**14 de marzode 2015**

Miguel Fernández Fuentes Santiago Marqués Rivera Cristina Montes Rubio Ángel Rodríguez Bobes Pablo Manuel Romero Martín Lucía Sánchez Díaz David Sariego Fernández Jorge Sierra Miranda

Universidad de Oviedo

14 de marzode 2015

Proyecto Trivial4a

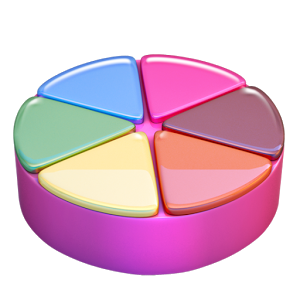


Tabla de contenido

1. [Planteamiento del problema 2](#_Toc416182847)
2. [Stakeholders 3](#_Toc416182848)
   * 1. [Responsables de NoGame 3](#_Toc416182849)
     2. [Desarrolladores de NoGame 3](#_Toc416182850)
     3. [Usuarios de la aplicación 3](#_Toc416182851)
3. [Requisitos de la aplicación 4](#_Toc416182852)
   1. [Requisitos funcionales 4](#_Toc416182853)
   2. [Requisitos no funcionales 4](#_Toc416182854)
4. [Atributos de Calidad 5](#_Toc416182855)
   * 1. [Disponibilidad 5](#_Toc416182856)
     2. [Modificabilidad 5](#_Toc416182857)
     3. [Seguridad 5](#_Toc416182858)
     4. [Testabilidad 5](#_Toc416182859)
     5. [Usabilidad 5](#_Toc416182860)
5. [Lista de Stakeholders 6](#_Toc416182861)
6. [Lista de Atributos de Calidad 7](#_Toc416182862)
7. [Atributos de Calidad e Interesados 8](#_Toc416182863)
8. [Escenarios de Calidad 9](#_Toc416182864)
9. [Vistas 10](#_Toc416182865)
   1. [Diagrama de paquetes 10](#_Toc416182866)
   2. [Diagrama de clases 12](#_Toc416182867)

# Planteamiento del problema

Una empresa del sector de videojuegos llamada NoGame está interesada en crear una nueva línea de productos basadas en el popular juego del Trivial.

En la primera fase de desarrollo de este proyecto, nos centraremos en la recopilación y el manejo de las preguntas que se utilizarán más tarde para abastecer el juego. Para ello, el objetivo es crear un extractor capaz de recibir, en principio, ficheros de preguntas en formato GIFT y pasarlos a formato JSON, que facilitará notablemente su posterior carga en una base de datos. La compañía ha pensado en MongoDb como candidata para ser utilizada como dicha base de datos, pero este dato aún no es seguro.

La idea principal es que el proceso sea sencillo y, lo que es más importante, que la conversión sea totalmente exitosa, por lo que se obviará la parte gráfica de la interfaz y el trabajo se centrará en mostrar los procedimientos llevados a cabo internamente para captar posibles errores que interrumpan el proceso o corrompan la conversión de los archivos. Por tanto, la etapa inicial de desarrollo se centrará en crear y perfeccionar el extractor, permitiendo como única “funcionalidad extra” la posible automatización de la conversión.

Aun así, la compañía deja abierta la puerta a posibles ampliaciones futuras con las que poder trabajar con más formatos, como por ejemplo XML y QTI.

# Stakeholders

### Responsables de NoGame

Se trata de los directivos de empresa.

Son responsables de los presupuestos y toman las decisiones al respecto, buscando un bajo coste de desarrollo, es decir, ajustar el tiempo de desarrollo del proyecto y el coste del mismo.

### Desarrolladores de NoGame

Serán los responsables de desarrollar el sistema resultante de la arquitectura.

#### Desarrolladores del Parser

Desean construir una aplicación sencilla, que lea y procese los datos, y posteriormente los almacene. Asimismo, buscan baja intensidad tecnológica en el proyecto, para obtener un proceso de desarrollo controlado y predecible. No se requiere que la conversión sea interactiva.

### Usuarios de la aplicación

Son los que utilizarán el sistema. Querrán hacerlo de manera sencilla y eficiente.

# Requisitos de la aplicación

## Requisitos funcionales

RF 1. El sistema deberá procesar un fichero de entrada con preguntas y sus respectivas respuestas en formato GIFT.

RF 2. El sistema deberá generar un fichero de salida con preguntas y sus respectivas respuestas en formato JSON.

RF 3. El sistema deberá almacenar las preguntas obtenidas en una base de datos externa.

RF 4. La conversión debe poder automatizarse para que se realice en cualquier momento.

RF 5. El sistema no deberá realizar la conversión de forma interactiva.

## Requisitos no funcionales

RNF 1. El equipo en el que se ejecute la aplicación deberá tener instalado Java para su correcto funcionamiento.

RNF 2. El sistema deberá funcionar ininterrumpidamente y a tiempo completo (sistema 24/7).

# Atributos de Calidad

### Disponibilidad

Disponibilidad del sistema para procesar datos 24x7.

### Modificabilidad

Facilidad de cambio: nuevos ficheros fuente, nuevos formatos de representación de las preguntas, posibles cambios de base de datos.

Escalabilidad del sistema: incremento del número de usuarios y de las preguntas almacenadas.

### Seguridad

Seguridad de los datos almacenados: preguntas y respuestas, datos de los usuarios.

### Testabilidad

Facilidad para probar la fiabilidad de sistema, garantía de que las conversiones (Parser) se han realizado correctamente.

### Usabilidad

Facilidad de uso para los usuarios finales del juego, y para los encargados del mantenimiento del sistema.

# Lista de Stakeholders

|  |  |  |
| --- | --- | --- |
| Código | Stakeholder | Intereses |
| ST01 | Responsables de NoGame | Bajo coste de desarrollo, es decir, ajustar el tiempo de desarrollo del proyecto y el coste del mismo. |
| ST02 | Desarrolladores de NoGame | Baja intensidad tecnológica en el proyecto, para obtener un proceso de desarrollo controlado y predecible. |
| ST03 | Usuarios | Facilidad de uso de la aplicación. |

# Lista de Atributos de Calidad

|  |  |  |
| --- | --- | --- |
| Código | Descripción | Tipo de Atributo |
| AT001 | Disponibilidad del sistema para procesar datos 24x7. | Disponibilidad |
| AT002 | Facilidad de cambio: nuevos ficheros fuente, nuevos formatos de representación de las preguntas, posibles cambios de base de datos. | Modificabilidad |
| AT003 | Escalabilidad del sistema, por posible incremento del número de usuarios y de las preguntas almacenadas. | Modificabilidad |
| AT004 | Seguridad de los datos almacenados: preguntas y respuestas, datos de los usuarios. | Seguridad |
| AT005 | Integridad de los datos almacenados | Seguridad |
| AT006 | Facilidad para probar la fiabilidad | Testabilidad |
| AT007 | Facilidad de uso para los usuarios finales del juego, y para los encargados del mantenimiento del sistema. | Usabilidad |

# Atributos de Calidad e Interesados

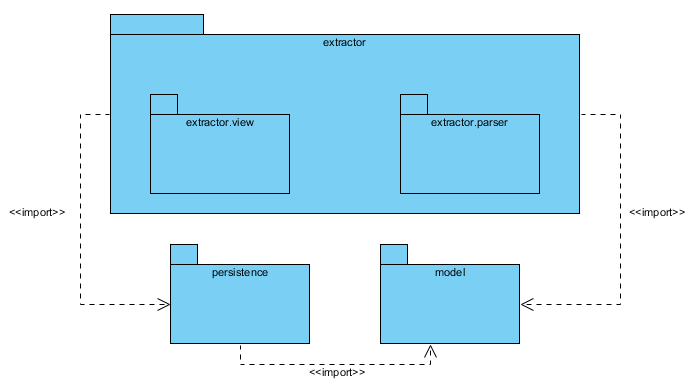
|  |  |  |  |
| --- | --- | --- | --- |
| Atributos  Vs.  Interesados | ST01 | ST02 | ST03 |
| AT001 |  |  | x |
| AT002 |  | x |  |
| AT003 | x |  |  |
| AT004 | x |  | x |
| AT005 |  | x |  |
| AT006 |  | x |  |
| AT007 |  | x | x |

# Escenarios de Calidad

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Nº | Fuente de estímulo | Estímulo | Entorno | Artefacto | Respuesta | Medición de la respuesta | Atributo de Calidad afectado |
| 1 | Desarrollador | Arranque del sistema | Desarrollo | Sistema | Tiempo para que el sistema esté funcionando razonablemente corto | Tiempo de arranque de la aplicación < 30 seg | AT001 |
| 2 | Desarrollador | Nuevo formato de preguntas | Desarrollo | Sistema | Implementar la nueva funcionalidad en un tiempo razonablemente corto | Tiempo de desarrollo < 2 días | AT002 |
| 3 | Desarrollador | Aumento del repositorio de preguntas | Desarrollo | Sistema/Base de datos | Adición de preguntas a la base de datos del sistema | Tiempo para la inclusión de las preguntas de un fichero < 2 min | AT003 |
| 4 | Usuario | Intento de acceso a datos privados del sistema | Explotación | Base de datos | Negación del sistema a realizar la operación pertinente | Nº de intrusiones y accesos no permitidos =0 | AT004 |
| 5 | Usuario | Almacenamiento de datos en el sistema | Explotación | Base de datos | Almacenamiento correcto de los datos del usuario en el sistema. | Sin pérdida de información | AT005 |
| 6 | Desarrollador | Pruebas del sistema | Desarrollo | Sistema | Ejecución de pruebas del sistema | Superación de las pruebas definidas para la aplicación | AT006 |
| 7 | Usuario | Utilización del sistema | Explotación | Sistema | El sistema será simple e intuitivo | Tiempo de aprendizaje para la utilización del sistema < 5 min | AT007 |

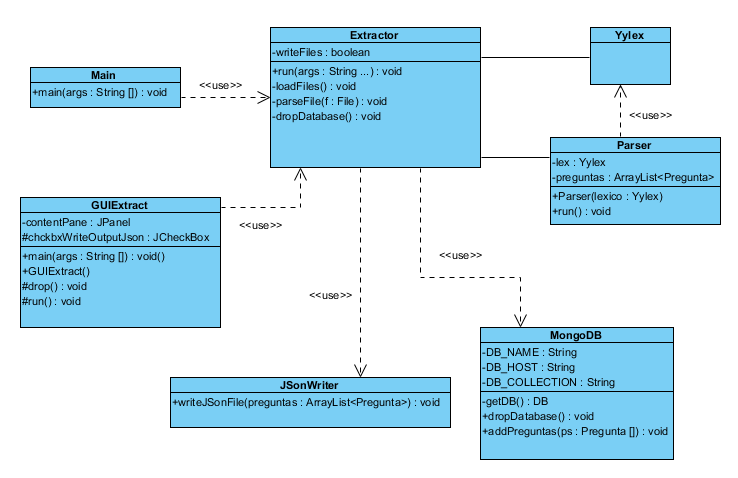
# Vistas

## Diagrama de paquetes



|  |  |
| --- | --- |
| Paquete | Descripción |
| extractor | Paquete principal de la aplicación. A su vez contiene dos paquetes:  -view  -parser |
| extractor.view | Encargado del despliegue de la aplicación. Contiene:  -Main.java  -GUIExtract.java |
| extractor.parser | Encargado de realizar la conversión de formatos. Contiene:  Contiene:  -Parser.java  -Yylex.java  -Extractor.java |
| Paquete | **Descripción** |
| persistence | Contenido relacionado con la persistencia y la base de datos.  Contiene:  -MongoDB.java  -JSonWriter.java |
| model | Contiene todas las clases del modelo de dominio:  -Pregunta.java  -Respuesta.java |

## Diagrama de clases



Además de las clases que figuran en el diagrama anterior, han de mencionarse las del modelo de dominio, que son utilizadas a lo largo de todo el proceso de extracción y conversión de las preguntas y sus respectivas respuestas. Son *Pregunta* y *Respuesta.*

Ambas clases representan a modo deatributos las características que conciernen a las preguntas que se recibirán en la aplicación y que, más tarde, serán de interés a la hora de la carga en la base de datos, como por ejemplo el identificador o la categoría de las preguntas y si cada una de sus respuestas es o no correcta.



Proyecto Trivial4a

MANUAL DE USUARIO





El propósito de este Manual es facilitar al usuario la navegación de las diferentes pantallas de la aplicación para poder usar correctamente el Trivial.

# DESCRIPCIÓN

Para la v1 de la entrega se ha implementado un Parser que nos permitirá transformar las preguntas en formato GIFT en preguntas con formato JSON, lo que permitirá su almacenamiento en una base de datos no relacional más fácilmente.

Para esta primera versión, las preguntas se almacenarán en ficheros agrupadas por categorías que mediante el parser se convertirán a JSON y se incluirán en la base de datos

# IMPLEMENTACIÓN DEL SISTEMA

### Requisitos de hardware

* Ordenador personal: Cualquier tipo de ordenador con unos requisitos mínimos para poder soportar la aplicación y base de datos es válido

### Requisitos de software

* **GitHub:** Herramienta para el control de versiones en el desarrollo de este proyecto. Se ha desarrollado la aplicación mediante cuatro ramas: tres ramas correspondientes cada una de las capas del MVC y una cuarta capa (master) donde se enlazaban los cambios realizados. Aunque no siempre se han mantenido activas todas estas ramas.
* **Sistema Operativo:** En el desarrollo del proyecto ha sido utilizado el sistema operativo Windows 7 y 8.1, pero cualquier sistema operativo compatible con los demás requisitos es también válido
* **Java:** Es el lenguaje en el que está desarrollado este proyecto, versión Java jdk 1.6 o superior, el cual utilizamos con las siguientes herramientas:
  + **Maven:** es una herramienta de software para la gestión y construcción de proyectos Java
  + **JSon:** La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX. Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos en este contexto es que es mucho más sencillo escribir un analizador sintáctico (parser) de JSON (Como nosotros hacemos en nuestro proyecto).
* **MongoDB:** es un sistema de base de datos NoSQL orientado a documentos. Utilizamos la versión 3.0.0

# INGRESANDO AL SISTEMA

Dentro de su navegador Web ingrese a la siguiente dirección: <https://www.java.com/es/download/> para proceder a la descarga de Java, necesario para el funcionamiento de la aplicación.

A continuación, desde una consola de comandos (cmd) utilizaremos el comando **“java –version”** para comprobar que java está correctamente instalado.

# INSTALACIÓN DE MongoDB

Nos descargamos el software de la dirección: <http://www.mongodb.org/downloads?_ga=1.54361128.1252863717.1426339425>.

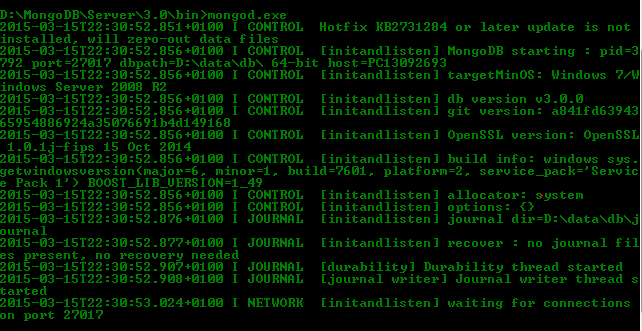
Descomprimimos el zip en la ubicación que queramos de nuestro sistema y creamos una carpeta data/db dentro del mismo directorio, aquí será donde se almacene la base de datos.

Para ejecutar MongoDB debemos usar un terminal de sistema, en él nos colocaremos en la carpeta /bin de la carpeta MongoDB donde descomprimimos el zip y ejecutamos el comando **mongod.exe**, con el que abriremos la base de datos y podremos trabajar con ella. A continuación se describe el proceso:

1.- Abrimos un terminal y nos colocamos en la carpeta donde se encuentra mongod.exeD:\ANGEL\Escritorio\Captura.PNG

2.-Ejecutamos mongod.exe para abrir la base de datos

D:\ANGEL\Escritorio\Captura.PNG  
3.- Deberíamos obtener algo así



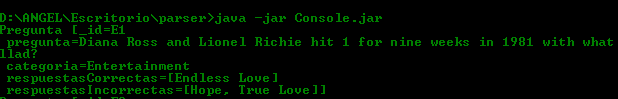
Una vez hecho esto ya podemos empezar a trabajar con el parser

# FUNCIONAMIENTO DEL PARSER

Se proporciona una carpeta en la que se incluye el parser en dos versiones diferentes, una versión para ser ejecutada en línea de comandos y otra que proporciona una pequeña interfaz gráfica.

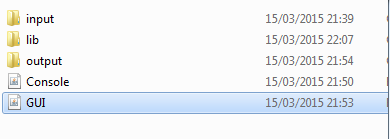
## EJECUCIÓN EN LÍNEA DE COMANDOS

En un terminal nos situamos en la carpeta del parser y ejecutamos **java -jar Console.jar**

****

Esto nos convierte las preguntas en formato .gift que debemos tener en una carpeta llamada input junto al .jar del parser en preguntas .json, y las almacena en la base de datos, es **muy importante tener abierta la base de datos.**

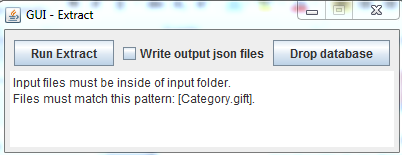
La estructura de directorios debería ser así



A continuación se detalla el contenido de cada carpeta:

* ***Input***: Contiene los ficheros con las preguntas en formato .gift, si queremos añadir nuevas preguntas debemos incluirlas en esta carpeta
* ***Lib:*** Bibliotecas necesarias para la utilización del parser
* ***Output***: Contiene los ficheros con las preguntas .gift convertidas a formato .json

## EJECUCIÓN CON INTERFAZ GRÁFICA

Para ejecutar la interfaz gráfica del parser basta con hacer doble click sobre GUI.jar, lo podemos encontrar junto al otro parser, esto nos mostrará una pequeña ventana con dos botones que nos permitirá realizar la conversión de las preguntas y almacenarlas en la base de datos o borrar la base de datos. 

Al igual que la opción anterior **es muy importante tener abierta la base de datos**, podemos marcar la opción de escribir los ficheros JSON para que se almacenen en la carpeta output para su posterior revisión.

# INTRODUCCIÓN DE NUEVAS PREGUNTAS

Para la introducción de nuevas preguntas se usará el formato GIFT, para ello se creará un nuevo fichero de texto con las preguntas en este formato y se incorporará a los demás bancos de preguntas dentro de la carpeta input que se encuentra junto a los .jar del parser, o bien se modificará un banco ya existente y se incluirán las nuevas preguntas en este formato.