

Entendendo a estrutura de um código Java

Veja neste artigo, como é a estrutura e anatomia de um código em java. Falaremos um pouco da estrutura básica do código(Arquivo fonte , classes , métodos , método main), assim como a anatomia básica dessas respectivas estruturas.

[Artigos Java](#) Entendendo a estrutura de um código Java

Nesse artigo, tentarei apresentar de forma simples para aqueles que desejam iniciar nessa plataforma , como é a estrutura e anatomia de um código em java. Falaremos um pouco da estrutura básica do java(Arquivo fonte , classes , métodos , método main), assim como a anatomia básica dessas respectivas estruturas.

Arquivo Fonte(Source File) :

Em java, cada classe(class) é colocada em um arquivo source, esses arquivos representam partes de uma aplicação ou toda a aplicação(no caso de programas muito pequenos).

Arquivos source são gerados com a extensão .java e podem ser facilmente criados utilizando o notepad por exemplo. Esses arquivos devem possuir o mesmo nome da classe que possuem.











Nome	Data de modificaç...	Tipo	Tamanho
 Candidato	02/05/2012 23:31	Arquivo JAVA	1 KB
 ControladorEleitor	03/05/2012 11:27	Arquivo JAVA	3 KB
 ControladorMesario	03/05/2012 11:31	Arquivo JAVA	1 KB
 Debug	02/05/2012 23:26	Arquivo JAVA	2 KB
 Eleitor	03/05/2012 00:15	Arquivo JAVA	1 KB
 ListaCandidato	02/05/2012 22:38	Arquivo JAVA	3 KB
 ListaEleitor	03/05/2012 20:02	Arquivo JAVA	5 KB
 Principal	03/05/2012 00:23	Arquivo JAVA	2 KB
 Relatorio	03/05/2012 11:31	Arquivo JAVA	2 KB
 TelaDebug	02/05/2012 23:19	Arquivo JAVA	2 KB
 TelaEleitor	03/05/2012 11:27	Arquivo JAVA	2 KB
 TelaMesario	02/05/2012 22:25	Arquivo JAVA	2 KB
 TelaPrincipal	03/05/2012 00:38	Arquivo JAVA	2 KB

Figura 1: Arquivos Source de um aplicativo que gerencia uma urna eletrônica

Classe(Class)

Em uma classe java são colocados os métodos(methods) ,funções ou procedimentos. Todo o código deve estar em alguma classe, pois quando executamos algum aplicativo java nós estamos, na verdade, executando uma classe.

Diferentemente de um arquivo fonte que só pode conter uma classe, uma classe pode conter vários métodos. Em java a classe deve estar em um Arquivo Fonte(Source File) e deve ir com um par de chaves “{}”, são nessas chaves que serão colocados os métodos. Lembrando que uma classe sempre inicia com letra maiúscula.

```
public class MyClass{  
    // código vai aqui  
}
```

Listagem 1. Exemplo de uma classe contida em um Source File MyClass.java

Métodos(Methods)

Os métodos, funções ou procedimentos são onde declararemos o código das nossas aplicações java.

Assim como classes, os métodos em java devem ser escritos acompanhados de um par de chaves “{}” no final. Lembrando que um método sempre inicia com letra minúscula.

```
public class MyClass{  
    public void meuMetodo(/*argumentos*/){  
    }  
}
```

Listagem 2. Exemplo de um método contido em uma Classe

Agora que vimos a estrutura básica do código em java, vamos analisar melhor o código e tentar entender o que declaramos quando criamos classes ou métodos.

Anatomia de uma Classe

```
public class MyClass{  
}
```

Listagem 3. Anatomia de uma classe em Java

- **public** = Refere-se a visibilidade desta classe. Quando dizemos que uma classe é de visibilidade “public”, estamos dizendo que esta classe poderá ser acessada por outras classes.
- **class** = Mostramos que estamos criando uma classe.
- **MyClass** = Refere-se ao nome da classe que estamos criando. Nesse caso, o nome da minha classe será “MyClass”.
- **{ }** = As chaves indicam até onde certa classe ou método se estende. O código que queremos inserir nesta classe deverá ser escrito dentro do espaço das chaves.

Anatomia de uma Método

```
public class MyClass{  
    public void meuMetodo(/*argumentos*/){  
    }  
}
```

Listagem 4. NOAnatomia de um método em JavaME

- **public** = Do mesmo modo que uma classe, refere-se a visibilidade deste método. Quando dizemos que o método é de visibilidade “public”, estamos dizendo que este método poderá ser acessado por outras classes.
- **void** = Refere-se ao tipo de retorno que esse método terá. Nesse caso, como o tipo de retorno é “void”, ou seja, “vazio”, esse método não retornará valor nenhum.
- **meuMetodo** = Assim como numa classe, refere-se ao nome do método que estamos criando. Nesse caso, o nome do meu método será “meuMetodo”.
- **(/*argumentos*/)** = Refere-se aos argumentos que serão passados para esse método, sendo opcional. Caso não seja necessário passar argumentos, simplesmente deixaríamos os parênteses vazios “()”. De contrário é necessário escrever o tipo da variável a ser passada e um nome para essa variável “(int valor)”.

- `{ }` = As chaves indicam até onde certa classe ou método se estende. O código que queremos inserir nesta classe deverá ser escrito dentro do espaço das chaves.

O Método main

Quando o java virtual machine(JVM) inicia, ele procura na sua classe principal por um método muito específico, chamado de método main.

Uma aplicação java obrigatoriamente deverá possuir pelo menos uma classe e um método main, pois é por esse método main que o JVM começará a executar. Como o método main é padrão para qualquer aplicação java, há algumas regras que devem ser cumpridas para o funcionamento desse método. Por regra, todo método main deverá ser: Público, estático, sem retorno(void), com nome de “main”, e deverá receber como argumento um array do tipo String.

```
public class MyClass{
    public static void main(String[] args){
    }
}
```

Listagem 5. Exemplo de um método main contido em uma Classe

Anatomia do método main

```
public class MyClass{
    public static void main(String[] args){
    }
}
```

Listagem 6. Anatomia do método main

- **public** = Do mesmo modo que um método comum, refere-se a visibilidade deste método. Quando dizemos que o método é de visibilidade “public”, estamos dizendo que este método poderá ser acessado por outras classes.
- **static** = Nos garante que somente haverá uma, e não mais que uma, referência para nosso método main, ou seja, todas as instâncias da classe irão compartilhar a mesma cópia do método main.
- **void** = Assim como um método comum, refere-se ao tipo de retorno que esse método terá. Nesse caso, como o tipo de retorno deve ser “void”, ou seja, “vazio”, esse método não retornará valor nenhum.
- **(String[] args)** = Refere-se aos argumentos que serão passados para esse método, sendo obrigatório no caso do método main
- `{ }` = Assim como um método comum, As chaves indicam até onde certa classe ou método se estende. O código que queremos inserir neste método deverá ser escrito dentro do espaço das chaves.

Fonte: Kathy Sierra, Bert Bates. Head First Java.2nd ed. O’Reilly Media.2005