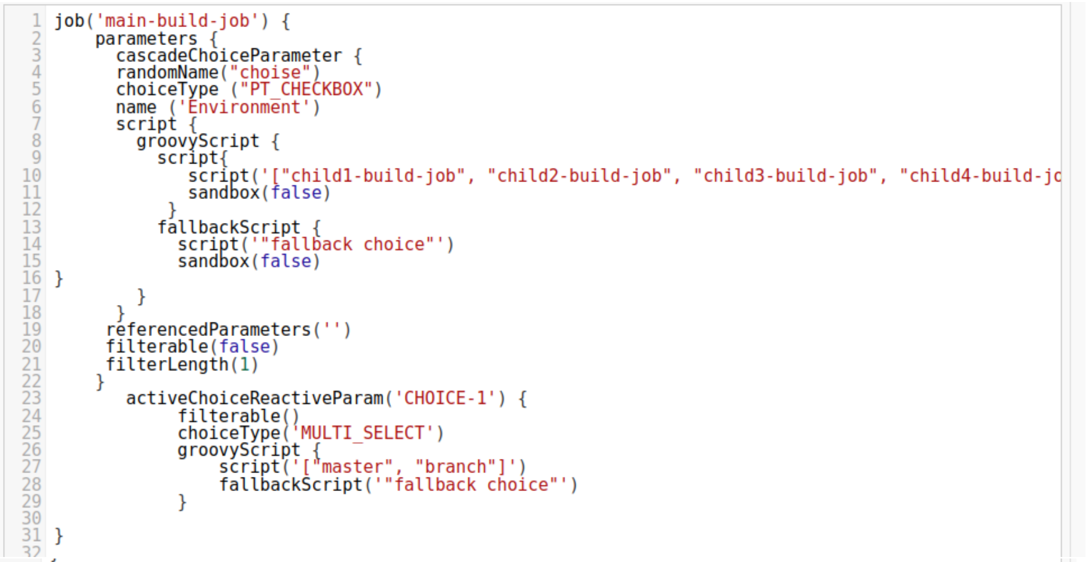
Gödel Technologies

Task 1. Automated Deployments

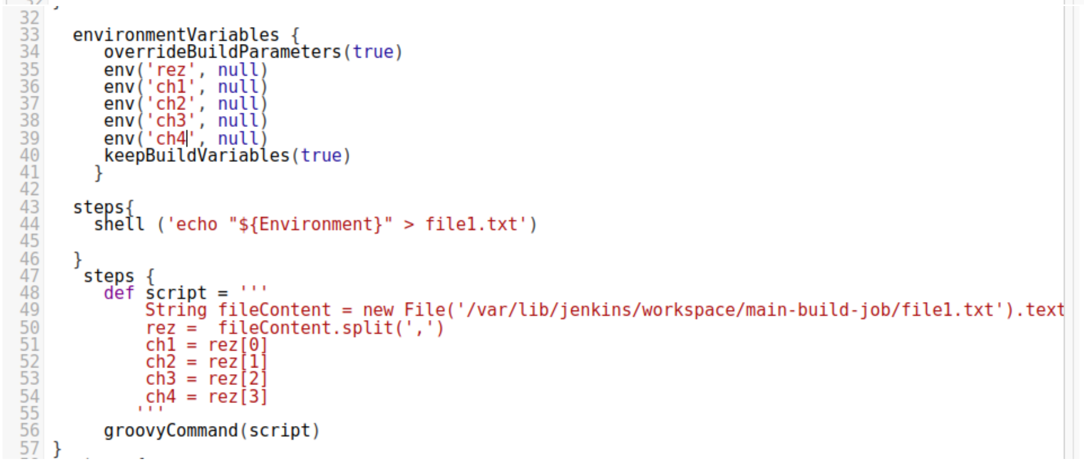
1) Install and configure DSL Plugin for the local Jenkins.
2) Create jobs.groovy file (DSL script) for automated creation the following Jenkins jobs: 1 main and 4 child.
3) All scripts are stored in github, Each student commits all step-by-step changes in own branch
4) Requirements for the jobs:

<u>The main job :</u>

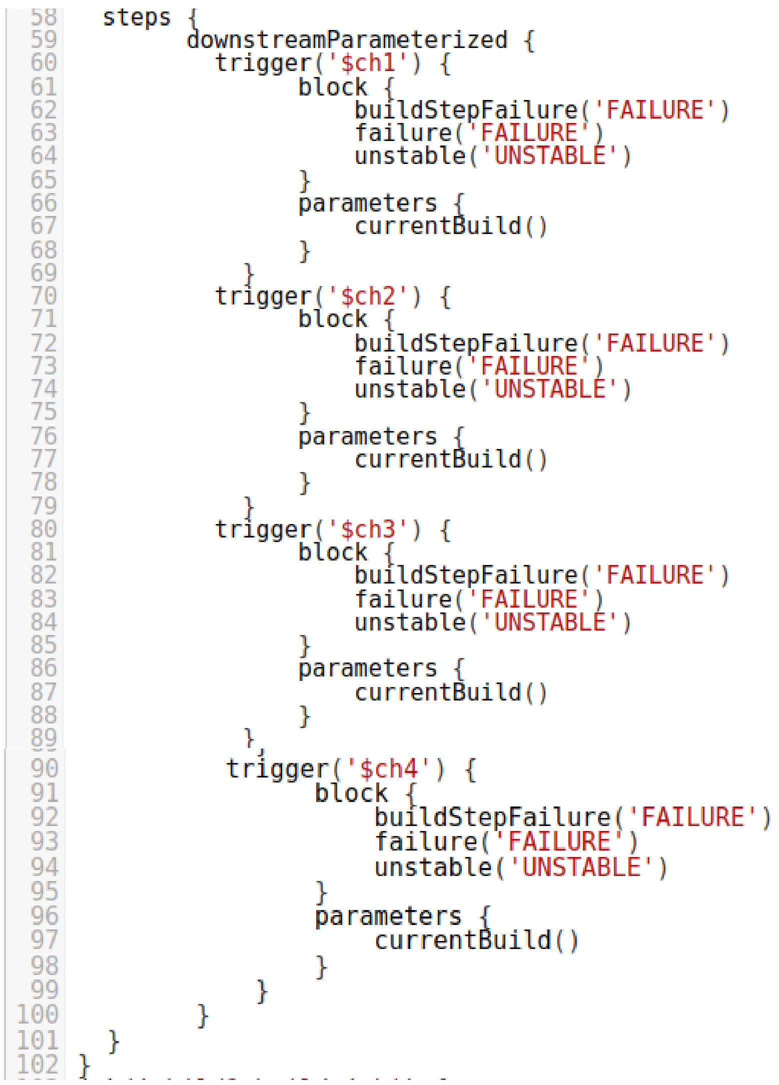
This job is required to trigger the rest four from one place. It provides ability to choose jobs should be executed (by checkboxes) and predefine string parameter BRANCH_NAME which has two options: branch and master. The main job should wait until all child jobs are executed and should be failing if even one of the triggered jobs is failed.



```
job('main-build-job') {
    parameters {
      cascadeChoiceParameter {
      randomName("choise")
      choiceType ("PT_CHECKBOX")
      name ('Environment')
      script {
        groovyScript {
          script{
             script('["child1-build-job", "child2-build-job", "child3-build-job", "child4-build-jc
             sandbox(false)
           }
          fallbackScript {
            script('"fallback choice"')
            sandbox(false)
}
        }
      }
     referencedParameters('')
     filterable(false)
     filterLength(1)
    }
       activeChoiceReactiveParam('CHOICE-1') {
            filterable()
            choiceType('MULTI_SELECT')
            groovyScript {
                script('["master", "branch"]')
                fallbackScript('"fallback choice"')
            }

}
```



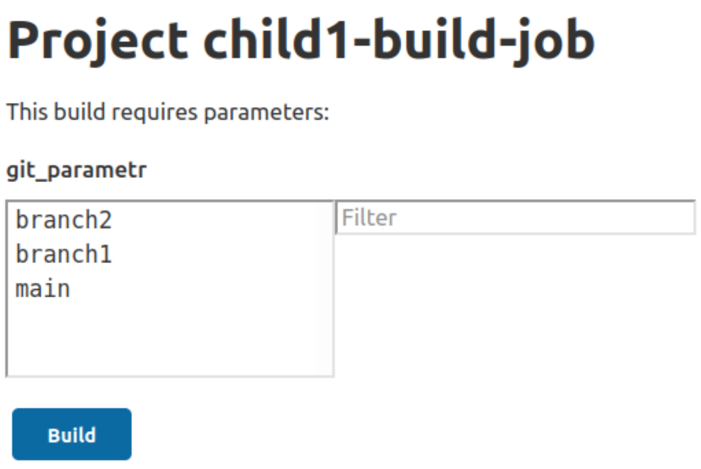
```

  environmentVariables {
     overrideBuildParameters(true)
     env('rez', null)
     env('ch1', null)
     env('ch2', null)
     env('ch3', null)
     env('ch4', null)
     keepBuildVariables(true)
    }

  steps{
    shell ('echo "${Environment}" > file1.txt')

  }
   steps {
     def script = '''
         String fileContent = new File('/var/lib/jenkins/workspace/main-build-job/file1.txt').text
         rez =  fileContent.split(',')
         ch1 = rez[0]
         ch2 = rez[1]
         ch3 = rez[2]
         ch4 = rez[3]
        '''
     groovyCommand(script)
}
```



```
  steps {
        downstreamParameterized {
          trigger('$ch1') {
                block {
                    buildStepFailure('FAILURE')
                    failure('FAILURE')
                    unstable('UNSTABLE')
                }
                parameters {
                    currentBuild()
                }
            }
          trigger('$ch2') {
                block {
                    buildStepFailure('FAILURE')
                    failure('FAILURE')
                    unstable('UNSTABLE')
                }
                parameters {
                    currentBuild()
                }
            }
          trigger('$ch3') {
                block {
                    buildStepFailure('FAILURE')
                    failure('FAILURE')
                    unstable('UNSTABLE')
                }
                parameters {
                    currentBuild()
                }
            },
          trigger('$ch4') {
                block {
                    buildStepFailure('FAILURE')
                    failure('FAILURE')
                    unstable('UNSTABLE')
                }
                parameters {
                    currentBuild()
                }
            }
        }
  }
}
```

The child jobs:

a) Each job has choice parameter BRANCH_NAME with list of all branches available in repo



Project child1-build-job

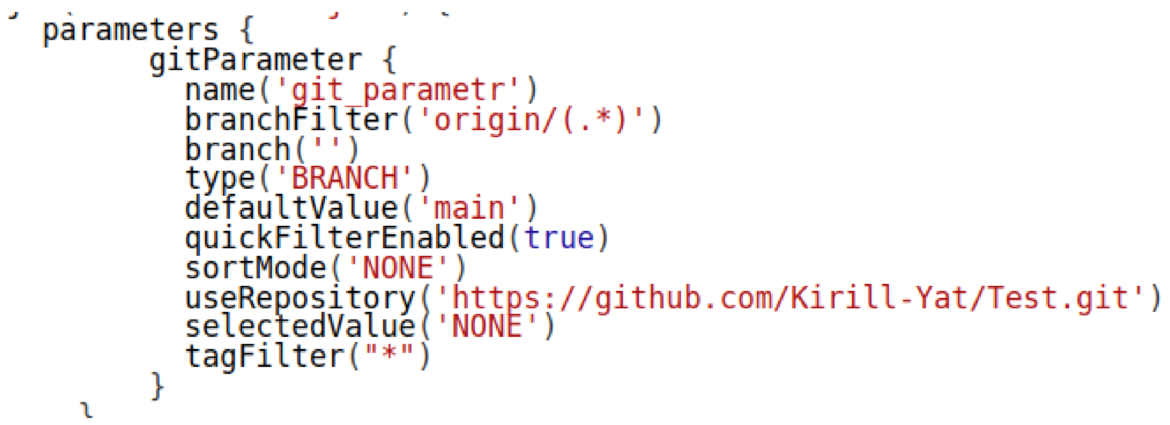
This build requires parameters:

git_parametr

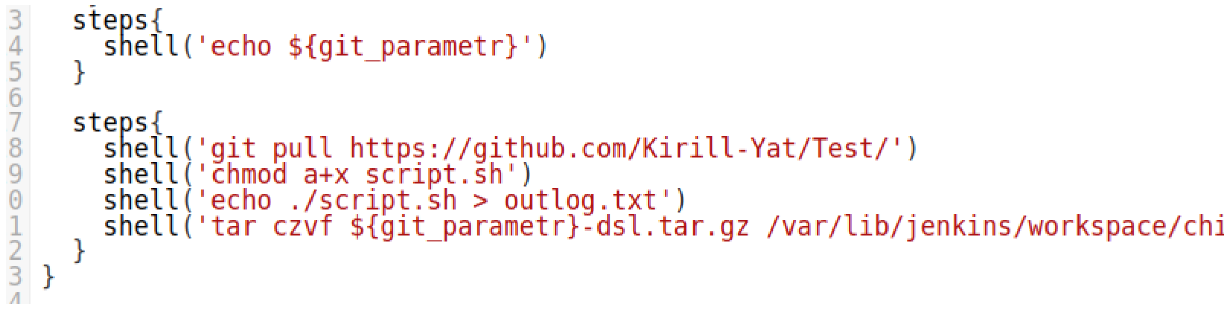
```
branch2
branch1
main
```

b) The job should execute the cloned ‘**script.sh**’ from branch propagated by main job. Output should be saved to the file ‘output.txt’. The file should be attached to job.

c) Templates cloned from appropriate branch should be archived to artefact ($BRANCH_NAME_dsl_script.tar.gz) and attached to each executed child job.



```
 parameters {
        gitParameter {
          name('git_parametr')
          branchFilter('origin/(.*)')
          branch('')
          type('BRANCH')
          defaultValue('main')
          quickFilterEnabled(true)
          sortMode('NONE')
          useRepository('https://github.com/Kirill-Yat/Test.git')
          selectedValue('NONE')
          tagFilter("*")
        }
    }
```

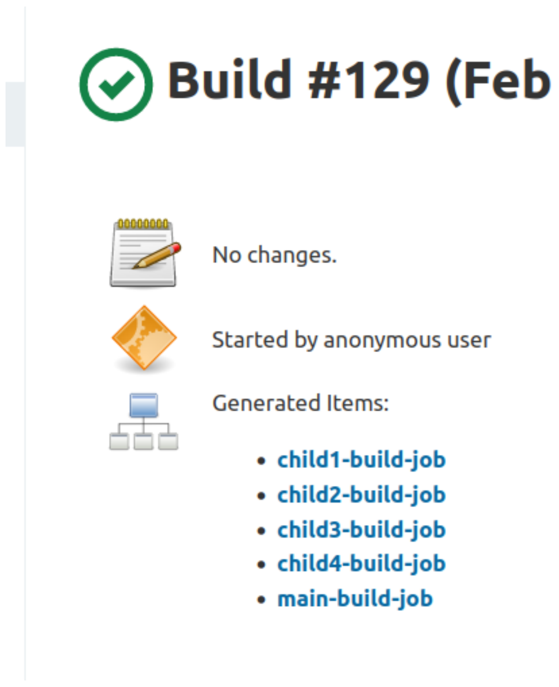


```
  steps{
    shell('echo ${git_parametr}')
  }

  steps{
    shell('git pull https://github.com/Kirill-Yat/Test/')
    shell('chmod a+x script.sh')
    shell('echo ./script.sh > outlog.txt')
    shell('tar czvf ${git_parametr}-dsl.tar.gz /var/lib/jenkins/workspace/chi
  }
}
```

<u>Job name convention:</u>
main-build-job
child1-build-job
child2-build-job
child3-build-job
child4-build-job

Check

1. The requested Jobs are created by automated way in the public Jenkins: 1 main and 4 childs



Build #129 (Feb

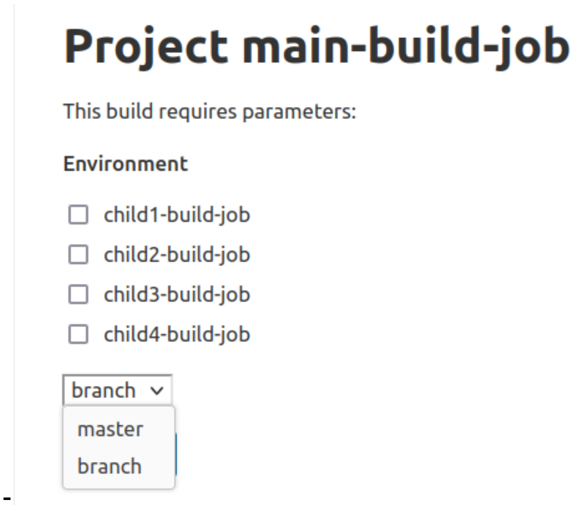
No changes.

Started by anonymous user

Generated Items:

- child1-build-job
- child2-build-job
- child3-build-job
- child4-build-job
- main-build-job

2. Main job provides ability to choose what the child jobs to run and set BRANCH_NAME (student name by default)



Project main-build-job

This build requires parameters:

Environment

☐ child1-build-job

☐ child2-build-job

☐ child3-build-job

☐ child4-build-job

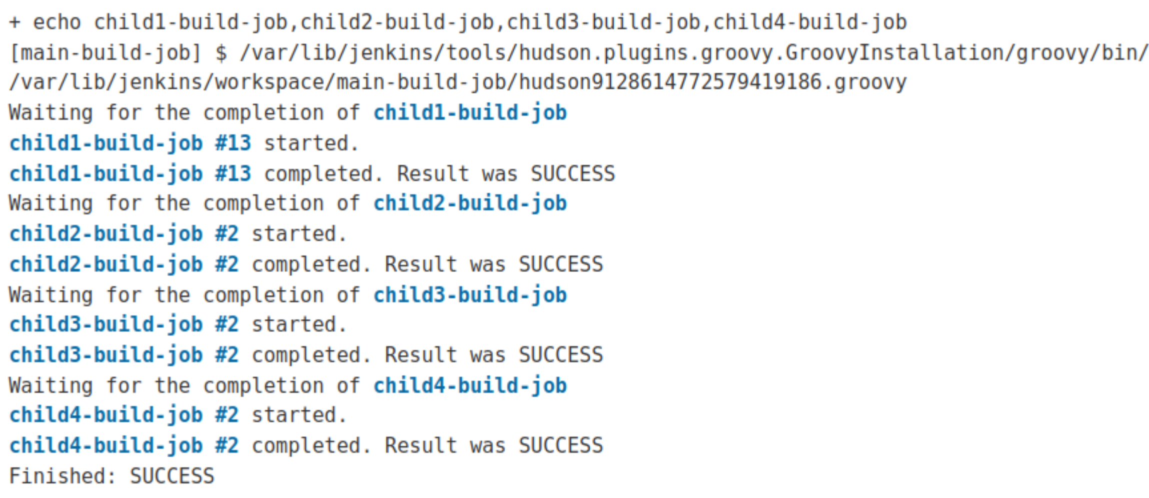
branch

master

branch

-

3. Running the main job triggers appropriate childs.



```
+ echo child1-build-job,child2-build-job,child3-build-job,child4-build-job
[main-build-job] $ /var/lib/jenkins/tools/hudson.plugins.groovy.GroovyInstallation/groovy/bin/
/var/lib/jenkins/workspace/main-build-job/hudson9128614772579419186.groovy
Waiting for the completion of child1-build-job
child1-build-job #13 started.
child1-build-job #13 completed. Result was SUCCESS
Waiting for the completion of child2-build-job
child2-build-job #2 started.
child2-build-job #2 completed. Result was SUCCESS
Waiting for the completion of child3-build-job
child3-build-job #2 started.
child3-build-job #2 completed. Result was SUCCESS
Waiting for the completion of child4-build-job
child4-build-job #2 started.
child4-build-job #2 completed. Result was SUCCESS
Finished: SUCCESS
```

4. Child job creates artefact attached to jobs with dsl script in it.