Regular Expressions

Web Programming

Uta Priss ZELL, Ostfalia University

2013

Web Programming Regular Expressions Slide 1/17

Outline

Basics

Basics

Wildcard and multipliers

Special characters

Negation

Other functions

Programming

Web Programming Regular Expressions Slide 2/17

NO match

Other functions

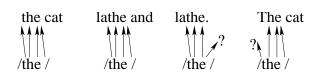
The cat

/the / i

Wildcard and multipliers

Basics

Character by character match



Note: "i" at the end means "ignore case"

NO match

Other functions

Programming

Special characters

. stands for "any character".

Wildcard and multipliers

Basics

Wildcard and multipliers

. stands for "any character".

Multipliers:

Basics

- + stands for "at least one character"
- * stands for "any number of characters (including 0)"
- ? stands for "at most one character" (i.e. either none or once)
- {n,m} stands for "at least n times, at most m times"

Web Programming Regular Expressions Slide 4/17

00

Other functions

000

Programming

000

Special characters

the

Basics

Wildcard and multipliers

000

Examples:

/t.{1,2}e//t.{1,2}e/

Exercise

What does /..\.19../ match: "12.1000" or "123.1900" or "12.2000"

Web Programming Regular Expressions Slide 6/17

Programming

Exercise

Basics

What does /..\.19../ match: "12.1000" or "123.1900" or "12.2000"

What does /hn*ell?o W...d/i match:
"Hello World" or "Hello Wood" or "Hell?o World"?

Web Programming Regular Expressions Slide 6/17

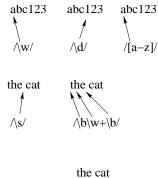
Special characters

Basics

```
word character (letter, digit or _)
\w
[a-zA-Z]
             letter
\W
             non-word character
[^a-zA-Z]
             not a letter
\d
             digit
\s
             space character (blank space, tab)
\backslash \mathsf{b}
             word boundary
             beginning of line or string
             end of line or string
```

Web Programming Regular Expressions Slide 7/17

abc123



the cat

\$a="the"

Basics

Which matches two consecutive words:

Basics

\$word $!\sim/a/$ means that "a" must not occur in \$word at all.

 $\$ word $= \sim /[\hat{a}]/$ means that $\$ word must have one character which is not "a".

Web Programming Regular Expressions Slide 10/17

Programming

000

Examples

Basics

the the
$$\uparrow$$
 !~ /t/ =~ /[^t]/ No match

gst0202 gst0202;

$$\Rightarrow /[^\w]/$$
 =~ /[^\w]/
No match match

Web Programming Regular Expressions Slide 11/17

Substitution

Basics

the table
$$\longrightarrow$$
 s/t/T/ \longrightarrow The table

the table
$$\longrightarrow$$
 s/t/T/g \longrightarrow The Table

the table
$$\Rightarrow$$
 s/<\/?p>//g \Rightarrow the table
 \Rightarrow s/<.*>//g

the table
$$\longrightarrow s/<.*?>//g \longrightarrow$$
 the table

Web Programming Regular Expressions Slide 12/17

Other functions

Remembering patterns

Wildcard and multipliers

Brackets are used for remembering patterns. The content of the first set of brackets can be retrieved with $\setminus 1$. The second set of brackets with $\setminus 2$, and so on.

Examples:

$$s/(the table)/\1//(.)\1/s/(.)(\1/2\1/$$

Web Programming Regular Expressions Slide 13/17

Other functions

Split and Join (Implode)

Wildcard and multipliers

```
$oldstring = "the,cat,sat,on,the,mat";
@array = split(/,/,$oldstring);
print @array;
# @array = ("the", "cat", "sat", "on", "the", "mat")
$newstring = join(" ",@array);
# $newstring ="the cat sat on the mat"
```

Web Programming Regular Expressions Slide 14/17

Wildcard and multipliers

Basics

Strategies

Instead of using one complicated regular expression, it is sometimes easier to use several simpler regular expressions combined with if statements.

For example: string starts with "a" and ends with "z":

```
if (\$string =\sim /^a.*z$/)
if (\$string =\sim /^a/ and \$string =\sim /z\$/)
```

Web Programming Regular Expressions Slide 15/17

More Strategies

Wildcard and multipliers

If a string needs to be processed ...

- ▶ from left to right, one character or one word at a time ⇒ split into array, then process array.
- from left to right, in some other regular manner ⇒ substr() can be used instead of regular expression.
- ▶ by checking whether some pattern exists ⇒ use regular expressions.

Web Programming Regular Expressions Slide 16/17

Use of regular expressions in PHP

Wildcard and multipliers

```
Searching:
    if (preg_match("/the /i", $line, $matches)) {
    echo $line,"<br> matches: ",$matches[0],"<br>";}
Replace:
    $line = preg_replace("/T/", 't', $line);
Split:
    $words = preg_split("/\s+/", $line);
Implode:
    $newstring = implode(" ", $array);
```