PRÀCTICA S1

MEMÒRIA

LS GALLOS

Noms complets:Laia Abad i Silvia Miralles
Logins:laia.abad i silvia.miralles

ÍNDEX

- Resum de l'enunciat…… 2
- Diagrama UML…….. 3
- Explicació del diagrama…….. 4
- Opcionals i extres……..…….. 5
- Mètode de proves utilitzat…….. 6
- Dedicació en hores……... 7
- Conclusions…….. 8
- Bibliografia……..…... 9

RESUM

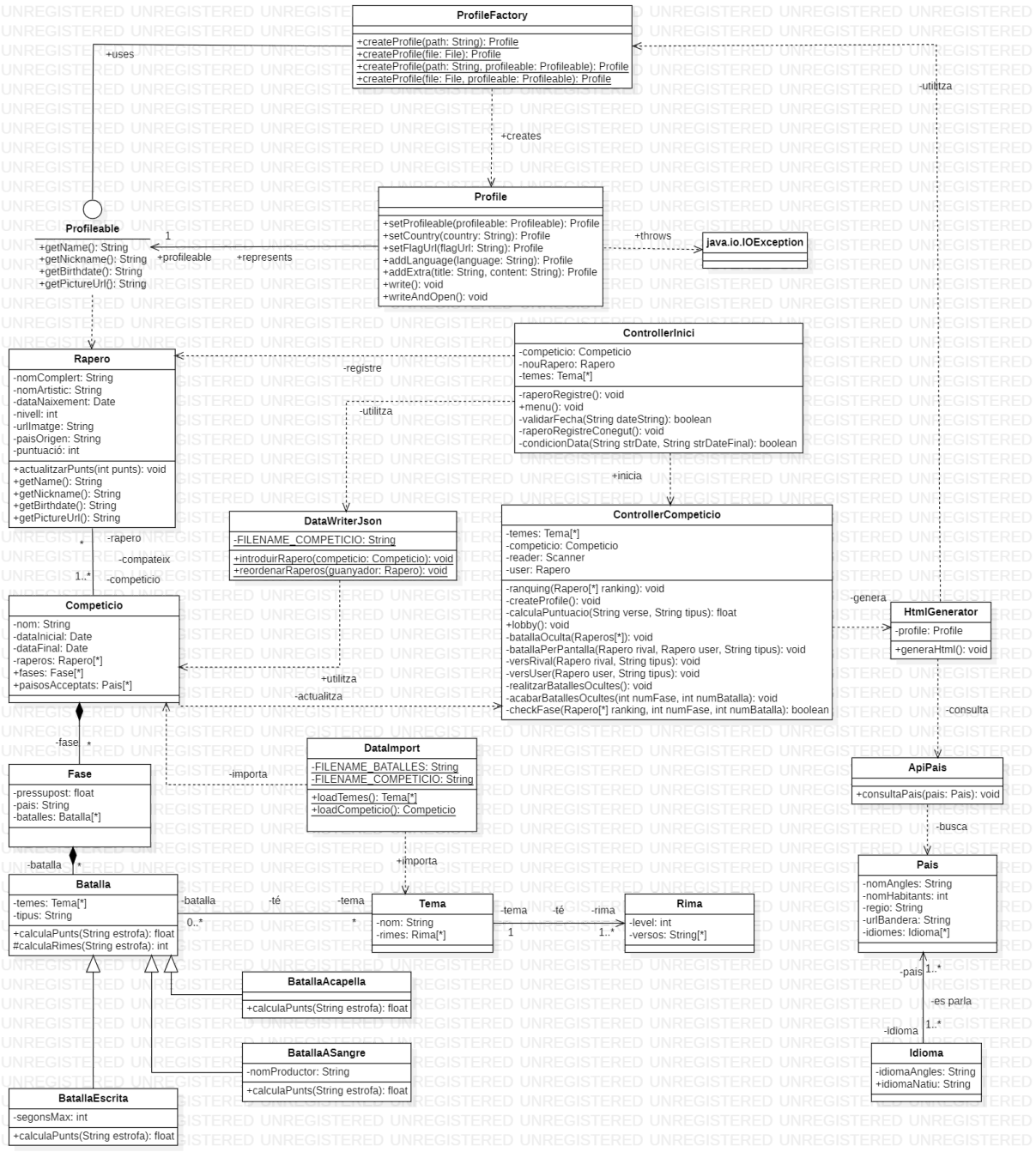
El projecte que hem desarrollat consisteix en un programa sobre batalles de galls, es realitzarà una única competicio. Primer de tot hem de configurar la competició amb la següent informació: nom, data d'inici, data de fi i llista de participants ja registrats, en cas de voler introduir més participants s'han de registrar manualment amb nom complet, nom
artístic, data de naixement, nivell d'expertesa, URL amb la seva foto i el nom del seu país
d'origen, i quan s'inici la competició no es podran registrar més participants.

La competició esta dividida en varies fases, cada una té dues batalles de tres tipus (escrita, a sangre oi acapella) i cada batalla tindrà assignada un conjunt de temes perquè els raperos puguin rapejar. Cada tipus de batalla té una condicions diferents: la batalla escrita tenen un temps límit, les batalles a sangre hauran d'indicar el nom del productor de les bases.

Durant la competició a mida que avança els participants van obtenint punts de les seves batalles que en funció de quin tipus de batalla sigui se'ls hi assignen uns o uns altres en funció de com ho han fet, ademes es pot consultar el ranking per saber en quina posició van.

Durant la competició o al acabar es pot sol·licitar el perfil de qualsevol participant introduint el nom del rapero, si tot va bé obtindrem el perfil del rapero amb el seu nom, el nom artístic, la seva fotografia, la bandera del seu país, la data de naixement i els idiomes, juntament amb informació addicional com els seus punts i la seva posició a la competició.

DIAGRAMA DE CLASSES UML DE DISSENY



EXPLICACIÓ DEL DIAGRAMA DE CLASSES QUE HEU DISSENYAT

La classe rapero esta relacionada amb una associació a competició ja que els raperos estan a la competició durant un temps determinat i no de manera esporàdica. El tema s'utilitza a la batalla mentre aquesta està passant, així que té el mateix tipus de relació, i el mateix amb la rima, que s'utilitza mentre s'utilitza un tema. El idioma, té una relació més duradora amb país que els mencionats anteriorment, però com no és part, ni herència del país, té el mateix tipus de relació. Les relacions entre Profileable, Profile i ProfileFactory estaven expressades a l'enunciat, però entenem que per a fer servir Profileable necessitem un Profile i per a fer servir ProfileFactory necessitem un Profileable.

Una competició no té sentit sense les seves fases, igual que una fase no té sentit sense les seves batalles, així que aquestes són composicions.

Els diferents tipus de batalla són clarament herències de batalla, ja que son tipus de batalla.

La resta de relacions que tenim son relacions esporàdiques, així que son només dependències.

Hem canviat un parell de coses respecte com teníem l'UML anteriorment, ja que vam tenir problemes al implementar l'altre, i algunes relacions innecessàries, per exemple país amb rapero, que vam reemplaçar amb un string a rapero que conté el país, i el mateix amb fase. Després de fer aquests canvis, no vam tenir més problemes a l'hora d'implementar-ho.

OPCIONALS I EXTRES

Durant la realització de les practica nosaltres hem realitzat la competició contemplant alguns aspectes que poder no son els usuals. En el cas de les batalles hem considerat que a cada fase tenia dues batalles en les quals hi ha dos temes diferent escollits de manera aleatoria.

En el cas de les rimes com depenent del tipus de batalla que fos es tenia que fer un càlcul diferent de la puntuació, com es pot veure a continuació:

$$Puntuacio\ Sangre = \frac{\pi R^2}{4} \quad (1)$$

$$Puntuacio\ Acapella = \frac{6\sqrt{R}+3}{2} \quad (2)$$

$$Puntuacio\ Escrita = \frac{16+2+128+64+256+4+32+512+1024+R}{1024+128+4+64+16+256+R+2+32+512} + 3R \quad (3)$$

Nosaltres hem considerat que com les rimes es miraven en funció del final de cada vers, és tenia que seguir un patró:
vers 1,
vers 2,
vers3,
vers 4.

El patró tal com es pot veure consisteix en què els tres primers versos tenen que anar seguits de una coma i l'últim vers d'un punt. Hem creat aquest patró ja que els exemples també es realitzava.

MÈTODE DE PROVES UTILITZAT

Per comprovar que les dades del JSON i l'API estaven ben introduïts, hem utilitzat el debugger posant stop points on es guardaven les dades, i anavem mirant que les dades introduïdes eren correctes comparades amb el document i que el nombre de dades també ho era.

Per a comprovar el correcte funcionament de la resta del programa, hem anat provant tots els casos possibles després d'executar el programa, tots els possibles errors que haviem de controlar i tots els casos bons.

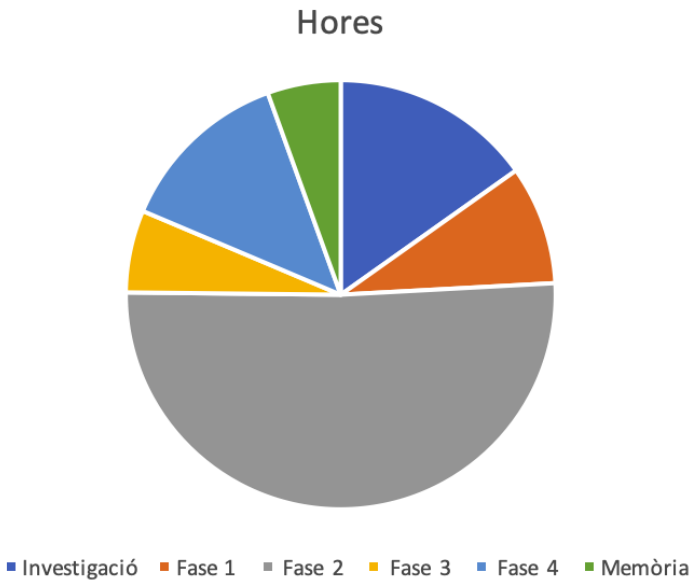
DEDICACIÓ EN HORES

En quant als costos del projecte, hi ha diverses punts a valorar:

1. Investigació: temps de cerca online, revisió d'exercicis anteriors fets a classe.
2. Fase 1: diagrama inicial amb les classes, atributs, funcionalitats, relacions. Ademes de revisar la informació del estudy per acabar de revisar que tot estigues correcte i que les relacions que posavem eren les adequades.
3. Fase 2: implementació del disseny de la fase 1 en Java
4. Fase 3: redissenyar la pràctica a partir de noves funcionalitats ademés d'incorporar blocs funcionals dissenyats per un tercer
5. Fase 4: implementació del disseny de la fase 3 en java, per crear el perfil de cada participant amb HTML.
6. Memòria: registre del treball realitzat durant tot el primer semestre.

Evidentment és difícil estimar el temps que hem dedicat a cada apartar, ja que algunes vegades ens hem encallat en algun punt i ens ha requerit més temps de lo usual, ademes del factor que el treball ha sigut de forma remota.

	HORES
INVESTIGACIÓ	21
FASE 1	16
FASE 2	74
FASE 3	11
FASE 4	16
MEMÒRIA	8



CONCLUSIONS

Gràcies a aquest projecte hem après la importància de llegir i entendre bé l'enunciat per tal de fer a nivell de disseny el que et demana adequadament i el plasmar els coneixements apresos a classe per tal de realitzar-ho correctament. És molt important tenir clar el disseny abans de posar-se amb el codi per tal de que el codi sigui el més òptim possible i que compleixi els quatre principis del disseny orientat a objectes, encapsulació, abstracció, herència i polimorfisme.

També hem après a fer lectura i escriptura de fitxers JSON i fer ús de APIs, a més d'utilitzar llibreries externes.

D'altre banda, hem hagut d'aplicar els coneixements apresos a classe com les herències, mètodes i classes estàtiques, entre d'altres.

Sobretot, aquest treball ens ha obligat a fer ús dels nostres recursos per tal de solucionar els problemes que hem tingut, fent búsquedes a internet i utilitzant la documentació de l'estudy.

BIBLIOGRAFIA

ENDRULLIS, S. 2013. How to sort an attribute of an object using Collections. *Stack Overflow* [en línea]. Disponible en: https://stackoverflow.com/questions/5932720/how-to-sort-an-attribute-of-an-object-using-collections.

GUPTA, L. [sin fecha]. How to swap two elements in ArrayList in Java. *HowToDoInJava* [en línea]. Disponible en: https://howtodoinjava.com/java/collections/arraylist/swap-two-elements-arraylist/.

GUPTA, M. 2018. Collections.shuffle() in Java with Examples. *GeeksforGeeks* [en línea]. Disponible en: https://www.geeksforgeeks.org/collections-shuffle-java-examples/.

MP 2009. Shrinking an ArrayList to a new size. *Stack Overflow* [en línea]. Disponible en: https://stackoverflow.com/questions/1184636/shrinking-an-arraylist-to-a-new-size.

PENG, Y. [sin fecha]. Java String Camel Case Format toCamelCase (String str). *Java2s.com* [en línea]. Disponible en: http://www.java2s.com/example/java-utility-method/string-camel-case-format/tocamelcase-string-str-d986d.html.

Random (Java Platform SE 8). *Docs.oracle.com* [en línea] [sin fecha].

ROCHA, D. 2016. randomly remove element from array java. *Stack Overflow* [en línea]. Disponible en: https://stackoverflow.com/questions/35254019/randomly-remove-element-from-array-java.

SINGH, C. 2014. How to sort ArrayList in descending order in Java. *beginnersbook.com* [en línea]. Disponible en: https://beginnersbook.com/2013/12/sort-arraylist-in-descending-order-in-java/.