

Mobile Internet Application Design Issues

There are numerous mobile device vendors with no common widely accepted standard regarding any aspect of their devices! Creating an application that works as intended on multiple devices is not possible. Further even the same vendor's different product lines (for example Nokia phone series) are incompatible. There are a number of technologies and mobile platforms available and here I specifically look at designing for Windows Mobile 2003 Pocket PC edition.

1. Available Technologies

There are a number of technologies to develop and present content on mobile devices. Client side display technologies are based on markup languages – HDML¹, WML², HTML³, cHTML⁴, XHTML³, VoiceXML⁵ are some major ones. Each has its own advantages and purpose and serves different kinds of mobile devices.

Dynamic content can be generated by all standard technologies currently used and no significant new resource is required. CGI⁶, Perl⁷, Java Servlets⁸, JSP⁹, ASP/ASP.NET¹⁰, PHP¹¹ etc are good examples.

Understanding the targeted devices is very important in determining the appropriate technology to use. Details of these specific technologies available at the references, here I am presenting a generic discussion that may be applied to your available device.

2. Device Independence

W3C [DI W3C] Device Independence group is attempting to create techniques for device independence. The goal is to prevent fragmentation of the Web into spots accessible only by certain types of devices. Principles related to the user, developer and delivery mode are suggested.

User Principles

These relate to the user experience of using multiple devices and access mechanisms for interacting with the World Wide Web.

¹ <http://www.w3.org/TR/NOTE-Submission-HDML-spec.html>

² <http://www.wapforum.org/what/technical.htm>

³ <http://www.w3.org/MarkUp/>

⁴ <http://www.w3.org/TR/1998/NOTE-compactHTML-19980209/>

⁵ <http://www.w3.org/TR/2004/REC-voicexml20-20040316/>

⁶ <http://hoohoo.ncsa.uiuc.edu/cgi/overview.html>

⁷ <http://perl.apache.org/>

⁸ <http://java.sun.com/products/servlet/>

⁹ <http://java.sun.com/products/jsp>

¹⁰ <http://msdn.microsoft.com/asp.net/>

¹¹ <http://www.php.net>

Principle 1: Device Independent Access “For some web content or application to be device independent, it should be possible for a user to obtain a functional user experience associated with its web page identifier via any access mechanism.”

Functional user experience relates to perceived stimuli provided to the user by the device (e.g. visual, audio, and force-feedback) that enables the user to complete a desired function as *intended by the developer*. The access mechanism is also important since not all mechanisms may permit the necessary affordances due to inherent limitations of the technology. The web-page identifier refers to the URI of a page.

It is important to note that this does not imply that the user should have the *same* experience on all devices. The user should get a *similar functional experience*, i.e. be able to complete the task.

Principle 2: Device Independent web page identifiers “A web page identifier that provides a functional user experience via one access mechanism should also provide a user experience of equivalent functionality via any other access mechanism.”

Developer (Author) Principles

These relate to the interpretation of the user request and subsequent processing and delivery of information.

Principle 3: Functionality “It should be possible to provide a functional user experience, in response to a request for a web page, in any given delivery context that has an adequate access mechanism.”

Principle 4: Incompatible access mechanism “If a functional user experience of an application cannot be provided due to inherent limitations in the access mechanism, an explanatory message should be provided to the user.”

Principle 5: Harmonization “If the author wishes, it should be possible to provide a harmonized user experience, in response to a request for a web page, in any given delivery context that has an adequate access mechanism.”

These principles emphasize that the developer/author is responsible for ensuring that different users get the same functional experience and if not possible due to inherent limitations then a suitable message should be displayed. Further user experience should be optimized for the access mechanism of the user.

Delivery Principles

These relate to the access mechanism, the user device and its capabilities to send requests and rendering the information for presentation to the user.

Principle 6: Characterization of delivery context “The user agent should be able to associate the characteristics of the delivery context with a request for a particular web page.”

Principle 7: Adaptation Preferences “It should be possible for a user to provide or update any adaptation preferences as part of the delivery context.”

The user agent is the mobile device and the user should have the ability to control the adaptation preferences. These client side adaptation preferences for example can determine if low resolution images should be requested.

Device independence is the goal, and my current work aims to achieve this by using the 5-layer application architecture as discussed. My currently implementations are optimized for Windows Mobile 2003 Pocket PC edition. For example the object oriented classes can be easily used to create pages optimized for Palm OS devices.

3. Thin-Client Application Design for Windows Mobile 2003¹² - Pocket PC

Developing thin-client applications for Pocket Internet Explorer is similar to development for desktop browsers. Pocket IE available with Windows Mobile 2003 supports CSS as well and pages are developed to conform to HTML 3.2. It must be noted that complex multi-layer interfaces especially with mouse-over function cannot be designed for Pocket IE. Macromedia Flash ActiveX for Pocket PCs can however be installed and such functionality made available however this might cause problems to users as the Flash ActiveX is not shipped with the device.

I have tested my applications on Intel X-Scale processor equipped HP iPAQ¹³ 3800 series and Dell Axim¹⁴ X30 series devices. All devices had 802.11 connectivity running Windows Mobile 2003, Second Edition for Pocket PC. This version of Pocket IE (4.0) supports SSL and CSS and both were used extensively.

3.1 Available Screen Real Estate

Generally a Pocket PC is used in the ‘portrait’ orientation and typically this gives 240x320 pixels. When designing for the Pocket Internet Explorer, of the available height of 320 pixels, 26 are used by the Windows Caption bar, 23 by the address bar, 80 by the Soft Input Panel (SIP), (when active), 11 by the horizontal scroll bar (when active) and 26 by the Menu Bar. Horizontally all 240 may be available unless 11 pixels used by the vertical scroll bar. Refer to Figure 1 and Figure 2.

¹² Note: this is specific to Windows Mobile 2003, older versions do not support features mentioned and used here

¹³ <http://welcome.hp.com/country/us/en/prodserv/handheld.html>

¹⁴ <http://www1.us.dell.com/content/topics/segtopic.aspx/vanity/axim?c=us&l=en&s=gen>

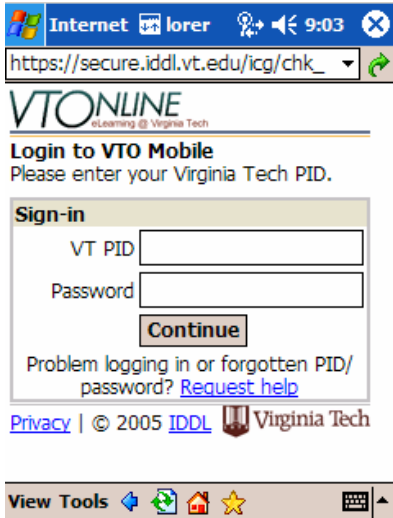


Figure 1: Available Screen Space - without soft input panel

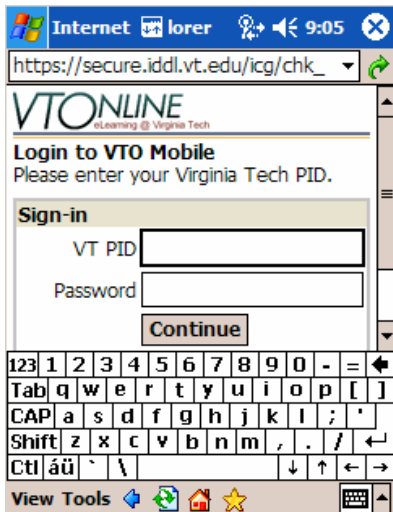


Figure 2: Available Screen Space - with soft keyboard (or transcriber, block recognizer etc)

3.2 Caveats

For developers used to developing for a full fledged web browser, a mobile browser (Pocket IE) presents a number of new challenges.

- No pop up windows are possible
- Multiple browser also cannot be open simultaneously, only one window is available
- “Mouse-over” events as available on desktop environments are not available
- Tapping once or tap and hold (equivalent of a right-click) are the only options
- This also does away with the availability of helpful tool-tips as used for many interfaces

While usability methods to employ, metaphors to use etc are all similar to any UI design (look up and usability engineering class) some special points to note for Mobile devices are now mentioned.

3.3 Design Considerations

MSDN provides some useful design guidelines for Windows Mobile¹⁵. The small screen and limited storage availability together with generally low bandwidth connectivity (Bluetooth, GPRS) present many challenges. The CPU is also not as powerful and power needs to be conserved as well. Meeting ‘users expectations’ is the most important consideration. Some issues to consider:

- Fonts: Normally very few fonts are available and use of serifs may result in poor display.
- Colors: Generally when powered on battery the screen dims (adjustable) to conserve power. This may result in poor color contrasts so ensure that colors used are visible in low light.
- Unnecessary sounds and animation should be avoided (conserve CPU, power)
- Sizes of interface elements: The size of the interface elements should be allowable to be tapped using (i) a stylus or (ii) a finger. Obviously for a finger tap the area need to be much larger and a good example is the password keypad display (Figure 3). A stylus can afford a smaller area for accurate tapping however this too can't be too small. (MSDN suggested: 21x21 pixels for stylus and 38x38 pixel for finger).

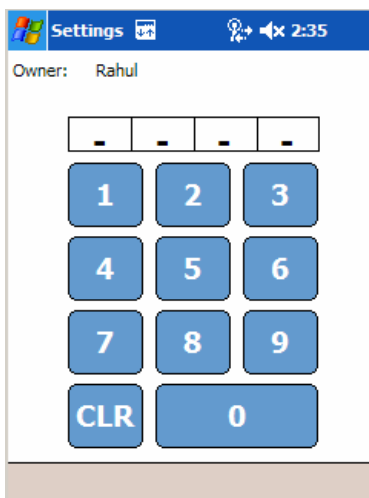


Figure 3: Finger tapping capable display

¹⁵ <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/mobilesdk5/html/mob5oriCreatingWindowsMobileApplications.asp>