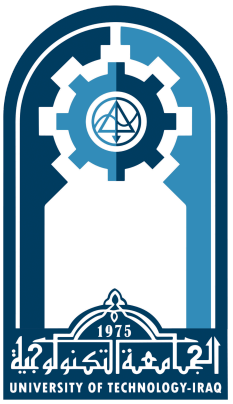

WEB LAB

Name : Zumuruda ahmed mohammed

Fourth stage - network - morning

Discussion

1- What is a regular expression and how is it used in PHP?

A regular expression, often abbreviated as regex or regexp, is a sequence of characters that define a search pattern. They are used for pattern matching within strings. Regular expressions are incredibly powerful tools for string manipulation, search, and validation tasks.

Here's a basic rundown of how regular expressions are used in PHP:

1-Creating a regular expression pattern: You define a pattern using specific characters and syntax that represent the sequence of characters you want to match. For example, `/pattern/` is a basic regular expression pattern.

2-Matching patterns: You can use functions like `preg_match()`, `preg_match_all()`, or `preg_replace()` to match patterns within strings

3-Modifiers: You can add modifiers to regular expression patterns to change their behavior. For example, adding the "i" modifier makes the pattern case-insensitive.

4-Character classes and quantifiers: Regular expressions support various special characters and symbols for defining character classes (e.g., `[a-zA-Z]` matches any letter), quantifiers (e.g., `+` matches one or more occurrences), and more.

5-Capturing groups: Parentheses `()` are used to create capturing groups within regular expressions. These groups allow you to extract specific parts of a matched string.

2- What are some of the most commonly used regular expression functions in PHP, and what do they do?

preg_match(): This function is used to perform a regular expression match against a string. It returns true if the pattern is found in the string, and false otherwise.

Example:

```
$string = "Hello, world!";
$pattern = "/world/";
if (preg_match($pattern, $string)) {
    echo "Pattern found!";
} else {
    echo "Pattern not found!";
}
```

preg_match_all(): Similar to preg_match(), but it returns the number of matches found and stores them in an array.

Example:

```
$string = "Hello, world! Hello, universe!";
$pattern = "/Hello/";
if (preg_match_all($pattern, $string, $matches)) {
    echo "Found " . count($matches[0]) . " matches!";
} else {
    echo "No matches found!";
}
```

preg_replace(): This function is used to perform a search and replace using a regular expression pattern. It replaces all occurrences of the pattern with a specified replacement string.

Example:

```
$string = "The quick brown fox jumps over the lazy dog.";
$pattern = "/fox/";
$replacement = "cat";
$new_string = preg_replace($pattern, $replacement, $string);
echo $new_string;
Output: "The quick brown cat jumps over the lazy dog."
```

preg_split(): This function splits a string into an array of substrings using a regular expression pattern as the delimiter.

Example:

```
$string = "apple,orange,banana";
$pattern = "/,/";
$array = preg_split($pattern, $string);
print_r($array);
// Output: Array ( [0] => apple [1] => orange [2] => banana )
```

3- What is the difference between the preg_match() and preg_match_all() functions?

preg_match():

- This function is used to search for a single match of a pattern in a string.
- It stops searching after the first match is found.
- It returns `true` **if the pattern is found at least once in the string, and** `false` **otherwise.**
- **If matches are found, the function returns** `1`**, indicating that a match was found.**

preg_match_all():

- This function is used to search for all matches of a pattern in a string.
- It continues searching for matches throughout the entire string.
- It returns the number of matches found.
- It also stores the matches in an array specified by the third parameter.

4- How do you use regular expressions to search and replace text in PHP? Give a code example.

In PHP, you can use the **`preg_replace()`** function to search and replace text using regular expressions.

lec1.php | function.php

function.php

```php
<!DOCTYPE html>
<html lang="en">
<head>

    <title>Document</title>
</head>
<body>
    <?php
    $string = "The quick brown fox jumps over the lazy dog.";
    $pattern = "/fox/";
    $replacement = "cat";
    $new_string = preg_replace($pattern, $replacement, $string);
    echo $new_string; // Output: "The quick brown cat jumps over the lazy dog."
    ?>
</body>
</html>
```

```
Output: "The quick brown cat jumps over the lazy dog."
```