Adding visible feedback to buttons links and clickable elements in Symbian Web Runtime apps

Overview

When designing intuitive UI for a widget, you might want to show a visible response when the user clicks on a button, on a link or other click-able elements. The method of using only the `:hover`, `:active`, or `:focus` CSS selectors is not working (see below for details).

A working approach is to make a javascript function to add a click routine to the element. This means that when the onmousedown event is received, the class of the element is changed to `activeButton`. When the onmouseup event is received or the focus is lost, the style is changed to `passiveButton`. If there are other classes in the button, they are not replaced or removed.

First of all, we have defined a button in the page as follows:

```
<button id="testButton" class="iconButton"  />
```

The CSS style of the element is given as follows:

```
button.iconButton {
  padding: 0px 40px;
  margin: 10px;
  border: 10px solid black;
  min-height: 140px;
  min-width: 140px;
}
```

So, to add a response to the button press, we add a following style declaration for it:

```
button.iconButton.activeButton {
  border: 10px solid green;
  background: white;
}
```

Then, we modify the init function so that the button is given to a `addClickRoutine` function, which adds the proper onclick event handlers for the button.

```
// Initializes the widget
function init() {
  addClickRoutine(document.getElementById("testButton"));
//    createMenu();
}

...

/*
  addClickRoutine - a method to add a click routine to a button or any other clickable
element

  @param element the element to which we attach the onclick handler
  @param styledElement the element of which the style is changed (defaults to @param
element)
```


(C) Copyright Nokia 2014. All rights reserved.

```
  **/
function addClickRoutine(element, styledElement) {
  if (!styledElement) {
    styledElement = element;
  }
  element.onmousedown = function() {
    var elementClass = styledElement.getAttribute("class");
    if (!elementClass) {
      elementClass = "activeButton";
    }
    // A better approach: use regexps like /(^ )passiveButton( $)/
    // and store them as a singleton instance
    if (elementClass.search("passiveButton") == -1) {
      if (elementClass.search("activeButton") == -1) {
        elementClass = elementClass+" activeButton";
      }
      // else already activeButton
    }
    else {
      elementClass = elementClass.replace(/passiveButton/, "activeButton");
    }
    styledElement.setAttribute("class", elementClass);
  }
  element.onmouseup = function() {
    window.setTimeout(function() {
      var elementClass = styledElement.getAttribute("class");
      styledElement.setAttribute("class", elementClass.replace(/activeButton/,
"passiveButton"));
    }, 250);
  }
  element.onblur = function() {
    var elementClass = styledElement.getAttribute("class");
    styledElement.setAttribute("class", elementClass.replace(/activeButton/,
"passiveButton"));
  }
}
```

The `addClickRoutine` function has an optional `styledElement` parameter, since sometimes you might want to change the style of some other element than the element which was clicked (for example, the style of a container element).

Postconditions

Now, when clicking the button, the button is highlighted for short time.

Further information

This section provides further information about the situation in 2010-10-29.

The CSS selector `:hover` is not working as expected, since when the user presses a button and then removes the finger from the screen, the hover effect is still shown.

The CSS selector `:focus` is not working at all.

The CSS selector `:active` is working only when the navigation type is set to none and the user presses the button long enough, so short taps are not shown. Furthermore, the response is not instant, and takes longer time than the javascript method presented here.

Tested with E7, software version 012.002, and N8, PR1.0, software version 011.008 (date 2010-08-19). A bug report submitted.

(C) Copyright Nokia 2014. All rights reserved.

Supplementary material

This code snippet is part of the stub concept, which means that it has been patched on top of a template application in order to be more useful for developers. The version of the WRT stub application used as a template in this snippet is v1.2.

- The patched, executable application that can be used to test the features described in this snippet is available for download at Media:adding visible feedback to buttons links and clickable elements widget.zip.
- For unpatched stub applications, see Example stub.

(C) Copyright Nokia 2014. All rights reserved.