GUIÓN WORKSHOP XAMARIN Y AZURE

1.Presentación

-Contar un poco de mí (estudios, a qué me dedico, contacto…).

-Xamarin y Azure:

-Xamarin: es una herramienta que permite hacer una aplicación multiplataforma usando un único código en **C#**. Nos permite la reutilización del código.

-Azure: es el sistema operativo en la nube de **Microsoft**. Proporciona un entorno gestionado para la ejecución y el despliegue de aplicaciones y servicios en la nube. **Windows Azure** proporciona a los desarrolladores un entorno de computación bajo demanda y almacenamiento alojado en los centros de datos de Microsoft para aplicaciones en la web.

-UNIA y Club.Net:

-**Webs:** gracias a **ASP.NET MVC*** se pueden crear webs de alta calidad y de una forma cómoda gracias a **Visual Studio**.

*El modelo arquitectónico Modelo-Vista-Controlador (MVC) separa una aplicación en tres componentes principales: el modelo, la vista y el controlador. El marco de ASP.NET MVC proporciona una alternativa al modelo de formularios Web Forms de ASP.NET para crear aplicaciones web. El marco de ASP.NET MVC es un marco de presentación de poca complejidad y fácil de comprobar que (como las aplicaciones basadas en formularios Web Forms) se integra con las características de ASP.NET existentes, tales como páginas maestras y la autenticación basada en pertenencia. El marco de MVC se define en el ensamblado System.Web.Mvc.

-Programas: con **WPF*** se pueden crear programas de escritorio mediante **C#** y **XAML*** (por ejemplo) en un entorno **.NET**.

*Windows Presentation Foundation (WPF) es un subsistema de presentación unificado para Windows, expuesto mediante WinFX, el modelo de programación de código administrado para Windows Vista que amplía Microsoft .NET Framework. WPF se compone de un motor de visualización y un marco de código administrado. WPF unifica la forma en que Windows crea, muestra y manipula documentos, elementos multimedia e interfaces de usuario (UI), lo que permite a programadores y diseñadores crear experiencias de usuario diferenciadas y visualmente sorprendentes.

*El lenguaje de marcado de aplicaciones extensibles, o XAML, es un lenguaje de marcado basado en XML desarrollado por Microsoft. XAML es el lenguaje que subyace a la presentación visual de una aplicación desarrollada en Microsoft Expression Blend, al igual que HTML es el lenguaje que subyace la presentación visual de una página web. La creación de una aplicación en Expression Blend supone tener que escribir código XAML, ya sea de forma manual o visual, mediante la vista Diseño de Expression Blend.

-**Aplicaciones multiplataforma:** con **Xamarin** y **Visual Studio** se pueden crear aplicaciones para **Android**, **IOS** y **Windows**.

-**Juegos:** hay muchos entornos para crear videojuegos pero destaca **Unity** en el que podemos programar con **C#.**

-**Servicios en el cloud:** mediante **.NET** y **Azure** podemos trabajar y aprender sobre los servicios cloud.

-**IT:** por supuesto en este entorno contamos con muchas herramientas para aprender sobre IT y Azure.

2.Proyecto

-¿Qué vamos a hacer?

Hoy crearemos una aplicación multiplataforma gracias a **Xamarin** conectada a la nube utilizando **Azure** que nos mostrará una lista de los conferenciantes del **Dev Days** con sus detalles (Avatar, nombre, descripción…).

-Manos a la obra:

-Abrimos la **solución**.

-Lo primero es restaurar los paquetes de **NuGet***, para ello tenemos que hacer **click derecho en la solución** y seleccionamos **restaurar paquetes de NuGet**.

*Es un manejador de paquetes que permite instalar y actualizar librerías y herramientas en Visual Studio 2010. Está basado en una extensión la cual se acopla a Visual Studio y es instalada desde el Extension Gallery, agregando funcionalidades de línea de comando (en powershell) e interfaz gráfica para realizar la búsqueda, instalación y actualización de paquetes.

-**Speaker.cs:**

-Añadimos las **propiedades**, que será la información del conferenciante del Dev Days que nos mostrará la aplicación.

-**SpeakersViewModel.cs:**

SpeakerViewModel.cs proporcionará toda la funcionalidad de cómo nuestra vista principal de **Xamarin.Forms** mostrará los datos. Consistirá en una lista de conferenciantes y un método que se puede llamar para obtener los oradores del servidor. También contendrá una booleana que indicará si estamos recibiendo datos en una tarea de fondo.

-Implementamos la interfaz **INotifyPropertyChanged.** Esto nos permitirá saber cuándo se cambia una propiedad.

-Creamos el evento y un método que se invoque cuando cambie una propiedad.

-IsBusy: creamos un método que nos dirá si la aplicación está cogiendo datos del servidor para que no podamos realizar varias operaciones a la vez. La interfaz nos notificará el cambio.

-ObservableCollection of Speaker: usaremos una **ObservableCollection** porque tiene soporte incorporado para eventos **CollectionChanged** cuando agregamos o quitamos de él. Esto es muy útil y así no tenemos que llamar a **OnPropertyChanged** cada vez.

-GetSpeakers Method: creamos el método de tipo **async Task (**Las tareas asíncronas se hacen necesarias cuando las aplicaciones se quedan congeladas debido a acciones que no terminan y no dejan que el usuario siga utilizándolas**)**.
Explicar el **try/catch/finally,** lo que es **JSON** (**JavaScript Object Notation** es un formato para el intercambios de datos, básicamente JSON describe los datos con una sintaxis dedicada que se usa para identificar y gestionar los datos. JSON nació como una **alternativa a XML**, el fácil uso en **javascript** ha generado un gran número de seguidores de esta alternativa. Una de las mayores ventajas que tiene el uso de JSON es que puede ser leído por **cualquier lenguaje de programación**. Por lo tanto, puede ser usado para el intercambio de información entre distintas tecnologías) y cómo funciona el **foreach**.

-GetSpeakers Command: explicar las **expresiones lambda (**Una expresión lambda es una función anónima que se puede usar para crear tipos delegados o de árbol de expresión. Al utilizar expresiones lambda, puede escribir funciones locales que se pueden pasar como argumentos o devolverse como valor de llamadas de función. Las expresiones lambda son especialmente útiles para escribir expresiones de consulta LINQ. Para crear una expresión lambda, especifique los parámetros de entrada (si existen) a la izquierda del operador lambda => y coloque el bloque de expresiones o de instrucciones en el otro lado).

-SpeakerPage.xaml:

Explicar todo lo que puedan preguntar (parte fácil, en las diapositivas está más o menos la explicación).

Explicar cómo funciona **x:Name** (Identifica de manera única elementos definidos por XAML en un ámbito de nombres XAML. Los ámbitos de nombres XAML y sus modelos de unicidad se pueden aplicar a los objetos con instancias, cuando los marcos de trabajo proporcionan las API o implementan comportamientos de acceso al gráfico de objetos creado por XAML en tiempo de ejecución).

Para determinar la apariencia de cada objeto usamos un **Item.Template**. Explicar lo que hay dentro de **Item.Template** (**Células** predeterminadas de **Xamarin.Forms**).

-Primera depuración: Explicar dónde solucionar los posibles errores para **Android:**

-aapt.exe exited with code o **Unsupported major.minor version 52:** El Java SDK puede no haberse configurado correctamente o tienes herramientas más recientes de las que soporta.

Consultar: **https://releases.xamarin.com/technical-bulletin-android-sdk-build-tools-24/**

O también: **http://motzcod.es/post/149717060272/fix-for-unsupported-majorminor-version-520**

-Si tienes problemas con los paquetes de Android consultar:

https://xamarinhelp.com/debugging-xamarin-android-build-and-deployment-failures/

-SpeakersPage.xaml.cs:

En el primer método comprobamos que el objeto seleccionado no es **null** y si no lo es abrimos otra página y deseleccionamos el objeto.

-DetailsPage.xaml:

Es parecido al **SpeakersPage.xaml** solo que introducimos el **StackLayout** en un **ScrollView** por si tenemos un texto muy largo.

Incluimos los controles y enlaces para las propiedades del conferenciante.

- DetailsPage.xaml.cs:

Añadimos el plugin **Text To Speech** para que la aplicación lea la descripción.

Explicar cómo abrir una URL con el buscador predeterminado (**OpenUri**).

-Conectar a Azure:

Si preguntan explicar lo que es el **back-end** (El programador back-end es aquel que se encuentra del lado del servidor, es decir, esta persona se encarga de lenguajes como PHP, Python, .Net, Java, etc, es aquel que se encarga de interactuar con bases de datos, verificar manejos de sesiones de usuarios, montar la página en un servidor, y desde este "servir" todas las vistas que el front-end crea, es decir, uno como back-end se encarga de la manipulación de los datos).

Seguir los pasos de las diapositivas. Como tarda en prepararse el **Quickstart** volvemos al código.

En cuanto a la parte de código explicar lo que son las **consultas LINQ** (Una *consulta* es una expresión que recupera datos de un origen de datos. Las consultas normalmente se expresan en un lenguaje de consultas especializado. A lo largo del tiempo se han ido desarrollando lenguajes diferentes para los distintos tipos de orígenes de datos, como SQL para las bases de datos relacionales y XQuery para XML. Por tanto, los desarrolladores han tenido que aprender un nuevo lenguaje de consulta para cada tipo de origen de datos o formato de datos que deben usar. LINQ simplifica esta situación al proporcionar un modelo coherente para trabajar con los datos de varios tipos de formatos y orígenes de datos. En una consulta LINQ, siempre se trabaja con objetos. Se utilizan los mismos modelos de codificación básicos para consultar y transformar datos de documentos XML, bases de datos SQL, conjuntos de datos ADO.NET, colecciones .NET y cualquier otro formato para el que haya disponible un proveedor LINQ).

Por último actualizar el **SpeakersViewModel.cs** para hacer una consulta en la tabla creada en vez de usar el **HttpClient** para obtener un **string**.

Terminar siguiendo las diapositivas de configurar **Azure.**

3. Final del taller

-Bonus challenges:

-Recordar los ejercicios extras para hacer en casa que hay al final de la página de los materiales.

-Dudas y fin.