

04 Release Management

Release Management is the process of managing software releases throughout an ongoing cycle of development, testing, and release.

[processes:manualSteps](#)

Introduction

Whenever we finish a set of features, enhancements or bug fixes in one of our products or projects, we conduct various methods and levels of testing. If the software is verified to be in a stable and functional state, it is *released* to the customer or to the public.

Each release gets a unique and “human-readable” identifier, a [version number](#). Some of these releases are distributed by copying an archive file of the software to our web server and providing download links.

We have defined and documented our own, personalised [process for releasing software](#). You can find details about that on our [Way of Working](#) pages. Many of our projects are released automatically through our [Release Automation Tools](#).

Accessing releases with Dokwiki

In order to keep things simple for you, we have developed a [Dokuwiki plugin](#) that serves as a simplified front-end to GitLab and our file servers, displaying a list of releases and allowing to download associated files directly within your Dokuwiki project page.

For your project, you will find the list of software releases as a subpage of your project page:

```
/projects/<customer>/<project>/releases
```

List of releases

Showing all releases for the appinstruments/qseries-firmware project.	
 v0.1.2 🔗 0916106d by Joerg Hampel (joerg@hampel.at) on 03.03.2015 09:17 fixed processing of new version number string	
Downloads	
 Q.series-Gateway_v0.1.2.lvappimg	24.50 MB
 v0.1.1 🔗 94ba5a11 by Joerg Hampel (joerg@hampel.at) on 03.03.2015 08:20 Pre-Build VI, tagging - pre-build vi will eventually be merged with appic-helpers/appbuilder and removed from library directory - new tag v0.1.1 for further tag-testing No downloads available.	
List of releases	

For each release, the list displays

- the release name (also called *version number* or *tag*)
- the first 8 digits of the git commit hash
- the user and email of the comitter
- the date this version was committed
- the commit message
- associated downloads (if any)

Version Numbers

Format

We loosely adhere to the [Semantic Versioning](#) standard. This means that a normal version number takes the form `X.Y.Z` where `X`, `Y`, and `Z` are non-negative integers:

- `X`: Major version - identifies versions that introduce *incompatible API changes*
- `Y`: Minor version - gets increased when *functionality* is added in a *backwards-compatible* manner
- `Z`: Patch version - shows versions with *backwards-compatible bug fixes*

Elements increase numerically. For instance: `1.9.0` → `1.10.0` → `1.11.0`.

From:

<https://dokuwiki.hampel-soft.com/> - **HAMPEL SOFTWARE ENGINEERING**

Permanent link:

<https://dokuwiki.hampel-soft.com/processes/release-management?rev=1603353112>

Last update: **2020/10/22 07:51**

