Соглашение о тегах

# Теги в git

В качестве аналога тегов модулей в SVN в git могут служить аннотированные теги (annotated tags)[[1]](#footnote-1). Тег создаётся для обозначения некоторой вехи в развитии модуля – как правило, стабильной версии, рекомендуемой для использования в прикладной разработке. По сравнению с SVN, теги в git являются легкими: их создание не требует операций копирования файлов; под их создание практически не расходуется место на жёстком диске локальной машины или в репозитории. Фактически, тег в git полностью соответствует своему названию (tag) – это метка, которой помечается коммит. При создании тега, помимо его имени, в базе данных git сохраняется следующая информация:

* наименование тега;
* контрольная сумма (SHA-1);
* автор тега;
* дата и время создания тега;
* описание тега.

**Тег создаётся для репозитория целиком**. В отличие от SVN, в git нельзя выбрать папку, на основе которой делается тег. В связи с этим правила именования тегов пересматриваются.

# Порядок создания и именования тегов

Рекомендуется следующий порядок разработки. Разработка модулей ведётся в главном бранче (master); при необходимости работать над задачей изолированно от других изменений для неё создаётся бранч. Когда доработка завершена и модуль готов к релизу, все необходимые изменения следует закоммитить и отправить в удалённый репозиторий (push).

Если изменению с момента создания последнего тега изменениям подвергся лишь один модуль (например, только JepRia), то имя тега формируется следующим образом:

Module-X.Y.Z, где Module – имя модуля. Пример: JepRia-8.5.0

Если изменению подвергались одновременно несколько модулей, то в теге их названия и номера указываются через запятую:

Module1-X1.Y1.Z1,Module2-X2.Y2.Z2,… Пример: JepRia-8.6.0,JepRiaToolkit-6.2.0

Примечание: **имя тега в git не может содержать пробела**, поэтому название модуля отделяется от версии знаком дефиса.

В описание (message) тега следует писать release notes – так же, как это для тегов в SVN, но с использованием новых соглашений по commit-сообщениям.

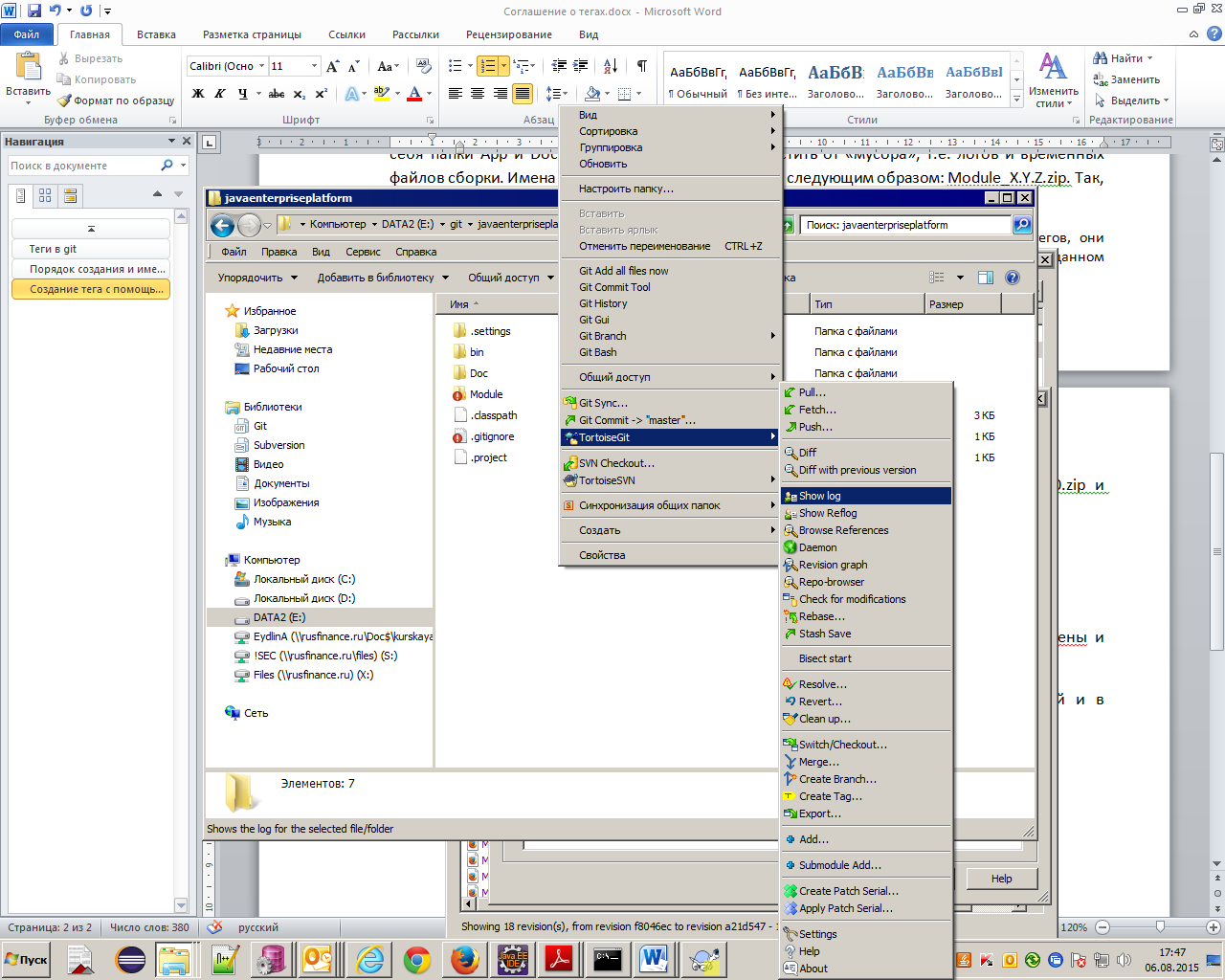
После создания тега из каждого модуля формируются отдельные zip-архивы, включающие в себя папки App и Doc. Папку App необходимо очистить от «мусора», т.е. логов и временных файлов сборки. Имена файлов при этом формируются следующим образом: Module-X.Y.Z.zip. Так, в примере выше с двумя модулями в теге будет создано два файла: JepRia-8.6.0.zip и JepRiaToolkit-6.2.0.zip.

Созданные архивы размещаются [на SourceForge в разделе Files](https://sourceforge.net/projects/javaenterpriseplatform/files/).

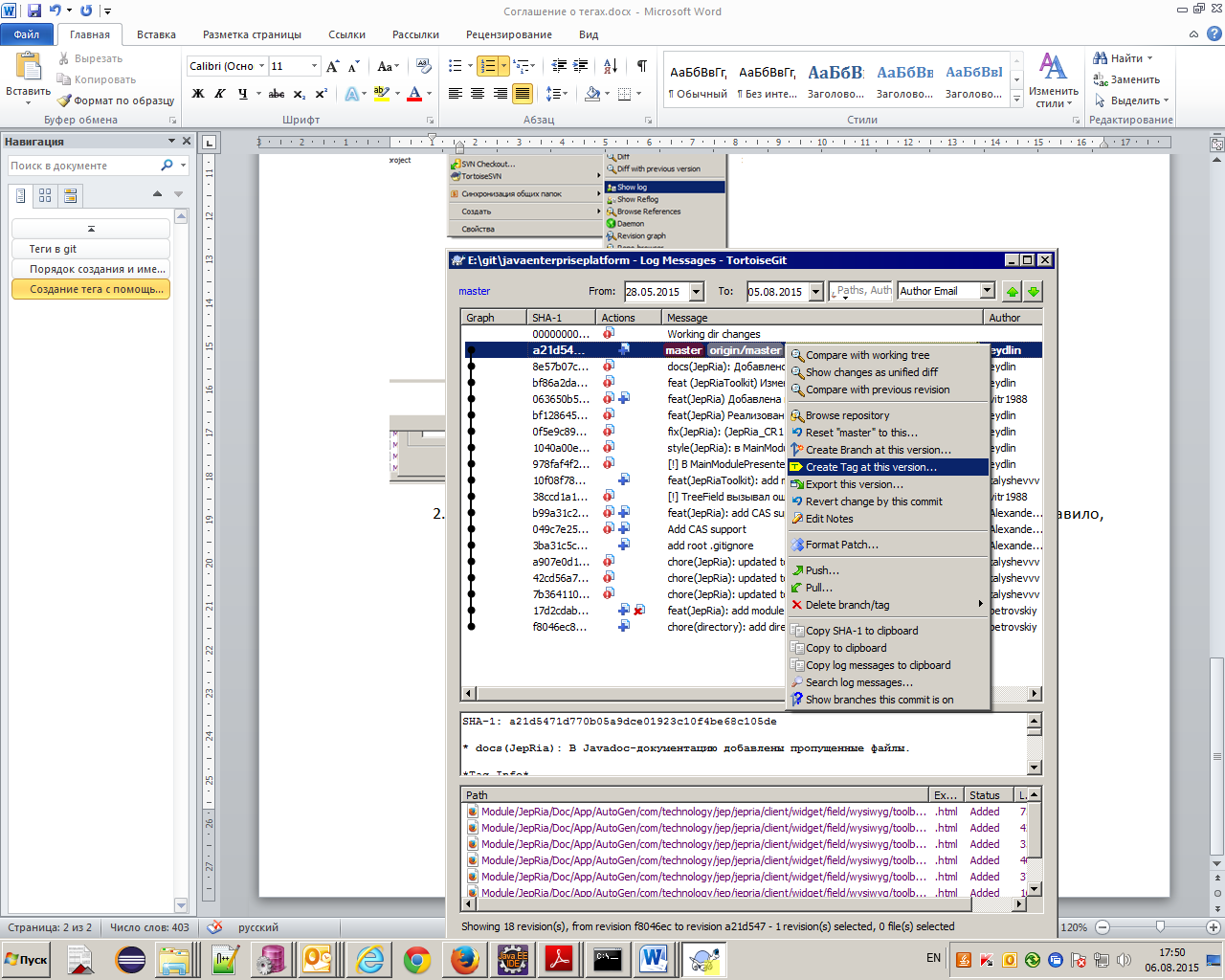
# Создание тега с помощью TortoiseGit

Предполагается, что все планируемые к включению в тег изменения закоммичены и отправлены в репозиторий.

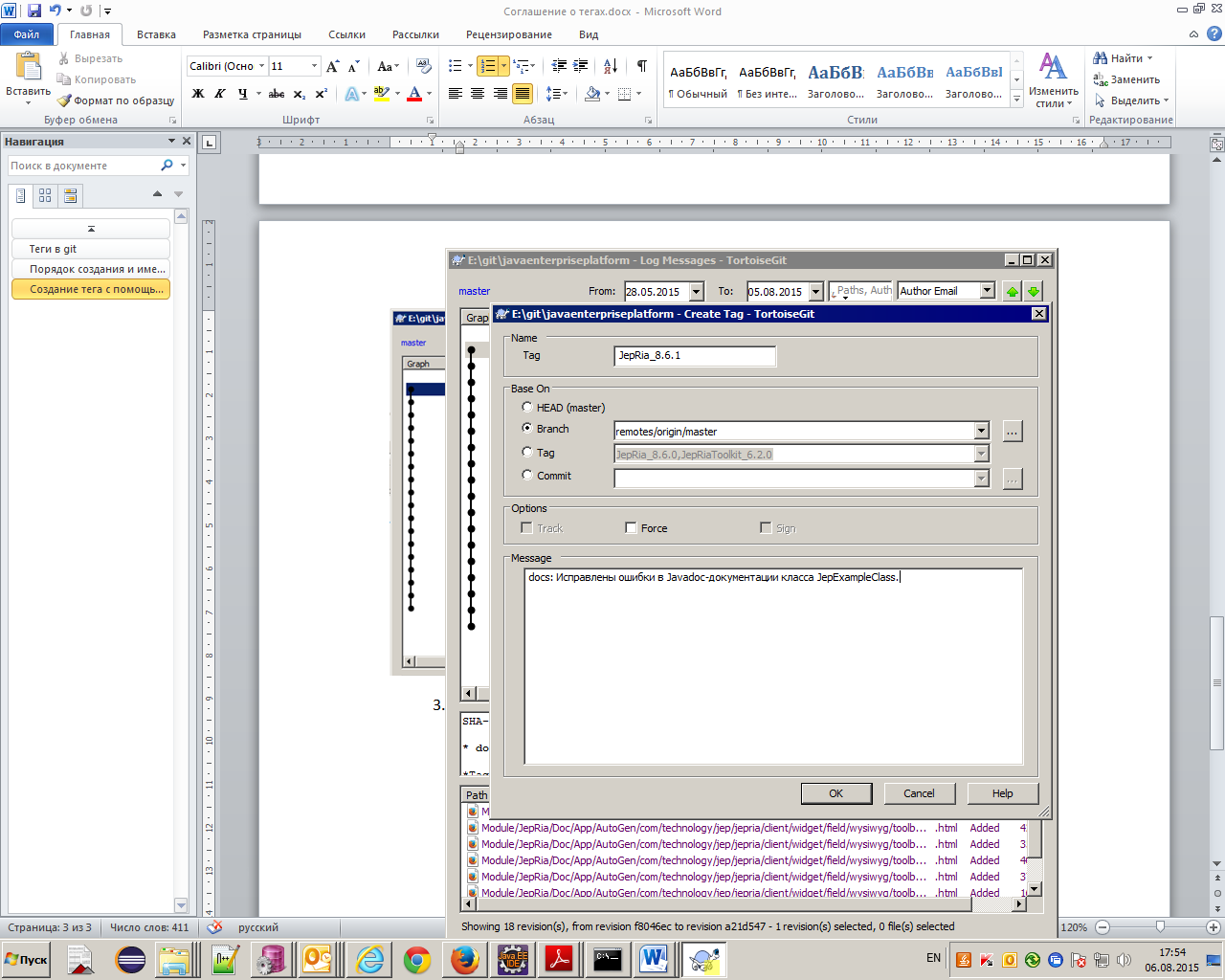
1. В папке репозитория на локальной машине необходимо кликнуть правой кнопкой и в открывшемся меню выбрать *TortoiseGit -> Show log…*



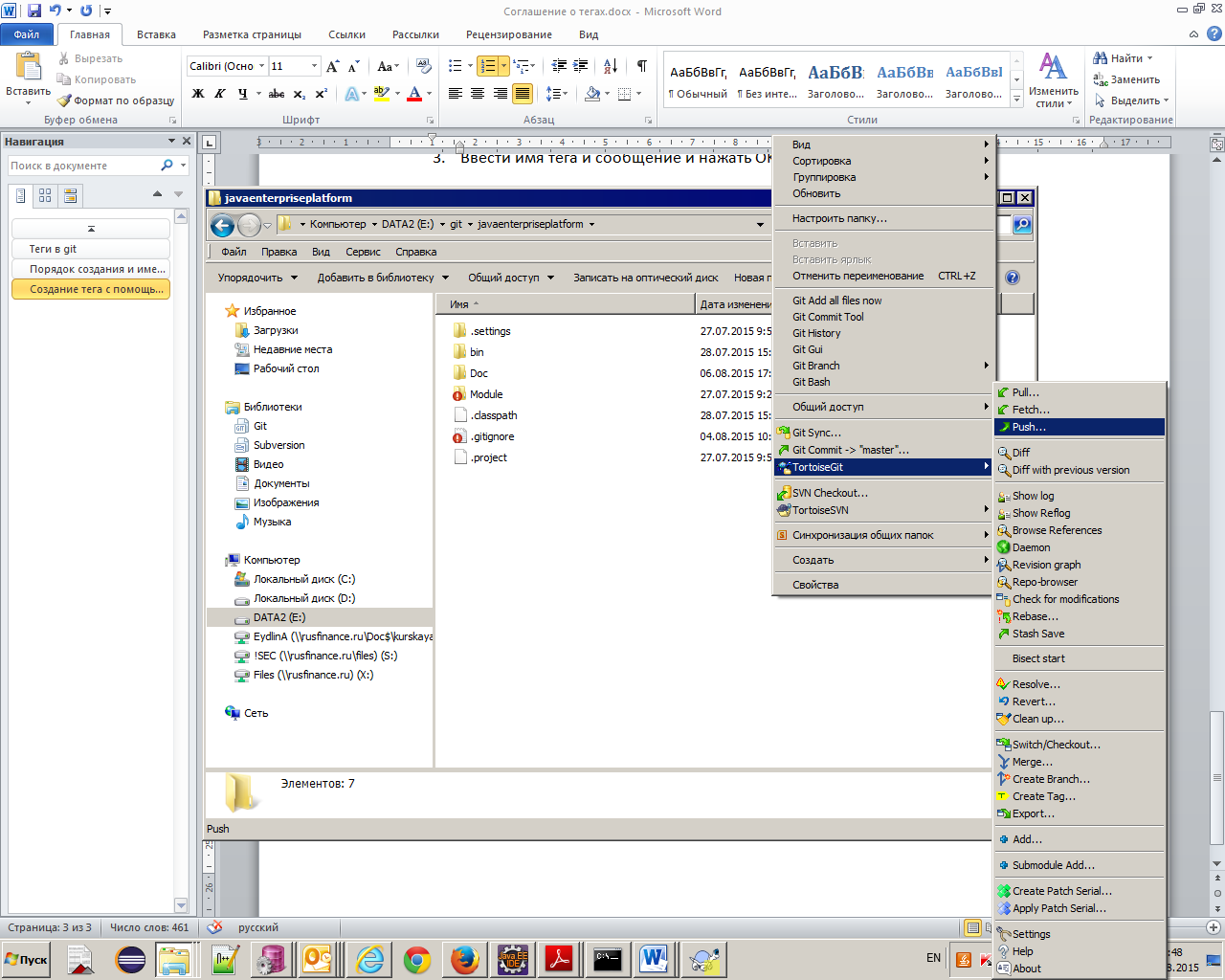
1. В открывшемся окне выбрать коммит, на основе которого создаётся тег (как правило, это будет master), кликнуть правой кнопкой и выбрать *Create Tag at this version…*



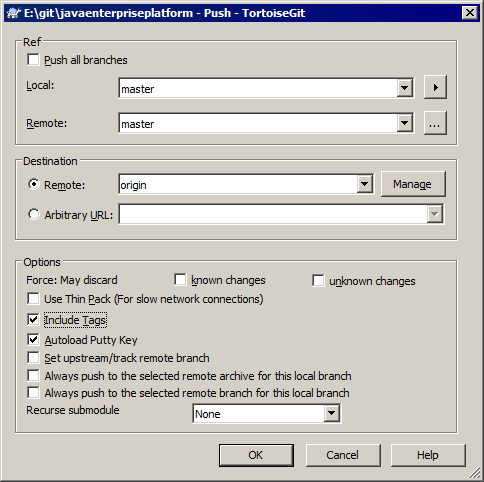
1. Ввести имя тега и сообщение и нажать *OK*.



1. Тег был создан локально, **в репозитории его пока нет**. Необходимо отправить его в репозиторий (push). Для этого необходимо в папке репозитория на локальной машине кликнуть правой кнопкой мыши и выбрать *TortoiseGit -> Push…*



1. В появившемся окне необходимо **поставить флаг** *Include tags* и нажать OK.



**Примечание:** Вместо шагов 4–5 можно воспользоваться открыть командную строку, перейти в папку репозитория и воспользоваться командой:

git push origin --tags

1. Существуют также «легковесные» теги (lightweight tag). В отличие от аннотированных тегов, они представляют собой просто указатель на коммит; под них не создаётся объект в базе данных git. В данном документе они **не** рассматриваются. [↑](#footnote-ref-1)