Introducción a XNA

Octubre 2014

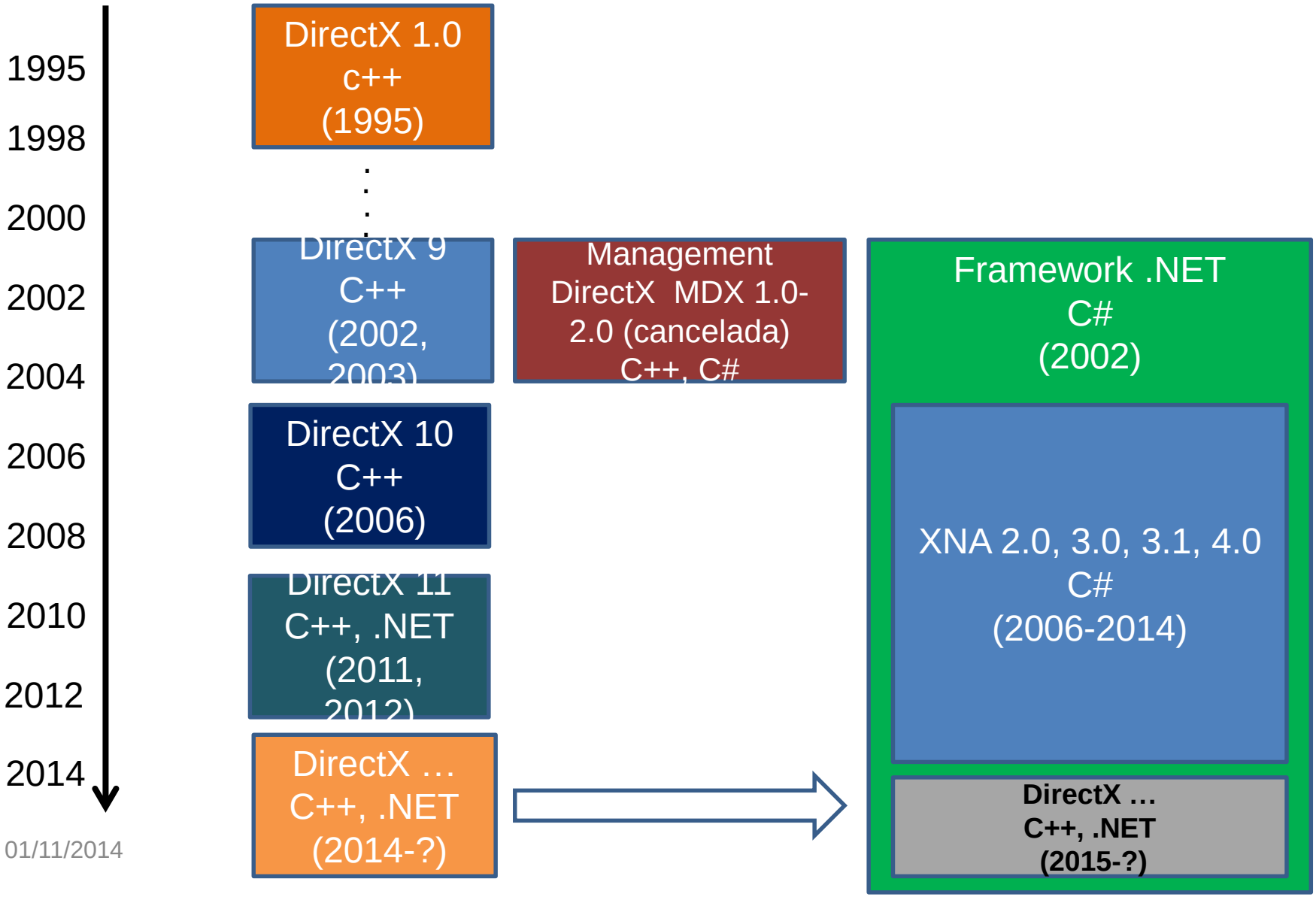
Origen…

- Desde el lanzamiento de Windows 95 se introdujo la API DirectX
  - Desarrollada para manejo de gráficos y multimedia (videojuegos, video)
- Busca integrar el desarrollo de aplicaciones con el hardware para gráficos
- El kit de desarrollo (SDK) es orientado a C++
  - Aunque hay lanzamientos no oficiales en otros lenguajes

Origen…

- El surgimiento de .NET (2002) hace posible escribir y compilar programas en varios lenguajes
  - Common Language Runtime (CLR)
  - Common Runtime Code
- XNA toma lugar en el Framework .NET

Origen…


1995
1998
2000
2002
2004
2006
2008
2010
2012
2014
DirectX 1.0
c++
(1995)
DirectX 9
C++
(2002, 2003)
Management
DirectX MDX 1.0-2.0 (cancelada)
C++, C#
DirectX 10
C++
(2006)
DirectX 11
C++, .NET
(2011, 2012)
DirectX …
C++, .NET
(2014-?)
Framework .NET
C#
(2002)
XNA 2.0, 3.0, 3.1, 4.0
C#
(2006-2014)
DirectX …
C++, .NET
(2015-?)

¿Qué es XNA?

- Consiste de un framework
- Conjunto de bibliotecas de código para realizar tareas relacionadas a:
  - Gráficos, Sonido => Juegos

- XNA Game Studio
  - Extensión de Visual Studio C# que incluye plantillas para la creación de proyectos

Proyecto XNA

- Los plantillas integran el loop (bluce|ciclo) del juego
- Fácil de utilizar, permite acceder a distintos dispositivos gráficos
- Incluye métodos para mostrar gráficos
- Soporta modelos 3D

Evolución de XNA

XNA Game Studio Express, XNA 1.0
Diciembre 2006
Windows – XBOX 360

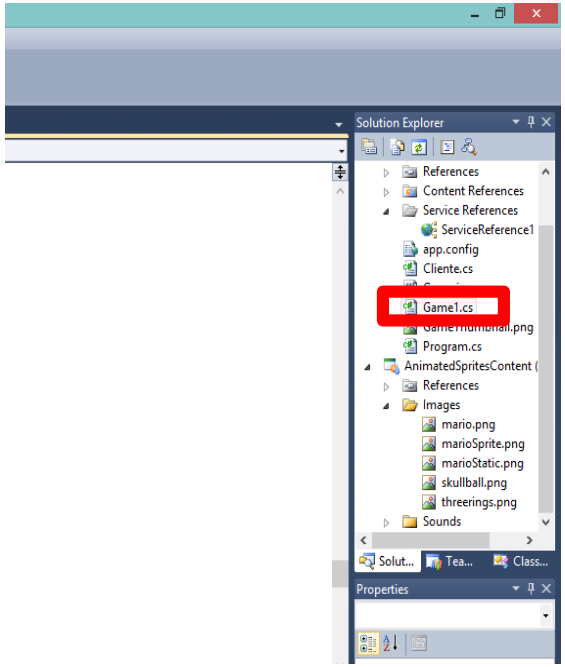
XNA 2.0
Visual Studio 2005
Free – Visual C# 2005 Express
Diciembre 2007
API Redes Xbox Live

XNA 3.0, 3.1, C# 3.0, LINQ
Octubre 2008, 2009
Windows – XBOX 360, Multiplayer, Avatars

XNA 4.0
Octubre 2010
Windows – XBOX 360
Windows Phone
Soporte hasta Abril 2014

Clase Game1

- Archivo generado automáticamente
- Contiene lo básico para la codificación del código
- Tiene 5 métodos principales para la personalización del proyecto XNA



Clase Game1

- Acceso al sistema de tarjeta de video – **GraphicsDeviceManager**
- **SpriteBatch** permite dibujar imágenes 2D de forma muy rá

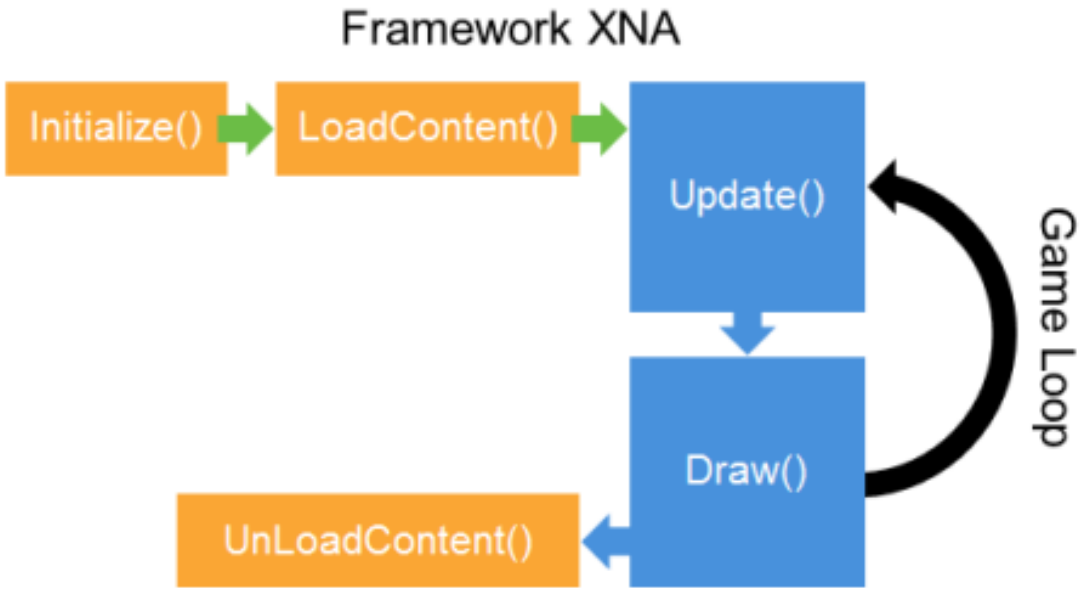
```
/// <summary>
/// This is the main type for your game
/// </summary>
public class Game1 : Microsoft.Xna.Framework.Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;

    public Game1()
    {
        graphics = new GraphicsDeviceManager(this);
        Content.RootDirectory = "Content";
    }

    /// <summary>
    /// Allows the game to perform any initialization it needs to before starting to run.
    /// This is where it can query for any required services and load any non-graphic
    /// related content.  Calling base.Initialize will enumerate through any components
    /// and initialize them as well.
    /// </summary>
```

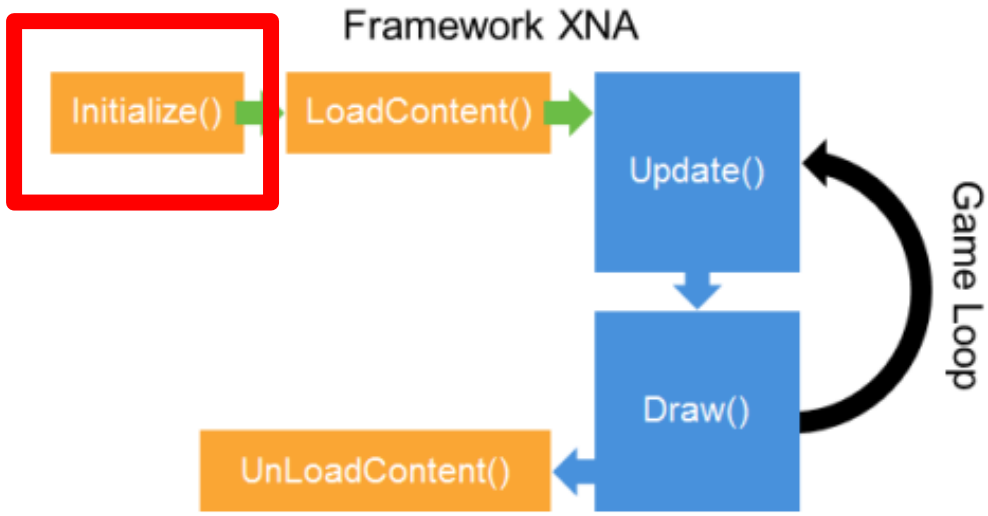
Funcionamiento de XNA

- Es importante conocer el orden de los métodos (invocado 60 veces por segundo – 60 FPS)



Método Inicializar (Initialize)

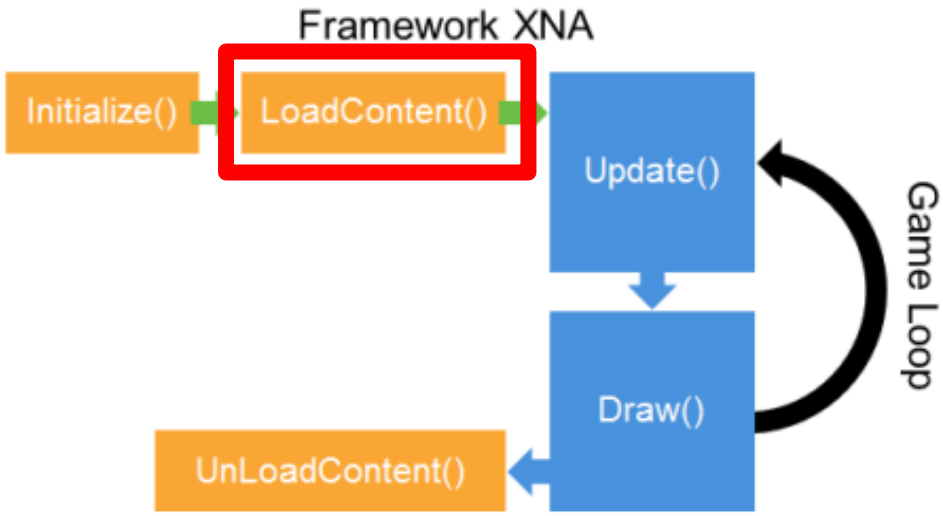
- Inicializar variables y otros objetos asociados con el objeto **Game1**
- Los objetos del dispositivo gráfico son inicializados en este punto
- Este método se usa para inicializar valores de puntuación y otros valores



Método LoadContent

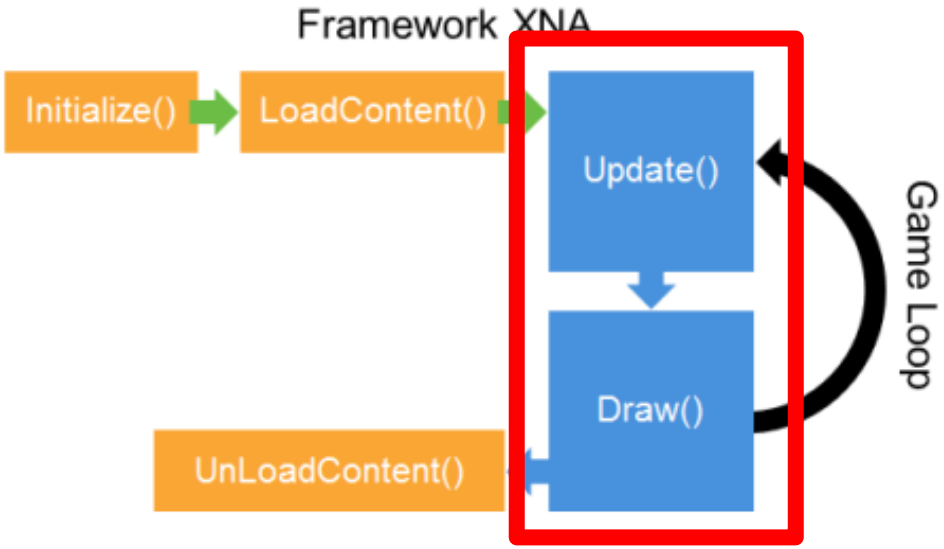
- Se puede volver a invocar cuando el dispositivo grafico debe ser reiniciado
  - P.ej. El jugador ha cambiado la configuración gráfica del juego

- El método **LoadContent** es donde se cargan todos los gráficos y contenidos requeridos por el juego
  - Incluyendo imágenes, modelos, sonidos, etc.



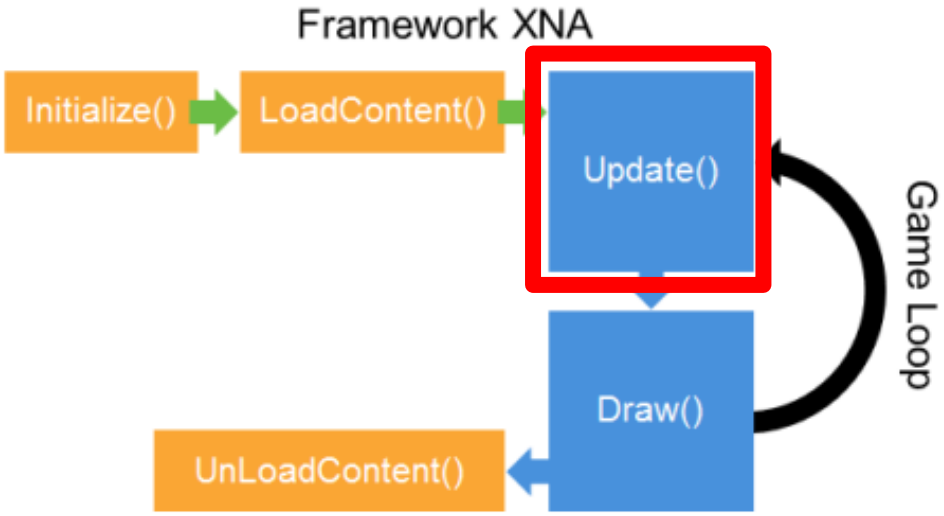
Método Update y Método Draw

- Cuando termina el método **LoadContent**, el juego entra en un estado conocido como **Game Loop**
- La mayoría de los videojuegos entran en un tipo de bucle independientemente si están escritos en XNA
- Un Game Loop consiste en una serie de métodos que son invocados una y otra vez hasta que el juego finaliza
- En XNA, el Game Loop consiste en dos métodos: **Update** y **Draw**
- Toda la lógica que afecta el juego actual se encontrará en el método Update o el método Draw



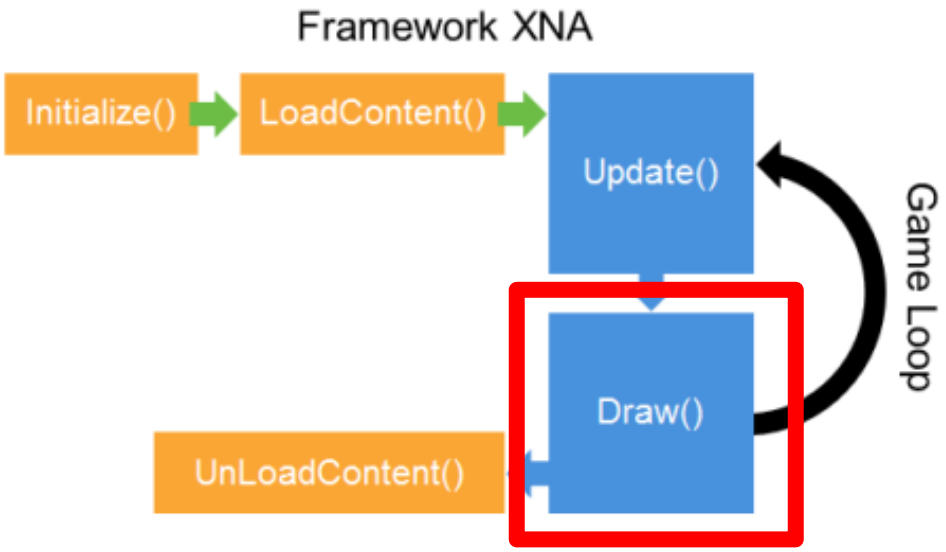
Método Update

- Lo necesario para la ejecución adecuada del juego se encontrará en el método **Update**
  - Mover objetos, colisiones, actualizar puntuaciones, checar condición de fin de juego



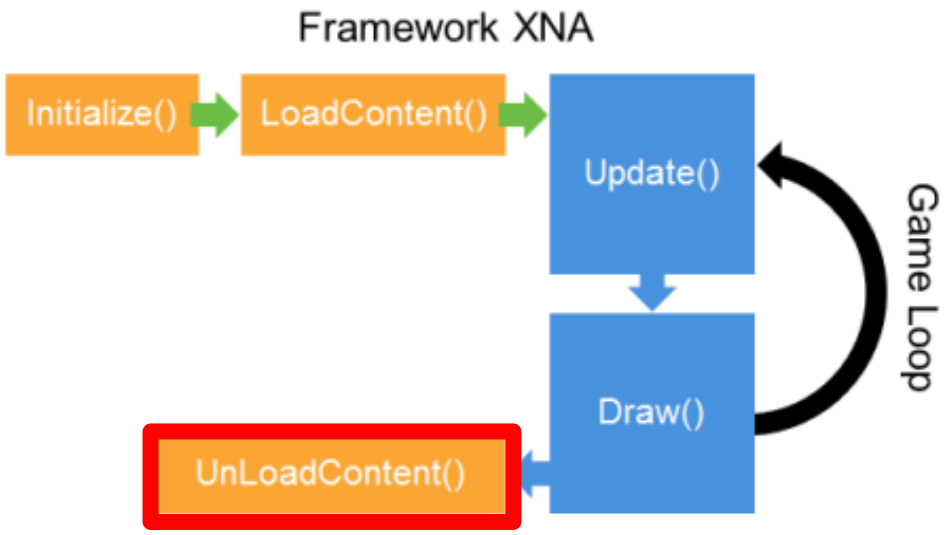
Método Draw

- Usado para dibujar elementos
- Implementar lo mínimo en esta área, excepto para **desplegar elementos** de una escena

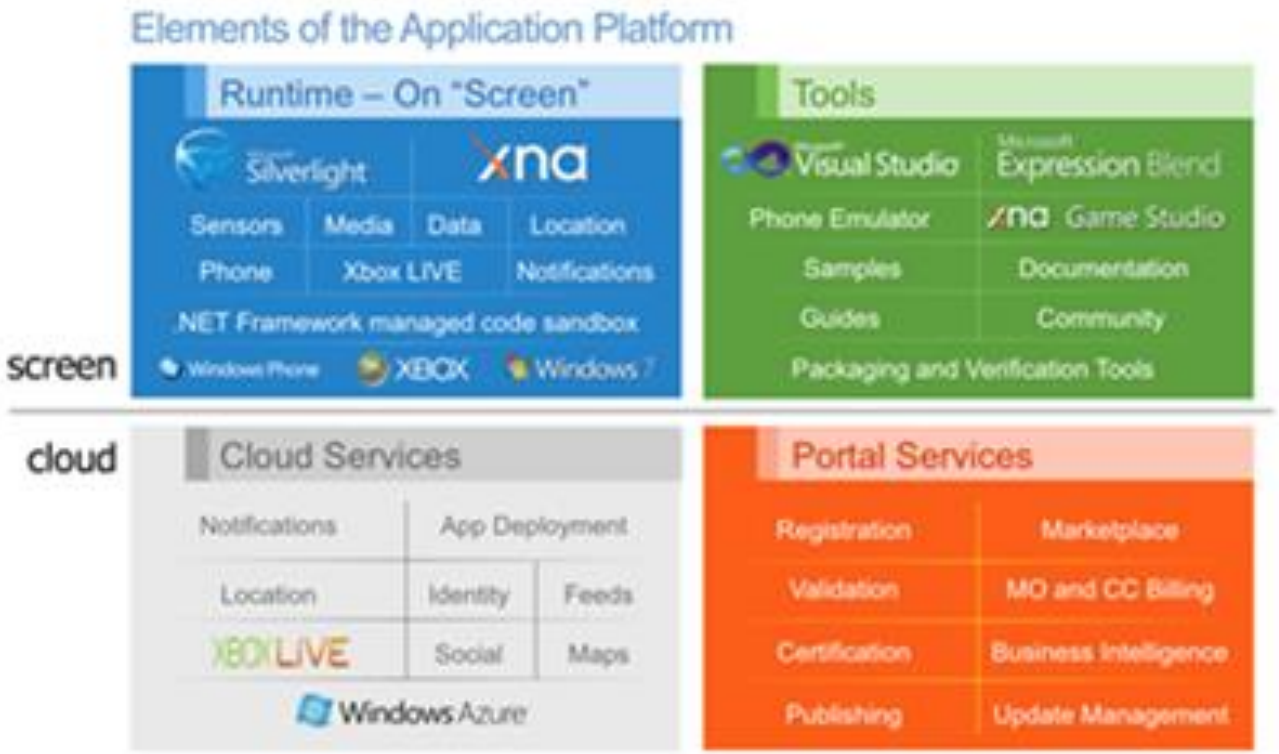


Método UnloadContent

- Cuando el Game Loop finalice se invoca automáticamente el método **UnloadContent**.
- Usado para liberar cualquier contenido cargado previamente en el método **LoadContent**
- Se gestiona la recolección de basura
- En el método **UnloadContent** se hace la liberación de contenido de la memoria



Arquitectura XNA



Fuentes de información

- Microsoft Academy
- Learning XNA Academy 4.0, Aaron Reed
- XNA 4.0 Game Development by Example, Kurt Jaegers