Class Block

java.lang.Object
    blockchaintask0.Block

```
public class Block
extends java.lang.Object
```

The Block Class

This class represents a simple Block.

Each Block object has an index - the position of the block on the chain. The first block (the so called Genesis block) has an index of 0.

Each block has a timestamp - a Java Timestamp object, it holds the time of the block's creation.

Each block has a field named data - a String holding the block's single transaction details.

Each block has a String field named previousHash - the SHA256 hash of a block's parent. This is also called a hash pointer.

Each block holds a nonce - a BigInteger value determined by a proof of work routine. This has to be found by the proof of work logic. It has to be found so that this block has a hash of the proper difficulty. The difficulty is specified by a small integer representing the minimum number of leading hex zeroes the hash must have.

Each block has a field named difficulty - it is an int that specifies the minimum number of left most hex digits needed by a proper hash. The hash is represented in hexadecimal. If, for example, the difficulty is 3, the hash must have at least three leading hex 0's (or,1 and 1/2 bytes). Each hex digit represents 4 bits.

Constructor Summary

Constructors

Constructor	Description
`Block(int index, java.sql.Timestamp timestamp, java.lang.String data, int difficulty)`	This the Block constructor.

Method Summary

		previousHash, nonce, and difficulty.
java.lang.String	**getData**()	Simple getter method
int	**getDifficulty**()	Simple getter method
int	**getIndex**()	Simple getter method
java.math.BigInteger	**getNonce**()	This method returns the nonce for this block.
java.lang.String	**getPreviousHash**()	Simple getter method
java.sql.Timestamp	**getTimestamp**()	Simple getter method
static void	**main**(java.lang.String[] args)	
java.lang.String	**proofOfWork**()	The proof of work methods finds a good hash.
void	**setData**(java.lang.String data)	Simple setter method
void	**setDifficulty**(int difficulty)	Simple setter method
void	**setIndex**(int index)	Simple setter method
void	**setPreviousHash**(java.lang.String previousHash)	Simple setter method
void	**setTimestamp**(java.sql.Timestamp timestamp)	Simple setter method
java.lang.String	**toString**()	Override Java's toString method

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

```
java.lang.String data,
int difficulty)
```

This the Block constructor.

Parameters:

`index` - This is the position within the chain. Genesis is at 0.

`timestamp` - This is the time this block was added.

`data` - This is the transaction to be included on the blockchain.

`difficulty` - This is the number of leftmost nibbles that need to be 0.

Method Detail

calculateHash

```
public java.lang.String calculateHash()
```

This method computes a hash of the concatenation of the index, timestamp, data, previousHash, nonce, and difficulty.

Returns:

a String holding Hexadecimal characters

getNonce

```
public java.math.BigInteger getNonce()
```

This method returns the nonce for this block. The nonce is a number that has been found to cause the hash of this block to have the correct number of leading hexadecimal zeroes.

Returns:

a BigInteger representing the nonce for this block.

This method calls calculateHash() to compute a hash of the concatenation of the index, timestamp, data, previousHash, nonce, and difficulty. If the hash has the appropriate number of leading hex zeroes, it is done and returns that proper hash. If the hash does not have the appropriate number of leading hex zeroes, it increments the nonce by 1 and tries again. It continues this process, burning electricity and CPU cycles, until it gets lucky and finds a good hash.

Returns:

a String with a hash that has the appropriate number of leading hex zeroes. The difficulty value is already in the block. This is the minimum number of hex 0's a proper hash must have.

getDifficulty

```
public int getDifficulty()
```

Simple getter method

Returns:

difficulty

setDifficulty

```
public void setDifficulty(int difficulty)
```

Simple setter method

Parameters:

`difficulty` - determines how much work is required to produce a proper hash

toString

```
public java.lang.String toString()
```

Override Java's toString method

Overrides:

`toString` in class `java.lang.Object`

Returns:

```
public void setPreviousHash(java.lang.String previousHash)
```

Simple setter method

Parameters:

`previousHash` - a hashpointer to this block's parent

getPreviousHash

```
public java.lang.String getPreviousHash()
```

Simple getter method

Returns:

previous hash

getIndex

```
public int getIndex()
```

Simple getter method

Returns:

index of block

setIndex

```
public void setIndex(int index)
```

Simple setter method

Parameters:

`index` - the index of this block in the chain

Parameters:

`timestamp` - of when this block was created

getTimestamp

```
public java.sql.Timestamp getTimestamp()
```

Simple getter method

Returns:

timestamp of this block

getData

```
public java.lang.String getData()
```

Simple getter method

Returns:

this block's transaction

setData

```
public void setData(java.lang.String data)
```

Simple setter method

Parameters:

`data` - represents the transaction held by this block

main

```
public static void main(java.lang.String[] args)
```