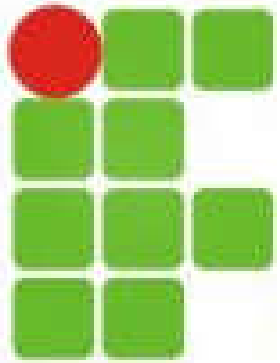
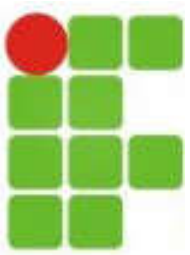
Fundamentos de Web Design 2

Professor Eng. Dr. Will Roger Pereira

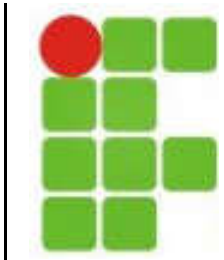


Conteúdo



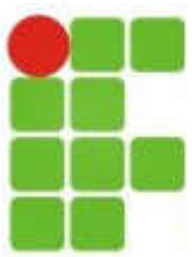
- Expressões regulares.

Introdução



- Expressão regular é, na teoria de linguagens formais, uma sequência de caracteres que define um **padrão de busca**;
- Normalmente utilizado em operações de busca em strings, para verificação de presença ou substituição;
- Ela institui vários conceitos, como metacaracteres, agrupamento, OU booleano e quantificação;
- Utilize com muito cuidado e preste atenção na sua expressão regular, pois um símbolo, e.g. ^, pode ter significados diferentes dependendo do contexto.

Sintaxe em JS



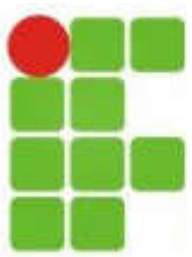
- Em Javascript, uma expressão regular está contida entre barras (/):

/expressão regular/

var exp = **/jovem/**

- A expressão regular exp procura pela string “jovem”.

Operações com Expressões Regulares

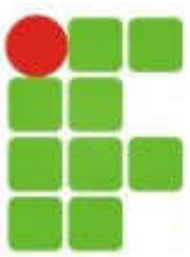


- Para exemplificar, uma expressão regular será chamada de **regexp**, e uma string de **string**;

- Operações:
  - Testar presença;
  - Retornar primeira combinação;
  - Substituir as combinações encontradas em uma string;
  - Dividir uma string em array baseado na regexp;

- **Seja curioso e procure, há várias funcionalidades!!!**

Teste de Presença

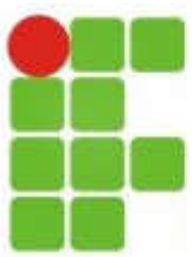


- Testa a expressão regular na string;
- Sintaxe:

```
regexp.test(string)
```

- Retorna **true** caso a expressão regular encontre correspondência na string, e **false** caso contrário;
- Ótimo para verificações em formulários.

Retorno da Primeira Combinação



- Testa a expressão regular na string e retorna a primeira combinação;

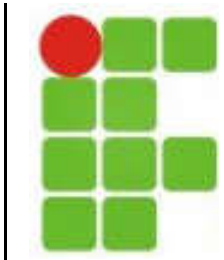
- Sintaxe:

```
regexp.exec(string)
```

- Retorna **uma string,** contendo a primeira combinação, caso se aplique, e **null** caso contrário;

- Ótimo para verificar o que foi combinado através de uma expressão regular mais complexa.

Substituição em uma String



- Testa a expressão regular na string, realiza a substituição da(s) combinação(ões), e retorna a string modificada;

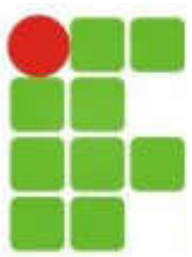
- Sintaxe:

```
string.replace(regexp, novastring)
```

- Nas combinações, novastring irá substituir a string encontrada;

- Caso nenhuma combinação for realizada, a string não será modificada.

Regra Básica



- Foco nas exceções;

- Os metacaracteres podem aparentar ser uma coisa mas podem representar outra completamente diferente;

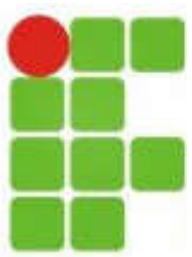
/abc/ → "abc"

/a.c/ ≠ somente "a.c"

/a.c/ → "abc", "a1c", "a=c", "a.c" ... exceto "a\nc"

- E.g., o metacaractere **.** representa qualquer caractere, exceto **\n**.

Letras Maiúsculas e Minúsculas



- As expressões regulares são case-sensitive, i.e., diferenciam letras maiúsculas e minúsculas;

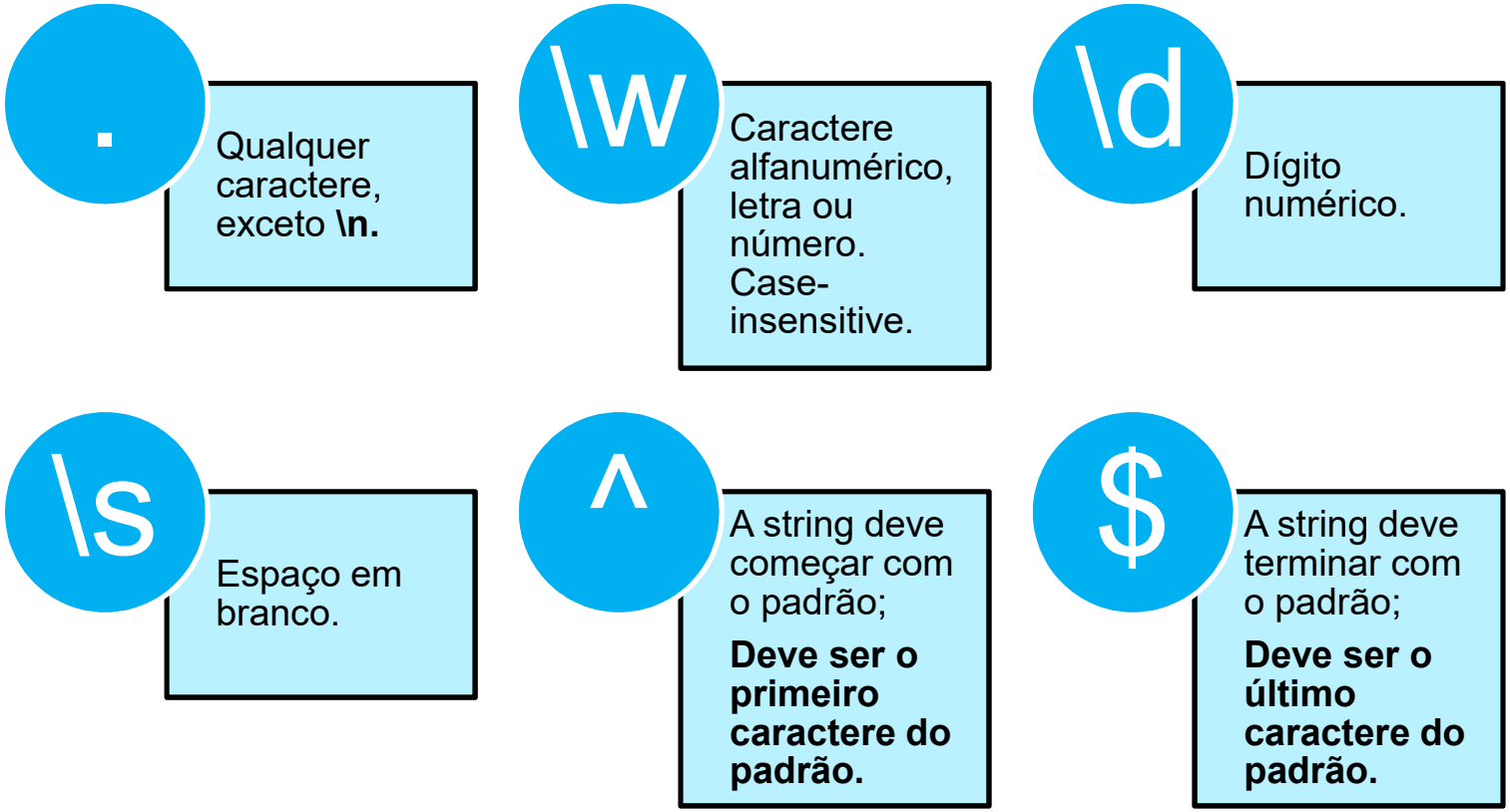
- Em Javascript, caso deseje-se suprimir esta diferenciação, acrescente um **i** após a expressão regular:

/jovem/ → "jovem"

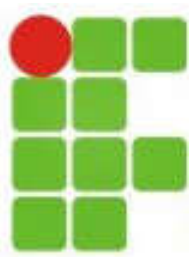
/jovem/i → "jovem", "JOVEM", "JoVeM"

Metacaracteres

- São caracteres que possuem combinação especial:



Exemplos com metacaracteres



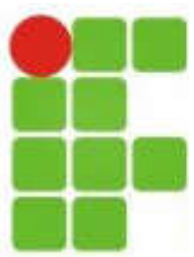
```
/\d\d\d\d\d-\d\d\d/
```

- Indica qualquer informação contendo 5 dígitos, seguido de um hífen (-), e por outros 3 dígitos.

```
/^\d\d\d\d\d-\d\d\d/
```

- Indica qualquer informação **que inicia** por 5 dígitos, seguido de um hífen (-), e por outros 3 dígitos.

Exemplos com metacaracteres



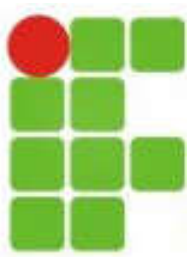
```
/^\d\d\d\d\d-\d\d\d$/
```

- Indica qualquer informação **composta exatamente** por 5 dígitos, seguido de um hífen (-), e por outros 3 dígitos.

```
/^\d\w\w$/
```

- Indica qualquer informação **composta exatamente** por 3 caracteres alfanuméricos, sendo o primeiro um dígito.

Desativando o significado de um metacaractere



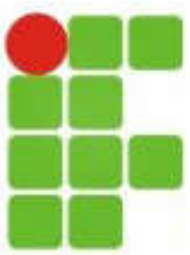
- Há momento que se deseja procurar literalmente um caractere que possui um significado especial, ou seja, é um metacaractere;

- Para suprimir seu significado especial, utiliza-se a barra invertida (\):

/a\.c/ → somente “a.c”

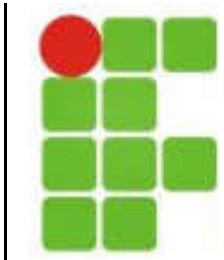
- Isto vale para os vários metacaracteres que vamos aprender.

Procurando uma gama de caracteres



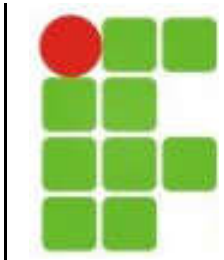
- FAZER

Estados de validação de um campo



Estado	Validação
tooLong	Indica se a quantidade de caracteres excede o especificado no atributo **maxlength.**
tooShort	Indica se a quantidade de caracteres é inferior ao especificado no atributo **minlength.**
badInput	Indica se o usuário especificou entrada que o navegador é incapaz de converter, e.g., **letra em Number**.
patternMismatch	Indica se o valor preenchido não corresponde ao padrão Regex especificado no atributo **pattern**.

Conclusão



- Com as ferramentas aprendidas aqui, temos uma maneira bem mais personalizada de validar formulários;
- Para impedir que um formulário seja submetido caso seja detectada falha em sua validação, retorne **false** na função de callback;
- Para permitir que o mesmo seja submetido normalmente, retorne **true;**
- Exemplos em **2-validação.html**