# Lecture 02 - What is a BlockChain - more on Go.

## News

1. Canadian Steel / Supply chain tracking on blockchain.
2. LinkedIn ranks blockchain as the number two technology skill companies need in 2020.
3. Digital Dollar – from fed report could effect stability of dollar.
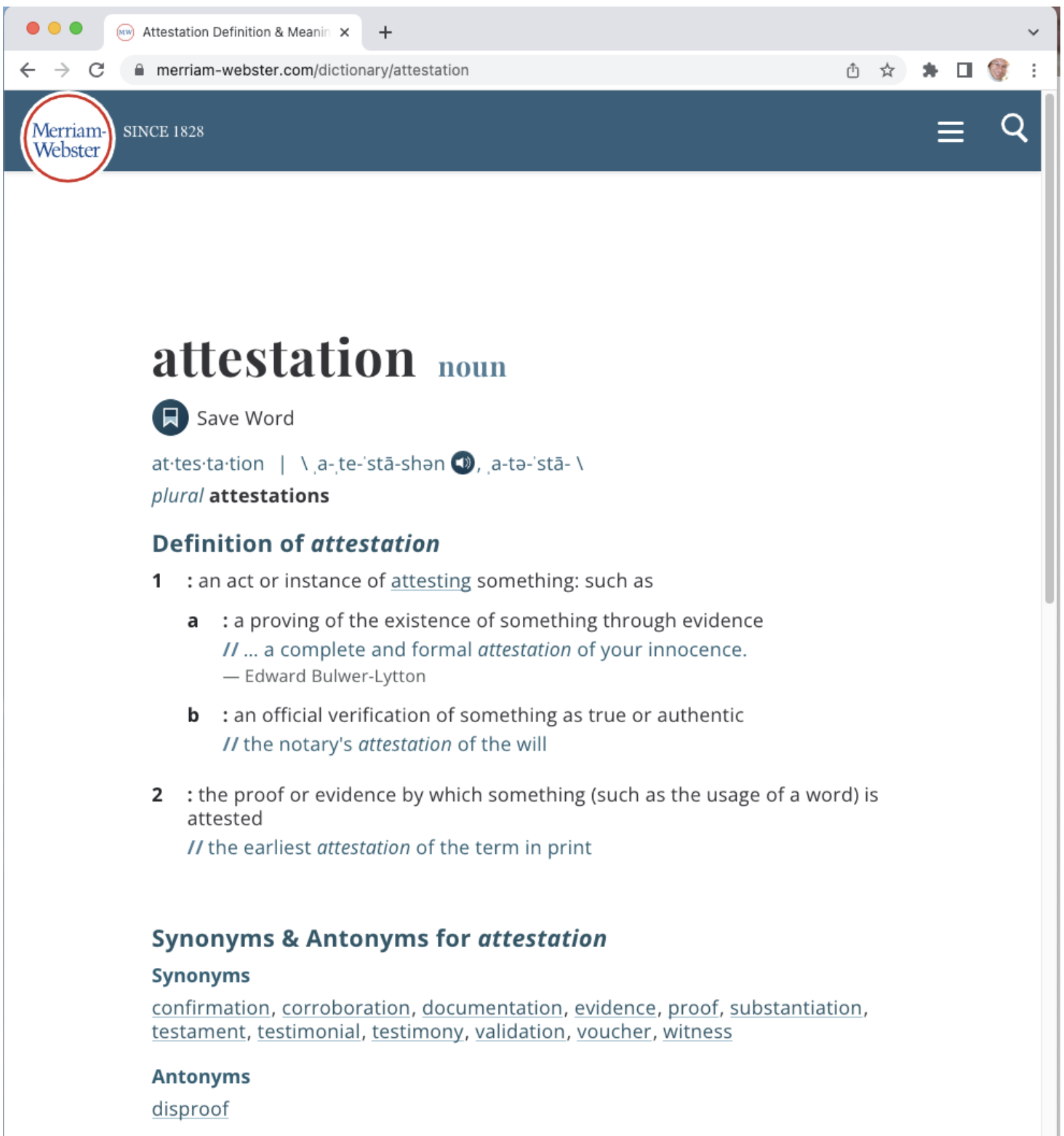4. NFTs - OpenSea trades 4.5 billion in one month.

## Question: What is a hash?

1. (math) a mapping form a range to a domain.

2. Text to a number.

```
abc               275f20377d6574b67399702947cb56849d2e02f7112c1d021603346c345b37f8
abd               3212601953780d3a8de118531b87bf6183edb8c81baf6982fdca260033a5f29b
war-and-peace.txt 67c570b0e09d70225d739aec9a7ea91631df1ea06ba44f0c9d9fe99e45f41756
```

3. Different kinds of hash, MD5, SHA1, SHA256, SHA3, SHA512, Keccak256.

## attestation - Definition of attestation - From Websters Dictionary.

## attestation noun

Save Word

at·tes·ta·tion | \ ˌa-ˌte-ˈstā-shən 🔊, ˌa-tə-ˈstā- \
*plural* **attestations**

### Definition of *attestation*

**1** : an act or instance of attesting something: such as

  **a** : a proving of the existence of something through evidence
  *//* ... a complete and formal *attestation* of your innocence.
  — Edward Bulwer-Lytton

  **b** : an official verification of something as true or authentic
  *//* the notary's *attestation* of the will

**2** : the proof or evidence by which something (such as the usage of a word) is
attested
*//* the earliest *attestation* of the term in print

### Synonyms & Antonyms for *attestation*

**Synonyms**

confirmation, corroboration, documentation, evidence, proof, substantiation,
testament, testimonial, testimony, validation, voucher, witness

**Antonyms**

disproof

https://www.merriam-webster.com/dictionary/attestation

1. an act or instance of attesting something: such as
    1. : a proving of the existence of something through evidence ... a complete and formal attestation of your innocence. —
       Edward Bulwer-Lytton
    2. : an official verification of something as true or authentic the notary's attestation of the will
2. the proof or evidence by which something (such as the usage of a word) is attested the earliest attestation of the term in
   print

## Economics of Blockchain

This is an up/down year in the cryptocurrency world. China, India and a few other countries have banned it. On the flip side the US and about $\frac{1}{2}$ of the world are looking to replace national currencies with it.

In 2009 - bitcoin invented.

In 2013 - smart contracts - Ethereum.

In 2021 - 6% of transactions in 6 African countries on it. China and Russia are moving to replace national currency with it. Goldman Sacks / Apple are using it. Visa expects $1T in transactions on it.

Ability to create trust between non-trusting parties.

Ability to create economic systems.

Merick - shipping 40% decrease in shipping times. World wide $380 Billion in trade. 90% of all the goods in the world are moved by ship. 38 days average shipping time. A 40% reduction is dropping that to 23 days. Difference is 15 days. 40% of 380 billion is 152 billion in capital that is not tied up - at an average capital cost of 10% = 15.2 billion - over the 23 days. $\frac{15}{365}$ * 15.2 billion. -- About 0.62 billion.

Estonia - All titles and property on the chain. In US 5.51 Million Houses. Average title search $3821. Title search in Estonia, $23.33 each. Title insurance $1408. My calculation $19 billion a year.

Marshal Islands - off of the dollar and onto a blockchain. The estimate is that the government will save around $5M a year or about $90 per resident a year. For entire US 327 million - that is $29 billion dollars.

Over 22 accredited educational educational institutions now issuing certificates based on blockchain. (Especial Interest in this because this is our project)

Perspective on 28 or 29 billion - free college tuition for all students in the United States is estimated to cost 75 billion.

State of Nevada - Marriage license on blockchain.

The largest "blockchain" sale is valued at 103 million. In a commercial property in Zurich.

# Go - Intro

Assignment 1 - Due Mon Feb 3

## Hello World - walk through

```
1: package main
2:
3: import "fmt"
4:
5: func main() {
6:     fmt.Printf("Hello Silly World\n")
7:     fmt.Printf("שלום עולם טיפשי\n")
8: }
```

## Echo - walk through

```
 1: package main
 2:
 3: import (
 4:     "fmt"
 5:     "os"
 6: )
 7:
 8: func main() {
 9:     for i, s := range os.Args {
10:         if i == 0 {
11:         } else if i == len(os.Args)-1 {
12:             fmt.Printf("%s\n", s)
13:         } else {
14:             fmt.Printf("%s ", s)
15:         }
16:     }
17: }
```

1st time - or when you change dependencies

```
$ mkdir echo
$ cd echo
$ vi main.go
$ go mod init
$ go mod tidy
$ go build
```

After that

```
$ go build
```

# Marshal and Unmarshal of data - walk through

```
 1: package main
 2:
 3: import (
 4:     "encoding/json"
 5:     "fmt"
 6: )
 7:
 8: type Demo struct {
 9:     Aa int
10:     Ab string
11: }
12:
13: func main() {
14:     d := Demo{
15:         Aa: 33,
16:         Ab: "Penguines are People Too...",
17:     }
18:
19:     buf, err := json.Marshal(d)
20:     if err != nil {
21:         fmt.Printf("Error: %s\n", err)
22:     }
23:
```

```
24:     fmt.Printf("%s\n", buf)
25:
26:     buf, err = json.MarshalIndent(d, "", "\t")
27:     if err != nil {
28:         fmt.Printf("Error: %s\n", err)
29:     }
30:
31:     fmt.Printf("%s\n", buf)
32: }
```

```
 1: package main
 2:
 3: import (
 4:     "encoding/json"
 5:     "fmt"
 6: )
 7:
 8: type Demo struct {
 9:     Aa int `json:"A_cx"`
10:     Ab string
11: }
12:
13: func main() {
14:     s := `{
15:         "A_cx": 33,
16:         "Ab": "Penguines are People Too...",
17:         "Ac": "skips this, no error"
18:     }`
19:
20:     var d Demo
21:     err := json.Unmarshal([]byte(s), &d)
22:     if err != nil {
23:         fmt.Printf("Error: %s\n", err)
24:     }
25:
26:     fmt.Printf("%+v\n", d)
27: }
```

## For loops

```
 1: package main
 2:
 3: import "fmt"
 4:
 5: var aSlice = []string{"abc", "def", "ghi"}
 6: var aMap = map[string]int{
 7:     "alice": 22,
 8:     "bob":   23,
 9:     "tom":   25,
10: }
11:
12: func main() {
13:     for i := 0; i < 5; i++ {
14:         fmt.Printf("Loop 1: %d\n", i)
15:     }
16:     fmt.Printf("\n")
17:
18:     for i, v := range aSlice {
19:         fmt.Printf("Loop 2: at:%d ->%s<-\n", i, v)
```

```
20:     }
21:     fmt.Printf("\n")
22:
23:     for key, val := range aMap {
24:         fmt.Printf("Loop 3: key:%s ->%v<-\n", key, val)
25:     }
26: }
```

## Functions

```
1: package main
2:
3: import "fmt"
4:
5: func Qs(ss []string) (rv []string) {
6:
7:     partition := func(arr []string, low, high int) ([]string, int) {
8:         pivot := arr[high]
9:         i := low
10:        for j := low; j < high; j++ {
11:            if arr[j] < pivot {
12:                arr[i], arr[j] = arr[j], arr[i]
13:                i++
14:            }
15:        }
16:        arr[i], arr[high] = arr[high], arr[i]
17:        return arr, i
18:     }
19:
20:     var quickSort func(arr []string, low, high int) []string
21:     quickSort = func(arr []string, low, high int) []string {
22:         if low < high {
23:             var p int
24:             arr, p = partition(arr, low, high)
25:             arr = quickSort(arr, low, p-1)
26:             arr = quickSort(arr, p+1, high)
27:         }
28:         return arr
29:     }
30:
31:     rv = quickSort(ss, 0, len(ss)-1)
32:     return
33: }
34:
35: func main() {
36:     r := Qs([]string{"def", "ghi", "abc", "ddd", "zzz"})
37:     fmt.Printf("%v\n", r)
38: }
```

```
1: package main_test
2:
3: import (
4:     "reflect"
5:     "testing"
6:
7:     main "github.com/Univ-Wyo-Education/S22-4010/class/lect/02/funcDemo"
8: )
9:
10: func Test_Qs(t *testing.T) {
```

```
11:      expect := []string{"abc", "ddd", "def", "ghi", "zzz"}
12:      data := []string{"def", "ghi", "abc", "ddd", "zzz"}
13:
14:      rv := main.Qs(data)
15:      if len(rv) != len(expect) {
16:          t.Errorf("Expected %v got %v\n", expect, rv)
17:      }
18:      if !reflect.DeepEqual(rv, expect) {
19:          t.Errorf("Expected %v got %v\n", expect, rv)
20:      }
21:
22: }
```