

Lecture 22

Validating user input

Mr. Mubashir Ali

Lecturer (Dept. of Computer Science)

dr.mubashirali1@gmail.com

Summary of the previous lecture

- **Super Global variables**
- **Passing form data**
- **Passing data with sessions**

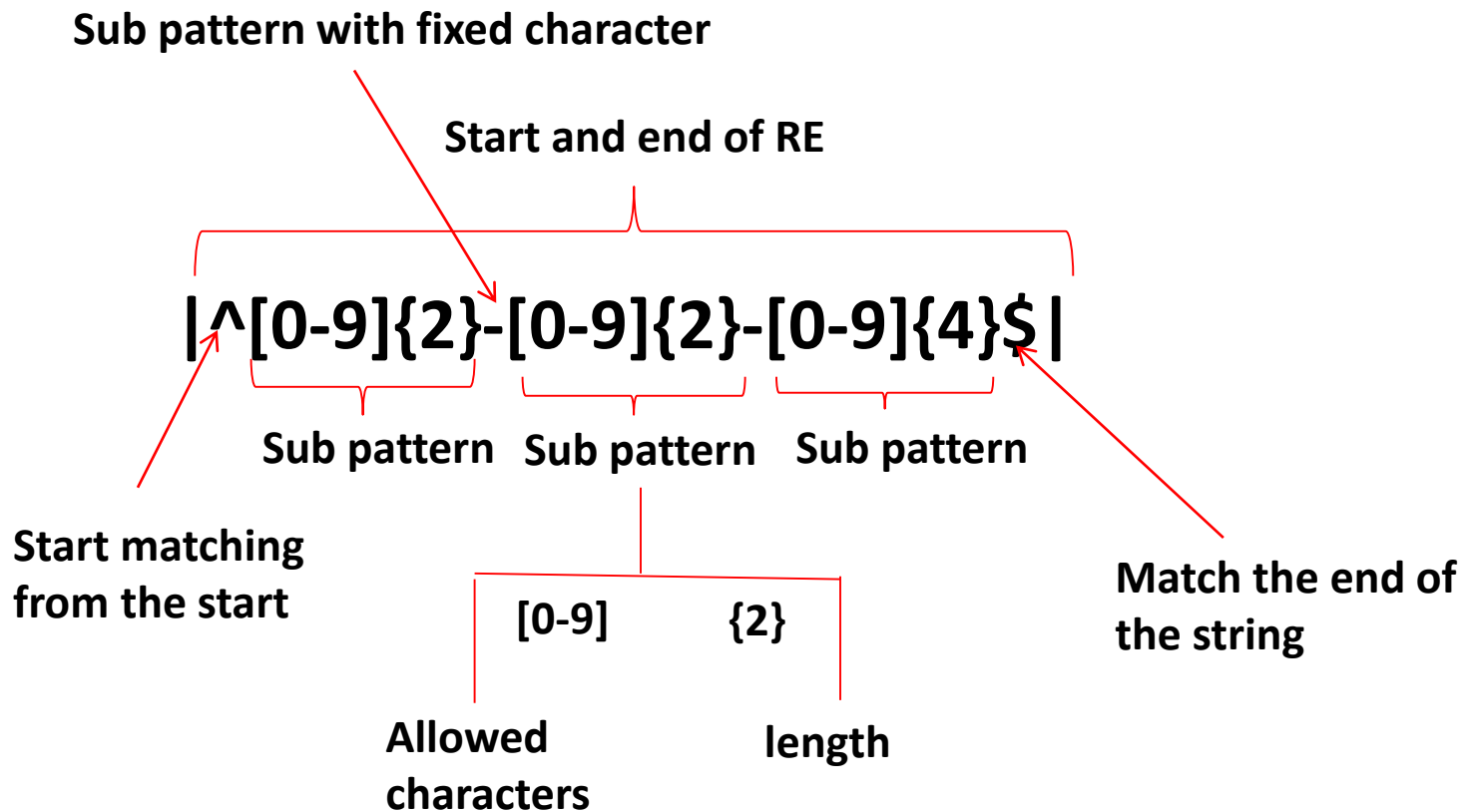
Outline

- **Regular expressions in PHP**
- **Validating user input at server**
- **String functions**

1. Regular expressions in PHP

- A regular expression is a **concise notation** to describe patterns in strings
- Regular expressions provide the foundation for **describing or matching** data according to defined **syntax rules**
 - Example: `|^[0-9]{2}-[0-9]{2}-[0-9]{4}$|`

1. Regular expressions in PHP...



1. Regular expressions in PHP...

- **Start and end of the RE:**
 - optional, | |
- **Sub-patterns:**
 - **range** of allowed characters
 - Allowed **length**
- **Sub-patterns with fixed character:**

1. Regular expressions in PHP...

- Matching from the **start**:

– 1212-12-2014

Pattern exists if do not
match from start

- Matching till end:

– 12-12-2014123

Pattern exists if do
not match till end

- For exact match we should use both **^** and **\$**

1.1 Notations for RE

- **^**: match strings that start with the given pattern
- **\$**: match strings that end with the given pattern
- **-**: means a range of characters
- **[]**: makes a class of characters
- **[^]**: negates the class of character

1.1 Notation for RE...

- **Quantifiers:**
- **{n}**: matches a character, class or sub-pattern for **n** times
- **{ n, m }**: matches a character, class or sub-pattern for minimum **n** times and maximum **m** times

1.1 Notation for RE...

- **?:** matches the character, class or sub-pattern **0 or 1** time
 - equal to $\{0,1\}$
- **+:** matches the character, class or sub-pattern **1 or more** times
 - equals to $\{1, \}$
- ***:** matches the character, class or sub-pattern **0 or any number of time**
 - equals $\{0, \}$

1.1 Notation for RE...

Predefined character ranges:

- **\d**: means exactly as [0-9]
- **\D**: means exactly as [^0-9]
- **\w**: means exactly as [a-zA-Z0-9_]

1.1 Notation for RE...

RE examples:

- Validating date:

– `|^\d{2}-\d{2}-\d{4}$|`

- Validating CNIC:

– `|^\d{5}-\d{7}-\d{1}$|`

- Validating Email:

– `|^[a-zA-Z0-9_\.]+\@[a-z]{3,5}\.[a-z]{2,3}$|`

1.1 Notation for RE...

- **Validating name:**
 - $|^{\wedge}[a-zA-Z]\{5,25\}\$ |$
- **Validating Password:**
 - must contain '@'
 - $|@|$

2. Validating user's input

- **preg_match():**
 - searches a **string** for a **specific pattern**
 - returns **TRUE** if it exists and **FALSE** otherwise
 - **preg_match("pattern",\$string);**

2. Validating user's input

User <u>Registration</u> Form		
Full Name	<input type="text"/>	Post, action.php
E-mail	<input type="text"/>	name
CNIC	<input type="text"/>	email
Date of Birth	<input type="text"/>	cnic
<input type="submit" value="Submit"/>		dob

2. Validating user's input

```
1 <?php
2 $name     = $_POST['name'];
3 $email    = $_POST['email'];
4 $cnic     = $_POST['cnic'];
5 $dob      = $_POST['dob'];
6
7 if(!preg_match("[a-zA-Z ]{3,25}$", $name))
8 echo "Invalid input for name";
```

Receiving values

Validating name

2. Validating user's input

```
9 echo "<br>";
10 if(!preg_match(
11   "|^[a-zA-Z0-9_]+@[a-z]{3,5}\.[a-z]{3}$|", $email)) } email
12 echo "<br>";
13 if(!preg_match("|^\d{5}-\d{7}-\d{1}$|", $cnic)) } CNIC
14 echo "Invalid CNIC";
15 echo "<br>";
16 if(!preg_match("|^\d{2}-\d{2}-\d{4}$|", $dob)) } DoB
17 echo "Invalid Date of Birth";
18 ?>
```

3. String functions in PHP

- **strlen():**
 - Returns the length of the string
 - `strlen($string);`
- **strcmp():**
 - Compares two strings
 - Returns 0 if strings are equal, 1 if first string is greater and -1 if second is greater
 - `strcmp($string1,$string2);`
- **Strcasecmp():**
 - Compares two strings in case insensitive manner
 - `strcasecmp($string1,$string2);`

3. String functions in PHP...

Change Password		Method=post
Name	<input type="text"/>	name
Type new password	<input type="text"/>	pass
Re-type new password	<input type="text"/>	pass1
		<input type="submit" value="Submit"/>

3. String functions in PHP...

```
1 <?php
2 $name = $_POST['name'];
3 $pass = $_POST['pass'];
4 $pass1 = $_POST['pass1'];
5 if(strlen($pass)<6)
6 echo "Too short password";
```

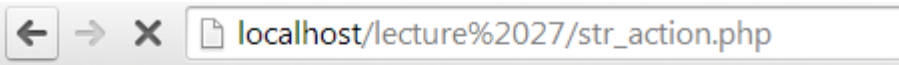
Getting variables

Using strlen()

3. String functions in PHP...

Change Password	
Name	<input type="text" value="Basharat"/>
Type new password	<input type="text" value="123"/>
Re-type new password	<input type="text" value="123"/>
<input type="button" value="Submit"/>	

Password is short



Too short password

3. String functions in PHP...

```
7 if (strcmp($pass, $pass1) <> 0)
8 echo "Password mismatch";
```

Compares pass and pass1

Change Password	
Name	<input type="text" value="Basharat"/>
Type new password	<input type="text" value="1233456"/>
Re-type new password	<input type="text" value="1232222"/>
<input type="button" value="Submit"/>	

localhost/lecture%2027/str_action.php

Password mismatch

3. String functions in PHP...

- **strtolower():**
 - Convert a string in lower case
 - `strtolower($string);`
- **strtoupper():**
 - Convert a string in upper case
 - `strtoupper($string);`
- **ucfirst():**
 - Convert the first character of a string to upper case
 - `ucfirst($string);`
- **ucwords():**
 - Convert the first character of each word in a string to upper case
 - `ucfirst($string);`

3. String functions in PHP...

```
9 echo strtolower($name) . "<br>";  
10 echo strtoupper($name) . "<br>";  
11 echo ucfirst($name) . "<br>";  
12 echo ucwords($name) . "<br>";
```

Converts name to
lowercase

Converts name
to uppercase

Using ucfirst()

Using ucwords()

3. String functions in PHP...

Change Password	
Name	<input type="text" value="Basharat mahmood"/>
Type new password	<input type="text" value="1233456"/>
Re-type new password	<input type="text" value="1233456"/>
<input type="button" value="Submit"/>	

localhost/lecture%2027/str_action.php

basharat mahmood ← **Lowercase**
BASHARAT MAHMOOD ← **uppercase**
Basharat mahmood ← **ucfirst()**
Basharat Mahmood ← **ucwords()**

3. String functions in PHP...

- **strpos():**
 - finds the position of the first case-sensitive occurrence of a substring in a string
 - `strpos($string,sub-string);`
- **strrpos():**
 - finds the position of the last case-sensitive occurrence of a substring in a string
 - `strrpos($string,sub-string);`
- **substr_count():**
 - returns the number of times one string occurs within another
 - `substr_count($string,sub-string);`

3. String functions in PHP...

Finding first occurrence of 'a'

```
13 echo strpos($name, 'a') . "<br>";  
14 echo strrpos($name, 'a') . "<br>";  
15 echo substr_count($name, 'a') . "<br>";  
16 ?>
```

Last occurrence of 'a'

Finding number of occurrences
of 'a'

3. String functions in PHP...

Change Password	
Name	<input type="text" value="Basharat mahmood"/>
Type new password	<input type="text" value="1233456"/>
Re-type new password	<input type="text" value="1233456"/>
<input type="button" value="Submit"/>	

localhost/lecture%2027/str_action.php

basharat mahmood

BASHARAT MAHMOOD

Basharat mahmood

Basharat Mahmood **First occurrence of 'a'**

1 **Last occurrence of 'a'**

10

4 **Number of occurrences of 'a'**

Summary

- **Writing regular expression in PHP**
- **Validating user's input**
- **String functions**

References

- **Chapter 9, “Beginning PHP and MySQL”** by W. Jason Gilmore, Apress publisher, 4th edition; 2010, ISBN-13 (electronic): 978-1-4302-3115-8.