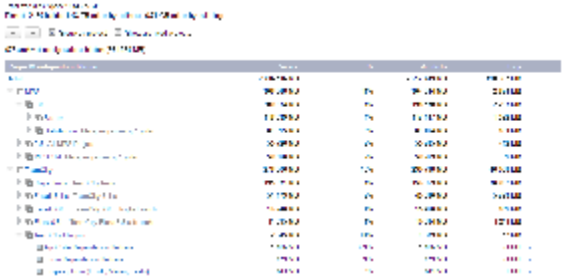
Gaya 8.0 EAP (build 27147) Release Notes

Project groups

Several important improvements and bug fixes were made in project groups feature.

- "Configure Visible Projects" dialog is now aware of project hierarchy.
- Projects hierarchy is shown on Administration | Projects page in the same way as it is shown in the Projects popup.
- Projects hierarchy is shown on Administration | Disk usage page.



- Projects hierarchy is shown in all drop downs where a list of projects or configurations is displayed.
- Investigations page for a project also shows all investigations from sub-projects. The same applies to muted problems, current problems, build chains and project change log pages.
- Notification rule defined for a project will be effective for sub-projects too.
- Builds schedule page for a project shows schedule from sub-projects too.
- It is now possible to define cleanup rules for projects, such cleanup rules serve as default for configurations from this project and its sub-projects. Default cleanup rule is now defined in the Root project.
- Remote run dialogs updated in IntelliJ IDEA plugin and Visual Studio add-in to accommodate for nested projects.
- The list of VCS roots in administration UI has been moved to the "VCS Roots" tab of a project. To view all VCS roots available on the server, see "VCS Roots" tab of the Root project.

VCS roots and templates

There is no notion of shared VCS root anymore. If a VCS root should be available to several projects it should be moved to the common parent of these projects or to the Root project. Therefore, when you upgrade to this EAP build, all shared VCS roots will be moved to the Root project.

The same applies to templates - build configuration can be attached to a template if the template belongs to the parent project. Starting from this EAP you won't be able to use a template from a separate project hierarchy. However, as this was possible before, TeamCity will continue to support this case and will load such configurations without errors.

External IDs for build configuration and template

It is now possible to define external ID for build configuration or template. It works the same way as external ID for project, i.e. it is used in URLs instead of internal ID, and it is used in configuration files. However, with build configuration external IDs add some benefits:

- if you're using dependency parameters, you can use external IDs there instead of cryptic btXXX: `dep.<my build configuration external id>.build.number`
- you can access build artifacts using a URL like this: `http://<host>/repository/download/<my build configuration external id>/lastSuccessful/artifact.zip`
- external ID can be used when build is triggered by HTTP request and in REST requests

Read more about using external IDs when accessing server by HTTP in our documentation: Accessing Server by HTTP, Patterns

For Accessing Build Artifacts

Project configuration files changes

We continue refactoring the way configuration files are stored on disk under TeamCity data directory. This EAP brings two important changes:

- from now on, build configuration internal ID is no longer stored in project configuration files, external ID is used instead. The same applies to build configuration template.
- since all VCS roots now belong to projects, we removed the global `vcs-roots.xml` file and moved VCS roots settings to corresponding projects.

Meta-runner

Meta-runner is a Build Runner consisting of one or more build steps, parameters and requirements and having user interface targeted to user domain. Let's see how it works.

Sometimes during the build process we need to replace some pattern in several files before making a distribution package. For example, if you build a TeamCity plugin you need to provide build number in teamcity-plugin.xml file bundled with the plugin. So this build runner should have the following parameters:

- pattern to search in files
- replacement string
- patterns for files where replacement must be performed

First of all I need to create a build configuration with Ant build step. I will use Ant built-in replaceregexp task to do all the work, so my build.xml will look like this:

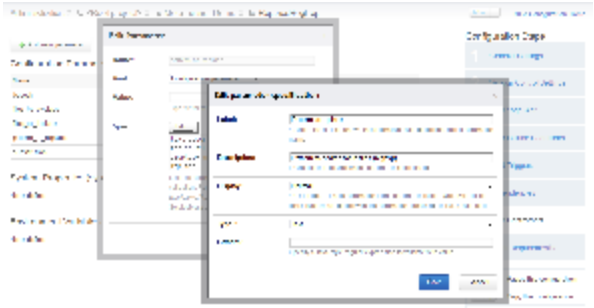
```
<project default="replace" name="ReplaceRegExp">

<target name="replace">

<replaceregexp flags="g">
  <regexp pattern="%pattern_to_replace%"/>
  <substitution expression="%substitution%"/>
  <fileset dir="%basedir%">
    <include name="%files_to_include%"/>
    <exclude name="%files_to_exclude%"/>
  </fileset>
</replaceregexp>

</target>

</project>
```

Note that in several places I used parameter references:

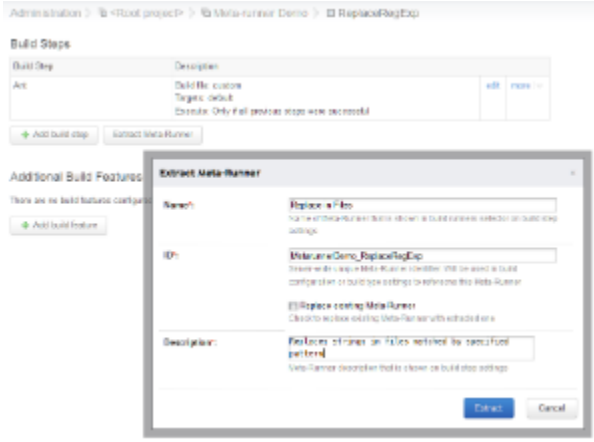
- %pattern_to_replace% - regexp pattern to search in files
- %substitution% - substitution string
- %basedir% - base directory where to perform replacement
- %files_to_include% and %files_to_exclude% - Ant include / exclude patterns for files

In the build configuration where this Ant build step is defined I will see these parameter references as undefined parameters on Build parameters tab, because no one provided values for them.

Now we need to associate labels and descriptions with those parameters, this can be done through the regular parameter editing dialog.



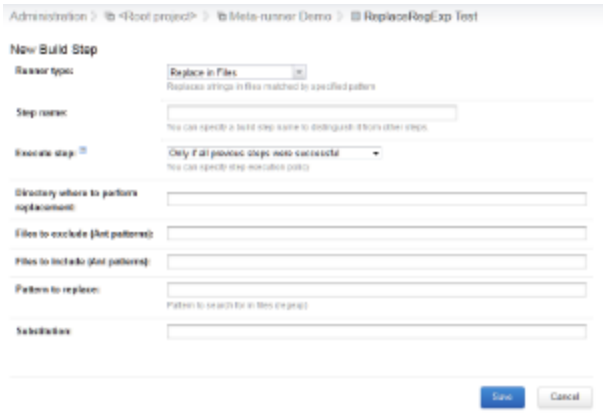
Once we described the parameters we can extract a meta-runner from the build steps page of the build configuration:



We need to provide the Name, ID and Description of the runner. Name and Description will be shown in web interface, ID is required to distinguish this meta-runner from others.

Upon clicking on "Extract" button, TeamCity will take definitions of all build steps, parameters, and requirements in this configuration and create a build runner out of them. In this example we did not use requirements, so extracted runner will not have them. However if runner depends on some specific software installed on agent, we could define requirements in configuration for this software, and these requirements would be extracted in meta runner too.

After successful extraction of the runner it can be used in any build configuration:



Definitions of meta runners are stored under `TeamCity Data Directory`/config/_meta_runners/ directory. Here is complete definition for Replace in Files build runner:

```
<?xml version="1.0" encoding="UTF-8"?>
<meta-runner name="Replace in Files">
  <description>Replaces strings in files matched by specified pattern</description>
  <settings>
    <parameters>
      <param name="basedir" value="" spec="text display='normal' label='Directory where to perform
replacement:'" />
      <param name="files_to_exclude" value="" spec="text display='normal' label='Files to exclude
(Ant patterns):'" />
      <param name="files_to_include" value="" spec="text display='normal' label='Files to include (Ant
patterns):'" />
      <param name="pattern_to_replace" value="" spec="text description='Pattern to search for in
files (regexp)' display='normal' label='Pattern to replace:'" />
      <param name="substitution" value="" spec="text display='normal' label='Substitution:'" />
    </parameters>
    <build-runners>
      <runner name="" type="Ant">
        <parameters>
          <param name="build-file"><![CDATA[<project default="replace" name="ReplaceRegExp">

<target name="replace">

<replaceregexp byline="true">
  <regexp pattern="%pattern_to_replace%"/>
  <substitution expression="%substitution%"/>
  <fileset dir="%basedir%">
    <include name="%files_to_include%"/>
    <exclude name="%files_to_exclude%"/>
  </fileset>
</replaceregexp>

</target>

</project>]]></param>
          <param name="build-file-path" value="build.xml" />
          <param name="teamcity.coverage.emma.include.source" value="true" />
          <param name="teamcity.coverage.emma.instr.parameters" value="-ix -*Test*" />
          <param name="teamcity.coverage.idea.includePatterns" value="*" />
          <param name="teamcity.step.mode" value="default" />
          <param name="use-custom-build-file" value="true" />
        </parameters>
      </runner>
    </build-runners>
    <requirements />
  </settings>
</meta-runner>
```

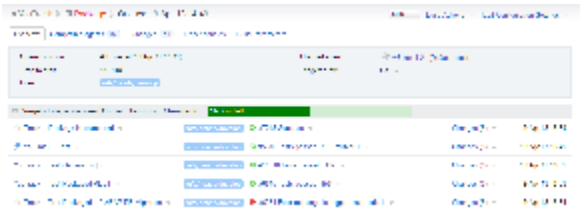
In order to install this build runner in your TeamCity 8.0 installation you need to save this definition to a file under `TeamCity Data Directory/config/_meta_runners/` directory. File should have name like:
<runner id>.xml, where <runner id> is unique identifier of this build runner. Server will detect this definition and will load it on the fly.

Current limitations:

- when meta-runner is extracted from the web interface there is no way to specify which steps you want to extract - all steps will be extracted
- extract meta-runner UI does not allow to reorder parameters, they are sorted alphabetically so far
- meta-runner definitions are global for the server, there is no way to store them in some project only

Queued build page

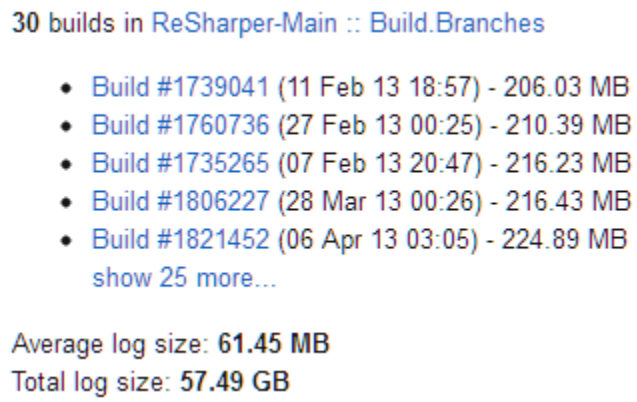
Dependencies progress was added on queued build page, you can now see estimates for all dependencies in one place.



Health status report

Large build logs (based on disk usage report)

This report shows builds with large log files. Large (hundreds of megabytes) build logs are rarely useful, it is hard to analyze them and in most of cases they just waste space on disk. We hope this report will help to find configurations that produce such log files and fix them.



Swabra clean checkout

This report finds incorrect settings of Swabra build feature which can cause unnecessary clean checkout.

Branch filters in triggers

VCS, Schedule and Finish build triggers have got a new setting: Branch filter. With the help of the new filter you can limit the set of branches where automatic triggering will be performed. Read more about branch filters in our documentation: Working with Feature Branches#Triggers

Mercurial subrepo improvements

We continue improving usability of Mercurial subrepo changes. Changes from subrepositories are marked with an icon:

On this screenshot you can see another new feature: dashed lines that visually connect changes in the main repository to changes in subrepositories.

Last but not least, change graph can be collapsed into a single line:

Expanded	Collapsed
	

Other

- Schedule trigger time can be set in a specific timezone.
- Builds schedule report allows to filter data by sub-projects
- Build problems UI has got several improvements, notifications for build problems investigations are now supported
- Pin and tag status is shown for builds on the change log page
- All fixed issues (http://youtrack.jetbrains.com/releaseNotes/TW?q=%23fixed+Fix+versions%3A+\{Gaya+8.0+EAP3+%2827147%29\}+-\{trunk+issue\}+-Documentation+-Internal&title=Gaya+8.0+EAP+%2827147%29&showDescription=false&showComments=false)