Отчет по практическим заданиям “ Continuous Integration with Jenkins ”

Martynovich Hanna

1. **I Can Win**

*1. Установите Jenkins*

done

*2. Создайте задачу, которая будет делать следующее:*

* *Клонировать проект*[*https://github.com/vitalliuss/helloci*](https://github.com/vitalliuss/helloci)

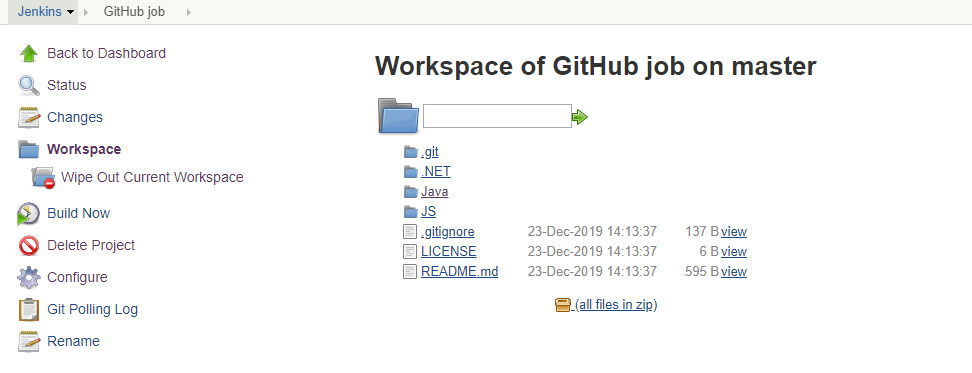
Чтобы создать проект, откроем страницу localhost:8080, выберем **New Item** и укажем имя (*Github job*) и тип *(****Freestyle project***) проекта.

Далее необходимо прописать настройки для работы с *git.* Для этого в *Sourse code Managemen*t указываем адрес репозитория (для этого используем http проекта "Hello, CI").

В *Global Tool Configuration* прописываем расположение нашего git.

С помощью Build Now клонируем проект.

В Workspace можно увидеть, что проект существует.

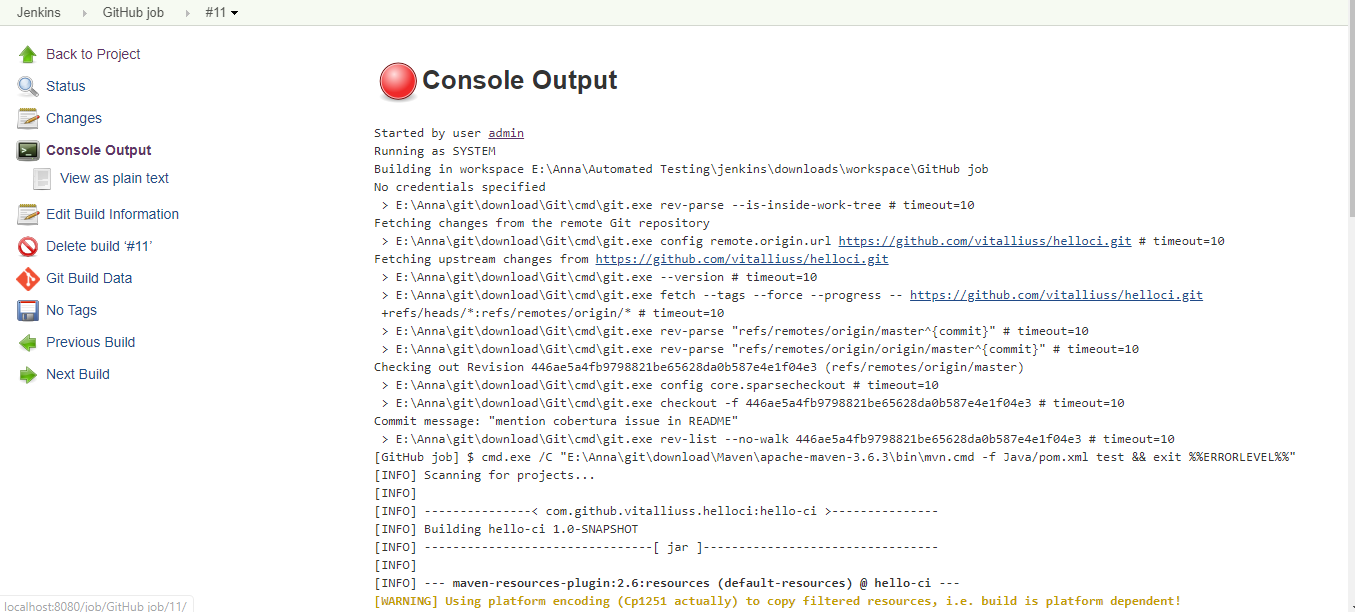


* *Запускать тесты из проекта в директори Java с помощью цели mvn test*

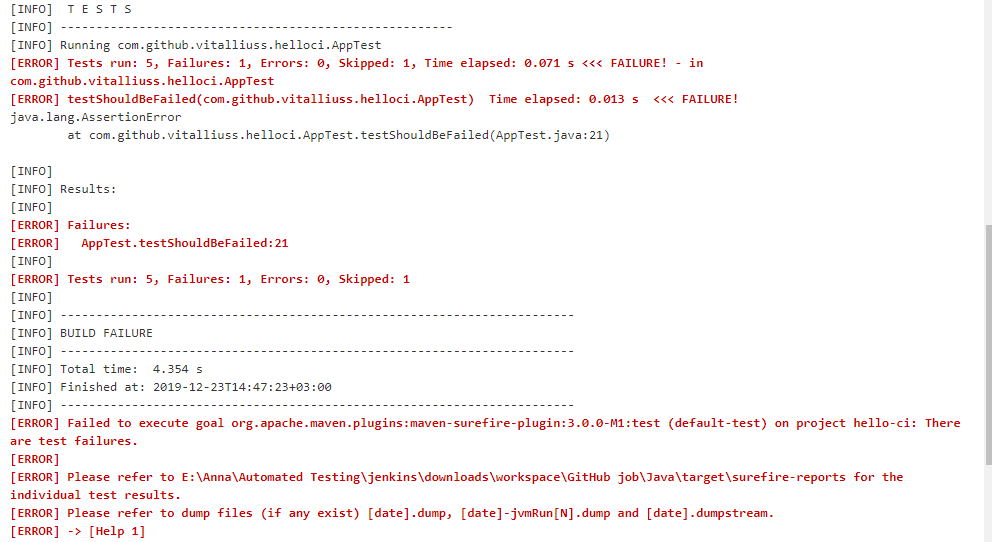
Для этого необходимо в *Global Tool Configuration* прописать расположение нашего maven.

В настройках *Build -> Invoke top-level Maven targets* указываем версию maven и задачу – test, а в *POM* – Java/pom.xml.

С помощью Build Now в Console Output просматриваем выполнение git и maven



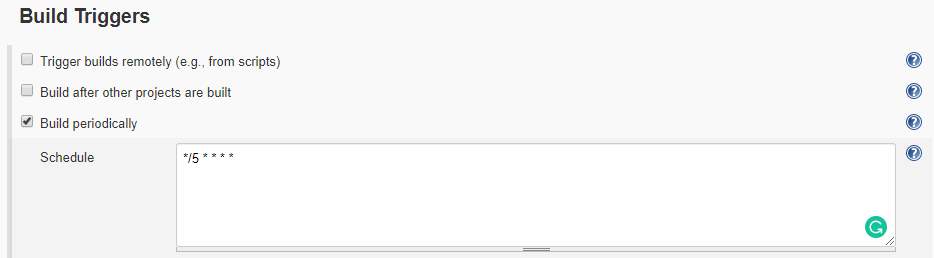
Так как в проекте имеется один “упавший” тест, то результат будет следующим:



*3. Настройте билд триггеры таким образом, чтобы задача выполнялась раз в 5 минут*

Зайдем в настройки Github job, а именно: *Configure -> Build Triggers*. Устанавливаем значок *Build periodically* (сборка по расписанию) и в *Schedule* прописываем:

\*/5 \* \* \* \* - выполнение каждые 5 минут.



1. **Bring It On**

*1. Установите Jenkins*

done

*2. Создайте задачу которая будет делать следующее:*

* *Клонировать проект*[*https://github.com/vitalliuss/helloci*](https://github.com/vitalliuss/helloci)

done

* *Запускать тесты из проекта в директори Java с помощью цели mvn test*

done

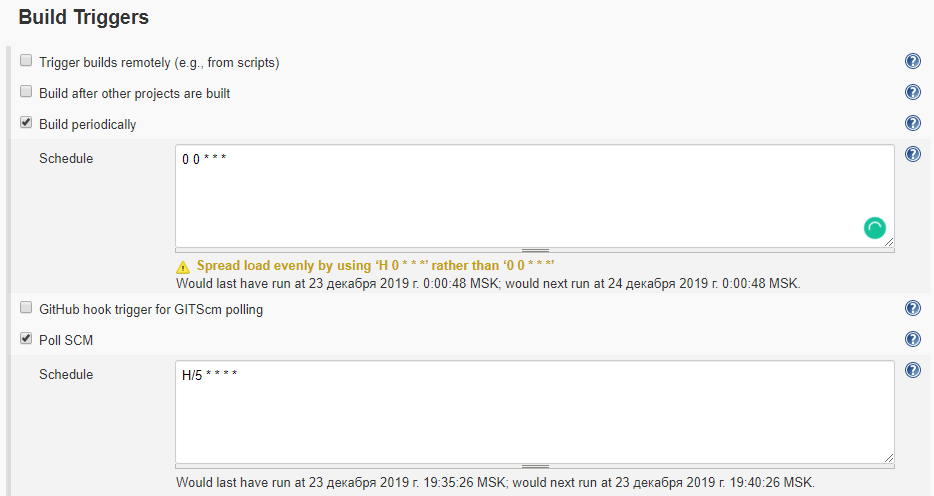
*3. Настройте билд триггеры:*

* *Запуск тестов не позднее чем через 5 минут после коммита в git*

Для это необходимо зайти в *Build Triggers и в Poll SCM* указать H/5 \* \* \* \* (H для распре-деления нагрузки)

* *Каждый будний день в полночь*

Там же, в *Build Triggers, в Build periodically* (сборка по расписанию) и в *Schedule* прописываем: 0 0 \* \* \* - 0 часов и 0 минут (полночь)



*4. Опубликуйте файл “Java\target\surefire-reports\com.github.vitalliuss.helloci.AppTest.txt” как артефакт*

Для этого необходимо открыть *Artifact generator -> Post-build Actions*, в *Add post-build action* и выбрать *Archive the artifacts*. В Files to archive укажем путь к файлу: *Java\target\surefire-reports\com.github.vitalliuss.helloci.AppTest.txt*

1. **Hurt Me Plenty**

*Данное задание выполняется на основе* ***Bring It On*** *задачи.*

*1. Сменить порт сервера на 8081*

С помощью services.msc остановим Jenkins и в загрузочном файле Jenkins откроем файл jenkins.xml, в <arguments> исправим название порта 8080 на 8081

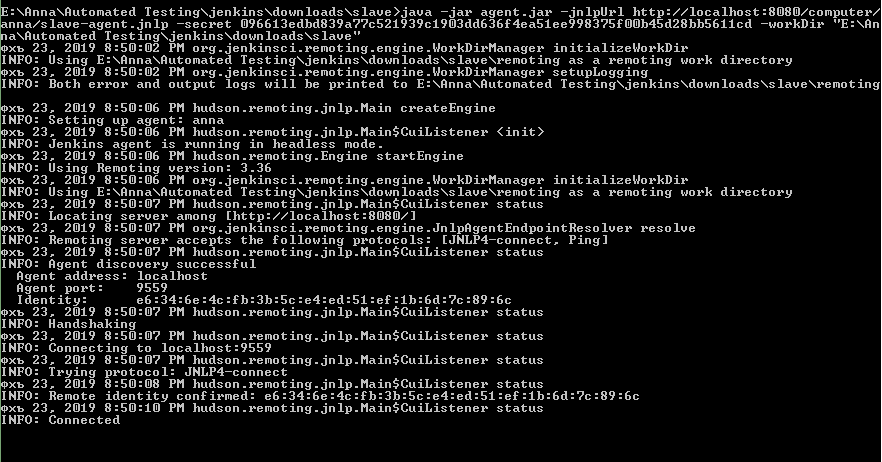
*2. Создать ноду и настроить сервер так, чтобы джоба выполнялась только на slave ноде*

Для начала необходимо выполнить следующее: *Build -> Executor Status -> New Node*.

Здесь укажем имя ноды(anna). В папке E:\Anna\Automated Testing\jenkins\downloads создадим папку slave и разместим файл agent.jar. В *Remote root directory* пропишем путь к файлу slave, а в *Launch Method* выберем *Launch agent by connecting it to the master*.

Скопируем строку, которая располагается в *Run from agent command line,* в командную строку и выполним.

Результат выполнения:



Далее настроим сервер так, чтобы джоба выполнялась только на slave ноде. Для этого зайдем в настройки Github job и в *Restrict where this project can be run* укажем **anna** в *Label Expression*

Для проверки нажмем Build Now и в Console можно увидеть следующее:

Building remotely on [anna](http://localhost:8080/computer/anna) (chrome win8.1 x86) in workspace E:\Anna\Automated Testing\jenkins\downloads\slave\workspace\GitHub job

*3. Настроить Job Config History и thinBackup*

Для установки ThinBackup (выполняет резервное копирование глобальной конфигурации и конкретной конфигурации) и Job Configuration History (позволяет отслеживать историю изменений файлов конфигурации jobs) необходимо зайти в Manage Plugins, добавить сами плагины и указать все необходимые настройки.