RA 2 Parte 1 Teoría – Programación de aplicaciones para dispositivos móviles

Herramientas y fases de construcción.
Interfaces de usuario.
Componentes disponibles -API
Contexto gráfico. Imágenes.
Eventos de teclado - Unity.
Técnicas de animación.
Técnicas de sonido.
Bases de datos y almacenamiento.
Persistencia de datos.
Envío y recepción de mensajes.
Servicios web.

Programación Multimedia y Dispositivos Móviles – 2º CFGS Desarrollo de Aplicaciones Multiplataforma – IES AGL – Curso 23/24

Herramientas y fases de construcción.

Desarrollo rápido de aplicaciones

- El **desarrollo rápido de aplicaciones** o **RAD** (acrónimo en inglés de *rapid application development*) es un proceso de desarrollo de software, desarrollado inicialmente por James Martin en 1991 basado en el trabajo hecho por Scott Shultz en los 80.
- El método comprende el desarrollo interactivo, la construcción de prototipos y el uso de utilidades CASE (ingeniería asistida por computadora). Tradicionalmente, el desarrollo rápido de aplicaciones tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución.[1][2]
- Hoy en día se suele utilizar para referirnos al desarrollo rápido de interfaces gráficas de usuario tales como Glade, o entornos de desarrollo integrado completos. Algunas de las plataformas más conocidas son Visual Studio, Lazarus, Gambas, Delphi, Foxpro, Anjuta o Game Maker.
- En el área de la autoría multimedia, software como Neosoft Neobook y MediaChance Multimedia Builder proveen plataformas de desarrollo rápido de aplicaciones, dentro de ciertos límites...

Herramientas y fases de construcción.

Principios básicos del Desarrollo Rápido de Aplicaciones:

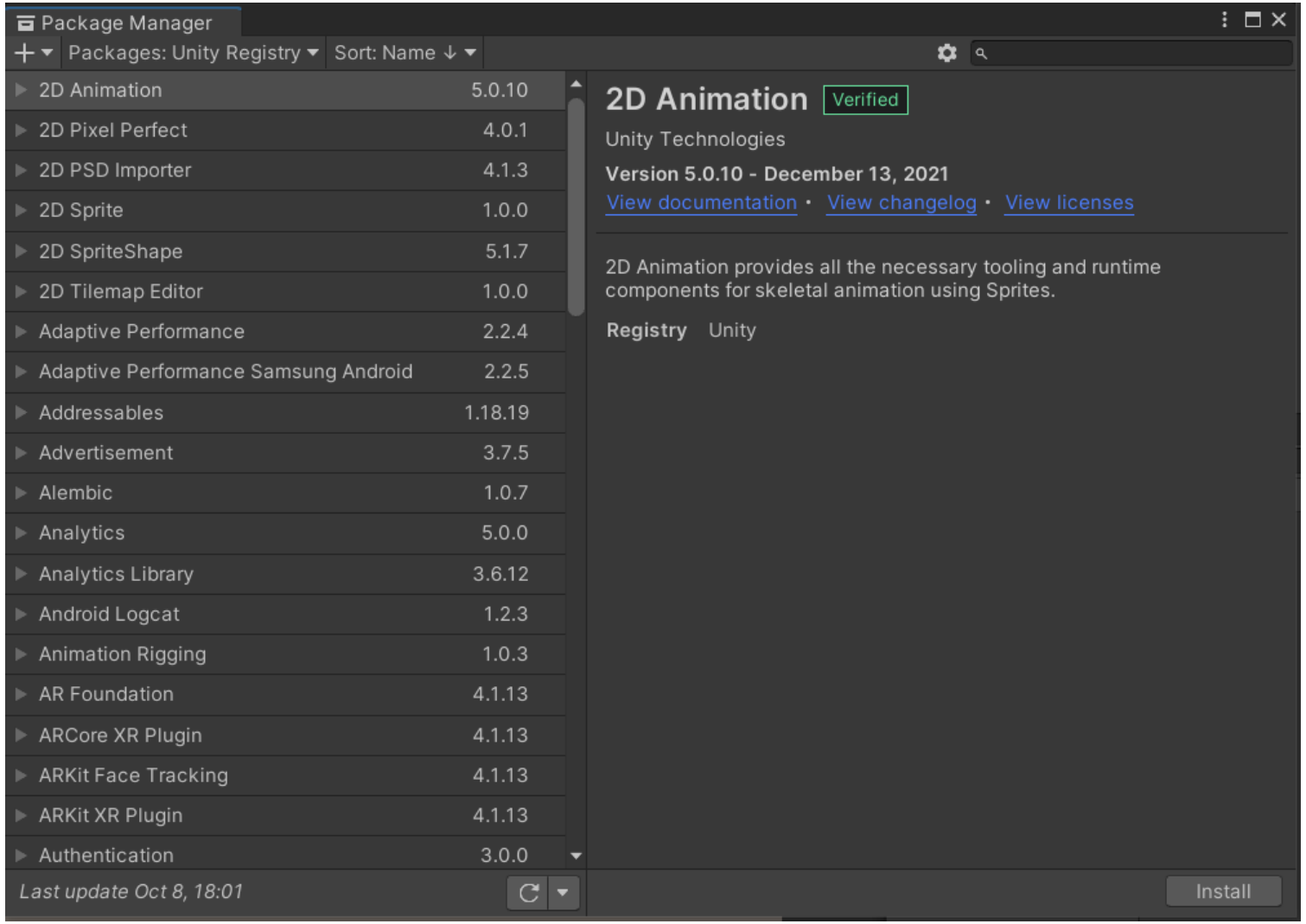
- Objetivo clave es para un rápido desarrollo y entrega de una alta calidad en un sistema de relativamente **bajo coste de inversión**.
- **Intenta reducir los riesgos inherentes del proyecto** partiéndolo en segmentos más pequeños y proporcionar más facilidad de cambio durante el proceso de desarrollo.
- Orientación dedicada a producir sistemas de alta calidad con rapidez, principalmente mediante el uso de iteración por prototipos (en cualquier etapa de desarrollo), promueve la participación de los usuarios y el uso de herramientas de desarrollo computarizadas.
- Estas herramientas pueden incluir constructores de Interfaz gráfica de usuario (GUI), Computer Aided Software Engineering (CASE) las herramientas, los sistemas de gestión de bases de datos (DBMS), lenguajes de programación de cuarta generación, generadores de código, y técnicas orientada a objetos.
- Hace especial hincapié en el cumplimiento de la necesidad comercial, mientras que la ingeniería tecnológica o la excelencia es de menor importancia.
- Control de proyecto implica el desarrollo de prioridades y la definición de los plazos de entrega. Si el proyecto empieza a aplazarse, **se hace hincapié en la reducción de requisitos para el ajuste**, no en el aumento de la fecha límite.
- La **participación activa de los usuarios** es imprescindible.
- Iterativamente realiza la **producción de software, en lugar de enfocarse en un prototipo.**
- Produce la **documentación necesaria** para facilitar el futuro desarrollo y mantenimiento.

Herramientas y fases de construcción.

- En el caso de Unity dispone de múltiples herramientas para facilitar la creación de un videojuego en 2D y 3D como el **ProBuilder, el ProGrids o el PolyBrush** para facilitar el diseño artístico o el **CodeCoverage** para realizar analíticas de uso de scripts de la aplicación sobre ejecución.
- Todas estas herramientas os las iré tramitando pero si queréis ir trasteando por vuestra cuenta en vuestros desarrollos se encuentran en Window – Package Manager

Herramientas y fases de construcción.

- Importante: **Package: Unity Registry** para que te aparezcan los disponibles para el motor gráfico.
- Con darle al Install os aparecerá una nueva Ventana en el panel de Unity: **Tools** donde podréis acceder a todas las que tengáis instaladas.



Interfaces de usuario.

- Hoy en día un buen diseño es el que nos atrae más a las personas y el que elegimos a simple vista. Para conseguir hacer una buena app y que resalte entre las otras, el truco está en la simplicidad y como saberla utilizar. Así que lo más importante es la simplicidad en el diseño, pero también la simplicidad y la facilidad a la hora de navegar por la aplicación.
- La interfaz de una aplicación es como la ropa que te pones para salir a la calle. Está compuesta por botones, gráficos, iconos y fondos de pantalla que tienen una apariencia visual diferente de cada uno de los sistemas operativos, Android, iOS y Windows Phone tienen su propia manera de entender el diseño. El trabajo del diseñador consiste en interpretar la personalidad de cada sistema operativo para conseguir aplicaciones que sean diferentes de las demás.

Interfaces de usuario.

- **Usabilidad:** Esto parece obvio, pero si no lo es la aplicación no duraría ni un mes en la tienda. Tenemos que tener presente que sea sencillo para el propio desarrollador pero también para los usuarios que la usarán.
- **Fácil de aprender:** Las buenas aplicaciones son intuitivas y no les hacen falta muchas explicaciones para que los usuarios las entiendan.
- **Código y significantes:** Los códigos que pones en cualquier lugar de la aplicación siempre tendrán que tener el mismo significado.
- **Diseño enfocado a metas:** conoce primero a tu usuario antes de empezar a diseñar, el diseño enfocado a metas orientado al que tu target busca te hará más sencillo determinar qué elementos son necesarios y qué no dentro de tu aplicación.
- **Patrones y texturas planas:** Lo más atractivo de hoy en día son los fondos con texturas planas.
- **Esquemas de colores simples:** El color se utiliza estratégicamente para crear interés visual, sin añadir ningún elemento de diseño aparte.
- **Efectos de desenfocamiento:** Los fondos desenfocados son una de las cosas que hacen más atractivas las aplicaciones.
- **Uso de tipografía:** El hecho de utilizar diferentes tipografías puede ser signo de una app descuidada. Por ello, lo más adecuado es utilizar una tipografía que sea legible a simple vista y que no cueste de leer.

Interfaces de usuario.

Iconos y pantalla inicial

- Dicen que la primera impresión es la que cuenta. En el mundo de las aplicaciones esta primera impresión está limitada a dos componentes visuales: el icono de lanzamiento y la pantalla inicial -también llamada splash- que se mostrará muchas veces al abrir la aplicación.
- Estos elementos se verán antes de nada, incluso, antes de empezar a utilizar realmente la aplicación. No despreciar su importancia y darlos la atención que merecen, garantiza arrancar con el pie derecho.

Grilla (Grid) o retícula de construcción

- En el diseño de interfaces para móviles, la grilla permite establecer márgenes y determinar la ubicación de los botones, la separación de la tipografía y el espacio interior y exterior de los contenedores. Por supuesto, cada uno de los sistemas operativos tiene diferentes retículas y por tanto, diferentes módulos.

Interfaces de usuario.

Tipografía

- Cómo en cualquier diseño, el objetivo de la tipografía es conseguir que el texto se lea con claridad. Esto se consigue no solo con una adecuada elección de la tipografía, sino también gestionando su tamaño, separación entre líneas, ancho de columnas y contraste visual con el fondo.

Color

- El color también es una de las cosas más importantes puesto que depende mucho del aspecto visual, ya que cada color puede significar una cosa. También sirve para separar apartados y así poder diferenciar y leer las cosas más fácilmente.

Interfaces nativas o personalizadas

- Las interfaces nativas son las que vienen ya hechas para facilitarte un poco el trabajo a la hora de crear la aplicación. Tienen un aspecto ya definido en cuanto a color, medida o tipografía.
- Es recomendable, al empezar, que definas la interfaz con elementos nativos, para que no sea necesario crear todos los elementos de nuevo. El inconveniente de las interfaces nativas es que no puedes personalizar el diseño 100 % a tu gusto y te tienes que adaptar a algunos elementos que quizás no te gustan o no puedes mover.

Componentes disponibles -API

- Una **API** (del inglés, *Application Programming Interface*, en español, *interfaz de programación de aplicaciones)* es una pieza de código que permite a diferentes aplicaciones comunicarse entre sí y compartir información y funcionalidades. Una API es un intermediario entre dos sistemas, que permite que una aplicación se comunique con otra y pida datos o acciones específicas.
- Por ejemplo, si se tiene una app para móviles acerca de recetas y se hace una búsqueda de una determinada receta, se puede utilizar una API para que esta aplicación se comunique con el sitio web de recetas y pida las recetas que cumplen con los criterios de búsqueda.
- La API entonces se encarga de recibir la solicitud, buscar las recetas apropiadas y regresar los resultados a la aplicación. Una API es una forma de conectar diferentes aplicaciones y hacer que trabajen juntas de manera más eficiente y efectiva. Son usadas generalmente en las bibliotecas de programación.

Componentes disponibles -API

- Una API representa la capacidad de comunicación entre componentes de *software*.
- Se trata del conjunto de llamadas a ciertas bibliotecas que ofrecen acceso a ciertos servicios desde los procesos y representa un método para conseguir **abstracción** en la **programación**, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del *software*.
- Uno de los principales propósitos de una API consiste en proporcionar un conjunto de **funciones** de uso general, por ejemplo, para dibujar **ventanas o iconos** en la **pantalla**. De esta forma, los **programadores** se benefician de las ventajas de las API haciendo uso de su funcionalidad, evitándose el trabajo de programar todo desde el principio.
- Las API asimismo son abstractas: el ***software*** que proporciona una cierta API generalmente es llamado la implementación de esa API.

Componentes disponibles - API

- Por ejemplo, se puede ver la tarea de escribir "*Hola Mundo*" sobre la pantalla en diferentes niveles de abstracción:

1. Haciendo todo el trabajo desde el principio:
    1. Traza, sobre papel milimetrado, la forma de las letras (y espacio) "*H, o, l, a, M, u, n, d, o*".
    2. Crea una matriz de cuadrados negros y blancos que se asemeje a la sucesión de letras.
    3. Mediante instrucciones en ensamblador, escribe la información de la matriz en la memoria intermedia (búfer) de pantalla.
    4. Mediante la instrucción adecuada, haz que la tarjeta gráfica realice el volcado de esa información sobre la pantalla.
2. Por medio de un sistema operativo para hacer parte del trabajo:
    1. Carga una fuente tipográfica proporcionada por el sistema operativo.
    2. Haz que el sistema operativo borre la pantalla.
    3. Haz que el sistema operativo dibuje el texto "*Hola Mundo*" usando la fuente cargada.

Componentes disponibles - API

3. Usando una aplicación (que a su vez usa el sistema operativo) para realizar la mayor parte del trabajo:
    1. Escribe un documento HTML con las palabras "*Hola Mundo*" para que un navegador web como Firefox, Chrome, Opera, Safari, Midori, Web o Microsoft Edge pueda representarlo en el monitor.

Como se puede ver, la primera opción requiere más pasos, cada uno de los cuales es mucho más complicado que los pasos de las opciones siguientes. Además, no resulta nada práctico usar el primer planteamiento para representar una gran cantidad de información, como un artículo enciclopédico sobre la pantalla, mientras que el segundo enfoque simplifica la tarea eliminando un paso y haciendo el resto más sencillos, y la tercera forma simplemente requiere escribir "*Hola Mundo*".

Sin embargo, las API de alto nivel generalmente pierden flexibilidad; por ejemplo, resulta mucho más difícil en un navegador web hacer girar texto alrededor de un punto con un contorno parpadeante, que programarlo a bajo nivel. Al elegir usar una API se debe llegar a un cierto equilibrio entre su potencia, simplicidad y pérdida de flexibilidad.

Contexto gráfico. Imágenes.

- Una **imagen** (del latín *imago*) es una representación visual, que manifiesta la apariencia visual de un objeto real o imaginario. Aunque el término suele entenderse como sinónimo de representación visual, también se aplica como extensión para otros tipos de percepción, como imágenes auditivas, olfativas, táctiles, sinestesias, etc.
- Las imágenes que la persona no percibe sino que vive interiormente, se las denominan **imágenes mentales**, mientras que las que representan visualmente un objeto mediante técnicas diferentes, se las designa como imágenes creadas, (o bien, imágenes reproducidas). Algunas de ellas son el dibujo, el diseño, la pintura, la fotografía o el vídeo, entre otras.

Contexto gráfico. Imágenes.

Se pueden distinguir una serie de funciones características de la imagen y dentro de estas encontramos: la función representativa, la simbólica, la semántica, la epistémica y la estética. A pesar de ello, resulta casi imposible separarlas o categorizarlas debido a que la función de la imagen casi siempre es compartida, lo mismo encuentran su función para informar como para causar sentimientos o significados.

- **Función representativa**: es aquella que se refiere a los casos en los que la imagen quiere decir lo que está reproduciendo. Un buen ejemplo sería una fotografía tomada en un viaje, ya que esa imagen refleja una información concreta.
- **Función simbólica**: es la que se refiere a la imagen con un significado o concepto añadido por las personas, que en un principio no tendríamos por qué haber relacionado con la misma. Hay que tener en cuenta que para comprender estas imágenes necesitamos una base de conocimientos, ya que el significado de estos conceptos añadidos a la imagen no está generalizados ni vienen predeterminados. Un buen ejemplo de ello sería una bandera. Si hablamos de forma cronológica posiblemente sería una de las primeras funciones de la imagen en su justificación con los símbolos religiosos y las manifestaciones de lo divino. Hoy en día esta función también encuentra lugar en la imagen publicitaria.
- **Función semántica**: es aquella que se produce cuando la propia imagen es la que actúa como signo, ya que la conexión entre dicha imagen y su sentido es completamente arbitraria. Si quisiéramos ejemplificar con una imagen la actividad de la lectura, podríamos utilizar la imagen de un niño con un libro, o un adulto con un papel, o personas leyendo el periódico.
- **Función epistémica**: "…las imágenes también sirven para dar informaciones acerca del mundo, son portadoras de conocimiento. Esta función sigue presente en gran cantidad de imágenes, aunque a través de la historia el valor informativo ha cambiado."
- **Función estética**: encuentra su justificación cuando la imagen produce sensaciones al momento de la percepción, estas sensaciones pueden inclusive estar relacionadas con las emociones.

Contexto gráfico. Imágenes.

- Una **imagen digital o gráfico digital** es una representación bidimensional de una imagen a partir de una matriz numérica, frecuentemente en binario (unos y ceros).
- Dependiendo de si la resolución de la imagen es estática o dinámica, puede tratarse de una imagen matricial (o mapa de bits) o de un gráfico vectorial. El mapa de bits es el formato más utilizado en informática.
- La mayoría de formatos de imágenes digitales están compuestos por una cabecera que contiene atributos (dimensiones de la imagen, tipo de codificación, etc.), seguida de los datos de la imagen en sí misma. La estructura de los atributos y de los datos de la imagen es distinto en cada formato.
- Además, los formatos actuales añaden a menudo una zona de metadatos ("metadata" en fotografía (Escala de sensibilidad, flash, etc.) Estos metadatos se utilizan muy a menudo en el formato extensión cámaras digitales y videocámaras.

Contexto gráfico. Imágenes.

Las **imágenes digitales** se pueden obtener de varias formas:

- Por medio de dispositivos de entrada conversión analógica-digital como los escáneres y las cámaras digitales.
- Directamente mediante programas informáticos editores de mapas de bits y dibujo vectorial, como por ejemplo realizando dibujos con el ratón o tableta digitalizadora gráfica incluyendo el lápiz óptico, por otro lado mediante un programa de renderización 3D a mapa de bits.

Las imágenes digitales se pueden modificar mediante filtros, añadir o suprimir elementos, modificar su tamaño, etc. y almacenarse en un dispositivo de grabación de datos como por ejemplo un disco duro.

- SVG para gráficos vectoriales, formato estándar del W3C (World Wide Web Consortium).

Contexto gráfico. Imágenes vectoriales.

- Una **imagen vectorial** es una imagen digital formada por objetos geométricos dependientes (segmentos, polígonos, arcos, muros, etc.), cada uno de ellos definido por atributos matemáticos de forma, de posición, etc. Por ejemplo un círculo de color rojo quedaría definido por la posición de su centro, su radio, el grosor de línea y su color.
- Este formato de imagen es completamente distinto al formato de las imágenes de mapa de bits, también llamados imágenes matriciales, que están formados por píxeles.
- El interés principal de los gráficos vectoriales es poder ampliar el tamaño de una imagen a voluntad sin sufrir la pérdida de calidad que sufren los mapas de bits. De la misma forma, permiten mover, estirar y retorcer imágenes de manera relativamente sencilla.
- Su uso también está muy extendido en la generación de imágenes en tres dimensiones tanto dinámicas como estáticas.
- Todos los ordenadores actuales traducen los gráficos vectoriales a mapas de bits para poder representarlos en pantalla al estar esta constituida físicamente por píxeles.

Contexto gráfico. Imágenes vectoriales

Generación de gráficos

- Se utilizan para crear logos ampliables a voluntad así como en el diseño técnico con programas de tipo CAD (*Computer Aided Design*, diseño asistido por computadora). Muy populares para generar escenas 3D.

Lenguajes de descripción de documentos

- Los gráficos vectoriales permiten describir el aspecto de un documento independientemente de la resolución del dispositivo de salida. Los formatos más conocidos son PostScript y PDF. A diferencia de las imágenes matriciales, se puede visualizar e imprimir estos documentos sin pérdida en cualquier resolución.

Tipografías

- La mayoría de aplicaciones actuales utilizan texto formado por imágenes vectoriales. Los ejemplos más comunes son TrueType, OpenType y PostScript.

Videojuegos

- En los videojuegos 3D es habitual la utilización de gráficos vectoriales.

Internet

- Los gráficos vectoriales que se encuentran en el World Wide Web suelen ser o bien de formatos abiertos VML y SVG, o bien SWF en formato propietario. Estos últimos se pueden visualizar con Adobe Flash Player.

Soportes publicitarios

- Los folletos, tarjetas de visita, vallas publicitarias y carteles utilizan gráficos vectoriales en formato AI

Contexto gráfico. Imágenes vectoriales

Impresión

- Un punto clave de las imágenes vectoriales es su práctica puesta a punto en el momento de la impresión ya que es posible escalarlas y aumentar su definición de forma ilimitada.
- Por ejemplo: se puede tomar el mismo logo vectorizado, imprimirlo en una tarjeta personal y, después, agrandarlo e imprimirlo en una valla manteniendo en ambas imágenes el mismo nivel de calidad. Los ejemplos más populares de formato de documentos que se deban imprimir son PDF y PostScript.
- Otra aplicación donde los gráficos vectoriales son importantes es el plotter de corte o impresora de corte de vinilo, ya que este, como su nombre indica, corta áreas de color diseñadas por el usuario a partir de un archivo digital.
- Estas figuras están construidas a partir de vectores que son interpretados por el plotter como las líneas límite por donde debe pasar la cuchilla que corta el material. Muy extendido en el gremio de la rotulación, la decoración ya sea de superficies planas o carrocerías de vehículos.

Contexto gráfico. Imágenes vectoriales

Ventajas

- Dependiendo de cada caso particular, las imágenes vectoriales pueden requerir menor espacio de almacenamiento que un mapa de bits. Las imágenes formadas por colores planos o degradados sencillos son más factibles de ser vectorizadas.
- A menor información para crear la imagen, menor será el tamaño del archivo. Dos imágenes con dimensiones de presentación distintas pero con la misma información vectorial, ocuparán el mismo espacio de almacenamiento.
- No pierden calidad al ser redimensionadas. En principio, se puede escalar una imagen vectorial de forma ilimitada. En el caso de las imágenes matriciales, se alcanza un punto en el que es evidente que la imagen está compuesta por píxeles.
- Los objetos definidos por vectores pueden ser guardados y modificados en el futuro.
- Algunos formatos permiten animación. Esta se realiza de forma sencilla mediante operaciones básicas como traslación o rotación y no requiere un gran acopio de datos, ya que lo que se hace es reubicar las coordenadas de los vectores en nuevos puntos dentro de los ejes *x*, *y*, y *z* en el caso de las imágenes 3D.

Contexto gráfico. Imágenes vectoriales

Desventajas

- Los gráficos vectoriales, en general, no son aptos para codificar fotografías o vídeos tomados en el «mundo real» (fotografías de la naturaleza, por ejemplo), aunque algunos formatos admiten una composición mixta (vector + mapa de bits). Prácticamente todas las cámaras digitales almacenan las imágenes en mapa de bits.
- Los datos que describen el gráfico vectorial deben ser procesados, es decir, el computador debe ser suficientemente potente para realizar los cálculos necesarios para formar la imagen final. Si el volumen de datos es elevado se puede volver lenta la representación de la imagen en pantalla, incluso trabajando con imágenes pequeñas.
- Por más que se construya una imagen con gráficos vectoriales su visualización tanto en pantalla, como en la mayoría de sistemas de impresión, en última instancia tiene que ser traducida a píxeles.

Contexto gráfico. Gráficos vectoriales escalables.

- Los **gráficos vectoriales escalables** o **gráficos vectoriales redimensionables** (del inglés: ***Scalable Vector Graphics*** (**SVG**) es un formato de gráficos vectoriales bidimensionales, tanto estáticos como animados, en formato de lenguaje de marcado extensible XML (Extensible Markup Language), es decir que se compone por código y cuya especificación es un estándar abierto desarrollado por el W3C desde 1999.
- A diferencia de aquellos gráficos codificados en JPG, PNG, o TIFF (Rasters), los SVG pueden ser interactivos y dinámicos y esto se debe a que no se componen por mapa de bits, sino que están compuestos por vectores, que son instrucciones matemáticas que se le dan al navegador o programas de ediciones de estos gráficos vectoriales, para escalarlos de manera infinita y sin perder resolución o calidad en el gráfico.
- Las imágenes SVG y sus comportamientos se definen en archivos de texto XML. Esto significa que se pueden buscar, indexar, codificar y comprimir. Como archivos XML, las imágenes SVG se pueden crear y editar con cualquier editor de texto o comúnmente editor de código, así como con software de dibujo.
- El SVG es un lenguaje usado para dibujar y representar gráficos, imágenes y logotipos, o sea que son gráficos que pueden manipularse con CSS y JavaScript.

Contexto gráfico. Gráficos vectoriales escalables.

Ventajas de SVG

- Las ventajas de usar el formato SVG sobre otros formatos de imagen son que estas se pueden:[3]
- Crear y editar con cualquier editor de texto.
- Buscar, indexar, codificar y comprimir.
- Imprimir con alta calidad en cualquier resolución.
- Pueden ser cambiadas de tamaño sin perder calidad de imagen o gráfico.
- Son escalables.
- SVG es un estándar abierto.
- Los archivos SVG están compuestos por código XML puro.

SVG usa formato XML

- Para poder entender un poco SVG desde el código se debe tener un conocimiento básico de HTML y XML.

Por lo tanto debe llevar:

- Etiquetas de apertura y cierre (*Al igual que HTML*)
- Atributos.
- Como dialecto XML, requiere que se defina el namespace, de lo contrario el navegador no lo interpretará correctamente.

Contexto gráfico. Gráficos vectoriales escalables.

La especificación de SVG permite tres tipos de objetos gráficos:

- Elementos geométricos vectoriales, como los caminos o trayectorias consistentes en rectas y curvas, y áreas limitadas por ellos.
- Imágenes de mapa de bits/digitales.
- Texto.

Los objetos gráficos pueden ser agrupados, transformados y compuestos en objetos previamente renderizados, y pueden recibir un estilo común. El texto puede estar en cualquier espacio de nombres XML admitido por la aplicación, lo que mejora la posibilidad de búsqueda y la accesibilidad de los gráficos SVG.

El conjunto de características incluye las transformaciones anidadas, las trayectorias de recorte, las máscaras alfa, los filtros de efectos, las plantillas de objetos y la extensibilidad.

Contexto gráfico. Imágenes - Unity

UI.Image – Muestra un Sprite para el sistema de UI.

- Más info: https://docs.unity3d.com/es/2018.4/ScriptReference/UI.Image.html

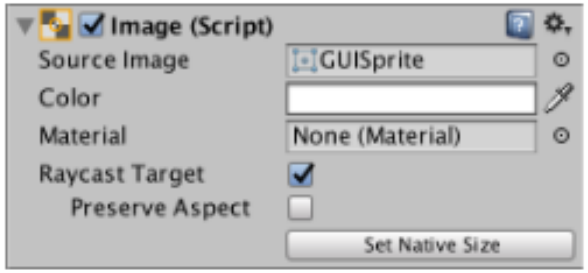
Image

- El control Imagen muestra una imagen no interactiva al usuario. Esto se puede usar para decoración, íconos, etc., y la imagen también se puede cambiar desde un script para reflejar cambios en otros controles.
- El control es similar al control Imagen sin formato pero ofrece más opciones para animar la imagen y archivar con precisión el rectángulo de control. Sin embargo, el control Imagen requiere que su textura sea un Sprite , mientras que la Imagen sin formato puede aceptar cualquier textura.

Contexto gráfico. Imágenes - Unity

- Link: https://docs.unity3d.com/560/Documentation/Manual/script-Image.html

Properties



Property:	Function:
Source Image	The texture that represents the image to display (which mu
Color	The color to apply to the image.
Material	The Material to use for rendering the image.
Raycast Target	Should this be considered a target for raycasting?
Preserve Aspect	Ensure image remains existing dimension.
Set Native Size	Button to set the dimensions of the image box to the origir

Details

The image to display must be imported as a Sprite to work with the Image contro

Eventos de teclado - Unity.

Teclado móvil

- En la mayoría de los casos, Unity manejará input de teclado automáticamente para elementos GUI pero también es fácil mostrar el teclado en demanda desde un script.

Elementos GUI

- El teclado aparecerá automáticamente cuando un usuario golpee en elementos GUI editables.
  Actualmente, GUI.TextField, GUI.TextArea y GUI.PasswordField va a mostrar el teclado; vea la documentación GUI class para más detalles.

Manejo del teclado manual

- Use la función **TouchScreenKeyboard.Open()** para abrir el teclado. Por favor ver la referencia del scripting TouchScreenKeyboard para los parámetros que está función toma.

Eventos de teclado - Unity.

Opciones de distribución del teclado

El teclado soporta las siguientes opciones:-

Propiedad:	Función:
TouchScreenKeyboardType.Default	Letras. Pueden ser cambiadas al teclado con números y puntuación.
TouchScreenKeyboardType.ASCIICapable	Letras. Pueden ser cambiadas al teclado con números y puntuación.
TouchScreenKeyboardType.NumbersAndPunctuation	Números y puntuación. Pueden ser cambiadas al teclado con letras.
TouchScreenKeyboardType.URL	Letras con barras y botones .com .Pueden ser cambiadas a teclado con números y puntuación.
TouchScreenKeyboardType.NumberPad	Solo números de 0 a 9.
TouchScreenKeyboardType.PhonePad	Teclado usado para introducir números de teléfono.
TouchScreenKeyboardType.NamePhonePad	Letras. Pueden ser cambiadas al teclado de teléfono.
TouchScreenKeyboardType.EmailAddress	Letras con signo @. Pueden ser cambiadas a teclado con números y puntuación.

Eventos de teclado - Unity.

Vista Preliminar de Texto

- Por defecto, una caja de edición va a ser creada y colocada encima del teclado después de que aparezca. Esto funciona como una vista preliminar del texto que el usuario esté escribiendo, entonces el texto es siempre visible para el usuario. No obstante, puede deshabilitar la vista preliminar del texto seleccionando **TouchScreenKeyboard.hideInput** a true.
- Tenga en cuenta que esto funciona solo para ciertos tipos de teclado y modos de input. Por ejemplo, esto no va a servir para los teclados de teléfono e input de texto de varias lineas. En esos casos, la caja de edición siempre va aparecer.
- **TouchScreenKeyboard.hideInput** es una variable global y va afectar todos los teclados.

Eventos de teclado - Unity.

Visibilidad y tamaño del teclado

Hay tres propiedades de teclado en TouchScreenKeyboard que determina estatus de la visibilidad del teclado y el tamaño en la pantalla.

Propiedad:	Función:
visible	Devuelve **true** si el teclado es completamente visible en la pantalla y puede ser usada para introducir caracteres.
area	Devuelve la posición y dimensiones del teclado.
active	Devuelve **true** si el teclado está activado. Esta propiedad no es una propiedad estática. Usted debe tener una instancia de teclado para usar esta propiedad.

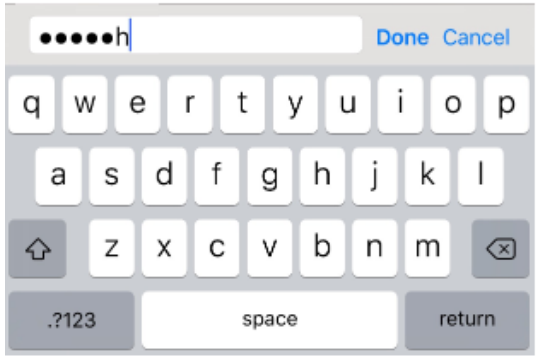
Tenga en cuenta que **TouchScreenKeyboard.area** va a devolver un Rect con una posición y tamaño establecido en 0 hasta que el teclado sea completamente visible en la pantalla. Usted no debe consultar este valor inmediatamente después de **TouchScreenKeyboard.Open()**. La secuencia de eventos de teclado es la siguiente:

- **TouchScreenKeyboard.Open()** es llamada. **TouchScreenKeyboard.active** devuelve true. **TouchScreenKeyboard.visible** devuelve false. **TouchScreenKeyboard.area** devuelve (0, 0, 0, 0).
- El teclado se desliza afuera de la pantalla. Todas las propiedades se mantienen igual.
- El teclado para de deslizarse. **TouchScreenKeyboard.active** devuelve true. **TouchScreenKeyboard.visible** devuelve true. **TouchScreenKeyboard.area** devuelve la verdadera posición y tamaño del teclado.

Eventos de teclado - Unity.

- Ejemplo:

```
TouchScreenKeyboard.Open("", TouchScreenKeyboardType.Default, false, false, true);
```



- Más información: https://docs.unity.cn/es/2017.4/Manual/MobileKeyboard.html

Técnicas de animación.

- La **animación** es un proceso utilizado por uno o más animadores, para dar la ilusión de movimiento a imágenes, dibujos u otro tipo de objetos inanimados (figuras de plastilina, por ejemplo). Se considera, normalmente, una ilusión óptica. Existen numerosas técnicas para realizar una animación que van más allá de los familiares dibujos animados.
- Los cuadros se pueden generar dibujando, pintando o fotografiando los minúsculos cambios hechos repetidamente a un modelo de la realidad o a un modelo tridimensional virtual; también es posible animar objetos reales y actores. Entre los formatos de archivo de animación (o que soportan animación) se encuentran el GIF, el SVG, etc. Las animaciones en GIF son guardadas imagen por imagen; sin embargo, existen animaciones que no se logran así, sino que son interpretadas y "armadas" en tiempo real al ejecutarse, como el SVG.

Técnicas de animación.

Técnicas de animación Dibujos animados.

- Los dibujos animados se crean dibujando los fotogramas, uno por uno (24 de ellos por cada segundo de animación), siguiendo la técnica desarrollada principalmente por los animadores de Disney a principios del siglo XX.
- El proceso comienza con el animador dibujando cada fotograma en papel. Después el dibujo se realiza de nuevo con tinta y se pinta en láminas de acetato. Finalmente el dibujo se fotografía con una cámara estática. Las fotografías se colocan en secuencia para dar la ilusión de movimiento. Esta práctica, sin embargo, ha estado cayendo en descenso desde hace dos décadas, debido a la aparición de la computadora y las facilidades que esta provee para la creación de animación de una manera más rápida y barata.

Técnicas de animación.

Técnicas de animación – Animación en Volumen.

- A la «**animación en volumen**» se le conoce también con los términos «**animación fotograma por fotograma**», «animación cuadro por cuadro», «parada de imagen», «paso de manivela» o «animación foto a foto». En los últimos años también se ha popularizado su nombre en inglés: *stop motion*.
- En ella no se animan dibujos o imágenes planas sino objetos estáticos e inmóviles colocados delante de una cámara. Consiste en aparentar el movimiento de dichos objetos capturando fotogramas: en cada uno de estos, se ha movido ligeramente el objeto y, en cada nuevo cambio de posición, debe haberse siempre orientado el objeto en una cierta dirección en relación con el cambio de posición y fotograma anteriores, guardando así la mayor continuidad lógica del movimiento que se quiere imitar.
- Más tarde, al reproducir los fotogramas uno detrás de otro (como se hace de hecho con cualquier proyección cinematográfica obtenida mediante filmación real), la proyección en pantalla crea la ilusión óptica de que el objeto se mueve por sí mismo.

Técnicas de animación.

Técnicas de animación – Animación en Volumen.

- Por otro lado, la animación en volumen tiene un realismo fotográfico completamente ausente en un dibujo animado, ya que en este la profundidad de campo es una simple ilusión óptica, realizada con mayor o menor verosimilitud. Mientras, en la animación en volumen, la profundidad de campo es auténtica, puesto que se obtiene mediante filmación convencional.
- La única diferencia con el *stop motion* es que éste es una filmación obtenida manualmente, fotograma por fotograma, y no automáticamente y en tiempo real como es el caso de una filmación convencional.
- En general, las animaciones que no entran en la categoría de dibujo animado, (es decir, que no fueron dibujadas ni pintadas, sino que fueron creadas tomando imágenes de la realidad), son referidas como animaciones en volumen o *stop motion*. Tradicionalmente, hay dos grandes grupos de animaciones *stop motion*: la animación de plastilina (o cualquier material maleable; en inglés *claymation)*, y las animaciones de objetos (más rígidos).

Técnicas de animación.

Animación con recortes o *cut-out*

- La animación con recortes, más conocida en inglés como *cut-out*, es la técnica en que se usan figuras recortadas, ya sea de papel o incluso fotografía.
- Los cuerpos de los personajes se construyen con los recortes de sus partes, moviendo y reemplazándolas. Así, se obtienen diversas poses y se da vida al personaje.
- Un ejemplo muy claro de esta técnica lo encontramos en el vídeo musical *Live for the moment*, de la banda Irlandesa Verona Riots, realizado por Alberto Serrano y Nívola Uyá en 2014.

Técnicas de animación.

Plastimación o *claymation*

- La plastimación, conocida en inglés como *claymation*, es la animación con arcilla, plastilina o cualquier otro material moldeable.
- Puede hacerse al «estilo libre», cuando no hay una figura definida, sino que las figuras se van transformando en el progreso de la animación (como lo hacen los gatos Mio y Mao en la serie italiana *Mio Mao*); o puede orientarse a personajes, que mantienen una figura constante en el transcurso de la película.

Técnicas de animación.

Pixilación

- La **pixilación** es una variante del *stop motion*, en la que los objetos animados son personas y auténticos objetos comunes (no modelos ni maquetas).
- Al igual que en cualquier otra forma de animación, estos objetos son fotografiados repetidas veces, y desplazados ligeramente entre cada fotografía.
- Norman McLaren popularizó esta técnica, empleada en su famoso corto animado *Neighbours";* pero, ya en 1908, el aragonés Segundo de Chomón, utilizaba en su obra *Hotel eléctrico* la misma técnica para animar objetos. Es ampliamente utilizada en los videoclips.

Técnicas de animación.

Go Motion.

- La animación *go motion* es una variante del *stop motion*, inventado por Phil Tippett para la película de 1980 *El Imperio contraataca*.
- El *go motion* consiste en obtener cada fotograma E E mientras se sacude ligeramente el objeto, una parte de este. El efecto borroso resultante sobre las partes en movimiento -el llamado barrido de movimiento- aumenta de este modo la sensación de realismo en la animación resultante.
- En las filmaciones realizadas en tiempo real, cuando un objeto es más rápido que la velocidad de obturación de la cámara, el objeto aparece borroso en algunos fotogramas, a pesar de que la proyección de la película sea de un realismo impecable, y este es el efecto buscado por la técnica de animación por *go motion*.

Técnicas de animación.

Rotoscopia

- La rotoscopia es una técnica de animación que recurre a una máquina llamada rotoscopio. El rotoscopio tiene una placa de vidrio sobre la que se pueden colocar láminas transparentes de acetato o papel.
- Debajo, un proyector ilumina el fotograma de la filmación realizada en tiempo e imagen real. De este modo se puede calcar el contorno de los objetos filmados. *Koko the Clown*, del estudio Fleischer estaba animado con rotoscopia.
- Se especula que en *Blancanieves*, de Walt Disney, se utilizó rotoscopia. Pero los artistas solo usaban modelos de acción real como referencias, no se calcaba el material filmado. En animación por computadora, la técnica análoga a la rotoscopia es la técnica por captura de movimiento.

Técnicas de animación.

Animación por computadora

- La **animación por computadora**, también llamada animación digital, animación informática o animación por ordenador; es la técnica que consiste en crear imágenes en movimiento mediante el uso de ordenadores o computadoras.
- Cada vez los gráficos creados en 3D son más, aunque los gráficos en 2D todavía se siguen usando ampliamente, para conexiones lentas y aplicaciones en tiempo real que necesitan renderizar rápido.
- Algunas veces, el objetivo de la animación es la computación en sí misma; otras, puede ser otro medio, como una película. Los diseños se elaboran con la ayuda de programas de diseño, modelado y, por último, renderizado.
- En la animación, sin embargo, las imágenes no se toman, sino que se producen individualmente, y, por ello, no tienen que cumplir necesariamente con el estándar del cine. Una película de animación tiene siempre 24 fotogramas por segundo, pero no necesariamente todos esos fotogramas muestran imágenes diferentes ya que suelen repetirse en varios fotogramas.

Así pues, tenemos varias tasas de animación:

- **Cada imagen es diferente**, sin repetición. 24 imágenes por segundo, una imagen cada fotograma.
- **Cada imagen se repite una vez**. 12 imágenes por segundo, una imagen cada 2 fotogramas.
- **Cada imagen se repite dos veces**. 8 imágenes por segundo, una imagen cada 3 fotogramas.

Técnicas de animación.

Time Lapse

- El Time Lapse es una técnica fotográfica que consiste en la captación de imágenes fijas que después son reproducidas a una velocidad mayor.
- El Time Lapse se ha usado bastante en la televisión. La suelen usar para hacer un salto de tiempo, ya sea un día entero, de la mañana a la noche, o cualquier salto de tiempo. El uso de la velocidad cambia dependiendo el tiempo que quieras que pase o lo rápido que quieres que se vea.

Otras técnicas

- Virtualmente, cualquier forma de producir imágenes o cualquier materia que pueda ser fotografiada puede utilizarse para animar. Existen muchas técnicas de animación que solo han sido utilizadas por unos y que son desconocidas para el gran público. Entre estas se incluyen pintura sobre cristal, animación de arena, pantalla de agujas, pintura sobre celuloide, *tweening*, etc.

Técnicas de sonido.

- La **reproducción y grabación de sonido** es la inscripción eléctrica o mecánica y la recreación de ondas sonoras, como la voz, el canto, la música instrumental o efectos sonoros. Las dos clases principales de tecnologías de grabación de sonido son la grabación analógica y la grabación digital.
- La **grabación analógica acústica** se logra con un pequeño micrófono de diafragma que puede detectar cambios en la presión atmosférica (ondas de sonido acústicas) y grabarlas como ondas de sonido gráficas en un medio como un fonógrafo (en el que un estilete hace surcos helicoidales sobre un cilindro de fonógrafo) o una cinta magnética (en la que la corriente eléctrica del micrófono es convertidas a fluctuaciones electromagnéticas que modulan una señal eléctrica).
- La **reproducción de sonido analógico** es el proceso inverso, en el que un altavoz de diafragma de mayor tamaño causa cambios en la presión atmosférica para formar ondas de sonido acústicas. Las ondas de sonido generadas por electricidad también pueden ser grabadas directamente mediante dispositivos como los altavoces de una guitarra eléctrica o un sintetizador, sin el uso de acústica en el proceso de grabación, más que la necesidad de los músicos de escuchar que tan bien están tocando durante las sesiones de grabación.

Técnicas de sonido.

Durante la grabación, se realiza un proceso de transducción en el cual la señal de audio es transformada en variaciones de voltaje que pueden almacenarse de distintos modos.

Las fuentes pregrabadas utilizan soportes muy diferentes donde almacenar la señal de audio, todo dependerá de la modalidad de grabación de sonido empleada.

- **Grabación analógica de sonido:**
  - Grabación magnética analógica o grabación electromagnética analógica.
  - Grabación óptica analógica o grabación fotográfica del sonido.
- **Grabación digital de sonido:**
  - Grabación magnética digital.
  - Grabación óptica digital.
  - Grabación magneto-óptica digital.

Bases de datos y almacenamiento.

- Una **base de datos** (del inglés: **database**) se encarga no solo de almacenar datos, sino también de conectarlos entre sí en una unidad lógica. En términos generales, una base de datos es un conjunto de datos estructurados que pertenecen a un mismo contexto y, en cuanto a su función, se utiliza para administrar de forma electrónica grandes cantidades de información.
- En este sentido; una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta.
- Actualmente, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital, siendo este un componente electrónico; por tanto, se ha desarrollado y se ofrece un amplio rango de soluciones al problema del almacenamiento de datos.

Bases de datos y almacenamiento.

Las bases de datos pueden clasificarse de varias maneras, de acuerdo con el contexto que se esté manejando, la utilidad de las mismas o las necesidades que satisfagan.

Según la variabilidad de la base de datos

Bases de datos estáticas

- Son bases de datos únicamente de lectura, utilizadas principalmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones, tomar decisiones y realizar análisis de datos para inteligencia empresarial.

Bases de datos dinámicas

- Son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización, borrado y edición de datos, además de las operaciones fundamentales de consulta. Un ejemplo puede ser la base de datos utilizada en un sistema de información de un supermercado.

Bases de datos y almacenamiento.

Según el contenido

Bases de datos bibliográficas

- Solo contienen una subrogante (representante) de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque si no, estaríamos en presencia de una base de datos a texto completo (o de fuentes primarias —ver más abajo). Como su nombre lo indica, el contenido son cifras o números. Por ejemplo, una colección de resultados de análisis de laboratorio, ayuda mucho a la redundancia de datos.

Bases de datos de texto completo

- Almacenan las fuentes primarias, cómo por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

Bases de datos y almacenamiento.

Directorios

- Estos directorios se pueden clasificar en dos grandes tipos dependiendo de si son personales o empresariales (llamadas páginas blancas o amarillas respectivamente).
- Los directorios empresariales hay de tres tipos:
  1. Tienen nombre de la empresa y dirección.
  2. Contienen teléfono y los más avanzados contienen correo electrónico.
  3. Contienen datos como facturación o número de empleados además de códigos nacionales que ayudan a su distinción.
- Los directorios personales solo hay de un tipo, ya que leyes como la LOPD en España protege la privacidad de los usuarios pertenecientes al directorio. La búsqueda inversa está prohibida en los directorios personales (a partir de un número de teléfono saber el titular de la línea).

Bases de datos o "bibliotecas" de información química o biológica

- Son bases de datos que almacenan diferentes tipos de información proveniente de la química, las ciencias de la vida o médicas. Se pueden considerar en varios subtipos:
- Las que almacenan secuencias de nucleótidos o proteínas. Las bases de datos de rutas metabólicas.
- Bases de datos de estructura, comprende los registros de datos experimentales sobre estructuras 3D de biomoléculas. Bases de datos clínicas. Bases de datos bibliográficas.

Bases de datos y almacenamiento - Unity.

Base de Datos de Activos

- Con la mayoría de los tipos de activos, Unity necesita convertir los datos del archivo fuente del activo a un formato que pueda usar en un juego o aplicación en tiempo real. Almacena estos archivos convertidos y los datos asociados con ellos en la **base de datos de activos** .
- El proceso de conversión es necesario porque la mayoría de los formatos de archivos están optimizados para ahorrar espacio de almacenamiento, mientras que en un juego o una aplicación en tiempo real, los datos de los recursos deben estar en un formato que esté listo para el hardware, como la CPU, los gráficos o hardware de audio, para utilizarlo inmediatamente. Por ejemplo, cuando Unity importa un archivo de imagen .png como textura, no utiliza los datos originales con formato .png en tiempo de ejecución. En cambio, cuando importa la textura, Unity crea una nueva representación de la imagen en un formato diferente que se almacena en la carpeta Biblioteca del proyecto . La clase Texture en el motor de Unity usa esta versión importada y Unity la carga en la GPU para su visualización en tiempo real.
- Si posteriormente modifica el archivo fuente de un recurso que ya importó (o si cambia cualquiera de sus dependencias), Unity vuelve a importar el archivo y actualiza la versión importada de los datos. Consulte Actualización de la base de datos de activos para obtener más información sobre este proceso.
- La base de datos de activos también proporciona una API de AssetDatabase que puede utilizar para acceder a los activos y controlar o personalizar el proceso de importación.

Bases de datos y almacenamiento - Unity.

Dependencias de importación de activos

- La base de datos de activos realiza un seguimiento de todas las dependencias de cada activo y mantiene un caché de las versiones importadas de todos los activos.
- Las dependencias de importación de un activo constan de todos los datos que podrían influir en los datos importados. Por ejemplo, el archivo fuente de un Activo es una dependencia, así como la configuración de importación del Activo (como el tipo de compresión de una textura) o la plataforma de destino de su Proyecto (por ejemplo, el hardware de PS4 requiere datos en un formato diferente al de Android). hardware). Si modifica cualquiera de estas dependencias, la versión almacenada en caché del activo importado deja de ser válida y Unity debe volver a importarlo para reflejar los cambios.

Bases de datos y almacenamiento - Unity.

Almacenamiento en caché de activos

- El **Asset Cache** es donde Unity almacena las versiones importadas de los activos. Debido a que Unity siempre puede recrear estas versiones importadas a partir del archivo de recursos de origen y sus dependencias, estas versiones importadas se tratan como un caché de datos precalculados, lo que ahorra tiempo cuando usas Unity. Por este motivo, debe excluir los archivos en Asset Cache de control de versiones sistemas.
- Unity usa un caché local de forma predeterminada, lo que significa que las versiones importadas de los Activos se almacenan en caché en la carpeta Biblioteca en la carpeta de su Proyecto en su máquina local. Deberías usar un ignorar archivo para excluir esta carpeta del control de versiones.
- Sin embargo, si trabaja como parte de un equipo y utiliza un sistema de control de versiones, podría resultar beneficioso utilizar también Unity Accelerator , que comparte el caché de activos en su LAN.
- Debido a que los Activos almacenados en caché no son adecuados para almacenar en un sistema de control de versiones, cuando su equipo trabaja en conjunto en un Proyecto y usa el almacenamiento en caché local, la copia de Unity de cada miembro del equipo realiza el proceso de importación si un Activo o una dependencia cambia, lo que puede llevar mucho tiempo.
- Unity proporciona una solución para esto llamada **Unity Accelerator** . Una de las características del Acelerador es un agente de software que almacena y entrega las versiones en caché de los Activos a todos los que trabajan juntos en el mismo Proyecto en la misma red local. Esto significa que solo un miembro del equipo necesita importar un activo determinado. Luego, la versión importada del activo se almacena en el Acelerador y los demás miembros del equipo pueden descargar la versión almacenada en caché en lugar de esperar el proceso de importación localmente.

Bases de datos y almacenamiento - Unity.

Activos y artefactos de origen

- Unity mantiene dos archivos de base de datos en la carpeta Biblioteca, que en conjunto se denominan Base de datos de activos. Estas dos bases de datos realizan un seguimiento de la información sobre los archivos de activos de origen y los artefactos, que es información sobre los resultados de la importación.

La base de datos de activos de origen

- La base de datos de activos de origen contiene metainformación sobre los archivos de activos de origen que Unity utiliza para determinar si el archivo se ha modificado y, por lo tanto, si se deben volver a importar los archivos. Esto incluye información como la fecha de la última modificación, un hash del contenido del archivo, GUID y otra metainformación.

La base de datos de artefactos

- Los artefactos son el resultado del proceso de importación. La base de datos de Artifact contiene información sobre los resultados de la importación de cada activo de origen. Cada artefacto contiene información de dependencia de importación, metainformación de artefacto y una lista de archivos de artefacto.
- Nota: Los archivos de la base de datos se encuentran en la carpeta Biblioteca de su proyecto y, como tal, debe excluirlos de los sistemas de control de versiones. Puedes encontrarlos en las siguientes ubicaciones:

Base de datos de activos de origen: Library\SourceAssetDB

Base de datos de artefactos: Library\ArtifactDB

Bases de datos y almacenamiento - Unity.

Cambio de plataforma y reimportación

- Cambiar entre plataformas puede hacer que Unity vuelva a importar tus activos. Esto suele ocurrir cuando la forma en que se importa el activo difiere entre plataformas, lo que suele ser el caso. Por ejemplo, diferentes plataformas tienen diferentes formatos de textura, por lo que las texturas se importan de forma diferente para cada plataforma.
- Cuando se utiliza Asset Database V2, la plataforma es parte del hash que Asset Database utiliza para almacenar los resultados de importación para los importadores integrados de Unity. Esto significa que los resultados de la importación de sus activos en diferentes plataformas se almacenan como datos separados en caché.
- El resultado de esta característica es que la primera vez que cambia de plataforma con nuevos recursos en su proyecto que aún no se han importado para esa plataforma, se vuelven a importar. Esto significa que debe esperar a que se complete ese proceso. Sin embargo, los nuevos datos reimportados no sobrescriben los datos antiguos importados en caché para la plataforma anterior.
- Esto significa que cada vez que vuelva a cambiar a una plataforma donde ya haya importado activos para esa plataforma, esos resultados de importación de activos ya estarán almacenados en caché y listos para usar, lo que hace que el cambio sea mucho más rápido.

Versión de la base de datos de activos

- Esta documentación hace referencia a la versión 2 de Asset Database, que es la predeterminada en los nuevos proyectos creados con Unity 2019.3 o posterior. La versión heredada (versión 1) era la predeterminada en versiones anteriores de Unity, que se comporta de manera diferente. La versión heredada no se puede utilizar en Unity 2020+.

Bases de datos y almacenamiento - Unity.

Importar un activo

Unity normalmente importa los activos automáticamente cuando se arrastran al proyecto, pero también es posible importarlos bajo el control de un script. Para hacer esto, puede utilizar el método AssetDatabase.ImportAsset como en el siguiente ejemplo.

```
using UnityEngine;
using UnityEditor;

public class ImportAsset {
    [MenuItem ("AssetDatabase/ImportExample")]
    static void ImportExample ()
    {
        AssetDatabase.ImportAsset("Assets/Textures/texture.jpg", ImportAssetOptions.Default);
    }
}
```

Bases de datos y almacenamiento - Unity.

Cargando un activo

El editor carga recursos solo cuando es necesario, por ejemplo, si se agregan a la escena o editado desde el Inspector panel. Sin embargo, puede cargar y acceder a activos desde una secuencia de comandos utilizando AssetDatabase.LoadAssetAtPath , AssetDatabase.LoadMainAssetAtPath , AssetDatabase.LoadAllAssetRepresentationsAtPath y AssetDatabase.LoadAllAssetsAtPath . Consulte la documentación de secuencias de comandos para obtener más detalles.

```
using UnityEngine;
using UnityEditor;

public class ImportAsset {
    [MenuItem ("AssetDatabase/LoadAssetExample")]
    static void ImportExample ()
    {
        Texture2D t = AssetDatabase.LoadAssetAtPath("Assets/Textures/texture.jpg", typeof(Texture2D)) as Texture2D;
    }
}
```

Más información: https://docs.unity3d.com/Manual/AssetDatabase.html

Persistencia de datos.

- La **persistencia de datos** es la representación residual de datos que han sido de alguna manera nominalmente borrados o eliminados. Este residuo puede ser debido a que los datos han sido dejados intactos por un operativo de eliminación nominal, o por las propiedades físicas del medio de almacenaje. La persistencia de datos posibilita en forma inadvertida la exhibición de información sensible si el medio de almacenaje es dejado en un ambiente sobre el que no se tiene control (p. ej., se tira a la basura, se le da a un tercero).
- Con el correr del tiempo, se han desarrollado varias técnicas para contrarrestar la persistencia de datos. Dependiendo de su efectividad y de su intención, a menudo se los clasifica como compensación o purga/higienización. Métodos específicos incluyen la sobre escritura, la desmagnetización, el cifrado, y la destrucción física.

Persistencia de datos.

Habitualmente se reconocen tres niveles de eliminación de datos persistentes:

Compensación

- Compensación es la remoción de los datos sensibles de un medio de almacenamiento de tal manera que hay seguridad de que los datos no podrán ser reconstruidos utilizando las funciones normales del sistema o programas de recuperación de archivos/datos. Los datos pueden aún ser recuperables, pero eso requerirá técnicas especiales de laboratorio.[1]

Purga

- Purga o higienización es la remoción de datos sensibles de un dispositivo de almacenamiento con el objeto de que los datos no puedan ser reconstruidos utilizando alguna de las técnicas conocidas. La purga, proporcional a la sensibilidad de los datos, generalmente se efectúa antes de dejar libres de control los dispositivos de almacenamiento, en los casos en que se descartan viejos medios de almacenamiento o se trasladan dichos medios a computadoras con diferentes requerimientos de seguridad.

Destrucción

- El medio de almacenamiento es físicamente destruido. Su efectividad varía. Dependiendo de la densidad de grabación del medio y/o de la técnica de destrucción, esta técnica puede dejar datos recuperables por métodos de laboratorio. A su vez, la destrucción física cuando se utilizan los métodos apropiados, es generalmente considerada como el método más seguro posible.

Envío y recepción de mensajes - Unity

Messaging System (Sistema de Mensajería)

- El nuevo sistema de UI utiliza un messaging system (sistema de mensajería) diseñado para remplazar SendMessage. El sistema es puramente C# y apunta a abarcar algunos de los problemas presentes con SendMessage. El sistema funciona utilizando interfaces personalizadas que pueden ser implementas en MonoBehaviour para indicar que el componente es capaz de recibir un callback del sistema de mensajería.
- Cuando la llamada es hecha, un GameObject objetivo es especificado; la llamada será emitida en todos los componentes del GameObject que implementan la infería específica que la llamada está emitida contra. El sistema de mensajería permite que datos del usuario personalizados sean pasados; así como hasta qué punto a través de la jerarquía del GameObject el evento debería ser propagado; esto es que debería ser ejecutado para el GameObject especifico, o debería también ejecutarse en hijos y padres. En adición a esto, el framework de mensajería proporciona funciones de ayuda para buscar y encontrar GameObjects que implementan un interfaz de mensajería dada.
- El sistema de mensajería es genérico y diseñado para el uso no solo por el sistema de UI pero también por código de juego general. Es relativamente trivial agregar eventos de mensajería personalizados y estos van a funcionar utilizando la misma framework que el sistema Ui utiliza para todo el manejo de eventos.

Envío y recepción de mensajes - Unity

Cómo Defino un Mensaje Personalizado?

Si usted desea definir un mensaje personalizado es relativamente simple. En el namespace UnityEngine.EventSystems hay una interfaz base llamada 'IEventSystemHandler'. Cualquier cosa que extienda de esto puede ser considerado como un objetivo para recibir eventos vía el sistema de mensajería.

```
public interface ICustomMessageTarget : IEventSystemHandler
{
    // functions that can be called via the messaging system
    void Message1();
    void Message2();
}
```

Envío y recepción de mensajes - Unity

Una vez esta interfaz haya sido definido entonces puede ser implementado por un MonoBehavior. Cuando sea implementando, éste define las funciones que serán ejecutadas si el mensaje dado es emitido contra ese GameObject de MonoBehaviours.

```
public class CustomMessageTarget : MonoBehaviour, ICustomMessageTarget
{
    public void Message1()
    {
        Debug.Log ("Message 1 received");
    }

    public void Message2()
    {
        Debug.Log ("Message 2 received");
    }
}
```

Envío y recepción de mensajes - Unity

Ahora que un script existe que puede recibir el mensaje, nosotros necesitamos emitir el mensaje. Normalmente esto sería en respuesta a algún evento débilmente acoplado que ocurre. Por ejemplo, en el sistema UI nosotros emitimos eventos para tales cosas como PointerEnter y PointerExit, al igual que una variedad de otras cosas que pueden suceder en respuesta al input del usuario a la aplicación.

Para enviar un mensaje, una clase estática de ayuda existe para esto. Como argumentos, requiere un objeto objetivo para el mensaje, algunos datos específicos del usuario, y un funtor que mapea a la función específica en la interfaz de mensajes que usted desea orientarse.

```
ExecuteEvents.Execute<ICustomMessageTarget>(target, null, (x,y)=>x.Message1());
```

Este código va a ejecutar la función Message1 en cualquier componente en el GameObject objetivo que implemente la interfaz ICustomMessageTarget. La documentación de scripting para la clase ExecuteEvents cubre otras formas de Execute (Ejecutar) funciones, como Executing (Ejecutar) en hijos o padres.

Servicios web.

- Un servicio web (en inglés: Web Service) es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet.
- La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web.
- Para mejorar la interoperabilidad entre distintas implementaciones de servicios web se ha creado el organismo WS-I, encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares. Es una máquina que atiende las peticiones de los clientes web y les envía los recursos solicitados.

Servicios web - Unity

Unity Services Web APIs

- Link a documentación:
- https://services.docs.unity.com/
- Dado que aún estamos iniciándonos con Unity os dejo las referencias con los Servicios Web de Unity para investigarlas. Lo iremos desarrollando a lo largo del curso.

Bibliografía

- https://es.wikipedia.org/wiki/Dise%C3%B1o_de_interfaces_de_aplicaciones
- https://es.wikipedia.org/wiki/Desarrollo_r%C3%A1pido_de_aplicaciones
- https://es.wikipedia.org/wiki/API
- https://es.wikipedia.org/wiki/Imagen
- https://es.wikipedia.org/wiki/Imagen_digital
- https://es.wikipedia.org/wiki/Animaci%C3%B3n
- https://es.wikipedia.org/wiki/Grabaci%C3%B3n_y_reproducci%C3%B3n_de_sonido
- https://docs.unity3d.com/Manual/AssetDatabase.html
- https://es.wikipedia.org/wiki/Base_de_datos
- https://docs.unity3d.com/es/530/Manual/MessagingSystem.html