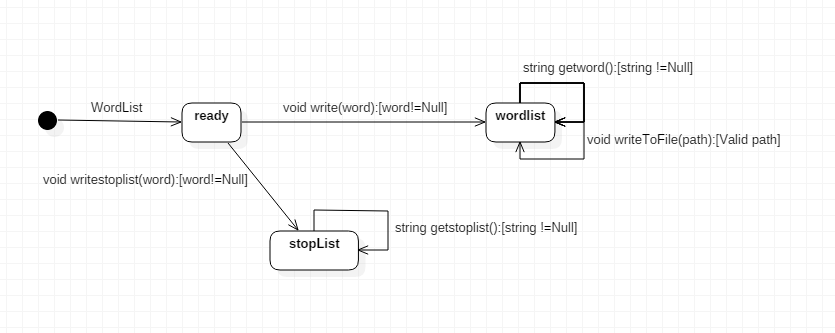
WordList

WordList is an object of the Wordlist Parser class. It’s the output of the Wordlist Parser class.The parser extracted punctuation and spaces to produce this list of words. WordList class takes two methods: read() that reads words of type string from the InputFileName and write() writes back the string of words in to the file.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Signature[Parameters]** | **Signature[Return]** | **Description** |
| getword | N/A | string | Gets a single word from wordlist |
| getstoplist | N/A | string | Gets a single word from stoplist |
| write | string | N/A | Write a word to wordlist |
| writestoplist | string | N/A | Writes word to stoplist |
| writeToFile | string | N/A | Writes wordlist to a file |
| Wordlist | N/A | N/A | Constructor |
| ~Wordlist | N/A | N/A | Destructor |



FreqList

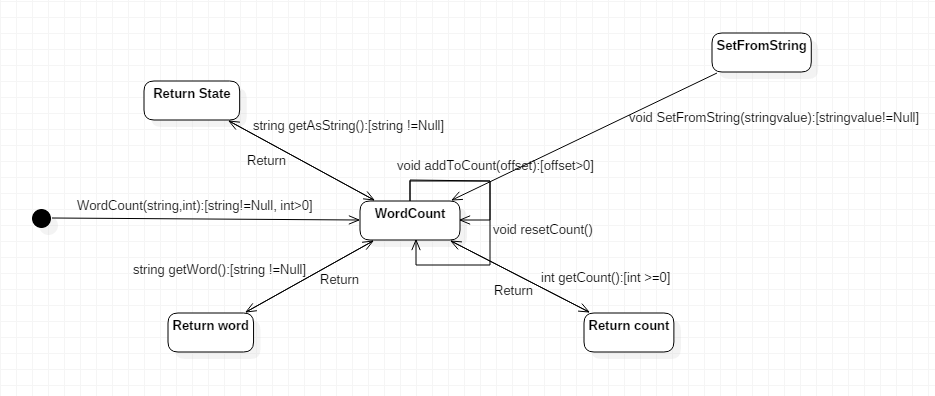
Holds a list of WordCount objects which contains each word and its associated count i.e. the number of times the word appears in the document. It has the read () and write () methods to map from a file. It also provides a method for sorting the list by their counts. It has a method that discards the WordCount objects with very less frequency.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Signature[Parameters]** | **Signature[Return]** | **Description** |
| read | string | N/A | Open File and get the mapitems |
| write | string | N/A | Close file and write the mapitems |
| getFirst | N/A | WordCount | Returns the first element of WordCount list |
| getLast | N/A | WordCount | Returns the last element of list |
| getNext | N/A | WordCount | Returns the next element of list |
| getSize | N/A | WordCount | Returns the size of the list |
| sortbycount | N/A | N/A | Sort the WordCount objects in the ascending order of frequency |
| addinlist | WordCount | N/A | Adds a new WordCount object to list |
| removefromlist | WordCount | N/A | Removes a WordCount object from list |
| updatelist | int | N/A | Updates the count value for a WordCount object |
| getlist | N/A | list<WordCount> | returns the WordCount list |
| Topn | int | N/A | Finalizes the Top n elements in list |
| FreqList | N/A | N/A | Constructor |
| ~ FreqList | N/A | N/A | Destructor |

WordCount

Holds a word and its associated count (which for our purposes is the number of times the word appears in the document). It enables us to get the word of the object through getword(). Returns the freq of word through setcount(). It must provide the asString() method that puts the object’s state (attribute values) into a string that it returns. It must also provide fromString() (the reverse of asString()).

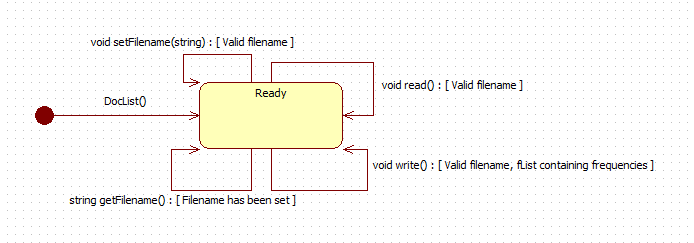
|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Signature[Parameters]** | **Signature[Return]** | **Description** |
| resetCount | N/A | N/A | Resets the values of the WordCount object |
| addToCount | int | N/A | Adds to the frequency of a word |
| setWord | int, string | N/A | Called implicitly in Constructor |
| getWord | N/A | string | Gets the word of WordCount object |
| setCount | int | N/A | Called implicitly in Constructor |
| getCount | N/A | int | Gets the count of a WordCount object |
| getAsString | N/A | string | Puts the object’s state into a string |
| setFromString | string | N/A | The reverse of getAsString() |
| WordCount | string, int | N/A | Parametrized Constructor, initializes the word and count value |
| ~ WordCount | N/A | N/A | Destructor |



DocList

Contains a FreqList and the filename associated with it. Read and Write save and load from that file.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Signature[Parameters]** | **Signature[Return]** | **Description** |
| read | N/A | N/A | Reads from the file |
| write | N/A | N/A | Writes to the file |
| getFilename | N/A | string | Returns the filename |
| setFilename | string | N/A | Setst the filename |
| fList | N/A | FreqList | The FrequencyList object |
| DocList | N/A | N/A | Constructor |
| ~DocList | N/A | N/A | Destructor |



**WordListParser**

Has a method that reads the input file (an ascii file), parses it into words, and strips out punctuation. The method returns a WordList object.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Signature[Parameters]** | **Signature[Return]** | **Description** |
| parser | String (file location) | wordlist | Reads the input and parses into words |
| wordlistparser | n/a | n/a | constructor |
| ~wordlistparser | n/a | n/a | destructor |

**ICloudLayout**

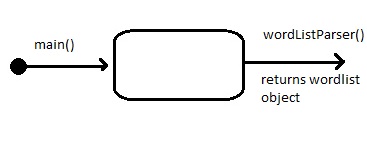
Specifies a method that will be implemented by the various layout classes. This method should update a WordMap by setting the locations (Point) for each word in the map.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Signature[Parameters]** | **Signature[Return]** | **Description** |
| ICloudLayout | n/a | n/a | Constructor |
| ~ICloudLayout | n/a | n/a | Destructor |
| setLocation | wordmap, Point | n/a | Sets the location for each word in a word map |

**StackedLayout**

Implements the method in ICloudLayout by setting with locations for each word. Words are stacked from the bottom of the page upwards, with font size going from biggest (at the bottom) to smallest (at the top).

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Signature[Parameters]** | **Signature[Return]** | **Description** |
| ICloudLayout::setLocation | wordmap, Point | n/a | Implements the method in ICloudLayout by setting the locations for a stacked layout |
| StackedLayout | n/a | n/a | constructor |
| ~ StackedLayout | n/a | n/a | destructor |



Our method is to be used within other parts of the program, taking in a text file and either parsing it as needed or parsing it and returning a wordlist object to the next step in the program.

Our ICloudLayout and StackedLayout are also to be used by other groups, making the text file readable by their respective methods.