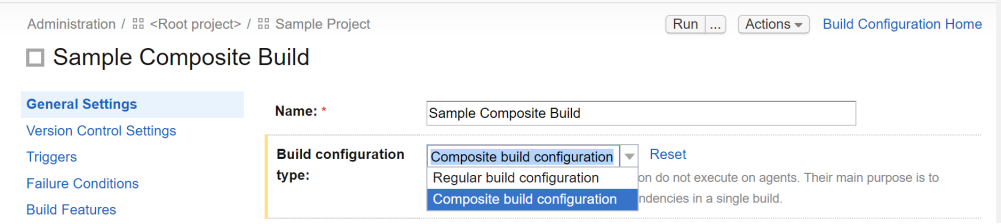
Indore 2017.2 EAP2 (build 49708) Release Notes

See also Indore 2017.2 EAP1 (build 49391) Release Notes

Composite Builds

In this EAP we're introducing a new kind of a build configuration - a composite one. The purpose of this build configuration is to aggregate results from several other builds combined by snapshot dependencies and present them in a single place.

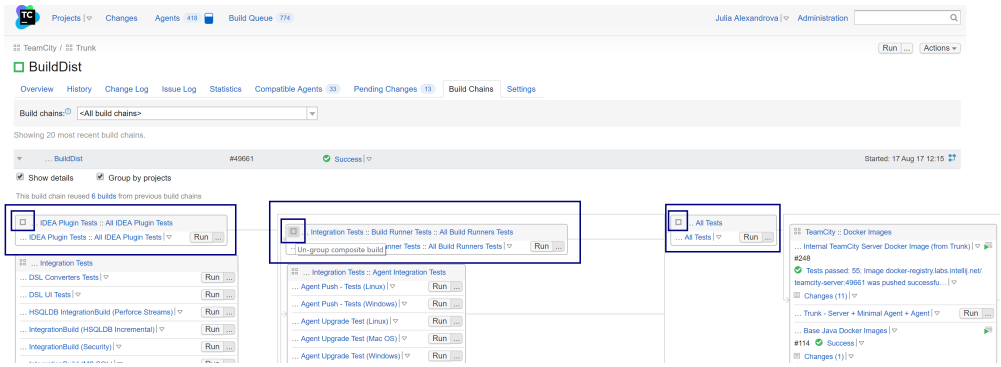
When creating a build configuration, you can now select the build configuration type:



There are important differences between such build configurations and usual configurations with snapshot dependencies:

- A composite build is shown as running at the same time when the first dependency of the build chain starts, and is shown as finished when the last dependency finishes
- A composite build does not occupy an agent, as a result some settings which require an agent, like build steps or requirements, are not applicable
- A composite build aggregates results from dependencies, for instance, it aggregates all of the tests and shows all failed/muted or ignored tests of the chain in a single place
- The status line of the composite builds reflects the current chain state: the number of running/queued/finished or failed dependencies
- The progress indicator of the composite build reflects all of the dependencies, so it actually shows when the whole chain is going to finish
- A composite build often acts as a single build despite the number of dependencies; so for instance, if one decides to stop this build or remove it from the queue, all dependencies are stopped / removed too
- Another example of a single build behavior is "maximum number of running builds" setting. If for a composite build it is set to 1, then, if some composite build is running, the next one will not start until the previous is finished, which means that dependencies of the next composite build will not start either and so the whole chain will wait till the current one finishes.

On the build chain view TeamCity also automatically hides all of the dependencies of the a composite build, showing a single node for such builds by default.Since this build behaves like a single build, it is possible to create a notification rule on "Build starts" or "The first build error occurs" event and get a notification when the first dependency starts or reports a build problem. To some extent, a composite build can be viewed as a build which consists of a number of parts which can be executed in parallel on different agents. All these parts will have synchronized snapshot of the source code and the results can be seen in a single place.



Docker Integration Improvements

In the previous EAP we announced the initial support for Docker. This EAP brings the following improvements:

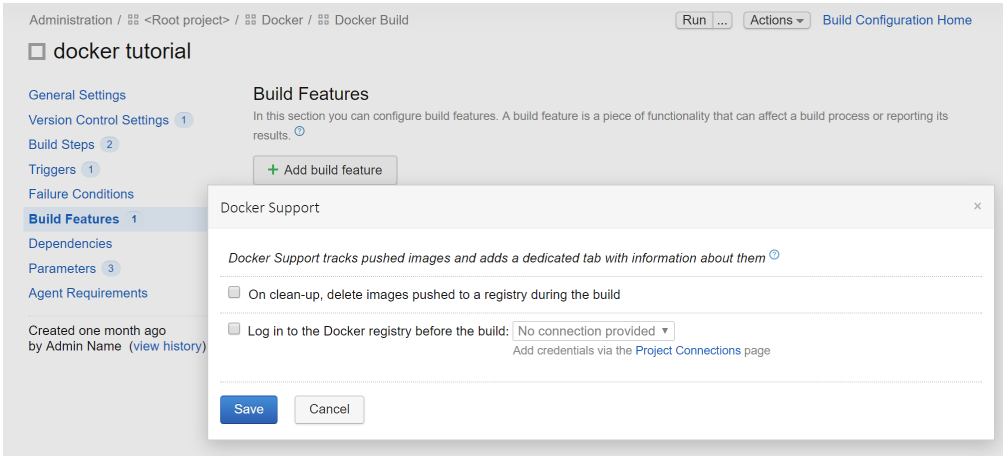
Setting Ownership of Files Created in Checkout Directory

When a process is run under Docker in the Linux environment, the owner of the created files is root and the files created by a step with the Docker Wrapper may be not writable by the parent build agent process. Rather than setting `umask 0` (as it was done in the previous EAP), access to such files is restored by the `"chown -R buildAgentUserId <checkout directory>"` command automatically run by TeamCity at the end of the build step with Docker Wrapper.

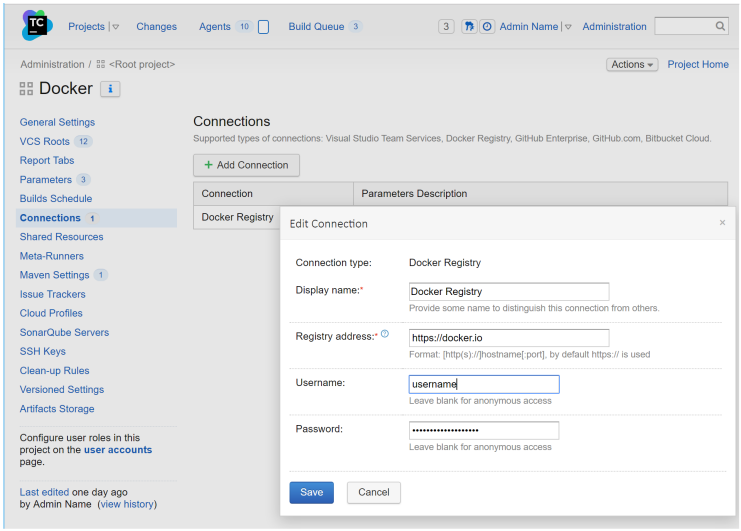
Docker Support Build Feature Improvements

The Docker Support build feature now provides 2 new options:

- the ability to clean-up the images
- automatic login to an authenticated registry before the build and log out of it after the build



These options require a configured connection to a docker registry: on the Project Settings > Connections page you can now configure an authorized connection to docker.io (default) or a private Docker registry. More than one connection can be added to the project.



Clean-up of images

If you have a build configuration which publishes images, you need to remove them at some point. You can select the corresponding option and instruct TeamCity to remove the images published by a certain build when the build itself is cleaned up. It works as follows: when an image is published, TeamCity stores the information about the registry of the images published by the build. On cleaning up a build, all the configured connections are searched for the address of this registry and the images published by the build are cleaned up using the credentials specified in the found connection.

Automatic Login/Logout to Docker Registry

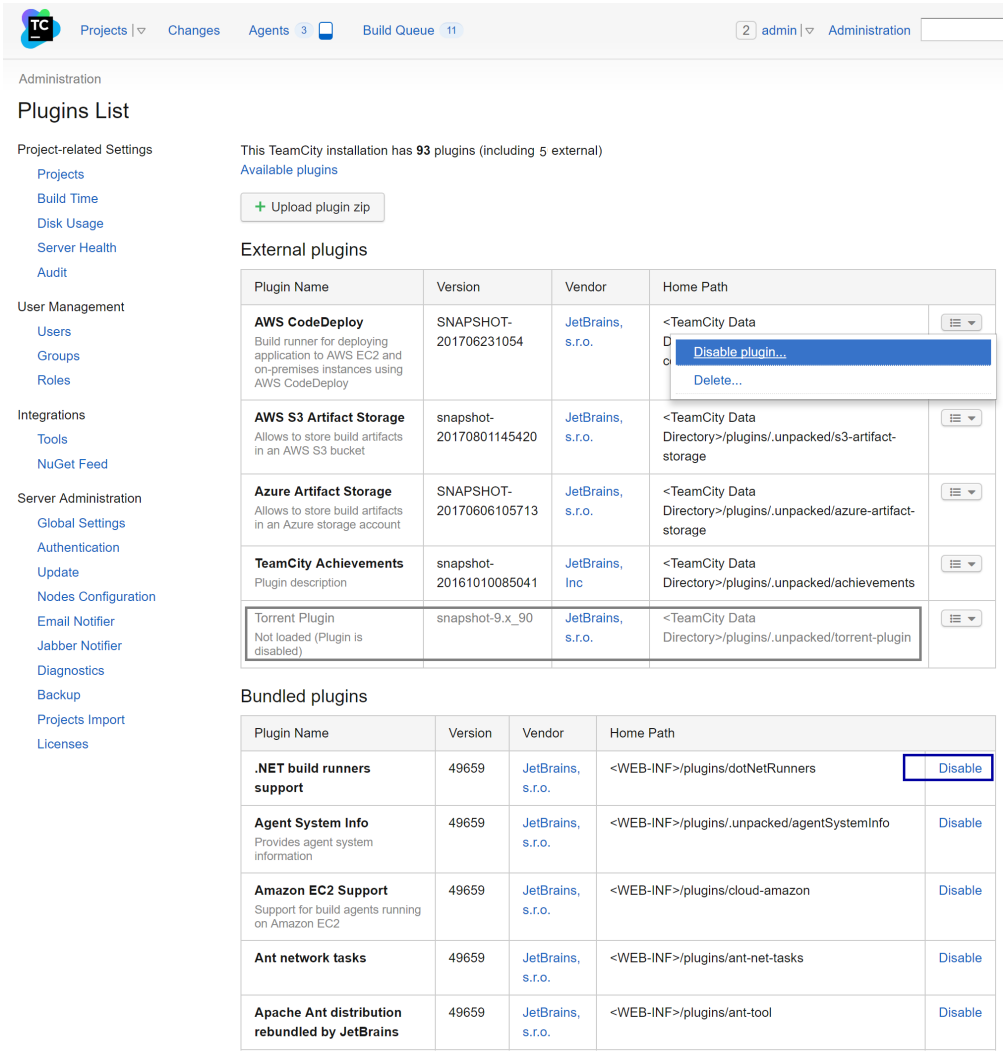
If you need to log in to a registry requiring authentication before a build, select the corresponding option and a connection to Docker configured in the project settings. Automatic logout will be performed after the build finishes.

Tracking Newly Uploaded Plugins

Now when a new plugin is uploaded to the server, it is displayed on the Administration | Plugins List page. If the uploaded plugin overwrites an existing one, a note is displayed warning the administrator that the plugin will be overwritten.The uploaded plugins are displayed, but they will be installed and enabled only after the server restart.

Disabling Plugins from UI

Starting from this version, any plugin can be disabled using the TeamCity UI: on the Administration | Plugins List page every e xternal plugins has a button, clicking which opens a popup with the corresponding option; the bundled plugins have a link which can be used to disable them. On disabling a plugin, a warning is displayed that this may impact other TeamCity components (e.g. other plugins). During the next server restart, the disabled plugin is not loaded. If there are other plugins depending on the disabled one, they will not be loaded either. Disabled plugins are greyed out in the list.



TeamCity Auto-Update

We’re also working on the TeamCity auto-update feature, which should further enhance your upgrade experience. TeamCity server startup scripts have been reworked significantly to handle the automatic upgrade process. There is also the Administrati on -> Update page which will show new versions of TeamCity where you can start the server upgrade process. One of the bonuses of developing this feature was implementation of the “Restart Server” button, which was announced in the previous EAP. We hope that this will make the upgrade of your server to the next EAP build much simpler.

Other Improvements

- The TeamCity Audit log now contains information on including / excluding roles and adding / deleting permissions
- Now TeamCity will automatically terminate a long running TFS app in case of inactivity (e.g. a user just wanted to test a TeamCity TFS connection) and when a certain run time is exceeded (e.g. to address memory leaks).
  These periods are controlled by the following configurable properties:
  - `teamcity.tfs.java.long.living.process.idleTime` defaults to 30 min
  - `teamcity.tfs.java.long.living.process.lifeTime` is disabled by default
- REST API now allows investigations assignment
- All fixed issues