

Kanpur 2019.2 EAP1 (build 70479) Release Notes

On this page:

- [Support of EC2 launch templates](#)
- [Code highlighting in build runners' scripts](#)
- [New features of experimental UI](#)
 - [Comparing builds](#)
 - [Extended build preview in build list](#)
- [Other improvements](#)
- [Notes on discontinued Running Builds Node](#)

Support of EC2 launch templates

TeamCity now supports [Amazon EC2 launch templates](#) for cloud instances. Many of the EC2 users appreciate the launch templates since they allow reusing a once defined launch specification for all new instances, which eliminates the need to describe the launch settings every time new instances are requested.

When you are [adding an image](#) in the TeamCity cloud profile connected to the Amazon server, TeamCity automatically detects images and instances available on this server. And now, it also detects existing launch templates. Simply select a template as a Source and specify its version, and TeamCity will request instances based on the template parameters.

Add Image ×

Custom Image

Name:

Source: * ▼
AMI Image or EBS Instance.

Template Version: * ▼


Max instances:
Leave blank to have no limit

Agent pool: ? ▼
The agents will be assigned to the selected pool

[🔑 Hide advanced options](#)

Code highlighting in build runners' scripts

We've added automatic code highlighting and line numbering for scripts of the following build runners: [Command Line](#), [Ant](#), [PowerShell](#), [Docker](#), [NAnt](#), and [Rake](#), as well as for the [configuration](#) of Amazon EC2 images.

To improve readability, you can also apply soft wraps to the code by clicking  next to the input form.

Example of Dockerfile highlighting in TeamCity:

New Build Step

Runner type:

Docker

Runner for Docker commands

Step name:

Optional, specify to distinguish this build step from other steps.

Execute step: ?

If all previous steps finished successfully

Specify the step execution policy.

Docker command:

build push other...

Docker Command Parameters


Dockerfile source: ?

File content

Content: *

```
1 FROM golang:1.11-alpine AS build
2
3 # Install tools required for project
4 # Run `docker build --no-cache .` to update dependencies
5 RUN apk add --no-cache git
6 RUN go get github.com/golang/dep/cmd/dep
7
8 # List project dependencies with Gopkg.toml and Gopkg.lock
9 # These layers are only re-built when Gopkg files are updated
10 COPY Gopkg.lock Gopkg.toml /go/src/project/
11 WORKDIR /go/src/project/
12 # Install library dependencies
13 RUN dep ensure -vendor-only
14
15 # Copy the entire project and build it
16 # This layer is rebuilt when a file changes in the project
   directory
17 COPY /go/src/project/
```

New features of experimental UI

The following features are currently available in the TeamCity experimental UI. To switch to the experimental UI, click  in the upper right corner of the page.

Comparing builds





With the Compare Builds feature, you can select two builds from the same configuration and review their details side-by-side:

- build environment and parameters, such as source branches, revisions, and agents used for running builds
- statistics
- failed, ignored, and passed tests

This feature allows for easier monitoring and is especially helpful when multiple users manage and monitor builds. For example, if a build has no changes in the project code but fails for no obvious reason, you can compare this build with the last successful build and analyze their differences to find the most probable cause of the failure.

To compare a finished build with another, open the action menu of this build in the build list, click Compare with, and select a build for comparison.

Compare Builds

| | | |
|--------|--|--|
| Build | #317  | #313  |
| Status |  Tests failed: 3 (3 new), passed: 7 |  Tests passed: 10 |
| Time | 15s, started at 15 Aug 18:48 | 17s, started at 15 Aug 18:44 |

Show changed only

Params Revisions **Statistics** Tests

| | |
|--|---|
| Build Artifacts Publishing Time 5.151s | Build Artifacts Publishing Time 3.681s |
| Build Checkout Time 1.603s | Build Checkout Time 1.346s |
| Build Duration (all stages) 14.831s | Build Duration (all stages) 17.084s |
| Build Duration (excluding Checkout Time) 6.581s | Build Duration (excluding Checkout Time) 11.865s |
| BuildTestStatus 3 | BuildTestStatus 1 |
| Number of Failed Tests 3 | Number of Passed Tests 10 |
| Number of Passed Tests 7 | Number of Passed Tests 10 |
| Queue wait reason: No available agents 26.135s | |
| Success Rate 0% | Success Rate 100% |
| Time Spent in Queue 26.148s | Time Spent in Queue 012ms |
| Total Artifacts Size 39.2 KB | Total Artifacts Size 36.1 KB |

Extended build preview in build list

TeamCity now allows you to preview the most important build results directly in the list of builds. On the Build Configuration Home page, click on a build in the list to see the following details:

- all build changes
- build problems and failed tests
- snapshot dependencies

With this preview, you also receive quick access to any tab of the [build results](#).

| Build number | Status | Changes | Agent | Started | Duration |
|--|--------------------------------------|-------------------------|----------|-----------------|----------|
| #317 | ❗ Tests failed: 3 (3 new), passed: 7 | 44381451+kse1920: 2 ▾ | 👤 Agent2 | 15 Aug 19 18:48 | 15s |
| <div style="display: flex; border-left: 1px solid #ccc; padding-left: 10px;"> <div style="width: 20%; border-right: 1px solid #ccc; padding-right: 10px;"> <p>Overview</p> <p>Changes 2</p> <p>Tests</p> <p>Build Log ↓</p> <p>Parameters</p> <p>Artifacts</p> <p>Maven Build Info</p> </div> <div style="padding-left: 10px;"> <p>In queue: 26s</p> <p>Triggered 4 days ago by: you</p> <p>1 Problem</p> <p>3 failed tests detected ▾</p> <p>3 Tests Failed 3 new</p> <ul style="list-style-type: none"> <input type="checkbox"/> ru.apache_maven.AppTest.testFailed2 <input type="checkbox"/> ru.apache_maven.AppTest.testFailed3 <input type="checkbox"/> ru.apache_maven.AppTest.testFailed4 <p>✅ 7 passed</p> <div style="display: flex; gap: 10px;"> <input type="button" value="Investigate..."/> <input type="button" value="Fix..."/> </div> <p>2 Changes</p> <p>Update AppTest.java 44381451+kse1920 1 file ▾</p> <p>Update AppTest.java 44381451+kse1920 1 file ▾</p> </div> </div> | | | | | |
| #313 | ✅ Tests passed: 10 | No changes | 👤 Agent2 | 15 Aug 19 18:44 | 17s |
| #312 | ✅ Tests passed: 10 | No changes | 👤 Agent2 | 15 Aug 19 18:44 | 19s |

Other improvements

- **Optimized multinode setup**
 Since this EAP, agents can load server-side patches from secondary nodes – not only from the main server, as it was before.
 Server-side patches are mostly used when an agent cannot find a VCS client executable (for example, Git or Perforce) on an agent machine. In this case, the agent will request the server to create a patch with VCS changes and send it to the agent. Now, if you assign the “VCS repositories polling” responsibility to a secondary node, the agents will be able to request patches from this node as well, which significantly reduces the load on the main server.
- **Improved Docker integration**
 Once the [Docker Compose](#) build runner creates a Docker image on an agent, it is stored on this agent to be used for the later builds. Now, TeamCity will automatically detect that the Dockerfile, used for building the image, has changed and will force the Docker Compose step to rebuild this image.
- **All fixed issues**

Notes on discontinued Running Builds Node

The [Running Builds Node](#) is no longer supported since this EAP. In a [multinode setup](#), you can instead configure a secondary node with the “[Processing data produced by running builds](#)” responsibility.

Note that the secondary node offers more features than the Running Builds Node and thus might require as many hardware resources as a regular TeamCity server. Refer to [Estimate Hardware Requirements for TeamCity](#) for notes on the recommended hardware.