How-to use the menu object in WRT widgets

JavaScript menu object

The menu object is supported since WRT version 1.0. The menu object is a built-in object in the widget engine. It is extended from the window object to provide widget-specific APIs for manipulating widget's options menu and softkeys. The menu object can be accessed via the window object window.menu or just simply menu. Thus, in the WRT environment, the term menu is a reserved keyword and should not be used as a user-defined global variable name for other type of objects. A widget's options menu can be constructed using the menu object. The options menu is automatically associated with the device's left softkey and it cannot be reassigned to associate with another softkey (for example, the right softkey). The label of the options menu (the left softkey) is the system-defined term in the current used system language (for example, Options for English). Thus, the left softkey label cannot be changed.

menu object methods

- void append(MenuItem menuItem)
- void remove(MenuItem menuItem)
- void replace(MenuItem oldMenuItem, MenuItem newMenuItem)
- Object MenuItem getMenuItemById(Integer menuItemId)
- Object MenuItem getMenuItemByName(String menuItemLabel)
- void setRightSoftkeyLebel(String label, Function callbackfunction)
- void showSoftkeys()
- void hideSoftkeys()
- void clear()

menu object property

- Void [Function] onShow

Detailed description of the menu object's methods

append()

Syntax: [void] window.menu.append(MenuItem menuItem) or [void] menu.append(MenuItem menuItem) Description: Call the append method to add a menu item to the top level of the options menu list. Menu items are shown on the options menu list in the order in which they are appended. Arguments:

- menuItem:

This is an instance of the MenuItem object which is being added to the top level of the options menu. Return value: This method does not return a value. Remarks: Cascading submenus are created using the MenuItem object. See also the description of the MenuItem.append method in Section 5.2.2.

remove()

Syntax: [void] window.menu.remove(MenuItem menuItem) or [void] menu.remove(MenuItem menuItem) Description: Call the remove method to remove a menu item from the top level of the options menu list. If the removed menu item has a cascading submenu in it, the submenu will also be removed. Arguments:

- menuItem:

This is an instance of the MenuItem object which is being removed from the top level of the options menu list. Return value: This method does not return a value. Remarks: Menu items in a cascading submenu list are removed using the MenuItem object's method.

replace()

THIS METHOD IS NO LONGER SUPPORTED IN WRT.

Use remove() and append() instead.

getMenuItemById()

Syntax: [MenuItem] window.menu.getMenuItemById(Integer id) or [MenuItem] menu.getMenuItemById(Integer id) Description: Call the getMenuItemById method with a specified menu item's identifier to retrieve the handle of the menu item instance. Arguments:

- id:

This is the identifier of an existing menu item whose handle is being retrieved. Return value: The returned value is an instance of the existing MenuItem object. If the id is invalid, the method returns "undefined". Remarks: N/A.

(C) Copyright Nokia 2013. All rights reserved.

getMenuItemByName()

Syntax: [MenuItem] window.menu.getMenuItemByName(String menuItemLabel) or [MenuItem] menu.getMenuItemByName(String menuItemLabel) Description: Call the getMenuItemByName method with a specified menu item's label to retrieve the handle of the menu item instance. Arguments:

- menuItemLabel:

This is the label of an existing menu item whose handle is being retrieved. See also the MenuItem object description in Section 5.1. Return value: The returned value is an instance of the existing MenuItem object.If the label is invalid, the method returns "undefined".

setRightSoftkeyLabel()

Syntax: [void] window.menu.setRightSoftkeyLabel(String label, Function callbackfunc) or [void] menu.setRightSoftkeyLabel(String label, Function callbackfunc) Description: Call the setRightSoftkeyLabel method to customize the label and the operation associated with the right softkey. By default, the right softkey of a device is assigned to the "Exit" function, which terminates a running widget. The default label depends on the current used system language (Exit for English). Arguments:

- label:

A text string specifies the label to be shown on the right softkey.

- callbackfunc:

A reference to the callback function, which will be called by the system when the right softkey is pressed. Return value: This method does not return a value. Remarks: The default label and "Exit" function of the right softkey can be restored by calling the setRightSoftkeyLabel with an empty string (‘’) specified for the label argument and the callbackfunc argument set to null. Example code:

```
function switchToSettingsView(){
    // implement and show settings
    ...
    // customize the right softkey
    window.menu.setRightSoftkeyLabel('Back', returnToMainView);
}

function returnToMainView(){
    // show main view
    ...
    // restore the default right softkey
    window.menu.setRightSoftkeyLabel('', null);
}
```

showSoftkeys()

Syntax: [void] window.menu.showSoftkeys(void) or [void] menu.showSoftkeys(void) Description: Call the showSoftkeys method to display the softkey pane at all times. By default, the device's softkey pane is hidden. The softkey pane is shown either by programmatically calling the showSoftkeys method or when the end user presses one of the softkeys. Arguments: This method takes no arguments. Return value: This method does not return a value. Example code:

```
// choose to show the softkey pane at all times
window.menu.showSoftkeys();
```

hideSoftkeys()

Syntax: [void] window.menu.hideSoftkeys(void) or [void] menu.hideSoftkeys(void) Description: Call the hideSoftkeys method to hide the softkey pane. By default, the device's softkey pane is hidden. The softkey pane is shown automatically when the end user presses one of the softkeys. Arguments: This method takes no arguments. Return value: This method does not return a value. Example code:

```
function useFullScreen(){
    // choose to hide the softkey pane to get more screen space
    window.menu.hideSoftkeys();
    ...
};
```

clear()

Syntax: [void] window.menu.clear(void) or [void] menu.clear(void) Description: Call the clear method to delete all the menu items in the options menu pane. This operation will also clear all submenus if such exist. Arguments: This method takes no arguments. Return value: This method does not return a value.

Description of the menu object's property

onShow

Syntax: [void] window.menu.onShow = function() { } or [void] menu.onShow = menupaneShown;

```
function menupaneShown(){
    // ...
};
```


(C) Copyright Nokia 2013. All rights reserved.

Description: The onShow property of the menu object is an event handler for the event of when the options menu is open. In other words, when the user presses the left softkey, the system will fire an event and a widget can catch the event by providing a call back function.

JavaScript MenuItem object

The MenuItem object is supported since WRT version 1.0. MenuItem is a built-in object in the widget engine. The MenuItem object is a reference type that provides APIs for manipulating a widget's menu items. The MenuItem object is designed to be used together with the menu object to create and manipulate the widget's main menu and cascading submenus. The MenuItem object is created by using the new operator. MenuItem object methods:

- Object MenuItem(String label, Integer Id)
- Void append(MenuItem childMenuItem)
- Void remove(MenuItem childMenuItem)
- Void replace(MenuItem oldMenuItem, MenuItem newMenuItem)
- Void setDimmed(Boolean true|false)

MenuItem object property:

- Void [Function] onSelect

Description of the MenuItem object's methods

Constructor

Syntax: [MenuItem] new MenuItem(String label, Integer id) Description: The new operator creates and instantiates an instance of the MenuItem object. Arguments:

- label:

A text string that defines the label for the menu item.

- id:

A unique integer that identifies the menu item. Return value: An initialized instance of the MenuItem object. Remarks: If the arguments are not valid, a JavaScript exception indicating the error type is thrown. A valid argument implies an integer type for id and a string type for label. The string length for the label is truncated to 40 characters if it exceeds that limit. Menu items on the whole menu tree are unique and identified by their id.

append()

Syntax: [void] MenuItem.append(MenuItem childMenuItem) Description: Call the append method to add a child menu item to the parent menu item in the options menu list. This results in the creation of a submenu list in the menu tree. Use this method to create a cascading submenu when needed. Menu items are shown on the options menu list in the order in which they are appended. Arguments:

- childMenuItem:

This is an instance of the MenuItem object which is being added to the parent menu item. See also the menu.append method description in Section 4.2.1 for instructions on how to append a menu item to the top level of the options menu list. Return value: This method does not return a value. Remarks: For devices with a small screen, it is not practical to create multi-level cascading submenus. Thus, a maximum of two-levels is recommended for submenus. Appending a new menu item fails if it tries to create a duplicate menu item (it either has the same id or the same label as an existing menu item).

remove()

Syntax: [void] MenuItem.remove(MenuItem childMenuItem) Description: Call the remove method to remove a child menu item and its children (if any) from the parent menu item. Arguments:

- childMenuItem:

This is an instance of the MenuItem object which is being removed from the parent menu item. See also the menu.remove method description in Section 4.2.2 for instructions on how to remove a menu item from the top level of the options menu list. Return value: This method does not return a value. Remarks: To remove a menu item and its children from the top level of the options menu list, use the menu object's method.

replace()

THIS METHOD IS NO LONGER SUPPORTED IN WRT. Use remove() and append() instead.

setDimmed()

Syntax: [void] MenuItem.setDimmed(Boolean flag) Description: Call the setDimmed method to show or hide an existing menu item. By default, a menu item is shown when it is appended to the options menu. Arguments:

- flag:

true to show the menu item, and false to hide the menu item. Return value: This method does not return a value.

Detailed description of the MenuItem object's property

onSelect

(C) Copyright Nokia 2013. All rights reserved.

Syntax: MenuItem.onSelect = function(Integer id) { } or MenuItem.onSelect = onMenuItemSelected;

```
function onMenuItemSelected(id){
    // ...
};
```

Description: The onSelect property of the MenuItem object is an event handler for the event of when the memenu item is selected. In other words, when the end user opens the options menu and selects a menu item either from the top-level menu list or from a submenu list, the system will fire an event and a widget can catch the event by providing a callback function. The callback function is passed with an argument, which is an integer identifier identifying the menu item that was just selected. It is possible to assign an individual callback function for each menu item so that the id argument can be ignored.
Remarks: Submenu item's callback function must be assigned to the onSelect property after the parent menu item is appended to the main menu pane.
Example code:

```
// Creating a menu:
window.onload = createMenu();


// function to create a menu
function createMenu() {

    // Create a Menu Object
    var optionsMenu = window.menu;
    // Set a callback function for Menu
    optionsMenu.onShow = function() {
      alert('Event Trigger: optionsMenu.onShow');
    }
    // Create two Menu items
    var m1 = new MenuItem('Beverages', 2001);
    var m2 = new MenuItem('Snacks', 2002);
    // Assign a callback function for the menu items
    m1.onSelect = menuEventHandler;
    m2.onSelect = menuEventHandler;
    // Append two Menu items to Menu
    optionsMenu.append(m1);
    optionsMenu.append(m2);
    // Create two more Menu items for Sub-Menu
    var m11 = new MenuItem('Coca Cola', 3001);
    var m12 = new MenuItem('Pepsi', 3002);
    // Assign a callback function for the submenu items
    m11.onSelect = submenuEventHandler;
    m12.onSelect = submenuEventHandler;
    // Append two Sub Menu Items to Menu 'Beverages'
    // get Menu Item reference by ID
    optionsMenu.getMenuItemById(2001).append(m11);
    // get Menu Item reference by Name
    optionsMenu.getMenuItemByName('Beverages').append(m12);
};


// Implement menu event handler:
function menuEventHandler(id) {
  switch (id) {
      case 2001:
      break;
      case 2002: // do something
      break;
  }
};

==Example Code==
[[File:Themenu.zip]]
```


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.