CONSTRUCT RELEASE CHANNELS

View online: https://www.construct.net/en/make-games/manuals/construct-3/tips-and-guides/release-channels

You can find every release of Construct, past and present, on the releases page. Construct releases are split in to three types of releases, called *release channels*: **beta**, **stable** and **LTS** (Long Term Support). Broadly speaking, beta releases are the most frequent releases but most likely to have unexpected issues, and LTS releases the most infrequent releases and least likely to have unexpected issues. This guide briefly describes the differences between them and why you might want to choose each.

Beta releases

Released about once a week, beta releases are mainly for testing purposes. Beta releases haven't had as much testing as other releases, so unexpected problems are more likely to come up. Be prepared for occasional disruption. You should probably not publish a finished project from a beta release. However these releases get the latest bug fixes, new features and other improvements the soonest. You may want to try a beta release to verify that a bug fix works correctly for your project. People who want the latest features can use beta releases to try out the latest improvements. Enthusiasts, or anyone else who wants to shape the development of Construct, may also be interested in using beta releases and providing feedback to help us make Construct in to what you want and need it to be - beta releases are where the main active development of Construct takes place.

Occasionally very serious bugs may warrant a patch release of a beta release (i.e. a .2 release the same week), but in general you should assume any issues will only be fixed in the following week's beta release at the earliest.

You can load the latest beta release by visiting: editor.construct.net/beta

Stable releases

Released every 2-3 months, stable releases are the culmination of previous work on beta releases. Once a suitable batch of work has been completed and any significant known issues are identified and resolved, a new stable release is published including all the changes from the past beta releases since the previous stable release. By this point all the changes are much more likely to be reliable and bug-free. However sometimes mistakes happen and new issues appear. Any major issues will likely be fixed in follow-up patch releases (i.e. a .2 or .3 release). Other issues may be dealt with in subsequent beta releases and so only reach the next stable release.

Most people use stable releases - when you visit editor.construct.net in the address bar, it will load the most recent stable release.

The time in between stable releases is called a release cycle. Usually the few beta releases prior to a stable release stop introducing new features and instead focus solely on bug fixes and

reliability to ensure the next stable release goes smoothly.

LTS releases

LTS stands for Long Term Support. These are based off a mid-year stable release (around June or July), and are supported with essential bug fixes and maintenance updates only for a period of 18 months (so until the end of the following year). The aim is to have as few changes as possible to ensure a consistent platform for purposes like testing and launching a major new project, or in education to minimize any potentially disruptive changes during the academic year. LTS releases start out as a stable release, but before the next stable release comes out, the stable release is "promoted" to an LTS release.

Most people should use stable releases over LTS releases. You should only choose an LTS release where the importance of predictability outweighs the importance of bug fixes and the latest new features. Software development is complex, and sometimes new bugs can accidentally make their way to new stable releases. It's even possible a project depends on a bug to work correctly, and fixing the bug causes the project to stop working; in a large project, fixing such issues can sometimes be hard work. Sometimes changes which affect compatibility are intentionally made for the long-term benefit of Construct. Suppose you are preparing to launch a game that has been in development for a few years - in this case the last thing you want is some new bug or compatibility difference to break your game just before launch. This is a good situation to choose an LTS release: as you approach launch and start to focus on testing your game, you may want to switch to an LTS release to minimize any unexpected changes to Construct.

The downside of using an LTS release is that you won't get access to the latest new features and updates, nor will you necessarily get the latest bug fixes either. If you are wondering if any particular fix or change will be applied to an LTS release, the default answer is "no". Every change brings a risk of breakage - as noted even bug fixes can cause breakages - and in order to provide a consistent platform, LTS releases will change as little as possible. In general only fixes which are absolutely critical will be applied to LTS releases. Generally this involves a new requirement makes it impossible to publish projects any more unless something is changed. For example if mobile publishing requirements are updated and a change must be made to continue publishing mobile apps, or if a browser bug or compatibility change meant existing projects were broken, that would warrant an update to an LTS release. In virtually all other cases, the change will only be made for the latest beta release and be rolled out to everyone at the following stable release. Even where LTS updates are made, they may only be some time after the change is made to other release channels, in order to verify that the change works as expected.

If you use LTS releases, there is a chance you run in to some bug that is only fixed in the latest stable release and does not meet the high bar to be applied to the LTS channel. In this case you will have to update to the latest stable release. Therefore whether to use an LTS release, and when to start using it, depends on your judgement of the trade-off between regular maintenance and a predictable platform to publish with.

If you use an LTS release for more than a year, you may wish to consider switching to the LTS release that comes out the following year. This choice depends on how far through your project

you are: if publishing is still some way off, it is probably a good idea to update and deal with any compatibility differences; if your project has already been published and you are just maintaining it and don't think you'll need many more updates, then you may want to just stick with the same LTS release.

You can load the latest LTS release by visiting: editor.construct.net/lts

Release numbering

Construct's release numbering scheme merely increments the release number for every beta release, and is prefixed with an "r" for "release". For example a beta r400 will be followed by another beta r401 about a week later. Every 2-3 months, a new stable release is made continuing the incrementing release numbers. Occasionally "patch" releases are made to apply changes to an existing release. These increment a number after a dot - e.g. r407.2 is a patch release for the original r407 release.

Once a year the latest stable release is "promoted" to a new LTS release. As LTS releases are supported for longer, they will tend to get more patch releases, such as r397.2, r397.3, and so on.

The life of a change

Suppose a change is made to Construct, such as a bug fix. Normally such a change will work its way through the release channels as follows.

1. If the change is for a critical issue, it may be released as a patch release for beta, stable or LTS releases. (Note the LTS release may be updated some time after other channels to verify the change works as expected.)
2. Otherwise the change will first appear in the following weeks' beta release.
3. Within a couple of months, the next stable release will be published including the change.
4. At the next annual update, the current stable release will be promoted to LTS, including the change.

The release channel you choose depends on the trade-off you want between the latest bug fixes and new features, and the chance that unexpected problems could cause disruption to your project.