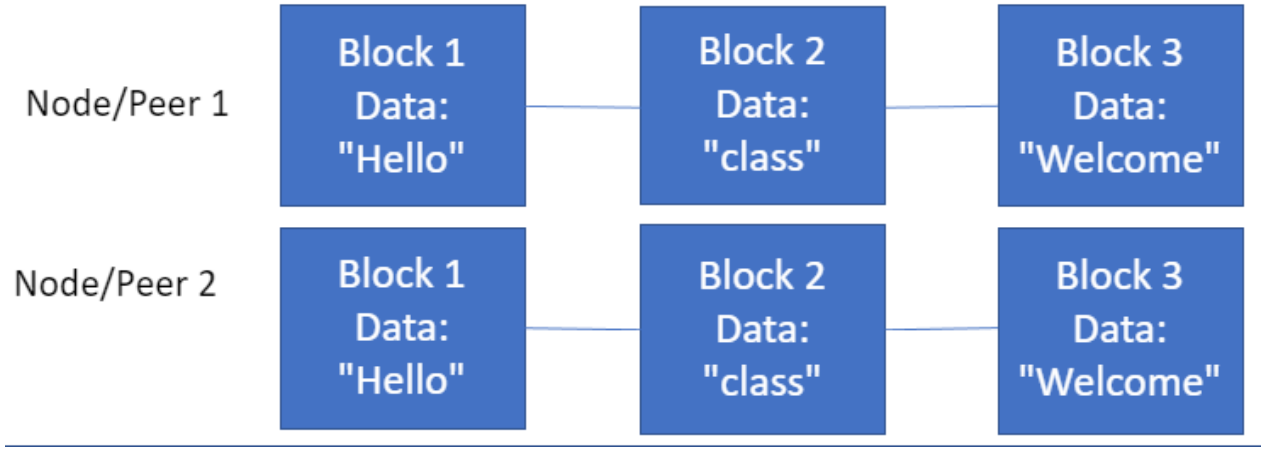
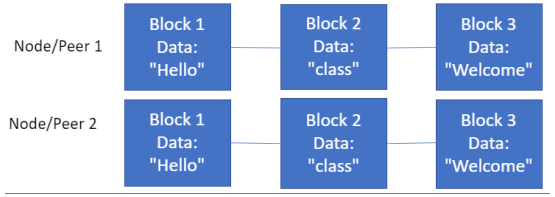
hochschule hof
University of Applied Sciences

Blockchain Breakdown

Joe Oakes

joe.oakes@psu.edu



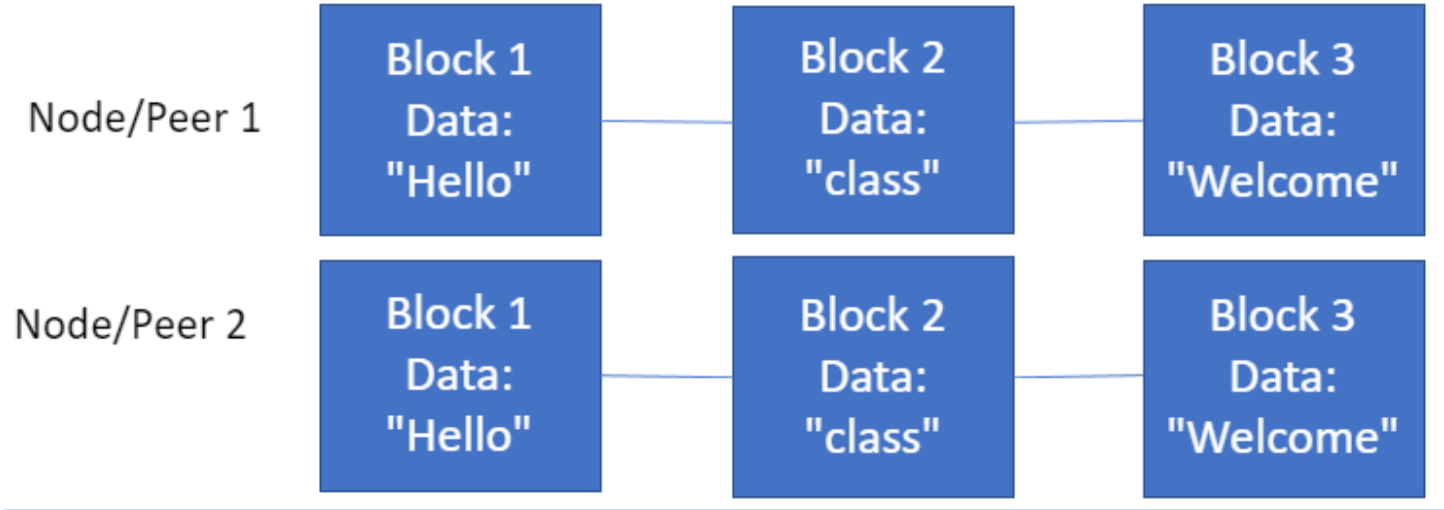


Blockchain Breakdown: Overview

hochschule hof
University of Applied Sciences

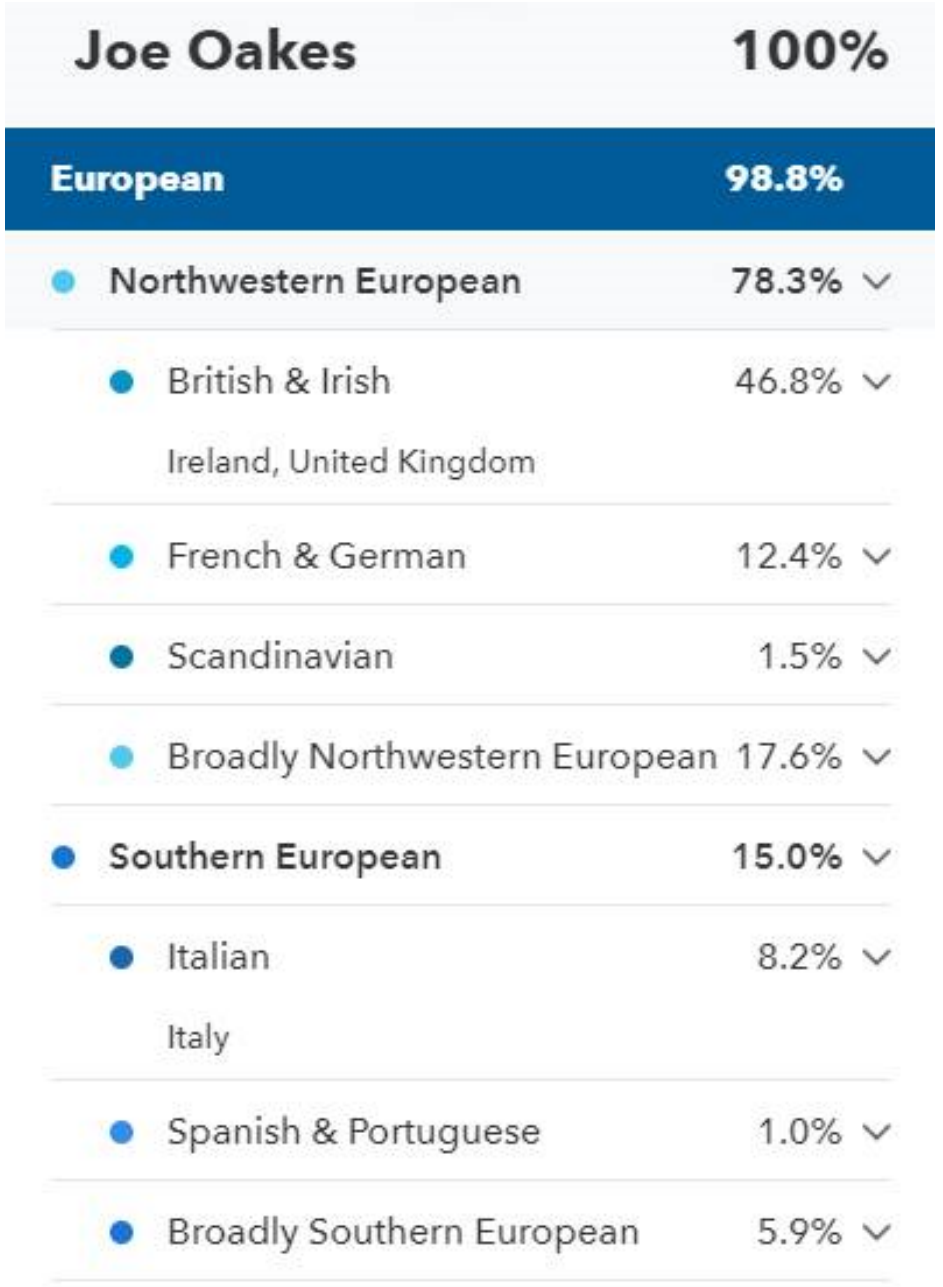
Joe Oakes

- What is **Joe Oakes**
- What is a **Blockchain**?
- What is a **Centralized Network**?
- What is a **P2P Network**?
- What is a **Block**?
  - What is a **Hash**?
  - Computing the **Block Hash**
  - What is a **Timestamp**?
  - What is a **Block Nonce**?
- What are **Mining and Signing**?
- What is a **Payload?**
- What is a **Node/Peer**?
- What is a **Distributed Blockchain?**
- What is a **Merkle Tree?**
- Review



What is Joe Oakes: Education

- Life Learner
- Associate Degree in **Computer Science**
- Undergraduate Degree from Jefferson University in **MIS Management Information Systems**
- Master Degree from Penn State in **Software Engineering**
- Master Degree from Penn State in **Information Systems**
- PhD in **Computer Science** Towson University Summer 2019

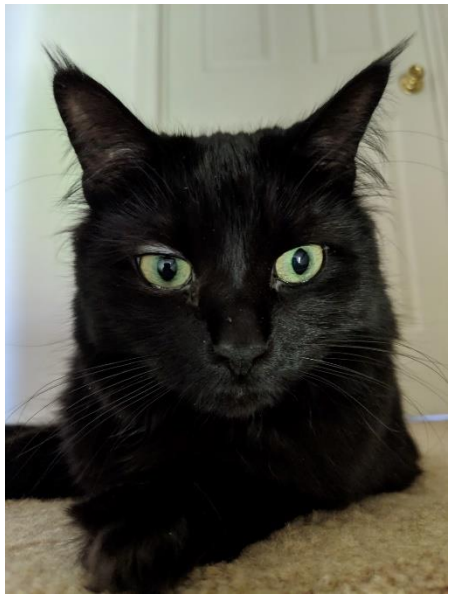


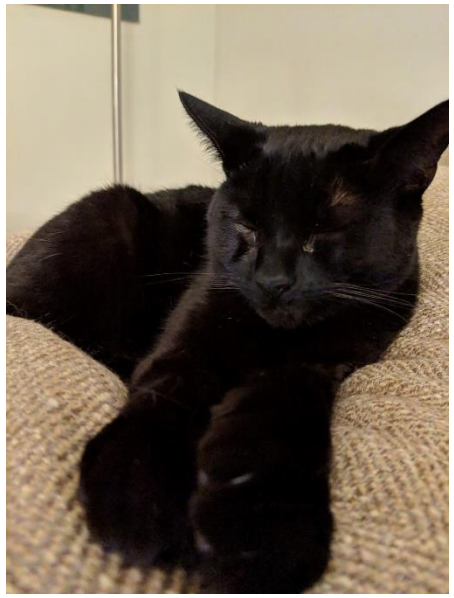
What is Joe Oakes: Employment

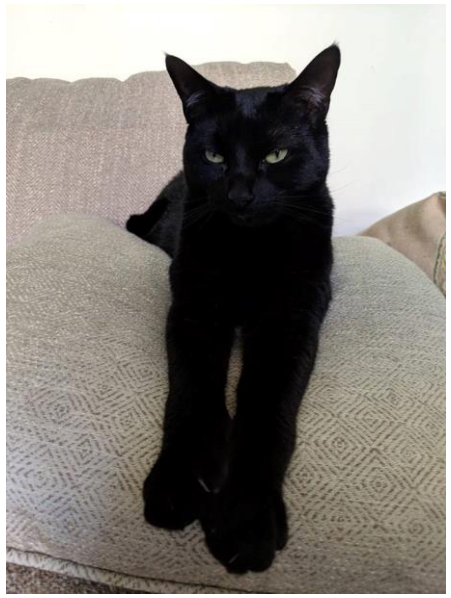
- I have worked for Penn State University for 21 years
- Involved with teaching at Hof for 15 years
- Worked for a Credit Card transaction processing company for 10 years
- Worked for ERP Enterprise Resource Planning Software company for 10 years
- Love all technologies: because they can make you $$$

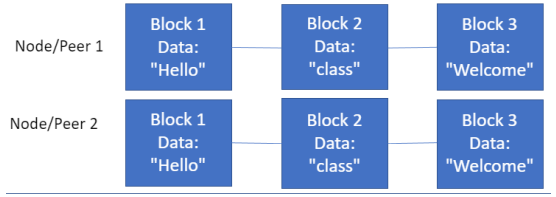
What is Joe Oakes: Personal

- From Philadelphia Pennsylvania USA
- Hardcore mountain biker
- Previous hardcore snowboarder
- Cat Lover: I have three awesome cats
  - Yoda, Explorer, Alpha







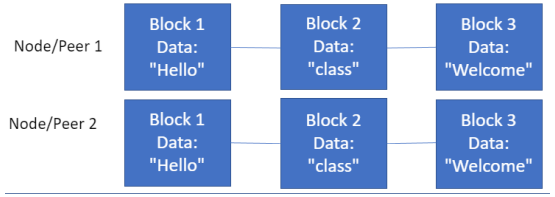


Blockchain Breakdown: Resources

hochschule hof
University of Applied Sciences

Joe Oakes

- Coding example: https://github.com/joeoakes/reNoobChain
- **Website Demo**: https://anders.com/blockchain/hash.html

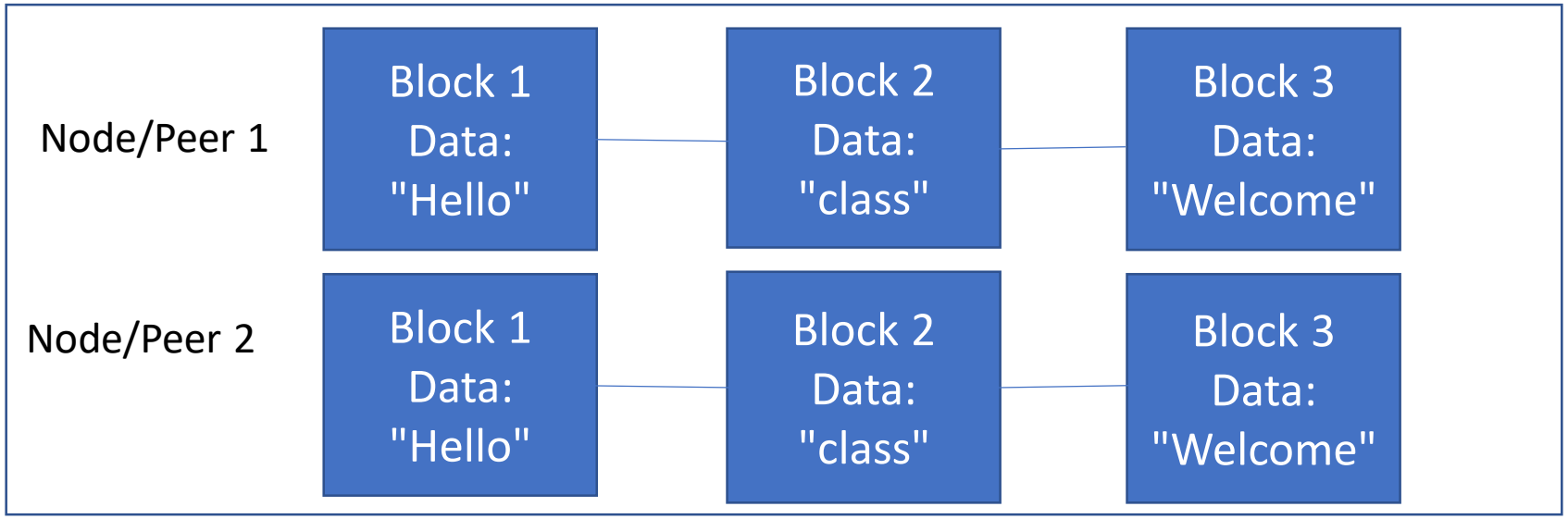


Blockchain: What is a Blockchain?

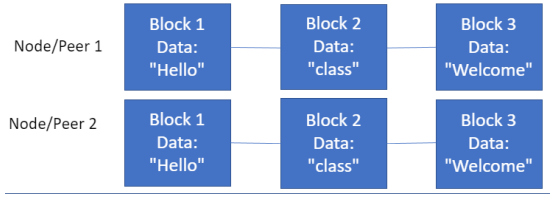
hochschule hof
University of Applied Sciences

Joe Oakes

- Blockchain is a **Peer to Peer** P2P database (which helps to make it secure) vs a **centralized** database system
- It is **distributed data storage** consisting of containers (data blocks) which are connected – chained together
- The same copy of the data is kept on **multiple machines** that are connected
- The multiple blocks of data are **ordered**



Distributed Data Network

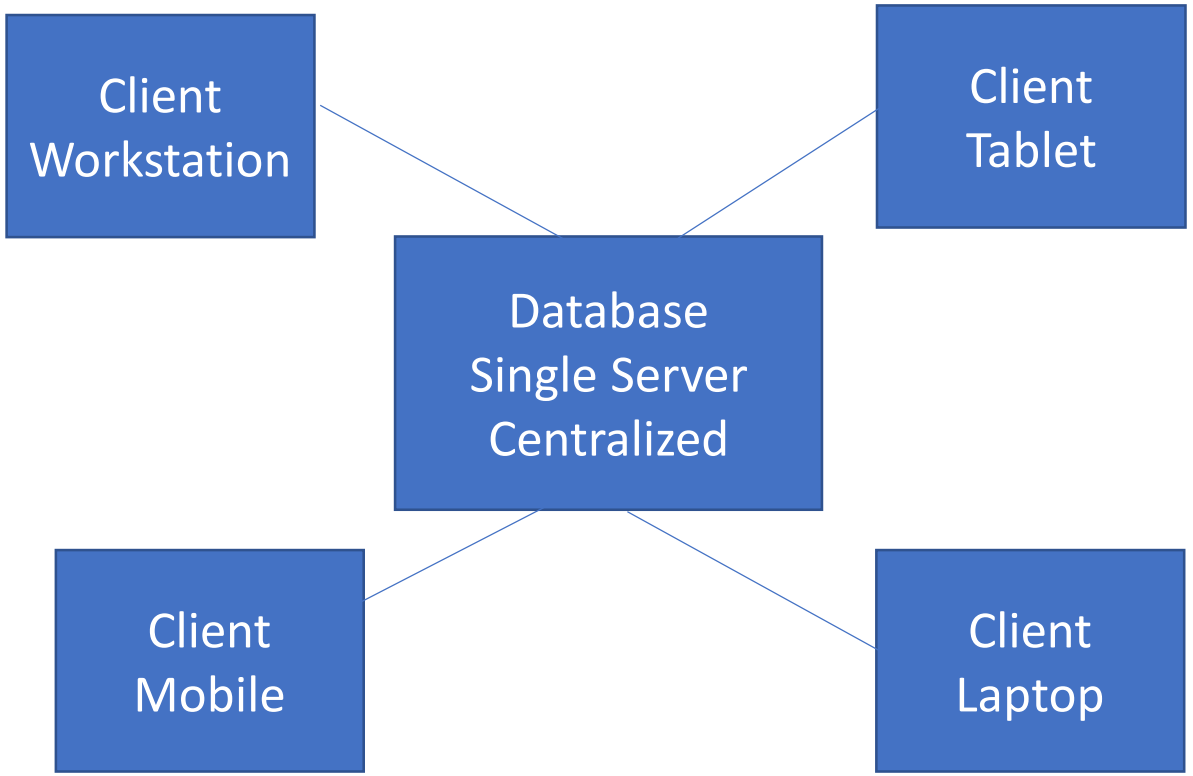


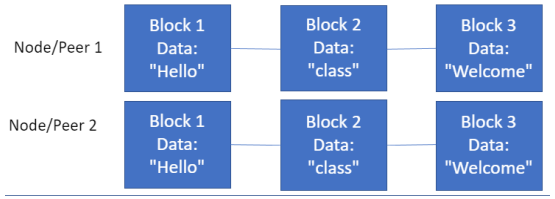
Blockchain: What is a Centralized Network?

hochschule hof
University of Applied Sciences

Joe Oakes

- The **Server** is the central authority storing and managing all the information
- **Centralized point** of failure
- **Security Issues**: can be hacked and now all the information is compromised
- **Centralized control** over the content, for example pictures on social media site



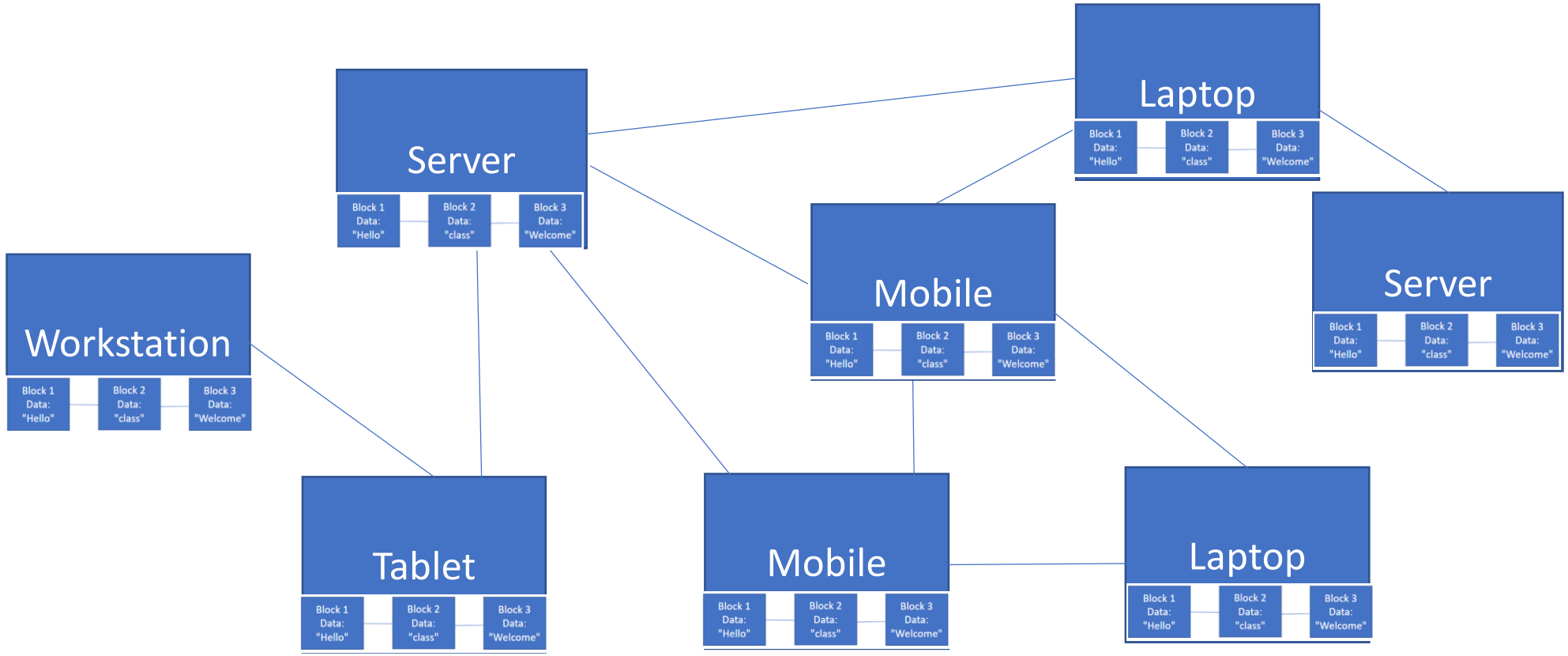


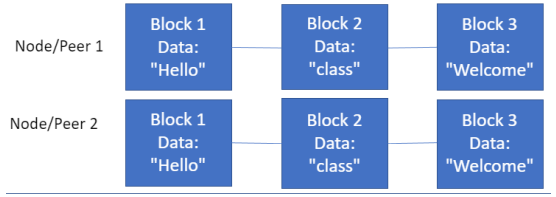
Blockchain: What is a Peer to Peer P2P network?

hochschule hof
University of Applied Sciences

Joe Oakes

- There is no **controlling party**, or **central storage**
- All the information on the network is **recorded and transferred** across the participants (nodes or peers) on the network
- All parties have the same **identical copies of the information**



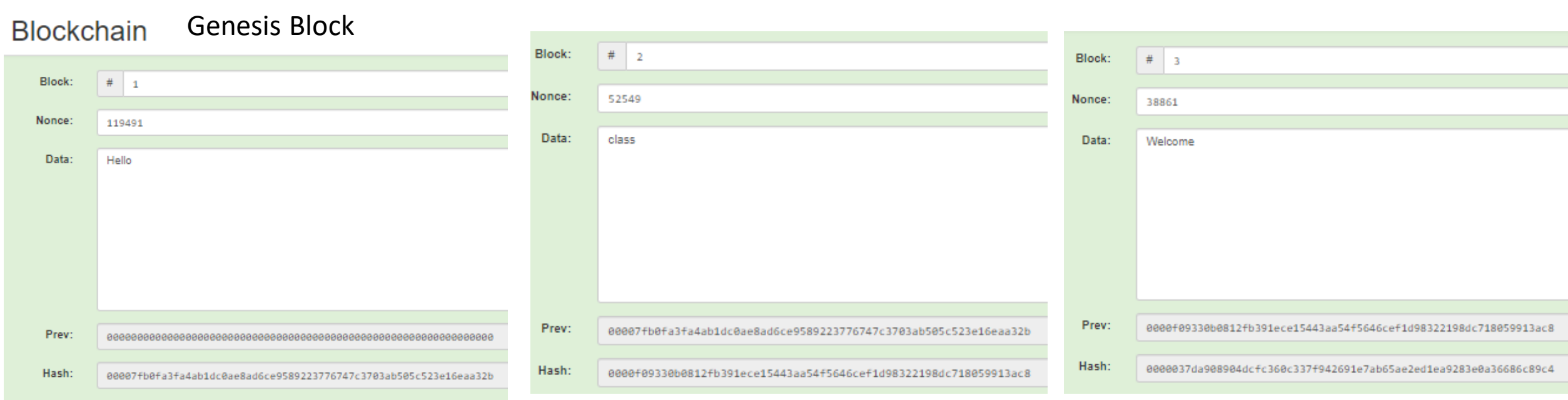


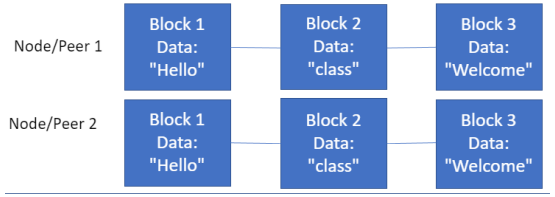
Blockchain: Process of building Blockchain

hochschule hof
University of Applied Sciences

Joe Oakes

1. Create a genesis block
2. Mine (get a valid hash) the block based on difficulty by changing the nonce value
3. Get the next block (mine (get a valid hash))



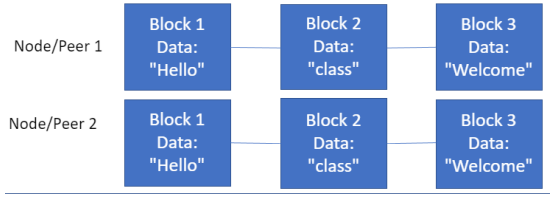


Blockchain: What is Block?

hochschule hof
University of Applied Sciences

Joe Oakes

- A block is a type of container - data structure
- A block is composed of a header and transactions
- A block can contain around 500 transactions
- A block size can average between 1MB and 8MB
- The block header contains metadata about the block
  - Previous block's hash
  - Mining completion – valid hash, timestamp, nonce, difficulty (longer it takes to get a valid hash)
  - Merkle tree root – data structure to summarize the transactions in the block
- The first block is called the Genesis block



Blockchain: What is a Block?

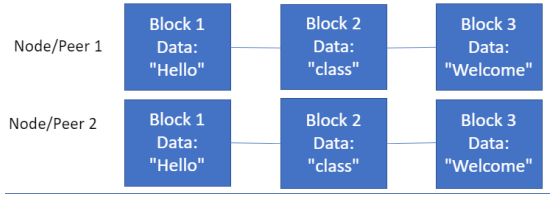
hochschule hof
University of Applied Sciences

Joe Oakes

- **Classification** of a Block - Java Class Block Example
- **Properties**: ID, Hash, Previous Hash, Data, Timestamp, Nonce
- **Hash:** digital signature, used to determine the integrity of the data, MD5, SHA1, SHA-2
- **Previous Hash**: zero for the first block, else contains the value of previous hash
- **Data**: payload or token
- **Timestamp**: the number of seconds that have elapsed since January 1, 1970
- **Nonce:** arbitrary number that can be used just once

Block Classification
Index, Hash, Previous Hash, Data: "Hello", Timestamp, Nonce

```
public class Block {
    //initialization of properties
    public String hash;     //Digital Signature
    public String previousHash;
    private String data; //our data will be a simple message.
    private long timeStamp; //as number of milliseconds since 1/1/1970.
    private int nonce;   //incremented number until the hash is valid
```



Blockchain: Block Data POJO

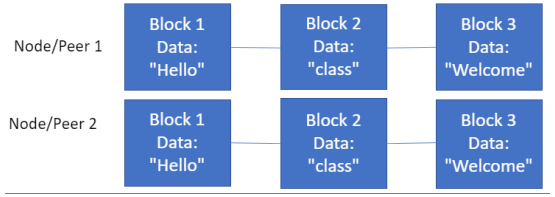
hochschule hof
University of Applied Sciences

Joe Oakes

- **Data**: payload: POJO Plain Old Java Object
  - Object: **Banking Record**

```
public class BankingRecord{
    //First define the fields in the banking record POJO
    private String accountNumber, accountType, transactionType, transactionAmount, balance;

    //Now start setters and getters
    public String getAccountNumber() {return accountNumber;}
    public void setAccountNumber(String _ActN) {this.accountNumber = _ActN;}
    public String getAccountType() {return accountType;}
    public void setAccountType(String _ActT) {this.accountType = _ActT;}
    public String getTransactionType() {return transactionType;}
    public void setTransactionType(String _TraT) {this.transactionType = _TraT;}
    public String getTransactionAmount() {return transactionAmount;}
    public void setTransactionAmount(String _TraA) {this.transactionAmount = _TraA;}
    public String getBalance() {return balance;}
    public void setBalance(String _Bal) {this.balance = _Bal;}
}
```



Blockchain: Block Data POJO

hochschule hof
University of Applied Sciences

Joe Oakes

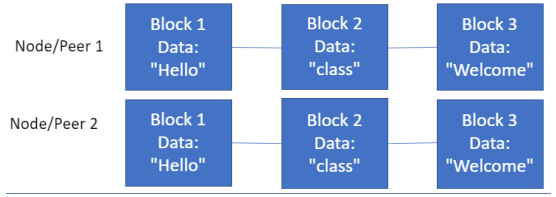
- **Data**: payload: POJO Plain Old Java Object
  - Object: **Banking Record**
  - The object was converted to JSON payload format
  - Then the payload was AES encrypted
  - The Data Block is then added to the Blockchain

```
BankingRecord bankRecord = new BankingRecord();
bankRecord.setAccountNumber("1234");
bankRecord.setAccountType("Savings");
bankRecord.setTransactionType("Deposit");
bankRecord.setTransactionAmount("100");
bankRecord.setBalance("200");

Gson gson = new Gson();
String jsonBank = gson.toJson(bankRecord);

String encryptedBankRecord = AES.encrypt(jsonBank, secretKey) ;

addBlock(new Block(encryptedBankRecord, previousHash: "0"));
```



Blockchain: Block Data JSON

hochschule hof
University of Applied Sciences

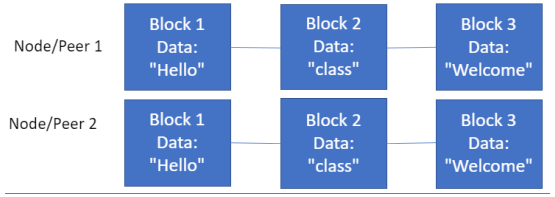
Joe Oakes

- **Data**: payload: **JSON (JavaScript Object Notation) Formatted payload**
  - Object: **Banking Record**
  - **AES encrypted and decrypted payload -** Advanced Encryption Standard symmetric **encryption algorithm**


```
 {
   "hash": "00000cd0331cbfa7d87553cf860699dba6878c73992f067e71f73682bcac6dac",
   "previousHash": "0",
   "data":
"Ftl/qPSGPcbMkufAsU5quDJQqMyjgqZFCbHO3JpfJJ5+CC+L7OYHvtTFsnCYaFn2HxVUT0SdOh9yc3HcjsA66N7PlepJ/8ChzHQ2GUG
Q+1mu6etIDh9zZJQajMdHew6M/YfKO6zZOx5EV4rxKbJtSK5K0k9JsjgOS5/TvqE/BLA\u003d",
   "timeStamp": 1558518234846,
   "nonce": 565048
 }
```


Decrypted block data for block #1:


```
{"accountNumber":"1234","accountType":"Savings","transactionType":"Deposit","transactionAmount":"100","balance":"200"}
```



Blockchain: Block Data POJO

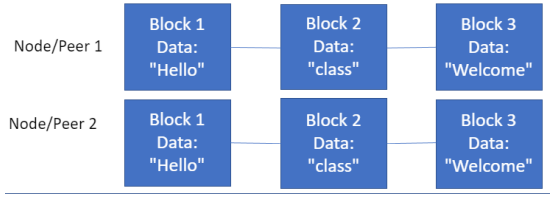
hochschule hof
University of Applied Sciences

Joe Oakes

- **Data**: payload: POJO Plain Old Java Object
  - Object: **Medical Record**

```
public class MedicalRecord{
    //First define the fields in the medical record POJO
    private String patientName, patientID, visitDate, doctorName, procedureCode;

    //Now start setters and getters
    public String getPatientName() {return patientName;}
    public void setPatientName(String _PatN) {this.patientName = _PatN;}
    public String getPatientID() {return patientID;}
    public void setPatientID(String _PatID) {this.patientID = _PatID;}
    public String getVisitDate() {return visitDate;}
    public void setVisitDate(String _VisD) {this.visitDate = _VisD;}
    public String getDoctorName() {return doctorName;}
    public void setDoctorName(String _DocN) {this.doctorName = _DocN;}
    public String getProcedureCode() {return procedureCode;}
    public void setProcedureCode(String _ProC) {this.procedureCode = _ProC;}
}
```



Blockchain: Block Data POJO

hochschule hof
University of Applied Sciences

Joe Oakes

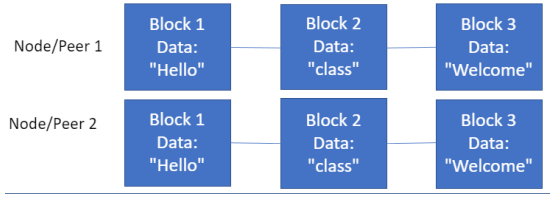
- **Data**: payload: POJO Plain Old Java Object
  - Object: **Medical Record**
  - The object was converted to JSON payload format
  - Then the payload was AES encrypted
  - The Data Block is then added to the Blockchain

```
MedicalRecord medicalRecord = new MedicalRecord();
medicalRecord.setDoctorName("Dr. Joseph Oakes");
medicalRecord.setPatientID("54469");
medicalRecord.setPatientName("Joseph Sliwka");
medicalRecord.setProcedureCode("582");
medicalRecord.setVisitDate("2/21/2019");

String jsonMedical = gson.toJson(medicalRecord);

String encryptedMedicalRecord = AES.encrypt(jsonMedical, secretKey);

addBlock(new Block(encryptedMedicalRecord,blockchain.get(blockchain.size()-1).hash));
```



Blockchain: Block Data JSON

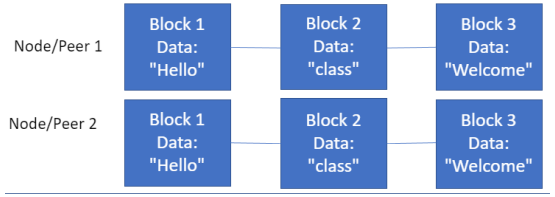
hochschule hof
University of Applied Sciences

Joe Oakes

- **Data**: payload: **JSON (JavaScript Object Notation) Formatted payload**
  - Object: **Medical Record**
  - **AES encrypted and decrypted payload -** Advanced Encryption Standard symmetric **encryption algorithm**

{
"hash": "000007b40cadf1bc50a52d528f812e0519b5c0b2768a9b2dff5455b84a5e1b50",
"previousHash": "00000cd0331cbfa7d87553cf860699dba6878c73992f067e71f73682bcac6dac",
"data":
"wq7RaImWbeLHhKWEbWbDy7qzviLLcpp6PJm62AyCmd3fiWITaQfoG8pD4iTKbFfVfVmGbR0Mi7El2wQkHVredS6xIxOgYK6ytwgS6BJiCpmL5wT2v5bMIZbspwg2cvNpl46v6CucuvPXRYAW0CGrhGXCIFZLXj44GeFF3qYJGTFeAk3lIbzV2Xr0PDWTR0Nn",
"timeStamp": 1558518236296,
"nonce": 359930
}

Decrypted block data for block #2: {"patientName":"Joseph Sliwka","patientID":"54469","visitDate":"2/21/2019","doctorName":"Dr. Joseph Oakes","procedureCode":"582"}



Blockchain: Block Data POJO

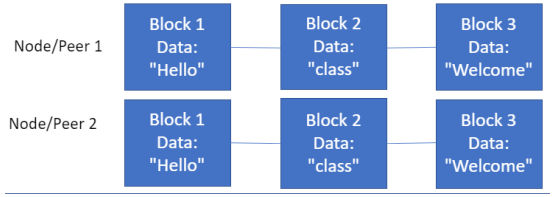
hochschule hof
University of Applied Sciences

Joe Oakes

- **Data**: payload: POJO Plain Old Java Object
  - Objects: **Credit Card Transaction**

```
public class CreditCard{
    //First define the fields in the credit card transaction POJO
    private String cardholderName, date, transactionType, businessName, status;

    //Now start setters and getters
    public String getCardholderName() {return cardholderName;}
    public void setCardholderName(String _CarN) {this.cardholderName = _CarN;}
    public String getDate() {return date;}
    public void setDate(String _Date) {this.date = _Date;}
    public String getTransactionType() {return transactionType;}
    public void setTransactionType(String _TraT) {this.transactionType = _TraT;}
    public String getBusinessName() {return businessName;}
    public void setBusinessName(String _BuiN) {this.businessName = _BuiN;}
    public String getStatus() {return status;}
    public void setStatus(String _Status) {this.status = _Status;}
}
```



Blockchain: Block Data POJO

hochschule hof
University of Applied Sciences

Joe Oakes

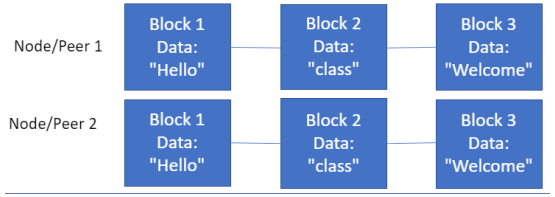
- **Data**: payload: POJO Plain Old Java Object
  - Object: **Credit Card Transaction**
  - The object was converted to JSON payload format
  - Then the payload was AES encrypted
  - The Data Block is then added to the Blockchain

```
CreditCard creditCardTransaction = new CreditCard();
creditCardTransaction.setBusinessName("Penn State University - Abington");
creditCardTransaction.setCardholderName("Joseph Sliwka");
creditCardTransaction.setDate("2/21/2019");
creditCardTransaction.setStatus("Pending");
creditCardTransaction.setTransactionType("PURCHASE");

String jsonCreditCardTransaction = gson.toJson(creditCardTransaction);

String jsonCreditCardTransactionEncrypted = AES.encrypt(jsonCreditCardTransaction, secretKey);

addBlock(new Block(jsonCreditCardTransactionEncrypted,blockchain.get(blockchain.size()-1).hash));
```

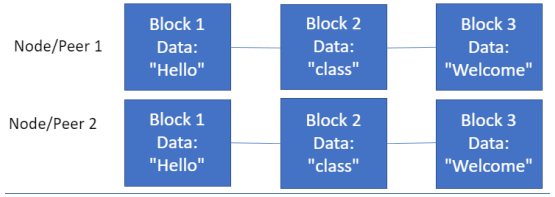


Blockchain: Block Data JSON

hochschule hof
University of Applied Sciences
Joe Oakes

- **Data**: payload: **JSON (JavaScript Object Notation) Formatted payload**
  - Object: **Credit Card Transaction**
  - **AES encrypted and decrypted payload -** Advanced Encryption Standard symmetric **encryption algorithm**

{
"hash": "00000f63dc7c2f187d2d2877e370f5381eec84f5e7abcce4a148697fda235e1b",
"previousHash": "000007b40cadf1bc50a52d528f812e0519b5c0b2768a9b2dff5455b84a5e1b50",
"data":
"rEcAm6QylZBVNe6/klHHjbfoupMBMDyeZAG8+yj9MlkvVBKbncCFSs4lxMJT1HCb6R3pBnxOLYeZ2Qo2ip6tHRZgDLHkyW5d90Ysv+Dz1RcD8ZGD18j3DnRolidyfTrpmuxLchPMjYd0PKdRYMOtFgp32Nj6VeL8PTojZfVlBVuwvMM8ZX++nCAzadg+TBVM958R/3M638VSu8VylleXMg\u003d\u003d",
"timeStamp": 1558518237089,
"nonce": 102517
}
Decrypted block data for block #3: {"cardholderName":"Joseph Sliwka","date":"2/21/2019","transactionType":"PURCHASE","businessName":"Penn State University - Abington","status":"Pending"}



Blockchain: Block Data POJO

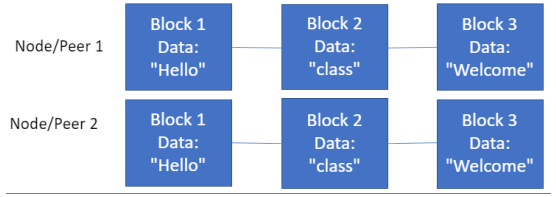
hochschule hof
University of Applied Sciences

Joe Oakes

- **Data**: payload: POJO Plain Old Java Object
  - Object: **Student Record**

```
public class Student {
    //First define the fields in the credit card transaction POJO
    private String studentID, studentName, studentMajor;

    //Now start setters and getters
    public String getStudentID() {return studentID;}
    public void setStudentID(String _StuID) {this.studentID = _StuID;}
    public String getStudentName() {return studentName;}
    public void setStudentName(String _StuN) {this.studentName = _StuN;}
    public String getStudentMajor() {return studentMajor;}
    public void setStudentMajor(String _StuM) {this.studentMajor = _StuM;}
}
```



Blockchain: Block Data POJO

hochschule hof
University of Applied Sciences

Joe Oakes

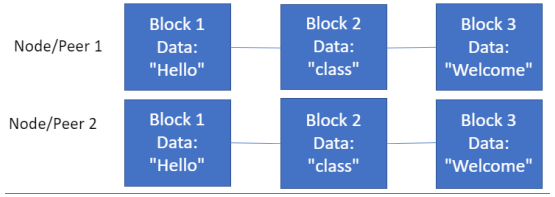
- **Data**: payload: POJO Plain Old Java Object
  - Object: **Student Record**
  - The object was converted to JSON payload format
  - Then the payload was AES encrypted
  - The Data Block is then added to the Blockchain

```
Student student1 = new Student();
student1.setStudentID("981493786");
student1.setStudentName("Joseph Sliwka");
student1.setStudentMajor("IST");

String jsonStudent = gson.toJson(student1);

String jsonStudentEncrypted = AES.encrypt(jsonStudent, secretKey);

addBlock(new Block(jsonStudentEncrypted,blockchain.get(blockchain.size()-1).hash));
```



Blockchain: Block Data JSON

hochschule hof
University of Applied Sciences
Joe Oakes

- **Data**: payload: **JSON (JavaScript Object Notation) Formatted payload**
  - Object: **Student Record**
  - **AES encrypted and decrypted payload -** Advanced Encryption Standard symmetric **encryption algorithm**


```
{
  "hash": "00000dc3cfb52d12562bb80ab99d71270d03c22a680584972a97584d5d987803",
  "previousHash": "00000f63dc7c2f187d2d2877e370f5381eec84f5e7abcce4a148697fda235e1b",
  "data":
"3c2nS5c9bRKbHyo+9XENdH+nevZy4N1qkBs6AoJhIIlKD1/RlpfrN6LIPcWF/lRJxqIEAtCcXuZp0uDlnTijuzAP5tPhWCZXhpRWR7aLlVU\
u003d",
  "timeStamp": 1558518237298,
  "nonce": 1897054
}
```

Decrypted block data for block #4: {"studentID":"981493786","studentName":"Joseph Sliwka","studentMajor":"IST"}

[{ "hash": "00000cd0331cbfa7d87553cf860699dba6878c73992f067e71f73682bcac6dac", "previousHash": "0",

"data":
"Ftl/qPSGPcbMkufAsU5quDJQqMyjgqZFCbHO3JpfJJ5+CC+L7OYHvtTFsnCYaFn2HxVUT0SdOh9yc3HcjsA66N7PlepJ/8ChzHQ2GUGQ+1mu6etIDh9zZJQajMdHew6M/YfKO6zZOx5EV4rxKbJtSK5K0k9JsjgOS5/TvqE/BLA\u003d",

"timeStamp": 1558518234846, "nonce": 565048},

{ "hash": "000007b40cadf1bc50a52d528f812e0519b5c0b2768a9b2dff5455b84a5e1b50", "previousHash":
"00000cd0331cbfa7d87553cf860699dba6878c73992f067e71f73682bcac6dac",

"data":
"wq7RaImWbeLHhKWEbWbDy7qzviLLcpp6PJm62AyCmd3fiWITaQfoG8pD4iTKbFfVfVmGbR0Mi7El2wQkHVredS6xIxOgYK6ytwgS6BJiCpmL5wT2v5bMlZbspwg2cvNpl46v6CucuvPXRYAW0CGrhGXCIFZLXj44GeFF3qYJGTFeAk3lIbzV2Xr0PDWTR0Nn",

"timeStamp": 1558518236296, "nonce": 359930},

{ "hash": "00000f63dc7c2f187d2d2877e370f5381eec84f5e7abcce4a148697fda235e1b", "previousHash":
"000007b40cadf1bc50a52d528f812e0519b5c0b2768a9b2dff5455b84a5e1b50",

"data":
"rEcAm6QylZBVNe6/klHHjbfoupMBMDyeZAG8+yj9MlkvVBKbncCFSs4lxMJT1HCb6R3pBnxOLYeZ2Qo2ip6tHRZgDLHkyW5d90Ysv+Dz1RcD8ZGD18j3DnRolidyfTrpmuxLchPMjYd0PKdRYMOtFgp32Nj6VeL8PTojZfVlBVuwvMM8ZX++nCAzadg+TBVM958R/3M638VSu8VylleXMg\u003d\u003d",

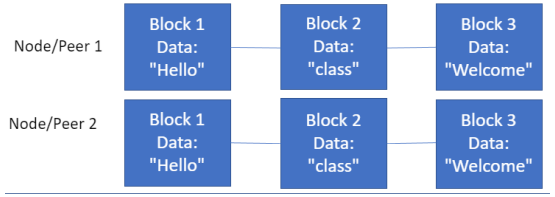
"timeStamp": 1558518237089, "nonce": 102517},

{"hash": "00000dc3cfb52d12562bb80ab99d71270d03c22a680584972a97584d5d987803","previousHash":
"00000f63dc7c2f187d2d2877e370f5381eec84f5e7abcce4a148697fda235e1b",

"data":
"3c2nS5c9bRKbHyo+9XENdH+nevZy4N1qkBs6AoJhIIlKD1/RlpfrN6LIPcWF/lRJxqIEAtCcXuZp0uDlnTijuzAP5tPhWCZXhpRWR7aLlVU\u003d",

"timeStamp": 1558518237298, "nonce": 1897054

}]



Blockchain: What is a Hash?

hochschule hof
University of Applied Sciences

Joe Oakes

Hash value is a **digital signature** derived from a mathematical algorithm

SHA2-256 Hash generator – has value of 256 bits or 64 bytes hex

"Hello" = 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969

Website Demo: https://anders.com/blockchain/hash.html

Block 1
Data:
"Hello"

SHA256 Hash

Data: Hello

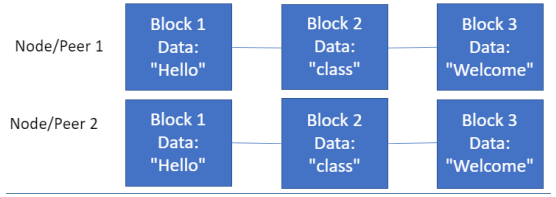
Hash: 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969

SHA256 Hash

Data: hello

Hash: 2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824

64 bytes hexadecimal – Notice changing one character of the data “H” to “h” will change the digital signature



Blockchain: What is a Block Previous Hash?

hochschule hof
University of Applied Sciences

Joe Oakes

- SHA2-256 Hash generator – has value of 256 bits or 64 bytes hexadecimal

"Hello" = 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969

"class" = 0889113e04d3203f0c401c17c0fd8b352b740dc607433779d3edcaa13320b001

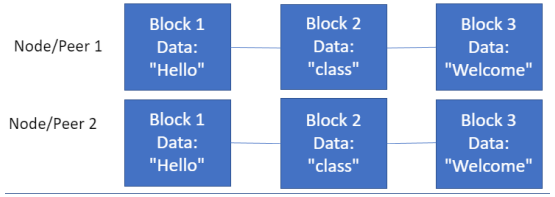
"Welcome" = 0e2226b5235f0ff94a276eb4d07a3bfea74b7e3b8b85e9efca6c18430f041bf8

- But remember we still need to hash in all the block information – **previous hash, timestamp, nonce**

Block 1
Hash: 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969
Previous Hash: 0
Data: "Hello"

Block 2
Hash: 0889113e04d3203f0c401c17c0fd8b352b740dc607433779d3edcaa13320b001
Previous Hash: 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969
Data: "class"

Block 3
Hash: 0e2226b5235f0ff94a276eb4d07a3bfea74b7e3b8b85e9efca6c18430f041bf8
Previous Hash: 0889113e04d3203f0c401c17c0fd8b352b740dc607433779d3edcaa13320b001
Data: "Welcome"



Blockchain: Calculate Hash

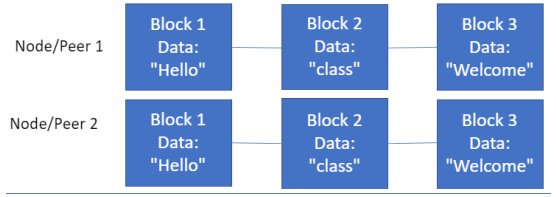
hochschule hof
University of Applied Sciences

Joe Oakes

{ "hash": "00000cd0331cbfa7d87553cf860699dba6878c73992f067e71f73682bcac6dac", "previousHash": "0",
"data":
"Ftl/qPSGPcbMkufAsU5quDJQqMyjgqZFCbHO3JpfJJ5+CC+L7OYHvtTFsnCYaFn2HxVUT0SdOh9yc3HcjsA66N7PlepJ/8ChzHQ2GUGQ+1mu6etIDh9zZJQajMdHew6M/YfKO6zZOx5EV4rxKbJtSK5K0k9JsjgOS5/TvqE/BLA\u003d",
"timeStamp": 1558518234846, "nonce": 565048},
{ "hash": "000007b40cadf1bc50a52d528f812e0519b5c0b2768a9b2dff5455b84a5e1b50", "previousHash":
"00000cd0331cbfa7d87553cf860699dba6878c73992f067e71f73682bcac6dac",
"data":
"wq7RaImWbeLHhKWEbWbDy7qzviLLcpp6PJm62AyCmd3fiWITaQfoG8pD4iTKbFfVfVmGbR0Mi7El2wQkHVredS6xIxOgYK6ytwgS6BJiCpmL5wT2v5bMIZbspwg2cvNpl46v6CucuvPXRYAW0CGrhGXCIFZLXj44GeFF3qYJGTFeAk3lIbzV2Xr0PDWTR0Nn",
"timeStamp": 1558518236296, "nonce": 359930}

```
public String calculateHash() {
    return StringUtil.applySha256( input: previousHash +
                    Long.toString(timeStamp) +
                    Integer.toString(nonce) +
                    data +
                    Long.toString(index)
    );
}
```

The data block is concatenated together into a string and applied to SHA256to calculate the Hash value
This can be used as a Hash Pointer.



Blockchain: What is a Timestamp?

hochschule hof
University of Applied Sciences
Joe Oakes

- **Date Formats:** Julian date, Unix date, Gregorian date, Mayan
- **Julian date** format number of seconds that have elapsed since January 1, 4713 B.C. (2458564.00961)
- **Unix date** format: number of seconds that have elapsed since January 1, 1970 (1553170397) (Java Date().getTime())
- **Gregorian date**: format 9:47 PM 3/14/19, 21:47 03/14/2019, March 3 2019
- **Mayan calendar**: Count of days since August 11, 3114 B.C. (0.0.0.1.5)

The Current Unix Timestamp

1558518699 seconds since Jan 01 1970. (UTC)

This epoch translates to:

05/22/2019 @ 9:51am (UTC)

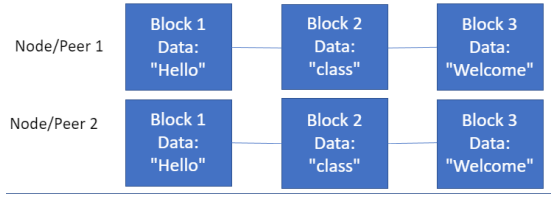
2019-05-22T09:51:39+00:00 in ISO 8601

Wed, 22 May 2019 09:51:39 +0000 in RFC 822, 1036, 1123, 2822

Wednesday, 22-May-19 09:51:39 UTC in RFC 2822

2019-05-22T09:51:39+00:00 in RFC 3339

{
"hash":
"00000cd0331cbfa7d87553cf860699dba6878c73992f067e71f73682bcac6dac",
"previousHash": "0",
"data":
"Ftl/qPSGPcbMkufAsU5quDJQqMyjgqZFCbHO3JpfJJ5+CC+L7OYHvtTFsnCYaFn2HxVUT
0SdOh9yc3HcjsA66N7PlepJ/8ChzHQ2GUGQ+1mu6etIDh9zZJQajMdHew6M/YfKO6zZ
Ox5EV4rxKbJtSK5K0k9JsjgOS5/TvqE/BLA\u003d",
"timeStamp": 1558518234846,
"nonce": 565048
}



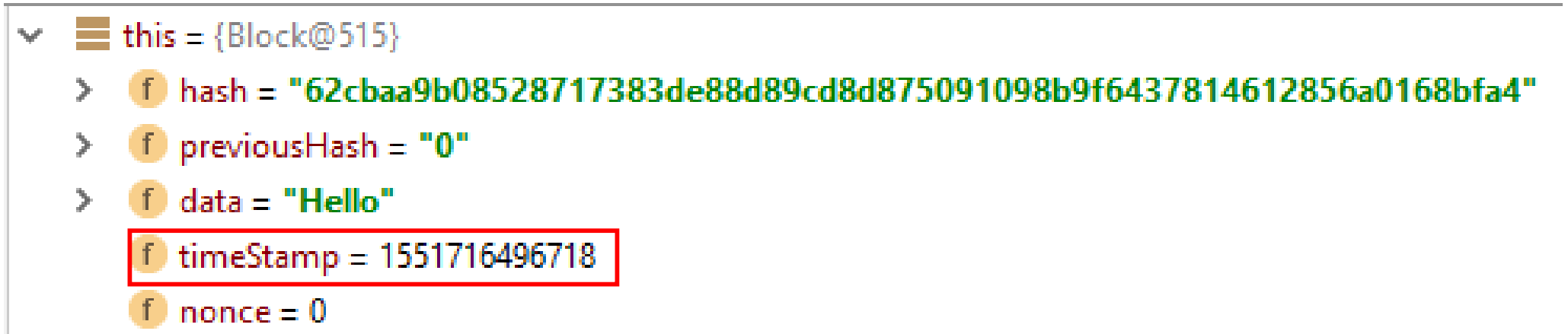
Blockchain: What is a Timestamp?

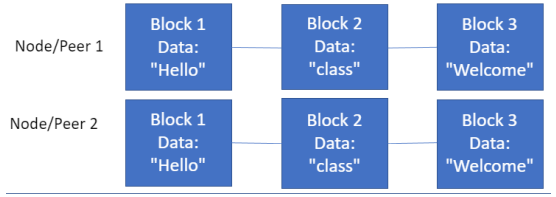
hochschule hof
University of Applied Sciences

Joe Oakes

The **Unix epoch** (or **Unix time** or **POSIX time** or **Unix timestamp**) is the number of seconds that have elapsed since January 1, 1970

```
public Block(String data,String previousHash ) {
    this.data = data;
    this.previousHash = previousHash;
    this.timeStamp = new Date().getTime();
    this.hash = calculateHash();
}
```





Blockchain: What is a Block Timestamp?

hochschule hof
University of Applied Sciences

Joe Oakes

this = {Block@515}
- hash = "62cbaa9b08528717383de88d89cd8d875091098b9f6437814612856a0168bfa4"
- previousHash = "0"
- data = "Hello"
- timeStamp = 1551716496718
- nonce = 0

The current Unix epoch time is **1551718791**

Convert epoch to human readable date and vice versa

1551716496718 | Timestamp to Human date | [batch convert]

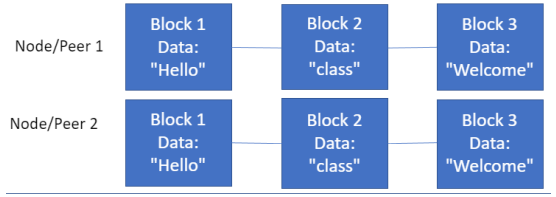
Assuming that this timestamp is in milliseconds:

GMT : Monday, March 4, 2019 4:21:36.718 PM

Your time zone : Monday, March 4, 2019 11:21:36.718 AM GMT-05:00

Relative : 4 minutes ago

Mon	Day	Yr	Hr	Min	Sec			
3	4	2019	4	24	36	PM	GMT	Human date to Timestamp



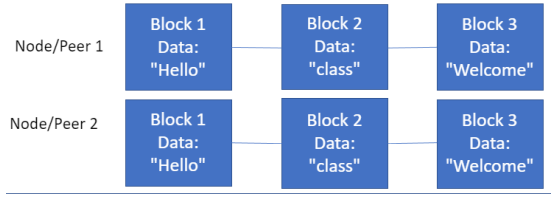
Blockchain: What is a Nonce?

hochschule hof
University of Applied Sciences

Joe Oakes

- Nonce: incremented number until the hash is valid
- The nonce is part of the mining process
- Nonce is an arbitrary number
- Example uses - incremented values starting 1

```
public void mineBlock(int difficulty) {
    //Set the target hash so that the block will stop being hashed once it reaches the target hash.
    String target = StringUtil.getDificultyString(difficulty); //Create a string with difficulty * "0"
    //Calculate the hash of the block and increase the nonce for every
    // iteration until the hash substring reaches the target value
    //the difficulty sets the number of zeros at the beginning of the hash.
    // We are hashing until the number of zeros at the beginning of
    // the hash value is equal to the integer set in the difficulty variable.
    while(!hash.substring( 0, difficulty).equals(target)) {
        nonce ++;
        hash = calculateHash();
    }
    System.out.println("Block Mined!!! : " + hash);
}
```



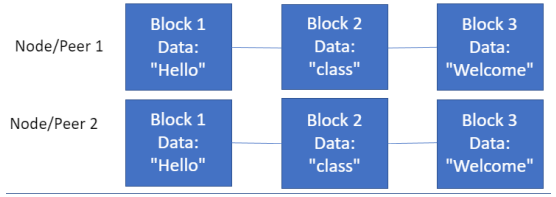
Blockchain: What is a Nonce?

hochschule hof
University of Applied Sciences

Joe Oakes

```java
public void mineBlock(int difficulty) {
    //Set the target hash so that the block will stop being hashed once it reaches the target hash.
    String target = StringUtil.getDificultyString(difficulty); //Create a string with difficulty * "0"
    //Calculate the hash of the block and increase the nonce for every
    // iteration until the hash substring reaches the target value
    //the difficulty sets the number of zeros at the beginning of the hash.
    // We are hashing until the number of zeros at the beginning of
    // the hash value is equal to the integer set in the difficulty variable.
    while(!hash.substring( 0, difficulty).equals(target)) {
        nonce ++;
        hash = calculateHash();
    }
    System.out.println("Block Mined!!! : " + hash);
}
```

```
> this = {Block@1269}
  p difficulty = 5
> target = "00000"
  nonce = 1
> hash = "b0d1c15ba31ac0c40d5c7586d49f3b6eacd7047ccc956df1bd4be2c517547f25"
```



Blockchain: What is a Block Nonce?

hochschule hof
University of Applied Sciences

Joe Oakes

- Notice the block in green has been mined by changing the Nonce value until the difficulty is reached – Hash starts with four 0000
- **Difficulty rule** - the hash value needs to start with **four zeros**

Block

Block: # 1

Nonce: 72608

Data: Hello

Hash: f23b5f6168e9c8fecd5aab55f34c992e51c7033cc50b9021e1042f9c7dde25be

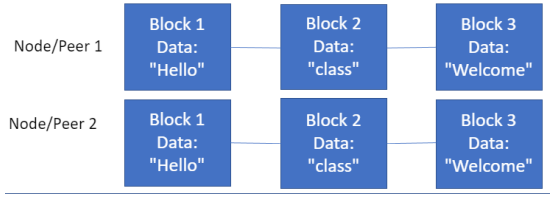
Block

Block: # 1

Nonce: 32904

Data: Hello

Hash: 0000[illegible]2462488067cf69de151b63b06aae9324f26023aa49b8d8ea3cfb2ec6e0b5

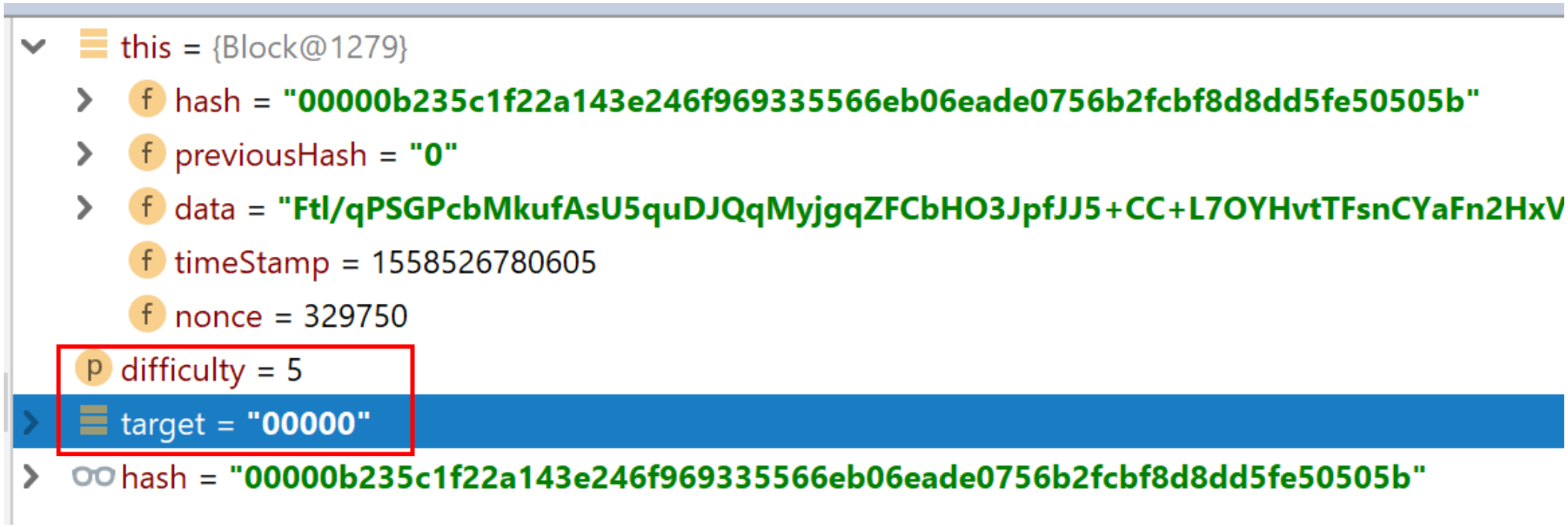


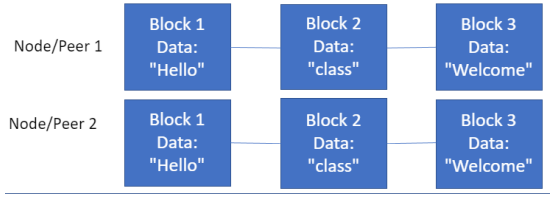
Blockchain: What is a Block Nonce?

hochschule hof
University of Applied Sciences

Joe Oakes

- Notice the block in green has been mined by changing the Nonce value until the difficulty is reached – Hash starts with five 00000
- **Difficulty rule** - the hash value needs to start with **five zeros**





Blockchain: Adding Data Blocks

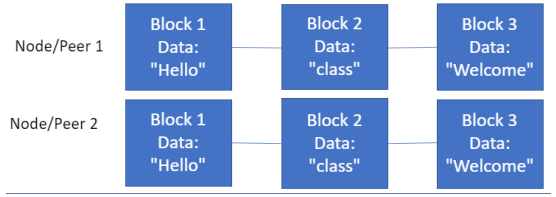
hochschule hof
University of Applied Sciences

Joe Oakes

- **Data**: adding data blocks to the array list data structure
- **Difficulty**: level setting

```
//arraylist for storing created blocks in the blockchain
public static ArrayList<Block> blockchain = new ArrayList<~>();
public static int difficulty = 5;
```

```
//method for adding a new block into the blockchain, accepts a Block object as a parameter
public static void addBlock(Block newBlock) {
    //make the computer do work by mining the passed block before adding the block to the blockchain
    // Pass in the difficulty of mining the block that was defined above.
    //This makes it harder or easier for the computer to mine the block.
    newBlock.mineBlock(difficulty);
    blockchain.add(newBlock);
}
```



Blockchain: What is a Block Nonce?

hochschule hof
University of Applied Sciences

Joe Oakes

Block

Block: # 1

Nonce: 72608

Data: Hello

Hash: f23b5f6168e9c8fecd5aab55f34c992e51c7033cc50b9021e1042f9c7dde25be

Block

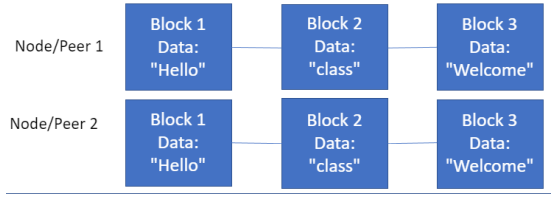
Block: # 1

Nonce: 32904

Data: Hello

Hash: 0000[illegible]2462488067cf69de151b63b06aae9324f26023aa49b8d8ea3cfb2ec6e0b5

```
public void mineBlock(int difficulty) {
    //Set the target hash so that the block will stop being ha
    String target = StringUtil.getDificultyString(difficulty);
    while(!hash.substring( 0, difficulty).equals(target)) {
        nonce ++;
        hash = calculateHash();
    }
    System.out.println("Block Mined!!! : " + hash);
}
```

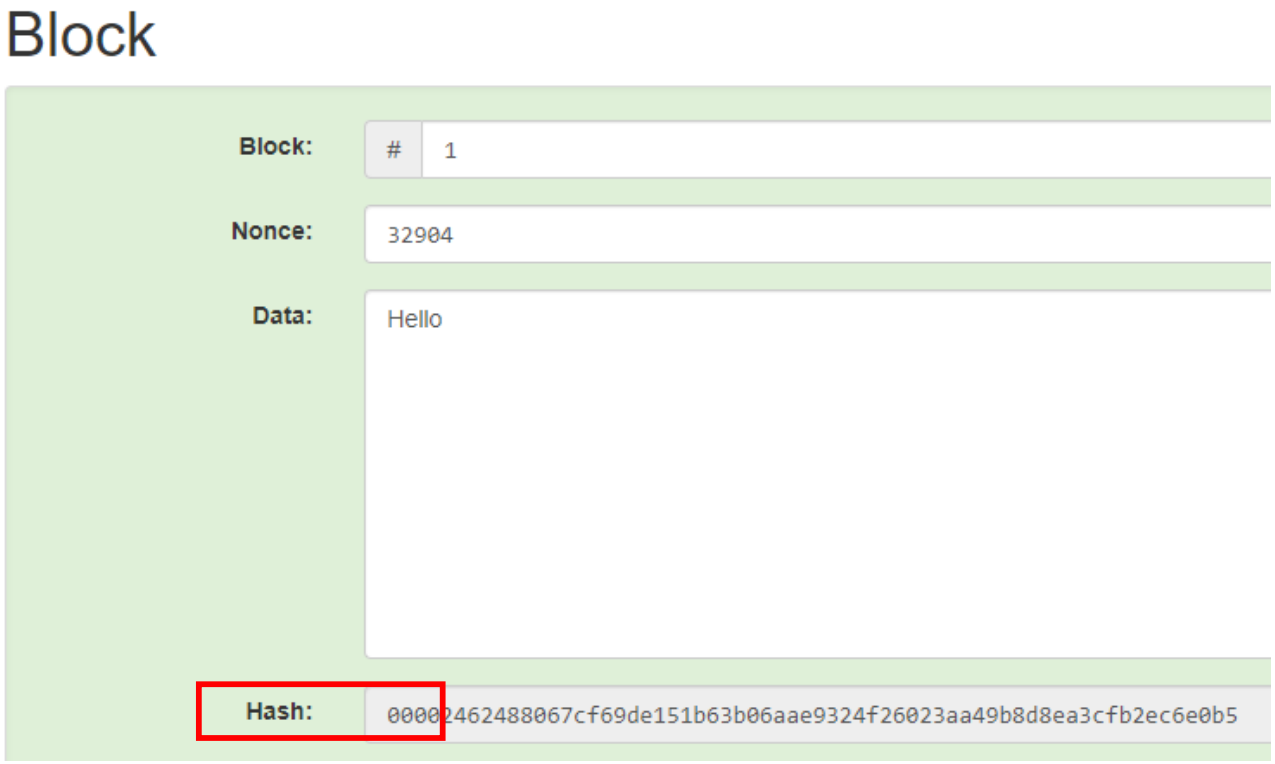


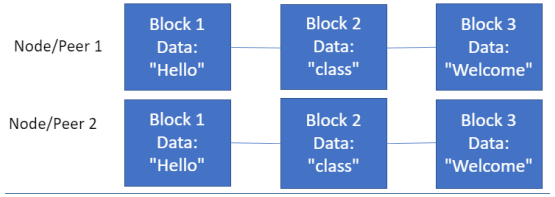
Blockchain: What are Block Mining and Signing?

hochschule hof
University of Applied Sciences

Joe Oakes

- **Mining** is the process of trying different Nonce values until the hash value matches the difficulty rule set
- For this example a valid hash must start with four zeros - **signed**



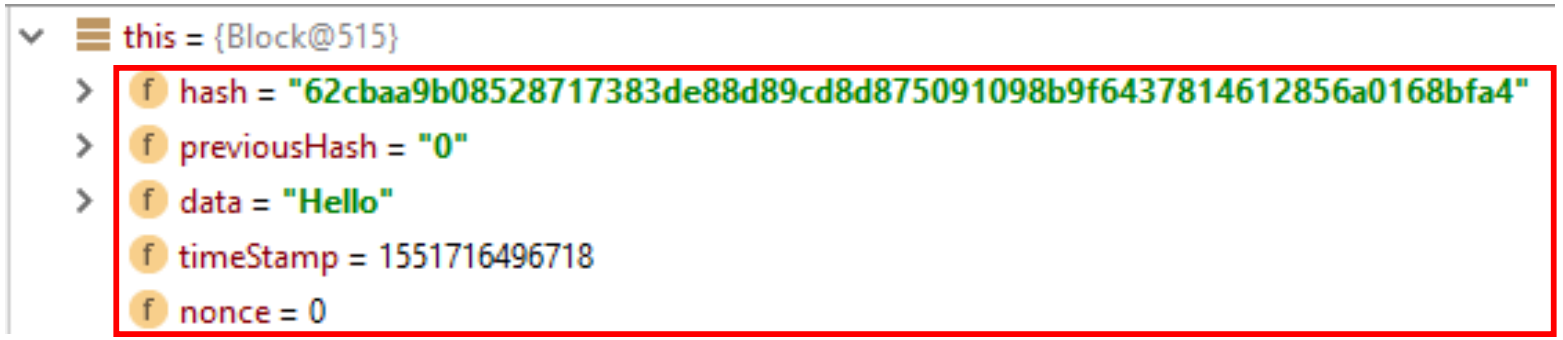


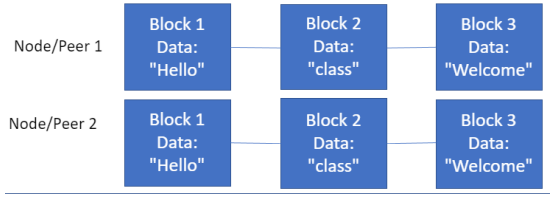
Blockchain: Computing the Block Hash

hochschule hof
University of Applied Sciences
Joe Oakes

Use the **previous hash** + **timestamp** + **nonce** + **data** to generate the block hash

```
public Block(String data,String previousHash ) {
        this.data = data;
        this.previousHash = previousHash;
        this.timeStamp = new Date().getTime();
        this.hash = calculateHash();
}
```





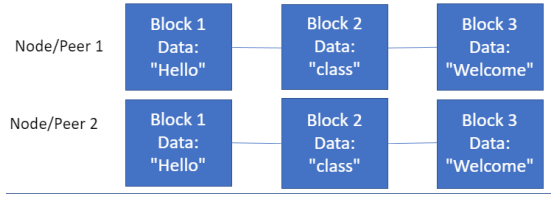
Blockchain: Computing the Block Hash

hochschule hof
University of Applied Sciences

- The CalculateHash() Method concatenates the **previous hash** + **timestamp** + **nonce** + **data** as a string and sends to applySha256() method to generate the block hash

```
//Calculates the hash of the new block object being created.
//Uses custom hash method stored in StringUtil.java.
//Takes input of previous block's hash, the timestamp of the new block's creation,
//the nonce of the block since this is a cryptocurrency mining simulator,
//and the block's data, which is a plaintext message.
public String calculateHash() {
    String calculatedhash = StringUtil.applySha256(
            input: previousHash +
                    Long.toString(timeStamp) +
                    Integer.toString(nonce) +
                    data
    );
    return calculatedhash; //returns the new calculated hash.
}
```

```
this = {Block@491}
  hash = null
  previousHash = "0"
  data = "Hello"
  timeStamp = 1551720474038
  nonce = 0
calculatedhash = "263102a607ca2221b82841c1dafbb9e9980905d57a5055d8a107a58a9b84273d"
```



Blockchain: Computing the Block Hash

hochschule hof
University of Applied Sciences

Joe Oakes

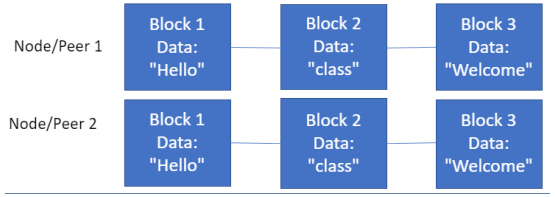
```java
//Applies Sha256 to a string and returns the result.
public static String applySha256(String input){

    try {
        MessageDigest digest = MessageDigest.getInstance("SHA-256");

        //Applies sha256 to our input,
        byte[] hash = digest.digest(input.getBytes( charsetName: "UTF-8"));

        StringBuffer hexString = new StringBuffer(); // This will contain hash as hexidecimal
        for (int i = 0; i < hash.length; i++) {
            String hex = Integer.toHexString( i: 0xff & hash[i]);
            if(hex.length() == 1) hexString.append('0');
            hexString.append(hex);
        }
        return hexString.toString();
    }
    catch(Exception e) {
        throw new RuntimeException(e);
    }
}
```

```
this = {Block@491}
  f hash = null
  f previousHash = "0"
  f data = "Hello"
  f timeStamp = 1551720474038
  f nonce = 0
  calculatedhash = "263102a607ca2221b82841c1dafbb9e9980905d57a5055d8a107a58a9b84273d"
```



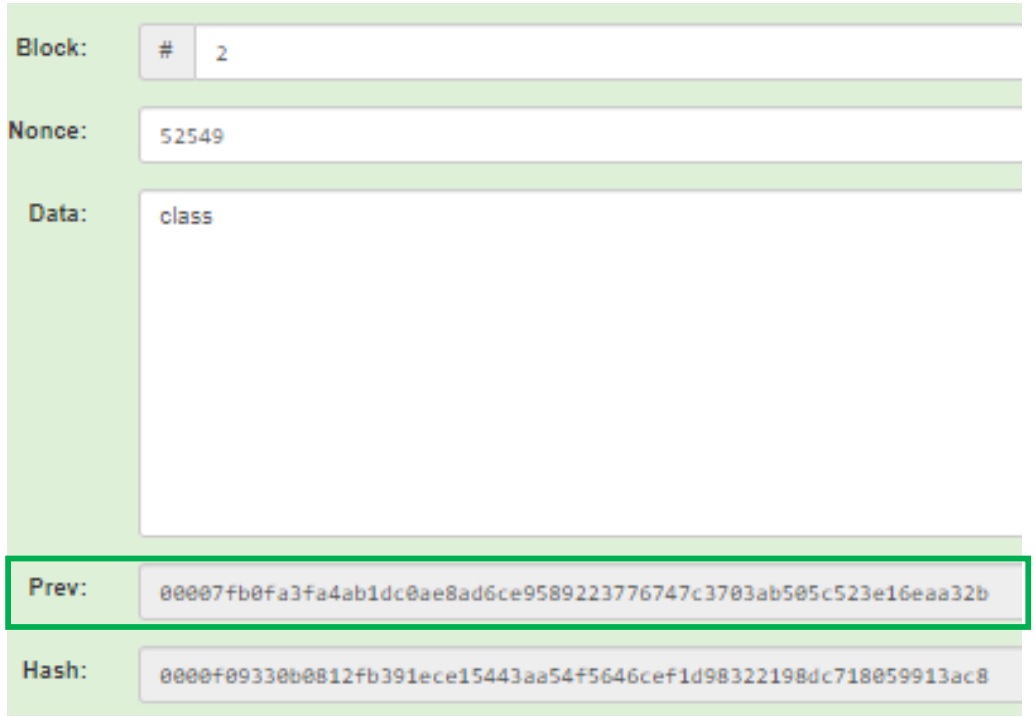
Blockchain: What is a Blockchain?

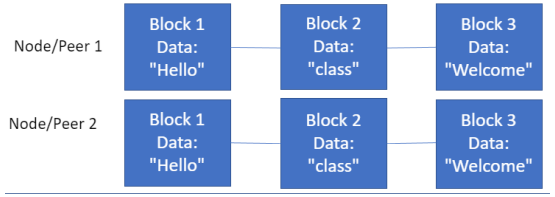
hochschule hof
University of Applied Sciences

Joe Oakes

- Blocks are chained together - linked list
- The two blocks are linked using the hash pointer

Genesis Block



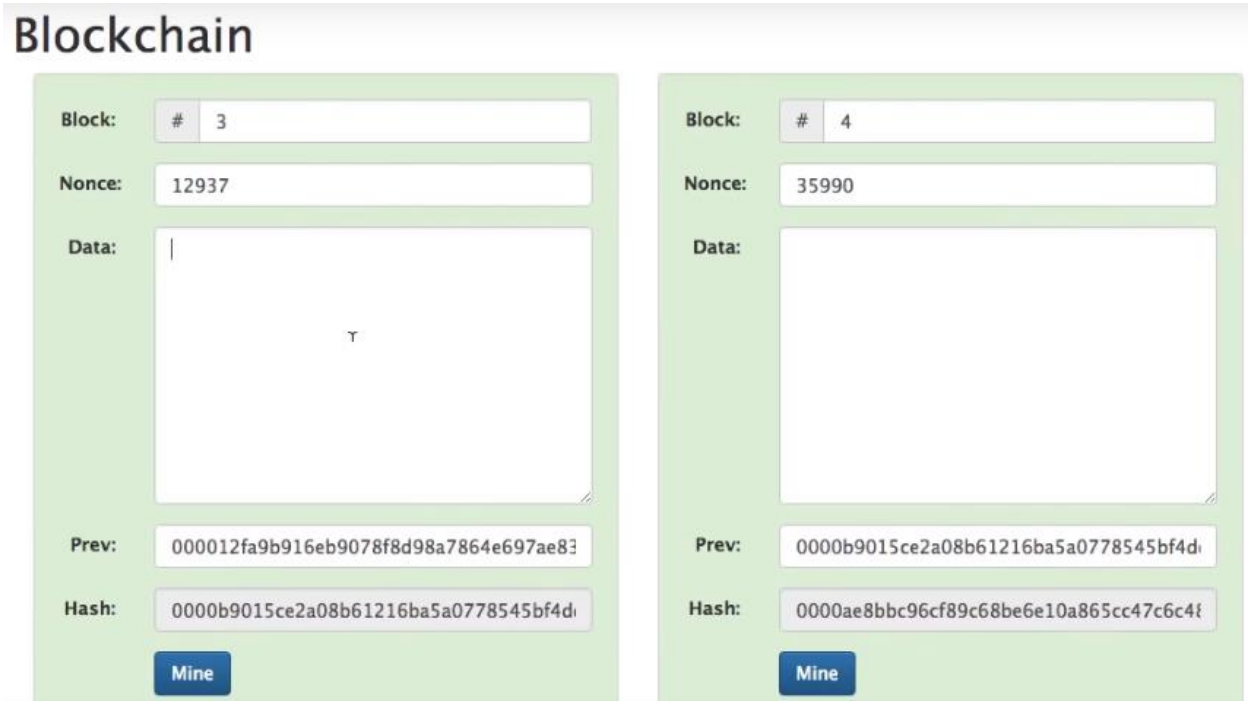


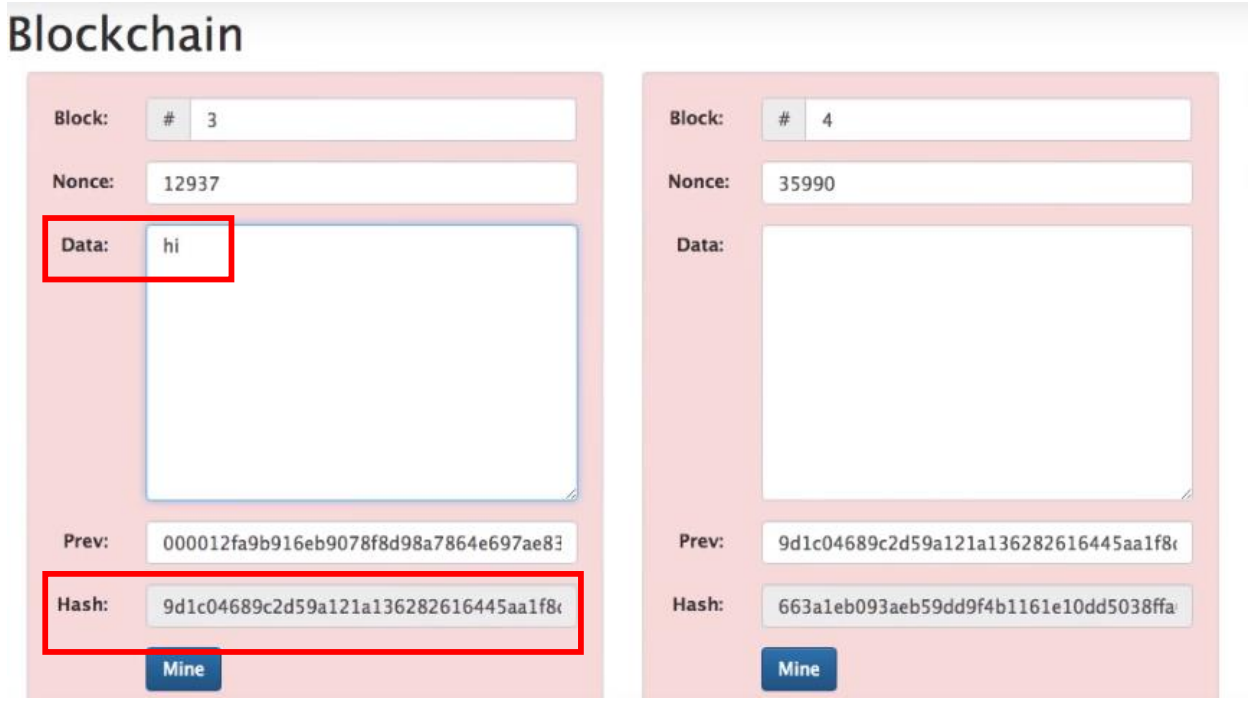
Blockchain: What is a Blockchain?

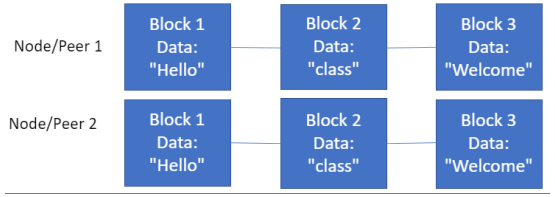
hochschule hof
University of Applied Sciences

Joe Oakes

- Changing any data in this list, will change the signature and will break the chain
- If any previous blocks were modified, then any blocks after it would all have to be mined again and would result in a different ending hash value





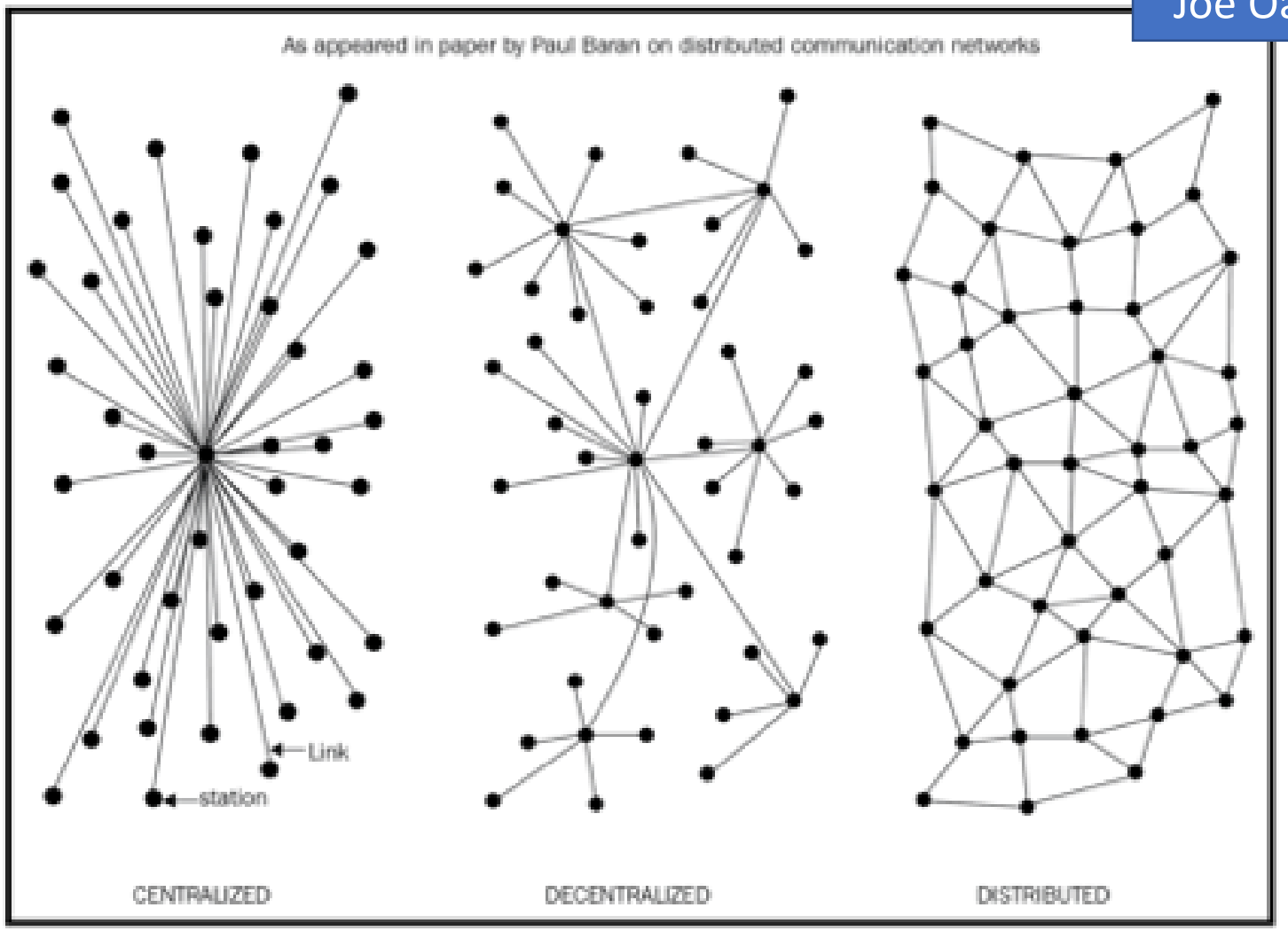


hochschule hof
University of Applied Sciences

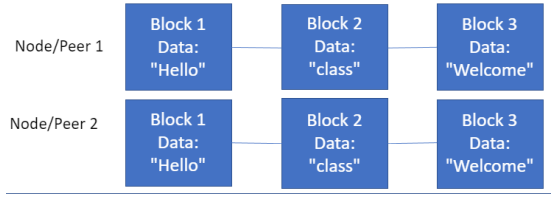
Joe Oakes

What is a decentralized network?

The key difference between a decentralized system and a distributed system is that in a distributed system, there still exists a central authority that governs the entire system, whereas in a decentralized system, no such authority exists.



Different types of network/system



Blockchain: What is a Distributed Blockchain?

hochschule hof
University of Applied Sciences

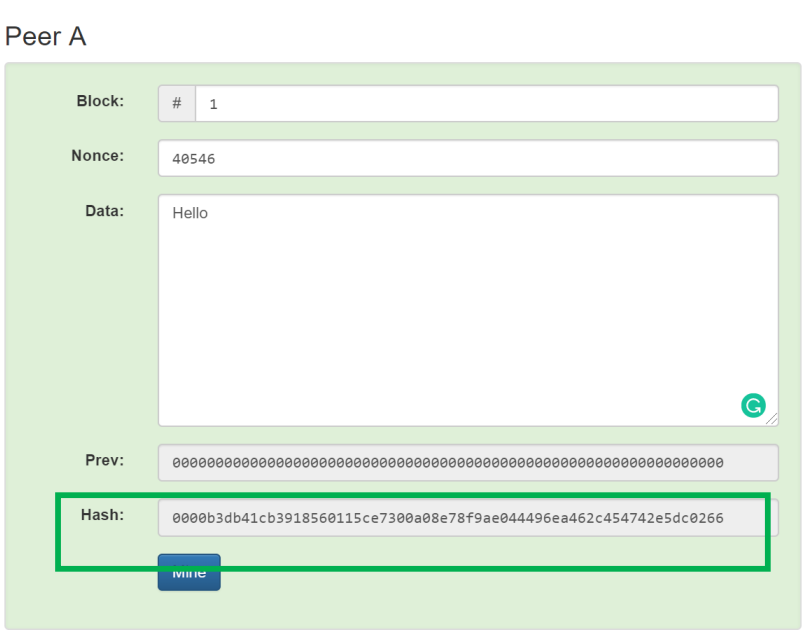
Joe Oakes

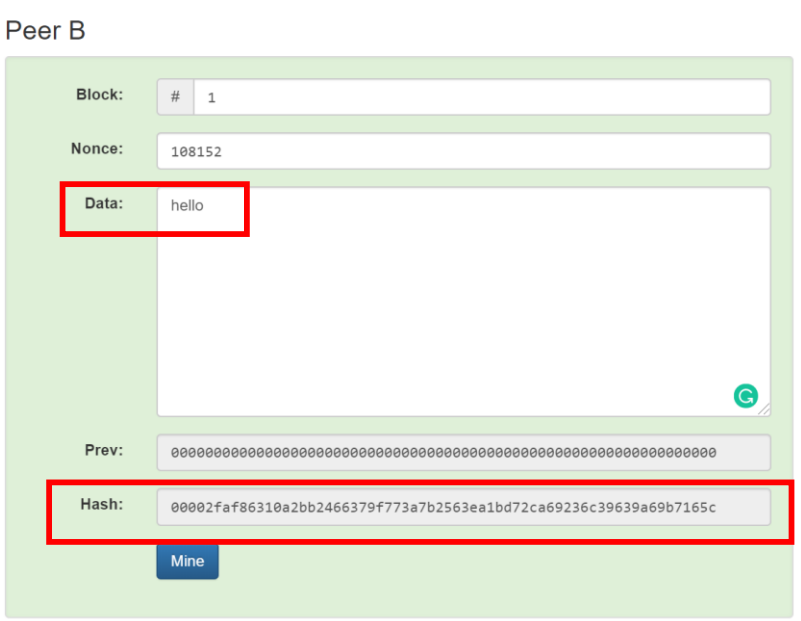
- Peers should have an exact complete copies of the blockchain and would know the if hash doesn't match and that it has been modified

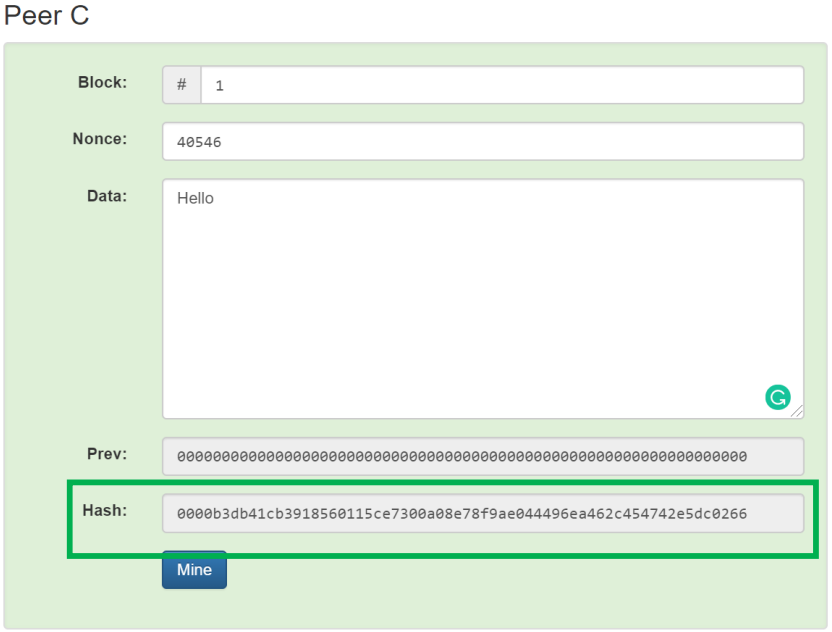
Peer A – hash matches

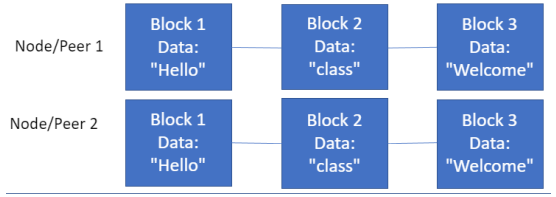
Peer B – hash does not match
Data has been modified

Peer C – hash matches







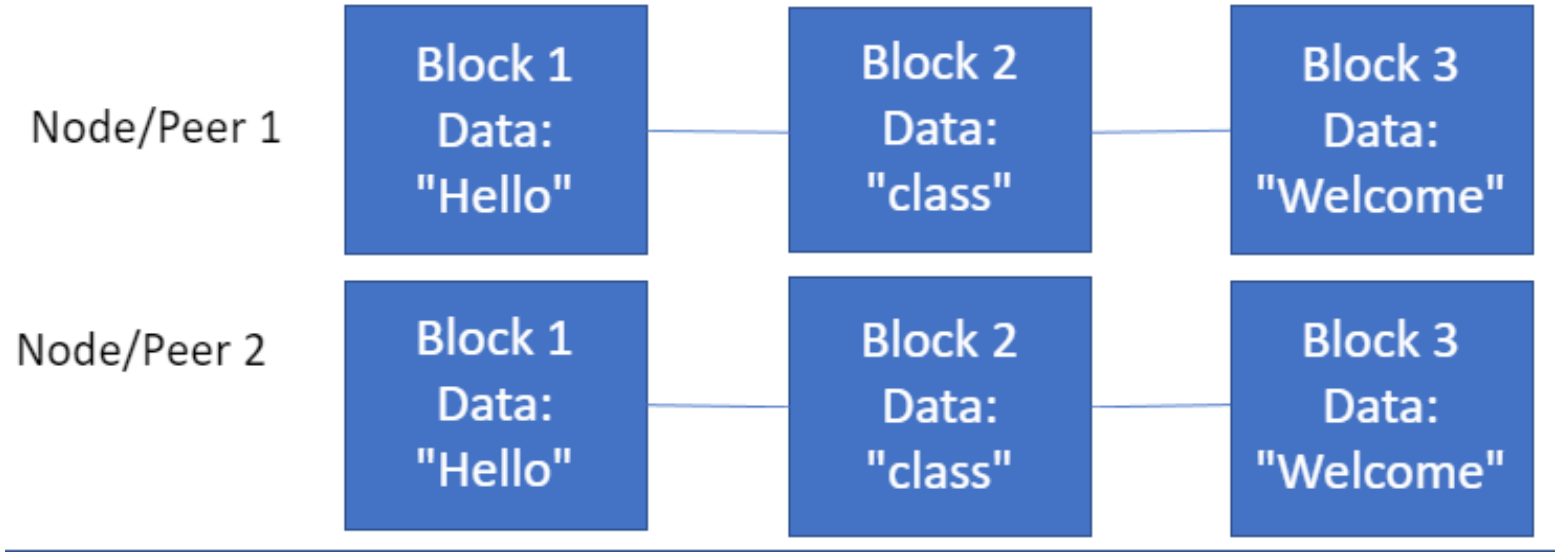


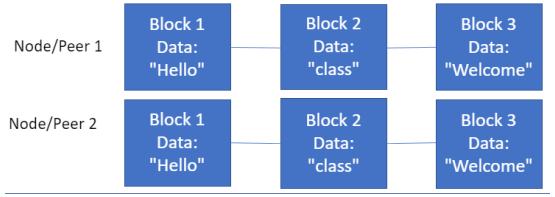
Blockchain: Nodes

hochschule hof
University of Applied Sciences

Joe Oakes

- A full node downloads a complete copy of a blockchain and checks any new transactions coming in based on the consensus protocol utilized by that particular cryptocurrency or utility token.
- It is the nodes on the network that confirm and validate transactions, putting them into blocks.
- A node can either be a communication endpoint or a point of communication redistribution, linking to other nodes.
- Every node on the network is considered equal, however certain nodes have different roles in the manner in which they support the network.



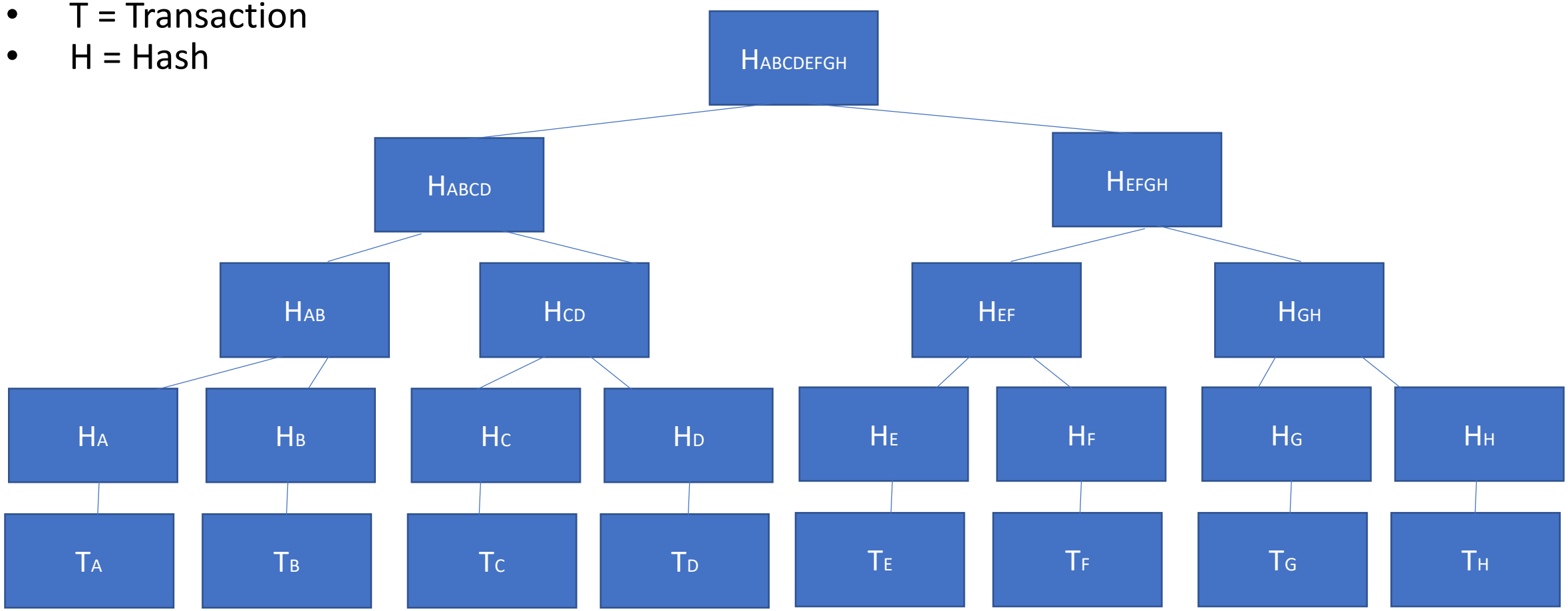


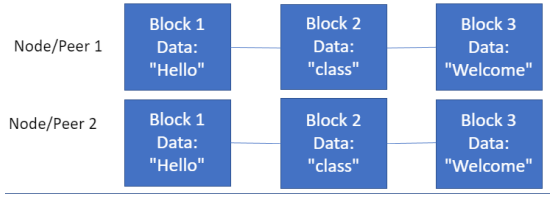
Blockchain: What is a Merkle Tree?

hochschule hof
University of Applied Sciences

Joe Oakes

- Merkle Tree
  - T = Transaction
  - H = Hash



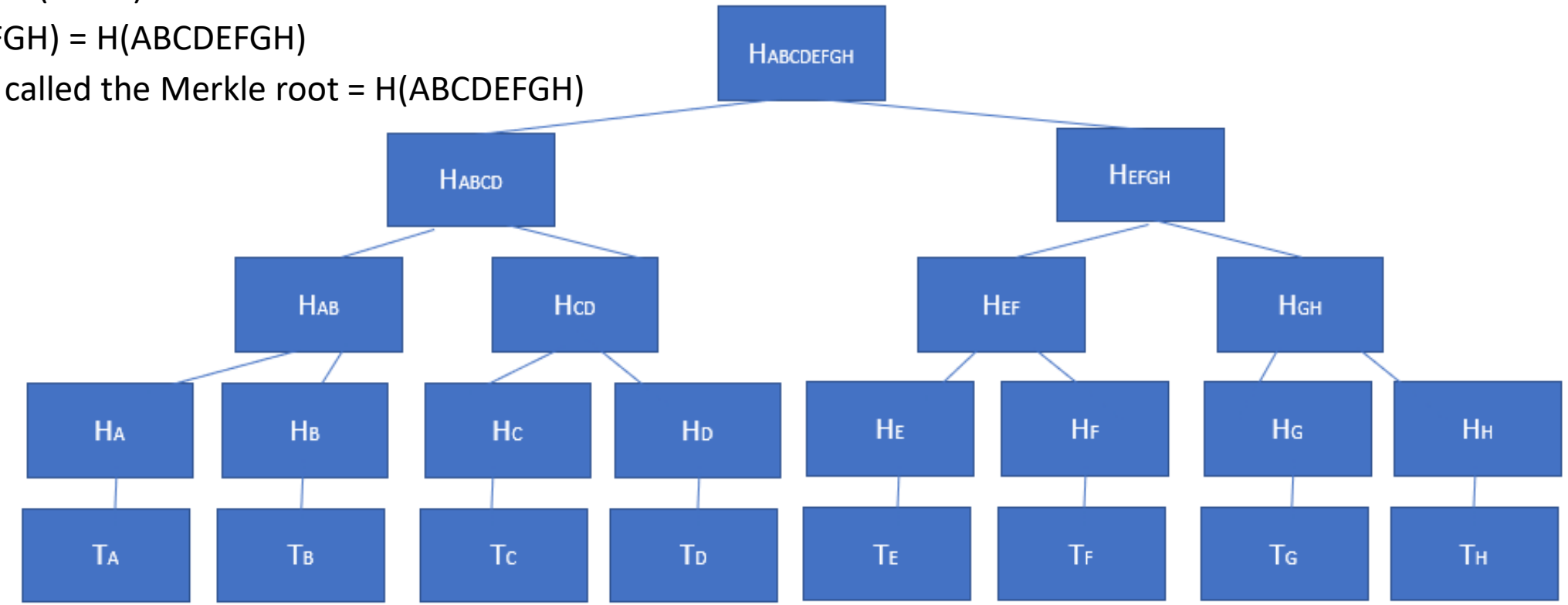


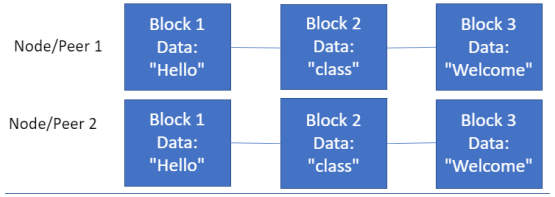
Blockchain: What is a Merkle Tree?

hochschule hof
University of Applied Sciences

Joe Oakes

- Having many transaction hashes can start to be problem for example each bitcoin block contains around 2,000 transactions
- This can cause bandwidth transmission performance issues
- A merkle tree solves this problem by pairing transactions up and hashing them together
- H(A) + H(B) = H(AB) and H(C)+H(D)=H(CD)
- H(AB) + H(CD) = H(ABCD)
- H(ABCD) + H(EFGH) = H(ABCDEFGH)
- A single hash is called the Merkle root = H(ABCDEFGH)



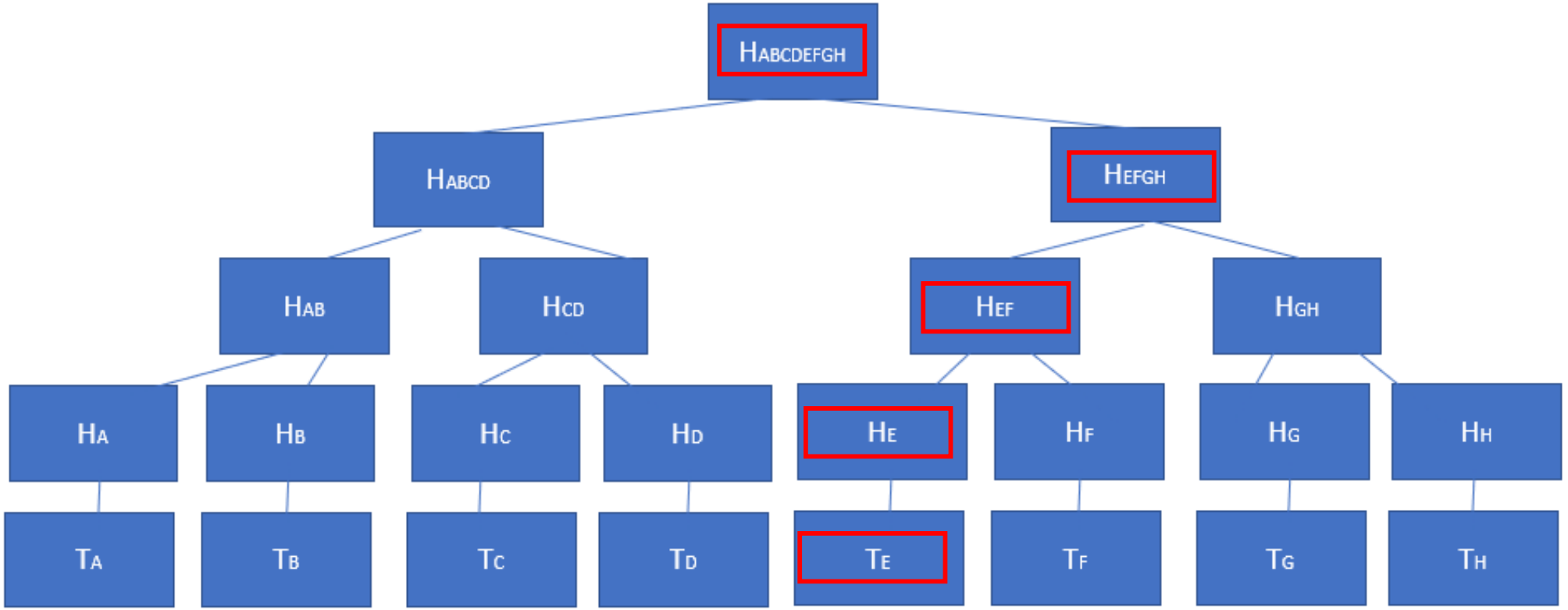


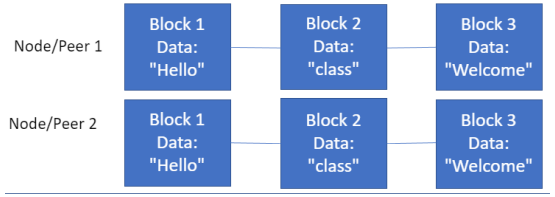
Blockchain: What is a Merkle Tree?

hochschule hof
University of Applied Sciences

Joe Oakes

- There are 8 leaves in this tree
- All the leaves in the tree depends on other leaves
- We construct the tree from the bottom up by pairing each leaf
- It is very complicated to cheat a blockchain
- Changing one transaction means you must change everything



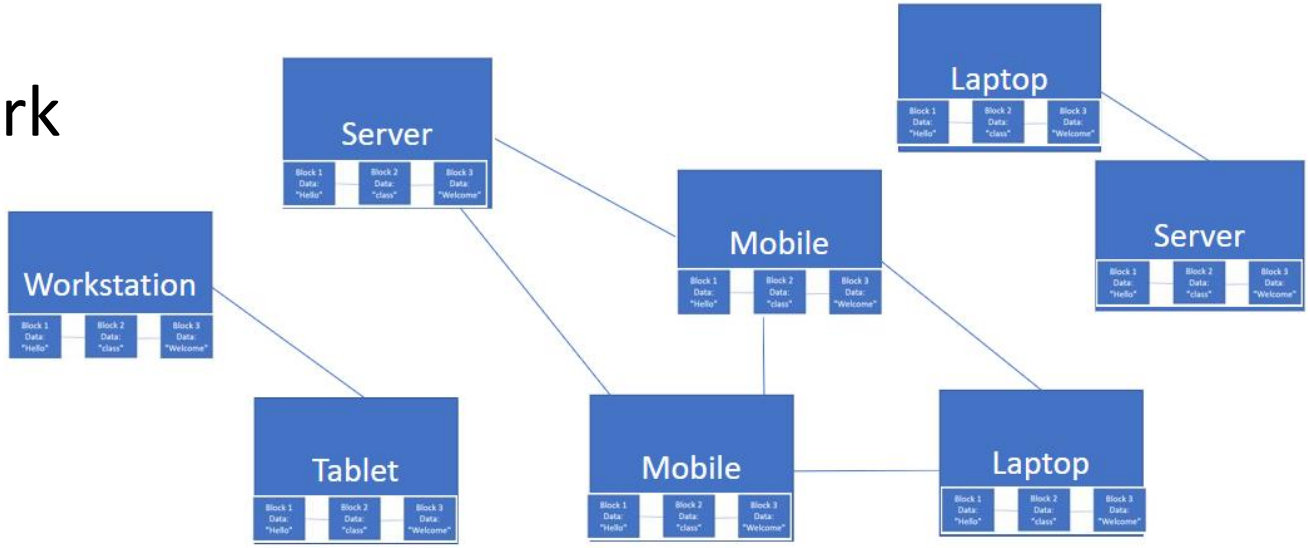


Blockchain: Review P2P Network, Block

hochschule hof
University of Applied Sciences

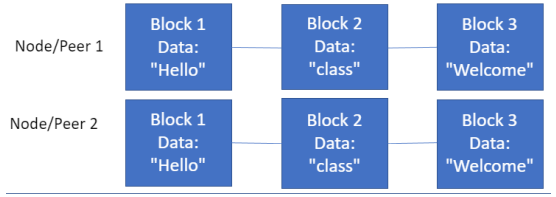
Joe Oakes

Decentralized **P2P** Network



Classification of a **Block**

Block Classification
Index, Hash, Previous Hash, Data: "Hello", Timestamp, Nonce



Blockchain: Review Data Blocks, Timestamp

hochschule hof
University of Applied Sciences

Joe Oakes

Data blocks are chained together

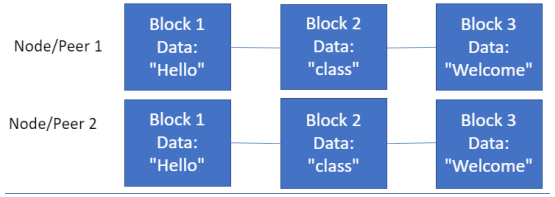
Block 1
Hash: 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969
Previous Hash: 0
Data: "Hello"

Block 2
Hash: 0889113e04d3203f0c401c17c0fd8b352b740dc607433779d3edcaa13320b001
Previous Hash: 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969
Data: "class"

Block 3
Hash: 0e2226b5235f0ff94a276eb4d07a3bfea74b7e3b8b85e9efca6c18430f041bf8
Previous Hash: 0889113e04d3203f0c401c17c0fd8b352b740dc607433779d3edcaa13320b001
Data: "Welcome"

Timestamp uses Unix time

```
this = {Block@515}
  f hash = "62cbaa9b08528717383de88d89cd8d875091098b9f6437814612856a0168bfa4"
  f previousHash = "0"
  f data = "Hello"
  f timeStamp = 1551716496718
  f nonce = 0
```



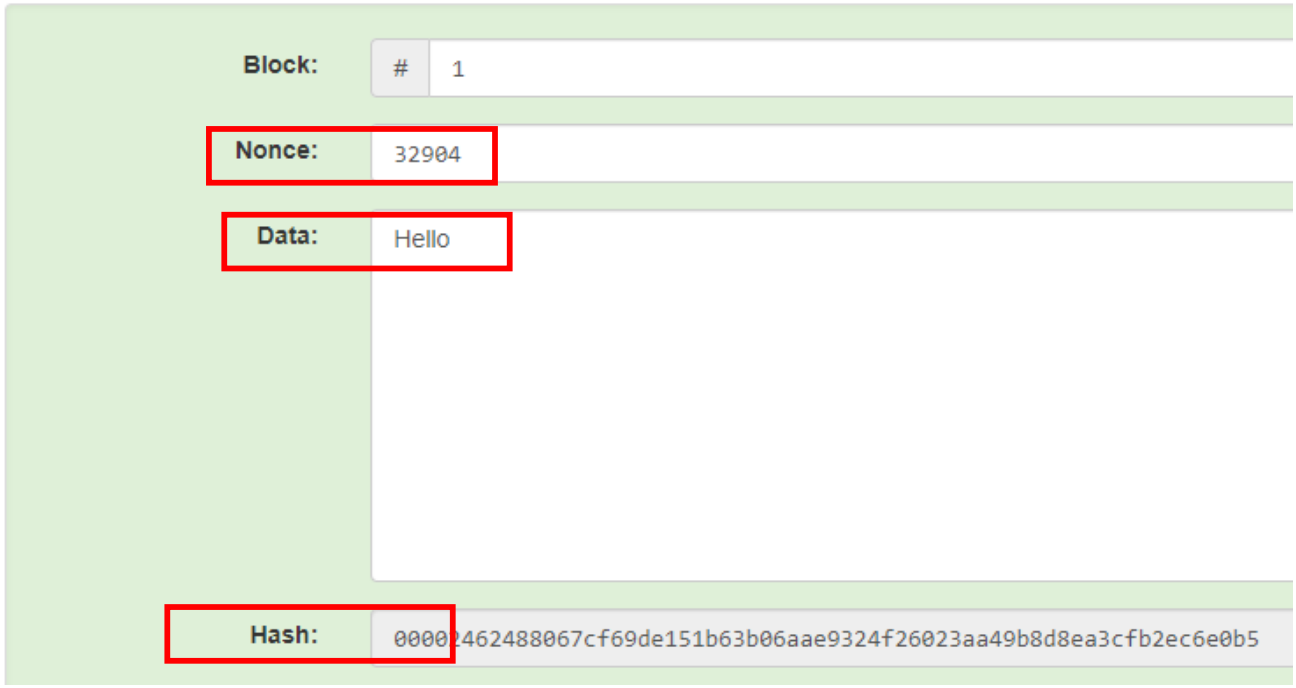
Blockchain: Review Mining, Nonce, Difficulty

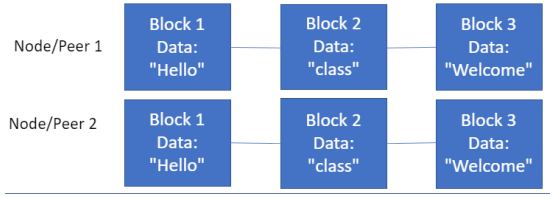
hochschule hof
University of Applied Sciences

Joe Oakes

Mining a finding a **nonce** number that matches the proper difficulty based on the data

Difficulty is how hard it is to compute the proper hash value



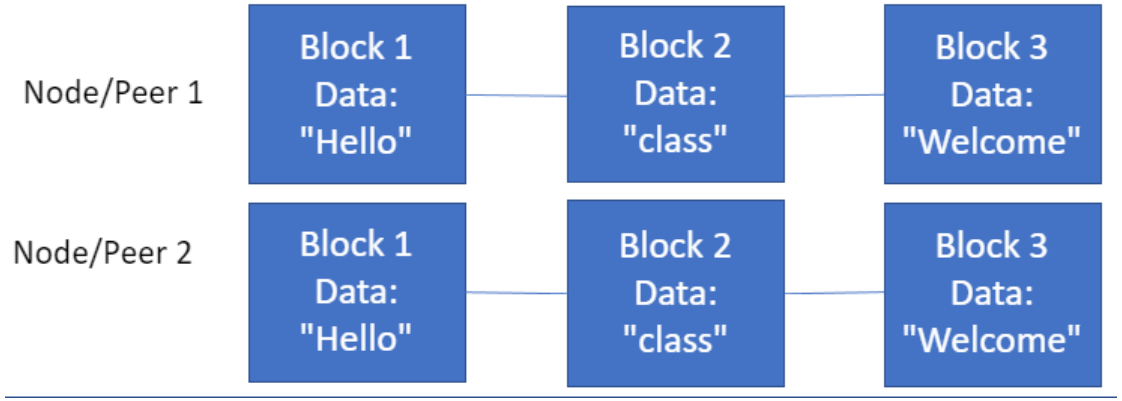


Blockchain: Review Nodes, Merkel Tree

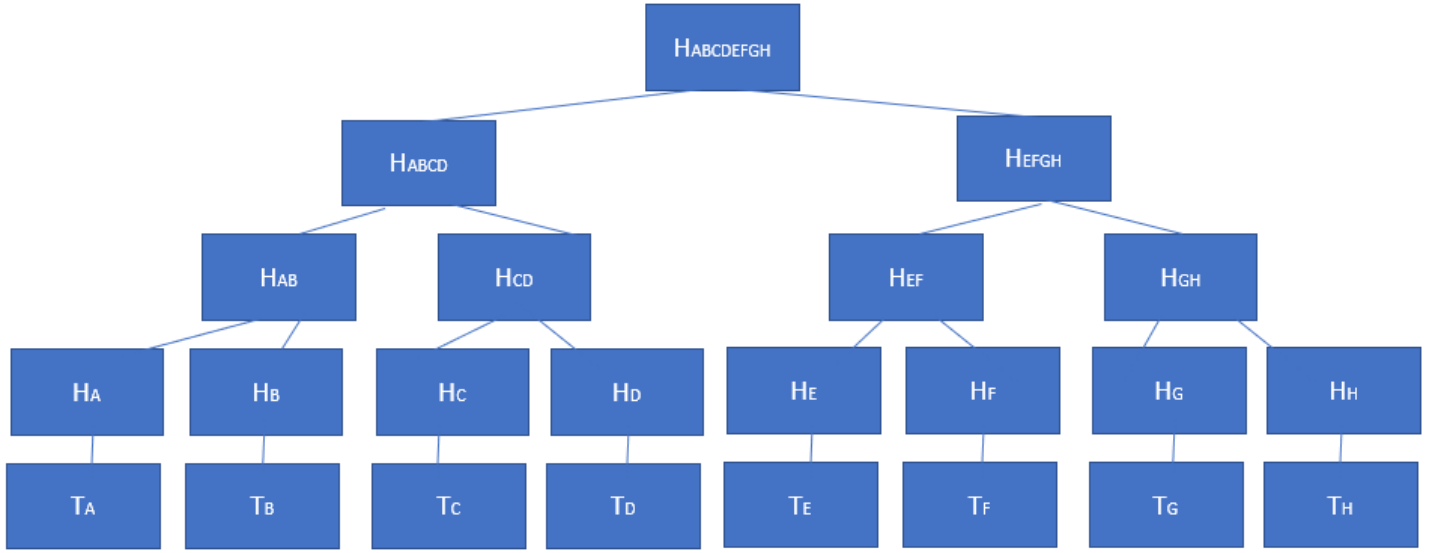
hochschule hof
University of Applied Sciences

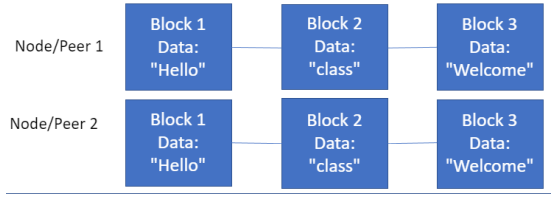
Joe Oakes

Nodes on the Network



Merkel Tree





Blockchain: Student Lab Work

hochschule hof
University of Applied Sciences

Joe Oakes

- Download the reNoobChain from github
- Clone the project using IntelliJ – need git installed
- GSON.jar is in the root folder google JSON library
- Add a new POJO for data containers: passport, real estate, pizza order, amazon item
- Modify reNoobChain.java and add the other blocks to the chain
- Change the difficulty to be 6,7,8 zeros does it take more time to process?
- Change the difficulty to be 2,3,4 zeros does it take less time to process?