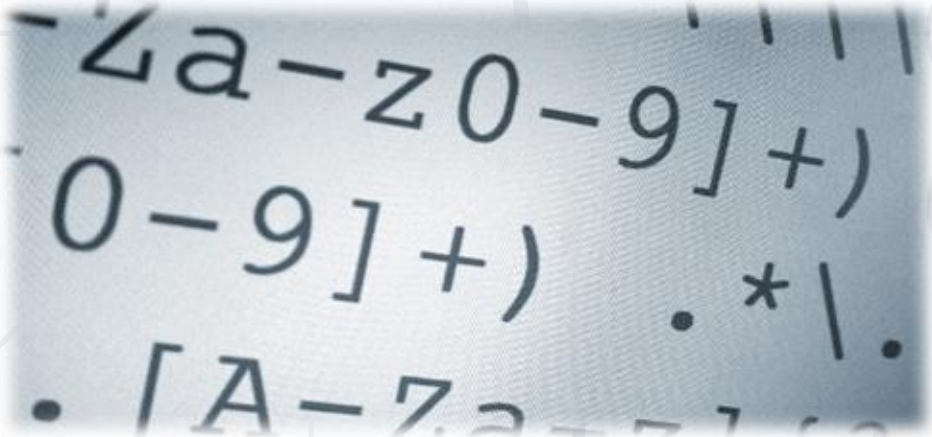
Regular Expressions (RegEx)

Regular Expressions Language Syntax



SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

Table of Contents

Have a Question?

Software University

sli.do

#fund-csharp

Definition and Classes

Regular Expressions

What Are Regular Expressions?

- Regular expressions (regex)
  - Match text by pattern
- Patterns are defined by special syntax, e.g.
  - `[0-9]+` matches non-empty sequence of digits
  - `[A-Z][a-z]*` matches a capital + small letters
- Play with regex live at: regexr.com, regex101.com

Regular Expression Pattern – Example

- Regular expressions (regex) describe a search pattern
- Used to find / extract / replace / split data from text by pattern

```
[A-Z][a-z]+ [A-Z][a-z]+
```

```
John Smith
```

```
Linda Davis
```

```
Contact: Alex Scott
```

Character Classes: Ranges

- `[nvj]` matches any character that is either **n**, **v** or **j**

```
node.js v0.12.2
```

- `[^abc]` – matches any character that is **not** **a**, **b** or **c**

```
Abraham
```

- `[0-9]` – character range: matches any digit from **0** to **9**

```
John is 8 years old.
```

Predefined Classes

- \w - matches any **word character** (a-z, A-Z, 0-9, _)
- \W - matches any **non-word character** (the opposite of \w)
- \s - matches any **white-space** character
- \S - matches any **non-white-space** character (the opposite of \s)
- \d - matches any **decimal digit** (0-9)
- \D - matches any **non-decimal character** (the opposite of \d)

Grouping

Quantifiers

Quantifiers

- * - matches the previous element zero or more times

```
\+\d*   +359885976002 a+b
```

- + - matches the previous element one or more times

```
\+\d+   +359885976002 a+b
```

- ? - matches the previous element zero or one time

```
\+\d?   +359885976002 a+b
```

- {3} - matches the previous element exactly 3 times

```
\+\d{3}   +359885976002 a+b
```

Grouping Constructs

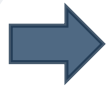
- `(subexpression)` - captures the matched subexpression as numbered group

```
\d{2}-(\w{3})-\d{4}
```

```
22-Jan-2015
```

- `(?:subexpression)` - defines a non-capturing group

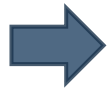
```
^(?:Hi|hello),\s*(\w+)$
```



```
Hi, Peter
```

- `(?<name>subexpression)` - defines a named capturing group

```
(?<day>\d{2})-(?<month>\w{3})-
(?<year>\d{4})
```



```
22-Jan-2015
```

Problem: Match All Words

- Write a regular expression in www.regex101.com that extracts all word char sequences from given text

```
_ (Underscores) are also word characters!
```

```
_|Underscores|are|also|word|characters
```

Problem: Match Dates

- Write a regular expression that extracts **dates** from text
  - Valid date format: **dd-MMM-yyyy**
  - Examples: **12-Jun-1999**, **3-Nov-1999**

```
I am born on 30-Dec-1994.
My father is born on the 9-Jul-1955.
01-July-2000 is not a valid date.
```

Problem: Email Validation

- Write a regular expression that performs simple **email validation**
  - An email consists of: **username @ domain name**
  - **Usernames** are **alphanumeric**
  - **Domain names** consist of **two strings**, separated by a **period**
  - **Domain names** may contain only **English letters**

Valid: `valid123@email.bg`

Invalid: `invalid*name@emai1.bg`

Numbered Capturing Group

Backreferences

Backreferences Match Previous Groups

- \number - matches the value of a numbered capture group

```
<(\w+)[^>]*>.*?<\/\1>
```

```
<b>Regular Expressions</b> are cool!
<p>I am a paragraph</p> … some text after
Hello, <div>I am a<code>DIV</code></div>!
<span>Hello, I am Span</span>
<a href="https://softuni.bg/">SoftUni</a>
```

Regex in C#

- C# supports a built-in regular expression class: **Regex**
  - Located in **System.Text.RegularExpressions** namespace

```
using System.Text.RegularExpressions;

static void Main()
{
   string pattern = @"A\w+";
   Regex regex = new Regex(pattern);
}
```

Validating String by Pattern

- `IsMatch(string text)`
  - Determines whether the text matches a given pattern

```
string text = "Today is 2015-05-11";
string pattern = @"\d{4}-\d{2}-\d{2}";

Regex regex = new Regex(pattern);
bool containsValidDate = regex.IsMatch(text);

Console.WriteLine(containsValidDate); // True
```

Checking for a Single Match

- `Match(string text)`
  - Returns the first match of a given pattern

```
string text = "Nakov: 123";
string pattern = @"([A-Z][a-z]+): (\d+)";
Regex regex = new Regex(pattern);
Match match = regex.Match(text);

Console.WriteLine(match.Groups.Count); // 3
Console.WriteLine("Matched text: \"{0}\"", match.Groups[0]);
Console.WriteLine("Name: {0}", match.Groups[1]); // Nakov
Console.WriteLine("Number: {0}", match.Groups[2]); // 123
```

Checking for Matches

- `Matches(string text)` - returns a collection of matches

```
string text = "Nakov: 123, Branson: 456";
string pattern = @"([A-Z][a-z]+): (\d+)";

Regex regex = new Regex(pattern);

MatchCollection matches = regex.Matches(text);
Console.WriteLine("Found {0} matches", matches.Count);

foreach (Match match in matches)
  Console.WriteLine("Name: {0}", match.Groups[1]);

// Found 2 matches
// Name: Nakov
// Name: Branson
```

Replacing with Regex

- `Replace(string text, string replacement)` – replaces all strings that match the pattern with the provided replacement

```
string text = "Nakov: 123, Branson: 456";
string pattern = @"\d{3}";
string replacement = "999";

Regex regex = new Regex(pattern);
string result = regex.Replace(text, replacement);

Console.WriteLine(result);
// Nakov: 999, Branson: 999
```

Splitting with Regex

- `Split(string text)` - splits the text by the pattern
  - Returns `string[]`

```
string text = "1   2 3      4";
string pattern = @"\s+";

string[] results = Regex.Split(text, pattern);
Console.WriteLine(string.Join(", ", results));
// 1, 2, 3, 4
```

Problem: Match Full Name

- You are given a list of names
  - Match all full names

```
Bethany Taylor, Oliver miller, sophia Johnson, SARah
Wilson, John Smith, Sam        Smith
```

```
Bethany Taylor John Smith
```

Check your solution here: https://judge.softuni.org/Contests/Practice/Index/1667#0

Solution: Match Full Names

```
string listOfNames = Console.ReadLine();

string pattern @"\b[A-Z][a-z]+ [A-Z][a-z]+";
Regex regex = new Regex(pattern);
MatchCollection validNames = regex.Matches(input);
foreach (Match name in validNames)
{
    Console.Write($"{name.Value} ");
}

Console.WriteLine();
```

Check your solution here: https://judge.softuni.org/Contests/Practice/Index/1667#0

Problem: Match Dates

- You are given a string
  - Match all dates in the format "**dd{separator}MMM{separator}yyyy**" and print them space-separated

```
13/Jul/1928, 01/Jan-1951
```

```
Day: 13, Month: Jul, Year: 1928
```

Check your solution here: https://judge.softuni.org/Contests/Practice/Index/1667#2

Solution: Match Dates

```
string input = Console.ReadLine();

string pattern = @"\b(?<day>\d{2})(\.|-|\/)
(?<month>[A-Z][a-z]{2})\1(?<year>\d{4})\b";

MatchCollection matches = Regex.Matches(input, pattern);

foreach (Match date in matches)
    Console.WriteLine($"Day: {date.Groups["day"].Value},
    Month: {date.Groups["month"].Value}, Year:
    {date.Groups["year"].Value}");
```

Check your solution here: https://judge.softuni.org/Contests/Practice/Index/1667#2

Summary

- **Regular expressions** describe **patterns** for searching through text
- Define **special characters**, **operators** and **constructs** for building complex pattern
- Can utilize **character classes**, **groups**, **quantifiers** and more

Questions?

SoftUni

Software University

SoftUni Svetlina

SoftUni Creative

SoftUni Digital

SoftUni Foundation

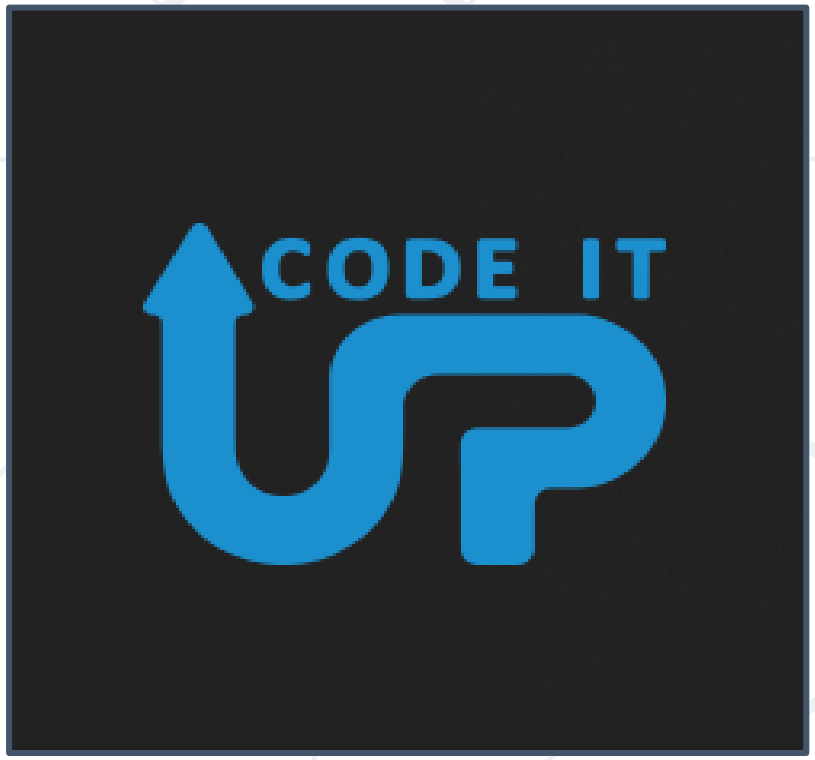
SoftUni Kids

SoftUni Diamond Partners

Software University

Educational Partners

Software University



Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://about.softuni.bg/
- © Software University – https://softuni.bg