Explicación de clases

Capa Domini

Clase Document

- Descripción: documento que gestiona el programa Seshat
- Cardinalidad: documento pertenece a un único autor y la cardinalidad es indeterminada para la clase Document_Set.
- Atributos:
    - ·title: el título que identifica el documento
    - ·id: identificador del documento
    - ·plainText: el contenido del documento
- Relaciones:
    - ·Relación de asociación con autor para saber saber quién es el autor del documento.
    - ·Relación de agregación con Document_Set.
- Mètodos:
    - Equals(Object o): método que determina si dos instancias de la clase Document son las mismas.

Clase DocumentSet

- Descripción: conjunto de documentos que gestiona el programa Seshat
- Cardinalidad: puede haber 0 o más documentos de la clase documento.
- Atributos:
    - ·docs: un map con un string de key mapeado a otro mapa con string de key y documento como valor.
- Relaciones:
    - ·Relación de agregación con Document.
    - ·Relación de relación con AuthorSet.
    - ·Relación de relación con BooleanExpressionSet.
    - ·Relación de relación con VectorialSpace.
- Mètodos:
    - exist(string author,string title): determina si existe una instancia de Documento con el nombre author y título title.
    - listDocuments(): lista los documentos con su autor y título.
    - listDocuments(string author):lista los títulos de los documentos con autor author.
    - addAuthor(Author author, string title, int id): añade a la lista un documento con autor Author, título title e identificador id.

- remove(string author,string title): elimina un documento con autor author y título title.
- haveTitles(string name): determina si hay un documento con título name.
- findSimilars(string author,string title,int k): determina los k documentos más similares al documento con autor author y título title.

Clase Author

- Descripción: usuarios que manipulan documentos.
- Cardinalidad: indeterminada con AuthorSet.
- Atributos:
  - ·name: nombre del autor
- Relaciones:
  - ·Relación de agregación con AuthorSet.
  - ·Relación de asociación con documento.
- Mètodos:
  - toString(): convierte la clase author en una string
  - Equals(Object o): método que determina si dos instancias de la clase Author son iguales.

Clase AuthorSet

- Descripción: conjunto de autores.
- Cardinalidad: indeterminada con autores.
- Atributos:
  - ·auths: set ordenador de autores
- Relaciones:
  - ·Relación de agregación con Author.

Clase VectorialSpace

- Descripción: implementación del algoritmo modelo espacio vectorial para ordenar documentos según similitud
- Cardinalidad: uno a uno con Document_Set
- Atributos:
  - ·library: conjunto de todos los documentos
- Relaciones:
  - ·Relación de asociación con Document_Set para poder aplicar el algoritmo sobre los documentos.
- Métodos:
  - existsTerm(Document Doc, string term): determina si existe en el contenido de Doc el término term.

- termFreq(Document Doc): calcula para el documento Doc los términos de su contenido y su frecuencia.
- documentWeight(Document Doc): calcula para el documento Doc el peso de los términos de su contenido.
- addAuthorTitle(TreeMap<Double, AuthorTitleList> similarity,Double sim,Document doc): añade en el Map de similaridad el nombre del autor y el título del documento de Doc para las claves repetidas.
- vectorSpaceModel(Document Doc): calcular la similaridades de otros documentos respecto al documento Doc.
- getSimilars(Document Doc,int k): muestra el título y el autor de los k documentos más parecidos a Doc.

Clase AuthorTitleList

- Descripción:Estructura de datos auxiliar para evitar anidación en la clase VectorialSpace, es un conjunto de un array con nombre de autor y nombre de documento.
- Cardinalidad: muchos a muchos con VectorialSpace
- Atributos: AuthorTitleList: arraylist de un Pair con autor y título.
- Relación de asociación con VectorialSpace.
- Mètodos:
  - add(Document d): añade a la lista el autor y el título del documento d.
  - addNames(string author,string title): añade directamente a la lista los nombres de los parámetros.
  - getAuthorTitleList(): devuelve el atributo principal en otro array.

Clase BooleanExpressionSet

- Descripción: Conjunto de expresiones booleanas.
- Cardinalidad: Uno a uno con DocumentSet.
- Atributos:
  ·bExpresions: treeMap que guarda todo el conjunto de expresiones booleanas donde cada expresión booleana tiene asociada una *id*.
- Relaciones:
  ·Relación de agregación con BooleanExpression.
- Mètodos:
  - BooleanExpressionSet(): función constructora.
  - add(String s): Añade una expresión booleana. Devuelve el id de la expresión booleana.
  - remove(int n): Elimina la expresión booleana con *id* ‘n’.
  - list(): Lista todas las expresiones booleanas con su respectivo *id*.
  - evaluate(int id, Document doc): Devuelve que documentos cumplen la expresión booleana con *id* ‘id’.

- getBooleanExpresion(int id): Devuelve el contenido de la expresión booleana con *id* ‘id.

Clase BooleanExpression

- Descripción: Guarda una expresión booleana
- Cardinalidad: Indeterminada con BooleanExpressionSet.
- Atributos:
  - ·expression: guarda el contenido de la expresión booleana en forma de string.
  - ·root: guarda el binTree de la expresión booleana.
- Relaciones:
  - ·Relación de agregación con BooleanExpressionSet.
  - ·Relación de asociación con Node.
- Mètodos:
  - Boolean_expression(String expression): Constructora de la clase.
  - correctNot(Node node): Corrige el binTree debido a los *not’s*.
  - getExpression(): Devuelve el contenido de la expresión booleana, es decir, el atributo expression.
  - evaluate (Document doc): Devuelve si el documento doc cumple con la expresión booleana.

Clase Node

- Descripción: Clase para poder simular un binTree.
- Cardinalidad: Una a una con BooleanExpression.
- Atributos:
  - ·value: string que guarda el valor del nodo
  - ·right: hijo derecho, que también es un nodo.
  - ·left: hijo izquierdo, que también es un nodo.
- Relaciones:
  - ·Dos relaciones de asociación con Node: left y right.
  - ·Relación de asociación con BooleanExpression.
- Mètodos:
  - Node(String value): Constructora de la clase.
  - setLeft(Node node): *Seteas* el nodo izquierdo.
  - setRight(Node node): *Seteas* el nodo derecho.
  - setValue(String value): *Seteas* el valor del nodo.
  - getValue(): Devuelve el valor del nodo.
  - getLeft(): Devuelve el nodo izquierdo.
  - getRight(): Devuelve el nodo derecho.
  - isLeaf(): Devuelve si el nodo es hoja, es decir, no tiene ni nodo izquierdo, ni nodo derecho.
  - hasLeftNode(): Devuelve si tiene nodo izquierdo.

- hasRightNode(): Devuelve si tiene nodo derecho.
- copy(Node n): Copia el nodo n, al nodo que llama la función.
- void printTree(Node aux, int level): Imprime por pantalla el *binTree*.

Clase Pair

- Descripción: implementación la estructura de dato pair
- Cardinalidad: uno a uno con Document_Set
- Atributos:
    - ·key: primer valor
    - ·value: segundo valor
- Relaciones:
    - ·Relación de asociación con Document_Set para poder aplicar la estructura de dato automáticamente.

Capa Persistencia:

Clase SaveData

- Descripción: es la clase que se encarga de cargar y guardar la información de índices del programa
- Cardinalidad: uno a uno con ControladorPersistencia

Clase SaveDocument

- Descripción: clase abstracta plantilla para manejar el guardado y las diferentes exportaciones/importaciones de documentos
- Cardinalidad: uno a uno con ControladorPersistencia

Clase SaveSerialized

- Descripción: clase extensión de SaveDocument que se encarga de guardar el documento con la extensión propia de nuestro software
- Cardinalidad: uno a uno con ControladorPersistencia

Clase SavePlainText

- Descripción: clase extensión de SaveDocument que se encarga de exportar/importar documentos en formato de texto plano
- Cardinalidad: uno a uno con ControladorPersistencia

Clase SaveXML

- Descripción: clase extensión de SaveDocument que se encarga de exportar/importar documentos en formato XML
- Cardinalidad: uno a uno con ControladorPersistencia

Capa Presentació:

Clase Main

- Descripción: clase que se encarga de gestionar el JFrame principal de la parte de interfaz gráfica del usuario
- Cardinalidad: uno a uno con ControladorPresentacion

Clase AuthorView

- Descripción: clase extensión de JPanel, recoge las funciones de filtrado de la búsqueda de autores, muestra la lista de autores y permite al usuario crear un nuevo autor
- Cardinalidad: uno a uno con ControladorPresentacion

Clase AuthorList

- Descripción: clase extensión de JPanel, que se encarga de organizar la visualización de los autores existentes
- Cardinalidad: uno a uno con AuthorView

Clase AuthorBox

- Descripción: clase extensión de JPanel, define el panel con el nombre del autor y el botón de eliminar
- Cardinalidad: muchos a uno con AuthorList

Clase DocumentView

- Descripción: clase extensión de JPanel, recoge las funciones de filtrado de la búsqueda de documentos, muestra la lista de documentos y permite al usuario crear un nuevo documento
- Cardinalidad: uno a uno con ControladorPresentacion

Clase DocumentList

- Descripción: clase extensión de JPanel, que se encarga de organizar la visualización de los documentos existentes
- Cardinalidad: uno a uno con DocumentView

Clase DocumentBox

- Descripción: clase extensión de JPanel, define el panel con el nombre del documento y el botón de eliminar
- Cardinalidad: muchos a uno con DocumentList

Clase NewDocument

- Descripción: clase extensión de JFrame, define el frame que le pedirá al usuario el nombre del documento y del autor del nuevo documento que quiere crear y crea un documento con estas características.
- Cardinalidad: uno a uno con ControladorPresentacion