**01Char e String**

**Transcrição**

A seguir, trabalharemos com caracteres e palavras! Criaremos uma nova classe mais uma vez, a "TestaCaracteres". Existe uma variável primitiva básica do Java que trabalha com *chars*, isto é, caracteres, cuja peculiaridade é guardar um único caractere de 16bits.

Usaremos as aspas simples para guardar a letra a, por exemplo:

public class TestaCaracteres {

public static void main(String[] args) {

char letra = 'a';

System.out.println(letra);

}

}Copiar código

Ao salvarmos e rodarmos este código, lê-se a no Console, nada muito especial.

Quando trabalhamos com chars, estamos realmente "presos" a um único caractere. Se substituirmos a do código acima por ab, o código não compilará, e o mesmo ocorrerá se utilizarmos aspas duplas em vez das simples. O char guarda em si um único código, um número da tabela de Unicode, como a ASCII, porém muito maior e sem limite definido.

letra, portanto, é um número e, se observarmos bem, o char guarda em seu valor um número, mas é uma variável do tipo numérico equivalente àquele short, mas ele contém apenas valores positivos, possuindo mais detalhes. No momento, é interessante sabermos que ele é um número que é convertido em uma letra, como no trecho a seguir:

char valor = 66;

System.out.println(valor);Copiar código

A partir do qual obteremos:

BCopiar código

Isto ocorre pois na tabela Unicode o 65 corresponde à letra a, portanto 66 refere-se a b. Testando-se o código abaixo,

valor = valor + 1;

System.out.println(valor);Copiar código

há um erro de compilação em valor + 1, por conta da regra do Java quando se trabalha com dois tipos distintos em uma mesma operação, de dar o resultado no maior deles. Neste caso, o valor é do tipo char, e 1 é um int, que é maior. O resultado desta operação, portanto, será dado em int. No entanto, um inteiro cabe em um char? Não! Porém, novamente, o inverso é possível.

Se queremos que isto seja válido, devemos informar que a resposta disso passará pelo *casting*, moldando-se para o char:

valor = (char) (valor + 1);

System.out.println(valor);Copiar código

Salvando e rodando o código, receberemos a letra C. O char é interessante, mas não é tão usado no dia a dia, como no caso de String, com S em maiúsculo. Ela não é palavra chave do Java, não guarda valor, é um tipo referência. As diferenças ficarão mais claras quando formos entender melhor sobre orientação a objetos.

Atenção: o funcionamento básico de uma String exige aspas duplas, e não simples, as quais podem inclusive ficar vazias (""). Em char, por outro lado, não é possível deixar as aspas simples sem nada dentro ('') - um espaço seria algo, e compilaria. Um char vazio, não.

String palavra = "alura cursos online de tecnologia";

System.out.println(palavra);Copiar código

Salvando e rodando o código, teremos a impressão alura cursos online de tecnologia, como esperado. E é possível utilizarmos o operador de soma (**+**) para concatenar Strings, criando uma nova, como no exemplo abaixo:

palavra = palavra + 2020;

System.out.println(palavra);Copiar código

Isto nos retornará alura cursos online de tecnologia2020. A String, então, não se comporta como um int ou um char, mas aparecerá recorrentemente. Em breve veremos que ela faz referência a um objeto e possui vários métodos. Ainda precisaremos aprender o básico e aprofundarmos nossos conhecimentos com calma!

**02Declarando String e char**

Rômulo é um grande entusiasta da linguagem Java e está sempre disposto a aprender novas coisas. A última coisa que Rômulo resolveu aprender são Strings e chars em Java, porém ele ainda não tem certeza de como deve declarar cada um dos tipos. Qual das alternativas têm a resposta correta para Rômulo?

Parte superior do formulário

* String frase = 'Estou sempre disposto a aprender novas tecnologias";
* char letra = 'Uma palavra';
* Alternativa correta
* String frase = "Estou sempre disposto a aprender novas tecnologias";

char letra = 'Uma palavra';

* Alternativa correta
* String frase = "Estou sempre disposto a aprender novas tecnologias";

char letra = 'a';

Correto! Lembre-se, uma String é declarada com aspas duplas " e pode ter zero ou mais caracteres. Um char é declarado com aspas simples ' e pode usar apenas um caractere!

* Alternativa correta
* string frase = "Estou sempre disposto a aprender novas tecnologias";

char letra = 'a';

* Alternativa correta
* String frase = 'Estou sempre disposto a aprender novas tecnologias';

char letra = "a";

Parte inferior do formulário

Parabéns, você acertou!

Este conteúdo foi útil para o seu aprendizado?

Sim

Não muito

**03Qual será o resultado?**

Rômulo, depois de assistir aos vídeos deste capítulo, resolveu testar o que foi dito em aula e fez um código para somar dois valores quaisquer.

String parcela1 = "10";

String parcela2 = "20";

System.out.println(parcela1 + parcela2);Copiar código

Rômulo terá o resultado esperado (30) com este trecho de código?

Parte superior do formulário

* Sim, o resultado será 30.
* Alternativa correta

Sim, porém o resultado será "30" e não 30.

* Alternativa correta

Não, pois não podemos utilizar o sinal + neste caso.

* Alternativa correta

Não, o resultado será 1020.

Correto. O resultado será a concatenação das duas parcelas.

Parte inferior do formulário

Parabéns, você acertou!

Este conteúdo foi útil para o seu aprendizado?

Sim

Não muito

**04Variáveis guardam valores**

**Transcrição**

Um último detalhe muito interessante sobre estas variáveis do tipo primitivo - todas aquelas que vimos exceto a String - é seu funcionamento interno. O que são guardadas na memória delas?

Vamos criar mais uma classe, o TestaValores. E para não ficarmos digitando public static void main(String[] args) {} à mão o tempo todo, aprenderemos um atalho. Digitaremos "main" e apertaremos "Ctrl + barra de espaço" que, assim como em outros editores, tem a ver com o *autocomplete*. No Eclipse, também envolve ***templates***.

Por meio deste atalho, aparecerão algumas opções, apertaremos a tecla "Enter", e o código aparece pronto no editor de texto. Isso passará a ser frequente para vocês.

Para entendermos como é guardado o valor de uma variável no Java, a **passagem por valor**, vamos fazer um desafio:

public class TestaValores {

public static void main(String[] args) {

int primeiro = 5;

int segundo = 7;

System.out.println(segundo);Copiar código

Ao rodarmos o código, obteremos 7.

public class TestaValores {

public static void main(String[] args) {

int primeiro = 5;

int segundo = 7;

segundo = primeiro;

System.out.println(segundo);Copiar código

Salvando e rodando este código, obteremos 5!

int primeiro = 5;

int segundo = 7;

segundo = primeiro;

primeiro = 10;

// quanto vale o segundo?

System.out.println(segundo);Copiar código

No segundo, tínhamos guardado o primeiro, mas agora primeiro vale 10. Quanto vale segundo?

As linguagens de programação trabalham de formas diferentes dependendo do uso de um símbolo específico, ou da existência de alguma referência, e por aí vai. Estas variáveis do tipo primitivo são trabalhadas com o valor do conteúdo, da variável, então, quando copiamos 5 para dentro de segundo, e depois copiamos 10 para primeiro, a linha segundo = primeiro; não diz nada.

Quando se faz uma atribuição no Java, não se diz que uma variável **sempre** segue a outra, e sim que estamos **copiando e colando valores**. Deste modo, primeiro = 10; não surtirá efeito para segundo. Confirmaremos isto rodando a aplicação, pois continuaremos recebendo 5.

Isso significa que a variável guarda um valor, e não uma referência, e este exemplo dará base para as entendermos melhor.

Estamos prontos para o próximo passo, que consiste em finalmente começarmos com controle de fluxos, com if, while e for, para estruturarmos nossos primeiros programas! E então veremos a orientação a objetos (O.O.) de maneira contra procedural. Vamos lá?

**05Concatenação de String e inteiros**

Paulo, vendo que Rômulo continuava com algumas dificuldades com Java, resolveu elaborar um desafio para seu amigo utilizando conceitos de concatenação de Strings, vistos neste capítulo. O desafio continha o seguinte trecho de código:

String saudacao = "Olá, meu nome é ";

String nome = "Rômulo ";

String continuacao = "e minha idade é ";

int idade = 100;

System.out.println(saudacao+nome+continuacao+idade);Copiar código

O código de Paulo para Rômulo possui algum problema?

Parte superior do formulário

* Sim, pois não é possível concatenar uma String com um int
* Alternativa correta

Sim, pois o correto é Int e não int

* Alternativa correta

Não há problemas, a concatenação pode ser feita sem problemas.

Correto!

Parte inferior do formulário

Parabéns, você acertou!

Este conteúdo foi útil para o seu aprendizado?

Sim

Não muito

**06Mão na massa: Praticando char e String**

Dessa vez, vamos praticar um pouco do uso dos tipos **char** e **String**.

1) Crie uma classe e estruture seu método main, na aula utilizamos o nome "**TestaCaracteres**"

2) Crie uma variável do tipo **char**, como fizemos abaixo:

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

}

}

Copiar código

3) Imprima a variável no console:

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

System.out.println(letra);

}

}

Copiar código

Nossa saída será: **a**

4) Como falado no vídeo, o tipo **char** guarda apenas um caractere da tabela **unicode**, por isso, quando passamos um valor numérico, teremos como saída o caractere correspondente ao número na tabela **unicode**. Escreva o seguinte código e veja:

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

System.out.println(letra);

char valor = 65; // Compila!

System.out.println(valor);

}

}

Copiar código

A saída do nosso segundo System.out.println será : **A**

5) Para um segundo teste, escreva o seguinte:

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

System.out.println(letra);

char valor = 65; // Compila!

System.out.println(valor);

valor = valor + 1; // Não compila!

System.out.println(valor);

}

}

Copiar código

Nosso código não compila, quando escrevemos essa soma, o Java automaticamente transforma o resultado para o **maior** tipo, que nesse caso é **int**, já que o número **1** é um inteiro.

6) Faça um casting para que esse código compile, para isso devemos escrever:

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

System.out.println(letra);

char valor = 65; // Compila!

System.out.println(valor);

valor = (char) (valor + 1); // Compila!

System.out.println(valor);

}

}

Copiar código

Repare que envolvemos a soma com parênteses, dessa forma, dizemos para o Java que queremos transformar o resultado dessa soma em **char**.

7) Crie agora uma variável do tipo **String**, repare que esse tipo é diferente dos que tinhamos usado até o momento, além disso, escrevemos com a inicial maiúscula.

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

System.out.println(letra);

char valor = 65; // Compila!

System.out.println(valor);

valor = (char) (valor + 1); // Compila!

System.out.println(valor);

String palavra = "alura cursos online de tecnologia";

}

}

Copiar código

Para **String**, usaremos obrigatóriamente aspas duplas.

8) Imprima a **String** no console. Nosso código terá a seguinte cara:

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

System.out.println(letra);

char valor = 65; // Compila!

System.out.println(valor);

valor = (char) (valor + 1);

System.out.println(valor);

String palavra = "alura cursos online de tecnologia";

System.out.println(palavra);

}

}

Copiar código

9) Concatene sua **String** com o número **2020** da seguinte forma:

public class TestaCaracteres{

public static void main(String[] args){

char letra = 'a';

System.out.println(letra);

char valor = 65; // Compila!

System.out.println(valor);

valor = (char) (valor + 1); // Compila!

System.out.println(valor);

String palavra = "alura cursos online de tecnologia";

System.out.println(palavra);

palavra = palavra + 2020;

System.out.println(palavra);

}

}

Copiar código

Repare que o número 2020 aparece no final da frase, isso acontece porque que o Java converte automaticamente o número para **String**, diferente do que acontecia com os tipos primitivos que vimos anteriormente.

**Opinião do instrutor**

Alguma dúvida? Pergunte no fórum!

**07O que aprendemos?**

Este capítulo apresentou:

* O conceito e como declarar char e String
* Como concatenar Strings
* Atalho para criação da main
* Variáveis guardam valores e não referências.

Este conteúdo foi útil para o seu aprendizado?

Sim

Não muito

**08Arquivos do projeto atual**

No link abaixo, você encontra o projeto até o momento atual do curso.

<https://github.com/alura-cursos/Curso-Java-parte-1-Primeiros-passos/archive/capitulo5.zip>

**Opinião do instrutor**

Lembre-se de usar o fórum em caso de dúvidas.