Day 3: java NIO package

java.nio package:

This java.nio package was introduced in java 1.4 version to implement high-speed IO operations. It is an alternative to the standard IO API.

java.nio.file package:

In this package there are some following classes and Interfaces are there by using which we can create a file or folder, we can read or write from a file in much more efficient way.

java.nio.file.Path inteface
java.nio.file.Paths class
java.nio.file.Files class

java.nio.file.Path interface:

The object of this Path represents actual location of a file or folder.

In computer every file and folder in a file system can be uniquely identified by the Path object.

A Path can be an absolute or relative, an absolute path means the location/address from the root to the file or folder. where as a relative path is the location/address which is relative to some other path.

We get the Path object by the help of java.nio.file.Paths class static method called get().

Example:

```
Path p = Paths.get("d://abc"); // for folder

Path p = Paths.get("d://abc//a1.txt"); // for files
```

Note:- Getting the object of Path doesn't mean that creating a new File or folder.

After getting the object of Path we need to supply this Path object to the some of the static methods of **java.nio.file.Files** class to perform manipulation on files and folders.

Some of the static method presents in java.nio.file.Files class:

createFile(); // used to create a file

createDirectory(); // used to create a folder

List<String> readAllLines(); // here we can read in the form of List of String , here no need to take BufferedReader, it is very fast.

byte[] readAllBytes(); // here we can read in the form of byte array.

Stream<String> lines(); // read all the line from a file and return the java.util.stream.Stream object

delete(); // used to delete a file or folder

deleteIfExist(); checks before deleting a file or folder

write(Path path, byte[] bytes); // Writes bytes to a file specified by the Path object.

copy(); // to copy a file //copy and paste

move() // to move a file // cut and paste

etc.

Example: creating a new File:

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;

class Main {

    public static void main(String[] args) throws IOException {

        Path p = Paths.get("d://abc//a1.txt"); // here d:/abc folder must be there,otherwise we get an exception

       if(Files.exists(p)) {
            System.out.println("File is aready exist");
        }else {
            Path p2 = Files.createFile(p);;
            System.out.println("created a file at :" + p2);

        }
    }
}
```

Example: writing some String to the File:

```
import java.io.IOException;
import java.nio.file.*;
import java.util.*;
class Main {

    public static void main(String[] args) throws IOException {

        Path p = Paths.get("d://a1.txt");

        String msg="welcome to java";

        //writing a normal string
        Files.write(p, msg.getBytes());

        List<String> list= Arrays.asList("delhi","mumbai","kolkata","chennai");

        //writing a List of String
        //Files.write(p, list);
```

```
        //In append mode
        Files.write(p, list,StandardOpenOption.APPEND);

        System.out.println("done...");


    }
}
```

Example: Reading a file line by line

```
import java.io.IOException;
import java.nio.file.*;
import java.util.*;
class Main {

    public static void main(String[] args) throws IOException {

        Path p = Paths.get("d://a1.txt");

        List<String> list= Files.readAllLines(p);

        System.out.println("Reading from the file");
        for(String line:list) {
            System.out.println(line);
        }

    }
}
```

Example: reading a file using Stream:

```
import java.io.IOException;
import java.nio.file.*;
import java.util.stream.Stream;

class Main {

    public static void main(String[] args) throws IOException {

        Path p = Paths.get("d://a1.txt");

        Stream<String> str= Files.lines(p);

        str.forEach(line -> System.out.println(line));

    }
}
```

Example: applying map to the file: (if some specific text is there then convert it in different text)

If any Line Admin is there, then convert it as “Welcome Admin”;

```
import java.io.IOException;
import java.nio.file.*;
import java.util.stream.Stream;

class Main {

    public static void main(String[] args) throws IOException {

        Path p = Paths.get("d://a1.txt");
ches
```

```
        Stream<String> str= Files.lines(p);

        str.map(line -> {

            if(line.contains("Admin"))
                return line.replace("Admin","Welcome Admin");
            else
                return line;

        }).forEach( line -> System.out.println(line));
    }
}
```

Example: Reading from one file and writing to another file:

```
import java.io.IOException;
import java.nio.file.*;
import java.util.List;

class Main {

    public static void main(String[] args) throws IOException {

        Path sourcePath = Paths.get("d://a1.txt");

        Path dPath = Paths.get("ab.txt");

        Files.createFile(dPath);

        List<String> list = Files.readAllLines(sourcePath);

        Files.write(dPath, list);

        System.out.println("done");


    }
}
```

Example: copying a file from one location to another

```
  Path source=Paths.get("somefile.zip");
  Path dest  =Paths.get("ab2.zip");


  Files.copy(source,dest);

  System.out.println("done");
```

Note : If we use move() method in place of copy() method then from the source the files will be removed(like cut and paste option).