

Javadoc. Основание для ведения документации

- Возобновление работы над проектом после продолжительного перерыва
- Переход проекта от одного человека (группы) к другому человеку (группе)
- Опубликование проекта для Open Source сообщества
- Совместная работа большой группы людей над одним проектом

Javadoc. Требования к документам

- Не документировать очевидные вещи (setter'ы и getter'ы, циклы по массивам и листам, вывод логов и прочее)

```
package java.se._01.javadoc;
public class DocRequirement {
    /**      Проверка: редактируема ли данная ячейка.
     *
     *      <p>В случае если данная ячейка редактируема -
возвращается true</p>
     *
     *      <p>В случае если данная ячейка не редактируема -
возвращается false</p>
     *
     *      @param column номер колонки для проверки
     *      @return результат проверки
     */
    public boolean isCellEditable(int column) {
        return column % 2 == 0 ? true : false;
    }
}
```

Javadoc. Требования к документам

- Поддерживать документацию в актуальном состоянии

```
package java.se._01.javadoc;
public class Parsing {
    /**
     * Произвести парсинг истории операций над невстроенной БД.
     * @throws XMLConfigurationParsingException
     */
    public void parseHistoryNotEmbeddedDB() throws XMLConfigurationParsingException {
        return;
    }
    /**
     *
     * Thread.currentThread().getContextClassLoader().
     * getResourceAsStream("ru/magnetosoft/magnet/em/cfg/db-configuration-not-
embedded.xml");
     *
     * String configXml = readStringFromStream(is);
     *
     * XmlConfigurationParserImpl parser = new
     * XmlConfigurationParserImpl(configXml); IEmConfiguration res =
     * parser.parse(); assertNotNull(res);
     *
     * assertFalse(res.getOperationHistoryStorageConfiguration().isEmbedded());
     *
     * assertEquals("HSQLDB",
     * res.getOperationHistoryStorageConfiguration().getStorageDBType());
     */
}
}
```

Javadoc. Требования к документам

- Описывать входящие параметры, если нужно

```
package java.se._01.javadoc;
public class EnterParamsDoc {
    /**      Создание нового экземпляра ядра.
     *
     *      @param contextName
     *      @param objectRelationManager
     *      @param xmlObjectPersister
     *      @param ohm
     *      @param snm
     *      @param initializationLatch
     *      @return
     */
    public static EmEngine newInstance(String contextName,
        IXmlObjectRelationManager objectRelationManager,
        IXmlObjectPersister xmlObjectPersister,
        OperationHistoryManager ohm,
        ISearchNotificationManager snm,
        CountDownLatch initializationLatch) {
        ...
    }
}
```

Javadoc. Синтаксис javadoc-комментария

- Обыкновенный комментарий

```
/* Calculates the factorial */
```

```
int factorial(int x) {
```

```
...
```

- Javadoc-комментарий (он может включать в себя HTML тэги и специальные javadoc тэги, которые позволяют включать дополнительную информацию и ссылки)

```
/** Calculates the factorial */
```

```
public double factorial(int x) {
```

Javadoc. Структура javadoc-комментария

Структура каждого javadoc-комментария такова:

- первая строка, которая попадает в краткое описание класса (отделяется точкой и пустой строкой);
- основной текст, который вместе с HTML тэгами копируется в основную документацию;
- входящие параметры (если есть);
- выбрасываемые исключения (если есть);
- возвращаемое значение (если есть);
- служебные javadoc-тэги.

Javadoc. Структура javadoc-комментария

```
package java.se._01.javadoc;

import java.se._01.javadoc.exception.EntityManagerException;
import java.se._01.javadoc.exception.XmlMagnetException;

public class WriterDocExample {
    /**
     * Произвести запись нового объекта.
     *
     * Произвести запись нового объекта. Тип для сохранения может быть
     * подклассом List (для реализации возможности работы с несколькими
     * объектами) или единичным объектом. В случае если произошла какая-либо
     * ошибка - выбрасывается исключение. В данном случае с базой не происходит
     * никаких изменений и ни один объект не затрагивается пред
     * операцией. Конкретный тип ошибки можно определить проверкой
     * возвращенного исключения.
     * @param object
     *      сохраняемый объект/объекты,
     * @return сохраняемый объект/объекты
     * @throws XmlMagnetException
     * @throws EntityManagerException
     */
    public Object insert(Object object) throws XmlMagnetException,
    EntityManagerException{
        return new Object();
    }
}
```

Method Summary

java.lang.Object	insert (java.lang.Object object)	Произвести запись нового объекта.
------------------	--	-----------------------------------

Method Detail

insert

```
public java.lang.Object insert(java.lang.Object object)
    throws java.se._01.javadoc.exception.XmlMagnetException,
    java.se._01.javadoc.exception.EntityManagerException
```

Произвести запись нового объекта. Произвести запись нового объекта. Тип для сохранения может быть подклассом List (для реализации возможности работы с несколькими объектами) или единичным объектом. В случае если произошла какая-либо ошибка - выбрасывается исключение. В данном случае с базой не происходит никаких изменений и ни один объект не затрагивается пред операцией. Конкретный тип ошибки можно определить проверкой возвращенного исключения.

Parameters:

object - сохраняемый объект/объекты.

Returns:

сохраненный объект/объекты

Throws:

java.se._01.javadoc.exception.XmlMagnetException
java.se._01.javadoc.exception.EntityManagerException

Javadoc. Типы тегов

- **Блочные теги**

- Начинается с `@tag` и оканчивается с началом следующего тега
- Пример

`@param x a value`

- **Строчные теги**

- Ограничены фигурными скобками
- Могут встречаться в теле других тегов
- Пример

Use a {@link java.lang.Math#log} for positive numbers.

Javadoc. Тег @param

- Описывает параметров методов и конструкторов
- Синтаксис

`@param <имя параметра> <описание>`

- Пример

`@param x a value`

Javadoc. Тер @return

- Описывает возвращаемое значение метода
- Синтаксис

`@return <описание>`

- Пример

`@return the factorial of <code>x</code>`

Javadoc. Тер @throws

- Описывает исключения, генерируемые методом/конструктором

- Синтаксис

`@throws <класс исключения> <описание>`

- Пример

`@throws IllegalArgumentException if <code>x</code> is less than zero`

Javadoc. Тэг @see

- Ссылка на дополнительную информацию
- Синтаксис
 - @see <имя класса>
 - @see [<имя класса>]#<имя члена>
 - @see "<Текст ссылки>"
- Примеры
 - @see Math#log10
 - @see "The Java Programming language Specifiecation, p. 142"

Javadoc. Тэг @version

- Текущая версия класса/пакета
- Синтаксис

`@version <описание версии>`

- Пример

`@version 5.0`

Javadoc. Тег @since

- Версия в которой была добавлена описываемая сущность
- Синтаксис

@since <описание версии>

- Пример

@since 5.0

Javadoc. Тэг @deprecated

- Помечает возможности, которые не следует использовать
- Синтаксис

`@deprecated <комментарий>`

- Пример

`@deprecated replaced by {@link #setVisible}`

Javadoc. Тэг @author

- Описывает автора класса/пакета
- Синтаксис

@author <имя автора>

- Пример

@author Josh Bloch

@author Neal Gafter

Javadoc. Тэг {@link}

- Ссылка на другую сущность
- Синтаксис

`{@link <класс>#<член> <текст>}`

- Примеры

`{@link java.lang.Math#Log10 Decimal Logarithm}`

`{@link Math}`

`{@link Math#Log10}`

`{@link #factorial() calculates factorial}`

Javadoc. Тэг {@docRoot}

- Заменяется на ссылку на корень документации
- Синтаксис

`{@docRoot}`

- Пример

`Copyright`

Javadoc. Тэг {@value}

- Заменяется на значение поля
- Синтаксис

`{@value <имя класса>#<имя поля>}`

- Пример

Default value is {@value #DEFAULT_TIME}

Javadoc. Тэг {@code}

- Предназначен для вставки фрагментов кода
- Внутри тэга **HTML** не распознается
- Синтаксис

{@code <код>}

- Пример

Is equivalent of **{@code Math.max(a, b)}**.

Javadoc. Описание пакета

- Есть возможность применять комментарии для пакетов. Для этого необходимо поместить файл `package.html` в пакет с исходными текстами.
- Данный файл должен быть обычным HTML-файлом с тегом `<body>`.
 - Первая строка файла до точки идет в краткое описание пакета, а полное идет вниз – под список всех классов и исключений.

Этот функционал позволяет описать что-то, что невозможно описать с помощью конкретных классов.

Javadoc. Применение тегов

Пакеты	Классы	Методы и конструкторы	Поля
@see @since {@link} {@docRoot}			
	@deprecated		
@author @version		@param @return @throws	{@value}

Javadoc. Наследование Javadoc

- Если какая-то часть информации о методе не указана, то описание копируется у ближайшего предка.
- Копируемая информация:
 - Описание
 - @param
 - @returns
 - @throws

Javadoc. Компиляция Javadoc

- Инструмент

Javadoc

- Применение

javadoc <опции> <список пакетов> <список файлов>

- Пример

javadoc JavadocExample1.java

Javadoc. Основные опции Javadoc

-sourcepath <path>	Местоположения исходных файлов
-classpath <path>	Местоположение используемых классов
-d <dir>	Каталог для документации
-public	Подробность информации
-protected	
-package	
-private	
-version	Информация о версии
-author	Информация об авторе

```

package java.se._01.javadoc;
import java.se._01.javadoc.exception.EntityManagerException;
import java.se._01.javadoc.exception.XmlMagnetException;
/**
 * Представитель модуля EntityManger на клиентской стороне.
 *
 * <p>
 *
 * Данный класс представляет средства доступ к возможностям модуля
 * EntityManager, минуя прямые вызовы веб-сервисов.
 *
 * </p>
 *
 * <p>
 *
 * Он самостоятельно преобразовывает ваши Java Беап'ы в XML и производит
 * обратную операцию, при получении ответа от модуля.
 *
 * </p>
 *
 * <p>
 *
 * Для получения экземпляра данного класса предназначены статические методы
 * {@link #getInstance(InputStream)} и {@link #getInstance(String)}
 *
 * </p>
 *
 * Screated 09.11.2006
 * (Aversion $Revision 738 $
 * @author MalyshkinF
 * @since 0.2.2
 */
public class EntityManagerInvoker {

```

}

Javadoc

The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays a project structure with packages like `kz.enu.omarbekova.exception`, `kz.enu.omarbekova.logic`, `kz.enu.omarbekova.main`, `kz.enu.omarbekova.matrix`, `kz.enu.omarbekova.util`, and `MatrixFactory`. The main editor shows the `Runner.java` file with the following code:

```
package kz.enu.omarbekova.main;

import kz.enu.omarbekova.exception.MultipleException;
import kz.enu.omarbekova.logic.Multiplicator;
import kz.enu.omarbekova.matrix.Matrix;
import kz.enu.omarbekova.util.MatrixFactory;

/**
 * @author omarbekova
 * @version 1.0
 */
public class Runner {

    public static void main(String[] args) {
        int n = 2, m = 3, l = 4;
        Matrix p = MatrixFactory.createRandomized(n, m);
        Matrix q = MatrixFactory.createRandomized(m, l);
        System.out.println("Matrix first is: " + p);
        System.out.println("Matrix second is: " + q);
    }
}
```

A context menu is open over the code, with the `Generate Javadoc...` option selected. The Console window at the bottom shows the output of the program:

```
<terminated> Runner (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (02.11.2013 20:51:50)
Matrix first is:
Matrix : 2x3
0 1 4
1 6 3

Matrix second is:
Matrix : 3x4
8 6 3 15
15 10 11 0
5 14 11 3

Matrix product is:
Matrix : 2x4
```

A status bar message at the bottom left states: "The word 'omarbekova' is not correctly spelled". The bottom of the screen shows the Windows taskbar with various open applications and the system clock displaying 21:06.

Javadoc

All Classes

[EntityManagerInvoker](#)



[Package](#) [Class](#) [Use](#) [Tree](#) [Deprecated](#) [Index](#) [Help](#)

[PREV PACKAGE](#) [NEXT PACKAGE](#)

[FRAMES](#) [NO FRAMES](#)

Package java.se._01.javadoc

Author:
Ivanov

Class Summary

[EntityManagerInvoker](#) Предс

[Package](#) [Class](#) [Use](#) [Tree](#) [Index](#) [Help](#)

[PREV PACKAGE](#) [NEXT PACKAGE](#)

Constructor Summary

[EntityManagerInvoker](#) ()

Method Summary

java.lang.Object	insert (java.lang.Object object) Произвести запись нового объекта.
------------------	---

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

EntityManagerInvoker

public EntityManagerInvoker ()

Method Detail

insert

```
public java.lang.Object insert(java.lang.Object object)
    throws java.se._01.javadoc.exception.XmlMagnetException,
           java.se._01.javadoc.exception.EntityManagerException
```

Произвести запись нового объекта. Произвести запись нового объекта. Тип для сохранения может быть подклассом List (для реализации возможности работы с несколькими объектами) или единичным объектом. В случае если произошла какая-либо ошибка - выбрасывается исключение. В данном случае с базой не происходит никаких изменений и ни один объект не был затрагивается предполагаемой операцией. Конкретный тип ошибки можно определить проверкой конкретного возвращённого