Golang Session

- Harsh Dusane

Topic : Files and Directories

Create an empty file

```
package main

import (
        "log"
        "os"
)

func main() {
        emptyFile, err := os.Create("empty.txt")
        if err != nil {
                log.Fatal(err)
        }
        log.Println(emptyFile)
        emptyFile.Close()
}
```

Go program to Create directory or folder if not exist

```
package main

import (
        "log"
        "os"
)

func main() {
        _, err := os.Stat("test")

        if os.IsNotExist(err) {
                errDir := os.MkdirAll("test", 0755)
                if errDir != nil {
                        log.Fatal(err)
                }

        }
}
```

Rename a file in Golang

```
package main

import (
        "log"
        "os"
)

func main() {
        oldName := "test.txt"
        newName := "testing.txt"
        err := os.Rename(oldName, newName)
        if err != nil {
                log.Fatal(err)
        }
}
```

Move a file from one location to another in Golang

```
package main

import (
        "log"
        "os"
)

func main() {
        oldLocation := "/var/www/html/test.txt"
        newLocation := "/var/www/html/src/test.txt"
        err := os.Rename(oldLocation, newLocation)
        if err != nil {
                log.Fatal(err)
        }
}
```

Golang Create Copy of a file at another location

```
package main

import (
        "io"
        "log"
        "os"
)

func main() {

        sourceFile, err := os.Open("/var/www/html/src/test.txt")
        if err != nil {
                log.Fatal(err)
        }
        defer sourceFile.Close()

        // Create new file
        newFile, err := os.Create("/var/www/html/test.txt")
        if err != nil {
                log.Fatal(err)
        }
        defer newFile.Close()

        bytesCopied, err := io.Copy(newFile, sourceFile)
        if err != nil {
                log.Fatal(err)
        }
        log.Printf("Copied %d bytes.", bytesCopied)
}
```

Get file information in Golang

```
package main

import (
        "fmt"
        "log"
        "os"
)

func main() {
        fileStat, err := os.Stat("test.txt")

        if err != nil {
                log.Fatal(err)
        }

        fmt.Println("File Name:", fileStat.Name())        // Base name of the file
        fmt.Println("Size:", fileStat.Size())             // Length in bytes for regular files
        fmt.Println("Permissions:", fileStat.Mode())      // File mode bits
        fmt.Println("Last Modified:", fileStat.ModTime()) // Last modification time
        fmt.Println("Is Directory: ", fileStat.IsDir())   // Abbreviation for Mode().IsDir()
}
```

Golang program to delete a specific file

```
package main

import (
        "log"
        "os"
)

func main() {
        err := os.Remove("/var/www/html/test.txt")
        if err != nil {
                log.Fatal(err)
        }
}
```

Go program to read a text file character by character

```
package main

import (
        "bufio"
        "fmt"
        "io/ioutil"
        "os"
        "strings"
)

func main() {
        filename := "test.txt"

        filebuffer, err := ioutil.ReadFile(filename)
        if err != nil {
                fmt.Println(err)
                os.Exit(1)
        }
        inputdata := string(filebuffer)
        data := bufio.NewScanner(strings.NewReader(inputdata))
        data.Split(bufio.ScanRunes)

        for data.Scan() {
                fmt.Print(data.Text())
        }
}
```

Reduce file size

```
package main

import (
        "log"
        "os"
)

func main() {
        err := os.Truncate("test.txt", 100)

        if err != nil {
                log.Fatal(err)
        }
}
```

Go program to add or append content at the end of text file

```
package main

import (
        "fmt"
        "os"
)

func main() {
        message := "Add this content at end"
        filename := "test.txt"

        f, err := os.OpenFile(filename, os.O_RDWR|os.O_APPEND|os.O_CREATE, 0660)

        if err != nil {
                fmt.Println(err)
                os.Exit(-1)
        }
        defer f.Close()

        fmt.Fprintf(f, "%s\n", message)
}
```

Golang Changing permissions, ownership, and timestamps

```
package main

import (
	"log"
	"os"
	"time"
)

func main() {
	// Test File existence.
	_, err := os.Stat("test.txt")
	if err != nil {
		if os.IsNotExist(err) {
			log.Fatal("File does not exist.")
		}
	}
	log.Println("File exist.")

	// Change permissions Linux.
	err = os.Chmod("test.txt", 0777)
	if err != nil {
		log.Println(err)
	}

	// Change file ownership.
	err = os.Chown("test.txt", os.Getuid(), os.Getgid())
	if err != nil {
		log.Println(err)
	}

	// Change file timestamps.
	addOneDayFromNow := time.Now().Add(24 * time.Hour)
	lastAccessTime := addOneDayFromNow
	lastModifyTime := addOneDayFromNow
	err = os.Chtimes("test.txt", lastAccessTime, lastModifyTime)
	if err != nil {
		log.Println(err)
	}
}
```