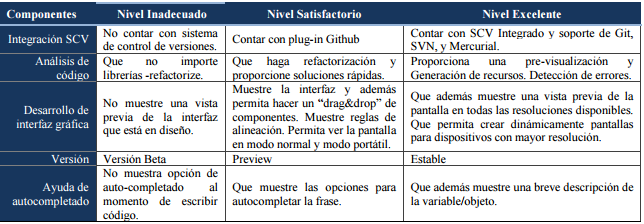
Herramientas y Lenguajes

Para la elección de la herramienta y lenguaje de programación se determinó las mejores prácticas de codificación, es decir el Entorno de Desarrollo Integrado (IDE). Se optó por utilizar la herramienta Microsoft .NET con Visual Studio y Xamarin Studio ya que presenta los mejores beneficios para el proyecto a desarrollar esto se determinó luego de haber realizado una investigación de los IDE existentes para el desarrollo de aplicaciones en dispositivos móviles con sistema operativo Android los cuales son los siguientes:

1. Eclipse.
2. NetBeans.
3. IntelliJ IDEA.
4. Microsoft .NET con Visual Studio y Xamarin Studio.

A estos a se les aplicó los siguientes criterios de evaluación



# Herramientas

**a) Eclipse:**

La definición que da el proyecto Eclipse (IBM, 2001) acerca de su software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular". También se lo define como un [programa informático](http://es.wikipedia.org/wiki/Programa_inform%C3%A1tico) compuesto por un conjunto de herramientas de programación de [código abierto](http://es.wikipedia.org/wiki/C%C3%B3digo_abierto) multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma, típicamente ha sido usada para desarrollar [entornos de desarrollo integrados](http://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado) (del inglés IDE)

El [entorno de desarrollo integrado](http://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado) (IDE) de Eclipse (Fundacion Wikimedia, 2015) emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la plataforma de cliente enriquecido, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software.

En cuanto a las aplicaciones clientes, Eclipse le provee al programador “[frameworks](http://es.wikipedia.org/wiki/Framework)” (marcos de trabajos) muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc.

El SDK (*kit de desarrollo de software)*de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. Mediante diversos plugins estas herramientas están también disponibles para otros lenguajes como C/C++ (Eclipse CDT) y en la medida de lo posible para lenguajes de script no tipados como PHP o Javascript. El IDE también hace uso de un espacio de trabajo, en este caso un grupo de [metadatos](http://es.wikipedia.org/wiki/Metadatos) en un espacio para archivos planos, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente.

**Características de Eclipse**

* Perspectivas, editores y vistas: en Eclipse el concepto de trabajo está basado en las perspectivas, que no es otra cosa que una preconfiguración de ventanas y editores, relacionadas entre sí, y que permiten trabajar en un determinado entorno de trabajo de forma óptima.
* Gestión de proyectos: el desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como puede ser el código fuente, documentación, ficheros configuración, árbol de directorios. El IDE proporcionará asistentes y ayudas para la creación de proyectos.
* Depurador de código: se incluye un potente depurador, de uso fácil e intuitivo, y que visualmente ayuda a mejorar el código, donde se muestra de forma ordenada toda la información necesaria para realizar dicha tarea.
* Extensa colección de plug-ins: están disponibles en una gran cantidad, unos publicados por Eclipse, otros por terceros, los hay gratuitos, de pago y bajo distintas licencias.

**Ventajas en la utilización de Eclipse**

* El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la [Plataforma](http://www.ecured.cu/index.php/Plataforma) de [Cliente](http://www.ecured.cu/index.php/Cliente) rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.
* Eclipse puede extenderse usando otros lenguajes de programación como son C/[C++](http://www.ecured.cu/index.php/C%2B%2B) y [Python](http://www.ecured.cu/index.php/Python), también trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos.
* La arquitectura plug-in permite escribir cualquier extensión deseada en el ambiente, como sería Gestión de la configuración.
* El SDK (kit de desarrollo de software) de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código.
* El IDE también hace uso de un espacio de trabajo, en este caso un grupo de metadata en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente.

**Desventajas en la utilización de Eclipse:**

* Su principal inconveniente, es el consumo de recursos del sistema.
* Carece de mucho soporte para webapps (.war, jsp y servlets).
* Los plugin no son muy potentes y son difíciles de usar.

**b) NetBeans**

Es un [entorno de desarrollo integrado](http://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado) [libre](http://es.wikipedia.org/wiki/Software_libre), hecho principalmente para el [lenguaje de programación](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) [Java](http://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)). NetBeans es un IDE que permite de forma rápida y fácil desarrollar aplicaciones Java de escritorio, móviles y aplicaciones web, utilizando tecnologías y lenguajes como HTML5 con HTML, Javascript y CSS. Proporciona un conjunto de herramientas para PHP, C y C++.

**Características:**

* Asistentes y Gestor de Proyectos: Cuenta con asistentes para configuración de distintos proyectos y selección de frameworks. Simplifica la gestión de proyectos con el uso de vistas, asistentes de ayuda y estructurando la visualización de información de manera ordenada.
* Editor de Código Multilenguaje: Sugerencias de código, coloreado de código, acceso a clases con tan solo un click control de versiones son algunas de las características del editor de NetBeans.
* Depurador de Código: Cuenta con un depurador muy útil, entre otras cosas se puede monitorizar en tiempo real los valores de las propiedades y variables, además de poder conectarse al debugger incluso cuando hay un proceso en ejecución.
* Acceso a Base de Datos y PlugIns: Desde el propio IDE se puede conectar a distintos gestores de base de datos y consultar tablas y datos.

**Ventajas**

* La plataforma Netbeans puede ser usada para desarrollar cualquier tipo de aplicación.
* Reutilización de Módulos.
* Permite el uso de la herramienta Update Center Module.
* Instalación y actualización simple.
* Incluye Templates y Wizards.
* Posee soporte para Php.

**Desventajas**

* Poca existencia de plugins para esta plataforma.
* Muy pesado.
* Hace falta documentación del Rich Client Plataform (RCP).
* No posee un editor de código HTML.
* Es complicado correr los Programas.
* Su Entorno no es muy amigable.

**c) IntelliJ IDEA**

Los creadores de IntelliJ IDEA (JetBrains, 2011) definen el mismo como un entorno inteligente para desarrollar aplicaciones Java, cliente y servidor, también para desarrollar aplicaciones para móviles (J2ME) y cuenta con un potente editor de código, compatible las tecnologías AJAX, JSP, EJB.

**Características:**

* Integración de Perforce, Rational ClearCase y Microsoft Team Foundation Server.
* Dependencia de la matriz de estructura.
* Manipulación de códigos avanzada con búsqueda estructural y sustitución.
* Refactorización y depuración JavaScript
* Compatibilidad completa con Java
* Compatibilidad con Groovy 1.8 y Spring 3.1
* Mejoras JavaScript, Android y Flex
* Integración Jetty
* Compatibilidad XSLT2
* Diseñador de interfaces gráficas para Android.

**Ventajas:**

* Autocompletado de código.
* Integración con sistemas de control de versiones.
* Viene con todo instalado de serie, existe un amplio set de plugins.
* Sensación de fiabilidad y robustez.
* Herramienta de refactorización extremadamente inteligente.

**Inconvenientes:**

* Es pago.
* Consume muchos recursos.
* Se torna lento a la hora de compilar los archivos.

**d) Microsoft .NET con Xamarin y con Visual Studio**

Microsoft .NET (Microsoft, 2009) es un framework que hace énfasis en la transparencia de [redes](http://es.wikipedia.org/wiki/Red_de_computadoras), con independencia de plataforma de [hardware](http://es.wikipedia.org/wiki/Hardware) y permite un rápido desarrollo de [aplicaciones](http://es.wikipedia.org/wiki/Aplicaci%C3%B3n_inform%C3%A1tica).

Los principales componentes del marco de trabajo son:

* El conjunto de [lenguajes de programación](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n).
* La biblioteca de clases base o BCL.
* El entorno común de ejecución para lenguajes, o CLR por sus siglas en inglés.

Microsoft .NET soporta más de 20 lenguajes de programación y es posible desarrollar cualquiera de los tipos de aplicaciones soportados en la plataforma con cualquiera de ellos, lo que elimina las diferencias que existían entre lo que era posible hacer con uno u otro lenguaje.

Algunos de los lenguajes desarrollados para el marco de trabajo .NET son: [C#](http://es.wikipedia.org/wiki/C_Sharp), [Visual Basic .NET](http://es.wikipedia.org/wiki/Visual_Basic_.NET), [Delphi](http://es.wikipedia.org/wiki/Delphi) ([Object Pascal](http://es.wikipedia.org/wiki/Object_Pascal)), [C++](http://es.wikipedia.org/wiki/C%2B%2B), [F#](http://es.wikipedia.org/wiki/F_Sharp), [J#](http://es.wikipedia.org/wiki/J_Sharp), [Perl](http://es.wikipedia.org/wiki/Perl), [Python](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_Python), [Fortran](http://es.wikipedia.org/wiki/Fortran), [Prolog](http://es.wikipedia.org/wiki/Prolog) (existen al menos dos implementaciones, el [P#](http://es.wikipedia.org/wiki/P_Sharp)[1](http://es.wikipedia.org/wiki/Microsoft_.NET#cite_note-1) y el [Prolog.NET](http://es.wikipedia.org/w/index.php?title=Prolog.NET&action=edit&redlink=1)[2](http://es.wikipedia.org/wiki/Microsoft_.NET#cite_note-2) ), [Cobol](http://es.wikipedia.org/wiki/Cobol) y [PowerBuilder](http://es.wikipedia.org/wiki/PowerBuilder)

**Características:**

El Common Language Runtime (CLR) es el verdadero núcleo del framework de .NET, entorno de ejecución en el que se cargan las aplicaciones desarrolladas en los distintos lenguajes, ampliando el conjunto de servicios del sistema operativo. Permite integrar proyectos en distintos lenguajes soportados por la plataforma .Net, como C++, Visual Basic, C#, entre otros.

La herramienta de desarrollo compila el código fuente de cualquiera de los lenguajes soportados por .NET en un código intermedio, el [CIL](http://es.wikipedia.org/wiki/Common_Intermediate_Language) (Common Intermediate Language) antes conocido como [MSIL](http://es.wikipedia.org/wiki/MSIL) (Microsoft Intermediate Language). Para generarlo, el compilador se basa en la especificación [CLS](http://es.wikipedia.org/w/index.php?title=CLS&action=edit&redlink=1) (Common Language Specification) que determina las reglas necesarias para crear el código MSIL compatible con el CLR.

Para ejecutarse se necesita un segundo paso, un compilador [JIT](http://es.wikipedia.org/wiki/Compilaci%C3%B3n_en_tiempo_de_ejecuci%C3%B3n) (Just-In-Time) es el que genera el [código máquina](http://es.wikipedia.org/wiki/C%C3%B3digo_m%C3%A1quina) real que se ejecuta en la plataforma del cliente. De esta forma se consigue con .NET independencia de la plataforma de hardware. La compilación JIT la realiza el CLR a medida que el programa invoca métodos. El código ejecutable obtenido se almacena en la memoria caché del ordenador, siendo recompilado de nuevo sólo en el caso de producirse algún cambio en el código fuente.



Ilustración Estructura interna del entorno de ejecución en lenguaje común.

Mono for Android: Mono es la implementación de código abierto (open source) de la plataforma .NET sobre Windows y Linux. En el caso de Android, Mono ejecuta sobre el Kernel de Linux, a la par de Dalvik, que es la máquina virtual de Java que soporta todo el ecosistema Android.

Mono permite crear, depurar y empaquetar una aplicación en un .apk y utilizarla directamente en un teléfono Android. A diferencia de otros productos, Mono for Android permite el desarrollo de aplicaciones nativas, significando que se integra al ecosistema Android e interactúa con las aplicaciones nativas creadas en Java, utiliza la interfaz (UI) nativa del sistema operativo y sigue el modelo de desarrollo de Android. Intenta sustituir Java por C#, dejando el resto de los componentes intactos, de forma tal que la interfaz de usuario (UI) se diseñe de igual manera y la aplicación desarrollada se integre al ecosistema Android como si fuese desarrollada propiamente en Java. Además, las herramientas de desarrollo se integran con Visual Studio, brindándole a un desarrollador .NET, la gran ventaja de desarrollar una aplicación Android sin salirse del entorno de desarrollo conocido y utilizando el lenguaje C# que domina.

**Xamarin Studio**

Según la compañía Xamarin (Proyecto Mono, 2011) el software es un CrossPlatform (programa que puede utilizarse en varias plataformas) que permite escribir aplicaciones móviles tanto para Android como para iOS y Windows Phone, y consiste en una implementación libre de la plataforma de desarrollo .NET para dispositivos Android, iOS y GNU/Linux. Es decir, con Xamarin se [evita tener que utilizar Java](http://www.elandroidelibre.com/2012/05/es-java-en-android-tan-importante-como-parece.html) para desarrollar una aplicación para Android. También permite generar aplicación para iOS (.APP) y para Android (.APK), la cual correrá de forma nativa. Gracias a esto, surge una de las grandes ventajas de Xamarin: la reutilización de código. Al poder desarrollar aplicaciones nativas para Windows Phone, Windows 8 y web en C# y .NET, resulta que la reutilización del código se exporta a aún más plataformas que iOS o Android.

Xamarin Studio es una versión personalizada para desarrollo móvil de MonoDevelop, que es el IDE oficial de Mono. Mono for Android, cuenta con una extensión para Visual Studio que habilita la compilación y la depuración de aplicaciones Android. El mismo proyecto/solución sirve tanto en Visual Studio como Xamarin Studio y se puede utilizar indistintamente tanto uno como el otro, ya que MonoDevelop, en el cual se basa Xamarin Studio, utiliza el mismo formato (.sln y .csproj) que Visual Studio.

**Características de Xamarin**

* Lenguaje de Programación C#: Permite utilizar una sintaxis conocida ya por muchos desarrolladores e integra características sofisticadas como Generics, Linq y Parallel Task.
* Framework Mono .NET: Proporciona una implementación multiplataforma utilizando todas las características con las que cuenta Microsoft .NET.
* Compilador: Dependiendo de la plataforma seleccionada produce una aplicación nativa o una aplicación .NET integrada.
* Herramientas IDE: Cuenta con el IDE de Xamarin Studio y el plug-in de Xamarin para integración con Visual Studio.
* Soporte de multiplataforma móvil: Xamarin ofrece soporte multiplataforma sofisticado para las tres principales plataformas móviles: iOS, Android y Windows Phone. Las aplicaciones pueden ser escritas para compartir hasta el 75% del código entre plataformas. Además, Xamarin ofrece acceso a los recursos comunes de las 3 plataformas.

**Ventajas**

* Permite utilizar C# y .NET Framework.
* Cuenta con su propio IDE llamado Xamarin Studio que cuenta con versión para MAC y Windows, también se puede seguir trabajando con Visual Studio.
* Es multiplataforma.
* Posee diseñador de Interfaz de Usuario nativa.
* Contiene Xamarin forms.
* Mayor accesibilidad a ciertas características nativas.
* Facilidad de importación a librerías externas.
* Acepta todas las características del framework .NET.
* Permite reutilizar el 90% del código.
* Incorpora un plugin para usarlo con Visual Studio.
* Genera código nativo.

**Desventajas**

* Es la herramienta que menos plataformas abarca. Tiene versiones para Android y iOS. Al ser C#, gran parte del código puede utilizarse en Windows Phone.
* En OS X sólo se puede usar su IDE. La opción de Visual Studio solo está en Windows.
* Necesita tiempo extra para compilar.
* Falta de documentación extensiva.
* Falta de soporte a ciertas características.

**Microsoft Visual Studio:**

Es un [entorno](http://es.wikipedia.org/wiki/Entorno) (IDE, por sus siglas en inglés) para sistemas operativos [Windows](http://es.wikipedia.org/wiki/Windows). Soporta múltiples lenguajes de programación tales como [C++](http://es.wikipedia.org/wiki/Visual_C%2B%2B), [C#](http://es.wikipedia.org/wiki/C_sharp),[Visual Basic .NET](http://es.wikipedia.org/wiki/Visual_Basic_.NET), [F#](http://es.wikipedia.org/wiki/F_Sharp), [Java](http://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)), [Python](http://es.wikipedia.org/wiki/Python), [Ruby](http://es.wikipedia.org/wiki/Ruby), [PHP](http://es.wikipedia.org/wiki/PHP).

**Características:**

* Editor de código: el núcleo del IDE de desarrollo, el propio Visual Studio, se rige sobre un editor de código/interfaz que soporta los más variados lenguajes, proporcionando un completo intellisense predictivo y múltiples herramientas de refactorización y aceleración de la codificación.
* Análisis del código y de las “Buenas prácticas”: la inclusión de pruebas de rendimiento, y del análisis estático del código, redondean un módulo que orienta al desarrollador hacia las mejores prácticas de codificación y de técnicas avanzadas de programación como el TDD o XP.
* Test de las aplicaciones: Para cumplir con las extensas necesidades de la programación actual, incluye un completo framework de test unitario y de integración; acompañado por la gestión completa de los planes de prueba por medio del módulo de Test Manager, e incluyendo test exploratorios grabados y automatizados de las interfaces de usuario, pudiendo ser programados con test CodedUI.
* Pruebas de carga y rendimiento: también incluye un módulo con entidad propia para realizar el análisis en profundidad de los diagnósticos de prestaciones que cualquier tipo de aplicación desarrollada en .Net.
* Arquitectura: módulo orientado a la documentación arquitectónica que permite, por ejemplo, modelar en UML toda la estructura del proyecto, incluso generando código desde los diagramas; navegar por la vista de clases; verificar las referencias circulares, etc.
* Team Explorer: desde esta ventana se podra gestionar todo el trabajo almacenado en el Team Foundation Server. Gestionar las tareas, el trabajo, explorar el repositorio de código y las operaciones de branching y merge, lanzar y programar las build, etc.
* Construcción y depuración: permite construir las aplicaciones para todos los dispositivos, plataformas y sistemas operativos soportados; y realizar decenas de operaciones y validaciones de depuración que permitan encontrar los fallos de manera fácil y sencilla; incluso permite añadirse a un proceso abierto en el equipo de desarrollo (o remotamente) para depurar aplicaciones no soportadas por .NET.
* Herramientas: permite conectarse contra una base de datos sql, comparar los esquemas, comparar los datos, lanzar querys; conectarse contra un SharePoint o un Azure; crear un GUID (identificador único); ofuscar y analizar código; configurar servicios WCF; obtener la ejecución detallada de procesos; y optimizar y configurar el propio IDE.

**Ventajas:**

* Permite crear aplicaciones para windows en poco tiempo.
* Versión colaborativa para llevar a cabo proyectos en conjunto.
* Permite generar librerías dinámicas de forma activa, mediante una reconfiguración en el proceso de colección o codificación.
* Gestión del Ciclo de Vida de las Aplicaciones (ALM), la cual permite crear soluciones de calidad y a medida, a la vez que se reducen el coste y el tiempo del desarrollo.
* Análisis o testeo de las funcionalidades.
* Control de pruebas y calidad.
* Permite desarrollar aplicaciones web sobre la base ASP.NET y DMBC, también para el cloud, para Windows Phone, Windows 8.1.
* Permite trabajar con Xamarin y usar C# para programar apps nativas también en Android y iOS.
* Posee “telemetría” o tracking de lo que pasa en la aplicación, para poder tener mejor información de cómo la app está siendo usada, dónde se está usando, y dónde están los picos de demanda.
* Visual Studio es gratuito hasta cinco usuarios

**Desventajas:**

* No se puede exportar el código a otras plataformas de windows.

# Lenguajes de Programación

Los dos lenguajes de programación más conocidos y utilizados para las IDE seleccionadas son:

* C sharp
* Java

**C sharp:**

C# (pronunciado si sharp en inglés) es un [lenguaje de programación](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n) [orientado a objetos](http://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos) desarrollado y estandarizado por [Microsoft](http://es.wikipedia.org/wiki/Microsoft) como parte de su plataforma [.NET](http://es.wikipedia.org/wiki/Microsoft_.NET).

**Características:**

* Orientado a objetos: C# soporta todas las características del paradigma de la programación orientada a objetos, como son la encapsulación, la herencia y el polimorfismo.
* Orientado a componentes: La sintaxis de C# incluye por ejemplo formas de definir propiedades, eventos o atributos.
* Recolección de basura: tiene a su disposición el recolector de basura del CLR. Esto implica que no es necesario incluir instrucciones de destrucción de objetos en el lenguaje.
* Declaraciones en el espacio de nombres: al empezar a programar, se puede definir una o más clases dentro de un mismo espacio de nombres.
* Atributos: cada miembro de una clase tiene un atributo de acceso del tipo público, protegido, interno, interno protegido y privado.
* Pase de parámetros: se puede declarar a los métodos para que acepten un número variable de parámetros.
* Métodos virtuales y redefiniciones: antes de que un método pueda ser redefinido en una clase base, debe declararse como virtual. El método redefinido en la subclase debe ser declarado con la palabra override.
* Propiedades: un objeto tiene intrínsecamente propiedades, y debido a que las clases en C# pueden ser utilizadas como objetos, C# permite la declaración de propiedades dentro de cualquier clase.
* Inicializador: un inicializador es como una propiedad, con la diferencia de que en lugar de un nombre de propiedad, un valor de índice entre corchetes se utiliza en forma anónima para hacer referencia al miembro de una clase.

**Ventajas:**

* Tipos de datos: en C# existe un rango más amplio y definido de tipos de datos que los que se encuentran en C, C++ o Java.
* Programación Orientada a Objetos.
* Control de versiones: permite mantener múltiples versiones de clases en forma binaria, colocándolas en diferentes espacios de nombres. Esto permite que versiones nuevas y anteriores de software puedan ejecutarse en forma simultánea.
* Sencillez de uso: elimina muchos elementos añadidos por otros lenguajes y que facilitan su uso y compresión, es por ello que se dice que C# es auto contenido. Además, no se incorporan al lenguaje elementos poco útiles.
* Seguridad de tipos: incluye mecanismos de control de acceso a tipos de datos, lo que garantiza que no se produzcan errores difíciles de detectar como un acceso a memoria de ningún objeto.
* Instrucciones seguras: se han impuesto una serie de restricciones en el uso de instrucciones de control más comunes.
* Unificación de tipos: todos los tipos derivan de una superclase común llamada System.Object, por lo que automáticamente heredarán todos los miembros definidos en esta clase. Es decir, son objetos.
* Compatible: Ofrece la posibilidad de acceder a código nativo escrito como funciones sueltas no orientadas a objetos, tales como las DLLs de la API de Win32

**Desventajas:**

* Se tiene que conseguir una versión reciente de Visual Studio .NET.
* Se tiene que tener algunos requerimientos mínimos del sistema para poder trabajar adecuadamente tales como contar con Windows NT 4 o superior, tener alrededor de 4 gigas de espacio libre para la pura instalación.
* Para quien no está familiarizado con ningún lenguaje de programación, le costará más trabajo iniciarse en su uso.

**JAVA**

Los desarrolladores (Sun Microsystems, James Gosling, 2014) definen Java como "Un lenguaje simple. Orientado al objeto, distribuido, interpretado, sólido, seguro, de arquitectura neutral, portable, de alto desempeño, de multihilos y dinámico".

Java está diseñado para que un programa escrito en este lenguaje sea ejecutado independientemente de la plataforma (hardware, software y sistema operativo) en la que se esté actuando.

**Características:**

* Lenguaje totalmente orientado a Objetos: todos los conceptos en los que se apoya esta técnica, encapsulación, herencia, polimorfismo, etc., están presentes en Java.
* Disponibilidad de un amplio conjunto de bibliotecas: posee la posibilidad de utilizar el conjunto de clases que Sun pone a disposición del programador y con las cuales es posible realizar prácticamente cualquier tipo de aplicación.
* Distribuido: Java proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets, establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas.
* Interpretado y compilado a la vez: Java es compilado, en la medida en que su código fuente se transforma en una especie de código máquina, los [bytecodes](http://es.wikipedia.org/wiki/bytecode), semejantes a las instrucciones de ensamblador. Por otra parte, es interpretado, ya que los bytecodes se pueden ejecutar directamente sobre cualquier máquina a la cual se hayan portado el intérprete y el sistema de ejecución en tiempo real (run-time).
* Robusto: Java fue diseñado para crear software altamente fiable. Para ello proporciona numerosas comprobaciones en compilación y en tiempo de ejecución.
* Indiferente a la arquitectura: Java está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red, desde Unix a Windows NT, pasando por Mac y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos.
* Portable.
* Multitarea: soporta sincronización de múltiples hilos de ejecución (multithreading) a nivel de lenguaje, especialmente útiles en la creación de aplicaciones de red distribuidas.
* Dinámico: Las clases sólo se enlazan a medida que son necesitadas. Se pueden enlazar nuevos módulos de código bajo demanda, procedente de fuentes muy variadas, incluso desde la Red.

**Ventajas:**

* Manejo automático de la memoria.
* Lenguaje Multiplataforma.
* Programación Orientada a Objetos.
* Puede correr en el explorador y en dispositivos móviles.
* Sintaxis similar a C/C++.

**Desventajas:**

* Requiere un intérprete.
* Algunas implementaciones y librerías pueden tener código rebuscado.
* Una mala implementación de un programa en java, puede resultar en algo muy lento.
* Los programas hechos en java no tienden a ser muy rápidos.
* Algunas herramientas tienen un costo adicional.
* Hay diferentes tipos de soporte técnico para la misma herramienta, por lo que el análisis de la mejor opción se dificulta.
* Para manejo a bajo nivel deben usarse métodos nativos, lo que limita la portabilidad.
* El diseño de interfaces gráficas con awt y swing no es simple. Existen herramientas como el JBuilder que permiten generar interfaces gráficas de manera sencilla, pero tienen un costo adicional.

**Semejanzas entre C# Java:**

* Son Lenguajes de última generación y de propósito general
* Son Lenguajes Orientado a Objetos.
* Incluyen numerosas restricciones para garantizar su seguridad y confiabilidad
* Para evitar errores comunes como se producían programando en otros lenguajes, en estos se han impuesto una serie de restricciones en el uso de instrucciones de control más comunes.

**Diferencias entre C# y Java**

* Java es un lenguaje portable.
* Java especifica los tamaños de sus tipos de datos básicos y el comportamiento de sus operadores aritméticos, de manera que los programas son iguales en todas las plataformas.
* Java puede ser usado para crear dos tipos de programas: aplicaciones independientes y applets.
* En C# todos los tipos derivan de una superclase común llamada System.Object, por lo que automáticamente heredarán todos los miembros definidos en esta clase. Es decir, son objetos. A diferencia de Java.

# ¿C Sharp o Java?

Luego de investigar acerca de las ventajas y desventajas que presentan los dos lenguajes de programación más utilizados se optó por la utilización de C Sharp debido a lo siguiente:

**Ventajas de C sharp frente a java:**

* El rendimiento es, por lo general, mucho mejor.
* CIL (el lenguaje intermedio de .NET) está estandarizado, mientras que los bytecodes de java no lo están.
* Soporta bastantes más tipos primitivos (value types), incluyendo tipos numéricos sin signo.
* Indizadores que permiten acceder a cualquier objeto como si se tratase de un array.
* Compilación condicional.
* Aplicaciones multi-hilo simplificadas.
* Soporta la sobrecarga de operadores, que aunque pueden complicar el desarrollo son opcionales y algunas veces muy útiles
* Permite el uso (limitado) de punteros cuando realmente se necesiten, como al acceder a librerías nativas que no se ejecuten sobre la máquina virtual.

**LA APUESTA POR .NET Y C#**

Si bien es atractivo programar para Android en C#, existen tres razones adicionales por lo cual es conveniente apostar por .NET y C# para el desarrollo de aplicaciones móviles:

* Desarrollo Multiplataforma.
* Reutilización de bibliotecas de código existentes.
* Interfaz de usuario nativa y rendimiento nativo.

Contar con C# y .NET nos permite reutilizar las bibliotecas de código existentes que se hayan desarrollado previamente (incluso para otras aplicaciones .NET) y explotar los algoritmos escritos en C# por la comunidad. Utilizando la característica Portable Class Library (PCL) de .NET, es posible incluso desarrollar la lógica de la aplicación una sola vez y desarrollar para cada plataforma sólo el código específico que se requiera. De esta forma se simplifica el mantenimiento de la aplicación. Al obtener un resultado que compila directamente e interactúa con las características nativas del dispositivo, se obtienen beneficios de rendimiento con respecto a las aplicaciones HTML5 que requieren capas de abstracción más complejas para su ejecución. Además, la experiencia de usuario (UI) se mantiene intacta, ya que se utilizan los mismos componentes nativos de la plataforma para el desarrollo de la interfaz de usuario.