Installing and Configuring the TeamCity Server

This page covers a new TeamCity server installation.
For upgrade instructions, please refer to Upgrade.

To install a TeamCity server, perform the following:

- Choose the appropriate TeamCity distribution (.exe, .tar.gz or Docker image) based on the details below. You can also run TeamCity on AWS stack.
- Download the distribution
- Review software requirements and hardware requirements notes and platform selection
- Review TeamCity Licensing Policy
- Install and configure the TeamCity server per instructions below

This page covers:

Installing TeamCity Server

After you selected one of the the TeamCity installation options, proceed with corresponding installation instructions:

- Windows .exe distribution - the executable which provides the installation wizard for Windows platforms and allows installing the server as a Windows service;
- .tar.gz distribution - the archive with a "portable" version suitable for all platforms;
- Docker image - check the instructions at the image page;
- .war distribution - for experienced users who want to run TeamCity in a separately installed Web application server. Please consider using .tar.gz distribution instead. .war is not recommended to use unless really required (please let us know the reasons).

Compared to the .war distribution, the .exe and .tar.gz distributions:

- include a Tomcat version which TeamCity is tested with, so it is known to be a working combination. This might not be the case with an external Tomcat.
- define additional JRE options which are usually recommended for running the server
- have the teamcity-server startup script which provides several convenience options (e.g. separate environment variable for memory settings) and configures TeamCity correctly (e.g. log4j configuration)
- (at least under Windows) provide better error reporting for some cases (like a missing Java installation)
- under Windows, allow running TeamCity as a service with the ability to use the same configuration as if run from the console
- come bundled with a build agent distribution and single startup script which allows for easy TeamCity server evaluation with one agent
- come bundled with the devPackage for TeamCity plugin development.
- may provide more convenience features in the future

After installation, the TeamCity web UI can be accessed via a web browser. The default addresses are http://localhost/ for Windows distribution and http://localhost:8111/ for tar.gz distribution.

If you cannot access the TeamCity web UI after successful installation, please refer to the #Troubleshooting TeamCity Installation Issues section.

The build server and one build agent will be installed by default for Windows, Linux or macOS. If you need more build agents, refer to the Installing Additional Build Agents section.

During the server setup you can select either an internal database or an existing external database. By default, TeamCity uses an HSQLDB database that does not require configuring. This database suites the purposes of testing and evaluating the system.
For production purposes, using a standalone external database is recommended.

Installing TeamCity via Windows installation package

For the Windows platform, run the executable file and follow the installation instructions. You have options to install the TeamCity web server and one build agent that can be run as a Windows service.

If you opted to install the services, use the standard Windows `Services` applet to manage the service.
Otherwise, use standard scripts.

If you did not change the default port (80) during the installation, the TeamCity web UI can be accessed via http://localhost/ in a web browser running on the same machine where the server is installed. Please note that port 80 can be used by other programs (e.g. Skype, or other web servers like IIS). In this case you can specify another port during the installation and use http://localhost:<port>/ address in the browser.

If you want to edit the TeamCity server's service parameters, memory settings or system properties after the installation, refer to the Configuring TeamCity Server Startup Properties section.

Service account
Make sure the user account specified for the service has:

- log on as service right (related Microsoft page)
- write permissions for the TeamCity Data Directory,
- write permissions for the TeamCity Home , i.e. directory where TeamCity was installed,
- all the necessary permissions to work with the source controls used. This includes:
  - access to Microsoft Visual SourceSafe database (if Visual SourceSafe integration is used).
  - the user, under whose account the TeamCity server service runs, and ClearCase view owner are the same (if the ClearCase integration is used).

By default, the Windows service is installed under the SYSTEM account. To change it, use the Services applet (Control Panel | Administrative Tools | Services)

Installing TeamCity bundled with Tomcat servlet container (Linux, macOS, Windows)

Review software requirements before the installation.

Unpack the `TeamCity<version number>.tar.gz` archive (for example, using the `tar xfz TeamCity<version number>.tar.gz` command under Linux, or the WinZip, WinRar or similar utility under Windows).
Please use GNU tar to unpack (for example, Solaris 10 tar is reported to truncate too long file names and may cause a `ClassNotFoundException` when using the server after such unpacking. Consider getting GNU tar at Solaris packages or using the `gtar xfz` command).

Ensure you have JRE or JDK installed and the JAVA_HOME environment variable is pointing to the Java installation directory. Java JDK 1.8.0_16 or later is required.

Installing TeamCity into Existing J2EE Container

It is not recommended to use .war distribution. Use the TeamCity .tar.gz distribution (bundled with Tomcat web server) instead. If you have important reasons to deploy TeamCity into existing web server and want to use .war distribution, please let us know the reasons.

1. Make sure your web application server is stopped.
2. Copy the downloaded `TeamCity<version number>.war` file into the web applications directory of your J2EE container under the `TeamCity.war` name (the name of the file is generally used as a part of the URL) or deploy the .war following

the documentation of the web server. Please make sure there is no other version of TeamCity deployed (e.g. do not preserve the old TeamCity web application directory under the web server applications directory).
3. Ensure the TeamCity web application gets sufficient amount of memory. Please increase the memory accordingly if you have other web applications running in the same JVM.
4. If you are deploying TeamCity to the Tomcat container, please add `useBodyEncodingForURI="true"` attribute to the main `Connector` tag for the server in the `Tomcat/conf/server.xml` file.
5. If you are deploying TeamCity to Jetty container version >7.5.5 (including 8.x.x), please make sure the system property `org.apache.jasper.compiler.disablejsr199` is set to `true`
6. Ensure that the servlet container is configured to unpack the deployed war files. Though for most servlet containers it is the default behavior, for some it is not (e.g. Jetty version >7.0.2) and should be explicitly configured. TeamCity is not able to work from a packed .war: if started this way, there will be a note on this the logs and UI.
7. Configure the appropriate TeamCity Data Directory to be used by TeamCity.

   > ⚠ Note that it is recommended to start with an empty TeamCity Data Directory. After completing the installation and performing the first TeamCity server start, the required data (e.g. database settings file) can be moved to the directory.

8. Check/configure the TeamCity logging properties by specifying the `log4j.configuration` and `teamcity_logs` internal properties.
9. Restart the server or deploy the application via the servlet container administration interface and access http://server:port/TeamCity/, where "TeamCity" is the name of the `war` file.

TeamCity J2EE container distribution is tested to work with Tomcat 7 servlet container. (See also Supported Platforms and Environments#The TeamCity Server)

> ⚠ If you're using Tomcat J2EE container, make sure the Apache Portable Runtime feature of this container is disabled (actually it is disabled by default). Otherwise due to issues in the Apache Portable Runtime, TeamCity may not work properly.

Unattended TeamCity server installation

For automated server installation, use the .tar.gz distribution.

Typically, you will need to unpack it and make the script perform the steps noted in Configuring Server for Production Use section.

If you want to get a pre-configured server right away, put files from a previously configured server into the Data Directory. For each new server you will need to ensure it points to a new database (configured in <Data Directory>\config\database.properties) and change `<Data Directory>\config\main-config.xml` file not to have the "uuid" attribute in the root XML element (so new one can be generated) and setting appropriate value for "rootURL" attribute.

Using another Version of Tomcat

To use another version of the Tomcat web server instead of the one bundled in the .tar.gz and .exe distributions), you can either use the .war TeamCity distribution (not recommended) or perform the Tomcat upgrade/patch for TeamCity installed from the .exe or .tar.gz distributions.

For the latter, you might want to:

- backup the current TeamCity home
- delete/move out the directories from the TeamCity home which are also present in the Tomcat distribution
- unpack the Tomcat distribution into the TeamCity home directory
- copy TeamCity-specific files from the previously backed-up/moved directories to the TeamCity home. Namely:
  - files under `bin` which are not present in the Tomcat distribution
  - review differences between the default Tomcat `conf` directory and one from TeamCity, update Tomcat files with TeamCity-specific settings (teamcity-* files, and portions of server.xml)
  - delete the default Tomcat `webapps/ROOT` directory and replace it with the one provided by TeamCity

Starting TeamCity server

Under Windows, if TeamCity server is installed as a Windows service, follow the usual procedure of starting and stopping services.

If TeamCity is installed using the .exe or .tar.gz distributions, the TeamCity server can be started and stopped by the `teamcity-server` scripts provided in the `<TeamCity home>/bin` directory.

The script accepts `run` (run in the same console), `start` (start new detached process and exit from the script) and `stop` commands.

- (evaluation only) To start/stop the TeamCity server and one default agent at the same time, use the `runAll` script, e.g.:
  - Use `runAll.bat start` to start the server and the default agent
  - Use `runAll.bat stop` to stop the server and the default agent
- To start/stop the TeamCity server only, use the `teamcity-server` scripts and pass the required parameters. Start the script without parameters to see the usage instructions.
  The `teamcity-server` scripts support the following options for the `stop` command:
  - `stop n` - Sends the stop command to the TeamCity server and waits up to n seconds for the process to end.
  - `stop n -force` - Sends the stop command to the TeamCity server, waits up to n seconds for the process to end, and terminates the server process if it did not stop.

The TeamCity server will restart automatically if the server process exits (crashes or is killed) without invoking "`teamcity-server stop`" script.

By default, TeamCity runs on http://localhost:8111/. See the information below for changing the server port.

If you need to pass special properties to the server, refer to Configuring TeamCity Server Startup Properties.

If TeamCity is installed into an existing web server (.war distribution), start the server according to its documentation. Make sure you configure TeamCity-specific logging-related properties and pass suitable memory options.

Autostart TeamCity server on macOS

Starting up TeamCity server on macOS is quite similar to starting Tomcat on macOS.

1. Install TeamCity and make sure it works if started from the command line with **`bin/teamcity-server.sh start`**. We'll assume that TeamCity is installed in the /Library/TeamCity folder
2. Create the `/Library/LaunchDaemons/jetbrains.teamcity.server.plist` file with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
 <key>WorkingDirectory</key>
 <string>/Library/TeamCity</string>
 <key>Debug</key>
 <false/>
 <key>Label</key>
 <string>jetbrains.teamcity.server</string>
 <key>OnDemand</key>
 <false/>
 <key>KeepAlive</key>
 <true/>
 <key>ProgramArguments</key>
 <array>
       <string>/bin/bash</string>
       <string>--login</string>
       <string>-c</string>
       <string>bin/teamcity-server.sh run</string>
   </array>
 <key>RunAtLoad</key>
 <true/>
 <key>StandardErrorPath</key>
 <string>logs/launchd.err.log</string>
 <key>StandardOutPath</key>
 <string>logs/launchd.out.log</string>
</dict>
</plist>
```

3. Test your file by running launchctl load /Library/LaunchDaemons/jetbrains.teamcity.server.plist . This command should start the TeamCity server (you can see this from `logs/teamcity-server.log` and in your browser).
4. If you don't want TeamCity to start under the root permissions, specify the UserName key in the plist file, e.g.:

```
<key>UserName</key>
 <string>teamcity_user</string>
```

The TeamCity server will now start automatically when the machine starts. To configure automatic start of a TeamCity Build Agent, see the dedicated section.

Installation Configuration

Troubleshooting TeamCity Installation

Upon successful installation, the TeamCity server web UI can be accessed via a web browser.
The default address that can be used to access TeamCity from the same machine depends on the installation package and installation options. (Port 80 is used for Windows installation, unless another port is specified, port 8111 for .tar.gz installation unless not changed in the server configuration).

If the TeamCity web UI cannot be accessed, please check:

- the "TeamCity Server" service is running (if you installed TeamCity as a Windows service);
- the TeamCity server process (Tomcat) is running (it is a `java` process run in the `<TeamCity home>/bin` directory);
- the console output if you run the server from a console,
- the `teamcity-server.log` and other files in the `<TeamCity home>\logs` directory for error messages.

One of the most common issues with the server installation is using a port that is already used by another program. See below on changing the default port.

Changing Server Port

If you use the TeamCity server Windows installer, you can set the port to be used during installation.

Use the following instructions to change the port if you use the .tar.gz distribution.
If another application uses the same port as the TeamCity server, the TeamCity server (Tomcat server) won't start and this will be identified by "Address already in use" errors in the server logs or server console.

To change the server port, in the `<TeamCity Home>/conf/server.xml` file, change the port number in the not commented "<Connector>" XML node (here the port number is 8111):

```
<Connector port="8111" ...
```

To apply the changes, restart the server. If the server was working with the old port previously, you would need to change the port in all the stored URLs of the server (browser bookmarks, agents' serverUrl property, URL in user's IDEs, "Server URL" setting on the "Administration | Global Settings" page).

If you run another Tomcat server on the same machine, you might need to also change other Tomcat server service ports (search for "port=" in the server.xml file).

If you want to use the https:// protocol, it should be enabled separately and the process is not specific to TeamCity, but rather for the web server used (Tomcat by default). See also Configure HTTPS for TeamCity Web UI

Changing Server Context

By default, the TeamCity server is accessible under the root context of the server address (e.g. http://localhost:8111/). To make it available under a nested path instead (e.g. http://localhost:8111/teamcity/), you need to:

- stop the TeamCity server;
- rename the `<TeamCity home>\webapps\ROOT` directory to the `<TeamCity home>\webapps\teamcity`;
- start the TeamCity server.

Note that after this change automatic update will be disabled for your installation and you will have to upgrade TeamCity manually.

Java Installation

The TeamCity server is a web application that runs in an J2EE application server (a JVM application). TeamCity server requires a Java SE JRE installation to run.

The TeamCity server requires JRE 8 to operate, the agent can run with Java 6-10, but Java 8 is recommended. The recommended Java download is AdoptOpenJDK.
It is recommended to use the 32-bit installation unless you need to dedicate more memory to TeamCity server. Please check the 64-bit Java notes before upgrade.
If you configured any native libraries for use with TeamCity (like a .dll for using Microsoft SQL database Integrated Security option), you need to update the libraries to match the JVM x86/x64 platform.

For TeamCity agent Java requirements, check Setting up and Running Additional Build Agents.

TeamCity selects the Java to run the server process as follows:

- If your TeamCity installation has a bundled JRE (there is the `<TeamCity Home>\jre` directory), it will always be used to run the TeamCity server process.

- If there is no <TeamCity Home>\jre directory present, set JRE_HOME or JAVA_HOME environment variables pointing to the installation directories of JRE or JVM (Java SDK) respectively. JRE will be used if both are present.

The necessary steps to update the Java installation depend on the distribution used.

- if your TeamCity installation has a bundled JRE (there is the <TeamCity Home>\jre directory), update it by installing a newer JRE per installation instructions and copying the content of the resulting directory to replace the content of the existing <TeamCity home>\jre directory.

  > ⚠ Note that on upgrade, TeamCity will overwrite the existing JRE with the bundled 32-bit version, so you'll have to update to the 64-bit JRE again after upgrade.

  If you also run a TeamCity agent from the <TeamCity home>\buildAgent directory, install JVM (Java SDK) installation instead of JRE and copy content of JVM installation directory into <TeamCity Home>\jre.
- if there is no <TeamCity Home>\jre directory present, set JRE_HOME or JAVA_HOME environment variables to be available for the process launching the TeamCity server (setting global OS environment variables and system restart is recommended). The variables should point to the home directory of the installed JRE or JVM (Java SDK) respectively and if both are present, the installed JRE will be used.
- if you use the .war distribution, Java update depends on the application server used. Please refer to the manual of your application server.

Using 64 bit Java to Run TeamCity Server

TeamCity server can run under both the 32- and 64-bit JVM.
It is recommended to use the 32-bit JVM unless you need to dedicate more than 1Gb of memory (via -Xmx JVM option) to the TeamCity process (see details) or your database requirements are different.

If you choose to use the 64-bit JVM, note that the memory usage is almost doubled when switching from the 32- to 64-bit JVM, so please make sure you specify at least twice as much memory as for 32-bit JVM, see #Setting Up Memory settings for TeamCity Server.

To update to the 64-bit Java:

- update Java to be used by the server
- set JVM memory options. It is recommended to set the following options for the 64-bit JVM: `-Xmx4g -XX:ReservedCodeCacheSize=350m`

Setting Up Memory settings for TeamCity Server

TeamCity server has the main process which can also launch child processes. Child processes use available memory on the machine, this section covers the memory settings of the main TeamCity server process only as it requires special configuration. As a JVM application, TeamCity main server process only utilizes memory devoted to the JVM. The required memory may depend on the JVM used (32 bit or 64 bit). The memory used by JVM usually consists of: heap (configured via -Xmx) and metaspace (limited by the amount of available native memory), internal JVM (usually tens of Mb), and OS-dependent memory features like memory-mapped files. TeamCity mostly depends on the heap memory and this settings can be configured for the TeamCity application manually by passing -Xmx (heap space) option to the JVM running the TeamCity server.

Once you start using TeamCity for production purposes or you want to load the server during evaluation, you should manually set the appropriate memory settings for the TeamCity server.

To change the memory settings, refer to Configuring TeamCity Server Startup Properties, or to the documentation of your application server, if you run TeamCity using the .war distribution.
Generally this means setting TEAMCITY_SERVER_MEM_OPTS environment variable to the value like -Xmx750m

> ⚠ The Permanent Generation (PermGen) space has been replaced with metaspace memory allocation. It is recommended to remove the -XX:MaxPermSize JVM option from TEAMCITY_SERVER_MEM_OPTS environment variable, if previously configured.

If slowness, OutOfMemory errors occur, or you consistently see a memory-related warning in the TeamCity UI, increase the setting to the next level.

- minimum setting (the 32-bit Java should be used (bundled in .exe distribution)): -Xmx750m
- recommended setting for medium server use (the 32-bit Java should be used): -Xmx1024m. Greater settings with the 32-bit Java can cause an OutOfMemoryError with "Native memory allocation (malloc) failed" JVM crashes or "unable to create new native thread" messages
- recommended setting for large server use (64-bit Java should be used): -Xmx4g -XX:ReservedCodeCacheSize=350m.

These settings should be suitable for an installation with up to two hundreds of agents and thousands of build configurations. Custom plugins installed might require increasing the values defined via the Xmx parameter.
- maximum settings for large-scale server use (64-bit Java should be used): `-Xmx10g -XX:ReservedCodeCacheSize=512m`. Greater values can be used for larger TeamCity installations. However, generally it is not recommended to use values greater than 10g without consulting TeamCity support.

Tips:

- the 32-bit JVM can reliably work with up to 1Gb heap memory (`-Xmx1024m`). (This can be increased to `-Xmx1200m`, but JVM under Windows might crash occasionally with this setting.) If more memory is necessary, the 64-bit JVM should be used with not less than 2.5Gb assigned (`-Xmx2500m`). Unless the TeamCity server runs with more than 100 agents or serves very active builds / thousands of users, it's unlikely that you will need to dedicate more than 4Gb of memory to the TeamCity process.
- A rule of thumb is that the 64-bit JVM should be assigned twice as much memory as the 32-bit for the same application. If you switch to the 64-bit JVM, make sure you adjust the memory setting (`-Xmx`) accordingly. It does not make sense to switch to 64 bit if you dedicate less than the double amount of memory to the application.
- Large TeamCity installations might benefit from more fine tuning of the memory settings. The amount of memory dedicated to TeamCity server JVM should not regularly exceed 60% of the total available physical memory on the machine (to allow for nested process and OS-level caches usage). Also, with heaps (–Xmx) set to more than 8Gb, if the machine has many CPU cores (e.g. more than 8) and current CPU usage is below 60%, enabling G1 JVM garbage collector via "`-XX:+UseG1GC`" JVM option might reduce the length of stop-the-world GC pauses.

The recommended approach is to start with initial settings and monitor for the percentage of used memory using the Administr ation | Diagnostics page. If the server uses more than 80% of memory consistently without drops for tens of minutes, that is probably a sign to increase the `-Xmx` memory value by another 20%.

The license could not be verified: License Certificate has expired!

Configuring TeamCity Server

- If you have a lot of projects or build configurations, we recommend you avoid using the Default agent in order to free up the TeamCity server resources. The TeamCity Administrator can disable the default agent on the Age nts page of the web UI.
- When changing the TeamCity data directory or database, make sure they do not get out of sync.

Configuring TeamCity Data Directory

The default placement of the TeamCity data directory can be changed. See corresponding section: TeamCity Data Directory for details.

Editing Server Configuration

After successful server start, any TeamCity page request will redirect to prompt for the server administrator username and password. Please make sure that no one can access the server pages until the administrator account is setup.

After administration account setup you may begin to create Project and Build Configurations in the TeamCity server. You may also want to configure the following settings in the Server Administration section:

- Server URL
- Email server address and settings
- Jabber server address and settings

Configuring Server for Production Use

Out-of-the-box TeamCity server installation is suitable for evaluation purposes. For production use you will need to perform additional configuration which typically includes:

- Check that the server is using due server port and configure access via https.
- Make sure server URL, email and (optionally) Jabber server settings are specified and are correct
- Configuring the server process for OS-dependent autostart on machine reboot
- Using reliable storage for TeamCity Data Directory

- Using external database
- Configuring recommended memory settings, use "maximum settings" for active or growing servers
- Planning for regular backups
- Planning for regular upgrades to the latest TeamCity releases
- (since TeamCity 10.0.3) Consider adding the `"teamcity.installation.completed=true"` line into the `<TeamCity Home Directory>\conf\teamcity-startup.properties file` - this will prevent the server from creating an administrator user if no such user is found

Please also review the notes on configuring the server for performance and security notes.

See also:

Installation and Upgrade: Setting up and Running Additional Build Agents