UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

Aplicación educativa para el entrenamiento de la memoria de personas con Síndrome de Down.

Cristina Correcher Esteve

Tutora: Clara Pérez Fuster

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2017-18

Valencia, 24 de diciembre de 2017

Resumen

El presente TFG pretende poner al alcance de un colectivo de personas con Síndrome de Down las nuevas tecnologías, y que estas sirvan para mejorar y potenciar sus capacidades de aprendizaje.

Para ello se ha realizado una aplicación con Unity3D, dividida en 3 etapas y 9 actividades en total, cuyo uso sea sencillo a la vez que divertido, y en el que ellos se sientan familiarizados y cómodos. El contenido está orientado a desarrollar su aprendizaje, sin necesidad de ir cargados con los pesados libros, a la vez que les resulte atractivo y les motive a practicar.

También cuenta con una base de datos que guardará las puntuaciones que van generando los usuarios para incentivar su competitividad.

Dichas actividades han sido supervisadas por profesionales de Asindown, entidad sin ánimo de lucro cuyo principal objetivo es facilitar y promover el desarrollo integral de las personas con Síndrome de Down.

Se ha hecho un estudio de costes de desarrollo para su inclusión en app-stores.

Resum

El present TFG preten posar a l'abast d'un col·lectiu de persones amb Síndrome de Down les noves tecnologies, i que estes servisquen per a millorar i potenciar les seues capacitats d'aprenentatge.

Per a aquest fi s'ha realitzat una aplicació amb Unity3D, dividida en 3 etapes i 9 activitats en total, l'ús del qual siga senzill al mateïx temps que divertit, i en el que ells es senten familiaritzats i còmodes. El contingut està orientat a desenvolupar el seu aprenentatge, sense necessitat d'anar carregats amb pesats llibres, al mateïx temps que els resulte atractiu i els motive a practicar.

També compta amb una base de dades que guardarà les puntuacions que van generant els usuaris per incentivar la seua competitivitat.

Aquestes activitats han sigut supervisades per professionals d'Asindown, entitat sense ànim de lucre el objectiu de la qual és facilitar i promoure el desenvolupament integral de les persones amb Síndrome de Down.

S'ha fet un estudi de costos de desenvolupament per a la seua inclusió en app-stores.

Abstract

The present TFG pretends to put within reach of a people's collective with Down's Syndrome the new technologies, and these to serve to improve and develop their capacities of learning.

To get it an app has been developed with Unity3D, divided in 3 steps and 9 activities, whose use has to be as easy as fun, and in which they feel familiar and comfortable. Content is oriented to develop their learning, without need to load heavy books, at the same time it to be attractive and motivates them to practice.

It also has a database that will save the punctuation that users generate to stimulate their competitivity.

Those activities have been supervised by professionals from Asindown, entity without encouragement of profit whose main objective is to make easy and promote the integral development of the people with Down's Syndrome.

A developments costs' study has been done for its inclusion in app-stores.

Índice

Capítulo 1. Introducción y antecedentes

1.1 Introducción

A medida que va avanzando el tiempo, la tecnología avanza con nosotros hasta formar un papel importante en el día a día. Hacer de la tecnología algo útil y que facilite la vida de las personas es el objetivo de todo proyecto. Siguiendo esta filosofía se pretende crear una aplicación educativa orientada a personas con discapacidad, tales como el Síndrome de Down, autismo, etc, con el fin de ayudar a estas personas a potenciar y mejorar sus capacidades de aprendizaje.

La idea es que estas personas puedan entrenarse y preparar su día a día de una forma divertida y sencilla con algo con lo que ellos se sienten familiarizados y cómodos, como son los móviles, con sistema operativo tanto Android como iOS. La aplicación debe cumplir las características propias de un juego que pueda usar todo el mundo desde cualquier lugar, en cualquier momento y a cualquier hora.

El proyecto deberá permitir a las personas que lo utilicen escoger la etapa en la que se encuentren (formación temprana, etapa educativa, formación adultos, etapa de empleo y envejecimiento), y realizar los distintos ejercicios de repaso y apoyo que correspondan a cada etapa. A la vez que deberán llevar un seguimiento de las puntuaciones para ver las mejoras.

Para su desarrollo se han de tener en cuenta los principios básicos que debe cumplir cualquier aplicación destinada a personas con estas discapacidades. Además, requerirá de una fase de pruebas detallada y precisa, teniendo en cuenta cualquier defecto o pequeño detalle que pueda ser importante para que los usuarios utilicen la aplicación con facilidad y sin problemas.

Como se verá en los capítulos de la memoria, todas las informaciones necesarias a la hora de diseñar los distintos ejercicios de la aplicación serán consultados y supervisados por personas profesionales en este sector, siendo la organización Asindown Valencia la encargada de su supervisión.

Esta memoria presenta de forma detallada el desarrollo del proyecto y de todas las fases de diseño, implementación y pruebas, contando con una pequeña introducción de las personas con estas discapacidades a las que está dedicado el proyecto.

1.2 Motivación

La tecnología juega un papel importante en las vidas de las personas. Con los avances que ha habido en las últimas dos décadas, hemos aprendido a utilizar múltiples dispositivos dejando casi de lado cualquier producto no digital. El avance de la tecnología ha sido muy importante.

A lo largo de mis estudios para la obtención del *Grado en Ingeniería y Servicios de las Telecomunicaciones* he realizado múltiples proyectos basados en la programación, tales como aplicaciones Android, aplicaciones con ensamblador, programación de componentes electrónico; y considero que he adquirido la capacidad de desarrollar nuevas aplicaciones basadas en otros entornos y lenguajes.

En la actualidad hay motores tales como Unity3D que cada vez están siendo más utilizados, por ello se apuesta por esta nueva plataforma, no vista durante el grado, que cambia de lenguaje y la forma de interacción con el entorno; y que supone para mí un nuevo reto para comprobar y demostrar las capacidades adquiridas.

Otra de las grandes motivaciones para la elección de este TFG ha sido la posibilidad de ayudar y facilitar la vida a personas con Síndrome de Down y otras discapacidades, como autismo y Asperger. La aplicación se ha realizado con el asesoramiento de Asindown Valencia, en cuanto a contenidos, nivel de dificultad, etc., para que los usuarios de dicha *app* puedan utilizarla y repasar los aspectos que más les interese según la etapa en la que se encuentren.

1.3 Justificación y Objetivos del Trabajo

Las justificaciones del presente trabajo son dos, una de ellas es de carácter académico y es la obtención del título de *Grado en Ingeniería y Servicios de las Telecomunicaciones;* y la otra de carácter técnico y social. Justificación técnica porque se trata de desarrollar una aplicación cómoda para el usuario, ya que solo con el dispositivo móvil o Tablet podrá realizar sus ejercicios diarios sin necesidad de cargar con libros, algo muy pesado para ellos; y social porque fundamentalmente va dirigida a un sector olvidado, y pretende que logren actualizarse en las nuevas tecnologías y puedan sacarle el provecho para aprender conceptos de manera visual e interactiva, y que se diviertan a la vez que aprenden.

Así que el objetivo central del trabajo es desarrollar una aplicación para móviles o Tablet enfocada a las personas con discapacidad, tales como Síndrome de Down, autismo… para que puedan repasar los conceptos que aprenden de manera más amena.

A continuación, se presentan los objetivos específicos que deberá cumplir la aplicación:

- Disponer de diferentes áreas, con niveles de dificultad según la etapa y madurez en la que se encuentre el usuario. Cada etapa deberá incluir una variedad de ejercicios.
- Crear una base de datos que permita tener un ranking con puntuaciones de los usuarios, así incentivar su competitividad.
- Realizar una aplicación que resulte sencilla, dirigida a todo el que lo desee, y utilizada tanto por adolescentes como personas mayores.
- Realizar una aplicación de aprendizaje visual y amena.
- Los ejercicios a realizar no deben ser sencillos, tienen que presentarles un reto a estas personas.
- Tiene que tener un punto competitivo, así poder incentivar el querer seguir jugando y a la vez aprendiendo.
- Investigar sobre aplicaciones similares.
- Subir la *app* a las diferentes *stores*.

1.4 Requisitos Software y Hardware

Este trabajo está ligado a la empresa en la que trabaja la autora, donde se ha formado en el entorno de Unity3D y mediante la que se ha puesto en contacto con la asociación Asindown.

Es por ello que se utilizará el programa Unity3D, que es un poderoso motor con un conjunto completo de herramientas intuitivas y flujos de trabajo rápido para crear contenido interactivo compatible con casi cualquier plataforma de las más conocidas.

Para desarrollar la aplicación se necesita un ordenador que soporte Unity3D, en este caso se ha utilizado un Toshiba con procesador i7.

Otro elemento importante será la disponibilidad de un dispositivo Android (*Huawei* P10Lite) para el testeo de la aplicación antes de subirla a la *play store*.

También será necesario un ordenador MAC para poder instalar el *Xcode* para compilar la aplicación y comprobar que todo funciona correctamente para subirla a la *Apple Store*.

Capítulo 2. Metodología de trabajo del TFG

2.1 Fases del desarrollo

Para la realización del presente trabajo se han llevado a cabo varias fases de desarrollo, distribuidas a continuación en orden cronológico ascendente.

2.1.1 Fase 1: Sistema Operativo

La aplicación se va a desarrollar para dispositivos móviles como Smartphone y tablets, así que va a ser necesario un estudio de los diferentes sistemas operativos existentes y de su presencia en el mercado. Los principales en cuanto a volumen de mercado son:

- **Android**: Es el sistema operativo para móviles más extendido; aunque si por algo se caracteriza es por ser de código abierto basado en Linux. El lenguaje de Android es Java y la extensión de las aplicaciones Android se denomina *Package* (*.apk*).
  Con respecto a la obtención de ingresos cuando se desarrolla una aplicación en esta plataforma son los anuncios. Otro factor a destacar es el perfil de usuario que desarrolla en Android (gente que no se dedica profesionalmente sino como hobby y profesionales).
  Google Play es la tienda de aplicaciones para Android, y para conseguir publicar aplicaciones en ella es necesario hacer un único pago para siempre; pero no todo son ventajas, al poder publicar cualquier tipo de aplicación sin pagar, hace que se cuelen aplicaciones ofensivas o perjudiciales. Aunque Android dispone de varios controles, siempre hay alguna forma de ignorarlos y conseguir introducirse en otros dispositivos. También cabe decir que el sistema operativo Android consume mucha batería debido a su capacidad multitarea.
  Pese a ello, sigue siendo el más utilizado en la actualidad, y es un gran mercado para los iniciados en el desarrollo de aplicaciones.

- **iOS**: No es de código abierto. En referencia a la obtención de ingresos la vía principal es mediante contratos de desarrollo, es decir, empresas que pagan al desarrollador para que realice una aplicación, aunque también se generan ingresos por publicidad. Quizá es por esto que el perfil más extendido en iOS es el de las empresas y emprendedores de las tecnologías de la información.
  Apple Store es la tienda de aplicaciones de iOS y requiere de un pago anual por desarrollador.

- **Windows Phone**: Como factor diferenciador más importante se encuentra la opción de seleccionar entorno de desarrollo. Su vía de ingresos más importante son los anuncios. Su perfil de desarrollador es gente que lo hace por diversión.
  Para poder vender la aplicación por Windows Phone es necesaria una licencia anual.

Después del estudio de los diferentes sistemas operativos que hay en el mercado, se ha llegado a la conclusión que la aplicación se va a desarrollar para *Android* e *iOS* ya que son los sistemas operativos más utilizados.

2.1.2 Fase 2: Entorno de desarrollo

Se va a utilizar Unity3D para el desarrollo de la aplicación, ya que es un motor de videojuegos que dispone de una versión totalmente gratuita y es multiplataforma. Además, incorpora una gran cantidad de utilidades y características a la hora de crear y manejar contenidos gráficos.

La mayor de sus ventajas es su gran documentación para distintos lenguajes de programación, hay gran cantidad de tutoriales y en el foro de Unity3D se puede encontrar la solución casi para cualquier duda.

Además, cuenta con la posibilidad de programar con distintos lenguajes de programación a la vez, permitiendo utilizar *C#* o *JavaScript*, lenguajes eficientes para el desarrollo en Unity3D.

2.1.2.1 Historia de Unity3D

Unity Tecnhologies empezó en el año 2004 cuando David Helgason, Nicholas Francis y Joachim Ante, decidieron dar un vuelco a su compañía de desarrollo de videojuegos tras el fracaso de "GoolBall". El juego no tuvo el éxito esperado; pero en su desarrollo habían creado unas herramientas muy potentes que sirvieron como principio para una idea que rondaba la cabeza del equipo. Crear un motor de videojuegos que pequeñas y grandes empresas pudieran utilizar por igual. Un entorno amigable tanto para programadores como artistas y diseñadores, que llegase a diferentes plataformas sin obligar a programar el juego en estos lenguajes.

El motor llegaría solo para sistema operativo Mac en principio y se presentaría en dos versiones, *Indie* y Profesional. La primera representaba un punto de acceso económico para los pequeños estudios que estaban empezando y no podrían permitirse un gasto elevado, tenía muchas funciones y su precio empezaba en 300 dólares. La versión Pro, de 1500 dólares de coste, traía todas las funciones del motor. Cabe destacar que incluso en la versión *Indie* estaba permitido vender el producto resultante. El motor funcionaba bien; pero estaba lejos de ser una alternativa comparable con los grandes.

En 2008 llegaría el gran salto al aprovechar el lanzamiento del *iPhone*. Un poco más tarde llegaría la versión para Android. El auge de Unity3D era imparable. En 2009 la versión *Indie* desapareció dejando paso a la versión gratuita, accesible para todo aquel que quisiera iniciarse en la plataforma. Incluso con esta versión gratuita se permitía distribuir el juego.

Los desarrolladores independientes no tardaron en convertir Unity3D en uno de los motores gráficos más usados. Pero todavía faltaría dar un salto de calidad para conquistar a los estudios de desarrollo que exigían más calidad gráfica. Este salto llegaría con la versión 4.0, la

cual llega acuerdos con Sony, Microsoft y Nintendo para que el motor sea compatible con sus sistemas.

La versión actual es compatible con muchísimas plataformas (PC, Mac, Linux, iOS, Android, PlayStation, Xbox, Wii, Wii U, Web…), pero siempre hay que retocar algunas cosas al cambiar de plataforma de desarrollo.

2.1.3 Fase 3: Introducción a la programación en C#

Puesto que se desarrolla una aplicación en el entorno de Unity3D se comienza por investigar los requerimientos necesarios para llevarla a cabo. El lenguaje de programación que se utiliza es el *C#* y, puesto que no se dispone de conocimiento alguno acerca de éste, hay que estudiar cuáles son las características del mismo, así como su sintaxis, etc. Para aprender a programar en *C#* hay que documentarse mediante Video Tutoriales en *Cursopedia*, algunas páginas web y el foro de Unity3D.

C# es un lenguaje de programación orientado a objetos desarrollado por Microsoft desde el año 2000. Su sintaxis básica deriva del C/C++ y utiliza un modelo de objetos similar al del Java. Algunas de sus características principales son:

- Lenguaje simple, moderno y de carácter general.
- Fácil de migración del programador familiarizado con C, C++ y Java.
- Adecuación para construir aplicaciones de cualquier tamaño.

```
using UnityEngine;
using System.Collections;

public class EjemploMoverObjeto : MonoBehaviour {

    public GameObject objetoMover;

    // Use this for initialization
    void Start () {

        objetoMover.transform.position = new Vector2 (-10, 10);

    }

}
```

Imagen 1. Función que permite mover un objeto en Unity3D desde un Script C#.

2.1.4 Fase 4: Despliegue del entorno de desarrollo

Como se ha dicho anteriormente, es necesario utilizar el entorno de Unity3D, que se instala para aprender su funcionamiento y familiarizarse con las herramientas de desarrollo.

Es necesario acceder a la página de Unity3D, ya que para descargarlo se necesita previamente una cuenta de usuario.

2.1.4.1 Introducción a Unity3D.

En este apartado se va a introducir el motor gráfico Unity3D, sus características y modo de uso. El aprendizaje sobre Unity3D ha sido autodidacta por medio de tutoriales, la comunidad de Unity3D, mediante la realización de varias aplicaciones y con ayuda de los compañeros de trabajo.

Los proyectos en Unity3D se basan en la utilización de *Assets* (paquetes de recursos), los cuales contienen materiales, efectos, *scripts, sprite, etc.* Unity3D dispone de la *Asset Store*, una tienda donde pueden adquirirse recursos ya creados por otros desarrolladores, gratuitos o de pago, que permiten reducir el tiempo de desarrollo.

Para los iniciados en este motor gráfico, Unity3D facilita ayuda mediante una extensa comunidad en la que se puede encontrar casi cualquier tipo de problema resuelto, una gran ayuda a la hora de empezar a utilizar un software desconocido y tan potente.

A continuación, se va a entrar en detalle para la explicación de las distintas partes que contiene el entorno.

- **Interfaz de usuario de Unity3D**: La interfaz es intuitiva y sencilla, no es difícil familiarizarse con ella en un corto periodo de tiempo. Se divide en varias secciones.
    1. ***Project***: en esta sección se muestra la estructura del proyecto, en ella se muestran los *Assets* que incluyen los recursos a usar en las escenas de la aplicación. En esta ventana se pueden tener multitud de recursos, pero esto no implica la utilización de todos ellos. Por ello, hay que mantener una buena estructura para facilitar el trabajo al usuario. La organización de esta vista se puede hacer mediante la creación de una jerarquía de carpetas. Para ello, existe el desplegable "*create*" que contiene las opciones que permiten una buena organización.

    2. ***Hierarchy***: objetos de la escena actual. La vista de jerarquía contiene todos los elementos que van a ser parte de la escena. Se pueden manipular todos los objetos de la escena a partir de esta vista. Cada vez que se introduce un elemento en la escena se añade una entrada para este elemento en la vista. Al seleccionar un objeto en esta lista, también será seleccionado en la escena y en el inspector. Lo cual, permite y facilita el movimiento, escalado, la rotación y el borrado del objeto o la edición de sus parámetros. Todo ello sin necesidad de tocar ni una línea de código.

    3. ***Inspector***: una de las partes más importantes de Unity3D. Aquí se muestran las características del objeto seleccionado. De esta manera al seleccionar un objeto en la escena, se pueden modificar sus parámetros en el inspector (posición, rotación, translación, etc). El inspector sirve también como panel de herramientas para algunos elementos. En la vista inspector se pueden ver los elementos que definen el comportamiento de los objetos, estos elementos se llaman componentes. Además, aquí mismo se puede proceder a la activación o desactivación de los objetos

y componentes asociados. Los componentes pueden ser *scripts*, animaciones, *colliders*, etc. Por último, aclarar que el inspector cambia según el elemento seleccionado.

4. ***Scene*:** Las aplicaciones en Unity3D se dividen en escenas y para la creación de estas se dispone de un entorno tanto 3D, como 2D. Esta vista permite la visualización del aspecto gráfico de la aplicación en todo el momento. Permite crear escenas de forma sencilla, se pueden arrastrar objetos desde la vista del proyecto y manejarlos (posicionar, escalar, rotar…). También se pueden editar los terrenos y moldearlos. En la vista superior derecha se encuentra el *Gizmo* que se puede observar a continuación. Permite ver la escena desde diferentes puntos de vista. El modo por defecto es en perspectiva 3D y se obtiene haciendo clic sobre la caja central gris.

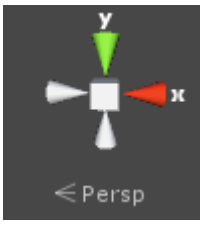


Imagen 2. Modos de visualización

- El cono verde "y" permite pasar al modo de vista aérea de la escena.
- El cono rojo "X" permite pasar al modo *Side*.
- Los conos grises llevan a los siguientes modos: *Back, Left, Bottom*.

5. ***Game*:** sección que muestra cómo se verá el juego al compilarse, hay que tener en cuenta la resolución de pantalla elegida. Se puede ver en esta vista el juego en movimiento para comprobar el buen funcionamiento. En esta ventana encontramos 4 botones que son:

   - El botón "*Maximize on Play*" que permite maximizar a pantalla completa la vista del juego cuando se da al *Play*.
   - El botón "*Gizmo*".
   - El botón "*Stats*".
   - Desplegable "*Free Aspect*", donde se podrá elegir la resolución de pantalla.

6. ***Console*:** muestra los mensajes de error y los que se desean mostrar por pantalla mediante *Debug.Log(" ");*

7. ***Game Controller*:** En Unity3D se puede reproducir el juego sin salir del editor.

   ***Play*:** ejecuta o detiene el juego.
   ***Pause*:** pausa el juego, permitiendo observar la escena sin necesidad de detener el juego por completo.

8. ***Animation*:** permite realizar animaciones para el objeto que se desee.

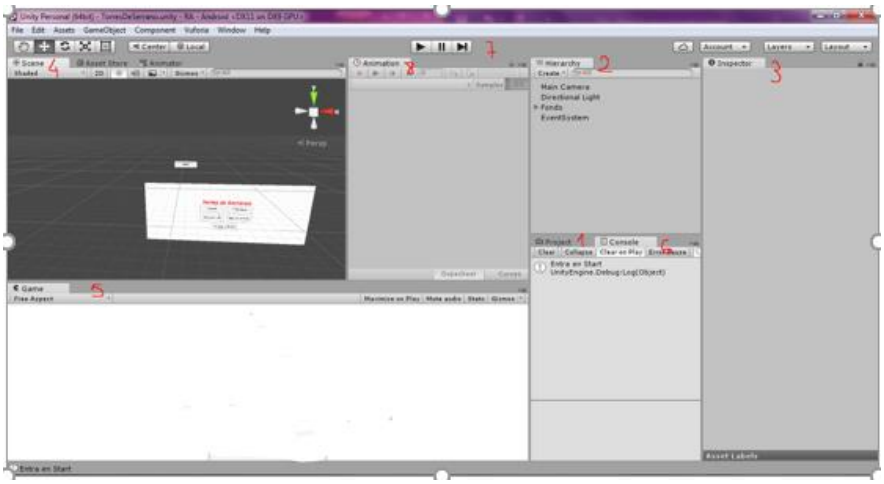


Imagen 3. Interfaz Unity3D

- Menú de la aplicación: A continuación, se muestra el menú de opciones de Unity3D que se encuentra en la parte superior izquierda.

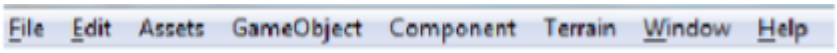


Imagen 4. Menú de la aplicación.

- Botones de control: Debajo del menú vienen definidos los botones de control tal y como se ve en la imagen que sigue:

Imagen 5. Botones de manipulación.

 Este botón permite moverse alrededor de la visa de escena:

- *ALT* permite rotar.
- *COMMAND/CTRL* permite hacer zoom.

- *SHIFT* incrementa la velocidad de movimiento mientras se usa la herramienta.

 Permite mover los objetos seleccionados en la escena, el movimiento se puede realizar en los ejes X, Y o Z.

 Permite rotar los objetos seleccionados en la escena, la rotación se puede realizar en los ejes X, Y o Z.

 Permite escalar los objetos seleccionados en la escena. En este caso también se puede escalar en cualquiera de los ejes X, Y o Z.

Al tratarse de una aplicación 2D, no van a utilizarse materiales ni físicas en 3D. Para materiales se utilizan los denominados *Sprites*.

A la hora de colocar los distintos objetos del menú de la aplicación se deberá utilizar esta característica, que tendrán todos los objetos en común.

Imagen 6. *Transform* Unity3D

Transform indica la posición, rotación y tamaño del objeto en las 3 dimensiones (X,Y,Z), al trabajar solo en 2 dimensiones, solo se hará caso a X e Y.

A estos *GameObjects* utilizados se les puede añadir todo tipo de Componentes (efectos, audios, físicas, *render*, *scripts*, *UI*, etc.) y es con lo que se compone el menú de la aplicación.

Unity3D facilita las cosas permitiendo crear *Prefabs*, que son objetos que predefines con unos componentes y puedes añadirlos las veces que quieras con solo arrastrarlos hacia la *Scene*.

Componentes necesarios para conocer:

- ***Sprite** Renderer*: convierte un objeto en un *sprite* 2D.
- ***Animator***: permite a un objeto tener distintos estados de animación.
- ***Physics 2D**:* permite a un objeto colisionar con otros.
- ***RigidBody**:* permite a un sprite interactuar con las físicas del juego.
- ***Camera***: convierte un objeto en una cámara.

- ***Particle System*:** convierte un objeto en generador de partículas.
- ***Script*:** permite añadir funcionalidades y comportamientos a los objetos.

Unity3D se compone de *GameObjects* que tienen distintos componentes, pero ¿cómo hacer que un botón pueda cambiar de escena o poder escoger opciones? De eso se encargan los *scripts*, donde se programan los comportamientos de los objetos, y el funcionamiento de la aplicación.

En cada *script* deberá programarse el código que realice la función requerida. Los *scripts* deben heredar de la clase *Monobehavior*, que es la clase principal de Unity3D, disponiendo de distintas funciones que siguen un ciclo de vida como:

- ***Awake* ():** inicializa el script
- ***Start* ():** se ejecuta una sola vez al activarse el script
- ***Update* ():** se ejecuta una vez casa *frame*.

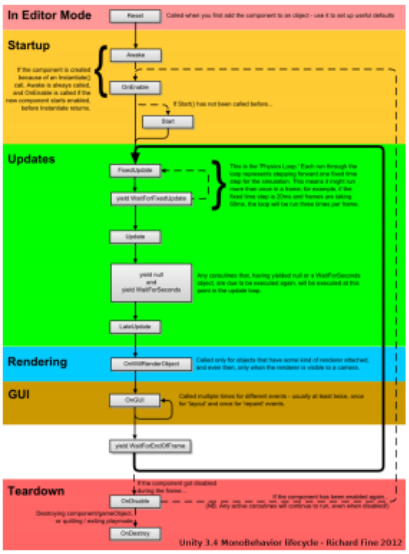


Imagen 7. Ciclo de vida de *Monobehavior*

Unity3D dispone de funciones tipo *Couroutine* que permiten pausar la ejecución del código y continuar con el siguiente *frame*.

La instrucción de retorno *yield return* es la encargada de pausar la ejecución y reanudarla en el siguiente *frame*. Para iniciar una *Couroutine* se debe llamar a la función *StartCouroutine (nombredelafuncion).* Este tipo de funciones serán utilizadas cuando no se necesite ejecutar en cada *frame* como haría el método *Update* (), de esta forma se reduce notablemente el número de comprobaciones.

La arquitectura de Unity3D se basa en componentes, permitiéndola ser modular y extensible. Esto quiere decir, que es a través de los Componentes de los *GameObjects* que se encuentran en las Escenas, como se comunica Unity3D con la lógica del juego controlada por *Monobehavior*. El flujo de ejecución de los *scripts* está controlado por el motor gráfico, el cual está controlado por el *Runtime* de *Monobehavior*. Unity3D compila los *assets* y sus componentes a través del *SDK* de Android.

Imagen 8. Arquitectura de Unity3D

La versión 4.6 de Unity3D introduce un nuevo sistema de Interfaz de Usuario que permite, de forma rápida y sencilla crear los menús y componentes de la UI, adaptándose a los distintos dispositivos en los que puede ser ejecutado.

2.1.5 Fase 5: Documentación para programación de aplicaciones Unity3D.

Una vez desplegado el entorno de desarrollo, se procede a la documentación para conocer el funcionamiento de las herramientas que dicho motor proporciona mediante tutoriales de *Cursopedia* y del foro de Unity3D. Paralelamente se van desarrollando diversas aplicaciones sencillas que se proponen en estos tutoriales para así adaptarse al entorno.

Lo primero que se debe saber es la creación de un nuevo proyecto, para ello hay que ir hasta la barra superior de Unity3D y hacer clic en *File* → *New Project* para que se abra la ventana del dialogo "*Create new Project*". Hay que seleccionar la carpeta en la que se quiere guardar el proyecto haciendo clic en "*Browse*".

Al crear el nuevo proyecto, se reinicia Unity3D creando la estructura del proyecto en la carpeta indicada. Se abre la pantalla de inicio de Unity3D en la que aparecen los *assets* y un *main camera* en la vista de escena y jerarquía.

2.1.6 Fase 6: Documentación de Síndrome de Down.

Previamente es necesario la búsqueda de información sobre esta discapacidad, para poder tener una idea de aplicación.

El síndrome de Down (SD) es un trastorno genético causado por la presencia de una copia extra del cromosoma 21, en vez de los dos habituales, por ello se denomina también trisomía del par 21. Se caracteriza por la presencia de un grado variable de discapacidad cognitiva y unos rasgos físicos peculiares que le dan un aspecto reconocible.

El SD es la principal causa de discapacidad intelectual y la alteración genética más común. Entre las discapacidades intelectuales están:

- Lentitud del desarrollo motor y cognitivo. La variabilidad es enorme, y aunque hay lentitud, si hay progresos.
- Su afectación cerebral produce lentitud para procesar y codificar la información, así como dificultades para interpretarla, elaborarla y la toma de decisiones. Pueden tener dificultades para retener información tanto a corto como a largo plazo.
- Suelen tener dificultades para centrar la atención durante tiempos prolongados, necesitando en ocasiones de entrenamiento específico para conseguir mejoras.

Una vez pensada la idea inicial se acude a la asociación 'Asindown Valencia' para el asesoramiento de actividades a realizar para las personas con estas discapacidades.

La Asociación Síndrome de Down de Valencia, nace para fomentar toda clase de medios de carácter pedagógico y científico que faciliten la educación, integración familiar, laboral y social de las personas con síndrome de Down. Fue creada por iniciativa de un grupo de padres en 1989 y declarada de utilidad pública en 2001, llevando a cabo un intenso trabajo de formación, de abrir nuevas perspectivas de futuro y de permanente presencia en la sociedad para facilitar la mejora de la calidad de vida de las personas con Síndrome de Down.

En 1994, la Asociación Síndrome de Down de Valencia funda la fundación Asindown, una entidad sin ánimo de lucro, cuyo objetivo es mejorar la calidad de vida y favorecer la inclusión y participación como ciudadanos de pleno derecho en todas las etapas de su vida, de las personas con Síndrome de Down u otras discapacidades intelectuales y sus familias, mediante la promoción de recursos y apoyos para la investigación y el desarrollo personal, social, educativo y laboral.

Asindown cuenta con multitud de proyectos, tales como:

- Servicio de atención social a las familias: las familias son el pilar fundamental para impulsar las capacidades de las personas con síndrome de Down desde que nacen.
- Servicio de atención psicológica: mejora la calidad de vida de las personas con SD y otras discapacidades intelectuales y su familia.
- Programa de atención temprana: atención a niños de 0 a 4 años que presentan trastornos en su desarrollo o riesgos de padecerlo con el objetivo de potenciar su capacidad de desarrollo y bienestar.
- Intervención socioeducativa: programa de apoyo para que las personas con SD sigan un itinerario educativo inclusivo.
- Iniciación al ocio: proporciona al alumnado oportunidades de aprendizaje en entornos reales.
- Formación para el empleo: es fundamental ofrecer una formación profesional de calidad, adaptada a las necesidades del alumnado.

- Formación básica de personas adultas: Las personas con síndrome de Down, precisan que se les ofrezca apoyo, para que su derecho a la educación se vea cumplido con igualdad y que sus necesidades, no sean un obstáculo para conseguirlo.
- Programa de empleo con apoyo: basado en la búsqueda, adaptación y seguimiento en el puesto de trabajo.

Una vez hablado con Asindown y vistas los programas que tienen, se llega a un acuerdo donde los psicólogos y profesores de la etapa educativa, formación adultos y empleo facilitarán la información necesaria para poder realizar las actividades que sirvan de ayuda a las personas con SD.

2.1.7 Fase 7: Boceto de las actividades a realizar

Tras el asesoramiento de Asindown se procede a realizar bocetos de todas las actividades que se pueden realizar a un nivel básico (sin arte), para que las personas asesoradoras pudiesen ver el tipo de actividades que se han pensado para las distintas etapas. Se va compartiendo un documento vía *Google Drive* con Asindown para ver las actividades que se pueden realizar.

Para la etapa educativa se pueden realizar ejercicios y minijuegos de dificultad media con el objetivo de potenciar el aprendizaje de la lectura, el habla y cálculo matemático básico. Las propuestas de esta etapa son las siguientes:

- Relación de actividades y juegos de lectura global.
- Asociación número y cantidad.
- Batería de vocabulario por categorías con imágenes.
- Problemas de matemáticas de primer ciclo.
- Ejercicios para trabajar el euro.

Para la etapa de formación de adultos los ejercicios y minijuegos deben estar destinados a la formación laboral/profesional del usuario y su comportamiento social, tales como:

- Nombres de los objetos
- Ordenas las fechas
- Resolución de problemas (pedir ayuda, llegar tarde, pedir día libre…)
- Señales de riesgos laborales

La etapa de empleo debe tener ejercicios destinados al refuerzo laboral, como son:

- Tareas administrativas de ordenar alfabéticamente, archivar documentos según nombre…
- Tareas de hostelería como montar correctamente una mesa con sus utensilios.
- Tareas de reponedor de cómo colocar los productos en el sitio, leer las etiquetas, reconocer productos caducados…

2.1.8 Fase 8: Acuerdo de las actividades

Después de realizar varias actividades, se concretan las actividades que mejor convienen para cada etapa ya con sus respectivos textos proporcionados por la asociación.

Se llega al consenso que las mejores actividades a realizar para la aplicación y para cada etapa son las siguientes:

Etapa educativa: Asociación número y cantidad, ejercicios para trabajar el euro y batería de vocabulario de animales.

Etapa Formación de adultos: Relacionar nombres con los objetos, ordenar las fechas y señales de riesgos laborales.

Etapa empleo: Clasifica los productos, archivar los documentos según el apellido, montar correctamente la mesa con sus utensilios.

Todos los ejercicios anteriores tendrán un incentivo competitivo contando con un ranking online donde los usuarios podrán registrar sus puntuaciones y entrar en la clasificación. En dicha clasificación se tendrá en cuenta el nivel que se ha escogido previamente y la actividad que se ha realizado.

2.1.9 Fase 9: Desarrollo de la aplicación final

Adquiridos todos los conocimientos e información necesarios, se comienza la implementación de la aplicación junto con el diseño de ésta. Durante la fase de implementación se realizan varias consultas a la asociación para concretar algunos temas de las actividades que no habían quedado del todo claros.

A continuación, se procede a la explicación de la aplicación y de las pantallas de las que consta.

2.1.9.1 Pantalla Principal

Imagen 9. Pantalla Principal

Esta es la primera pantalla que se muestra al abrir la aplicación. Como se observa, muestra el logo de Asindown, fundación con la que se ha trabajado, y un botón de empezar.

Es la pantalla principal, la que nos va a permitir entrar a las siguientes pantallas, tan solo apretando en el botón empezar.

2.1.9.2 Pantalla elección de etapa

Imagen 10. Pantalla elección de etapa

Esta pantalla es la que va a permitir escoger la etapa en la que se encuentra el usuario o aquella de la que desea realizar las actividades. Se cuenta con las tres etapas dichas anteriormente, solo hay que apretar el botón que se desee. Los botones se han hecho de distintos colores para poder permitir a los usuarios de la app reconocerlos mejor.

2.1.9.3 Pantalla Etapa Educativa

Imagen 11. Pantalla etapa educativa

En esta escena se pueden apreciar las tres actividades de las que consta la etapa educativa.

Si el usuario se equivoca al entrar en la etapa tiene un botón de atrás para volver a la pantalla anterior y escoger la que más se adapte a sus preferencias.

2.1.9.4 Pantalla Elegir dificultad

Imagen 12. Pantalla elegir dificultad

Al principio de cada actividad el usuario tiene la posibilidad de escoger el nivel en el que se desea realizar la actividad. Dependiendo de esta elección el usuario va a tener más o menos tiempo para realizar las actividades. Además, en la actividad de ordenar las fechas si se escoge el nivel fácil se tendrán 3 fechas, el medio 5 y el difícil 6, aumentado así la dificultad de este ejercicio.

2.1.9.5 Pantalla asociación número y cantidad

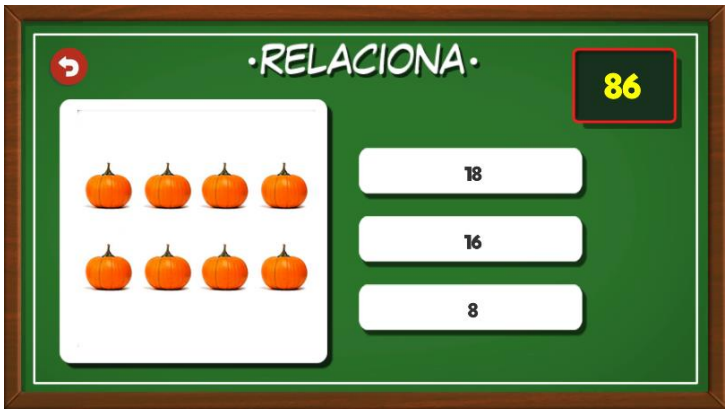


Imagen 13. Pantalla asociación número y cantidad

En esta actividad el jugador tiene que relacionar la cantidad de objetos que van saliendo en pantalla con el número correcto, por ejemplo, en esta pantalla hay 8 calabazas por lo que se debería pulsar el botón que muestra el 8. Una vez pulsado este botón se pondría en color verde mostrando que el usuario ha acertado la respuesta y sumaría a la puntuación. En cambio, si el usuario falla el texto del botón se vuelve rojo y restaría a la puntuación. Además de cambiar el color de la respuesta, también se instancian unos emoticonos feliz y triste para que el usuario pueda ver mejor si ha acertado o no.

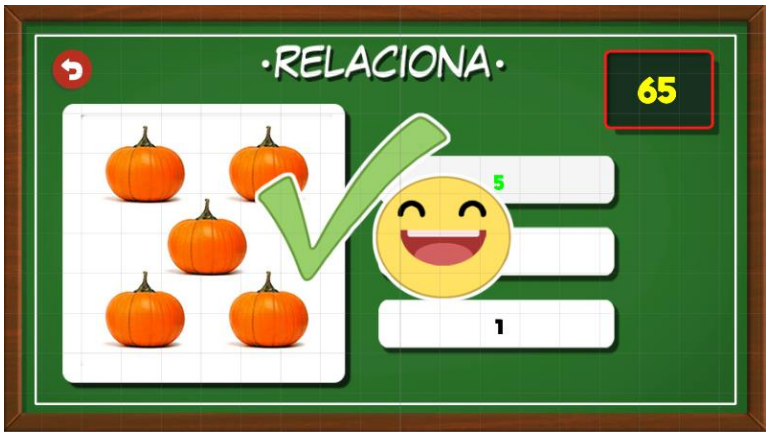


Imagen 14. Ejemplo acierto

Como se puede observar hay un temporizador que va mostrando el tiempo que queda para que la actividad finalice.

Para la realización de esta actividad son necesarios dos *arrays* de *strings*, uno para los nombres de las imágenes con los objetos y otro con los números, en este caso números del 0 al 20.

De manera *Random* (con la función *Random.Range(0,20));* se escogerá el número del *array* de *strings* de los nombres de las imágenes, una vez se sabe esto, se carga el *Sprite* con este nombre anterior, mostrando la imagen.

Código:

```
i = Random.Range (0, 20);

ImagenMostrada.sprite = Resources.Load<Sprite> (ListaImagenes[i]);
```

Imagen 15. Código cargar *sprite*

Hay que saber que ImagenMostrada es el *Image* dentro de la escena, previamente definido en el *Awake().*

Una vez cargada la imagen mostrada, es el turno de mostrar el texto de los botones utilizando el siguiente código:

```
int a = Random.Range (0, 20);

while (ImagenMostrada.sprite.name == TextosBotones [a]) {
    a = Random.Range (0, 20);
}

Boton [0].text = TextosBotones [a];

int b = Random.Range (0, 20);
while (ImagenMostrada.sprite.name == TextosBotones [b] || TextosBotones[b] == TextosBotones[a]) {
    b = Random.Range (0, 20);
}

Boton [1].text = TextosBotones [b];

int c = Random.Range (0, 20);
while (ImagenMostrada.sprite.name == TextosBotones [c] || TextosBotones [c] == TextosBotones[a] || TextosBotones [c] == TextosBotones[b]) {
    c = Random.Range (0, 20);
}

Boton [2].text = TextosBotones [c];

int z = Random.Range (0, 3);

Boton [z].text = ImagenMostrada.sprite.name;
```

Imagen 16. Código texto botones

De manera *random* se escoge el texto que se va a poner en cada uno de los botones, pero sin que coincidan entre ellos, para eso se hace uso de los *while*, para que mientras salga el mismo texto que en los botones anteriores siga generando de manera *random* otra opción. Una vez generado el texto de los tres botones hay que asegurar que la opción correcta está entre una de las tres posibilidades, para ello, de manera *random* se escoge uno de los botones y se le asigna el nombre de la imagen mostrada, que previamente han sido llamadas igual que los textos que hay que sacar por pantalla. Así para comprobar si el usuario ha acertado o fallado solo hay que comprobar si el nombre de la imagen coincide con el texto del botón de la manera siguiente:

```
if (ImagenMostrada.sprite.name == variable) {

    Debug.Log ("Respuesta Correcta");
    Aciertos++;
    Boton [0].color = Color.green;
} else {
    Debug.Log ("Respuesta Incorrecta");
    Boton [0].color = Color.red;
    Fallos++;
    Boton [0].color = Color.red;
}

StartCoroutine (next ());
```

Imagen 17. Código comprobación

Tanto si se acierta como si se falla se llamará a la corrutina *next* para que se vuelva a generar la imagen y el texto así poder seguir practicando.

2.1.9.6 Pantalla practica con los euros

Imagen 18. Pantalla practica con los euros

En esta actividad el usuario tiene que pulsar sobre las monedas y billetes que sean necesarios para poder pagar el objeto. Si el usuario se equivoca se le da la oportunidad de volver a pulsar sobre la moneda equivocada para poder restar este valor a la suma total. Los billetes y monedas tendrán un reborde amarillo si están pulsadas, para que así el usuario pueda apreciarlo mejor.

El procedimiento de esta actividad es parecido a la anterior, pero en este caso habrá un *float* que irá sumando la cantidad que el usuario va añadiendo pulsando sobre los botones con las monedas y billetes, una vez la cantidad sea correcta saldrá una imagen mostrando que está todo correcto y cambiará la imagen y los botones para poder seguir jugando. Para poder saber si hay que sumar o restar según se apriete el botón se hace uso de *booleanos*, cuando se apreté el botón el valor de dicho *booleano* cambia a true por lo que suma a la cuenta total, por el contrario, si se vuelve a apretar dicho botón el booleano cambia a false por lo que se restará dicha cifra a la cuenta total.

2.1.9.7 Pantalla Relaciona animales con su nombre

Imagen 19. Pantalla relaciona animales

Esta actividad es muy parecida a la actividad de asociación de cantidad con los números, pero esta vez hay que relacionar animales con sus nombres. Seguirá las mismas mecánicas anteriores.

2.1.9.8 Pantalla Etapa formación de adultos

Imagen 20. Pantalla formación adultos

En esta escena se pueden apreciar las tres actividades de las que consta la etapa de formación de adultos.

Si el usuario se equivoca al entrar en la etapa tiene un botón de atrás para volver a la pantalla anterior y escoger la que más se adapte a sus preferencias.

2.1.9.9 Pantalla relacionar los objetos con sus nombres

Imagen 21. Pantalla relaciona objetos

En este ejercicio habrá que relacionar los nombres de una batería de objetos pertenecientes a el ámbito administrativo para que así repasen aquellos objetos que serán necesarios a la hora de trabajar.

Las mecánicas de este ejercicio serán iguales que las otras dos actividades de relacionar.

2.1.9.10 Pantalla Ordenar las fechas

Imagen 22. Pantalla ordenar fechas fáciles

Imagen 23. Pantalla ordenar fechas medio

Imagen 24. Pantalla ordenar fechas difíciles

Esta actividad consta de ordenar fechas de menor a mayor. Aparecerán en pantalla diferentes fechas y el usuario deberá ordenarlas cronológicamente. Las fechas se muestran numéricamente.

Como se puede observar según el nivel escogido saldrán más o menos fechas a ordenar. El tiempo para empezar es el mismo y según se va pasando de ronda el tiempo va disminuyendo de 10 en 10 hasta llegar el punto que solo se tienen 10 segundos para ordenar las fechas.

Según se van acertando las fechas estas se van colocando en los huecos que hay abajo para mostrar el orden para que los usuarios no se equivoquen. Si se equivocan la fecha se pondrá en rojo hasta que sea el turno de esta.

Para realizar esta actividad ha sido necesario seguir el siguiente proceso.

1. Se crean unos *arrays* para poder almacenar las posibles fechas que se van a mostrar.

2. Se crea una función para comprobar que no se repite ninguna de las fechas que se muestran.

```
int GetRandomNumberWithoutRepeat(){

    int ran = Random.Range (0, list.Count);
    int value = list [ran];
    list.RemoveAt (ran);
    return value;
}
```

Imagen 25. Código función no repetir

3. Para cada botón se crea una función para comprobar que la fecha que está escrita en este es la menor posible. Si es correcto el botón desaparece y la fecha se escribe en los huecos dejados abajo. Si es incorrecto el texto del botón se volverá rojo para mostrar que es un error que siga intentándolo. Habrá un contador que en todo momento sabrá en qué posición del *array* se encuentra para así saber qué fecha debe ir.

```
public void b0(){

    string variable;
    variable = Boton[0].GetComponent<Text> ().text;
    Debug.Log (variable);
    if (cont == 0) {

        Debug.Log ("Variable b0: " + variable);

        for (int i = 0; i < 3; i++) {

            Debug.Log (ListaNumeros1 [i]);

            if (variable == ListaNumeros1 [i]) {
                fecha1.text = ListaNumeros1[i];

                auxbool0 = true;
                Boton [0].color =  Color.green;

                StartCoroutine (contador ());
                Boton1.SetActive(false);
                aciertos += 1;


            }

        } if (auxbool0 == false) {

            Boton [0].color = Color.red;

        }
            auxbool0 = false;

    }
```

Imagen 26. Código comprobación

Imagen 27. Error fechas

4. Una vez ordenadas todas las fechas se lanza una *corrutina* que decrementará en 10 segundos el tiempo que se tiene para ordenar las fechas y lanzará otras fechas nuevas a ordenar. La actividad terminará cuando el tiempo llegue a 0.

2.1.9.11 Pantalla señales de riesgos laborales

Imagen 26. Pantalla señales riesgos laborales

En este ejercicio habrá que relacionar las señales de riesgos laborales con sus significados.

Las mecánicas de este ejercicio serán iguales que las otras actividades de relacionar.

2.1.9.12 Pantalla Etapa de empleo

Imagen 27. Pantalla etapa de empleo

En esta escena se pueden apreciar las tres actividades de las que consta la etapa de empleo.

2.1.9.13 Pantalla Clasifica los productos

Imagen 28. Pantalla clasifica los productos

En esta actividad el usuario tendrá que pulsar sobre los productos de la estantería para ver su fecha de caducidad y así saber si pueden estar en la estantería o no. Para ello se estanciará un *prefab* previamente creado, que cambiará de *Sprite* dependiendo la imagen que haya sobre el botón que se ha pulsado.

Imagen 29. Pantalla elección SI o NO

La aplicación siempre sabrá en que día se encuentra el usuario, gracias a una función de Unity3D que permite esto. Con esta función se puede saber el día, el mes, el año e incluso la hora en que se encuentra el usuario.

```
diahoy = System.DateTime.Now.Day;
Debug.Log ("Dia: " + diahoy);

meshoy = System.DateTime.Now.Month;
Debug.Log ("Mes: " + meshoy);

añohoy = System.DateTime.Now.Year;
Debug.Log ("Año: " + añohoy);
```

Imagen 30. Código saber fecha

Una vez se sabe el dato anterior, solo habrá que comparar dicho dato con la fecha de caducidad del producto, que una vez más es generada de manera *random*. Primero se comprobará si el año es mayor que en el que estamos, si es así, sí que se podrá poner en la estantería, en caso contrario se pasaría al mes y podría llegar incluso a compararse el día.

Conforme se vayan aceptando o fallando se irán incrementando los marcadores de acierto y fallo, cuando se haya acabado con una estantería aparecerá otra, así hasta que el tiempo se agote.

2.1.9.14 Pantalla de archivar los documentos

Imagen 31. Pantalla archivar documentos

En esta actividad hay que archivar los ficheros según el apellido, para ello, se cuenta con un *scroll* donde habrá carpetas con las letras de abecedario. Cada letra contiene un *tag* y un *collider* para así poder detectar que el archivo está sobre la carpeta y saber si el *tag* es el mismo o distinto. Si el *tag* es distinto significa que el apellido no coincide con la carpeta seleccionada

por lo que el archivo vuelve al centro de la pantalla, en caso de ser el mismo el archivo cambiará de apellido. Para realizar esto ha sido necesario utilizar la siguiente parte de código.

```
void OnCollisionEnter2D(Collision2D col){


    if (col.collider.tag == GetComponent<Collider2D>().tag) {
        gameObject.transform.localPosition = new Vector2(0,0);
        StartCoroutine (otra ());
        Aciertos++;


        }

    if (col.collider.tag != GetComponent<Collider2D>().tag) {

        gameObject.transform.localPosition = new Vector2(0,0);
        Fallos++;

    }


}
```

Imagen 32. Código *Collider*

2.1.9.15 Pantalla Hostelería

Imagen 33. Pantalla hostelería

Esta actividad trata de arrastrar los objetos, que hay bajo en el *scroll*, sobre las siluetas de encima de la mesa, para así ayudar a recordar donde va cada cosa. Será necesario el uso de *tags* para poder identificar que el objeto que se ha arrastrado es el correcto, en caso contrario el objeto volverá a su posición en el *scroll*.

Imagen 34. Pantalla hostelería aciertos

Si se arrastra el objeto al lugar adecuado, éste desaparecerá y la silueta tomará el color adecuado.

2.1.9.16 Pantalla puntuación

Imagen 35. Pantalla puntuación

Esta pantalla muestra la puntuación obtenida en la actividad realizada, pidiendo el nombre del jugador para así poder mandar dicha puntuación junto con el nombre, la actividad realizada y la dificultad a la base de datos del *ranking*. Una vez escrito el nombre hay que apretar sobre el botón de enviar puntuación.

2.1.9.17 Pantalla Clasificación

Imagen 36. Pantalla clasificación

Esta es la pantalla que muestra el *ranking* de puntuaciones. Se puede observar que muestra el nombre del jugador, la puntuación obtenida, la actividad realizada y la dificultad. Para realizar esta parte ha sido necesario tener conocimientos de base de datos, MySQL y PHP.

Una base de datos es una herramienta para recopilar y organizar información. No solo permite almacenar información sobre casi cualquier cosa, sino que es posible también sistematizarla, modificarla, consultarla, etc. Muchas bases de datos empiezan siendo una lista en un programa de procesamiento de texto o en una hoja de cálculo. A medida que crece la lista, empiezan a aparecer repeticiones e inconsistencias en los datos y cada vez resulta más complicado comprender los datos representados. Es aconsejable transferir la información a una base de datos creada mediante un sistema de administración de base de datos. Se denominan así porque utilizan tablas de datos relacionadas por un campo común.

MySQL es un sistema de administración relacional de base de datos. Una base de datos relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo, lo que aporta velocidad y flexibilidad a la hora de introducirlos, editarlos y consultarlos. Las tablas están conectadas por relaciones definidas que hacen posible combinar los datos de diferentes tablas en función de las necesidades del usuario. En este caso solo hará falta una tabla donde estará almacenada toda la información.

Aunque MySQL tiene su propia interfaz de trabajo se prefiere usar una aplicación llamada *phpMyAdmin*, que facilita las tareas de crear las bases de datos, sus tablas, insertar, modificar y

borrar datos. Ésta es una herramienta de software libre escrito en PHP, destinado a facilitar la administración de MySQL a través de la Web.

Se ha utilizado como servidor para el almacenamiento de la base de datos (MySQL) *hostinguer.es*. Para la realización de esta ha sido necesario *phpMyAdmin*, donde se ha creado una tabla, en este caso llamada RankingPuntuaciones y tres filas dentro de esta:

- Fila nombre, de tipo *varchar* con longitud de 30 y cotejamiento *utf8_unicode_ci*.
- Fila puntuación, de tipo *int*.
- Fila actividad, de tipo *varchar* con longitud de 30 y cotejamiento *utf8_unicode_ci*.

Una vez creada la base de datos hay que aprender a interactuar con esta. Para ello, es necesario tener algún conocimiento previo en PHP, lenguaje elegido del lado del servidor para interpretar las partes dinámicas de la aplicación.

Resulta una opción muy interesante ya que se trata de un software libre y fácilmente se puede encontrar una gran cantidad de información, tutoriales y diferentes formas de ayuda a través de internet, lo que facilita el desarrollo y permite captar nuevas ideas gracias a todo el material disponible.

PHP es un lenguaje de programación del lado del servidor que permite el desarrollo de contenido dinámico. Una de las ventajas de este lenguaje es que es procesado en el servidor, que es el que genera el contenido dinámico, de manera que no consume recursos del cliente. Es posible su uso en la mayoría de los servidores web, al igual que en casi todos los sistemas operativos sin costo.

PHP significa *Hypertext Pre-processor* y fue creado por Rasmus Lerdof en 1994, aunque la implementación principal de PHP la lleva a cabo The PHP Group.

Para la realización de la clasificación han sido necesarias dos partes, la parte del servidor y la del cliente.

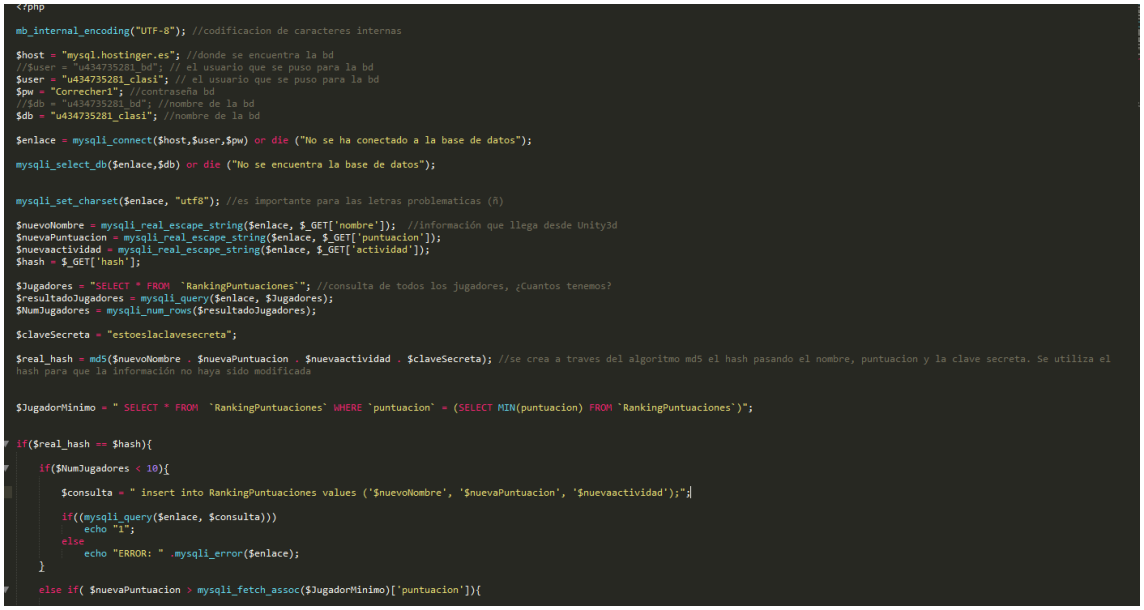
- La parte del servidor ha sido realizada con PHP con el editor de código "*Sublime Text 3*". Han sido necesarios dos *scripts*, el de ver la puntuación y el de ingresar la puntuación, ambos compartirán código entre ellos.

```php
<?php

mb_internal_encoding("UTF-8"); //codificacion de caracteres internas

$host = "mysql.hostinger.es"; //donde se encuentra la bd
//$user = "u434735281_bd"; // el usuario que se puso para la bd
$user = "u434735281_clasi"; // el usuario que se puso para la bd
$pw = "Correcher1"; //contraseña bd
//$db = "u434735281_bd"; //nombre de la bd
$db = "u434735281_clasi"; //nombre de la bd


$enlace = mysqli_connect($host,$user,$pw,$db) or die ("No se ha conectado a la base de datos");

mysqli_select_db($enlace,$db) or die ("No se encuentra la base de datos");


mysqli_set_charset($enlace, "utf8"); //es importante para las letras problematicas (ñ)

$consulta = "SELECT * FROM  `RankingPuntuaciones` ORDER BY  `RankingPuntuaciones`.`puntuacion` DESC LIMIT 10";







//consulto a la bd en la tabla rankingpuntuaciones y saldrá ordenado por la parte de puntuacion de mayor a menor
$resultado = mysqli_query($enlace,$consulta);

$num = mysqli_num_rows($resultado); //saldrá el número de columnas que hay de puntuacion


for($i = 0; $i < $num; $i++){

    $row = mysqli_fetch_array($resultado, MYSQLI_ASSOC);
    echo $row['nombre'] . ";" . $row['puntuacion'] . ";" . $row['actividad'] . ";";
}

?>
```

Imagen 37. Código conectarse a la base de datos

Como se puede observar en los comentarios del *script*, es necesaria una autentificación, para saber a qué base de datos conectarse, quien es el usuario y la contraseña.

Una vez conectados a la base de datos se realiza una consulta a ésta para que muestre en pantalla las 10 mejores puntuaciones obtenidas. Primero se consulta en la tabla Ranking-Puntuaciones y saldrá ordenado por la parte de puntuaciones de mayor a menor, a continuación, con la función *mysqli_num_rows* saldrá el número de columnas que tiene la base de datos, dentro de un bucle *for* se irán mostrando las columnas hasta llegar a 10 o al número de columnas que hay siempre y cuando sea menor que 10.



```php
<?php

mb_internal_encoding("UTF-8"); //codificacion de caracteres internas

$host = "mysql.hostinger.es"; //donde se encuentra la bd
//$user = "u434735281_bd"; // el usuario que se puso para la bd
$user = "u434735281_clasi"; // el usuario que se puso para la bd
$pw = "Correcher1"; //contraseña bd
//$db = "u434735281_bd"; //nombre de la bd
$db = "u434735281_clasi"; //nombre de la bd

$enlace = mysqli_connect($host,$user,$pw) or die ("No se ha conectado a la base de datos");

mysqli_select_db($enlace,$db) or die ("No se encuentra la base de datos");


mysqli_set_charset($enlace, "utf8"); //es importante para las letras problematicas (ñ)

$nuevoNombre = mysqli_real_escape_string($enlace, $_GET['nombre']);  //información que llega desde Unity3d
$nuevaPuntuacion = mysqli_real_escape_string($enlace, $_GET['puntuacion']);
$nuevaactividad = mysqli_real_escape_string($enlace, $_GET['actividad']);
$hash = $_GET['hash'];

$Jugadores = "SELECT * FROM  `RankingPuntuaciones`"; //consulta de todos los jugadores, ¿Cuantos tenemos?
$resultadoJugadores = mysqli_query($enlace, $Jugadores);
$NumJugadores = mysqli_num_rows($resultadoJugadores);

$claveSecreta = "estoeslaclavesecreta";

$real_hash = md5($nuevoNombre . $nuevaPuntuacion . $nuevaactividad . $claveSecreta); //se crea a traves del algoritmo md5 el hash pasando el nombre, puntuacion y la clave secreta. Se utiliza el
hash para que la información no haya sido modificada


$JugadorMinimo = " SELECT * FROM  `RankingPuntuaciones` WHERE `puntuacion` = (SELECT MIN(puntuacion) FROM `RankingPuntuaciones`)";


if($real_hash == $hash){

    if($NumJugadores < 10){

        $consulta = " insert into RankingPuntuaciones values ('$nuevoNombre', '$nuevaPuntuacion', '$nuevaactividad');";

        if((mysqli_query($enlace, $consulta)))
            echo "1";
        else
            echo "ERROR: " .mysqli_error($enlace);
    }

    else if( $nuevaPuntuacion > mysqli_fetch_assoc($JugadorMinimo)['puntuacion']){
```

Imagen 38. Código consulta base de datos I

```php
$JugadorMinimo = " SELECT * FROM  `RankingPuntuaciones` WHERE `puntuacion` = (SELECT MIN(puntuacion) FROM `RankingPuntuaciones`)";


if($real_hash == $hash){

    if($NumJugadores < 10){

        $consulta = " insert into RankingPuntuaciones values ('$nuevoNombre', '$nuevaPuntuacion', '$nuevaactividad');";

        if((mysqli_query($enlace, $consulta)))
            echo "1";
        else
            echo "ERROR: " .mysqli_error($enlace);
    }

    else if( $nuevaPuntuacion > mysqli_fetch_assoc($JugadorMinimo)['puntuacion']){

        $Eliminado = "DELETE FROM RankingPuntuaciones ORDER BY puntuacion ASC LIMIT 1";
        $consulta = "insert into RankingPuntuaciones values ('$nuevoNombre', '$nuevaPuntuacion', '$nuevaactividad');";

        if((mysqli_query($enlace, $consulta)) && (mysqli_query($enlace, $Eliminado)))
            echo "2";
        else
            echo "ERROR : " .mysqli_error($enlace);

    }

    else
        echo "3";

}

?>
```

Imagen 39. Código consulta base de datos II

Para introducir una nueva puntuación se hace uso de una función *hash* para asegurarse que los datos no han sido modificados. Pero antes también será necesaria la autentificación en la base de datos.

Si el *hash* obtenido en el cliente es igual que el del servidor es cuando se introducirá en la base de datos la nueva puntuación.

Estos archivos PHP se tienen que subir a la base de datos creada, para ello se utiliza el programa "*filezilla*", cliente FTP multiplataforma de código abierto y software libre que permite conectarse desde el ordenador a un servidor web. Permite subir o bajar archivos a un servidor Web. La pantalla principal de este programa pide:

- Nombre o IP del servidor donde está alojada la base de datos en este caso la ip es 93.188.160.180.
- Nombre de usuario: u434735281
- Contraseña: **********
- Puerto: 21

Una vez proporcionados estos datos se estará conectado al servidor y se podrán incorporar los scripts anteriormente descritos.

- La parte del cliente, en este caso Unity3d, también van a ser necesarios dos *scripts*, uno para ver las puntuaciones y el otro para introducir la puntuación.

```
IEnumerator ObtenerJugadores(){

    txtCargando.text = "Cargando...";
    WWW DataServer = new WWW ("http://" + URLVerPuntuacion);
    yield return DataServer;

    if (DataServer.error != null) {

        Debug.Log ("Problema al intentar obtener los jugadores de la base de datos: " + DataServer.error);
        txtCargando.text = DataServer.error;
    } else {

        txtCargando.text = "";
        ObtenerRegistros (DataServer);
        VerRegistros ();

    }
}

void ObtenerRegistros(WWW DataServer){

    CurrentArray = System.Text.Encoding.UTF8.GetString (DataServer.bytes).Split (";" [0]);
    for (int i = 0; i < CurrentArray.Length-3; i += 3) {

        rankingPuntuaciones.Add(new Jugador(CurrentArray[i],CurrentArray[i+1],CurrentArray[i+2]));
    }

}

void VerRegistros(){

    for (int i = 0; i < rankingPuntuaciones.Count; i++) {

        GameObject obj = Instantiate (Panelpre);
        Jugador jg = rankingPuntuaciones [i];
        obj.GetComponent<setScore> ().SetScore (jg.nombre, jg.puntuacion, jg.actividad);
        obj.transform.SetParent (tfPanelCargarDatos);
        obj.GetComponent<RectTransform> ().localScale = new Vector3 (1, 1, 1);
    }

}
```

Imagen 40. Código Ver Jugadores

Para realizar esta parte, primero es necesario conectarse con la base de datos, después se recorre un bucle *for*, donde previamente se ha buscado cuantos jugadores hay en el *ranking*, que va desde 0 hasta el máximo de jugadores que hay. Una vez realizado esto, se instancia un *Prefab*

llamado Panelpre en el que se han introducido los jugadores, las puntuaciones y el nivel que han realizado la actividad.

```
public string Md5Sum(string strToEncrypt){

    System.Text.UTF8Encoding ue = new System.Text.UTF8Encoding ();
    byte[] bytes = ue.GetBytes (strToEncrypt);
    System.Security.Cryptography.MD5CryptoServiceProvider md5 = new System.Security.Cryptography.MD5CryptoServiceProvider ();
    byte[] hashBytes = md5.ComputeHash (bytes);
    string hashString = "";
    for (int i = 0; i < hashBytes.Length; i++) {

        hashString += System.Convert.ToString (hashBytes [i], 16).PadLeft (2, '0');
    }

    return hashString.PadLeft(32, '0');
}


IEnumerator EnviarJugadores(string nombreJugador, int puntuacionJugador, string actividadpuntuacion){

    Debug.Log (nombreJugador);
    Debug.Log (puntuacionJugador);
    Debug.Log (actividadpuntuacion);

    string hash = Md5Sum (nombreJugador + puntuacionJugador +actividadpuntuacion+ claveSecreta);
    string PostURL = NuevaPuntuacionURL + "nombre=" + WWW.EscapeURL (nombreJugador) + "&puntuacion=" + puntuacionJugador + "&actividad=" + actividadpuntuacion + "&hash=" + hash;

    WWW DataPost = new WWW ("http://" + PostURL);
    yield return DataPost;

    if (DataPost.error != null) {

        print ("Problema al intentar enviar el jugador y su puntuación a la base de datos: " + DataPost.error);
    } else {
        Debug.Log ((System.Text.Encoding.UTF8.GetString (DataPost.bytes)));
    }
```

Imagen 41. Código Enviar Puntuación

Antes que nada, para enviar la puntuación y saber que no se ha modificado al enviarla, la puntuación se encripta, una vez encriptada la puntuación se procede a enviar a la base de datos.

2.2 Diagrama temporal.

Tareas	Fecha inicio	Dias trabajados	Fecha final
Elección del sistema operativo	20-may.-17	2	22-may.-17
Elección del entorno de desarrollo	22-may.-17	2	24-may.-17
Introducción a la programación en C#	25-may.-17	13	7-jun.-17
Despliegue del entorno de desarrollo	25-may.-17	13	7-jun.-17
Documentación para programación de aplicaciones en Unity3D	25-may.-17	20	14-jun.-17
Documentación del Síndrome de Down	19-jun.-17	4	23-jun.-17
Boceto de las actividades a realizar	21-jun.-17	7	28-jun.-17
Acuerdo de las actividades	28-jun.-17	2	30-jun.-17
Desarrollo de la aplicación final	30-jun.-17	41	10-ago.-17
Desarrollo de la memoria	20-ago.-17	93	21-nov.-17

Imagen 42. Tabla diagrama temporal

Imagen 43. Diagrama de Gantt

Capítulo 3. Modelo de negocio

3.1 Introducción al modelo de negocio

Esta aplicación pretende ser una aplicación de ayuda para las personas con Síndrome de Down y otras enfermedades del mismo tipo. Con esta aplicación se pretende sacar beneficio para poder donarlo a Asindown, y para ello es necesario definir primero el modelo de negocio de esta.

3.1.1 ¿Qué es un modelo de negocio?

Un modelo de negocio es el plan de negocio que define qué se va a ofrecer al mercado, cómo hacerlo, quién será el público objeto, cómo se venderá el producto o servicio y cuál será el método para generar ingresos.

Alexander Osterwalder e *Yves Pigneur*, son los creadores de la "plantilla" de negocio más popular en el mundo. Por ello se va a hacer uso de ésta para realizar este modelo de negocio.

3.1.2 ¿Cuáles son los beneficios de utilizar el Canvas de Modelo de Negocio?

- El método Canvas es una herramienta muy práctica ya que permite modificar todo lo que se desee sobre la misma a medida que se va a avanzando en su análisis y testeando la hipótesis.
- El *Canvas model* es muy sencillo, un lienzo muy intuitivo.
- Permite trabajar en equipo.
- Es muy visual, permite ver de manera global todos los aspectos importantes que configuran el Canvas del modelo de negocio.

3.2 Los nueve elementos clave del modelo de negocio

Antes de empezar a realizar el modelo de negocio, primero es necesario definir los diferentes bloques que lo componen.

- **Segmentos de clientes:** Cómo seleccionar a los clientes.
- **Proposición de Valor:** Cómo crear utilidad para los clientes.
- **Canales de venta:** Cómo sale al mercado, con qué comunicación, distribución y canales de vente.
- **Relaciones con el cliente:** Cómo consigue y conserva a los clientes.
- **Ingresos:** Son el resultado de propuestas de valor ofrecidas con éxito a los clientes.
- **Recursos clave:** requeridos para ofrecer y distribuir su propuesta de valor.
- **Actividades clave:** Aquellas que resultan clave para desarrollar el modelo.
- **Colaboraciones clave:** Algunas actividades pueden ser subcontratadas fuera de la empresa.
- **Estructura de costes:** Muestran el resultado de los elementos del modelo de negocio.

3.3 Segmentos de clientes

En este apartado, se deben identificar los diferentes grupos de personas a los que se va a dirigir la aplicación. Es el primer punto que se debe desarrollar a la hora de hacer un modelo de negocio, es necesario tener claro a quien se va a intentar satisfacer las necesidades a través de la app.

A la hora de identificar los posibles clientes, hay que estudiar las propuestas similares que ya se encuentran en el mercado. Esto puede ayudar a estimar a que clientes les puede interesar la idea y que clientes nos pueden interesar a nosotros. Evidentemente no vale encontrar una audiencia a la que le interese la idea pero que detrás no haya ningún poder adquisitivo para comprar dicha solución, por eso se pueden buscar terceras personas interesadas en dicha aplicación que pueda comprarla.

Lo esencial, es localizar uno o más intereses cotidianos para así poder agrupar características comunes. Pero para todo ello, es necesario hacerse unas preguntas primero.

- **¿Para quién se ha creado la aplicación?**

  Esta aplicación está enfocada a todas aquellas personas con discapacidad, tales como el Síndrome de Down, autismo, Asperger, etc., en España. Más adelante se podrá ampliar para más lugares. Como va a ser una aplicación benéfica, también va dirigida para todas aquellas personas que quieran participar en esta causa, ya sea comprando la aplicación o patrocinando alguna de las actividades.

  De todo esto se llega a la conclusión de cuál es el posible segmento de clientes de la aplicación.

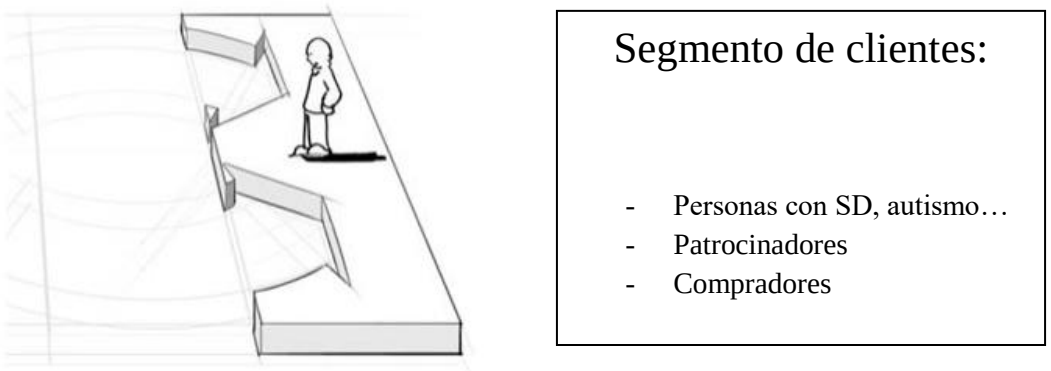


Imagen 44. Segmento de clientes

Así, que ya se puede contar con un bloque para poder construir el modelo de negocio de la aplicación.

3.4 La proposición de valor

Una vez desarrollado lo anterior, y en relación con los segmentos identificados, se deben determinar los valores añadidos y soluciones que se van a ofrecer a los clientes, en otras palabras, cuáles son los puntos estrella o claves de la propuesta y que va es lo que se va a diferenciar de los demás.

Para poder definir correctamente la propuesta de valor, para un determinado segmento de clientes, se tienen que tener en cuenta varios factores que pueden ayudar:

Imagen 45. Propuesta de valor

1. **Precio:** Ofrecer el mismo valor por un precio menor para alcanzar clientes sensibles al precio.
2. **Novedad:** Crear nuevos mercados para satisfacer necesidades que los clientes no tenían identificados explícitamente.
3. **Calidad:** Orientado a entregar un nivel de calidad superior al de los competidores.
4. **Conveniencia:** Enfoque a facilitar "la vida" al cliente, optimizando su tiempo y esfuerzo.
5. **Marca/Status:** Productos asociados a la pertenencia a un cierto grupo social, moda o tendencia.
6. **Desempeño:** Garantizar desempeño superior a los productos de los competidores.
7. **Reducción de riesgos:** Enfoque en minimizar el riesgo que el cliente incurre al comprar el producto o servicio.
8. **Reducción de costes:** Minimizar los costos de la aplicación.
9. **Diseño:** Enfatizar el diseño como elemento identificador.
10. **Customización:** Permitir la adaptación de la oferta a las necesidades y gustos de cada cliente.

Lógicamente, la propuesta de valor no enfocará a todos los puntos anteriores, sino, a un par de ellos. Para ello, hay que contestar a la siguiente cuestión que puede ayudar a describir la propuesta de valor de la idea de aplicación.

- **¿Qué es lo que ofrece al mercado?**

  Se ofrece una aplicación tanto para iOS como para Android, que ayuda a las personas con ciertas discapacidades intelectuales a repasar aquellos conceptos aprendidos ya sea en la escuela como en el lugar de trabajo. Además de poder colaborar con Asindown para que estos puedan invertir más en nuevos proyectos de ayuda.

  Según Asindown en el mercado hay algunas aplicaciones para personas con Síndrome de Down, pero no han sido contrastadas por expertos y no se adaptan totalmente a lo que necesitan. Con esto ya se puede contar con otra pieza más del Canvas.

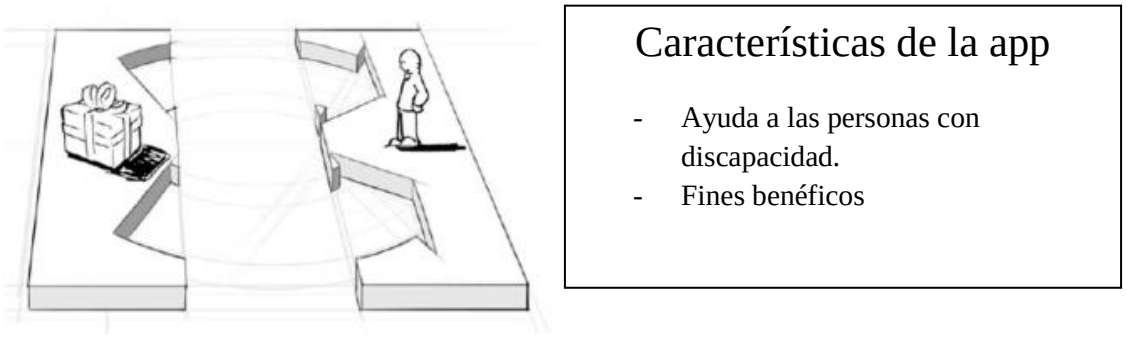


Imagen 46. Canvas Propuesta de valor

3.5 Canales de venta

Una de las partes más importantes de la aplicación son los canales de venta. Para poder vincular la app con el mercado objetivo, es necesario construir los correspondientes canales de distribución, que servirán de mecanismo de comunicación entre ambos y dará soporte para poder entregar las propuestas de valor.

Se deben estudiar las posibilidades de constituir canales directos o indirectos, así como propios o recurrir a otros socios comerciales, con el objetivo de analizar cuál es el más adecuado, rentable y eficiente.

- **¿Por qué canales prefieren los clientes ser contactados?**

  Compradores de la app: Hoy en día la gran mayoría de la población cuenta con un teléfono móvil con acceso a internet, así que ésta será la manera de distribuir la app.

  Patrocinadores: Para poder encontrar patrocinadores será necesario concertar una cita con algunos dueños de tiendas y comercios para venderles la idea.

- **¿Cuál es el canal que mejor funciona?**

  Los sistemas operativos más utilizados son iOS y Android, así que la aplicación estará destinada para las dos plataformas más utilizadas.

- **¿Cuál es el más eficiente?**

  Hoy en día los canales más eficientes para la distribución de la aplicación son la *Play Store y la Apple Store*, por eso la aplicación estará subida para las dos plataformas más utilizadas en el mercado.

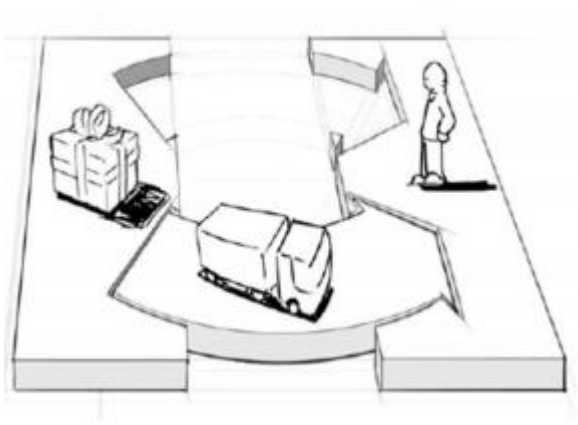
- **¿Cómo descubre el cliente la propuesta de valor?**

  Asindown tiene contactos con la prensa, televisión… así que se anunciará por estas vías, además de publicitarla por las redes sociales más utilizadas hoy en día.

- **¿Cómo evalúa el cliente la propuesta de valor?**

  Las *stores*, a las que se pretende subir la aplicación, cuentan con la posibilidad de dar la opinión sobre las aplicaciones descargadas y sugerir cambios. También estará disponible un perfil en las redes sociales al que se le podrá enviar mensajes directos para hacer peticiones y sugerencias de cambios.

Una vez analizadas las preguntas anteriores, se tendrá completo otro bloque del Canvas del modelo de negocio.



Canales de venta

- Publicación en la APP Store
- Búsqueda de patrocinadores
- Redes sociales

Imagen 47. Canvas Canales de venta

3.6 Las relaciones con los clientes

De forma paralela y coherente con lo anterior, se debe establecer qué tipo de relaciones se pueden establecer entre la causa y los segmentos de mercado, incluyendo el alcance y la integración efectiva. Determinar si el foco se establece en la captación, la fidelización o la sugestión, llevará asociados diferentes tipos de relaciones, como ofrecer servicios personalizados.

Es importante mantener una buena relación con los clientes, en este caso con los patrocinadores y usuarios de la app, para poder retenerles e incentivar la recompra. Para que esto ocurra se debe administrar de un modo satisfactorio las necesidades del cliente y escuchar las sugerencias de mejora.

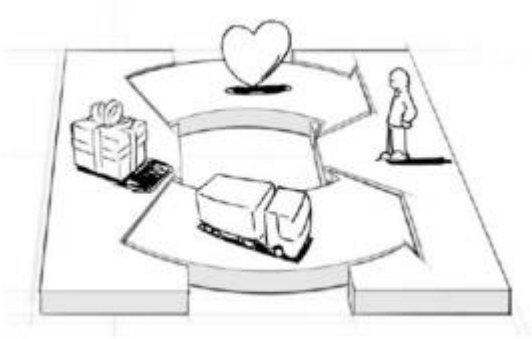
En este punto hay que preguntarse varias cosas.

¿Qué tipo de relación se quiere tener con el cliente? ¿Qué relaciones hay establecidas en este momento? ¿Son costosas? ¿Cómo se integran con el resto de modelo de negocio?

Para responder a las cuestiones anteriores es necesario conocer algunas de las formas más comunes de relacionarse.

1. **Relación de asistencia personal.** Es la interacción entre el cliente y un dependiente que lo ayude en el proceso de ventas o post-compra. En el caso de la app pensada, estaría el caso personal de hablar con los patrocinadores para publicitarles la aplicación.
2. **Relación de asistencia personal dedicada.** Involucra asignar un responsable a la atención específica del cliente. En el caso de la aplicación estarían las redes sociales, donde una vez al mes se revisarían respondiendo a las dudas y haciendo mejoras que hayan sido consideradas.
3. **Relación de autoservicio.** Aquí la compañía no mantiene relación directa con los clientes. Este es el caso de subir la aplicación a las *stores* tanto de Android como de iOS, donde los clientes se descargarían la aplicación, ya sea para el uso o simplemente por el fin benéfico.
4. **Relación automatizada.** Es automatizar procesos simulando una relación personal para crear valor en la atención a segmentos de clientes típicamente masivos. La app no contaría con este apartado de relación con los clientes.
5. **Comunidades.** Algunas compañías crean comunidades en las que buscan comunicarse y entender mejor a sus clientes potenciales y sus clientes actuales. No será necesario crear una comunidad en nuestra aplicación ya que contará con las redes sociales donde se podrán hacer las preguntas necesarias y serán contestadas en la menor brevedad de tiempo.
6. **Co-creación.** Se trata de crear valor junto a los clientes. Este apartado también estaría dentro de las redes sociales y de las opiniones encontradas en las *stores*.

Otra parte del Canvas terminada.



Las relaciones con los clientes

- Redes sociales
- Relaciones personales
- App Store

Imagen 48. Canvas Las relaciones con los clientes

3.7 La estructura de ingresos

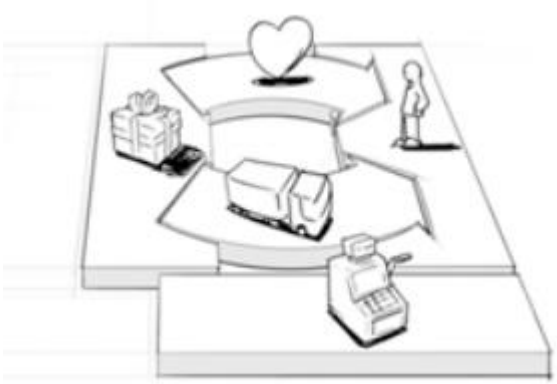
Para terminar de visualizar el modelo de negocios es necesario que se defina la estructura de ingresos, donde se retrata las fuentes a través de las cuales se obtendrán los ingresos benéficos, asegurando la sostenibilidad del modelo. Será necesario apuntar una serie de conceptos relacionados con este bloque.

1. Se pueden definir fuentes de ingresos a partir de los distintos segmentos de clientes.
2. Los ingresos se pueden dividir en dos grandes grupos: los ingresos transaccionales, que provienen de una compra o pago único y los ingresos recurrentes, que provienen de pagos periódicos, constantes…

En un modelo de negocio pueden coexistir diferentes tipos de fuentes de ingresos.

1. **Venta de productos.** La aplicación se venderá en las *stores* para todo aquel que quiera utilizarla o simplemente colaborar por el fin benéfico.
2. **Cobro por uso.** Este apartado no será utilizado en la aplicación.
3. **Publicidad.** Se les cobrará mensualmente a los patrocinadores de la aplicación por tener publicidad en ella.

Quedando clara la estructura de ingresos, se tendrá otro bloque completado del Canvas.



Estructura de ingresos

- Pago por la aplicación.
- Pago por poner publicidad del comercio en la aplicación

Imagen 49. Canvas Estructura de ingresos

3.8 Recursos clave

En este bloque se tendrán que anotar cuáles son los activos más importantes para que el modelo de negocio funcione.

Para empezar, hay que pensar que activos se pueden utilizar para crear una buena propuesta de valor, cuáles permitirán alcanzar los mercados a los que se ha planeado llegar y cuales ayudarán a mantener una buena relación con los segmentos de clientes.

A continuación, se incluyen algunos recursos que se podrían tomar en cuenta.

1. **Recursos Físicos.** Son todos aquellos recursos materiales, como instalaciones, edificios, vehículos, máquinas, puntos de venta, redes de distribución… que otorgan una posición ventajosa frente a los competidores. Serán necesarios, teléfonos móviles, bases de datos y licencias de software.
2. **Intelectuales.** Pueden ser una marca, una patente, una base de datos… Son recursos que se pueden explotar para obtener una ventaja frente a la competencia porque típicamente son únicos. Se necesitan profesionales que sepan hacer la aplicación, deberán saber:

    - Conocimientos avanzados en programación en Unity3D.
    - Conocimientos de Android e iOS.
    - Conocimientos de PHP y MySQL.
    - Conocimientos en diseño de aplicaciones para hacer diseños agradables, sencillos y llamativos.

3. **Humanos.** En toda empresa se requiere una porción de trabajo humano. Pero en aquellas empresas en las que se requiere un uso intensivo de conocimiento o de creatividad, las personas son recursos especialmente valiosos. En esta aplicación hará falta contar con un programador y un diseñador, ya que no solo vale tener una aplicación desarrollada, sino que se necesita de un diseño llamativo y fácil de utilizar.

4. **Financieros.**

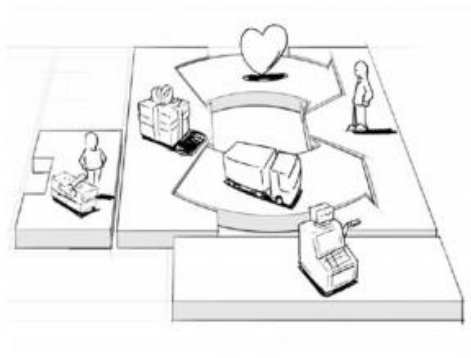


Imagen 50. Canvas Recursos clave

Recursos claves:

- Programadores
- Diseñadores
- Recursos físicos

3.9 Actividades clave

No hay duda de que todo modelo de negocio requiere establecer una serie de actividades para funcionar correctamente, pero no son las mismas para todos los negocios, ni todas las actividades son clave. Con esto en mente, hay que observar los bloques anteriormente trabajados y pensar cuales encierran actividades verdaderamente determinantes para el éxito de la aplicación.

Un negocio puede verse como una serie de actividades o procesos. Dependiendo del giro y de la manera en la que se generan ingresos, algunas actividades son más importantes que otras. Así, por ejemplo, si vendes por Internet, la seguridad en el manejo de transacciones y la administración de datos del cliente son actividades muy importantes; por lo que debes poner especial atención en invertir en la tecnología de protección adecuada y de ser posible en obtener una certificación que avale la seguridad de tu sitio.

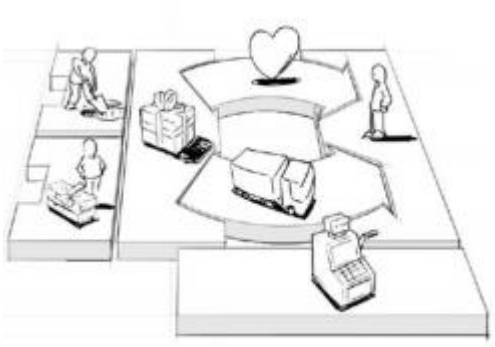
Yves Pigneur y *Alex Osterwalder*, los creadores del Canvas, sugieren una clasificación sencilla que se puede usar para establecer las actividades clave.

1. **Actividades de producción.** Los procesos del diseño y programación de la aplicación son actividades clave, ya que sin ellas no habría aplicación. Sobre todo, si se requiere hacer una gran aplicación con buena calidad.

2. **Actividades de solución de problemas.** Si la oferta es materializar la creatividad y el conocimiento en soluciones para los problemas individuales de los clientes, la actividad de solución es clave.

3. **Actividades de creación de plataformas y construcción de redes.** El mantenimiento de la app es clave. También sería clave mantener las redes sociales actualizadas respondiendo a las cuestiones de los usuarios.

4. **Promociones de la aplicación.** Será necesario promocionar la aplicación para que tenga más ventas.



Actividades clave:

- Venta de la aplicación
- Promoción de la aplicación
- Realización de la aplicación
- Mantenimiento
- Mejoras con las sugerencias de los usuarios

Imagen 51. Canvas Actividades clave

3.10 Colaboraciones clave

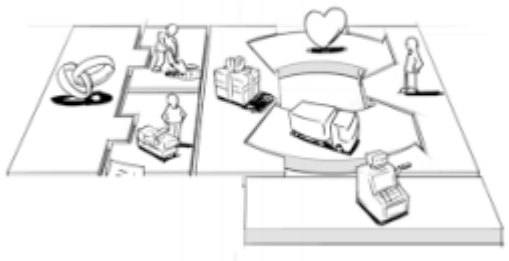
Para que el modelo de negocio desarrollado pueda llevarse a cabo con éxito, es necesario elevar a plano de consciencia la red de clientes clave con los que va a interactuar. Identificar los proveedores y socios nucleares, sin los que el nivel de riesgo pone en peligro la viabilidad del proyecto, evaluar el tipo de asociación que se desarrollará en virtud de alcanzar una adecuada estabilidad y delimitar el alcance de las colaboraciones son puntos clave del estudio para poder fijar adecuadamente los objetivos marcados en este campo.

En el caso de esta aplicación, van a ser necesarios varios colaboradores clave, ya que sin ellos no tendría mucho éxito.

El colaborador más primordial son los patrocinadores de la app, que son los que darían esa pequeña ayuda para poder empezar con el proyecto.

Otro de los colaboradores también esenciales sería Asindown junto con la prensa, ya que ellos son los encargados de publicitar la aplicación para que la gente la conozca y sepa la causa a la que va destinada.

Así que éstos son los colaboradores clave de la aplicación, teniendo así otro bloque del Canvas.



Colaboradores clave:
- Patrocinadores - Asindown - Prensa

Imagen 52. Canvas Colaboradores clave

3.11 Estructura de costo

Finalmente se deben identificar los costes que lleva asociado el lanzamiento del modelo de negocio desarrollado. Para ello se realiza un análisis de 360º a todo el modelo, valorando aquellos directos como los tangenciales y derivados del desarrollo y mantenimiento del modelo. Evidentemente se trata de quedarse con las principales fuentes de coste, puesto que son ellos los que permitirán concluir que el modelo sea rentable (o no), priorizando en el tipo de costes sobre los que se puede actuar a partir del tipo de posicionamiento del modelo desarrollado.

La estructura de costo describe todos los costos incurridos para operar un modelo de negocio, la creación y entrega de valor, el mantenimiento de las relaciones con los clientes y la generación de todos los ingresos. Una vez definidos los recursos clave, las actividades clave… se pueden calcular con facilidad los costos. Pero primero será necesario repasar algunas definiciones del concepto "coste".

- Se entiende por "coste" la medida y valoración del consumo realizado o previsto por la aplicación racional de los factores para la obtención de un producto, trabajo o servicio.
- Se denomina "coste" el total de los gastos incorporados a la producción hasta un momento dado.
- "Coste" es la suma de valores que hay que sacrificar para producir algo.
- El "coste" desde un punto de vista económico, puede considerarse como la suma de bienes y servicios que se utilizan para llevar a cabo un determinado acto productivo.

Es necesario conocer el precio de coste de un producto o servicio, no solo para saber si se debe fabricar o no, sino que también para poder fijar un precio de venta. A continuación, se verán unas definiciones que aclaran lo anterior.

1. El precio del coste es el total de costes relativos a un producto o servicio hasta la fase final de entrega al cliente. Un mismo producto puede tener varios costes, pero sólo tendrá un precio de coste.
2. El precio de coste puede considerarse escalonado de la siguiente forma:
    a. Por el coste de los consumos realizados directa y proporcionalmente para obtener un producto (coste de producción básico).
    b. Por el coste de producción básicos más los gastos correspondientes a los servicios generales de fabricación. (coste industrial).

c. Por el coste industrial más los gastos de distribución y venta. (coste comercial).
d. Por los costes comerciales más los gastos generales de administración y dirección. (Coste total).

Quedando así el siguiente esquema de coste.

Coste total = Coste comercial + Coste industrial + Coste básico

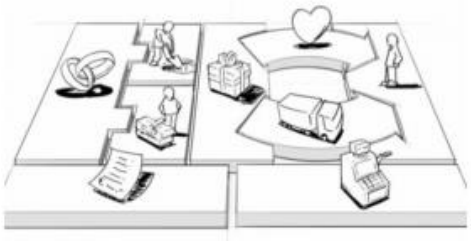
Por otro lado, es conveniente enunciar brevemente algunas de las diversas clasificaciones que se pueden realizar de los costes:

- **Costes externos:** Costes externos son aquellos cuya procedencia son de gastos surgidos en el ambiente externo o comunicados por la contabilidad general.
- **Costes de actividad**: Costes de actividad son aquellos vinculados a la actividad productiva.
- **Costes funcionales:** De acuerdo con la función de la empresa a la que puedan quedar afectados, los costos serán de: Compras, Producción, Administración, Ventas y Distribución.
- **Costos directos:** Costes directos son aquellos que se vinculan directamente al centro o al producto que los ha motivado; no hay necesidad de aplicar método de reparto alguno, tal y como ocurre con las materias primas incorporadas al producto y la mano de obra.
- **Costes indirectos:** Son aquellos que se vinculan al tiempo del periodo productivo. No pueden ser atribuidos a ningún producto o centro en particular.
- **Costes semidirectos:** Son aquellos que, si bien no pueden ser aplicados directamente a un producto, pedido u orden de fabricación, si pueden aplicarse directamente a un centro o función, por ejemplo, el sueldo del programador
- **Costes fijos:** Son aquellos que permanecen constantes e independientemente de las variaciones de la producción para un periodo de tiempo conocido. Por ejemplo, el alquiler de un local para trabajar, seguros…
- **Costes variables**: Son aquellos que varían directamente en función de las unidades de producción o del tiempo.

Se van a reducir a un costo donde se incluirá lo anterior.

- **Costos fijos:** se mantienen iguales a pesar del volumen del producto o servicio ofrecido. Aquí estarían:
    - Costes de servidor y terminales móviles de prueba.
    - Costes de cuotas del programador.
    - Costes de cuotas del diseñador.
    - Costes de la licencia de software.

Así que ya se ha finalizado el último bloque del modelo de negocio.



Estructura de costos:

- Software, licencias, hardware
- Programador, diseñador…

Imagen 53. Canvas estructura de costos

Capítulo 4. Viabilidad económica

Una vez analizado todo el modelo de negocio es necesario saber la viabilidad económica, es decir, si va a salir rentable o no realizar dicha aplicación.

4.1 Inversión Inicial

El desarrollo de esta aplicación requerirá de una inversión inicial, ya que hará uso de algunas herramientas que para ser utilizadas se necesita su previa compra. Así que la inversión inicial sería la siguiente:

1. **Servidores:** Mientras haya pocos usuarios se utiliza un servidor gratuito, pero en un futuro si el nivel de usuarios crece notablemente será necesaria la compra de un servidor.
2. **Licencias:** El uso de Unity3D es gratuito en cierto modo, en un principio se usará la licencia gratuita pero una vez se empiecen a tener ventas, será necesario comprar dicha licencia. Además, serán necesarias las licencias de la *Play Store* de Google y de la *Apple Store*.

Será necesario comprar ordenadores tanto para el programador como el diseñador para que pueden realizar la app. Además de la compra de un terminal para poder probar que funciona todo correctamente.

4.2 Análisis de costes

Se encuentran los siguientes costes de la aplicación:

- **Costes de licencias:** En el apartado anterior se han mencionado los costes de licencias, las cuales en un momento inicial no son necesarias comprar; pero una vez vendida la aplicación sí que lo serían. Por lo tanto, serían los siguientes coses de licencias:

    - Licencia de Unity3D, programa donde se realizará la aplicación, en un principio se comprará la licencia plus ya que permite ingresos hasta 200 mil dólares. El precio de esta licencia son **35 €/mes**. Más tarde se podría considerar comprar la licencia de empresa.
    - Cuota de desarrollador Android, se necesitará para poder subir la aplicación a la Play Store. El precio de esta licencia son **25 €**.
    - Cuota de desarrollador iOS, se necesitará para poder subir la aplicación a la Apple Store. El precio de esta licencia está alrededor de los **100 €.**

    El precio total de las licencias asciende a **160 € al mes**, al año serían (35*12) + 125 = **545 €/año**. Después del primer año habrá que descontar 25 € ya que la cuota de Android son 25 € para toda la vida.

- **Costes salariales:** van a ser necesarias varias personas para poder llevar a cabo esta aplicación y la futura venta. Para ello es necesario tener unos costes salariales que se desglosaran a continuación.

- Programador. Será necesario un programador para que realice la aplicación. En este caso el programador también hará de Project manager dirigiendo en todo momento el proyecto tanto a la hora de programar como a la de montar la aplicación con los diseños previamente estudiados con el diseñador.

- Diseñador. También es importante tener un diseñador para que haga una interfaz fácil y sencilla.

Tanto el diseñador como el programador estarán contratados a jornada completa, y tendrán un sueldo de **1022.25€** netos, que supone un coste de **1485.96€** a la empresa por trabajador.

El papel de comercial para realizar las visitas para poder vender la aplicación se llevará a cabo por las personas de Asindown ya que ellos tienen varios contactos a los que les puede interesar la idea.

Así que el coste salarial en un mes asciende a **2971.92 €.** Teniendo en cuenta que el programador y diseñador tendrán la aplicación hecha en 4 meses como mucho al año saldría un coste de (2971.92 * 4) = **11887.68 €**.

- **Costes de Hardware:** Como se ha dicho anteriormente van a ser necesarios algunos aparatos de hardware para poder realizar dicho proyecto.

  - Ordenadores HP Intel Core i7: **700€ * 2 = 1400€.**
  - Ordenador Mac: **1200€.**
  - Terminal móvil Huawei P10 Lite: **200€.**

Costes	Cantidad mes|unidad	Meses|Unidades	TOTAL
Licencia Unity3D	35 €	12	420 €
Cuota desarrollador Android	25 €	1	25 €
Cuota desarrollador iOS	100 €	1	100 €
Programador	1.486 €	4	5.944 €
Diseñador	1.486 €	4	5.944 €
Ordenador HP	700 €	2	1.400 €
Ordenador MAC	1.200 €	1	1.200 €
Teléfono móvil Huawei P10Lite	200 €	1	200 €
Total			**15.233 €**

Imagen 54. Tabla de costes

4.3 Análisis de ingresos.

1. Compra del servicio de la APP: Los patrocinadores son los que compraran el servicio de la aplicación. Tendrán que pagar una cuota mensual de 300€, que para ellos puede resultar algo simbólico.
2. Compra de la APP: La aplicación al ser benéfica será de pago, así que saldrá a la venta por el precio simbólico de 1€, para que así todo el mundo se pueda permitir aportar a la causa.

   Haciendo una hipótesis de ingresos: 9 comercios serían los patrocinadores de la aplicación, esto sería 9*300 = **2700** € al mes, al año sería **32400 €**, para que los patrocinadores no tengan mucho gasto, cada año se buscarían 9 patrocinadores distintos.; Y pensando que se descargaría la app un promedio de 2000 personas para empezar serían **2000** € más.

Capítulo 5. Trabajo futuro

Una vez transcurrido un periodo de uso de esta aplicación, se podrán recoger las sugerencias y las mejoras por parte de los usuarios y de las personas de su entorno. Así como, comprobar su aceptación y si están interesados en este. Actualmente queda pendiente la aceptación por parte de Asindown, para formalizar los trámites y subirla a las *stores*. En caso afirmativo, habría muchas líneas de trabajo a futuro que realizar; a continuación, se nombran algunas de ellas.

1. Añadir las etapas de atención temprana y envejecimiento.
2. Añadir más actividades propuestas por el centro.
3. Toma de contacto con los usuarios de la app para que aporten mejoras e indiquen lo que no les gusta.
4. Subida de la aplicación a las *stores*.
5. Enfocarlo hacia otro tipo de casos de uso como puede ser el Alzheimer.

Capítulo 6. Conclusión

Tras la finalización del trabajo final de grado, se puede llegar a la conclusión de que cumple los objetivos definidos anteriormente, ya que:

- Hay tres etapas distintas con tres ejercicios para cada etapa.
- Se ha creado una base de datos donde está alojado el ranking de las puntuaciones de los usuarios.
- La aplicación es muy sencilla e intuitiva.
- Es muy visual todo.

Queda pendiente subir la app a las *stores* una vez que den la conformidad los responsables de Asindown; esta última fase está en proceso y no depende de la autora.

Capítulo 7. Bibliografía

1. Cursos en Cursopedia:
    - Unity3D para principiantes: http://www.cursopedia.com/Panel-Unity-3D-para-principiantes
    - Creación de videojuegos en Unity3D: http://www.cursopedia.com/Panel-Creacion-de-Videojuegos-con-Unity-3D

2. Foro Unity3D:
    - Unity Spain: http://www.unityspain.com/
    - Unity3D: https://forum.unity.com/

3. Wikipedia Unity3D: https://es.wikipedia.org/wiki/Unity_(motor_de_juego)
4. Wikipedia Síndrome de Down: https://es.wikipedia.org/wiki/S%C3%ADndrome_de_Down

5. Asindown: https://www.asindown.org/

6. Wikipedia MySQL: https://es.wikipedia.org/wiki/MySQL

7. Hostinguer: https://www.hostinger.es/

8. Apple Store: https://itunes.apple.com/es/genre/ios/id36?mt=8

9. Play Store: https://play.google.com/store/apps?hl=es

10. Canvas Modelo de negocio: Asignatura tecnologías y modelo de negocio. Máster en desarrollo de aplicaciones sobre dispositivos móviles.

11. Alexander Osterwalder: http://alexosterwalder.com/