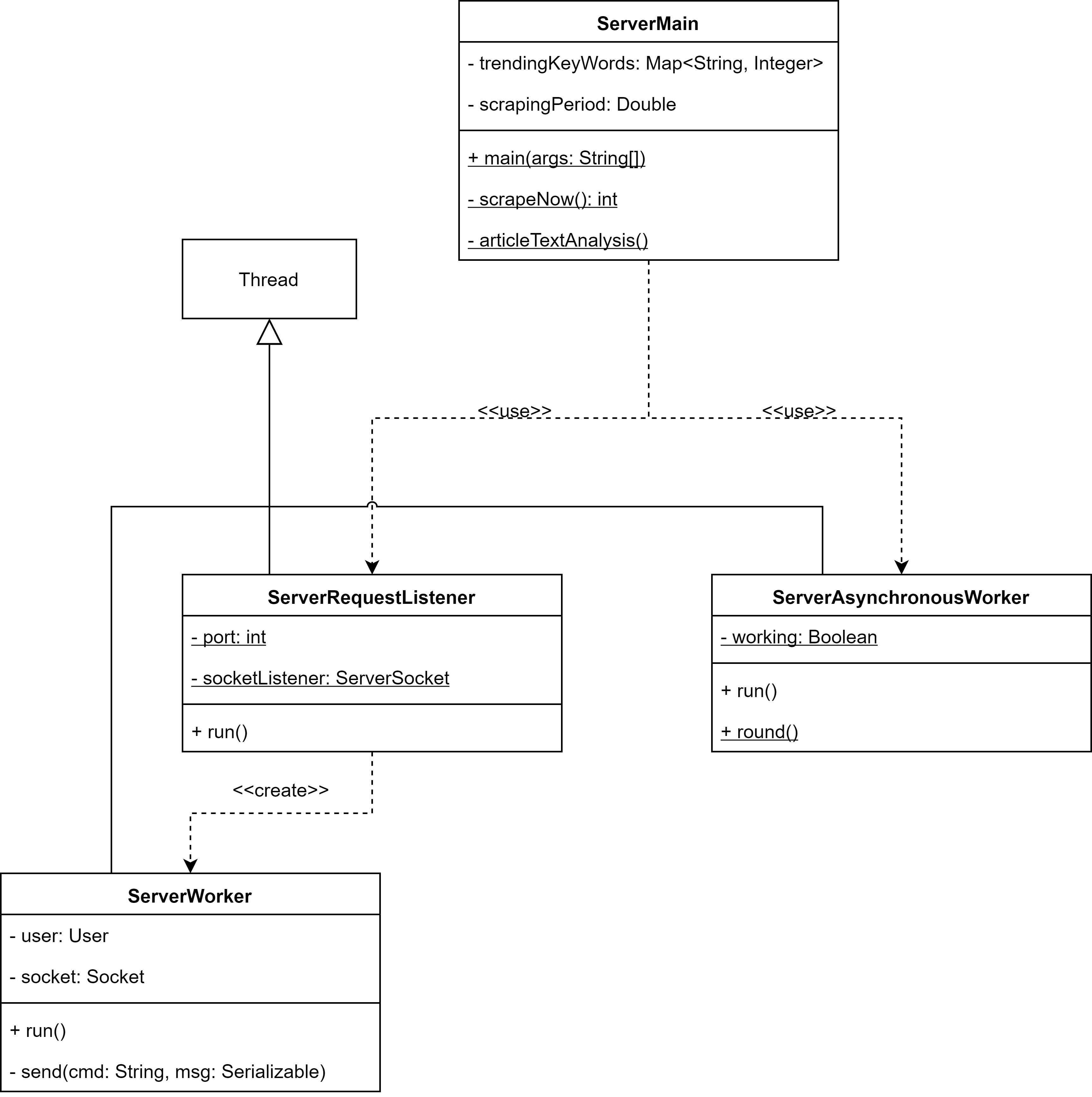
# Server’s architecture

## Scheme:



## ServerMain Class:

ServerMain is the main class of the server side of the application. It has the task to initialize the main structures of the server, to launch the worker threads, and to call the back-end methods for the articles’ text analysis and for forcing a scrape round (collecting articles from all sites backwards in time until the last scraped article).

When the server is launched, mongoDB indexes are created first thing. Then the listener thread ServerRequestListener and the scraping thread ServerAsynchronousWorker are created (more details in the following paragraphs).

It contains the scrapingPeriod variable, which determines the cool down period for the scraping process. It’s accessed in a thread-safe way, so that its modifications due to admin requests are concurrent with its readings.

trendingKeywords is a structure representing the top 10 trending keywords in the reference period and their frequency. It’s updated after each scraping round, and sent to each user that logs in to build the trending keyword chart.

When an admin wants to force a scraping round, the scrapeNow() method is called. It at first checks that no scraping threads are already running; if it is, this method returns with an error code. If no scraping threads are active, this method creates a ServerAsynchronousWorker object, and calls its round() method: this will start a round of scraping.

Moreover, each time a scraping round is completed, the articleTextAnalysis() method is called, which invokes the back-end methods for the retrieval of stored articles and the methods for text analysis collected in TextAnalyzer class.

## ServerRequestListener Class:

ServerRequestListener is the core of the multi-threaded server architecture. It implements the main thread for the communications with the clients, creating a socket on which requests to the server are listened. Each time a new connection is accepted, a new thread (ServerWorker) is created.

## ServerWorker Class:

ServerWorker is a thread communicating with a specific user. It collects all the methods for the intercommunication with the client’s socket.

Each time a connected user sends a request to the server, his correlated thread receives it and, according to the command, calls a specific method of ServerWorker class to handle it.

This class contains a method for each action provided for by the communication protocol; these methods are not represented in the scheme above, because we opted for a compact version of this diagram to make it more readable.

## ServerAsynchronousWorker Class:

ServerAsynchronousWorker is a thread that periodically works for updating the database of the system. In particular, each ServerMain.scrapingPeriod seconds, it does a scraping round (collecting all the articles until the last one seen during last round), it calls the methods for updating the articles present in the database by applying them their text analysis, and it finally invokes the method for calculating the new trending keywords.

Since a scraping round can be forced by the admin, only one scraping process must be done at a time: this implies the presence of a semaphore (‘working’ boolean variable) that is set to true each time the round() function is called, and it is set to false when the round is completed. Before invoking this function, this variable is checked: if ‘working’ is true, then the round function returns immediately.

Accessing to working variable is thread-safe, due to the presence of a synchronization on the same variable; this is important to avoid situations in which two processes attempt to change the value of this variable.