Vocabulari

- **Nom de la classe:** Vocabulari
- **Descripció:** Classe que conté el llistat de totes les paraules que els documents tenen.
- **Cardinalitat:** 1 instància
- **Descripció dels atributs:**
    - **arrel:** Arbre que conté totes les paraules dels documents.
- **Descripció dels mètodes:**
    - **Vocabulari()** Constructora per defecte de Vocabulari.
    - **decrementarOcurrencia(Paraula p)** Decrementa en una unitat el nombre d'ocurrencies de la paraula p.
    - **esborrarParaula(Paraula p)** Esborra una paraula del diccionari.
    - **Paraula inserirObtenirParaula(String s)** Obtenir la classe Paraula que correspon a la seqüència s, la crea si no existeix aquesta.
    - **Paraula obtenirParaula(String s)** Obtenir la classe Paraula que correspon a la seqüència s.
- **Descripció de les relacions:**
    - Relació d'associació amb la classe "Paraula": indica quines paraules formen part del vocabulari.

Paraula

- **Nom de la classe:** Paraula
- **Descripció:** Paraula que pot ser continguda a algun document.
- **Cardinalitat:** 0 o més instàncies
- **Descripció dels atributs:**
    - **index:** Identificador de la paraula.
    - **mot:** La pròpia paraula en format String.
    - **ocurrència:** Número de vegades que la paraula apareix entre tots els continguts.
    - **static proxim_index:** Pròxim identificador a asignar.
- **Descripció dels mètodes:**
    - **Paraula(String p)** Constructora per defecte de paraula.
    - **decrementarOcurrencia()** Decrementa en una unitat el nombre d'ocurrencies.
    - **int getId()** Retorna l'índex de la paraula.
    - **int getOcurrencia()** Retorna el nombre d'ocurrencies total.
    - **String getParaula()** Retorna la paraula representada per la classe.
    - **incrementarOcurrencia()** Incrementa en una unitat el nombre d'ocurrencies.
- **Descripció de les relacions:**
    - Relació d'associació amb la classe "Vocabulari": indica el vocabulari on aquesta paraula està inclosa.

- Relació d'associació amb la classe "Contingut": indica els continguts on apareix la paraula.

Contingut

- **Nom de la classe:** Contingut
- **Descripció:** Classe que conté el contingut d'un document.
- **Cardinalitat:** 1 instància per a cada document
- **Descripció dels atributs:**
  - **plaintext:** Emmagatxema tot el text del contingut en un sol String.
  - **phrases:** Array de Frases que formen el contingut.
  - **words:** HashMap que mapeja cada índex de paraula amb el seu nombre d'ocurrències al contingut.
  - **n_paraules:** Nombre de paraules total al contingut.
- **Descripció dels mètodes:**
  - **Contingut (String plaintext, Frase [ ] phrases):**
    Donat el plaintext i l'array de Frases, construeix el Contingut
  - **double getTFofWord (int index)**
    Donat l'índex d'una paraula, retorna el Term Frequency d'aquesta al contingut (o -1 si no existeix la paraula al contingut).
  - **MAP<Integer, Double> getTF ()**
    Retorna un HashMap que mapeja cada índex de paraula amb el seu TF al contingut.
  - **boolean conteSequencia (String seq)**
    Retorna *true* si, i només si, la sequüència *seq* es troba en el text del contingut.
- **Descripció de les relacions:**

ConsultaSemblant

- **Nom de la classe:** ConsultaSemblant
- **Descripció:** Classe que implementa la consulta per semblança.
- **Cardinalitat:** 0 instàncies. La classe només té un mètode static.
- **Descripció dels atributs o relacions: -**
- **Descripció dels Mètodes:**
    - **LIST< PAIR < Double, Document > > executeQuery (Llibreria lib, Document doc, int k, int mode)**
      Únic mètode static de la classe ConsultaSemblant que implementa l'execució de la Query donada una llibreria (és a dir, un conjunt de documents), un document *doc*, un enter *k*, i un mètode (0 ó 1) per a assignar pesos a les paraules.
      El mètode retorna una llista de parells, on cada parell tenim un document i el seu cosinus amb el document *doc*. La llista són els *k* documents més semblants a *doc*, on definim el concepte "semblança" com el cosinus entre els angles de dos vectors de paraules i pesos TF-IDF (mode 0) ó pesos Nombre d'Ocurrències (mode 1).

DocumentCtrl

- **Nom de la classe:** DocumentCtrl
- **Descripció:** Classe que conté mètodes relacionats amb la gestió de documents; com poden ser crear, desar, eliminar, modificar, o ordenar llistes de documents.
- **Cardinalitat:** 1 instància
- **Descripció dels atributs:**
    - **Vocabulari vocab:** El vocabulari de paraules creades
    - **Llibreria lib:** El conjunt de documents de l'aplicació
    - **ConsultaData CD:** La consultora per dates
    - **ConsultaTitol CT:** La consultora per títol
    - **ConsultaPreferit CP:** La consultora dels documents preferits
    - **ConsultaAutors CA:** La consultora dels autors que mai han existit.
    - **(4X) Comparator<Document> documentComparator**
      Són quatre comparadors de Documents que serveixen per a ordenar-los segons dates, preferits, autors i títols.
- **Descripció dels mètodes:**
    - **LIST<Document> sortDocuments(LIST<Document>, int type)**
      Mètode que ordena i retorna una ArrayList de Document segons un criteri, especificat per l'enter *type*. (0 = per data de creació; 1 = preferits primer; 2 = autor alfabèticament; 3 = títol alfabèticament).
    - **LIST<Document> sortDocuments(SET<Document>, int type)**
      Mateix mètode que l'anterior, però admet com a paràmetre un conjunt de documents en comptes d'una llista.
    - **DocumentCtrl (…)**

Constructora per defecte, simplement assigna els sis atributs.
    - **LIST<String> decomposeWords(String frase)**
      Mètode que descomposa una String en paraules. És a dir, separa cadenes de lletres per espais ó signes especials i llavors insereix aquestes paraules a una llista a retornar.
    - **PAIR<Document, Boolean> getDocument (nomAutor, nomTitol)**
      Mètode que retorna el Document identificat per l'String nomAutor i nomTitol. Si existeix el booleà del Pair que retorna serà *true*, altrament *fals*. En cas que sigui *fals* el document retornar serà *null*.
    - **void crearDocument (LocalDate dia, Boolean isFav), eliminarDocument (), modificarDocument()**
      Crea, elimina o modifica un document interaccionant amb l'usuari
    - **void eliminarDocument (Document d)**
      Elimina el document *d* de la llibreria
    - **void togglePreferit (Document d)**
      Fa un toggle del booleà preferit del document *d*

Expressio Booleana

- **Nom de la classe:** ExpressioBooleana
- **Descripció:** Expressió formada per combinacions de instancies d'operadors, paraules o expressions més breus. Les expressions requereixen una paraula o una altra expressió.
- **Cardinalitat:** 0 o més instàncies
- **Descripció dels atributs:**
    - **String Cos:** Guarda l'expressió booleana tal qual arriba a l'instancia de la classe.
    - **String Nom:** Guarda el nom associat a l'expressió booleana. Pot ser null.
    - **Node Root:** Node arrel de l'arbre d'expressió generat per la classe ExpressionTree
    - **Set<Document> Resultat:** Conjunt de instancies de la classe Document que només conté els documents que compleixen l'expressió booleana.
    -
- **Descripció dels mètodes:**
    - **Set<Document> getResultat(Llibreria l):** Retorna els Documents que compleixen l'expressió booleana de Cos.
    - **ExpressioBooleana():** Creadora per defecte.

- **ExpressioBooleana(String cos):** Creadora de la classe donada l'expressió.
- **ExpressioBooleana(String nom, String cos):** Creadora de la classe donats el nom i l'expressió.
- **ExpressioBooleana ExpressioBooleana (String cos):** Torna una instancia d'ExpressioBooleana donada una expressió.
- **Set<Docment> ConsultaBooleana (Llibreria l, Node consulta):** Torna un conjunt de documents que compleixen l'expressió booleana donats una llibreria i un node arrel.
- **Boolean ConteParaula (String query, Document doc):** Torna cert si el document doc conté la paraula query
- **Set<Document>ConsultaBooleanaRec (Node consulta, Set<Document> tots):** Torna recursivament un conjunt de documents que contenen els documents que compleixen la condició del node donats un node i un conjunt de documents.

ExpressioBooleanaCtrl

- **Nom de la classe:** ExpressioBooleanaCtrl
- **Descripció:** Classe controlador que conté mètodes per a la creació, ús, eliminació i cerca d'expressions booleanes
- **Cardinalitat:** 1
- **Descripció dels atributs:**
  - **SetDeExpressions<String, ExpressioBooleana>:** Es un HashMap que conté totes les expressions booleanes i fa servir el seu nom com a identificador.

- **Descripció dels metodes:**
  - **Boolean isEmpty():** Torna cert si no hi ha cap expressió booleana a SetDeExpressions.
  - **ExpressioBooleanaCtrl():** Creadora per defecte.
  - **Set<String> GetNomExpressions():** Torna un conjunt amb tots els identificadors de SetDeExpressions.
  - **ExpressioBooleana GetExpressioBooleana (String nom):** Torna l'instancia d'ExpressioBooleana corresponent al parametre.
  - **ExpressioBooleana ExpressioBooleanaTemporal (String cos):** Torna una nova instància d'ExpressioBooleana sense nom donat el seu cos.
  - **DeleteExpressioBooleana(String nom):** Elimina l'expressió booleana del conjunt donat el seu nom.

    - **SetExpressioBooleana(String nom, String cos):** Substitueix una expressió booleana donats el seu nom i el nou cos. La crea si aquesta no existia previament.
- **Descripció de les relacions:**
    - Relació amb la classe ExpressioBooleana: Hi ha diverses instancies d'ExpressioBooleana totes associades a la mateixa instancia d'ExpressioBooleanaCtrl

ConsultaAutors

- **Nom de la classe:** ConsultaAutors
- **Descripció:** Representa el tipus de consulta per Autor, descrita a l'apartat 2 de l'Enunciat.
- **Cardinalitat:** 1
- **Descripció dels Atributs:**
    - **TernaryTreeAutor autors:** Arbre ternary emprat per guardar els autors. Per cada autor es guarda el seu nom sencer, el seu nom sense la primera paraula (es a dir només cognoms), el seu nom sense la primera i la segona paraula (segón cognom i de més, si tingués) fins a haver inserit només la última paraula. Això es fa per tal de poder trobar prefixos a qualsevol dels cognoms d'un autor, a més de en el nom.
- **Descripció dels mètodes**
    - **addAutor(Frase autor)**
      Insereix l'autor en l'arbre *autors* tal i com s'ha comentat a la descripció de l'atribut.
    - **Set<Frase> donaAutors(String prefix)**
      Retorna un set on estan els noms dels autors que compleixen el prefix, passat per paràmetre, en forma Frase. El set resultant és el resultat d'una crida a la funció *obtenirAutors(String=prefix, int=0)* de la classe *TernaryTreeAutor*.

- **Descripció de les Relacions: -**

ConsultaTítol

- **Nom de la classe:** ConsultaTítol
- **Descripció:** Representa el tipus de consulta per Títol, descrita a l'apartat 1 de l'Enunciat.
- **Cardinalitat:** 1
- **Descripció dels Atributs:**
    - **coleccions:** Conjunt de documents, una per a cada autor.
- **Descripció dels mètodes:**

    - **ConsultaTitol()** Constructora per defecte.
    - **afegirDocument(Document d)** Afegeix el document al conjunt corresponent.
    - **eliminarDocument(Document d)** Elimina el document del conjunt corresponent.
    - **Set<Document> getDocAutor(Frase autor)** Retorna el conjunt de documents corresponent a l'autor.
- **Descripció de les Relacions: -**

Frase

- **Nom de la classe:** Frase
- **Descripció:** Classe que representa una frase i guarda d'aquesta les seves paraules.
- **Cardinalitat:** 0 o més instàncies.
- **Descripció dels atributs:**
    - **Paraula[] Oracio:** Conjunt d'instàncies de la classe Paraula que formen la Frase.
    - **int n_paraules:** Número de paraules de la frase.
    - **String text:** Contingut de la frase original en format String.
- **Descripció dels mètodes**
    - **Frase(ArrayList<Paraula> words, String frase)**
      Constructora d'una frase donada una ArrayList de les paraules que formen part de la frase i un string (per l'atribut text).
    - **Frase(Paraula[] words, String frase)**
      Constructora d'una frase similar a l'anterior però ara les paraules venen donades per Paraula[].
    - **Frase(String frase)**
      Constructora d'una frase donat només un string. Mitjançant les funcions String to paraula, decompose i isPuntuacio s'aconsegueixen les paraules.
    - **boolean conteSequencia(String[] Paraules)**
      Retorna *true* si es troben les paraules concatenades (és a dir una darrere de l'altre en ordre i sense interrupcions), passades per atribut, en la frase.
    - **boolean conteCaracters(String lletres)**
      Retorna *true* si la frase conté els caràcters que formen el String *lletres* en ordre i concatenats.
    - **boolean conteParaula(String paraula)**
      Retorna *true* si la frase conté la paraula passada per paràmetre.
    - **HashMap<Integer, Integer> donaWords()**

Retorna un HashMap on, per cada paraula diferent de la Frase, la *key* és el *Id* de la paraula i el *value* és el número d'aparicions d'aquesta en la frase.

- **boolean isPuntuacio(char c)**
  Retorna *true* si el caràcter c és igual a una sèrie de caràcters que ens interesa identificar (els quals en aparèixer marquen el final d'una paraula).
- **ArrayList<String> decompose(String frase)**
  Retorna, donat una Frase en format String, un ArrayList de les paraules d'aquesta frase. Es va creant un String llegint els caràcters que són lletres de *frase* que serà una paraula a insertar; quan arribem a un signe de puntuació (isPuntuacio) sabem que la paraula ha acabat i podem inserir el String a l'ArrayList.
- **ArrayList<Paraula> stringToParaules(String frase)**
  Donat un String obté primer un ArrayList<String> de les paraules gràcies a decompose i després retorna un ArrayList<Paraula> on per cada string de l'ArrayList retornat per decompose es crea una instància de la classe Paraula i s'afegeix al ArrayList resultat.

- **Descripció de les relacions:**
  - Relació d'associació amb la classe "Paraula": indica les paraules per les quals la frase està formada.
  - Una instància de la classe Frase pot ser títol o autor de diversos documents.
  - Un contingut està format de diverses frases.

Document

- **Nom de la classe:** Document
- **Descripció:** Document donat d'alta.
- **Cardinalitat:** 0 o més instàncies.
- **Descripció dels atributs:**
  - **LocalDate date:** Indica la data de creació del document.
  - **path:** Path relatiu on es troba el fitxer del document amb el contingut en qualsevol dels tres formats acceptats.(txt, xml, prop)
  - **boolean isFav:** Indica si el document ha estat marcat com a preferit
  - **Frase title:** Instància de la classe Frase que fa referència al títol del document.
  - **Frase author:** Instància de la classe Frase que fa referència a l'autor del document.
  - **Contingut cont:** Instància de la classe Contingut que té tot el text del contingut del document
- **Descripció dels Mètodes:**

- **Document (autor, titol, isFav, path, date, cont)**
  Constructora d'un Document que inicialitza els atributs.
- **double getTFofWord(int index)**
  Donat l'índex d'una paraula, retorna el Term Frequency d'aquesta al contingut del document (o -1 si no existeix la paraula al contingut).
- **MAP<Integer, Double> getTF ()**
  Retorna un HashMap que mapeja cada índex de paraula amb el seu TF al contingut.
- **boolean conteSequencia (String seq)**
  Retorna *true* si, i només si, la sequüència *seq* es troba en el text del contingut, en la String de l'Autor, o en el títol.

- **Descripció de les relacions:**
  - Document té un únic autor i títol; tots dos són instàncies de la classe Frase.
  - Document té un Contingut

ConsultaData

- **Nom de la classe:** ConsultaData
- **Descripció:** Representa el tipus de consulta per Data, una cerca extra basada en la consulta de Documents segons la seva data (atribut de la classe Document).
- **Cardinalitat:** 1.
- **Descripció dels atributs:**
  - **LocalDate anterior:** Indica la data a partir de la qual es vol buscar.
  - **LocalDate posterior:** Indica la data màxima per trobar un document.
  - **ArrayList<Document> docs:** Conjunt de tots els documents donats d'alta del sistema ordenats per la seva data.
  - **int n_docs:** número de documents a docs.
- **Descripció dels Mètodes:**
  - **ConsultaData()**
    Crea una consulta sense documents i amb anterior = data mínima permesa per LocalDate i posterior = data màxima permesa.
  - **ConsultaData(LocalDate ant, LocalDate post)**
    Crea una consulta sense documents amb anterior = ant i posterior = post.
  - **addDoc(Document D)**
    Afegeix el document D a docs mantenint l'ordre per data. En cas d'empatar amb algún / alguns altres Documents s'insereix per ordre alfabètic (dintre del grup amb els qui empata en data). S'incrementa l'atribut *n_docs*.
  - **deleteDoc(Document D)**
    S'esborra el document D de docs i es decrementa el *n_docs*.
  - **ArrayList<Document> consultaAnterior(LocalDate max)**

Retorna els documents de *docs* amb data anterior o igual a max.
    - **ArrayList<Document> consultaPosterior(LocalDate min)**
      Retorna els documents de *docs* amb data posterior o igual a min.
    - **ArrayList<Document> consulta(LocalDate ant, LocalDate pos)**
      Es retorna el resultat de consulta() assignant prèviament anterior = ant i posterior.
    - **ArrayList<Document> consulta()**
      Retorna els documents amb data <= posterior i amb data >= anterior. Es fa mitjançant dues cerques lineals, una començant pel final per acotar superiorment i una començant per l'inici per acotar inferiorment *docs*, on s'aconsegueixen dos iteradors que marquen els indexos on l'interval de documents requerit comença i acaba. Es fa un recorregut lineal i s'insereix cada document desde l'index inicial fins al final al ArrayList resultat i es retorna. En el cas de ser un interval sense documents es retorna un ArrayList empty.

- **Descripció de les relacions:-**

ConsultaPreferit

- **Nom de la classe:** ConsultaPreferit
- **Descripció:** Consulta documents marcats com a preferits.
- **Cardinalitat:** 1
- **Descripció dels atributs:**
    - **preferits:** Conjunt de documents marcats com a preferits.
- **Descripció dels mètodes:**
    - **ConsultaPreferit()** Constructora per defecte.
    - **afegirDocument(Document d)** Afegeix el document al conjunt.
    - **Set<Document> getDocPreferit()** Retorna el conjunt de documents preferits.
- **Descripció de les relacions:** -

ConsultaRellevancia

- **Nom de la classe:** ConsultaRellevancia
- **Descripció:** Representa el tipus de consulta per Títol, descrita a l'apartat 4.3 de l'Enunciat.
- **Cardinalitat:** 1.
- **Descripció dels atributs:**
    - **Integer k:** número de documents que es volen obtenir.
    - **Paraula[] query:** conjunt de paraules que formen la *query*.
    - **Llibreria docs:** conjunt dels *k* documents més rellevants.
- **Descripció dels Mètodes:**
  ConsultaRellevancia(Integer k_docs, Paraula[] words, String frase, Integer mode, Llibreria documents)

Constructora on es retorna una Consulta Rellevancia amb *k*= k_docs, *query*= words; docs s'obtè depenent del mètode (seleccionat per *mode*) que volguem utilitzar (recordem que aquesta consulta ofereix dues formes d'obtenir els k documents més rellevants a la query). El primer mètode suma les aparicions de les paraules de la query en cada document i guarda, en docs, els k documents amb la suma més gran (el primer el que té la suma màxima). El segón mètode es basa en crear un document trivial, on títol,autor i contingut és igual a la query (pasada per paràmetre i en format String en *frase*), i fer una cerca per Semblança utilitzant aquest document com el de referència, amb *k*= k_docs i Llibreria com a conjunt de documents a partir dels quals volem obtenir els k més rellevants.

Llibreria

- **Nom de la classe:** Llibreria
- **Descripció:** La llibreria representa un conjunt de documents.
- **Cardinalitat:** 1 princiapl (més si es retorna una Llibreria en alguna consulta)
- **Descripció dels atributs:**
    - **LIST < PAIR < Document, MAP < Integer, Double > > > docs0**
      Llista de documents, on cada document té un HashMap que serveix com a representació del vector sparsed de paraules amb pesos TF-IDF
    - **LIST < PAIR < Document, MAP < Integer, Double > > > docs1**
      Llista de documents, on cada document té un HashMap que serveix com a representació del vector sparsed de paraules amb pesos el nombre d'ocurrències de cada paraula al document.
    - **MAP < Integer, Integer > word_ocurrences**
      HashMap que assigna cada índex de paraula amb el seu nombre d'ocurrències totals a tota la llibreria. Necessàri pel càlcul de l'IDF eficient.
    - **int nDocs:** Nombre de documents en total a la llibreria
- **Descripció dels Mètodes:**
    - **Llibreria ()**
      Constructora d'una llibreria, inicialment buida.
    - **double computeCosinus(Document d1, Document d1, int mode)**
      Donats dos documents, calcula el cosinus entre els dos vectors de paraules que representen cada document. Els pesos de les paraules són o bé TF-IDF (si *mode = 0*) o el nombre d'ocurrències (mode = 1).
    - **void addDocument(Document d)**
      Afegeix el document *d* a la llibreria i actualitza tots els índex IDF i ocurrències de les paraules.
    - **void deleteDocument(Document d)**
      Elimina el document *d* de la llibreria i actualitza tots els índexs IDF i ocurrències de les paraules.

    - **PAIR < Document, Boolean > getDocument (String aut, String tit)** Mètode que retorna el Document identificat per l'String *aut* i *tit*. Si existeix el booleà del Pair que retorna serà *true*, altrament *fals*. En cas que sigui *fals* el document retornar serà *null*.
    - **Llibreria getPreferits()**: Retorna una Llibreria dels documents preferits
    - **Document getIessim(int i):** Retorna l'i-éssim document afegit a la Llibreria.
    - **SET < Document > getSetDocuments ( )** Retorna tots els documents de la llibreria en un conjunt.

- **Descripció de les relacions:**
    - Llibreria té referències als Documents que conté

ConsultaAvançada

- **Nom de la classe:** ConsultaAvancada
- **Descripció:** Consulta documents que conté la seqüència indicada.
- **Cardinalitat:** Qualsevol
- **Descripció dels atributs:**
- **Descripció dels mètodes:**
    - **ConsultaAvancada()** Constructora per defecte de ConsultaAvançada.
    - **static Set<Document> obtenirDocuments(Llibreria l, String s)** Retorna el conjunt de documents dins de l que contenen la seqüència s.
- **Descripció de les relacions:** -

TernaryTree

- **Nom de la classe:** TernaryTree
- **Descripció:** Estructura de dades per contenir el diccionari de paraules.
- **Cardinalitat:** 1
- **Descripció dels atributs:**
    - **esquerra:** Node fill de l'esquerra.
    - **dreta:** Node fill de la dreta.
    - **centre:** Node fill del centre.
    - **paraula:** Paraula que el node fa referència.
    - **lletra:** Lletra que conté el node.
- **Descripció dels mètodes:**
    - **TernaryTree()** Constructora per defecte de TernaryTree.
    - **TernaryTree(char c)** Constructora per als nodes fills.
    - **esborrarParaula(String s, int i, TernaryTree esborrable, int dir)** Esborra la paraula corresponent a s juntament amb els nodes innecessaris.
    - **Paraula inserirObtenirParaula(String s, int i)** Obtenir la classe Paraula que correspon a la seqüència s, la crea si no existeix aquesta, també actualitza el número d'ocurrències si aquesta ja existia.

    - **Paraula obtenirParaula(String s, int i)** Obtenir la classe Paraula que correspon a la seqüència s.
- **Descripció de les relacions:** -

TernaryTreeAutor

- **Nom de la classe:** TernaryTreeAutor
- **Descripció:** Estructura de dades per contenir l'arbre de prefixos dels autors.
- **Cardinalitat:** 1
- **Descripció dels atributs:**
    - **esquerra:** Node fill de l'esquerra.
    - **dreta:** Node fill de la dreta.
    - **centre:** Node fill del centre.
    - **autors:** Conjunt d'autors que contenen el prefix d'aquell node.
    - **lletra:** Lletra que conté el node.
- **Descripció dels mètodes:**
    - **TernaryTreeAutor()** Constructora per defecte de TernaryTreeAutor.
    - **TernaryTreeAutor(char c)** Constructora per als nodes fills.
    - **inserirAutor(Frase a, String s, int i)** Insereix l'autor a l'arbre.
    - **Set<Frase> obtenirAutors(String s, int i)** Obtenir els conjunt d'autors que conten el prefix s.
- **Descripció de les relacions:** -