X.Org Version Numbering Schemes

The XFree86 Project, Inc

Updated for X.Org by Keith Packard,
Kevin E. Martin, and Alan Coopersmith

November 2010

Abstract

X.Org has adopted the same basic numbering scheme used by the XFree86 Project, Inc. for their releases. The actual numbers are different, but the basic scheme is the same. This document reflects the policy that X.Org uses.

Table of Contents

Module Versions

Starting with the X11R7.0 release, each module has its own version number. For those without a natural starting point, the version numbers started at 1.0. For instance, the X11R7.0 release included the xorg-server 1.0 module. As modules are released independently from the rest of the window system, the module version is the most accurate source of version information. For instance, there are many X server releases in a year, but generally only one window system release, so an X server version number such as 1.7.7 is more informative than the X11R7.5 version for the window system “katamari” release.

Unfortunately, up through the X server 1.3 release, the X server used the Window System version when reporting its version number in log files, the -version option, and the connection setup string (displayed by xdpyinfo). This was corrected with X server 1.3, which caused the visible version number string to appear to jump backwards from 7.2 to 1.3.

Releases, Development Streams and Branches

X.Org has two release branches for the X server software, and several other modules with active ongoing development. First is the trunk of the git repository. This is the main development stream, where all new work and work for future releases is done.

Second is the stable bugfix branch for the latest full release. It is created around the time of the release. The branch will be named for the release version, such as “`server-1.9-branch`” for the X server 1.9.x series of releases. Fixes for bugs found in the release will be added to this branch (as well as the trunk), and updates to this release (if any) will be cut from this branch. Similar stable branches are present for previous full releases.

The X.Org Foundation is planning to make full releases from the main development stream at regular intervals in the 6-12 month range. The feature freezes for these releases will usually be 2-3 months before the release dates. This general plan is a goal, not a binding commitment. The actual release intervals and dates will depend to a large degree on the resource available to X.Org. Update/bugfix releases will be made on an as-required basis, depending also on the availability of resources, and will generally be limited to serious bug and security fixes. New features will not usually be added in update releases.

Aside from actual releases, snapshots of the active release branches are tagged in the git repository from time to time. Each such snapshot has an identifiable version number.

Current Version Numbering Scheme

Starting with the main development branch after X11R6.7, the X.Org versions are numbered according to the scheme outlined here.

The version numbering format is `M.m.P.s`, where *M* is the major version number, *m* is the minor version number, *P* is the patch level, and *s* is the snapshot number. Full releases have *P* set to zero, and it is incremented for each subsequent bug fix release on the post-release stable branch. The snapshot number *s* is present only for between-release snapshots of the development and stable branches.

Development Branch

Immediately after forming a release stable branch, the patch level number for the main development branch is bumped to 99, and the snapshot number is reset. The snapshot number is incremented for each tagged development snapshot. The git tag for snapshots is “`xorg-server-M.m.P.s`”. When the development branch enters feature freeze, the snapshot number is bumped to 900. A stable branch may be created for the next full release at any time after the feature freeze. When it is, the branch is called “`server-M.m-branch`”. The snapshot number is incremented from there until the release is finalised. Each of these snapshots is a “release candidate”. When the release is finalised, the minor version is incremented, the patch level is set to zero, and the snapshot number removed.

Here's an example which shows the version number sequence for the development leading up to version 1.8:

`1.7.99.1`	The first snapshot of the pre-1.8 development branch.
`1.7.99.23`	The twenty-third snapshot of the pre-1.8 development branch.
`1.7.99.900`	The start of the 1.8 feature freeze.
`1.7.99.903`	The third 1.8 release candidate.
`1.8.0`	The 1.8 release.
`1.8.99.1`	The first pre-1.9 development snapshot, which is the first main branch snapshot after creating the 1.8 stable branch.

Stable Branch

After a full release, the stable branch for the release will be maintained with bug fixes and important updates until the next full release. Any snapshots on this branch are considered “release candidates,” which is indicated by setting *s* to a number above 900. The snapshot number is incremented for each release candidate until the update release is finalised. The patch level value (*P*) is incremented for each update release.

Here's an example which shows a version number sequence for a 1.8.x stable branch:

`1.8.0`	The 1.8 release.
`1.8.0.901`	The first pre 1.8.1 snapshot.
`1.8.0.903`	The third pre 1.8.1 snapshot, also known as the third 1.8.1 release candidate.
`1.8.1`	The 1.8.1 release.
`1.8.1.901`	The first pre 1.8.2 snapshot.
`1.8.2`	The 1.8.2 release.

Finding the X.Org X Server Version From a Client

The X.Org X servers report a `VendorRelease` value that matches the X.Org version number. There have been some cases of releases where this value wasn't set correctly. The rules for interpreting this value as well as the known exceptions are outlined here.

As noted above, the version reported by `VendorRelease` changed from the window system version to the X server version starting in the xorg-server 1.3 release.

For all X.Org development and release versions using this numbering scheme, the `VendorRelease` value is *MMmmPPsss*. That is, version *M.m.P.s* has `VendorRelease` set to `M * 10000000 + m * 100000 + P * 1000 + s`.

The following is a code fragment taken from `xdpyinfo.c` that shows how the `VendorRelease` information can be interpreted.

```
    if (strstr(ServerVendor(dpy), "X.Org")) {
        int vendrel = VendorRelease(dpy);

        printf("X.Org version: ");
 printf("%d.%d.%d", vendrel / 10000000,
        (vendrel /   100000) % 100,
        (vendrel /     1000) % 100);
 if (vendrel % 1000) {
     printf(".%d", vendrel % 1000);
```

```
}
    }
```