OpenEyes - Keyboard Navigation

Editors: G W Aylward
Version: 0.9:
Date issued: 18 June 2010

Target Audience

General Interest	✔
Healthcare managers	
Ophthalmologists	✔
Developers	✔

Amendment Record

Issue	Description	Author	Date
0.9	First issue	G W Aylward	18 June 2010

Table of Contents

Introduction

OpenEyes will be used for extended periods of time during busy outpatient clinics, and therefore ease of use, and speed of navigation are of paramount importance. Using the mouse as an input device can be slow, so it is sensible to make extensive use of keyboard navigation. This document sets out the design, presentation, and operation of keyboard navigation throughout OpenEyes.

User Interface

Keyboard navigation should behave in a consistent manner, similar to a Desktop application, and be independent of the type of browser being used. This section describes the expected behaviour of each key

Return Key

The return (enter) key causes form submission in many browsers. This is prevented in OpenEyes using javascript, and submission only occurs if the return key is pressed while the keyboard focus is on a submit element, a button element attached to a method which leads to submission, or for certain text boxes (the class attribute is set to “submit”). The action of the return key depends on which control has focus, but is summarised in the following table.

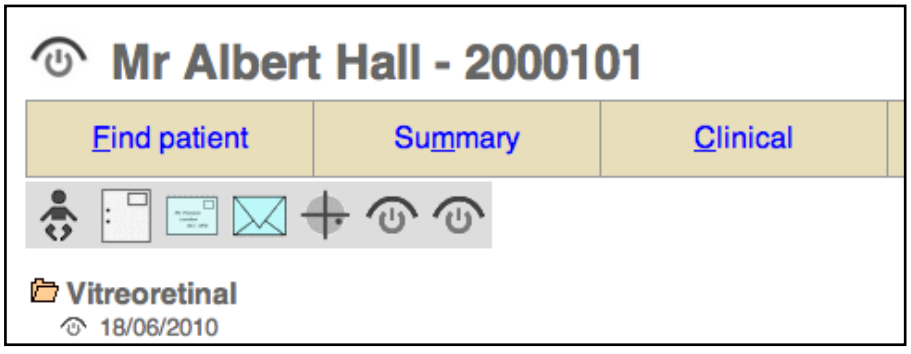
Element with focus	Action
Text box	Move to next element or control (ie same as TAB)
Text Area	Insert a carriage return
Select	Confirm selection and move to next element or control

Tab Key

The tab key is used to move the keyboard focus between data input fields and controls. When the user reaches the last control on the page, focus should return to the first.

Keyboard shortcuts

Keyboard shortcuts are triggered by a key combination consisting of an action key or keys, and another key on the keyboard (usually a letter, number, or punctuation mark). Shortcuts are preferably displayed to the user by one of the letters in the label being underlined as in the following screen shot;



The action key is pressed in combination with the control and alt key. This combination has been chosen for OpenEyes as it minimises conflict with built in shortcuts in the major browsers.

Implementation

The HTML standard defines the accesskey attribute which can be used to trigger actions using a combination of a trigger key and the accesskey. For example, the attribute accesskey="H" would allow the combination of the trigger key and the 'H' key to activate the element. Unfortunately browser developers have implemented the concept in a variable and inconsistent manner. For example the key combination required differs between browsers as indicated in the following table

Browser	Key combination
Internet Explorer	Alt + [the accesskey]
Firefox for Windows	Shift + Alt + [the accesskey]
Opera for Windows or Mac	Shift + Esc + [the accesskey]

In addtion, there are often conflicts with the browser's own shortcuts, particularly in Internet Explorer which has allocated most Alt - key combinations for its own use. For these reasons, OpenEyes uses javascript rather than the accesskey attribute in order to implement keyboard shortcuts (see appendix 1). Any element to be activated by a keyboard shortcut should be given an id attribute consisting of the access key with the prefix 'oeak' (abbreviation for OpenEyes Access Key). For the previous example the attribute would be id="oeakH";

Shortcuts

The following table details the shortcuts found throughout OpenEyes.

All Modes:

Key	Action
S	Saves the currently edited form
K	Cancels the current edit and dismisses the form
E	Edits the currently displayed item or page (if date and permissions allow)

Admin Mode:

Key	Action
F	Launches the find patient page
A	Launches the administration modules

Key	Action
B	Launches the booking module
D	Launches the audit modules
P	Launches the profile module
H	Loads the home page
O	Logs out of the system

Patient Mode:

Key	Action
F	Launches the find patient page
M	Displays the patient summary
C	Displays the clinical modules
D	Displays the full list of diagnoses
T	Displays the contacts page
H	Loads the home page
O	Logs out of the system

Appendix 1

Javascript method

The following javascript function (found in common.js) intercepts all keystrokes in the HTML document and deals with them accordingly.

```
document.onkeydown = keyPressed;

// Function to handle key presses
function keyPressed(e)
{
    // IE needs a window.event
    var event = e ? e : window.event;

    // Check for ctrl and alt key pressed
    if (event)
    {
          if (event.altKey && event.ctrlKey)
          {
                // Don't trigger on alt key alone
                if (event.keyCode != 18)
                {
                      // Generate id of target element
                      var elementId = "oeak" +
String.fromCharCode(event.keyCode);

                      // Attempt to get element with that id
                      var element = document.getElementById(elementId);

                      // If exists, it will either be an <a> with an href at-
tribute or an element with a click method
                      if (element)
                      {
                            // Shift focus to it
                            element.focus();

                            // Generate action
                            var url = element.href;
                            if (url)
                            {
                                  window.location = url;
                            }
                            else
                            {
                                  element.click();
                            }
                      }
                }
          }
          // Return key should not submit form, unless active element is of
type submit
          if (event.keyCode == 13)
          {
                // Attempt to get active element type
                var element = event.srcElement;
```

```
                // If its a submit click it, otherwise return false
                if (element && element.type == "submit")
                {
                      element.click();
                }
                // If its a text input with class 'submit' then allow submis-
sion
                else if (element.className == "submit")
                {
                      return true;
                }
                // Otherwise prevent submission
                else
                {
                      return false;
                }
          }
    }
}
```