Python String Manipulation: A Beginner's Guide

Before diving into regular expressions, it's important to have a good understanding of basic string manipulation in Python. Here's a brief overview:

Lesson 1: String Basics

In Python, a string is a sequence of characters. You can create a string by enclosing characters in single quotes `'` or double quotes `"`.

```
s = "Hello, world!"
```

Lesson 2: Accessing Characters

You can access individual characters in a string using indexing. Python uses zero-based indexing, so the first character is at index 0, the second character is at index 1, and so on.

```
s = "Hello, world!"
print(s[0])  # Outputs: H
```

Lesson 3: Slicing Strings

You can access a range of characters in a string by using slicing. The syntax for slicing is `s[start:stop]`, where `start` is the index to start at and `stop` is the index to stop at (but not include).

```
s = "Hello, world!"
print(s[0:5])  # Outputs: Hello
```

Lesson 4: String Length

You can get the length of a string (the number of characters) using the `len()` function.

```
s = "Hello, world!"
print(len(s))  # Outputs: 13
```

Lesson 5: String Methods

Python provides a number of useful methods for manipulating strings. Here are a few examples:

- `s.lower()`: Returns a copy of the string with all characters converted to lowercase.
- `s.upper()`: Returns a copy of the string with all characters converted to uppercase.
- `s.replace(old, new)`: Returns a copy of the string with all occurrences of the substring `old` replaced by `new`.
- `s.split(sep)`: Returns a list of substrings separated by the string `sep`.

```
s = "Hello, world!"
print(s.lower())  # Outputs: hello, world!
print(s.upper())  # Outputs: HELLO, WORLD!
print(s.replace("world", "Python"))  # Outputs: Hello, Python!
print(s.split(","))  # Outputs: ['Hello', ' world!']
```

With this basic understanding of string manipulation in Python, you should be well-prepared to start learning about regular expressions.