Известно достаточно много разработчиков, которые не могут понять как же нумеруются версии.
Иногда нужно просто сесть и разобраться, конечно в сети есть куча информации, но иногда ее слишком много, а где-то наоборот. Хочу поделиться как обстоят дела с нумерацией программного обеспечения, манера изложения понятна и доступна даже простому пользователю, которому так же не плохо знать, что же обозначают эти странные циферки после названия программы.

Формат номера версии

Формат номера версии A.B.C.D[r], где:

- A – главный номер версии (major version number).
- B – вспомогательный номер версии (minor version number).
- C – номер сборки, номер логической итерации по работе над функционалом версии A.B (build number).
- D – Номер ревизии, сквозной номер назначаемый автоматически программным обеспечением хранения версий (SVN). Номер ревизии SVN должен синхронизироваться с номером ревизии в AssemblyInfo при каждой сборке релиза (revision number).
- [r] – условное обозначение релиза.

A.B

Совокупность главного и вспомогательного номеров версии (A.B) дают информацию о функционале приложения. Главный номер версии увеличивается только при очень серьёзном изменении функционала. Пользователи, купившие продукт и оплатившие техническую поддержку получают новые версии только в рамках постоянного главного номера версии, соответственно при выпуске новой главной версии пользователи не смогут получить её в рамках технической поддержки и будут вынуждены оплачивать её покупку заново.

C

Номер сборки (билда) (C) должен увеличиваться (зачастую) руководителем проекта по разработке всякий раз, когда продукт передаётся на тестирование.

D

Номер ревизии (D) увеличивается системой контроля версий (SVN) автоматически при работе с ней. Задача руководите проекта по разработке синхронизировать номер ревизии, генерируемый SVN, с номером указанным в AssemblyInfo в модулях проекта. Выполнять эту операцию нужно одновременно с увеличением номера билда (C).

[r]

Обозначение релиза соответствует этапу работы над проектом в рамках жизненного разработки. Выделяют следующие релизы:
• **Pre-alpha** (pa) – соответствует этапу начала работ над версией. Характеризуется большими изменениями в функционале и большим количеством ошибок. Pre-alpha релизы не покидают отдела разработки ПО.
• **Alpha**(a) – соответствует этапу завершения разработки нового функционала. Начиная с alpha версии новый функционал не разрабатывается, а все заявки на новый функционал уходят в план работ по следующей версии. Этап характеризуется высокой активностью по тестированию внутри подразделения разработки ПО и устранению ошибок.
• **Beta** (b) – соответствует этапу публичного тестирования. Это первый релиз, который выходит за пределы отдела разработки ПО. На этом этапе принимаются замечания от пользователей по интерфейсу продукта и прочим найденным пользователями ошибкам и неточностям.
• **Release Candidate** (rc) – весь функционал реализован и полностью оттестирован, все найденные на предыдущих этапах ошибки исправлены. На этом этапе могут вноситься изменения в документацию и конфигурации продукта.
• **Release to manufacturing** или **Release to marketing** (rtm) – служит для индикации того, что ПО соответствует всем требованиям качества, и готово для массового распространения. RTM не определяет способа доставки релиза (сеть или носитель) и служит лишь для индикации того, что качество достаточно для массового распространения.
• **General availability** (ga) – финальный релиз, соответствующий завершению всех работ по коммерциализации продукта, продукт полностью готов к продажам через веб или на физических носителях.
• **End of life** (eol) – работы по развитию и поддержке продукта завершены.
В скобках указаны сокращения, используемые для формирования номера релиза. Если в номере не указано ни одного сокращения, то считается что это релиз General availability (ga).
Помимо сокращённого обозначения в наименовании версии обозначение релиза должно указываться в исходных файлах проекта через атрибут:

```
[AssemblyConfiguration]
```

В случае большого количества проектов в решении рекомендуется использовать один файл GlobalAssemblyInfo.cs (или GlobalAssemblyInfo.vb) с указанием ссылки на него во всех проектах решения и именно в нём проставлять вид релиза.
Пример C#:

```
using System.Reflection;
[assembly: AssemblyConfiguration("Beta")]
```

Пример VB.NET:

```
Imports System.Reflection
```

Примеры

HBR 2.3.1.1260b – релиз HBR версии 2.3, сборка 1, ревизия 1260, бета.
HBR 2.3.2.1370rc – релиз HBR версии 2.3, сборка 2, ревизия 1370, релиз-кандидат.
HBR 2.3.5.1432 – релиз HBR версии 2.3, сборка 5, ревизия 1432, финальный релиз.

Версии модулей/дополнение

Если в составе ПО выделены модули или дополнения, то можно применять два подхода к ведению номеров их версий.
1. **Синхронная нумерация** – нумерация модулей и дополнений совпадает с версией самого приложения.
2. **Индивидуальная нумерация** – нумерация версии модуля или дополнения ведётся индивидуально как для отдельного самостоятельного приложения.
Первый подход рекомендуется применять на этапе активной разработки приложения до выхода первого ga-релиза в текущей версии. Если функционал модуля устоялся и не требует изменений при развитии других модулей или самого приложения, то рекомендуется применять второй подход.

Имя файла дистрибутива

Имя дистрибутива должно однозначно указывать продукт и полный номер версии.
При сборке дистрибутива как набора несжатых файлов корневая папка, в которой располагаются подпапки и несжатые файлы дистрибутива именуется по формату «<Имя продукта> A_B_C_D[r]».
При сборке дистрибутива как msi-файла, msi-файл должен переименовываться в

«<Имя продукта> A_B_C_D[r]».
При сжатии в архив каталога с файлами дистрибутива архив должен именоваться аналогично: «<Имя продукта> A_B_C_D[r]».
Такой принцип нумерации версий использует большинство разработчиков десктопных приложений.

Содержимое статьи чуть менее, чем полностью взято у автора https://habrahabr.ru/users/ifmalex/