****

REGULAR EXPRESSIONS

****

1. Regular Expressions
2. Pattern
3. Matcher
4. Character Classes
5. Predefined Character Classes
6. Quantifiers
7. Pattern Class Split()
8. String Class split()
9. String Tokenizer

If we want to represent a Group of strings according to a Particular Pattern, then we should go for Regular Expression.

EX1: We can write a Regular Expression to represent All Mobile Numbers and emails.

1. Validation forms
2. Pattern matches applications
3. TO develop digital circuits
4. Digital circuits

****

Pattern:

1. Pattern Object Represents a compiled version of regular Expression.
2. We can create a Pattern Object by using compile() of Pattern Class.

Public static Pattern Compile(String regularexpression)

Pattern p = Pattern.complie("ab");

Matcher:

1. We can matcher objects to match the given pattern in the target String.

2. Wecan Create Matcher Objects by Using matcher() of Pattern Class.
3. Public Matcher matcher(String target);
4. Matcher m = p.matcher("ababbaab");

Methods of Matcher Object:

1. Public boolean find():
    a. It attempts to find the next match and returns true if its available. Otherwise returns false.
2. Public int start()
    a. Returns Start index of match.
3. Public int end()
    a. Returns end+1 index of match
4. Public string group();
    a. Returns a Matched pattern.

Pattern and Matcher are from java.util.regex package. And 1.4v of java.

```java
import java.util.regex.*;

public class demo {
    public static void main(String args[])
    {
        Pattern p = Pattern.compile("ab");
        // By using class name if we are call a menthod that return the
        //same object is called Static fatory mthods.

        Matcher m = p.matcher("ababbaab");
        while(m.find())
        {
            System.out.println(m.start());
        }

    }
}
```

```java
import java.util.regex.*;

public class demo {
    public static void main(String args[])
    {
        Pattern p = Pattern.compile("ab");
        // By using class name if we are call a menthod that return the
        //same object is called Static fatory mthods.

        Matcher m = p.matcher("ababbaab");
        while(m.find())
        {
            System.out.println(m.start()+"----"+m.end()+"----"+m.group());
        }

    }
}
```

Console × Problems Debug Shell

```
<terminated> demo [Java Application] G:\Eclipse Setup\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.5.v2022
0----2----ab
2----4----ab
6----8----ab
```

Character Classes:

CHAPTER-11 **REGULAR EXPRESSIONS** PART-**4 /** SLIDE

Character Classes:

[abc]	Either a OR b OR c
[^abc]	Except a, b and c
[a-z]	Any Lower Case Alphabet Symbol
[A-Z]	Any Upper Case Alphabet Symbol
[a-z A-Z]	Any Alphabet Symbol
[0-9]	Any Digit from 0 to 9
[a-z A-Z 0-9]	Any Alpha Numeric Symbol
[^a-z A-Z 0-9]	Except Alpha Numeric Symbol (Special Characters)

Pre - Defined Character Classes:

\s	Space Character
\S	Any Character Except Space
\d	Any Digit from [0-9]
\D	Any Character Except Digit
\w	Any Word Character [Any Alpha Numeric Character] [a-Za-z0-9]
\W	Except Word Character (Special Character)
.	Any Symbol including Special Character Also

```java

import java.util.regex.*;

public class demo {
    public static void main(String args[])
    {
        Pattern p = Pattern.compile("\\s");
        String s[] = p.split("Chandu is my FRD");
        for(String ss : s)
            System.out.println(ss);

    }
}

```

```java
import java.util.regex.*;

public class demo {
    public static void main(String args[])
    {
        Pattern p = Pattern.compile("[.]");
        String s[] = p.split("www.chandu.com");
        for(String ss : s)
            System.out.println(ss);

    }
}
```

StringTokenizer:

- **Specially designed class for tokenization activities.**
- **Java.util package**

StringTokenizer st = StringTokenizer("Chandu Soft ware");

while(st.hasmoreTokens())

{

S.o.p(st.nextToken);

}

```java
import java.util.regex.*;
import java.util.*;

public class demo {
    public static void main(String args[])
    {
        StringTokenizer st = new StringTokenizer("Chandu Soft Ware");
        while(st.hasMoreTokens())
        {
            System.out.println(st.nextToken());
        }

    }
}
```

Console × Problems Debug Shell

```
<terminated> demo [Java Application] G:\Eclipse Setup\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_6
Chandu
Soft
Ware
```

Validate the Mobile Number By REGEX.

```java
import java.util.regex.*;

public class demo {
    public static void main(String args[])
    {
        Pattern p= Pattern.compile("[6-9][0-9]{9}");
        String MyMobile = new String("9950264821");
        Matcher m = p.matcher(MyMobile);

        if(m.find() && m.group().equals(MyMobile))
        {
            System.out.println("Valid mobile Number");

        }
        else {
            System.out.println("Invalid");
        }

    }
}
```

Console × Problems Debug Shell

```
<terminated> demo [Java Application] G:\Eclipse Setup\eclipse\plugins\org.eclipse.justj.openjdk.hot
Valid mobile Number
```

```java
import java.util.regex.*;

public class demo {
    public static void main(String args[])
    {
//      email regex
//      Pattern p = Pattern.compile("[a-zA-Z0-9]+@gmail[[.][a-z]+]+");
//      Matcher m = p.matcher("7amireddychandu77@gmail.co.in");

//      Mobile Number
        Pattern p = Pattern.compile("(0|91)?[6-9][0-9]{9}");
        Matcher m = p.matcher("919959574853");

        if(m.find())
        {
            System.out.println("Number Verified "+m.group());
        }
        else {
            System.out.println("inValid");
        }

    }
}
```

```java
import java.util.regex.*;
import java.io.*;

public class demo {
    public static void main(String args[]) throws Exception
    {
        PrintWriter out = new PrintWriter("mobile.txt");
        Pattern p = Pattern.compile("[6-9][0-9]{9}");
        BufferedReader bf = new BufferedReader(new FileReader("data.txt"));
        String line = bf.readLine();
        while(line != null)
        {
            Matcher m = p.matcher(line);
            while(m.find()) {
                out.println(m.group());
            }
            line = bf.readLine();
        }

        out.flush();
    }

}
```

```java
    int count = 0;
    Pattern p = Pattern.compile("[a-zA-Z0-9][a-zA-Z0-9_$.]*[.]txt");
    File f = new File("C:\\durga_classes");
    String[] s = f.list();
    for(String s1 : s)
    {
        Matcher m = p.matcher(s1);
        if(m.find() && m.group().equals(s1))
        {
            count++;
            System.out.println(s1);
        }
    }
    System.out.println(count);
  }
}
```

DURGASOFT
www.durgasoft.com