: ‘12key’ or ‘qwerty’
Return Value	None
Remarks	You can use this function when you only need one keyboard mode.(‘12key’ or ‘qwerty’)
Example	imeObj. setInputMode (imeObj,’ 12key’);

Function getInputObj	
Get input object from IME object	
Syntax	getInputObj ()
Parameter	• None

Return Value	input object from IME object
Remarks	You can use this function when you need to get the HTML input object attached to an IME object.
Example	var inputObj = imeObj.getInputObj();

Function
setAnyKeyFunc

Set a callback function that will be called each time a key is pressed, no matter what the key.

Syntax	setAnyKeyFunc(func)
Parameter	• func: callback function with following parameters: o keyCode: code of the key that was pressed o No return value
Return Value	None
Remarks	The function will be called after key handle process.

Example	var callback = function(keyCode)
	{
	alert("Key pressed, code " + keyCode);
	}
	imeObj.setAnyKeyFunc(callback);

Function
setKeyFunc

Set a callback function that will be called when when a specific key is pressed.

Syntax	setKeyFunc(keyCode, func)
Parameter	• keyCode: code for the specific key that should trigger the callback • func: callback function with the following parameters: o keyCode: the key code that was pressed o Return value: ▪ true – IME will also handle this key if it needs to. ▪ false – IME will not handle the key at all
Return Value	none

Remarks	Some keys are usually handled by the IME itself. For example, numeric keys are used for text input, and the left and right keys are used for moving the cursor. It is possible to register callbacks for these special keys using setKeyFunc(). The IME will call the callback function first, before handling the key itself. If the callback function returns true (or null), it will continue as normal to handle the key and take the usual action (like entering text or moving the cursor). If the callback function returns false, the IME will not take action for this key. This allows the client to block certain default behaviour of the IME (like cursor movement for example). This feature should be used with care as it prevents normal functioning of the IME.

Example	var tvKey = new Common.API.TVKeyValue();
	var callback = function(keyCode)
	{
	if (keyCode == tvKey.KEY_LEFT)
	{
	alert("Blocking left key");
	return false;
	}
	else
	{
	alert("Key code received: " + keyCode + ", allow IME to process");
	/* Do some action here based on keyCode */
	return true;
	}
	}
	imeObj.setKeyFunc(tvKey.KEY_LEFT, callback);
	imeObj.setKeyFunc(tvKey.KEY_INFO, callback);
	imeObj.setKeyFunc(tvKey.KEY_MENU, callback);

Function setKeypadPos	
Set the position for the display of the IME keypad window	
Syntax	setKeypadPos(x, y, z)
Parameter	• x: horizontal offset of the IME keypad display from the left edge of the screen, in pixels • y: vertical offset of the IME keypad display from the top edge of the screen, in pixels • z: z-index offset of the IME keypad. This value deside CSS z-index of the keypad. This parameter is optional.
Return Value	None
Remarks	None
Example	imeObj.setKeypadPos(500, 40);

Function setQWERTYPos	
Set the position for the display of the IME QWERTY keypad window	
Syntax	setQWERTYPos (x, y, z)
Parameter	• x: horizontal offset of the IME keypad display from the left edge of the screen, in pixels • y: vertical offset of the IME keypad display from the top edge of the screen, in pixels • z: z-index offset of the IME keypad. This value deside CSS z-index of the keypad. This parameter is optional.
Return Value	None
Remarks	None
Example	imeObj. setQWERTYPos (500, 40);

Function setEnterFunc
Set a callback function that will be called when the enter key is pressed on the input box.

Syntax	setEnterFunc(func)
Parameter	• func: callback function with the following parameters: o string: character string currently entered in the input box o No return value
Return Value	none
Remarks	This is commonly used to indicate that entry of text on a form is completed.

Example	var callback = function(string)
	{
	alert("User pressed enter, final text is " + string);
	}
	imeobj. setEnterFunc(callback);

Function
setMaxLengthFunc

Set a callback function that will be called when the maximum length of the HTML input element is reached.

Syntax	setMaxLengthFunc(func)
Parameter	• func: callback function with no parameters or return value
Return Value	none
Remarks	The maximum length is specified as the HTML property "maxlength". This callback will not be triggered if a "maxlength" property has not been set for the input element.

Example	var callback = function()
	{
	alert("Maximum length reached");
	}
	imeObj.setMaxLengthFunc(callback);

Function
setPrevEventOnLeftFunc

Set a callback function that will be called when the left key is pressed, and the cursor position is at the start of the input box.

Syntax	setPrevEventOnLeftFunc(func)
Parameter	• func: callback function with no parameters or return value
Return Value	none
Remarks	none

Example	var callback = function()
	{
	alert("Left key pressed at left side of input box");
	}
	imeObj.setPrevEventOnLeftFunc(callback);

Function
setOnCompleteFunc

Set a callback function that will be called when the entry of one character is completed.

Syntax	setOnCompleteFunc(func)
Parameter	• func: callback function with no parameters or return value
Return Value	none
Remarks	This function will be called when: 1. Right key pressed 2. Delete key pressed (delete key is indicated by IME display) 3. A space character is added 4. Entry of a character is completed For example, in a text style input box, the user may press a numeric key 3 times to choose the correct character. This callback waill not be triggered until a short time is elapsed after this, or the user presses a different numeric key – this means that the user has chosen to keep this character in the string, and so the entry of one character is completed. This callback can be used, for example, to create a list of suggestions for completing entries in a search box.

Example	var callback = function()
	{
	alert("Entry of a character is complete");
	}
	imeObj. setOnCompleteFunc(callback);

Function
setBlockSpace

Block space

Syntax	setBlockSpace(block)
Parameter	• block (optional, default: true) o true – IME will not allow the space character to be entered

	o false – IME will allow the space character to be entered
Return Value	None
Remarks	This function can be used to block entry of the space character in an input box. For example, this can be useful when entering a user id or password.
Example	imeObj.setBlockSpace (true);

Function setString	
Set the string for display in the HTML input box associated with the IME object	
Syntax	setString(string)
Parameter	• string: character string to display in the input box
Return Value	none
Remarks	This function should be used instead of directly setting the value property of the HTML input box (element.value="…"). Directly setting the value property will cause unexpected behaviour in the IME.
Example	imeObj.setString("Hello world");

Function setKeySetFunc	
Set the string for start keypad layer with the IME object	
Syntax	setKeySetFunc(string)
Parameter	• string: keypad layer type, '12key' or 'qwerty'
Return Value	none
Remarks	This function should be used set keypad layer type of the IME object.
Example	imeObj.setKeySetFunc("qwerty");

Function setKeypadChangeFunc
Set a callback function that will be called when exchange to 12key or qwerty with the IME object

Syntax	setKeypadChangeFunc (string, func)
Parameter	• string: keypad layer type, '12key' or 'qwerty' • func: callback function with no parameters or return value
Return Value	none
Remarks	Change position of the HTML Inputbox div When the IME keypad change event times.
Example	imeObj.setKeypadChangeFunc('12key',move12KeyPosition); imeObj.setKeypadChangeFunc('qwerty',moveqwertyPosition);

Function
setInputHightlight

Set a callback function that will be called when input box highlight.

Syntax	setInputHighlightFunc(func);
Parameter	• func: callback function with no parameters or return value
Return Value	none
Remarks	'12key' mode – always highlight focus 'qwerty' mode – move highlight focus with remocon direction keypress
Example	imeObj.setInputHighlightFunc(inputHighlight);

Global value
_g_ime_cn.dim_use_YN

Set background dim div use or not use with the IME object when use mouse

Syntax	_g_ime_cn.dim_use_YN = false;
Parameter	• bool: true or false, default: true
Return Value	Nothing
Remarks	Background dim div use or not use When use mouse If Apps use mouse, focus move to input box When backgound dim click. It's Global value of IME object. Add position: Before new IMEShell function. **This value is only support by 2012th TV**
Example	_g_ime_cn.dimm_use_YN = true;

	imeObj = new IMECNShell("searchText1", ime_init_text,this);
	if(!ime){
	alert("object for IMEShell create failed", 3);
	}

Global value _g_ime_cn. pluginRecog_use_YN	
Set the voice recognize guide message show or hide with the IME object	
Syntax	_g_ime_cn. pluginRecog_use_YN = false;
Parameter	• bool: true or false, default: true
Return Value	Nothing
Remarks	Voice recognize use or not use on Apps. If TV support voice recognize, Apps can set show or hide voice recognize guide message of IME keypad bottom. It's Global value of IME object. Voice recognize don't support on SDK, Add position: Before new IMEShell function **This value is only support by 2012th TV**

Example	_g_ime_cn. pluginRecog_use_YN = false;
	
	imeObj = new IMECNShell("searchText1", ime_init_text,this);
	if(!ime){
	alert("object for IMECNShell create failed", 3);
	}

Global value _g_ime_cn.pluginMouse_use_YN	
Get mouse support with the IME object	
Syntax	var mouse_use = _g_ime_cn.pluginMouse_use_YN;
Parameter	Nothing
Return Value	• bool: true or false, default value : true
Remarks	Mouse use or not use on Apps. If Apps support mouse, Apps can get status or mouse use. It's Global value of IME object.

	Add position: After new IMEShell function **This value is only support by 2012th TV**

Example	if(_g_ime_cn.pluginMouse_use_YN){
	imeObj._focus();
	document.getElementById('searchText1').focus();
	}
	else{
	document.getElementById('searchText1').focus();
	}

Function _focus()	
Set show and start of IME keypad when mouse use.	
Syntax	imeObj._focus()
Parameter	Nothing
Return Value	Nothing
Remarks	IME show and start function is defferent when Mouse use on Apps. If Apps not use mouse, show IME keypad by focus() event. If Apps use mouse, show IME keypad by _focus() function. **This function is only support by 2012th TV**

Example	if(_g_ime.pluginMouse_use_YN){
	imeObj._focus();
	document.getElementById('searchText1').focus();
	}
	else{
	document.getElementById('searchText1').focus();
	}

Function _blur()	
Set hide and close of IME keypad when mouse use.	
Syntax	imeObj._blur()

Parameter	Nothing
Return Value	Nothing
Remarks	IME hide and close function is defferent when Mouse use on Apps. If Apps not use mouse, hide IME keypad by blur() event. If Apps use mouse, hide IME keypad by _blur() function. **This function is only support by 2012th TV**

Example	if(_g_ime.pluginMouse_use_YN){
	imeObj._blur();
	imeObj2._focus();
	document.getElementById('searchText2').focus();
	}
	else{
	document.getElementById('searchText2').focus();
	}