As linguagens regulares constituem a classe de linguagens com menor poder de representação, sendo possível desenvolver algoritmos de reconhecimento, existindo várias aplicações, como a analise léxica, sistemas de animação, hipertextos e hipermédia (Menezes, 2000). As linguagens regulares podem ser presentadas por um autómato finito e por uma Expressão Regular (REGEX).

REGEX – Expressões Regulares – uma forma sequencial de especificar uma linguagem regular, através de um padrão de Strings que descreve o mesmo que pode ser descrito por um autómato finito. Um exemplo, em notação UNIX de uma REGEX " [A-Z][a-z]*[ ][A-Z][A-Z]" representa uma palavra iniciada com maiúscula, seguida de espaço e duas maiúsculas, nesta seria aceite a sequência "Porto PT".

## Simulation REGEX

String:

Regex: / /

Flags: ignore case ( /i ) global ( /g ) multiline ( /m )

Test

### Replacement Methods

**Examples**

**Try Youself**

Test Test





V1.1 [03/01/2021]
Adicionada Esta Página de Suporte

Adicionado Repositório GitHub
V1.0 [14/09/2020]
Professor Doutor Jorge Morais apresenta tutorial do UAbALL: Vídeo

Ficheiros Exemplo para DFA:

1. Aceitação Strings binarias que não tenham consecutivos 1's: Download

2. Aceitação Strings binarias que não tenham consecutivos 1's [conceito encravamento]: Download

3. Aceitação Strings binarias de multimplos de 8: Download

4. Aceitação Strings numéricas de número par: Download

5. Aceitação Strings terminadas em "ing": Download

6. Aceitação Strings binárias com quatidade impar de 1's: Download

Manual v1.0

Relatório do Projecto - http://hdl.handle.net/10400.2/10079

O Automata Learning Lab da Universidade Aberta (UAbALL), pretende ser um laboratório integrado de simulação de autómatos. Numa primeira fase focado na construção da base e introduzindo a Simulação de Autómatos Finitos Deterministas (DFA). Este Laboratório ambiciona gozar de capacidade de extensibilidade e adaptabilidade, sendo este documento uma base técnica e científica para que no futuro sejam produzidas as restantes componentes, assim como adaptado a novas realidades tecnológicas e plataformas de distribuição.



HOME DFA NFA NFA-ε PDA TURING REGEX GRAMMAR HELP

As linguagens regulares constituem a classe de linguagens com menor poder de representação, sendo possível desenvolver algoritmos de reconhecimento, existindo várias aplicações, como a analise léxica, sistemas de animação, hipertextos e hipermédia (Menezes, 2000). As linguagens regulares podem ser presentadas por um autómato finito e por uma Expressão Regular (REGEX).

REGEX – Expressões Regulares – uma forma sequencial de especificar uma linguagem regular, através de um padrão de Strings que descreve o mesmo que pode ser descrito por um autómato finito. Um exemplo, em notação UNIX de uma REGEX " [A-Z][a-z]*[ ][A-Z][A-Z]" representa uma palavra iniciada com maiúscula, seguida de espaço e duas maiúsculas, nesta seria aceite a sequência "Porto PT".

## Simulation REGEX

String:

Regex: / /

Flags: ignore case ( /i ) global ( /g ) multiline ( /m )

Test

### Replacement Methods

**Examples**

Test

**Try Youself**

Test

Projeto de Engenharia Informática 2020 de André Maciel Sousa V.1.1

HOME DFA NFA NFA-ε PDA TURING REGEX GRAMMAR HELP

V1.1 [03/01/2021]
Adicionada Esta Página de Suporte

Adicionado Repositório GitHub
V1.0 [14/09/2020]
Professor Doutor Jorge Morais apresenta tutorial do UAbALL: Vídeo

Ficheiros Exemplo para DFA:

1. Aceitação Strings binarias que não tenham consecutivos 1's: Download

2. Aceitação Strings binarias que não tenham consecutivos 1's [conceito encravamento]: Download

3. Aceitação Strings binarias de multimplos de 8: Download

4. Aceitação Strings numéricas de número par: Download

5. Aceitação Strings terminadas em "ing": Download

6. Aceitação Strings binárias com quatidade impar de 1's: Download

Manual v1.0

Relatório do Projecto - http://hdl.handle.net/10400.2/10079

O Automata Learning Lab da Universidade Aberta (UAbALL), pretende ser um laboratório integrado de simulação de autómatos. Numa primeira fase focado na construção da base e introduzindo a Simulação de Autómatos Finitos Deterministas (DFA). Este Laboratório ambiciona gozar de capacidade de extensibilidade e adaptabilidade, sendo este documento uma base técnica e científica para que no futuro sejam produzidas as restantes componentes, assim como adaptado a novas realidades tecnológicas e plataformas de distribuição.

Projeto de Engenharia Informática 2020 de André Maciel Sousa V.1.1

V1.1 [03/01/2021]
Adicionada Esta Página de Suporte

Adicionado Repositório GitHub
V1.0 [14/09/2020]
Professor Doutor Jorge Morais apresenta tutorial do UAbALL: Vídeo

Ficheiros Exemplo para DFA:

1. Aceitação Strings binarias que não tenham consecutivos 1's: Download

2. Aceitação Strings binarias que não tenham consecutivos 1's [conceito encravamento]: Download

3. Aceitação Strings binarias de multimplos de 8: Download

4. Aceitação Strings numéricas de número par: Download

5. Aceitação Strings terminadas em "ing": Download

6. Aceitação Strings binárias com quatidade impar de 1's: Download

Manual v1.0

Relatório do Projecto - http://hdl.handle.net/10400.2/10079

O Automata Learning Lab da Universidade Aberta (UAbALL), pretende ser um laboratório integrado de simulação de autómatos. Numa primeira fase focado na construção da base e introduzindo a Simulação de Autómatos Finitos Deterministas (DFA). Este Laboratório ambiciona gozar de capacidade de extensibilidade e adaptabilidade, sendo este documento uma base técnica e científica para que no futuro sejam produzidas as restantes componentes, assim como adaptado a novas realidades tecnológicas e plataformas de distribuição.

V1.1 [03/01/2021]
Adicionada Esta Página de Suporte

Adicionado Repositório GitHub
V1.0 [14/09/2020]
Professor Doutor Jorge Morais apresenta tutorial do UAbALL: Vídeo

Ficheiros Exemplo para DFA:

1. Aceitação Strings binarias que não tenham consecutivos 1's: Download

2. Aceitação Strings binarias que não tenham consecutivos 1's [conceito encravamento]: Download

3. Aceitação Strings binarias de multimplos de 8: Download

4. Aceitação Strings numéricas de número par: Download

5. Aceitação Strings terminadas em "ing": Download

6. Aceitação Strings binárias com quatidade impar de 1's: Download

Manual v1.0

Relatório do Projecto - http://hdl.handle.net/10400.2/10079

O Automata Learning Lab da Universidade Aberta (UAbALL), pretende ser um laboratório integrado de simulação de autómatos. Numa primeira fase focado na construção da base e introduzindo a Simulação de Autómatos Finitos Deterministas (DFA). Este Laboratório ambiciona gozar de capacidade de extensibilidade e adaptabilidade, sendo este documento uma base técnica e científica para que no futuro sejam produzidas as restantes componentes, assim como adaptado a novas realidades tecnológicas e plataformas de distribuição.

As linguagens regulares constituem a classe de linguagens com menor poder de representação, sendo possível desenvolver algoritmos de reconhecimento, existindo várias aplicações, como a analise léxica, sistemas de animação, hipertextos e hipermédia (Menezes, 2000). As linguagens regulares podem ser presentadas por um autómato finito e por uma Expressão Regular (REGEX).

REGEX – Expressões Regulares – uma forma sequencial de especificar uma linguagem regular, através de um padrão de Strings que descreve o mesmo que pode ser descrito por um autómato finito. Um exemplo, em notação UNIX de uma REGEX " [A-Z][a-z]*[ ][A-Z][A-Z]" representa uma palavra iniciada com maiúscula, seguida de espaço e duas maiúsculas, nesta seria aceite a sequência "Porto PT".

---

## Simulation REGEX

String:

Regex: / /

Flags: ignore case ( /i ) global ( /g ) multiline ( /m )

Test

---

## Replacement Methods

### Examples

Test

### Try Youself

Test

As linguagens regulares constituem a classe de linguagens com menor poder de representação, sendo possível desenvolver algoritmos de reconhecimento, existindo várias aplicações, como a analise léxica, sistemas de animação, hipertextos e hipermédia (Menezes, 2000). As linguagens regulares podem ser presentadas por um autómato finito e por uma Expressão Regular (REGEX).

REGEX – Expressões Regulares – uma forma sequencial de especificar uma linguagem regular, através de um padrão de Strings que descreve o mesmo que pode ser descrito por um autómato finito. Um exemplo, em notação UNIX de uma REGEX " [A-Z][a-z]*[ ][A-Z][A-Z]" representa uma palavra iniciada com maiúscula, seguida de espaço e duas maiúsculas, nesta seria aceite a sequência "Porto PT".

## Simulation REGEX

String:

Regex: / /

Flags: ignore case ( /i ) global ( /g ) multiline ( /m )

Test

## Replacement Methods

**Examples**

**Try Youself**

Test

Test