Algoritmo Berkeley (java) con sockets REQ/REP ZeroMMQ

J.JavierGutiérrezGil(jogugi@posgrado.upv.es)V0.1

Generado por Doxygen 1.8.16

Capítulo 1

Indice de namespaces

1.1. Paquetes

Aquí van los paquetes con una breve descripción (si etá disponible):

Capítulo 2

Indice jerárquico

2.1. Jerarquía de la clase

Esta lista de herencias esta ordenada aproximadamente por orden alfabético:

Capítulo 3

índice de clases

3.1. Lista de clases

Lista de las clases, estructuras, uniones e interfaces con una breve descripción:

Capítulo 4

Ìndice de archivos

4.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

Capítulo 5

Documentación de namespaces

5.1. Paquetes org

Paquetes

- package **jogugi**

5.2. Paquetes org.jogugi

Paquetes

- package **berkeley**

5.3. Paquetes org.jogugi.berkeley

Clases

- class **AbstractNode**

  Clase abstracta que implementa la interfaz ***INode*** (p. **??**).
- class **Config**

  Clase de configuración para los nodos en el sistema.
- class **Follower**

  Clase que representa un nodo seguidor en un sistema distribuido.
- class **FollowerInfo**

  Clase que representa información detallada de un nodo seguidor en el sistema.
- enum **FollowerState**

  Enumeración que define los posibles estados de un seguidor durante el proceso de actualización de la hora del sistema.
- interface **INode**

  Interfaz que define las operaciones básicas para un nodo en el sistema.
- class **Leader**

  Representa un nodo líder en el sistema.
- class **LoggerManager**

  La clase.
- class **Main**

  Clase principal que gestiona la inicialización y ejecución del líder y los seguidores en el sistema.
- class **SocketZeroMQException**

  Excepción personalizada para errores en la operación de sockets de ZeroMQ.

Capítulo 6

Documentación de las clases

6.1. Referencia de la Clase org.jogugi.berkeley.AbstractNode

Clase abstracta que implementa la interfaz **`INode`** (p. **??**).

Herencias **org.jogugi.berkeley.INode**.

Heredado por **org.jogugi.berkeley.Follower** y **org.jogugi.berkeley.Leader**.

Métodos públicos

- void **initializeNode** (String **name**, String **address**, int **timeout**)

  Inicializa el nodo con el nombre, dirección y tiempo de espera.
- void **initializeNode** (String **name**, String **address**, int **timeout**, HashMap< String, String > listNode↩ Addresses)

  Inicializa el nodo con el nombre, dirección, tiempo de espera y la lista de nodos.
- String **sendMessageSync** (String **address**, String message) throws SocketZeroMQException

  Envía un mensaje de manera síncrona y espera una respuesta.
- void **sendMessageAsync** (String **address**, String message) throws SocketZeroMQException

  Envía un mensaje de manera asíncrona.
- void **startListening** () throws SocketZeroMQException

  Inicia el proceso de escucha de mensajes en el nodo.
- void **close** () throws SocketZeroMQException

  Cierra los recursos utilizados por el nodo, incluyendo el socket y el contexto.
- void **startAlgorithm** () throws SocketZeroMQException, InterruptedException, ExecutionException

  Método para iniciar el algoritmo del nodo.

Métodos protegidos

- abstract String **handleProcess** (String message) throws SocketZeroMQException

  Método abstracto para manejar el procesamiento de los mensajes recibidos.

Atributos protegidos

- ZContext **context** = null

  Contexto de ZeroMQ utilizado para la creación de sockets.
- ZMQ.Socket **socket** = null

  Socket REP del nodo utilizado para recibir mensajes.
- String **name**

  Nombre del nodo.
- String **address**

  Dirección del nodo en formato `host:puerto`*.*
- int **timeout**

  Tiempo de espera en milisegundos para operaciones de socket.
- Map< String, String > **nodeAddresses** = null

  Mapa que contiene las direcciones de otros nodos en el sistema.
- List< ZMQ.Socket > **listSocketsREQ**

  Lista de sockets REQ utilizados para enviar solicitudes a otros nodos.

Atributos privados estáticos

- static final Logger **logger** = LoggerFactory.getLogger(AbstractNode.class)

  Logger utilizado para registrar información, advertencias y errores.

6.1.1. Descripción detallada

Clase abstracta que implementa la interfaz **`INode`** (p. **??**).

La clase **`AbstractNode`** (p. **??**) proporciona la implementación base para un nodo en el sistema. Maneja el contexto de ZeroMQ, los sockets de comunicación y contiene las operaciones comunes para el envío y recepción de mensajes, además de almacenar la información del nodo como su nombre, dirección y tiempos de espera.

Definición en la línea 23 del archivo AbstractNode.java.

6.1.2. Documentación de las funciones miembro

6.1.2.1. close()

`void org.jogugi.berkeley.AbstractNode.close ( ) throws` **`SocketZeroMQException`**

Cierra los recursos utilizados por el nodo, incluyendo el socket y el contexto.

Este método cierra el socket asociado al nodo y también el contexto ZeroMQ, liberando así los recursos utilizados por el nodo. Una vez cerrados estos recursos, el nodo deja de poder enviar o recibir mensajes. El proceso de cierre es registrado en los logs del sistema.

Excepciones

SocketZeroMQException *(*p. **??***)*	Si ocurre un error al intentar cerrar el socket o el contexto.

Implementa **org.jogugi.berkeley.INode** (p. **??**).

Reimplementado en **org.jogugi.berkeley.Leader** (p. **??**).

Definición en la línea 240 del archivo AbstractNode.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.context, org.jogugi.berkeley.AbstractNode.logger, org.jogugi.↩ berkeley.AbstractNode.name y org.jogugi.berkeley.AbstractNode.socket.

Referenciado por org.jogugi.berkeley.AbstractNode.startListening().

6.1.2.2. handleProcess()

```
abstract String org.jogugi.berkeley.AbstractNode.handleProcess (
            String message ) throws  SocketZeroMQException  [abstract], [protected]
```

Método abstracto para manejar el procesamiento de los mensajes recibidos.

Este método debe ser implementado en las subclases específicas para definir el comportamiento que debe seguir el nodo al recibir un mensaje. El mensaje es procesado y se devuelve una respuesta basada en la lógica del nodo.

Parámetros

message	El mensaje recibido que debe ser procesado.

Devuelve

La respuesta procesada al mensaje.

Excepciones

SocketZeroMQException *(*p. **??***)*	Si ocurre un error al procesar el mensaje.

Reimplementado en **org.jogugi.berkeley.Leader** (p. **??**) y **org.jogugi.berkeley.Follower** (p. **??**).

Referenciado por org.jogugi.berkeley.AbstractNode.startListening().

6.1.2.3. initializeNode() [1/2]

```
void org.jogugi.berkeley.AbstractNode.initializeNode (
            String name,
```

```
            String address,
            int timeout )
```

Inicializa el nodo con el nombre, dirección y tiempo de espera.

Este constructor inicializa un nodo con los parámetros básicos: nombre del nodo, dirección del nodo y el tiempo de espera. También crea el contexto de ZeroMQ para la comunicación.

Parámetros

name	Nombre del nodo.
address	Dirección del nodo en formato `host:puerto`.
timeout	Tiempo de espera para las operaciones de socket.

Implementa **org.jogugi.berkeley.INode** (p. **??**).

Definición en la línea 78 del archivo AbstractNode.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.AbstractNode.logger, org.↩ jogugi.berkeley.AbstractNode.name y org.jogugi.berkeley.AbstractNode.timeout.

6.1.2.4. initializeNode() [2/2]

```
void org.jogugi.berkeley.AbstractNode.initializeNode (
            String name,
            String address,
            int timeout,
            HashMap< String, String > listNodeAddresses )
```

Inicializa el nodo con el nombre, dirección, tiempo de espera y la lista de nodos.

Este constructor inicializa un nodo con los parámetros básicos: nombre del nodo, dirección del nodo, tiempo de espera y la lista de direcciones de otros nodos en el sistema. También crea el contexto de ZeroMQ para la comunicación.

Parámetros

name	Nombre del nodo.
address	Dirección del nodo en formato `host:puerto`.
timeout	Tiempo de espera para las operaciones de socket.
listNodeAddresses	Lista de direcciones de otros nodos en el sistema.

Implementa **org.jogugi.berkeley.INode** (p. **??**).

Reimplementado en **org.jogugi.berkeley.Leader** (p. **??**).

Definición en la línea 100 del archivo AbstractNode.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.AbstractNode.logger, org.↩ jogugi.berkeley.AbstractNode.name y org.jogugi.berkeley.AbstractNode.timeout.

6.1.2.5. sendMessageAsync()

```
void org.jogugi.berkeley.AbstractNode.sendMessageAsync (
            String address,
            String message ) throws  SocketZeroMQException
```

Envía un mensaje de manera asíncrona.

Este método crea un socket PUSH para enviar un mensaje de manera asíncrona a la dirección especificada. No espera una respuesta, sino que simplemente envía el mensaje y termina la operación.

Parámetros

address	La dirección a la que se enviará el mensaje en formato `host:puerto`.
message	El mensaje que se va a enviar.

Excepciones

SocketZeroMQException *(*p. **??***)*	Si ocurre un error al enviar el mensaje.

Implementa **org.jogugi.berkeley.INode** (p. **??**).

Definición en la línea 159 del archivo AbstractNode.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.AbstractNode.context, org.↩ jogugi.berkeley.AbstractNode.logger y org.jogugi.berkeley.SocketZeroMQException.ErrorType.SEND_ERROR.

6.1.2.6. sendMessageSync()

```
String org.jogugi.berkeley.AbstractNode.sendMessageSync (
            String address,
            String message ) throws  SocketZeroMQException
```

Envía un mensaje de manera síncrona y espera una respuesta.

Este método crea un socket REQ para enviar un mensaje a una dirección específica y espera una respuesta del servidor. Si no se recibe respuesta dentro del tiempo de espera o si ocurre un error durante el envío/recepción, se lanza una excepción.

Parámetros

address	La dirección a la que se enviará el mensaje en formato `host:puerto`.
message	El mensaje que se va a enviar.

Devuelve

La respuesta recibida del servidor.

Excepciones

SocketZeroMQException *(*p. **??***)*	Si ocurre un error al enviar o recibir el mensaje.

Implementa **org.jogugi.berkeley.INode** (p. **??**).

Definición en la línea 123 del archivo AbstractNode.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.AbstractNode.context, org.↩ jogugi.berkeley.AbstractNode.logger, org.jogugi.berkeley.SocketZeroMQException.ErrorType.RECEIVE_ERROR y org.jogugi.berkeley.SocketZeroMQException.ErrorType.SEND_ERROR.

6.1.2.7. startAlgorithm()

`void org.jogugi.berkeley.AbstractNode.startAlgorithm ( ) throws` **`SocketZeroMQException,`** `Interrupted↩ Exception, ExecutionException`

Método para iniciar el algoritmo del nodo.

Este método se sobrecarga por el tipo de nodo (por ejemplo, nodo líder o seguidor) y debe ser implementado en las subclases específicas. Si no se implementa en una subclase, lanza un error.

Excepciones

SocketZeroMQException *(*p. **??***)*	Si ocurre un error en la comunicación con ZeroMQ.
InterruptedException	Si el hilo es interrumpido mientras se ejecuta el algoritmo.
ExecutionException	Si ocurre un error durante la ejecución del algoritmo.

Implementa **org.jogugi.berkeley.INode** (p. **??**).

Reimplementado en **org.jogugi.berkeley.Follower** (p. **??**) y **org.jogugi.berkeley.Leader** (p. **??**).

Definición en la línea 267 del archivo AbstractNode.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.logger.

6.1.2.8. startListening()

`void org.jogugi.berkeley.AbstractNode.startListening ( ) throws` **`SocketZeroMQException`**

Inicia el proceso de escucha de mensajes en el nodo.

Este método pone al nodo en un estado en el que puede escuchar mensajes entrantes en su socket. El nodo recibe mensajes de forma continua en un hilo separado, los procesa y, opcionalmente, responde con un mensaje. Si recibe un mensaje que indica que el nodo debe cerrarse (por ejemplo, un mensaje con la operación `"CLOSE"`), el nodo cierra su socket y termina el proceso de escucha.

Excepciones

SocketZeroMQException *(p. **??**)*	Si el socket no está inicializado o ocurre un error en el proceso de escucha.

Implementa **org.jogugi.berkeley.INode** (p. **??**).

Definición en la línea 184 del archivo AbstractNode.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.close(), org.jogugi.berkeley.SocketZeroMQException.Error↩ Type.CONNECTION_ERROR, org.jogugi.berkeley.AbstractNode.handleProcess(), org.jogugi.berkeley.Abstract↩ Node.logger, org.jogugi.berkeley.AbstractNode.name, org.jogugi.berkeley.SocketZeroMQException.ErrorType.R↩ ECEIVE_ERROR y org.jogugi.berkeley.AbstractNode.socket.

Referenciado por org.jogugi.berkeley.Follower.startAlgorithm().

6.1.3. Documentación de los datos miembro

6.1.3.1. address

```
String org.jogugi.berkeley.AbstractNode.address  [protected]
```

Dirección del nodo en formato `host:puerto`.

Definición en la línea 43 del archivo AbstractNode.java.

Referenciado por org.jogugi.berkeley.Follower.handleProcess(), org.jogugi.berkeley.Follower.initializeNode(), org.↩ jogugi.berkeley.AbstractNode.initializeNode(), org.jogugi.berkeley.Leader.initializeNode(), org.jogugi.berkeley.↩ Leader.processFollowers(), org.jogugi.berkeley.AbstractNode.sendMessageAsync(), org.jogugi.berkeley.Abstract↩ Node.sendMessageSync() y org.jogugi.berkeley.Follower.startAlgorithm().

6.1.3.2. context

```
ZContext org.jogugi.berkeley.AbstractNode.context = null  [protected]
```

Contexto de ZeroMQ utilizado para la creación de sockets.

Definición en la línea 28 del archivo AbstractNode.java.

Referenciado por org.jogugi.berkeley.AbstractNode.close(), org.jogugi.berkeley.Leader.close(), org.jogugi.↩ berkeley.Leader.processFollowerTime(), org.jogugi.berkeley.Leader.sendCloseMessage(), org.jogugi.berkeley.↩ AbstractNode.sendMessageAsync(), org.jogugi.berkeley.AbstractNode.sendMessageSync(), org.jogugi.berkeley.↩ Leader.sendTimeUpdateToFollower() y org.jogugi.berkeley.Follower.startAlgorithm().

6.1.3.3. listSocketsREQ

```
List<ZMQ.Socket> org.jogugi.berkeley.AbstractNode.listSocketsREQ  [protected]
```

Lista de sockets REQ utilizados para enviar solicitudes a otros nodos.

Definición en la línea 58 del archivo AbstractNode.java.

6.1.3.4. logger

```
final Logger org.jogugi.berkeley.AbstractNode.logger = LoggerFactory.getLogger(AbstractNode.↩
class)  [static], [private]
```

Logger utilizado para registrar información, advertencias y errores.

Definición en la línea 63 del archivo AbstractNode.java.

Referenciado por org.jogugi.berkeley.AbstractNode.close(), org.jogugi.berkeley.AbstractNode.initializeNode(), org.jogugi.berkeley.AbstractNode.sendMessageAsync(), org.jogugi.berkeley.AbstractNode.sendMessageSync(), org.jogugi.berkeley.AbstractNode.startAlgorithm() y org.jogugi.berkeley.AbstractNode.startListening().

6.1.3.5. name

```
String org.jogugi.berkeley.AbstractNode.name  [protected]
```

Nombre del nodo.

Definición en la línea 38 del archivo AbstractNode.java.

Referenciado por org.jogugi.berkeley.AbstractNode.close(), org.jogugi.berkeley.Follower.displayLeaderMessage(), org.jogugi.berkeley.Follower.handleProcess(), org.jogugi.berkeley.Follower.initializeNode(), org.jogugi.berkeley.↩ Leader.initializeNode(), org.jogugi.berkeley.AbstractNode.initializeNode(), org.jogugi.berkeley.Follower.mod↩ SystemTime(), org.jogugi.berkeley.Leader.processFollowers(), org.jogugi.berkeley.Leader.processFollowerTime(), org.jogugi.berkeley.Leader.sendCloseMessage(), org.jogugi.berkeley.Leader.sendTimeUpdateToFollower(), org.↩ jogugi.berkeley.Follower.startAlgorithm() y org.jogugi.berkeley.AbstractNode.startListening().

6.1.3.6. nodeAddresses

```
Map<String, String> org.jogugi.berkeley.AbstractNode.nodeAddresses = null  [protected]
```

Mapa que contiene las direcciones de otros nodos en el sistema.

Definición en la línea 53 del archivo AbstractNode.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime(), org.jogugi.berkeley.Leader.↩ processFollowers() y org.jogugi.berkeley.Leader.processFollowerTime().

6.1.3.7. socket

```
ZMQ.Socket org.jogugi.berkeley.AbstractNode.socket = null  [protected]
```

Socket REP del nodo utilizado para recibir mensajes.

Definición en la línea 33 del archivo AbstractNode.java.

Referenciado por org.jogugi.berkeley.AbstractNode.close(), org.jogugi.berkeley.Leader.close(), org.jogugi.↩ berkeley.Follower.startAlgorithm() y org.jogugi.berkeley.AbstractNode.startListening().

6.1.3.8. timeout

```
int org.jogugi.berkeley.AbstractNode.timeout  [protected]
```

Tiempo de espera en milisegundos para operaciones de socket.

Definición en la línea 48 del archivo AbstractNode.java.

Referenciado por org.jogugi.berkeley.Follower.initializeNode(), org.jogugi.berkeley.AbstractNode.initializeNode(), org.jogugi.berkeley.Leader.initializeNode(), org.jogugi.berkeley.Leader.processFollowers(), org.jogugi.berkeley.↩ Leader.processFollowerTime(), org.jogugi.berkeley.Leader.sendCloseMessage(), org.jogugi.berkeley.Leader.↩ sendCloseMessagesToFollowers() y org.jogugi.berkeley.Leader.sendTimeUpdateToFollower().

La documentación para esta clase fue generada a partir del siguiente fichero:

- **AbstractNode.java**

6.2. Referencia de la Clase org.jogugi.berkeley.Config

Clase de configuración para los nodos en el sistema.

Clases

- class **FollowerConfig**

  Configuración del nodo seguidor.
- class **LeaderConfig**

  Configuración del nodo líder.

Atributos públicos

- **LeaderConfig leader**

  Nodo líder del sistema.
- List< **FollowerConfig** > **followers**

  Lista de nodos seguidores.
- int **timeout**

  Tiempo de espera global para todos los nodos.

6.2.1. Descripción detallada

Clase de configuración para los nodos en el sistema.

Esta clase proporciona la configuración necesaria para inicializar los nodos en el sistema, incluyendo la configuración del líder y los seguidores, así como el tiempo de espera global.

Definición en la línea 12 del archivo Config.java.

6.2.2. Documentación de los datos miembro

6.2.2.1. followers

`List< FollowerConfig> org.jogugi.berkeley.Config.followers`

Lista de nodos seguidores.

Esta propiedad contiene una lista de configuraciones de nodos seguidores, cada una con su nombre y dirección. Los seguidores se conectan al líder.

Definición en la línea 73 del archivo Config.java.

Referenciado por org.jogugi.berkeley.Main.main().

6.2.2.2. leader

`LeaderConfig org.jogugi.berkeley.Config.leader`

Nodo líder del sistema.

Esta propiedad contiene la configuración del nodo líder, que incluye su nombre y dirección.

Definición en la línea 64 del archivo Config.java.

Referenciado por org.jogugi.berkeley.Main.main().

6.2.2.3. timeout

`int org.jogugi.berkeley.Config.timeout`

Tiempo de espera global para todos los nodos.

Este campo define el tiempo máximo que los nodos esperarán por una respuesta antes de considerar que ocurrió un error de tiempo de espera.

Definición en la línea 82 del archivo Config.java.

Referenciado por org.jogugi.berkeley.Main.main().

La documentación para esta clase fue generada a partir del siguiente fichero:

- **Config.java**

6.3. Referencia del enum org.jogugi.berkeley.SocketZeroMQException.ErrorType

Enumeración de los tipos de errores posibles.

Atributos públicos

- **CONNECTION_ERROR**
  Error relacionado con la conexión de ZeroMQ.
- **SEND_ERROR**
  Error al enviar un mensaje a través del socket.
- **RECEIVE_ERROR**
  Error al recibir un mensaje del socket.

6.3.1. Descripción detallada

Enumeración de los tipos de errores posibles.

Esta enumeración se utiliza para clasificar los diferentes tipos de errores que pueden ocurrir durante las operaciones con sockets ZeroMQ.

Definición en la línea 23 del archivo SocketZeroMQException.java.

6.3.2. Documentación de los datos miembro

6.3.2.1. CONNECTION_ERROR

`org.jogugi.berkeley.SocketZeroMQException.ErrorType.CONNECTION_ERROR`

Error relacionado con la conexión de ZeroMQ.

Definición en la línea 24 del archivo SocketZeroMQException.java.

Referenciado por org.jogugi.berkeley.Follower.startAlgorithm() y org.jogugi.berkeley.AbstractNode.startListening().

6.3.2.2. RECEIVE_ERROR

`org.jogugi.berkeley.SocketZeroMQException.ErrorType.RECEIVE_ERROR`

Error al recibir un mensaje del socket.

Definición en la línea 26 del archivo SocketZeroMQException.java.

Referenciado por org.jogugi.berkeley.AbstractNode.sendMessageSync() y org.jogugi.berkeley.AbstractNode.↩ startListening().

6.3.2.3. SEND_ERROR

`org.jogugi.berkeley.SocketZeroMQException.ErrorType.SEND_ERROR`

Error al enviar un mensaje a través del socket.

Definición en la línea 25 del archivo SocketZeroMQException.java.

Referenciado por org.jogugi.berkeley.AbstractNode.sendMessageAsync() y org.jogugi.berkeley.AbstractNode.↩ sendMessageSync().

La documentación para este enum ha sido generada a partir del siguiente fichero:

- **SocketZeroMQException.java**

6.4. Referencia de la Clase org.jogugi.berkeley.Follower

Clase que representa un nodo seguidor en un sistema distribuido.

Herencias **org.jogugi.berkeley.AbstractNode**.

Métodos públicos

- void **initializeNode** (String **name**, String **address**, String **leaderAddress**, int **timeout**) throws IOException, SocketZeroMQException

  Inicializa el nodo seguidor con la información específica.
- void **startAlgorithm** () throws SocketZeroMQException

  Inicia el algoritmo configurando y enlazando el socket ZeroMQ.

Métodos protegidos

- String **handleProcess** (String message) throws SocketZeroMQException

  Maneja y procesa los mensajes recibidos del líder.

Métodos privados

- long **displayLeaderMessage** (String leaderName, String leaderMessage, long T0, long localTime)

  Muestra el mensaje del líder y calcula un valor intermedio de tiempo (TP).
- String **modSystemTime** (long delta)

  Modifica el tiempo local del sistema en función de un delta.
- long **getCurrentTime** ()

  Obtiene la hora actual del sistema en milisegundos desde la época Unix.

Atributos privados

- String **leaderAddress**

  Dirección del nodo líder en formato `host:puerto`.

Atributos privados estáticos

- static final Logger **logger** = LoggerFactory.getLogger(Follower.class)

  Logger para registrar eventos y mensajes del nodo seguidor.

Otros miembros heredados

6.4.1. Descripción detallada

Clase que representa un nodo seguidor en un sistema distribuido.

La clase **`Follower`** (p. **??**) extiende la funcionalidad de la clase abstracta **`AbstractNode`** (p. **??**). Representa un nodo seguidor que interactúa con un nodo líder en un sistema distribuido. El seguidor es capaz de recibir y procesar mensajes del líder, así como manejar operaciones específicas como obtener la hora, modificarla, y cerrar la conexión.

Definición en la línea 21 del archivo Follower.java.

6.4.2. Documentación de las funciones miembro

6.4.2.1. displayLeaderMessage()

```
long org.jogugi.berkeley.Follower.displayLeaderMessage (
             String leaderName,
             String leaderMessage,
             long T0,
             long localTime )  [private]
```

Muestra el mensaje del líder y calcula un valor intermedio de tiempo (TP).

Este método registra el mensaje del líder recibido, la hora proporcionada por el líder (T0), y la hora local del seguidor. Además, calcula un tiempo promedio (TP) basado en T0 y el tiempo local del seguidor, y lo registra en el log.

Parámetros

leaderName	El nombre del líder que envía el mensaje.
leaderMessage	El mensaje recibido del líder.
T0	La marca de tiempo enviada por el líder en milisegundos desde la época.
localTime	La hora local actual del seguidor en milisegundos desde la época.

Devuelve

El tiempo promedio (TP) calculado a partir de T0 y la hora local del seguidor.

Definición en la línea 131 del archivo Follower.java.

Hace referencia a org.jogugi.berkeley.Follower.logger y org.jogugi.berkeley.AbstractNode.name.

Referenciado por org.jogugi.berkeley.Follower.handleProcess().

6.4.2.2. getCurrentTime()

```
long org.jogugi.berkeley.Follower.getCurrentTime ( )  [private]
```

Obtiene la hora actual del sistema en milisegundos desde la época Unix.

Este método registra el tiempo actual en milisegundos (TP) junto con la dirección del seguidor y también lo convierte en una fecha legible para su registro.

Devuelve

El tiempo actual del sistema en milisegundos desde el 1 de enero de 1970 (época Unix).

Definición en la línea 175 del archivo Follower.java.

Hace referencia a org.jogugi.berkeley.Follower.logger.

Referenciado por org.jogugi.berkeley.Follower.handleProcess().

6.4.2.3. handleProcess()

```
String org.jogugi.berkeley.Follower.handleProcess (
            String message ) throws  SocketZeroMQException  [protected]
```

Maneja y procesa los mensajes recibidos del líder.

Este método deserializa un mensaje JSON enviado por el líder, identifica la operación solicitada (por ejemplo, obtener la hora, modificar el tiempo o cerrar la conexión) y responde de acuerdo con la operación especificada.

Parámetros

message	El mensaje JSON recibido del líder.

Devuelve

Una cadena JSON con la respuesta apropiada para la operación realizada.

Excepciones

SocketZeroMQException *(*p. **??***)*	En caso de errores relacionados con el socket.

Reimplementado de **org.jogugi.berkeley.AbstractNode** (p. **??**).

Definición en la línea 69 del archivo Follower.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.Follower.displayLeader↩ Message(), org.jogugi.berkeley.Follower.getCurrentTime(), org.jogugi.berkeley.Follower.leaderAddress, org.↩ jogugi.berkeley.Follower.logger, org.jogugi.berkeley.Follower.modSystemTime() y org.jogugi.berkeley.Abstract↩ Node.name.

6.4.2.4. initializeNode()

```
void org.jogugi.berkeley.Follower.initializeNode (
            String name,
            String address,
            String leaderAddress,
            int timeout ) throws IOException,  SocketZeroMQException
```

Inicializa el nodo seguidor con la información específica.

Este método configura el nodo seguidor asignándole su nombre, dirección, dirección del líder y tiempo de espera. Extiende la funcionalidad de la clase base **`AbstractNode`** (p. **??**).

Parámetros

name	Nombre del nodo seguidor.
address	Dirección del nodo seguidor en formato `host:puerto`.
leaderAddress	Dirección del líder en formato `host:puerto`.
timeout	Tiempo de espera (en milisegundos) para las operaciones del nodo.

Excepciones

IOException	Si ocurre un error al inicializar el nodo.
SocketZeroMQException (p. **??**)	Si ocurre un error relacionado con ZeroMQ.

Definición en la línea 49 del archivo Follower.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.Follower.leaderAddress, org.↩ jogugi.berkeley.AbstractNode.name y org.jogugi.berkeley.AbstractNode.timeout.

Referenciado por org.jogugi.berkeley.Main.main().

6.4.2.5. modSystemTime()

```
String org.jogugi.berkeley.Follower.modSystemTime (
            long delta )  [private]
```

Modifica el tiempo local del sistema en función de un delta.

Este método simula un cambio en el tiempo del sistema sumando un delta al tiempo local actual. También registra el tiempo antes y después de la modificación en los logs y devuelve una representación JSON de la operación.

Parámetros

delta	El desplazamiento en milisegundos a aplicar al tiempo local actual.

Devuelve

Una cadena JSON que contiene el nombre del seguidor, el nuevo tiempo local modificado y el estado de la operación.

Definición en la línea 154 del archivo Follower.java.

Hace referencia a org.jogugi.berkeley.Follower.logger y org.jogugi.berkeley.AbstractNode.name.

Referenciado por org.jogugi.berkeley.Follower.handleProcess().

6.4.2.6. startAlgorithm()

`void org.jogugi.berkeley.Follower.startAlgorithm ( ) throws` **`SocketZeroMQException`**

Inicia el algoritmo configurando y enlazando el socket ZeroMQ.

Este método configura un socket de tipo REP (respuesta) para recibir mensajes y lo enlaza a la dirección especificada. Luego inicia el proceso de escucha de mensajes entrantes.

Excepciones

SocketZeroMQException *(*p. **??***)*	Si ocurre un error al configurar o enlazar el socket.

Reimplementado de **org.jogugi.berkeley.AbstractNode** (p. **??**).

Definición en la línea 196 del archivo Follower.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.SocketZeroMQException.↩ ErrorType.CONNECTION_ERROR, org.jogugi.berkeley.AbstractNode.context, org.jogugi.berkeley.Follower.logger, org.jogugi.berkeley.AbstractNode.name, org.jogugi.berkeley.AbstractNode.socket y org.jogugi.berkeley.Abstract↩ Node.startListening().

Referenciado por org.jogugi.berkeley.Main.main().

6.4.3. Documentación de los datos miembro

6.4.3.1. leaderAddress

`String org.jogugi.berkeley.Follower.leaderAddress [private]`

Dirección del nodo líder en formato `host:puerto`.

Definición en la línea 26 del archivo Follower.java.

Referenciado por org.jogugi.berkeley.Follower.handleProcess() y org.jogugi.berkeley.Follower.initializeNode().

6.4.3.2. logger

```
final Logger org.jogugi.berkeley.Follower.logger = LoggerFactory.getLogger(Follower.class)
[static], [private]
```

Logger para registrar eventos y mensajes del nodo seguidor.

Definición en la línea 31 del archivo Follower.java.

Referenciado por org.jogugi.berkeley.Follower.displayLeaderMessage(), org.jogugi.berkeley.Follower.getCurrent↩ Time(), org.jogugi.berkeley.Follower.handleProcess(), org.jogugi.berkeley.Follower.modSystemTime() y org.↩ jogugi.berkeley.Follower.startAlgorithm().

La documentación para esta clase fue generada a partir del siguiente fichero:

- **Follower.java**

6.5. Referencia de la Clase org.jogugi.berkeley.Config.FollowerConfig

Configuración del nodo seguidor.

Atributos públicos

- String **name**

  Nombre del nodo seguidor.
- String **address**

  Dirección del nodo seguidor.

6.5.1. Descripción detallada

Configuración del nodo seguidor.

Esta clase define las propiedades del nodo seguidor, incluyendo su nombre y dirección.

Definición en la línea 41 del archivo Config.java.

6.5.2. Documentación de los datos miembro

6.5.2.1. address

```
String org.jogugi.berkeley.Config.FollowerConfig.address
```

Dirección del nodo seguidor.

Este campo almacena la dirección de cada nodo seguidor, que se usará para establecer conexiones.

Definición en la línea 55 del archivo Config.java.

Referenciado por org.jogugi.berkeley.Main.main().

6.5.2.2. name

`String org.jogugi.berkeley.Config.FollowerConfig.name`

Nombre del nodo seguidor.

Este campo almacena el nombre de cada nodo seguidor en el sistema.

Definición en la línea 48 del archivo Config.java.

Referenciado por org.jogugi.berkeley.Main.main().

La documentación para esta clase fue generada a partir del siguiente fichero:

- **Config.java**

6.6. Referencia de la Clase org.jogugi.berkeley.FollowerInfo

Clase que representa información detallada de un nodo seguidor en el sistema.

Métodos públicos

- **FollowerInfo** (String address, String name, long localTime, long communicationTime, long nowTime)
  Constructor de la clase ***FollowerInfo*** (p. **??**).
- String **getAdrressFollower** ()
  Obtiene la dirección del seguidor.
- String **getName** ()
  Obtiene el nombre del seguidor.
- Date **getDateFollower** ()
  Obtiene la fecha y hora local del seguidor.
- long **getDiffTime** ()
  Obtiene la diferencia de tiempo entre el líder y el seguidor.
- long **getLocalTime** ()
  Obtiene la hora local del seguidor.
- long **getCommunicationTime** ()
  Obtiene el tiempo de comunicación entre el líder y el seguidor.
- long **getDelta** ()
  Obtiene la diferencia global de tiempo (delta).
- void **setDelta** (long delta)
  Establece la diferencia global de tiempo (delta) para el seguidor.
- **FollowerState** **getState** ()
  Obtiene el estado actual del seguidor.
- void **setState** (**FollowerState** state)
  Establece el estado del seguidor.
- String **toString** ()
  Representación en cadena del objeto ***FollowerInfo*** (p. **??**).

6.6.1. Descripción detallada

Clase que representa información detallada de un nodo seguidor en el sistema.

La clase **FollowerInfo** (p. **??**) encapsula información sobre un seguidor, incluyendo su nombre, dirección, tiempos relacionados con la comunicación, diferencias horarias y estado actual.

Definición en la línea 11 del archivo FollowerInfo.java.

6.6.2. Documentación del constructor y destructor

6.6.2.1. FollowerInfo()

```
org.jogugi.berkeley.FollowerInfo.FollowerInfo (
            String address,
            String name,
            long localTime,
            long communicationTime,
            long nowTime )
```

Constructor de la clase **FollowerInfo** (p. **??**).

Inicializa un objeto **FollowerInfo** (p. **??**) con los valores proporcionados.

Parámetros

address	Dirección del seguidor.
name	Nombre del seguidor.
localTime	Hora local del seguidor en milisegundos desde la época UNIX.
communicationTime	Tiempo de comunicación en milisegundos.
nowTime	Hora actual del líder en milisegundos desde la época UNIX.

Definición en la línea 51 del archivo FollowerInfo.java.

Hace referencia a org.jogugi.berkeley.FollowerState.REQUEST_NOT_SENT.

6.6.3. Documentación de las funciones miembro

6.6.3.1. getAdrressFollower()

```
String org.jogugi.berkeley.FollowerInfo.getAdrressFollower ( )
```

Obtiene la dirección del seguidor.

Definición en la línea 63 del archivo FollowerInfo.java.

Referenciado por org.jogugi.berkeley.Leader.sendTimeUpdateToFollower().

6.6.3.2. getCommunicationTime()

```
long org.jogugi.berkeley.FollowerInfo.getCommunicationTime ( )
```

Obtiene el tiempo de comunicación entre el líder y el seguidor.

Definición en la línea 88 del archivo FollowerInfo.java.

6.6.3.3. getDateFollower()

```
Date org.jogugi.berkeley.FollowerInfo.getDateFollower ( )
```

Obtiene la fecha y hora local del seguidor.

Definición en la línea 73 del archivo FollowerInfo.java.

6.6.3.4. getDelta()

```
long org.jogugi.berkeley.FollowerInfo.getDelta ( )
```

Obtiene la diferencia global de tiempo (delta).

Definición en la línea 93 del archivo FollowerInfo.java.

6.6.3.5. getDiffTime()

```
long org.jogugi.berkeley.FollowerInfo.getDiffTime ( )
```

Obtiene la diferencia de tiempo entre el líder y el seguidor.

Definición en la línea 78 del archivo FollowerInfo.java.

6.6.3.6. getLocalTime()

```
long org.jogugi.berkeley.FollowerInfo.getLocalTime ( )
```

Obtiene la hora local del seguidor.

Definición en la línea 83 del archivo FollowerInfo.java.

6.6.3.7. getName()

```
String org.jogugi.berkeley.FollowerInfo.getName ( )
```

Obtiene el nombre del seguidor.

Definición en la línea 68 del archivo FollowerInfo.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime().

6.6.3.8. getState()

```
FollowerState org.jogugi.berkeley.FollowerInfo.getState ( )
```

Obtiene el estado actual del seguidor.

Definición en la línea 106 del archivo FollowerInfo.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime() y org.jogugi.berkeley.Leader.↩ processFollowers().

6.6.3.9. setDelta()

```
void org.jogugi.berkeley.FollowerInfo.setDelta (
            long delta )
```

Establece la diferencia global de tiempo (delta) para el seguidor.

Parámetros

delta	Valor de la diferencia global de tiempo en milisegundos.

Definición en la línea 101 del archivo FollowerInfo.java.

6.6.3.10. setState()

```
void org.jogugi.berkeley.FollowerInfo.setState (
             FollowerState state )
```

Establece el estado del seguidor.

Parámetros

state	Nuevo estado del seguidor.

Definición en la línea 114 del archivo FollowerInfo.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime(), org.jogugi.berkeley.Leader.↩ processFollowers(), org.jogugi.berkeley.Leader.processFollowerTime() y org.jogugi.berkeley.Leader.sendTime↩ UpdateToFollower().

6.6.3.11. toString()

```
String org.jogugi.berkeley.FollowerInfo.toString ( )
```

Representación en cadena del objeto **`FollowerInfo`** (p. **??**).

Devuelve

Una descripción detallada de los atributos del seguidor.

Definición en la línea 124 del archivo FollowerInfo.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **FollowerInfo.java**

6.7. Referencia del enum org.jogugi.berkeley.FollowerState

Enumeración que define los posibles estados de un seguidor durante el proceso de actualización de la hora del sistema.

Atributos públicos

- **RESPONDED**

  El seguidor ha respondido correctamente a la solicitud del líder.
- **NO_RESPONSE**

  El seguidor no ha respondido a la solicitud dentro del tiempo esperado.
- **CONNECTION_ERROR**

  Hubo un error de conexión al intentar comunicarse con el seguidor.
- **REQUEST_NOT_SENT**

  La solicitud de actualización de la hora no fue enviada debido a algún error.
- **REQUEST_DELTA_SENT**

  La solicitud de actualización de la hora fue enviada al seguidor para actualizar su hora.
- **TIME_ERROR_SENT_UPDATE**

  La solicitud de actualización de la hora fue enviada pero hubo un error en ese envío.
- **TIME_UPDATED**

  La hora del sistema del seguidor se actualizó correctamente.

6.7.1. Descripción detallada

Enumeración que define los posibles estados de un seguidor durante el proceso de actualización de la hora del sistema.

Cada estado representa una etapa en la interacción entre el líder y el seguidor.

Definición en la línea 7 del archivo FollowerState.java.

6.7.2. Documentación de los datos miembro

6.7.2.1. CONNECTION_ERROR

`org.jogugi.berkeley.FollowerState.CONNECTION_ERROR`

Hubo un error de conexión al intentar comunicarse con el seguidor.

Definición en la línea 22 del archivo FollowerState.java.

6.7.2.2. NO_RESPONSE

`org.jogugi.berkeley.FollowerState.NO_RESPONSE`

El seguidor no ha respondido a la solicitud dentro del tiempo esperado.

Definición en la línea 17 del archivo FollowerState.java.

Referenciado por org.jogugi.berkeley.Leader.processFollowers() y org.jogugi.berkeley.Leader.processFollower↩ Time().

6.7.2.3. REQUEST_DELTA_SENT

`org.jogugi.berkeley.FollowerState.REQUEST_DELTA_SENT`

La solicitud de actualización de la hora fue enviada al seguidor para actualizar su hora.

Definición en la línea 32 del archivo FollowerState.java.

6.7.2.4. REQUEST_NOT_SENT

`org.jogugi.berkeley.FollowerState.REQUEST_NOT_SENT`

La solicitud de actualización de la hora no fue enviada debido a algún error.

Definición en la línea 27 del archivo FollowerState.java.

Referenciado por org.jogugi.berkeley.FollowerInfo.FollowerInfo(), org.jogugi.berkeley.Leader.processFollowers() y org.jogugi.berkeley.Leader.processFollowerTime().

6.7.2.5. RESPONDED

`org.jogugi.berkeley.FollowerState.RESPONDED`

El seguidor ha respondido correctamente a la solicitud del líder.

Definición en la línea 12 del archivo FollowerState.java.

Referenciado por org.jogugi.berkeley.Leader.processFollowers() y org.jogugi.berkeley.Leader.processFollower↩ Time().

6.7.2.6. TIME_ERROR_SENT_UPDATE

`org.jogugi.berkeley.FollowerState.TIME_ERROR_SENT_UPDATE`

La solicitud de actualización de la hora fue enviada pero hubo un error en ese envío.

Definición en la línea 36 del archivo FollowerState.java.

Referenciado por org.jogugi.berkeley.Leader.sendTimeUpdateToFollower().

6.7.2.7. TIME_UPDATED

`org.jogugi.berkeley.FollowerState.TIME_UPDATED`

La hora del sistema del seguidor se actualizó correctamente.

Definición en la línea 40 del archivo FollowerState.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime() y org.jogugi.berkeley.Leader.send↩ TimeUpdateToFollower().

La documentación para este enum ha sido generada a partir del siguiente fichero:

- **FollowerState.java**

6.8. Referencia de la Interfaz org.jogugi.berkeley.INode

Interfaz que define las operaciones básicas para un nodo en el sistema.

Heredado por **org.jogugi.berkeley.AbstractNode**.

Métodos públicos

- void **initializeNode** (String name, String address, int timeout)

  Inicializa el contexto y el socket para el nodo.
- void **initializeNode** (String name, String address, int timeout, HashMap< String, String > listNode↩ Addresses)

  Inicializa el contexto y el socket para el nodo, incluyendo una lista de direcciones de otros nodos.
- String **sendMessageSync** (String address, String message) throws SocketZeroMQException

  Envía un mensaje síncrono a un nodo y espera su respuesta.
- void **sendMessageAsync** (String address, String message) throws SocketZeroMQException

  Envía un mensaje de manera asincrónica a un nodo.
- void **startListening** () throws SocketZeroMQException

  Inicia el nodo en modo escucha, esperando recibir mensajes.
- void **startAlgorithm** () throws SocketZeroMQException, InterruptedException, ExecutionException

  Inicia un algoritmo especial para el nodo (puede ser utilizado para procesos específicos como el cálculo del tiempo o sincronización).
- void **close** () throws SocketZeroMQException

  Cierra los recursos del nodo, liberando cualquier conexión o socket abierto.

6.8.1. Descripción detallada

Interfaz que define las operaciones básicas para un nodo en el sistema.

La interfaz **`INode`** (p. **??**) establece los métodos que deben implementarse para inicializar un nodo, enviar mensajes (sincrónicos y asincrónicos), iniciar el algoritmo del nodo y cerrar sus recursos.

Definición en la línea 12 del archivo INode.java.

6.8.2. Documentación de las funciones miembro

6.8.2.1. close()

`void org.jogugi.berkeley.INode.close ( ) throws` **`SocketZeroMQException`**

Cierra los recursos del nodo, liberando cualquier conexión o socket abierto.

Excepciones

SocketZeroMQException *(p. **??**)*	Si ocurre un error al cerrar el nodo.

Implementado en **org.jogugi.berkeley.Leader** (p. **??**) y **org.jogugi.berkeley.AbstractNode** (p. **??**).

6.8.2.2. initializeNode() [1/2]

```
void org.jogugi.berkeley.INode.initializeNode (
            String name,
            String address,
            int timeout )
```

Inicializa el contexto y el socket para el nodo.

Parámetros

name	Nombre del nodo.
address	Dirección del nodo en formato `host:puerto`.
timeout	Tiempo de espera en milisegundos.

Implementado en **org.jogugi.berkeley.AbstractNode** (p. **??**).

6.8.2.3. initializeNode() [2/2]

```
void org.jogugi.berkeley.INode.initializeNode (
            String name,
            String address,
            int timeout,
            HashMap< String, String > listNodeAddresses )
```

Inicializa el contexto y el socket para el nodo, incluyendo una lista de direcciones de otros nodos.

Parámetros

name	Nombre del nodo.
address	Dirección del nodo en formato `host:puerto`.
timeout	Tiempo de espera en milisegundos.
listNodeAddresses	Mapa con las direcciones de otros nodos.

Implementado en **org.jogugi.berkeley.AbstractNode** (p. **??**) y **org.jogugi.berkeley.Leader** (p. **??**).

6.8.2.4. sendMessageAsync()

```
void org.jogugi.berkeley.INode.sendMessageAsync (
            String address,
            String message ) throws  SocketZeroMQException
```

Envía un mensaje de manera asincrónica a un nodo.

Parámetros

address	Dirección del nodo receptor en formato `host:puerto`.
message	Mensaje a enviar.

Excepciones

SocketZeroMQException *(p. **??**)*	Si ocurre un error durante la comunicación.

Implementado en **org.jogugi.berkeley.AbstractNode** (p. **??**).

6.8.2.5. sendMessageSync()

```
String org.jogugi.berkeley.INode.sendMessageSync (
            String address,
            String message ) throws  SocketZeroMQException
```

Envía un mensaje síncrono a un nodo y espera su respuesta.

Parámetros

address	Dirección del nodo receptor en formato `host:puerto`.
message	Mensaje a enviar.

Devuelve

Respuesta recibida del nodo.

Excepciones

SocketZeroMQException *(p. **??**)*	Si ocurre un error durante la comunicación.

Implementado en **org.jogugi.berkeley.AbstractNode** (p. **??**).

6.8.2.6. startAlgorithm()

```
void org.jogugi.berkeley.INode.startAlgorithm ( ) throws  SocketZeroMQException, Interrupted↩
Exception, ExecutionException
```

Inicia un algoritmo especial para el nodo (puede ser utilizado para procesos específicos como el cálculo del tiempo o sincronización).

Excepciones

SocketZeroMQException *(p. **??**)*	Si ocurre un error durante la ejecución del algoritmo.
InterruptedException	Si el hilo es interrumpido durante la ejecución.
ExecutionException	Si ocurre un error en la ejecución del algoritmo.

Implementado en **org.jogugi.berkeley.AbstractNode** (p. **??**), **org.jogugi.berkeley.Follower** (p. **??**) y **org.↩ jogugi.berkeley.Leader** (p. **??**).

6.8.2.7. startListening()

`void org.jogugi.berkeley.INode.startListening ( ) throws` **`SocketZeroMQException`**

Inicia el nodo en modo escucha, esperando recibir mensajes.

Excepciones

SocketZeroMQException *(p. **??**)*	Si ocurre un error al iniciar el socket.

Implementado en **org.jogugi.berkeley.AbstractNode** (p. **??**).

La documentación para este interfaz fue generada a partir del siguiente fichero:

- **INode.java**

6.9. Referencia de la Clase org.jogugi.berkeley.Leader

Representa un nodo líder en el sistema.

Herencias **org.jogugi.berkeley.AbstractNode**.

Métodos públicos

- void **initializeNode** (String **address**, String **name**, int **timeout**, HashMap< String, String > followers)

  Inicializa el nodo líder con la dirección, nombre, timeout y lista de seguidores.
- void **startAlgorithm** () throws InterruptedException, ExecutionException

  Inicia el algoritmo de sincronización de relojes entre el líder y los seguidores.
- void **close** () throws SocketZeroMQException

  Cierra el socket del líder y el contexto de ZeroMQ.

Métodos protegidos

- String **handleProcess** (String message)

  Función que debe sobrecargarse e implementar la lógica para procesar el mensaje recibido y realizar la lógica de negocio correspondiente.

Métodos privados

- void **processFollowers** () throws InterruptedException

  Procesa las respuestas de todos los seguidores en paralelo.
- **FollowerInfo** **processFollowerTime** (String followerName, String followerAddress, long T0)

  Procesa la solicitud de hora a un seguidor y calcula el tiempo de comunicación.
- long **calculateDeltaTimeDifference** ()

  Calcula la diferencia de tiempo (delta) entre el tiempo local y los tiempos de los seguidores válidos.
- void **callFollowersWithUpdatedTime** (long delta) throws ExecutionException

  Envía actualizaciones de tiempo a los seguidores en paralelo utilizando un `ExecutorService`.
- **FollowerInfo** **sendTimeUpdateToFollower** (**FollowerInfo** follower, long delta)

  Envía una solicitud para actualizar el tiempo de un seguidor en el sistema.
- void **sendCloseMessage** (String followerAddress)

  Envía un mensaje de cierre a un seguidor especificado.
- void **sendCloseMessagesToFollowers** () throws ExecutionException

  Envía un mensaje de cierre a todos los seguidores que respondieron con éxito.
- String **getFollowerName** (Future< **FollowerInfo** > future, Map< String, Future< **FollowerInfo** >> futureMap)

  Obtiene el nombre del seguidor asociado al futuro pasado como argumento.
- void **printResults** ()

  Imprime los resultados del algoritmo, mostrando los diferentes estados de los seguidores.

Atributos privados

- ConcurrentHashMap< String, **FollowerInfo** > **unreachableFollowers** = new ConcurrentHashMap<>()

  Mapa de seguidores no alcanzables.
- ConcurrentHashMap< String, **FollowerInfo** > **successfulFollowers** = new ConcurrentHashMap<>()

  Mapa de seguidores exitosos.
- ConcurrentHashMap< String, **FollowerInfo** > **nonRespondingFollowers** = new ConcurrentHash↩ Map<>()

  Mapa de seguidores que no respondieron.
- ConcurrentHashMap< String, **FollowerInfo** > **timeUpdatedFollowers** = new ConcurrentHashMap<>()

  Mapa de seguidores con tiempo actualizado.
- ConcurrentHashMap< String, **FollowerInfo** > **failedFollowers** = new ConcurrentHashMap<>()

  Mapa de seguidores fallidos.

Atributos privados estáticos

- static final Logger **logger** = LoggerFactory.getLogger(Leader.class)

  *Logger para registrar la actividad de la clase **Leader** (p. **??**).*

Otros miembros heredados

6.9.1. Descripción detallada

Representa un nodo líder en el sistema.

La clase **Leader** (p. **??**) gestiona el estado de los seguidores (followers) en el sistema. Administra los seguidores que están alcanzables, exitosos, no responden, actualizados o fallidos, y se comunica con ellos a través de mensajes.

Definición en la línea 33 del archivo Leader.java.

6.9.2. Documentación de las funciones miembro

6.9.2.1. calculateDeltaTimeDifference()

```
long org.jogugi.berkeley.Leader.calculateDeltaTimeDifference ( )  [private]
```

Calcula la diferencia de tiempo (delta) entre el tiempo local y los tiempos de los seguidores válidos.

Esta función calcula un promedio de los tiempos ajustados de los seguidores que han enviado respuestas válidas, y calcula la diferencia entre el tiempo local del líder y el promedio calculado. El valor de delta se usa para sincronizar el tiempo entre el líder y los seguidores. Si no se reciben respuestas válidas de los seguidores, se retorna 0.

Devuelve

La diferencia de tiempo (delta) calculada en milisegundos. Si no hay seguidores válidos, retorna 0.

Definición en la línea 361 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.Leader.logger y org.jogugi.berkeley.Leader.successfulFollowers.

Referenciado por org.jogugi.berkeley.Leader.startAlgorithm().

6.9.2.2. callFollowersWithUpdatedTime()

```
void org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime (
            long delta ) throws ExecutionException  [private]
```

Envía actualizaciones de tiempo a los seguidores en paralelo utilizando un `ExecutorService`.

Esta función recorre los seguidores exitosos y envía una solicitud de actualización de tiempo a cada uno de ellos. Las actualizaciones se realizan en paralelo utilizando un pool de hilos. Se gestionan las respuestas de cada seguidor y se espera un resultado con un tiempo de espera definido. Si no se recibe respuesta a tiempo, se registra un mensaje de advertencia.

Parámetros

delta	El valor diferencial de tiempo que se debe aplicar a cada seguidor.

Excepciones

ExecutionException	Si ocurre un error durante la ejecución de alguna de las tareas en paralelo.

Definición en la línea 410 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.Leader.failedFollowers, org.jogugi.berkeley.Leader.getFollowerName(), org.jogugi.berkeley.FollowerInfo.getName(), org.jogugi.berkeley.FollowerInfo.getState(), org.jogugi.berkeley.↩

Leader.logger, org.jogugi.berkeley.AbstractNode.nodeAddresses, org.jogugi.berkeley.Leader.sendTimeUpdate↩ ToFollower(), org.jogugi.berkeley.FollowerInfo.setState(), org.jogugi.berkeley.Leader.successfulFollowers, org.↩ jogugi.berkeley.FollowerState.TIME_UPDATED y org.jogugi.berkeley.Leader.timeUpdatedFollowers.

Referenciado por org.jogugi.berkeley.Leader.startAlgorithm().

6.9.2.3. close()

```
void org.jogugi.berkeley.Leader.close ( ) throws  SocketZeroMQException
```

Cierra el socket del líder y el contexto de ZeroMQ.

Este método se encarga de cerrar el socket del líder si está abierto y el contexto de ZeroMQ, liberando los recursos asociados. Si el socket o el contexto no están inicializados, se emite un mensaje de advertencia. Si ocurre un error durante el cierre, se registra un mensaje de error correspondiente.

Excepciones

SocketZeroMQException *(*p. **??***)*	Si ocurre un error durante el cierre del socket.

Reimplementado de **org.jogugi.berkeley.AbstractNode** (p. **??**).

Definición en la línea 688 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.context, org.jogugi.berkeley.Leader.logger y org.jogugi.↩ berkeley.AbstractNode.socket.

Referenciado por org.jogugi.berkeley.Main.main().

6.9.2.4. getFollowerName()

```
String org.jogugi.berkeley.Leader.getFollowerName (
            Future<  FollowerInfo > future,
            Map< String, Future<  FollowerInfo >> futureMap )  [private]
```

Obtiene el nombre del seguidor asociado al futuro pasado como argumento.

Este método busca el nombre de un seguidor en el mapa de futuros (`futureMap`) en función del objeto `Future<`**`FollowerInfo`** (p. **??**)`>`. Si el futuro corresponde a uno de los seguidores registrados en el mapa, se devuelve el nombre asociado a ese seguidor. Si no se encuentra el seguidor, se devuelve un nombre por defecto: "Unknown".

Parámetros

future	El objeto `Future<`**`FollowerInfo`** (p. **??**)`>` que se está buscando en el mapa de futuros.
futureMap	El mapa de futuros (`Map<String, Future<`**`FollowerInfo`** (p. **??**)`>>`) que asocia nombres de seguidores con sus futuros.

Devuelve

El nombre del seguidor asociado al futuro, o "Unknown" si no se encuentra el seguidor.

Definición en la línea 722 del archivo Leader.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime() y org.jogugi.berkeley.Leader.↩ processFollowers().

6.9.2.5. handleProcess()

```
String org.jogugi.berkeley.Leader.handleProcess (
            String message )  [protected]
```

Función que debe sobrecargarse e implementar la lógica para procesar el mensaje recibido y realizar la lógica de negocio correspondiente.

Parámetros

message	El mensaje que se recibirá para ser procesado. Este mensaje podría ser de cualquier tipo, dependiendo de la implementación futura.

Devuelve

El resultado del procesamiento del mensaje. Actualmente siempre retorna null, pero se espera que devuelva un resultado significativo en el futuro según la lógica de procesamiento.

Excepciones

Exception	Si ocurre algún error durante el procesamiento del mensaje, se lanzará una excepción.

Reimplementado de **org.jogugi.berkeley.AbstractNode** (p. **??**).

Definición en la línea 784 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.Leader.logger.

6.9.2.6. initializeNode()

```
void org.jogugi.berkeley.Leader.initializeNode (
            String address,
            String name,
            int timeout,
            HashMap< String, String > followers )
```

Inicializa el nodo líder con la dirección, nombre, timeout y lista de seguidores.

Este método configura el nodo líder. Llama al método de inicialización de la clase base para configurar el contexto y los parámetros comunes del nodo.

Parámetros

address	Dirección del nodo líder.
name	Nombre del nodo líder.
timeout	Tiempo de espera en milisegundos para las operaciones de comunicación.
followers	Lista de direcciones de los seguidores que están conectados al líder.

Reimplementado de **org.jogugi.berkeley.AbstractNode** (p. **??**).

Definición en la línea 78 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.Leader.logger, org.jogugi.↩ berkeley.AbstractNode.name y org.jogugi.berkeley.AbstractNode.timeout.

Referenciado por org.jogugi.berkeley.Main.main().

6.9.2.7. printResults()

```
void org.jogugi.berkeley.Leader.printResults ( )  [private]
```

Imprime los resultados del algoritmo, mostrando los diferentes estados de los seguidores.

Este método imprime tres grupos de seguidores:

- Seguidores inalcanzables: Aquellos que no pudieron ser contactados.
- Seguidores que no respondieron a tiempo: Aquellos que no enviaron su respuesta dentro del plazo esperado.
- Seguidores que respondieron correctamente: Aquellos que respondieron con éxito y dentro del tiempo esperado.

Definición en la línea 739 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.Leader.failedFollowers, org.jogugi.berkeley.Leader.logger, org.jogugi.↩ berkeley.Leader.nonRespondingFollowers, org.jogugi.berkeley.Leader.successfulFollowers, org.jogugi.berkeley.↩ Leader.timeUpdatedFollowers y org.jogugi.berkeley.Leader.unreachableFollowers.

Referenciado por org.jogugi.berkeley.Leader.startAlgorithm().

6.9.2.8. processFollowers()

```
void org.jogugi.berkeley.Leader.processFollowers ( ) throws InterruptedException  [private]
```

Procesa las respuestas de todos los seguidores en paralelo.

Este método solicita la hora a todos los seguidores y luego procesa las respuestas a medida que llegan. Los seguidores son procesados en paralelo utilizando un `ExecutorService` y `ExecutorCompletionService` para manejar la ejecución de múltiples hilos.

Fases:

- **Fase 1.1**: Lanzar tareas concurrentes para solicitar la hora a todos los seguidores.
- **Fase 1.2**: Procesar los resultados conforme se completan.

Los resultados de las respuestas de los seguidores se almacenan en dos mapas:

- **successfulFollowers**: Los seguidores que respondieron correctamente.
- **nonRespondingFollowers**: Los seguidores que no respondieron o respondieron incorrectamente.

Excepciones

InterruptedException	Si el hilo principal es interrumpido mientras espera los resultados.

Definición en la línea 149 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.address, org.jogugi.berkeley.Leader.getFollowerName(), org.↩ jogugi.berkeley.FollowerInfo.getState(), org.jogugi.berkeley.Leader.logger, org.jogugi.berkeley.AbstractNode.name, org.jogugi.berkeley.FollowerState.NO_RESPONSE, org.jogugi.berkeley.AbstractNode.nodeAddresses, org.↩ jogugi.berkeley.Leader.nonRespondingFollowers, org.jogugi.berkeley.Leader.processFollowerTime(), org.jogugi.↩ berkeley.FollowerState.REQUEST_NOT_SENT, org.jogugi.berkeley.FollowerState.RESPONDED, org.jogugi.↩ berkeley.FollowerInfo.setState(), org.jogugi.berkeley.Leader.successfulFollowers, org.jogugi.berkeley.Abstract↩ Node.timeout y org.jogugi.berkeley.Leader.unreachableFollowers.

Referenciado por org.jogugi.berkeley.Leader.startAlgorithm().

6.9.2.9. processFollowerTime()

```
FollowerInfo org.jogugi.berkeley.Leader.processFollowerTime (
            String followerName,
            String followerAddress,
            long T0 )  [private]
```

Procesa la solicitud de hora a un seguidor y calcula el tiempo de comunicación.

Este método establece una conexión con un seguidor a través de un socket de tipo REQ/REP utilizando ZMQ. Envía una solicitud para obtener la hora local del seguidor y calcula el tiempo que tarda en recibir la respuesta. Si se recibe una respuesta válida, devuelve un objeto **`FollowerInfo`** (p. **??**) con la información del seguidor. Si no se recibe respuesta o ocurre un error, se registra la incidencia y devuelve una respuesta con valores por defecto.

El proceso de solicitud de tiempo se realiza de la siguiente manera:

- Se crea un socket de tipo REQ para realizar la solicitud de hora.
- Se envía una solicitud en formato JSON al seguidor con la hora inicial (`T0`).
- Se espera la respuesta con la hora local del seguidor.
- Si la respuesta es válida, se calcula el tiempo de comunicación y se devuelve el resultado.
- Si no se recibe respuesta o ocurre un error, se gestionan estos casos mediante valores predeterminados.

Parámetros

followerName	Nombre del seguidor al que se le solicita la hora.
followerAddress	Dirección del seguidor al que se conecta.
T0	Tiempo de referencia desde el cual se mide el tiempo de comunicación.

Devuelve

Un objeto **`FollowerInfo`** (p. **??**) con la hora del seguidor, el tiempo de comunicación, y otros datos relevantes.

Excepciones

JsonProcessingException	Si ocurre un error al procesar el JSON.

Definición en la línea 267 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.context, org.jogugi.berkeley.Leader.logger, org.jogugi.↩ berkeley.AbstractNode.name, org.jogugi.berkeley.FollowerState.NO_RESPONSE, org.jogugi.berkeley.Abstract↩ Node.nodeAddresses, org.jogugi.berkeley.FollowerState.REQUEST_NOT_SENT, org.jogugi.berkeley.Follower↩ State.RESPONDED, org.jogugi.berkeley.FollowerInfo.setState() y org.jogugi.berkeley.AbstractNode.timeout.

Referenciado por org.jogugi.berkeley.Leader.processFollowers().

6.9.2.10. sendCloseMessage()

```
void org.jogugi.berkeley.Leader.sendCloseMessage (
             String followerAddress )  [private]
```

Envía un mensaje de cierre a un seguidor especificado.

Este método crea una solicitud de cierre para desconectar un seguidor del líder. El mensaje se envía de manera síncrona a la dirección del seguidor proporcionada. Si el seguidor responde, se registra la respuesta, indicando si la operación de cierre fue realizada correctamente.

Parámetros

followerAddress	Dirección del seguidor al que se enviará el mensaje de cierre.

Nota

Si no se recibe respuesta del seguidor, se emite una advertencia en los logs.

Definición en la línea 566 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.context, org.jogugi.berkeley.Leader.logger, org.jogugi.↩ berkeley.AbstractNode.name y org.jogugi.berkeley.AbstractNode.timeout.

Referenciado por org.jogugi.berkeley.Leader.sendCloseMessagesToFollowers().

6.9.2.11. sendCloseMessagesToFollowers()

```
void org.jogugi.berkeley.Leader.sendCloseMessagesToFollowers ( ) throws ExecutionException
[private]
```

Envía un mensaje de cierre a todos los seguidores que respondieron con éxito.

Este método utiliza un `ExecutorService` con un grupo de hilos para enviar un mensaje de cierre a cada uno de los seguidores que han respondido correctamente. El mensaje de cierre es enviado de manera asíncrona, y el método espera las respuestas o el tiempo de espera para cada uno de los seguidores.

Para cada seguidor:

- Se envía un mensaje de cierre en un hilo separado.
- Se espera hasta que el mensaje se haya enviado correctamente o se alcance un tiempo de espera.
- Se registran las respuestas o se manejan los errores (por ejemplo, tiempo de espera alcanzado).

Excepciones

ExecutionException	Si ocurre un error durante la ejecución de las tareas asíncronas.

Definición en la línea 625 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.Leader.logger, org.jogugi.berkeley.Leader.sendCloseMessage(), org.↩ jogugi.berkeley.Leader.successfulFollowers y org.jogugi.berkeley.AbstractNode.timeout.

Referenciado por org.jogugi.berkeley.Leader.startAlgorithm().

6.9.2.12. sendTimeUpdateToFollower()

```
FollowerInfo org.jogugi.berkeley.Leader.sendTimeUpdateToFollower (
            FollowerInfo follower,
           long delta )  [private]
```

Envía una solicitud para actualizar el tiempo de un seguidor en el sistema.

Esta función establece una conexión de tipo `REQ` con un seguidor a través de ZeroMQ, enviando una solicitud de actualización de tiempo con un valor delta proporcionado. Si la respuesta es recibida, se procesa y se muestra en el log. Si no se recibe respuesta o ocurre un error, se registra un mensaje adecuado.

Parámetros

followerAddress	Dirección del seguidor a donde enviar la solicitud.
delta	El valor diferencial de tiempo que se debe aplicar en el seguidor.

Definición en la línea 493 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.AbstractNode.context, org.jogugi.berkeley.FollowerInfo.getAdrressFollower(),

org.jogugi.berkeley.Leader.logger, org.jogugi.berkeley.AbstractNode.name, org.jogugi.berkeley.FollowerInfo.set↩ State(), org.jogugi.berkeley.FollowerState.TIME_ERROR_SENT_UPDATE, org.jogugi.berkeley.FollowerState.TI↩ ME_UPDATED y org.jogugi.berkeley.AbstractNode.timeout.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime().

6.9.2.13. startAlgorithm()

```
void org.jogugi.berkeley.Leader.startAlgorithm ( ) throws InterruptedException, Execution↩
Exception
```

Inicia el algoritmo de sincronización de relojes entre el líder y los seguidores.

El algoritmo se ejecuta en varias fases:

- **Fase 1**: Solicitar la hora a todos los seguidores en paralelo.
- **Fase 2**: Calcular el delta entre los tiempos de los seguidores obteniendo la media de esos tiempos (delta).
- **Fase 3**: Actualizar los relojes de los seguidores con el delta calculado.
- **Fase 4**: Enviar un mensaje de cierre a los seguidores.
- **Fase 5**: Mostrar los resultados finales de la sincronización.

El método utiliza un pool de hilos para ejecutar la solicitud de hora a cada seguidor en paralelo. Se gestionan las respuestas de cada seguidor y se realiza el cálculo necesario para sincronizar los relojes. En esta versión, para simplificar el código, generramos tantos hilos en el pool como seguidores teenmos.

Excepciones

InterruptedException	Si el hilo es interrumpido durante la ejecución.
ExecutionException	Si ocurre un error durante la ejecución de las tareas en paralelo.

Reimplementado de **org.jogugi.berkeley.AbstractNode** (p. **??**).

Definición en la línea 101 del archivo Leader.java.

Hace referencia a org.jogugi.berkeley.Leader.calculateDeltaTimeDifference(), org.jogugi.berkeley.Leader.call↩ FollowersWithUpdatedTime(), org.jogugi.berkeley.Leader.logger, org.jogugi.berkeley.Leader.printResults(), org.↩ jogugi.berkeley.Leader.processFollowers() y org.jogugi.berkeley.Leader.sendCloseMessagesToFollowers().

Referenciado por org.jogugi.berkeley.Main.main().

6.9.3. Documentación de los datos miembro

6.9.3.1. failedFollowers

```
ConcurrentHashMap<String, FollowerInfo> org.jogugi.berkeley.Leader.failedFollowers = new
ConcurrentHashMap<>() [private]
```

Mapa de seguidores fallidos.

Definición en la línea 63 del archivo Leader.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime() y org.jogugi.berkeley.Leader.print↩ Results().

6.9.3.2. logger

```
final Logger org.jogugi.berkeley.Leader.logger = LoggerFactory.getLogger(Leader.class) [static],
[private]
```

Logger para registrar la actividad de la clase **Leader** (p. **??**).

Definición en la línea 38 del archivo Leader.java.

Referenciado por org.jogugi.berkeley.Leader.calculateDeltaTimeDifference(), org.jogugi.berkeley.Leader.call↩ FollowersWithUpdatedTime(), org.jogugi.berkeley.Leader.close(), org.jogugi.berkeley.Leader.handleProcess(), org.jogugi.berkeley.Leader.initializeNode(), org.jogugi.berkeley.Leader.printResults(), org.jogugi.berkeley.Leader.↩ processFollowers(), org.jogugi.berkeley.Leader.processFollowerTime(), org.jogugi.berkeley.Leader.sendClose↩ Message(), org.jogugi.berkeley.Leader.sendCloseMessagesToFollowers(), org.jogugi.berkeley.Leader.sendTime↩ UpdateToFollower() y org.jogugi.berkeley.Leader.startAlgorithm().

6.9.3.3. nonRespondingFollowers

```
ConcurrentHashMap<String, FollowerInfo> org.jogugi.berkeley.Leader.nonRespondingFollowers =
new ConcurrentHashMap<>() [private]
```

Mapa de seguidores que no respondieron.

Definición en la línea 53 del archivo Leader.java.

Referenciado por org.jogugi.berkeley.Leader.printResults() y org.jogugi.berkeley.Leader.processFollowers().

6.9.3.4. successfulFollowers

```
ConcurrentHashMap<String, FollowerInfo> org.jogugi.berkeley.Leader.successfulFollowers = new
ConcurrentHashMap<>() [private]
```

Mapa de seguidores exitosos.

Definición en la línea 48 del archivo Leader.java.

Referenciado por org.jogugi.berkeley.Leader.calculateDeltaTimeDifference(), org.jogugi.berkeley.Leader.call↩ FollowersWithUpdatedTime(), org.jogugi.berkeley.Leader.printResults(), org.jogugi.berkeley.Leader.process↩ Followers() y org.jogugi.berkeley.Leader.sendCloseMessagesToFollowers().

6.9.3.5. timeUpdatedFollowers

```
ConcurrentHashMap<String, FollowerInfo> org.jogugi.berkeley.Leader.timeUpdatedFollowers = new
ConcurrentHashMap<>() [private]
```

Mapa de seguidores con tiempo actualizado.

Definición en la línea 58 del archivo Leader.java.

Referenciado por org.jogugi.berkeley.Leader.callFollowersWithUpdatedTime() y org.jogugi.berkeley.Leader.print↩ Results().

6.9.3.6. unreachableFollowers

```
ConcurrentHashMap<String, FollowerInfo> org.jogugi.berkeley.Leader.unreachableFollowers = new
ConcurrentHashMap<>() [private]
```

Mapa de seguidores no alcanzables.

Definición en la línea 43 del archivo Leader.java.

Referenciado por org.jogugi.berkeley.Leader.printResults() y org.jogugi.berkeley.Leader.processFollowers().

La documentación para esta clase fue generada a partir del siguiente fichero:

- **Leader.java**

6.10. Referencia de la Clase org.jogugi.berkeley.Config.LeaderConfig

Configuración del nodo líder.

Atributos públicos

- String **name**
  Nombre del nodo líder.
- String **address**
  Dirección del nodo líder.

6.10.1. Descripción detallada

Configuración del nodo líder.

Esta clase define las propiedades del nodo líder, incluyendo su nombre y dirección.

Definición en la línea 19 del archivo Config.java.

6.10.2. Documentación de los datos miembro

6.10.2.1. address

`String org.jogugi.berkeley.Config.LeaderConfig.address`

Dirección del nodo líder.

Este campo almacena la dirección del nodo líder, que se usará para establecer conexiones.

Definición en la línea 33 del archivo Config.java.

Referenciado por org.jogugi.berkeley.Main.main().

6.10.2.2. name

`String org.jogugi.berkeley.Config.LeaderConfig.name`

Nombre del nodo líder.

Este campo almacena el nombre del nodo líder en el sistema.

Definición en la línea 26 del archivo Config.java.

Referenciado por org.jogugi.berkeley.Main.main().

La documentación para esta clase fue generada a partir del siguiente fichero:

- **Config.java**

6.11. Referencia de la Clase org.jogugi.berkeley.LoggerManager

La clase.

Métodos públicos estáticos

- static Logger **getLogger** (Class<?> clazz)

  Obtiene el logger correspondiente a la clase que invoca este método.

6.11.1. Descripción detallada

La clase.
`LoggerManager`

es responsable de gestionar los loggers para las clases que lo invocan. Utiliza la biblioteca **slf4j** (p. **??**) para crear instancias de loggers.

Esta clase proporciona un único método estático que devuelve el logger correspondiente a la clase que lo invoca.

Autor

Tu Nombre

Versión

1.0

Desde

2024-12-07

Definición en la línea 19 del archivo LoggerManager.java.

6.11.2. Documentación de las funciones miembro

6.11.2.1. getLogger()

```
static Logger org.jogugi.berkeley.LoggerManager.getLogger (
            Class<?> clazz )  [static]
```

Obtiene el logger correspondiente a la clase que invoca este método.

Parámetros

clazz	La clase para la cual se desea obtener el logger.

Devuelve

Un objeto **Logger** (p. **??**) correspondiente a la clase
`clazz`

.

Ver también

Logger

Definición en la línea 28 del archivo LoggerManager.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **LoggerManager.java**

6.12. Referencia de la Clase org.jogugi.berkeley.Main

Clase principal que gestiona la inicialización y ejecución del líder y los seguidores en el sistema.

Métodos públicos estáticos

- static void **main** (String[ ] args) throws IOException, InterruptedException, SocketZeroMQException
  Método principal que inicia el proceso de inicialización y ejecución del sistema.

Atributos privados estáticos

- static final Logger **logger** = LoggerFactory.getLogger(Main.class)

6.12.1. Descripción detallada

Clase principal que gestiona la inicialización y ejecución del líder y los seguidores en el sistema.

Esta clase se encarga de cargar la configuración desde un archivo JSON, inicializar los nodos de los seguidores y el líder, y coordinar la ejecución de los algoritmos en el sistema. Además, gestiona las excepciones y asegura que el flujo de trabajo continúe o se detenga dependiendo de la correcta inicialización de los componentes.

Los pasos clave en este proceso son:

1. Cargar la configuración desde un archivo JSON.
2. Crear e inicializar los seguidores y el líder.
3. Iniciar los algoritmos en los seguidores y el líder.
4. Gestionar el tiempo de espera para la sincronización.
5. Cerrar el nodo del líder de forma segura.

Autor

[Tu nombre]

Versión

1.0

Desde

[Fecha de creación]

Definición en la línea 31 del archivo Main.java.

6.12.2. Documentación de las funciones miembro

6.12.2.1. main()

```
static void org.jogugi.berkeley.Main.main (
            String[] args ) throws IOException, InterruptedException,  SocketZeroMQException
[static]
```

Método principal que inicia el proceso de inicialización y ejecución del sistema.

Parámetros

args	Argumentos de línea de comandos. No se utilizan en esta implementación.

Excepciones

IOException	Si ocurre un error al leer el archivo de configuración o durante la inicialización de los nodos.
InterruptedException	Si la ejecución es interrumpida durante los períodos de espera.
SocketZeroMQException *(*p. **??***)*	Si ocurre un error al intentar establecer comunicación a través de ZeroMQ.

Definición en la línea 42 del archivo Main.java.

Hace referencia a org.jogugi.berkeley.Config.LeaderConfig.address, org.jogugi.berkeley.Config.FollowerConfig.↩ address, org.jogugi.berkeley.Leader.close(), org.jogugi.berkeley.Config.followers, org.jogugi.berkeley.Follower.↩ initializeNode(), org.jogugi.berkeley.Leader.initializeNode(), org.jogugi.berkeley.Config.leader, org.jogugi.↩ berkeley.Main.logger, org.jogugi.berkeley.Config.LeaderConfig.name, org.jogugi.berkeley.Config.FollowerConfig.↩ name, org.jogugi.berkeley.Leader.startAlgorithm(), org.jogugi.berkeley.Follower.startAlgorithm() y org.jogugi.↩ berkeley.Config.timeout.

6.12.3. Documentación de los datos miembro

6.12.3.1. logger

```
final Logger org.jogugi.berkeley.Main.logger = LoggerFactory.getLogger(Main.class)  [static],
[private]
```

Definición en la línea 32 del archivo Main.java.

Referenciado por org.jogugi.berkeley.Main.main().

La documentación para esta clase fue generada a partir del siguiente fichero:

- **Main.java**

6.13. Referencia de la Clase org.jogugi.berkeley.SocketZeroMQException

Excepción personalizada para errores en la operación de sockets de ZeroMQ.

Herencias Exception.

Clases

- enum **ErrorType**
  Enumeración de los tipos de errores posibles.

Métodos públicos

- **SocketZeroMQException** (String message, **ErrorType errorType**)
  Constructor de la excepción.
- **ErrorType getErrorType** ()
  Obtiene el tipo de error asociado a esta excepción.

Atributos privados

- **ErrorType errorType**
  Tipo de error que ocurrió.

Atributos privados estáticos

- static final long **serialVersionUID** = 1L

6.13.1. Descripción detallada

Excepción personalizada para errores en la operación de sockets de ZeroMQ.

Esta clase extiende de `Exception` y se utiliza para manejar errores específicos que ocurren durante las operaciones de sockets utilizando ZeroMQ. Los tipos de errores cubiertos son relacionados con la conexión, el envío y la recepción de mensajes.

Definición en la línea 10 del archivo SocketZeroMQException.java.

6.13.2. Documentación del constructor y destructor

6.13.2.1. SocketZeroMQException()

```
org.jogugi.berkeley.SocketZeroMQException.SocketZeroMQException (
            String message,
             ErrorType errorType )
```

Constructor de la excepción.

Parámetros

message	El mensaje de error que describe el problema ocurrido.
errorType	El tipo de error que ocurrió, basado en la enumeración **ErrorType** (p. **??**).

Definición en la línea 37 del archivo SocketZeroMQException.java.

Hace referencia a org.jogugi.berkeley.SocketZeroMQException.errorType.

6.13.3. Documentación de las funciones miembro

6.13.3.1. getErrorType()

ErrorType org.jogugi.berkeley.SocketZeroMQException.getErrorType ( )

Obtiene el tipo de error asociado a esta excepción.

Devuelve

El tipo de error que se produjo durante la operación de ZeroMQ.

Definición en la línea 47 del archivo SocketZeroMQException.java.

Hace referencia a org.jogugi.berkeley.SocketZeroMQException.errorType.

6.13.4. Documentación de los datos miembro

6.13.4.1. errorType

ErrorType org.jogugi.berkeley.SocketZeroMQException.errorType [private]

Tipo de error que ocurrió.

Definición en la línea 29 del archivo SocketZeroMQException.java.

Referenciado por org.jogugi.berkeley.SocketZeroMQException.getErrorType() y org.jogugi.berkeley.SocketZeroM↩QException.SocketZeroMQException().

6.13.4.2. serialVersionUID

final long org.jogugi.berkeley.SocketZeroMQException.serialVersionUID = 1L [static], [private]

Definición en la línea 15 del archivo SocketZeroMQException.java.

La documentación para esta clase fue generada a partir del siguiente fichero:

- **SocketZeroMQException.java**

Capítulo 7

Documentación de archivos

7.1. Referencia del Archivo AbstractNode.java

Clases

- class **org.jogugi.berkeley.AbstractNode**

  Clase abstracta que implementa la interfaz ***INode*** (p. **??**).

Paquetes

- package **org.jogugi.berkeley**

7.2. Referencia del Archivo Config.java

Clases

- class **org.jogugi.berkeley.Config**

  Clase de configuración para los nodos en el sistema.
- class **org.jogugi.berkeley.Config.LeaderConfig**

  Configuración del nodo líder.
- class **org.jogugi.berkeley.Config.FollowerConfig**

  Configuración del nodo seguidor.

Paquetes

- package **org.jogugi.berkeley**

7.3. Referencia del Archivo Follower.java

Clases

- class **org.jogugi.berkeley.Follower**

  Clase que representa un nodo seguidor en un sistema distribuido.

Paquetes

- package **org.jogugi.berkeley**

7.4. Referencia del Archivo FollowerInfo.java

Clases

- class **org.jogugi.berkeley.FollowerInfo**

  Clase que representa información detallada de un nodo seguidor en el sistema.

Paquetes

- package **org.jogugi.berkeley**

7.5. Referencia del Archivo FollowerState.java

Clases

- enum **org.jogugi.berkeley.FollowerState**

  Enumeración que define los posibles estados de un seguidor durante el proceso de actualización de la hora del sistema.

Paquetes

- package **org.jogugi.berkeley**

7.6. Referencia del Archivo INode.java

Clases

- interface **org.jogugi.berkeley.INode**

  Interfaz que define las operaciones básicas para un nodo en el sistema.

Paquetes

- package **org.jogugi.berkeley**

7.7. Referencia del Archivo Leader.java

Clases

- class **org.jogugi.berkeley.Leader**

  Representa un nodo líder en el sistema.

Paquetes

- package **org.jogugi.berkeley**

7.8. Referencia del Archivo LoggerManager.java

Clases

- class **org.jogugi.berkeley.LoggerManager**
  La clase.

Paquetes

- package **org.jogugi.berkeley**

7.9. Referencia del Archivo Main.java

Clases

- class **org.jogugi.berkeley.Main**
  Clase principal que gestiona la inicialización y ejecución del líder y los seguidores en el sistema.

Paquetes

- package **org.jogugi.berkeley**

7.10. Referencia del Archivo SocketZeroMQException.java

Clases

- class **org.jogugi.berkeley.SocketZeroMQException**
  Excepción personalizada para errores en la operación de sockets de ZeroMQ.
- enum **org.jogugi.berkeley.SocketZeroMQException.ErrorType**
  Enumeración de los tipos de errores posibles.

Paquetes

- package **org.jogugi.berkeley**

Índice alfabético