Espressioni regolari

Prof. Andrea Bonardi

Espressioni regolari

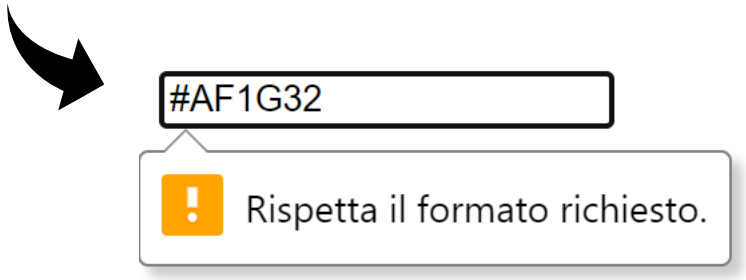
- Un'**espressione regolare**, nota anche come *regex* o *regexp*, è una sequenza di caratteri che identifica un insieme di stringhe.
- Le espressioni regolari consentono di:
  - Ricercare una stringa all'interno di un testo
  - Convalidare un input
  - Sostituire del testo

99

Espressioni regolari

- Convalidare un input

```
<form action="http://andreabonardi.altervista.org/ricevidati.php" method="post">
    <input type="text" id="color" name="color"  pattern="#[0-9A-F]{6}">
    <br><br>
    <input type="submit" value="Invia">
</form>
```



- NB. Se l'utente non compila il campo, la validazione con *pattern* viene saltata. Se necessario, aggiungere *required*.

Espressioni regolari

- **Classi di caratteri**
  - **.** un qualsiasi carattere (tranne newline)
    - \. per indicare il carattere punto
  - **\d** una cifra numerica
  - **\w** un carattere alfanumerico o underscore
  - **\s** un carattere di spaziatura (spazio, tab, newline)
  - **[ ]** un insieme di caratteri validi
    - [aeiou] corrisponde a una qualsiasi vocale
    - [a-z] corrisponde a qualsiasi lettera minuscola
    - [^aeiou] corrisponde a un qualsiasi carattere che non sia una vocale

Espressioni regolari

- **Quantificatori**
  - **?** 0 o 1 occorrenza
  - ***** 0 o più occorrenze
  - **+** 1 o più occorrenze
  - **{ }** per specificare il numero esatto di occorrenze
    - \d{3} esattamente 3 cifre
    - \d{3,5} da 3 a 5 cifre
    - \d{3,} almeno 3 cifre

Espressioni regolari

- **Gruppi**
  - ( ) raggruppa una porzione di espressione regolare
  - | alternativa tra due espressioni
    - (Milano|Roma)

Espressioni regolari: esercizi

- Un bit
- Una sequenza di 8 bit
- Una cifra esadecimale
- Un colore in formato RGB
- Il sesso di una persona (m o f)
- Una data (GG/MM/AAAA)

- Un numero con la virgola
  - Esempi: -3.454, 10.8, +3.0

Espressioni regolari: esercizi

- Un bit
  - [01]
- Una sequenza di 8 bit
  - [01]{8}
- Una cifra esadecimale
  - [0-9A-F]
- Un colore in formato RGB
  - #[0-9A-F]{6}
- Il sesso di una persona (m o f)
  - [mMfF]
- Una data (GG/MM/AAAA)
  - ((0[1-9])|([12]\d)|(3[01]))\/((0[1-9])|(1[012]))\/\d{4}
- Un numero con la virgola
  - Esempi: -3.454, 10.8, +3.0
  - [\-\+]?\d+\.\d+

Devo utilizzare il carattere di escape \, in quanto -, + e . fanno parte della sintassi delle regex

Espressioni regolari: esercizi

- Il nome dei file da consegnare al prof. Zola

ATTENZIONE: Ogni file dovrà essere nominato secondo lo standard: **Cog_Nom_Cl_Argomento.***

Cog: le prime tre lettere del cognome
Nom: le prime tre lettere del nome
Cl: la classe e sezione di appartenenza
Argomento: una o più parole che identifichi il file

La non consegna verrà considerata come un rifiuto (voto: 2)

Es:

- Zol_Fra_3AI_ver1.pdf
- Zol_Fra_3AI_ver2.pdf
- ...
- ...
- Zol_Fra_3AI_codici.zip

Espressioni regolari: esercizi

- Il nome dei file da consegnare al prof. Zola
  - [A-Z][a-z]{2}_[A-Z][a-z]{2}_[1-5][A-C][IT]_[A-Za-z0-9_]+\.(pdf|zip)

ATTENZIONE: Ogni file dovrà essere nominato secondo lo standard: **Cog_Nom_Cl_Argomento.***

Cog: le prime tre lettere del cognome
Nom: le prime tre lettere del nome
Cl: la classe e sezione di appartenenza
Argomento: una o più parole che identifichi il file

La non consegna verrà considerata come un rifiuto (voto: 2)

Es:

- Zol_Fra_3AI_ver1.pdf
- Zol_Fra_3AI_ver2.pdf
- ...
- ...
- Zol_Fra_3AI_codici.zip

Espressioni regolari: esercizi

- Il codice fiscale

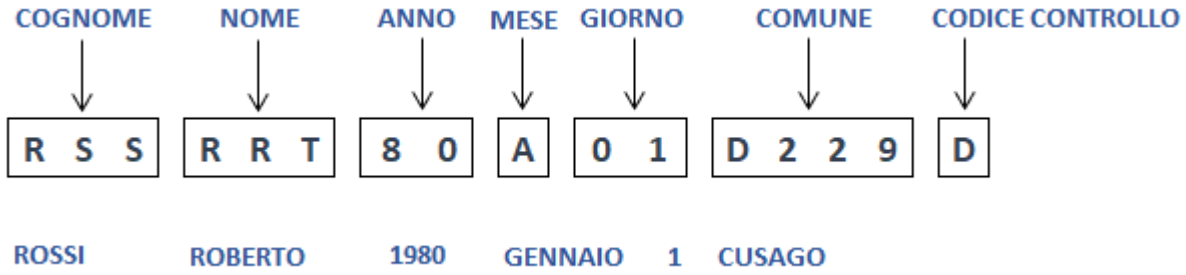


Tabella Conversione del mese di nascita

Gennaio = A	Maggio = E	Settembre = P
Febbraio = B	Giugno = H	Ottobre = R
Marzo = C	Luglio = L	Novembre = S
Aprile = D	Agosto = M	Dicembre = T

Espressioni regolari: esercizi

- Il codice fiscale

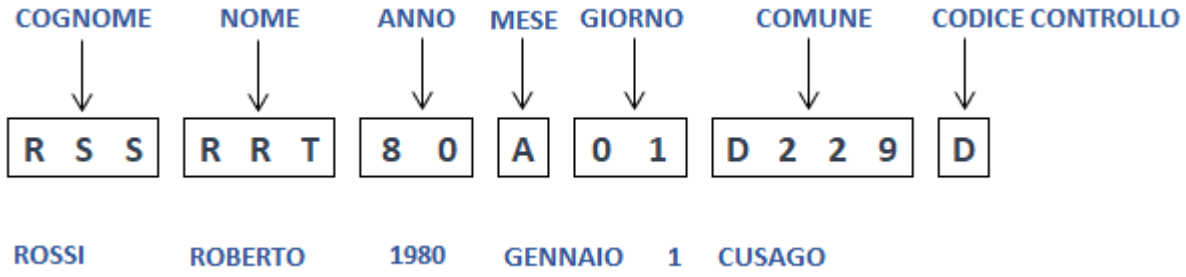


Tabella Conversione del mese di nascita

Gennaio = A	Maggio = E	Settembre = P
Febbraio = B	Giugno = H	Ottobre = R
Marzo = C	Luglio = L	Novembre = S
Aprile = D	Agosto = M	Dicembre = T

- [A-Z]{6}\d{2}[A-EHLMPRST]((0[1-9])|([12]\d)|(3[01]))[A-Z]\d{3}[A-Z]

Espressioni regolari: esercizi

- La carta di credito
  - Un numero di carta di credito è una sequenza di 16 cifre, divise in gruppi di 4, separati da spazi o trattini
  - 1234-5678-9101-1121 o 4321 8765 2109 3210

Espressioni regolari: esercizi

- La carta di credito
  - Un numero di carta di credito è una sequenza di 16 cifre, divise in gruppi di 4, separati da spazi o trattini
  - 1234-5678-9101-1121 o 4321 8765 2109 3210

    - ((\d{4}\-){3}\d{4}) | ((\d{4}){3}\d{4})

Espressioni regolari: esercizi

- Una password complessa
  - Almeno 8 caratteri
  - Almeno una lettera maiuscola
  - Almeno una lettera minuscola
  - Almeno una cifra
  - Almeno un carattere speciale (@, #, $, %, &).

- In una regexp un **lookahead positivo** verifica una condizione senza consumare caratteri.
- Sintassi: (?= ……)
- Es. stringa con almeno 1 cifra: (?=.*\d)

Espressioni regolari: esercizi

- Una password complessa
  - Almeno 8 caratteri
  - Almeno una lettera maiuscola
  - Almeno una lettera minuscola
  - Almeno una cifra
  - Almeno un carattere speciale (@, #, $, %, &).

- In una regexp un **lookahead positivo** verifica una condizione senza consumare caratteri.
- Sintassi: (?= ……)
- Es. stringa con almeno 1 cifra: (?=.*\d)

- (?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[@#$%&]).{8,}

Regexp in Java

- Libreria: *java.util.regex*

```
String regex = "\\d+";
String input = "HelloWorld";

Pattern pattern = Pattern.compile(regex);
Matcher matcher = pattern.matcher(input);

if (matcher.matches()) {
    System.out.println("Input valido");
} else {
    System.out.println("Input non valido");
}
```

Regexp in Java

- Estrapolazione di tutti gli indirizzi email da un testo

```
String text = "Contatta mario.rossi@gmail.com o anna.bianchi@azienda.it per
    info. Anche info@example.com è valido.";

String emailRegex = "\\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,}\\b";

Pattern pattern = Pattern.compile(emailRegex);
Matcher matcher = pattern.matcher(text);

while (matcher.find()) {
    System.out.println(matcher.group());
}
```

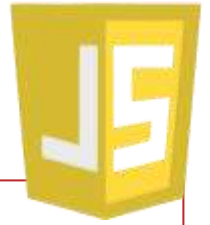
Regexp in Java

- Sostituzione in un testo

```
String testo = "Password: 1234. Pin: 5678";

String risultato = testo.replaceAll("\\d", "*");

System.out.println("Originale: " + testo);

System.out.println("Modificato: " + risultato);
//OUTPUT: Password: ****. Pin: ****
```

Regexp in Javascript

- Le regex in JavaScript sono integrate direttamente nel linguaggio



```
const email = document.getElementById('email').value;
const emailRegex = /^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$/;

if (!emailRegex.test(email)) {
    document.getElementById('msgEmail').innerHTML = 'Email non valida';
}
```