# XMLScore

12 de febrero de 2015

Asier Esteban García

# Parte I
# Resumen

El proyecto propuesto consiste en el desarrollo de una aplicación para el sistema operativo móvil Android OS que permita al usuario crear y editar composiciones musicales mediante pentagramas que una vez finalizados serán exportados al formato abierto MusicXML, de manera que puedan ser reproducidos, editados y visualizados en herramientas de terceros que soporten dicho formato.

Este proyecto consta de diversas fases, a saber, la planificación integral del proyecto, tanto temporal como económica, el diseño de la aplicación con el correspondiente estudio de las tecnologías requeridas para el mismo, la implementación del código que compondrá la aplicación, las pruebas que garanticen en la medida de lo posible la calidad y robustez del código y la defensa de este proyecto ante un tribunal.

Este proyecto no tiene como única finalidad la creación de la aplicación mencionada, sino que también tiene por objeto la puesta en práctica de los conocimientos adquiridos durante el período formativo y conocer de primera mano las distintas facetas que compone un proyecto, como son los problemas inesperados que surgen y la capacidad para resolverlos de manera eficaz en el menor tiempo posible, la gestión del tiempo invertido en las tareas y el cuidado de los detalles, que son de gran importancia y que necesitan una gran cantidad de tiempo para su limado.

# Índice

# Índice de figuras

# Índice de cuadros

# Parte II
# Memoria.

## 1. Introducción.

El gran avance experimentado en la tecnología durante estos últimos años ha dotado a la sociedad de nuevas herramientas de trabajo. Además, las generaciones actuales de estudiantes y con más fuerza aún las generaciones venideras, van a emplear estas plataformas de manera más asidua y por ello es importante dotar a este sector de la sociedad de herramientas apropiadas para su formación. Es, por tanto, importante formarse en estas tecnologías para poder satisfacer adecuadamente la demanda social que esto supone.

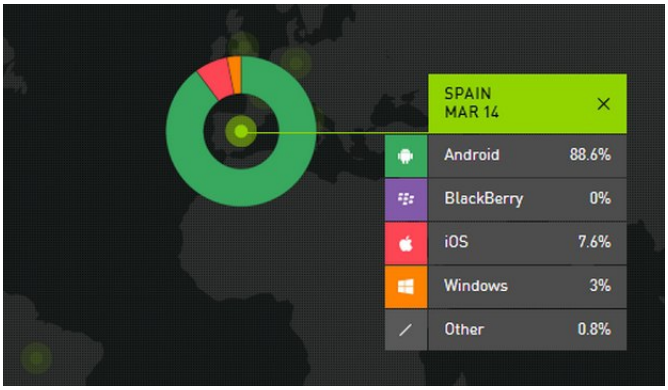



Figura 1: Distribución de los sistemas operativos en dispositivos móviles.

De todas las tecnologías que han proliferado, la de los dispositivos móviles es la que más goza de estar en auge en la sociedad. Sobre estos terminales podemos destacar dos grandes grupos atendiendo a cuotas de mercado, que apuntan a los dos gigantes en el enfoque de estos aparatos: iOS, desarrollado por Apple Inc. y Android, desarrollado por Google Inc.

El enfocar este proyecto en esta dirección tiene por motivos, la formación en esta tecnología, la cual tiene un gran potencial tanto presente como futuro e incluye la investigación de la misma y la puesta en práctica de los diferentes conocimientos adquiridos durante la formación. Además será un buen indicador para la estimación de tiempo que se tarda en realizar cada una de las actividades de las que está compuesto el proyecto.

Este proyecto tiene como uno de sus pilares el estándar MusicXML, el cual posibilita al usuario la visualización de sus composiciones con una gran variedad de herramientas de visualización y edición para ampliar los proyectos realizados

en esta aplicación.

Por tanto, la aplicación que se va a desarrollar servirá para crear y componer música basándose en pentagramas, arrastrando figuras musicales sobre un pentagrama dividido en compases.

# 2. Planteamiento inicial.

Los objetivos de este proyecto consisten en investigación sobre la programación en Android y las posibilidades que ofrece, establecer una base para la estimación de duración de las tareas, planificación de las tareas a realizar, ampliación de las competencias en materia de programación, así como la iniciación en la dirección de proyectos con los problemas que esto conlleva.

## 2.1. Objetivos.

Los objetivos marcados para este proyecto son los siguientes:

- Los pentagramas creados serán en clave de sol.
- La clave tendrá, en caso de que el usuario lo requiera, una armadura, la cual no se podrá modificar una vez creado el pentagrama.
- Las notas de que se dispondrá serán: Redonda, Blanca, Negra, Corchea, Semicorchea, Fusa y Semifusa, así como sus figuras de silencio relacionadas.
- El pentagrama a crear tendrá el compás que el usuario requiera. Sin embargo, esta opción estará limitada a unos valores previamente establecidos.(2/4, 3/4, 4/4, 6/8, 9/8).
- El formato de guardado de los pentagramas creados será MusicXML.
- Se podrán abrir pentagramas creados anteriormente para editarlos o bien seguir elaborándolos.
- Solo habrá un idioma disponible. No obstante, la programación de la aplicación está diseñada de tal manera que la adición de un nuevo idioma sea sencilla.

## 2.2. Planificación temporal.

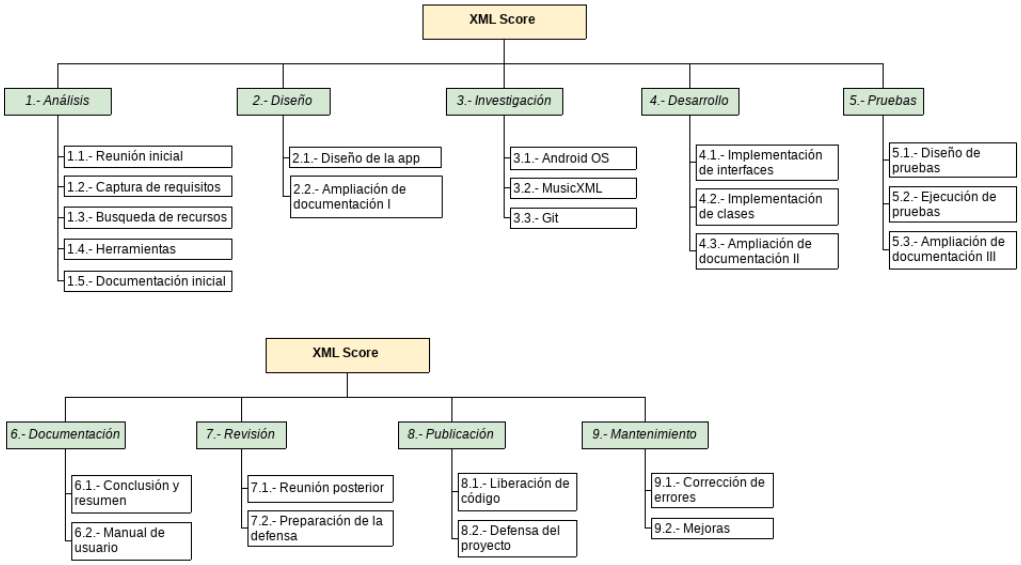
### 2.2.1. Estructura de Descomposición del Trabajo.

Las fases en que se ha dividido el proyecto se especifican a continuación:

1. Análisis.
2. Diseño.
3. Investigación.
4. Desarrollo.
5. Pruebas.
6. Documentación.

7. Revisión.
8. Publicación.
9. Mantenimiento.

Una vez presentadas las fases y definidas inicialmente, se procede a detallar las tareas en las que se subdivide el proyecto.




Cuadro 1: Estructura de descomposición de trabajo.

A continuación se describe detalladamente cada tarea.

**Análisis.**

1.- Análisis
Paquete de trabajo: 1.1.- Reunión inicial.
Duración: 1 hora.
Descripción.
En esta tarea se concretan con el tutor las carácteristicas deseadas para la aplicación, así como los métodos a emplear durante el proyecto en distintos ámbitos (reuniones intermedias, entregas...).
Entradas.
Ninguna.
Salidas/Entregables.
Información acerca de la aplicación y la normativa para el desarrollo del proyecto.
Recursos necesarios.
Ninguno.
Precedencias.
Ninguna.

1.- Análisis
Paquete de trabajo: 1.2.- Captura de requisitos.
Duración: 3 horas.
Descripción.
Tras analizar la información obtenida de la reunión, se establecen los requisitos de la aplicación. El objetivo es desarrollar el comportamiento de la aplicación.
Entradas.
Los datos de la reunión.
Salidas/Entregables.
Diagramas de casos de uso y modelo del dominio.
Recursos necesarios.
Un PC.
Precedencias.
1.1.

1.- Análisis
Paquete de trabajo: 1.3.- Búsqueda de recursos.
Duración: 8 horas.
Descripción.
Una búsqueda de los recursos necesarios para el desarrollo del proyecto: Procesador de textos, herramientas para diagramas, entorno de programación, bibliografía para el aprendizaje...
Entradas.
Ninguna.
Salidas/Entregables.
Colección de útiles a emplear durante el proyecto.
Recursos necesarios.
Un PC, Internet.
Precedencias.
Ninguna.

1.- Análisis
Paquete de trabajo: 1.4.- Herramientas.
Duración: 2 horas.
Descripción.
Selección de las herramientas preferidas para el desarrollo de la app.
Entradas.
La colección de recursos.
Salidas/Entregables.
Selección de las herramientas.
Recursos necesarios.
Un PC.
Precedencias.
1.3.

1.- Análisis
Paquete de trabajo: 1.5.- Documentación inicial.
Duración: 10 horas.
Descripción.
Una vez realizado el estudio preliminar, se procede a redactar el documento de objetivos del proyecto.
Entradas.
Los datos iniciales.
Salidas/Entregables.
El documento de objetivos de proyecto.
Recursos necesarios.
Un PC.
Precedencias.
1.1.

**Diseño.**

2.- Diseño
Paquete de trabajo: 2.1.- Diseño de la app.
Duración: 4 horas.
Descripción.
Se diseña la estructura de la aplicación. Se desarrollan esquemas con las diferentes pantallas de la aplicación, teniendo en cuenta las funciones a realizar en cada una de ellas.
Entradas.
Los casos de uso y el modelo del dominio
Salidas/Entregables.
Bocetos del diseño.
Recursos necesarios.
Un PC.
Precedencias.
1.2.

2.- Diseño.
Paquete de trabajo: 2.2.- Ampliación de documentación I.
Duración: 3 horas.
Descripción.
Se transcribe el proceso de diseño a la documentación.
Entradas.
La documentación, los bocetos.
Salidas/Entregables.
Documentación actualizada.
Recursos necesarios.
Un PC.
Precedencias.
1.5, 2.1.

**Investigación.**

3.- Investigación.
Paquete de trabajo: 3.1.- Android OS.
Duración: 10 horas.
Descripción.
De la lista de recursos, se analizan los correspondientes al sistema operativo Android.
Entradas.
Recursos de Android.
Salidas/Entregables.
Conocimiento acerca de Android.
Recursos necesarios.
Un PC.
Precedencias.
1.3.

3.- Investigación.
Paquete de trabajo: 3.2.- MusicXML.
Duración: 10 horas.
Descripción.
De la lista de recursos, se analizan los correspondientes a MusicXML
Entradas.
Recursos de MusicXML.
Salidas/Entregables.
Conocimiento acerca de MusicXML
Recursos necesarios.
Un PC.
Precedencias.
1.3.

3.- Investigación.
Paquete de trabajo: 3.3.- Git.
Duración: 3 horas.
Descripción.
De la lista de recursos, se analizan los correspondientes al sistema Git.
Entradas.
Recursos de Git.
Salidas/Entregables.
Conocimiento de Git.
Recursos necesarios.
Un PC.
Precedencias.
1.3.

**Desarrollo.**

4.- Desarrollo.
Paquete de trabajo: 4.1.- Implementación de interfaces.
Duración: 20 horas.
Descripción.
Se producen las interfaces de la aplicación en el entorno de desarrollo y se programan las clases asociadas.
Entradas.
Los bocetos.
Salidas/Entregables.
Interfaces realizadas.
Recursos necesarios.
Un PC, IDE de Android
Precedencias.
2.1, 3.1, 3.3.

4.- Desarrollo.
Paquete de trabajo: 4.2.- Implementación de clases.
Duración: 70 horas.
Descripción.
Implementación de las clases que conforman el grueso de la aplicación.
Entradas.
Los bocetos, diagrama de clases.
Salidas/Entregables.
Las clases implementadas.
Recursos necesarios.
Un PC, IDE de Android.
Precedencias.
2.1, 3.1, 3.3.

4.- Desarrollo.
Paquete de trabajo: 4.3.- Ampliación de documentación II.
Duración: 5 horas.
Descripción.
Transcripción de los procesos realizados a la documentación.
Entradas.
La documentación.
Salidas/Entregables.
Documentación actualizada.
Recursos necesarios.
Un PC, IDE de Android.
Precedencias.
4.1, 4.2.

**Pruebas.**

5.- Pruebas
Paquete de trabajo: 5.1.- Diseño de pruebas.
Duración: 3 horas.
Descripción.
En este paso se procede a realizar un profundo estudio de las partes críticas de la aplicación y se determinan las pruebas que se habrán de realizar sobre la aplicación para probar la fiabilidad de la misma.
Entradas.
La aplicación.
Salidas/Entregables.
Las pruebas a realizar.
Recursos necesarios.
Un PC.
Precedencias.
2.2.

5.- Pruebas
Paquete de trabajo: 5.2.- Ejecución de pruebas.
Duración: 3 horas.
Descripción.
Se ejecutan las pruebas acordadas. Si alguna fallara se procederá a revisar la aplicación y arreglar los errores encontrados.
Entradas.
Las pruebas diseñadas.
Salidas/Entregables.
Posibles errores. La aplicación con las pruebas superadas.
Recursos necesarios.
Un PC. Un terminal con Android y la app instalada.
Precedencias.
4.2, 5.1

5.- Pruebas.
Paquete de trabajo: 5.3.- Ampliación de documentación III.
Duración: 2 horas.
Descripción.
Transcripción de los resultados de las pruebas y los arreglos realizados en caso de haberlos realizado.
Entradas.
El informe de las pruebas.
Salidas/Entregables.
Documentación actualizada.
Recursos necesarios.
Un PC. Un terminal con Android
Precedencias.
5.2.

**Documentación.**

6.- Documentación.
Paquete de trabajo: 6.1.- Conclusión y resumen.
Duración: 10 horas.
Descripción.
Una vez acabada la aplicación se hará una retrospectiva del trabajo realizado.
Entradas.
El proyecto realizado.
Salidas/Entregables.
Documentación actualizada.
Recursos necesarios.
Un PC.
Precedencias.
Todo lo anterior ha de estar completo.

6.- Documentación.
Paquete de trabajo: 6.2.- Manual de usuario.
Duración: 3 horas.
Descripción.
Se detallará en lenguaje natural el funcionamiento de la aplicación y se escribirá pensando en el usuario, las acciones que deberá y/o podrá realizar, posibles problemas, etc...
Entradas.
La aplicación.
Salidas/Entregables.
Documentación actualizada: Manual del usuario completo.
Recursos necesarios.
Un PC.
Precedencias.
4

**Revisión.**

7.- Revisión.
Paquete de trabajo: 7.1.- Reunión posterior.
Duración: 3 horas.
Descripción.
Reunión con el profesor para dar el visto bueno al trabajo realizado. También se podrán proponer mejoras.
Entradas.
El trabajo realizado.
Salidas/Entregables.
El proyecto completo.
Recursos necesarios.
Ninguno.
Precedencias.
Todo lo anterior ha de estar finalizado.

7.- Revisión.
Paquete de trabajo: 7.2.- Preparación de la defensa.
Duración: 20 horas.
Descripción.
Se preparará la presentación para el proyecto mediante diapositivas.
Entradas.
Ninguna.
Salidas/Entregables.
Las diapositivas para la defensa.
Recursos necesarios.
Un PC. Suite ofimática.
Precedencias.
El proyecto ha de estar finalizado y con el visto bueno del tutor.

**Publicación.**

8.- Publicación.
Paquete de trabajo: 8.1.- Liberación de código.
Duración: 1 horas.
Descripción.
Se pondrá a disposición del público general el repositorio con el código de la aplicación para su libre distribución.
Entradas.
El código subido al repositorio.
Salidas/Entregables.
El repositorio liberado.
Recursos necesarios.
Un PC.
Precedencias.
El proyecto terminado.

8.- Publicación.
Paquete de trabajo: 8.2.- Defensa del proyecto.
Duración: 1 horas.
Descripción.
Se expondrá ante el tribunal el proyecto realizado mediante diapositivas.
Entradas.
Ninguna.
Salidas/Entregables.
Ninguna.
Recursos necesarios.
Las diapositivas.
Precedencias.
7.2

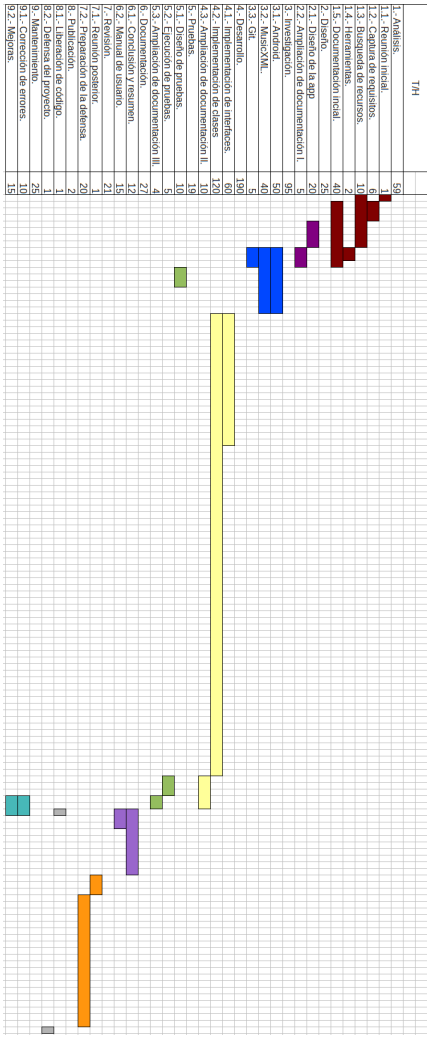
**Mantenimiento.**

9.- Mantenimiento.
Paquete de trabajo: 9.1.- Corrección de errores.
Duración: 3 horas.
Descripción.
Tras analizar la información obtenida de la reunión, se establecen los requisitos de la aplicación. El objetivo es desarrollar el comportamiento de la aplicación.
Entradas.
Los datos de la reunión-
Salidas/Entregables.
Diagramas de casos de uso y modelo del dominio.
Recursos necesarios.
Un PC.
Precedencias.
La reunión con el tutor.

9.- Mantenimiento.
Paquete de trabajo: 9.2.- Mejoras.
Duración: 3 horas.
Descripción.
Tras analizar la información obtenida de la reunión, se establecen los requisitos de la aplicación. El objetivo es desarrollar el comportamiento de la aplicación.
Entradas.
Los datos de la reunión-
Salidas/Entregables.
Diagramas de casos de uso y modelo del dominio.
Recursos necesarios.
Un PC.
Precedencias.
La reunión con el tutor.

### 2.2.2. Gantt

El diagrama de Gantt referente a las tareas descritas anteriormente se presenta a continuación.




Cuadro 2: Planificación temporal.

## 2.3. Herramientas.

Las herramientas que se usarán en este proyecto son las siguientes:

- LyX: Editor de texto basado en LaTeX.
- Cacoo: Herramienta online para la creación de diagramas.
- Visual Paradigm: Herramienta para creación de diagramas.
- JSON: (JavaScript Object Notation) Formato empleado para intercambio de datos, en este caso para guardar pentagramas.
- Android Studio: Entorno de desarrollo integrado de Java basado en el editor IntelliJ Idea, orientado a programar en Android. Recomendado por Google Inc.
- BitBucket: Alojamiento en la nube para proyectos de informática, el cual soporta diferentes sistemas de control de versiones (Git, SVN, Mercurial).
- Git: Sistema de control de versiones para proyectos de programación.
- Sublime Text 2: Editor de texto avanzado con capacidad sintáctica para varios lenguajes de programación.
- Dropbox: Alojamiento en la nube de todo tipo de ficheros.

- 1 PC
- 1 portátil
- Un smartphone con Android: Para pruebas físicas se hará uso de un Nexus 5 con la versión 4.4.4 en adelante (4.4.4, 5.0, 5.0.1), de LG.

## 2.4. Gestión de riesgos.

La gestión de riesgos de este proyecto se detalla a continuación:

| | |
|---|---|
| Descripción | Retraso en la planificación temporal establecida al principio del proyecto. |
| Prevención | Reuniones de seguimiento para asegurar que se cumplen los plazos preestablecidos. |
| Plan de contingencia | Restablecer la planificación y valorar por qué se retrasa. |
| Probabilidad | ALTA. Debido a otras labores a realizar, existe una posibilidad real de que las previsiones temporales sufran desvíos. |
| Impacto | MEDIO/ALTO. Retrasaría los plazos posteriores en la panificación temporal y en última instancia la fecha de finalización del proyecto. |
| Prioridad | ALTA. Valorando la probabilidad, el impacto y la magnitud, de dicho riesgo, se tratará como uno de prioridad alta. |

| | |
|---|---|
| Descripción | Pérdida del contenido del proyecto desarrollado hasta el momento. |
| Prevención | Sincronización de los diferentes módulos de que se compone el proyecto en la nube. Dropbox para documentación y Bitbucket para código fuente. |
| Plan de contingencia | Restablecer la información haciendo uso de dichos backups y valorar como tuvo lugar la pérdida de información para evitar que se produzca de nuevo en un futuro. |
| Probabilidad | BAJA. Los servicios en la nube tienen a su vez copias de seguridad y al estar duplicados en local en 2 maquinas diferentes, en caso de pérdida en la nube existen otras copias. |
| Impacto | BAJO/ALTO. El impacto dependería de en qué fase del proyecto nos encontráramos si este riesgo llegara a hacerse efectivo. Si tuviera lugar al principio el impacto sería mínimo pero de ocurrir en las fases finales del desarrollo, el impacto sería de gran importancia. |
| Prioridad | ALTA. Valorando los puntos anteriores y aunque la probabilidad de que ocurra sea muy baja, se tratará como un riesgo de prioridad alta ya que el impacto y la magnitud podrían llegar a ser de gran importancia |

| | |
|---|---|
| Descripción | Inhabilitación de una máquina de trabajo. (Robo, funcionamiento erróneo, fallo de algún componente). |
| Prevención | Tener diversas máquinas de trabajo preparadas. Concretamente 3. |
| Plan de contingencia | Se usará la máquina secundaria y en caso de fallo de esta, la terciaria. En caso de fallar esta última se considerará la posibilidad de usar una máquina de menor potencia (netbook) para continuar con el trabajo. |
| Probabilidad | MEDIA. Dado que se van a usar una distribución de Linux avanzada (Arch Linux) que es muy sensible a los cambios y una versión técnica de Windows (Windows 10), la probabilidad de que alguna máquina falle es alta. |
| Impacto | BAJO. Al tener más máquinas el impacto que produciría la perdida de un una de ellas no sería grande. Además si el fallo en Linux es debido a algún fallo del sistema, la recuperación de la máquina es casi inmediata y sin pérdida de información importante. |
| Prioridad | BAJA. Considerando los puntos anteriores, la prioridad establecida para este riesgo es baja. |

## 2.5. Evaluación económica.

El proyecto tiene unos costes fijos y variables que se definen a continuación.

| Gastos fijos | | Gastos variables | |
|---|---|---|---|
| Concepto | Precio (hora) | Concepto | Precio (hora) |
| Luz | 2.50€ | Equipos informáticos | 5€ |
| Agua | 1.80€ | Consumibles | 2€ |
| Teléfono | 0.25€ | Salario | 12€ |
| Internet | 0.50€ | | |
| Oficina | 1.90€ | | |
| **Total** | **6.95€** | **Total** | **19€** |

Se expone ahora el flujo de caja esperado para el proyecto. Nótese que el salario está prorrateado en cada tarea en función de las horas necesarias para realizar cada tarea. Las horas totales invertidas en este proyecto suman un total de 463 horas, lo que supondría un salario de 5556€ en total para el desarrollador.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Gastos | 13,95 | 41,85 | 111,6 | 27,9 | 139,5 | 55,8 | 41,85 | 139,5 | 139,5 | 41,85 |
| Ingresos | 110 | 36 | 96 | 24 | 120 | 48 | 134 | 120 | 120 | 36 |
| Flujo de caja | 96,05 | -5,85 | -15,6 | -3,9 | -19,5 | -7,8 | 92,15 | -19,5 | -19,5 | -5,85 |
| Acumulado | 96,05 | 90,2 | 74,6 | 70,7 | 51,2 | 43,4 | 135,55 | 116,05 | 96,55 | 90,7 |
| | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Gastos | 279 | 976,5 | 69,75 | 41,85 | 41,85 | 27,9 | 139,5 | 41,85 | 41,85 | 279 |
| Ingresos | 240 | 840 | 158 | 36 | 36 | 24 | 120 | 36 | 134 | 240 |
| Flujo de caja | -39 | -136,5 | 88,25 | -5,85 | -5,85 | -3,9 | -19,5 | -5,85 | 92,15 | -39 |
| Acumulado | 51,7 | -84,8 | 3,45 | -2,4 | -8,25 | -12,15 | -31,65 | -37,5 | 54,65 | 15,65 |
| | 20 | 21 | 22 | 23 | | | | | | |
| Gastos | 13,95 | 13,95 | 41,85 | 41,85 | | | | | | |
| Ingresos | 12 | 12 | 36 | 36 | | | | | | |
| Flujo de caja | -1,95 | -1,95 | -5,85 | -5,85 | | | | | | |
| Acumulado | 13,7 | 11,75 | 5,9 | 0,05 | | | | | | |

Figura 2: Flujo de caja del proyecto.

Analizando la figura anterior:

- El apartado de gastos incluye los gastos fijos en función de las horas invertidas en cada tarea.
- El apartado de ingresos comprende el salario en función de las horas necesarias para cada tarea.
- La inversión máxima que hay que afrontar para este proyecto es de 84,80€.
- Este es un proyecto sin ánimo de lucro, por tanto, el beneficio final es un valor simbólico.
- Durante las tareas 1, 2, 4 y 6 se reciben pagos de 98€ para sufragar gastos.

# 3. Análisis de antecedentes.

El reto propuesto tiene diversas aproximaciones. Se puede orientar a una aplicación para computadoras, sin embargo, esta opción plantea diversos problemas como pueden ser requisitos mínimos de la máquina y sistema y disponibilidad de computadoras. Por la rama de los dispositivos móviles, se aprecian trabas como la versión del sistema del dispositivo y también la disponibilidad. Sin embargo, para este último caso, las trabas son menos difíciles de superar. Para realizar este proyecto, habrá que iniciarlo desde 0 ya que, a pesar de que existen proyectos parecidos, no se posee código base. No obstante se reutilizarán librerías de terceros en este proyecto.

Uno de los puntos importantes de esta aplicación es que los pentagramas resultantes se exportarán en formato Music MXL, el cual es un estándar formato de código abierto que cada vez usan más programas comerciales. Es por esto que se descarta un formato propio para la aplicación.

## 3.1. Comparativas con otros proyectos existentes.

Para la plataforma escogida, Android OS, los proyectos existentes más destacados en éste ámbito son: Pitch Genie y Music Composition.

**Pitch Genie. (Ref. 8 de la bibliografía)** Este proyecto es el más parecido al propuesto en este proyecto, sin embargo su usabilidad es algo confusa y sus interfaces han quedado obsoletas para las nuevas versiones de Android.

**Music Composition. (Ref. 2 de la bibliografía)** Esta aplicación además de crear pentagramas con una configuración avanzada es capaz de reproducir los sonidos de las mismas con diferentes instrumentos. No exporta a MXL, solamente exporta una imagen PNG del pentagrama y/o un fichero MIDI con los sonidos del mismo. Su interfaz de usuario también ha quedado obsoleta para las nuevas versiones de Android.

## 3.2. MusicXML

Esta aplicación tiene como pilar la capacidad de exportar los pentagramas en formato MusicXML.

MXML es un formato de código abierto basado en XML empleado para la notación musical. Lo que hace que este formato tome fuerza ante otras opciones es su versatilidad a la hora de usarlo en diferentes plataformas.

> Desde 2008 se incluyen entre estas plataformas programas como Finale, Sibelius, Cubase, Musescore y Rosegarden.[1]

El proyecto a desarrollar contiene una clase de exportación a este formato de creación propia, y para su diseño e implementación se emplearán algunas partituras extraídas de Musescore y serán analizadas mediante ingeniería inversa. Éste será un proceso simple, dado que el formato es abierto (como se ha indicado anteriormente) y es legible en cualquier editor de textos. En este caso se hará uso de la versión comunitaria de Sublime Text 2.

MusicXML es un formato que está formado por una gran cantidad de etiquetas, las cuales posibilitan un nivel de detalle muy profundo para los pentagramas. En esta aplicación, las etiquetas empleadas se muestran y explican a continuación. (Nótese que son extractos de la clase Writer, encargada de la creación de ficheros en formato MXL)

- ```
  <?xml version="1.0" encoding="UTF-8" standalone="no"?>
  <!DOCTYPE score-partwise PUBLIC
          "-//Recordare//DTD MusicXML 3.0 Partwise//EN"
          "http://www.musicxml.org/dtds/partwise.dtd">
  ```

  Estas etiquetas conforman la definición del documento y lo identifica como documento XML con su versión y codificación junto con una descripción.

- ```
  <score-partwise version="3.0">
          <part-lists/>
          <parts/>
  </score-partwise>
  ```

  Esta etiqueta es la que engloba el pentagrama, indicando en primer lugar su versión (en este caso 3.0), a continuación la lista de partes que componen el pentagrama y por último la definición de cada una de las partes. Estas partes mencionadas son los hilos de música que componen un pentagrama, los cuales pueden ser instrumentos o voces.

- ```
  <part-list>
          <score-part id="P1">
                  <part-name>Music</part-name>
          </score-part>
  </part-list>
  ```

[1] Extracto de http://es.wikipedia.org/wiki/Musicxml
  ```

Las partes están descritas con un identificador y un nombre que describa la parte. Este nombre aparece a la izquierda del pentagrama. También existe una etiqueta de abreviado para las siguientes filas de un pentagrama pero no se hace uso de ellas en esta aplicación.

```
<part id="P1">
        <measure number="1">
                <attributes>
                        <divisions></divisions>
                        <key>
                                <fifths></fifths>
                                <mode></mode>
                        </key>
                        <time>
                                <beats></beats>
                                <beat-type></beat-type>
                        </time>
                        <clef>
                                <sign>G</sign>
                                <line>2</line>
                        </clef>
                </attributes>
                <las notas/>
        </measure>
</part>
```

Cada una de las partes se especifica indicando en primer lugar el identificador que le corresponde (véase el punto anterior), seguido de los compases de los que consta la parte en cuestión mediante la etiqueta measure, que va acompañada del número del compás. Cada compás está definido por unos atributos incluidos entre las etiquetas <attributes>.

Estos son divisions, que indica la medida de la nota más pequeña usada en cada compás. No obstante, en esta aplicación será un valor fijo, 16 (es este valor y no otro ya que el valor 1 corresponde a la nota negra), que define la unidad más pequeña posible de la misma (corresponde a una semifusa, siendo 1 la negra, 2 la corchea, 4 la semicorchea, 8 la fusa y 16 la semifusa).

A continuación se establece la armadura bajo la etiqueta <key> que consta de dos atributos, fifths que expresa el número de bemoles (números negativos desde -1 a -7) o sostenidos (números positivos de 1 a 7) junto con el modo musical, que en esta aplicación es «major» por defecto, o ninguna (número 0) en cuyo caso no se requiere el modo musical.

También se define el tiempo del compás con la etiqueta <time> con los atributos <beats> (el numerador del compás) y <beats-type> (el denominador del compás).

Por último se define la clave con <clef> (en esta aplicación es por defecto <sign> G y <line> 2, ya que sólo se crean pentagramas en clave de Sol).

Además también se incluyen las notas pertenecientes a cada compás.

- 
```
<note>
        <pitch>
                <step></step>
                <octave></octave>
        </pitch>
        <duration></duration>
        <type></type>
        <stem></stem>
        <accidental></accidental>
</note>
```

Cada nota viene definida por 3 campos, <pitch> que sitúa la nota en el pentagrama con los valores <step> que nombra la nota en formato anglosajón (C (do), D (re), E (mi), F (fa), G (sol), A (la), B (si)) y la octava mediante <octave> que indica a qué octava pertenece (las octavas son cada una de las nueve repeticiones de las notas a lo largo de la escala musical. En esta aplicación está limitada a las octavas 4º, 5º y 6º).En caso de ser la nota un silencio, el primer campo <pitch> es sustituido por el campo <rest/>. También definen a la nota su duración con <duration>, valor logrado multiplicando el atributo <divisions> por el peso de la nota (el peso de la nota se establece marcando la nota más pequeña como valor unitario y siendo cada nota ascendente el doble de la anterior), y por el valor <type> que es el que indica el tipo de la nota: whole (redonda), half (blanca), quarter (negra), eighth (corchea), 16th (semicorchea), 32th (fusa), 64th (semifusa). Se guardan también los atributos <stem>, que dicta el sentido de la plica (up o down) y <accidental> que permite definir accidentes (sostenido, bemol o becuadro). En caso de tener la nota puntillo se agrega el campo <dot/>.

# 4. Captura de requisitos.

El trabajo a desarrollar se define a continuación mediante unos breves diagramas de casos de uso y modelo del dominio.

## 4.1. Casos de uso.

A continuación se expone el diagrama de casos de uso, en el cual se pueden apreciar las acciones que puede realizar el usuario en la aplicación a desarrollar. Los Casos de uso extendidos se encuentran en el A.1

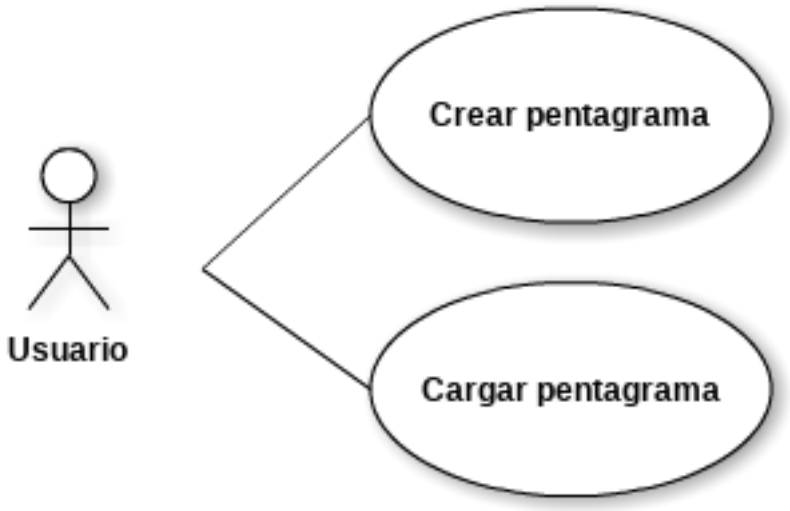



Figura 3: Diagrama de casos de uso

## 4.2. Modelo de dominio.

La siguiente figura representa el modelo de dominio de la aplicación.

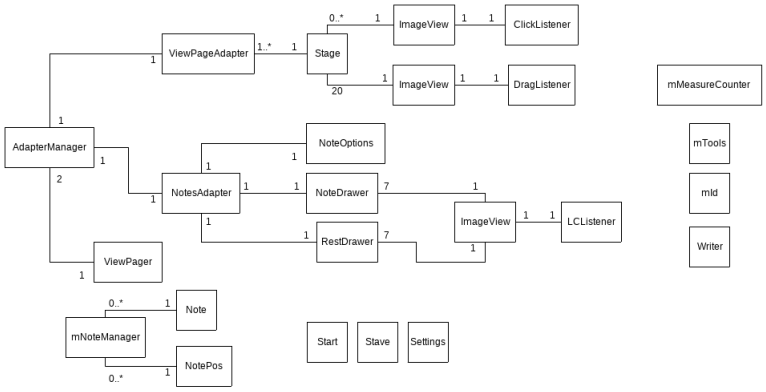



Figura 4: Modelo de dominio

### 4.2.1. MAEs.

El uso de diversas herramientas a lo largo del proyecto ha hecho que la opción del patrón sigleton sea la más adecuada para tener dichas funciones al alcance en cualquier sitio. Son las siguientes:

- AdapterManager.
- mNoteManager.
- mTools.
- mId.
- Writer.

### 4.2.2. Interfaces.

La aplicación está compuesta de diferentes interfaces para que el usuario interactúe con ella. Puesto que para definir las particularidades de estas hay que instanciarlas, se dispone de las clases:

- Start.
- Stave.
- Settings.

#### 4.2.3. Adaptadores.

Los adaptadores son interfaces de facto en android, que al ser empleadas requieren de la reescritura de alguno de sus métodos.

- ViewPageAdapter.
- NotesAdapter.

# 5. Análisis y diseño.

Este proyecto consta de varias partes:

- Clases Singleton.

El uso de este tipo de clases es debido a la necesidad de obtener datos en diferentes clases. Dicha información existe unitariamente, cumpliendo así la característica principal del patrón singleton.

- Clases Standard.

La existencia reiterada de distintos objetos dicta que han de usarse clases que estructuren y encapsulen la información referente a dichos objetos.

- Clases de control de interfaces de usuario.

Las acciones realizadas por las interfaces de usuario han de ser controladas desde clases inherentes a cada interfaz.

- Layouts de interfaces de usuario.

Los diseños de las interfaces de usuario se almacenan en este tipo de ficheros.

## 5.1. Diagrama de clases.

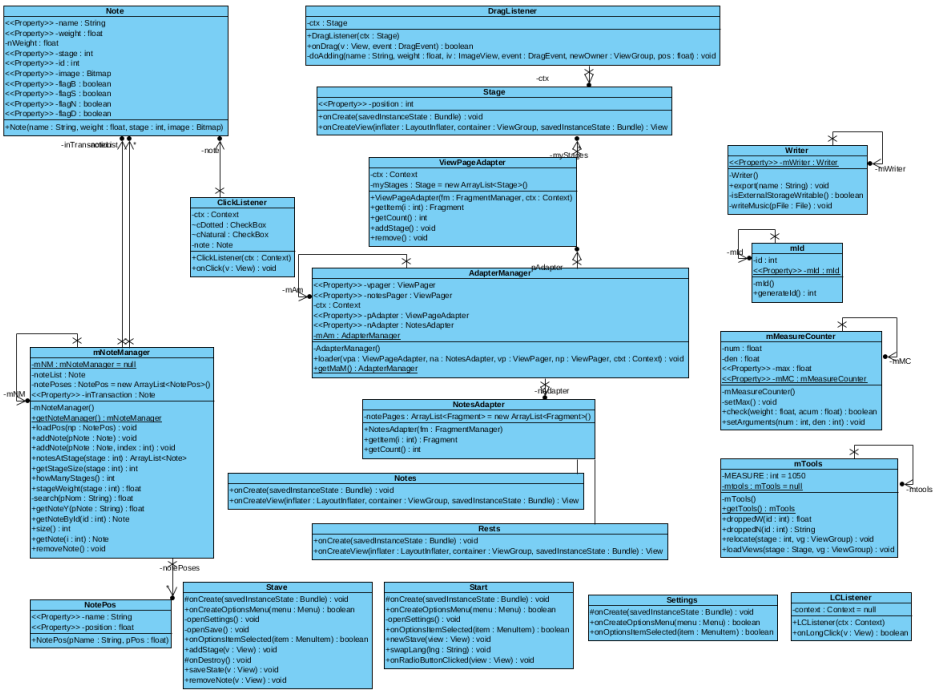



Figura 5: Diagrama de clases

### 5.1.1. Clases.

- Note: La clase que define las notas usadas en la aplicación.
- mNoteManager: El contenedor de notas y sus funciones asociadas para el manejo de las mismas.
- NotePos: Clase que almacena cada octava musical con su posición asociada.
- ClickListener: Clase empleada para la detección de pulsaciones sobre los objetos que la tengas asociada (Las notas situadas en los compases).
- LCListener: Clase usada para detectar pulsaciones largas y reaccionar con vibración como feedback.
- DragListener: Esta clase contiene la definición de las acciones a realizar cuando se detecta movimiento en los objetos que lo tienen asociado.
- Stage: Clase encargada de gestionar cada compás musical del pentagrama.
- ViewPageAdapter: Es un adaptador de vistas infinitas. Su peculiaridad es que las crea y destruye a medida que son necesarias para mantener estable el uso de memoria.
- AdapterManager: Se encarga de gestionar los adaptadores de los objetos de las interfaces. El ViewPageAdapter es el encargado de gestionar los trozos de pentagrama a medida que se crean. NotesAdapter gestiona el cajón de las notas y silencios, así como el panel de edición de las notas y los dos ViewPager son los objetos que contienen dichos adaptadores.
- NotesAdapter: Es un adaptador de vistas finitas, en este caso 3. Siempre están activas en memoria.
- Notes: Clase que gestiona la interfaz que contiene los elementos de notas.
- Rests: Clase que gestiona la interfaz que contiene los elementos de silencios.
- Stave: El pentagrama donde se arrastran las notas.
- Start: La interfaz de inicio. En ella el usuario concreta el pentagrama.
- Settings: Interfaz para las opciones de la aplicación.
- mId: Generador de identificadores para notas.
- Writer: Clase encargada de la exportación y guardado en JSON.
- mTools: Conjunto de herramientas para tratar imágenes.
- mMeasureCounter: Clase que gestiona la capacidad de los compases.

## 5.2. Diagramas de secuencia.

Los diagramas de secuencia más destacables, los cuales son los relativos a las funciones onDrag(), en la clase DragListener, encargada de añadir cada nota al pentagrama, y relocate(), en la clase mTools, encargada de posicionar correctamente las notas sobre los compases, se encuentran en el A.2

# 6. Desarrollo.

Durante el desarrollo del proyecto han surgido diversas necesidades que han alterado de alguna manera los objetivos marcados en un principio. Problemas como cambios necesarios en la implementación, en forma de clases nuevas o modificación de métodos y estructuras debidos a las propiedades del propio sistema han sido de carácter habitual. El problema subyacente ha sido el desconocimiento de la plataforma Android y su funcionamiento interno a pesar de haber leído documentación al respecto.

Un problema de gran importancia surgió a la hora de querer almacenar pentagramas no acabados para su posterior edición. La primera solución en que se pensó fue el diseño de una clase que cargase pentagramas en formato MXL, previa exportación de los mismos al cierre del programa. Esta solución supone una gran inversión en tiempo y esfuerzo, resultando en una vía poco o nada rentable. Más adelante se descubrió el formato JSON, que debido a su composición estructural hace de él la solución más eficiente. La estructura mencionada se basa en pares (identificador, valor) para objetos JSONObject y una colección de JSONObjects para objetos del tipo JSONArray. Al observar esto, y sabiendo que la composición de los datos necesarios para almacenar las notas que forman el pentagrama no es otra que los atributos que la definen con sus respectivos valores, se forma la idea de usar este formato para guardado interno de pentagramas en el sistema. Remarcar que el guardado de estos JSONs es dependiente de la aplicación, es decir, si la aplicación es borrada del terminal, estos archivos se eliminan también.

El otro problema grave que ha existido en el proyecto tiene que ver con la planificación temporal de las tareas. La estimación inicial propuesta fue demasiado ambiciosa y optimista, y tras la realización de las primeras tareas y dada cuenta del retraso, se decidió optar por una revisión de la planificación más realista. Una planificación que debería contemplar en cada tarea las posibles adversidades y los detalles que podrían aparecer (y que de hecho han aparecido). Así, tras la revisión se obtuvo una planificación diseñada más en profundidad y con un carácter más cercano a la realidad.

Un último problema a remarcar fue un fallo humano durante una actualización de sistema del equipo informático principal de desarrollo que tuvo como consecuencia la inoperabilidad del sistema. No se sufrió pérdida de información, y al tener prevista la incidencia y disponiendo de un equipo secundario a disposición, se pudo proseguir con el proyecto con la mayor brevedad posible.

El editor empleado para la implementación de la aplicación ha sido también un punto importante. Esto se debe a la decisión inicial de usar Eclipse, dada la ventaja adquirida durante el transcurso de la carrera sobre el funcionamiento y la familiaridad con el editor y sus funciones en sí. Sin embargo a la hora de desarrollar la aplicación surgieron ciertos inconvenientes que hicieron patente

una necesidad de cambio. La respuesta de las necesidades viene de mano de Android Studio, un entorno de desarrollo específicamente diseñado para desarrollar aplicaciones en Android, que aunque tiene algunos puntos en contra que han pesado (como la necesidad de configurar el sistema de control de versiones Git empleado en el proyecto manualmente), finalmente ha dado la talla más que de sobra.

# 7. Verificación y evaluación.

Las pruebas de este proyecto se basan en la realización de diferentes actividades con la aplicación, enfocando principalmente la atención en los puntos críticos, a saber, la creación inicial del pentagrama, el añadido correcto de cada nota y por último la exportación correcta al formato MusicXML. Para esto último se hará uso de un software de terceros para abrir los ficheros generados durante las pruebas y ver el resultado de las mismas.

Los tests diseñados se detallan a continuación, junto con los resultados de los mismos.

| Descripción. | Creación de un pentagrama con [compás] y [armadura]. |
|---|---|
| Resultado. | Excepción en compás 12/8. |
| Observaciones. | Resuelto el error. (El check de compás no se hacía correctamente). |
| Restest. | Todas las combinaciones crean pentagramas correctamente. |

| Descripción. | Insertar notas en el pentagrama. (Delante/Detrás + Overflow) |
|---|---|
| Resultado. | OK. |
| Observaciones. | |
| Retest. | - |

| Descripción. | Borrar una nota de un compás. |
|---|---|
| Resultado. | OK. Anomalía durante la inserción de nota, si la nota es la última del compás: Se crea un compás extra. |
| Observaciones. | |
| Retest. | - |

| Descripción. | Editar una nota. |
|---|---|
| Resultado. | Error. No controla bien el añadido de puntillo a la nota cuando hay overflow. |
| Observaciones. | Error subsanado. Configuración errónea de la condición. |
| Retest. | OK |

| Descripción. | Guardar un pentagrama en MXL. |
|---|---|
| Resultado. | Errores sintácticos varios y duración errónea |
| Observaciones. | Sintaxis corregida. Duración corregida tras varios intentos. |
| Retest. | OK |

| Descripción. | Guardado automático de un pentagrama en formato JSON a la salida de la edición. |
|---|---|
| Resultado. | OK |
| Observaciones. | |
| Retest. | - |

| | |
|---|---|
| Descripción. | Cargado de un pentagrama a partir de su JSON + Detección de JSONS en el sistema. |
| Resultado. | Errores varios por cambios de tipo de las variables. |
| Observaciones. | Reconfigurados los tipos de variables. |
| Retest. | OK |

# 8. Conclusiones y trabajo futuro.

El proyecto realizado ha permitido dar un paso más en la formación sobre los pilares tratados, tal como se especifica en los objetivos. Entre estos se destaca la programación, y la capacidad de resolver de la mejor manera los problemas surgidos en esta área. Además, esta aplicación ha permitido descubrir de primera mano los efectos derivados de 2 riesgos contemplados en esta documentación y por tanto poner en práctica las contingencias asociadas.

Se ha desarrollado a su vez un interés en el desarrollo de aplicaciones para Android, especialmente en los nuevos estándares de diseño establecidos por Google, denominado Material Design para las versiones actuales de su sistema.

Durante la implementación de la aplicación ha habido muchos cambios, sobre todo estructurales, los cuales se deben a problemas derivados de las propiedades del sistema, teniendo que implementar más clases, cambiando métodos, etc...

Sin embargo, este proyecto no acaba aquí. La aplicación concebida tiene por objeto la formación escolar en materia musical, y como tal ha de ser mejorada en muchas maneras: La mejora y optimización de código para una ejecución más ágil, una interfaz renovada según los estándares marcados por Google (el arriba mencionado Material Design), la posibilidad de reproducir los pentagramas con audio basado en MIDI o similar, aumentar el número de claves disponibles para la creación de pentagramas, mayor variedad de idiomas (la aplicación está diseñada de tal forma que agregar un idioma diferente no lleva demasiado esfuerzo) y una revisión completa para seguir arreglando los fallos ocultos que contiene la aplicación.

La valoración general sobre el trabajo realizado es positiva, pero existen algunos matices que mejorados, permitirán en un futuro mayor eficiencia y eficacia. El matiz en que más hay que trabajar es la gestión del tiempo de trabajo y la organización del mismo.

# 9. Bibliografía.

## Referencias


[1] Jet Brains. Android studio. http://developer.android.com/tools/studio/index.html.

[2] AY Electronics. Music composition. https://play.google.com/store/apps/details?id=com.ayelectronics.MusicComposition&hl=es.

[3] Computer Hoy. Distribucion de los sistemas operativos en dispositivos moviles. http://cdn3.computerhoy.com/sites/computerhoy.com/files/editores/user-11130/cuota_mercado2.jpg.

[4] http://developer.android.com/reference/packages.html. Android developers.

[5] http://es.wikipedia.org/. Wikipedia.

[6] http://stackoverflow.com/. Stack overflow.

[7] http://www.musicxml.com/. Musicxml.

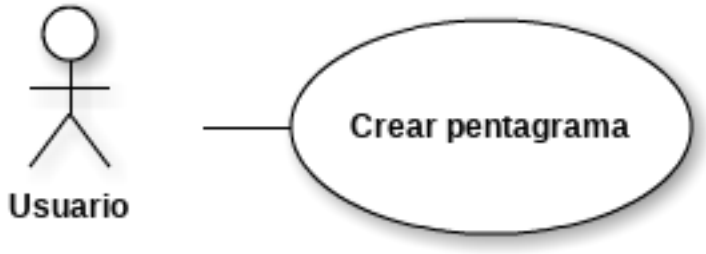
[8] lingyu. Pitch genie. https://play.google.com/store/apps/details?id=com.PitchGenie&hl=es.

[9] Scott McCracken. *Android 4. Desarrollo profesional de aplicaciones.* De profesional a profesional. Inforbook's, 2013.

[10] Unknown. Clef image. http://wrhsonline.net/wrhs-media/2013/05/10/french-orchestra-visits-woodland/.

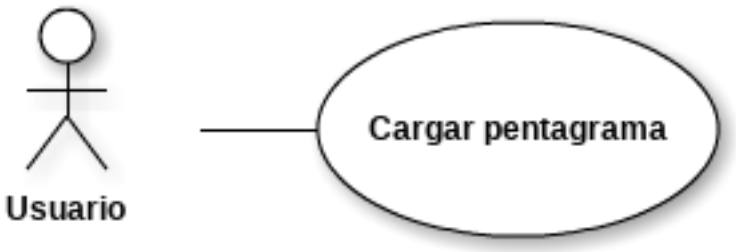
# A. Anexo I. Diagramas de diseño.

## A.1. Casos de uso extendidos.




| |
|---|
| **Nombre:** Crear pentagrama. |
| **Descripción:** El usuario crea un pentagrama y lo rellena con notas. |
| **Actores:** El usuario. |
| **Precondiciones:** Ninguna. |
| **Requisitos no funcionales:** Ninguno. |
| **Flujo de eventos:**<br>1.- El usuario selecciona los parámetros de compás y armadura (si la desea) para su pentagrama. (Imagen <–>)<br>2.- El usuario presiona el botón «Crear pentagrama».<br>3.- Se abre el editor de pentagramas.<br>A.- Si el usuario desea añadir una nueva nota:<br>1.- Selecciona la nota deseada y la arrastra al pentagrama.<br>A'.- Si la nota cabe en el compás:<br>1'.- Se añade la nota.<br>B'.- Si la nota no cabe en el compás:<br>1'.- Se muestra un mensaje de error.<br>B.- Si el usuario desea editar una nota:<br>1.- Selecciona la nota en el pentagrama.<br>2.- El usuario edita la nota y pulsa Aceptar.<br>C.- Si el usuario desea eliminar una nota:<br>1.- Selecciona la nota en el pentagrama y pulsa Borrar.<br>D.- Si el usuario pulsa Guardar:<br>1.- Se guarda el pentagrama en formato MXL.<br>E.- Si el usuario pulsa Atrás:<br>1.- Se guarda el pentagrama.<br>2.- Se vuelve a la pantalla de Inicio.<br>F.- Si el usuario cierra la aplicación:<br>1.- Se guarda el pentagrama. |
| **Post-condiciones:** Se habrá creado el pentagrama. |
| **Interfaz:** Las interfaces se encuentran en el A.3 |

Cuadro 3: Casos de uso extendidos I.




| |
|---|
| **Nombre:** Cargar pentagrama. |
| **Descripción:** El usuario carga un pentagrama creado previamente y lo edita. |
| **Actores:** El usuario. |
| **Precondiciones:** Deben existir pentagramas creados previamente. |
| **Requisitos no funcionales:** Ninguno. |
| **Flujo de eventos:**<br>1.- El usuario selecciona Cargar pentagrama en el menú de inicio. (Imagen <–>)<br>2.- El usuario selecciona el pentagrama que desea editar.<br>3.- Pulsa aceptar y se muestra el pentagrama.<br>A.- Si el usuario desea añadir una nueva nota:<br>1.- Selecciona la nota deseada y la arrastra al pentagrama.<br>A'.- Si la nota cabe en el compás:<br>1'.- Se añade la nota.<br>B'.- Si la nota no cabe en el compás:<br>1'.- Se muestra un mensaje de error.<br>B.- Si el usuario desea editar una nota:<br>1.- Selecciona la nota en el pentagrama.<br>2.- El usuario edita la nota y pulsa Aceptar.<br>C.- Si el usuario desea eliminar una nota:<br>1.- Selecciona la nota en el pentagrama y pulsa Borrar.<br>D.- Si el usuario pulsa Guardar:<br>1.- Se guarda el pentagrama en formato MXL.<br>E.- Si el usuario pulsa Atrás:<br>1.- Se guarda el pentagrama.<br>2.- Se vuelve a la pantalla de Inicio.<br>F.- Si el usuario cierra la aplicación:<br>1.- Se guarda el pentagrama. |
| **Post-condiciones:** Se habrá actualizado el pentagrama. |
| **Interfaz:** Las interfaces se encuentran en el A.3 |

Cuadro 4: Casos de uso extendidos II.

## A.2. Diagramas de secuencia.

En la siguiente figura se exponen los diagramas de secuencia correspondientes a la función onDrag(), perteneciente a la clase DragListener, la cual es la encargada de la adición de notas al pentagrama, y la función relocate(), perteneciente a la clase mTools, la cual organiza los elementos de un compás determinado.

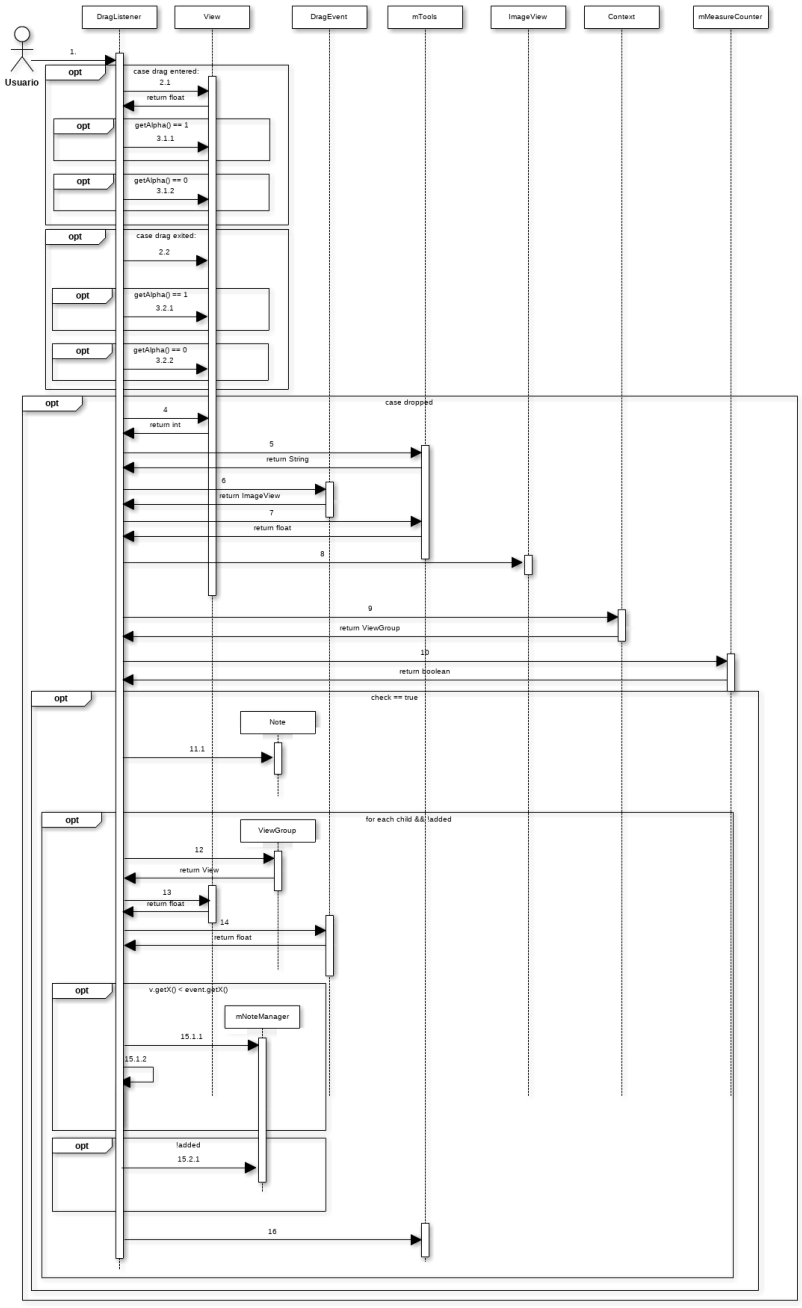



Figura 6: Diagrama de secuencia I (onDrag). Flujo de creación de una nota.

onDrag()
1.- El usuario arrastra la nota sobre el pentagrama.
[case drag entered]
2.1.- getAlpha() : float
[if getAlpha() == 1]
3.1.1.- setAlpha(0)
[if getAlpha() == 0]
3.1.2.- setAlpha(1)
[case drag exited]
2.2.- getAlpha() : float
[if getAlpha() == 1]
3.2.1.- setAlpha(0)
[if getAlpha() == 0]
3.2.2.- setAlpha(1)
[case dropped]
4.- v.getId()
5.- mTools.getTools().droppedN(v.getId())
6.- (ImageView) event.getLocalState()
7.- mTools.getTools().droppedW(target.getId())
8.- event.getLocalState()).setDrawingCacheEnabled(true)
9.- (ViewGroup) ctx.getActivity().findViewById(R.id.notePlace)
10.- mMeasureCounter.getmMC().check(weight, mNoteManager.getNoteManager().stageWeight(ctx.getPosition()))
[if check == true]
11.1.- new Note(name, weight, ctx.getPosition(), ((ImageView) event.getLocalState()).getDrawingCache())
[for each element && !added]
12.- newOwner.getChildAt(i)
13.- getX()
14.- getX()
[if v.getX() < event.getX()]
15.1.1.- mNoteManager.getNoteManager().addNote(note, i); (La i representa la posición de la nota en el compás)
15.1.2.- added = true
[if !added]
15.2.1.- mNoteManager.getNoteManager().addNote(note)
16.- mTools.getTools().relocate(this.ctx.getPosition(), newOwner)

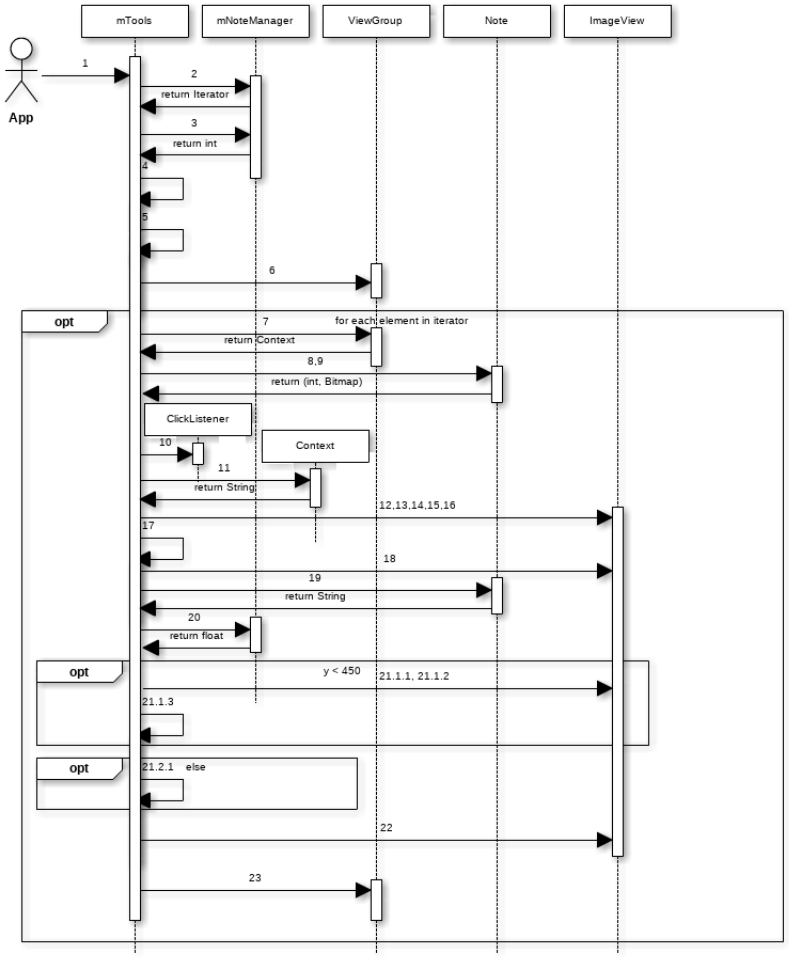



Figura 7: Diagrama de secuencia II (relocate). Flujo de organización de notas.

1.- relocate()
2.- mNoteManager.getNoteManager().notesAtStage(stage).iterator()
3.- mNoteManager.getNoteManager().notesAtStage(stage).size()
4.- step = MEASURE / (size + 1)
5.- accumulate = 0
6.- vg.removeAllViews()
  [for each element]
    7.- vg.getContext()
    8.- note.getId()
    9.- note.getImage()
    10.- new ClickListener(Context ctx)
    11.- getString(R.string.added_notes)
    12.- new ImageView(Context ctx)
    13.- iv.setImageBitmap(Bitmap img)
    14.- iv.setId(int id)
    15.- iv.setOnClickListener(ClickListener cl)
    16.- iv.setContentDescription(String str)
    17.- accumulate += step
    18.- iv.setX(accumulate -150)
    19.- note.getName()
    20.- mNoteManager.getNoteManager().getNoteY(String name)
    [if y < 450]
      21.1.1.- iv.setRotationX(180)
      21.1.2.- iv.setRotationY(180)
      21.1.3.- y -= 50
    [else]
      21.2.1.- y -= 230
    22.- iv.setY(y)
    23.- vg.addView(iv)

## A.3. Interfaces gráficas de los casos de uso.

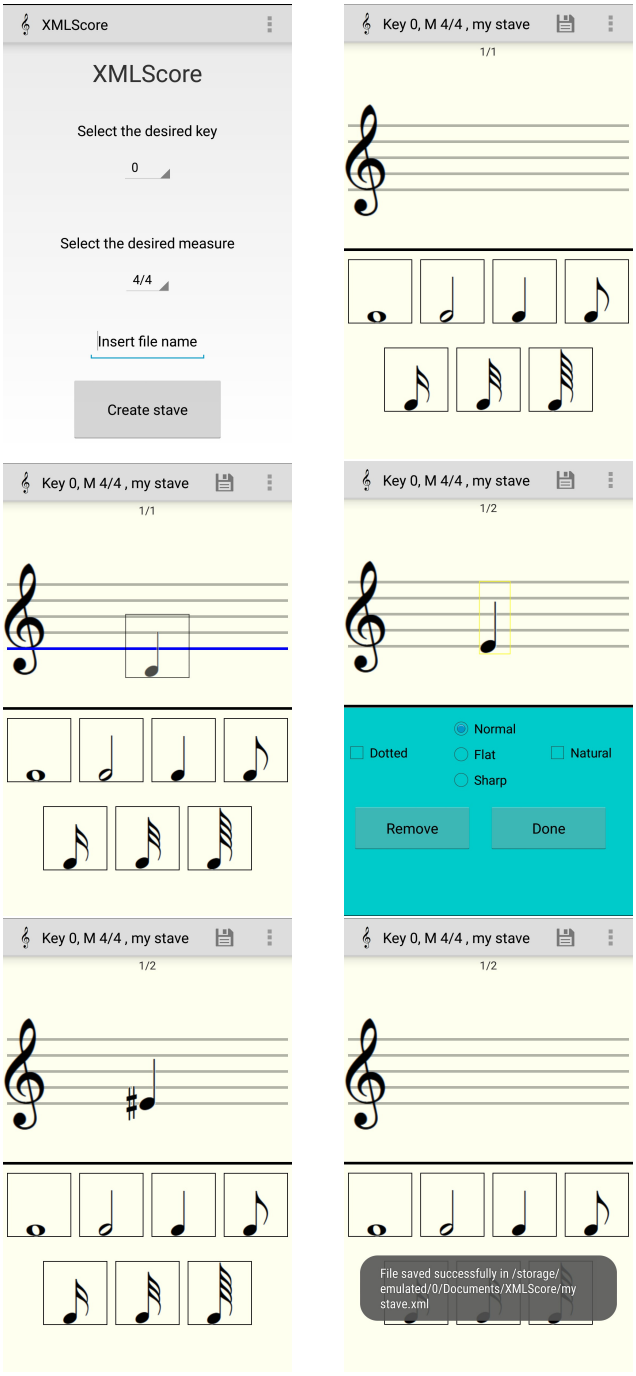



Figura 8: Interfaces del caso de uso Crear Pentagrama.

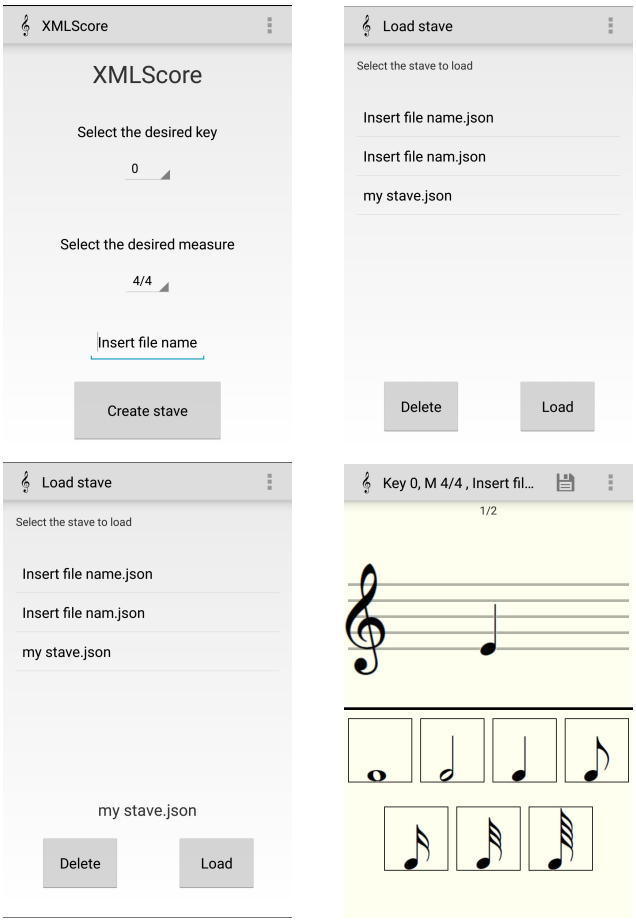



Figura 9: Interfaces del caso de uso Cargar Pentagrama.

# B. Anexo II. Manuales y otros documentos.

## B.1. Manual de usuario.

### B.1.1. Instalación de la aplicación.

El proceso de instalación de esta aplicación puede cursarse de varias maneras. Ha de tenerse en cuenta que existe un requisito mínimo para ejecutar esta aplicación: El terminal donde se instale deberá funcionar bajo Android 4.4.4 como mínimo.

- A través del código fuente:

  El código fuente se encuentra tanto en el repositorio en línea sobre el cual se ha trabajado: BitBucket, como en el paquete adjunto al proyecto. Este puede ser descargado en cualquier momento como archivo comprimido en ZIP. Este archivo se puede importar directamente al entorno de desarrollo Android Studio y bien ejecutando una compilación sobre un dispositivo compatible conectado, o bien generando un archivo instalable (véase punto 2), puede obtenerse la aplicación instalada. La generación del archivo instalable se realiza siguiendo los siguientes pasos:

  - Se desempaqueta el archivo comprimido contenido en el proyecto y se abre abre el Android Studio.
  - En el menú inicial de Android Studio, se ha de seleccionar la opción de Abrir un proyecto de Android Studio ya existente y seleccionar la carpeta creada en el paso anterior.
  - Una vez el proyecto ha sido cargado en Android Studio, seleccionar la opción Generate signed apk del menú Build.
  - Generar una llave de identificación y seguir los pasos para generar el ejecutable.

- Instalación del fichero .apk:

  El archivo .apk puede instalarse habilitando la opción «Orígenes Desconocidos» en el menú de Ajustes, sección de Seguridad, apartado Administración del dispositivo. A continuación, y tras haber transferido el fichero al móvil, ejecutarlo desde cualquier explorador de archivos o aplicación dedicada para instalar aplicaciones.

### B.1.2. Uso de la aplicación.

Bienvenido a la aplicación XMLScore, con la cual podrá crear pentagramas musicales y exportarlos para su continuación, edición y reproducción en otras plataformas con soporte MusicXML.

Una vez iniciada la aplicación se encontrará en el menú principal, en el cual podrá, en caso de querer crear un nuevo pentagrama, seleccionar la armadura

de clave y compás que desee (Recuerde que no podrá cambiar estos valores una vez creado el pentagrama). Además, de manera recomendable, aunque no necesaria, podrá escoger un nombre para su composición, el cual se usará después para nombrar el fichero de exportación. No podrá crear un pentagrama con el nombre ya existente de un pentagrama guardado previamente.

Una vez en la pantalla de edición del pentagrama, encontrará usted en la barra superior los valores previamente seleccionados. Justo en la parte inferior a estos datos se encuentra el índice de compases que le muestra el compás en que se encuentra y los compases que componen su pentagrama. A continuación se encuentra el compás donde podrá depositar las notas. La navegación a través de los diferentes compases de su pentagrama se realiza desplazando a derecha e izquierda cada compás. Por último, en la parte inferior de la pantalla podrá ver el apartado de notas. Este incluye 2 vistas, las cuales contienen las notas y sus respectivos silencios. Podrá moverse entre ellas deslizándolas a derecha e izquierda.

Para seleccionar una nota o silencio, deberá mantener la pulsación sobre la figura hasta que note una ligera vibración del terminal, no debería tomar más de 2 segundos. Sin soltar la pulsación podrá entonces desplazar la figura sobre el compás activo (el que se visualiza en ese instante) y deberá soltar la pulsación en la línea que guste cuando esta se torne azul. El añadido de notas puede realizarse a ambos lados de una determinada nota para situarla delante o detrás.

Cada nota puede ser editada pulsando una vez sobre ella. En la parte inferior le aparecerá el menú de edición donde podrá optar entre aplicar un accidente, agregarle un puntillo o un bemol a la nota. La edición se hará efectiva al momento de pulsar Aceptar. También puede borrar la nota pulsando borrar.

Cuando esté satisfecho con su pentagrama puede guardarlo pulsando el botón de la parte superior que contiene una imagen de un disco de $3^{1/2}$. Los archivos generados se almacenan en su carpeta de Documentos.

Pulsando la tecla atrás en su dispositivo hace que el pentagrama que está editando se guarde en el sistema, y éste puede ser reabierto de nuevo en la aplicación accediendo al menú de carga situado en la parte superior derecha de la pantalla de menú principal. Una vez accedido a este contexto, le aparecerá la lista de los pentagramas que haya guardado y seleccionando uno, le aparecerá el nombre correspondiente en la parte central de la pantalla. Puede usted eliminar dicho pentagrama con el botón Eliminar o bien cargarlo pulsando Cargar.

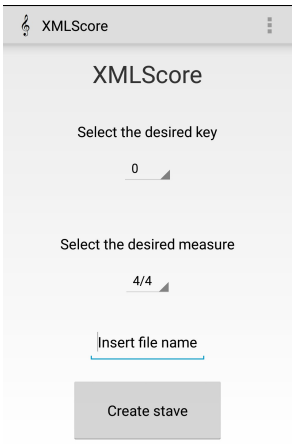



Figura 10: Menú Principal.

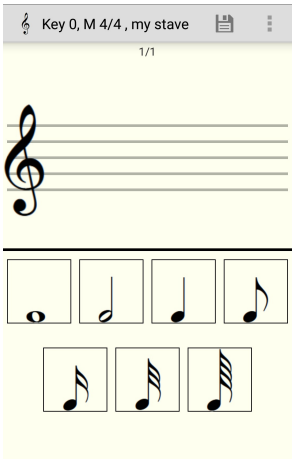



Figura 11: Editor.

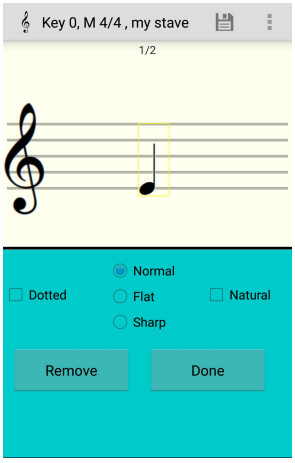
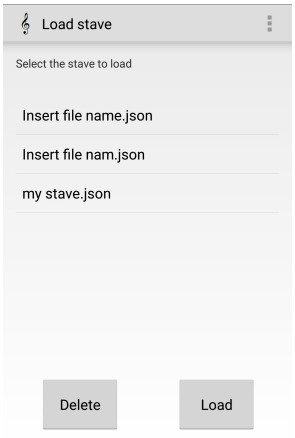



Figura 12: Opciones.




Cuadro 5: Menú de carga.