



Grado en Ingeniería en Tecnologías de la Telecomunicación

Curso Académico 2018/2019

Trabajo Fin de Grado

TALLERES DIDÁCTICOS ORIENTADOS A LA
DIFUSIÓN DE EXPERIENCIAS VR ENTRE
ALUMNOS DE ENSEÑANZAS MEDIAS.

Autor : Jonatan Santana Pero

Tutor : Pedro de las Heras Quirós

Trabajo Fin de Grado/Máster

Título del Trabajo con Letras Capitales para Sustantivos y Adjetivos

Autor : Jonatan Santana Pero

Tutor : Pedro de las Heras Quirós

La defensa del presente Proyecto Fin de Carrera se realizó el día de
de 20XX, siendo calificada por el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de 2019

*Dedicado a
mi mismo*

Agradecimientos

AGRADECIMIENTOS

Resumen

Desde el principio se pensó en este proyecto como una serie de talleres o actividades para introducir a una serie de tecnologías a estudiantes de instituto que estén interesados en introducirse en ellas.

El objetivo de este proyecto es introducir 3 tecnologías , Bettleblocks, A-Frame, y el Micro-Bit como interfaz de usuario.

- ¿De qué va este proyecto? ¿Cuál es su objetivo principal?
- ¿Cómo se ha realizado? ¿Qué tecnologías están involucradas?
- ¿En qué contexto se ha realizado el proyecto? ¿Es un proyecto dentro de un marco general?

Lo mejor es escribir el resumen al final.

Índice general

| | |
|--|----------|
| 1. Introducción | 1 |
| 1.1. Experiencias VR | 1 |
| 1.1.1. Utilidad VR en la educación | 2 |
| 1.2. Diseño de figuras 3D | 2 |
| 1.2.1. Entornos VR con figuras en 3D | 2 |
| 1.3. Microbit | 3 |
| 1.3.1. Microbit como interfaz de usuario | 3 |
| 1.4. Estructura de la memoria | 3 |
| 2. Objetivos | 5 |
| 2.1. Objetivos | 5 |
| 2.1.1. Implementar entornos VR en la educación | 5 |
| 2.2. Motivación | 5 |
| 2.3. Metodología y Plan de Trabajo | 6 |
| 2.3.1. Metodología | 6 |
| 2.3.2. Plan de trabajo | 7 |
| 3. Creación de experiencias VR | 9 |
| 3.1. Tecnologías | 9 |
| 3.2. Diseño 3D | 9 |
| 3.2.1. Programación mediante bloques | 9 |
| 3.2.2. Beetleblocks | 10 |
| 3.2.3. Blender | 14 |
| 3.3. A-frame | 15 |

| | |
|---|-----------|
| 3.3.1. Programación A-frame mediante HTML | 15 |
| 3.3.2. Programación A-frame mediante JS | 15 |
| 3.4. Microbit | 16 |
| 3.4.1. Programación Microbit mediante bloques(Makecode) | 18 |
| 3.4.2. Micropython | 18 |
| 4. Diseño e implementación de los talleres | 19 |
| 4.1. Introducción y propósito de los talleres | 19 |
| 4.2. Taller 1 (Creación y exportación de un objeto 3D) | 20 |
| 4.2.1. Bettleblocks | 20 |
| 4.2.2. Blender | 20 |
| 4.3. Taller 2 (Creación de un entorno VR) | 22 |
| 4.3.1. Creación de entorno VR con A-Frame | 22 |
| 4.3.2. Animación básica con A-frame. | 24 |
| 4.3.3. Anexo: Lleva tu experiencia VR a tu smartphone | 25 |
| 4.4. Taller 3(Microbit como IU en entorno VR) | 27 |
| 5. Conclusiones | 29 |
| 5.1. Consecución de objetivos | 29 |
| 5.2. Aplicación de lo aprendido | 29 |
| 5.3. Lecciones aprendidas | 29 |
| 5.4. Trabajos futuros | 30 |
| A. Manual de usuario | 31 |
| Bibliografía | 33 |
| B. Introducción | 35 |
| B.1. Sección | 35 |
| B.1.1. Estilo | 35 |
| B.2. Estructura de la memoria | 36 |
| C. Estado del arte | 37 |
| C.1. Sección 1 | 37 |

Índice de figuras

| | |
|---|----|
| 3.1. Blender | 14 |
| 4.1. Blender | 20 |
| B.1. Página con enlaces a hilos | 36 |
| C.1. Estructura del parser básico | 38 |

ÍNDICE DE FIGURAS

Capítulo 1

Introducción

1.1. Experiencias VR

La realidad virtual, es la sensación de estar inmerso en un entorno con objetos o escenas de apariencia real. El usuario puede sumergirse en imágenes 3D realistas, generadas por ordenador, a través de tecnología como los visores de realidad virtual una experiencia que luego puede enriquecerse con otros dispositivos.

En los últimos años hemos visto un crecimiento en su importancia y uso, se trata de una tecnología relativamente temprana tanto en su concepción como en su aplicación, hasta los años 90 no se consiguió simplificar los grandes simuladores de realidad virtual en dispositivos portátiles, desde cascos de realidad virtual a gafas en los que los smartphones se encargan de hacer realidad la experiencia.

Entre los años 2015 y 2016 se vivió el gran avance en estos dispositivos al animarse grandes marcas como HTC o Sony a sacar estos dispositivos a la venta para el público.

A la hora de hablar de la realidad virtual nos centraremos en las denominadas experiencias, se trata de entornos en realidad virtual que nos permiten situarnos en medio de la escena, de la que podemos ser espectadores, o interactuar con ella, ayudándonos de sensores o controladores.

Si bien es cierto que el mayor uso de la realidad virtual va dirigido a videojuegos, también es empleada en campos como el de la medicina o la educación.

1.1.1. Utilidad VR en la educación

El uso de la realidad virtual en la educación, es algo que se está empezando a introducir en la sociedad, ha estado limitada durante años a la formación de pilotos aéreos en carísimos simuladores pero gracias a la mejora en la tecnología y la reducción en costes, podemos ir viendo como poco a poco más universidades e institutos introducen la realidad virtual, como apoyo para la enseñanza.

Una gran baza de esta tecnología es su accesibilidad, con unas gafas de cartón como las Cardboard de Google y un móvil, podemos ser capaces de experimentar miles de experiencias en realidad virtual.

Tampoco podemos obviar su inmersión que aumenta la motivación y aporta un mayor impacto en los procesos de aprendizaje.

Declaraciones como las de Baptiste Grève, creador de Unimersiv, una plataforma de experiencias virtuales, señala lo positivo de esta tecnología dado que el cerebro humano retiene el

10

1.2. Diseño de figuras 3D

El diseño de las figuras en 3D se realiza mediante software de diseño como Blender o de una forma mas sencilla mediante una programación por bloques como la que nos ofrece Beetleblocks.

A la hora de diseñar nuestra figura si nos centramos en la programación por bloques, deberemos tener en cuenta que que nuestra figura puede ser tan compleja como nosotros queramos, pero que para aprender a diseñar, la forma más correcta será empezar con una figura base, para más tarde crear una más compleja ayudándonos de la combinación de figuras simples.

1.2.1. Entornos VR con figuras en 3D

Los entornos de realidad virtual se componen de figuras en 3 dimensiones.

Gracias a utilidades como A-Frame podemos crear estos entornos y a herramientas como Beetleblocks y Blender crear figuras para verlas representadas en nuestras experiencias.

1.3. Microbit

BBC micro: bit es una pequeña tarjeta programable de 4x5 cm diseñada para que aprender a programar sea fácil, divertido y al alcance de todos.

Gracias a la gran cantidad de sensores que incorpora, sólo con la tarjeta se pueden llevar a cabo centenares de proyectos. BBC micro: bit también es una plataforma IoT (Internet of Things), lo que la hace muy interesante para usuarios avanzados.

Y es Open Source, por supuesto. Tanto el hardware como el software de micro:bit es de código abierto.

1.3.1. Microbit como interfaz de usuario

Dentro de las múltiples aplicaciones que le podemos dar al Micro: bit , nos centraremos en usarlo como interfaz de usuario, es decir, usarlo como un controlador para movernos por nuestra experiencia de realidad virtual.

1.4. Estructura de la memoria

Tras esta introducción en la que hemos visto una visión de la realidad virtual , los entornos , el diseño de objetos en 3D mediante bloques, y el Microbit y sus multiples posibilidades, continuaremos describiendo los objetivos que se presentan en este trabajo de fin de grado, en el capítulo 2.

A continuación en el capítulo 3 aprenderemos a usar las tecnologías que se nos presentan , tanto para el diseño 3D , Beetleblocks y Blender, la creación de entornos de realidad virtual, A-Frame, y por último el uso del Microbit como controlador para nuestra experiencia.

En el capítulo 4 nos encontramos la serie de 3 talleres o prácticas de las que consta este trabajo, el primero centrado en el diseño de figuras en 3D, el segundo para aprender a crear nuestro entorno de realidad virtual, y por último un tercero para usar nuestro Microbit como controlador en el entorno y poder movernos con él.

Para finalizar tenemos el capítulo 5 con las Conclusiones, donde se analizan los resultados obtenidos de las prácticas y las posibles mejoras futuras que se podían aplicar.

Capítulo 2

Objetivos

2.1. Objetivos

Ya hemos visto una breve introducción de las experiencias de realidad virtual, y del diseño de objetos en 3 dimensiones, ahora toca centrarse en los objetivos de este trabajo de fin de grado.

2.1.1. Implementar entornos VR en la educación

El primer objetivo a la hora de realizar este trabajo de fin de grado, es exponer la utilidad de la realidad virtual en la educación, y como gracias a ella podemos llegar a aprender a programar figuras sencillas en 3D y experimentar con un entorno de realidad virtual de una forma sencilla e interactiva.

2.2. Motivación

La principal motivación que ha llevado a realizar este trabajo de fin de grado, es la propia tecnología en sí, siempre me ha llamado la atención , tanto el diseño de objetos en 3D como las experiencias en realidad virtual.

El gran potencial que tienen, y los múltiples usos que se le pueden dar me han parecido muy interesantes desde un primer momento, gracias a la realización de este trabajo podía a aprender acerca de todo esto y además darle una utilidad en forma de talleres para que alguien aprenda de una forma sencilla y amena.

2.3. Metodología y Plan de Trabajo

En esta sección se expone la metodología y el plan de trabajo que se han llevado a cabo para la realización de este trabajo de fin de grado.

2.3.1. Metodología

La metodología que se ha seguido para la realización de este trabajo de fin grado, esta orientada en la construcción de una serie de talleres didácticos.

Un taller es un programa educacional corto e intensivo, para una cantidad relativamente pequeña de personas, en un área de conocimientos determinada que hace énfasis en la participación para la resolución de problemas.

Los pasos que se han seguido para la creación de los talleres son los siguientes:

- Definir los objetivos del taller.
- Adecuarlos a los participantes, en este caso se trata de alumnos de enseñanzas medias por lo que deben ser acordes a sus conocimientos.
- Definir el formato del taller, será un taller en el que los alumnos harán experiencias guiadas pero en las que participaran activamente en ciertos momentos para involucrarlos con el taller.

Cada taller a su vez seguirá una metodología común que es la siguiente.

- Presentación del tema general del taller.
- Planteamiento de los objetivos del taller.
- Presentación del software que se usará.
- Fomentar la participación activa
- Una vez finalizado resumir la sesión y pedir feedback al grupo.

2.3.2. Plan de trabajo

En base a la realización de los talleres se debía tener un conocimiento profundo de todo el software que se usa en ellos, debido a esto se ha realizado un plan de trabajo en el que primeramente se estudiaron las 2 tecnologías Beetleblocks y A-Frame. Gracias a la ayuda del tutor mediante tutorías a través de Hanghouts y mediante la investigación y realización de tutoriales se obtuvo una base sólida acerca de ellas. Posteriormente se contempló el uso de Tensorflow para añadir un tipo de interfaz de usuario a la experiencia de realidad virtual, este primer planteamiento fue desechado en detrimento de la introducción del Micro: bit, la popularidad y variedad de aplicaciones de este último fue determinante, y por último se desarrolló una interfaz de usuario en la que el controlador era el Micro: bit.

Una vez cimentado un conocimiento acerca de Beetleblocks, A-Frame y Micro:bit se pasó a plantear los talleres y se dividieron en una serie de 3 talleres autoconclusivos pero a su vez recursivos entre ellos. Con esto conseguimos que realizando uno de ellos se obtuvieran conocimientos acerca de la tecnología en cuestión y a su vez alguien que realizara los tres obtuviera una experiencia completa de realidad virtual .

Capítulo 3

Creación de experiencias VR

3.1. Tecnologías

En este capítulo nos centraremos en conocer el software y el hardware que vamos a usar en este trabajo, además aprenderemos a usarlo, y veremos unos pequeños ejemplos.

3.2. Diseño 3D

Para diseñar nuestros objetos en 3D, nos ayudaremos de Beetleblocks una potente herramienta online que usa la programación mediante bloques para realizar nuestros objetos de una manera más gráfica y amena.

Una vez tengamos nuestra figura deseada en Beetleblocks, usaremos la segunda de las herramientas para el diseño en 3D, se trata de Blender un software open source. Blender posee numerosas utilidades, pero en este caso nos centraremos solamente en valernos de él, para lograr la extensión deseada de nuestro objeto 3D.

3.2.1. Programación mediante bloques

La programación mediante bloques, facilita mucho las cosas si no tenemos experiencia de ningún tipo con la programación, en este trabajo de fin de grado la veremos tanto en Beetleblocks como cuando usemos el Micro: bit.

3.2.2. Beetleblocks

Beetleblocks es un proyecto creado por Eric Rosenbaum, Duks Koschitz, y Bernat Romagosa, además de la ayuda en la programación de Jens Mönig.

Beetle Blocks esta basado en Scratch e implementado usando Snap! y ThreeJS.









A la hora de comenzar a usar Beetle Blocks, estos son los pasos a seguir:

- Creamos una cuenta.
- Categorías de los bloques.
 1. Motion: Donde se engloban los bloques para movernos por la malla
 2. Shapes: Figuras y formas predeterminadas
 3. Sensing: funciones para usar con nuestros bloques
 4. Variables: creación y uso de variables de entorno
 5. Control: operadores para nuestros conjuntos de bloques, eventos , funciones etc..
 6. Colors: Bloques para dar color a nuestros diseños
 7. Operators: operadores matemáticos para usar en nuestros bloques
 8. Myblocks: bloques propios creados en el proyecto.
- En el capítulo ?? (ojo, otra referencia automática) se muestran los objetivos del proyecto.



Guía de uso Beetleblocks

Para empezar a usar beetleblocks vamos a empezar a crear una serie de figuras básicas. Para aprender a usar beetleblocks expongo a continuación una serie de ejemplos prácticos, en los que enseño como crear figuras básicas, crear círculos, moverse en los 3 ejes espaciales y por último un objeto que usa todo lo aprendido para la creación de unos auriculares.

■ Figuras

1. Dibujar línea: Usamos la orden  que se encuentra en la pestaña de Shapes
2. Creación de variable: En la pestaña Variables tenemos una primera opción make a variable que nos permite crear una variable global para utilizar.
3. Creamos una variable x a la que podemos asignar un valor gracias al bloque  en el que introducimos el valor que queremos que tenga
4. Ahora usamos un bucle para repetir el proceso de la creación de las líneas de una figura geométrica, los bucles vienen indicados en beetleblocks por el bloque  en el hueco del bloque insertaremos nuestra variable, esto provoca que todos los bloques que insertemos a posteriori se repiten el número de veces que valor tenga nuestra variable.
5. Dentro del bucle insertaremos 2 bloques
 - El bloque  el cual dibujará una línea que tendrá la longitud igual al valor que insertemos en su hueco. En este hueco insertaremos nuestra variable arrastrándola, de tal forma que quedara así 
 - El bloque  nos permite girar la dirección de nuestro puntero en este caso lo queremos girar tantos grados como lados tenga nuestra figura geométrica esto lo haremos de una forma muy sencilla con el bloque  Con el que dividimos por el valor de nuestra variable y los combinamos 
 - Por lo tanto, nuestro bucle completo quedaría así.



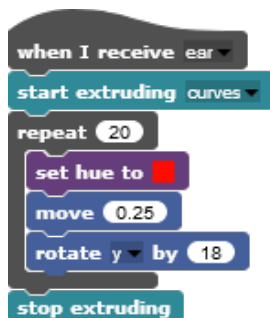
6. Dos bloques más que nos serán muy útiles para crear cualquier figura son  que nos inicia la serie de bloques que anidemos cuando hagamos click. Y  que nos deja el plano en blanco.

7. Finalmente, nuestro bloque para crear figuras geométricas quedaría así.




■ Círculos

Para crear círculos de una manera muy sencilla simplemente empezando a dibujar curvas o líneas nos moveremos hasta completar los 360 grados del círculo girando x grados uno de los ejes. En este ejemplo dibujaríamos un círculo rojo moviéndonos 0.25 hacia delante cada vez que giramos 18 grados, esto lo haríamos 20 veces para completar los 360 grados.

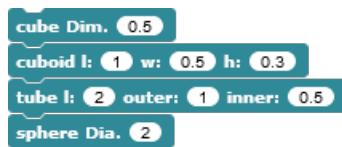


■ Moverse por los ejes

Dependiendo de como dibujemos nuestras figuras deberemos movernos por los ejes y orientar la mariquita que sirve como puntero de orientación para dibujar. El bloque que se usa para movernos y situarnos en donde queremos es este. 

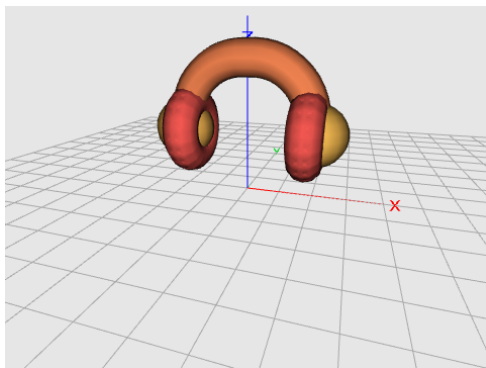
- Rellenar
- Figuras predeterminadas

Las figuras predeterminadas que nos ofrece bettleblocks son el cubo (en dos formas), el cilindro y la esfera cuyos bloques se corresponden con:



- Auriculares

Para probar el manejo de Bettleblocks he creado una figura sencilla , se trata de unos cascos en los que uso una serie de bloques parra dibujarlos.



Formato

Por último para continuar usaremos AFRAME para poder visualizar nuestra imagen en 3D y para esto necesitaremos exportar nuestra figura en formato .STL. El siguiente programa que usaremos será Blender para poder dar un formato compatible con A-Frame a nuestra figura.

3.2.3. Blender

Blender es una potente herramienta open source de creación 3D. En este trabajo de fin de grado le vamos a dar un uso muy concreto, se trata de pasar nuestra figura creada con Beetle-blocks a un formato compatible con A-Frame (.obj)

Abrimos Blender lo primero que vemos será una pantalla como esta

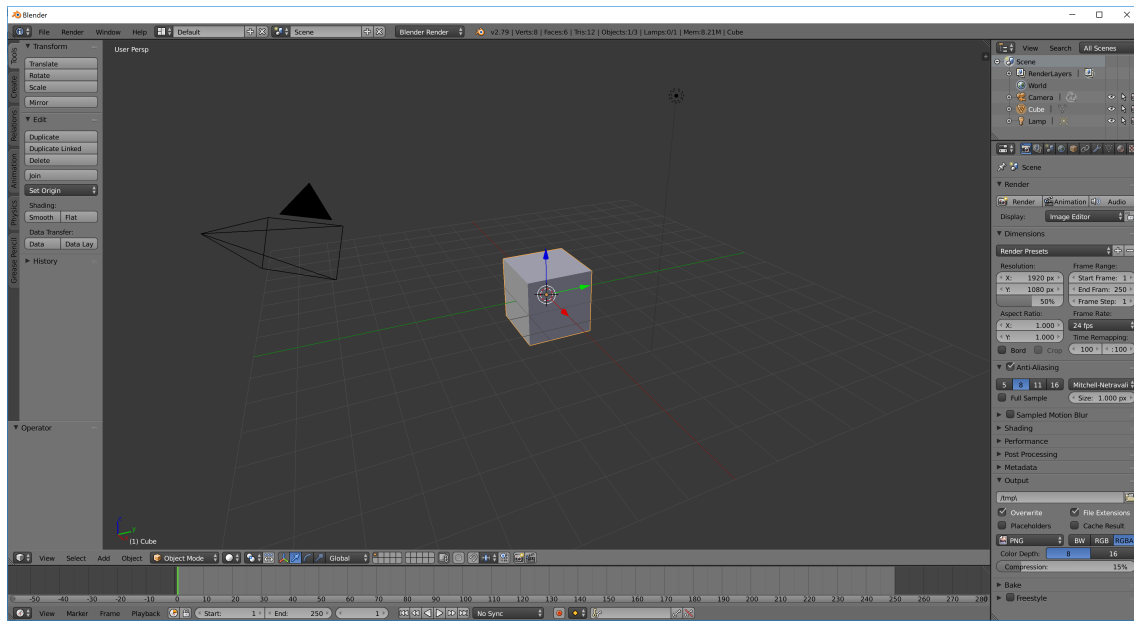


Figura 3.1: Blender

- Los pasos a seguir para exportar nuestra figura en **.obj** y **.mtl** son los siguientes:
 1. Nos posicionamos encima del cubo y presionamos **x** para borrarlo.
 2. pulsamos espacio y escribimos **import stl**.
 3. Pulsamos espacio de nuevo y escribimos **export obj** seleccionando las checkboxes **Write Normals**, **Write Materials**

3.3. A-frame

Como ya sabemos A-Frame es el framework que nos permite crear nuestra experiencia de realidad virtual, su aplicación es posible mediante HTML o JavaScript aunque en la serie de talleres, el uso que le daremos será mediante documentos HTML, en este capítulo aprenderemos a usarlo en las dos vertientes.

A continuación expongo una breve guía de uso de ambas

a-Frame is a web framework for building virtual reality (VR) experiences. A-Frame is based on top of HTML, making it simple to get started. But A-Frame is not just a 3D scene graph or a markup language; the core is a powerful entity-component framework that provides a declarative, extensible, and composable structure to three.js.

Originally conceived within Mozilla and now maintained by the co-creators of A-Frame within Supermedium, A-Frame was developed to be an easy yet powerful way to develop VR content. As an independent open source project, A-Frame has grown to be one of the largest VR communities.

A-Frame supports most VR headsets such as Vive, Rift, Windows Mixed Reality, Daydream, GearVR, Cardboard, Oculus Go, and can even be used for augmented reality. Although A-Frame supports the whole spectrum, A-Frame aims to define fully immersive interactive VR experiences that go beyond basic 360° content, making full use of positional tracking and controllers.

3.3.1. Programación A-frame mediante HTML

REKLLLENAR

3.3.2. Programación A-frame mediante JS

Gracias a la importación de archivos en JavaScript podemos añadir múltiples funcionalidades a A-Frame.

Un paquete muy utilizado es el `aframe-extras`¹ se trata de una serie de Add-ons y helpers para A-Frame, e incluye componentes de control, modelos predefinidos, primitivos ...

¹<https://github.com/donmccurdy/aframe-extras>.

Otro recurso muy utilizado para A-Frame y basado en JavaScript son los components², estos nos habilitan una serie de funciones y funcionas como módulos, existe una amplia biblioteca de components, entre ellos hay algunos que se organizan en paquetes como superframe.³

El uso de cada component es particular y puede encontrarse en su documentación, pero generalmente el uso conlleva a vincularlo con una entidad `<a-entity>` y configurarlo.

Tomamos como ejemplo particle system, su uso sería de la siguiente forma:

- Importamos el component de la misma manera que ya lo hicimos con A-Frame en nuestro index.html

```
<head>
  <script src="https://aframe.io/[...].min.js">
    </script>
  <script src="https://unpkg.com/aframe-[...]-component.min.js">
    </script>
</head>
```

- Y a continuación vinculamos particle system con una entidad

```
<body>
  <a-scene>
    <a-entity particle-system="preset: snow" position="0 0 -10">
      </a-entity>
    </a-scene>
  </body>
```

3.4. Microbit

”Micro:bit es una pequeña computadora programable, diseñada para hacer que el aprendizaje y la enseñanza sean fáciles y divertidos!”

²<https://aframe.io/docs/0.9.0/introduction/entity-component-system.html>.

³<https://github.com/supermedium/superframe>.

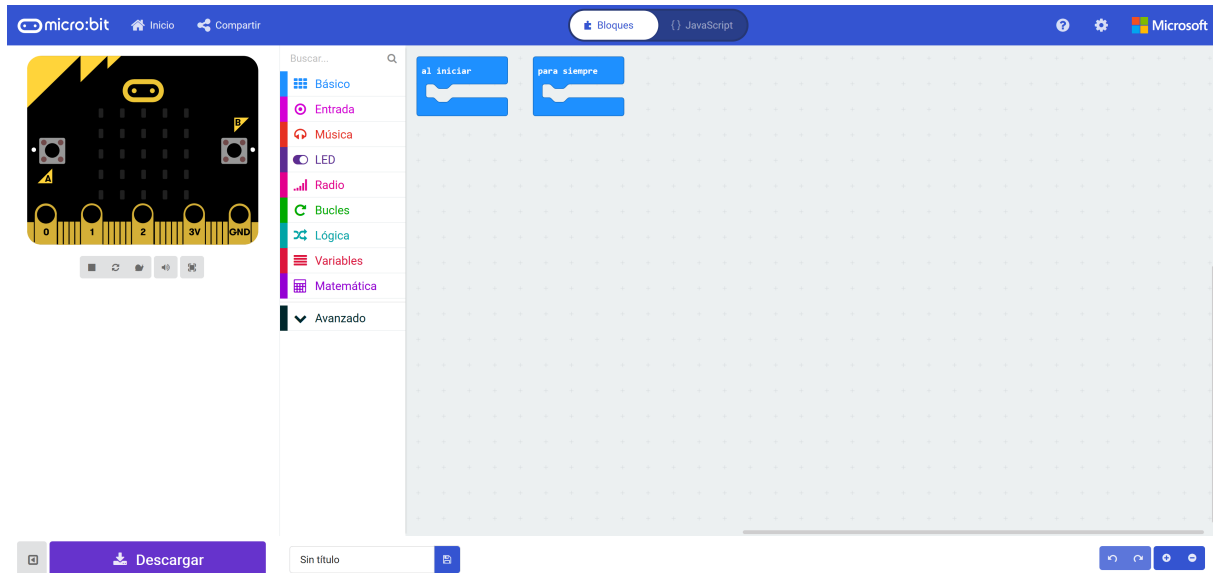
Esta es la frase que encontramos al entrar a <https://microbit.org/es>, y eso es Micro:bit, un sistema de Hardware embebido, diseñado por la BBC para la enseñanza informática en UK. Su primera aparición fue el 12 de Marzo de 2015, y se empezó a distribuir en Febrero de 2016, consta de un procesador ARM CortexM0, acelerómetro y magnetómetro, conectividad Bluetooth y USB, 25 leds, 2 botones programables y puede ser alimentada por medio de pilas o USB. Las entradas y salidas del dispositivo están formadas por 5 anillos conectores que forman parte de un conector mayor de 23 pines.



Hay dos principales editores de código en la web de Micro:bit, Makecode, basado en la programación mediante bloques, y MicroPython una variación de Python.

3.4.1. Programación Microbit mediante bloques(Makecode)

Para la programación mediante bloques del Micro:bit vamos a utilizar *Makecode* ⁴



Con una interfaz muy parecida a la de Beetleblocks divide sus bloques en grupos, que aportan distintas funciones como Bucles , acciones para los LEDs, Lógica, Matemática ... etc

En la web de Makecode disponemos de numerosos tutoriales para familiarizarnos con su uso.

3.4.2. Micropython

Micropython es el otro editor de código que esta presente en la web oficial de Micro:bit, permite al usuario escribir scripts de Python en el editor web de Micro bit los cuales se combinan con el firmware de MicroPython y se cargan en el dispositivo.

⁴<https://makecode.microbit.org>

Capítulo 4

Diseño e implementación de los talleres

4.1. Introducción y propósito de los talleres

El principal motivo de este trabajo de fin de grado es introducir todas estas herramientas que hemos explicado previamente a alumnos de enseñanzas medias, y para ello voy a desarrollar una serie de tres talleres, estos son autoconclusivos pero a su vez cada uno de ellos utiliza lo que aprendimos en el anterior haciendo que una vez finalizamos los tres tengamos una experiencia de realidad virtual completa creada con A-Frame en la que veamos un objeto creado con Beetleblocks y que puede ser controlada con nuestro Micro:bit para movernos por ella.

Por tanto el desarrollo y contenido de los talleres es el siguiente:

- **Taller 1:** Crea tu objeto 3D con Beetleblocks.

En este taller nos centraremos en la creación de un objeto 3D con Beetleblocks, aprenderemos a exportarlo y a darle el formato adecuado para el siguiente taller.

- **Taller 2:** Crea tu experiencia de realidad virtual con A-frame.

En este taller aprenderemos a usar A-Frame creando un entorno de realidad virtual en el que esta incluido nuestro objeto creado anteriormente con Beetleblocks.

- **Taller 3:** Micro:bit como interfaz de usuario para tu experiencia de realidad virtual.

En este taller convertiremos nuestro Micro:bit en un controlador para nuestra experiencia de realidad virtual

4.2. Taller 1 (Creación y exportación de un objeto 3D)

Este primer taller tiene el objetivo de que aprendas a crear un objeto en 3D con Beetleblocks y su accesible método de programación mediante bloques. Además en la última parte del taller dejaremos nuestro objeto preparado para usarlo en un entorno VR creado con A-Frame.

4.2.1. Bettleblocks

Beetleblocks nos pone las cosas fáciles a la hora de crear objetos 3D, con este taller aprenderemos a ...

4.2.2. Blender

Ya hemos aprendido como crear nuestro objeto, ahora solo nos falta que tenga una extensión .obj para ello vamos a usar un software OpenSource de diseño 3D como es Blender.

Los pasos a realizar son muy sencillos, solo usaremos Blender como si de una herramienta de conversión se tratase.

Abrimos Blender lo primero que vemos será una pantalla como esta

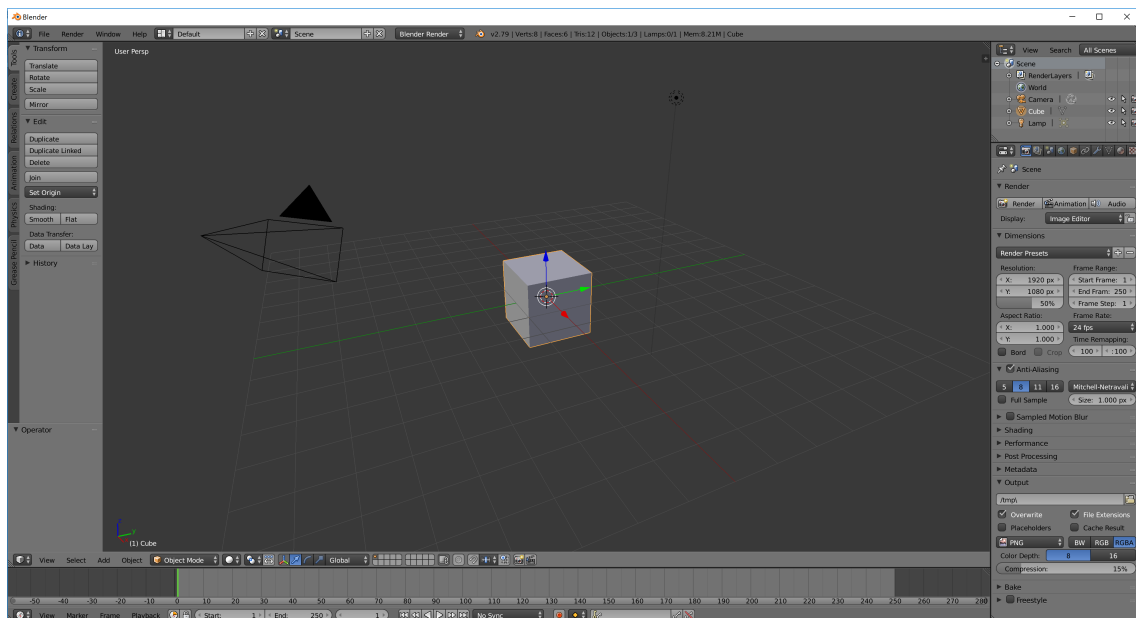
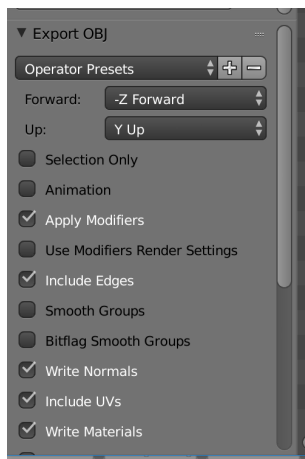


Figura 4.1: Blender

- Los pasos a seguir para exportar nuestra figura en *.obj* y *.mtl* son los siguientes:

1. Nos posicionamos encima del cubo y presionamos *x* para borrarlo.
2. pulsamos espacio y escribimos *import stl*.
3. Pulsamos espacio de nuevo y escribimos *export obj* seleccionando las checkboxes *Write Normals, Write Materials*



4.3. Taller 2 (Creación de un entorno VR)

En este segundo taller aprenderemos a crear una experiencia de realidad virtual con A-Frame, además importaremos nuestro objeto creado en el primer taller con Beetleblocks, y podremos visualizarla en nuestro smartphone.

4.3.1. Creación de entorno VR con A-Frame

Para crear nuestra experiencia de realidad virtual el primer paso es crear nuestro `index.html` en el directorio de nuestro proyecto, para esto crearemos un repositorio de la siguiente manera: En un terminal unix escribimos

```
mkdir A-Frame
cd A-Frame
makefile index.html
```

A-Frame puede ser utilizado desde un documento plano de HTML sin la necesidad de instalar nada.

Para empezar a usar A-Frame en nuestro `index.html` debemos importarlo en la cabecera de esta manera:

```
<head>
  <script src="https://aframe.io/releases/0.9.0/aframe.min.js"></script>
```

Con esto ya hemos cargado la versión de A-Frame y podemos dar paso a nuestra primera prueba.

■ Primera figura

Un ejemplo sencillo para aprender a usar A-Frame consiste en cargar un primitive, plantillas de elementos que están dentro del código de A-Frame, gracias a ellos podemos crear un cubo , un cilindro, una esfera . . . de una forma muy sencilla. A la hora de usarlos en nuestro *index.html* basta con nombrarlos como si se tratara de una entidad, a los primitives se le pueden dar diferentes propiedades. Siempre que cargamos una figura en A-Frame tiene que ser dentro de la escena, en nuestro documento html `<a-scene>`.

Cargamos un cubo y una esfera en nuestra escena de la siguiente manera:

```

<body>
  <a-scene>
    <a-box position="-1 0.5 -3" rotation="0 45 0" color="#4CC3D9">
    </a-box>
    <a-sphere position="0 1.25 -5" radius="1.25" color="#EF2D5E">
    </a-sphere>
    <a-sky color="#ECECEC"></a-sky>
  </a-scene>
</body>

```

Además podemos crear nuestros propios primitivos, algo que veremos en la siguiente sección de este capítulo, exportando un archivo JavaScript.

Ahora que ya sabemos la metodología para cargar objetos en A-Frame, el siguiente paso a realizar es cargar nuestro objeto creado con Beetleblocks, ya con el formato **.obj** que le hemos conseguido dar con Blender.

- El proceso para cargarlo en nuestro index.html es el siguiente:

1. Inicializamos nuestras variables **.obj**, **.mtl**, y **textura**.

```

<a-entity
  obj-model="obj: #crate-obj;" mtl="#crate-mtl"
  material="src: #crate-texture">

</a-entity>.

```

2. Cargamos el **.obj**, **.mtl**, y **textura**¹ que queramos².

```

<a-assets>
  <a-asset-item id="crate-obj" src="models/cascos3.obj"></a-asset-item>
  <a-asset-item id="crate-mtl" src="models/cascos3.mtl"></a-asset-item>
  
</a-assets>

```

¹Para la textura nos vale cualquier **.jpg**.

²En la propiedad **src** indicamos la ruta del archivo.

4.3.2. Animación básica con A-frame.

A-Frame nos permite añadir algo de animación básica a nuestra figura, gracias a la importación de paquetes en JavaScript.

Nuestro objetivo va a ser que nuestra figura rote, para hacerlo vamos a valernos del paquete `aframe-animation-component`.

Lo importamos en nuestra cabecera `<head>`

```
<script src="https://rawgit.com/ngokevin/aframe-animation-component
  /master/dist/aframe-animation-component.min.js">

</script>
```

Para usar la animación usamos `<a-animation>` dentro de `<a-entity>`

```
<a-entity [...]>

  <a-animation attribute="rotation" begin="click"
    repeat="5" to="0 360 0"> </a-animation>

  <a-event name="mouseenter" scale="4 1 6">
</a-event>

</a-entity>
```

Con esto conseguimos que el objeto rote cuando hacemos click con el ratón en él.

Al estar en un entorno de realidad virtual, no disponemos de nuestro ratón y lo que funciona como tal es el centro de nuestra mirada, para conseguir apuntar al objeto creamos un cursor en el centro de la pantalla que no es otra cosa que el centro de la perspectiva que tenemos en cada momento. El cursor se crea una vez definimos la posición inicial de la cámara, en este caso le daremos un color rojo que resalte con el fondo.

```
[...]
```

```
<a-camera position="1.5 1 6.5">  
  
<a-cursor color="#FF0000">  
  
</a-camera>  
</a-entity>  
[...]
```

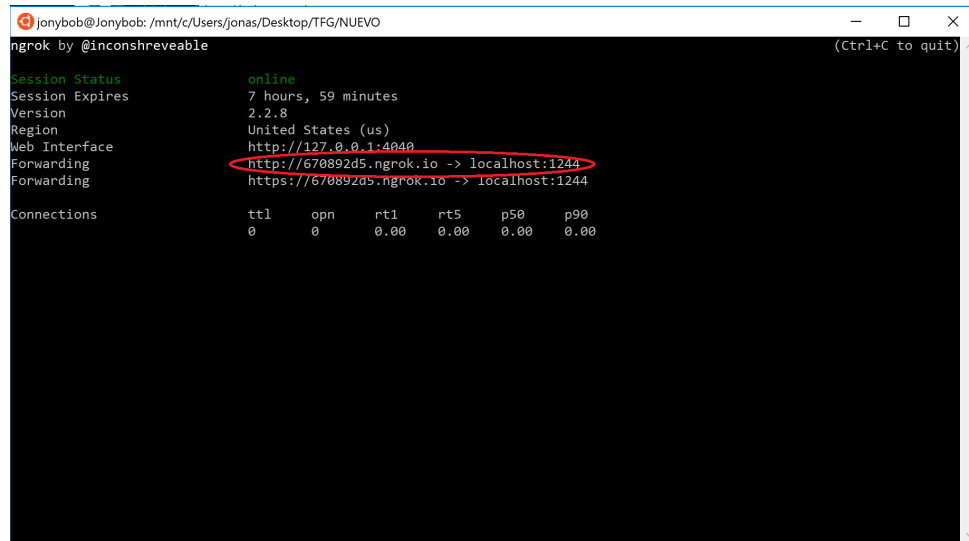
4.3.3. Anexo: Lleva tu experiencia VR a tu smartphone

Ya tenemos nuestra figura cargada en A-Frame de forma local. Pero como ya sabemos para visualizar el entorno de realidad virtual, necesitamos algo más y es un visor, la forma más rápida y sencilla de ver nuestra experiencia, es a través de nuestro smartphone, para exportar nuestro index.html a internet nos vamos a ayudar de una utilidad llamada ngrok.

- Para usar ngrok hacemos lo siguiente
 1. Abrimos una terminal y no vamos a nuestra carpeta raíz del proyecto, lanzamos nuestro servidor de forma local:

```
python -m SimpleHTTPServer 1234
```
 2. instalamos ngrok, simplemente descargando el .zip y descomprimiéndolo en nuestra carpeta raíz.
 3. después de esto lo ejecutamos pasándole como argumento el puerto de nuestro servidor.

```
./ngrok http 1234
```
 4. Ngrok nos devuelve una pantalla con la Url de nuestra aplicación.

A screenshot of a terminal window titled 'jonybob@Jonybob: /mnt/c/Users/jonas/Desktop/TFG/NUEVO'. The terminal shows the output of the 'ngrok by @inconshreveable' command. The status is 'online'. The session expires in 7 hours and 59 minutes. The version is 2.2.8. The region is United States (us). The web interface is at http://127.0.0.1:4040. The forwarding URL is http://670892d5.ngrok.io -> localhost:1244, which is circled in red. The connections table shows 0 connections with 0.00 ms latency.

```
jonybob@Jonybob: /mnt/c/Users/jonas/Desktop/TFG/NUEVO
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Session Expires     7 hours, 59 minutes
Version             2.2.8
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://670892d5.ngrok.io -> localhost:1244
                    https://670892d5.ngrok.io -> localhost:1244

Connections
  ttl    opn    rt1    rt5    p50    p90
    0     0     0.00  0.00  0.00  0.00
```

5. Introduciendo esta url en un navegador compatible como Google Chrome podemos visualizar nuestra experiencia en nuestro smartphone u otro dispositivo compatible.

4.4. Taller 3(Microbit como IU en entorno VR)

Capítulo 5

Conclusiones

5.1. Consecución de objetivos

Esta sección es la sección espejo de las dos primeras del capítulo de objetivos, donde se planteaba el objetivo general y se elaboraban los específicos.

Es aquí donde hay que debatir qué se ha conseguido y qué no. Cuando algo no se ha conseguido, se ha de justificar, en términos de qué problemas se han encontrado y qué medidas se han tomado para mitigar esos problemas.

5.2. Aplicación de lo aprendido

Aquí viene lo que has aprendido durante el Grado/Máster y que has aplicado en el TFG/TFM. Una buena idea es poner las asignaturas más relacionadas y comentar en un párrafo los conocimientos y habilidades puestos en práctica.

1. a

2. b

5.3. Lecciones aprendidas

Aquí viene lo que has aprendido en el Trabajo Fin de Grado/Máster.

1. a

2. b

5.4. Trabajos futuros

Ningún software se termina, así que aquí vienen ideas y funcionalidades que estaría bien tener implementadas en el futuro.

Es un apartado que sirve para dar ideas de cara a futuros TFGs/TFM.

Apéndice A

Manual de usuario

Bibliografía

- [1] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Inc., 1999.
- [2] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Inc., 1999.

Apéndice B

Introducción

No te olvides de echarle un ojo a la página con los cinco errores de escritura más frecuentes¹.

Aconsejo a todo el mundo que mire y se inspire en memorias pasadas. Las mías están todas almacenadas en mi web del GSyC².

B.1. Sección

Esto es una sección, que es una estructura menor que un capítulo.

Por cierto, a veces me comentáis que no os compila por las tildes. Eso es un problema de codificación. Cambiad de “UTF-8” a “ISO-Latin-1” (o viceversa) y funcionará.

B.1.1. Estilo

Recomiendo leer los consejos prácticos sobre LaTeX de Diomidos Spinellis³.

Sobre el uso de las comas⁴

A continuación, viene una figura, la Figura **??**. Observarás que el texto dentro del a referencia es el identificador de la figura (que se corresponden con el “label” dentro de la misma). También habrás tomado nota de cómo se ponen las “comillas dobles” para que se muestren

¹<http://www.tallerdeescritores.com/errores-de-escritura-frecuentes>

²<https://gsyc.urjc.es/~grex/pfcs/>

³<https://github.com/dspinellis/latex-advice>

⁴<http://narrativabreve.com/2015/02/opiniones-de-un-corrector-de-estilo-11-recetas-par>
html



Figura B.1: Página con enlaces a hilos

correctamente. Volviendo a las referencias, nota que al compilar, la primera vez se crea un diccionario con las referencias, y en la segunda compilación se “rellenan” estas referencias. Por eso hay que compilar dos veces.

B.2. Estructura de la memoria

En el capítulo ?? (ojo, otra referencia automática) se muestran los objetivos del proyecto.

Apéndice C

Estado del arte

Puedes citar libros, como el de Bonabeau et al. sobre procesos estigmérgicos [?].

C.1. Sección 1

Hemos hablado de cómo incluir figuras. Pero no hemos dicho nada de tablas. A mí me gustan las tablas. Mucho. Aquí un ejemplo de tabla, la Tabla ??.

Si tu proyecto es un software, siempre es bueno poner la arquitectura (que es cómo se estructura tu programa a “vista de pájaro”).

Por ejemplo, puedes verlo en la figura ??.

Si utilizas una base de datos, no te olvides de incluir también un diagrama de entidad-relación.

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

Cuadro C.1: Ejemplo de tabla

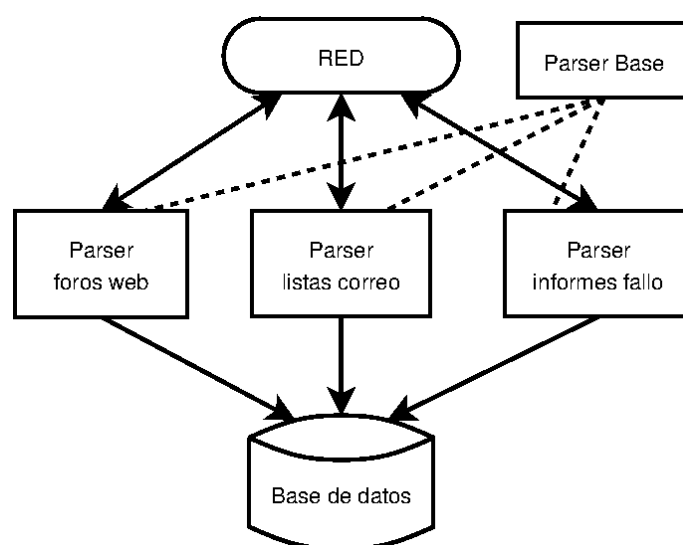


Figura C.1: Estructura del parser básico