

Embarcadero RAD Studio XE2, Las nuevas fronteras de la programación

Dirigido a Equipos de desarrollo | Luis Alfonso Rey | sp@danysoft.com

Quien haya trabajado alguna vez con Delphi, C++ Builder o RAD Studio no podrá negar que son ambientes de trabajo agradables y productivos, pero podíamos pensar que ya no tecnológicamente punteros, en comparación con los lenguajes manejados. Sin embargo Embarcadero está dispuesto a hacernos cambiar de idea con RAD Studio XE2...

Estamos acostumbrados a número de modificaciones determinado incluidas con cada nueva versión de Delphi, C++ Builder o RAD Studio, dada la estrategia de lanzamientos anuales que primero Borland y luego Embarcadero han seguido. Esto se ha hecho así para ir adoptando las nuevas mejoras en un plazo de tiempo menor, pero si las mejoras no estaban relacionadas con nuestro trabajo directo, podía haber menos razones para actualizar a una versión concreta.

Los cambios que son populares para algunos porque les afectan, no lo son tanto para otros porque no lo hacen. Incluso ha habido quien ha tenido que adaptarse a cambios que aparentemente no le afectan, como en el caso de Unicode, que sin embargo, son fundamentales para otros.

Sin embargo todo esto ha tenido su recompensa. Esta es el nuevo conjunto de tecnologías que acompañan a XE2, son tan grandes y profundas que por primera vez ocultan otras más pequeñas por falta de espacio y/o tiempo. Ser exhaustivos con las modificaciones acometidas en esta versión es tarea casi imposible, así que intentaremos centrarnos en lo importante dejándolo lo más claro posible.

FireMonkey no es solo gráficos

Cuando hablamos de FireMonkey podemos hablar de una plataforma de diseño de interfaces avanzados, que incluyen 3D o HD y multiplataforma, puede quedar muy simplista, nada más lejos de la realidad.

FireMonkey nos permite crear aplicaciones en tres plataformas diferentes:

- Windows (32bit y 64bit).
- Mac OS.
- e IOS que es el sistema operativo de iPad e iPhone.



En principio los proyectos Windows y MacOS son 100% compatibles y en los de IOS tenemos una serie de recortes propios de una plataforma más limitada. Se incluyen dos tipos de proyectos, uno 3D y otro HD, lo que nos permite definir con exactitud qué tipo de proyecto se adapta a nuestra necesidad. Además esta plataforma hace uso intensivo



del hardware grafico disponible (GPU), ya que se sitúa como una delgada capa sobre tecnologías como GDI+, DirectX o OpenGL. Tecnologías todas ellas accesibles previamente desde Delphi o C++, pero de manera sumamente complicada, poco productiva y muy bajo nivel.

Sin embargo hay que tener en cuenta un par de cosas. Por un lado FireMonkey no sustituye a la VCL. Todos, y digo bien, todos los proyectos anteriores siguen ahí y ninguno ha sido sustituido por razón de FireMonkey, es más, algunos han sufrido evoluciones importantes como

veremos a continuación.

FireMonkey es una plataforma de desarrollo que solo implica presentación, esto significa que puede ser combinada con cualquier tecnología Delphi excepto obviamente las visuales.

FireMonkey puede ser una plataforma para desarrollo de



aplicaciones de datos que nos permita desde crear efectos visuales más atractivos para vender mejor nuestro producto, hasta crear un nuevo interfaz de aplicaciones del siglo XXI.



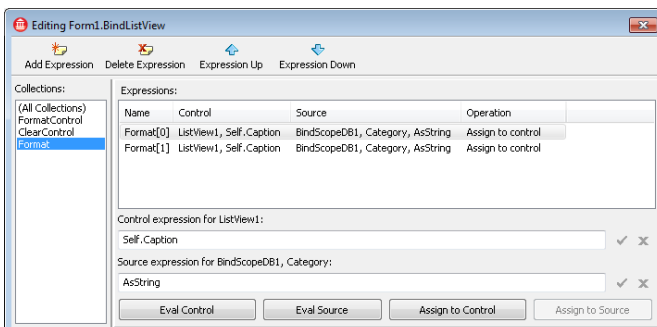
El único punto débil es el enlace a datos ya que FireMonkey no es compatible con la VCL, por tanto, ¿qué podemos hacer? En Embarcadero han pensado en todo y han creado los LiveBindings para solucionar este aspecto.

LiveBindings

La concepción de enlace a datos de Delphi inicialmente, fue genial. Componentes arrastrables que representaban tablas o consultas, cursores, que se conectaban con controles mediante un *TDataSource* y se manipulaban mediante una simple interfaz de métodos como *First*, *Next*, *Prior*,...

Sin embargo los problemas rápidamente aparecieron. El primero y más evidente fue el rendimiento. Gestionar la información de esa manera tiene un coste muy importante a nivel de capacidad de proceso y movimiento de datos, incrementado seriamente en aplicaciones cliente servidor. Por tanto desde las primeras versiones Delphi incorporaba

tecnologías de acceso “optimizado” a datos. La sublimación de estos mecanismos fue DBX4, a la vez un acceso de alto rendimiento y la madre de tecnologías tan importantes como DataSnap, en conjunción con *TClientDataSet*.



Pero existen otros problemas relacionados con el enlace a datos

que estaban asfixiando el desarrollo de Delphi como lenguaje y la evolución dentro del paradigma de la programación orientada a objetos.

Pongamos por caso que siguiendo un análisis exhaustivo creamos una jerarquía de clases en nuestro programa. Dichas clases estarán encargadas de leer datos de la base de datos y mostrarlos en las ventanas de nuestra aplicación al tiempo que recogen los nuevos datos y los almacenan de vuelta en la base de datos.

Este esquema, conocido por desarrollo por capas, está insinuando un modelo extremadamente difícil de implementar en Delphi y sugiere repeticiones en las lecturas de los datos en la base de datos, ¿cuáles? Muy sencillo si tengo una clase que lee la base de datos y ya contiene los datos de esta, ¿para qué voy a poner un dataset que vuelva a leer los datos para mostrarlos en componentes?, es más ¿cómo voy luego a sincronizar ambos orígenes de datos?

No tenía fácil respuesta esta pregunta, lo que normalmente hacíamos, o bien realizábamos la extracción de los datos de la clase y la inserción de estos en los controles mediante interminables listas de asignaciones por código, o bien prescindíamos de la jerarquía de objetos limitando nuestra arquitectura de software a lo de siempre, arrastro control y suelto. Otras soluciones, tecnológicamente más complejas podían incluir el desarrollo de datasets que estuvieran diseñados para mostrar listas de objetos, pero estas eran sensiblemente más complicadas y difíciles de depurar.

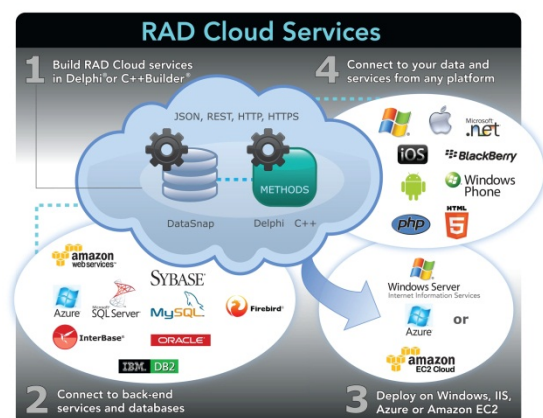
Los LiveBindings vienen a llenar este agujero. Para ello se crea una serie de componentes que nos permiten enlazar cualquier propiedad de cualquier objeto con cualquiera otra de otro, todo ello de manera visual, y sin renunciar por ello a un enlace a datos directamente contra un dataset, ya que se sigue pudiendo usar el modelo tradicional, aunque también existe soporte en los LiveBindings para realizar esta tarea.

Esto tiene dos grandes implicaciones directas. La primera es que ya no estamos limitados a los componentes *DBAware* para que muestren los datos de nuestras aplicaciones, la segunda es que ya no solo la propiedad Text o Caption es la receptora de los datos, ahora también podemos sincronizar propiedades de color, anchura, altura, etc.

Pero efecto colateral importante, y que merece la pena nombrar por las implicaciones de futuro que tiene es la posibilidad de sincronizar elementos no visuales, creando modelos de sincronización muy simples y a la vez poderosos.

DataSnap y los Mobile Connectors

Otra de las novedades y ésta ya casi un clásico, es la relativa a DataSnap. Desde que en la versión 2009 apareciera, no ha habido una versión que no incluya modificaciones. En el caso de XE2 son muchas, mejora en el proceso de los *callbacks*, mejora en los procesos de seguridad, inclusión del protocolo https,... Sin embargo hay dos de importancia notable, la primera la capacidad de



monitorización de la actividad y las sesiones, la segunda es tan importante que Embarcadero la trata como una modificación separada como si nada tuviera que ver con DataSnap, que son los Mobile Connectors.

Estos conectores sirven para conectar casi desde cualquier plataforma con un servidor DataSnap, en principio parece que solo basado en REST. Para ello solo tenemos que solicitar al servidor que nos “genere” las clases para que nuestra aplicación consuma sus funcionalidades expuestas, y es aquí donde viene lo más interesante, sea nuestro programa escrito en Delphi y C++ o en Java. El número de plataformas no está cerrado pero parece que incluirá IOS (C++), Android (Java), Window7(C# y Embarcadero Prism), Blackberry, Windows y Mac.

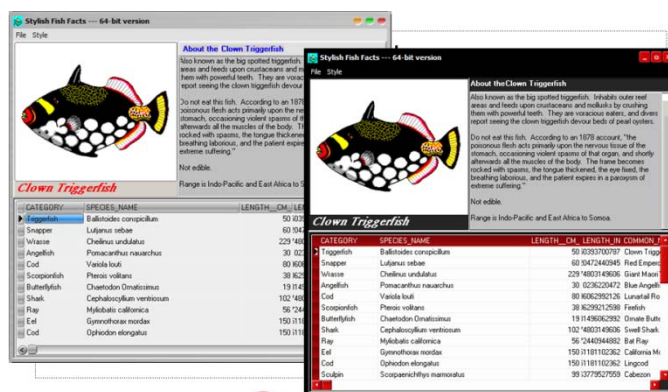


Otras modificaciones

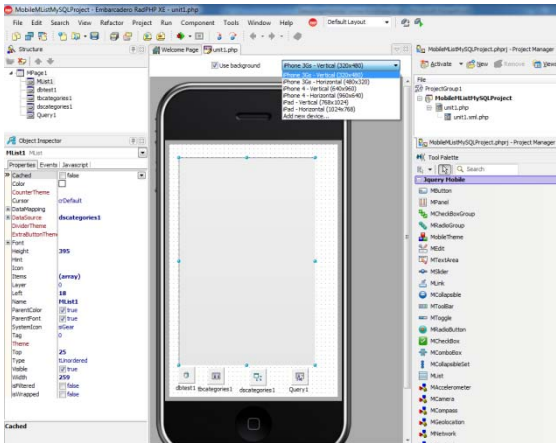
La lista de modificaciones parece interminable en esta versión y no me gustaría dejarme en el tintero alguna, pero aun a riesgo de ello voy a tratar de enumerar las más importantes dentro de las no comentadas ya, para por lo menos dar una idea de la amplitud que tiene esta nueva versión.

El compilador y plataforma completos para Windows 64-bit, comentado como de pasada con anterioridad, supone “de facto” realizar aplicaciones para Windows 64Bit. Inicialmente está reservado solo a Delphi (aunque parece que en el horizonte se encuentra C++), y para los sistemas operativos Windows Vista y 7 de 64bit. Aumentando de manera dramática la cantidad de memoria que nuestra aplicación puede usar, al aumentar también el rango de direccionamiento.

Los estilos VCL personalizables que complementan los estilos de FireMonkey, y que nos permiten variar el “Look’n’Feel” de nuestra aplicación sin recompilar y con tan solo variar un fichero, que además podemos editar con un nuevo editor desarrollado al efecto.



No olvidar también novedades en el desarrollo para la Nube incluyendo soporte en esta versión para los servicios de datos de Amazon EC2 como en su día se hizo para Azure, incluyendo además una nueva herramienta de despliegues.



Fuera de Delphi y C++ advertir también que RAD PHP trae también importantes mejoras, entre las que destaca el desarrollo web para móviles, y una interesante herramienta para convertir estas aplicaciones en aplicaciones nativas llamada Phone Gap. No puedo ser más específico porque todavía no la he podido ver en profundidad...

Importante es recalcar también la inclusión de dos nuevas herramientas, el conocidísimo generador de informes FastReport, casi un

estándar que complementa, no sustituye, a Rave Reports y la herramienta de generación de documentación Insight doc.

Conclusión

Muchas versiones hemos visto, algunas ejecutadas con más acierto que otras, pero nunca en la historia reciente (y en toda ella nos tendríamos que remontar a las épocas de Delphi 3 ó 4) hemos tenido una versión que prometa tantos cambios al mismo tiempo, y abra tantas puertas al futuro como RAD Studio XE2.

Para más información le rogamos contacte Danysoft en el 902 123146, o visitando www.danysoft.com | Danysoft es el representante exclusivo en la península ibérica de las soluciones de Embarcadero technologies.