JAVADOC: gera a documentação do código.

O preview da documentação aparece na área do console como o botão Javadoc.

Para incluir um comentário na documentação oficial utilizamos uma /** e fechamos com */ ou /** + enter.

Para gerar a documentação, no cabeçalho do Eclipse, clicamos em "Project -> Generate Javadoc".

Ao darmos o "Ok", serão percorridas todas as classes e será criada uma página HTML que contém todas as informações do projeto. Foi criada uma pasta doc, e vários arquivos a partir dessa varredura de classes, mas o que nos interessa neste momento é o index.

Normalmente os comentários especiais possuem algum valor importante dentro do código. Por exemplo, especificar a função de alguma classe ou fornecer informações de valor acerca de algum elemento, como faz a classe Exception.

O primeiro passo para repassar um projeto é justamente gerar essa documentação. O Java iniciou a proposta de incluir a documentação dentro do código, como vimos nos comentários especiais. Se você realizar alguma modificação, as atualizações na documentação podem ser realizadas facilmente.

- ❖ Algumas tags do javadoc:
  - ➢ @author (usado na classe ou interface)
  - ➢ @version (usado na classe ou interface)
  - ➢ @param (usado no método e construtor)
  - ➢ @return (usado apenas no método)
  - ➢ @exception ou @throws (no método ou construtor)
  - ➢ @see
  - ➢ @since
  - ➢ @serial

➢ @deprecated

Diferentemente das anotações (ex: @Override) que são interpretadas pelo compilador e são muito mais sofisticadas e poderosas. Elas só entraram na plataforma Java a partir da versão 1.5 enquanto o javadoc está presente desde o nascimento da plataforma Java. O interessante é que as anotações foram inspiradas pelas tags do javadoc.

JAR (**J**ava **AR**chive): É o formato padrão do mundo Java para distribuir código compilado. Nada mais do que um arquivo ZIP, mas como a extensão .jar

❖ Para exportar um projeto:

Exportar -> jar file. Não exportamos os arquivos internos do Eclipse (.classpath e .project), e sim, todo o conteúdo da pasta src. Lembrando que não estamos exportando o código fonte, apenas o código compilado, que o Eclipse denomina "class files".

❖ Para usar o arquivo JAR:

Criar um projeto -> clicar em cima do projeto e criar um folder com o nome libs (pasta que geralmente armazena as bibliotecas) -> Arrastar o arquivo .jar para dentro desse folder. Para que as classes se tornem visíveis e usuais, adicionar essa biblioteca ao path, pressionamos o botão direito do mouse, e selecionamos as opções "Build Path > Add to Build Path". Assim feito, veremos que uma representação gráfica de

jarra surge ao lado do nome do arquivo, apontado como Referenced Libraries, e dentro dele são exibidos todos os pacotes, que por sua vez, armazenam as classes.

Para usar as classes desse arquivo, importá-las (sugestão do eclipse) para dentro do arquivo que desejar.

Java é uma plataforma de desenvolvimento completa que se destaca com sua grande quantidade de projetos open source. Para a maioria dos problemas no dia a dia do desenvolvedor já existem bibliotecas para resolver.

Aí existe a necessidade de organizar, centralizar e versionar os JARs dessa biblioteca e gerenciar as dependências entre elas. Para resolver isso, foram criadas ferramentas específicas e no mundo Java se destacou o Maven. O Maven organiza os JARs (código compilado, código fonte e documentação) em um repositório central que é público e pode ser pesquisado: https://mvnrepository.com/.

Lá você pode ver e até baixar os JARs, mas o melhor é que a ferramenta Maven pode fazer isso para você.