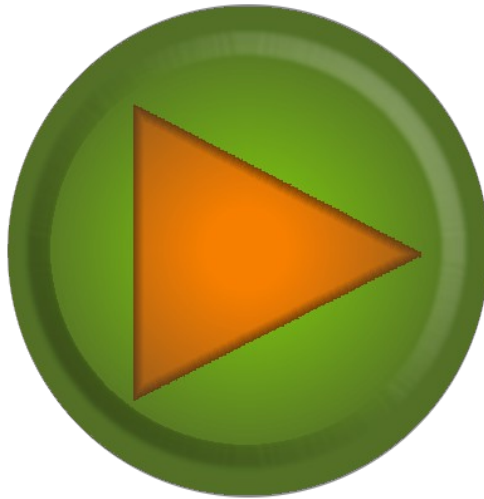


Play US



Arquitectura e Integración de Sistemas Software
Grado de Ingeniería del Software

García Sánchez, Alberto (albgarsanuni@yahoo.es)
Muñoz Moya, David (damu-cazalla@hotmail.es)
Sánchez Crespo, Juan Luis (juanlu-28a@hotmail.com)

Tutor: Ortega Rodríguez, Francisco Javier
Número de grupo: 6

Enlace de la aplicación: play-us.appspot.com
Enlace de proyecto en gitHub: <https://github.com/Juanlu991/Play-US>
Enlace de proyecto en Projetsii:
<https://projetsii.informatica.us.es/projects/h36kfmvc9p3g2yyzyq4>

HISTORIAL DE VERSIONES

Fecha	Versión	Detalles	Participantes
14/03/2014	1.0	- Incluye introducción, prototipos de las interfaces de usuario y diagramas UML de componentes y despliegue.	Muñoz Moya, David García Sánchez, Alberto Sánchez Crespo, Juan Luis
25/04/2014	1.1	-Incluye introducción, prototipos de las interfaces de usuario, diagramas UML de componentes y despliegue, prototipo funcional y documentación API REST	Muñoz Moya, David García Sánchez, Alberto Sánchez Crespo, Juan Luis

AUTOEVALUACIÓN

Alumno	Entregable 1	Entregable 2	Final
Sánchez Crespo, Juan Luis	100%	100%	
Muñoz Moya, David	100%	50%	
García Sánchez, Alberto	100%	60%	

0% = No he podido participar y por tanto debo aceptar tener un cero en el entregable.

50% = He trabajado aproximadamente la mitad que mis compañeros y por tanto merezco la mitad de la nota del trabajo.

100% = He trabajado como el que más y merezco la nota dada al trabajo.

Índice

1	Introducción.....	5
1.1	Aplicaciones integradas.....	5
1.2	Evolución del proyecto.....	6
2	Prototipos de interfaz de usuario.....	6
2.1	Vista Inicio.....	6
2.2	Vista Configuración.....	7
3	Arquitectura.....	7
3.1	Diagrama de componentes.....	7
3.2	Diagrama de despliegue.....	8
4	Manual de usuario.....	9
4.1	Mahup.....	9
4.2	API REST.....	11
4.2.1	/oauth2/token.....	12
4.2.2	/users.....	12
4.2.3	/users/{id}.....	12
4.2.4	/users/{id}/playlists.....	13
4.2.5	/users/{id}/playlists/{playlist_name}.....	13
4.2.6	/songs.....	14
4.2.7	/songs/{server}/{id}.....	15
	Referencias.....	16

1 Introducción

Hay ocasiones en las que la canción que buscamos no se encuentra en nuestro reproductor de música online. Otras veces, tenemos varias canciones que nos gustan, pero hay una que no está en nuestro reproductor preferido, por lo que no podemos hacer listas de reproducciones de todas las canciones que deseamos. Algunas veces deseamos hacer un comentario en alguna red social, sobre la canción que estamos escuchando, y en ese momento no estamos en ella.

Para la solución de esos problemas y más, vamos a realizar una aplicación que englobe los 3 reproductores principales de música online: Grooveshark, SoundCloud y Spotify. También vamos a conectar dos redes sociales: Twitter y Facebook. Podremos realizar listas de reproducción con canciones de diferentes reproductores, hacer comentarios mientras estamos escuchando nuestras canciones sin tener que abrir otra pestaña e incluso mencionar nuestra canción en nuestro comentario con el símbolo |> .

Con todo esto, agrupamos lo mejor de cada reproductor en un mismo sitio web, además con la posibilidad de realizar comentarios.

Como servicios vamos a ofrecer la posibilidad de realizar búsquedas de canciones, devolveremos una lista de canciones y en qué servidor la podemos encontrar. También se podrá realizar la búsqueda de canciones en servidores concretos.

1.1 Aplicaciones integradas

Nombre aplicación	URL documentación API
Facebook	https://developers.facebook.com/docs/graph-api/using-graph-api/
SoundCloud	http://developers.soundcloud.com/docs/api/reference
Twitter	https://dev.twitter.com/docs/api/1.1
Spotify	https://developer.spotify.com/technologies/web-api/
Grooveshark	http://developers.grooveshark.com/

TABLA 1. APLICACIÓN INTEGRADAS

1.2 Evolución del proyecto

2 Prototipos de interfaz de usuario

2.1 Vista Inicio

En esta vista tenemos a nuestro alcance todos los componentes que forman nuestro mashup.

En el panel de la izquierda, podemos encontrar las canciones específicas de cada reproductor de nuestra lista de reproducción actual.

En el panel central, tenemos primero un buscador de canciones, en el que especificamos el servidor en el que queremos buscarlo. A continuación aparecería la búsqueda realizada. Debajo de la búsqueda, hayamos un recuadro, en el cual podemos introducir nuestro comentario y elegir la red social en el que vamos a publicarlo.

En el ultimo panel, encontramos la lista de reproducción actual, además tenemos la opción de guardar nuestra lista.

En la parte inferior de nuestra pantalla, siempre encontraremos una herramienta de reproducción con la cual podemos controlar la canción actual.

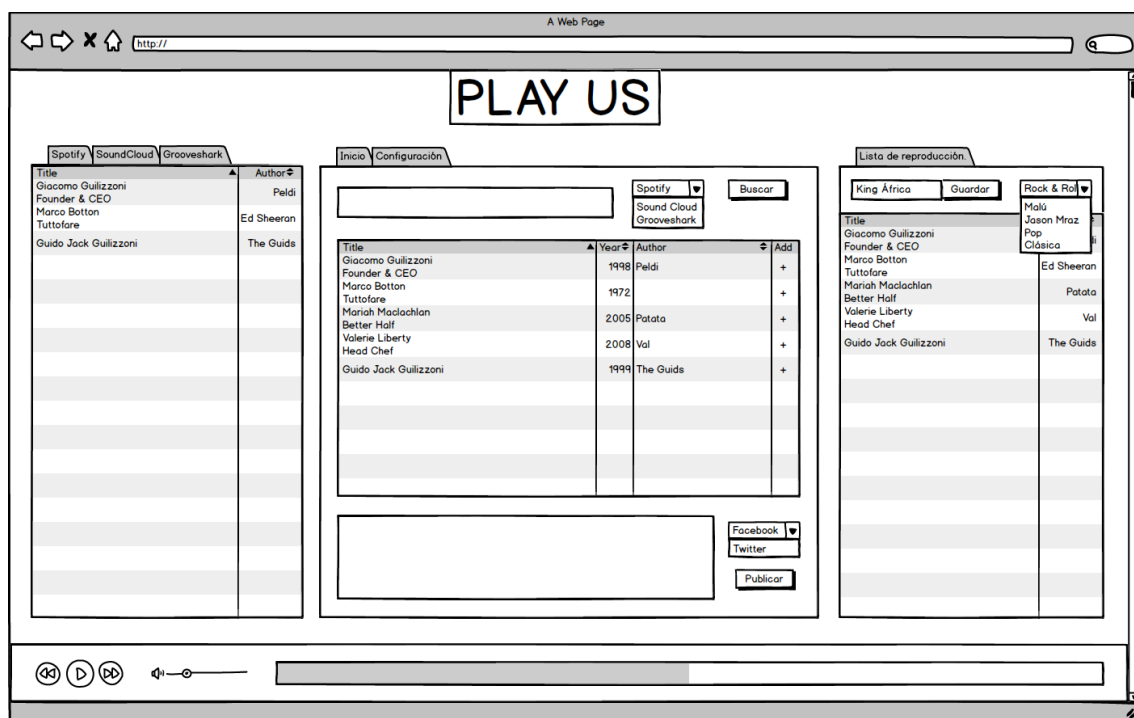


FIGURA 1. PROTOTIPO DE INTERFAZ DE USUARIO DE LA VISTA INICIO

2.2 Vista Configuración

En esta vista, el único cambio que encontramos es el del panel central, que ahora muestra una serie de opciones para autenticarnos en las distintas web.

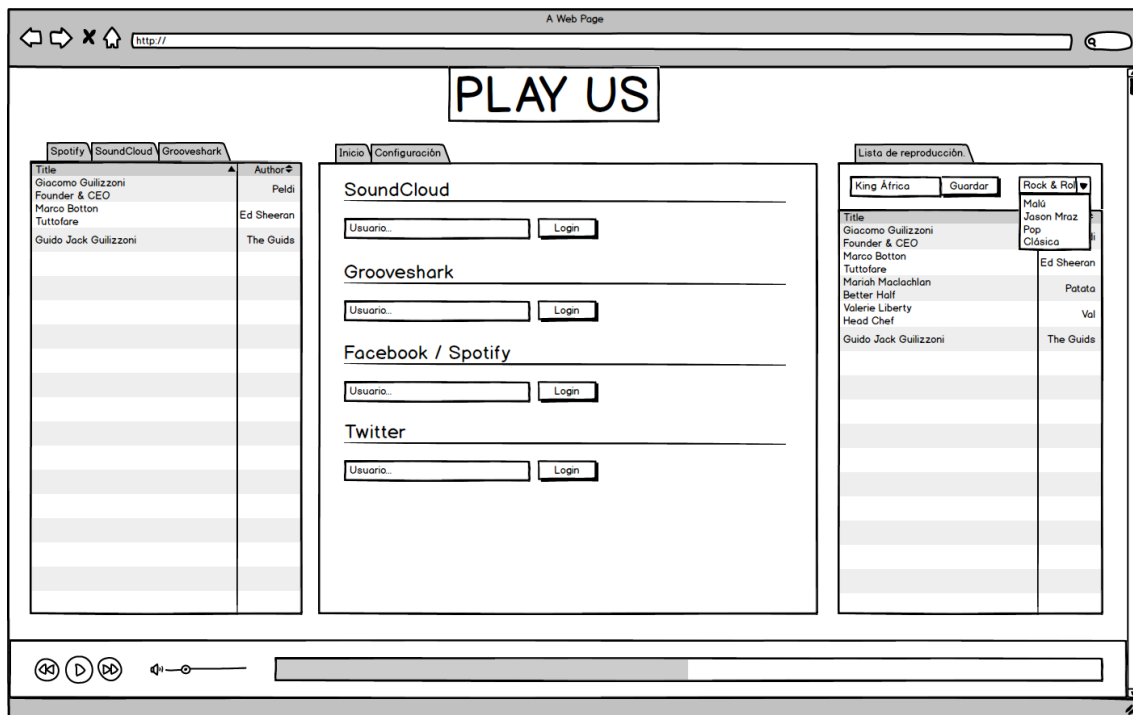


FIGURA 2. PROTOTIPO DE INTERFAZ DE USUARIO DE LA VISTA CONFIGURACIÓN.

3 Arquitectura

3.1 Diagrama de componentes.

Nuestra aplicación va a utilizar diferentes servicios web, entre ellos SoundCloud, Spotify y Groovespark para reproducir canciones y Twitter y Facebook como redes sociales.

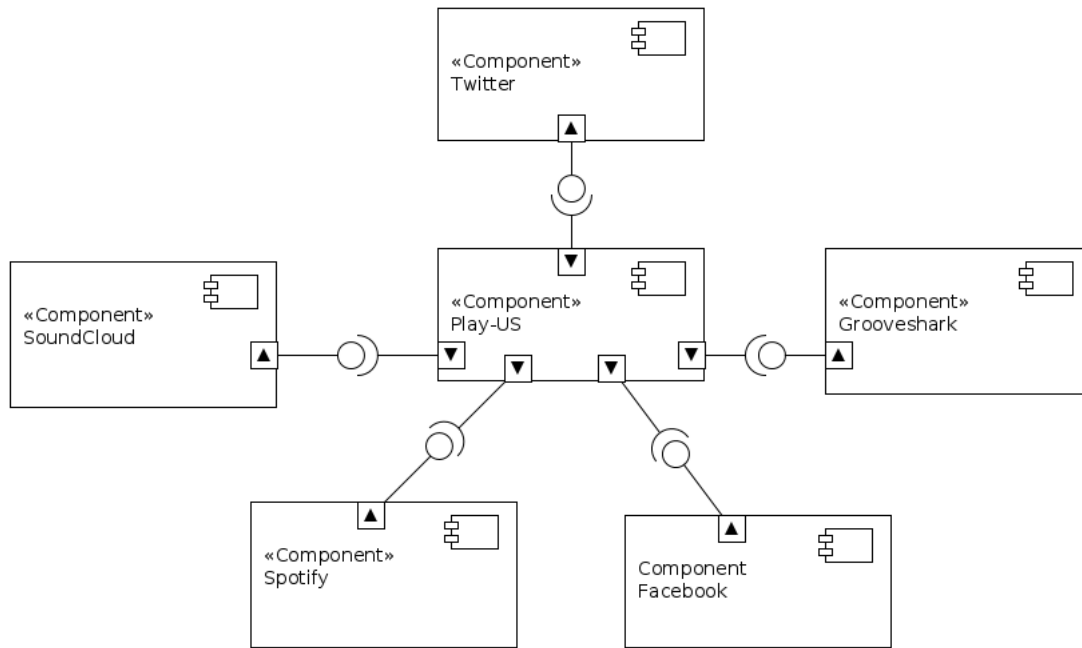


FIGURA 3. PROTOTIPO DE DIAGRAMA DE COMPONENTES.

3.2 Diagrama de despliegue

El mashup estará desarrollado en tecnología GWT y se ejecutará en AppEngine en un servidor de Google. Se podrá acceder a través de un navegador web.

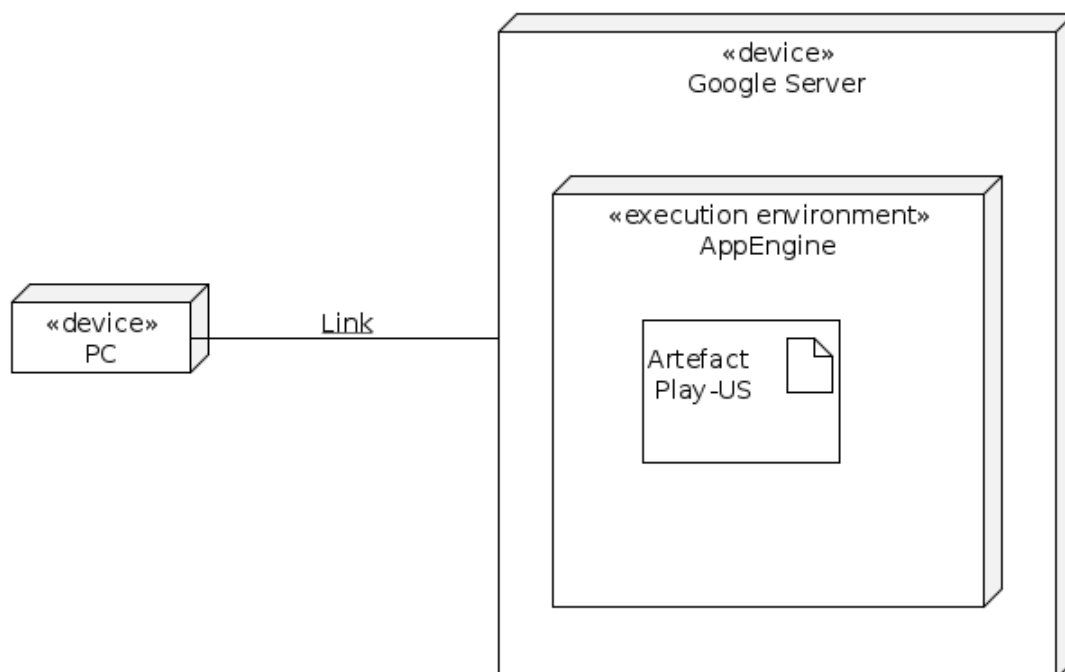
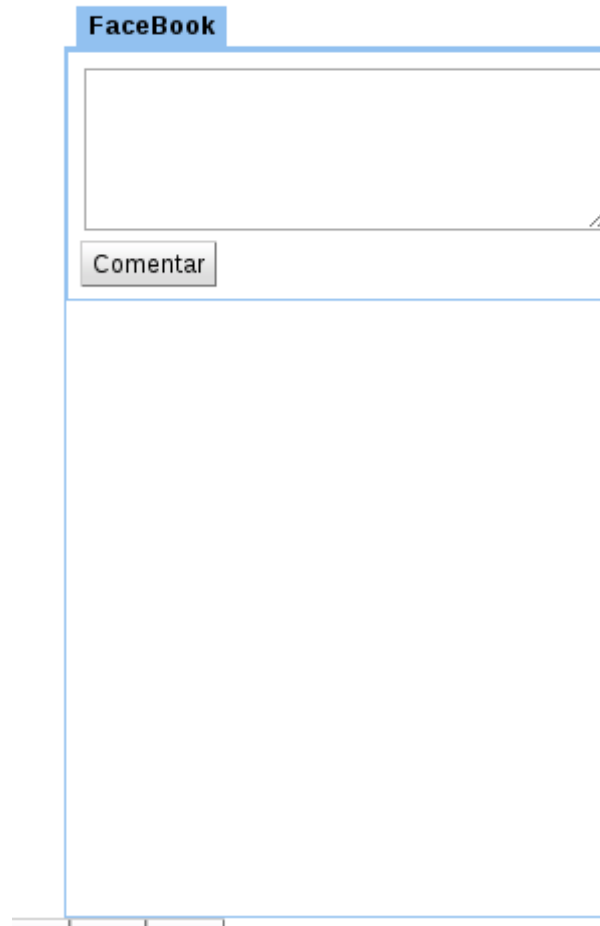


FIGURA 4 . PROTOTIPO DE DIAGRAMA DE DESPLIEGUE.

4 Manual de usuario.

4.1 Mahup.

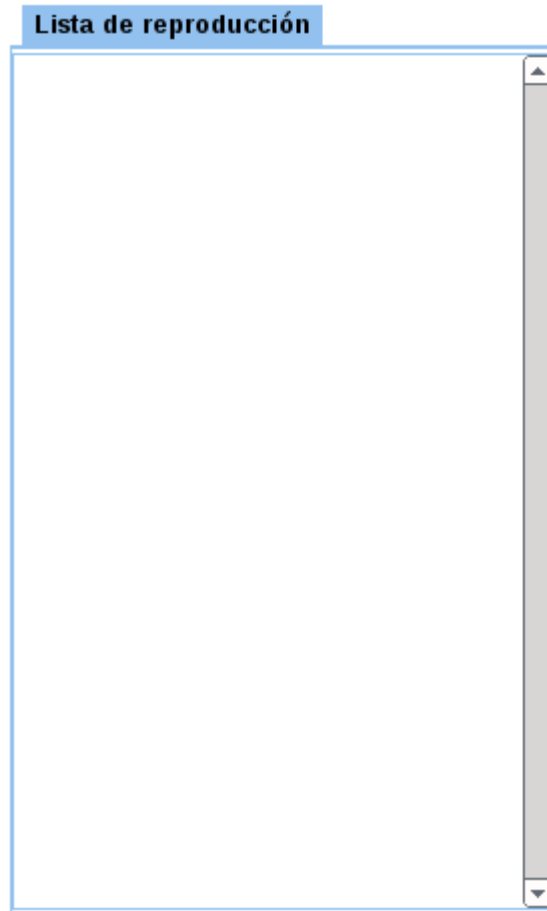


The image shows a user interface for a Facebook-like comment section. At the top, there is a blue tab labeled "FaceBook". Below the tab is a large, empty rectangular text input area. To the right of the input area, there is a small icon of a speech bubble. Below the input area is a button labeled "Comentar". Below the button is a large, empty rectangular area for displaying comments.

Esta parte de la aplicación está reservada para las redes sociales, será desde donde se pueden crear los comentarios.(Todavía no esta funcional).



Vista central de nuestra aplicación. En ella, podemos buscar canciones en diferentes servidores y añadirlas a nuestra lista de reproducción con doble click. También tenemos la pestaña de configuración, donde podremos iniciar sesión en los servidores que sea necesario, como puede ser facebook. En la última pestaña, información, podemos encontrar una descripción del proyecto.(Ahora podemos logearnos en facebook, buscar en todos los servidores y añadir las canciones a las listas de reproducción.)



En esta lista podemos encontrar las canciones que hemos añadido. En versiones siguientes, también tendremos la opción de seleccionar nuestras propias listas guardadas. (Actualmente las canciones de soundcloud funcionan perfectamente, las de grooveshark funcionan en local, pero en el servidor no se añaden correctamente y en spotify solo se añade el nombre de la canción)



Reproductor actual, funciona con las canciones de soundcloud y las de grooveshark en local. No funciona en firefox (No soporta mp3).

4.2 API REST

La API Play-US ofrece recursos propios tales como listas de reproducción o canciones mediante llamadas a otras API's haciendo peticiones HTTP, como GET, POST, DELETE o PUT.

Todos los recursos son accedidos y manipulados de manera similar:

/[nombre del recurso]

Estos recursos pueden tener otros que dependan de él, de la forma:

/[nombre del recurso]/[nombre del subrecurso]

La URI de punto de entrada a la API es:

http://api.play-us.appspot.com

4.2.1 /oauth2/token

Este es el punto de entrada a la API y sólo aceptará peticiones GET. Estas peticiones proveerán de un token de acceso una vez que el usuario haya autorizado a la aplicación.

Parámetros

Parámetro	Tipo	Descripción
client_id	String	ID de la aplicación.
redirect_uri	String	URI de redirección una vez se haya autorizado a la aplicación

4.2.2 /users

Provee usuarios.

Propiedades

Parámetro	Tipo	Descripción
user_list	User[]	Lista de usuarios.

Métodos

Método	Ruta	Descripción
GET	/users	Retorna una lista de usuarios.

Filtros

Parámetro	Tipo	Descripción
q	String	Cadena de filtrado.

4.2.3 /users/{id}

Provee de la información de usuario.

Propiedades

Parámetro	Tipo	Descripción
id	String	ID del usuario.
username	String	Nombre y apellidos de usuario.

Métodos

Método	Ruta	Descripción
GET	/users/{id}	Devuelve el usuario.
POST	/users/{id}	Modifica información de usuario.

4.2.4 /users/{id}/playlists

Provee información de las listas de reproducción de un usuario concreto.

Propiedades

Parámetro	Tipo	Descripción
playlist_list	PlayList[]	Conjunto de listas de reproducción.

Métodos

Método	Ruta	Descripción
GET	/users/{id}/playlists	Información de las listas de reproducción.

Filtros

Parametro	Tipo	Descripción
q	String	Cadena de filtrado.

4.2.5 /users/{id}/playlists/{playlist_name}

Provee información de una lista de reproducción de un usuario concreto.

Propiedades

Parámetro	Tipo	Descripción
id_list	Integer	ID de la lista de reproducción

playlist_name	String	Nombre de la lista de reproducción.
size	Integer	Número de canciones de la lista de reproducción.
song_list	Song[]	Lista de canciones de la lista de reproducción.

Métodos

Método	Ruta	Descripción
GET	/users/{id}/playlists/{playlist_name}	Información de la lista de reproducción.
POST	/users/{id}/playlists/{playlist_name}	Modifica una lista de reproducción
PUT	/users/{id}/playlists/{playlist_name}	Añade una lista de reproducción
DELETE	/users/{id}/playlists/{playlist_name}	Elimina una lista de reproducción.

4.2.6 /songs

Lista de canciones.

Propiedades

Parámetro	Tipo	Descripción
spotify_list	Song[]	Lista de canciones de Spotify
grooveshark_list	Song[]	Lista de canciones de Grooveshark
soundcloud_list	Song[]	Lista de canciones de SoundCloud

Métodos

Método	Ruta	Descripción
GET	/songs	Lista de canciones.

Filtros

Parámetro	Tipo	Descripción
q	String	Cadena de filtrado.

4.2.7 /songs/{server}/{id}

Provee información sobre una canción de un servidor concreto.

Propiedades

Parámetro	Tipo	Descripción
id_song	String	ID de la canción.
name_song	String	Nombre de la canción.
server_song	String	Servidor de la canción.
length_song	Double	Duración de la canción.
artist_song	String	Artista o grupo de la canción.

Métodos

Método	Ruta	Descripción
GET	/songs/{server}/{id}	Información de la canción

Referencias

- [1] *Balsamiq*. <http://balsamiq.com/>
- [2] J. Webber, S. Parastatidis y I. Robinson. *REST in Practice: Hypermedia and Systems Architecture*. O'Reilly Media. 2010.
- [3] UMLet. <http://www.umlet.com/>
- [4] <http://www.gwtproject.org/>