Strings

Programação Orientada a Objetos — QXD0007

UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

Prof. Atílio Gomes Luiz

gomes.atilio@ufc.br

Universidade Federal do Ceará

2º semestre/2021

Leituras para esta aula

- **Capítulo 14** (Strings, caracteres e expressões regulares) do livro Java Como Programar, Décima Edição, Disponível no link: `http://libgen.lc/ads.php?md5=728636A04ACA056038BB5F079403AC96`

Strings

- Strings são sequências de caracteres, que podem conter letras, dígitos e caracteres especiais.

- No Java, uma string é um objeto da classe `String`.

Strings

- Strings são sequências de caracteres, que podem conter letras, dígitos e caracteres especiais.

- No Java, uma string é um objeto da classe `String`.

- String literais são sequências de caracteres no código escritas entre aspas duplas. Ex.: "Amanda Costa".
  - São armazenadas na memória como objetos da classe `String`

Strings

- Strings são sequências de caracteres, que podem conter letras, dígitos e caracteres especiais.
- No Java, uma string é um objeto da classe `String`.
- String literais são sequências de caracteres no código escritas entre aspas duplas. Ex.: "Amanda Costa".
  - São armazenadas na memória como objetos da classe `String`
- No Java, strings são constantes: o seu valor não pode ser mudado depois que elas são criadas.

Strings

- Strings são sequências de caracteres, que podem conter letras, dígitos e caracteres especiais.
- No Java, uma string é um objeto da classe `String`.
- String literais são sequências de caracteres no código escritas entre aspas duplas. Ex.: "Amanda Costa".
  - São armazenadas na memória como objetos da classe `String`
- No Java, strings são constantes: o seu valor não pode ser mudado depois que elas são criadas.

A fim de conservar memória, o Java trata todas as string literais que possuem o mesmo conteúdo como um único objeto `String`.

Classe `String` – Construtores

- A classe `String` provê diversos construtores, alguns dos quais estão listados abaixo.

Construtor	Descrição
`String`()	Instancia um objeto String que representa uma sequência de caracteres vazia.
`String`(String s)	Instancia um objeto String que é uma cópia da string do argumento.
`String`(char[] v)	Aloca uma nova String que representa a sequência de caracteres no array.
`String`(char[] v, int offset, int count)	Aloca uma nova String que contém caracteres de uma subvetor do array de caracteres passado como argumento. `offset` é a posição inicial e `count` é o número de caracteres a ser copiado a partir dessa posição.

Classe `String` – Construtores

- A classe `String` provê diversos construtores, alguns dos quais estão listados abaixo.

Construtor	Descrição
`String`()	Instancia um objeto String que representa uma sequência de caracteres vazia.
`String`(String s)	Instancia um objeto String que é uma cópia da string do argumento.
`String`(char[] v)	Aloca uma nova String que representa a sequência de caracteres no array.
`String`(char[] v, int offset, int count)	Aloca uma nova String que contém caracteres de uma subvetor do array de caracteres passado como argumento. `offset` é a posição inicial e `count` é o número de caracteres a ser copiado a partir dessa posição.

- Exemplo: arquivo `StringConstructors.java`

Classe `String` – Métodos

- Alguns métodos básicos da classe `String` servem para que possamos processar os caracteres que formam a string.

Método	Descrição
int `length`()	Retorna o número de caracteres da string.
char `charAt`(int index)	Retorna o caractere no índice especificado.
void `getChars`(int begin, int end, char[] dest, int destBegin)	Copia caracteres desta string para o array de caracteres dest passado por parâmetro. Os caracteres são copiados a partir da posição begin até a posição end-1. O último argumento especifica a posição inicial onde os caracteres copiados devem ser colocados no array dest

Classe `String` – Métodos

- Alguns métodos básicos da classe `String` servem para que possamos processar os caracteres que formam a string.

Método	Descrição
int `length`()	Retorna o número de caracteres da string.
char `charAt`(int index)	Retorna o caractere no índice especificado.
void `getChars`(int begin, int end, char[] dest, int destBegin)	Copia caracteres desta string para o array de caracteres dest passado por parâmetro. Os caracteres são copiados a partir da posição begin até a posição end-1. O último argumento especifica a posição inicial onde os caracteres copiados devem ser colocados no array dest

- Analisar os arquivos `JogoDaForca.java` e `DemoJogoDaForca.java`

Classe `String` – Métodos

Comparando duas strings

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

- É frequente a necessidade de comparação de valores de variáveis ou atributos com outras variáveis, atributos ou constantes, para tomada de decisões em programas.

- Instâncias da classe `String` não podem ser comparadas com o operador de igualdade $==$.

- Analisar o arquivo `ComparandoStrings.java`

Classe `String` – Métodos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Comparando duas strings

Método	Descrição
boolean equals(Object obj)	Compara esta string com o objeto especificado. O resultado é verdadeiro se e somente se o argumento não for null e for um objeto String que representa a mesma sequência de caracteres que este objeto.
boolean equalsIgnoreCase(String str)	Compara esta string com str, ignorando o case.
int compareTo(String str)	Compara duas strings lexicograficamente. Retorna um inteiro negativo se esta String for menor que o argumento str. Retorna um inteiro positivo se esta String for maior que o argumento. Retorna 0 se as strings forem iguais.
boolean regionMatches(int toffset, String other, int ooffset, int len)	Determina se uma substring desta String é igual a uma substring da String other do argumento. A substring desse objeto String a ser comparada começa no índice toffset e tem comprimento len. A substring de other a ser comparada começa no índice ooffset e tem comprimento len.

Classe `String` – Métodos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Comparando duas strings

Método	Descrição
boolean equals(Object obj)	Compara esta string com o objeto especificado. O resultado é verdadeiro se e somente se o argumento não for null e for um objeto String que representa a mesma sequência de caracteres que este objeto.
boolean equalsIgnoreCase(String str)	Compara esta string com str, ignorando o case.
int compareTo(String str)	Compara duas strings lexicograficamente. Retorna um inteiro negativo se esta String for menor que o argumento str. Retorna um inteiro positivo se esta String for maior que o argumento. Retorna 0 se as strings forem iguais.
boolean regionMatches(int toffset, String other, int ooffset, int len)	Determina se uma substring desta String é igual a uma substring da String other do argumento. A substring desse objeto String a ser comparada começa no índice toffset e tem comprimento len. A substring de other a ser comparada começa no índice ooffset e tem comprimento len.

Analisar arquivo `StringCompare.java`.

Classe `String` – Substrings

Método	Descrição
int indexOf(int ch)	Retorna o índice da primeira ocorrência do caractere ch nesta string. Retorna -1 se ch não for encontrado
int indexOf(String str)	Retorna o índice da primeira ocorrência da string str dentro desta string. Retorna -1 se str não for encontrado.
int indexOf(String str, int index)	Retorna o índice da primeira ocorrência da string str dentro desta string, começando a partir do índice index. Retorna -1 se str não for encontrado.
int lastIndexOf(int ch)	Retorna o índice da última ocorrência do caractere ch nesta string. Retorna -1 se ch não for encontrado.
int lastIndexOf(String str)	Retorna o índice da última ocorrência da string str dentro desta string. Retorna -1 se str não for encontrado.
int lastIndexOf(String str, int i)	Retorna o índice da última ocorrência da string str dentro desta string, começando de trás para frente a partir do índice i. Retorna -1 se str não for encontrado.

Classe `String` – Substrings

Método	Descrição
String substring(int from)	Retorna uma string que é uma substring desta string. A substring começa com o caractere na posição from e vai até o final da string original.
String substring(int from, int to)	Retorna uma string que é uma substring desta string. A substring vai da posição from até a posição to-1 da String original.
boolean endsWith(String suffix)	Testa se esta String termina com o sufixo especificado no argumento.
boolean startsWith(String preffix)	Testa se esta String inicia com o prefixo especificado no argumento.

Classe `String` – Substrings

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
String substring(int from)	Retorna uma string que é uma substring desta string. A substring começa com o caractere na posição from e vai até o final da string original.
String substring(int from, int to)	Retorna uma string que é uma substring desta string. A substring vai da posição from até a posição to-1 da String original.
boolean endsWith(String suffix)	Testa se esta String termina com o sufixo especificado no argumento.
boolean startsWith(String preffix)	Testa se esta String inicia com o prefixo especificado no argumento.

- Analisar exemplo SubString.java e StringStartEnd.java

Classe `String` – Métodos diversos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
String concat(String str)	Concatena a string str, passado como argumento, ao final desta String.
String toUpperCase()	Converte todos os caracteres desta String para maiúsculo.
String toLowerCase()	Converte todos os caracteres desta String para minúsculo.
String replace(char oldChar, char newChar)	Retorna uma string resultado da troca de todas as ocorrências de oldChar nesta string por newChar.
String trim()	Retorna uma string cujo valor é esta string, com todos os espaços iniciais e finais removidos.
char[] toCharArray()	Converte esta String para um novo array de caracteres do tamanho da string.

Classe `String` – Métodos diversos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
String concat(String str)	Concatena a string str, passado como argumento, ao final desta String.
String toUpperCase()	Converte todos os caracteres desta String para maiúsculo.
String toLowerCase()	Converte todos os caracteres desta String para minúsculo.
String replace(char oldChar, char newChar)	Retorna uma string resultado da troca de todas as ocorrências de oldChar nesta string por newChar.
String trim()	Retorna uma string cujo valor é esta string, com todos os espaços iniciais e finais removidos.
char[] toCharArray()	Converte esta String para um novo array de caracteres do tamanho da string.

- Analisar arquivo StringMiscellaneous2.java

Classe `String` – Conversão com `valueOf`

- A classe `String` fornece várias versões sobrecarregadas do método estático `valueOf` que recebem um argumento de tipo nativo e o converte para String.

- Analisar arquivo `StringValueOf.java`

Tokenização de strings

- Tokens são palavras individuais que transmitem um significado dentro de um texto.

- Os tokens são separados entre si por delimitadores, em geral caracteres de espaçamento como espaço, tabulação e nova linha. Porém, outros caracteres também podem ser utilizados como delimitadores para separar tokens.

Tokenização de strings

- Tokens são palavras individuais que transmitem um significado dentro de um texto.

- Os tokens são separados entre si por delimitadores, em geral caracteres de espaçamento como espaço, tabulação e nova linha. Porém, outros caracteres também podem ser utilizados como delimitadores para separar tokens.

- Em Java, uma string pode ser dividida em tokens usando o método `split` da classe `String`.

Método	Descrição
String [] split(String delim)	Retorna um array de strings que são os tokens da string original. Os tokens são gerados com base no delimitador delim passado como entrada.

Tokenização de strings

- Tokens são palavras individuais que transmitem um significado dentro de um texto.

- Os tokens são separados entre si por delimitadores, em geral caracteres de espaçamento como espaço, tabulação e nova linha. Porém, outros caracteres também podem ser utilizados como delimitadores para separar tokens.

- Em Java, uma string pode ser dividida em tokens usando o método `split` da classe `String`.

Método	Descrição
String [] split(String delim)	Retorna um array de strings que são os tokens da string original. Os tokens são gerados com base no delimitador delim passado como entrada.

Analisar o arquivo TokenTest.java

Exercício

- Escreva um programa que lê um parágrafo e mostra as palavras individuais como uma lista.
  - Primeiro mostre todas as palavras.
  - Então, mostre todas as palavras em ordem reversa.
  - Então, mostre-as de modo que todas as palavras em plural estejam com letra maiúscula.
  - Por fim, mostre-as com todas as palavras em plural removidas.

Exercício

- Escreva um programa que lê um parágrafo e mostra as palavras individuais como uma lista.
  - Primeiro mostre todas as palavras.
  - Então, mostre todas as palavras em ordem reversa.
  - Então, mostre-as de modo que todas as palavras em plural estejam com letra maiúscula.
  - Por fim, mostre-as com todas as palavras em plural removidas.

Solução: Ver o arquivo Paragrafo.java

Character

A classe wrapper `Character`

- A maioria dos métodos da classe `Character` são métodos estáticos projetados para processar valores `char` individuais.
- Esses métodos aceitam pelo menos um argumento caractere e realizam um teste ou uma manipulação do caractere.
- Consulte a API: `https://docs.oracle.com/en/java/javase/16/docs/api/java.base/java/lang/Character.html`

Métodos estáticos da classe `Character`

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
boolean isDefined(char ch)	Determina se ch é um caractere Unicode.
boolean isJavaIdentifierStart(char ch)	Determina se ch é um caractere que pode ser o primeiro caractere de um identificador no Java.
boolean isUpperCase(char ch)	Determina se ch é caractere maiúsculo.
boolean isLowerCase(char ch)	Determina se ch é caractere minúsculo.
boolean isLetter(char ch)	Determina se ch é uma letra
boolean isDigit(char ch)	Determina se ch é um dígito (0..9)
boolean isLetterOrDigit(char ch)	Determina se ch é letra ou dígito.
boolean isWhitespace(char ch)	Determina se ch é um espaço em branco
char toUpperCase(char ch)	Converte ch para maiúsculo.
char toLowerCase(char ch)	Converte ch para minúsculo.
String toString(char ch)	Converte ch para String.

Métodos estáticos da classe `Character`

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
boolean isDefined(char ch)	Determina se ch é um caractere Unicode.
boolean isJavaIdentifierStart(char ch)	Determina se ch é um caractere que pode ser o primeiro caractere de um identificador no Java.
boolean isUpperCase(char ch)	Determina se ch é caractere maiúsculo.
boolean isLowerCase(char ch)	Determina se ch é caractere minúsculo.
boolean isLetter(char ch)	Determina se ch é uma letra
boolean isDigit(char ch)	Determina se ch é um dígito (0..9)
boolean isLetterOrDigit(char ch)	Determina se ch é letra ou dígito.
boolean isWhitespace(char ch)	Determina se ch é um espaço em branco
char toUpperCase(char ch)	Converte ch para maiúsculo.
char toLowerCase(char ch)	Converte ch para minúsculo.
String toString(char ch)	Converte ch para String.

- Analisar o arquivo StaticCharMethods.java

StringBuilder

Classe `StringBuilder`

- StringBuilder é uma string modificável. Toda StringBuilder possui uma capacidade. Se sua capacidade for excedida, ela é expandida a fim de acomodar os caracteres adicionais.

- Se um programa executa muitas operações de concatenação, ou outras modificações de strings, pode ser mais eficiente implementar essas modificações com a classe `StringBuilder`.

- API do Java: `https://docs.oracle.com/en/java/javase/16/docs/api/java.base/java/lang/StringBuilder.html`

`StringBuilder` — Construtores

- `StringBuilder` fornece 4 construtores, três deles exibidos abaixo.

Método	Descrição
StringBuilder()	Constrói uma StringBuilder vazia com capacidade inicial para 16 caracteres.
StringBuilder(int cap)	Constrói uma StringBuilder vazia com capacidade inicial igual a cap.
StringBuilder(String str)	Constrói uma StringBuilder e a inicializa com o valor de str.

`StringBuilder` — Construtores

- `StringBuilder` fornece 4 construtores, três deles exibidos abaixo.

Método	Descrição
StringBuilder()	Constrói uma StringBuilder vazia com capacidade inicial para 16 caracteres.
StringBuilder(int cap)	Constrói uma StringBuilder vazia com capacidade inicial igual a cap.
StringBuilder(String str)	Constrói uma StringBuilder e a inicializa com o valor de str.

Exemplo: ver arquivo StringBuilderConstructors.java

StringBuilder — Métodos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
int length()	Retorna o número de caracteres atualmente na StringBuilder
int capacity()	Retorna o número total de caracteres que pode ser armazenado.
void ensureCapacity(int min)	Garante que a capacidade seja pelo menos igual ao mínimo min especificado. A nova capacidade é o maior valor entre min e duas vezes a capacidade anterior mais 2.
void setLength(int n)	A sequência é alterada para uma nova sequência de caracteres cujo comprimento é especificado pelo argumento $n \geq 0$. Se n for maior que a string atual, o valor excedente é preenchido com null '\u0000'

StringBuilder — Métodos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
int length()	Retorna o número de caracteres atualmente na StringBuilder
int capacity()	Retorna o número total de caracteres que pode ser armazenado.
void ensureCapacity(int min)	Garante que a capacidade seja pelo menos igual ao mínimo min especificado. A nova capacidade é o maior valor entre min e duas vezes a capacidade anterior mais 2.
void setLength(int n)	A sequência é alterada para uma nova sequência de caracteres cujo comprimento é especificado pelo argumento $n \geq 0$. Se n for maior que a string atual, o valor excedente é preenchido com null '\u0000'

Analisar o arquivo StringBuilderCapLen.java

StringBuilder — Métodos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
char charAt(int index)	Retorna o caractere no índice especificado
void setCharAt(int i, char ch)	O caractere no índice i é modificado para ch
void getChars(int srcBegin, int srcEnd, char[] dest, int destBegin)	Os caracteres são copiados desta string para o array dest. O primeiro caractere a ser copiado está na posição srcBegin; o último caractere a ser copiado está na posição srcEnd-1. Os caracteres são copiados no array dest começando no índice destBegin.
StringBuilder reverse()	Esta sequência de caracteres é invertida.

StringBuilder — Métodos

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Método	Descrição
char charAt(int index)	Retorna o caractere no índice especificado
void setCharAt(int i, char ch)	O caractere no índice i é modificado para ch
void getChars(int srcBegin, int srcEnd, char[] dest, int destBegin)	Os caracteres são copiados desta string para o array dest. O primeiro caractere a ser copiado está na posição srcBegin; o último caractere a ser copiado está na posição srcEnd-1. Os caracteres são copiados no array dest começando no índice destBegin.
StringBuilder reverse()	Esta sequência de caracteres é invertida.

Analisar arquivo StringBuilderChars.java

StringBuilder — Método append

- A classe StringBuilder fornece versões sobrecarregadas do método append, que recebe um único valor como argumento e anexa a string representante deste valor à StringBulder atual. Este método retorna a StringBuilder resultante.

- Versões do método append são fornecidas para os tipos nativos, arrays de caracteres, Strings e Objects.

`StringBuilder` — Método `insert`

- StringBuilder fornece versões sobrecarregadas do método `insert`.
- Este método recebe dois argumentos: o primeiro é o índice em que o valor deve ser inserido e o segundo argumento é o valor a ser inserido.
  - O índice deve ser maior ou igual a 0 e menor ou igual ao comprimento da sequência.

`StringBuilder` — Método `insert`

- StringBuilder fornece versões sobrecarregadas do método `insert`.
- Este método recebe dois argumentos: o primeiro é o índice em que o valor deve ser inserido e o segundo argumento é o valor a ser inserido.
  - O índice deve ser maior ou igual a 0 e menor ou igual ao comprimento da sequência.
- Versões do método insert são fornecidas para os tipos nativos, arrays de caracteres, Strings e Objects.

`StringBuilder` — Removendo

- StringBuilder fornece os métodos `delete` e `deleteCharAt` para deletar caracteres em qualquer posição de uma StringBuilder.

- `StringBuilder delete(int start, int end)`
  - A substring a ser excluída começa na posição start e termina na posição end-1 ou vai até o fim desta sequência se essa posição não existir. Se start == end, nenhuma modificação é feita.

- `StringBuilder deleteCharAt(int index)`
  - Remove o char na posição especificada no argumento.

`StringBuilder` — Removendo

- StringBuilder fornece os métodos `delete` e `deleteCharAt` para deletar caracteres em qualquer posição de uma StringBuilder.

- `StringBuilder delete(int start, int end)`
  - A substring a ser excluída começa na posição start e termina na posição end-1 ou vai até o fim desta sequência se essa posição não existir. Se start == end, nenhuma modificação é feita.

- `StringBuilder deleteCharAt(int index)`
  - Remove o char na posição especificada no argumento.

- Analisar o arquivo StringBuilderInsertDelete.java

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

FIM