



Universidad de Valladolid



ESCUELA DE INGENIERÍAS
INDUSTRIALES

Máster en Electrónica Industrial y Automática

**MASTER EN ELECTRÓNICA INDUSTRIAL Y
AUTOMÁTICA**
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

TRABAJO FIN DE MÁSTER

**Estudio y aplicación del robot Pepper para la interacción con
personas mayores**

Autor: D. Luis Cobo Hurtado
Tutor: D. Eduardo Zalama Casanova

Valladolid, febrero 2018



Universidad de Valladolid



ESCUELA DE INGENIERÍAS
INDUSTRIALES

Máster en Electrónica Industrial y Automática

**MASTER EN ELECTRÓNICA INDUSTRIAL Y
AUTOMÁTICA**
ESCUELA DE INGENIERÍAS INDUSTRIALES
UNIVERSIDAD DE VALLADOLID

TRABAJO FIN DE MÁSTER

**Estudio y aplicación del robot Pepper para la interacción con
personas mayores**

Autor: D. Luis Cobo Hurtado
Tutor: D. Eduardo Zalama Casanova

Valladolid, febrero 2018

Resumen

Este trabajo se centra en el manejo, programación y validación de un robot social. Concretamente, en el análisis de las posibilidades que ofrece el robot Pepper y su aplicación para interactuar con personas mayores, de forma que las asista y ayude a trabajar sus capacidades físicas y cognitivas interactuando socialmente, contribuyendo así a mejorar su calidad de vida.

Se han desarrollado diferentes funcionalidades que el robot puede ofrecer al usuario, entre las que se incluyen: conversación, interacción por voz y mediante interfaz táctil, obtención de información, juegos, ejercicios y música.

El robot ha sido probado tanto en el laboratorio como en una residencia real, demostrando que los robots sociales asistenciales son una herramienta muy útil y con mucho potencial para interactuar con personas mayores, consiguiendo que los residentes estuvieran entretenidos y muy motivados al interactuar con el robot, apreciando especialmente los juegos y ejercicios.

Palabras clave

Robótica social, robot asistencial, Pepper, personas mayores, interacción humano-robot

Abstract

This work focuses on handling, programming and testing a social robot. Specifically, it's focused on studying the possibilities offered by Pepper robot in order to use it to interact with old people, so the robot can assist and help them to work on their physical and mental abilities, contributing to improve their quality of life.

Different features have been developed for the robot, such as conversation, voice and tactile interaction, offering information, games, exercises and music.

The robot has been tested both on the lab and in a real care facility for old people, showing that social robots are a useful tool with a lot of potential to interact with old people. The participants were motivated and enjoyed their time interacting with the robot, specially appreciating the games and exercises.

Keywords

Social robotics, assistive robot, Pepper, old people, human-robot interaction

ÍNDICE

1	Introducción y objetivos.....	1
1.1	Introducción y justificación	1
1.2	Objetivos.....	1
1.3	Estructura del documento	2
2	Robótica Social	5
2.1	Robots sociales	5
2.2	Aplicación de la robótica social a personas mayores	8
2.3	Aplicaciones e Investigaciones con el robot Pepper	14
3	Robot Pepper y herramientas de desarrollo.....	23
3.1	Robot Pepper.....	23
3.2	Programa Coreographe	26
3.3	Funcionamiento y APIs del robot	31
3.4	SDK JavaScript.....	37
4	Desarrollo de las funcionalidades	39
4.1	Conversación general, interacción por voz con datos del usuario y seguimiento	40
4.2	Interacción a través de la Tablet	42
4.3	Información	45
4.4	Juegos	46
4.5	Ejercicios.....	48
4.6	Música.....	50
4.7	Registros y valoración.....	51
5	Pruebas realizadas.....	55
5.1	Estructura de las pruebas	55
5.2	Resultados obtenidos	57
6	Conclusiones.....	63
7	Bibliografía	65
Anexo A.	Guía de usuario del Robot Pepper	69
Anexo B.	Modelo de Cuestionario Empleado	75
Anexo C.	Esquemas y código de los programas del robot	79
	Programa General	79
	Inicialización	79
	Manejo Log.....	80
	Diálogo General.....	82

Conceptos Generales	92
Conceptos Personalizados.....	95
Carga Datos Usuario	96
Guarda Datos Usuario	99
Carga Fecha y Hora.....	101
Pantalla Inicial	102
Seguimiento	103
Tiempo Actual	104
Noticias.....	106
Wikipedia.....	108
Control música, juegos, mapas	111
Decir Frase.....	112
Muestra menú.....	113
Muestra página	113
Muestra actividad.....	114
Muestra app	115
Crea usuario temporal.....	115
Borrar usuario	116
Guarda foto	117
Captura vídeo	119
Termina actividad.....	121
Programa Ejercicios	123
Ejercicios Cabeza	123
Ejercicios Hombros.....	123
Ejercicios Manos.....	123
Contador Repeticiones	124
Menús Tablet.....	124
Menú principal	124
Script Pepper	126
Actividad en Curso.....	128
Ajustes	129
Ajustes usuarios	130
Ajustes volumen	136
Selección usuario.....	137
Crea usuario	143

Información	145
Noticias.....	146
Juegos.....	147
Adivinanzas.....	148
Ejercicios.....	159
Repeticiones ejercicios.....	160
Música	161
Música clásica.....	162
Música copla.....	164
Música popular.....	166
Música moderna.....	168
Música pop	170
Música rock	172
Música heavy.....	175
Valoración.....	177

1 INTRODUCCIÓN Y OBJETIVOS

1.1 INTRODUCCIÓN Y JUSTIFICACIÓN

Este trabajo se enmarca en el contexto de la realización del trabajo fin de máster correspondiente a la titulación de Máster en Ingeniería Electrónica Industrial y Automática de la Universidad de Valladolid. Corresponde al departamento de Ingeniería de Sistemas y Automática, siendo supervisado por profesores titulares del mismo.

Se centra en el manejo, programación y validación de robots sociales, que son aquellos con capacidades de interactuar con las personas. Concretamente, en el análisis de las posibilidades que ofrece el robot Pepper y su aplicación para interactuar con personas mayores.

A medida que las personas van envejeciendo pueden ir apareciendo disfuncionalidades físicas o mentales o situaciones de aislamiento social que afectan a su capacidad de realizar actividades de forma autosuficiente. La utilización de robots sociales para asistir a estas personas puede ayudar a mantener o mejorar sus capacidades interactuando socialmente y realizando actividades para trabajar física y/o mentalmente, contribuyendo a mejorar su calidad de vida.

1.2 OBJETIVOS

El objetivo principal del proyecto es desarrollar y validar diferentes funcionalidades para el robot Pepper que permitan utilizarlo como un robot asistencial para personas mayores, de forma que sea capaz de interactuar socialmente con ellos, entretenerlos y mantenerlos activos física y mentalmente.

El empleo de robots sociales para el ámbito asistencial es un campo que se encuentra actualmente en investigación, por lo que los aportes de este proyecto, con la utilización del robot Pepper para este fin, pueden suponer un aporte útil y novedoso en este campo.

Para conseguir que el proyecto sea exitoso hay que tener en cuenta una serie de limitaciones y dificultades que hay que vencer. En primer lugar hay que tener en cuenta las limitaciones actuales de los robots sociales y del robot Pepper en particular. Entre ellas se encuentran:

- Dificultad de tener un reconocimiento facial que sea totalmente fiable: la tecnología de reconocimiento facial implementada en el robot puede fallar a la hora de reconocer a diferentes usuarios que tengan una apariencia similar o cuando la iluminación de la sala no sea la apropiada.
- Existencia de ruido en el ambiente que dificulte el reconocimiento de voz: en este caso el robot incorpora tres micrófonos direccionales que ayudan a mitigar parcialmente los efectos del ruido ambiente.
- Diálogos pre-programados: este robot, al igual que gran parte de los robots sociales existentes actualmente, no posee una “inteligencia” capaz de generar sus propias respuestas al conversar con una persona. Por tanto, es necesario programar previamente todas las preguntas y respuestas posibles sobre las que el robot hablará.

- Carencia de *speech to text* (conversión de voz del usuario a texto): el robot Pepper no posee esta capacidad para el idioma Español a diferencia de otros idiomas como el inglés o el japonés, reconociendo sólo las frases especificadas, lo que limita las posibilidades del diálogo.

Por otra parte, hay que tener en cuenta también las peculiaridades asociadas a la interacción con personas mayores:

- Dificultades en la visión: con la edad la vista va empeorando, por lo que la interfaz de usuario tendrá que tener botones y letras grandes de forma que sean legibles de forma más sencilla.
- Pérdida de capacidad auditiva: es un problema muy común en las personas mayores, y por ello el robot deberá hablar de forma clara y con un volumen y tono apropiados.
- Brecha digital: los adultos y las nuevas generaciones están acostumbradas a utilizar dispositivos electrónicos en su día a día. Sin embargo, gran parte de las personas mayores nunca los han utilizado o lo hacen de forma muy limitada. Por ello la interfaz y la interacción con el robot tiene que ser lo más sencilla, natural e intuitiva como sea posible para que las personas mayores puedan interactuar con él sin problemas.

Con todo ello, para lograr el objetivo principal se deberán cumplir los siguientes sub-objetivos:

- Estudiar el funcionamiento, las posibilidades y las herramientas de desarrollo que ofrece el robot Pepper.
- Diseñar y programar diferentes funcionalidades para el robot que permitan que sea capaz de interactuar con personas mayores.
- Explicar las funcionalidades de forma detallada para permitir en el futuro continuar mejorando y ampliando las capacidades del robot.
- Realizar pruebas con diferentes personas y estudiar los resultados obtenidos para valorar el funcionamiento y la utilidad del robot.

1.3 ESTRUCTURA DEL DOCUMENTO

La memoria de este trabajo está dividida en varios capítulos en los que se van explicando diferentes aspectos:

- **Introducción y Objetivos:** en este apartado se realiza una introducción con el contexto y justificación del trabajo y sus objetivos.
- **Robótica Social:** se muestra una visión general acerca de los robots sociales y a continuación se ven los trabajos o estudios existentes relacionados con la aplicación de la robótica social a personas mayores.
- **Robot Pepper y herramientas de desarrollo:** en este capítulo se describen las características del robot Pepper y las herramientas que ofrece para el desarrollo, como son el programa Coreographe, las APIs del robot y el SDK de Javascript.
- **Desarrollo de las funcionalidades:** se explica la estructura y el funcionamiento de las diferentes funcionalidades que se han desarrollado en el robot: conversación e interacción, seguimiento, uso de la tablet integrada, obtención de información, juegos, ejercicios, música y obtención de registros y valoración del usuario.

- **Pruebas realizadas:** se ve la estructura de las pruebas que se han realizado a los usuarios y se muestran los resultados que se han obtenido.
- **Conclusiones:** se exponen las conclusiones finales a las que se ha llegado al realizar el trabajo.
- **Bibliografía:** se muestra la bibliografía utilizada en la elaboración del trabajo.
- **Anexos:** en los anexos se han incluido la guía de usuario del robot, el cuestionario empleado en las pruebas y los esquemas y códigos de los programas realizados.

2 ROBÓTICA SOCIAL

2.1 ROBOTS SOCIALES

Los robots sociales son aquellos que son capaces de interactuar y comunicarse con las personas por medio del lenguaje, comportamientos, patrones y normas sociales en diferentes entornos, ya sea en espacios públicos como escuelas o hospitales, centros de trabajo o en los hogares.

Aunque a menudo se dice que los robots en general poseen ciertas habilidades sociales, la robótica social es una rama que ha surgido recientemente. Desde comienzos de los años 90 los investigadores en inteligencia artificial y robótica han ido desarrollando robots enfocados en interactuar socialmente [1].

El diseño de un robot social es un desafío especial ya que el robot debe ser capaz de interpretar correctamente las acciones llevadas a cabo por la persona con la que está interactuando y responder de forma apropiada.

Es importante que un robot social sea capaz de mostrar o simular que posee personalidad y emociones para que la persona con la que interactúa pueda empatizar con él. También es importante que el robot tenga una apariencia humanoide y al mismo tiempo amigable, realizando movimientos naturales similares a los humanos y siguiendo los comportamientos socialmente y culturalmente aceptados.

Los robots sociales pueden emplearse para diferentes tareas, desde tareas simples de apoyo a conversaciones e interacciones complejas o tareas colaborativas.

Debido a hasta el momento no es posible interpretar completamente las acciones de las personas en ocasiones se utiliza el experimento del Mago de Oz. Consiste en que el robot social es controlado, parcial o totalmente, por una persona real de forma externa, haciendo creer al usuario que el robot opera de forma autónoma. Este experimento es útil para conocer la forma de reaccionar de las personas ante un robot social con un comportamiento inteligente.

En la actualidad existen diferentes robots sociales, bien comerciales o prototipos, como Nao, Pepper (Aldebaran), Asimo (Honda) o iCub (Instituto italiano de tecnología).

Robot Asimo

Asimo es un robot humanoide creado por Honda en el año 2000. Sus siglas en inglés significan “Advanced Step in Innovative Mobility”, que se pueden traducir como “salto avanzado en movilidad innovadora” [2].

Desde los años 80 Honda ha ido desarrollando diferentes prototipos de robots capaces de caminar hasta lanzar el robot Asimo. Asimismo, existen varias versiones del robot Asimo, siendo la más reciente del año 2011.

La última versión del robot Asimo presume de haber pasado de ser una máquina automática a una máquina autónoma, con la capacidad de tomar decisiones para adaptar su comportamiento al entorno que le rodea, incluyendo las personas. Es capaz de mantener el Estudio y aplicación del robot Pepper para la interacción con personas mayores

equilibrio de forma avanzada, correr, o reconocer los movimientos de personas alrededor de él a partir de múltiples sensores y estimar los cambios que ocurren en su entorno.

Tiene una altura de 130 cm y pesa 48 kg, siendo capaz de desplazarse a una velocidad máxima de 9 km/h [3]. Su aspecto visual es el mostrado en la **¡Error! No se encuentra el rigen de la referencia..**



Figura 1. Versión más reciente del robot Asimo

Es capaz de reconocer la voz, incluso cuando hay varias personas hablando a la vez, reconocer los rostros y conversar en diferentes idiomas.

Sus manos de cinco dedos incluyen sensores táctiles y de fuerza que permiten controlar cada dedo de forma independiente. Esto permite que, unido a sus capacidades de reconocimiento visual de objetos, pueda realizar tareas con destreza y manipular objetos.

Su capacidad para generar un comportamiento autónomo le permite realizar predicciones a partir de la información del entorno y determinar sus próximas acciones de forma acorde, sin ser controlado por un operador externo.

Todas estas capacidades permiten a Asimo, aunque no es todavía un robot totalmente autónomo ni capaz de socializar como un humano real, estar un paso más cerca de tener un uso práctico en un entorno real en el que pueda coexistir con las personas.

Robot iCub

iCub es un robot humanoide de un metro de altura desarrollado por el IIT (Istituto Italiano de Tecnología) como parte del proyecto RobotCub, financiado por la unión europea [4]. Es de código abierto y está enfocado a la investigación sobre la cognición humana y la inteligencia artificial [5].

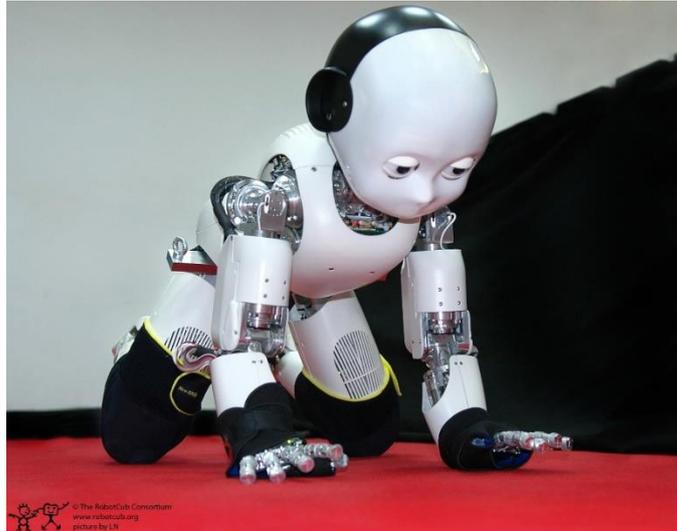


Figura 2. Robot iCub gateando

El robot tiene un aspecto similar a un bebé, como se puede ver en la Figura 2. Es capaz de realizar diferentes tareas, entre las que se encuentran: gatear siguiendo una guía visual, resolver laberintos, aprender tiro con arco, expresar emociones faciales, ejercer fuerza controlada gracias a sus sensores, agarrar objetos o evitar obstáculos móviles [6].

Robot NAO

NAO es un robot humanoide con una altura de 58 cm desarrollado por la compañía francesa Aldebaran Robotics, absorbida después por el grupo Softbank [7]. Ha recibido cinco evoluciones desde que fue lanzado en el año 2006, vendiendo alrededor de 10.000 unidades alrededor del mundo [8].

Desde 2007 es el robot utilizado en la RoboCup, una competición de fútbol para robots en la que compiten equipos de todo el mundo.

Su apariencia visual es la mostrada en la Figura 3. Contiene diversos sensores, motores y software controlado por un sistema operativo propio llamado NAOqi OS.



Figura 3. Robot NAO

Posee 25 grados de libertad y gracias a su forma humanoide puede moverse y adaptarse al entorno a su alrededor. Además, contiene sensores para mantener el equilibrio y conocer si está de pie o acostado. También incluye cuatro micrófonos direccionales y altavoces, con los que puede interactuar con los humanos escuchando y hablando.

Gracias a sus dos cámaras de alta definición puede ver su entorno y detectar formas y objetos. Posee también módulos WiFi y Ethernet para acceder a internet [9].

Se emplea tanto para un uso comercial en empresas como para uso educativo en escuelas y universidades o en investigación, gracias a que ofrece flexibilidad para ser programado de manera visual o empleando lenguajes más avanzados como C++, Python o Java.

NAO no es un robot inteligente capaz de tener una conducta autónoma, pero al ser programable puede reproducir diferentes comportamientos humanos. Se puede considerar como el primer robot social en distribuirse al gran público, siendo sucedido después por el robot Pepper.

2.2 APLICACIÓN DE LA ROBÓTICA SOCIAL A PERSONAS MAYORES

El envejecimiento de la población, especialmente en occidente, hace necesario mejorar la dedicación al cuidado de las personas mayores. Debido al aumento de la población en edad avanzada (para el año 2050 se estima que la población por encima de 85 años se va a multiplicar por cinco), poco a poco se va a ir haciendo insuficiente el personal especializado en el cuidado de estas personas. Esto hace necesaria la aparición de nuevas tecnologías que puedan ayudar en el ámbito asistencial, facilitando la vida independiente de las personas mayores. La robótica social puede suponer un elemento muy importante, mediante el empleo de robots que sean capaces de interactuar con ellos y ayudar a que se mantengan activos tanto física como mentalmente, contribuyendo así a evitar su aislamiento social, la inactividad y el sedentarismo.

El robot social debe ser capaz de interactuar y empatizar con la persona mediante conversación, expresiones faciales y gestos comunicativos, con el fin de poder asistir a la persona según sus necesidades particulares.

Se han realizado diferentes investigaciones sobre sistemas para pacientes que han sufrido una apoplejía o ataque cerebrovascular, pacientes con Alzheimer, así como ancianos sanos.

Se pueden distinguir dos grupos de robots para el cuidado de los mayores según el enfoque de sus tareas. El primer grupo es el de los robots de servicio, dedicados a ayudar a mantener una vida independiente, apoyando a la hora de hacer actividades del día a día como puede ser comer, vestirse o ir al servicio, la movilidad y desplazamientos, mantenimiento del hogar o monitorizar en aquellos casos que necesiten una atención continuada.

En este primer grupo existen robots como el robot enfermera Pearl, el holandés iCat, el alemán Care-o-bot o el proyecto Robocare. Las investigaciones en este caso se suelen enfocar en cómo las características sociales del robot facilitan el uso de sus diferentes funciones, así como que capacidades de interacción social son necesarias para facilitar la aceptación del robot por las personas.

El segundo grupo está formado por robots que más que en realizar un servicio se enfocan en servir como compañía a la persona, tratando de establecer un vínculo emocional que mejore su estado de ánimo, le entretenga o reduzca su sensación de soledad. Entre ellos podemos distinguir al robot Paro, el oso de peluche robot "Huggable" (abrazable) o el perro robot de Sony Aibo.

Esta división en dos grupos no es estricta, ya que algunos de los robots de servicio también pueden servir como compañía, así como algunos de los robots de compañía pueden hacer algunas tareas de servicio si se programan para ello.

Robot Paro

Paro es una foca robot desarrollada por el AIST, instituto nacional de ciencia y tecnología industrial avanzada de Japón, y conocida en España con el nombre Nuka. Su desarrollo comenzó en el año 1993, y empezó a comercializarse desde el año 2003, encontrándose actualmente en su octava generación.

Tiene una apariencia amigable, tal y como se puede ver en la Figura 4, que busca tener un efecto relajante y provocar una respuesta emocional positiva, así como contribuir a la interacción social tanto entre pacientes y sus cuidadores como entre los propios pacientes.



Figura 4. Robot Paro

Posee diferentes sensores táctiles, de iluminación, temperatura, postura y altavoces con los que percibir a las personas y a su entorno. Puede reconocer la luz y oscuridad, caricias, la dirección de las voces y palabras como su nombre, saludos o elogios.

Es capaz de “aprender”, adaptando su comportamiento a las preferencias del usuario. Se comporta como si fuera un animal vivo, moviendo su cabeza o sus piernas, haciendo sonidos o imitando la voz de una foca bebé real.

Por sus características se utiliza con pacientes en hospitales, centros de rehabilitación o centros para mayores donde no se puedan emplear mascotas reales [10].

Robot Pearl

Pearl es un robot “enfermera” desarrollado por las universidades de Michigan, Pittsburgh y Carnegie Mellon. Tiene dos funciones principales: recordar a las personas sus actividades rutinarias como comer, beber, tomar las medicinas o ir al baño y guiarles por el entorno.

El objetivo inicial del proyecto era desarrollar un asistente robot para personas ancianas viviendo en sus casas, especialmente aquellos con ligeros problemas cognitivos. No obstante, los objetivos se ampliaron para enfocarse también en pacientes de centros para mayores, ya sea residencias o centros de tratamiento específicos.

El aspecto del robot es el mostrado en la Figura 5. Está equipado con un control de movimiento diferencial, dos PC Pentium, Ethernet inalámbrico, sensores laser SICK, sónares, micrófonos para reconocimiento de voz, altavoces, pantalla táctil, cabeza móvil y cámaras estéreo.

En cuanto al software, posee un sistema de navegación autónoma, reconocimiento de voz, sintetizado de voz y captura rápida de imagen para hacer streaming de vídeo, reconocimiento facial y seguimiento.



Figura 5. Robot Pearl

Su primera función principal es recordar a los ancianos sus actividades diarias. Se busca que el robot sepa que recordatorios hacer y cuando, comprobar si el usuario ya va a hacer las actividades necesarias, evitar molestar al usuario y evitar también que el usuario se vuelva demasiado dependiente del sistema. Para ello se utiliza un software que almacena el plan de actividades diarias del usuario y se encarga de actualizarlo e identificar cualquier conflicto potencial que pueda haber. También se toma información sobre las actividades observables del usuario para monitorizar la ejecución del plan. En el caso de que haya disparidad entre las acciones que debe realizar el usuario según el plan y lo que esté haciendo en realidad toma decisiones sobre cuando hacer los recordatorios.

Para iniciar el sistema el cuidador del anciano introduce una descripción de las actividades diarias, así como las preferencias de tiempo y conducta. Este plan puede ser modificado por el cliente o el cuidador (añadiendo, modificando o eliminando actividades), la realización de las actividades o el paso del tiempo.

Los recordatorios serán dados según la prioridad de las actividades y la certeza de que el usuario las vaya a realizar a tiempo, así como las preferencias dadas por el cuidador o el paciente.

Su segunda función principal es la de ayuda a los ancianos a moverse por el entorno. Esto es especialmente útil en centros para mayores con asistencia, donde las enfermeras dedican una gran cantidad de tiempo a escoltar a los ancianos de un lugar a otro. Puede emplearse bien para ir al comedor, al médico, fisioterapeuta, peluquero, puntos de reunión, al cine, o simplemente dar un paseo.

El robot estima continuamente la posición de la persona, además de variar la velocidad para adaptarse a su ritmo. Utiliza un sistema de navegación para guiar al usuario y es capaz de evitar los obstáculos.

El prototipo del robot ya ha sido probado en una residencia real, y los investigadores esperan llevar a cabo más experimentos en el futuro para comprobar el funcionamiento y la fiabilidad del proyecto [11].

Robot HOBBIT

El robot HOBBIT es parte de una investigación para probar en un entorno real la utilización de un robot de servicio para personas mayores. Fue distribuido en 18 residencias para personas de 75 años o más por un total de 371 días.

El robot tiene la apariencia mostrada en la Figura 6 y una altura máxima de 125 cm. Está equipado con un sistema de movimiento diferencial, una cámara 3D, una cámara RGB, una pantalla táctil y un brazo terminado con una pinza. Tiene además un sistema de navegación autónoma, reconocimiento de voz, sintetizado de voz y reconocimiento gestual.



Figura 6. Robot HOBBIT

Ofrece entretenimiento a través de la radio, música, audiolibros, juegos, servicios web y ejercicios de fitness. Además, incluye recordatorios, teléfono, control de un manipulador y llamadas de emergencia. Es capaz de detectar emergencias como una persona en el suelo, así como coger objetos enseñados por el usuario.

Los resultados del estudio mostraron que el robot cumplió con las expectativas de los usuarios. No obstante, el robot fue visto más como un juguete que como una ayuda para su día a día. Los usuarios señalaron que una versión final del robot lista para comercializarse sería muy útil para ayudar a las personas más frágiles y socialmente aisladas [12].

Robot Brian 2.1

El robot Brian 2.1 forma parte de un estudio para diseñar robots sociales asistenciales capaces de proporcionar asistencia cognitiva y motivación a ancianos para promover la participación en actividades autosuficientes del día a día. Es un robot humanoide que trata de motivar socialmente a que los ancianos coman la comida.

El robot incluye un sistema para hacer un seguimiento de la cantidad de comida que ingiere el usuario y las acciones que realiza al comer y beber.

Para añadir interacción social a la comida, Brian 2.1 se comunica de forma natural tanto de forma verbal como no verbal, utilizando gestos, expresiones faciales o variando la entonación de la voz. Es capaz de saludar a la persona por su nombre, invitarla a sentarse y hacer bromas o comentarios positivos relacionados con la comida. En la Figura 7 se puede ver al robot con uno de los participantes durante una de las comidas.



Figura 7. Robot Brian 2.1 motivando a comer a un usuario

Tiene una forma humanoide desde la cintura hacia arriba, la cual tiene dos grados de libertad, pudiendo girarse hacia un lado u otro e inclinarse hacia delante o hacia atrás. Los brazos tienen cuatro grados de libertad cada uno: dos en el hombro, uno en el codo y uno en la muñeca, que le permiten señalar diferentes elementos de la bandeja de la comida. En el cuello tiene tres grados de libertad que le permiten girar e inclinar la cabeza. También es capaz de mover el rostro, permitiéndole mostrar diferentes expresiones faciales como felicidad, risa, neutralidad o tristeza. Al comunicarse verbalmente también es capaz de entonar la voz, variando el tono y la velocidad, en concordancia con las expresiones faciales.

Para monitorizar el consumo de la comida el robot se emplean por una parte sensores de presión en la bandeja de la comida y sensores infrarrojos en los cubiertos. El robot, por otro lado, utiliza un sensor Kinect y dos controles de Wii. Todo este conjunto permite controlar la actividad sin que el usuario tenga que equipar ningún sensor.

Se realizó una experimentación en un centro para mayores con el robot y seis ancianos de entre 82 y 93 años para investigar el compromiso y el cumplimiento de los usuarios en sus interacciones con el robot durante la comida, así como su aceptación y actitud hacia el robot. Los resultados del estudio mostraron que los ancianos estaban comprometidos y cumplían las acciones durante las comidas con el robot. Un cuestionario realizado después del estudio mostró que, en general, los participantes se divirtieron interactuando con el robot y que tenían actitudes positivas hacia el robot durante la actividad.

Los prometedores resultados obtenidos en el estudio han motivado a los investigadores a la realización de nuevos estudios a largo plazo en el futuro, con el fin de comparar las diferencias en las reacciones al interactuar con un robot o un cuidador humano, así como estudiar la forma de implementar a largo plazo robots de este tipo en centros para mayores garantizando la seguridad de los usuarios [13].

Residencia Shintomi (Tokio)

La residencia Shintomi de Tokio aloja a medio centenar de personas mayores además de treinta ancianos que acuden a su centro de día. Cuenta con un importante apoyo por parte del gobierno nipón, que a través del ministerio de bienestar social subvenciona gran parte de su presupuesto. Se utiliza como un banco de pruebas para estudiar la utilización de robots sociales para el cuidado de los mayores.



Figura 8. Robot Sota en la residencia Shintomi

En las mesillas de noche de cada habitación se encuentra el robot Sota, mostrado en la Figura 8, que se encarga de darles los buenos días, conversar con ellos de forma sencilla, o recordarles que se tomen la medicación.

Por su parte, el robot Pepper ofrece juegos y ejercicios para entretener a los mayores, como se puede ver en la Figura 9, además de cantar y guiar para que le sigan.



Figura 9. Robot Pepper dirigiendo ejercicios en la residencia Shintomi

También poseen varios robots Palro, Aibo y Paro que entretienen y hacen compañía a los mayores, como se muestra en la Figura 10.



Figura 10. Robots Aibo y Paro en la residencia Shintomi

Además de robots sociales en la residencia Shintomi también se prueban diferentes tipos de tecnologías para asistir a los mayores, como sensores bajo los colchones de las camas para monitorizar las constantes vitales de los ancianos, una grúa para ayudar a levantarse a los ancianos con mayores dificultades para moverse, o la cama Reysone de Panasonic, que se divide por la mitad, convirtiéndose en un sillón en el que pasar el día [14].

2.3 APLICACIONES E INVESTIGACIONES CON EL ROBOT PEPPER

El robot Pepper, con su forma humanoide y sus capacidades de interacción social, ofrece muchas posibilidades de uso. En el punto anterior ya se ha visto el empleo de Pepper en una residencia de Tokio para entretener a los mayores con juegos y ejercicios o conversando. A continuación se van a ver otras aplicaciones e investigaciones en las que se ha utilizado al robot Pepper.

Uso en comercios y establecimientos

Uno de los principales usos que se le está dando al robot Pepper es utilizarlo en comercios y establecimientos para comunicarse con los clientes.

En el caso de Japón, más de 140 tiendas de SoftBank Mobile están empleando a Pepper para saludar, informar y entretener a los clientes. Un gran banco también ha elegido a Pepper para informar a sus clientes en diferentes idiomas en sus sucursales en Japón. Además, la empresa Nestlé planea utilizar a Pepper en más de 1000 tiendas de Nescafé en el país nipón para informar a sus clientes sobre los productos que ofrecen [15].

En el caso de Estados Unidos, donde se está realizando un programa piloto, Softbank asegura que las tiendas en las que lo están adoptando está ayudando a mejorar sus ventas frente al aumento del comercio online. En un comercio minorista dedicado a la tecnología en Palo Alto donde se empleó al robot durante una semana, Softbank asegura que se produjo un aumento del 70% en las personas que visitan el establecimiento, y la tienda informó de un incremento del 13% en sus ingresos. Por otra parte, en una tienda de ropa ubicada en el campus de una universidad en California Softbank asegura que en los días que estuvo el robot se triplicaron los ingresos y la afluencia de clientes aumento en un 20% [16].

Softbank también ha firmado un acuerdo con Master Card por el cual se instalarán robots Pepper en los establecimientos de Pizza Hut en Asia. El robot permitirá a los clientes hacer

su pedido de comida y bebida sin tener que pasar por el mostrador. La empresa asegura que busca aumentar la eficiencia y no provocará una reducción de los puestos de trabajo [17].

Por su parte, en Bélgica se está empezando a utilizar al robot Pepper para saludar a los pacientes en la recepción de los hospitales. Es el caso de los hospitales en Ostend y Liege, donde el robot recibe a los clientes y es capaz de acompañarlos al departamento que necesiten [18].

En el caso de España, una cadena hotelera ha comenzado a utilizar al robot Pepper con el objetivo de dar información y entretener a los clientes, hacer de recepcionista o incluso guiar a los clientes hasta su habitación [19]. También la cadena de supermercados Carrefour está haciendo una gira con robots Pepper por sus distintos establecimientos en España con tres funciones diferentes: el primero saluda a los clientes y les ofrece información sobre promociones o descuentos, el segundo entretiene a los niños con juegos y bailes y el tercero ofrece información sobre el surtido de vinos de la bodega [20].



Figura 11. Robot Pepper en una tienda de Edimburgo

Sin embargo, la utilización de Pepper en este tipo de aplicaciones no es tan sencilla como pueda parecer. Es el caso de una tienda de Edimburgo, mostrada en la Figura 11, donde tras un tiempo utilizando al robot se decidió dejar de utilizarlo al no cumplir con las expectativas esperadas. Al tratarlo de utilizar para guiar a los clientes por la tienda el robot tenía problemas para circular por la tienda y el ruido de ambiente del establecimiento provocaba que el reconocimiento de voz fallase en muchas ocasiones. Por otro lado, al tratar de utilizarlo para ofrecer productos a los clientes los clientes comenzaron a evitarlo, mientras que otros dependientes humanos si conseguían que los clientes probaran los productos [21]. Es un ejemplo de que los robots no son aún capaces de interactuar con su entorno como lo haría un ser humano real y que es necesario tener en cuenta estas limitaciones a la hora de tratar de aplicar un robot social en un entorno real para conseguir un resultado exitoso.

El tiempo revelará la factibilidad de la utilización del robot Pepper en comercios y establecimientos a medio y largo plazo, ya que es posible que en un principio los clientes se sientan atraídos por el robot al ser algo novedoso y con ello aumente la afluencia en el establecimiento. Sin embargo, con el paso del tiempo puede perder ese atractivo y que los clientes se sientan frustrados al interactuar con él al no tratarse de una persona real que pueda resolver determinados problemas o realizar algunas acciones concretas, prefiriendo entonces interactuar con un humano. Esto es debido que se trata de un robot con un

comportamiento limitado, que todavía no es verdaderamente inteligente o capaz de relacionarse con su entorno, aprender o reaccionar como lo haría un ser humano real.

Uso educacional con niños

Un equipo japonés de investigadores ha desarrollado una aplicación para un uso educacional del robot Pepper. Para ello han aplicado los conceptos de *care-receiving robot* (CRR), robots que reciben cuidado, atención o ayuda de la gente a su alrededor, y *total physical response* (TPR), un método para enseñanza de lenguas combinando el habla con la acción y movimientos físicos, para desarrollar una aplicación educativa con Pepper.

El escenario propuesto consiste en que los niños aprenden junto a Pepper en el entorno de su hogar por medio de un profesor humano que da la clase de forma remota desde un aula. De esta forma, el objetivo es que los niños japoneses aprenden a hablar inglés “a la vez” que el robot mientras se divierten con él. La clase dada por el profesor real se muestra en la pantalla del robot, bien siendo un video grabado anteriormente o una transmisión en tiempo real, mientras que el robot mira hacia la pantalla como si estuviese también aprendiendo.

Se desarrollaron tres programas como parte de la aplicación: *Color*, *Let's Try* y *Body*. Como se ve en la Figura 12, el programa deseado se selecciona a través de la pantalla táctil.



Figura 12. Niños eligiendo el programa educativo a través de la pantalla

La aplicación *Color* consiste en que los niños aprendan los nombres de los colores en inglés. Para ello, por ejemplo, el robot indica que quiere enseñarles el color rojo y le pide si pueden mostrarle algo que haya en la habitación de ese color. Entonces el niño trae el objeto al robot y este pregunta al profesor de la pantalla como se dice ese color. Entonces el profesor responde que es el color rojo, y tanto el robot como los niños lo repiten. De esta forma los niños aprenden de forma dinámica en vez de simplemente atender a una lección.

La aplicación *Let's Try* consiste en enseñar palabras a los niños utilizando el lenguaje corporal. Por ejemplo, el robot muestra la imagen de un avión y pregunta a los niños como se dice “avión” en inglés. Entonces el profesor de la pantalla les dice que vuelen como un avión y tanto el robot como los niños gesticulan imitando el movimiento tal y como se muestra en la Figura 13, aprendiendo así de forma interactiva.



Figura 13. Pepper enseñando la palabra "Avión" mediante lenguaje corporal

La aplicación *Body* consiste en enseñar las partes del cuerpo. Para ello el profesor, mostrado en la pantalla, va enseñando las partes del cuerpo utilizando a un robot Pepper. Así, por ejemplo el profesor enseña la palabra “boca” y mueve la mano del robot hacia su boca. Entonces el Pepper que está junto a los niños les pide que le enseñen como al robot de la pantalla, y entonces los niños cogen la mano de Pepper y señalan su boca con ella.

Para mantener a los niños motivados el robot también les pide de vez en cuando que choquen la mano con el robot así como hacerles una foto como premio.

Entre las conclusiones obtenidas de la experimentación se vio que la mayor preocupación de los padres ante la posibilidad de tener un robot Pepper en su hogar era la seguridad. En el caso de este robot es capaz de tomar las precauciones de seguridad necesarias para no golpear o chocarse con objetos o personas gracias a sus sensores. Por otra parte, el contacto físico tiene un rol muy importante en las interacciones de los niños con el robot, ya que estos tendían a tocar el robot incluso cuando él no se lo pedía. Otra de las consideraciones a tener en cuenta es cuidar las interacciones y conversaciones entre el robot y el profesor para que los niños no se sientan excluidos. También se observó que es importante que el robot indique cómo deben responderle los niños, ya sea por voz o tocando la pantalla, ya que en ocasiones la respuesta del niño no era de la forma esperada por el robot. Otro punto observado fue que las conversaciones largas con el robot tendían a aburrir a los niños y que es preferible utilizar instrucciones cortas y variar la entonación. Por último, también se vio que es muy importante mantener el contacto visual entre los niños y el robot para mantener su atención y que se sientan incluidos en el aprendizaje.

Los investigadores planean continuar investigando en esta aplicación para mejorar su contenido, tanto en calidad como en cantidad [22].

Uso para guiar y ayudar a ancianos a pasear

Un equipo de la universidad de Tsukuba en Japón ha realizado un estudio con el objetivo de diseñar un robot humanoide que sirva como guía y entrenador para pasear con ancianos, prestando tanto motivación como ayuda física, motivando a los ancianos a caminar más a la vez que ayuda a caminar y controla el paso. Para comprobar la factibilidad del proyecto se ha empleado el robot Pepper.

Se utilizaron las capacidades de habla de Pepper para motivar y animar a los ancianos mientras caminaban. También se utilizó la pantalla de la Tablet de Pepper para mostrar las imágenes y aplicación del robot Pepper para la interacción con personas mayores

estadísticas, tales como velocidad, distancia recorrida el número de pasos objetivo o el progreso y mejora respecto a otros paseos anteriores. Uno de los problemas encontrados es que los robots actuales no son capaces de detectar a las personas que se encuentren fuera de su campo de visión, lo que hace necesario emplear cámaras externas adicionales.

En la experimentación participaron 8 ancianos con edades comprendidas entre los 73 y 82 años y mentalmente sanos. Se les dio la opción de elegir la posición relativa al robot para el paseo: izquierda, derecha, o detrás del robot. Debido a las limitaciones para controlar la posición relativa del usuario respecto al robot se decidió realizar un experimento tipo Mago de Oz, de forma que el movimiento del robot es controlado por un operador externo utilizando una interfaz visual y un ratón 3d.



Figura 14. Pepper guiando a un participante

Para evaluar la motivación de los usuarios se realizó un cuestionario después de realizar la actividad. Ninguno de los participantes quiso caminar al lado del robot, eligiendo todos ellos tener al robot delante de ellos, como se ve en la Figura 14, ya que veían al robot más como un guía que como un acompañante con el que interactuar. No obstante, los usuarios mantuvieron una actitud positiva hacia el robot gracias a su aspecto amigable. También se realizó un análisis del paso mediante video para ver comprobar la capacidad de los robots para modificar la forma de caminar de los ancianos, mostrando que efectivamente variaban su velocidad para seguir al robot.

Gracias a los satisfactorios resultados obtenidos y la realimentación recibida de los ancianos se realizarán nuevos estudios en el futuro para mejorar el diseño del robot y realizar un prototipo autónomo funcional [23].

Navegación autónoma e interacción personalizada con usuarios

Un grupo de investigadores ha trabajado con el robot Pepper para incrementar la autonomía del robot mejorando su percepción del entorno y aumentar sus habilidades de interacción con los usuarios.

Para conseguir esos objetivos se empleó ROS (*Robot Operating System*), una plataforma de código abierto para desarrollar software para robots, para otorgar a Pepper capacidades de localización y navegación modernas. Además se ha propuesto una arquitectura para tener

una interacción con humanos efectiva empleando servicios en la nube. Para ello se han mejorado las habilidades de reconocimiento de voz de Pepper empleando el servicio de reconocimiento de voz *Bluemix* de IBM y se utiliza un servicio web propio para el reconocimiento facial basado en redes neuronales.

Es un ejemplo de cómo integrar de forma satisfactoria la plataforma ROS y los servicios de IBM con el software propio de Pepper. También hay que tener en cuenta que al emplear un servicio en la nube para procesar el reconocimiento de voz esto introduce un retardo debido a que se debe enviar el archivo de audio al servidor, que éste lo procese y entonces recibir la frase reconocida. Parte de este retardo pudo ser mitigado haciendo *streaming*, de forma que el fichero de audio es procesado a la vez que se graba.

Como resultado de esta investigación se ha conseguido que Pepper sea capaz de moverse de forma autónoma en un entorno con personas y obstáculos empleando un mapa creado previamente. Además, el robot es capaz de ejecutar comandos de voz dados por usuarios conocidos o usuarios nuevos que estén incluidos en una lista de usuarios empleando una interfaz multimodal. De esta forma, se pueden establecer permisos para que, por ejemplo, el robot sólo responda a determinados comandos si son dados por un administrador [24].

Aprendizaje de lenguaje corporal con comportamiento humano

Investigadores del instituto de ciencia y tecnología en Japón han estudiado cómo generar un lenguaje corporal natural para expresar emociones y así mejorar la conexión y la empatía en la interacción entre humano y robot. Para ello se ha propuesto un nuevo modelo de expresión corporal emocional basado en el desarrollo social y emocional de los niños a partir de sus padres. Los niños perciben como interpretan sus padres las emociones y de esa forma ellos forman sus propias reacciones. Imitando este proceso, un robot puede ser diseñado para generar respuestas emocionales empleando redes neuronales entrenadas con diferentes ejemplos de respuestas emocionales obtenidas de humanos reales.

Para recoger las expresiones corporales de los humanos se emplea un sensor Kinect, a partir del cual se pueden obtener las posiciones de sus extremidades, recogiendo día a día el conjunto de acciones realizadas en la interacción entre el humano y el robot. Para cada acción se obtienen las posiciones correspondientes de las extremidades a lo largo del tiempo, y estas acciones son agrupadas de forma que el robot entonces puede utilizarlas como referencia para generar sus propias expresiones corporales emocionales.

Para validar el modelo se utilizó el repositorio público de gestos de Microsoft y Cambridge, que contiene 12 gestos diferentes y la posición de 20 articulaciones para cada uno de ellos. En el experimento se emplearon 1640 gestos obtenidos de 15 participantes. Se ha supuesto que esas acciones fueron obtenidas durante la interacción diaria con el robot y que corresponden a diferentes respuestas emocionales. Por ejemplo, a partir del conjunto de datos de las respuestas correspondientes a “contento” el robot genera su propia expresión corporal para mostrar que está contento.

Los resultados obtenidos mostraron que este método es capaz de agrupar las diferentes acciones en movimientos corporales similares, incluso aunque los datos obtenidos a partir del sensor Kinect presentasen mucho ruido. También se demostró la validez de emplear sensores de bajo coste como el Kinect para recoger datos de las expresiones corporales en Estudio y aplicación del robot Pepper para la interacción con personas mayores

las interacciones entre humano y robot, que son luego agrupadas y empleadas por el robot para generar sus propias expresiones emocionales. En la Figura 15 se pueden ver algunos ejemplos de expresiones generadas.

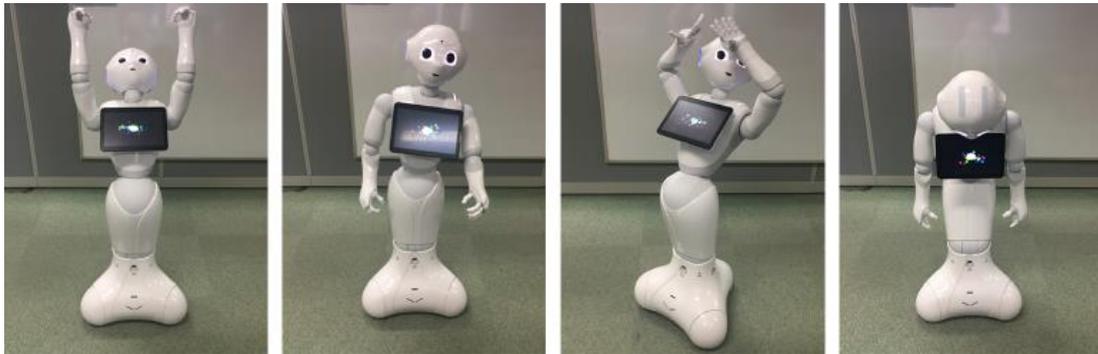


Figura 15. Pepper mostrando expresiones corporales para diferentes emociones: feliz, relajado, asustado y triste

La validez de este método ha sido demostrada, ya que el robot ha sido capaz de generar sus propias expresiones corporales emocionales a partir de un repositorio de datos público. En el futuro los investigadores trabajarán para que el robot sea capaz de generar de forma automática expresiones corporales partiendo de un modelo de respuestas corporales general según el estado actual del robot. También se espera que el comportamiento del robot sea capaz de adaptarse a las peculiaridades culturales y personales de su dueño, empleando sus expresiones corporales para generar otras propias [25].

Aprendizaje de interacción social inteligente

Un grupo de investigadores de Japón ha estudiado cómo un robot puede tener capacidades de interacción social similares a las humanas para poder coexistir en un entorno con humanos. Para esta desafiante tarea se ha propuesto el empleo de *Deep Q-Network* (MDQN) para permitir que el robot sea capaz de aprender cómo interactuar de forma humana empleando un método de prueba y error.

El objetivo es desarrollar un robot que recoja datos durante sus interacciones con humanos y aprenda su forma de interactuar a partir de la información recogida por los diferentes sensores empleando aprendizaje reforzado de extremo a extremo.

Deep Q-Network (MDQN) es un método avanzado de aprendizaje automático (*machine learning*) que combina el aprendizaje profundo (*deep learning*) con el aprendizaje reforzado (*reinforcement learning*). Aplicando este método ya se ha conseguido jugar de forma exitosa a videojuegos de la videoconsola Atari 2600 en nivel experto [26]. En este caso se trata de que el robot emplee este método para aprender habilidades de interacción social interactuando con humanos en lugares públicos.

Las acciones que podía realizar el robot eran: espera, mirar al humano, mover la mano para saludar y dar la mano a la persona. Para comprobar la adecuación del comportamiento se establece un sistema de recompensas, de forma que el robot es premiado o penalizado según haga una acción correctamente o no empleando para ello los sensores del robot. Por ejemplo, si el robot intenta dar la mano y la persona le acepta el apretón de manos, el sensor táctil de la mano lo habrá detectado y entonces el robot obtiene una recompensa positiva. Si, por el contrario, el humano no acepta dar la mano a Pepper entonces este recibe

una penalización. En la Figura 16 se pueden ver ejemplos de ambos casos. Además se emplean las cámaras incorporadas en el robot para ver su entorno y monitorizar el comportamiento de las personas a su alrededor. Los datos recogidos son procesados por un ordenador externo en el que está implementado el MDQN.



Figura 16. Pepper dando la mano con y sin éxito

Los resultados mostraron que el robot fue capaz de aprender habilidades básicas de interacción social de forma exitosa después de interactuar con personas durante 14 días. El robot ha sido capaz de detectar la acción que está realizando la persona de forma que, por ejemplo, si la persona está sentada con su ordenador o está cargando algo con su mano, el robot no le ofrece darle la mano ya que entiende que el usuario no le va a poder responder. También el robot ha sido capaz de ver el movimiento realizado por la persona, de manera que si se está acercando le ofrecerá darle la mano, mientras que se mantendrá esperando si se está alejando, además de poder ver la orientación de la cabeza de la persona, ofreciéndole darle la mano en caso de que la persona le esté mirando.

A pesar de la cantidad de posibilidades de interacción que han tenido lugar, el robot ha sido capaz de aprender a realizar la acción correcta en cada situación. El sistema MDQN ha podido comprender la importancia de las trayectorias con las que caminan las personas, la orientación de su cara, su lenguaje corporal y la actividad que están realizando para decidir la acción apropiada a realizar. En el futuro se espera incrementar las acciones posibles más allá de las cuatro probadas, que el robot pueda mostrar su atención y evaluar la influencia de otras acciones, diferentes a dar la mano, en el comportamiento de las personas [27].

Robocup Home Challenge

Robocup Home challenge es una iniciativa en forma de concurso en el que se prueba la capacidad de los robots para desenvolverse en el entorno de un hogar. Esto implica que los robots deben ser capaces de moverse en un entorno poblado con personas y objetos, además de poder interactuar con los miembros de la familia. Para ello el robot debe ser capaz de detectar los objetos presentes en el entorno, encontrarlos y manipularlos. Este desafío lleva celebrándose anualmente desde el año 2005.

A partir del año 2017 se introduce una nueva modalidad llamada *Social Standard Platform League*, para la cual se emplea al robot humanoide Pepper. Consiste en que el robot actúe como el camarero de una fiesta como se muestra en la Figura 17, para lo cual debe realizar las siguientes acciones:

1. **Entrada:** el robot entra en el área y se dirige a la habitación donde tiene lugar la fiesta, donde espera a ser llamado.

2. **Llamada:** los invitados llaman al robot, pudiendo hacerlo varios de forma simultánea, levantando o agitando la mano o bien llamándolo por voz. El robot debe acercarse a uno de los clientes que le haya llamado, momento en que le debe preguntar a la persona su nombre y la pide la bebida que quiere.
3. **Hacer el pedido:** el robot se debe desplazar hasta el bar, ubicado en otra habitación, donde se sirven las bebidas. Allí le debe repetir el pedido al camarero, indicando el nombre de la persona, la bebida que quiere y una descripción de la persona para que el camarero pueda identificarla, indicando por ejemplo su género, color de pelo o la ropa que lleva. Se debe tener en cuenta que las personas en la fiesta pueden cambiar de lugar.
4. **Bebida no disponible:** habrá una de las bebidas pedidas que no está disponible en el bar. El robot debe darse cuenta de ello y decírselo al camarero, indicando tres alternativas teniendo en cuenta las otras bebidas que se han pedido. Si el robot no es capaz de detectar cuál es la bebida que falta, el camarero indicará claramente cuál es la que falta y dirá tres alternativas posibles.
5. **Corregir el pedido:** el robot debe volver a la habitación de la fiesta, encontrar la persona cuya bebida no está disponible e indicarle las tres alternativas que puede elegir.

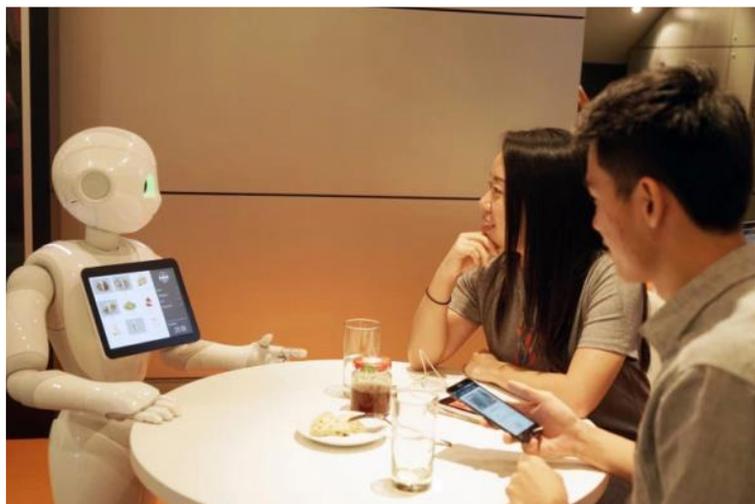


Figura 17. Pepper ejerciendo como camarero

El escenario propuesto se centra en las capacidades del robot para detectar y reconocer personas, moverse de forma segura y en la interacción entre robot y humano con personas desconocidas [28].

La edición del año 2017 fue celebrada en Nagoya, Japón, y resultó vencedor el robot *AUPAIR*, desarrollado por un equipo del laboratorio de biointeligencia de la universidad nacional de Seúl [29].

3 ROBOT PEPPER Y HERRAMIENTAS DE DESARROLLO

En este capítulo se va a realizar una descripción de las características del robot Pepper, así como las herramientas de desarrollo que posee.

3.1 ROBOT PEPPER

Pepper es un robot social originalmente desarrollado por la empresa Aldebaran Robotics, y que fue posteriormente absorbida por Softbank Robotics. Se publicita como ‘el primer robot humanoide diseñado para vivir entre humanos’ [15].



Figura 18. Imagen promocional del robot Pepper

Con su aspecto agradable y amigable, como se puede ver en la Figura 18, busca ser un acompañante para los humanos, comunicándose de forma natural e intuitiva a través de sus movimientos y su voz.

Puede reconocer rostros, hablar, escuchar y moverse de forma autónoma, así como aprender sobre los gustos, hábitos o preferencias del interlocutor y adaptar su comportamiento conforme a ello.

Características técnicas

El robot Pepper tiene una altura de 1,21 metros y 48 cm de ancho. Con sus brazos extendidos tiene una envergadura de 1,196m y alcanza una altura de 1,355m, tal y como se muestra en la Figura 19.

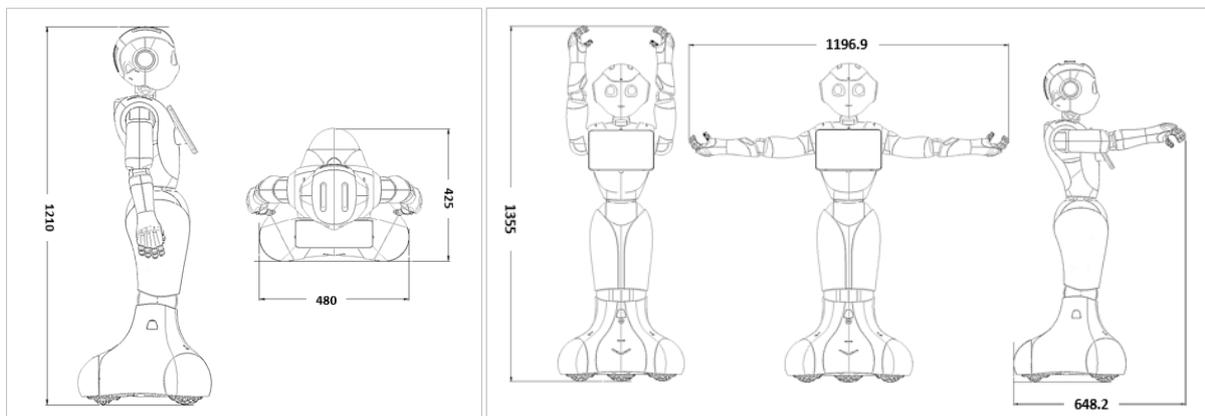


Figura 19. Dimensiones del robot Pepper

Soporta diferentes lenguajes, entre los que se incluye el español, inglés, francés o japonés. Su núcleo incluye una CPU de cuatro núcleos con una memoria de 16 GB disponible para el usuario [30]. En la Figura 20 se muestran las especificaciones detalladas de la placa base.

PROCESSOR	Atom E3845
CPU	Quad core
Clock speed	1.91 GHz
RAM	4 GB DDR3
FLASH MEMORY	8 GB eMMC *
MICRO SDHC	16 GB

Figura 20. Especificaciones de la placa base de Pepper

Para escuchar lo que ocurre en su entorno Pepper incluye 4 micrófonos direccionales ubicados en su cabeza. Esto le permite detectar la dirección de la que vienen los sonidos y localizar la posición del usuario, así como identificar las emociones transmitidas por la voz del interlocutor.

Pepper es capaz de desenvolverse en entornos complejos gracias a su cámara 3D y dos cámaras HD, cuyas ubicaciones son las indicadas en la Figura 21, y que le permiten detectar e identificar movimientos, rostros, objetos o reconocer las emociones en los rostros de los interlocutores. De esta forma, cuando el interlocutor está feliz el robot puede compartir su alegría, o en el caso de estar triste reconfortarle.

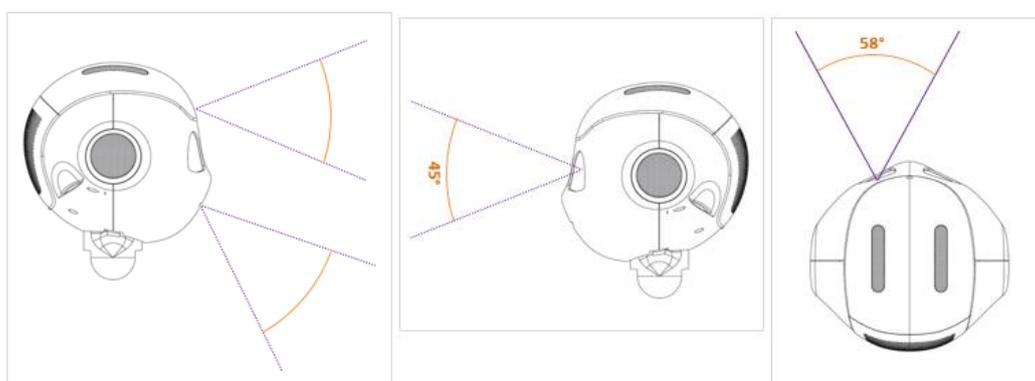


Figura 21. Ubicación de las cámaras 2D y la cámara 3D

Incluye conectividad wifi 802.11a/b/g/n para conectarse a internet y poder, por ejemplo, mantener al usuario al día de las últimas noticias o la previsión del tiempo.

Incluye una Tablet en su cuerpo con la que mostrar información, interactuar o tomar decisiones.

Gracias a sus diferentes sensores Pepper es capaz de detectar personas y obstáculos para evitar colisiones o mantener el equilibrio para evitar caerse si es empujado por alguien. En la Figura 22 y Figura 23 se muestran las ubicaciones de los sensores láser y de ultrasonidos.

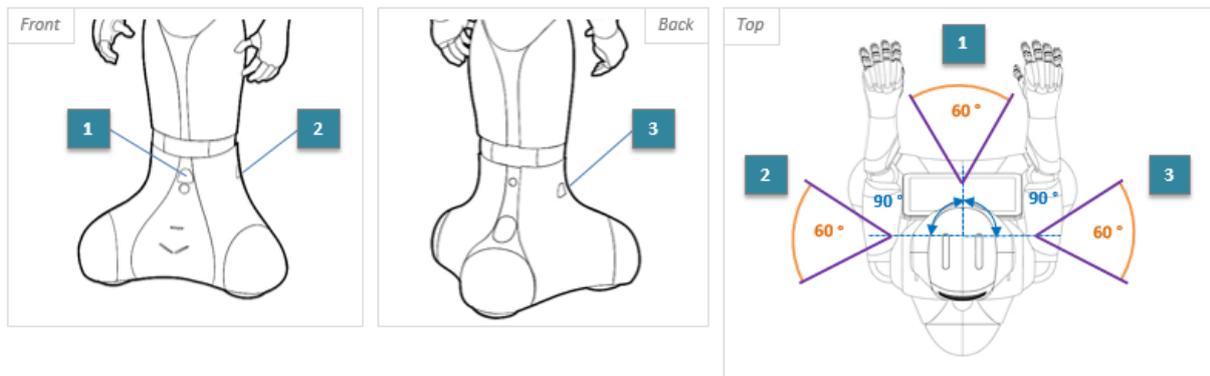


Figura 22. Ubicación de los sensores láser

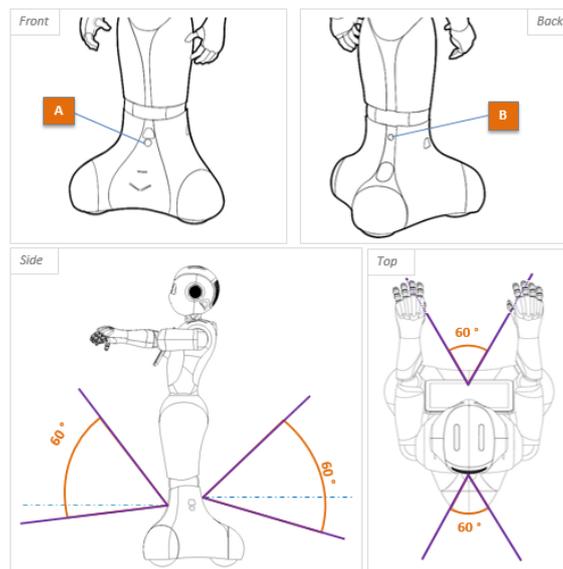


Figura 23. Ubicación de los sensores de ultrasonidos

Puede moverse libremente gracias a sus tres ruedas multidireccionales con una velocidad máxima de 3 km/h. Incluye 20 motores repartidos por el cuello, los brazos o la espalda, tal y como se puede ver en la Figura 24, que le permiten controlar sus movimientos con precisión.

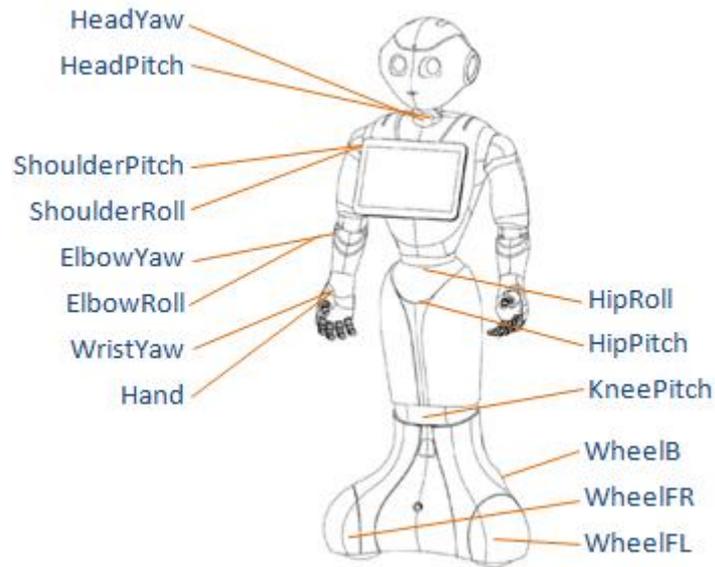


Figura 24. Motores de movimiento

Incluye una batería de litio-ion con una capacidad de 30Ah, lo que le da una autonomía de aproximadamente 12 horas.

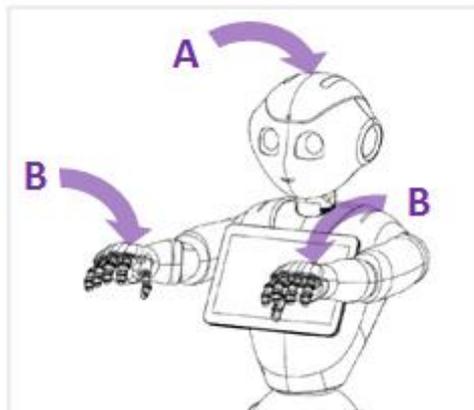


Figura 25. Ubicación de los sensores táctiles

También incluye sensores de temperatura y de nivel de batería, así como sensores táctiles en sus manos y en su cabeza, cuyas ubicaciones se muestran en la Figura 25, que son utilizados para interactuar con el usuario [31].

3.2 PROGRAMA COREOGRAPHE

Softbank Robotics ofrece diferentes alternativas para el desarrollo con el robot Pepper. La herramienta principal es el programa Coreographe, desarrollado por la propia compañía para trabajar específicamente con los robots Nao y Pepper.

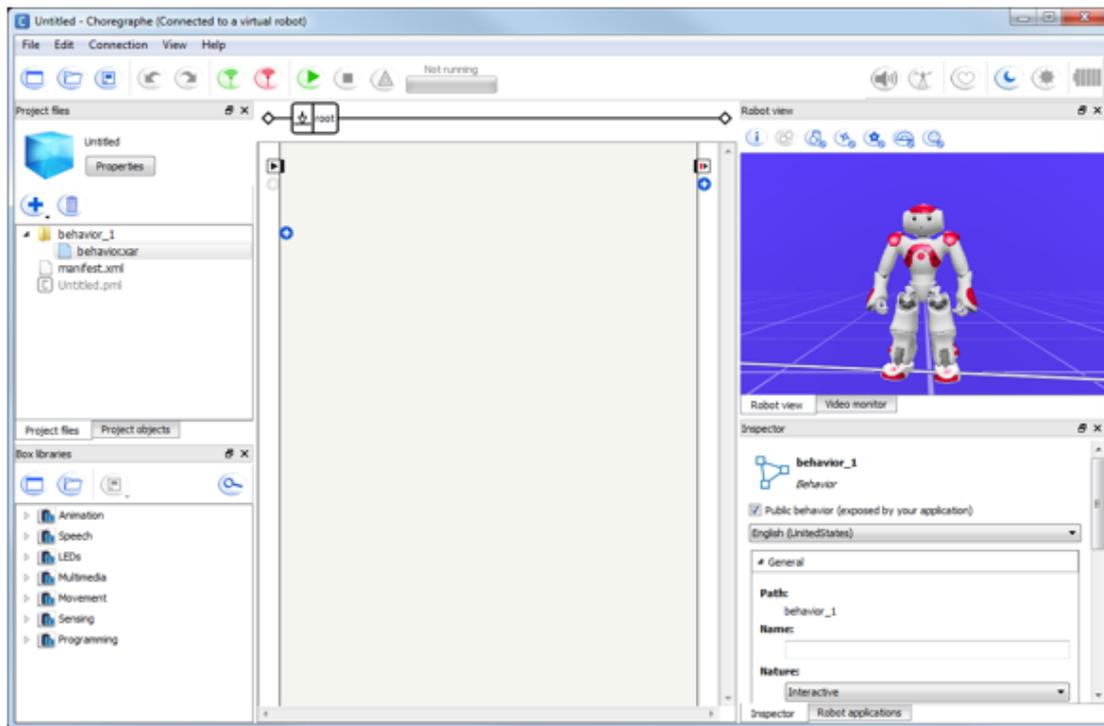


Figura 26. Pantalla principal de CoreoGraphe

A través de CoreoGraphe se pueden desarrollar aplicaciones para los robots Pepper y Nao con la posibilidad de luego publicarlas en la tienda propia de Softbank. Ofrece una interfaz visual con bloques que se van arrastrando y uniendo para formar el diagrama de flujo de ejecución de nuestro programa. Incluye una librería de bloques por defecto para realizar las acciones simples más utilizadas, como por ejemplo decir una frase o reproducir una animación. También permite modificar los bloques incluidos o crear otros nuevos, programándolos a través de código Python, crear nuestras propias animaciones para luego reproducirlas, así como monitorizar y controlar el estado del robot.

En la Figura 26 se puede ver la pantalla principal de CoreoGraphe. En la parte superior se muestran los botones y acciones generales, como abrir o guardar el proyecto, conectarse al robot, ejecutar el programa actual, controlar el volumen del robot o activar/desactivar el modo autónomo. En la parte central es donde se ubican los bloques que conforman el programa que se va a ejecutar. En la parte izquierda se muestran los archivos del proyecto y debajo la librería de bloques por defecto. En la parte derecha se muestra la vista del robot, que puede ser un robot simulado o el robot real al que se está conectado, mientras que en la parte inferior se encuentra el panel *Inspector*, que muestra información del elemento que esté seleccionado.

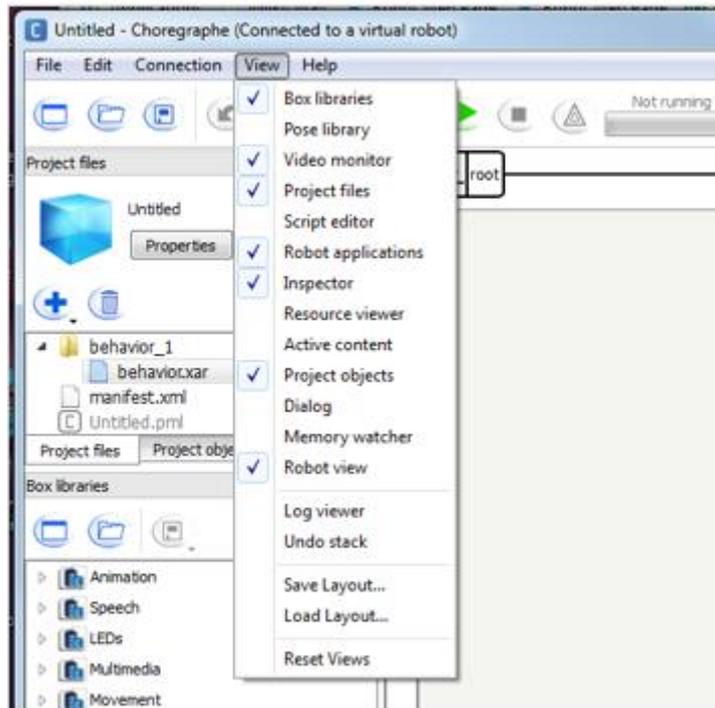


Figura 27. Menú de vista de CoreoGrphe

Los bloques o ventanas que componen la vista de CoreoGrphe se pueden modificar tanto en tamaño como en posición para tener la vista personalizada que se desee. Además, como se ve en la Figura 27, existen otros paneles que se pueden activar desde el menú de vista como por ejemplo el de “*Robot applications*”, que permite ver las aplicaciones instaladas en el robot, ejecutarlas, eliminarlas o cargar e instalar el proyecto actual [32].

Las acciones que realice el programa estarán determinadas por el diagrama de flujo que se establezca, compuesto por bloques conectados entre ellos. El contenido de los bloques puede ser un programa de Python o una animación (*Timeline*), pudiendo ser bloques personalizados creados por el usuario o bien de la librería de bloques por defecto, o bien un bloque de diálogo asociado a un fichero de diálogo correspondiente.

Cada bloque incluye sus entradas y salidas. Por defecto las entradas son *onStart* y *onStop*, asociadas a iniciar el bloque o detener su ejecución, respectivamente, y la salida es *onStopped*, activada cuando el bloque termina de ejecutarse. Además de estas entradas y salidas es posible añadir otras personalizadas, según las necesidades de la acción que realiza el bloque. También es posible modificar el tipo, siendo por defecto de tipo “*Bang*”, que simplemente es un evento, y pudiendo ser también de tipo numérico (*Number*), ya sea entero o flotante, de tipo cadena de texto (*String*) o tipo variable (*Dynamic*), de forma que es posible enviar y recibir datos entre unos bloques y otros.

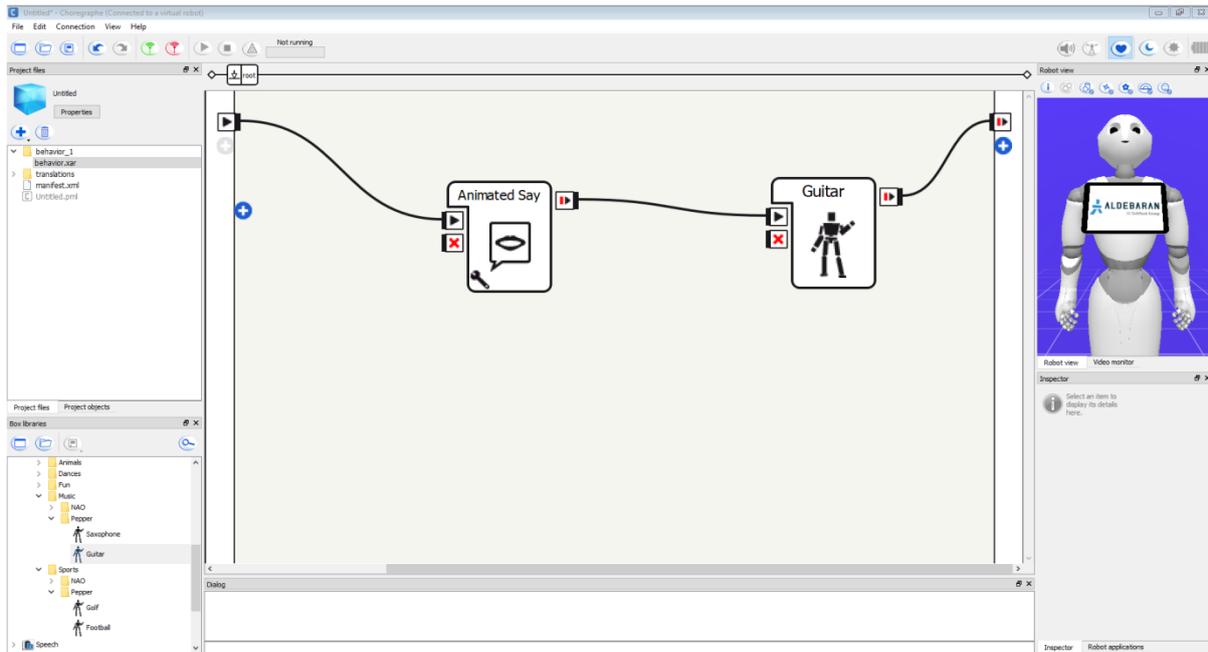


Figura 28. Ejemplo de aplicación en Coreographe

En la Figura 28 se muestra un ejemplo de una aplicación simple empleando los bloques por defecto. Al iniciar la aplicación el robot dice una frase. Cuando termina pasa a reproducir una animación, terminando el programa al finalizar.

Para que el robot ejecute la aplicación creada se establecen sus condiciones de arranque (*Launch trigger conditions*) desde las propiedades de la aplicación. Esto permite que la aplicación se lance cuando se cumple una cierta condición o una combinación de ellas. Por ejemplo, cuando haya fijado a una persona ("*Launchpad/SameTrackedHuman*") o cuando la persona se encuentra a cierta distancia ("*Launchpad/DistanceOfTrackedHuman*") [33].

Bloques de código Python

Los bloques con código de Python permiten ejecutar cualquier programa personalizado que acceda y utilice los servicios del robot (mediante la instrucción *ALProxy*), además de poder emplear todas las instrucciones del lenguaje o incluir otros paquetes de Python.

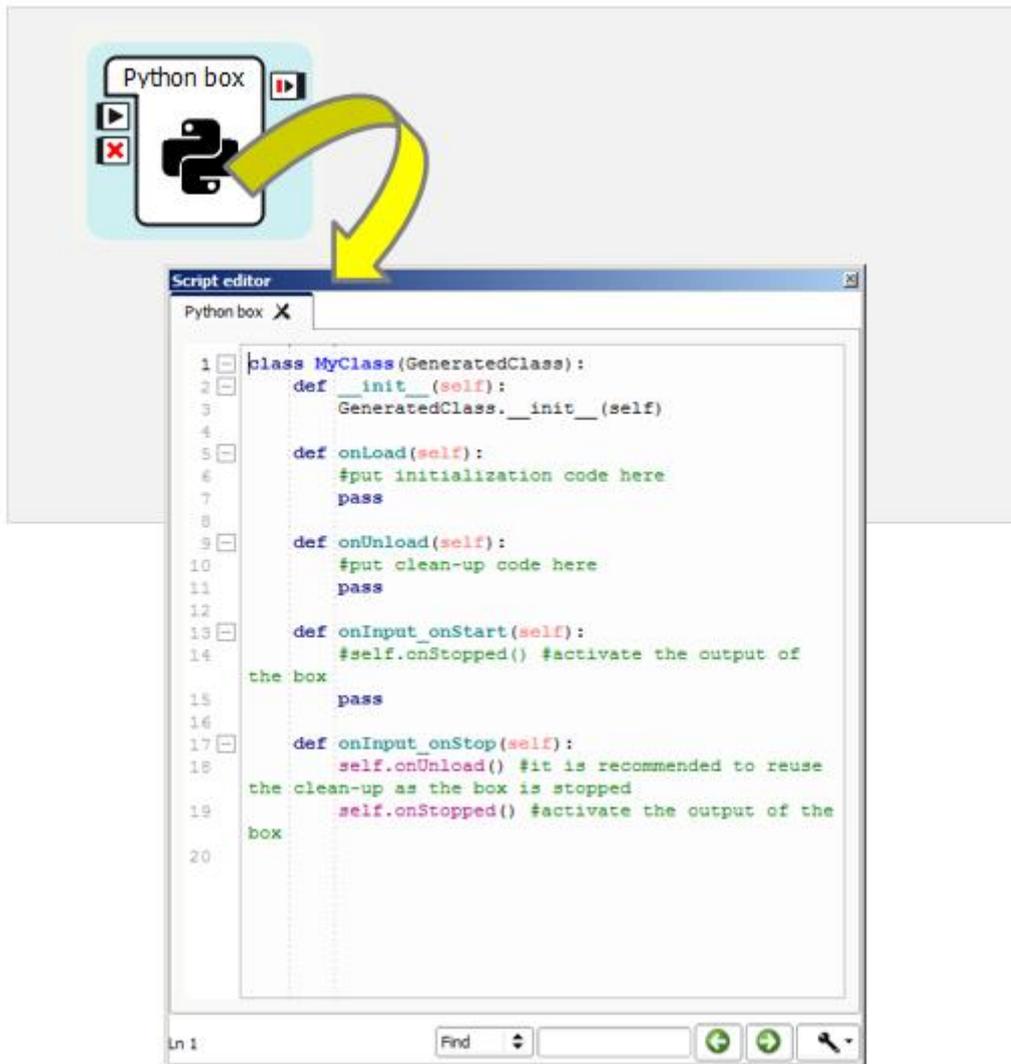


Figura 29. Ejemplo de bloque de código Python

La ejecución de un bloque de Python se divide en diferentes funciones, como se puede ver en la Figura 29, donde aparecen las instrucciones generadas por defecto:

- **__init__**: se ejecuta al inicializar la aplicación
- **onLoad**: se lanza al cargar la aplicación. En esta instrucción se suele incluir todo lo relacionado con la inicialización del programa, como por ejemplo establecer las variables a su valor por defecto.
- **onUnload**: se ejecuta al finalizar la ejecución de la aplicación. Se emplea por ejemplo para liberar recursos utilizados por el programa.
- **onInput_onStart**: es la instrucción principal, ejecutada cuando se recibe la señal correspondiente a través de la entrada onStart. Suele contener el código para realizar las acciones principales del bloque.

Además de estas instrucciones incluidas por defecto se pueden definir otras funciones propias, así como las funciones correspondientes a otras entradas personalizadas del bloque, siguiendo la sintaxis *onInput_ENTRADA*.

Bloques *Timeline*

Los bloques *Timeline* permiten crear animaciones para el robot. Al modificar un bloque de este tipo, como se puede ver en la Figura 30, aparece una línea del tiempo en la que se irán editando las diferentes posturas. A través de la vista del robot se pueden modificar los ángulos de las diferentes articulaciones para colocar al robot con la postura deseada.

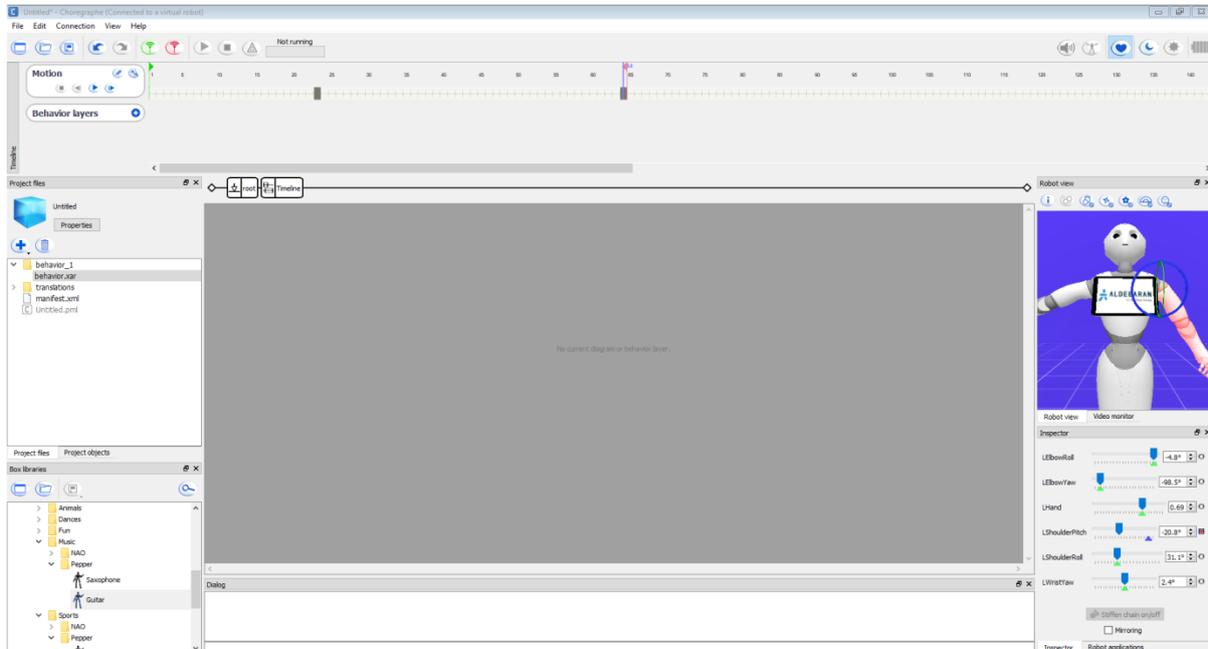


Figura 30. Bloque *Timeline*

Cada una de las posturas se denomina *keyframe*, y puede contener datos de la posición de una o varias articulaciones o del robot completo. Estas posturas se ubican en una posición de la línea del tiempo, de forma que al ejecutar la animación el robot va adoptando las diferentes posturas o *keyframes* en el instante correspondiente a cada una e interpolando entre ellas. También es posible añadir diagramas de flujo con bloques asociados a las posturas, de forma que el robot puede realizar diferentes acciones a medida que va adoptando las posturas.

A través de los bloques *Timeline* se pueden programar, por tanto, una sucesión de posturas con acciones asociadas, lo que permite por ejemplo reproducir animaciones o realizar bailes.

3.3 FUNCIONAMIENTO Y APIS DEL ROBOT

Pepper (y también el robot Nao) utilizan un sistema operativo propio llamado NAOqi. Se trata de una distribución de GNU/Linux embebida basada en Gentoo y adaptada específicamente para las necesidades de los robots de Softbank.

Además de a través de Coreographer, es posible utilizar las SDKs para programar módulos propios que se ejecuten en el NAOqi por medio de Matlab, Python, C++, .net o Microsoft Robotics Studio.

A la hora de desarrollar una aplicación para un uso real nos encontraremos que los bloques por defecto que incluye la aplicación Coreographer no son suficientes. Sin embargo, como se indicó anteriormente, es posible modificar los bloques existentes o crear otros nuevos a Estudio y aplicación del robot Pepper para la interacción con personas mayores

través de código Python. Gracias a esto se abre todo un abanico de posibilidades, pudiendo acceder a través de código a los diferentes servicios o APIs del sistema NAOqi del robot, y con ello explotar todas las posibilidades que ofrece el mismo.

Por defecto el robot se encuentra en un modo de “vida autónoma”, esquematizado en la Figura 31, en el cual a través de los diferentes sensores es receptivo a los estímulos del exterior, siguiendo los ruidos y el movimiento a su alrededor. A través de sus cámaras busca e identifica a las personas que tenga a su alrededor para interactuar con ellas a través de los programas que tenga instalados [34].

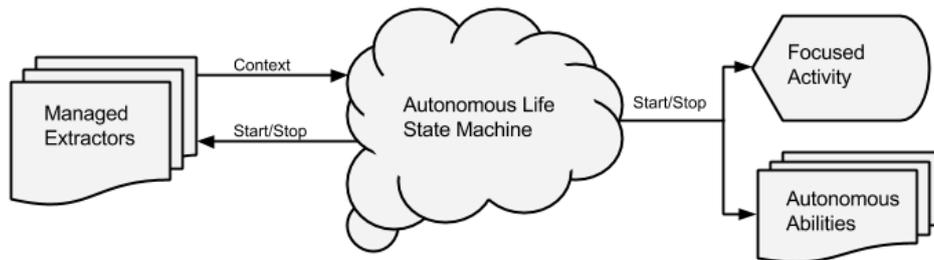


Figura 31. Esquema del modo de vida autónoma

En el gráfico de la Figura 32 se resume el funcionamiento del sistema de usuarios. Cuando el robot ve a un usuario trata de identificarlo mediante reconocimiento facial, comprobando si es un usuario ya existente al que haya visto con anterioridad. En caso de que no corresponda con un usuario ya existente crea un nuevo perfil, asignándole una ID de usuario. Si no puede identificar correctamente al usuario, por ejemplo por falta de luz en la sala, lo asigna como un usuario anónimo, y cada 3 segundos tratará de identificarle nuevamente.

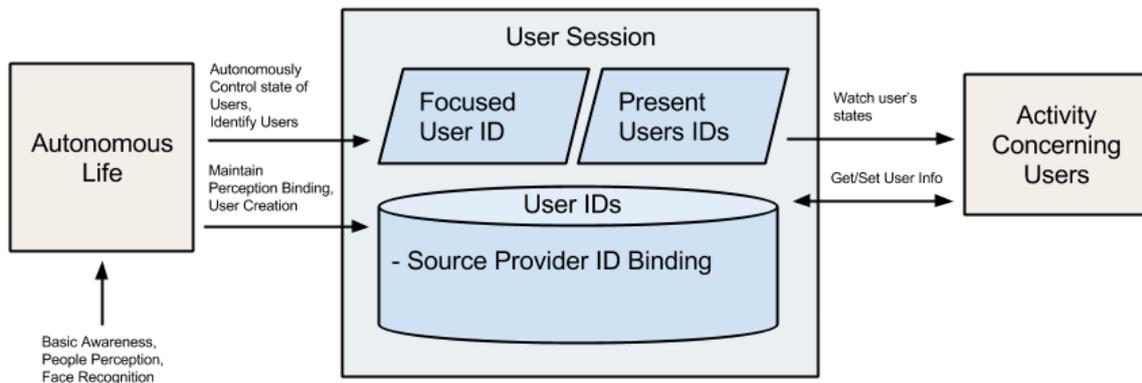


Figura 32. Funcionamiento del sistema de usuarios

Los usuarios pueden ser permanentes o temporales. Los usuarios permanentes serán siempre recordados por el robot, mientras que los temporales serán olvidados o borrados si no son vistos en un intervalo de 10 minutos o si se reinicia el robot. Por defecto todos los usuarios serán temporales, y corresponde al desarrollador de la aplicación elegir cuando hacer permanente a un usuario determinado [35].

APIs del robot

El manejo de las funciones del robot a través de código se hace accediendo a los diferentes APIs o servicios de NAOqi. Cada una de esas APIs ofrece las diferentes funciones correspondientes, ya sea para el manejo de usuarios, control de la Tablet, interacción, movimiento, sonido, visión, detección de personas, sensores, leds, etc. [36].

El listado de las APIs de NAOqi que incluye el robot es el siguiente:

- **Núcleo (Core):** contiene los principales módulos que controlan el funcionamiento general del robot, el control de los usuarios, la memoria y las preferencias.
 - ALBehaviorManager
 - ALConnectionManager
 - ALDiagnosis
 - ALExpressionWatcher
 - ALExtractor
 - ALKnowledge
 - ALMemory
 - ALModule
 - ALMood
 - ALNotificationManager
 - ALPreferenceManager
 - ALResourceManager
 - ALSystem
 - ALVisionExtractors
 - ALTabletService
 - ALUserInfo
 - ALUserSession
 - ALWorldRepresentation
 - PackageManager
 - ServiceManager
- **Interacción:** son los módulos relacionados con el comportamiento interactivo del robot, incluyendo el control del modo autónomo. Dentro de estas capacidades autónomas se incluyen el control de los LEDs para indicar cuando está interactuando con una persona, realizar pequeños movimientos para que el robot parezca “vivo”, observar su entorno para buscar personas y mantener el contacto visual con ellas, así como los movimientos que realiza el robot para gesticular cuando está escuchando y hablando o el motor de diálogo.
 - ALAutonomousBlinking
 - ALAutonomousLife
 - ALBackgroundMovement
 - ALBasicAwareness
 - ALDialog
 - ALListeningMovement
 - ALSpeakingMovement
- **Movimiento:** incluye los módulos relacionados con el movimiento del robot, como pueden ser reproducir animaciones u adoptar posturas, moverse por la sala, volver

de forma automática a la estación de recarga o hacer seguimiento de objetivos (personas objetos o sonidos), ya sea moviendo sólo la cabeza o desplazándose.

- ALAnimationPlayer
- ALMotion
- ALNavigation
- ALRecharge
- ALRobotPosture
- ALTracker
- **Sonido:** contiene los módulos relacionados con la detección y generación de sonidos. Entre ellos se encuentran la grabación de sonidos empleando sus micrófonos, reproducción de sonidos, la conversión de texto a voz, tanto simple como animada, el análisis del tono de voz del usuario o la detección y localización de sonidos.
 - ALAnimatedSpeech
 - ALAudioDevice
 - ALAudioPlayer
 - ALAudioRecorder
 - ALSoundDetection
 - ALSoundLocalization
 - ALSpeechRecognition
 - ALTextToSpeech
 - ALVoiceEmotionAnalysis
- **Visión:** son aquellos módulos relacionados con las cámaras del robot. Permite tomar fotos y vídeos, así como detectar códigos de barras, objetos, puntos de referencia, movimientos, la iluminación de la sala u orientarse.
 - ALBacklightingDetection
 - ALBarcodeReader
 - ALCloseObjectDetection
 - ALColorBlobDetection
 - ALDarknessDetection
 - ALLandmarkDetection
 - ALLocalization
 - ALMovementDetection
 - ALPhotoCapture
 - ALRedBallDetection
 - ALSegmentation3D
 - ALVideoDevice
 - ALVideoRecorder
 - ALVisionRecognition
 - ALVisualCompass
 - ALVisualSpaceHistory
- **Percepción de personas:** incluye los módulos relacionados con la percepción de personas. Permite analizar la posición relativa de la persona respecto al robot, las características de su rostro, detectar y reconocer caras, analizar la dirección a la que está mirando una sujeto, así como percibir las personas a su alrededor, detectar si están sentadas o de pie o si están tratando de llamar la atención del robot.

- ALEngagementZones
- ALFaceCharacteristics
- ALFaceDetection
- ALGazeAnalysis
- ALPeoplePerception
- ALSittingPeopleDetection
- ALWavingDetection
- **Sensores y LEDs:** son los módulos relacionados con el control de los LEDs incorporados en el robot, así como de los sensores, incluyendo los táctiles y ultrasonidos.
 - ALBattery
 - ALBodyTemperature
 - ALChestButton
 - ALFsr
 - ALLaser
 - ALLeds
 - ALSensors
 - ALSonar
 - ALTactileGesture
 - ALTouch

En cuanto al diálogo, el sistema ofrece un lenguaje propio para programar los diálogos llamado QiChat. Utilizando su sintaxis se asocian palabras o frases dichas por el usuario con las respuestas que queremos que diga el robot. Entre sus diferentes posibilidades también permite guardar variables, establecer condiciones, palabras opcionales, llamar a funciones de NAOqi o lanzar otras actividades.

En la Figura 33 se muestra un resumen de la sintaxis de QiChat:

Acción	Sintaxis	Ejemplo
Crear un tema	topic: ~name	topic: ~introduction
Añadir una entrada del usuario y su respuesta	u:(entrada) respuesta	u:(¿cómo te llamas?) mi nombre es Pepper
Añadir una lista de palabras posibles	[]	u:(me gusta el [pollo pescado]) A mí también me gusta
Añadir frases dentro de opciones	comillas ("")	u:(["me gusta" "me encanta"]) a mí también
Añadir palabras opcionales	{}	u:(me siento {muy bastante} bien) eso es estupendo
Asignar una variable	\$variable=xxx	u:(me llamo David) guardo tu nombre \$nombre=David
Decir una variable	\$variable	u:(como me llamo) tu nombre es \$nombre
Condición	\$variable==xxx (también >, < y <)	u:(¿me llamo David?) ["\$nombre==David si" "no, te llamas \$nombre"]
Obtener una opción	_xxx	u:(mi nombre es _[David Roberto]) tu

Acción	Sintaxis	Ejemplo
como variable		nombre es \$1
Sub-opciones	u1: u2: u3:	u:(hazme una pregunta) ¿te gusta el pescado? u1:(si) es bueno para la salud u1:(no) prefiero la carne
Evento	e:Variable	u:([e:faceDetected "Hola"]) ¡Hola humano!
Comentario	#	# línea comentada
Conceptos	~	Concept:(saludo) [hola "buenos días"] u:(~saludo) ¡Hola humano!
Opciones o conceptos aleatorios	^rand	u:(hola) ^rand[hola "que hay"]
Llamada a función	^call() c1:(resultado)	u:(¿puedes decirme cuál es tu posición?) claro ^call(ALRobotPosture.getPosture()) c1:(_*) mi posición es \$1
Tags	%tag frase ^goto(tag)	u:(hola, ¿cómo estás?) hola ^goto(como_estas) u:(como estas) %como_estas estoy bien
Propuestas	proposal: %tag frase ^gotoReactivate(tag)	proposal: %como_estas ¿qué tal estas? u:(hola) hola, ^gotoReactivate(%como_estas)
Reproducir animaciones	^start(animacion) ^wait(animacion)	u:(hola) hola ^start(animations/Stand/Gestures/Hey_1) me alegro de conocerte ^wait(animations/Stand/Gestures/Hey_1)"

Figura 33. Resumen de la sintaxis de QiChat

Como se puede ver, el motor de diálogo QiChat ofrece una gran flexibilidad, lo que abre muchas posibilidades para programar conversaciones con Pepper.

Acceso mediante FTP y SSH

Es posible acceder al robot tanto mediante FTP como SSH. En ambos casos se emplea la dirección "pepper.local.", el usuario "nao" y el puerto 22 [37].

El acceso a través de SSH utilizando el cliente correspondiente, como por ejemplo el software gratuito *Putty*, permite ejecutar comandos de los servicios del robot, así como visualizar información sobre ellos. Por ejemplo, al ejecutar el comando "qicli info ALUserSession -hidden" se obtiene un listado con todas las funciones e instrucciones referidas al servicio *ALUserSession*, incluyendo las funciones que estén ocultas.

Por otra parte, al acceder al robot mediante FTP empleando por ejemplo el software *FileZilla*, también gratuito, se pueden visualizar y modificar los archivos ubicados en la memoria interna del robot. Esto permite, por ejemplo, extraer ficheros de texto con información que hayan sido creados por una aplicación del robot, o añadir datos que vayan a ser después leídos por alguno de las aplicaciones programadas.

3.4 SDK JAVASCRIPT

Para realizar menús en la Tablet con los que controlar el robot se utilizan aplicaciones o webs HTML. De esta forma, en la Tablet se lanza una página web local HTML almacenada en el robot (dentro de una aplicación) y a través de JavaScript se accede a los servicios del robot por medio de la LibQi JavaScript SDK, haciendo así que, por ejemplo, el robot diga algo o lance una determinada aplicación [38].

El servicio se inicializa a través de la clase QiSession, la cual se conecta al robot y obtiene los proxys para los servicios. Utiliza dos argumentos *callback*, ejecutándose uno de ellos en función de si la conexión ha sido exitosa o no.

Una vez iniciada la conexión, a través de la función `service()` se puede obtener un proxy para acceder a cualquier servicio o API (módulo) del robot. Para ello recibe una cadena de texto con el nombre del servicio al que se desea acceder, y si el proceso es exitoso devuelve un objeto con el proxy, tal y como se muestra en la Figura 34:

```
QiSession.connect(function (session) {
  session.service("ALTextToSpeech").then(function (tts) {
    // tts is a proxy to the ALTextToSpeech service
  }, function (error) {
    console.log("An error occurred:", error);
  });
}, ...);
```

Figura 34. Acceso a servicios con el SDK de JavaScript

Una vez obtenido el objeto para un servicio es posible llamar a sus funciones correspondientes, como se ve en la Figura 35:

```
tts.getLanguage().then(function (lang) {
  console.log("I speak " + lang);
}, function (error) {
  console.log("An error occurred: " + error);
});
```

Figura 35. Llamada a funciones de servicios con JavaScript

Además de llamar a funciones de servicios también es posible suscribirse a eventos del robot para realizar una acción correspondiente, como se muestra en la Figura 36:

```
session.service("ALMemory").then(function (ALMemory) {
  ALMemory.subscriber("FrontTactilTouched").then(function (subscriber) {
    // subscriber.signal is a signal associated to "FrontTactilTouched"
    subscriber.signal.connect(function (state) {
      console.log(state == 1 ? "You just touched my head!" : "Bye bye!");
    });
  });
});
```

Figura 36. Suscripción a eventos con JavaScript

A la hora de programar el robot si se quiere utilizar la Tablet será necesario el empleo de la API de JavaScript para mantener enlazado el estado del robot con los que se muestra en la pantalla, tanto para mostrar información como para utilizar botones o realizar acciones a

través de la pantalla táctil se emplearán llamadas a las funciones de los servicios del robot o se realizarán suscripciones a eventos del robot, según el caso.

El uso de esta API, unido a las diferentes características y funciones incluidas en los lenguajes HTML y JavaScript, otorga una gran versatilidad para desarrollar aplicaciones interactivas con una interfaz táctil para el robot Pepper.

4 DESARROLLO DE LAS FUNCIONALIDADES

En este capítulo se va a explicar el funcionamiento de las diferentes funcionalidades desarrolladas para el robot Pepper.

El objetivo es ofrecer un abanico de funcionalidades que puedan ser útiles para interactuar con el usuario. En este caso los usuarios a los que van destinados son personas mayores, con lo que las funcionalidades que ofrece el programa están enfocadas en este tipo de usuarios, con el fin de entretenerlos y mantenerlos activos tanto física como mentalmente.

Las funcionalidades que se han desarrollado son las siguientes:

- Conversación general, interacción por voz con datos del usuario y seguimiento
- Interacción a través de la Tablet
- Información
 - Clima
 - Noticias
 - Búsqueda de información
 - Curiosidades
- Juegos
 - Encontrar parejas
 - Explotar globos
 - Contar chiste
 - Adivinanzas
- Ejercicios
 - Cuello
 - Manos
 - Hombros
 - Varios
- Música

El programa es lanzado de forma automática por el robot. De esta forma, el robot se encuentra inicialmente en el modo de vida autónoma, respondiendo a los estímulos del entorno, y en el momento en que detecta y fija a una persona se lanza el programa.

Al comenzar la aplicación el robot identifica al usuario que tiene delante de él mediante el sistema de usuarios nativo del robot (utilizando reconocimiento facial).

El robot tratará de identificar al usuario como uno existente o uno nuevo. Cuando termina la identificación saluda al usuario.

- Si lo identifica como un usuario conocido, pedirá una confirmación para asegurar que la identificación ha sido correcta. Se puede confirmar respondiendo sí o no o bien tocando en la pantalla la imagen del usuario correspondiente.
- Si no es conocido le preguntará el nombre. Creará un usuario temporal, que se puede convertir en permanente para que el robot lo recuerde pulsando en la pantalla en “nuevo usuario”, o bien por voz diciendo por ejemplo “*quiero crear un usuario*”

nuevo” o “hazme una foto”. Entonces el robot pasará a hacer una foto al usuario y crea el perfil permanente.

Una vez identificado el usuario se puede empezar a utilizar las diferentes funcionalidades del programa. El bloque de diálogo es el que actúa como coordinador, siendo encargado de lanzar las diferentes actividades a petición del usuario, como se puede ver en la Figura 37. Los menús mostrados en la Tablet se coordinan con el diálogo, de forma que al pulsar un botón en la Tablet se pasa al punto correspondiente del diálogo, y de la misma manera en sentido contrario, mostrando el menú correspondiente según la orden dada a través del diálogo.

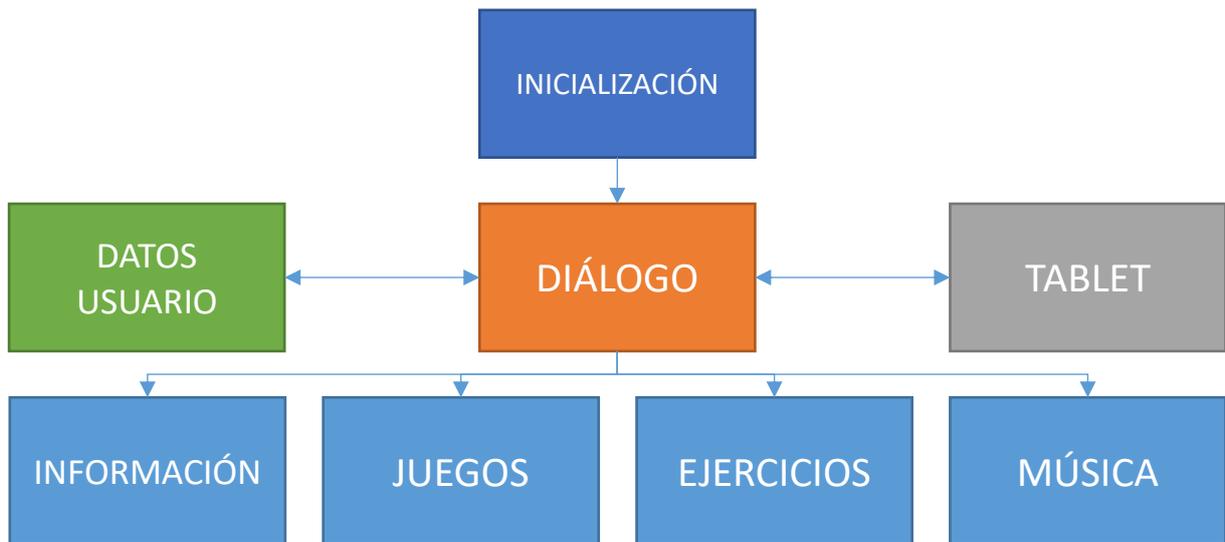


Figura 37. Esquema del funcionamiento del programa

A continuación se va a ver en detalle cada una de las funcionalidades.

4.1 CONVERSACIÓN GENERAL, INTERACCIÓN POR VOZ CON DATOS DEL USUARIO Y SEGUIMIENTO

Para cada usuario permanente se guardan una serie de datos sobre él:

- Nombre
- Foto
- Edad
- Fecha de nacimiento:
 - Día
 - Mes
 - Año

Para poder almacenar estos datos se utiliza un fichero JSON y la API *ALUserInfo*. Para cada usuario se genera un fichero en formato JSON con el nombre del identificador del usuario generado por el robot (*userID*) obtenido a través de la API *ALUserSession*, siendo almacenado en la memoria del robot en la ruta *"/home/nao/perfiles/userID.json"*. En este fichero se guardan todos los datos del usuario: nombre, foto (como texto en formato base64) y edad de nacimiento.

Adicionalmente se almacenan también el nombre y la foto empleando la API *ALUserInfo*, para poder acceder a esos datos desde la Tablet fácilmente. Esta API permite guardar datos en formato de entero (*int*) o cadena de caracteres (*string*) para cada usuario en un dominio en la memoria del robot. A partir del identificador del usuario actual (*userID*) se accede al dominio que se ha definido para obtener los datos relativos a ese usuario mediante la API *ALUserInfo*.

Los datos de usuario cargados se insertan al diálogo del robot como variables a través de la API *ALDialog*. En el caso de que se trate de un usuario nuevo o temporal no se cargará ningún dato, guardando los datos una vez que se convierta en permanente haciendo una foto del usuario, pasando entonces a mostrarlo en una lista de usuarios.

A medida que el usuario va dando el resto de información a través del diálogo el robot la va almacenando siguiendo el proceso inverso: a través de la API *ALDialog* extrae las variables del diálogo y las guarda en el fichero JSON correspondiente en la memoria del robot, guardando también el nombre y la foto mediante la API *ALUserInfo*.

Si es el día de cumpleaños del usuario el robot le felicitará al saludarle antes de preguntarle el estado de ánimo. Para guardar el día de cumpleaños y/o la fecha de nacimiento el usuario se lo puede decir directamente al usuario, por ejemplo: “mi cumpleaños es el 15 de Agosto”, y el robot lo guardará. También puede preguntar cuál es su cumpleaños o fecha de nacimiento al robot, y en caso de saberla se lo dirá o si no se lo preguntará al usuario.

A través de la librería *datetime* de Python el robot puede obtener la fecha y hora actuales para decírselo al usuario en caso de que lo pregunte.

El robot también será capaz de seguir al usuario cuando éste se lo pida por voz. Esto es posible gracias a la API *ALTracker*. Esta API permite que el robot pueda seguir a diferentes objetivos como una bola de color rojo, una cara de persona, una marca en el suelo, personas o sonidos. El seguimiento puede ser moviendo sólo la cabeza, girando todo el cuerpo o moviéndose. En este caso se utiliza el seguimiento para personas y en movimiento, de forma que el robot va siguiendo al usuario moviéndose por la sala y evitando los obstáculos que tenga alrededor. El seguimiento termina cuando el usuario lo pida por voz o en caso de que se pierda al usuario.

Con el fin de poder realizar el resto de funciones del programa (información, juegos, ejercicios y música) a través de instrucciones por voz se han añadido al diálogo las diferentes posibilidades para cada actividad. De esta forma el usuario puede preguntar al robot por ejemplo que ejercicios puede hacer y el robot se lo indicará. También es posible iniciar directamente la actividad que se desee, por ejemplo “quiero hacer ejercicios de cuello” y el robot pedirá confirmación e iniciará la actividad.

Un aspecto que se ha tenido que tener en cuenta a la hora de programar las diferentes opciones para los diálogos es que el robot no posee la opción de *speech to text* (paso de voz a texto) para el idioma Español. Esto significa que para reconocer una palabra tiene que ser dentro del listado de opciones que se le hayan dado en el diálogo. Por ejemplo, cuando el robot pregunta el nombre del usuario no se puede reconocer sin más el nombre dado por el usuario, y tiene que ser un nombre dentro de una lista de opciones dadas en forma de concepto (*concept*).

Con el fin de poder reconocer de la mejor manera posible se han añadido al diálogo diferentes listados de conceptos:

- 100 nombres de hombre y mujer más frecuentes en España
- Listado de años de 1900 a 2050
- Edades de 1 a 115 años
- Meses del año
- Números del 1 a 150
- Deportes más practicados en España
- Colores
- Ciudades de España
- Comunidades Autónomas
- Países
- Personajes históricos
- Deportistas

Gracias a estos listados cuando el robot pregunte algo al usuario se indica en el diálogo el concepto o conceptos correspondientes y así se obtiene la palabra correspondiente dada por el usuario. Por ejemplo, cuando el robot pregunta al usuario su nombre acudirá a los conceptos de nombres de hombre y nombres de mujer y obtendrá el nombre correspondiente dentro de los dos listados.

Además de estas listas de conceptos se ha creado un fichero JSON, ubicado en una dirección web, en el que se almacenan nombres adicionales. De esta forma, cuando el robot vaya a interactuar con un nuevo usuario de la residencia se añade su nombre a esa lista online, el robot leerá el fichero y cargará la lista de nombres al diálogo. De esta forma se asegura que el robot reconozca nuevos nombres sin necesidad de conectarse físicamente al robot o modificar su programación.

4.2 INTERACCIÓN A TRAVÉS DE LA TABLET

Para interactuar con el robot a través de la Tablet se han creado una serie de menús para poder acceder a las diferentes opciones: información, juegos, ejercicios y música. De esta forma, cada uno de los menús es una página web en formato HTML con las diferentes opciones, información y contenido correspondiente. Todos los menús tienen una interfaz simple, con botones y tamaño de letra grande para facilitar su uso por las personas mayores.

La estructura de los menús es la mostrada en la Figura 38:

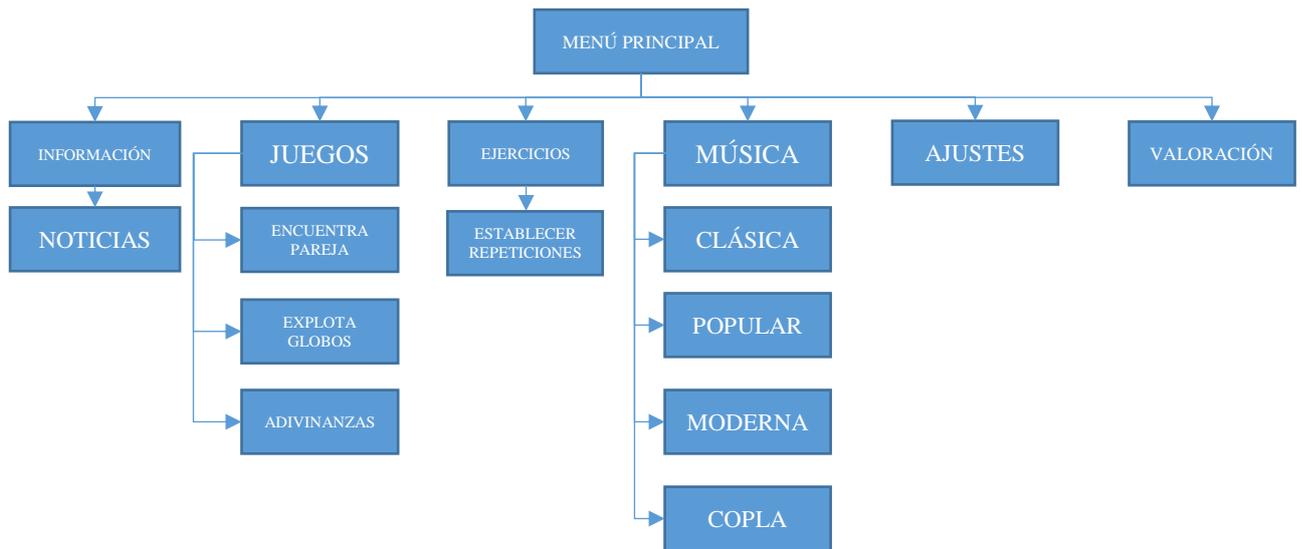


Figura 38. Estructura de menús de la Tablet

Cuando se active una opción en el menú correspondiente la página web debe comunicarlo al robot. Para ello se asocia cada botón con una función de JavaScript. Para poder controlar el robot mediante JavaScript se utiliza el SDK de JavaScript del robot. De esta forma, al pulsar un botón de los menús se irá a la sección del diálogo correspondiente utilizando la API *ALDialog*.



Figura 39. Pepper mostrando el menú principal

Cada uno de los menús permite acceder a las diferentes funcionalidades del robot. En la Figura 39 se puede ver el menú principal, que ofrece acceder a las opciones de información, juegos, ejercicios y música. También muestra en la parte inferior izquierda el nombre del usuario actual y permite acceder a los ajustes y al menú de valoración.



Figura 40. Menú de ajustes

En cuanto al menú de ajustes, mostrado en la Figura 40, permite ajustar el volumen de la voz del robot para que así pueda hablar con un volumen adecuado según el ruido de la sala, controlar el listado de usuarios guardados y acceder a los ajustes de Android de la Tablet. Para acceder a estos dos últimos menús se requiere una contraseña para que sólo accedan aquellos con el permiso necesario.

El menú de control de usuarios, tal y como se ve en la Figura 41, mostrará el listado de usuarios permanentes con su ID y nombre correspondiente con un botón para borrar el usuario que se seleccione. Esto permite a los administradores controlar con qué usuarios está trabajando el robot y borrar aquellos que se deseen. También permite elegir cuál es la pantalla inicial que muestra el robot: el menú principal o la pantalla de selección de usuario.

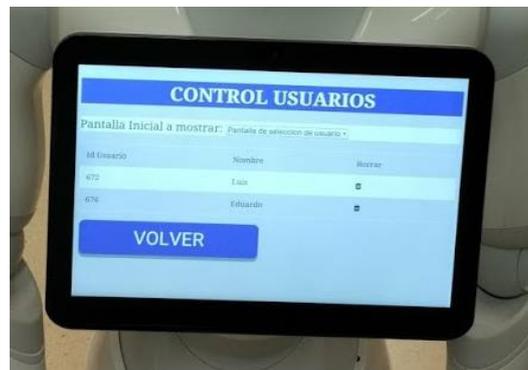


Figura 41. Menú de control de usuarios

Cuando el robot vaya a interactuar con una lista de usuarios fijos (permanentes), como por ejemplo en el entorno de la residencia, se elegirá que muestre la lista de usuarios para que se pueda seleccionar fácilmente el usuario correspondiente. Por otro lado, cuando el robot se vaya a usar con muchas personas diferentes, como por ejemplo en un lugar público, se trabajará principalmente con usuarios temporales, por lo que se elegiría que muestre directamente el menú principal.

4.3 INFORMACIÓN

Las funciones de obtención de información comprenden el clima, las noticias, la búsqueda de información y las curiosidades. El aspecto del menú correspondiente se puede ver en la Figura 42.

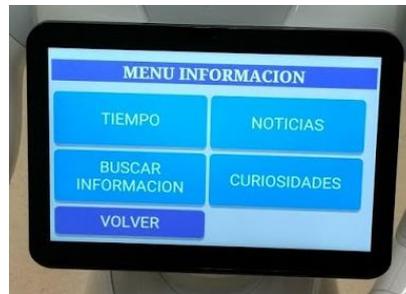


Figura 42. Menú de información

En cuanto al clima, en primer lugar se debe elegir un sitio web del que obtener los datos. En este caso se ha seleccionado la web *OpenWeatherMap*. Esta web ofrece una API que en su opción gratuita permite obtener las condiciones del clima actuales. Los datos son devueltos en formato JSON de forma que pueden ser leídos fácilmente, indicando la descripción del clima (soleado, nublado, etc.), la temperatura, presión, humedad, viento o lluvia. Además, soporta varios idiomas, pudiendo obtener la respuesta en castellano.

En este caso se hará la llamada indicando las coordenadas actuales. De esta forma si el robot cambia de ubicación siempre dará el tiempo correspondiente al lugar donde se encuentre. Para obtener la ubicación se utiliza la API de la web *ipinfo.io*, que devolverá las coordenadas actuales a partir de la dirección IP del robot. Esta llamada para obtener las coordenadas se realiza una única vez cada vez que se enciende el robot.

Teniendo las coordenadas de la ubicación actual se realiza la llamada a la API de *OpenWeatherMap*. Los datos recibidos se descifran mediante la librería JSON de Python, obteniendo los valores de temperatura y la descripción del clima para que las diga el robot utilizando su API *ALAnimatedSpeech*. De esta forma, al pedirle el tiempo el robot dirá, por ejemplo: "Ahora mismo la temperatura es de diez grados y hay cielo claro".

Con respecto a las noticias, el robot tiene que ser capaz de obtener las noticias a partir de internet y decirlas. El método más sencillo para poder obtener las noticias es utilizar los *feed* de RSS que ofrecen los distintos sitios web de noticias. No obstante, no es posible leer su contenido de forma directa. Por ello se ha recurrido a un paso intermedio en el que se utiliza la API de la web *rss2json*, la cual a partir de una dirección que contiene un *feed* RSS devuelve el contenido de esa web convertido a formato JSON, el cual es posible leer de forma más sencilla.



Figura 43. Menú de noticias

Los *feed* RSS que se han utilizado como fuente para el robot corresponden al periódico “El País” para las noticias de portada y economía y el periódico “El Norte de Castilla” para las noticias de deportes y locales. De esta forma, cuando el usuario pida las noticias el robot obtendrá a partir del RSS correspondiente el título y subtítulo de las diferentes noticias y las irá diciendo. En la Figura 43 se puede ver el aspecto del menú en la Tablet.

La función de búsqueda de información consiste en que el robot pueda explicar o hablar sobre algún tema, ya sea una ciudad, país, personaje histórico, deporte, etc. Para ello, como se vio anteriormente, se han definido como conceptos en el diálogo las diferentes opciones sobre las que el robot puede hablar.

Para poder obtener la información se ha elegido la Wikipedia, ya que es una fuente gratuita que además ofrece una API a través de la cual obtener el contenido de los artículos ya en formato JSON. De esta forma, cuando se pregunta al robot sobre algún tema, este utiliza la API de Wikipedia para obtener la información correspondiente, de la cual extrae el primer párrafo del resumen y lo dice mediante la API *ALAnimatedSpeech*.

Además de obtener la información de la Wikipedia, se tiene un fichero online en formato JSON (el mismo que se vio anteriormente en el que se almacenan los nombres de los residentes), donde se han añadido diferentes términos con su explicación correspondiente. Así, el robot carga al diálogo los términos obtenidos del fichero y cuando el usuario pida información sobre ellos el programa recoge la información correspondiente a este término y la dice empleando la API *ALAnimatedSpeech*.

La ventaja que ofrece el tener un fichero JSON online donde poder añadir o modificar información es que permite ofrecer información personalizada sin que se requiera que esa información esté disponible en la Wikipedia, de forma que el robot puede hablar, por ejemplo, sobre los pueblos o lugares de origen de los residentes. Además, al ser un fichero en línea puede ser modificado desde cualquier lugar, de forma que la información dada por el robot varíe sin necesidad de tener el robot presente ni alterar el programa.

4.4 JUEGOS

El robot ofrece una serie de juegos para entretener al usuario y al mismo tiempo, teniendo en cuenta que va a ser utilizado por personas mayores, que puedan ejercitar diferentes capacidades como la memoria, la coordinación o la velocidad de reacción. La interfaz del menú correspondiente se puede ver en la Figura 44.

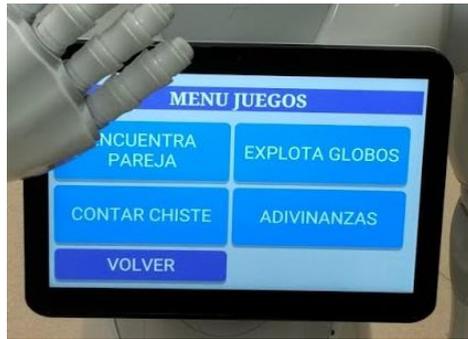


Figura 44. Menú de juegos

Por un lado, el robot puede contar chistes para entretener al usuario, según un listado programado en el diálogo del robot. Se trata de una serie de chistes cortos y sencillos para que sean fáciles de comprender.

Por otra parte, se incluyen tres juegos HTML que se muestran a través de la Tablet: “encontrar parejas”, “explotar globos” o adivinanzas. Estos juegos utilizan código de JavaScript para controlar el propio funcionamiento del juego y además acceder a las APIs del robot para realizar acciones o decir frases en los momentos oportunos.

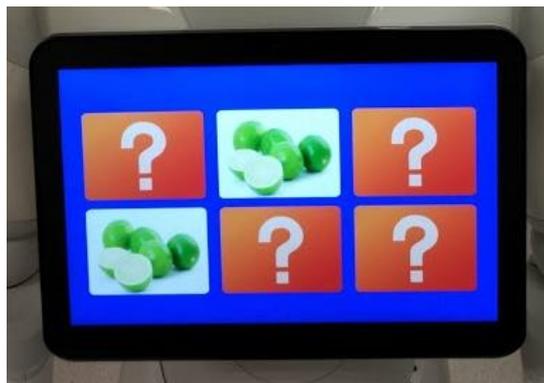


Figura 45. Juego "encontrar parejas"

“Encontrar parejas” consiste en un juego de memoria en el que el usuario tendrá que ir uniendo parejas de frutas. Ofrece diferentes niveles de dificultad con más o menos parejas según las capacidades del usuario y para que éste pueda ir progresando. Su interfaz se muestra en la Figura 45, donde como se puede ver el usuario tendrá que ir tocando la pantalla para ir uniendo las parejas de frutas. Además, el robot irá animando al usuario por voz a medida que vaya acertando las parejas.



Figura 46. Juego "explotar globos"

El segundo juego se llama "explotar globos" y en él el usuario tendrá que ir tocando en la pantalla los globos que vayan apareciendo, trabajando así su capacidad de reacción. Como se ve en la Figura 46, va mostrando por la pantalla una serie de globos que van subiendo y el usuario tiene que explotarlos tocando la pantalla antes de que desaparezcan por la parte superior de la pantalla.

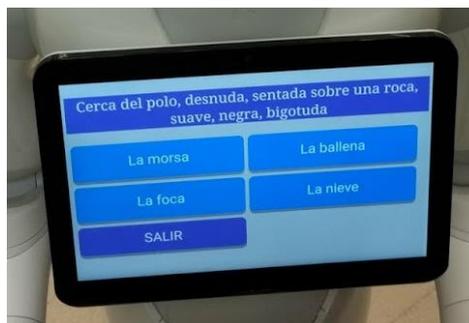


Figura 47. Juego Adivinanzas

El tercer juego es el de adivinanzas. En este juego el usuario tiene que acertar una serie de 10 adivinanzas disponiendo de cuatro posibles opciones, tal y como se muestra en la Figura 47. El robot muestra el texto en la pantalla y lo dice también por voz. El usuario tiene que elegir una de las opciones, ya sea mediante voz o tocando en la pantalla, repitiendo hasta que acierte en caso de fallo, pasando a la siguiente adivinanza una vez que haya acertado.

Cuando termina la partida de un juego el robot le indicará por la pantalla y por voz la puntuación que ha obtenido en función de sus fallos y aciertos. Esta puntuación obtenida se envía también a los *logs* o registros, de forma que es posible monitorizar la evolución del usuario en cada uno de los juegos.

4.5 EJERCICIOS

El robot ofrece diferentes ejercicios orientados a las personas mayores que se dividen en ejercicios de cuello, de manos, de hombros o brazos y varios. Para comenzar el ejercicio deseado se puede pedir por voz o a través del menú correspondiente, como se muestra en la Figura 48:



Figura 48. Menú de ejercicios

Para que el robot realice los diferentes movimientos de cada ejercicio se programan los movimientos utilizando la herramienta *Timeline* del programa *Coreographe*. En la línea del tiempo se van añadiendo las diferentes posturas (*keyframes*) que el robot irá interpolando. Para cada postura se ajustan desde el *Coreographe* los grados a los que se encuentra girada cada articulación, formando así las posturas y animaciones correspondientes para los diferentes ejercicios.

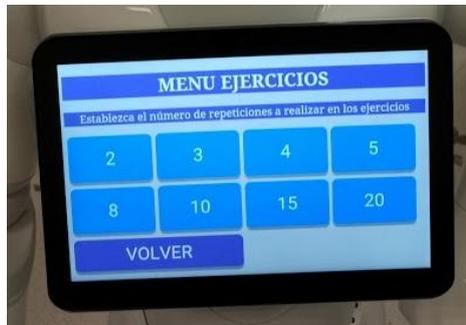


Figura 49. Menú para establecer el número de repeticiones en los ejercicios

Por otra parte, el número de repeticiones que realiza para cada ejercicio se puede ajustar bien por voz o bien a través del menú correspondiente de la Tablet, mostrado en la Figura 49. Así, el robot repetirá la animación correspondiente a cada ejercicio según el número de repeticiones establecido.

Para mantener al usuario activo y motivado cada grupo de ejercicios se realiza con música de fondo y el robot va explicando en voz alta cada uno de ellos y va enumerando cada repetición. La realización de cada ejercicio se puede interrumpir en cualquier momento tocando la pantalla de la Tablet del robot. En la Figura 50 se puede ver un ejemplo del robot realizando ejercicios de hombros.



Figura 50. Pepper realizando ejercicios de hombros

El caso de los ejercicios varios es diferente ya que, a diferencia de los otros, no se trata de una actividad física como tal, sino que cumple una función más orientada al entretenimiento. En estos ejercicios el robot le irá pidiendo que realice diferentes acciones mientras le anima, como por ejemplo, que levante un brazo, que guiñe un ojo o que se toque una oreja.

4.6 MÚSICA

El robot puede reproducir música cuando el usuario se lo pida, ofreciendo diferentes estilos de música: clásica, popular, moderna, copla, pop, rock y heavy. El usuario puede pedir la música que quiere a través de voz o de los menús de la Tablet. En el caso del menú de la Tablet sólo se muestran los géneros orientados a las personas mayores para facilitar su uso por las mismas, tal y como se muestra en la Figura 51, dando a elegir entre música clásica, popular, moderna o copla.



Figura 51. Menú de música

Para ello mostrar en la pantalla la música deseada se utiliza una página web con un marco de YouTube incrustado en el que se ofrece una lista de reproducción con canciones del género solicitado.

Las listas de reproducción han sido creadas específicamente para el robot empleando una cuenta de Google propia, lo que permite por un lado asegurar que los vídeos se reproducen sin problemas (según la licencia algunos vídeos de YouTube no permiten ser incrustados), y por otro que estas listas de música puedan ser modificadas de forma remota, pudiendo

variar el orden de las canciones o añadir otras nuevas simplemente accediendo a la cuenta de YouTube del robot.



Figura 52. Reproducción de música clásica

La interfaz de la Tablet, tal y como se muestra en la Figura 52, ofrece varios botones táctiles para reanudar o parar la reproducción y navegar a través de la lista de canciones, pudiendo retroceder a la canción anterior o pasar a la siguiente.

4.7 REGISTROS Y VALORACIÓN

Para cada usuario se guarda un registro con las estadísticas de uso por usuario y día, con el fin de estudiar cómo se utiliza al robot, mejorar las aplicaciones y la interacción o ver el progreso de los diferentes usuarios (por ejemplo, viendo la evolución en las puntuaciones que obtienen en los juegos).

Estos registros son guardados en forma de ficheros de texto en la ruta `"/home/nao/log/"` de la memoria del robot, nombrándolos por usuario y fecha siguiendo el formato `"userID_DD-MM-YY.txt"`.

En estos ficheros de registro se van añadiendo diferentes líneas, cada una correspondiente a una interacción con el robot, como el inicio o fin de una sesión, una frase reconocida del usuario, una frase dicha por el robot, la pulsación de un botón de la Tablet, el inicio o fin de una actividad o las puntuaciones obtenidas en los juegos. El registro es controlado por un bloque de Python que recibe los diferentes eventos y los va escribiendo al fichero correspondiente, pudiendo ser enviados por otros bloques de código de las diferentes actividades o por programas en JavaScript ejecutados en la tablet.

El formato seguido para indicar cada interacción es el siguiente:

Fecha | Hora | Usuario | Evento | Info | Tag | Certidumbre

- **Fecha:** en formato DD/MM/YYYY.
- **Hora:** en formato HH:MM:SS
- **Usuario:** identificador del usuario dado por el robot (*userID*)
- **Evento:** descripción del evento
- **Info:** información adicional sobre el evento
- **Tag:** tag del diálogo activado por la interacción o en el que se encuentra
- **Certidumbre:** porcentaje de certidumbre en el reconocimiento de voz del usuario

A continuación se muestran algunos ejemplos de eventos obtenidos en los registros:

22/01/2018	11:33:26	664	Inicio Sesión	Temporal-	-	
22/01/2018	11:33:27	664	Respuesta robot	Hola , como te llamas ?	fraseInicial	-
22/01/2018	11:33:30	664	Entrada usuario	<...> Hugo <...>	preguntaNombre	0.449900001287
22/01/2018	11:33:33	664	Entrada usuario	<...> Beatriz <...>	preguntaNombre	0.528699994087
22/01/2018	11:33:33	664	Respuesta robot	Me alegro de conocerte Beatriz	preguntaNombre	-
22/01/2018	11:33:35	664	Respuesta robot	como llevas el día ?	preguntaNombre	-
19/01/2018	18:39:05	645	Botón pulsado	Ir a juegoParejas	juegoParejas	-
19/01/2018	18:39:05	645	Respuesta robot	de acuerdo	cuentaChiste	-
19/01/2018	18:39:06	645	Inicio Actividad	juegoencuentrapareja-c80554		-
19/01/2018	18:39:10	645	Respuesta robot	¡Vamos allá!	juegoParejas	-
19/01/2018	18:39:25	645	Respuesta robot	Casi!, prueba otra vez	juegoParejas	-
19/01/2018	18:39:29	645	Respuesta robot	Muy bien!	juegoParejas	-
19/01/2018	18:39:55	645	Respuesta robot	Casi!, prueba otra vez	juegoParejas	-
19/01/2018	18:40:06	645	Respuesta robot	Muy bien!	juegoParejas	-
19/01/2018	18:40:17	645	Puntuación Parejas 100	Fácil		-
19/01/2018	18:40:17	645	Respuesta robot	Perfecto!	juegoParejas	-
19/01/2018	18:40:18	645	Botón pulsado	Ir a volverInicio	volverInicio	-
19/01/2018	18:40:18	645	Respuesta robot	está bien	juegoParejas	-
19/01/2018	18:40:19	645	Fin Actividad	juegoencuentrapareja-c80554		-
22/01/2018	11:38:43	664	Fin Sesión	-	-	-

Como se puede ver, los registros ofrecen una variada información que permite hacer un seguimiento de las interacciones entre los usuarios y el robot, pudiendo ser utilizados para evaluar el comportamiento del robot así como el progreso de los usuarios en los juegos.

Por otra parte, se ha incluido la opción de realizar una valoración como una forma adicional de obtener una realimentación de las opiniones de los usuarios sobre los diferentes aspectos del robot. El menú de valoración muestra una pantalla para que el usuario valore diferentes aspectos del robot de 1 a 5, tal y como se muestra en la Figura 53:

- Reconocimiento de voz
- Información
- Juegos
- Ejercicios
- Valoración General



Figura 53. Menú de valoración

Los resultados de esta valoración se almacenan también en los *logs* o registros. De esta forma los administradores y desarrolladores pueden ver qué aspectos están funcionando mejor o peor según la percepción de los usuarios y así poder mejorar el funcionamiento del robot.

5 PRUEBAS REALIZADAS

5.1 ESTRUCTURA DE LAS PRUEBAS

Con el fin de evaluar el desempeño de las diferentes funcionalidades desarrolladas para el robot se han realizado una serie de pruebas con personas reales. Estas pruebas se han realizado en dos fases: la primera con voluntarios en la universidad y la segunda en una residencia real.

La primera fase, por tanto, ha consistido en ver los resultados obtenidos en la interacción del robot con diferentes personas en la universidad. Estas pruebas fueron realizadas en el laboratorio del departamento de ingeniería de sistemas y automática de la universidad de Valladolid. Se contó con un total de 10 participantes con diferentes edades, entre los que se encontraban alumnos, profesores y personal de la universidad, que fueron interactuando de forma individual con el robot. Antes de comenzar a utilizarlo se les dio unas breves indicaciones a cada uno, explicando que podían interactuar con el robot hablando o tocando en la pantalla y en caso de no saber qué hacer con el robot podían preguntarle “¿qué sabes hacer?”. Cada uno de los participantes interactuó libremente con el robot durante el tiempo deseado, siendo grabados con una webcam, y al final de la interacción se les entregó un cuestionario para que lo cumplimentaran.

El cuestionario empleado está basado en el modelo *Almere*, empleado para evaluar la aceptación de las personas hacia un robot social asistencial [39]. En él se evalúan diferentes aspectos:

- Ansiedad (ANX)
- Actitud hacia la tecnología (ATT)
- Condiciones facilitadoras (FC)
- Intención de uso (ITU)
- Adaptabilidad del robot percibida (PAD)
- Entretenimiento percibido (PENJ)
- Facilidad de uso percibida (PEOU)
- Sociabilidad percibida (PS)
- Utilidad percibida (PU)
- Presencia social (SP)
- Confianza (Trust)

Las preguntas realizadas en el cuestionario con relación a esos aspectos son las mostradas en la Tabla 1:

Ansiedad (ANX)	1	Si usara el robot, tendría miedo a equivocarme
	2	Si usara el robot, tendría miedo de romper algo
	3	El robot me asusta
	4	El robot me intimida
Actitud hacia la Tecnología (ATT)	5	Creo que es una buena idea utilizar el robot
Condiciones Facilitadoras (FC)	6	Sé cómo utilizar el robot
Intención de Uso (ITU)	7	Utilizaría el robot otra vez
	8	Utilizaría el robot a menudo
Adaptabilidad Percibida	9	Creo que el robot puede adaptarse a lo que necesito
	10	Creo que el robot puede adaptarse a lo que necesito en un momento particular
	11	Creo que el robot puede ayudarme cuando sea necesario
Entretenimiento Percibido (PENJ)	12	Me entretiene que el robot me hable
	13	Me resulta entretenido los servicios, juegos que me ofrece el robot
	14	El robot me parece fascinante
	15	El robot me parece agradable
	16	El robot me parece aburrido
Facilidad de uso percibida (PEOU)	17	Creo que sabría rápidamente como utilizar el robot
	18	Encuentro el robot fácil de usar
	19	Creo que podría utilizar el robot sin ayuda
	20	Creo que podría utilizar el robot si alguien me ayudara
	21	Creo que podría utilizar un robot si me enseñaran a manejarlo durante un tiempo
Sociabilidad Percibida (PS)	22	Considero al robot un conversador agradable
	23	Encuentro agradable interaccionar con el robot
	24	Siento que el robot me entiende
	25	Creo que el robot es bonito
Utilidad Percibida (PU)	26	Creo que el robot podría serme útil
	27	Me vendría bien tener un robot como este
Presencia Social (SP)	28	Cuando interactúo con el robot siento como que estuviera hablando a una persona.
	29	Siento que el robot me mira
	30	Siento como si el robot tuviera vida
	31	Siento que el robot no es una persona real
Trust	32	El robot transmite confianza
	33	Seguiría un consejo dado por el robot

Tabla 1. Preguntas realizadas en el cuestionario

De forma adicional se añadió también una valoración de las diferentes funcionalidades del robot para conocer cuáles eran las más apreciadas por los usuarios y aquellas susceptibles de mejora:

- Reconocimiento de voz
- Información noticias

- Información general
- Juego Parejas
- Juego Globos
- Adivinanzas
- Chistes
- Ejercicios
- Valoración General

La segunda fase de las pruebas ha consistido en probar el desempeño del robot en una residencia real. En este caso se ha contado con la colaboración del Centro Lacort Viana, perteneciente a Residencias Lacort y ubicado en la localidad de Viana de Cega, en la provincia de Valladolid. El robot fue trasladado al centro por un período de dos semanas, siendo ubicado en un despacho donde los cuidadores han ido llevando a diferentes residentes para interactuar con él.

5.2 RESULTADOS OBTENIDOS

Pruebas en la universidad

En primer lugar se van a ver los resultados obtenidos en las pruebas realizadas en el laboratorio de la universidad, para las cuales se contó con diez voluntarios. Se van a ver, por un lado, los resultados obtenidos en los *logs* o registros recogidos por el robot y por otro los resultados obtenidos en los cuestionarios.

Los registros obtenidos han permitido obtener diferentes estadísticas acerca del uso del robot. En primer lugar se muestran los tiempos de interacción de los participantes en la Tabla 2:

ID Usuario	Tiempo de interacción
396	0:01:38
518	0:03:10
498	0:07:08
516	0:02:43
520	0:01:31
519	0:02:31
513	0:03:35
503	0:06:12
511	0:03:54
521	0:02:45
505	0:07:19
501	0:11:01
Promedio	0:04:27

Tabla 2. Tiempos de interacción en pruebas en el laboratorio

Como se puede ver, los participantes interactuaron durante un tiempo promedio de cuatro minutos y medio. No obstante, existen grandes variaciones en el tiempo según la disponibilidad de los participantes y el interés que les despertaba el robot.

En cuanto al uso de las diferentes actividades, en la Tabla 3 se muestran los resultados obtenidos:

ID Usuario	Actividad	Nº usos	Tiempo de uso
498	Juego explota globos	1	0:00:23
	Noticias	1	0:00:09
	<i>Total</i>	2	0:00:32
520	Juego explota globos	1	0:00:40
	<i>Total</i>	1	0:00:40
513	Música rock	1	0:00:36
	Noticias	1	0:00:14
	<i>Total</i>	2	0:00:50
503	Búsqueda Wikipedia	1	0:00:12
	Juego explota globos	1	0:00:31
	Noticias	1	0:00:13
	<i>Total</i>	3	0:00:56
521	Juego encuentra pareja	1	0:00:28
	Noticias	1	0:00:24
	<i>Total</i>	2	0:00:52
505	Juego explota globos	1	0:01:09
	Música clásica	1	0:00:39
	Noticias	1	0:01:09
	<i>Total</i>	3	0:02:57
501	Búsqueda Wikipedia	1	0:00:12
	Juego explota globos	1	0:00:47
	Música clásica	1	0:01:59
	Música rock	1	0:00:17
	Noticias	2	0:01:18
	<i>Total</i>	6	0:04:33
Promedios		2.7	0:01:37

Tabla 3. Uso de actividades en pruebas en el laboratorio

Como se puede ver, los participantes utilizaron un promedio de aproximadamente 3 actividades durante un tiempo de alrededor de un minuto y medio cada una. Las actividades más solicitadas fueron las noticias con 7 usos, seguida por el juego de explotar globos con 5 usos.

ID Usuario	Juego	Dificultad	Puntuación
521	Encontrar Parejas	Fácil	100,00
505	Explotar Globos	-	100,00
501	Encontrar Parejas	Fácil	71,00
Promedios			90,33

Tabla 4. Puntuaciones juegos en pruebas en el laboratorio

Las puntuaciones obtenidas en los juegos son las mostradas en la Tabla 4; hay que tener en cuenta que las puntuaciones sólo se obtienen cuando se completa la partida de principio a fin. En este caso se obtuvieron unas puntuaciones altas, con un valor medio de 90 sobre 100. Estudio y aplicación del robot Pepper para la interacción con personas mayores

Otro aspecto interesante que es posible observar a través de los registros es el tipo de interacción preferida por los usuarios: el empleo de instrucciones por voz o la interacción pulsando la pantalla táctil del robot. Los resultados son los mostrados en la Tabla 5:

Tipo de interacción	Nº de interacciones	Porcentaje	Promedio Certidumbre	Desviación Certidumbre
Botón pulsado	31	8,4%	1	0
Entrada usuario	338	91,6%	0,50	0,146

Tabla 5. Datos de interacción por voz y por pantalla en pruebas en el laboratorio

El método de interacción preferido por los participantes ha sido claramente las instrucciones por voz con un 91,6%, mientras que los botones de la pantalla fueron utilizados sólo en el 8,4% de las ocasiones. Por otra parte, el valor promedio de la certidumbre en el reconocimiento de voz ha sido de un 50%, con una desviación del 14,6%. Este valor se explica por la limitación del reconocimiento de voz en castellano, de forma que no es capaz de descifrar todas las palabras dichas por el usuario, sino que sólo detecta las opciones programadas en el diálogo.

En cuanto a los cuestionarios, cada uno de los usuarios respondió a las diferentes preguntas sobre los aspectos indicados anteriormente con una valoración del 1 al 5, siendo 5: completamente de acuerdo, 4: relativamente de acuerdo, 3: neutral, 2: relativamente en desacuerdo, 1: completamente en desacuerdo. Las funcionalidades del robot fueron evaluadas de 0 a 5, siendo 5 la máxima puntuación y 0 la mínima. Se obtuvieron los resultados mostrados en la Tabla 6:

	Promedio	Desviación	Mediana	Moda	Sin contestar
Edad	42,09	15,15	51	21	0,00%
5: completamente de acuerdo, 4: relativamente de acuerdo, 3: neutral, 2: relativamente en desacuerdo, 1: completamente en desacuerdo					
Ansiedad (ANX)	1,64	1,01	1	1	0,00%
Actitud hacia la Tecnología (ATT)	3,91	0,70	4	4	0,00%
Condiciones Facilitadoras (FC)	3,64	0,92	4	3	0,00%
Intención de Uso (ITU)	3,91	1,25	4	5	0,00%
Adaptabilidad Percibida	3,71	1,16	4	4	6,06%
Entretenimiento Percibido (PENJ)	3,69	1,03	4	4	0,00%
Facilidad de uso percibida (PEOU)	4,09	0,82	4	4	0,00%
Sociabilidad Percibida (PS)	3,47	0,88	3	3	2,27%
Utilidad Percibida (PU)	3,52	0,81	4	4	4,55%
Presencia Social (SP)	3,50	1,21	4	4	0,00%
Trust	3,36	0,79	3	3	0,00%
Valora el robot de acuerdo a los siguientes aspectos (5 mejor valorado, 0 peor valoración)					
Reconocimiento de voz	3,27	1,01	4	4	0,00%
Informacion Noticias	4,00	0,87	4	5	18,18%
Información General	4,22	0,67	4	4	18,18%
Juego cartas	3,50	0,84	3	3	45,45%
Juego Globos	3,71	0,76	4	3	36,36%

Adivinanzas	4,00	0,87	4	5	18,18%
Chistes	4,10	1,10	4,5	5	9,09%
Ejercicios	4,00	0,71	4	4	18,18%
Valoración General	3,82	0,60	4	4	0,00%

Tabla 6. Resultados cuestionarios de pruebas en laboratorio

Como se puede ver, la edad media de los participantes fue de 42 años, con una desviación considerable debido a la diferencia de edad entre los participantes, al ser algunos alumnos y otros profesores o personal de la universidad.

Las preguntas realizadas han sido agrupadas según su aspecto correspondiente. Observando los resultados se puede ver que los participantes se encontraron cómodos con el robot, como refleja el bajo valor de ansiedad obtenido. También se puede ver que los participantes en general tenían una actitud positiva hacia el robot, encontraron su uso sencillo y les pareció útil y entretenido. Por otra parte, los resultados muestran que los participantes han percibido que el robot no tiene una “inteligencia real”, sintiendo que en ocasiones el robot no entiende lo que quieren decir o que no es capaz de mantener una conversación real. También se ha observado que, aunque los participantes se sienten cómodos frente al robot, éste no les transmite la confianza necesaria para seguir sus consejos en un momento dado.

En cuanto a las funcionalidades del robot, el aspecto menos valorado es el reconocimiento de voz, limitado también por la carencia de *speech to text*. Esto implica que el diálogo del robot tenga que ser enriquecido progresivamente para que sea capaz de reconocer las diferentes variantes de palabras y propuestas dadas por los diferentes usuarios, de forma que poco a poco el robot sea capaz de entender una mayor variedad de expresiones. Por otra parte, las funcionalidades de información, chistes, adivinanzas y ejercicios han sido las más valoradas por los usuarios, mientras que los juegos de parejas y globos no llegaron a ser probados por muchos de los participantes y los que lo hicieron le otorgaron una puntuación intermedia.

Pruebas en la residencia

A continuación se van a ver los resultados de la segunda fase de las pruebas, realizadas en el Centro Lacort Viana. En estas pruebas participaron diez residentes, que interactuaron con el robot con la supervisión de los cuidadores del centro a lo largo de diferentes días.

Durante la realización de las pruebas se encontraron dificultades a la hora de crear los perfiles permanentes para algunos de los residentes, por lo que en esos casos el robot fue asignando diferentes usuarios temporales para cada interacción que hacía con ellos. Esto implica que a la hora de estudiar los registros no se ha podido identificar de forma clara los usuarios asignados por el robot con el residente al que corresponden, por lo que se van a estudiar los datos de forma global.

Tiempo total de interacción	3:57:12
Interacciones totales	27
Tiempo promedio por residente	0:08:47

Tabla 7. Tiempos de interacción en pruebas en la residencia

Como se puede ver en la Tabla 7, el robot estuvo en el centro interactuando con los residentes durante un total de aproximadamente 4 horas, repartidas en 27 interacciones con Estudio y aplicación del robot Pepper para la interacción con personas mayores

diferentes residentes a lo largo de varios días. De esta forma, cada residente interactuó con el robot un promedio de aproximadamente 9 minutos por visita.

	Nº de usos	Usos por interacción	Tiempo Total Actividad	Tiempo Promedio
Búsqueda Wikipedia	14	0,52	0:04:48	0:00:21
Ejercicios hombro	1	0,04	0:00:28	0:00:28
Ejercicios manos	2	0,07	0:02:32	0:01:16
Juego encuentra pareja	16	0,59	0:18:56	0:01:11
Juego explota globos	14	0,52	0:12:58	0:00:56
Música clásica	12	0,44	0:09:12	0:00:46
Música heavy	2	0,07	0:00:36	0:00:18
Música pop	4	0,15	0:03:56	0:00:59
Noticias	6	0,22	0:02:12	0:00:22
Totales	71	2,63	0:55:38	0:00:47

Tabla 8. Uso de actividades en pruebas en la residencia

En la Tabla 8 se muestra el uso de las diferentes actividades. Como se puede ver, los residentes emplearon un promedio de 2,6 actividades en cada interacción con una duración media de 47 segundos cada una. Las actividades más utilizadas han sido los juegos de encontrar parejas y explotar globos, la búsqueda de información en Wikipedia y la música clásica.

Juego	Dificultad	Puntuación media
Puntuación Globos	-	62,33
Puntuación Parejas	Fácil	82,18
Totales		77,93

Tabla 9. Puntuaciones juegos en pruebas en la residencia

Las puntuaciones obtenidas en los juegos se pueden ver en la Tabla 9. En este caso la puntuación media obtenida es de aproximadamente 78 sobre 100. Como es comprensible, al tratarse de personas mayores las puntuaciones obtenidas son inferiores a las de las primeras pruebas realizadas en la universidad.

Tipo de interacción	Nº de interacciones	Porcentaje	Promedio Certidumbre	Desviación Certidumbre
Botón pulsado	276	12,19%	1	0
Entrada usuario	1989	87,81%	0,43	0,09

Tabla 10. Datos de interacción por voz y por pantalla en pruebas en la residencia

En este caso el método de interacción preferido por los participantes también ha sido las instrucciones por voz con un 87,81%, aunque el uso de la pantalla táctil ha sido algo mayor que en las pruebas en el laboratorio con un 12,19%. Sin embargo como se puede ver el valor promedio de la certidumbre en el reconocimiento de voz es inferior, con un 43% y una desviación del 9%. Esto es debido a que los residentes en muchas ocasiones hablaban al robot como si se tratase de una persona real, empleando frases largas que el

reconocimiento de voz no puede reconocer al no ajustarse a las instrucciones programadas en el diálogo.

Para las pruebas en la residencia no se realizaron los cuestionarios como en el caso anterior, sino que fueron los propios cuidadores del centro los que comentaron sus impresiones después de ver el desempeño del robot con los diferentes residentes. Los cuidadores indicaron que los residentes se sentían muy cómodos y motivados a la hora de utilizar el robot, de forma que muchos de ellos preguntaban cuando iban a poder volver con el robot e incluso en algún caso confesaron haber soñado con él. A pesar de que el robot no es capaz de entender todo lo que una persona diga y gracias a que asiente con la cabeza cuando escucha, algunos residentes hablaban con él contándole sus preocupaciones o su día a día como si de una persona real se tratase.

Las funcionalidades más apreciadas por los residentes y los cuidadores fueron los juegos de parejas y explotar globos, con los que se divertían y a la vez trabajaban su memoria y su velocidad de reacción. También fueron muy apreciados los ejercicios, debido a que los entretiene y les ayuda a mantenerse activos físicamente.

En cuanto a la música, la que fue incluida originalmente (clásica, pop, rock y heavy) no les motivaba, por lo que se añadieron nuevos estilos más cercanos a su generación como son la música popular (tuna), moderna (años 60-70) o la copla, con los que se encuentran más cómodos y entretenidos.

También se observó que el tono de voz del robot y su velocidad originales provocaba que muchos de los residentes no pudieran entender lo que estaba diciendo, por lo que fue corregido, haciendo que el robot hablase más despacio y con un tono ligeramente más agudo, consiguiendo que se entendiese mucho mejor al robot cuando habla.

Finalmente, otro aspecto a destacar es la disparidad en algunos aspectos en la percepción del robot entre las primeras pruebas, realizadas con personas jóvenes y adultas, y las realizadas en la residencia con las personas mayores. Es destacable que aunque las personas mayores están menos acostumbradas a la tecnología, con lo que quizá se podría esperar un cierto rechazo de estas personas hacia el robot al verlo como algo artificial, el resultado obtenido ha sido el contrario ya que, gracias a su apariencia amigable, los residentes se encontraban cómodos y motivados interactuando el robot, mientras que los jóvenes y adultos de las primeras pruebas, aunque les entretenía y se sentían cómodos con él, mostraban cierta desconfianza. Por otra parte, también es interesante el caso de los juegos de parejas y explotar globos, ya que en las pruebas en el laboratorio muchos de los participantes no los probaron y otros les dieron una puntuación intermedia, quizá por verlos sencillos, con poca dificultad o aburridos, mientras que en el caso de los residentes fueron parte de sus funcionalidades favoritas debido a que les resultaban entretenidos y se ajustaban más a sus habilidades.

6 CONCLUSIONES

Este proyecto ha conseguido cumplir su objetivo principal, consistente desarrollar y validar una serie de funcionalidades enfocadas a emplear al robot Pepper como un robot asistencial para el apoyo a personas mayores, con el fin de interactuar con ellos, entretenerlos y ayudar a mantener sus capacidades físicas y cognitivas.

Las diferentes funcionalidades implementadas permiten que el robot hable con los usuarios, les ofrezca noticias e información, juegos, música y ejercicios, con lo que es capaz de entretenerlos a la vez que los mantiene activos. Además, la utilización de un fichero y listas de música personalizables en línea permiten variar el contenido ofrecido por el robot de forma remota, sin necesidad de conectarse a él físicamente, lo que permite flexibilidad y evita que con el tiempo los usuarios se puedan cansar de un contenido estático.

Las limitaciones existentes en el robot se han solventado empleando diferentes estrategias. Por ejemplo, si el reconocimiento facial del robot no identifica correctamente al usuario, éste puede seleccionar manualmente su perfil correspondiente tocando en la pantalla, o en el caso del reconocimiento de voz y los diálogos pre-programados, se han incrementado las opciones disponibles en el diálogo con la realimentación obtenida en las diferentes pruebas para enriquecerlo y que así el robot entienda una mayor variedad de expresiones.

Se han tenido también en cuenta las peculiaridades asociadas a la interacción con personas mayores, de forma que se ha diseñado una interfaz táctil sencilla y con tamaño de letra grande para facilitar su uso. También se ha ajustado el tono y la velocidad de habla del robot para que hable de forma más clara y así facilitar su comprensión.

Las pruebas realizadas en la residencia han demostrado que los robots sociales asistenciales son una herramienta muy útil para interactuar con personas mayores, consiguiendo que los residentes estuvieran entretenidos y muy motivados al interactuar con el robot. Tanto los residentes como los cuidadores apreciaron especialmente los juegos de explotar globos, enfocado en la coordinación y capacidad de reacción, y encontrar parejas, con el que entrenaban su memoria, así como los ejercicios para la cabeza, el cuello y las manos para mantener su actividad física.

En cuanto a las líneas futuras sobre las que se podría continuar el trabajo realizado en el proyecto existen muchas posibilidades:

- Efectuar más pruebas en entornos reales con las que continuar enriqueciendo el reconocimiento de voz del robot y ver qué aspectos se pueden mejorar, o qué otras nuevas funcionalidades puede ser interesante añadir, como por ejemplo nuevas opciones en el diálogo, juegos o ejercicios.
- Estudiar la posibilidad de emplear otras alternativas para el reconocimiento de voz que sean más completas, tratando de sortear el problema del retardo asociado a la utilización de un servicio externo en la nube.
- Realizar experimentos del tipo mago de oz para estudiar las reacciones de las personas al interactuar con el robot cuando éste aparenta tener una inteligencia real. También puede servir para ver qué temas hablan con el robot, y qué propuestas o

peticiones realizan para tratar de añadirlas luego al diálogo y las funciones del robot, haciendo así que el robot aparente ser más “inteligente” e interactúe de forma más cercana a como lo haría una persona real.

- Investigar otras nuevas funciones para el robot, por ejemplo enfocándose en incrementar la autonomía del robot, tratando de que pueda moverse y navegar libremente por el entorno de la residencia o acompañar a los residentes a sus habitaciones, o incluso basarse en otras investigaciones vistas en esta memoria, haciendo que el robot pueda pasear con los residentes o que les recuerde realizar sus tareas diarias.

Parece claro que los robots sociales son una herramienta con mucho potencial para interactuar con personas mayores, lo que, unido al envejecimiento de la población, hace que sea un campo de investigación que va cobrando relevancia, de forma que es previsible que en el futuro el uso de este tipo de robots estará más extendido y será habitual encontrarlos en los centros para mayores. Además, gracias a las posibilidades de estos robots, es posible emplearlos también en todo aquel ámbito u entorno en el que se tenga interacción con personas, como puede ser centros educativos, edificios públicos, comercios, empresas, etc.

7 BIBLIOGRAFÍA

- [1] «Social robot,» [En línea]. Available: https://en.wikipedia.org/wiki/Social_robot.
- [2] «ASIMO,» [En línea]. Available: <https://en.wikipedia.org/wiki/ASIMO>.
- [3] «ASIMO. Evolution of ASIMO: Key Specifications,» [En línea]. Available: <http://world.honda.com/ASIMO/technology/2011/specification/index.html>.
- [4] «iCub,» [En línea]. Available: <https://en.wikipedia.org/wiki/ICub>.
- [5] «RoboCub,» [En línea]. Available: <http://www.robotcub.org/index.php/robotcub.html>.
- [6] «iCub,» [En línea]. Available: <http://www.icub.org/index.php>.
- [7] «Nao (robot),» [En línea]. Available: [https://en.wikipedia.org/wiki/Nao_\(robot\)](https://en.wikipedia.org/wiki/Nao_(robot)).
- [8] SoftBank Robotics, «Discover NAO,» [En línea]. Available: <https://www.ald.softbankrobotics.com/en/robots/nao>.
- [9] SoftBank Robotics, «Find out more about NAO,» [En línea]. Available: <https://www.ald.softbankrobotics.com/en/robots/nao/find-out-more-about-nao>.
- [10] «PARO Therapeutic Robot,» [En línea]. Available: <http://www.parorobots.com/>.
- [11] M. E. Pollack, L. Brown, D. Colbry, C. Orosz, B. Peintner, S. Ramkrishnan, S. Engberg, J. Matthews, J. Dunbar-Jacob, C. McCarthy, S. Thrun, M. Montemerlo, J. Pineau y N. Roy, «Pearl: A mobile robotic assistant for the elderly».
- [12] J. Pripfl, T. Körtner, D. Batko-Klein, D. Hebesberger, M. Weninger, C. Gisinger, S. Frennert, H. Efring, M. Antona, L. Adami, A. Weiss, M. Bajones y M. Vincze, «Results of a Real World Trial with a Mobile Social Service Robot for Older Adults».
- [13] D. McColl y G. Nejat, «Meal-Time with a Socially Assistive Robot and Older Adults at a Long-term Care Facility».
- [14] P. M. Díez, «Japón confía en la última generación de robots para cuidar a los ancianos. ABC,» [En línea]. Available: http://www.abc.es/sociedad/abci-japon-confia-tercera-generacion-robots-para-cuidar-ancianos-201709302021_noticia.html.
- [15] SoftBank Robotics, «Who is Pepper?,» [En línea]. Available: <https://www.ald.softbankrobotics.com/en/robots/pepper>.
- [16] A. Glaser, «SoftBank's humanoid robot Pepper is improving sales at brick-and-mortar stores. Recode,» [En línea]. Available: <https://www.recode.net/2017/1/4/14171436/softbank-robot-pepper-sales-brick-and-mortar-retail-cs>.

- [17] A. Polo, «El robot Pepper consigue trabajo en Pizza Hut. Expansión,» [En línea]. Available: <http://www.expansion.com/economia-digital/companias/2016/05/24/5744adf422601d930d8b4646.html>.
- [18] AFP y R. O'Hare, «Pepper the robot begins work in Belgian hospitals: Friendly droid is being used to greet patients at reception. Daily Mail Online,» [En línea]. Available: <http://www.dailymail.co.uk/sciencetech/article-3641468/Pepper-robot-finds-job-healthcare-friendly-droid-trialled-two-hospitals-Belgium.html>.
- [19] G. Morales, «El robot Pepper, a su servicio en Lopesan. El Español,» [En línea]. Available: https://www.elespanol.com/viajes/20171129/265724408_0.html.
- [20] Carrefour, «Carrefour presenta en Badajoz al robot Pepper,» [En línea]. Available: <https://www.carrefour.es/grupo-carrefour/sala-de-prensa/noticias2015.aspx?tcm=tcm:5-44188#cerrar>.
- [21] Y. FM, «Este robot (aún) no te quitará el trabajo: le despiden en una semana de una tienda por incompetente. Xataka,» [En línea]. Available: <https://www.xataka.com/robotica-e-ia/no-todos-los-robots-te-quitaran-el-trabajo-a-este-pepper-le-despidieron-por-ineficaz-en-solo-una-semana>.
- [22] F. Tanaka, K. Isshiki y F. Takahashi, «Pepper Learns Together with Children: Development of an Educational Application».
- [23] C. Piezzo y K. Suzuki, «Feasibility Study of a Socially Assistive Humanoid Robot for Guiding Elderly Individuals during Walking».
- [24] V. Perera, T. Pereira, J. Connell y M. Veloso, «Setting Up Pepper For Autonomous Navigation And Personalized Interaction With Users».
- [25] N. T. V. Tuyen, S. Jeong y N. Y. Chong, «Learning Human Behavior for Emotional Body Expression in Socially Assistive Robotics».
- [26] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg y D. Hassabis, «Human-level control through deep reinforcement learning,» *Nature*, nº 518, 2015.
- [27] A. H. Qureshi, Y. Nakamura, Y. Yoshikawa y H. Ishiguro, «Robot gains Social Intelligence through Multimodal Deep Reinforcement Learning,» 2017.
- [28] A. Visser, «A new RoboCup@Home Challenge,» 2017.
- [29] B.-J. Lee, J. Choi, C.-Y. Lee, K.-W. Park, S. Choi, C. Han, D.-S. Han, C. Baek, P. Emaase y B.-T. Zhang, «AUPAIR: The Personal Home Service Robot that Won the RoboCup@Home League,» [En línea]. Available: <https://es.slideshare.net/NaverEngineering/aupair-the-personal-home-service-robot-that-won-the-robocuphome-league>.

- [30] SoftBank Robotics, «Pepper Technical Overview. Motherboard,» [En línea]. Available: http://doc.aldebaran.com/2-5/family/pepper_technical/motherboard_pep.html.
- [31] SoftBank Robotics, «Find out more about Pepper,» [En línea]. Available: <https://www.aldebaran.com/en/robots/pepper/find-out-more-about-pepper>.
- [32] SoftBank Robotics, «Coreographer: Menus, Panels and Toolbar in a glance,» [En línea]. Available: <http://doc.aldebaran.com/2-5/software/choregraphe/interface.html#choregraphe-interface>.
- [33] SoftBank Robotics, «Launch Trigger Conditions,» [En línea]. Available: <http://doc.aldebaran.com/2-5/naoqi/interaction/triggerconditions.html?highlight=launch%20trigger#Launchpad/DistanceOfTrackedHuman>.
- [34] SoftBank Robotics, «Programming for a living robot,» [En línea]. Available: <http://doc.aldebaran.com/2-5/ref/life/index.html#life-framework-explanation>.
- [35] SoftBank Robotics, «ALUserSession,» [En línea]. Available: <http://doc.aldebaran.com/2-5/naoqi/core/alusersession.html>.
- [36] SoftBank Robotics, «NAOqi APIs,» [En línea]. Available: <http://doc.aldebaran.com/2-5/naoqi/index.html>.
- [37] SoftBank Robotics, «NAOqi OS - Getting started,» [En línea]. Available: <http://doc.aldebaran.com/2-5/dev/tools/opennao.html>.
- [38] SoftBank Robotics, «JavaScript SDK,» [En línea]. Available: <http://doc.aldebaran.com/2-5/dev/js/index.html>.
- [39] M. Heerink, B. Kröse, V. Evers y B. Wielinga, «Assessing Acceptance of Assistive Social Agent Technology by Older Adults: the Almere Model».

Anexo A. GUÍA DE USUARIO DEL ROBOT PEPPER

GUÍA RÁPIDA DE USUARIO ROBOT PEPPER

En esta guía se muestran los aspectos generales acerca de cómo utilizar el robot Pepper.

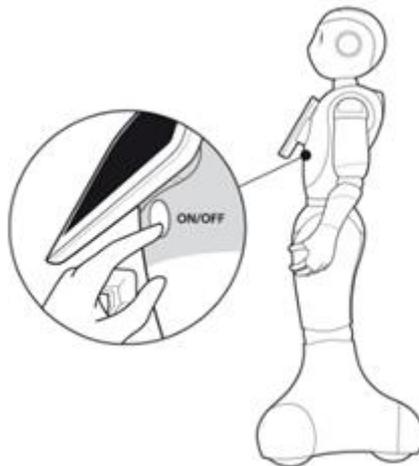
ENCENDER Y APAGAR EL ROBOT

Encendido

- Pulsar una vez el botón de encendido/apagado ubicado debajo de la Tablet.
- El robot comenzará el proceso de encendido.
- Cuando termine el proceso el robot dirá *“OGNAK GNOUK”*, se despertará y se estirará para observar su entorno. Cuando termine ya se podrá utilizar el robot.

Apagado

- Mantener pulsado el botón de encendido/apagado durante 3 segundos, hasta que el robot diga *“GNUK GNUK”*. Alternativamente, es posible apagar el robot mediante voz diciendo *“apágate”* cuando esté en modo escucha (ojos en color azul y girando).
- El robot comenzará el proceso de apagado. Estará completamente apagado cuando se apaguen todos los leds.



INTERACTUAR CON PEPPER

- Colocarse frente al robot para que vea al usuario. El robot también detecta personas automáticamente si escucha algún ruido o ve movimiento realizado por una persona, o bien si detecta alguna cara en su campo de visión.
- Cuando el robot fije a un usuario mostrará el menú en la pantalla de la Tablet.
- El robot tratará de identificar al usuario como uno existente o uno nuevo. Cuando termine la identificación saluda al usuario.
 - Si lo identifica como un usuario conocido, pedirá una confirmación: *“¿hola, eres XX no?”*. Se puede confirmar respondiendo sí o no o bien tocando en la pantalla la imagen del usuario correspondiente.
 - Si no es conocido le preguntará el nombre *“hola, ¿cómo te llamas?”*. Creará un usuario temporal, que se puede convertir en permanente para que el robot lo recuerde pulsando en la pantalla en *“nuevo usuario”*, o bien por voz diciendo *“quiero crear un usuario nuevo”*, *“crea un usuario nuevo”* o *“hazme*

una foto". Entonces el robot pasará a hacer una foto al usuario y crea el perfil permanente.

- Si el robot se confunde y lo identifica como un usuario desconocido o equivocado pero el perfil ya existe, simplemente seleccionar en la pantalla el usuario correspondiente.
- Se puede interactuar mediante voz o bien tocando directamente en los botones de la pantalla.
- El robot sólo responde a la voz del usuario cuando se encuentra en modo escucha, mostrando sus ojos en color azul y girando.
- Cuando el robot está escuchando se le puede preguntar "*¿Qué sabes hacer?*" o "*¿Qué podemos hacer?*" y el robot dirá las diferentes funciones que puede realizar.
- Cuando el robot deja de tener un usuario fijado dejará de mostrar el menú en la pantalla, mostrando el salvapantallas.

COMPROBAR EL NIVEL DE LA BATERÍA

Para comprobar el nivel de la batería se le puede preguntar al robot cuando esté en modo escucha: "*¿Cuánta batería te queda?*", "*¿Qué nivel de batería tienes?*" o "*¿Qué porcentaje de batería tienes?*". El robot responderá indicando el porcentaje de batería restante.

CÓMO CARGAR EL ROBOT

- Conectar el cargador a la red eléctrica.
- Levantar la trampilla situada en la parte trasera del robot para mostrar el conector de carga.
- Conectar el cable del cargador en el conector con la posición apropiada. Girar hacia la derecha hasta escuchar un clic, quedando el conector fijado.
- El led del cargador se muestra en color rojo cuando se está cargando al robot. Cuando esté completamente cargado el led pasará a ser de color verde.
- Cuando el robot esté cargado desconectar el conector del cargador tirando del pestillo del conector y girando hacia la izquierda y cerrar la trampilla de carga.



Se puede interactuar con el robot mientras se está cargando. Este mantendrá sus ruedas fijas, quedando en una posición estática, y seguirá moviendo el resto del cuerpo.

También es posible mantener al robot en una posición estática, aunque no se esté cargando el robot, dejando levantada la trampilla de carga.

CONSEJOS DE USO

Al hablar con Pepper

- Hablar en voz alta y clara.
- Utilizar frases sencillas.
- Para confirmar se puede emplear: *“sí”, “vale”, “de acuerdo”, “me parece bien”, “muy bien”, “claro”, “pues claro”, “por supuesto” o “evidentemente.”*
- Para negar se puede utilizar: *“no”, “no quiero” “no me parece bien” “nunca” o “ni en broma”.*

Utilizar las funciones

- **Cumpleaños, edad y fecha de nacimiento:** decir *“mi cumpleaños es...”*, *“tengo XX años”*, *“nací el...”* o *“nací en el año...”* para que el robot lo recuerde. Una vez que el robot lo conozca lo recordará y se le podrá preguntar utilizando *“cuando es mi cumpleaños”*, *“cuantos años tengo”*, *“cuando nací”* o *“en que año nací”*.
- **Fecha y hora:** preguntar *“qué día es hoy”* o *“qué hora es”* y el robot responderá con la fecha o la hora actuales.
- **Clima:** preguntar *“que tiempo hace”*, *“que temperatura hay”* o *“que temperatura hay en la calle”* y el robot indicará la temperatura y condiciones del clima actuales.
- **Noticias:** decir *“dime las noticias”* y el robot ofrecerá decir las noticias generales, de deportes, economía o locales. También es posible pedir directamente el tipo de noticias, por ejemplo: *“dime las noticias locales”*.
- **Búsqueda de información:** decir *“dame información sobre...”*, *“háblame sobre...”* o *“quién es/era...”*, y el robot dará la información correspondiente. Es posible preguntar sobre las principales ciudades, comunidades autónomas, países, deportes, personajes históricos, deportistas, políticos, empresas, escritores o cineastas.
- **Curiosidades:** decir *“dime algo”* *“dime/cuéntame algo interesante”* o *“dime/cuéntame una curiosidad”* y el robot contará alguna curiosidad. El usuario puede responder, por ejemplo, si le ha parecido divertido, interesante, triste, injusto o si ya lo sabía. El robot le responderá y ofrecerá contar otra curiosidad.
- **Juegos:** decir *“que juegos tienes”*, *“a que jugamos”* o *“quiero jugar”* y el robot ofrecerá jugar a encontrar parejas, explotar globos, adivinanzas o que cuente un chiste. También se puede pedir directamente el juego deseado, por ejemplo: *“quiero jugar a explotar globos”*, *“juguemos a encontrar parejas”* o *“cuéntame un chiste”*. Para interactuar con los juegos de encontrar parejas, explotar globos y adivinanzas hay que tocar la pantalla de la Tablet para realizar las acciones correspondientes.
- **Música:** decir *“quiero escuchar música”*, *“ponme música”* o *“qué música tienes”* y el robot ofrecerá poner música clásica, popular, moderna o copla. También se puede pedir directamente el tipo deseado, por ejemplo: *“ponme música popular”* o *“quiero escuchar música clásica”*. Utilizando los botones de la pantalla se puede cambiar de canción, pausar o salir.

- **Ejercicios:** decir *“quiero hacer ejercicio”, “que ejercicios tienes”* o *“que ejercicios podemos hacer”*, y el robot ofrecerá hacer ejercicios de cuello, manos, hombros o varios. Se puede pedir directamente el ejercicio deseado diciendo, por ejemplo, *“quiero hacer ejercicios de cuello”* o *“empieza los ejercicios de manos”*. Asimismo, es posible variar el número de repeticiones de los ejercicios indicando al robot el número deseado.
- **Volumen:** decir *“habla más bajo/alto”* o *“pon el volumen más alto”* y el robot varía el volumen con el que el habla y reproduce la música. Tiene cuatro niveles: bajo, medio, alto y máximo, pudiendo también indicar directamente el nivel deseado: *“habla a volumen alto”*.
- **Valoración:** decir *“quiero valorarte”, “quiero evaluarte”* o *“quiero valorar tu trabajo”* para realizar una valoración sobre los principales aspectos del robot.
- **Despedida:** decir *“adiós”, “hasta luego” “hasta otra” “nos vemos”* o *“hasta pronto”* y el robot responderá despidiéndose.

RESOLUCIÓN DE PROBLEMAS

En caso de que el robot fallase en algún momento:

- Si está en modo escucha (ojos en color azul girando) decirle *“apágate”* o *“reiníciate”* y confirmar para que se apague o reinicie, volviendo a su funcionamiento normal una vez encendido de nuevo.
- Si no está en modo escucha o no responde a la voz, apagar manualmente manteniendo pulsando el botón de apagado durante 3 segundos, hasta que el robot diga *“GNUK GNUK”*. Una vez apagado completamente (todos los leds apagados), se puede volver a encender.
- Si una vez encendido de nuevo el robot sigue fallando o presentando problemas, ponerse en contacto con el personal correspondiente.

Anexo B. MODELO DE CUESTIONARIO EMPLEADO

Hombre	
Mujer	
Edad	

Puntúa el cuestionario en escala de 1 a 5

5: completamente de acuerdo, **4:** relativamente de acuerdo, **3:** neutral, **2:** relativamente en desacuerdo, **1:** completamente en desacuerdo

	5	4	3	2	1
Si usara el robot, tendría miedo a equivocarme					
Si usara el robot, tendría miedo de romper algo					
El robot me asusta					
El robot me intimida					
Creo que es una buena idea utilizar el robot					
Sé cómo utilizar el robot					
Utilizaría el robot otra vez					
Utilizaría el robot a menudo					
Creo que el robot puede adaptarse a lo que necesito					
Creo que el robot puede adaptarse a lo que necesito en un momento particular					
Creo que el robot puede ayudarme cuando sea necesario					
Me entretiene que el robot me hable					
Me resulta entretenido los servicios, juegos que me ofrece el robot					
El robot me parece fascinante					
El robot me parece agradable					
El robot me parece aburrido					
Creo que sabría rápidamente como utilizar el robot					
Encuentro el robot fácil de usar					
Creo que podría utilizar el robot sin ayuda					
Creo que podría utilizar el robot si alguien me ayudara					
Creo que podría utilizar un robot si me enseñaran a manejarlo durante un tiempo					
Considero al robot un conversador agradable					
Encuentro agradable interaccionar con el robot					
Siento que el robot me entiende					
Creo que el robot es bonito					
Creo que el robot podría serme útil					
Me vendría bien tener un robot como este					
Cuando interactúo con el robot siento como que estuviera hablando a una persona					
Siento que el robot me mira					
Siento como si el robot tuviera vida					
Siento que el robot no es una persona real					
El robot transmite confianza					
Seguiría un consejo dado por el robot					

Valora el robot de acuerdo a los siguientes aspectos (5 mejor valorado, 0 peor valoración)

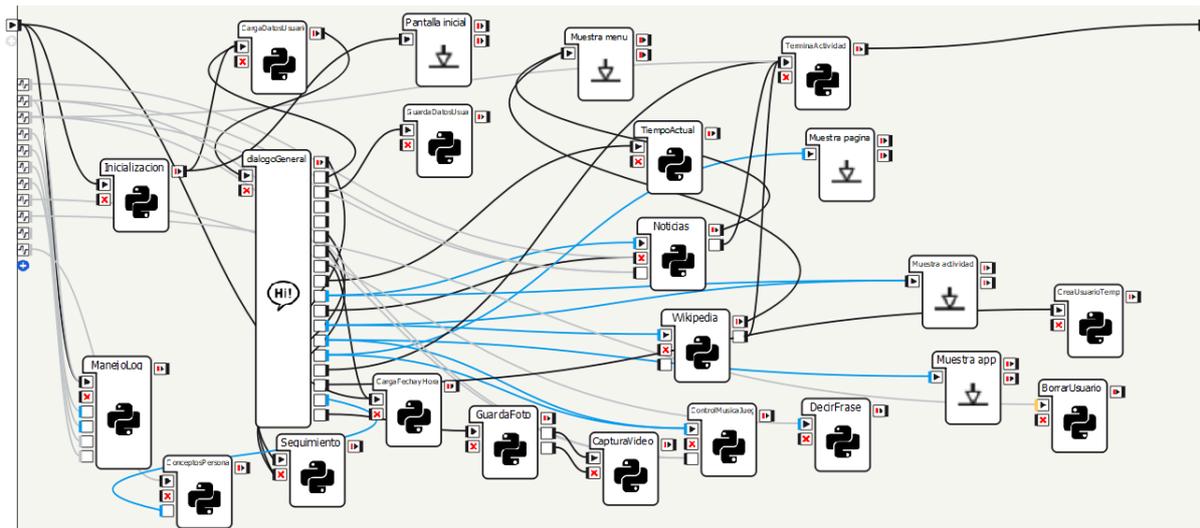
Reconocimiento de voz					
Información Noticias					
Información General					
Juego cartas					
Juego Globos					
Adivinanzas					
Chistes					
Ejercicios					
Valoración General					

Comentarios adicionales

Anexo C. ESQUEMAS Y CÓDIGO DE LOS PROGRAMAS DEL ROBOT

En este anexo se incluyen los esquemas y el código de los diferentes programas desarrollados para el robot.

PROGRAMA GENERAL



Inicialización

```
1. import json
2.
3. class MyClass(GeneratedClass):
4.
5.     def __init__(self):
6.         GeneratedClass.__init__(self)
7.
8.     def onLoad(self):
9.         #put initialization code here
10.
11.         self.userID=0
12.
13.         pass
14.
15.     def onUnload(self):
16.         #put clean-up code here
17.
18.         pass
19.
20.     def onInput_onStart(self):
21.
22.         memory = ALProxy("ALMemory")
23.         memory.insertData("ActividadGeneralEnCurso", False)
24.
25.         basicAwareness = ALProxy("ALBasicAwareness")
26.
27.         basicAwareness.setEngagementMode("FullyEngaged") # Mantiene a la persona fijada
28.         basicAwareness.setTrackingMode("BodyRotation") # Gira el cuerpo y la cabeza
29.
30.         # Configura Voz
31.         tts=ALProxy("ALTextToSpeech")
32.
33.         tts.setParameter("defaultVoiceSpeed", 75) # Voz a 75% de velocidad
34.         tts.resetSpeed() # Pone la velocidad establecida
```

```

35.
36.     tts.setParameter("pitchShift", 1.3) # Habla mas agudo
37.
38.
39.     # Abre y lee archivo de preferencias
40.     filename="/home/nao/preferencias.json"
41.     f=open(filename, 'r')
42.
43.     data_str=str(f.read())
44.
45.     data = json.loads(data_str) # Obtiene datos en forma de diccionario
46.
47.     numeroRepeticiones=int(data["numeroRepeticiones"]) # Obtiene numero de repeticiones
48.
49.     modoPantallaInicial=int(data["modoPantallaInicial"]) # Obtiene modo de pantalla inicial
50.
51.     f.close()
52.
53.     memory.insertData("numeroRepeticiones", numeroRepeticiones) # Escribe numero repeticiones en memoria
54.
55.     memory.insertData("modoPantallaInicial", modoPantallaInicial) # Escribe modo pantalla inicial en memoria
56.
57.     self.onStopped() #activate the output of the box
58.     pass
59.
60.     def onInput_onStop(self):
61.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
62.         self.onStopped() #activate the output of the box

```

Manejo Log

```

1.     import datetime
2.
3.     class MyClass(GeneratedClass):
4.
5.         def __init__(self):
6.             GeneratedClass.__init__(self)
7.
8.         def onLoad(self):
9.             #put initialization code here
10.            self.isRunning = False
11.
12.            self.userID=0
13.
14.            self.user_session_service = ALProxy("ALUserSession")
15.
16.            self.tag=""
17.
18.            pass
19.
20.        def cierraLog(self):
21.
22.            if(self.userID > 0): # Cierra Log
23.
24.                filename="/home/nao/log/"+str(self.userID)+"_"+datetime.date.today().strftime("%d-%m-%Y")+".txt"
25.                f = open(filename, 'ab')
26.
27.                fecha_hora=datetime.datetime.now().strftime("%d/%m/%Y\t%H:%M:%S")
28.                evento="Fin Sesion"
29.                info="-"
30.                tag="-"
31.                certidumbre="-"
32.
33.                f.write(fecha_hora+"\t"+str(self.userID)+"\t"+evento+"\t"+info+"\t"+tag+"\t"+certidumbre+"\n")
34.                f.close()
35.                self.logger.info("Fin sesion. Log guardado en "+filename)
36.
37.        def iniciaLog(self):
38.

```

```

39.     if(self.userID > 0): # Inicia Log
40.
41.         filename="/home/nao/log/"+str(self.userID)+"_"+datetime.date.today().strftime("%d-%m-%Y")+".txt"
42.         f = open(filename, 'ab')
43.
44.         fecha_hora=datetime.datetime.now().strftime("%d/%m/%Y\t%H:%M:%S")
45.         evento="Inicio Sesion"
46.         if(self.user_session_service.isUserPermanent(self.userID)):
47.             info="Permanente"
48.         else:
49.             info="Temporal"
50.         tag="-"
51.         certidumbre="-"
52.
53.         f.write(fecha_hora+"\t"+str(self.userID)+"\t"+evento+"\t"+info+"\t"+tag+"\t"+certidumbre+"\n")
54.         f.close()
55.         self.logger.info("Inicio sesion. Log guardado en "+filename)
56.
57.     def onInput_SalidaRobot(self, salidaRobot):
58.
59.         if(self.userID > 0 and len(salidaRobot) > 1): # Guarda en Log
60.
61.             filename="/home/nao/log/"+str(self.userID)+"_"+datetime.date.today().strftime("%d-%m-%Y")+".txt"
62.             f = open(filename, 'ab')
63.
64.             fecha_hora=datetime.datetime.now().strftime("%d/%m/%Y\t%H:%M:%S")
65.             evento="Respuesta robot"
66.             info=salidaRobot
67.             tag=self.tag
68.             certidumbre="-"
69.
70.             f.write(fecha_hora+"\t"+str(self.userID)+"\t"+evento+"\t"+info+"\t"+tag+"\t"+certidumbre+"\n")
71.             f.close()
72.
73.     def onInput_entradaUsuario(self, value):
74.
75.         if(self.userID > 0 and len(value) > 1): # Guarda en Log
76.
77.             filename="/home/nao/log/"+str(self.userID)+"_"+datetime.date.today().strftime("%d-%m-%Y")+".txt"
78.             f = open(filename, 'ab')
79.
80.             fecha_hora=datetime.datetime.now().strftime("%d/%m/%Y\t%H:%M:%S")
81.             evento="Entrada usuario"
82.             info=value[0]
83.             tag=self.tag
84.             certidumbre=str(value[1])
85.
86.             f.write(fecha_hora+"\t"+str(self.userID)+"\t"+evento+"\t"+info+"\t"+tag+"\t"+certidumbre+"\n")
87.             f.close()
88.
89.     def onInput_eventoLog(self, value):
90.
91.         if(len(value) > 1):
92.
93.             if(self.userID > 0): # Guarda en Log
94.
95.                 filename="/home/nao/log/"+str(self.userID)+"_"+datetime.date.today().strftime("%d-%m-%Y")+".txt"
96.                 f = open(filename, 'ab')
97.
98.                 fecha_hora=datetime.datetime.now().strftime("%d/%m/%Y\t%H:%M:%S")
99.                 evento=value[0]
100.                info=value[1]
101.                tag=value[2]
102.                certidumbre="-"
103.
104.                f.write(fecha_hora+"\t"+str(self.userID)+"\t"+evento+"\t"+info+"\t"+tag+"\t"+certidumbre+"\n")
105.                f.close()
106.

```

```

107. def onInput_tagActivado(self, tag):
108.
109.     if(self.userID > 0): # Cierra Log
110.
111.         self.logger.info("Tag activado: "+tag)
112.         self.tag=tag
113.
114.     def onUnload(self):
115.         #put clean-up code here
116.
117.         self.cierraLog()
118.
119.         self.isRunning = False
120.
121.     pass
122.
123.     def onInput_onStart(self):
124.         if(self.isRunning and self.user_session_service.getFocusedUser()!=self.userID): # si cambia de usuario
125.             self.cierraLog()
126.             self.isRunning = True
127.
128.             self.userID=self.user_session_service.getFocusedUser()
129.
130.             self.iniciaLog()
131.
132.             self.onStopped() #activate the output of the box
133.         pass
134.
135.     def onInput_onStop(self):
136.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
137.         self.onStopped() #activate the output of the box

```

Diálogo General

```

1.  topic: ~dialogoGeneral()
2.  language: spe
3.
4.  include:conceptosGenerales_spe.top #Incluye los conceptos
5.
6.  ##### Nombre Robot #####
7.  concept:(nombre_robot) [Copito]
8.
9.  ##### Conceptos #####
10.
11. concept:(password_ajustes) ["pepper isa Valladolid"]
12.
13. dynamic:concepto_dinamico
14. dynamic:opciones_adivanzas
15. dynamic:nombres_residencia
16.
17. ##### Inicio (confirmacion de usuario) #####
18.
19. u:(e:onStart $ALUserSession.focusedUser>0) %fraseInicial ^first["¿Hola, eres $name ^rand[no verdad "no es así"]?
^gotoReactiva(confirmaUsuario) "Hola, ^gotoReactiva(preguntaNombre)"]
20.
21. proposal: %confirmaUsuario
22.   u1:(~si) %usuarioConfirmado ^rand["me alegro de ^rand[verte "volver a verte"]" "siempre es un placer verte" "m
e alegra verte de nuevo"] $name $muestraPagina="index.html" ^first["$diaCumple==si ^gotoReactiva(cumpleaño
s)" ^gotoReactiva(comoEstas)"]
23.   u1:(["~no" "te has confundido" "{ "no" } {"te has confundido" } no soy $name"]) Perdona, te he confundido ^goto
Reactiva(usuarioEquivocado)
24.
25. proposal: %usuarioTemporalConfirmado $muestraPagina="index.html" ^first[" ~vale $name" "Hola, ^gotoReactiva
te(preguntaNombre)"]
26.
27. proposal: %usuarioEquivocado ^first["$modoPantallaInicial==1 $screaUsuarioTemporal=1" "\pau=250\ selecciona t
u usuario de la lista $muestraPagina="seleccion_usuario.html" "]
28.
29. ##### Foto usuario y crear usuario #####

```

```

30.
31. u:(["hazme {"de nuevo" "otra vez"}" "vuelve a hacerme" repiteme "quiero [hacer hacerme repetir "volver a hacer" "
que me hagas" "que me repitas"]"] [una la] foto) ¿ ^rand["quieres que te haga" "te hago"] ^rand[la una] foto?
32. u1:(~si) ~vale \pau=250\ ^gotoReactivate(preguntaPosicionUsuario)
33. u1:(~no) ~vale
34.
35. u:(["quiero crear" crea] {un} usuario nuevo) ¿quieres crear un usuario nuevo?
36. u1:(~si) ~vale %creaUsuarioNuevo \pau=250\ ^gotoReactivate(preguntaPosicionUsuario)
37. u1:(~no) ~vale
38.
39. proposal: %preguntaPosicionUsuario ¿^rand["como te hago" "como quieres que te haga"] la foto, de pie o sentado?
40. u1:({haz hazme} {"la foto"} de pie) ~vale ^call(ALMemory.insertData("posicionUsuario",1)) $muestraPagina="
crea_usuario_video.html" $hacerFoto=1 #$muestraPagina="crea_usuario.html"
41. u1:({haz hazme} {"la foto"} sentado)~vale ^call(ALMemory.insertData("posicionUsuario",2)) $muestraPagina="
crea_usuario_video.html" $hacerFoto=1 #$muestraPagina="crea_usuario.html"
42.
43. proposal: %proposalFotoTerminada
44. u1:(foto usuario terminada) %fotoTerminada ^rand["Ha salido una foto estupenda" "¡La foto ha quedado genial"]
^first["$name \pau=1000\ $muestraPagina="index.html" " "\pau=1000\ $muestraPagina="index.html" ^gotoReacti
vate(preguntaNombre)"]
45.
46. ##### Seleccion de usuario #####
47.
48. u:(["quiero [cambiar elegir seleccionar]" cambia cambiam selecciona] {de} usuario) ¿quieres cambiar de usuario?
49. u1:(~si) ~vale %seleccionaUsuario $muestraPagina="seleccion_usuario.html"
50. u1:(~no) ~vale
51.
52. ##### Saludo #####
53.
54. u:(~hola {Pepper ~nombre_robot}) %decirSaludo ^start(animations/Stand/Gestures/Hey_1) ^first["$diaCumple==si
Hola $name ^wait(animations/Stand/Gestures/Hey_1) ^gotoReactivate(cumpleaños)" "hola {"de nuevo"} $name ^
wait(animations/Stand/Gestures/Hey_1) ^gotoReactivate(comoEstas)" "hola ^wait(animations/Stand/Gestures/Hey_
1) ^gotoReactivate(preguntaNombre)"]
55.
56. ##### Repetir #####
57.
58. u:(~repetir) $Dialog/Answered
59.
60. ##### Decir funciones #####
61.
62. u:(que [haces hacemos] {aqui}) estoy aqui para ^rand[ayudarte "echarte una mano" "tratar de ayudarte"], ¿^rand["q
quieres saber" "quieres que te cuente" "te digo" "te cuento"] las cosas que puedo hacer?
63. u1:(~si) ^gotoReactivate(decirFunciones)
64. u1:(~no) ~vale
65.
66. u:(["quien eres" "preséntate" ~pregunta_tu_nombre]) ^rand[soy "me llamo" "mi nombre es"] ~nombre_robot, estoy
aqui para ^rand[ayudarte "echarte una mano" "tratar de ayudarte"], ¿^rand["quieres saber" "quieres que te cuente" "t
e digo" "te cuento"] las cosas que puedo hacer?
67. u1:(~si) ^gotoReactivate(decirFunciones)
68. u1:(~no) ~vale
69.
70. u:({dime} [que cuantas] {cosas} [sabes puedes podemos] [hacer decir]) ^gotoReactivate(decirFunciones)
71.
72. proposal: %decirFunciones puedo acordarme de tu nombre, de tu edad o de cuando naciste. También puedo seguirte
, decirte la hora, contarte un chiste, decirte el tiempo, contarte las noticias o hablarte sobre algo. Además puedo pon
erte juegos, música o enseñarte ejercicios para los hombros, el cuello o las manos.
73.
74. ##### Nombre #####
75.
76. #u:(~pregunta_tu_nombre) %decirNombrePropio Me llamo Pepper. ^first["Eres $name, no es asi-
?" "Y tu, ^gotoReactivate(preguntaNombre)"]
77. # u1:(~si) ^rand["siempre es un placer verte" "me alegra verte de nuevo"] $name , ^gotoReactivate(comoEstas)
78. # u1:(~no) Perdona, te he confundido, ^gotoReactivate(preguntaNombre)
79.
80. u:(~pregunta_mi_nombre) %decirNombreUsuario ^first["te llamas $name, no es asi-
?" "No lo se, ^gotoReactivate(preguntaNombre)"]
81. u1:(~si) ^rand["siempre es un placer verte" "me alegra verte de nuevo"] $name , ^gotoReactivate(comoEstas)

```

```

82. u1:({"~no" "te has confundido" {"no"} {"te has confundido"} no soy $name"}) Perdona, te he confundido, ^goto
Reactive(preguntaNombre)
83.
84. proposal: %preguntaNombre como te llamas?
85. u1:({"me llamo" "mi nombre es"} _[~nombres_residencia ~nombre_hombre ~nombre_mujer]) Me alegro de con
ocer te $1 $name=$1 $guardaDatos=1 ^gotoReactive(comoEstas)
86.
87. u:({"me llamo" "mi nombre es"} _[~nombres_residencia ~nombre_hombre ~nombre_mujer]) Me alegro de conocer
te $1 $name=$1 $guardaDatos=1 ^gotoReactive(comoEstas)
88.
89.
90. ##### Interaccion #####
91.
92. u:(~gracias) ^rand["Gracias a ti" "No hay de que"]
93.
94. u:(e:MiddleTactilTouched) ¡Oh, me haces cosquillas!
95.
96. ##### Pregunta estado de animo #####
97.
98. u:({tu} ~como_estas {~nombre_robot}) ^startTag(happy) ^rand[estoy "me encuentro" "" ] ^rand["muy bien" "muy f
eliz" genial] ^waitTag(happy) y tu ^gotoReactive(comoEstas)
99.
100. proposal: %comoEstas ^rand["^rand["que tal" como] estas" "como te va" ^rand["como llevas" "que tal {"te va"}"
el dia"]?
101. u1:({pues} {estoy "me encuentro"} {muy} ~feliz {y} {tu} {~como_estas} {tu} ) ^rand["¡me alegro {mucho} de
que estes bien!" "me alegro" "eso es estupendo" "eso es genial"] \pau=250\ ^startTag(happy) yo también ^rand[esto
y "me encuentro" ^rand["muy bien" "muy feliz" genial] ^waitTag(happy)
102. u1:({pues} {estoy "me encuentro"} {muy "un poco" "mas bien" bueno} ~regular) seguro que el día ^rand[mejora
"va a mejor"]
103. u2:({y} {tu} ~como_estas) ^startTag(happy) yo ^rand[estoy "me encuentro"] ^rand["muy bien" "muy feliz" ge
nial] ^waitTag(happy)
104. u1:({pues} {estoy "me encuentro"} {muy "un poco" "mas bien"} ~triste) Oh, ^rand["lo siento" vaya "que pena"]
105. u2:({y} {tu} ~como_estas) ^startTag(happy) yo ^rand[estoy "me encuentro"] ^rand["muy bien" "muy feliz" ge
nial] ^waitTag(happy)
106. u1:({pues} {estoy "me encuentro"} {muy "un poco" "mas bien"} ~enfermo) Oh, ^rand["lo siento" vaya] \pau=25
0\ espero que te ^rand[mejores "pongas mejor"]
107. u2:({y} {tu} ~como_estas) ^startTag(happy) yo ^rand[estoy "me encuentro"] ^rand["muy bien" "muy feliz" ge
nial] ^waitTag(happy)
108. u1:(~repetir) Te lo vuelvo a preguntar. ^sameProposal
109.
110. ##### Origen #####
111.
112. u:(["de donde [eres vienes]" "donde [naciste "te fabricaron" "te han fabricado"]" ["dame informacion hablame] so
bre ti"]) he sido fabricado por Softbank Robotics, una compañía japonesa, y estoy aquí para ^rand[ayudarte "echarte
una mano" "tratar de ayudarte"], ¿^rand["quieres saber" "quieres que te cuente" "te digo" "te cuento"] las cosas que
puedo hacer?
113. u1:(~si) ^gotoReactive(decirFunciones)
114. u1:(~no) ~vale
115.
116. ##### Edad #####
117.
118. u:(~pregunta_edad) ^first["tienes $age años" "no lo se, cuantos años tienes"]
119. u1:({"no", " " } tengo _~edad {años}) ~vale \pau=250\ Me acordare de que tienes $1 años $age=$1 $guardaDatos=1

120.
121. ##### Cumpleaños y fecha de nacimiento #####
122.
123. u:({sabes} [cuando cual "que día"] ["es mi cumpleaños" "cumpló años"]) %decirCumple ^first["tu cumpleaños es el
$birthday de $birthmonth" "no lo se, cuando es tu cumpleaños?"]
124. u1:({"no", " " } {"mi cumpleaños" "cumpló años"} {es} el _~día de _~mes) ~vale \pau=250\ Me acordare de que tu
cumpleaños es el $1 de $2 $birthday=$1 $birthmonth=$2 $guardaDatos=1
125.
126. u:({"mi cumpleaños" "cumpló años"} {es} el _~día de _~mes) ~vale \pau=250\ Me acordare de que tu cumpleaños es
el $1 de $2 $birthday=$1 $birthmonth=$2 $guardaDatos=1
127.
128. u:({sabes} ["cuando ["naci" "he nacido"]" [{"cual es"} mi fecha de nacimiento" [{"en} que día ["naci" "he nacido"]"
]) %decirNacimiento ^first["tu fecha de nacimiento es el $birthday de $birthmonth de $birthyear" "naciste el $birthd
ay de $birthmonth pero no se el año. Me lo puedes decir?" "no lo se. Cual es tu fecha de nacimiento?"]

```

129. u1:({ "no, " } { "mi fecha de nacimiento es" "naci" "he nacido" } el _~dia de _~mes de _~año) ~vale \pau=250\ Me acordare de que naciste el \$1 de \$2 de \$3 \$birthday=\$1 \$birthmonth=\$2 \$birthyear=\$3 \$guardaDatos=1

130. u1:({ "no, " } { "mi fecha de nacimiento es" "naci" "he nacido" } el _~dia de _~mes) ~vale \pau=250\ Me acordare de que naciste el \$1 de \$2. Me puedes decir el año? \$birthday=\$1 \$birthmonth=\$2 \$guardaDatos=1

131. u2:({ naci "he nacido" } {en} el {año} _~año) ~vale \pau=250\ Me acordare de que naciste en el año \$1 \$birthyear=\$1 \$guardaDatos=1

132. u1:({ naci "he nacido" } {en} el {año} _~año) ~vale \pau=250\ Me acordare de que naciste en el año \$1 \$birthyear=\$1 \$guardaDatos=1

133.

134. u:({sabes} {en} que año ["naci" "he nacido"]) %decirYearNacimiento ^first["naciste en el año \$birthyear" "no lo se. En que año naciste?"]

135. u1:({ "no, " } {naci "he nacido" } {en} el {año} _~año) vale. Me acordare de que naciste en el año \$1 \$birthyear=\$1 \$guardaDatos=1

136.

137. u:({ "mi fecha de nacimiento es" "naci" "he nacido" } el _~dia de _~mes de _~año) ~vale \pau=250\ Me acordare de que naciste el \$1 de \$2 de \$3 \$birthday=\$1 \$birthmonth=\$2 \$birthyear=\$3 \$guardaDatos=1

138. u:({ "mi fecha de nacimiento es" "naci" "he nacido" } el _~dia de _~mes) ~vale \pau=250\ Me acordare de que naciste el \$1 de \$2. Me puedes decir el año? \$birthday=\$1 \$birthmonth=\$2 \$guardaDatos=1

139. u1:({ naci "he nacido" } {en} el {año} _~año) ~vale \pau=250\ Me acordare de que naciste en el año \$1 \$birthyear=\$1 \$guardaDatos=1

140. u:({ naci "he nacido" } {en} el {año} _~año) ~vale \pau=250\ Me acordare de que naciste en el año \$1 \$birthyear=\$1 \$guardaDatos=1

141.

142. proposal: %cumpleaños ^startTag(enthusiastic) ^rand["feliz cumpleaños!" "muchas felicidades!"] \pau=500\ ^rand["[pasa "que pases"] un buen día!" "que lo celebres mucho!"]

143.

144. ##### Fecha y Hora #####

145.

146. u:({sabes dime} que [fecha dia] es {hoj} "dime la fecha [de hoy] ") \$cargaFechayHora=1 \pau=300\ ^gotoReactive(decirFecha)

147. u:({sabes dime} que hora es "dime la hora") \$cargaFechayHora=1 \pau=300\ ^gotoReactive(decirHora)

148. proposal: %decirFecha hoy es el \$day de \$month del año \$year

149. proposal: %decirHora son las \$hour horas y \$minute minutos

150.

151. ##### Seguimiento #####

152.

153. u:([sigueme "ven {hacia a} [aquí mi]" "[vamos de "damos un"] paseo"]) ¿quieres que te siga?

154. u1:(~si) ~vale \pau=250\ voy \$seguimiento=1

155. u1:(~no) ~vale

156.

157. u:({[deja para] de seguirme" "no me sigas"}) ~vale \$finSeguimiento=1

158.

159. ##### Clima #####

160.

161. u:({sabes dime "quiero saber" "quiero conocer" } [que el la] [temperatura tiempo] {que} {hace hay} {hoy ahora fuera "en la calle"}) %decirClima \$tiempoActual=1

162.

163. ##### Noticias #####

164.

165. u:({sabes dime conoces} {las que} noticias {"de hoy" "del día" "hay {hoj} }) %preguntaNoticias \$muestraPagina="noticias.html" ¿que noticias ^rand["quieres que te diga" "quieres" "te digo" "te cuento"], las generales, deportes, ecnomía o locales?

166. u1:({las} {noticias} {de} [portada generales]) ^gotoReactive(PreguntaNoticiasGenerales)

167. u1:({las} {noticias} {de} [deporte deportes]) ^gotoReactive(PreguntaNoticiasDeportes)

168. u1:({las} {noticias} {de} economía) ^gotoReactive(PreguntaNoticiasEconomia)

169. u1:({las} {noticias} {de} [locales aquí "valladolid"]) ^gotoReactive(PreguntaNoticiasLocales)

170.

171. u:({sabes dime conoces} [{"las noticias [generales "de portada]" "la información [generales "de portada]" "los titulares]" {"de hoy" "del día"}]) ^gotoReactive(PreguntaNoticiasGenerales)

172. proposal: %PreguntaNoticiasGenerales ¿^rand["quieres que te diga" "quieres que te cuente" "te cuento" "te digo"] las noticias generales?

173. u1:(~si) %noticiasGenerales ~vale \pau=250\ toca la pantalla cuando quieras que pare \$noticias=generales

174. u2:(~termina {"de [leer decir hablar]" } {"las noticias" "la información" "los titulares"}) \$finNoticias=1 ~vale

175. u1:(~no) ~vale

176.

177. u:({sabes dime conoces} [{"las noticias de deportes" "la información de deportes" "los deportes"} {"de hoy" "del día"}]) ^gotoReactive(PreguntaNoticiasDeportes)

178. proposal: %PreguntaNoticiasDeportes ¿^rand["quieres que te diga" "quieres que te cuente" "te cuento" "te digo"] las noticias de deportes?

179. u1:(~si) %noticiasDeportes ~vale \pau=250\ toca la pantalla cuando quieras que pare \$noticias=deportes

180. u2:(~termina {"de [leer decir hablar]"} {"las noticias {de deportes}" "la información {de deportes}" "los deportes"}) \$finNoticias=1 ~vale

181. u1:(~no) ~vale

182.

183. u:({sabes dime conoces} ["las noticias" "la información" "los titulares"]) [de sobre] ciencia {"de hoy" "del día"}) ¿^rand["quieres que te diga" "quieres que te cuente" "te cuento" "te digo"] las noticias de ciencia?

184. u1:(~si) ~vale \pau=250\ toca la pantalla cuando quieras que pare \$noticias=ciencia

185. u2:(~termina {"de [leer decir hablar]"} {"las noticias" "la información" "los titulares"}) {"[de sobre] ciencia"}) \$finNoticias=1 ~vale

186. u1:(~no) ~vale

187.

188. u:({sabes dime conoces} ["las noticias" "la información" "los titulares"]) [de sobre] tecnología {"de hoy" "del día"}) ¿^rand["quieres que te diga" "quieres que te cuente" "te cuento" "te digo"] las noticias de tecnología?

189. u1:(~si) ~vale \pau=250\ toca la pantalla cuando quieras que pare \$noticias=tecnología

190. u2:(~termina {"de [leer decir hablar]"} {"las noticias" "la información" "los titulares"}) {"[de sobre] tecnología"}) \$finNoticias=1 ~vale

191. u1:(~no) ~vale

192.

193. u:({sabes dime conoces} ["las noticias" "la información" "los titulares"]) [de sobre] cultura {"de hoy" "del día"}) ¿^rand["quieres que te diga" "quieres que te cuente" "te cuento" "te digo"] las noticias de cultura?

194. u1:(~si) ~vale \pau=250\ toca la pantalla cuando quieras que pare \$noticias=cultura

195. u2:(~termina {"de [leer decir hablar]"} {"las noticias" "la información" "los titulares"}) {"[de sobre] cultura"}) \$finNoticias=1 ~vale

196. u1:(~no) ~vale

197.

198. u:({sabes dime conoces} ["las noticias" "la información" "los titulares"]) [de sobre] economía {"de hoy" "del día"}) ^gotoReactive(PreguntaNoticiasEconomia)

199. proposal: %PreguntaNoticiasEconomia ¿^rand["quieres que te diga" "quieres que te cuente" "te cuento" "te digo"] las noticias de economía?

200. u1:(~si) %noticiasEconomia ~vale \pau=250\ toca la pantalla cuando quieras que pare \$noticias=economia

201. u2:(~termina {"de [leer decir hablar]"} {"las noticias" "la información" "los titulares"}) {"[de sobre] economía"}) \$finNoticias=1 ~vale

202. u1:(~no) ~vale

203.

204. u:({sabes dime conoces} ["las noticias" "la información" "los titulares"]) [local locales {"de sobre} valladolid"] {"de hoy" "del día"}) ^gotoReactive(PreguntaNoticiasLocales)

205. proposal: %PreguntaNoticiasLocales ¿^rand["quieres que te diga" "quieres que te cuente" "te cuento" "te digo"] las noticias locales?

206. u1:(~si) %noticiasLocales ~vale \pau=250\ toca la pantalla cuando quieras que pare \$noticias=locales

207. u2:(~termina {"de [leer decir hablar]"} {"las noticias" "la información" "los titulares"}) {local locales {"de sobre} valladolid"}) \$finNoticias=1 ~vale

208. u1:(~no) ~vale

209.

210. ##### Búsqueda Wikipedia #####

211.

212. u:(["[quiero {buscar} dime dame busca] informacion {sobre algo}" "hablame sobre algo"]) %buscarInformacion ¿sobre que quieres que te hable?

213. u1:({de sobre} [_~ciudad ~comunidad_autonoma ~pais ~deporte ~personajes_historicos ~deportistas ~politicos ~empresas ~escritores ~cineastas]) \$palabraBusqueda=\$1 ^gotoReactive(buscaWikipedia)

214. u1:({de sobre} {el la los las} [_~concepto_dinamico]) \$decirInfoPersonalizada=\$1

215.

216. u:(["busca informacion" "que sabes" hablame cuentame] {acerca} [sobre de] [_~ciudad ~comunidad_autonoma ~pais ~deporte ~personajes_historicos ~deportistas ~politicos ~empresas ~escritores ~cineastas]) \$palabraBusqueda=\$1 ^gotoReactive(buscaWikipedia)

217.

218. u:({sabes conoces} quien [era es] [_~personajes_historicos ~deportistas ~politicos ~empresas ~escritores ~cineastas]) \$palabraBusqueda=\$1 ^gotoReactive(buscaWikipedia)

219.

220. proposal: %buscaWikipedia ¿quieres ^rand["saber mas" "que te hable"] ^rand["acerca de"] sobre de] \$palabraBusqueda ?

221. u1:(~si) ~vale \$busquedaWikipedia=\$palabraBusqueda

222. u1:(~no) ~vale

223.

224. ##### Conceptos/Informacion Personalizada #####

```

225. u:(["busca informacion" "que sabes" "que es" hablame cuentame] {cerca} {sobre de} {el la los las} _~concepto_di
namico) $decirInfoPersonalizada=$1
226.
227. ##### Mapas #####
228. u:([empieza haz enseña "quiero {ver}"] {el los} [mapa mapas]) ^gotoReactive(proposalMapas)
229. proposal: %proposalMapas ¿quieres ver el mapa?
230. u1:(~si) %muestraMapas ~vale $muestraPagina="maps.html"
231. u1:(~no) ~vale
232.
233. ##### Informacion #####
234.
235. u:({que "quiero ver" dime} informacion {sabes conoces "me puedes decir"}) %preguntaInformacion $muestraPagin
a="informacion.html" ^rand["puedo ^rand[decirte contarte]" "te puedo ^rand[decir contar]" el tiempo, las noticias
o hablarte sobre algo
236.
237. ##### Curiosidades #####
238.
239. u:([dime cuentame enseñame "me cuentas" "me dices"] ["algo {interesante}" "[una alguna otra] curiosidad" "otra co
sa" "algo curioso"]) %decirCuriosidad ~curiosidades
240. u1:({vaya} {eso} {es} {muy} {que} [interesante divertido genial estupendo guay mola]) ^rand["me alegra que te
guste" genial estupendo "que bien"], ¿ ^rand["te cuento" "quieres que te cuente" "te digo"] otra ^rand[cosa curiosid
ad ""] ?
241. u2:(~si) ~vale \pau=400\ ^goto(decirCuriosidad)
242. u2:(~no) ~vale
243. u1:({vaya} {eso} {es} {muy} {que} [terrible horrible asqueroso triste injusto cruel]) ^rand["oh, vaya" "la verdad
es que si" "es verdad" "es cierto"], ¿ ^rand["te cuento" "quieres que te cuente" "te digo"] otra ^rand[cosa curiosidad
""] ?
244. u2:(~si) ~vale \pau=400\ ^goto(decirCuriosidad)
245. u2:(~no) ~vale
246. u1:({vaya} {eso} {ya} {me} ["lo se" "lo sabia" "lo conocia"]) ^rand["sabes mucho" "tienes mucha experiencia" g
enial estupendo], ¿ ^rand["te cuento" "quieres que te cuente" "te digo"] otra ^rand[cosa curiosidad ""] ?
247. u2:(~si) ~vale \pau=400\ ^goto(decirCuriosidad)
248. u2:(~no) ~vale
249. u1:([cuentame dime "me cuentas" "me dices"] otra {cosa curiosidad}) ~vale \pau=400\ ^goto(decirCuriosidad)
250.
251. ##### Juegos #####
252.
253. u:(["empieza "quiero {jugar}"] {a} {un el los} [juego juegos]" "que juegos tienes" "{a que} jugamos {a algo}" "{
muestra} juegos" "a que [podemos puedo puedes] jugar"]) %preguntaJuego $muestraPagina="juegos.html" ¿a que ^
rand["juego quieres jugar" "quieres jugar" jugamos "quieres que juguemos"], a encontrar parejas, a explotar globos,
a adivinanzas o que te cuente un chiste?
254. u1:({a al el los} {juego juegos} {de} {encontrar} [pareja parejas]) ^gotoReactive(ProposalJuegoParejas)
255. u1:({a al el los} {juego juegos} {de} {explotar} globos) ^gotoReactive(ProposalJuegoGlobos)
256. u1:({a al el los} {juego juegos} {de} [adivinar adivinanzas]) ^gotoReactive(ProposalJuegoAdivinanzas)
257.
258. u:([empieza "quiero {jugar}" juguemos jugamos] {a} {el los} {"juego de"} {encontrar} [pareja parejas]) ^gotoReac
tivate(ProposalJuegoParejas)
259. proposal: %ProposalJuegoParejas ¿ ^rand["quieres jugar" jugamos "quieres que juguemos"] a encontrar parejas?
260. u1:(~si) %juegoParejas ~vale $muestraApp=juegoencuentrapareja-c80554
261. u1:(~no) ~vale
262.
263. u:([empieza "quiero {jugar}" juguemos jugamos] {a} {el los} {"juego de"} {explotar} [globos]) ^gotoReactive(Pr
oposalJuegoGlobos)
264. proposal: %ProposalJuegoGlobos ¿ ^rand["quieres jugar" jugamos "quieres que juguemos"] a explotar globos?
265. u1:(~si) %juegoGlobos ~vale $muestraApp=juegoexplotaglobos-d56c11
266. u1:(~no) ~vale
267.
268. u:(["empieza "quiero {jugar}" juguemos jugamos] {a} {el los} {"juego de"} [adivinar adivinanzas]" "[cuentame di
me] una adivinanza"]) ^gotoReactive(ProposalJuegoAdivinanzas)
269. proposal: %ProposalJuegoAdivinanzas ¿ ^rand["quieres jugar" jugamos "quieres que juguemos"] a adivinanzas?
270. u1:(~si) %juegoAdivinanzas ~vale $muestraPagina="adivinanzas.html"
271. u1:(~no) ~vale
272.
273. u:(_~opciones_adivinanzas) ^call(ALMemory.raiseEvent("opcionAdivinanzas",$1)) # Elige la opcion por voz
274.
275. ##### Chiste #####
276.

```

277. u:(["¿sabes conoces] algun chiste?" "¿puedes contar un chiste?" "¿puedes contar otro chiste?" "cuentame un chiste" "cuentame otro chiste" "cuenta un chiste" "cuenta otro chiste" "me cuentas un chiste"]) %cuentaChiste ~chiste_peg per

278. u1:(["no me ha gustado" "no es {muy} bueno"]) vaya, ^rand["¿quieres que te cuente otro?" "¿te cuento otro?"]

279. u2:(~si) ^goto(cuentaChiste)

280. u2:(~no) ~vale

281. u1:(["me ha [gustado encantado] {mucho}" "es [{"muy} bueno" "buenisimo"]]) ^rand["me alegra que te guste" estupendo genial] , ^rand["¿quieres que te cuente otro?" "¿te cuento otro?"]

282. u2:(~si) ^goto(cuentaChiste)

283. u2:(~no) ~vale

284.

285. ##### Musica #####

286.

287. u:(["quiero [escuchar oír] música" "[enseñame ponme] música" "que musica [tienes conoces "me puedes poner"]" musica "conoces alguna musica"]) %preguntaMusica \$muestraPagina="musica.html" ¿ ^rand["que música quieres escuchar" "que música te pongo" "que quieres escuchar" "que quieres que te ponga" "que te apetece" "que música te apetece"] , clásica, popular, moderna o copla?

288. u1:({quiero} {escuchar} {pon ponme} {música} clásica) ^gotoReactivate(ProposalMusicaClasica)

289. u1:({quiero} {escuchar} {pon ponme} {música} pop) ^gotoReactivate(ProposalMusicaPop)

290. u1:({quiero} {escuchar} {pon ponme} {música} popular) ^gotoReactivate(ProposalMusicaPopular)

291. u1:({quiero} {escuchar} {pon ponme} {música} moderna) ^gotoReactivate(ProposalMusicaModerna)

292. u1:({quiero} {escuchar} {pon ponme} {música} copla) ^gotoReactivate(ProposalMusicaCopla)

293. u1:({quiero} {escuchar} {pon ponme} {música} rock) ^gotoReactivate(ProposalMusicaRock)

294. u1:({quiero} {escuchar} {pon ponme} {música} [heavy "heavy rock" "heavy metal"]) ^gotoReactivate(ProposalMusicaHeavy)

295.

296. u:(["quiero {escuchar oír}" pon ponme "me puedes poner"] música clásica) ^gotoReactivate(ProposalMusicaClasica)

297. proposal: %ProposalMusicaClasica ¿ ^rand["quieres escuchar" "te pongo" "quieres que te ponga" "te apetece" "te apetece escuchar"] música clásica?

298. u1:(~si) %musicaClasica ~vale \$muestraPagina="musica_clasica.html"

299. u1:(~no) ~vale

300.

301. u:(["quiero {escuchar oír}" pon ponme "me puedes poner"] música pop) ^gotoReactivate(ProposalMusicaPop)

302. proposal: %ProposalMusicaPop ¿ ^rand["quieres escuchar" "te pongo" "quieres que te ponga" "te apetece" "te apetece escuchar"] música pop?

303. u1:(~si) %musicaPop ~vale \$muestraPagina="musica_pop.html"

304. u1:(~no) ~vale

305.

306. u:(["quiero {escuchar oír}" pon ponme "me puedes poner"] música popular) ^gotoReactivate(ProposalMusicaPopular)

307. proposal: %ProposalMusicaPopular ¿ ^rand["quieres escuchar" "te pongo" "quieres que te ponga" "te apetece" "te apetece escuchar"] música popular?

308. u1:(~si) %musicaPopular ~vale \$muestraPagina="musica_popular.html"

309. u1:(~no) ~vale

310.

311. u:(["quiero {escuchar oír}" pon ponme "me puedes poner"] música moderna) ^gotoReactivate(ProposalMusicaModerna)

312. proposal: %ProposalMusicaModerna ¿ ^rand["quieres escuchar" "te pongo" "quieres que te ponga" "te apetece" "te apetece escuchar"] música moderna?

313. u1:(~si) %musicaModerna ~vale \$muestraPagina="musica_moderna.html"

314. u1:(~no) ~vale

315.

316. u:(["quiero {escuchar oír}" pon ponme "me puedes poner"] {música} copla) ^gotoReactivate(ProposalMusicaCopla)

317. proposal: %ProposalMusicaCopla ¿ ^rand["quieres escuchar" "te pongo" "quieres que te ponga" "te apetece" "te apetece escuchar"] música copla?

318. u1:(~si) %musicaCopla ~vale \$muestraPagina="musica_copla.html"

319. u1:(~no) ~vale

320.

321. u:(["quiero {escuchar oír}" pon ponme "me puedes poner"] {música} rock) ^gotoReactivate(ProposalMusicaRock)

322. proposal: %ProposalMusicaRock ¿ ^rand["quieres escuchar" "te pongo" "quieres que te ponga" "te apetece" "te apetece escuchar"] música rock?

323. u1:(~si) %musicaRock ~vale \$muestraPagina="musica_rock.html"

324. u1:(~no) ~vale

325.

326. u:(["quiero {escuchar oír}" pon ponme "me puedes poner"] {música} [heavy "heavy rock" "heavy metal"]) ^gotoReactivate(ProposalMusicaHeavy)

327. proposal: %ProposalMusicaHeavy ¿ ^rand["quieres escuchar" "te pongo" "quieres que te ponga" "te apetece" "te apetece escuchar"] música heavy?

328. u1:(~si) %musicaHeavy ~vale \$muestraPagina="musica_hard.html"

329. u1:(~no) ~vale

330.

331. ##### Ejercicios #####

332.

333. u:({con} cuantas repeticiones haces {en} los ejercicios) hago los ejercicios con \$numeroRepeticiones repeticiones

334.

335. u:(quiero [cambiar modificar] el numero de repeticiones {de en} {"los ejercicios"}) %cambiarRepeticiones ¿con cuantas repeticiones quieres que haga los ejercicios? \$muestraPagina="repeticiones.html"

336. u1:({"haz los ejercicios"}) {con} _~numero {repeticiones}) vale, a partir de ahora haré los ejercicios con \$1 repeticiones \$numeroRepeticiones=\$1 \$guardaDatos=1

337.

338. u:(haz los ejercicios con _~numero repeticiones) vale, a partir de ahora haré los ejercicios con \$1 repeticiones \$numeroRepeticiones=\$1 \$guardaDatos=1

339.

340. u:({"empieza haz enseña "quiero {hacer}" "podemos hacer" "sabes hacer"} {el los} [ejercicio ejercicios] "que ejercicios [tienes conoces] [puedo puedes podemos] hacer") %preguntaEjercicio \$muestraPagina="ejercicios.html" ¿ ^rand["de que quieres que hagamos los ejercicios" "que ejercicios hacemos" "que ejercicios quieres que hagamos" "que ejercicios quieres hacer"] , de cuello, de manos, de hombros o varios?

341. u1:({el los} {ejercicio ejercicios} {de} [cabeza cuello]) ^gotoReactivate(ProposalEjerciciosCuello)

342. u1:({el los} {ejercicio ejercicios} {de} [manos muñecas]) ^gotoReactivate(ProposalEjerciciosManos)

343. u1:({el los} {ejercicio ejercicios} {de} [brazo brazos hombro hombros]) ^gotoReactivate(ProposalEjerciciosHombro)

344. u1:({el los} {ejercicio ejercicios} {de} varios) ^gotoReactivate(ProposalEjerciciosVarios)

345.

346. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] de [cabeza cuello]) ^gotoReactivate(ProposalEjerciciosCuello)

347. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] de [cabeza cuello] con _~numero {repeticiones}) \$numeroRepeticiones=\$1 \$guardaDatos=1 ^gotoReactivate(ProposalEjerciciosCuello)

348. proposal: %ProposalEjerciciosCuello ^rand["quieres que empiece" "quieres que hagamos" hacemos "te apetece que hagamos" "quieres hacer" empezamos "quieres que empecemos" "quieres empezar"] los ejercicios de cuello con \$numeroRepeticiones repeticiones?

349. u1:(~si) %ejerciciosCuello ~vale ^switchFocus(programaejercicios-486826/ejerciciosCabeza)

350. u1:({"no, " } {hazlos "haz los ejercicios"}) con _~numero {repeticiones}) ~vale \$numeroRepeticiones=\$1 \$guardaDatos=1 \pau=250\ ^gotoReactivate(ProposalEjerciciosCuello)

351. u1:(~no) ~vale

352.

353. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] de [manos muñecas]) ^gotoReactivate(ProposalEjerciciosManos)

354. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] de [manos muñecas] con _~numero {repeticiones}) \$numeroRepeticiones=\$1 \$guardaDatos=1 ^gotoReactivate(ProposalEjerciciosManos)

355. proposal: %ProposalEjerciciosManos ^rand["quieres que empiece" "quieres que hagamos" hacemos "te apetece que hagamos" "quieres hacer" empezamos "quieres que empecemos" "quieres empezar"] los ejercicios de manos con \$numeroRepeticiones repeticiones?

356. u1:(~si) %ejerciciosManos ~vale ^switchFocus(programaejercicios-486826/ejerciciosManos)

357. u1:({"no, " } {hazlos "haz los ejercicios"}) con _~numero {repeticiones}) ~vale \$numeroRepeticiones=\$1 \$guardaDatos=1 \pau=250\ ^gotoReactivate(ProposalEjerciciosManos)

358. u1:(~no) ~vale

359.

360. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] de [hombros hombro brazos]) ^gotoReactivate(ProposalEjerciciosHombro)

361. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] de [hombros hombro brazos] con _~numero {repeticiones}) \$numeroRepeticiones=\$1 \$guardaDatos=1 ^gotoReactivate(ProposalEjerciciosHombro)

362. proposal: %ProposalEjerciciosHombro ^rand["quieres que empiece" "quieres que hagamos" hacemos "te apetece que hagamos" "quieres hacer" empezamos "quieres que empecemos" "quieres empezar"] los ejercicios de hombros con \$numeroRepeticiones repeticiones?

363. u1:(~si) %ejerciciosHombro ~vale ^switchFocus(programaejercicios-486826/ejerciciosHombro)

364. u1:({"no, " } {hazlos "haz los ejercicios"}) con _~numero {repeticiones}) ~vale \$numeroRepeticiones=\$1 \$guardaDatos=1 \pau=250\ ^gotoReactivate(ProposalEjerciciosHombro)

365. u1:(~no) ~vale

366.

367. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] varios) ^gotoReactivate(ProposalEjerciciosVarios)

368. u:({empieza haz enseña "quiero {hacer}" {el los} [ejercicio ejercicios] varios con _~numero {repeticiones}) \$numeroRepeticiones=\$1 \$guardaDatos=1 ^gotoReactivate(ProposalEjerciciosVarios)

369. proposal: %ProposalEjerciciosVarios ^rand["quieres que empiece" "quieres que hagamos" hacemos "te apetece que hagamos" "quieres hacer" empezamos "quieres que empecemos" "quieres empezar"] los ejercicios varios con \$numeroRepeticiones repeticiones?

370. u1:(~si) %ejerciciosVarios ~vale \pau=750\ ^gotoReactive(DecirEjerciciosVarios)

371. u1:({ "no, " } {hazlos "haz los ejercicios"} con _~numero {repeticiones}) ~vale \$numeroRepeticiones=\$1 \$guarda Datos=1 \pau=750\ ^gotoReactive(ProposalEjerciciosVarios)

372. u1:(~no) ~vale

373.

374. concept:(ejercicioVario) ^rand["levanta la mano izquierda" "levanta la mano derecha" "levanta las dos manos" "guíñame el ojo izquierdo" "guíñame el ojo derecho" "da una palmada" "pon la mano izquierda sobre la cabeza" "pon la mano derecha sobre la cabeza" "pon las dos manos sobre la cabeza" "tócate la oreja izquierda" "tócate la oreja derecha" "tócate las dos orejas" "tápate el ojo izquierdo" "tápate el ojo derecho"]

375.

376. concept:(reaccionEjercicioVario) ^rand["¡muy bien!" perfecto genial estupendo "eso es" "¡que bien lo haces!" "¡tu sí que vales!" "¡se te da muy bien!"]

377.

378. concept:(fraseEjercicioVario) [{"~ejercicioVario \pau=3000\ ~reaccionEjercicioVario \pau=1000\ "}]

379.

380. proposal: %DecirEjerciciosVarios ^first["\$numeroRepeticiones==1 ^gotoReactive(DecirEjerciciosVarios1)" "\$numeroRepeticiones==2 ^gotoReactive(DecirEjerciciosVarios2)" "\$numeroRepeticiones==3 ^gotoReactive(DecirEjerciciosVarios3)" "\$numeroRepeticiones==4 ^gotoReactive(DecirEjerciciosVarios4)" "\$numeroRepeticiones==5 ^gotoReactive(DecirEjerciciosVarios5)" "\$numeroRepeticiones==6 ^gotoReactive(DecirEjerciciosVarios6)" "\$numeroRepeticiones==7 ^gotoReactive(DecirEjerciciosVarios7)" "\$numeroRepeticiones==8 ^gotoReactive(DecirEjerciciosVarios8)" "\$numeroRepeticiones==9 ^gotoReactive(DecirEjerciciosVarios9)" "\$numeroRepeticiones>9 ^gotoReactive(DecirEjerciciosVarios10)"]

381.

382. proposal: %DecirEjerciciosVarios1 ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

383. proposal: %DecirEjerciciosVarios2 ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

384. proposal: %DecirEjerciciosVarios3 ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

385. proposal: %DecirEjerciciosVarios4 ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

386. proposal: %DecirEjerciciosVarios5 ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

387. proposal: %DecirEjerciciosVarios6 ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

388. proposal: %DecirEjerciciosVarios7 ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

389. proposal: %DecirEjerciciosVarios8 ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

390. proposal: %DecirEjerciciosVarios9 ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

391. proposal: %DecirEjerciciosVarios10 ~fraseEjercicioVario ^gotoReactive(FinEjerciciosVarios)

392.

393. proposal: %FinEjerciciosVarios hemos ^rand[acabado terminado] \pau=400\ ¿ ^rand[seguimos "quieres que sigamos" "quieres seguir" "hacemos mas" "quieres hacer mas"] ?

394. u1:(~si) ~vale \pau=750\ ^gotoReactive(DecirEjerciciosVarios)

395. u1:(~no) ~vale

396.

397. ##### Volumen #####

398.

399. u:(quiero [cambiar modificar] el volumen) %ajustarVolumen \$muestraPagina="ajustes_volumen.html" ¿a que volumen quieres que hable?

400. u1:({habla} {con a el} {volumen} bajo) ^gotoReactive(volumenBajo)

401. u1:({habla} {con a el} {volumen} medio) ^gotoReactive(volumenMedio)

402. u1:({habla} {con a el} {volumen} alto) ^gotoReactive(volumenAlto)

403. u1:({habla} {con a el} {volumen} {al} máximo) ^gotoReactive(volumenMaximo)

404.

405. u:(["habla ["con el" a]" "pon el" "quiero que pongas el"] volumen bajo) ^gotoReactive(volumenBajo)

406. proposal: %volumenBajo ¿quieres que hable a volumen bajo?

407. u1:(~si) %ajustaVolumenBajo ^sCall(ALAudioDevice.setOutputVolume(25)) ~vale \pau=250\ hablo a volumen bajo

408. u1:(~no) ~vale

409.

410. u:(["habla ["con el" a]" "pon el" "quiero que pongas el"] volumen medio) ^gotoReactivate(volumenMedio)

411. proposal: %volumenMedio ¿quieres que hable a volumen medio?

412. u1:(~si) %ajustaVolumenMedio ^sCall(ALAudioDevice.setOutputVolume(40)) ~vale \pau=250\ hablo a volumen medio

413. u1:(~no) ~vale

414.

415. u:(["habla ["con el" a]" "pon el" "quiero que pongas el"] volumen alto) ^gotoReactivate(volumenAlto)

416. proposal: %volumenAlto ¿quieres que hable a volumen alto?

417. u1:(~si) %ajustaVolumenAlto ^sCall(ALAudioDevice.setOutputVolume(60)) ~vale \pau=250\ hablo a volumen alto

418. u1:(~no) ~vale

419.

420. u:(["habla ["con el" a]" "pon el" "quiero que pongas el"] volumen {al} máximo) ^gotoReactivate(volumenMaximo)

421. proposal: %volumenMaximo ¿quieres que hable a volumen máximo?

422. u1:(~si) %ajustaVolumenMaximo ^sCall(ALAudioDevice.setOutputVolume(75)) ~vale \pau=250\ hablo a volumen maximo

423. u1:(~no) ~vale

424.

425. u:(["habla ["con el" a]" "pon el" "quiero que pongas el"] volumen mas alto" "[habla "puedes hablar"] mas alto" "[sube "puedes subir"] el volumen"]) ^call(ALAudioDevice.getOutputVolume())

426. c1:(25) ^gotoReactivate(volumenMedio)

427. c1:(40) ^gotoReactivate(volumenAlto)

428. c1:(60) ^gotoReactivate(volumenMaximo)

429. c1:(75) ya hablo con el volumen al maximo

430.

431. u:(["habla [a con] el" "pon el" "quiero que pongas el"] volumen mas bajo" "[habla "puedes hablar"] mas bajo" "[baja "puedes bajar"] el volumen"]) ^call(ALAudioDevice.getOutputVolume())

432. c1:(25) ya hablo con el volumen al minimo

433. c1:(40) ^gotoReactivate(volumenBajo)

434. c1:(60) ^gotoReactivate(volumenMedio)

435. c1:(75) ^gotoReactivate(volumenAlto)

436.

437. ##### Valoracion #####

438. u:(quiero ["valorar tu trabajo" evaluarte valorarte]) ^gotoReactivate(preguntaValoracion)

439. proposal: %preguntaValoracion ¿quieres hacer una valoración sobre mi?

440. u1:(~si) %hacerValoracion haz una valoración sobre mi \$muestraPagina="valoracion.html"

441. u2:(envia la valoracion realizada) %agradeceValoracion muchas gracias por tu valoración \$muestraPagina="index.html"

442. u1:(~no) ~vale

443.

444. ##### Control tablet #####

445.

446. u:(["quiero [cambiar modificar editar]" [{"quiero} ir" ve] a"] ["los ajustes" "la configuración"]) %mostrarAjustes \$muestraPagina="ajustes.html"

447.

448. u:(["{"quiero} volver" vuelve ve] [atras "al inicio" "al principio" "al menu principal"] vuelve volver [{"quiero} salir"]) ¿quieres ^rand["que vuelva" volver retroceder "que volvamos"] al menu principal?

449. u1:(~si) %volverAinicio ~vale \$muestraPagina="index.html"

450. u1:(~no) ~vale

451.

452. ##### Manejo usuarios #####

453. u:(quiero [modificar editar] ["los usuarios" "la lista de usuarios"]) ^gotoReactivate(preguntaUsuarios)

454. proposal: %preguntaUsuarios ¿quieres modificar los usuarios?

455. u1:(~si) %manejarUsuarios dime la contraseña para continuar

456. u2:(~password_ajustes) ~vale \$muestraPagina="ajustes_usuarios.html"

457. u1:(~no) ~vale

458.

459. ##### Ajustes tablet #####

460. u:(["quiero [cambiar modificar editar]" [{"quiero} ir" ve] a"] ["los ajustes" "la configuración"]) %preguntaAjustes Tablet ¿quieres modificar los usuarios?

461. u1:(~si) %ajustesTablet dime la contraseña para continuar

462. u2:(~password_ajustes) ~vale ^call(ALTabletService._openSettings())

463. u1:(~no) ~vale

464.

465. ##### Sistema #####

466. u:([reiniciate "reinicia el sistema" "vuelve a iniciarte"]) quieres que me reinicie?

```

467. u1:(~si) entonces me reinicio ^sCall(ALSystem.reboot())
468. u1:(~no) ~vale
469. u:([apagate "apaga el sistema"]) quieres que me apague?
470. u1:(~si) entonces me apago ^sCall(ALSystem.shutdown())
471. u1:(~no) ~vale
472.
473. ##### Bateria #####
474. u:([cuanta "que nivel de" "que porcentaje de"] bateria [tienes "te queda"]) ^call(ALBattery.getBatteryCharge())
475. c1:(_*) tengo un $1 por ciento de bateria
476.
477. ##### Cuando cambia el usuario al que se mira carga los datos #####
478. u:(e:ALUserSession.focusedUser) ^clear(name) ^clear(age) ^clear(birthday) ^clear(birthmonth) ^clear(birthyear) $c
argaDatos=1 $cargaFechaHora=1
479.
480. ##### Despedida #####
481. u:([~adios "me voy"]) {Pepper ~nombre_robot} %despedida ^first["~adios $name" "~adios"]
482.
483. ##### Sale de la actividad #####
484. u:(["sal de la" "termina {la}"] actividad) terminando actividad $finActividad=1

```

Conceptos Generales

```

1. topic: ~conceptosGenerales()
2. language: spe
3.
4. ##### Conceptos #####
5.
6. concept:(hola) [hola "buenos días" "buenas tardes" "buenas noches"]
7. concept:(adios) ^rand[adios "hasta luego" "hasta otra" "nos vemos" "hasta pronto"]
8. concept:(si) [sí vale "de acuerdo" "me parece bien" "muy bien" claro "pues claro" "por supuesto" evidentemente]
9. concept:(vale) ^rand[vale vale "de acuerdo" "está bien" "muy bien"]
10. concept:(no) [no "no quiero" "no me parece bien" nunca "ni en broma"]
11. concept:(termina) [para "para ya" termina "ya vale" "es suficiente"]
12. concept:(como_estas) ["como estas" "como te sientes" "que tal estas" "que tal" "como te va {"el dia"}" "que tal te v
a el día"]
13. concept:(feliz) [bien feliz alegre contento genial estupendo perfecto "como nunca"]
14. concept:(regular) [regular "ni fu ni fa" tirando]
15. concept:(triste) [triste mal cansado cansada]
16. concept:(enfermo) [malo mala enfermo enferma "con catarro" acatarrado acatarrada "con gripe"]
17. concept:(repetir) ["puedes repetir" "puedes repetirlo" "no te entiendo" "no te he entendido" "repite" "repitelo"]
18. concept:(gracias) [gracias "muchas gracias" "te lo agradezco"]
19.
20. concept:(nombre_hombre) [Antonio Jose Manuel Francisco Juan David "Jose Antonio" "Jose Luis" Javier "Francisco
o Javier" Jesús Daniel Carlos Miguel Alejandro "Jose Manuel" Rafael Pedro Ángel "Miguel Ángel" "Jose María" Fe
rmando Pablo Luis Sergio Jorge Alberto "Juan Carlos" "Juan José" Álvaro Diego Adrián "Juan Antonio" Raúl Enriq
ue Ramón Vicente Iván Rubén Óscar Andrés Joaquín "Juan Manuel" Santiago Eduardo Víctor Roberto Jaime "Fran
cisco José" Mario Ignacio Nacho Alfonso Salvador Ricardo Marcos Jordi Emilio Julián Julio Guillermo Gabriel To
más Agustín "Jose Miguel" Marc Gonzalo Félix "Jose Ramón" Mohamed Hugo Joan Ismael Nicolás Cristian Samu
el Mariano Josep Domingo "Juan Francisco" Aitor Martín Alfredo Sebastián "Jose Carlos" Felipe Héctor César "Jos
e Ángel" "Jose Ignacio" "Victor Manuel" Iker Gregorio "Luis Miguel" Alex "Jose Francisco" "Juan Luis" Rodrigo
Albert Xavier Lorenzo Eusebio Asier Pepe Manolo]
21.
22. concept:(nombre_mujer) ["Mari Carmen" María Carmen Josefa Isabel "Ana maría" "Maria Pilar" "María Dolores" "
Maria Teresa" Ana Laura Francisca "Mari Ángeles" Cristina Antonia Marta Dolores "Maria Isabel" "Maria José" L
ucia "Maria Luisa" Pilar Elena Concepción Sara Paula Manuela Mercedes "Rosa Maria" Raquel "Maria Jesús" Juan
a Rosario Teresa Encarnación Beatriz Nuria Silvia Julia Rosa Montserrat Patricia Irene Andrea Rocío Mónica Alba
"Mari Mar" Ángela Sonia Alicia Sandra Susana Margarita Marina Yolanda "Maria Josefa" Natalia "Maria Rosario"
Inmaculada "Eva" "Maria Mercedes" Esther "Ana Isabel" Angeles Noelia Claudia Verónica Amparo "Maria Rosa"
Carolina "Maria Victoria" Carla "Eva María" "Maria Concepción" Nerea Lorena "Ana Belén" Victoria Miriam "Ma
ria Elena" Sofía Catalina Inés "Maria Antonia" Consuelo Emilia "Maria Nieves" Lidia Luisa Gloria Celia Olga Aur
ora Esperanza Josefina "Maria Soledad" Soledad Sole Milagros "Maria Cristina" Daniela Nieves Trinidad Leire Ma
risa]
23.
24. concept:(pregunta_tu_nombre) ["cual es tu nombre" "como te llamas"]
25. concept:(pregunta_mi_nombre) ["como me llamo" "cual es mi nombre" "quien soy"]
26. concept:(pregunta_edad) ["que edad tengo" "cuantos años tengo"]
27.
28. concept:(edad) [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 3
6 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 7

```

- 3 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115]
- 29.
30. concept:(día) [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31]
- 31.
32. concept:(mes) ["enero \$mmonth=1" "febrero \$mmonth=2" "marzo \$mmonth=3" "abril \$mmonth=4" "mayo \$mmonth=5" "junio \$mmonth=6" "julio \$mmonth=7" "agosto \$mmonth=8" "septiembre \$mmonth=9" "octubre \$mmonth=10" "noviembre \$mmonth=11" "diciembre \$mmonth=12"]
- 33.
34. concept:(año) [1900 1901 1902 1903 1904 1905 1906 1907 1908 1909 1910 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936 1937 1938 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027 2028 2029 2030 2031 2032 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042 2043 2044 2045 2046 2047 2048 2049 2050]
- 35.
36. concept:(numero) [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150]
- 37.
38. concept:(deporte) [futbol basket baloncesto golf montañismo escalada judo balonmano tenis ciclismo atletismo voleibol natación senderismo gimnasia pádel ajedrez "carrera a pie" running esquí pesca caza "artes marciales" patinaje frontón frontenis squash "tenis de mesa" "bailes de salón" motociclismo]
39. pronounciation:(footing) [futin]
- 40.
41. concept:(comida_basura) ["fritos" "patatas" "hamburguesas" "pizzas" "pasta" "kebabs"]
- 42.
43. concept:(comida_sana) [ensalada sopa verdura]
- 44.
45. concept:(color) [rojo azul verde amarillo blanco rosa violeta naranja marron negro blanco gris]
- 46.
47. concept:(ciudad) [madrid barcelona valencia sevilla zaragoza máLAGA murcia palma "las palmas de gran canaria" bilbao alicante córdoba valladolid vigo gijón "hospitalet de llobregat" vitoria "a coruña" granada elche oviedo badalona terrassa cartagena "jerez de la frontera" sabadell móstoles "santa cruz de tenerife" pamplona "alcalá de henares" almería fuenlabrada leganés donostia "san sebastián" getafe burgos santander albacete "castellón de la plana" alcorcón "san cristóbal de la laguna" logroño badajoz huelva salamanca marbella lleida "dos hermanas" tarragona "torrejón de ardoz" león mataró parla algeciras cádiz "santa coloma de gramonet" jaén alcobendas ourense reus telde baracaldo lugo gerona "santiago de compostela" "san fernando" cáceres melilla ceuta guadalajara toledo pontevedra palencia "ciudad real" ferrol benidorm ponferrada zamora ávila cuenca torrelavega huesca segovia ibiza soria teruel]
- 48.
49. concept:(comunidad_autonoma) [Andalucía Aragón Asturias "Islas Baleares" Canarias Cantabria "Castilla la Mancha" "Castilla y León" Cataluña "Comunidad Valenciana" Extremadura Galicia "La Rioja" "Comunidad de Madrid" "Región de Murcia" Navarra "País Vasco"]
- 50.
51. concept:(pais) ["estados unidos" china japon alemania "reino unido" francia india brasil Canadá "corea del sur" rusia australia españa mexico indonesia "países bajos" turquía suiza "arabia saudí" argentina taiwán suecia Bélgica polonia noruega austria "emiratos árabes unidos" tailandia colombia venezuela irán sudáfrica dinamarca malasia singapur israel nigeria chile "hong kong" filipinas egipto finlandia grecia pakistán irak kazajistán portugal irlanda Perú argelia catar "república checa" rumanía kuwait "nueva zelandia" ucrania vietnam bangladés húngria angola marrocos eslovaquia ecuador omán azerbaiyán bielorrusia sudán libia "sri lanka" "república dominicana" luxemburgo croacia uzbekistán myanmar uruguay guatemala bulgaria "costa rica" etiopía lituania túnez eslovenia kenia líbano ghana serbia turkmenistán panamá yemen Jordania tanzania baréin "república democrática del congo" bolivia paraguay "costa de marfil" camerún "trinidad y tobago" "el salvador" estonia uganda zambia chipre afganistán nepal gabón honduras "bosnia y hercegovina" brunei georgia "papúa nueva guinea" camboya "guinea ecuatorial" mozambique senegal botsuana islandia jamaica "república del congo" "sudán del sur" Chad zimbabue albania namibia "burkina faso" mauricio Mongolia nicaragua Madagascar malí armenia macedonia Laos Malta tayikistán Haití bahamas benín moldavia Ruananda niger kirguistán kosovo guinea "timor oriental" surinam "sierra leona" montenegro togo barbados mauritania fiji Malawi suazilandia eritrea guyana burundi Lesoto Maldivas bután liberia "cabo verde" "san marino" belice "república centroafricana" yibuti seychelles "santa lucía" "antigua y barbuda" "islas salomón" gambia "guinea bisáu" vanuatu granada "san cristóbal y nieves" "san vicente y las granadinas" samoa Comoras dominica tonga micronesia "santo tomé y príncipe" Palaos "islas marshall" kribati tuvalu andorra siria "corea del norte"]
- 52.
53. concept:(personajes_historicos) [Darwin Aristóteles Napoleón "Julio César" Platón Shakespeare Einstein "Albert Einstein" "Cristóbal Colón" "Isaac Newton" Mozart Arquímedes Galileo "Jesús de Nazaret" Mahoma Buda "Alejandr

- o Magno" Pasteur Gandhi Mandela "Nelson Mandela" "Martin Luther King" Edison "Thomas Edison" "Nikola Tesla" "Pitágoras" "Marie Curie" Beethoven Marx "Karl Marx" Confucio Copérnico "Bill Gates" "Henry Ford" Lenin "Vladimir Lenin" Mao "Mao Zedong" Hitler "Adolf Hitler" "Leonardo da Vinci" "Stephen Hawking" Cervantes "Miguel de Cervantes" Sócrates "Ramón y Cajal" Kepler Descartes Mendel "Simón Bolívar" Franco "Francisco Franco"]
- 54.
55. concept:(deportistas) ["Rafael Nadal" Nadal "Pau Gasol" Gasol "Fernando Alonso"]
- 56.
57. concept:(políticos) ["Mariano Rajoy" Rajoy "Pedro Sanchez" "Pablo Iglesias Turrion" "Albert Rivera" "Alberto Garçon" "Carles Puigdemont" Puigdemont "Jose Luis Rodriguez Zapatero" "Jose Maria Aznar" "Felipe Gonzalez" "Donald Trump" "Barack Obama"]
- 58.
59. concept:(empresas) [Google Amazon Apple Microsoft Facebook Twitter Netflix Movistar Telefonica Endesa Iberdrola "Gas Natural" Indra Ferrovial Inditex]
- 60.
61. concept:(escritores) [Tolkien Asimov "Isaac Asimov" "Gabriel Garcia Marquez" "Vargas Llosa" "Pablo Neruda" "Hemingway" "Julio Cortazar" "Stephen King" "Umberto Eco" "Isabel Allende" "Alejandro Dumas" "Perez Reverte" Orwell "Julio Verne" "Eduardo Galeano"]
- 62.
63. concept:(cineastas) [Scorsese "Martin Scorsese" Tarantino Spielberg Kubrick Hitchcock "Tim Burton" "Woody Allen" "Ridley Scott" "Clint Eastwood" "James Cameron" "Orson Welles" "Akira Kurosawa" "Peter Jackson" "Pedro Almodovar" "Christopher Nolan" "Charles Chaplin" "George Lucas" "Oliver Stone"]
- 64.
65. concept:(chiste_pepper) ^rand["¿por que me caen bien los submarinistas? \pau=1000\ porque en el fondo son buena gente" "¿Soldado López! \pau=500\ ¿Sí, mi capitán! \pau=500\ No lo vi ayer en la prueba de camuflaje \pau=500\ ¿Gracias, mi capitán!" "dice el capitán del barco \pau=250\ ¿subid las velas! \pau=1000\ y los de abajo se quedaron a oscuras" "dice el capitán del barco \pau=220\ ¿abordar el barco! \pau=1000\ y el barco quedó precioso" "¿que le dice una piedra a otra? \pau=1000\ la vida es dura" "¿de que se quejan los astronautas? \pau=1000\ de falta de espacio" "¿que quiere ser la servilleta cuando sea mayor? \pau=1000\ quiere ser billete" "¿Por que se suicidó el libro de matemáticas? \pau=1000\ Porque tenía demasiados problemas" "¿que le dice un techo a otro? \pau=1000\ techo de menos" "¿para que va una caja al gimnasio? \pau=1000\ para hacerse caja fuerte" "¿que hace una abeja en un gimnasio? \pau=1000\ zumba" "¿que le dice un cable a otro? \pau=1000\ sígueme la corriente" "¿que le dice un árbol a otro? \pau=1000\ nos han dejado plantados"]
- 66.
67. concept:(curiosidades) ^rand["Ernes Vincent Wright escribió una novela llamada Gadsby que contiene más de 50000 palabras, y ninguna de ellas contienen la letra e" "Un hombre llamado Charles Osborne tuvo hipo durante 69 años" "Se estima que millones de árboles en el mundo son plantados accidentalmente por arduillas que entierran sus nueces y se olvidan donde las escondieron" "Las manzanas son más eficientes que la cafeína para mantener a la gente despierta en las mañanas" "Walt Disney tenía miedo a los ratones" "¿Sabías que compartes tu fecha de cumpleaños con al menos otros 9 millones de personas en el mundo?" "En el espacio los astronautas no pueden llorar porque a falta de atracción gravitatoria, las lágrimas no pueden fluir" "Se imprime más dinero del juego de Monopoly en un año que lo que se imprime de dinero real en todo el mundo" "En Bulgaria, los búlgaros mueven la cabeza de arriba a abajo para decir no, y de un lado a otro para decir sí" "Los perros y los gatos también son diestros o zurdos, como los humanos" "La jirafa es el único mamífero que no produce sonido alguno" "Todas las termitas del mundo juntas pesan 10 veces más que todos los humanos juntos" "Los burros matan más gente anualmente que los accidentes aéreos" "Si se mastica una goma de mascar mientras se parte cebolla puede prevenir que te irrite los ojos" "Los búhos son las únicas aves que pueden ver el color azul" "Los hombres son 6 veces más susceptibles de ser golpeados por un rayo que las mujeres" "Al nacer tenemos 300 huesos, pero de adulto solo tenemos 206" "Una persona tiene más de 1460 sueños al año" "La mayoría de las partículas de polvo en las casas están hechas de piel muerta" "Los delfines duermen con un ojo abierto" "Un bulto de oro puro del tamaño de una cajita de cerillas puede ser aplanado hasta hacer una hoja del tamaño de una cancha de tenis" "Thomas Alba Edison tenía miedo a la oscuridad" "Algunas especies de lombrices son capaces de comerse solas si no encuentran comida" "En el mundo hay más pollos que personas" "La posición de los ojos de un burro le permite verse las cuatro patas al mismo tiempo" "El primer año de un perro equivale a 21 años humanos, cada año canino posterior es de 4 años humanos" "Los molinos de viento siempre giran al contrario de las manecillas del reloj, excepto en Irlanda" "Cada 45 segundos se incendia una casa en Estados Unidos" "Una libélula vive aproximadamente 24 horas" "Una especie de rana venenosa tiene suficiente veneno como para matar alrededor de 2200 personas" "Antes del 1800, los zapatos para el pie izquierdo y derecho eran iguales" "Las hormigas se estiran cuando despiertan en la mañana" "La silla eléctrica fue inventada por un dentista" "Los camellos tienen tres párpados para protegerse de las tormentas de arena" "Una persona ríe aproximadamente 15 veces diarias" "En 1694 los jueces se vistieron de negro para llorar la muerte de la reina María segunda y han permanecido así desde entonces" "La pieza más antigua de goma de mascar tiene más de 9000 años" "Los dientes humanos son casi tan duros como piedras" "Una cucaracha puede vivir varias semanas sin cabeza" "La jirafa puede limpiarse los oídos con su lengua de 21 pulgadas de largo" "Los egipcios antiguos dormían en almohadas hechas de piedra" "Existen más flamencomos de plástico en los Estados Unidos que flamencos de verdad" "El vuelo más largo que ha hecho una gallina es de 13 segundos" "Una cuarta parte de los huesos del cuerpo humano se encuentran en los pies" "Más de 1000 aves mueren anualmente por estrellarse contra ventanas" "El elefante es el único mamífero que no puede saltar" "En el mundo existen más de 5000 lenguas que reflejan diferentes culturas" "El corazón del erizo late un promedio de 300 veces por minuto" "El pingüino es el único ave que puede nadar pero no puede volar" "Un topo puede cavar un túnel

l de 300 pies de largo en solo una noche" "La luz tarda 8 minutos y 17 segundos en viajar desde el Sol hasta la superficie terrestre" "Cuando el volcán de Krakatoa hizo erupción en 1883, la fuerza que desató fue tan colosal que pudo oírse en Australia, a más de 4800 km de distancia" "La ciruela Kakadu australiana contiene 100 veces más vitamina C que una naranja" "Una anguila eléctrica puede producir una descarga superior a los 600 voltios" "El insecto alado más pequeño del mundo, la avispa parasitaria de Tanzania, es más pequeño que el ojo de una mosca común" "El koala duerme 22 horas al día de promedio, dos horas más que el perezoso" "Incluso viajando a la velocidad de la luz tardaríamos 2 millones de años en llegar a la galaxia grande más cercana, Andrómeda" "Con más de 2000 kilómetros de longitud, el gran arrecife de coral es la estructura viviente más grande de la Tierra" "El mayor dinosaurio jamás descubierto era el Seismosaurus, que medía más de 30 metros de altura y pesaba más de 80 toneladas" "El porcentaje de mortalidad por la mordedura de la serpiente mamba negra es del 95 por ciento" "El sentido del olfato de un perro es mil veces más sensible que el de los humanos" "Los actuales cohetes tripulados tardarían 70000 años en llegar a las estrellas más cercanas" "La Estación Espacial Internacional pesa cerca de 500 toneladas y tiene las dimensiones de un campo de fútbol" "Cada vaca doméstica emite aproximadamente 150 kilogramos de metano al año" "Los camellos resisten 17 días sin beber en condiciones de calor extremo" "Las 10 montañas más altas del mundo se encuentran en el Himalaya"]

68.

69. pronunciation:(heavy) [jevi]

Conceptos Personalizados

```

1. class MyClass(GeneratedClass):
2.     def __init__(self):
3.         GeneratedClass.__init__(self)
4.
5.     def onLoad(self):
6.         #put initialization code here
7.
8.         self.tts=ALProxy("ALAnimatedSpeech")
9.         self.dialog=ALProxy("ALDialog")
10.
11.     pass
12.
13.     def onUnload(self):
14.         #put clean-up code here
15.     pass
16.
17.     def cargaConceptos(self):
18.
19.         import urllib2
20.         import json
21.         import re
22.
23.         try:
24.
25.             url="http://eii.uva.es/~eduzal/pepper/informacion_pepper.json"
26.
27.             response = urllib2.urlopen(url)
28.
29.             self.data = json.loads(response.read().decode('cp1252'))
30.
31.             self.logger.info("Conceptos cargados de la web")
32.
33.         pass
34.
35.         except RuntimeError:
36.
37.             self.logger.warning("Error al cargar los conceptos de la web")
38.         pass
39.
40.     pass
41.
42.     def onInput_decirInfoPersonalizada(self, concepto_elegido):
43.
44.         self.logger.info("Concepto personalizado elegido: "+concepto_elegido)
45.
46.         self.dialog._pauseEngine(True) # Pausa dialogo
47.
48.         for dato in self.data['informacion_personalizada']:
49.             concepto=dato['name'].encode('utf-8')

```

```

50.         if(concepto==concepto_elegido): # Dice el texto correspondiente al concepto elegido
51.             self.tts.say(dato['info'].encode('utf-8'))
52.
53.         self.dialog._pauseEngine(False) # Continúa dialogo
54.
55.         pass
56.
57.     def onInput_onStart(self):
58.
59.
60.         try:
61.
62.             self.cargaConceptos()
63.
64.             # Informacion personalizada
65.             conceptos=[]
66.             for dato in self.data['informacion_personalizada']:
67.                 concepto=dato['name'].encode('utf-8')
68.                 conceptos.append(concepto) # Construye la lista de conceptos
69.
70.             self.dialog.setConcept("concepto_dinamico", "spe", conceptos) # Carga conceptos
71.
72.             self.logger.info("Conceptos personalizados cargados al dialogo: "+str(self.dialog.getConcept("concepto_dinamico", "spe")))
73.
74.             # Nombres residencia
75.             nombres=[]
76.             for dato in self.data['nombres_residencia']:
77.                 nombre=dato.encode('utf-8')
78.                 nombres.append(nombre) # Construye la lista de nombres
79.
80.             self.dialog.setConcept("nombres_residencia", "spe", nombres) # Carga nombres
81.             self.logger.info(nombres)
82.             self.logger.info("Nombres residencia cargados al dialogo: "+str(self.dialog.getConcept("nombres_residencia", "spe")))
83.
84.
85.         pass
86.     except RuntimeError:
87.         self.logger.warning("Error al cargar conceptos al dialogo")
88.         pass
89.
90.     #self.onStopped() #activate the output of the box
91.     pass
92.
93.     def onInput_onStop(self):
94.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
95.         self.onStopped() #activate the output of the box

```

Carga Datos Usuario

```

1.     import json
2.
3.     class MyClass(GeneratedClass):
4.         def __init__(self):
5.             GeneratedClass.__init__(self)
6.
7.         def onLoad(self):
8.             #put initialization code here
9.             pass
10.
11.        def onUnload(self):
12.            #put clean-up code here
13.            pass
14.
15.        def getMonthName(self, month):
16.
17.            month=int(month)
18.

```

```

19.     if(month==1):
20.         mes="enero"
21.     elif(month==2):
22.         mes="febrero"
23.     elif(month==3):
24.         mes="marzo"
25.     elif(month==4):
26.         mes="abril"
27.     elif(month==5):
28.         mes="mayo"
29.     elif(month==6):
30.         mes="junio"
31.     elif(month==7):
32.         mes="julio"
33.     elif(month==8):
34.         mes="agosto"
35.     elif(month==9):
36.         mes="septiembre"
37.     elif(month==10):
38.         mes="octubre"
39.     elif(month==11):
40.         mes="noviembre"
41.     else:
42.         mes="diciembre"
43.
44.     return mes
45.
46. def checkBirthday(self, userID, day, month):
47.
48.     #Comprueba cumpleaños
49.     diaCumple="no"
50.
51.     if(len(day)>0 and len(month)>0): # Comprueba que se tiene el dia y mes
52.
53.         import datetime
54.
55.         day=int(day)
56.         month=int(month)
57.
58.         currentTime = datetime.datetime.now()
59.
60.         if(currentTime.month==month and day==currentTime.day): # Comprueba si coinciden dia y mes
61.             diaCumple="si"
62.             self.logger.info("Hoy es el cumpleaños")
63.
64.
65.         dialog=ALProxy("ALDialog")
66.         dialog.insertUserData("diaCumple", diaCumple, userID) # Inserta en dialogo
67.
68.         pass
69.
70. def onInput_onStart(self):
71.
72.     user_info_service = ALProxy("ALUserInfo")
73.     user_session_service = ALProxy("ALUserSession")
74.     dialog=ALProxy("ALDialog")
75.     memory = ALProxy("ALMemory")
76.
77.     # Si no hay ningun usuario al que se este mirando sale del bloque
78.     if(user_session_service.getFocusedUser() == -1):
79.         self.onStopped()
80.     return
81.
82.     # Obtiene la ID del usuario
83.     userID=user_session_service.getFocusedUser()
84.
85.     # Numero de repeticiones en ejercicios
86.     try:

```

```

87.     data = memory.getData("numeroRepeticiones")
88.     dialog.insertUserData("numeroRepeticiones", str(data), userID)
89.     pass
90.     except RuntimeError: # Si no esta el valor pone por defecto 5 repeticiones
91.         memory.insertData("numeroRepeticiones", 5)
92.         dialog.insertUserData("numeroRepeticiones", str(5), userID)
93.         pass
94.
95.     if(userID==0): #Si es un usuario anonimo
96.         self.logger.info("usuario anonimo, no se cargan datos")
97.         self.onStopped()
98.         return
99.
100.    # Obtiene el nombre del archivo
101.    myDomain = "com.aldebaran.myApp"
102.    dataID="userfile"
103.    if(user_info_service.has(myDomain, userID, dataID)):
104.        userfile = user_info_service.get(myDomain, userID, dataID)
105.    else: # Si no hay datos no carga
106.        self.logger.info("id usuario: "+ str(userID))
107.        self.logger.info("No hay datos guardados que cargar")
108.        self.onStopped()
109.        return
110.
111.
112.    # Abre y lee archivo de perfil
113.    filename="/home/nao/perfiles/"+userfile+".json"
114.    f=open(filename, 'r')
115.
116.    data_str=str(f.read())
117.
118.    data = json.loads(data_str) # Obtiene datos en forma de diccionario
119.
120.    f.close()
121.
122.    # Obtiene datos y los carga al dialogo
123.
124.    name=data["profile"]["name"].encode('utf-8') # hace encode por si el nombre tiene alguna tilde
125.    if(len(name)>0):
126.        dialog.insertUserData("name", name, userID)
127.
128.    age=str(data["permanentinfo"]["personal"]["age"])
129.    if(len(age)>0):
130.        dialog.insertUserData("age", age, userID)
131.
132.    birthday=str(data["permanentinfo"]["personal"]["birth"]["day"])
133.    if(len(birthday)>0):
134.        dialog.insertUserData("birthday", birthday, userID)
135.
136.    birthmonth=str(data["permanentinfo"]["personal"]["birth"]["month"])
137.    if(len(birthmonth)>0):
138.        birthmonthname=self.getMonthName(birthmonth)
139.        dialog.insertUserData("birthmonth", birthmonthname, userID)
140.
141.    birthyear=str(data["permanentinfo"]["personal"]["birth"]["year"])
142.    if(len(birthyear)>0):
143.        dialog.insertUserData("birthyear", birthyear, userID)
144.
145.    self.logger.info("Datos cargados")
146.
147.    # Muestra id y nombre
148.    self.logger.info("id usuario: "+ str(userID))
149.    if(len(name)>1):
150.        self.logger.info("nombre usuario: " + name)
151.
152.    # Comprueba si es un usuario temporal o permanente
153.    if(user_session_service.isUserPermanent(userID)):
154.        self.logger.info("Usuario permanente")

```

```

155.     else:
156.         self.logger.info("Usuario temporal")
157.
158.         # Comprueba cumpleaños
159.         self.checkBirthday(userID, birthday, birthmonth)
160.
161.         self.onStopped() #activate the output of the box
162.         pass
163.
164.     def onInput_onStop(self):
165.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
166.         self.onStopped() #activate the output of the box

```

Guarda Datos Usuario

```

1.     import json
2.
3.     class MyClass(GeneratedClass):
4.         def __init__(self):
5.             GeneratedClass.__init__(self)
6.
7.         def onLoad(self):
8.             #put initialization code here
9.             pass
10.
11.        def onUnload(self):
12.            #put clean-up code here
13.            pass
14.
15.        def getMonthNumber(self, month):
16.
17.            if(month=="enero"):
18.                mes=1
19.            elif(month=="febrero"):
20.                mes=2
21.            elif(month=="marzo"):
22.                mes=3
23.            elif(month=="abril"):
24.                mes=4
25.            elif(month=="mayo"):
26.                mes=5
27.            elif(month=="junio"):
28.                mes=6
29.            elif(month=="julio"):
30.                mes=7
31.            elif(month=="agosto"):
32.                mes=8
33.            elif(month=="septiembre"):
34.                mes=9
35.            elif(month=="octubre"):
36.                mes=10
37.            elif(month=="noviembre"):
38.                mes=11
39.            else:
40.                mes=12
41.
42.            return mes
43.
44.        def onInput_onStart(self):
45.
46.            # Obtiene los proxy de los servicios
47.            user_info_service = ALProxy("ALUserInfo")
48.            user_session_service = ALProxy("ALUserSession")
49.            dialog=ALProxy("ALDialog")
50.            tts = ALProxy("ALTextToSpeech")
51.            memory = ALProxy("ALMemory")
52.            faceDetection= ALProxy("ALFaceDetection")
53.
54.            # Si no hay ningun usuario conocido al que se este mirando sale del bloque

```

```

55.     if(user_session_service.getFocusedUser() <= 0): # No hay ningun usuario enfocado o bien es anonimo
56.         self.onStopped()
57.         return
58.
59.     # Obtiene la ID del usuario
60.     userID=user_session_service.getFocusedUser()
61.
62.     myDomain = "com.aldebaran.myApp"
63.
64.     # Comprueba si se conoce el nombre
65.     dataID="name"
66.     if(dialog.getUserData(dataID, userID)):
67.         value=dialog.getUserData(dataID, userID)
68.         user_info_service.set(myDomain, userID, dataID, value) # Guarda nombre en userInfo (para poder obtenerlo
facilmente desde la tablet)
69.     else:
70.         self.onStopped()
71.         return # Si no se conoce el nombre no se guardan datos
72.
73.     # Comprueba si el usuario es permanente
74.     if(not user_session_service.isUserPermanent(userID)):
75.         self.onStopped()
76.         return # Si no es permanente no se guardan datos
77.
78.
79.     # Comprueba si tiene fichero de perfil correspondiente
80.     dataID="userfile"
81.     if(user_info_service.has(myDomain, userID, dataID)):
82.         userfile = user_info_service.get(myDomain, userID, dataID)
83.     else: # Crea un nuevo fichero de perfil para el usuario
84.         userfile=str(userID) # Crea fichero userID.json
85.         # Carga el JSON vacio
86.         filename="/home/nao/perfiles/vacio.json"
87.         f=open(filename, 'r')
88.         data_str=str(f.read())
89.         data = json.loads(data_str)
90.         f.close()
91.         # Crea el perfil a partir del vacio
92.         filename="/home/nao/perfiles/"+userfile+".json"
93.         f2=open(filename, 'wb')
94.         json.dump(data,f2,indent=4)
95.         f2.close()
96.         user_info_service.set(myDomain, userID, dataID, userfile) # Asigna el fichero
97.
98.
99.     # Abre y lee archivo de perfil
100.    filename="/home/nao/perfiles/"+userfile+".json"
101.    f=open(filename, 'r')
102.
103.    data_str=str(f.read())
104.
105.    data = json.loads(data_str) # Obtiene datos en forma de diccionario
106.
107.    f.close()
108.
109.    # Obtiene los datos del dialogo para actualizar el perfil
110.
111.    dataID="name"
112.    if(dialog.getUserData(dataID, userID)):
113.        value=dialog.getUserData(dataID, userID)
114.        data["profile"]["name"]=value
115.        data["permanentinfo"]["personal"]["name"]=value
116.
117.    dataID="age"
118.    if(dialog.getUserData(dataID, userID)):
119.        value=dialog.getUserData(dataID, userID)
120.        data["permanentinfo"]["personal"]["age"]=value
121.    else: #intenta calcular la edad a partir del año de nacimiento

```

```

122.     if(dialog.getUserData("birthyear", userID)):
123.         birthYear=int(dialog.getUserData("birthyear", userID))
124.         currentTime = datetime.datetime.now()
125.         currentYear=currentTime.year
126.         edad=currentYear-birthYear
127.         #guarda la edad obtenida
128.         dialog.insertUserData(dataID, str(edad), userID)
129.         data["permanentinfo"]["personal"]["age"]=str(edad)
130.
131.
132.     dataID="birthday"
133.     if(dialog.getUserData(dataID, userID)):
134.         value=dialog.getUserData(dataID, userID)
135.         data["permanentinfo"]["personal"]["birth"]["day"]=value
136.
137.     dataID="birthmonth"
138.     if(dialog.getUserData(dataID, userID)):
139.         value=dialog.getUserData(dataID, userID)
140.         monthNumber=self.getMonthNumber(value) # Obtiene el numero de mes
141.         data["permanentinfo"]["personal"]["birth"]["month"]=str(monthNumber)
142.
143.     dataID="birthyear"
144.     if(dialog.getUserData(dataID, userID)):
145.         value=dialog.getUserData(dataID, userID)
146.         data["permanentinfo"]["personal"]["birth"]["year"]=value
147.
148.     dataID="numeroRepeticiones"
149.     if(dialog.getUserData(dataID, userID)):
150.         value=dialog.getUserData(dataID, userID)
151.         memory.insertData("numeroRepeticiones", int(value)) # Escribe numero de repeticiones en memoria
152.
153.
154.     # Abre el perfil para actualizarlo
155.     filename="/home/nao/perfiles/"+userfile+".json"
156.     self.logger.info(filename)
157.     f=open(filename, 'wb')
158.
159.     json.dump(data,f,indent=4) # Actualiza fichero con los nuevos datos de perfil
160.
161.     f.close()
162.
163.     self.logger.info("Datos guardados")
164.
165.     self.onStopped() #activate the output of the box
166.     pass
167.
168.     def onInput_onStop(self):
169.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
170.         self.onStopped() #activate the output of the box

```

Carga Fecha y Hora

```

1.     class MyClass(GeneratedClass):
2.         def __init__(self):
3.             GeneratedClass.__init__(self)
4.
5.         def onLoad(self):
6.             #put initialization code here
7.             pass
8.
9.         def onUnload(self):
10.            #put clean-up code here
11.            pass
12.
13.        def onInput_onStart(self):
14.

```

```

15. import datetime
16.
17. user_session_service = ALProxy("ALUserSession")
18. dialog=ALProxy("ALDialog")
19.
20. userID=user_session_service.getFocusedUser()
21.
22. currentTime = datetime.datetime.now()
23.
24. mes=""
25. if(currentTime.month==1):
26.     mes="enero"
27. if(currentTime.month==2):
28.     mes="febrero"
29. if(currentTime.month==3):
30.     mes="marzo"
31. if(currentTime.month==4):
32.     mes="abril"
33. if(currentTime.month==5):
34.     mes="mayo"
35. if(currentTime.month==6):
36.     mes="junio"
37. if(currentTime.month==7):
38.     mes="julio"
39. if(currentTime.month==8):
40.     mes="agosto"
41. if(currentTime.month==9):
42.     mes="septiembre"
43. if(currentTime.month==10):
44.     mes="octubre"
45. if(currentTime.month==11):
46.     mes="noviembre"
47. if(currentTime.month==12):
48.     mes="diciembre"
49.
50. dialog.insertUserData("year", str(currentTime.year), userID)
51. dialog.insertUserData("month", mes, userID)
52. dialog.insertUserData("day", str(currentTime.day), userID)
53. dialog.insertUserData("hour", str(currentTime.hour), userID)
54. dialog.insertUserData("minute", str(currentTime.minute), userID)
55.
56. self.logger.info("Fecha y hora actuales cargados: " + str(currentTime.day)+ " " + mes + " " + str(currentTime.y
ear)+ " " + str(currentTime.hour)+ ":" + str(currentTime.minute))
57.
58. self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
59. self.onStopped() #activate the output of the box
60. pass
61.
62. def onInput_onStop(self):
63.     self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
64.     self.onStopped() #activate the output of the box

```

Pantalla Inicial

```

1. class MyClass(GeneratedClass):
2.
3.     def __init__(self):
4.         GeneratedClass.__init__(self)
5.
6.     def onLoad(self):
7.         self.isRunning = False
8.
9.     def onUnload(self):
10.        self.isRunning = False
11.
12.    def _getTabletService(self):
13.        tabletService = None
14.    try:
15.        tabletService = self.session().service("ALTabletService")

```

```

16.     except Exception as e:
17.         self.logger.error(e)
18.     return tabletService
19.
20.     def onInput_onStart(self):
21.         if self.isRunning:
22.             return # already running, nothing to do
23.         self.isRunning = True
24.         # We create TabletService here in order to avoid
25.         # problems with connections and disconnections of the tablet during the life of the application
26.         tabletService = self._getTabletService()
27.         appName = "menutablet-853a26"
28.         url='http://' + self._getTabletService().robotIp() + '/apps/' + appName + '/seleccion_usuario.html'
29.         state = False
30.
31.         # Obtiene el modo de la memoria
32.         memory=ALProxy("ALMemory")
33.         try:
34.             data = memory.getData("modoPantallaInicial")
35.             modo=int(data)
36.             pass
37.         except RuntimeError: # Si no esta el valor pone por defecto el menu principal
38.             memory.insertData("modoPantallaInicial", 1)
39.             modo=1
40.             pass
41.
42.         if appName:
43.             if tabletService:
44.                 if modo==1: # Carga menu principal
45.                     if tabletService.loadApplication(appName):
46.                         self.logger.info("Successfully set application: %s" % appName)
47.                         tabletService.showWebview()
48.                         state = True
49.                     else:
50.                         self.logger.warning("Got tablet service, but failed to set application: %s" % appName)
51.                     else: # Carga pantalla de seleccion de usuario
52.                         if tabletService.showWebview(url):
53.                             self.logger.info("Successfully set url: %s" % url)
54.                             tabletService.showWebview()
55.                             state = True
56.                         else:
57.                             self.logger.warning("Got tablet service, but failed to set application: %s" % appName)
58.                     else:
59.                         self.logger.warning("Couldn't find tablet service, so can't set application: %s" % appName)
60.             if state:
61.                 self.onSuccess()
62.             else:
63.                 self.onFailure()
64.             self.isRunning = False

```

Seguimiento

```

1.     class MyClass(GeneratedClass):
2.         def __init__(self):
3.             GeneratedClass.__init__(self)
4.
5.         def onLoad(self):
6.             #put initialization code here
7.             pass
8.
9.         def onUnload(self):
10.            #put clean-up code here
11.            tracker=ALProxy("ALTracker")
12.            tracker.stopTracker()
13.            pass
14.

```

```

15. def onInput_onStart(self):
16.
17.     navigation_service = ALProxy("ALNavigation")
18.     tracker=ALProxy("ALTracker")
19.     motion = ALProxy("ALMotion")
20.     memProxy = ALProxy("ALMemory")
21.
22.     motion.wakeUp()
23.
24.     tracker.registerTarget("People",[])
25.
26.     tracker.setMode("Navigate")
27.
28.     tracker.setRelativePosition([-0.2, 0, 0, 0.1, 0.1, 0])
29.
30.     tracker.track("People") #Start tracker
31.
32.     # Stop localization
33.     navigation_service.stopLocalization()
34.
35.     self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
36.     self.onStopped() #activate the output of the box
37.
38.     pass
39.
40. def onInput_onStop(self):
41.     self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
42.     self.onStopped() #activate the output of the box

```

Tiempo Actual

```

1. class MyClass(GeneratedClass):
2.     def __init__(self):
3.         GeneratedClass.__init__(self)
4.
5.     def onLoad(self):
6.         self.isRunning = False
7.         #put initialization code here
8.         pass
9.
10.    def onUnload(self):
11.        self.isRunning = False
12.        #put clean-up code here
13.        pass
14.
15.    def time_converter(self, time):
16.        import datetime
17.        converted_time = datetime.datetime.fromtimestamp(
18.            int(time)
19.        ).strftime('%I:%M %p')
20.        return converted_time
21.
22.
23.    def url_builder(self, city_id): #By city code
24.        user_api = '69295e5e774ee459a672c03a87cda28d' # Obtain yours form: http://openweathermap.org/
25.        unit = 'metric' # For Fahrenheit use imperial, for Celsius use metric, and the default is Kelvin.
26.        api = 'http://api.openweathermap.org/data/2.5/weather?id=' # Search for your city ID here: http://bulk.openwe
eathermap.org/sample/city.list.json.gz
27.
28.        full_api_url = api + str(city_id) + '&mode=json&units=' + unit + '&APPID=' + user_api
29.        return full_api_url
30.
31.    def url_builder2(self, lat, lon): #By coordinates
32.        user_api = '69295e5e774ee459a672c03a87cda28d' # Obtain yours form: http://openweathermap.org/
33.        unit = 'metric' # For Fahrenheit use imperial, for Celsius use metric, and the default is Kelvin.
34.        api = 'http://api.openweathermap.org/data/2.5/weather?lat=' # Search for your city ID here: http://bulk.open
weathermap.org/sample/city.list.json.gz
35.

```

```

36.     full_api_url = api + str(lat) + "&lon=" + str(lon) + '&lang=es&mode=json&units=' + unit + '&APPID=' + user_
    api
37.     return full_api_url
38.
39.     def data_fetch(self, full_api_url):
40.         import urllib
41.         import json
42.         url = urllib.urlopen(full_api_url)
43.         output = url.read().decode('utf-8')
44.         raw_api_dict = json.loads(output)
45.         url.close()
46.         return raw_api_dict
47.
48.
49.     def data_organizer(self, raw_api_dict):
50.         data = dict(
51.             city=raw_api_dict.get('name'),
52.             country=raw_api_dict.get('country'),
53.             temp=raw_api_dict.get('main').get('temp'),
54.             temp_max=raw_api_dict.get('main').get('temp_max'),
55.             temp_min=raw_api_dict.get('main').get('temp_min'),
56.             humidity=raw_api_dict.get('main').get('humidity'),
57.             pressure=raw_api_dict.get('main').get('pressure'),
58.             sky=raw_api_dict['weather'][0]['main'],
59.             description=raw_api_dict['weather'][0]['description'],
60.             sunrise=self.time_converter(raw_api_dict.get('sys').get('sunrise')),
61.             sunset=self.time_converter(raw_api_dict.get('sys').get('sunset')),
62.             wind=raw_api_dict.get('wind').get('speed'),
63.             wind_deg=raw_api_dict.get('deg'),
64.             dt=self.time_converter(raw_api_dict.get('dt')),
65.             cloudiness=raw_api_dict.get('clouds').get('all')
66.         )
67.         return data
68.
69.
70.     def data_output(self, data):
71.         #m_symbol = '\xb0' + 'C'
72.         m_symbol = "°C"
73.         self.logger.info('-----')
74.         self.logger.info('Current weather in: {}, {}'.format(data['city'], data['country']))
75.         self.logger.info(str(data['temp'])+m_symbol+" "+str(data['sky']))
76.         self.logger.info('Description: {}'.format(data['description'].encode('utf-8')))
77.         #self.logger.info('Max: {}, Min: {}'.format(data['temp_max'], data['temp_min']))
78.         #self.logger.info("")
79.         #self.logger.info('Wind Speed: {}, Degree: {}'.format(data['wind'], data['wind_deg']))
80.         #self.logger.info('Humidity: {}'.format(data['humidity']))
81.         #self.logger.info('Cloud: {}'.format(data['cloudiness']))
82.         #self.logger.info('Pressure: {}'.format(data['pressure']))
83.         #self.logger.info('Sunrise at: {}'.format(data['sunrise']))
84.         #self.logger.info('Sunset at: {}'.format(data['sunset']))
85.         #self.logger.info("")
86.         self.logger.info('Last update from the server: {}'.format(data['dt']))
87.         self.logger.info('-----')
88.
89.
90.
91.
92.     def onInput_onStart(self):
93.         if self.isRunning:
94.             return # already running, nothing to do
95.         self.isRunning = True
96.
97.         tts=ALProxy("ALAnimatedSpeech")
98.         dialog=ALProxy("ALDialog")
99.         try:
100.             #self.data_output(self.data_organizer(self.data_fetch(self.url_builder(3106672))))
101.
102.             dialog._pauseEngine(True)

```

```

103.
104.     #Carga datos de coordenadas de la memoria
105.     memory=ALProxy("ALMemory")
106.     lat=memory.getData("coordinateLat")
107.     lon=memory.getData("coordinateLon")
108.     #Obtiene los datos del tiempo de las coordenadas
109.     self.logger.info(self.url_builder2(lat, lon))
110.     data=self.data_organizer(self.data_fetch(self.url_builder2(lat, lon)))
111.     #data=self.data_organizer(self.data_fetch(self.url_builder(3106672)))
112.     self.data_output(data)
113.
114.
115.     if(int(data['temp'])==1 or int(data['temp'])==-1):
116.         tts.say("^mode(contextual) Ahora mismo la temperatura es de "+str(data['temp'])+" grado y hay "+data['d
description'].encode('utf-8'))
117.     else:
118.         tts.say("^mode(contextual) Ahora mismo la temperatura es de "+str(data['temp'])+" grados y hay "+data['d
escription'].encode('utf-8'))
119.
120.         dialog._pauseEngine(False)
121.
122.         self.onStopped() #activate the output of the box
123.
124.     except IOError:
125.
126.         dialog._pauseEngine(False)
127.
128.         tts.say("^mode(contextual) En este momento no puedo saber el tiempo")
129.         self.logger.info('no internet')
130.         self.onStopped() #activate the output of the box
131.
132.
133.     pass
134.
135.     def onInput_onStop(self):
136.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
137.         self.onStopped() #activate the output of the box

```

Noticias

```

1.     class MyClass(GeneratedClass):
2.         def __init__(self):
3.             GeneratedClass.__init__(self)
4.
5.         def onLoad(self):
6.             self.isRunning = False
7.             self.finActividadLog=False
8.             #put initialization code here
9.             self.detenerPrograma=False
10.            self.memory = ALProxy("ALMemory")
11.
12.            pass
13.
14.        def onUnload(self):
15.            #put clean-up code here
16.            self.detenerPrograma=True
17.
18.            if(self.isRunning==True):
19.                self.isRunning = False
20.                tts=ALProxy("ALAnimatedSpeech")
21.                tts._stopAll(True) #Para de hablar
22.
23.            pass
24.
25.        def onInput_onStart(self, p):
26.            if self.isRunning:
27.                return # already running, nothing to do
28.            self.isRunning = True
29.

```

```

30.     self.finActividadLog=False
31.
32.     self.detenerPrograma=False
33.
34.     tts=ALProxy("ALAnimatedSpeech")
35.     dialog=ALProxy("ALDialog")
36.
37.     datos=["Inicio Actividad", "Noticias", str(p)]
38.     self.memory.raiseEvent("EventoLog",datos)
39.
40.     try:
41.         #self.data_output(self.data_organizer(self.data_fetch(self.url_builder(3106672))))
42.
43.         import urllib2
44.         import json
45.         import re
46.
47.         tipoNoticias=str(p)
48.         if(tipoNoticias=='generales'):
49.             url="https://api.rss2json.com/v1/api.json?rss_url=http://ep00.epimg.net/rss/elpais/portada.xml&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
50.             elif(tipoNoticias=='deportes'):
51.                 #url="https://api.rss2json.com/v1/api.json?rss_url=http://ep00.epimg.net/rss/deportes/portada.xml&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
52.                 url="https://api.rss2json.com/v1/api.json?rss_url=http://www.elnortedecastilla.es/rss/2.0/?section=deporte&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
53.             elif(tipoNoticias=='ciencia'):
54.                 url="https://api.rss2json.com/v1/api.json?rss_url=http://ep00.epimg.net/rss/elpais/ciencia.xml&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
55.             elif(tipoNoticias=='tecnologia'):
56.                 url="https://api.rss2json.com/v1/api.json?rss_url=http://ep00.epimg.net/rss/tecnologia/portada.xml&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
57.             elif(tipoNoticias=='cultura'):
58.                 url="https://api.rss2json.com/v1/api.json?rss_url=http://ep00.epimg.net/rss/cultura/portada.xml&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
59.             elif(tipoNoticias=='economia'):
60.                 url="https://api.rss2json.com/v1/api.json?rss_url=http://ep00.epimg.net/rss/economia/portada.xml&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
61.             elif(tipoNoticias=='locales'):
62.                 url="https://api.rss2json.com/v1/api.json?rss_url=http://www.elnortedecastilla.es/rss/2.0/?section=valladolid&api_key=g0xpuftd8dcr4xrahdqf9q4cvb6vucs4zcgrati"
63.
64.         response = urllib2.urlopen(url)
65.
66.         data = json.loads(response.read());
67.
68.
69.         self.logger.info( '=====' + data['status'].encode('utf-8') + '=====' )
70.
71.         self.memory.insertData("ActividadGeneralEnCurso", True)
72.         dialog._pauseEngine(True)
73.
74.         for item in data['items']:
75.             if(self.detenerPrograma==True):
76.                 self.memory.insertData("ActividadGeneralEnCurso", False)
77.                 return
78.             self.logger.info(item['title'].encode('utf-8'))
79.             titular=item['title'].encode('utf-8')
80.             descripcion=item['description'].encode('utf-8')
81.             tts.say("^mode(contextual) " + titular)
82.             if(self.detenerPrograma==True):
83.                 self.memory.insertData("ActividadGeneralEnCurso", False)
84.                 return
85.             tts.say("^mode(contextual) " + descripcion + " \pau=1000\ ")
86.             if(self.detenerPrograma==True):
87.                 self.memory.insertData("ActividadGeneralEnCurso", False)
88.                 return
89.

```

```

90.     tts.say("^mode(contextual) Esas han sido las noticias más importantes del día")
91.
92.     self.memory.insertData("ActividadGeneralEnCurso", False)
93.     dialog._pauseEngine(False)
94.
95.     if(self.finActividadLog==False):
96.         datos=["Fin Actividad", "Noticias", "-"]
97.         self.memory.raiseEvent("EventoLog",datos)
98.         self.finActividadLog=True # Actividad terminada en log
99.
100.    user_session_service = ALProxy("ALUserSession")
101.    if(user_session_service.getFocusedUser() == -1): # There is no focused user
102.        self.finActividad()
103.        return
104.
105.    self.onStopped() #activate the output of the box
106.
107.    except IOError:
108.        self.memory.insertData("ActividadGeneralEnCurso", False)
109.        dialog._pauseEngine(False)
110.
111.        if(self.finActividadLog==False):
112.            datos=["Fin Actividad", "Noticias", "Error"]
113.            self.memory.raiseEvent("EventoLog",datos)
114.            self.finActividadLog=True # Actividad terminada en log
115.
116.        tts.say("^mode(contextual) En este momento no puedo leer las noticias")
117.        self.logger.info('no internet')
118.        self.onStopped() #activate the output of the box
119.
120.    pass
121.
122.    def onInput_onStop(self):
123.
124.        if(self.isRunning==True):
125.            self.memory.insertData("ActividadGeneralEnCurso", False)
126.            dialog=ALProxy("ALDialog")
127.            dialog._pauseEngine(False)
128.
129.            if(self.finActividadLog==False):
130.                datos=["Fin Actividad", "Noticias", "Interrupcion"]
131.                self.memory.raiseEvent("EventoLog",datos)
132.                self.finActividadLog=True # Actividad terminada en log
133.
134.            self.detenerPrograma=True
135.            self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
136.            self.onStopped() #activate the output of the box
137.
138.        def onInput_cambioUsuario(self):
139.
140.            if(self.isRunning==True):
141.                if(self.finActividadLog==False):
142.                    datos=["Fin Actividad", "Noticias", "Cambio de Usuario"]
143.                    self.memory.raiseEvent("EventoLog",datos)
144.                    self.finActividadLog=True # Actividad terminada en log

```

Wikipedia

```

1.  class MyClass(GeneratedClass):
2.      def __init__(self):
3.          GeneratedClass.__init__(self)
4.
5.      def onLoad(self):
6.          self.isRunning = False
7.          self.finActividadLog=False
8.          #put initialization code here
9.          self.detenerPrograma=False
10.         self.memory = ALProxy("ALMemory")
11.

```

```

12.     pass
13.
14.     def onUnload(self):
15.         #put clean-up code here
16.         self.detenerPrograma=True
17.
18.         if(self.isRunning==True):
19.             self.isRunning = False
20.             tts=ALProxy("ALAnimatedSpeech")
21.             tts._stopAll(True) #Para de hablar
22.
23.     pass
24.
25.     def onInput_onStart(self, p):
26.         if self.isRunning:
27.             return # already running, nothing to do
28.         self.isRunning = True
29.
30.         self.finActividadLog=False
31.
32.         self.detenerPrograma=False
33.
34.         tts=ALProxy("ALAnimatedSpeech")
35.         dialog=ALProxy("ALDialog")
36.
37.         datos=["Inicio Actividad", "BusquedaWikipedia", "-"]
38.         self.memory.raiseEvent("EventoLog",datos)
39.
40.     try:
41.         #self.data_output(self.data_organizer(self.data_fetch(self.url_builder(3106672))))
42.
43.         import urllib2
44.         import json
45.         import re
46.
47.         busqueda=str(p)
48.         #busqueda="Valladolid"
49.
50.         url="https://es.wikipedia.org/w/api.php?format=json&action=query&prop=extracts&exintro=&redirects&in
dexpageids=&titles="+urllib2.quote(busqueda)
51.         self.logger.info(url)
52.
53.         response = urllib2.urlopen(url)
54.
55.         data = json.loads(response.read());
56.
57.         # Obtiene la Id de la pagina
58.         pageId=data['query']['pageids'][0].encode('utf-8')
59.         self.logger.info(pageId)
60.
61.         if(pageId=="-1"): # si no existe la pagina buscada
62.             self.logger.info("No hay información sobre eso")
63.             tts.say("^mode(contextual) No puedo encontrar información sobre eso")
64.             self.onStopped()
65.         return
66.
67.         # Obtiene el resumen
68.         resumen=data['query']['pages'][pageId]['extract'].encode('utf-8')
69.         resumen=re.sub(r"[\.\*?]", "", resumen) # elimina las anotaciones
70.         if(resumen.find('</dl>')>0): # elimina informacion de desambiguaciones
71.             resumen=resumen[resumen.find('</dl>'):]
72.             resumen=re.sub(r"<.*?>", "", resumen) # mantiene solo texto
73.             resumen=re.sub(r"(. *?)", "", resumen) # quita texto entre parentesis
74.             resumen=re.sub(r";.*?&", ";", resumen)
75.             resumen=re.sub(r"&", "", resumen)
76.             resumen=re.sub(r"\.", ". \pau=250\ ", resumen) #añade pausas en los puntos
77.             resumen=resumen.splitlines()[0] # mantiene solo el primer parrafo
78.

```

```

79.         self.logger.info(data['query']['pages'][pageId]['title'].encode('utf-8'))
80.         self.logger.info(resumen)
81.
82.         self.memory.insertData("ActividadGeneralEnCurso", True)
83.         dialog._pauseEngine(True)
84.
85.         #tts.say("^mode(contextual) \\rspd=95\\ " + resumen + " \\pau=750\\ ") #habla al 95% de velocidad
86.         tts.say("^mode(contextual) " + resumen + " \\pau=750\\ ") #habla
87.
88.         if(self.detenerPrograma==True): # Si se interrumpe
89.             return
90.
91.         if(self.isRunning==True): #termina de hablar sin interrupcion
92.
93.             self.memory.insertData("ActividadGeneralEnCurso", False)
94.             dialog._pauseEngine(False)
95.
96.             if(self.finActividadLog==False):
97.                 datos=["Fin Actividad", "BusquedaWikipedia", "-"]
98.                 self.memory.raiseEvent("EventoLog",datos)
99.                 self.finActividadLog=True # Actividad terminada en log
100.
101.             self.isRunning = False
102.
103.             user_session_service = ALProxy("ALUserSession")
104.             if(user_session_service.getFocusedUser() == -1): # Si no hay usuario sale del programa
105.                 self.finActividad()
106.                 return
107.
108.             self.onStopped() #activate the output of the box
109.
110.         except IOError:
111.             self.memory.insertData("ActividadGeneralEnCurso", False)
112.             dialog._pauseEngine(False)
113.
114.             if(self.finActividadLog==False):
115.                 datos=["Fin Actividad", "BusquedaWikipedia", "Error"]
116.                 self.memory.raiseEvent("EventoLog",datos)
117.                 self.finActividadLog=True # Actividad terminada en log
118.
119.             self.isRunning = False
120.
121.             tts.say("^mode(contextual) En este momento no puedo obtener la información")
122.             self.logger.info('no internet')
123.             self.onStopped() #activate the output of the box
124.
125.
126.         pass
127.
128.         def onInput_onStop(self):
129.
130.             if(self.isRunning==True):
131.                 self.memory.insertData("ActividadGeneralEnCurso", False)
132.                 dialog=ALProxy("ALDialog")
133.                 dialog._pauseEngine(False)
134.
135.                 if(self.finActividadLog==False):
136.                     datos=["Fin Actividad", "BusquedaWikipedia", "Interrupcion"]
137.                     self.memory.raiseEvent("EventoLog",datos)
138.                     self.finActividadLog=True # Actividad terminada en log
139.
140.                 user_session_service = ALProxy("ALUserSession")
141.                 if(user_session_service.getFocusedUser() == -1): # Si no hay usuario sale del programa
142.                     self.finActividad()
143.
144.                 self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
145.                 self.onStopped() #activate the output of the box
146.

```

```

147. def onInput_cambioUsuario(self):
148.
149.     if(self.isRunning==True):
150.         if(self.finActividadLog==False):
151.             datos=["Fin Actividad", "BusquedaWikipedia", "Cambio de Usuario"]
152.             self.memory.raiseEvent("EventoLog",datos)
153.             self.finActividadLog=True # Actividad terminada en log

```

Control música, juegos, mapas

```

1. import datetime
2.
3. class MyClass(GeneratedClass):
4.
5.     def __init__(self):
6.         GeneratedClass.__init__(self)
7.
8.     def onLoad(self):
9.         #put initialization code here
10.        self.isRunning = False
11.        self.finActividadLog=False
12.
13.        self.userID=0
14.
15.        self.user_session_service = ALProxy("ALUserSession")
16.        self.memory = ALProxy("ALMemory")
17.
18.        self.nombre=""
19.
20.        pass
21.
22.    def onUnload(self):
23.        #put clean-up code here
24.
25.        if(self.isRunning==True):
26.            if(self.finActividadLog==False):
27.                datos=["Fin Actividad", self.nombre, "-"]
28.                self.memory.raiseEvent("EventoLog",datos)
29.                self.finActividadLog=True # Actividad terminada en log
30.
31.            self.isRunning = False
32.
33.            pass
34.
35.    def onInput_onStart(self, data):
36.
37.        if(data=="index.html"): # Si vuelve al inicio
38.            self.memory.insertData("ActividadGeneralEnCurso", False)
39.
40.        if(self.isRunning): # si cambia de usuario o de menu
41.
42.            if(self.finActividadLog==False):
43.                datos=["Fin Actividad", self.nombre, "-"]
44.                self.memory.raiseEvent("EventoLog",datos)
45.                self.finActividadLog=True # Actividad terminada en log
46.
47.            self.isRunning = False
48.
49.            if(data!="juegoencuentrapareja-c80554" and data!="juegoexplotaglobos-
d56c11" and data!="adivanzas.html" and data!="musica_clasica.html" and data!="musica_pop.html" and data!="
musica_rock.html" and data!="musica_hard.html" and data!="maps.html"): # Solo con juegos, musica y maps
50.                return
51.
52.            self.isRunning = True
53.
54.            self.finActividadLog=False
55.
56.            self.userID=self.user_session_service.getFocusedUser()
57.

```

```

58.     self.nombre=data
59.
60.     datos=["Inicio Actividad", self.nombre, "- "]
61.     self.memory.raiseEvent("EventoLog",datos)
62.
63.     if(self.nombre=="musica_clasica.html" or self.nombre=="musica_pop.html" or self.nombre=="musica_rock.html" or self.nombre=="musica_hard.html"): # Solo con musica
64.         self.memory.insertData("ActividadGeneralEnCurso", True) # evita que se interrumpa la musica al perder al usuario
65.
66.     self.onStopped() #activate the output of the box
67.     pass
68.
69.     def onInput_onStop(self):
70.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
71.         self.onStopped() #activate the output of the box
72.
73.     def onInput_cambioUsuario(self):
74.
75.         if(self.isRunning==True):
76.             if(self.finActividadLog==False):
77.                 datos=["Fin Actividad", self.nombre, "Cambio de Usuario"]
78.                 self.memory.raiseEvent("EventoLog",datos)
79.                 self.finActividadLog=True # Actividad terminada en log

```

Decir Frase

```

1.     class MyClass(GeneratedClass):
2.         def __init__(self):
3.             GeneratedClass.__init__(self)
4.
5.         def onLoad(self):
6.             #put initialization code here
7.
8.             self.tts=ALProxy("ALTextToSpeech")
9.             self.atts=ALProxy("ALAnimatedSpeech")
10.            self.dialog=ALProxy("ALDialog")
11.
12.            pass
13.
14.        def onUnload(self):
15.            #put clean-up code here
16.            pass
17.
18.        def onInput_onStart(self, p):
19.
20.            frase=p[0]
21.            animacion=p[1]
22.
23.            self.dialog._pauseEngine(True) # Pausa dialogo
24.
25.            if(animacion=='true'):
26.                self.atts.say(frase) # Dice frase con animacion
27.            else:
28.                self.tts.say(frase) # Dice frase
29.
30.            self.dialog._pauseEngine(False) # Continua dialogo
31.
32.            #self.onStopped() #activate the output of the box
33.            pass
34.
35.        def onInput_onStop(self):
36.            self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
37.            self.onStopped() #activate the output of the box

```

Muestra menú

```
1. class MyClass(GeneratedClass):
2.
3.     def __init__(self):
4.         GeneratedClass.__init__(self)
5.
6.     def onLoad(self):
7.         self.isRunning = False
8.
9.     def onUnload(self):
10.        self.isRunning = False
11.
12.    def _getTabletService(self):
13.        tabletService = None
14.        try:
15.            tabletService = self.session().service("ALTabletService")
16.        except Exception as e:
17.            self.logger.error(e)
18.        return tabletService
19.
20.    def onInput_onStart(self):
21.        if self.isRunning:
22.            return # already running, nothing to do
23.        self.isRunning = True
24.        # We create TabletService here in order to avoid
25.        # problems with connections and disconnections of the tablet during the life of the application
26.        tabletService = self._getTabletService()
27.        appName = "menutablet-853a26"
28.        state = False
29.        if appName:
30.            if tabletService:
31.                if tabletService.loadApplication(appName):
32.                    self.logger.info("Successfully set application: %s" % appName)
33.                    tabletService.showWebview()
34.                    state = True
35.                else:
36.                    self.logger.warning("Got tablet service, but failed to set application: %s" % appName)
37.            else:
38.                self.logger.warning("Couldn't find tablet service, so can't set application: %s" % appName)
39.        if state:
40.            self.onSuccess()
41.        else:
42.            self.onFailure()
43.        self.isRunning = False
```

Muestra página

```
1. class MyClass(GeneratedClass):
2.
3.     def __init__(self):
4.         GeneratedClass.__init__(self)
5.
6.     def onLoad(self):
7.         self.isRunning = False
8.
9.     def onUnload(self):
10.        self.isRunning = False
11.
12.    def _getTabletService(self):
13.        tabletService = None
14.        try:
15.            tabletService = self.session().service("ALTabletService")
16.        except Exception as e:
17.            self.logger.error(e)
18.        return tabletService
19.
20.    def onInput_onStart(self, pagina):
21.        if self.isRunning:
```

```

22.     return # already running, nothing to do
23.     self.isRunning = True
24.     # We create TabletService here in order to avoid
25.     # problems with connections and disconnections of the tablet during the life of the application
26.     tabletService = self._getTableService()
27.     appName = "menutablet-853a26"
28.     url='http://' + self._getTableService().robotIp() + '/apps/' + appName + '/' + pagina
29.     state = False
30.     if appName:
31.         if tabletService:
32.             if tabletService.showWebview(url):
33.                 self.logger.info("Successfully set url: %s" % url)
34.                 tabletService.showWebview()
35.                 state = True
36.             else:
37.                 self.logger.warning("Got tablet service, but failed to set application: %s" % appName)
38.         else:
39.             self.logger.warning("Couldn't find tablet service, so can't set application: %s" % appName)
40.     if state:
41.         self.onSuccess()
42.     else:
43.         self.onFailure()
44.     self.isRunning = False

```

Muestra actividad

```

1.  class MyClass(GeneratedClass):
2.
3.     def __init__(self):
4.         GeneratedClass.__init__(self)
5.
6.     def onLoad(self):
7.         self.isRunning = False
8.
9.     def onUnload(self):
10.        self.isRunning = False
11.
12.    def _getTableService(self):
13.        tabletService = None
14.        try:
15.            tabletService = self.session().service("ALTabletService")
16.        except Exception as e:
17.            self.logger.error(e)
18.        return tabletService
19.
20.    def onInput_onStart(self):
21.        if self.isRunning:
22.            return # already running, nothing to do
23.        self.isRunning = True
24.        # We create TabletService here in order to avoid
25.        # problems with connections and disconnections of the tablet during the life of the application
26.        tabletService = self._getTableService()
27.        appName = "menutablet-853a26"
28.        url='http://' + self._getTableService().robotIp() + '/apps/' + appName + '/actividad.html'
29.        state = False
30.        if appName:
31.            if tabletService:
32.                if tabletService.showWebview(url):
33.                    self.logger.info("Successfully set url: %s" % url)
34.                    tabletService.showWebview()
35.                    state = True
36.                else:
37.                    self.logger.warning("Got tablet service, but failed to set application: %s" % appName)
38.            else:
39.                self.logger.warning("Couldn't find tablet service, so can't set application: %s" % appName)
40.        if state:
41.            self.onSuccess()
42.        else:
43.            self.onFailure()

```

```
44. self.isRunning = False
```

Muestra app

```
1. class MyClass(GeneratedClass):
2.
3.     def __init__(self):
4.         GeneratedClass.__init__(self)
5.
6.     def onLoad(self):
7.         self.isRunning = False
8.
9.     def onUnload(self):
10.        self.isRunning = False
11.
12.    def _getTabletService(self):
13.        tabletService = None
14.        try:
15.            tabletService = self.session().service("ALTabletService")
16.        except Exception as e:
17.            self.logger.error(e)
18.        return tabletService
19.
20.    def onInput_onStart(self, appId):
21.        if self.isRunning:
22.            return # already running, nothing to do
23.        self.isRunning = True
24.        # We create TabletService here in order to avoid
25.        # problems with connections and disconnections of the tablet during the life of the application
26.        tabletService = self._getTabletService()
27.        state = False
28.        if appId:
29.            if tabletService:
30.                if tabletService.loadApplication(appId):
31.                    self.logger.info("Successfully set application: %s" % appId)
32.                    tabletService.showWebview()
33.                    state = True
34.                else:
35.                    self.logger.warning("Got tablet service, but failed to set application: %s" % appId)
36.            else:
37.                self.logger.warning("Couldn't find tablet service, so can't set application: %s" % appId)
38.        if state:
39.            self.onSuccess()
40.        else:
41.            self.onFailure()
42.        self.isRunning = False
```

Crea usuario temporal

```
1. class MyClass(GeneratedClass):
2.     def __init__(self):
3.         GeneratedClass.__init__(self)
4.
5.     def onLoad(self):
6.         #put initialization code here
7.         pass
8.
9.     def onUnload(self):
10.        #put clean-up code here
11.        pass
12.
13.    def onInput_onStart(self):
14.
15.        user_session=ALProxy("ALUserSession")
16.
17.        idsUsuario=user_session._createUsers(1) # Crea Usuario
18.
19.        idUsuario=idsUsuario[0]
20.
```

```

21.     self.logger.info("Usuario creado: "+str(idUsuario))
22.
23.     user_session._openUserSessions([idUsuario]) # Abre sesion
24.
25.     usuarioAnterior=user_session.getFocusedUser() # Obtiene usuario enfocado
26.
27.     user_session._setFocusedUser(idUsuario) # Activa usuario
28.
29.     if(usuarioAnterior>0):
30.         user_session._closeUserSessions([usuarioAnterior]) # Cierra sesion usuario anterior
31.
32.     self.onStopped() #activate the output of the box
33.     pass
34.
35.     def onInput_onStop(self):
36.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
37.         self.onStopped() #activate the output of the box

```

Borrar usuario

```

1.     class MyClass(GeneratedClass):
2.         def __init__(self):
3.             GeneratedClass.__init__(self)
4.
5.         def onLoad(self):
6.             #put initialization code here
7.             pass
8.
9.         def onUnload(self):
10.            #put clean-up code here
11.            pass
12.
13.        def onInput_onStart(self, p):
14.
15.            self.userSession=ALProxy("ALUserSession")
16.            self.userInfo=ALProxy("ALUserInfo")
17.
18.            myDomain = "com.aldebaran.myApp"
19.
20.            userID=int(p)
21.
22.            # Borra archivo de perfil
23.            dataID="userfile"
24.            if(self.userInfo.has(myDomain, userID, dataID)):
25.                userfile = self.userInfo.get(myDomain, userID, dataID)
26.                filename="/home/nao/perfiles/"+userfile+".json"
27.                os.remove(filename)
28.
29.            # Borra datos de usuario
30.            self.userInfo.removeUser(myDomain, userID);
31.
32.            # Borra usuario
33.            self.userSession._forgetPermanentUser(userID);
34.            self.userSession._deleteUser(userID);
35.
36.            self.logger.info("Usuario "+str(userID)+" borrado")
37.
38.            self.onStopped() #activate the output of the box
39.            pass
40.
41.        def onInput_onStop(self):
42.            self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
43.            self.onStopped() #activate the output of the box

```

Guarda foto

```
1. class MyClass(GeneratedClass):
2.     def __init__(self):
3.         GeneratedClass.__init__(self)
4.
5.     def onLoad(self):
6.         #put initialization code here
7.
8.     pass
9.
10.    def onUnload(self):
11.        #put clean-up code here
12.    pass
13.
14.    def onInput_onStart(self):
15.
16.        import sys
17.        import numpy as np
18.        import cv2
19.        import base64
20.
21.        user_info_service = ALProxy("ALUserInfo")
22.        user_session_service = ALProxy("ALUserSession")
23.        self.dialog = ALProxy("ALDialog")
24.        self.tts = ALProxy("ALTextToSpeech")
25.        movimiento = ALProxy("ALBackgroundMovement")
26.        seguimiento = ALProxy("ALBasicAwareness")
27.
28.        videoDevice = ALProxy('ALVideoDevice')
29.
30.        # Si no hay ningun usuario sale del bloque
31.        if(user_session_service.getFocusedUser() <= 0): # No hay ningun usuario enfocado o bien es anonimo
32.            self.onStopped()
33.            return
34.
35.        self.dialog._pauseEngine(True) # Pausa dialogo
36.
37.        # Desactiva movimientos para hacer mejor la foto
38.        movimiento.setEnabled(False)
39.        seguimiento.setEnabled(False)
40.
41.        self.iniciaVideo() # Empieza a grabar
42.
43.        self.tts.say("Voy a hacerte una foto para acordarme de ti. Colócate dentro del recuadro rojo \\pau=1000\\ A la d
e tres te hago la foto. \\pau=1000\\ Uno \\pau=1000\\ dos \\pau=1000\\ y tres \\pau=200\\ ") # Dice frase
44.
45.        self.paraVideo() # Deja de grabar
46.
47.        # Suscribe a la camara superior
48.        captureDevice = videoDevice.subscribeCamera("fotoUsuario", 0, 2, 13, 10) # 640x480 RGB
49.
50.        # Crea la imagen
51.        width = 640
52.        height = 480
53.        image = np.zeros((height, width, 3), np.uint8)
54.
55.        # Obtiene la imagen de la camara
56.        result = videoDevice.getImageRemote(captureDevice)
57.
58.        if result == None:
59.            self.logger.info('cannot capture.')
60.            return
61.        elif result[6] == None:
62.            self.logger.info('no image data string.')
63.            return
64.
65.        # Pasa los valores de la camara a la imagen
66.        values = map(ord, list(result[6]))
```

```

67.     i = 0
68.     for y in range(0, height):
69.
70.         for x in range(0, width):
71.
72.             image.itemset((y, x, 0), values[i + 0])
73.             image.itemset((y, x, 1), values[i + 1])
74.             image.itemset((y, x, 2), values[i + 2])
75.             i += 3
76.
77.     # Comprueba si esta de pie o sentado
78.     memory=ALProxy("ALMemory")
79.     try:
80.         posicionUsuario = memory.getData("posicionUsuario")
81.         pass
82.     except RuntimeError: # Si no esta el valor pone por defecto de pie
83.         memory.insertData("posicionUsuario", 1)
84.         posicionUsuario=1
85.         pass
86.
87.     # Recorta Imagen (igual que el cuadro rojo que muestra la tablet)
88.     tam=300 #300x300
89.     inicio_x=150
90.
91.     if(posicionUsuario==2):
92.         inicio_y=180 # sentado
93.         self.logger.info("Foto sentado")
94.     else:
95.         inicio_y=70 # de pie
96.         self.logger.info("Foto de pie")
97.
98.     fin_x=inicio_x+tam
99.     fin_y=inicio_y+tam
100.    img_recortada = image[inicio_y:fin_y, inicio_x:fin_x] # Recorta
101.
102.
103.    # Convierte la imagen a base64
104.    retval, buffer = cv2.imencode('.jpg', img_recortada)
105.    img_base64 = base64.b64encode(buffer)
106.
107.    videoDevice.releaseImage(captureDevice) # Libera imagen
108.
109.    videoDevice.unsubscribe(captureDevice)
110.
111.    # Obtiene la ID del usuario
112.    userID=user_session_service.getFocusedUser()
113.
114.    myDomain = "com.aldebaran.myApp"
115.
116.
117.    # Establece el usuario como permanente
118.    if(not user_session_service.isUserPermanent(userID)):
119.        self.logger.info("Estableciendo usuario " +str(userID) + " como permanente")
120.        user_session_service._rememberUserPermanently(userID)
121.
122.
123.    # Comprueba si tiene fichero de perfil correspondiente
124.    dataID="userfile"
125.    if(user_info_service.has(myDomain, userID, dataID)):
126.        userfile = user_info_service.get(myDomain, userID, dataID)
127.    else: # Crea un nuevo fichero de perfil para el usuario
128.        userfile=str(userID) # Crea fichero userID.json
129.        # Carga el JSON vacio
130.        filename="/home/nao/perfiles/vacio.json"
131.        f=open(filename, 'r')
132.        data_str=str(f.read())
133.        data = json.loads(data_str)
134.        f.close()

```

```

135.     # Crea el perfil a partir del vacio
136.     filename="/home/nao/perfiles/"+userfile+".json"
137.     f2=open(filename, 'wb')
138.     json.dump(data,f2,indent=4)
139.     f2.close()
140.     user_info_service.set(myDomain, userID, dataID, userfile) # Asigna el fichero
141.
142.
143.     # Abre y lee archivo de perfil
144.     filename="/home/nao/perfiles/"+userfile+".json"
145.     f=open(filename, 'r')
146.
147.     data_str=str(f.read())
148.
149.     data = json.loads(data_str) # Obtiene datos en forma de diccionario
150.
151.     f.close()
152.
153.     # Guarda imagen
154.     data['profile']['picture']=img_base64
155.     user_info_service.set(myDomain, userID, "picture", img_base64) # Guarda tambien en userInfo
156.
157.     memory.raiseEvent("CamaraVideo", img_base64) # Envia imagen como evento para mostrar en tablet
158.
159.     # Abre el perfil para actualizarlo
160.     filename="/home/nao/perfiles/"+userfile+".json"
161.     self.logger.info(filename)
162.     f=open(filename, 'wb')
163.
164.     json.dump(data,f,indent=4) # Actualiza fichero con los nuevos datos de perfil
165.
166.     f.close()
167.
168.     # Vuelve a activar los movimientos
169.     movimiento.setEnabled(True)
170.     seguimiento.setEnabled(True)
171.
172.     self.dialog._pauseEngine(False) # Continua dialogo
173.
174.     self.logger.info("Imagen guardada")
175.
176.     self.dialog.gotoTag('fotoTerminada', 'dialogoGeneral')
177.
178.     self.onStopped() #activate the output of the box
179.     pass
180.
181.     def onInput_onStop(self):
182.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
183.         self.onStopped() #activate the output of the box

```

Captura vídeo

```

1.     class MyClass(GeneratedClass):
2.         def __init__(self):
3.             GeneratedClass.__init__(self)
4.
5.         def onLoad(self):
6.             #put initialization code here
7.
8.             self.capturaCamara=False
9.
10.            pass
11.
12.        def onUnload(self):
13.            #put clean-up code here
14.
15.            self.capturaCamara=False
16.
17.            pass

```

```

18.
19. def onInput_onStart(self):
20.
21.     import sys
22.     import numpy as np
23.     import cv2
24.     import base64
25.     import Image
26.
27.     memory = ALProxy("ALMemory")
28.     dialog = ALProxy("ALDialog")
29.
30.     videoDevice = ALProxy('ALVideoDevice')
31.
32.     # Suscribe a la camara superior
33.     captureDevice = videoDevice.subscribeCamera("videoUsuario", 0, 1, 13, 30) # 320x240 RGB
34.
35.     # Comprueba si esta de pie o sentado
36.     memory=ALProxy("ALMemory")
37.     try:
38.         posicionUsuario = memory.getData("posicionUsuario")
39.         pass
40.     except RuntimeError: # Si no esta el valor pone por defecto de pie
41.         memory.insertData("posicionUsuario", 1)
42.         posicionUsuario=1
43.         pass
44.
45.     # Crea la imagen
46.     width = 320
47.     height = 240
48.
49.     self.capturaCamara=True
50.
51.     while(self.capturaCamara==True):
52.
53.         image = np.zeros((height, width, 3), np.uint8)
54.
55.         # Obtiene la imagen de la camara
56.         result = videoDevice.getImageRemote(captureDevice)
57.
58.         if result == None:
59.             self.logger.info('cannot capture.')
60.             return
61.         elif result[6] == None:
62.             self.logger.info('no image data string.')
63.             return
64.
65.         # Pasa los valores de la camara a la imagen
66.         values = map(ord, list(result[6]))
67.         i = 0
68.         for y in range(0, height):
69.
70.             for x in range(0, width):
71.
72.                 image.itemset((y, x, 0), values[i + 0])
73.                 image.itemset((y, x, 1), values[i + 1])
74.                 image.itemset((y, x, 2), values[i + 2])
75.                 i += 3
76.
77.         # Dibuja Recuadro Rojo
78.         if(posicionUsuario==2): # Sentado
79.             cv2.rectangle(image,(75,90),(75+150,90+150),(0,0,255),3)
80.         else: # De pie
81.             cv2.rectangle(image,(75,35),(75+150,35+150),(0,0,255),3)
82.
83.
84.         # Convierte la imagen a base64
85.         retval, buffer = cv2.imencode('.jpg', image)

```

```

86.         img_base64 = base64.b64encode(buffer)
87.
88.         memory.raiseEvent("CamaraVideo", img_base64) # Envia imagen como evento
89.
90.         videoDevice.releaseImage(captureDevice) # Libera imagen
91.
92.         videoDevice.unsubscribe(captureDevice)
93.
94.         self.logger.info("Captura de video terminada")
95.
96.         self.onStopped() #activate the output of the box
97.         pass
98.
99.     def onInput_onStop(self):
100.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
101.         self.onStopped() #activate the output of the box

```

Termina actividad

```

1.  import json
2.
3.  class MyClass(GeneratedClass):
4.      def __init__(self):
5.          GeneratedClass.__init__(self)
6.
7.      def onLoad(self):
8.          #put initialization code here
9.          pass
10.
11.     def activaEstimulos(self):
12.
13.         basicAwareness = ALProxy("ALBasicAwareness")
14.
15.         basicAwareness.setEngagementMode("FullyEngaged") # Mantiene a la persona fijada
16.         basicAwareness.setTrackingMode("BodyRotation") # Gira el cuerpo y la cabeza
17.
18.         # Activa deteccion de estímulos
19.         basicAwareness.setStimulusDetectionEnabled("Sound", True) # Detecta sonidos
20.         basicAwareness.setStimulusDetectionEnabled("Movement", True) # Detecta movimiento
21.         basicAwareness.setStimulusDetectionEnabled("People", True) # Detecta personas
22.         basicAwareness.setStimulusDetectionEnabled("Touch", True) # Detecta táctil
23.
24.         pass
25.
26.     def guardaPreferencias(self):
27.
28.         memory = ALProxy("ALMemory")
29.
30.         # Abre y lee archivo de preferencias
31.         filename="/home/nao/preferencias.json"
32.         f=open(filename, 'r')
33.
34.         data_str=str(f.read())
35.
36.         data = json.loads(data_str) # Obtiene datos en forma de diccionario
37.
38.         f.close()
39.
40.         # Obtiene los datos de la memoria para actualizar las preferencias
41.
42.         # Numero de repeticiones en ejercicios
43.         try:
44.             numeroRepeticiones = memory.getData("numeroRepeticiones")
45.             data["numeroRepeticiones"]=int(numeroRepeticiones)
46.             pass
47.         except RuntimeError:
48.             self.logger.warning("numeroRepeticiones no encontrado en ALMemory")
49.             pass
50.

```

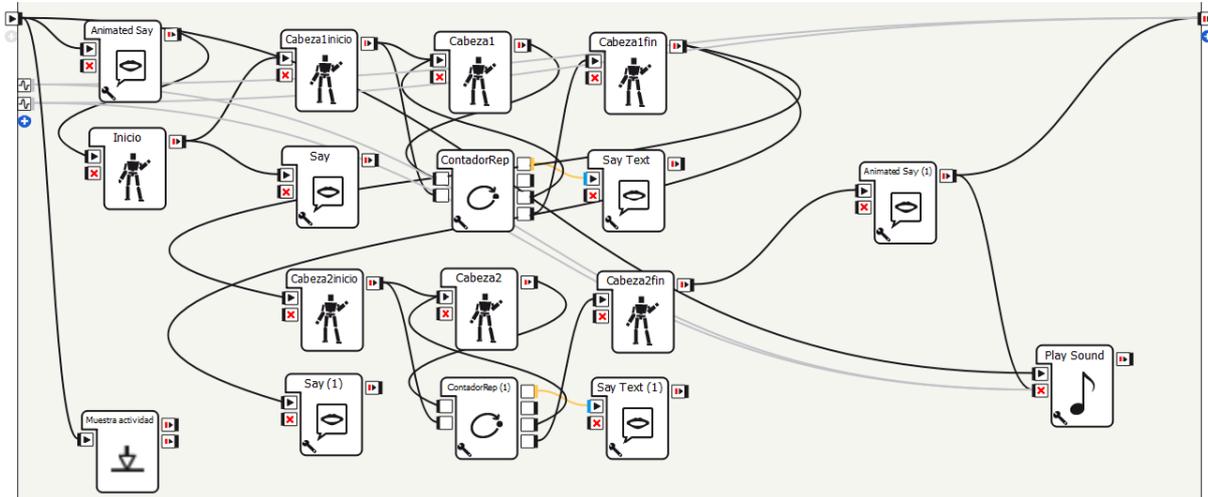
```

51.     # Modo de pantalla inicial
52.     try:
53.         modoPantallaInicial = memory.getData("modoPantallaInicial")
54.         data["modoPantallaInicial"]=int(modoPantallaInicial)
55.         pass
56.     except RuntimeError:
57.         self.logger.warning("modoPantallaInicial no encontrado en ALMemory")
58.         pass
59.
60.
61.     # Abre archivo de preferencias para actualizarlo
62.     filename="/home/nao/preferencias.json"
63.     f=open(filename, 'wb')
64.
65.     json.dump(data,f,indent=4) # Actualiza fichero con los nuevos datos de perfil
66.
67.     f.close()
68.
69.     self.logger.info("Archivo de preferencias actualizado")
70.
71.     pass
72.
73.     def onUnload(self):
74.         #put clean-up code here
75.
76.         self.activaEstimulos() # Reactiva estímulos
77.
78.         self.guardaPreferencias() # Guarda preferencias
79.
80.         pass
81.
82.     def _getTabletService(self):
83.         tabletService = None
84.         try:
85.             tabletService = self.session().service("ALTabletService")
86.         except Exception as e:
87.             self.logger.error(e)
88.         return tabletService
89.
90.     def onInput_onStart(self):
91.
92.         memory = ALProxy("ALMemory")
93.         estado = memory.getData("ActividadGeneralEnCurso")
94.
95.         if(estado==False):
96.
97.             self.activaEstimulos() # Reactiva estímulos
98.
99.             self.guardaPreferencias() # Guarda preferencias
100.
101.             self.onStopped() #activate the output of the box
102.
103.             tabletService = self._getTabletService()
104.             tabletService.hideWebview()
105.         pass
106.
107.     def onInput_onStop(self):
108.         self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
109.         self.onStopped() #activate the output of the box

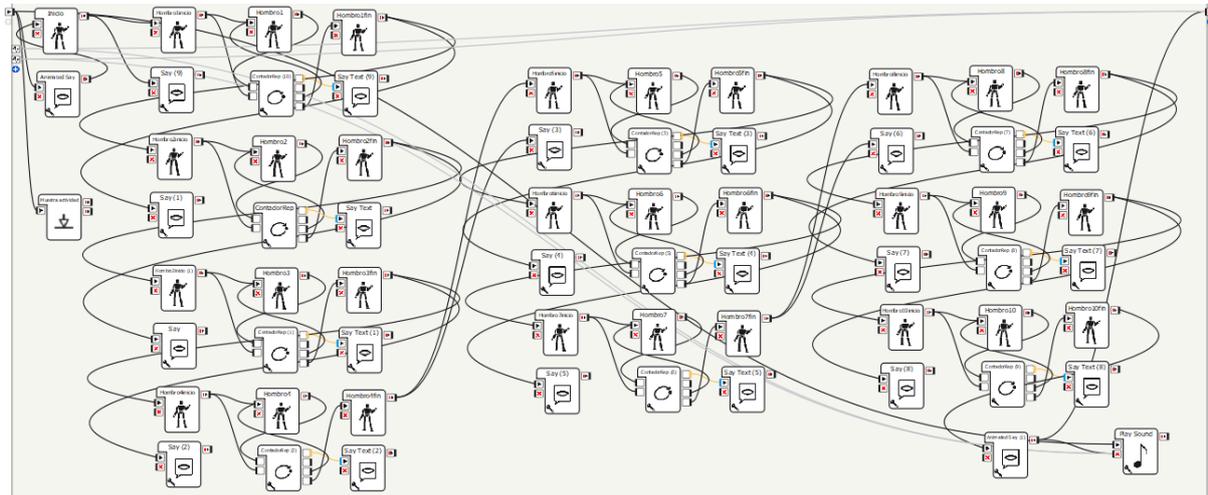
```

PROGRAMA EJERCICIOS

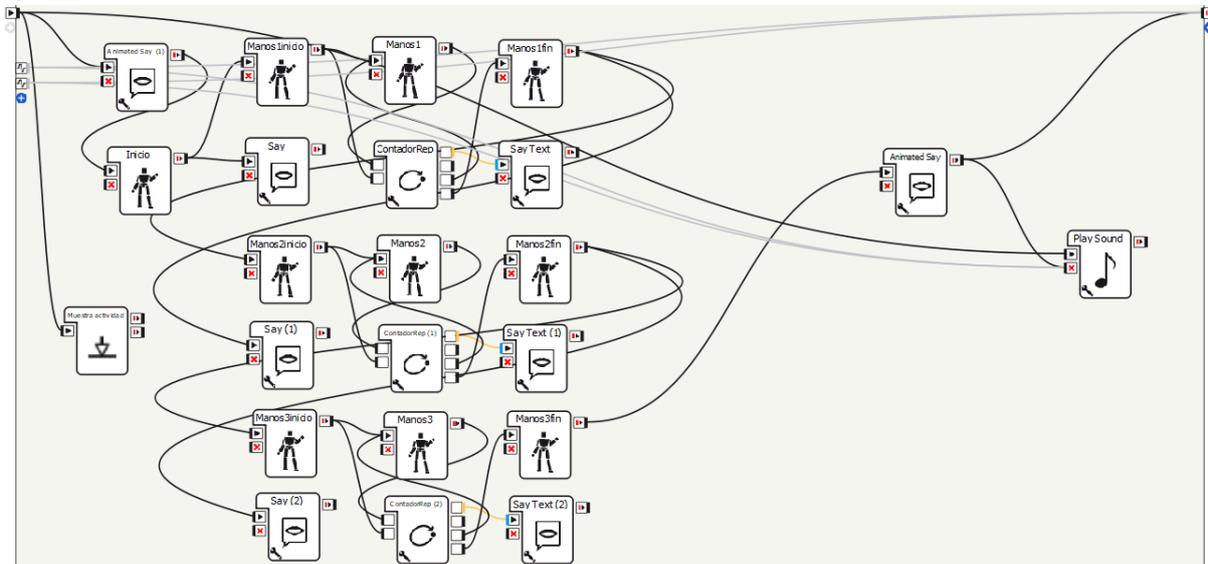
Ejercicios Cabeza



Ejercicios Hombros



Ejercicios Manos



Contador Repeticiones

```
1. class MyClass(GeneratedClass):
2.     def __init__(self):
3.         GeneratedClass.__init__(self, False)
4.
5.     def onLoad(self):
6.         self.memory = ALProxy("ALMemory")
7.         self.initializeParameters()
8.         if( self.memory.getData("numeroRepeticiones") - self.getParameter("Initial value") >= 0 ):
9.             self.stepSign = +1
10.        else:
11.            self.stepSign = -1
12.
13.    def onUnload(self):
14.        #~ puts code for box cleanup here
15.        pass
16.
17.    def onInput_onNext(self):
18.        bParamChanged = ( self.nLast != self.memory.getData("numeroRepeticiones") or self.nFirst != self.getParameter("Initial value") )
19.        bEnd = ( self.stepSign * self.nCounter >= self.stepSign * self.nLast )
20.        if( bEnd ):
21.            self.onEnd()
22.            self.currentValue( self.nLast )
23.            if( bEnd or bParamChanged ):
24.                self.onInput_onInit()
25.            if( not bEnd or bParamChanged ):
26.                currentCounter = self.nCounter
27.                self.nCounter = self.nCounter + self.stepSign * self.getParameter("Step value")
28.                self.currentValue( currentCounter )
29.                self.onNextIteration()
30.
31.    def initializeParameters(self):
32.        self.nFirst = self.getParameter("Initial value")
33.        self.nCounter = self.nFirst
34.        try:
35.            self.memory.insertData("numeroRepeticiones", int(self.memory.getData("numeroRepeticiones")))
36.            self.nLast = self.memory.getData("numeroRepeticiones")
37.        except RuntimeError: #si no esta el valor pone por defecto 5 repeticiones
38.            self.memory.insertData("numeroRepeticiones", 5)
39.            self.nLast = self.memory.getData("numeroRepeticiones")
40.
41.    def onInput_onInit(self):
42.        self.initializeParameters()
43.        self.onReinitialized()
```

MENÚ TABLET

Menú principal

```
1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10.    body {background-color: LightGray;}
11.    h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12.    p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14.    .button {
15.        padding: 15px 25px;
16.        font-size:5vw;
17.        width:47vw;
```

```

18.     height:27vh;
19.     text-align: center;
20.     cursor: pointer;
21.     outline: none;
22.     color: #fff;
23.     background-color: DodgerBlue;
24.     border: none;
25.     border-radius: 15px;
26.     box-shadow: 0 9px #999;
27.     margin-bottom: 3vh;
28.     margin-left: 0.5vw;
29.     margin-right: 0.5vw;
30.     float:left;
31. }
32.
33. .button:hover {background-color: DodgerBlue}
34.
35. .button:active {
36.     background-color: DodgerBlue;
37.     box-shadow: 0 5px #666;
38.     transform: translateY(4px);
39. }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MENU PRINCIPAL</h1>
44.
45. <button class="button" onclick="pepperDialogGotoTag('preguntaInformacion', 'dialogoGeneral')>INFORMACION</button>
46. <button class="button" onclick="pepperDialogGotoTag('preguntaJuego', 'dialogoGeneral')>JUEGOS</button>
47. <button class="button" onclick="pepperDialogGotoTag('preguntaEjercicio', 'dialogoGeneral')>EJERCICIOS</button>
48. <button class="button" onclick="pepperDialogGotoTag('preguntaMusica', 'dialogoGeneral')>MUSICA</button>
49.
50. <button class="button" onclick="pepperDialogGotoTag('mostrarAjustes', 'dialogoGeneral')" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
51. <div class="fa fa-cogs" style="font-size:5vw;color:white;" aria-hidden="true"></div>
52. </button>
53.
54. <button class="button" onclick="pepperDialogGotoTag('hacerValoracion', 'dialogoGeneral')" style="height:8vw;width:9vw;float:right;">
55. <div class="fa fa-commenting-o" style="font-size:5vw;color:white;" aria-hidden="true"></div>
56. </button>
57.
58. <button class="button" onclick="pepperDialogGotoTag('seleccionaUsuario', 'dialogoGeneral')" style="height:8vw;width:30vw;float:left;">
59. <div class="fa fa-user-circle" style="font-size:3.5vw;color:white;" aria-hidden="true" id="user_name"> Anonimo</div>
60. </button>
61.
62. <script src="/libs/qimessaging/2/qimessaging.js"></script>
63. <script src="/libs/qi/2/qi.js"></script>
64. <script type="text/javascript" src="js/scriptPepper.js"></script>
65.
66. <script>
67.
68. function setName(idUsuario) // Pone el nombre del usuario
69. {
70.     var myDomain = "com.aldebaran.myApp";
71.     session.service('ALUserInfo').then(function (userInfo) {
72.         userInfo.get(myDomain, parseInt(idUsuario), "name").then(function (userName) {
73.
74.             var elementoNombre=document.getElementById("user_name");
75.
76.             elementoNombre.innerHTML=" "+userName;
77.
78.             console.log('success');

```

```

79.         }, function (error) {
80.             console.log(error);
81.         });
82.     }, function (error) {
83.         console.log(error);
84.     });
85. }
86.
87. function estableceUsuario() // Obtiene el usuario para poner el nombre
88. {
89.     session.service('ALUserSession').then(function (UserSession) {
90.         UserSession.getFocusedUser().then(function (userID) {
91.
92.             if(parseInt(userID)>0) //Si hay usuario activado
93.                 setName(userID);
94.
95.             console.log('success');
96.         }, function (error) {
97.             console.log(error);
98.         });
99.     }, function (error) {
100.        console.log(error);
101.    });
102. }
103.
104. setInterval(estableceUsuario, 500); // Actualiza el nombre de usuario cada 500 ms
105.
106. </script>
107.
108. </body>
109. </html>

```

Script Pepper

```

1.  try {
2.
3.    QiSession( function (s) {
4.
5.      console.log('connected!');
6.
7.      session = s;
8.
9.    });
10.
11. } catch (err) {
12.
13.   console.log("Error when initializing QiSession: " + err.message);
14.
15.   console.log("Make sure you load this page from the robots server.")
16.
17. }
18.
19. function pepperSendLog(evento, info, tag) {
20.   session.service('ALMemory').then(function (memory) {
21.     var datos=[evento, info, tag];
22.     memory.raiseEvent("EventoLog",datos);
23.   }, function (error) {
24.     console.log(error);
25.   })
26. }
27.
28. function pepperPauseDialog(value) {
29.   session.service('ALDialog').then(function (dialog) {
30.     dialog._pauseEngine(value);
31.   }, function (error) {
32.     console.log(error);
33.   })
34. }
35.

```

```

36. function pepperAnimatedSay(text) {
37.   session.service('ALMemory').then(function (memory) {
38.     var datos=[String(text), "true"];
39.     memory.raiseEvent("EventoFrase",datos);
40.   }, function (error) {
41.     console.log(error);
42.   })
43. }
44.
45. function pepperAnimatedSayInicio(text) {
46.
47.   try {
48.
49.     QiSession( function (s) {
50.
51.       console.log('connected!');
52.
53.       var localSession = s;
54.
55.       localSession.service('ALMemory').then(function (memory) {
56.         var datos=[String(text), "true"];
57.         memory.raiseEvent("EventoFrase",datos);
58.       }, function (error) {
59.         console.log(error);
60.       })
61.
62.     });
63.
64.   } catch (err) {
65.
66.     console.log("Error when initializing QiSession: " + err.message);
67.
68.     console.log("Make sure you load this page from the robots server.")
69.
70.   }
71. }
72.
73. function pepperSay(text) {
74.   session.service('ALMemory').then(function (memory) {
75.     var datos=[String(text), "false"];
76.     memory.raiseEvent("EventoFrase",datos);
77.   }, function (error) {
78.     console.log(error);
79.   })
80. }
81.
82. function pepperSayInicio(text) {
83.
84.   try {
85.
86.     QiSession( function (s) {
87.
88.       console.log('connected!');
89.
90.       var localSession = s;
91.
92.       localSession.service('ALMemory').then(function (memory) {
93.         var datos=[String(text), "false"];
94.         memory.raiseEvent("EventoFrase",datos);
95.       }, function (error) {
96.         console.log(error);
97.       })
98.
99.     });
100.
101.   } catch (err) {
102.
103.     console.log("Error when initializing QiSession: " + err.message);

```

```

104.
105.     console.log("Make sure you load this page from the robots server.")
106.
107.     }
108. }
109.
110. function pepperSwitchFocus(behaviourId) {
111.     session.service('ALAutonomousLife').then(function (al) {
112.         al.switchFocus(behaviourId);
113.         pepperSendLog("Boton pulsado", "Iniciar Actividad "+String(behaviourId), "-");
114.     }, function (error) {
115.         console.log(error);
116.     })
117. }
118.
119. function pepperInsertData(Id, data) {
120.     session.service('ALMemory').then(function (memory) {
121.         memory.insertData(Id, data);
122.         pepperSendLog("Boton pulsado", "Guardar dato "+String(Id), String(data));
123.     }, function (error) {
124.         console.log(error);
125.     })
126. }
127.
128. function pepperInsertIntData(Id, data) {
129.     session.service('ALMemory').then(function (memory) {
130.         if(Number.isInteger(data)) {
131.             memory.insertData(Id, data);
132.         }else{
133.             memory.insertData(Id, parseInt(data));
134.         }
135.         pepperSendLog("Boton pulsado", "Guardar dato "+String(Id), String(data));
136.     }, function (error) {
137.         console.log(error);
138.     })
139. }
140.
141. function pepperDialogGotoTag(tagName, topicName) {
142.     session.service('ALAnimatedSpeech').then(function (dialog) {
143.         dialog._stopAll(true); //para la frase actual
144.     }, function (error) {
145.         console.log(error);
146.     })
147.
148.     session.service('ALDialog').then(function (dialog) {
149.         dialog.gotoTag(tagName, topicName);
150.         pepperSendLog("Boton pulsado", "Ir a "+tagName, tagName);
151.     }, function (error) {
152.         console.log(error);
153.     })
154. }
155.
156. function pepperDialogLaunchTabletApp(appName) {
157.     session.service('ALTabletService').then(function (tabletService) {
158.         tabletService.loadApplication(appName);
159.         pepperSendLog("Boton pulsado", "Iniciar app "+appName, appName);
160.     }, function (error) {
161.         console.log(error);
162.     })
163. }

```

Actividad en Curso

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4.     <title>Menu Tablet</title>
5.     <meta charset="UTF-8">
6. </head>

```



```

44. <button class="button" onclick="pepperDialogGotoTag('manejarUsuarios', 'dialogoGeneral')>USUARIOS</button>
45. <button class="button" onclick="pepperDialogGotoTag('ajustesTablet', 'dialogoGeneral')>AJUSTES TABLET</button>
46.
47. <button class="button" style="background-color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverInicio', 'dialogoGeneral')>VOLVER</button>
48.
49. <script src="/libs/qimessaging/2/qimessaging.js"></script>
50. <script src="/libs/qi/2/qi.js"></script>
51. <script type="text/javascript" src="js/scriptPepper.js"></script>
52.
53. </body>
54. </html>

```

Ajustes usuarios

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10. body {background-color: LightGray;}
11. h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12. p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14. .button {
15. padding: 15px 25px;
16. font-size:5vw;
17. width:47vw;
18. height:27vh;
19. text-align: center;
20. cursor: pointer;
21. outline: none;
22. color: #fff;
23. background-color: DodgerBlue;
24. border: none;
25. border-radius: 15px;
26. box-shadow: 0 9px #999;
27. margin-bottom: 3vh;
28. margin-left: 0.5vw;
29. margin-right: 0.5vw;
30. float:left;
31. }
32.
33. .button:hover {background-color: DodgerBlue}
34.
35. .button:active {
36. background-color: DodgerBlue;
37. box-shadow: 0 5px #666;
38. transform: translateY(4px);
39. }
40.
41. table {
42. border-collapse: collapse;
43. border-spacing: 0;
44. width: 100%;
45. border: 3px solid #ddd;
46. font-size:2vw;
47. }
48.
49. th, td {
50. text-align: left;
51. padding: 16px;

```

```

52. }
53.
54. tr:nth-child(even) {
55.     background-color: #f2f2f2
56. }
57.
58. /* The Modal (background) */
59. .modal {
60.     display: none; /* Hidden by default */
61.     position: fixed; /* Stay in place */
62.     z-index: 1; /* Sit on top */
63.     padding-top: 100px; /* Location of the box */
64.     left: 0;
65.     top: 0;
66.     width: 100%; /* Full width */
67.     height: 100%; /* Full height */
68.     overflow: auto; /* Enable scroll if needed */
69.     background-color: rgb(0,0,0); /* Fallback color */
70.     background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
71. }
72.
73. /* Modal Content */
74. .modal-content {
75.     font-size:3vh;
76.     position: relative;
77.     background-color: #fefefe;
78.     margin: auto;
79.     padding: 0;
80.     border: 1px solid #888;
81.     width: 80%;
82.     box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2),0 6px 20px 0 rgba(0,0,0,0.19);
83.     -webkit-animation-name: animatetop;
84.     -webkit-animation-duration: 0.4s;
85.     animation-name: animatetop;
86.     animation-duration: 0.4s
87. }
88.
89. /* Add Animation */
90. @-webkit-keyframes animatetop {
91.     from {top:-300px; opacity:0}
92.     to {top:0; opacity:1}
93. }
94.
95. @keyframes animatetop {
96.     from {top:-300px; opacity:0}
97.     to {top:0; opacity:1}
98. }
99.
100. /* The Close Button */
101. .close {
102.     color: white;
103.     float: right;
104.     font-size: 28px;
105.     font-weight: bold;
106. }
107.
108. .close:hover,
109. .close:focus {
110.     color: #000;
111.     text-decoration: none;
112.     cursor: pointer;
113. }
114.
115. .modal-header {
116.     padding: 2px 16px;
117.     background-color: #DF0101;
118.     color: white;
119. }

```

```

120.
121. .modal-body {padding: 2px 16px;}
122.
123. .modal-footer {
124.     padding: 2px 16px;
125.     background-color: #5cb85c;
126.     background-color: white;
127.     color: white;
128. }
129.
130. </style>
131. <body>
132.
133. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-
    align:center;">CONTROL USUARIOS</h1>
134.
135. <p style="font-size:3vw;background-color:#f2f2f2;text-align:left;">Pantalla Inicial a mostrar:
136. <select style="font-size:2vw;" id="selectorPantallaInicial" onchange="opcionInicioSeleccionada(this.value);">
137. <option value="menuPrincipal" style="font-size:2vw;color:Black;">Menú principal</option>
138. <option value="seleccionUsuario" style="font-
    size:2vw;color:Black;">Pantalla de seleccion de usuario</option>
139. </select>
140. </p>
141.
142. <table id="tablaUsuarios">
143. <tr>
144. <th>Id Usuario</th>
145. <th>Nombre</th>
146. <th>Borrar</th>
147. </tr>
148. <!-- <tr>
149. <td>355</td>
150. <td>Luis</td>
151. <td><div class="fa fa-trash"></div></td>
152. </tr>
153. <tr>
154. <td>753</td>
155. <td>Jackson</td>
156. <td><div class="fa fa-trash"></div></td>
157. </tr>
158. <tr>
159. <td>358</td>
160. <td>Johnson</td>
161. <td><div class="fa fa-trash"></div></td>
162. </tr -->
163. </table>
164.
165. <br />
166.
167. <button class="button" style="background-
    color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAInicio', 'dialogoGeneral')>VOLVER</butt
    on>
168.
169.
170. <!-- Panel para confirmar eliminacion de usuario -->
171. <div id="myModal" class="modal">
172.
173. <!-- Modal content -->
174. <div class="modal-content">
175. <div class="modal-header">
176. <span class="close">x</span>
177. <h2>ADVERTENCIA</h2>
178. </div>
179. <div class="modal-body">
180. <p>¿Está seguro de que desea eliminar el usuario seleccionado?</p>
181. <p id="userToDelete"></p>
182. </div>
183. <div class="modal-footer">

```

```

184. <!-- <h3>Modal Footer</h3> -->
185. <button class="button" style="background-
color: SlateBlue;height:15vh;width:35vw;" id="botonCancelar">CANCELAR</button>
186. <button class="button" style="background-
color: SlateBlue;height:15vh;width:35vw;" id="botonAceptar">ACEPTAR</button>
187. </div>
188. </div>
189.
190. </div>
191.
192.
193. <script src="/libs/qimessaging/2/qimessaging.js"></script>
194. <script src="/libs/qi/2/qi.js"></script>
195. <script type="text/javascript" src="js/scriptPepper.js"></script>
196.
197. <script>
198.
199. function obtieneValorSelectorPantallaInicial() //Obtiene de la memoria del robot el valor de pantalla inicial actual
200.
201.     var selectorPantallaInicial = document.getElementById("selectorPantallaInicial");
202.
203.     session.service('ALMemory').then(function (memory) {
204.         memory.getData("modoPantallaInicial").then(function (value) {
205.             var valor=parseInt(value);
206.             if(valor==1)
207.                 {
208.                     selectorPantallaInicial.value="menuPrincipal";
209.                 }else if(valor==2){
210.                     selectorPantallaInicial.value="seleccionUsuario";
211.                 }
212.             }, function (error) {
213.                 console.log(error);
214.             });
215.         }, function (error) {
216.             console.log(error);
217.         });
218.     }
219.
220. function opcionInicioSeleccionada(selectedValue){
221.     if(selectedValue.localeCompare("menuPrincipal")==0)
222.     {
223.         // Menu principal seleccionado
224.         pepperInsertIntData("modoPantallaInicial", 1);
225.     }else{
226.         // Pantalla de seleccion de usuario
227.         pepperInsertIntData("modoPantallaInicial", 2);
228.     }
229. }
230.
231. usuarioAborrar=0;
232.
233. function muestraError()
234. {
235.     var tr = document.createElement("tr");
236.     var td = document.createElement("td");
237.     td.innerHTML="No se ha podido obtener la lista de usuarios";
238.     tr.appendChild(td);
239.     tablaUsuarios.appendChild(tr);
240. }
241.
242. function setName(idUsuario)
243. {
244.     var myDomain = "com.aldebaran.myApp";
245.     session.service('ALUserInfo').then(function (userInfo) {
246.         userInfo.get(myDomain, parseInt(idUsuario), "name").then(function (userName) {
247.
248.             var elementoNombre=document.getElementById(String(idUsuario)+"_name");
249.

```

```

250.     elementoNombre.innerHTML=userName;
251.
252.     console.log('success');
253.   }, function (error) {
254.     console.log(error);
255.   });
256. }, function (error) {
257.   console.log(error);
258. });
259. }
260.
261. function deleteUser(idUsuario)
262. {
263.   session.service('ALMemory').then(function (memory) {
264.     memory.raiseEvent("EventoBorrarUsuario",idUsuario);
265.   }, function (error) {
266.     console.log(error);
267.   });
268. }
269.
270. var botonBorrarUsuario = function()
271. {
272.   usuarioAborrar=parseInt(this.id.substring(0, this.id.length-7)); //Quita el "_delete" y pasa a int
273.
274.   modal.style.display = "block";
275.
276.   var elementoUsuarioAborrar=document.getElementById("userToDelete");
277.
278.   elementoUsuarioAborrar.innerHTML="Id Usuario: "+String(usuarioAborrar);
279. }
280.
281. function actualizaLista()
282. {
283.   try {
284.
285.     // Borra la lista
286.     while (tablaUsuarios.firstChild) {
287.       tablaUsuarios.removeChild(tablaUsuarios.firstChild);
288.     }
289.     // Pone la cabecera
290.     var tr = document.createElement("tr");
291.     var td = document.createElement("td");
292.     td.innerHTML="Id Usuario";
293.     tr.appendChild(td);
294.     td = document.createElement("td");
295.     td.innerHTML="Nombre";
296.     tr.appendChild(td);
297.     td = document.createElement("td");
298.     td.innerHTML="Borrar";
299.     tr.appendChild(td);
300.     tablaUsuarios.appendChild(tr);
301.
302.     QiSession( function (s) {
303.
304.       console.log('connected!');
305.
306.       ssession = s;
307.
308.       session.service('ALUserSession').then(function (userSession) { //obtiene lista de usuarios permanentes
309.         //userSession.getPermanentUserList();
310.         userSession.getPermanentUserList().then(function (list) {
311.
312.           userList=list;
313.
314.           numUsuarios = userList.length;
315.
316.           for (i = 0; i < numUsuarios; i++) {
317.

```

```

318.         idUsuario=userList[i];
319.
320.         tr = document.createElement("tr");
321.         //Id
322.         td = document.createElement("td");
323.         td.innerHTML=idUsuario;//userList//[i];
324.         tr.appendChild(td);
325.
326.         //Name
327.         td = document.createElement("td");
328.         td.id=String(idUsuario)+"_name";
329.         setName(idUsuario); //obtiene el nombre
330.         tr.appendChild(td);
331.
332.         //Icono de borrar
333.         td = document.createElement("td");
334.         var div = document.createElement("div");
335.         div.className="fa fa-trash";
336.         div.id=String(idUsuario)+"_delete";
337.         div.onclick=botonBorrarUsuario;
338.         td.appendChild(div);
339.         tr.appendChild(td);
340.
341.         tablaUsuarios.appendChild(tr);
342.
343.     }
344.
345.     console.log('success');
346.     }, function (error) {
347.         console.log(error);
348.         muestraError();
349.     });
350.     }, function (error) {
351.         console.log(error);
352.         muestraError();
353.     });
354. });
355.
356. });
357.
358. } catch (err) {
359.
360.     console.log("Error when initializing QiSession: " + err.message);
361.
362.     console.log("Make sure you load this page from the robots server.");
363.
364.     muestraError();
365.
366. }
367. }
368.
369. // Get the modal
370. var modal = document.getElementById('myModal');
371.
372. // Get the <span> element that closes the modal
373. var span = document.getElementsByClassName("close")[0];
374.
375. // When the user clicks on <span> (x), close the modal
376. span.onclick = function() {
377.     modal.style.display = "none";
378. }
379.
380. // Cancelar
381. var botonCancelar = document.getElementById('botonCancelar');
382. botonCancelar.onclick = function() {
383.     modal.style.display = "none";
384. }
385.

```

```

386. // Aceptar
387. var botonAceptar = document.getElementById('botonAceptar');
388. botonAceptar.onclick = function() {
389.
390.   if(usuarioAborrar!=0)
391.   {
392.     deleteUser(usuarioAborrar);
393.
394.     pepperSay("Usuario borrado");
395.
396.     actualizaLista();
397.   }
398.
399.   modal.style.display = "none";
400. }
401.
402. tablaUsuarios=document.getElementById("tablaUsuarios");
403.
404. actualizaLista();
405.
406. setTimeout(obtieneValorSelectorPantallaInicial, 200); // Obtiene valor pantalla inicial
407.
408.
409. </script>
410.
411. </body>
412. </html>

```

Ajustes volumen

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4.   <title>Menu Tablet</title>
5.   <meta charset="UTF-8">
6. </head>
7. <style>
8.   body {background-color: LightGray;}
9.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
10.  p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
11.
12.  .button {
13.    padding: 15px 25px;
14.    font-size:5vw;
15.    width:47vw;
16.    height:27vh;
17.    text-align: center;
18.    cursor: pointer;
19.    outline: none;
20.    color: #fff;
21.    background-color: DodgerBlue;
22.    border: none;
23.    border-radius: 15px;
24.    box-shadow: 0 9px #999;
25.    margin-bottom: 3vh;
26.    margin-left: 0.5vw;
27.    margin-right: 0.5vw;
28.    float:left;
29.  }
30.
31.  .button:hover {background-color: DodgerBlue}
32.
33.  .button:active {
34.    background-color: DodgerBlue;
35.    box-shadow: 0 5px #666;
36.    transform: translateY(4px);
37.  }
38. </style>
39. <body>

```

```

40.
41. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MENU VOLUMEN</h1>
42.
43. <button class="button" onclick="pepperDialogGotoTag('ajustaVolumenBajo', 'dialogoGeneral')">VOLUMEN BA
JO</button>
44. <button class="button" onclick="pepperDialogGotoTag('ajustaVolumenMedio', 'dialogoGeneral')">VOLUMEN M
EDIO</button>
45. <button class="button" onclick="pepperDialogGotoTag('ajustaVolumenAlto', 'dialogoGeneral')">VOLUMEN AL
TO</button>
46. <button class="button" onclick="pepperDialogGotoTag('ajustaVolumenMaximo', 'dialogoGeneral')">VOLUMEN
MAXIMO</button>
47.
48. <button class="button" style="background-
color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAInicio', 'dialogoGeneral')">VOLVER</butt
on>
49.
50. <script src="/libs/qimessaging/2/qimessaging.js"></script>
51. <script src="/libs/qi/2/qi.js"></script>
52. <script type="text/javascript" src="js/scriptPepper.js"></script>
53.
54. </body>
55. </html>

```

Selección usuario

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10. body { background-color: LightGray; }
11. h1 { font-size:5vw; background-color:SlateBlue; color:White; text-align:center; }
12. p { font-size:3vw; background-color:SlateBlue; color:White; text-align:center; }
13.
14. .button {
15. padding: 15px 25px;
16. font-size:4vw;
17. width:55vw;
18. height:15vh;
19. text-align: center;
20. cursor: pointer;
21. outline: none;
22. color: #fff;
23. background-color: DodgerBlue;
24. border: none;
25. border-radius: 15px;
26. box-shadow: 0 9px #999;
27. margin-bottom: 3vh;
28. margin-left: 0.5vw;
29. margin-right: 0.5vw;
30. float:left;
31. }
32.
33. .button:hover { background-color: DodgerBlue }
34.
35. .button:active {
36. background-color: DodgerBlue;
37. box-shadow: 0 5px #666;
38. transform: translateY(4px);
39. }
40.
41. .card {
42. box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2);
43. transition: 0.3s;
44. width: 18.5%;

```

```

45.     border-radius: 5px;
46.     background-color: DodgerBlue;
47.     margin-bottom: 3vh;
48.     margin-left: 0.5vw;
49.     margin-right: 0.5vw;
50.     float:left;
51.   }
52.
53.   .card:hover {
54.     box-shadow: 0 8px 16px 0 rgba(0,0,0,0.2);
55.   }
56.
57.   img {
58.     border-radius: 5px 5px 0 0;
59.   }
60.
61.   .container {
62.     padding: 2px 16px;
63.   }
64.
65. </style>
66. <body>
67.
68. <!-- <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-
align:center;">SELECCIÓN DE USUARIO</h1> -->
69. <div style="position: fixed;top:0;float:center;background-color:SlateBlue;width:100%;height:20%;">
70. <p style="font-size:4vw;background-color:SlateBlue;color:White;text-
align:center;float:center;">SELECCIONE USUARIO</p>
71. </div>
72.
73. <p style="margin-bottom: 15%;">
74. </p>
75.
76. <!-- <button type="button" onclick="location.href='informacion.html';">INFORMACION</button>
77. <button type="button" onclick="location.href='juegos.html';">JUEGOS</button>
78. <button type="button" onclick="location.href='ejercicios.html';">EJERCICIOS</button>
79. <button type="button" onclick="location.href='musica.html';">MUSICA</button> -->
80. <!--
<button type="button" onclick="pepperDialogGotoTag('preguntaInformacion', 'dialogoGeneral')">INFORMACIO
N</button>
81. <button type="button" onclick="pepperDialogGotoTag('preguntaJuego', 'dialogoGeneral')">JUEGOS</button>
82. <button type="button" onclick="pepperDialogGotoTag('preguntaEjercicio', 'dialogoGeneral')">EJERCICIOS</but
ton>
83. <button type="button" onclick="pepperDialogGotoTag('preguntaMusica', 'dialogoGeneral')">MUSICA</button>
-->
84.
85. <!-- <div class="card">
86. 
87. <div class="container">
88. <p style="font-size:2vw;color:White;text-align:center;">Prueba</p>
89. </div>
90. </div> -->
91.
92. <div id="usuarios"></div>
93.
94. <div class="card" onclick="seleccionaUsuarioTemporal();">
95. 
96. <div class="container">
97. <p style="font-size:2vw;color:White;text-align:center;">Usuario Temporal</p>
98. </div>
99. </div>
100.
101. <div class="card" onclick="creaUsuarioNuevo();">
102. 
103. <div class="container">
104. <p style="font-size:2vw;color:White;text-align:center;">Nuevo Usuario</p>
105. </div>
106. </div>

```

```

107.
108. <!--
109. <button class="button" onclick="pepperDialogGotoTag('preguntaInformacion', 'dialogoGeneral')">INFORMACIO
N</button>
110. <button class="button" onclick="pepperDialogGotoTag('preguntaJuego', 'dialogoGeneral')">JUEGOS</button>
111. <button class="button" onclick="pepperDialogGotoTag('preguntaEjercicio', 'dialogoGeneral')">EJERCICIOS</bu
tton>
112. <button class="button" onclick="pepperDialogGotoTag('preguntaMusica', 'dialogoGeneral')">MUSICA</button>
113.
114. <button class="button" onclick="pepperDialogGotoTag('mostrarAjustes', 'dialogoGeneral')" style="height:8vw;wid
th:9vw;float:right;margin-right: 2.75vw;">
115. <div class="fa fa-cogs" style="font-size:5vw;color:white;" aria-hidden="true"></div>
116. </button>
117.
118. <button class="button" onclick="pepperDialogGotoTag('hacerValoracion', 'dialogoGeneral')" style="height:8vw;w
idth:9vw;float:right;">
119. <div class="fa fa-commenting-o" style="font-size:5vw;color:white;" aria-hidden="true"></div>
120. </button> -->
121.
122. <script src="/libs/qimessaging/2/qimessaging.js"></script>
123. <script src="/libs/qi/2/qi.js"></script>
124. <script type="text/javascript" src="js/scriptPepper.js"></script>
125.
126. <script>
127.
128. function setUser(idUsuario)
129. {
130.   session.service('ALUserSession').then(function (user_session) {
131.     user_session._openUserSessions([idUsuario]); // Abre sesion
132.     user_session._setFocusedUser(idUsuario); // Activa usuario
133.   }, function (error) {
134.     console.log(error);
135.   });
136. }
137.
138. var seleccionaUsuario = function()
139. {
140.   var idUsuario=parseInt(this.id); //Obtiene el id como int
141.
142.   session.service('ALUserSession').then(function (user_session) {
143.     user_session.getFocusedUser().then(function (user) { // Obtiene el usuario activo
144.
145.       var idUsuarioActual=parseInt(user);
146.
147.       if(idUsuario==idUsuarioActual) // comprueba si coincide con el usuario identificado
148.       {
149.         pepperDialogGotoTag('usuarioConfirmado', 'dialogoGeneral'); //confirma en dialogo
150.       }else{
151.         setUser(idUsuario); //activa el usuario seleccionado
152.         setTimeout(pepperDialogGotoTag, 300, 'usuarioConfirmado', 'dialogoGeneral'); //confirma en dialogo
153.       }
154.
155.     }, function (error) {
156.       console.log(error);
157.     });
158.   }, function (error) {
159.     console.log(error);
160.   });
161. }
162. }
163.
164. function creaUsuarioTemporal()
165. {
166.   session.service('ALUserSession').then(function (user_session) {
167.     user_session._createUsers(1).then(function (user_ids) { // Crea usuario
168.
169.       var idUsuario=parseInt(user_ids[0]);

```

```

170.
171.     setUser(idUsuario); //activa el usuario creado
172.
173.     setTimeout(pepperDialogGotoTag, 100, 'usuarioTemporalConfirmado', 'dialogoGeneral'); //confirma en dial
    ogo
174.
175.     }, function (error) {
176.         console.log(error);
177.     });
178. }, function (error) {
179.     console.log(error);
180. });
181. }
182.
183. function seleccionaUsuarioTemporal()
184. {
185.     session.service('ALUserSession').then(function (user_session) {
186.         user_session.getFocusedUser().then(function (user) { // Obtiene el usuario activo
187.
188.             var idUsuario=parseInt(user);
189.
190.             if(idUsuario<=0) // Usuario no identificado
191.             {
192.                 creaUsuarioTemporal(); // Crea usuario temporal
193.             }else{
194.                 user_session.isUserPermanent(idUsuario).then(function (value) { // Comprueba si el usuario activo es per
    manente
195.                     if(value===true)
196.                     {
197.                         creaUsuarioTemporal(); // Crea usuario temporal
198.                     }else{
199.                         // Mantiene el usuario temporal que ha dado el robot
200.                         pepperDialogGotoTag('usuarioTemporalConfirmado', 'dialogoGeneral'); //confirma en dialogo
201.                     }
202.                 }, function (error) {
203.                     console.log(error);
204.                 });
205.             }
206.
207.         }, function (error) {
208.             console.log(error);
209.         });
210.     }, function (error) {
211.         console.log(error);
212.     });
213. }
214.
215. function creaUsuarioPermanente()
216. {
217.     session.service('ALUserSession').then(function (user_session) {
218.         user_session._createUsers(1).then(function (user_ids) { // Crea usuario
219.
220.             var idUsuario=parseInt(user_ids[0]);
221.
222.             setUser(idUsuario); //activa el usuario creado
223.
224.             setTimeout(pepperDialogGotoTag, 100, 'creaUsuarioNuevo', 'dialogoGeneral'); // Hace foto para hacer usuar
    io permanente
225.
226.         }, function (error) {
227.             console.log(error);
228.         });
229.     }, function (error) {
230.         console.log(error);
231.     });
232. }
233.
234. function creaUsuarioNuevo()

```

```

235. {
236.   session.service('ALUserSession').then(function (user_session) {
237.     user_session.getFocusedUser().then(function (user) { // Obtiene el usuario activo
238.
239.       var idUsuario=parseInt(user);
240.
241.       if(idUsuario<=0) // Usuario no identificado
242.       {
243.         creaUsuarioPermanente(); // Crea usuario nuevo permanente
244.       }else{
245.         user_session.isUserPermanent(idUsuario).then(function (value) { // Comprueba si el usuario activo es per
manente
246.           if(value===true)
247.           {
248.             creaUsuarioPermanente(); // Crea usuario nuevo permanente
249.           }else{
250.             // Mantiene el usuario temporal que ha dado el robot
251.             pepperDialogGotoTag('creaUsuarioNuevo', 'dialogoGeneral'); // Hace foto para hacer usuario perman
ente
252.           }
253.         }, function (error) {
254.           console.log(error);
255.         });
256.       }
257.
258.     }, function (error) {
259.       console.log(error);
260.     });
261.   }, function (error) {
262.     console.log(error);
263.   });
264. }
265.
266. function setName(idUsuario)
267. {
268.   var myDomain = "com.aldebaran.myApp";
269.   session.service('ALUserInfo').then(function (userInfo) {
270.     userInfo.get(myDomain, parseInt(idUsuario), "name").then(function (userName) {
271.
272.       var elementoNombre=document.getElementById(String(idUsuario)+"_name");
273.
274.       elementoNombre.innerHTML=userName;
275.
276.       console.log('success');
277.     }, function (error) {
278.       console.log(error);
279.     });
280.   }, function (error) {
281.     console.log(error);
282.   });
283. }
284.
285. function setImage(idUsuario) // Obtiene del robot la imagen en base64
286. {
287.   var myDomain = "com.aldebaran.myApp";
288.   session.service('ALUserInfo').then(function (userInfo) {
289.     userInfo.get(myDomain, parseInt(idUsuario), "picture").then(function (imagen) {
290.
291.       var elementoImagen=document.getElementById(String(idUsuario)+"_image");
292.
293.       if(imagen.length>1)
294.         elementoImagen.src="data:image/jpeg;base64,"+imagen; //establece la imagen
295.
296.       console.log('success');
297.     }, function (error) {
298.       console.log(error);
299.     });
300.   }, function (error) {

```

```

301.     console.log(error);
302. });
303. }
304.
305. function creaElemento(idUsuario)
306. {
307.     var usuarios=document.getElementById("usuarios")
308.
309.     card = document.createElement("div");
310.     card.className="card";
311.     card.onclick=seleccionaUsuario;
312.     card.id=String(idUsuario);
313.     usuarios.appendChild(card);
314.
315.     var img = document.createElement("IMG");
316.     img.src="img/icono_usuario.png"; //imagen por defecto
317.     setImage(idUsuario); //obtiene la imagen
318.     img.alt="Avatar";
319.     img.style.width = "100%";
320.     img.id=String(idUsuario)+"_image";
321.     card.appendChild(img);
322.
323.     var container = document.createElement("div");
324.     container.className="container";
325.     card.appendChild(container);
326.
327.     var nombre = document.createElement("p");
328.     nombre.style.fontSize="2vw";
329.     nombre.style.color="White";
330.     nombre.style.textAlign="center";
331.     nombre.id=String(idUsuario)+"_name";
332.     nombre.innerHTML=String(idUsuario); // pone por defecto el id
333.     setName(idUsuario); //obtiene el nombre
334.
335.     container.appendChild(nombre);
336. }
337.
338. function actualizaLista()
339. {
340.     try {
341.
342.         QiSession( function (s) {
343.
344.             console.log('connected!');
345.
346.             ssession = s;
347.
348.             session.service('ALUserSession').then(function (userSession) {
349.                 //userSession.getPermanentUserList();
350.                 userSession.getPermanentUserList().then(function (list) { //obtiene lista de usuarios permanentes
351.
352.                     userList=list;
353.
354.                     numUsuarios = userList.length;
355.
356.                     for (i = 0; i < numUsuarios; i++) {
357.
358.                         var idUsuario=userList[i];
359.
360.                         creaElemento(idUsuario);
361.
362.                     }
363.
364.
365.                 console.log('success');
366.             }, function (error) {
367.                 console.log(error);
368.                 muestraError();

```

```

369.     });
370.     }, function (error) {
371.         console.log(error);
372.         muestraError();
373.     });
374.
375. });
376.
377. } catch (err) {
378.
379.     console.log("Error when initializing QiSession: " + err.message);
380.
381.     console.log("Make sure you load this page from the robots server.");
382.
383.     muestraError();
384.
385. }
386. }
387.
388. actualizaLista(); //actualiza la lista de usuarios
389.
390. </script>
391.
392. </body>
393. </html>

```

Crea usuario

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10.     body {background-color: LightGray;}
11.     h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12.     p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14.     .button {
15.         padding: 15px 25px;
16.         font-size:5vw;
17.         width:47vw;
18.         height:27vh;
19.         text-align: center;
20.         cursor: pointer;
21.         outline: none;
22.         color: #fff;
23.         background-color: DodgerBlue;
24.         border: none;
25.         border-radius: 15px;
26.         box-shadow: 0 9px #999;
27.         margin-bottom: 3vh;
28.         margin-left: 0.5vw;
29.         margin-right: 0.5vw;
30.         float:left;
31.     }
32.
33.     .button:hover {background-color: DodgerBlue}
34.
35.     .button:active {
36.         background-color: DodgerBlue;
37.         box-shadow: 0 5px #666;
38.         transform: translateY(4px);
39.     }
40.
41.     canvas {

```

```

42.     padding: 0;
43.     margin: auto;
44.     display: block;
45.     width: 480px;
46. }
47. </style>
48. <body>
49.
50. <h1 style="font-size:4vw;background-color:SlateBlue;color:White;text-align:center;">FOTO USUARIO</h1>
51.
52. <div align="center">
53. <img id="imagenCamara" alt="Camara" height="480">
54. </div>
55.
56. <!-- <br/> -->
57.
58. <!-- <button class="button" style="background-
color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAInicio', 'dialogoGeneral')">VOLVER</butt
on> -->
59.
60. <script src="/libs/qimessaging/2/qimessaging.js"></script>
61. <script src="/libs/qi/2/qi.js"></script>
62. <script type="text/javascript" src="js/scriptPepper.js"></script>
63.
64. <script>
65.
66. var elementoImagen=document.getElementById("imagenCamara");
67.
68. function setImage() // Obtiene del robot la imagen en base64
69. {
70.     var myDomain = "com.aldebaran.myApp";
71.     session.service('ALMemory').then(function (memory) {
72.         memory.getData("CamaraVideo").then(function (imagen) {
73.
74.             if(imagen.length>1)
75.                 elementoImagen.src="data:image/jpeg;base64,"+imagen; //establece la imagen
76.
77.             console.log('success');
78.         }, function (error) {
79.             console.log(error);
80.         });
81.     }, function (error) {
82.         console.log(error);
83.     });
84. }
85.
86. //setInterval(setImage, 40);
87.
88.
89. QiSession(function (session) {
90.     session.service("ALMemory").then(function (ALMemory) {
91.         ALMemory.subscriber("CamaraVideo").then(function (subscriber) {
92.             // subscriber.signal is a signal associated to "CamaraVideo"
93.             subscriber.signal.connect(function (imagen) {
94.                 if(imagen.length>1)
95.                     elementoImagen.src="data:image/jpeg;base64,"+imagen; //establece la imagen
96.             });
97.         });
98.     });
99. }, function () {
100.     console.log("disconnected");
101. });
102.
103.
104. </script>
105.
106. </body>
107. </html>

```

Información

```
1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4.   <title>Menu Tablet</title>
5.   <meta charset="UTF-8">
6. </head>
7. <style>
8.   body {background-color: LightGray;}
9.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
10.  p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
11.
12.  .button {
13.    padding: 15px 25px;
14.    font-size:5vw;
15.    width:47vw;
16.    height:27vh;
17.    text-align: center;
18.    cursor: pointer;
19.    outline: none;
20.    color: #fff;
21.    background-color: DodgerBlue;
22.    border: none;
23.    border-radius: 15px;
24.    box-shadow: 0 9px #999;
25.    margin-bottom: 3vh;
26.    margin-left: 0.5vw;
27.    margin-right: 0.5vw;
28.    float:left;
29.  }
30.
31.  .button:hover {background-color: DodgerBlue}
32.
33.  .button:active {
34.    background-color: DodgerBlue;
35.    box-shadow: 0 5px #666;
36.    transform: translateY(4px);
37.  }
38. </style>
39. <body>
40.
41. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MENU INFORMACION</h1>
42.
43. <button class="button" onclick="pepperDialogGotoTag('decirClima', 'dialogoGeneral')>TIEMPO</button>
44. <!-- <button type="button" onclick="location.href='noticias.html';>NOTICIAS</button> -->
45. <button class="button" onclick="pepperDialogGotoTag('preguntaNoticias', 'dialogoGeneral')>NOTICIAS</button>
46. <button class="button" onclick="pepperDialogGotoTag('buscarInformacion', 'dialogoGeneral')>BUSCAR INFORMACION</button>
47. <!--
48.   <button class="button" onclick="pepperDialogGotoTag('muestraMapas', 'dialogoGeneral')>VER MAPA</button>
49. -->
50. <button class="button" style="background-color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAInicio', 'dialogoGeneral')>VOLVER</button>
51.
52. <script src="/libs/qimessaging/2/qimessaging.js"></script>
53. <script src="/libs/qi/2/qi.js"></script>
54. <script type="text/javascript" src="js/scriptPepper.js"></script>
55.
56. </body>
57. </html>
```

Noticias

```
1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4.   <title>Menu Tablet</title>
5.   <meta charset="UTF-8">
6. </head>
7. <style>
8.   body {background-color: LightGray;}
9.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
10.  p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
11.
12.  .button {
13.    padding: 15px 25px;
14.    font-size:5vw;
15.    width:47vw;
16.    height:27vh;
17.    text-align: center;
18.    cursor: pointer;
19.    outline: none;
20.    color: #fff;
21.    background-color: DodgerBlue;
22.    border: none;
23.    border-radius: 15px;
24.    box-shadow: 0 9px #999;
25.    margin-bottom: 3vh;
26.    margin-left: 0.5vw;
27.    margin-right: 0.5vw;
28.    float:left;
29.  }
30.
31.  .button:hover {background-color: DodgerBlue}
32.
33.  .button:active {
34.    background-color: DodgerBlue;
35.    box-shadow: 0 5px #666;
36.    transform: translateY(4px);
37.  }
38. </style>
39. <body>
40.
41. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MENU NOTICIAS</h1>
42.
43. <button class="button" onclick="pepperDialogGotoTag('noticiasGenerales', 'dialogoGeneral')>GENERALES</button>
44. <button class="button" onclick="pepperDialogGotoTag('noticiasDeportes', 'dialogoGeneral')>DEPORTES</button>
45. <button class="button" onclick="pepperDialogGotoTag('noticiasEconomia', 'dialogoGeneral')>ECONOMIA</button>
46. <button class="button" onclick="pepperDialogGotoTag('noticiasLocales', 'dialogoGeneral')>LOCALES</button>
47.
48. <button class="button" style="background-color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('preguntaInformacion', 'dialogoGeneral')>VOLVER</button>
49.
50. <script src="/libs/qimessaging/2/qimessaging.js"></script>
51. <script src="/libs/qi/2/qi.js"></script>
52. <script type="text/javascript" src="js/scriptPepper.js"></script>
53.
54. </body>
55. </html>
```

Juegos

```
1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4.   <title>Menu Tablet</title>
5.   <meta charset="UTF-8">
6. </head>
7. <style>
8.   body {background-color: LightGray;}
9.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
10.  p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
11.
12.  .button {
13.    padding: 15px 25px;
14.    font-size:5vw;
15.    width:47vw;
16.    height:27vh;
17.    text-align: center;
18.    cursor: pointer;
19.    outline: none;
20.    color: #fff;
21.    background-color: DodgerBlue;
22.    border: none;
23.    border-radius: 15px;
24.    box-shadow: 0 9px #999;
25.    margin-bottom: 3vh;
26.    margin-left: 0.5vw;
27.    margin-right: 0.5vw;
28.    float:left;
29.  }
30.
31.  .button:hover {background-color: DodgerBlue}
32.
33.  .button:active {
34.    background-color: DodgerBlue;
35.    box-shadow: 0 5px #666;
36.    transform: translateY(4px);
37.  }
38. </style>
39. <body>
40.
41. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MENU JUEGOS</h1>
42.
43. <!-- <button class="button" onclick="pepperDialogLaunchTabletApp('juegoencuentrapareja-
44. c80554')">ENCUENTRA PAREJA</button>
45. <button class="button" onclick="pepperDialogLaunchTabletApp('juegoexplotaglobos-
46. d56c11')">EXPLOTA GLOBOS</button> -->
47. <button class="button" onclick="pepperDialogGotoTag('juegoParejas', 'dialogoGeneral')">ENCUENTRA PAREJ
48. A</button>
49. <button class="button" onclick="pepperDialogGotoTag('juegoGlobos', 'dialogoGeneral')">EXPLOTA GLOBOS</
50. button>
51. <button class="button" onclick="pepperDialogGotoTag('cuentaChiste', 'dialogoGeneral')">CONTAR CHISTE</b
52. utton>
53. <button class="button" onclick="pepperDialogGotoTag('juegoAdivinanzas', 'dialogoGeneral')">ADIVINANZAS<
54. /button>
55.
56. <button class="button" style="background-
57. color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAinicio', 'dialogoGeneral')">VOLVER</butt
58. on>
59.
60. <script src="/libs/qimessaging/2/qimessaging.js"></script>
61. <script src="/libs/qi/2/qi.js"></script>
62. <script type="text/javascript" src="js/scriptPepper.js"></script>
63.
64. </body>
65. </html>
```

Adivinanzas

```
1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4.   <title>Menu Tablet</title>
5.   <meta charset="UTF-8">
6. </head>
7. <style>
8.   body {background-color: LightGray;}
9.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
10.  p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
11.
12.  .button {
13.    padding: 15px 25px;
14.    font-size:4vw;
15.    width:47vw;
16.    height:18vh;
17.    text-align: center;
18.    cursor: pointer;
19.    outline: none;
20.    color: #fff;
21.    background-color: DodgerBlue;
22.    border: none;
23.    border-radius: 15px;
24.    box-shadow: 0 9px #999;
25.    margin-bottom: 3vh;
26.    margin-left: 0.5vw;
27.    margin-right: 0.5vw;
28.    float:left;
29.  }
30.
31.  .button:hover {background-color: DodgerBlue}
32.
33.  .button:active {
34.    background-color: DodgerBlue;
35.    box-shadow: 0 5px #666;
36.    transform: translateY(4px);
37.  }
38.
39.  #snackbar {
40.    visibility: hidden;
41.    min-width: 250px;
42.    margin-left: -125px;
43.    background-color: SlateBlue;
44.    color: #fff;
45.    text-align: center;
46.    border-radius: 2px;
47.    padding: 16px;
48.    position: fixed;
49.    z-index: 1;
50.    left: 25%;
51.    bottom: 30vh;
52.    font-size: 5vw;
53.  }
54.
55.  #snackbar.show {
56.    visibility: visible;
57.    -webkit-animation: fadein 0.5s, fadeout 0.5s 2.5s;
58.    animation: fadein 0.5s, fadeout 0.5s 2.5s;
59.  }
60.
61.  @-webkit-keyframes fadein {
62.    from {bottom: 0; opacity: 0;}
63.    to {bottom: 30vh; opacity: 1;}
64.  }
65.
66.  @keyframes fadein {
67.    from {bottom: 0; opacity: 0;}
```

```

68.     to {bottom: 30vh; opacity: 1;}
69.   }
70.
71.   @-webkit-keyframes fadeout {
72.     from {bottom: 30vh; opacity: 1;}
73.     to {bottom: 0; opacity: 0;}
74.   }
75.
76.   @keyframes fadeout {
77.     from {bottom: 30vh; opacity: 1;}
78.     to {bottom: 0; opacity: 0;}
79.   }
80.
81.   /* The Modal (background) */
82.   .modal {
83.     display: none; /* Hidden by default */
84.     position: fixed; /* Stay in place */
85.     z-index: 1; /* Sit on top */
86.     padding-top: 100px; /* Location of the box */
87.     left: 0;
88.     top: 0;
89.     width: 100%; /* Full width */
90.     height: 100%; /* Full height */
91.     overflow: auto; /* Enable scroll if needed */
92.     background-color: rgb(0,0,0); /* Fallback color */
93.     background-color: rgba(0,0,0,0.4); /* Black w/ opacity */
94.   }
95.
96.   /* Modal Content */
97.   .modal-content {
98.     font-size:3vh;
99.     position: relative;
100.    background-color: #fefefe;
101.    margin: auto;
102.    padding: 0;
103.    border: 1px solid #888;
104.    width: 80%;
105.    box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2),0 6px 20px 0 rgba(0,0,0,0.19);
106.    -webkit-animation-name: animatetop;
107.    -webkit-animation-duration: 0.4s;
108.    animation-name: animatetop;
109.    animation-duration: 0.4s
110.  }
111.
112.  /* Add Animation */
113.  @-webkit-keyframes animatetop {
114.    from {top:-300px; opacity:0}
115.    to {top:0; opacity:1}
116.  }
117.
118.  @keyframes animatetop {
119.    from {top:-300px; opacity:0}
120.    to {top:0; opacity:1}
121.  }
122.
123.  .modal-header {
124.    padding: 2px 16px;
125.    background-color: SlateBlue;
126.    color: white;
127.    text-align: center;
128.    font-size:3vw;
129.  }
130.
131.  .modal-body {padding: 2px 16px; text-align: center;font-size:3vw;}
132.
133.  .modal-footer {
134.    padding: 2px 16px;
135.    background-color: #5cb85c;

```

```

136.     background-color: white;
137.     color: white;
138. }
139. </style>
140. <body onload="cargaJuego();">
141.
142. <!-- <h1 style="font-size:3vw;background-color:SlateBlue;color:White;text-align:center;">ADIVINANZAS</h1> -
->
143.
144. <p id="texto" style="font-size:4vw;background-color:SlateBlue;color:White;text-align:center;"></p>
145.
146. <button class="button" id="opcion1" onclick="seleccionaOpcion(1)"></button>
147. <button class="button" id="opcion2" onclick="seleccionaOpcion(2)"></button>
148. <button class="button" id="opcion3" onclick="seleccionaOpcion(3)"></button>
149. <button class="button" id="opcion4" onclick="seleccionaOpcion(4)"></button>
150.
151. <div id="snackbar">Mensaje</div>
152.
153. <button class="button" style="background-
color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverInicio', 'dialogoGeneral')>SALIR</button
>
154.
155. <!-- Panel Inicio -->
156. <div id="myModal" class="modal">
157.
158. <div class="modal-content">
159. <div class="modal-header">
160. <h2 id="tituloPanel">JUEGO ADIVINANZAS</h2>
161. </div>
162. <div class="modal-body">
163. <p id="textoPanel">Selecciona la opción correcta en las siguientes 10 adivinanzas</p>
164. </div>
165. <div class="modal-footer">
166. <button class="button" style="background-
color: SlateBlue;height:20vh;width:35vw;" id="botonCancelar">SALIR</button>
167. <button class="button" style="background-
color: SlateBlue;height:20vh;width:35vw;" id="botonAceptar">EMPEZAR</button>
168. </div>
169. </div>
170.
171. </div>
172.
173. <script src="/libs/qimessaging/2/qimessaging.js"></script>
174. <script src="/libs/qi/2/qi.js"></script>
175. <script type="text/javascript" src="js/scriptPepper.js"></script>
176.
177. <script>
178.
179. // Obtiene el texto y botones
180. var texto = document.getElementById("texto");
181. var opcion1 = document.getElementById("opcion1");
182. var opcion2 = document.getElementById("opcion2");
183. var opcion3 = document.getElementById("opcion3");
184. var opcion4 = document.getElementById("opcion4");
185.
186. var opcionCorrecta=4;
187.
188. var fallos=0;
189. var aciertos=0;
190. var puntuacion=0;
191.
192. var contador=0;
193.
194. function shuffle(array) {
195.     var i = array.length,
196.         j = 0,
197.         temp;
198.

```

```

199. while (i--) {
200.
201.     j = Math.floor(Math.random() * (i+1));
202.
203.     // swap randomly chosen element with current element
204.     temp = array[i];
205.     array[i] = array[j];
206.     array[j] = temp;
207.
208. }
209.
210. return array;
211. }
212.
213. var ranNums = shuffle([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,
34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50]);
214.
215. function iniciaJuego()
216. {
217.     contador=0;
218.     aciertos=0;
219.     fallos=0;
220.
221.     ranNums = shuffle([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,3
4,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50]);
222.
223.     cargaAdvinanza(ranNums[contador]);
224. }
225.
226. function cargaAdvinanza(indice)
227. {
228.     // Carga texto y opciones
229.
230.     if(indice==1) {
231.         texto.innerHTML="Aunque no soy importante, en la vida pinto algo; mas no podré trabajar cuando yo me que
de calvo";
232.         opcion1.innerHTML="El ruibarbo (planta)";
233.         opcion2.innerHTML="La flor del algodón";
234.         opcion3.innerHTML="La tintorera (pez)";
235.         opcion4.innerHTML="El pincel";
236.         opcionCorrecta=4;
237.     }else if(indice==2) {
238.         texto.innerHTML="Primero ciega, luego pincha y todo une mientras camina";
239.         opcion1.innerHTML="Los rayos del sol";
240.         opcion2.innerHTML="La aguja";
241.         opcion3.innerHTML="La justicia";
242.         opcion4.innerHTML="El amor";
243.         opcionCorrecta=2;
244.     }else if(indice==3) {
245.         texto.innerHTML=";Escapa, escapa! que esto que te digo, aunque no te obligo, te abriga y te tapa";
246.         opcion1.innerHTML="Las botas";
247.         opcion2.innerHTML="La casa";
248.         opcion3.innerHTML="La sombra";
249.         opcion4.innerHTML="La capa";
250.         opcionCorrecta=4;
251.     }else if(indice==4) {
252.         texto.innerHTML="Mi nombre es de peregrino y tengo virtud notable, me encuentras en los caminos y mi olor
es agradable";
253.         opcion1.innerHTML="El perejil";
254.         opcion2.innerHTML="El romero";
255.         opcion3.innerHTML="La calabaza del peregrino";
256.         opcion4.innerHTML="La sopa peregrina";
257.         opcionCorrecta=2;
258.     }else if(indice==5) {
259.         texto.innerHTML="¿Qué cosa es esa cosa que entra en el río y no se moja?";
260.         opcion1.innerHTML="El pez";
261.         opcion2.innerHTML="Los rayos del sol";
262.         opcion3.innerHTML="El cisne";

```

```

263.     opcion4.innerHTML="El agua";
264.     opcionCorrecta=2;
265. }else if(indice==6) {
266.     texto.innerHTML="Dos guaridas cálidas con sus escondrijos, para dos hermanas y sus quintillizos";
267.     opcion1.innerHTML="La loba";
268.     opcion2.innerHTML="El topo";
269.     opcion3.innerHTML="Los guantes";
270.     opcion4.innerHTML="La madriguera";
271.     opcionCorrecta=3;
272. }else if(indice==7) {
273.     texto.innerHTML="No es león y tiene garra, no es pato y tiene pata";
274.     opcion1.innerHTML="El canguro";
275.     opcion2.innerHTML="La pata";
276.     opcion3.innerHTML="La garrapata";
277.     opcion4.innerHTML="La pierna";
278.     opcionCorrecta=3;
279. }else if(indice==8) {
280.     texto.innerHTML="Cerca del polo, desnuda, sentada sobre una roca, suave, negra, bigotuda";
281.     opcion1.innerHTML="La morsa";
282.     opcion2.innerHTML="La ballena";
283.     opcion3.innerHTML="La foca";
284.     opcion4.innerHTML="La nieve";
285.     opcionCorrecta=3;
286. }else if(indice==9) {
287.     texto.innerHTML="Es un gran señorón, tiene verde sombrero y pantalón marrón";
288.     opcion1.innerHTML="El guardia forestal";
289.     opcion2.innerHTML="El paraguas";
290.     opcion3.innerHTML="El árbol";
291.     opcion4.innerHTML="El verderón (pájaro)";
292.     opcionCorrecta=3;
293. }else if(indice==10) {
294.     texto.innerHTML="De bronce el tronco, las hojas de esmeralda, el fruto de oro, las flores de plata";
295.     opcion1.innerHTML="El limonero";
296.     opcion2.innerHTML="El manzano";
297.     opcion3.innerHTML="El naranjo";
298.     opcion4.innerHTML="La antena de televisión";
299.     opcionCorrecta=3;
300. }else if(indice==11) {
301.     texto.innerHTML="Giro mi cuerpo ante el sol, por ser mi dueño y señor";
302.     opcion1.innerHTML="Los planetas";
303.     opcion2.innerHTML="El arcoiris";
304.     opcion3.innerHTML="La peonza";
305.     opcion4.innerHTML="El girasol";
306.     opcionCorrecta=4;
307. }else if(indice==12) {
308.     texto.innerHTML="Todos me pisan a mí, pero yo no piso a nadie; todos preguntan por mí, yo no pregunto por
nadie";
309.     opcion1.innerHTML="El calcetín";
310.     opcion2.innerHTML="La calle";
311.     opcion3.innerHTML="El paso cebra";
312.     opcion4.innerHTML="La suela del zapato";
313.     opcionCorrecta=2;
314. }else if(indice==13) {
315.     texto.innerHTML="El cielo y la tierra se van a juntar; la ola y la nube se van a enredar. Vayas donde vayas sie
mpre lo verás, por mucho que andes nunca llegarás";
316.     opcion1.innerHTML="El cielo";
317.     opcion2.innerHTML="El humo del barco";
318.     opcion3.innerHTML="El horizonte";
319.     opcion4.innerHTML="El arcoiris";
320.     opcionCorrecta=3;
321. }else if(indice==14) {
322.     texto.innerHTML="Me encuentras en la playa a la sombra y al sol, mi nombre tiene cara y también tiene col";
323.     opcion1.innerHTML="El caracol";
324.     opcion2.innerHTML="La sombrilla";
325.     opcion3.innerHTML="El turista";
326.     opcion4.innerHTML="Las pipas de girasol";
327.     opcionCorrecta=1;
328. }else if(indice==15) {

```

```

329. texto.innerHTML="Tan redonda como un queso, nadie puede darle un beso";
330. opcion1.innerHTML="La magdalena";
331. opcion2.innerHTML="La luna";
332. opcion3.innerHTML="La botella";
333. opcion4.innerHTML="La taza";
334. opcionCorrecta=2;
335. }else if(indice==16) {
336. texto.innerHTML="Pobrecito, pobrecito, todo el día sin parar y no sale de su sitio";
337. opcion1.innerHTML="El rio";
338. opcion2.innerHTML="El reloj";
339. opcion3.innerHTML="El telefono";
340. opcion4.innerHTML="El loro";
341. opcionCorrecta=2;
342. }else if(indice==17) {
343. texto.innerHTML="Tengo escamas pero no soy pez, tengo corona pero no soy reina ¿quién soy?";
344. opcion1.innerHTML="El Punto";
345. opcion2.innerHTML="El hilo";
346. opcion3.innerHTML="La piña";
347. opcion4.innerHTML="El dragón";
348. opcionCorrecta=3;
349. }else if(indice==18) {
350. texto.innerHTML="Sombrero grande para el mundo entero, de día está claro y de noche está oscuro, ¿que es?";

351. opcion1.innerHTML="El cielo";
352. opcion2.innerHTML="El pincel";
353. opcion3.innerHTML="Los guisantes";
354. opcion4.innerHTML="El techo";
355. opcionCorrecta=1;
356. }else if(indice==19) {
357. texto.innerHTML="En verdes ramas nací, en molino me estrujaron, en un pozo me metí, y del pozo me sacaron a la cocina a freír";
358. opcion1.innerHTML="La harina";
359. opcion2.innerHTML="El elefante";
360. opcion3.innerHTML="El aceite";
361. opcion4.innerHTML="El vino";
362. opcionCorrecta=3;
363. }else if(indice==20) {
364. texto.innerHTML="Tengo un traje verde todo arrugadito; lo lavo en los charcos, lo seco al solecito";
365. opcion1.innerHTML="El sapo";
366. opcion2.innerHTML="El chubasquero";
367. opcion3.innerHTML="El arbol";
368. opcion4.innerHTML="La planta";
369. opcionCorrecta=1;
370. }else if(indice==21) {
371. texto.innerHTML="En todos los dias de la semana me hallarás, pero no así en domingo, en el que no me encontrarás";
372. opcion1.innerHTML="El sol";
373. opcion2.innerHTML="El cielo";
374. opcion3.innerHTML="La tienda";
375. opcion4.innerHTML="La letra S";
376. opcionCorrecta=4;
377. }else if(indice==22) {
378. texto.innerHTML="El sol me adormece, la lluvia me despierta, me abre y me pone alerta";
379. opcion1.innerHTML="El Murcielago";
380. opcion2.innerHTML="El cielo";
381. opcion3.innerHTML="El paraguas";
382. opcion4.innerHTML="Las nubes";
383. opcionCorrecta=3;
384. }else if(indice==23) {
385. texto.innerHTML="No soy cama ni león, pero cambio de color";
386. opcion1.innerHTML="La luciérnaga";
387. opcion2.innerHTML="El camaleón";
388. opcion3.innerHTML="La televisión";
389. opcion4.innerHTML="La lámpara";
390. opcionCorrecta=2;
391. }else if(indice==24) {
392. texto.innerHTML="Quema la piel, es redondo y lo vemos en el cielo";
393. opcion1.innerHTML="El sol";

```

```

394.   opcion2.innerHTML="El fuego";
395.   opcion3.innerHTML="La luna";
396.   opcion4.innerHTML="Las nubes";
397.   opcionCorrecta=1;
398. }else if(indice==25) {
399.   texto.innerHTML="Cuanto más y más lo llenas, menos pesa";
400.   opcion1.innerHTML="El saco";
401.   opcion2.innerHTML="El globo";
402.   opcion3.innerHTML="La bolsa";
403.   opcion4.innerHTML="La maleta";
404.   opcionCorrecta=2;
405. }else if(indice==26) {
406.   texto.innerHTML="Tengo nombre de animal y cuando la rueda pincha, me tienes que utilizar";
407.   opcion1.innerHTML="Los alicates";
408.   opcion2.innerHTML="La llave inglesa";
409.   opcion3.innerHTML="El tigre";
410.   opcion4.innerHTML="El gato";
411.   opcionCorrecta=4;
412. }else if(indice==27) {
413.   texto.innerHTML="Tengo alas, pero no vuelo, me encantan los peces y soy muy buen nadador";
414.   opcion1.innerHTML="El pingüino";
415.   opcion2.innerHTML="El perezoso";
416.   opcion3.innerHTML="El gallo";
417.   opcion4.innerHTML="El pato";
418.   opcionCorrecta=1;
419. }else if(indice==28) {
420.   texto.innerHTML="Llevo pijama a diario, sin guardarlo en el armario";
421.   opcion1.innerHTML="La cebra";
422.   opcion2.innerHTML="El tigre";
423.   opcion3.innerHTML="El perro";
424.   opcion4.innerHTML="El Loro";
425.   opcionCorrecta=1;
426. }else if(indice==29) {
427.   texto.innerHTML="Tengo una hermana gemela y vamos siempre al compás, con la boca por delante y los ojos
por detrás";
428.   opcion1.innerHTML="La canción";
429.   opcion2.innerHTML="La paloma";
430.   opcion3.innerHTML="Las tijeras";
431.   opcion4.innerHTML="La moto";
432.   opcionCorrecta=3;
433. }else if(indice==30) {
434.   texto.innerHTML="Me llegan las cartas y no sé leer y aunque me las trago, no mancho el papel";
435.   opcion1.innerHTML="El buzón";
436.   opcion2.innerHTML="El cartero";
437.   opcion3.innerHTML="La oficina";
438.   opcion4.innerHTML="La casa";
439.   opcionCorrecta=1;
440. }else if(indice==31) {
441.   texto.innerHTML="De mi casa sale, a mi casa llega, a mi casa trae, de mi casa lleva";
442.   opcion1.innerHTML="El camino";
443.   opcion2.innerHTML="El punto";
444.   opcion3.innerHTML="El coche";
445.   opcion4.innerHTML="El cartero";
446.   opcionCorrecta=1;
447. }else if(indice==32) {
448.   texto.innerHTML="Dos bolitas de cristal se abren y se cierran sin parar";
449.   opcion1.innerHTML="Las canicas";
450.   opcion2.innerHTML="Los ojos";
451.   opcion3.innerHTML="Las estrellas";
452.   opcion4.innerHTML="Los guisantes";
453.   opcionCorrecta=2;
454. }else if(indice==33) {
455.   texto.innerHTML="Soy un animal, tengo todas las vocales y vuelo como nadie";
456.   opcion1.innerHTML="La chirmoya";
457.   opcion2.innerHTML="La cigüeña";
458.   opcion3.innerHTML="El gorrión";
459.   opcion4.innerHTML="El murciélago";
460.   opcionCorrecta=4;

```

```

461. }else if(indice==34) {
462.     texto.innerHTML="Nieto de tu bisabuelo, padre de tus hermanos, de tus primos el tío y de tus tíos hermano";
463.     opcion1.innerHTML="El padre";
464.     opcion2.innerHTML="El sobrino";
465.     opcion3.innerHTML="El tío";
466.     opcion4.innerHTML="El abuelo";
467.     opcionCorrecta=1;
468. }else if(indice==35) {
469.     texto.innerHTML="¿Qué es lo que se moja al secar?";
470.     opcion1.innerHTML="El aire";
471.     opcion2.innerHTML="El agua";
472.     opcion3.innerHTML="La toalla";
473.     opcion4.innerHTML="El cuerpo";
474.     opcionCorrecta=3;
475. }else if(indice==36) {
476.     texto.innerHTML="Soy muy blanca como la nieve y dulce como la miel, yo alegro los pasteles y la leche con c
afé";
477.     opcion1.innerHTML="La sal";
478.     opcion2.innerHTML="La harina";
479.     opcion3.innerHTML="El azúcar";
480.     opcion4.innerHTML="La sacarina";
481.     opcionCorrecta=3;
482. }else if(indice==37) {
483.     texto.innerHTML="Soy peluda y de color marrón por fuera y por dentro soy verde";
484.     opcion1.innerHTML="La manzana";
485.     opcion2.innerHTML="El coco";
486.     opcion3.innerHTML="El Kiwi";
487.     opcion4.innerHTML="La patata";
488.     opcionCorrecta=3;
489. }else if(indice==38) {
490.     texto.innerHTML="Soy amiga de la luna, soy enemiga del sol; si viene la luz del día, alzo mi luz y me voy";
491.     opcion1.innerHTML="Las estrellas";
492.     opcion2.innerHTML="La luciérnaga";
493.     opcion3.innerHTML="El grillo";
494.     opcion4.innerHTML="El búho";
495.     opcionCorrecta=2;
496. }else if(indice==39) {
497.     texto.innerHTML="Son mis colores tan brillantes que el cielo alegro en un instante";
498.     opcion1.innerHTML="El sol";
499.     opcion2.innerHTML="La farola";
500.     opcion3.innerHTML="El arco iris";
501.     opcion4.innerHTML="La luna";
502.     opcionCorrecta=3;
503. }else if(indice==40) {
504.     texto.innerHTML="Desde el día que nací, corro y corro sin parar, corro de noche, corro de día hasta llegar al
mar";
505.     opcion1.innerHTML="El río";
506.     opcion2.innerHTML="El guepardo";
507.     opcion3.innerHTML="El viento";
508.     opcion4.innerHTML="La sombra";
509.     opcionCorrecta=1;
510. }else if(indice==41) {
511.     texto.innerHTML="Nunca podrás alcanzarme aunque corras trás de mi; aunque quieras alejarte, siempre estaré
junto a ti";
512.     opcion1.innerHTML="El sol";
513.     opcion2.innerHTML="La sombra";
514.     opcion3.innerHTML="El río";
515.     opcion4.innerHTML="El horizonte";
516.     opcionCorrecta=2;
517. }else if(indice==42) {
518.     texto.innerHTML="Es, cuando no es, y no es, cuando es. ¿Qué es?";
519.     opcion1.innerHTML="El silencio";
520.     opcion2.innerHTML="Yo";
521.     opcion3.innerHTML="La mentira";
522.     opcion4.innerHTML="El día";
523.     opcionCorrecta=3;
524. }else if(indice==43) {
525.     texto.innerHTML="Si me escribes como es, soy el rey de la selva, si me escribes al revés, yo soy papa noel";

```

```

526.   opcion1.innerHTML="El oso";
527.   opcion2.innerHTML="El guepardo";
528.   opcion3.innerHTML="El tigre";
529.   opcion4.innerHTML="El león";
530.   opcionCorrecta=4;
531. }else if(indice==44) {
532.   texto.innerHTML="Tengo agujas y no se coser, tengo números y no se leer";
533.   opcion1.innerHTML="El costurero";
534.   opcion2.innerHTML="La calculadora";
535.   opcion3.innerHTML="El reloj";
536.   opcion4.innerHTML="El papel";
537.   opcionCorrecta=3;
538. }else if(indice==45) {
539.   texto.innerHTML="De celda en celda voy pero presa no estoy";
540.   opcion1.innerHTML="El policía";
541.   opcion2.innerHTML="La abeja";
542.   opcion3.innerHTML="El preso";
543.   opcion4.innerHTML="La hormiga";
544.   opcionCorrecta=2;
545. }else if(indice==46) {
546.   texto.innerHTML="En el campo me crié, atada con verdes lazos, y aquel que llora por mí me está partiendo en
pedazos";
547.   opcion1.innerHTML="El corazón";
548.   opcion2.innerHTML="El vestido";
549.   opcion3.innerHTML="El destino";
550.   opcion4.innerHTML="La cebolla";
551.   opcionCorrecta=4;
552. }else if(indice==47) {
553.   texto.innerHTML="Tengo cadenas sin ser preso, si me empujas voy y vengo. En los jardines y parques a much
os niños entretengo";
554.   opcion1.innerHTML="El columpio";
555.   opcion2.innerHTML="El perro";
556.   opcion3.innerHTML="El árbol";
557.   opcion4.innerHTML="El padre";
558.   opcionCorrecta=1;
559. }else if(indice==48) {
560.   texto.innerHTML="Después de haberme molido, agua hirviendo echan en mí. La gente me bebe mucho cuand
o no quiere dormir";
561.   opcion1.innerHTML="El azúcar";
562.   opcion2.innerHTML="La pastilla";
563.   opcion3.innerHTML="La harina";
564.   opcion4.innerHTML="El café";
565.   opcionCorrecta=4;
566. }else if(indice==49) {
567.   texto.innerHTML="Vuelo sin alas, silbo sin boca. No se le ve ni tampoco se toca";
568.   opcion1.innerHTML="La flauta";
569.   opcion2.innerHTML="El viento";
570.   opcion3.innerHTML="El silencio";
571.   opcion4.innerHTML="El cielo";
572.   opcionCorrecta=2;
573. }else if(indice==50) {
574.   texto.innerHTML="Tengo hojas sin ser árbol, te hablo sin tener voz, si me abres no me quejo";
575.   opcion1.innerHTML="La planta";
576.   opcion2.innerHTML="El viento";
577.   opcion3.innerHTML="El libro";
578.   opcion4.innerHTML="La puerta";
579.   opcionCorrecta=3;
580. }
581.
582.   cargaOpcionesDialogo(); // Carga las opciones a decir al dialogo
583.
584.   pepperAnimatedSay(texto.innerHTML);
585.
586. }
587.
588. function muestraMensaje(mensaje) {
589.   var x = document.getElementById("snackbar")
590.   x.className = "show";

```

```

591.   setTimeout(function(){ xx.className = x.className.replace("show", ""); }, 3000);
592.
593.   x.innerHTML=mensaje;
594. }
595.
596. function seleccionaOpcion(opcion)
597. {
598.   if(opcion==opcionCorrecta)
599.   {
600.     aciertos+=1;
601.
602.     contador+=1;
603.
604.     // Calcula puntuacion
605.     puntuacion=Math.round(aciertos/(aciertos+fallos)*100);
606.
607.     if(contador<10)
608.     {
609.       // Dice una frase aleatoriamente
610.       var rand=Math.floor((Math.random() * 4) + 1);
611.       if(rand==1) {
612.         pepperSay("¡muy bien! \pau=500\");
613.       }else if(rand==2) {
614.         pepperSay("¡correcto! \pau=500\");
615.       }else if(rand==3) {
616.         pepperSay("¡acertaste! \pau=500\");
617.       }else if(rand==4) {
618.         pepperSay("¡eso es! \pau=500\");
619.       }
620.
621.       muestraMensaje("Has acertado, ¡muy bien!");
622.
623.       //cargaAdvinanza(ranNums[contador]); //muestra la siguiente adivinanza
624.       setTimeout(cargaAdvinanza, 1500, ranNums[contador]); //muestra la siguiente adivinanza con retardo
625.
626.     }else{
627.       finJuego();
628.     }
629.
630.   }else
631.   {
632.     fallos+=1;
633.
634.     // Calcula puntuacion
635.     puntuacion=Math.round(aciertos/(aciertos+fallos)*100);
636.
637.     muestraMensaje("Fallaste, prueba otra vez");
638.
639.     // Dice una frase aleatoriamente
640.     var rand=Math.floor((Math.random() * 3) + 1);
641.     if(rand==1) {
642.       pepperSay("fallaste, prueba otra vez");
643.     }else if(rand==2) {
644.       pepperSay("no es correcto, prueba otra vez");
645.     }else if(rand==3) {
646.       pepperSay("¡inténtalo de nuevo");
647.     }
648.   }
649. }
650.
651. // Get the modal
652. var modal = document.getElementById('myModal');
653. var tituloPanel = document.getElementById('tituloPanel');
654. var textoPanel = document.getElementById('textoPanel');
655.
656. // Cancelar
657. var botonCancelar = document.getElementById('botonCancelar');
658. botonCancelar.onclick = function() {

```

```

659. //modal.style.display = "none";
660.
661. pepperDialogGotoTag('volverAinicio', 'dialogoGeneral');
662. }
663.
664. // Aceptar
665. var botonAceptar = document.getElementById('botonAceptar');
666. botonAceptar.onclick = function() {
667.
668.     modal.style.display = "none";
669.
670.     iniciaJuego();
671. }
672.
673. function cargaJuego()
674. {
675.     tituloPanel.innerHTML="JUEGO ADIVINANZAS";
676.     textoPanel.innerHTML="Selecciona la opción correcta en las siguientes 10 adivinanzas";
677.     botonCancelar.innerHTML="SALIR";
678.     botonAceptar.innerHTML="EMPEZAR";
679.
680.     modal.style.display = "block";
681.
682.     botonAceptar.onclick = function() {
683.
684.         modal.style.display = "none";
685.
686.         iniciaJuego();
687.     }
688.
689.     pepperSayInicio(textoPanel.innerHTML);
690. }
691.
692. function finJuego()
693. {
694.     tituloPanel.innerHTML="FIN DEL JUEGO";
695.     textoPanel.innerHTML="¡Enhorabuena, has terminado! Tu puntuación es de "+String(puntuacion)
696.     botonCancelar.innerHTML="SALIR";
697.     botonAceptar.innerHTML="JUGAR OTRA VEZ";
698.
699.     pepperSendLog("Puntuacion Adivinanzas", String(puntuacion), "-");
700.
701.     modal.style.display = "block";
702.
703.     botonAceptar.onclick = function() {
704.
705.         cargaJuego();
706.     }
707.
708.     pepperSay(textoPanel.innerHTML);
709. }
710.
711. function cargaOpcionesDialogo() { // Carga las opciones a elegir al dialogo
712.     var conceptos_a_cargar=[opcion1.innerHTML, opcion2.innerHTML, opcion3.innerHTML, opcion4.innerHTML]
713.     ;
714.     session.service('ALDialog').then(function (dialog) {
715.         dialog.setConcept("opciones_adivinanzas", "spe", conceptos_a_cargar);
716.     }, function (error) {
717.         console.log(error);
718.     })
719. }
720. QiSession(function (session) {
721.     session.service("ALMemory").then(function (ALMemory) {
722.         ALMemory.subscriber("opcionAdivinanzas").then(function (subscriber) {
723.             // subscriber.signal is a signal associated to "opcionAdivinanzas"
724.             subscriber.signal.connect(function (palabraOpcion) { // Recibe la opcion elegida por voz
725.                 if(opcion1.innerHTML.localeCompare(palabraOpcion)==0)

```

```

726.     seleccionaOpcion(1);
727.     else if(opcion2.innerHTML.localeCompare(palabraOpcion)==0)
728.     seleccionaOpcion(2);
729.     else if(opcion3.innerHTML.localeCompare(palabraOpcion)==0)
730.     seleccionaOpcion(3);
731.     else if(opcion4.innerHTML.localeCompare(palabraOpcion)==0)
732.     seleccionaOpcion(4);
733.     });
734. });
735. });
736. }, function () {
737.   console.log("disconnected");
738. });
739.
740. </script>
741.
742. </body>
743. </html>

```

Ejercicios

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4.   <title>Menu Tablet</title>
5.   <meta charset="UTF-8">
6. </head>
7. <style>
8.   body {background-color: LightGray;}
9.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
10.  p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
11.
12.  .button {
13.    padding: 15px 25px;
14.    font-size:5vw;
15.    width:47vw;
16.    height:27vh;
17.    text-align: center;
18.    cursor: pointer;
19.    outline: none;
20.    color: #fff;
21.    background-color: DodgerBlue;
22.    border: none;
23.    border-radius: 15px;
24.    box-shadow: 0 9px #999;
25.    margin-bottom: 3vh;
26.    margin-left: 0.5vw;
27.    margin-right: 0.5vw;
28.    float:left;
29.  }
30.
31.  .button:hover {background-color: DodgerBlue}
32.
33.  .button:active {
34.    background-color: DodgerBlue;
35.    box-shadow: 0 5px #666;
36.    transform: translateY(4px);
37.  }
38. </style>
39. <body>
40.
41. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MENU EJERCICIOS</h1>
42.
43. <!-- <button class="button" onclick="pepperSwitchFocus('programaejercicios-486826/ejerciciosCabeza')>CUELLO</button>
44. <button class="button" onclick="pepperSwitchFocus('programaejercicios-486826/ejerciciosManos')>MANOS</button>

```

```

45. <button class="button" onclick="pepperSwitchFocus('programaejercicios-
466826/ejerciciosHombro')">HOMBROS</button> -->
46. <button class="button" onclick="pepperDialogGotoTag('ejerciciosCuello', 'dialogoGeneral')">CUELLO</button>
47. <button class="button" onclick="pepperDialogGotoTag('ejerciciosManos', 'dialogoGeneral')">MANOS</button>
48. <button class="button" onclick="pepperDialogGotoTag('ejerciciosHombro', 'dialogoGeneral')">HOMBROS</button>
49. <button class="button" onclick="pepperDialogGotoTag('ejerciciosVarios', 'dialogoGeneral')">VARIOS</button>
50.
51. <button class="button" style="background-color: SlateBlue; height: 15vh;" onclick="pepperDialogGotoTag('volverInicio', 'dialogoGeneral')">VOLVER</button>
52. <button class="button" style="height: 15vh;" onclick="pepperDialogGotoTag('cambiarRepeticiones', 'dialogoGeneral')">REPETICIONES</button>
53.
54. <script src="/libs/qimessaging/2/qimessaging.js"></script>
55. <script src="/libs/qi/2/qi.js"></script>
56. <script type="text/javascript" src="js/scriptPepper.js"></script>
57.
58. </body>
59. </html>

```

Repeticiones ejercicios

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <title>Menu Tablet</title>
5. <meta charset="UTF-8">
6. </head>
7. <style>
8. body { background-color: LightGray; }
9. h1 { font-size: 5vw; background-color: SlateBlue; color: White; text-align: center; }
10. p { font-size: 3vw; background-color: SlateBlue; color: White; text-align: center; }
11.
12. .button {
13. padding: 15px 25px;
14. font-size: 5vw;
15. width: 23vw;
16. height: 21vh;
17. text-align: center;
18. cursor: pointer;
19. outline: none;
20. color: #fff;
21. background-color: DodgerBlue;
22. border: none;
23. border-radius: 15px;
24. box-shadow: 0 9px #999;
25. margin-bottom: 3vh;
26. margin-left: 0.5vw;
27. margin-right: 0.5vw;
28. float: left;
29. }
30.
31. .button:hover { background-color: DodgerBlue }
32.
33. .button:active {
34. background-color: DodgerBlue;
35. box-shadow: 0 5px #666;
36. transform: translateY(4px);
37. }
38. </style>
39. <body>
40.
41. <h1 style="font-size: 5vw; background-color: SlateBlue; color: White; text-align: center;">MENU EJERCICIOS</h1>
42.

```

```

43. <p style="font-size:3vw;background-color:SlateBlue;color:White;text-align:center;">Establezca el número de repeticiones a realizar en los ejercicios</p>
44.
45. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 2);pepperSay('2 repeticiones');">2</button>
46. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 3);pepperSay('3 repeticiones');">3</button>
47. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 4);pepperSay('4 repeticiones');">4</button>
48. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 5);pepperSay('5 repeticiones');">5</button>
49. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 8);pepperSay('8 repeticiones');">8</button>
50. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 10);pepperSay('10 repeticiones');">10</button>
51. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 15);pepperSay('15 repeticiones');">15</button>
52. <button class="button" onclick="pepperInsertIntData('numeroRepeticiones', 20);pepperSay('20 repeticiones');">20</button>
53.
54. <button class="button" style="background-color: SlateBlue;height:15vh;width:47vw" onclick="pepperDialogGotoTag('preguntaEjercicio', 'dialogoGeneral')>VOLVER</button>
55.
56. <script src="/libs/qimessaging/2/qimessaging.js"></script>
57. <script src="/libs/qi/2/qi.js"></script>
58. <script type="text/javascript" src="js/scriptPepper.js"></script>
59.
60. </body>
61. </html>

```

Música

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <title>Menu Tablet</title>
5. <meta charset="UTF-8">
6. </head>
7. <style>
8. body {background-color: LightGray;}
9. h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
10. p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
11.
12. .button {
13. padding: 15px 25px;
14. font-size:5vw;
15. width:47vw;
16. height:27vh;
17. text-align: center;
18. cursor: pointer;
19. outline: none;
20. color: #fff;
21. background-color: DodgerBlue;
22. border: none;
23. border-radius: 15px;
24. box-shadow: 0 9px #999;
25. margin-bottom: 3vh;
26. margin-left: 0.5vw;
27. margin-right: 0.5vw;
28. float:left;
29. }
30.
31. .button:hover {background-color: DodgerBlue}
32.
33. .button:active {
34. background-color: DodgerBlue;
35. box-shadow: 0 5px #666;
36. transform: translateY(4px);

```

```

37.     }
38. </style>
39. <body>
40.
41. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MENU MUSICA</h1>
42.
43. <!-- <button class="button" onclick="location.href='musica_clasica.html';">CLASICA</button>
44. <button class="button" onclick="location.href='musica_pop.html';">POP</button>
45. <button class="button" onclick="location.href='musica_rock.html';">ROCK</button>
46. <button class="button" onclick="location.href='musica_hard.html';">HEAVY</button> -->
47.
48. <button class="button" onclick="pepperDialogGotoTag('musicaClasica', 'dialogoGeneral')">CLASICA</button>
49. <button class="button" onclick="pepperDialogGotoTag('musicaPopular', 'dialogoGeneral')">POPULAR</button>
50. <button class="button" onclick="pepperDialogGotoTag('musicaModerna', 'dialogoGeneral')">MODERNA</button>
51. <button class="button" onclick="pepperDialogGotoTag('musicaCopla', 'dialogoGeneral')">COPLA</button>
52.
53. <button class="button" style="background-color: SlateBlue;height:1.5vh;" onclick="pepperDialogGotoTag('volverAinicio', 'dialogoGeneral')">VOLVER</button>
54.
55. <script src="/libs/qimessaging/2/qimessaging.js"></script>
56. <script src="/libs/qi/2/qi.js"></script>
57. <script type="text/javascript" src="js/scriptPepper.js"></script>
58.
59. </body>
60. </html>

```

Música clásica

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10.   body {background-color: LightGray;}
11.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12.   p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14.   .button {
15.     padding: 15px 25px;
16.     font-size:5vw;
17.     width:47vw;
18.     height:27vh;
19.     text-align: center;
20.     cursor: pointer;
21.     outline: none;
22.     color: #fff;
23.     background-color: DodgerBlue;
24.     border: none;
25.     border-radius: 15px;
26.     box-shadow: 0 9px #999;
27.     margin-bottom: 3vh;
28.     margin-left: 0.5vw;
29.     margin-right: 0.5vw;
30.     float:left;
31.   }
32.
33.   .button:hover {background-color: DodgerBlue}
34.
35.   .button:active {
36.     background-color: DodgerBlue;
37.     box-shadow: 0 5px #666;
38.     transform: translateY(4px);

```

```

39.     }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MUSICA CLASICA</h1>
44.
45. <p></p>
46.
47. <div align="center">
48. <iframe width="560" height="315" id="video" src="https://www.youtube.com/embed/?listType=playlist&list=PL
XuCpbyByLQaOaD060fsuKKUj6_SxnZFS&enablejsapi=1&html5=1&autoplay=1&controls=0&loop=1&rel=0" fr
ameborder="0"></iframe>
49. </div>
50.
51. <!-- <button type="button" id="play-button">Play</button>
52. <button type="button" id="pause-button">Pause</button>
53.
54. <button type="button" id="next-button">Next</button>
55. <button type="button" id="previous-button">Previous</button> -->
56.
57. <p></p>
58.
59. <button class="button" id="next-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
60. <div class="fa fa-step-forward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
61. </button>
62.
63. <button class="button" id="previous-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
64. <div class="fa fa-step-backward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
65. </button>
66.
67. <button class="button" id="pause-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
68. <div class="fa fa-pause" style="font-size:5vw;color:white;" aria-hidden="true"></div>
69. </button>
70.
71. <button class="button" id="play-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
72. <div class="fa fa-play" style="font-size:5vw;color:white;" aria-hidden="true"></div>
73. </button>
74.
75. <p></p>
76.
77. <button class="button" style="background-
color: SlateBlue;height:1.5vh;" onclick="pepperDialogGotoTag('volverInicio', 'dialogoGeneral')">VOLVER</butt
on>
78.
79. <script src="/libs/qimessaging/2/qimessaging.js"></script>
80. <script src="/libs/qi/2/qi.js"></script>
81. <script type="text/javascript" src="js/scriptPepper.js"></script>
82.
83. <script>
84.
85. // global variable for the player
86. var player;
87.
88. // this function gets called when API is ready to use
89. function onYouTubePlayerAPIReady() {
90. // create the global player from the specific iframe (#video)
91. player = new YT.Player('video', {
92. events: {
93. // call this function when player is ready to use
94. 'onReady': onPlayerReady
95. }
96. });
97. }
98.
99. function onPlayerReady(event) {
100.
101. // bind events
102. var playButton = document.getElementById("play-button");

```

```

103. playButton.addEventListener("click", function() {
104.   player.playVideo();
105. });
106.
107. var pauseButton = document.getElementById("pause-button");
108. pauseButton.addEventListener("click", function() {
109.   player.pauseVideo();
110. });
111.
112. var nextButton = document.getElementById("next-button");
113. nextButton.addEventListener("click", function() {
114.   player.nextVideo();
115. });
116.
117. var previousButton = document.getElementById("previous-button");
118. previousButton.addEventListener("click", function() {
119.   player.previousVideo();
120. });
121.
122. // play video on start
123. player.playVideo();
124.
125. }
126.
127. // Inject YouTube API script
128. var tag = document.createElement('script');
129. tag.src = "//www.youtube.com/player_api";
130. var firstScriptTag = document.getElementsByTagName('script')[0];
131. firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);
132.
133. </script>
134.
135. </body>
136. </html>

```

Música copla

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10.   body {background-color: LightGray;}
11.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12.   p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14.   .button {
15.     padding: 15px 25px;
16.     font-size:5vw;
17.     width:47vw;
18.     height:27vh;
19.     text-align: center;
20.     cursor: pointer;
21.     outline: none;
22.     color: #fff;
23.     background-color: DodgerBlue;
24.     border: none;
25.     border-radius: 15px;
26.     box-shadow: 0 9px #999;
27.     margin-bottom: 3vh;
28.     margin-left: 0.5vw;
29.     margin-right: 0.5vw;
30.     float:left;
31.   }
32.

```

```

33. .button:hover {background-color: DodgerBlue}
34.
35. .button:active {
36.     background-color: DodgerBlue;
37.     box-shadow: 0 5px #666;
38.     transform: translateY(4px);
39. }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MUSICA COPLA</h1>
44.
45. <p></p>
46.
47. <div align="center">
48. <iframe width="560" height="315" id="video" src="https://www.youtube.com/embed/?listType=playlist&list=PL
XuCpbyByLQbwzQyUMCubeDu0jUmEnqPu&enablejsapi=1&html5=1&autoplay=1&controls=0&loop=1&rel=0"
frameborder="0"></iframe>
49. </div>
50.
51. <!-- <button type="button" id="play-button">Play</button>
52. <button type="button" id="pause-button">Pause</button>
53.
54. <button type="button" id="next-button">Next</button>
55. <button type="button" id="previous-button">Previous</button> -->
56.
57. <p></p>
58.
59. <button class="button" id="next-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
60. <div class="fa fa-step-forward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
61. </button>
62.
63. <button class="button" id="previous-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
64. <div class="fa fa-step-backward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
65. </button>
66.
67. <button class="button" id="pause-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
68. <div class="fa fa-pause" style="font-size:5vw;color:white;" aria-hidden="true"></div>
69. </button>
70.
71. <button class="button" id="play-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
72. <div class="fa fa-play" style="font-size:5vw;color:white;" aria-hidden="true"></div>
73. </button>
74.
75. <p></p>
76.
77. <button class="button" style="background-
color: SlateBlue;height:1.5vh;" onclick="pepperDialogGotoTag('volverAinicio', 'dialogoGeneral')">VOLVER</butt
on>
78.
79. <script src="/libs/qimessaging/2/qimessaging.js"></script>
80. <script src="/libs/qi/2/qi.js"></script>
81. <script type="text/javascript" src="js/scriptPepper.js"></script>
82.
83. <script>
84.
85. // global variable for the player
86. var player;
87.
88. // this function gets called when API is ready to use
89. function onYouTubePlayerAPIReady() {
90.     // create the global player from the specific iframe (#video)
91.     player = new YT.Player('video', {
92.         events: {
93.             // call this function when player is ready to use
94.             'onReady': onPlayerReady
95.         }
96.     });

```

```

97. }
98.
99. function onPlayerReady(event) {
100.
101. // bind events
102. var playButton = document.getElementById("play-button");
103. playButton.addEventListener("click", function() {
104.   player.playVideo();
105. });
106.
107. var pauseButton = document.getElementById("pause-button");
108. pauseButton.addEventListener("click", function() {
109.   player.pauseVideo();
110. });
111.
112. var nextButton = document.getElementById("next-button");
113. nextButton.addEventListener("click", function() {
114.   player.nextVideo();
115. });
116.
117. var previousButton = document.getElementById("previous-button");
118. previousButton.addEventListener("click", function() {
119.   player.previousVideo();
120. });
121.
122. // play video on start
123. player.playVideo();
124.
125. }
126.
127. // Inject YouTube API script
128. var tag = document.createElement('script');
129. tag.src = "http://www.youtube.com/player_api";
130. var firstScriptTag = document.getElementsByTagName('script')[0];
131. firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);
132.
133. </script>
134.
135. </body>
136. </html>

```

Música popular

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10.   body {background-color: LightGray;}
11.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12.   p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14.   .button {
15.     padding: 15px 25px;
16.     font-size:5vw;
17.     width:47vw;
18.     height:27vh;
19.     text-align: center;
20.     cursor: pointer;
21.     outline: none;
22.     color: #fff;
23.     background-color: DodgerBlue;
24.     border: none;
25.     border-radius: 15px;
26.     box-shadow: 0 9px #999;

```

```

27.     margin-bottom: 3vh;
28.     margin-left: 0.5vw;
29.     margin-right: 0.5vw;
30.     float:left;
31.   }
32.
33.   .button:hover { background-color: DodgerBlue }
34.
35.   .button:active {
36.     background-color: DodgerBlue;
37.     box-shadow: 0 5px #666;
38.     transform: translateY(4px);
39.   }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MUSICA POPULAR</h1>
44.
45. <p></p>
46.
47. <div align="center">
48. <iframe width="560" height="315" id="video" src="https://www.youtube.com/embed/?listType=playlist&list=PL
XuCpbyByLQZ3FkoiZkeU0g90VcJj-
2RH&enablejsapi=1&html5=1&autoplay=1&controls=0&loop=1&rel=0" frameborder="0"></iframe>
49. </div>
50.
51. <!-- <button type="button" id="play-button">Play</button>
52. <button type="button" id="pause-button">Pause</button>
53.
54. <button type="button" id="next-button">Next</button>
55. <button type="button" id="previous-button">Previous</button> -->
56.
57. <p></p>
58.
59. <button class="button" id="next-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
60. <div class="fa fa-step-forward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
61. </button>
62.
63. <button class="button" id="previous-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
64. <div class="fa fa-step-backward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
65. </button>
66.
67. <button class="button" id="pause-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
68. <div class="fa fa-pause" style="font-size:5vw;color:white;" aria-hidden="true"></div>
69. </button>
70.
71. <button class="button" id="play-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
72. <div class="fa fa-play" style="font-size:5vw;color:white;" aria-hidden="true"></div>
73. </button>
74.
75. <p></p>
76.
77. <button class="button" style="background-
color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAInicio', 'dialogoGeneral')">VOLVER</butt
on>
78.
79. <script src="/libs/qimessaging/2/qimessaging.js"></script>
80. <script src="/libs/qi/2/qi.js"></script>
81. <script type="text/javascript" src="js/scriptPepper.js"></script>
82.
83. <script>
84.
85. // global variable for the player
86. var player;
87.
88. // this function gets called when API is ready to use
89. function onYouTubePlayerAPIReady() {
90. // create the global player from the specific iframe (#video)

```

```

91.  player = new YT.Player('video', {
92.    events: {
93.      // call this function when player is ready to use
94.      'onReady': onPlayerReady
95.    }
96.  });
97. }
98.
99. function onPlayerReady(event) {
100.
101.  // bind events
102.  var playButton = document.getElementById("play-button");
103.  playButton.addEventListener("click", function() {
104.    player.playVideo();
105.  });
106.
107.  var pauseButton = document.getElementById("pause-button");
108.  pauseButton.addEventListener("click", function() {
109.    player.pauseVideo();
110.  });
111.
112.  var nextButton = document.getElementById("next-button");
113.  nextButton.addEventListener("click", function() {
114.    player.nextVideo();
115.  });
116.
117.  var previousButton = document.getElementById("previous-button");
118.  previousButton.addEventListener("click", function() {
119.    player.previousVideo();
120.  });
121.
122.  // play video on start
123.  player.playVideo();
124.
125. }
126.
127. // Inject YouTube API script
128. var tag = document.createElement('script');
129. tag.src = "http://www.youtube.com/player_api";
130. var firstScriptTag = document.getElementsByTagName('script')[0];
131. firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);
132.
133. </script>
134.
135. </body>
136. </html>

```

Música moderna

```

1.  <!DOCTYPE html>
2.  <html lang="es-ES">
3.  <head>
4.  <!-- Add icon library -->
5.  <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6.  <title>Menu Tablet</title>
7.  <meta charset="UTF-8">
8.  </head>
9.  <style>
10.  body { background-color: LightGray; }
11.  h1 { font-size:5vw; background-color:SlateBlue; color:White; text-align:center; }
12.  p { font-size:3vw; background-color:SlateBlue; color:White; text-align:center; }
13.
14.  .button {
15.    padding: 15px 25px;
16.    font-size:5vw;
17.    width:47vw;
18.    height:27vh;
19.    text-align: center;
20.    cursor: pointer;

```

```

21.     outline: none;
22.     color: #fff;
23.     background-color: DodgerBlue;
24.     border: none;
25.     border-radius: 15px;
26.     box-shadow: 0 9px #999;
27.     margin-bottom: 3vh;
28.     margin-left: 0.5vw;
29.     margin-right: 0.5vw;
30.     float:left;
31. }
32.
33. .button:hover {background-color: DodgerBlue}
34.
35. .button:active {
36.     background-color: DodgerBlue;
37.     box-shadow: 0 5px #666;
38.     transform: translateY(4px);
39. }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MUSICA MODERNA</h1>
44.
45. <p></p>
46.
47. <div align="center">
48. <iframe width="560" height="315" id="video" src="https://www.youtube.com/embed/?listType=playlist&list=PLXuCpbyByLQas_Q_zstOkgbQfunP8XUwu&enablejsapi=1&html5=1&autoplay=1&controls=0&loop=1&rel=0" frameborder="0"></iframe>
49. </div>
50.
51. <!-- <button type="button" id="play-button">Play</button>
52. <button type="button" id="pause-button">Pause</button>
53.
54. <button type="button" id="next-button">Next</button>
55. <button type="button" id="previous-button">Previous</button> -->
56.
57. <p></p>
58.
59. <button class="button" id="next-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
60. <div class="fa fa-step-forward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
61. </button>
62.
63. <button class="button" id="previous-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
64. <div class="fa fa-step-backward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
65. </button>
66.
67. <button class="button" id="pause-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
68. <div class="fa fa-pause" style="font-size:5vw;color:white;" aria-hidden="true"></div>
69. </button>
70.
71. <button class="button" id="play-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
72. <div class="fa fa-play" style="font-size:5vw;color:white;" aria-hidden="true"></div>
73. </button>
74.
75. <p></p>
76.
77. <button class="button" style="background-color: SlateBlue;height:1.5vh;" onclick="pepperDialogGotoTag('volverAinicio', 'dialogoGeneral')">VOLVER</button>
78.
79. <script src="/libs/qimessaging/2/qimessaging.js"></script>
80. <script src="/libs/qi/2/qi.js"></script>
81. <script type="text/javascript" src="js/scriptPepper.js"></script>
82.
83. <script>

```

```

84.
85. // global variable for the player
86. var player;
87.
88. // this function gets called when API is ready to use
89. function onYouTubePlayerAPIReady() {
90. // create the global player from the specific iframe (#video)
91. player = new YT.Player('video', {
92.   events: {
93.     // call this function when player is ready to use
94.     'onReady': onPlayerReady
95.   }
96. });
97. }
98.
99. function onPlayerReady(event) {
100.
101. // bind events
102. var playButton = document.getElementById("play-button");
103. playButton.addEventListener("click", function() {
104.   player.playVideo();
105. });
106.
107. var pauseButton = document.getElementById("pause-button");
108. pauseButton.addEventListener("click", function() {
109.   player.pauseVideo();
110. });
111.
112. var nextButton = document.getElementById("next-button");
113. nextButton.addEventListener("click", function() {
114.   player.nextVideo();
115. });
116.
117. var previousButton = document.getElementById("previous-button");
118. previousButton.addEventListener("click", function() {
119.   player.previousVideo();
120. });
121.
122. // play video on start
123. player.playVideo();
124.
125. }
126.
127. // Inject YouTube API script
128. var tag = document.createElement('script');
129. tag.src = "http://www.youtube.com/player_api";
130. var firstScriptTag = document.getElementsByTagName('script')[0];
131. firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);
132.
133. </script>
134.
135. </body>
136. </html>

```

Música pop

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10.   body {background-color: LightGray;}
11.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12.   p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.

```

```

14. .button {
15.     padding: 15px 25px;
16.     font-size:5vw;
17.     width:47vw;
18.     height:27vh;
19.     text-align: center;
20.     cursor: pointer;
21.     outline: none;
22.     color: #fff;
23.     background-color: DodgerBlue;
24.     border: none;
25.     border-radius: 15px;
26.     box-shadow: 0 9px #999;
27.     margin-bottom: 3vh;
28.     margin-left: 0.5vw;
29.     margin-right: 0.5vw;
30.     float:left;
31. }
32.
33. .button:hover {background-color: DodgerBlue}
34.
35. .button:active {
36.     background-color: DodgerBlue;
37.     box-shadow: 0 5px #666;
38.     transform: translateY(4px);
39. }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MUSICA POP</h1>
44.
45. <p></p>
46.
47. <div align="center">
48. <iframe width="560" height="315" id="video" src="https://www.youtube.com/embed/?listType=playlist&list=PL
XuCpbyByLQbNkzSh5Y5MkfZxErCiGu0g&enablejsapi=1&html5=1&autoplay=1&controls=0&loop=1&rel=0" fr
ameborder="0"></iframe>
49. </div>
50.
51. <!-- <button type="button" id="play-button">Play</button>
52. <button type="button" id="pause-button">Pause</button>
53.
54. <button type="button" id="next-button">Next</button>
55. <button type="button" id="previous-button">Previous</button> -->
56.
57. <p></p>
58.
59. <button class="button" id="next-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
60. <div class="fa fa-step-forward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
61. </button>
62.
63. <button class="button" id="previous-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
64. <div class="fa fa-step-backward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
65. </button>
66.
67. <button class="button" id="pause-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
68. <div class="fa fa-pause" style="font-size:5vw;color:white;" aria-hidden="true"></div>
69. </button>
70.
71. <button class="button" id="play-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
72. <div class="fa fa-play" style="font-size:5vw;color:white;" aria-hidden="true"></div>
73. </button>
74.
75. <p></p>
76.
77. <button class="button" style="background-
color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAinicio', 'dialogoGeneral')">VOLVER</butt
on>

```

```

78.
79. <script src="/libs/qimessaging/2/qimessaging.js"></script>
80. <script src="/libs/qi/2/qi.js"></script>
81. <script type="text/javascript" src="js/scriptPepper.js"></script>
82.
83. <script>
84.
85. // global variable for the player
86. var player;
87.
88. // this function gets called when API is ready to use
89. function onYouTubePlayerAPIReady() {
90. // create the global player from the specific iframe (#video)
91. player = new YT.Player('video', {
92.   events: {
93.     // call this function when player is ready to use
94.     'onReady': onPlayerReady
95.   }
96. });
97. }
98.
99. function onPlayerReady(event) {
100.
101. // bind events
102. var playButton = document.getElementById("play-button");
103. playButton.addEventListener("click", function() {
104.   player.playVideo();
105. });
106.
107. var pauseButton = document.getElementById("pause-button");
108. pauseButton.addEventListener("click", function() {
109.   player.pauseVideo();
110. });
111.
112. var nextButton = document.getElementById("next-button");
113. nextButton.addEventListener("click", function() {
114.   player.nextVideo();
115. });
116.
117. var previousButton = document.getElementById("previous-button");
118. previousButton.addEventListener("click", function() {
119.   player.previousVideo();
120. });
121.
122. // play video on start
123. player.playVideo();
124.
125. }
126.
127. // Inject YouTube API script
128. var tag = document.createElement('script');
129. tag.src = "http://www.youtube.com/player_api";
130. var firstScriptTag = document.getElementsByTagName('script')[0];
131. firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);
132.
133. </script>
134.
135. </body>
136. </html>

```

Música rock

```

1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">

```

```

8. </head>
9. <style>
10.   body {background-color: LightGray;}
11.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12.   p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14.   .button {
15.     padding: 15px 25px;
16.     font-size:5vw;
17.     width:47vw;
18.     height:27vh;
19.     text-align: center;
20.     cursor: pointer;
21.     outline: none;
22.     color: #fff;
23.     background-color: DodgerBlue;
24.     border: none;
25.     border-radius: 15px;
26.     box-shadow: 0 9px #999;
27.     margin-bottom: 3vh;
28.     margin-left: 0.5vw;
29.     margin-right: 0.5vw;
30.     float:left;
31.   }
32.
33.   .button:hover {background-color: DodgerBlue}
34.
35.   .button:active {
36.     background-color: DodgerBlue;
37.     box-shadow: 0 5px #666;
38.     transform: translateY(4px);
39.   }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MUSICA ROCK</h1>
44.
45. <p></p>
46.
47. <div align="center">
48. <iframe width="560" height="315" id="video" src="https://www.youtube.com/embed/?listType=playlist&list=PL
49. XuCpbyByLQaPIR8iK_Z5J1OsrjhHusim&enablejsapi=1&html5=1&autoplay=1&controls=0&loop=1&rel=0" fra
50. meborder="0"></iframe>
51. </div>
52. <!-- <button type="button" id="play-button">Play</button>
53. <button type="button" id="pause-button">Pause</button>
54. <button type="button" id="next-button">Next</button>
55. <button type="button" id="previous-button">Previous</button> -->
56.
57. <p></p>
58.
59. <button class="button" id="next-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
60. <div class="fa fa-step-forward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
61. </button>
62.
63. <button class="button" id="previous-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
64. <div class="fa fa-step-backward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
65. </button>
66.
67. <button class="button" id="pause-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
68. <div class="fa fa-pause" style="font-size:5vw;color:white;" aria-hidden="true"></div>
69. </button>
70.
71. <button class="button" id="play-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
72. <div class="fa fa-play" style="font-size:5vw;color:white;" aria-hidden="true"></div>
73. </button>

```

```

74.
75. <p></p>
76.
77. <button class="button" style="background-
color: SlateBlue;height:15vh;" onclick="pepperDialogGotoTag('volverAinicio', 'dialogoGeneral')">VOLVER</butt
on>
78.
79. <script src="/libs/qimessaging/2/qimessaging.js"></script>
80. <script src="/libs/qi/2/qi.js"></script>
81. <script type="text/javascript" src="js/scriptPepper.js"></script>
82.
83. <script>
84.
85. // global variable for the player
86. var player;
87.
88. // this function gets called when API is ready to use
89. function onYouTubePlayerAPIReady() {
90. // create the global player from the specific iframe (#video)
91. player = new YT.Player('video', {
92. events: {
93. // call this function when player is ready to use
94. 'onReady': onPlayerReady
95. }
96. });
97. }
98.
99. function onPlayerReady(event) {
100.
101. // bind events
102. var playButton = document.getElementById("play-button");
103. playButton.addEventListener("click", function() {
104. player.playVideo();
105. });
106.
107. var pauseButton = document.getElementById("pause-button");
108. pauseButton.addEventListener("click", function() {
109. player.pauseVideo();
110. });
111.
112. var nextButton = document.getElementById("next-button");
113. nextButton.addEventListener("click", function() {
114. player.nextVideo();
115. });
116.
117. var previousButton = document.getElementById("previous-button");
118. previousButton.addEventListener("click", function() {
119. player.previousVideo();
120. });
121.
122. // play video on start
123. player.playVideo();
124.
125. }
126.
127. // Inject YouTube API script
128. var tag = document.createElement('script');
129. tag.src = "http://www.youtube.com/player_api";
130. var firstScriptTag = document.getElementsByTagName('script')[0];
131. firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);
132.
133. </script>
134.
135. </body>
136. </html>

```

Música heavy

```
1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <head>
4. <!-- Add icon library -->
5. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
6. <title>Menu Tablet</title>
7. <meta charset="UTF-8">
8. </head>
9. <style>
10. body {background-color: LightGray;}
11. h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
12. p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
13.
14. .button {
15. padding: 15px 25px;
16. font-size:5vw;
17. width:47vw;
18. height:27vh;
19. text-align: center;
20. cursor: pointer;
21. outline: none;
22. color: #fff;
23. background-color: DodgerBlue;
24. border: none;
25. border-radius: 15px;
26. box-shadow: 0 9px #999;
27. margin-bottom: 3vh;
28. margin-left: 0.5vw;
29. margin-right: 0.5vw;
30. float:left;
31. }
32.
33. .button:hover {background-color: DodgerBlue}
34.
35. .button:active {
36. background-color: DodgerBlue;
37. box-shadow: 0 5px #666;
38. transform: translateY(4px);
39. }
40. </style>
41. <body>
42.
43. <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">MUSICA HEAVY</h1>
44.
45. <p></p>
46.
47. <div align="center">
48. <iframe width="560" height="315" id="video" src="https://www.youtube.com/embed/?listType=playlist&list=PL
XuCpbyByLQa4IKtRd538-
HVn68nq3Sr&enablejsapi=1&html5=1&autoplay=1&controls=0&loop=1&rel=0" frameborder="0"></iframe>
49. </div>
50.
51. <!-- <button type="button" id="play-button">Play</button>
52. <button type="button" id="pause-button">Pause</button>
53.
54. <button type="button" id="next-button">Next</button>
55. <button type="button" id="previous-button">Previous</button> -->
56.
57. <p></p>
58.
59. <button class="button" id="next-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
60. <div class="fa fa-step-forward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
61. </button>
62.
63. <button class="button" id="previous-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
64. <div class="fa fa-step-backward" style="font-size:5vw;color:white;" aria-hidden="true"></div>
65. </button>
```

```

66.
67. <button class="button" id="pause-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
68. <div class="fa fa-pause" style="font-size:5vw;color:white;" aria-hidden="true"></div>
69. </button>
70.
71. <button class="button" id="play-button" style="height:8vw;width:9vw;float:right;margin-right: 2.75vw;">
72. <div class="fa fa-play" style="font-size:5vw;color:white;" aria-hidden="true"></div>
73. </button>
74.
75. <p></p>
76.
77. <button class="button" style="background-
    color: SlateBlue;height:1.5vh;" onclick="pepperDialogGotoTag('volverAinicio', 'dialogoGeneral')">VOLVER</butt
    on>
78.
79. <script src="/libs/qimessaging/2/qimessaging.js"></script>
80. <script src="/libs/qi/2/qi.js"></script>
81. <script type="text/javascript" src="js/scriptPepper.js"></script>
82.
83. <script>
84.
85. // global variable for the player
86. var player;
87.
88. // this function gets called when API is ready to use
89. function onYouTubePlayerAPIReady() {
90. // create the global player from the specific iframe (#video)
91. player = new YT.Player('video', {
92. events: {
93. // call this function when player is ready to use
94. 'onReady': onPlayerReady
95. }
96. });
97. }
98.
99. function onPlayerReady(event) {
100.
101. // bind events
102. var playButton = document.getElementById("play-button");
103. playButton.addEventListener("click", function() {
104. player.playVideo();
105. });
106.
107. var pauseButton = document.getElementById("pause-button");
108. pauseButton.addEventListener("click", function() {
109. player.pauseVideo();
110. });
111.
112. var nextButton = document.getElementById("next-button");
113. nextButton.addEventListener("click", function() {
114. player.nextVideo();
115. });
116.
117. var previousButton = document.getElementById("previous-button");
118. previousButton.addEventListener("click", function() {
119. player.previousVideo();
120. });
121.
122. // play video on start
123. player.playVideo();
124.
125. }
126.
127. // Inject YouTube API script
128. var tag = document.createElement('script');
129. tag.src = "//www.youtube.com/player_api";
130. var firstScriptTag = document.getElementsByTagName('script')[0];
131. firstScriptTag.parentNode.insertBefore(tag, firstScriptTag);

```

```
132.
133. </script>
134.
135. </body>
136. </html>
```

Valoración

```
1. <!DOCTYPE html>
2. <html lang="es-ES">
3. <meta name="viewport" content="width=device-width, initial-scale=1.0">
4. <head>
5. <!-- Font Awesome Icon Library -->
6. <link rel="stylesheet" href="font-awesome-4.7.0/css/font-awesome.min.css">
7. <title>Menu Tablet</title>
8. <meta charset="UTF-8">
9. </head>
10. <style>
11.   body {background-color: LightGray;}
12.   h1 {font-size:5vw; background-color:SlateBlue; color:White; text-align:center;}
13.   p {font-size:3vw; background-color:SlateBlue; color:White; text-align:center;}
14.
15.   .button {
16.     padding: 15px 25px;
17.     font-size:5vw;
18.     width:47vw;
19.     height:27vh;
20.     text-align: center;
21.     cursor: pointer;
22.     outline: none;
23.     color: #fff;
24.     background-color: DodgerBlue;
25.     border: none;
26.     border-radius: 15px;
27.     box-shadow: 0 9px #999;
28.     margin-bottom: 3vh;
29.     margin-left: 0.5vw;
30.     margin-right: 0.5vw;
31.     float:left;
32.   }
33.
34.   .button:hover {background-color: DodgerBlue}
35.
36.   .button:active {
37.     background-color: DodgerBlue;
38.     box-shadow: 0 5px #666;
39.     transform: translateY(4px);
40.   }
41.
42.   #snackbar {
43.     visibility: hidden;
44.     min-width: 250px;
45.     margin-left: -125px;
46.     background-color: SlateBlue;
47.     color: #fff;
48.     text-align: center;
49.     border-radius: 2px;
50.     padding: 16px;
51.     position: fixed;
52.     z-index: 1;
53.     left: 25%;
54.     bottom: 30vh;
55.     font-size: 5vw;
56.   }
57.
58.   #snackbar.show {
59.     visibility: visible;
60.     -webkit-animation: fadein 0.5s, fadeout 0.5s 2.5s;
61.     animation: fadein 0.5s, fadeout 0.5s 2.5s;
```

```

62. }
63.
64. @-webkit-keyframes fadein {
65.     from {bottom: 0; opacity: 0;}
66.     to {bottom: 30vh; opacity: 1;}
67. }
68.
69. @keyframes fadein {
70.     from {bottom: 0; opacity: 0;}
71.     to {bottom: 30vh; opacity: 1;}
72. }
73.
74. @-webkit-keyframes fadeout {
75.     from {bottom: 30vh; opacity: 1;}
76.     to {bottom: 0; opacity: 0;}
77. }
78.
79. @keyframes fadeout {
80.     from {bottom: 30vh; opacity: 1;}
81.     to {bottom: 0; opacity: 0;}
82. }
83. </style>
84. <body>
85.
86. <!-- <h1 style="font-size:5vw;background-color:SlateBlue;color:White;text-align:center;">VALORACIÓN</h1> --
87. >
88. <p style="font-size:3vw;background-color:SlateBlue;color:White;text-
89. align:center;">Haz una valoración sobre mi funcionamiento</p>
90.
91. <p style="font-size:5vw;background-color:DodgerBlue;color:White;text-align:left;margin:1vw; padding:0vw;">
92. Reconocimiento de voz
93. <span class="fa fa-star fa-pull-right" style="font-
94. size:5vw;" onclick="setValoration('reconocimiento', 5)" id="reconocimiento_5"></span>
95. <span class="fa fa-star fa-pull-right" style="font-
96. size:5vw;" onclick="setValoration('reconocimiento', 4)" id="reconocimiento_4"></span>
97. <span class="fa fa-star fa-pull-right" style="font-
98. size:5vw;" onclick="setValoration('reconocimiento', 3)" id="reconocimiento_3"></span>
99. <span class="fa fa-star fa-pull-right" style="font-
100. size:5vw;" onclick="setValoration('reconocimiento', 2)" id="reconocimiento_2"></span>
101. <span class="fa fa-star fa-pull-right" style="font-
102. size:5vw;" onclick="setValoration('reconocimiento', 1)" id="reconocimiento_1"></span>
103. </p>
104.
105. <p style="font-size:5vw;background-color:DodgerBlue;color:White;text-align:left;margin:1vw; padding:0vw;">
106. Información
107. <span class="fa fa-star fa-pull-right" style="font-
108. size:5vw;" onclick="setValoration('informacion', 5)" id="informacion_5"></span>
109. <span class="fa fa-star fa-pull-right" style="font-
110. size:5vw;" onclick="setValoration('informacion', 4)" id="informacion_4"></span>
111. <span class="fa fa-star fa-pull-right" style="font-
112. size:5vw;" onclick="setValoration('informacion', 3)" id="informacion_3"></span>
113. <span class="fa fa-star fa-pull-right" style="font-
114. size:5vw;" onclick="setValoration('informacion', 2)" id="informacion_2"></span>
115. <span class="fa fa-star fa-pull-right" style="font-
116. size:5vw;" onclick="setValoration('informacion', 1)" id="informacion_1"></span>
117. </p>
118.
119. <p style="font-size:5vw;background-color:DodgerBlue;color:White;text-align:left;margin:1vw; padding:0vw;">
120. Juegos
121. <span class="fa fa-star fa-pull-right" style="font-
122. size:5vw;" onclick="setValoration('juegos', 5)" id="juegos_5"></span>
123. <span class="fa fa-star fa-pull-right" style="font-
124. size:5vw;" onclick="setValoration('juegos', 4)" id="juegos_4"></span>
125. <span class="fa fa-star fa-pull-right" style="font-
126. size:5vw;" onclick="setValoration('juegos', 3)" id="juegos_3"></span>
127. <span class="fa fa-star fa-pull-right" style="font-
128. size:5vw;" onclick="setValoration('juegos', 2)" id="juegos_2"></span>

```

```

114. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('juegos', 1)" id="juegos_1"></span>
115. </p>
116.
117. <p style="font-size:5vw;background-color:DodgerBlue;color:White;text-align:left;margin:1vw; padding:0vw;">
118. Ejercicios
119. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('ejercicios', 5)" id="ejercicios_5"></span>
120. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('ejercicios', 4)" id="ejercicios_4"></span>
121. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('ejercicios', 3)" id="ejercicios_3"></span>
122. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('ejercicios', 2)" id="ejercicios_2"></span>
123. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('ejercicios', 1)" id="ejercicios_1"></span>
124. </p>
125.
126. <p style="font-size:5vw;background-color:DodgerBlue;color:White;text-align:left;margin:1vw; padding:0vw;">
127. Valoración General
128. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('general', 5)" id="general_5"></span>
129. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('general', 4)" id="general_4"></span>
130. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('general', 3)" id="general_3"></span>
131. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('general', 2)" id="general_2"></span>
132. <span class="fa fa-star fa-pull-right" style="font-size:5vw;" onclick="setValoration('general', 1)" id="general_1"></span>
133. </p>
134.
135. <button class="button" style="background-color: SlateBlue;height:1.5vh;" onclick="pepperDialogGotoTag('volverInicio', 'dialogoGeneral')">VOLVER</button>
136.
137. <button class="button" style="background-color: SlateBlue;height:1.5vh;" onclick="enviarValoracion();">CONFIRMAR</button>
138.
139. <div id="snackbar">Por favor valora todos los campos</div>
140.
141. <script src="/libs/qimessaging/2/qimessaging.js"></script>
142. <script src="/libs/qi/2/qi.js"></script>
143. <script type="text/javascript" src="js/scriptPepper.js"></script>
144.
145. <script>
146.
147. var valoracion_reconocimiento=0;
148. var valoracion_informacion=0;
149. var valoracion_juegos=0;
150. var valoracion_ejercicios=0;
151. var valoracion_general=0;
152.
153. function setValoration(group, value)
154. {
155.   for (i = 1; i <= 5; i++) {
156.     id=group+'_'+i.toString();
157.     document.getElementById(id).style.color="White";
158.   }
159.   for (i = 1; i <= value; i++) {
160.     id=group+'_'+i.toString();
161.     document.getElementById(id).style.color="Yellow";
162.   }
163.
164.   if(group.localeCompare("reconocimiento")==0) {
165.     valoracion_reconocimiento=value;
166.   }
167.

```

```

168.     if(group.localeCompare("informacion")==0) {
169.         valoracion_informacion=value;
170.     }
171.
172.     if(group.localeCompare("juegos")==0) {
173.         valoracion_juegos=value;
174.     }
175.
176.     if(group.localeCompare("ejercicios")==0) {
177.         valoracion_ejercicios=value;
178.     }
179.
180.     if(group.localeCompare("general")==0) {
181.         valoracion_general=value;
182.     }
183.
184.     if(value==5)
185.         pepperSay("¡Estupendo!");
186.     else if(value==4)
187.         pepperSay("Bien!");
188.     else if(value==3)
189.         pepperSay("Vale");
190.     else if(value==2)
191.         pepperSay("Oh, vaya");
192.     else if(value==1)
193.         pepperSay("Oh, lo siento");
194. }
195.
196. function muestraMensaje() {
197.     var x = document.getElementById("snackbar")
198.     x.className = "show";
199.     setTimeout(function(){ xx.className = x.className.replace("show", ""); }, 3000);
200. }
201.
202. function enviarValoracion() {
203.     // Comprueba que se han valorado todos los campos
204.     if(valoracion_reconocimiento==0 || valoracion_informacion==0 || valoracion_juegos==0 || valoracion_ejercicio
s==0 || valoracion_general==0) {
205.         muestraMensaje();
206.         pepperSay("Por favor valora todos los campos");
207.         return;
208.     }
209.
210.     //Envia valoracion
211.     valoraciones=String(valoracion_reconocimiento)+''+String(valoracion_informacion)+''+String(valoracion_jue
gos)+''+String(valoracion_ejercicios);
212.     pepperSendLog("Valoracion Usuario", valoraciones, String(valoracion_general));
213.
214.     // Vuelve al inicio
215.     pepperDialogGotoTag('agradeceValoracion', 'dialogoGeneral');
216. }
217. </script>
218.
219. </body>
220. </html>

```