На этой странице я объясню, как писать комментарии JavaDoc для DelphiCodeToDoc в среде Delphi. Вы можете прочитать интересную статью от Sun: «Как написать комментарии к документации для инструмента Javadoc».

Ниже приведен лишь краткий обзор этого документа. Полный список тегов см. в конце данного документа.

Формат комментария к документации
Комментарий к документации записывается на HTML и должен предшествовать объявлению класса, поля, конструктора или метода. Он состоит из двух частей – описания, за которым следуют блочные теги. В этом примере есть два блочных тега: @param и @see.

Пример

```
{*-----------------------------------------------------------------------------
  Настроить вывод сообщений для пользователя и отладку, чтобы информировать о
статусе
  Класс TDOCProject полностью отделён от внешнего пользовательского интерфейса.
  Однако, чтобы сообщать о состоянии проекта, ему нужно знать, куда выводить
информацию.
  Это может быть визуальный элемент управления, файл или любой другой объект,
  содержащий ссылку на TStrings для работы.

  @param dmUser Куда будут записываться сообщения для пользователя
  @param dmDebug Куда будут записываться отладочные сообщения
  @see   WriteMessage
-----------------------------------------------------------------------------*}
procedure TDOCProject.SetDestinationMessages(const dmUser, dmDebug: TStrings);
begin
  FdmUserMessages := dmUser;
  FdmDebugMessages := dmDebug;
end;
```

Примечание
*Результатирующий HTML-код после выполнения DelphiCodeToDoc показан ниже.
*Каждая строка выше выровнена по коду под комментарием (блок begin/end).
Первая строка содержит начальную метку комментария "{" вместе с префиксом JavaDoc " и украшением, которое является необязательным.
*Напишите первое предложение как краткое резюме метода, так как DelphiCodeToDoc автоматически помещает его в таблицу сводки методов (и индекс).
*Вставьте пустую строку между описанием и списком тегов, как показано.
*Первая строка, начинающаяся с символа "@", завершает описание. В одном комментарии к документации может быть только один блок описания; вы не можете продолжить описание после блочных тегов.
*Последняя строка содержит конечную метку комментария "}" без украшения.
*Строки не переносятся, ограничивайте любые строки комментариев к документации до 80 символов.

Тег JavaDoc - Описание

Описание

*Первое предложение каждого комментария к документации должно быть предложением-саммари, содержащим лаконичное, но полное описание элемента API. Это означает первое предложение каждого описания члена, класса, интерфейса или пакета. Инструмент Javadoc копирует это первое предложение в соответствующую сводку членов, классов/интерфейсов или пакетов. Это делает важным написание четких и содержательных начальных предложений, которые могут стоять сами по себе.

*Остальная часть описания представляет собой пояснение и выходит за рамки саммари. Лучшие имена членов являются "самодокументирующимися", то есть они рассказывают вам в основном о том, что делает этот элемент. Если комментарий к документации просто повторяет имя члена в форме предложения, он не предоставляет дополнительной информации. Например, если описание метода использует только те слова, которые появляются в имени метода, тогда оно ничего не добавляет к тому, что можно было бы предположить. Идеальный комментарий идет дальше этих слов и всегда должен вознаграждать вас какой-то информацией, которая сразу не была очевидной из названия.

```
{*-----------------------------------------------------------------------------
  Обработка файлов BDF

  BDFfont - это библиотека для обработки шрифтов формата BDF (формата
распределения битовых карт)
```

```
  с использованием Delphi.
  Эта библиотека устранит недостаток поддержки растровых шрифтов в MS-Windows.
  Также доступна полная документация и несколько примеров.
  Получите последнюю версию по адресу: http://sourceforge.net/projects/bdffont/
  Авторские права TridenT 2003 года - Лицензия GNU GPL

  @Author TridenT
  @Version 2003/08/16 TridenT v0.1 Начальная версия
  @Version 2003/08/18 TridenT v0.2 Добавлена прозрачность и поддержка пробела
  @Version 2003/08/24 TridenT v0.3 Добавлено сообщение при загрузке шрифта
  @Version 2003/09/02 TridenT v0.4 Новый метод хранения (быстрый и компактный)
  @Version 2003/09/10 TridenT v0.5 Добавлена функция автоматического подбора
размера текста
-------------------------------------------------------------------------------}
unit u_BDFfont;
...
```

==========

Тег JavaDoc - @Author

@Author имя-текст
Добавляет запись "Автор" с указанным именем-текстом в сгенерированную документацию, когда используется опция -author. Комментарий к документации может содержать несколько тегов @author. Вы можете указать одно имя на каждый тег @author или несколько имен на один тег. Весь текст просто копируется в сгенерированный документ без анализа. Поэтому, если хотите, вы можете использовать несколько имен в одной строке, чтобы получить другой локализированный разделитель имен вместо запятой.

```
{*-------------------------------------------------------------------------------
  Базовый класс всех генераторов
  Компонент генератора создает форматированный файл с данными из
  экстрактора. Все генераторы будут наследоваться от TDOCGenerator,
  описанного в данной единице.

  http://dephicodetodoc.sourceforge.net/
  Авторские права TridenT 2003 года - Лицензия GNU GPL

  @Author TridenT
  @Version 2003/12/01 TridenT v0.0 Начальная версия
  @Todo Добавить комментарии ToDo (опционально)
-------------------------------------------------------------------------------}
unit DocGen;
...
```

==========

Тег JavaDoc - @version

@version текст-версии
Добавляет подзаголовок "Версия" с указанным текстом версии в сгенерированные документы, когда используется опция -version. Этот тег предназначен для хранения текущего номера версии программного обеспечения, частью которого является данный код (в отличие от @since, который хранит номер версии, начиная с которой был введен этот код). У текст-версии нет специальной внутренней структуры. Чтобы увидеть, где можно использовать тег версии, см. Где могут использоваться теги.
Комментарий к документации может содержать несколько тегов @version. Если имеет смысл, вы можете указать один номер версии на каждый тег @version или несколько номеров версий на один тег. В первом случае инструмент Javadoc вставляет запятую (,) и пробел между

именами. Во втором случае весь текст просто копируется в сгенерированный документ без анализа. Поэтому, если хотите, вы можете использовать несколько имен в одной строке, чтобы получить другой локализованный разделитель имен вместо запятой.

==========

Тег JavaDoc - @param

@param имя-параметра описание
Добавляет параметр в раздел "Параметры". Описание может продолжаться на следующей строке. Этот тег допустим только в комментарии к документации для метода или конструктора.
За тегом @param следует имя (не тип данных) параметра, за которым следует описание параметра. По соглашению, первым существительным в описании является тип данных параметра. (Артикли вроде "a", "an" и "the" могут предшествовать существительному.) Исключение делается для примитивного типа int, где обычно опускается тип данных. Дополнительные пробелы могут быть вставлены между именем и описанием, чтобы описания выравнивались в блоке. Дефисы или другие знаки препинания не должны вставляться перед описанием, поскольку инструмент Javadoc вставляет одну черту.
Имена параметров по соглашению записываются строчными буквами. Тип данных начинается со строчной буквы, чтобы обозначить объект, а не класс. Описание начинается со строчной буквы, если это фраза (не содержит глагола), или с заглавной буквы, если это предложение. Завершайте фразу точкой только в том случае, если за ней следует другая фраза или предложение.

```
{*----------------------------------------------------------------------------
  Настроить вывод сообщений для пользователя и отладку, чтобы информировать о
статусе
  Класс TDOCProject полностью отделён от внешнего пользовательского интерфейса.
  Однако, чтобы сообщать о состоянии проекта, ему нужно знать, куда выводить
информацию.
  Это может быть визуальный элемент управления, файл или любой другой объект,
  содержащий ссылку на TStrings для работы.

  @param dmUser  Куда будут записываться сообщения для пользователя
  @param dmDebug Куда будут записываться отладочные сообщения
----------------------------------------------------------------------------}
procedure TDOCProject.SetDestinationMessages(const dmUser, dmDebug: TStrings);
```

==========

Тег JavaDoc - @return

@return описание
Добавляет раздел "Возвращаемое значение" с текстом описания. Этот текст должен описывать возвращаемый тип и допустимый диапазон значений. Этот тег допустим только в комментарии к документации для метода.
Не включайте @return для процедур и конструкторов; включайте его для всех остальных методов, даже если его содержимое полностью дублирует описание метода. Явная метка @return облегчает кому-либо быстрое нахождение значения возврата. Всякий раз, когда возможно, предоставляйте возвращаемые значения для особых случаев (например, указывая значение, возвращаемое при передаче аргумента вне диапазона).

```
{*----------------------------------------------------------------------------
  Построение проекта путем создания выходных файлов
  Проверяется готовность проекта к построению, затем вызывается генератор
  методом execute.
  Дополнительно состояние проекта устанавливается в psBuilt в случае успеха
```

```
  @return TRUE, если успешно, иначе FALSE
-------------------------------------------------------------------------------}
function TDOCProject.Build: Boolean;
...
```

==========

Тег JavaDoc - @throws

@throws имя-класса описание
Добавляет подзаголовок "Исключения" в сгенерированную документацию с именем класса и текстом описания. Имя класса — это название исключения, которое может быть выброшено методом. Этот тег допустим только в комментарии к документации для метода или конструктора. Если этот класс не полностью указан, инструмент Javadoc использует порядок поиска для поиска этого класса. В одном комментарии к документации можно использовать несколько тегов @throws для одного и того же или разных исключений.
Не функционально: Чтобы убедиться, что все проверяемые исключения задокументированы, если тег @throws не существует для исключения в предложении throws, инструмент Javadoc автоматически добавляет это исключение в выходной HTML-файл (без описания), как если бы оно было задокументировано с помощью тега @throws.
Документация @throws копируется из переопределенного метода в подкласс только тогда, когда исключение явно объявлено в переопределенном методе. То же самое относится и к копированию из метода интерфейса в реализующий метод.

```
{*---------------
Тест процедуры класса!
@throws IExcept Неполная строка!!!
---------------*}
procedure TNewClass.ClassProc;
begin
  ...
  raise Exception.Create('Неполная строка: ');
end;
```

==========

Тег JavaDoc - @todo

@todo текст
Добавляет подзаголовок "Задачи" с указанным текстом в сгенерированную документацию. Используйте этот тег таким же образом, как функцию TODO в Delphi.

```
{*-------------------------------------------------------------------------------
  Обработка файлов BDF

  BDFfont - это библиотека для обработки шрифтов формата BDF (формата
распределения битовых карт)
  с использованием Delphi.
  Эта библиотека устранит недостаток поддержки растровых шрифтов в MS-Windows.
  Также доступна полная документация и несколько примеров.
  Получите последнюю версию по адресу: http://sourceforge.net/projects/bdffont/
  Авторские права TridenT 2003 года - Лицензия GNU GPL

  @Author  TridenT
  @Todo    Обработать исключение при ошибке доступа к файлу
-------------------------------------------------------------------------------}
unit u_BDFfont;
...
```

==========

Тег JavaDoc - @comment

Есть два способа работы с этим новым тегом: в виде блочного комментария с тегом @comment или в виде встроенного расширенного комментария внутри метода или функции.

Блочный комментарий

@comment текст
Добавляет подзаголовок "Дополнительный комментарий" с указанным текстом в сгенерированную документацию. Используйте этот тег для добавления другой информации к модулю, классу, методу или функции.

```
{*-----------------------------------------------------------------------------
  Обработка файлов BDF

  BDFfont - это библиотека для обработки шрифтов формата BDF (формата
распределения битовых карт)
  с использованием Delphi.
  Эта библиотека устранит недостаток поддержки растровых шрифтов в MS-Windows.
  Также доступна полная документация и несколько примеров.
  Получите последнюю версию по адресу: http://sourceforge.net/projects/bdffont/
  Авторские права TridenT 2003 года - Лицензия GNU GPL

  @Author  TridenT
  @Comment Обработайте исключение при ошибке доступа к файлу
-----------------------------------------------------------------------------}
unit u_BDFfont;
...
```

Встроенный расширенный комментарий

Добавляет подзаголовок "Дополнительный комментарий" с указанным текстом в сгенерированную документацию. Используйте встроенный тег аналогично тегу описания, но внутри метода или функции. Стандартный синтаксис — двойная косая черта плюс префикс JavaDoc для встроенных элементов.
Этот синтаксис полезен для добавления алгоритма метода в код и в документацию.

```
{*---------------------
  Тест процедуры класса!
---------------------*}
procedure TNewClass.ClassProc;
begin
  /// Это расширенное описание
  if ((Attr and FileInfo.Attr) <> 0) and SameText(ExtractFileExt(FileInfo.Name),
   PAS_FILE_EXT) then
  /// И другое описание
  Inc(i);
end;
```

==========

ANN: Выпущена версия DelphiCodeToDoc v0.23beta - Бесплатная система документации для Delphi

Выпущена последняя версия DelphiCodeToDoc (v0.23b), которую можно скачать здесь:

http://downloads.sourceforge.net/dephicodetodoc/DelphiCodeToDoc_exe_v0.23b.zip

DelphiCodeToDoc - это бесплатная система документации для Delphi, выпущенная под лицензией GNU General Public License. Она использует информацию о символах исходного кода и отформатированных комментариях в файлах для создания точной документации вашего приложения и компонентов. Теперь поддерживается синтаксис JavaDoc.

http://dephicodetodoc.sourceforge.net

Что изменилось в последней версии?

В последней версии v0.23beta внесены следующие изменения:

* Исправлена ошибка парсера с пустым описанием @param (Tracker 2781298)
* Исправлена проблема с некоторыми непарсируемыми файлами из-за специфического атрибута NTFS (Tracker 2523851)
* Добавлена возможность исключить ресурсные строки (Tracker 2736801, 1106583)
* Исправлен баг, связанный с тем, что качество тегов не соответствует категории фильтрации вывода (Tracker 1855474)
* Добавлен генератор PDF

Известные ошибки программы

* Некоторые сообщения не переведены.
* Классы различают поля и переменные, которые должны быть объединены.

Известные ограничения программы

* Переменные класса могут комментироваться только встроенным комментарием непосредственно после объявления.
* Поддерживается только английский язык для генерируемой документации.
* Тег @see еще не реализован.
* При открытом проекте изменение языка не работает для страницы конфигурации.

Что будет включено или изменено в будущих версиях?

* Исправление критических и серьезных ошибок