

ALPOO


Teoria - Java

Javadoc

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

1



Disciplina: ALPOO


Tema: Documentando Sistemas - JavaDoc

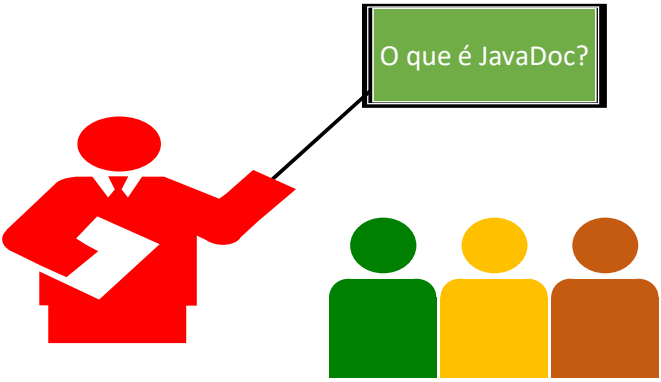
Professor: Ricardo Drudi

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

2






O que é JavaDoc?

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

3



JavaDoc

- **Javadoc** é um programa para gerar documentação.
- O Javadoc lê arquivos fontes em Java, procura por marcações especiais e gera um HTML com a documentação das entidades do código fonte.
- **Input:** códigos fonte em Java (.java)
 - Arquivos individuais ou um diretório com vários fontes
- **Output:** arquivo em HTML com a documentação do código Java
 - Gera arquivos separados por fonte e por *package*

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

4

Comentários



- A **documentação** do código é definida por linhas de comentário.

```
/**
 *
 * Este é um exemplo de comentário em múltiplas linhas.
 * Você pode fazer o comentário do tamanho que quiser,
 * mas para o Javadoc todas essas linhas são apenas um único
 * comentário, e serão agrupadas em um único texto.
 * Entretanto, separar as linhas no código facilita a leitura
 * do próprio código fonte.
 *
 */

/** Este exemplo faz tudo em uma linha só */
```

Localização



- Os comentários devem vir imediatamente antes da classe, interface, construtor, método ou atributo a que se referem. Qualquer código entre eles gera erros na documentação.

```
/**
 * A classe Ligacao armazena os dados dos registros telefônicos
 * da agenda.
 */

import java.util.ArrayList; // isso está errado!

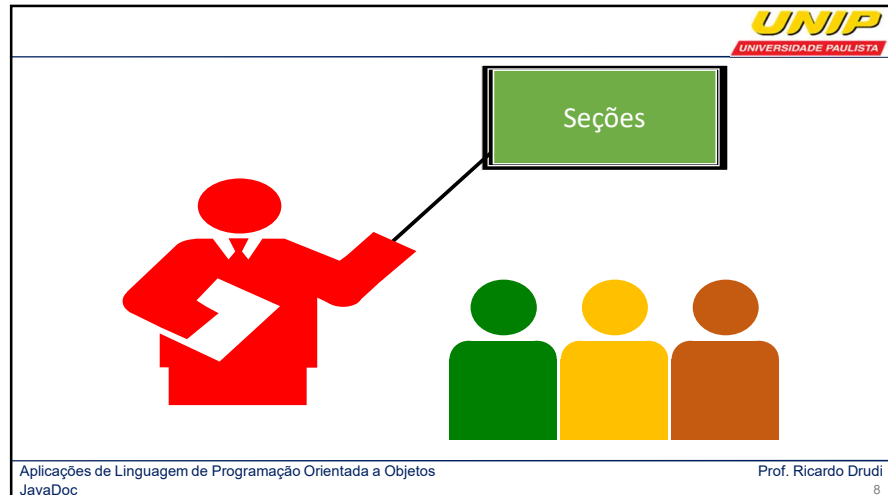
public class Ligacao {
    ...
}
```

Localização



```
import java.util.ArrayList;

/**
 * A classe Ligacao armazena os dados dos registros telefônicos
 * da agenda.
 */
public class Ligacao { // assim está certo!
    /** Número do telefone */
    private String fone;
    /**
     * Esse método retorna o número do telefone da ligação
     */
    public String getFone() {
        return (fone);
    }
}
```



Seções

- A documentação de cada elemento do código fonte é dividida em duas seções:
 - Descritivo:** texto com explicação geral sobre a função do elemento (classe, método, atributo). Deve ser o primeiro parágrafo dos comentários.
 - Tags:** cada elemento do Java têm tags que exercem funções especiais na documentação do sistema.

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

Seções do Javadoc

```

/**
 * Retorna o elemento armazenado na posição específica do array.
 * O índice deve estar entre 0 e <tamanho do array> - 1.
 *
 * @param indice o índice da posição a ser retornada
 * @return o valor armazenado na posição do índice
 * @author Ricardo Drudi
 * @see java.lang.Array
 */
public int getInt(int indice) {
    if (indice < tamanho) {
        return (array[indice]);
    }
}

```

Descrição

Tags

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

Tags

- Block tags:** devem ser utilizadas somente na seção de *tags*. Seguem o formato: **@tag**.
- Inline tags:** podem ser utilizadas em qualquer lugar dos comentários, inclusive na descrição das block tags. Devem aparecer entre chaves: **{@tag}**

```

/**
 * @deprecated Deve ser substituído por {@link #getValor(int)}
 */
public int getInt(int indice) {
    if (indice < tamanho) {
        return (array[indice]);
    }
}

```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

HTML

- Os comentários podem ter código HTML para formatação de seu conteúdo. Como o Javadoc gera um fonte em HTML, todas as tags HTML são válidas nos comentários.

```

/**
 * Retorna o <b>elemento armazenado</b> na posição específica
 * do array. O índice deve estar entre 0 e <tamanho do array> - 1.
 */
public int getValor(int indice) {
    if (indice < tamanho) {
        return (array[indice]);
    }
}

```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

Descritivo

- A primeira sentença de cada comentário do Javadoc deve ser um resumo **descritivo** da entidade que está sendo documentada. O Javadoc utiliza a primeira sentença para identifica-los na listagem dos métodos da classe.
- Você pode declarar mais de uma variável por linha em Java, mas não pode documentar mais de uma variável por linha.

```
/**
 * Valor da primeira coordenada da matriz.
 * Valor da segunda coordenada da matriz.
 */
private int x,y; // isso não funciona!
```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

Tags

Tag	Descrição
@author	Autor da classe / interface / método.
@deprecated	Informa que o método está obsoleto e não deve mais ser utilizado.
@exception @throws	Insere o comentário sobre quais Exceptions podem ser lançadas pelo método, em seção específica no document HTML.
{@link}	Insere um link para um método (pode ser de outro package) ou uma página da web com informações adicionais sobre o tema.
@param	Inclui o parâmetro especificado na seção correspondente na documentação do método. Cada parâmetro deve ser documentado individualmente.
@return	Insere o texto especificado na seção Return da documentação.
@see	Permite que a documentação sugira um texto complementar sobre o tema. Usando geralmente quando um método chama outro método.
@since	Tag utilizada para documentar a versão da aplicação (ou do Java) em que o método foi criado.
@version	Informa a versão da entidade em questão. Deve ser utilizada com cuidado, pois o controle manual de versões é bem complicado.

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

@param

- Permite a explicação dos parâmetros requeridos pelo método.
- Deve haver um @param para cada parâmetro.
- Só é válido para métodos e construtores.

```
/**
 * Tests a character and notifies an observer
 * according to the value of the character
 * @param ch the character to be tested
 * @param obs the observer to be notified
 */
public void testCharacter(char ch, Observer obs)
```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

@return

- Permite a explicação do valor de retorno de um método.
- Só pode haver um @return para cada método.
- Só é válido para métodos.

```
/**
 * Tests a character and notifies an observer
 * according to the value of the character
 * @param ch the character to be tested
 * @param obs the observer to be notified
 * @return result of the action performed by the
 * observer on the character
 */
public int testCharacter(char ch, Observer obs)
```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

@see

- Inclui uma referência “See Also...” na documentação.
- Há 3 variações possíveis:
- @see *string*: Adiciona o texto na documentação.
- @see label: Adiciona um hiperlink na documentação. Segue as regras do HTML.
- @see *package.class#member label*: Adiciona um link para outro método do próprio projeto, ou das APIs Java.
 - Ex: @see String#equals(Object) equals

See Also:
[equals](#)

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

@see

```
@see java.lang.String // String
@see java.lang.String The String class // The String class
@see String // String
@see String#equals(Object) // String.equals(Object)
@see String#equals // String.equals(java.lang.Object)
@see java.lang.Object#wait(long) // java.lang.Object.wait(long)
@see Character#MAX_RADIX // Character.MAX_RADIX
@see <a href="spec.html">Java Spec</a> // Java Spec
@see "The Java Programming Language" // The Java Programming Language
```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

@deprecated

- Informa que o método está obsoleto, e não deve mais ser utilizado.
- A princípio o método deve ser mantido por questões de retrocompatibilidade.

```
/**
 * @deprecated desde jan/2018. Utilize o método
 * {@link #novoMetodo(String,Observer)}
 */
public int testCharacter(char ch, Observer obs)
```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi

@throws

- Cria uma seção **Throws** na documentação gerada pelo Javadoc.
- Sintaxe: `@throws classe-exception descrição`

```
/**
 * Draws the shape of the object on the screen
 * instance given in the parameter.
 * @throws NullPointerException If the screen object is null.
 * @throws ScreenSizeException If the object does
 * not fit on the screen.
 */
public boolean drawShape(Screen screen)
    throws NullPointerException, ScreenSizeException{
    ...
}
```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi
21

Controle de Versão

- **@version**: cria um seção **Version** na documentação. O objetivo dessa *tag* é determinar a versão do sistema da qual o elemento faz parte. Só pode ser utilizada para classes e interfaces. A classe deve ser compilada com a opção `-version`.
- **@since**: cria um seção **Since** na documentação. Indica em que versão da aplicação (ou do Java) o método ou atributo foi introduzido.


```
/**
 * @since 1.4
 */
public int testCharacter(char ch, Observer obs)
```

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi
22

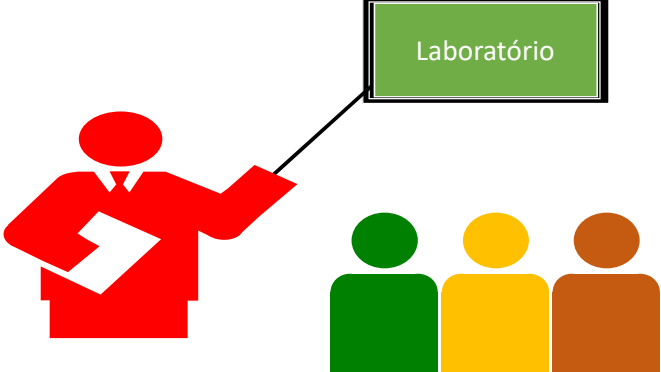
Netbeans

- Para gerar a documentação Javadoc na IDE **Netbeans**, utilize a opção *Run/Generate Javadoc*.
- Serão gerados um HTML por classe, agrupados em pastas por package, e um `index.html` na pasta `/<nomeDoProjeto>/dist/javadoc`.




Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi
23



Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi
24




Exercício

- 1) Documente, utilizando as *tags Javadoc* que julgar importantes, as classe, os atributos e os métodos do SSS, inclusive os métodos das classes utilitárias.
- 2) Crie um novo método na classe `Principal` que utilize os métodos da classe que você documentou, e veja que o *Netbeans* apresenta automaticamente um *help* com a documentação *Javadoc* que você criou.
- 3) Gere a documentação HTML do *Javadoc*, procure e abra o `index.html` do projeto, e percorra as classes que você documentou.

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi
25



Muito obrigado a todos.

Copyright © 2018 Prof. Ricardo Drudi

Todos direitos reservados. A reprodução ou divulgação total ou parcial deste documento é expressamente proibida sem o consentimento formal, por escrito, do professor Ricardo Drudi.

Aplicações de Linguagem de Programação Orientada a Objetos
JavaDoc

Prof. Ricardo Drudi
26