Documentação

Documentação

- Fundamental para
  - Manter e melhorar o software existente
  - Permitir gestão de equipas de desenvolvimento
  - Partilhar código
  - Grandes projectos
  - Grandes equipas de desenvolvimento
  - Desenvolvimento distribuído

Javadoc

- Formato de anotações (e ferramentas associadas) que permite a geração automática de documentação.
- Gera páginas html que descrevem as classes, interfaces, construtores, métodos, atributos, etc.
- Comentários Javadoc começam com /** em vez de /*
- Aparecem sempre imediatamente antes do elemento que documentam (uma classe, um método, um atributo, etc.)
- Mais informações:
  - http://www.oracle.com/technetwork/java/javase/documentation/javadoc-137458.html
  - Em particular o guia de estilo em: http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html
- Há outras ferramentas para gerar documentação, com sintaxes ligeiramente diferentes, por exemplo Doxygen (http://www.doxygen.org)

Principais anotações suportadas (Javadoc)

- @author: Nome do autor da classe
- @param: Descreve um parâmetro de um método
- @return: Descreve o tipo de valor devolvido por um método
- @throws: Descreve a excepção que pode ser lançada por um método
- @exception: Descreve uma excepção
- Para gerar documentação para uma classe em Eclipse seleccione Export / Java / Javadoc / Next ou Project/Generate Javadoc (os comentários no código serão gravados em HTML). Pode ser necessário indicar onde se encontra a aplicação. e.g. .../jdk1.6.0_04/bin/javadoc.exe)

Exemplo

```
/**
   * Inventory contains the ammount of
   coffee stores
   * @author John Smith
*/
public class Inventory {
/**
 * Coffee inventory
 */
private int ammountOfCoffe;
```

Exemplo

Class Inventory

```
java.lang.Object
  └ Inventory
```

```
public class Inventory
extends java.lang.Object
```

Inventory contians the ammount of coffe stores

Author:
John Smith

Constructor Summary

Inventory()   Construtor, default initilization 15 units of coffe

Method Summary

int	**quantidadeDeCafe**() Returns the existing ammount of coffe

Methods inherited from class java.lang.Object
equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Exemplo

```
/**
 * Construtor, default initilization 15 units of coffe
 */
   public Inventory() {  ammountOfCoffee = 15; }
/**
 * Returns the existing ammount of coffee
 * @return int
*/
  public int quantidadeDeCafe() {
    return ammountOfCoffee;
  }
}
```

Exemplo

Constructor Detail

Inventory

```
public Inventory()
```

Construtor, default initilization 15 units of coffe

Method Detail

quantidadeDeCafe

```
public int quantidadeDeCafe()
```

Returns the existing ammount of coffe

Exemplo: getImage(…)

```
/**
 * Returns an Image object that can then be painted on the screen.
 * The url argument must specify an absolute {@link URL}.
 * The name argument is a specifier that is relative to the url argument.
 * This method always returns immediately, whether or not the  image
 * exists. When this applet attempts to draw the image on
 * the screen, the data will be loaded. The graphics primitives that draw
 * the image will incrementally paint on the screen.

 * @param  url  an absolute URL giving the base location of the image
 * @param  name the location of the image, relative to the url argument
 * @return      the image at the specified URL
 * @see         Image

 */

public Image getImage(URL url, String name)
```

Exemplo: getImage(…)

getImage

`public Image getImage(URL url,`

`String name)`

Returns an `Image` object that can then be painted on the screen. The `url` argument must specify an absolute URL. The `name` argument is a specifier that is relative to the `url` argument.

This method always returns immediately, whether or not the image exists. When this applet attempts to draw the image on the screen, the data will be loaded. The graphics primitives that draw the image will incrementally paint on the screen.

Parameters:

`url` - an absolute URL giving the base location of the image.

`name` - the location of the image, relative to the `url` argument.

Returns:

the image at the specified URL.

See Also:

`Image`

Exemplo: Java.util.LinkedList

- Source code: http://developer.classpath.org/doc/java/util/LinkedList-source.html

- API: http://docs.oracle.com/javase/6/docs/api/java/util/LinkedList.html

Convenções para o JavaDoc

- Todos os comentários devem ser feitos em inglês
- Devem ser claros e corretamente inseridos pois muitas vezes são lidos dentro do código

```
/**
 * Return lateral location of the specified position.
 * If the position is unset, NaN is returned.
 * @param x    X coordinate of position.
 * @param y    Y coordinate of position.
 * @param zone Zone of position.
 * @return     Lateral location.
 * @throws IllegalArgumentException  If zone is <= 0.
 */
public double computeLocation(double x, double y, int zone) throws
IllegalArgumentException { ...}
```

Java Code Conventions

- **Java Programming Style Guidelines:** http://geosoft.no/development/javastyle.html
- **Google Java Style:** http://google-styleguide.googlecode.com/svn/trunk/javaguide.html#s7.1-javadoc-formatting

Mais informação / Referências

Y. Daniel Liang, *Introduction to Java Programming*, 7.ª edição, Prentice-Hall, 2008.

Sumário

- Documentação
- Javadoc