**Sintaxe básica de Java e autonomia de Classes**

**Objetivo do curso**

Apresentar regras essenciais para construção de códigos.

**Pré-requisitos**

* JDK instalado
* IDE escolhida
* Diretório do projeto definido

**Assuntos**

* Estrutura inicial
* Padrão de nomenclatura
* Declarando variáveis e métodos
* Identação
* Organizando arquivo
* Java Beans

**Anatomia das Classes**

Uma classe bem estruturada não quer guerra com ninguém.

Verifique com a empresa a qual você trabalha como é efeito, pois existem empresas que o código deve ser todo em inglês.

**Criando diretório (projeto) java**

1. Entre no Vscode
2. Use o atalho Ctrl+Shift+P
3. Digite **>java** aparecera Create java Project
4. Clique nele
5. Escolha o diretório
6. Nomei o projeto

**Criando uma Classe**

1. Na pasta src, clique com o botão direito e escolha new file
2. Nomei a classe com CamelCase
3. Todo Classe é .java
4. Ao dar enter o Vscode já cria toda a estrutura

**O que é um método main (principal) dentro da classe**

O método main (principal) é o que executa todo o programa

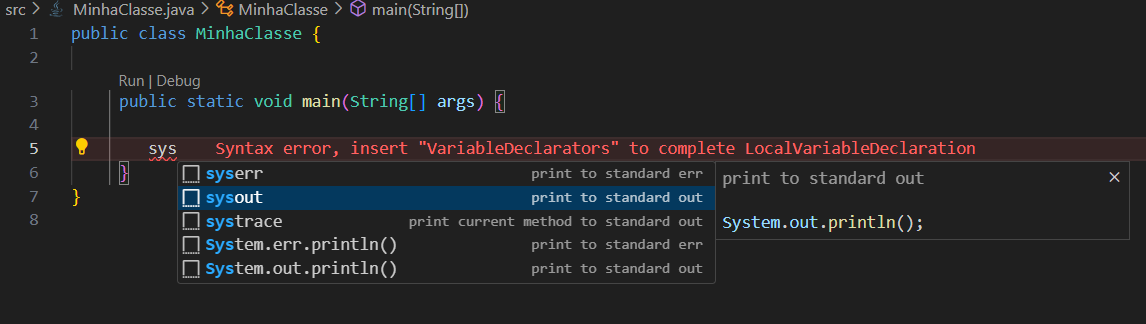
Texto

Descrição gerada automaticamente

**Imprimindo uma mensagem de boas-vindas**

Para isso devemos utilizar uma classe chamada System.out.print();

1. Digite sys e veja que aparece várias opções
2. Escolha a sysout clique ou selecione com as setas e de enter



1. Mensagens de texto serão escritas dentro dos parênteses () e sempre com aspas “” exemplo: (“Olá, mundo! Seja bem-vindo”)

Texto

Descrição gerada automaticamente

**Declaração de métodos, variáveis etc. (convenções)**

* **Arquivo .java =** padrão CamelCase iniciando com **LetraMaiúscula**
* **Nome da classe no arquivo =** A classe deve possuir o nome do arquivo.
* **Nome de variáveis =** padrão CamelCase iniciando com **letraMinúscula.**

**OBS.:** Existe uma exceção sobre a nomenclatura de variável que é do tipo **final.** Neste caso por ser uma variável que não pode ser alterada todas as letras são MAIÚSCULAS e em caso de **nomes compostos** usa-se underline. Veja exemplos:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

**Podemos usar números, caracteres e símbolos nas declarações?**

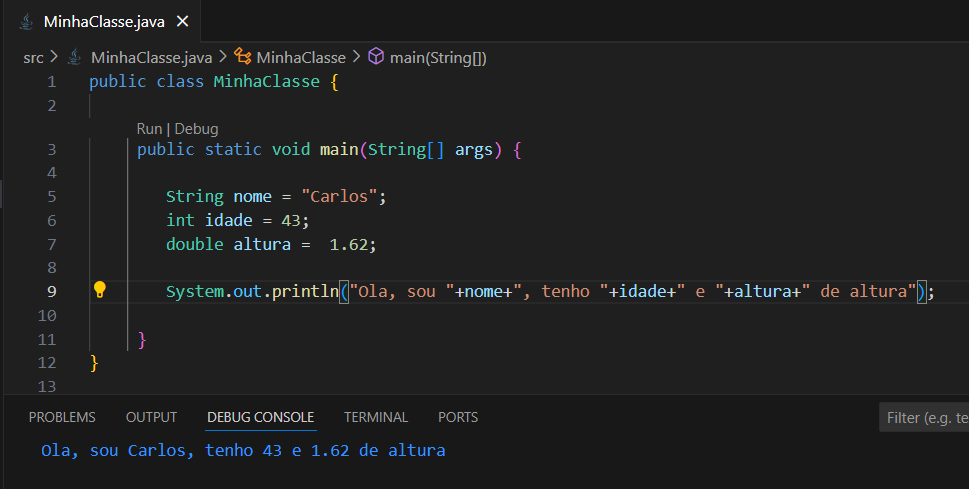
Sim é possível declarar uma varial com , porém com algumas regras

* Deve-se conter apenas letras, \_(underline), $ ou números de 0 a 9
* Deve-se **obrigatoriamente** se iniciar por uma letra (preferencialmente), \_ ou $, **jamais com número**
* Deve-se iniciar com uma **letra minúscula**
* Não pode conter espaços
* Não podemos usar **palavras-chave** da linguagem4
* O nome deve ser único dentro de um escopo

**Declarando variáveis**

Toda variável tem

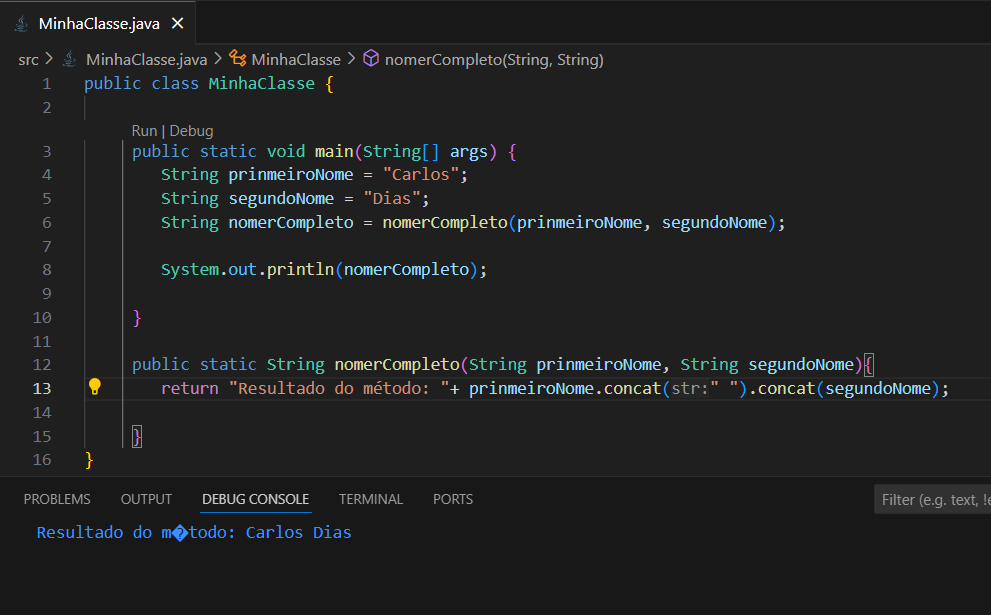
1. Uma tipagem = double, int, boolean
2. NomeBemDefinido
3. Valor (que é opcional em alguns casos)
4. Nome

****

**Declarando métodos**

Todo método tem

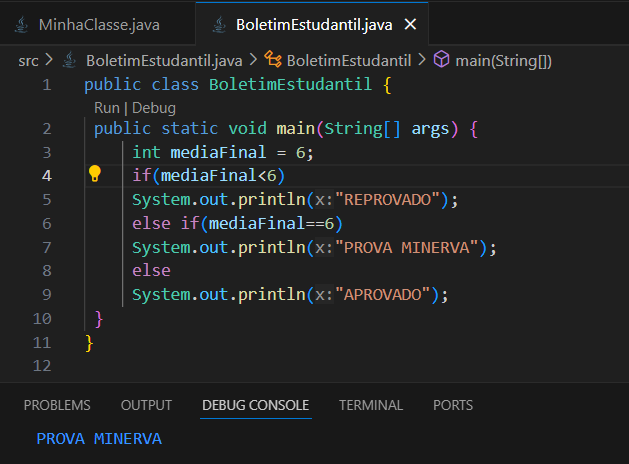
1. Tipo de retorno
2. Nome do objetivo no infinitivo = format**ar**, cadastr**ar**
3. Parâmetro (pode ter vários), porém **é opcional**, ele pode ou não ter



**Identação**

Identar é um termo usado para escrever um código do programa e separá-lo em blocos de código, facilitando assim o entendimento, interpretação, visualização e a localização da escrita. Tabulando o escopo, tudo fica mais bonito. Em Java isso não interfere no algoritmo em outra linguagem pode ser que sim.

**TEXTO SEM IDENTAÇÃO**



O programa funciona, no entanto, fica difícil entender

**TEXTO COM IDENTAÇÃO**

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Pode-se ter vários níveis de **profundidade** **hierarquia de identação**, mas caso isso ocorra revise sua lógica e, de preferência, encaminhe para outros processos.

**Organização dos arquivos (packages)**

À medida que nosso programa evolui, surge vários arquivos (códigos fontes) que devem ser organizados através de pacotes (packages). Existe uma convenção para criar tudo isso.

1. Inicia-se com o modelo que a empresa do projeto desenvolve
2. comercial (com)
3. organizacional (org)
4. opensource (org ou opensource)
5. Seguido do nome da empresa
6. Tipo de projeto = estoque, compras

Exemplo:

**Nome da empresa:** Improvision Comercio LTDA

com.improvision.compras

Sempre separaremos com . (pontos) a estrutura.

**Sugestão:**

Desenvolvas um package com seus arquivos de estudos, por exemplo:

**edu.carlos.tema.java**

edu = seria o modelo de projeto

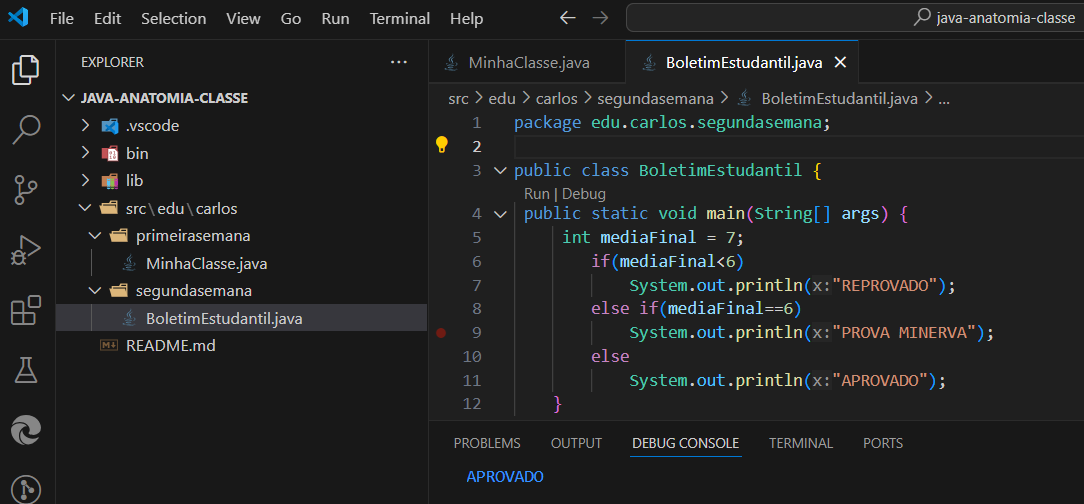
carlos = seria seu nome ou da empresa

tema = tipo do projeto

java = o tipo do estudo

**Criando o packages**

1. Entre no Vscode
2. Use o atalho Ctrl+Shift+P
3. Digite **>java** aparecera Create java Project
4. Clique nele
5. Escolha o diretório
6. Nomei o projeto
7. Clique na pasta src com o botão direito e escolha new folder
8. Nomei com modelo da empresa
9. Clique com o botão direito **nesta pasta**, escolha new folder e nomei-a com o nome da empresa
10. Clique como o botão direito nesta pasta, escolha nem folder e nomei-a com o tipo do projeto

****

**Java Beans**

Iniciativa conceitual da empresa de padronizar o melhor possível, através de convenções, as formas de escrita dos algoritmos tentando facilitar a compreensão de todos (times, equipes e de você mesmo) no futuro.

**Variáveis definições sugeridas pelo Beans**

* Clara, sem abreviações, com sentido (coesa)
* Sempre no singular, exceto em array ou coleção
* Com idioma definido pelo time

**Métodos definições sugeridas pelo Beans**

* Nomeados como verbos no infinitivo (comprar, vender, atualizar, excluir)
* Inicia-se com letra minúscula e padrão camelCase
* Algumas bibliotecas trazem métodos em inglês

**Tipos e variáveis**

Tipos primitivos são armazenados diretamente na pilha de memória (Memory Stack) são eles:

* int
* byte
* short
* long
* float
* double
* boolean
* char

Esses tipos não são considerados objetos, ou sejam não podem ser manipulados como um, mas podem ser modificados dentro do escopo e utilizam a memória raiz para armazenamento.

**Constantes**

São armazenadas em memória e não podem ser alteradas, são representadas pela palavra **final seguida do tipo.**

Por convenção, constantes são sempre escritas em caixa alta.