ProcessTreeKiller

This feature is available since 1.260

To reliably kill processes spawned by a job during a build, Jenkins contains a bit of native code to list up such processes and kill them. This is tested on several platforms and architectures, but if you find a show-stopper problem because of this, you can disable this feature by setting a Java property named "hudson.util.ProcessTree.disable" to the value "true".

This can be done as a parameter to the "java" binary when starting Jenkins:

```
java -Dhudson.util.ProcessTree.disable=true -jar jenkins.war
```

Depending on how you run your container, this can be different. In the distributed build environment, this system property needs to be set on slave JVMs.

Older versions of Hudson (<1.315) used the Java Property `hudson.util.ProcessTree`**`Killer`**`.disable`, but the class ProcessTreeKiller has been depecated since. For compatibility reasons, currently both property names are supported (as of version 1.404).

How it works

The ProcessTreeKiller takes advantage of the fact that by default a new process gets a copy of the environment variables of its spawning/creating process.

It sets a specific environment variable in the process executing the build job. Later, when the user requests to stop the build job's process it gets a list of all processes running on the computer and their environment variables, and looks for the environment variable that it initially set for the build job's process.

Every job with that environment variable in its environment is then terminated.

If your build wants to leave a daemon running behind...

A convenient way to achieve that is to change the environment variable BUILD_ID which Jenkins's ProcessTreeKiller is looking for. This will cause Jenkins to assume that your daemon is not spawned by the Jenkins build. For example:

```
BUILD_ID=dontKillMe /usr/apache/bin/httpd
```

In case of Jenkins Pipeline use `JENKINS_NODE_COOKIE` instead of `BUILD_ID`