



Introducción a .NET Compact Framework 2.0 # Anatomía de una dispositivo móvil

INTRODUCCIÓN A .NET COMPACT FRAMEWORK 2.0

Derecho de Autor © 2007 José Miguel Torres.

Permiso para copiar, distribuir y/o modificar este documento bajo los términos de la Licencia de Documentación Libre GNU, Versión 1.1 o cualquier otra versión posterior publicada por la *Free Software Foundation*; con las Secciones Invariantes siendo desarrolloMobile.NET, con los siendo desarrolloMobile.NET el texto de la Cubierta Frontal, y siendo desarrolloMobile.NET el texto de la Cubierta Posterior. Una copia de la licencia es incluida en la sección titulada "Licencia de Documentación Libre GNU".

Objetivos de este documento:

Este documento es el tercero de una serie de varios y que hablará sobre como empezar a desarrollar aplicaciones en .NET CF 2.0.

Se recomienda una mínima formación sobre programación y nociones muy básicas sobre la plataforma .NET.

El objetivo de este documento es conocer las características más importantes de un dispositivo móvil que debemos conocer para el desarrollo de aplicaciones.

NOTA IMPORTANTE: Existen gran cantidad de artículos en Internet que hablan sobre temas más específicos de hardware que se tratan en este documento. En algunos casos mostraré el enlace a estos artículos puesto que entiendo que es mejor redirigirte a un artículo de calidad que tratar de hacer lo mismo yo aquí.

Conceptos de hardware a tener en cuenta.

En el desarrollo de una aplicación Windows, Web o Middleware no tenemos en cuenta si el PC o servidor dónde se esté ejecutando tiene o no alimentación, espacio disponible o cobertura GPRS. Son conceptos que se dan por supuestos (o sencillamente no son necesarios como la cobertura GPRS) que estarán y no debemos (o no debiéramos) preocuparnos de ellos.

.NET Compact Framework está enfocado a dispositivos móviles, esto es, con determinadas características en la administración de memoria, procesos y recursos. Las aplicaciones que desarrollemos sobre ellos, aunque gracias al CLR de .NET CF 2.0 en ocasiones son transparentes, no debemos olvidar que el dispositivo móvil puede no tener suficiente batería como para, por ejemplo, efectuar una conexión GPRS/3G a un servicio Web, o bien no tiene el suficiente espacio y eficiencia como para almacenar un gran volumen de datos. Estos escenarios no nos los encontramos en aplicaciones Windows o Web.

Una premisa que debéis tener en cuenta es que todos los dispositivos son distintos entre sí. Evidentemente no los que son del mismo modelo y marca. Un PC clónico con Windows XP SP2 es distinto a uno “de marca” con el mismo sistema operativo pero esas diferencias (RAM, Caché, disco, procesador, controladores, etc...) no son **temas** a tener en cuenta antes del desarrollo. Tampoco pretendo asustaros, las diferencias entre dispositivos tampoco son abismales. Pero entonces, ¿a que tipo de diferencias me estoy refiriendo? Me explico...

Windows Mobile es instalado por parte del fabricante a cada uno de sus modelos siguiendo unas especificaciones técnicas. Microsoft, por su parte, provee todas las funcionalidades básicas a Windows Mobile sin embargo no todas esas funcionalidades están disponibles. Por ejemplo, un dispositivo de la marca X incorpora un controlador de Bluetooth de Microsoft (**Microsoft Bluetooth Stack**). Windows Mobile soporta este tipo de controlador por defecto así que, no habrá problema alguno. Otros fabricantes optan por utilizar Bluetooth de WIDCOMM. En ese caso el controlador Bluetooth que incorpora Windows Mobile por defecto es sustituido por el de WIDCOMM. Este controlador reside en ROM, así que forma parte del sistema operativo. Si decides hacer cualquier tipo de código de control de Bluetooth para un determinado dispositivo que utiliza Microsoft Bluetooth Stack no te servirá de nada si lo ejecutas en otro dispositivo con WIDCOMM. Lo mismo ocurre con el control de intensidad de la pantalla o teclado. Windows Mobile aporta la funcionalidad de modificarla programáticamente sin embargo los fabricantes pueden optar por mantenerlo o sustituirlo por controlarlo por librerías nativas, por ejemplo.

Son casos, los mencionados, algo rebuscados, pero si tienes intención de crear aplicaciones en un entorno empresarial verás como esas funcionalidades (u otras) las requerirás.

En definitiva, ten en cuenta estas premisas para futuras implementaciones en dispositivos móviles.

Diferencias entre PocketPC y Smartphone

Otro dilema a la hora de crear aplicaciones multiplataforma es conocer la propia plataforma. La principal diferencia entre Smartphone y Pocket PC es que esta última incorpora pantalla táctil mientras que Smartphone la navegación es por botones. Así las diferencia en la utilización de controles Windows para Smartphone y Pocket PC se resumen en el siguiente enlace:

[http://msdn2.microsoft.com/en-us/library/ms228834\(VS.80\).aspx](http://msdn2.microsoft.com/en-us/library/ms228834(VS.80).aspx)

Además existe otro artículo interesante donde muestra más en profundidad las diferencias de desarrollo.

<http://www.oreillynet.com/pub/a/wireless/2004/01/07/smartphone.html>



Normalmente, las capacidades de almacenamiento y velocidades del procesador son superiores en las Pocket PC. No es así en cuestiones de batería, puesto que depende de cada fabricante. Cabe la posibilidad de que existan Smartphone con duración de sus baterías superiores a otros Pocket PC pero puede ocurrir perfectamente lo contrario.

Administración de memoria en Windows Mobile

Windows Mobile 5 cambia la forma de administrar la memoria respecto a sus antecesores. Yo me centraré en explicar lo que necesitamos saber a partir de Windows Mobile 5 en adelante. Si queréis ampliar la información sobre diferencias y demás echad un vistazo a

http://www.pocketpcfaq.com/faqs/5.0/memory_management.htm

En un dispositivo móvil podemos distinguir a grandes rasgos 3 tipos de memoria. Memoria RAM, ROM y Storage Card.

La memoria RAM (Random Access Memory) requiere de una fuente de alimentación constante sin embargo no lo necesita la ROM (Read Only Memory). Es por esta razón que tanto el sistema operativo, como nuestros datos residen en ROM. La memoria RAM, por otro lado, es mas rápida que la ROM. Normalmente el tamaño de la capacidad de memoria ROM es mayor. Cuando instalamos un programa lo hacemos en la ROM pero cuando se ejecuta se hace en la RAM. En algún sitio he visto que la ROM es como una memoria SD interna (para hacernos un idea, claro)

Respecto a tarjetas de memorias externas sólo decir que es algo más lento (el acceso) que la ROM así que en aplicaciones de acceso a base de datos notareis una degradación en el tiempo de acceso a la base de datos. Para obtener más información acerca este tipo de memorias otro artículo:

<http://www.microsoft.com/windowsmobile/articles/storage.mspix>

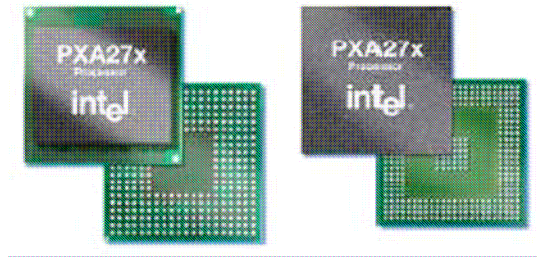
Si queréis, además, saber como desde que ejecutamos una aplicación .NET CF 2.0 hasta que finaliza el sistema operativo gestiona la memoria, que papel juega el CLR y qué memoria entra en escena os recomendaré un artículo <http://msdn2.microsoft.com/en-us/library/s6xoc3a4.aspx>. Pero si realmente queréis profundizar sobre el tema, si os atrevéis, no os perdáis este artículo de Mike Zintel (.Net Compact Framework Advanced Memory Management). Sencillamente impresionante.

<http://blogs.msdn.com/mikezintel/archive/2004/12/08/278153.aspx>

Procesadores

Seguro que alguna vez te has preguntado por qué cuando genero un archivo CAB o bien me voy a instalar SQL Server CE existen varias archivos CAB en base a distintos procesadores. Básicamente, son todos los procesadores que existen actualmente en el mercado sin embargo muchos de ellos están obsoletos. ARM, MIPS o SH3 son procesadores predecesores de los actuales.

Actualmente se está utilizando la tecnología XScale de Intel. Antes estuvo el Strong ARM. Con la aparición de Pocket PC 2002 Microsoft estandarizó un conjunto de instrucciones bajo el nombre de ARMv4. Todos los dispositivos a partir de entonces son compatibles con ARM. Posteriormente Intel evolucionó hasta XScale, como decía. Éste soporta tanto el conjunto de instrucciones v4 como le evolución v5. Dentro de este apartado encontramos procesadores que llegan hasta 624Mhz (PXA27X) con soporte para conectividad inalámbrica y soporte multimedia MMX.



Si quieres ampliar más información échale un vistazo a:

- Microprocesador ARM <http://es.wikipedia.org/wiki/ARM>
- Microprocesador MIPS <http://es.wikipedia.org/wiki/MIPS>
- Microprocesador StrongArm <http://es.wikipedia.org/wiki/StrongARM>
- Microprocesador Intel XScale http://es.wikipedia.org/wiki/Intel_XScale

La comprensión del funcionamiento, mejoras y especificaciones de los procesadores es importante aunque no imprescindible en .NET Compact Framework. Te animo a que si quieres profundizar en el tema eches un vistazo a los enlaces que te adjunto.

Navegación

Si de una cosa se caracterizan los dispositivos móviles es de la carencia de teclados, ratones y demás. Informática de mano, esto es, todo lo necesario lo tenemos en el dispositivo. Sin embargo, y como comenté en las diferencias entre Smartphone y Pocket PC, la pantalla táctil no es un requisito, ni tampoco el teclado (QWERTY o “telefónico”).

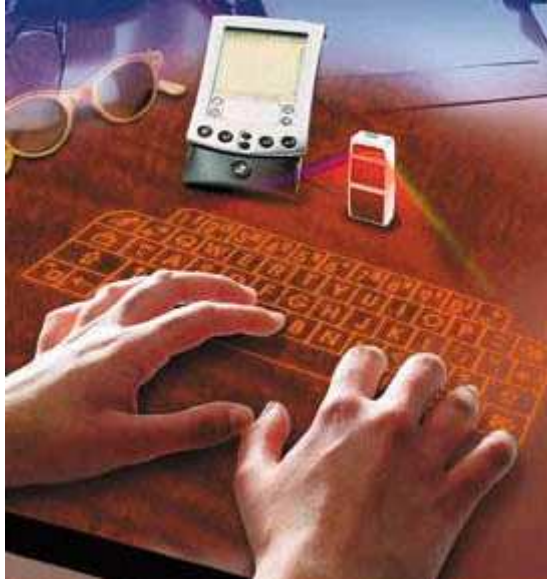
De todos modos existen dispositivos de entrada para todos los gustos:



http://www.hama.de/portal/articleId*110196/picType*awd2/action*2599?picURL=%2Fbilder%2F00036%2Fawd%2F00036_990awd2.jpg#picture

An advertisement for the E-TEN glofish M700 mobile phone. The phone is shown in a three-quarter view, highlighting its silver and black design and the attached QWERTY keyboard. The screen displays a menu with options like 'Pocket Mail', 'SMS Messages', 'MMS Mailbox', 'Expanded Mailbox', 'MMS Mailbox', and 'Pocket MMS Help'. To the left of the phone, there are icons for WiFi, GPS built-in, and QWERTY Keyboard. The text 'M700' is prominently displayed in large white letters. Below the phone, the text reads: 'E-mailing and messaging made easy with our new 'QWERTY' Keyboard! Featuring blue light night-glowing key pad'.

E-TEN glofish M700



iBIZ Technology Corp.



MPX300

En definitiva, cuando desarrollemos nuestros programas debemos ser conscientes de qué modo de entrada de datos tiene la plataforma para la cual desarrollamos de forma predeterminada. Evidentemente no podemos hacer una especificación para cada una de las PDA, o hardware de entrada, en mayor medida ya se ocupa (o debiera hacerlo) el sistema operativo.

Resumiendo, no hagamos de nuestra aplicación una experiencia similar para el usuario como si de una operación de cirugía cardiovascular se tratara o una combinación tediosa por teclado; al fin y al cabo es cuestión de botones, cuadros de texto (bueno más o menos ;-))

Conectividad

Otro aspecto a tener en cuenta es la conectividad de nuestro dispositivo. Este es un aspecto relativamente importante según sea el propósito de nuestras aplicaciones. En todo caso, no está de más que nos familiaricemos con las posibilidades que brinda nuestra PDA en cuanto a comunicaciones Bluetooth, IrDA, WiFi, además de las siempre soportadas por ActiveSync.

Creo que aquí es el lugar más idóneo de hablar de GSM, 3G, UMTS y demás. En definitiva son un montón de siglas, que pese a que muchas de ellas existen desde hace bastante, ahora empiezan a sonarnos, como HSDPA, que fue presentada en el

Introducción a .NET Compact Framework 2.0 # Anatomía de un dispositivo móvil
3GSM de Barcelona del año 2006. De manera muy superficial, y en cuanto a comunicaciones de telefonía móvil se refiere, se catalogan todas estas siglas en una línea temporal desde 0G (allá por los años 40) hasta 4G, pasando por 2G (GSM), 2.5G (GPRS), 3G(W-CDMA), 3.5G(HSDPA) y 3.75G(HSUPA). En fin, te dejo este enlace si quieres empaparte del tema:

<http://es.wikipedia.org/wiki/GPRS>

Servicios de cobertura 3G y GSM(1800/900) en España:

http://www.gsmworld.com/roaming/gsminfo/cou_es.shtml

No me atrevo a profundizar más(*), por ahora, en este aspecto. Quizás requiera de un documento entero. Una reflexión/experiencia en el tema que tuve hace tiempo, más concretamente sobre Bluetooth, la podéis encontrar en forma de artículo en la revista dotNetManía (nº35) y en el site

<http://www.desarrollomobile.net/libreria/libreria.aspx>.

Ah! Olvida compartir un enlace curioso respecto a IrDA. Se trata de un artículo de cómo utilizando IrDA podemos utilizar nuestra PDA de mando a distancia para el televisor... <http://www.desarrollomobile.net/explore2.aspx?item=28>

() Aunque no me atreva a profundizar por lo abstracto del tema, no puedo obviarlo; al fin y al cabo hablamos de Anatomía de un dispositivo móvil*

Complementos

Muchas veces tendemos a pensar lo siguiente: *en mi pc de desarrollo tengo el Visual Studio 2005 .NET con .net compact framework 2.0; además tengo una pda, con cámara, gps y lector de huellas dactilares. ¿Podré utilizar todos estos complementos desde una aplicación .net compact framework mía?*

La respuesta es **todos no**. Bueno sí, me explico (que lío). Si recuerdas, en el primer punto de este mismo documento hablé sobre aspectos del hardware a tener en cuenta, si un fabricante pone si otro quita, etc. Pues esto es lo mismo. ¿Podemos utilizar la cámara de fotos desde nuestra aplicación .net cf? Pues sí, pero necesitamos de un SDK, de una guía, de “algo” (sean librerías administradas, librerías nativas,...) que nos diga el qué y cómo hacerlo. **Todos** los fabricantes de dispositivos móviles tienen su SDK, otra cosa es que la forma de obtenerlo sea mediante pago (lamentablemente muy popular) o gratuito. Sin embargo, el código que utilices para utilizar la cámara de tu PDA no te servirá (lo mas probable) en otro tipo de PDA (me refiero a marca y modelo).

El tema del GPS ha sido solucionado por Microsoft. Lo que han hecho ha sido crear una capa intermedia que aísla el tipo y conectividad del GPS, retransmitiendo sólo los datos (secuencias NMEA ya estructuradas) en forma estructuras nativas. La mala noticia es que estas funciones y estructuras son nativas (en C++). De todas formas en el SDK de Windows Mobile 5.0 y superior encontrarás un ejemplo de utilización.

Setting up GPS on Windows Mobile 5.0

<http://blogs.msdn.com/windowsmobile/archive/2006/06/07/620387.aspx>

GPS Intermediate Driver

<http://msdn2.microsoft.com/en-us/library/ms850332.aspx>



Introducción a .NET Compact Framework 2.0 # Anatomía de una dispositivo móvil

Otros enlaces:

Peculiaridades del uso de menus en Smartphone

<http://jmtorres.blogspot.com/2006/09/peculiaridades-del-uso-de-mens-en.html>

¿eres cirujano plástico.net?

<http://jmtorres.blogspot.com/2007/06/eres-cirujano-plsticonet.html>

SD Memory

<http://blogs.msdn.com/windowsmobile/archive/2007/01/12/everything-you-want-to-know-about-sd.aspx>

Próxima: <<En desarrollo>>

Más consultas contactar en:

jtorres_diaz@terra.es?subject=Introduccion%20a%20CF%203