Ребята, цель данного туториала – познакомиться с одной из систем контроля версий **git.**

В пределах данного туториала – мы будем работать, исходя из следующего плана:

1. Регистрация на **GitHub**;
2. Разворачивание собственного репозитория на **GitHub**;
3. Установка **GitBash**;
4. Создание простого мавен-проекта Hello world;
5. Демонстрация возможности управления контентом проекта посредством git:

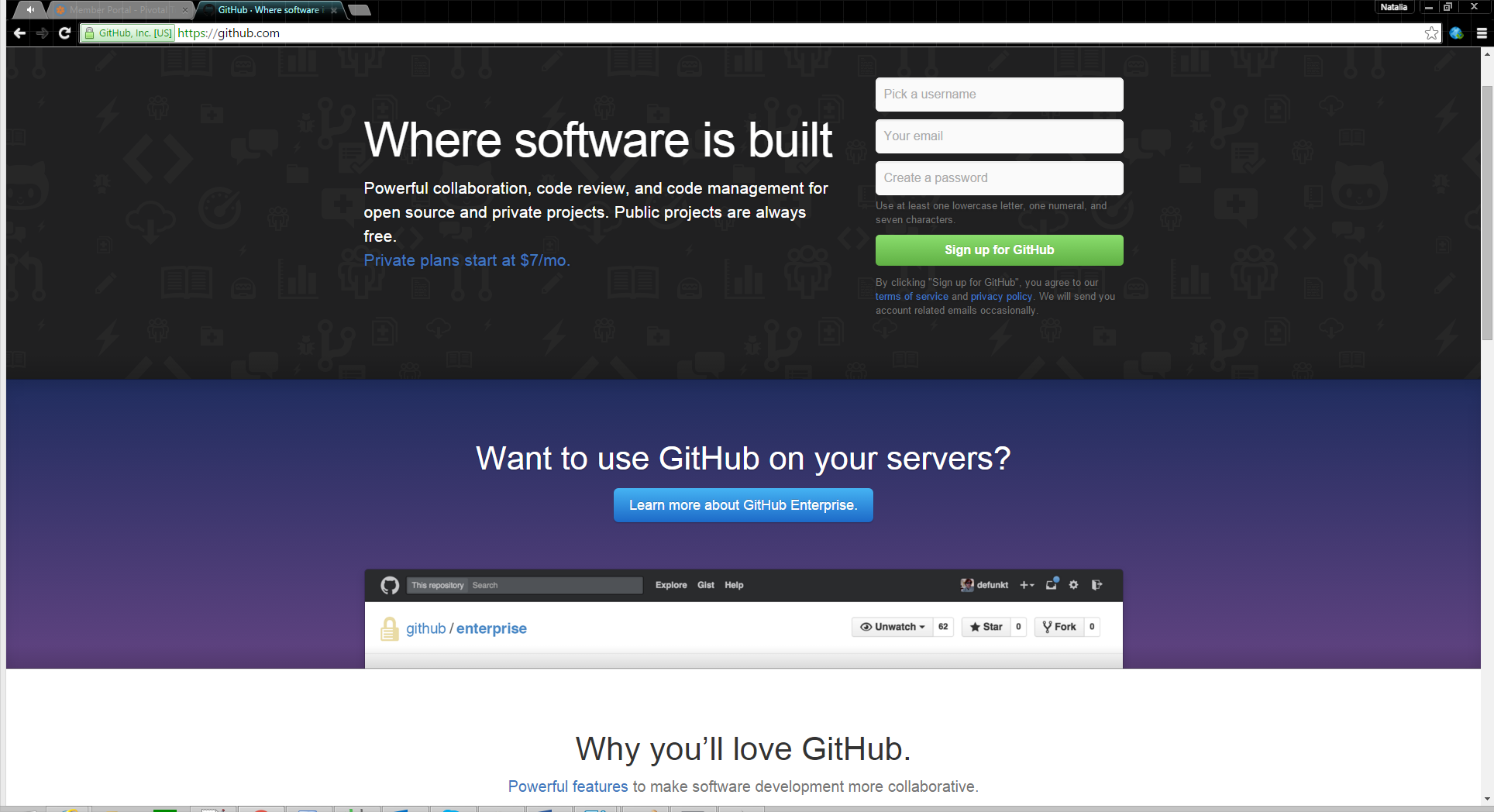
* Создание проекта
* Добавление/модификация структуры проекта
* Аппрув, и фиксирование изменений (работа в **gitbash** консоли)

Начнем по порядку:

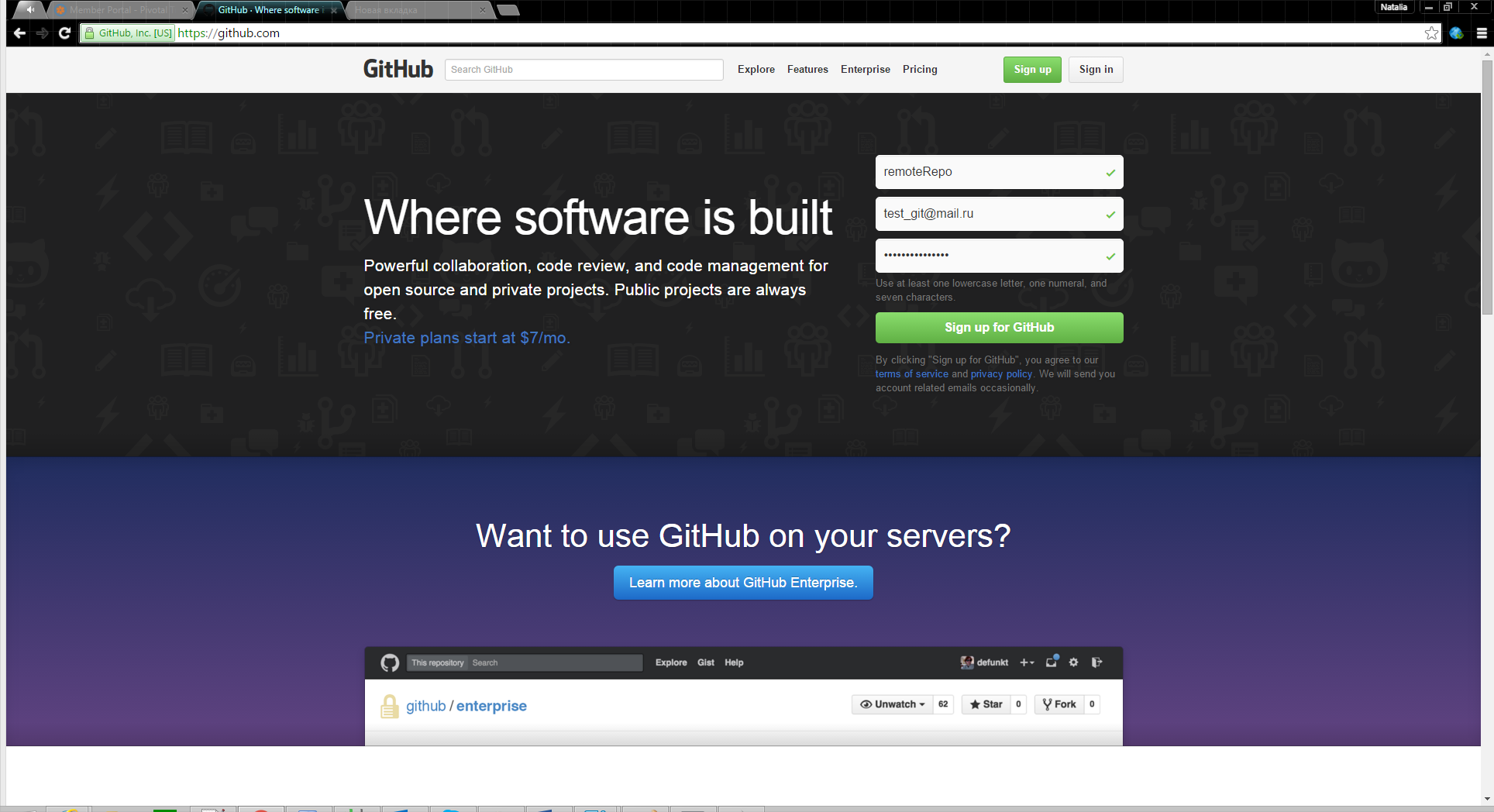
1. Перейдем к выполнению 1-го пункта – а именно регистрация Вашего репозитория на **GitHub**;

Для этого перейдем на официальный сайт **GitHub:**

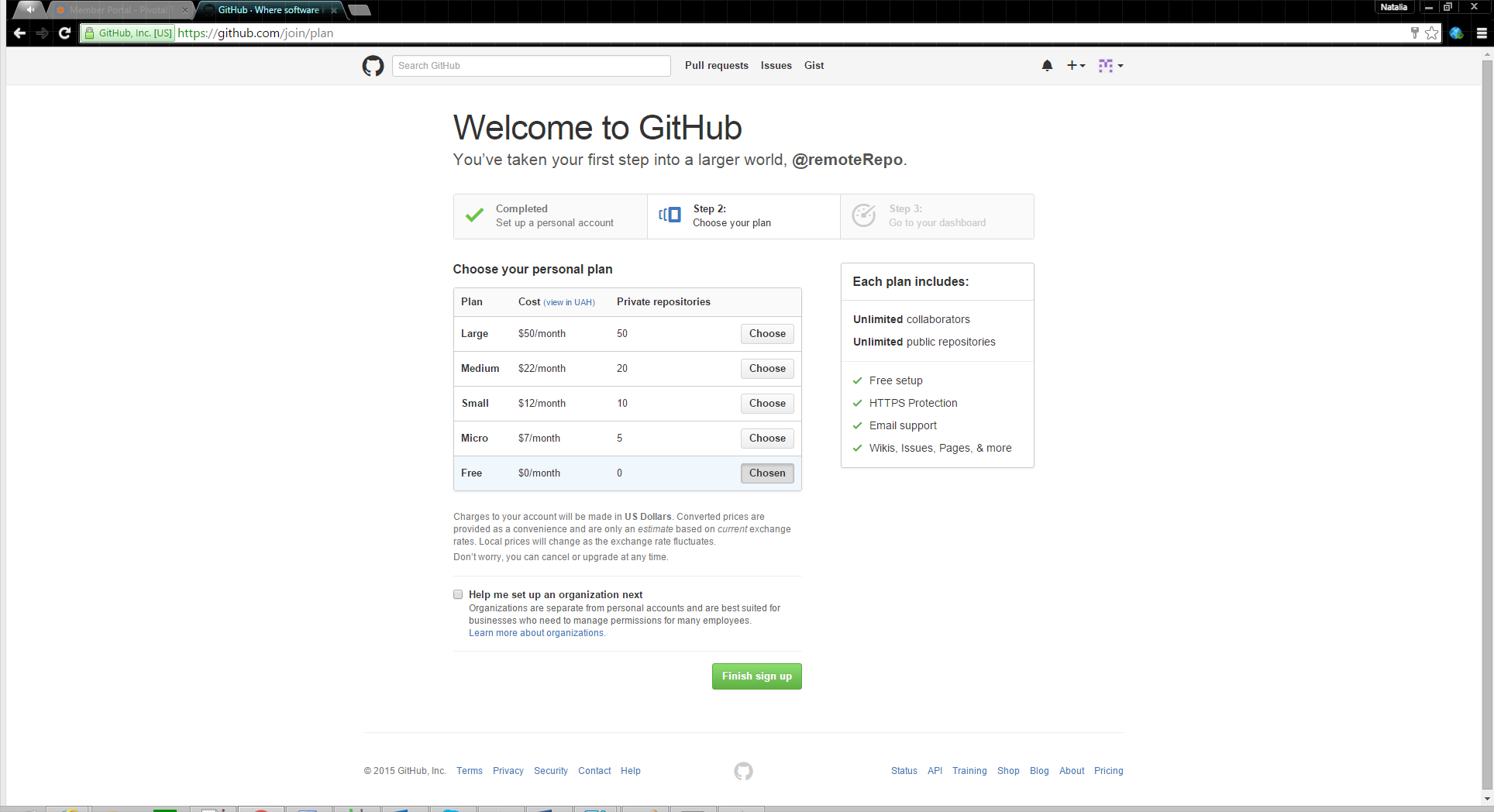
[**https://github.com/**](https://github.com/)



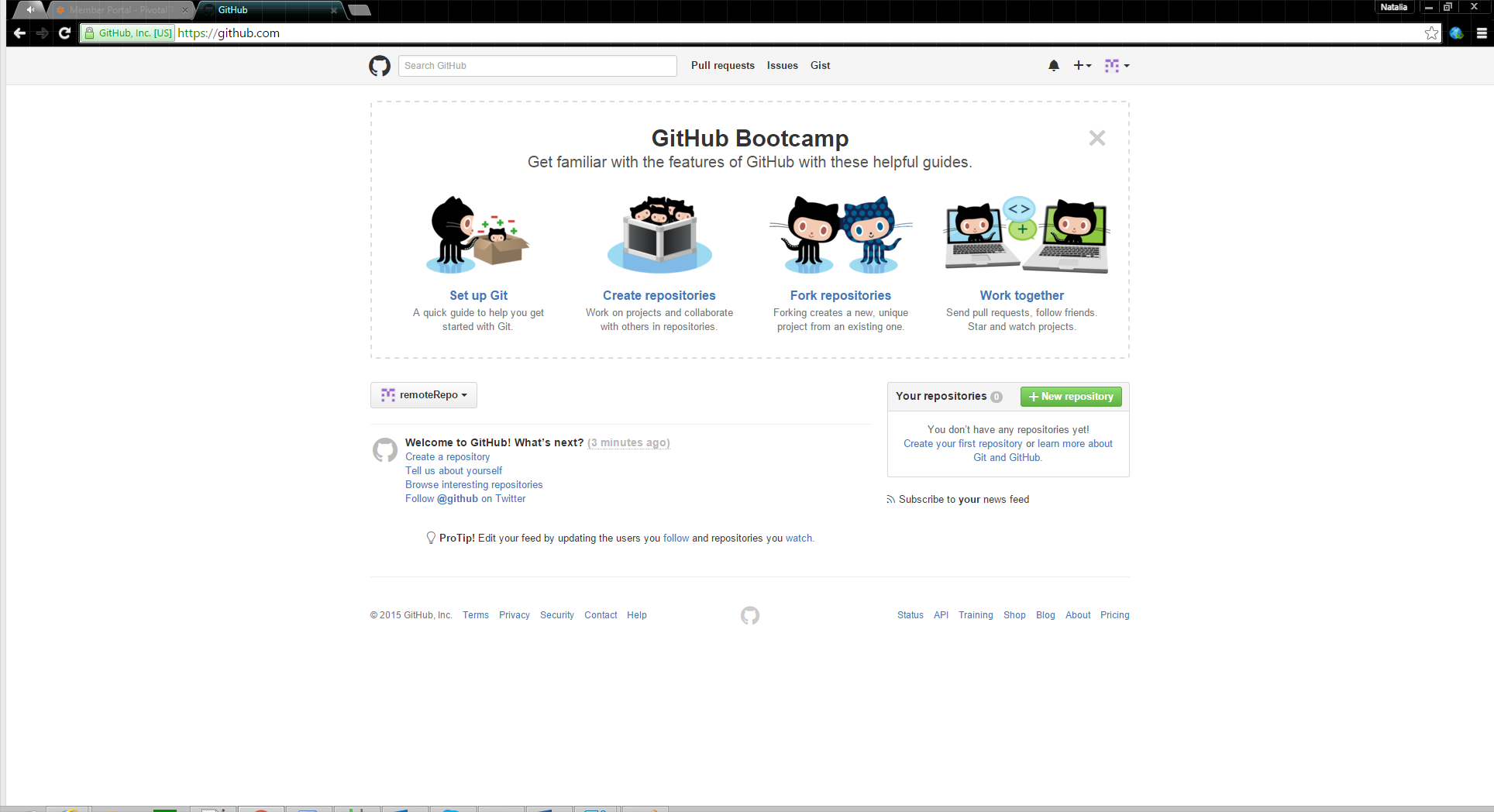
Регистрируемся



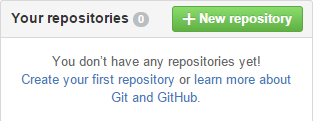
После регистрации



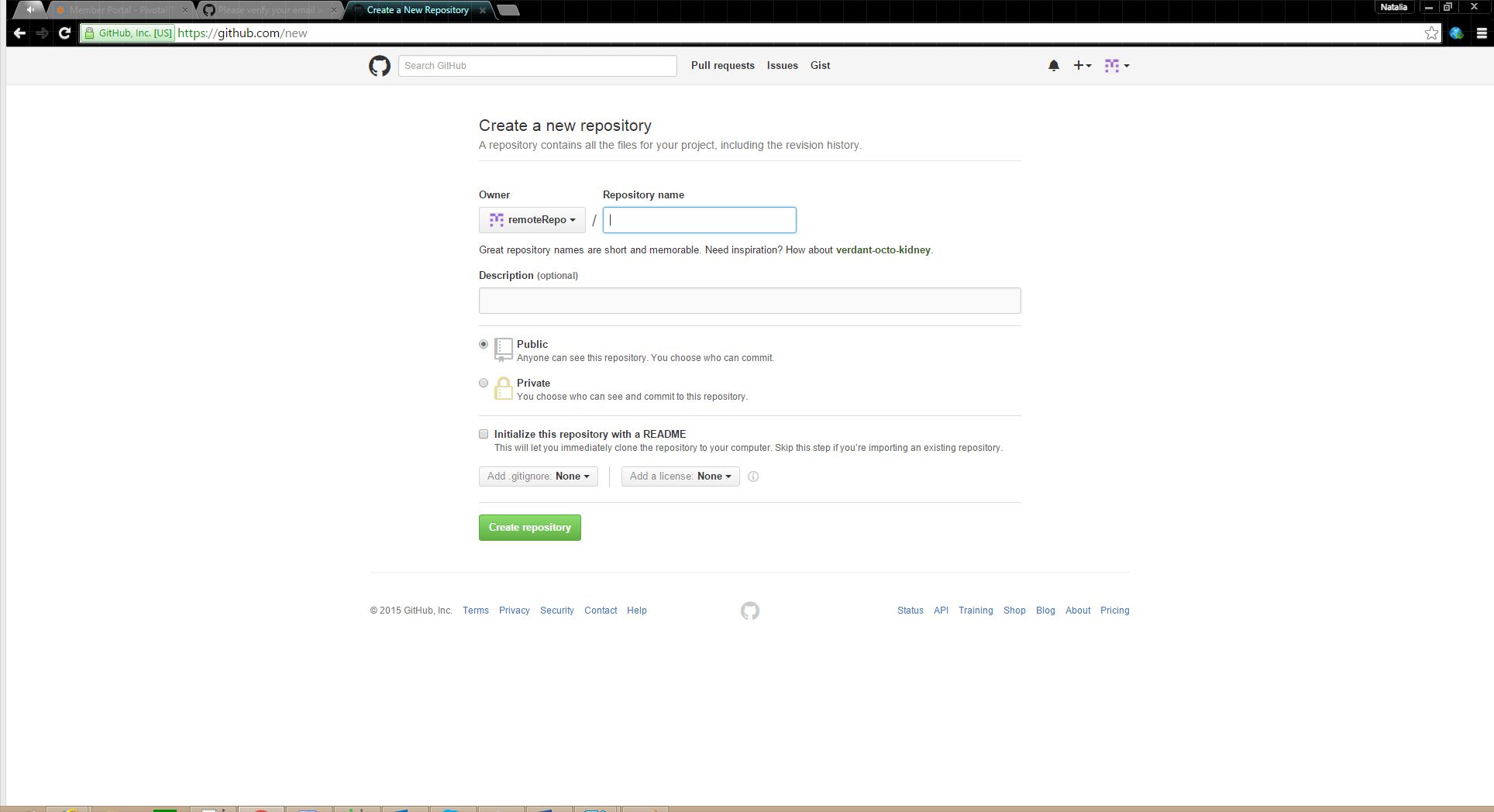
Нажимаем зеленую кнопку 



1. Находим глазками местоположение большой кнопки с надписью [**New repository]**

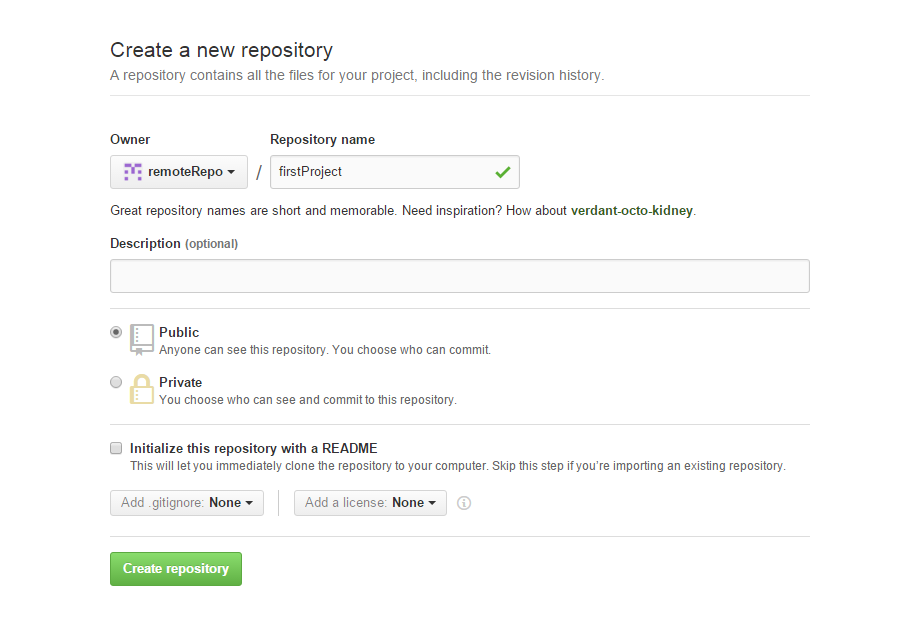


Кликаем на нее

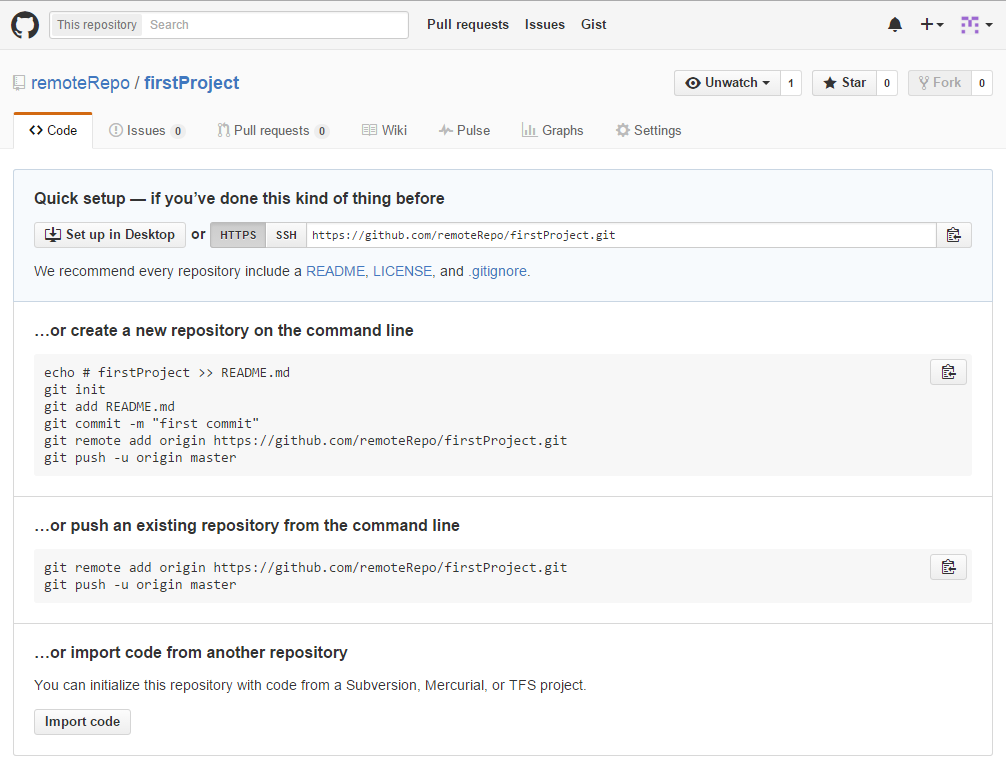
в качестве:

**Repository name –** указываем **firstProject**

Видимостьдля проекта оставляем **– Public**



Будьте внимательны после успешного создания проекта, должно появится окно, с описанием основных ссылок на Ваш только-что созданный репозиторий - по HTTPS or SSH



Прошу скопировать/запомнить следующую ссылку

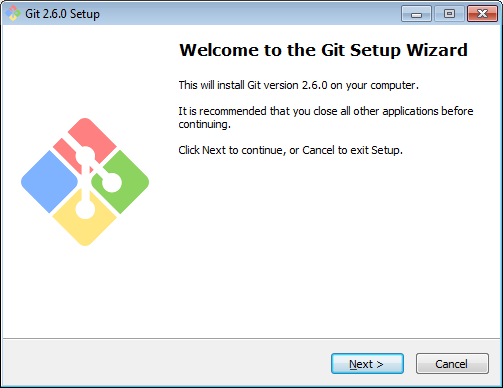
<https://github.com/remoteRepo/firstProject.git>

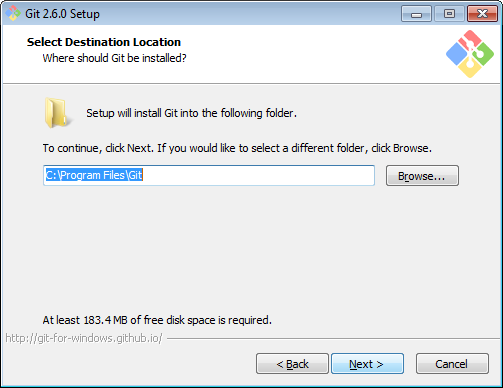
именно этот **url** будет использован нами для коннекта к репозиторию

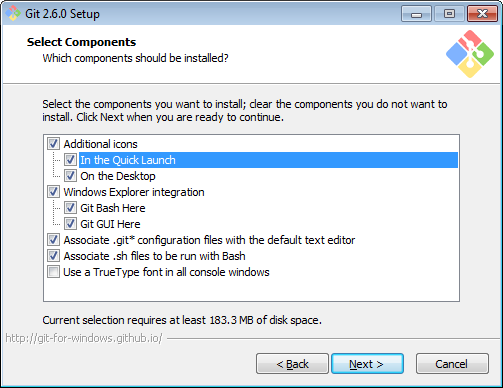
1. **Теперь перейдем к установке** **GitBash**

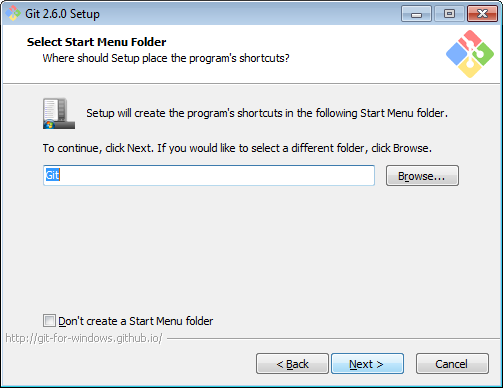
Переходим на страницу с ресурсами

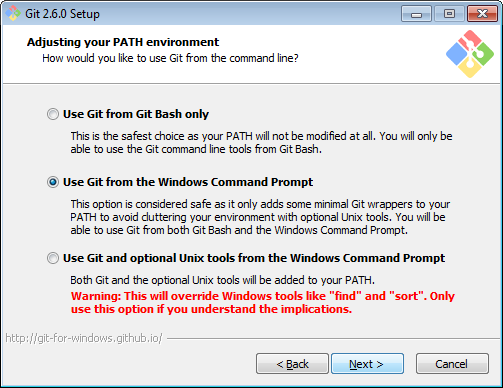
<https://git-scm.com/downloads>

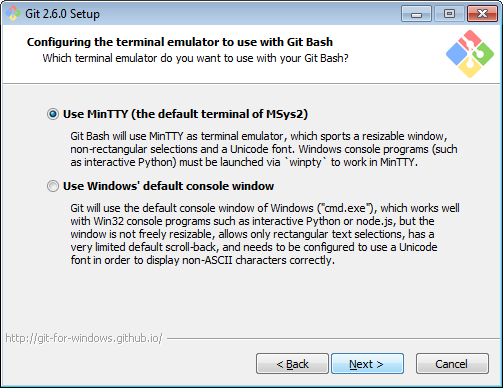
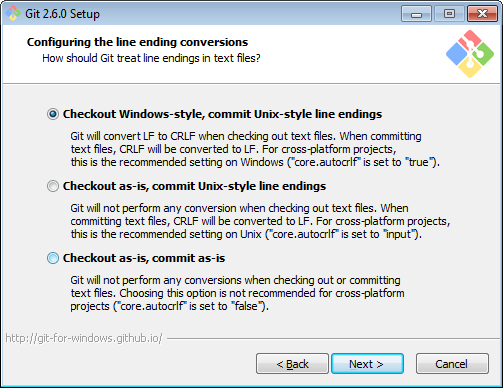


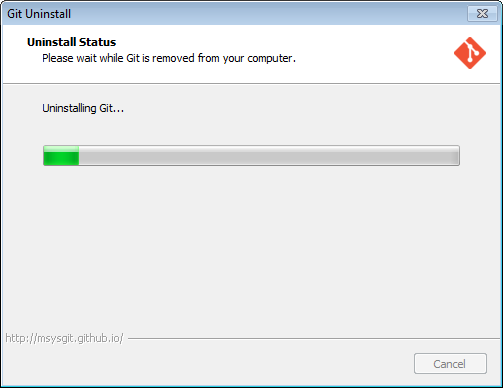


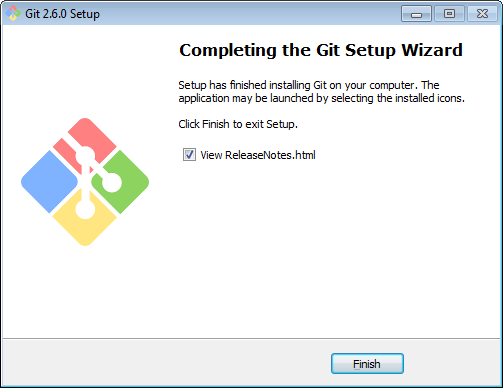






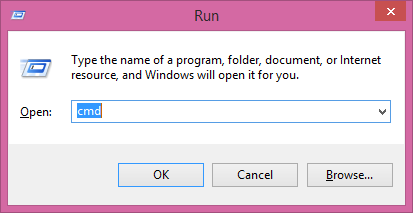






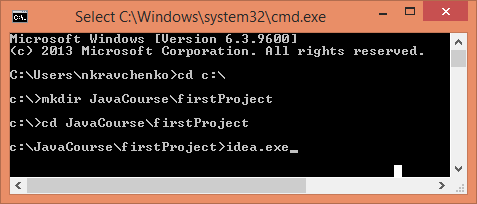
Для хранения проект**ов** в будущем – создайте в любом удобном для Вас месте файловой системы папочку, например: **JavaCourse** (именно она будет хранить все Ваши домашние задания); давайте попробуем сымитировать выполнение дз, и в пределах **JavaCourse** папки создадим еще одну папку **-> gitProject,** который будет хранить первую программку;

Из командной строки, выполнение вышеперечисленного плана выглядит так:

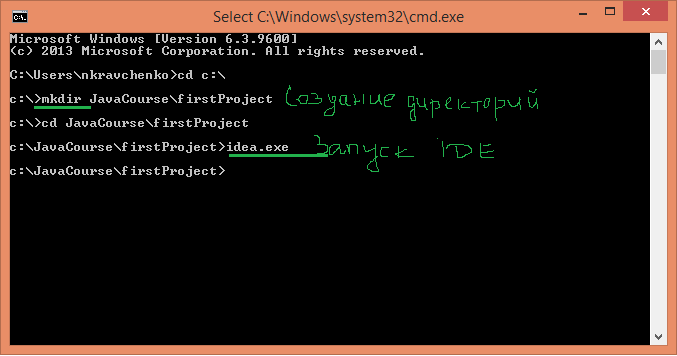


Наша цель:

* создать **JavaCourse** папку, в ее пределахразвернуть еще один folder **gitProject**;
* перейдя в **JavaCourse -> gitProject**, создадим мавен проект(я запустила **IDE** из консольки); все команды выглядят так:



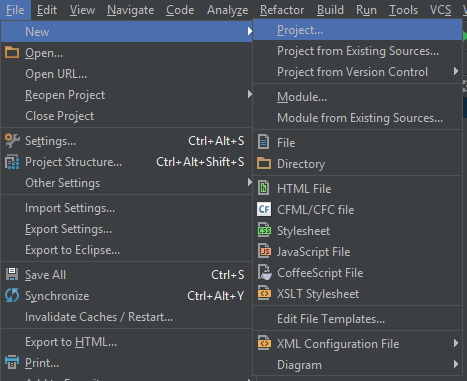
Более подробно:

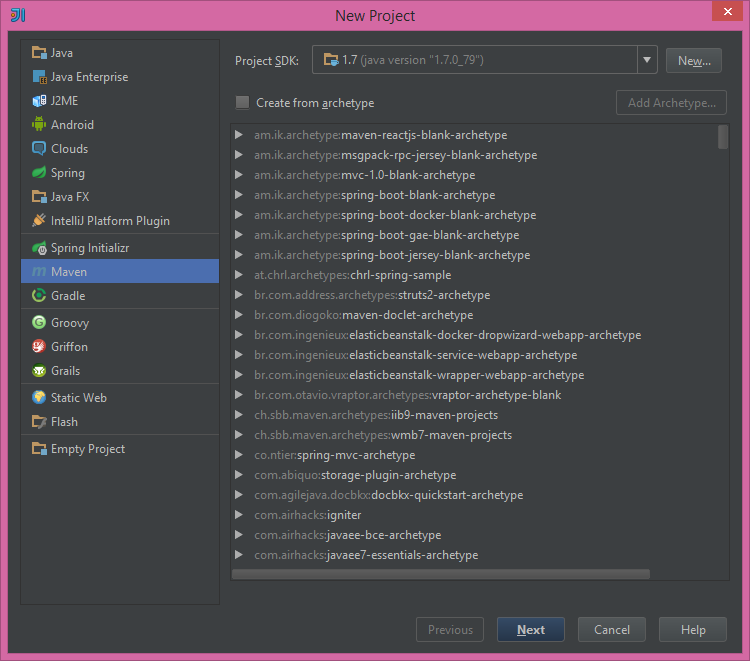




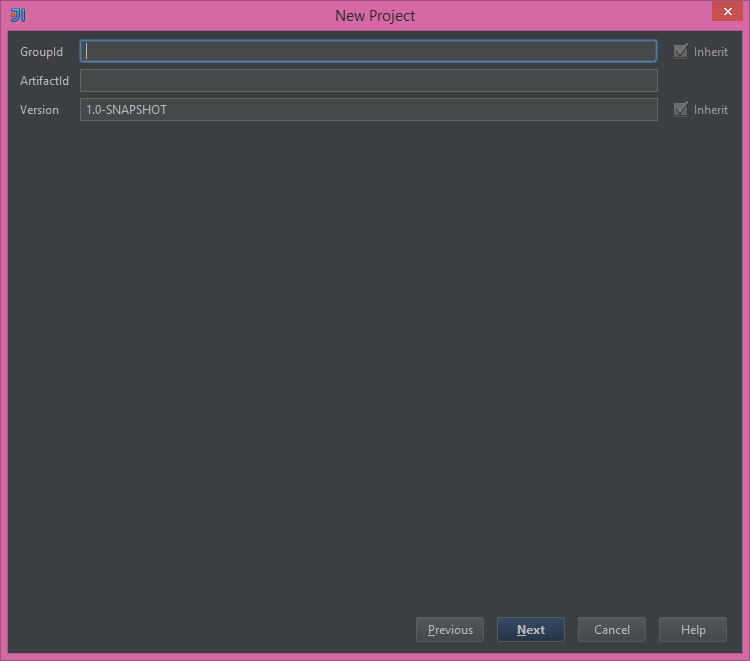
1. После запуска нашей **IDE**, приступим к созданию проекта, для этого выполним:

**File -> New Project**





Щелкаем **Next**

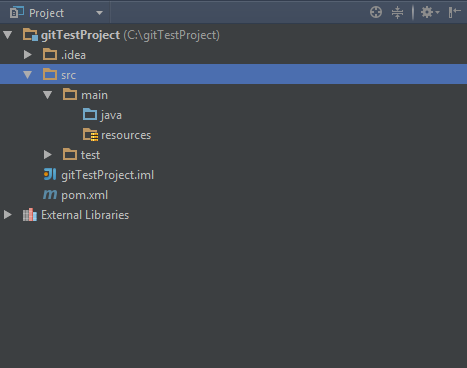


В качестве:

**GroupId -> testGitHubGroupId**

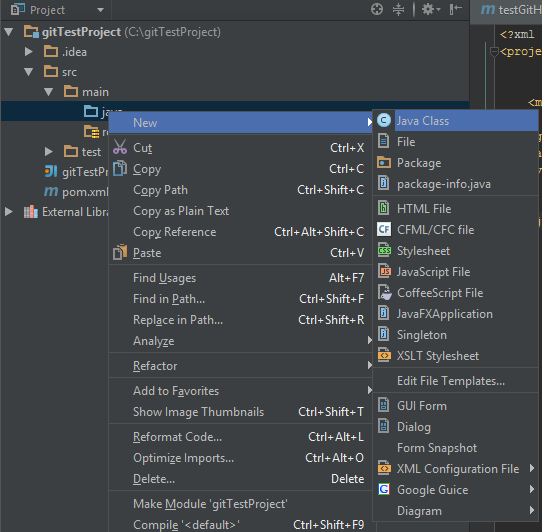
**ArtifactId -> testGitHubArtifactId**

Структура проекта:

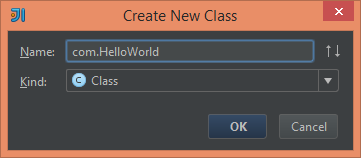
****

Кликаем правой кнопкой на **src -> main -> java**

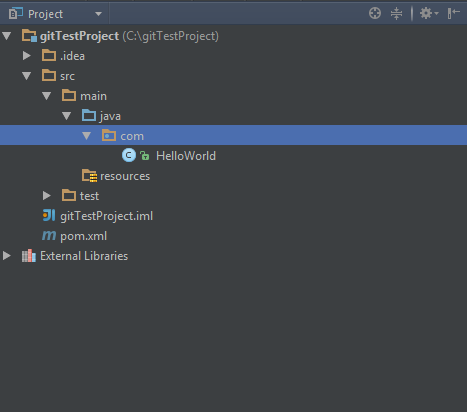
Выбираем **New** -> **Java Class**

****

Дадим название классу: **HelloWorld,** приставка **com**, будет определять пакет, в котором будет находится класс



Структура проекта после добалвения класса:

****

**Source of HelloWorld.java:**

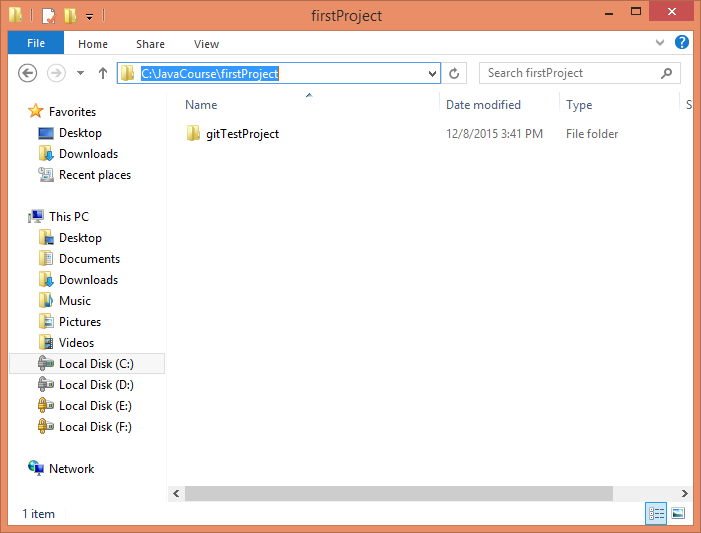
package com;  
  
public class HelloWorld {  
 public static void main(String[] args) {  
 System.*out*.println("Hello, world");  
 }  
}

Давайте проанализируем, что у нас есть:

* Удаленный репозиторий, доступный по ссылке (смотрите пункт 1)

<https://github.com/remoteRepo/firstProject.git>

* **Maven** проект, развернутый в пределах **C:\JavaCourse\firstProject**



* Установленный **GitBash** для работы с git посредством консольки

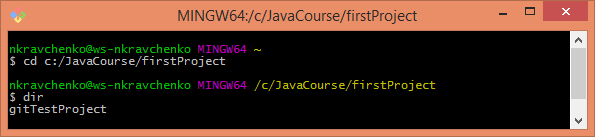
Давайте теперь попробуем связать удаленный репозиторий с нашим проектом, если говорить дословно, попробуем наш проект **gitTestProject** отправить на хранение в наш удаленный репозиторий, это будет начало нашего 5-го пункта из общего плана

1. Для начала проинициализируем наш локальный репозиторий, для того, чтобы **git** начал следить за нашим проектом, для этого:

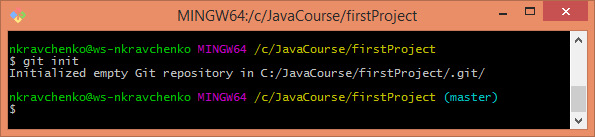
* Запустим **gitBash,** затемперейдем в **C:\JavaCourse\firstProject,** то-есть на уровень чуть выше нашего созданного maven проекта

nkravchenko@ws-nkravchenko MINGW64 ~

$ cd c:/JavaCourse/firstProject



* Выполним команду **git init –** для инициализации нашего локального репозитория



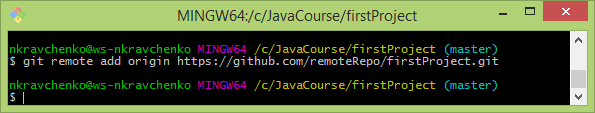
* После чего добавим алиасы на удаленный репозиторий путем использования команды **remote add [alias\_name] remote\_repository\_url**

Таким образом мы с Вами сможем обращаться к нашему удаленном репозиторию уже без использования основного урл(а), в нашем случае – это опять же смотрите пункт 1 <https://github.com/remoteRepo/firstProject.git> , а с использованием так называемого алиаса(синонима);

итак добавим алиас(безусловно Вы можете определить его как угодно), в моем случае это **origin** :

nkravchenko@ws-nkravchenko MINGW64 /c/JavaCourse/firstProject (master)

$ git remote add **origin** https://github.com/remoteRepo/firstProject.git



для того, чтобы было понятно, что все прошло нормально – проверьте настройки для удаленного репозитория, - выполним команду:

nkravchenko@ws-nkravchenko MINGW64 /c/JavaCourse/firstProject (master)

$ **git remote -v**

origin https://github.com/remoteRepo/firstProject.git (fetch)

origin https://github.com/remoteRepo/firstProject.git (push)

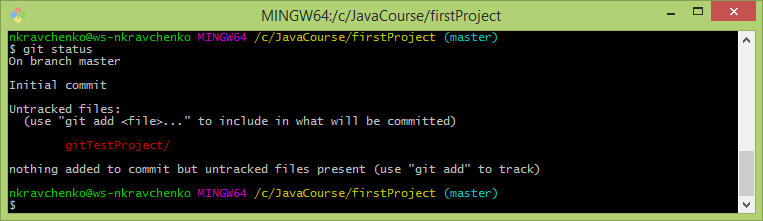


Видим, что все гуд)

После выполения всех вышеперечисленных пунктов, Вы должны обнаружить, что **git** стал следить за Вашей папкой с проектами; проверить просто – достаточно выполнить команду

nkravchenko@ws-nkravchenko MINGW64 /c/JavaCourse/firstProject (master)

$ git status

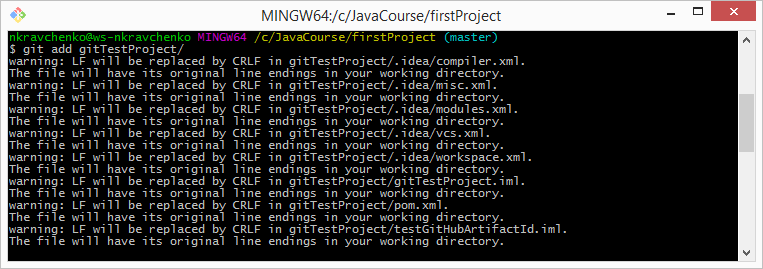


1. мы находимся в пределах бранча под названием **master**
2. красный цвет – определяет то, что наш **maven-project** не был добавлен в локальный репозиторий, и соответственно изменения в нем не фиксируются системой контроля версий – **git**(наш проект таким образом находится в рабочем каталоге, не стейджинге)**;** то-есть что бы я ни сделала в пределах папки **gitTestProject**, отображаться или фиксироваться в **git** не будет;

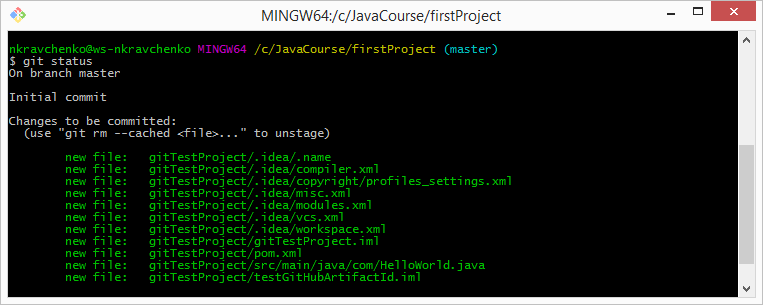
поэтому для того, чтобы наш maven-проект добавить в пределы видимости **git** выполним следующую команду:

nkravchenko@ws-nkravchenko MINGW64 /c/JavaCourse/firstProject (master)

$ git add gitTestProject/



Снова проверим **git status**



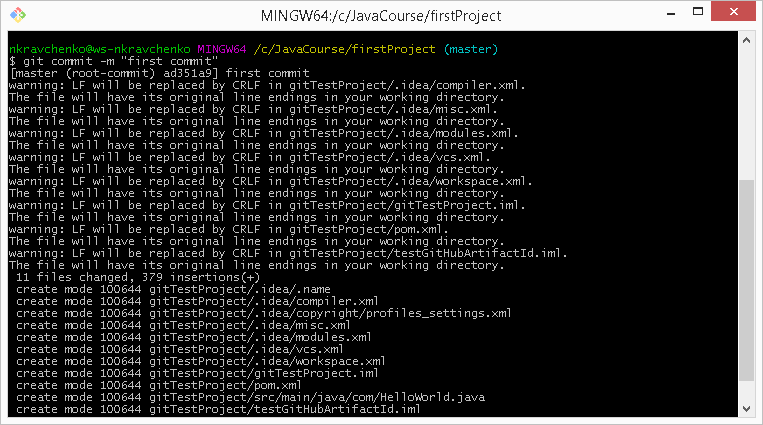
Видим, что все файлики стали зелеными – следовательно попали в стейджинг каталог, как следствие могут быть закоммичены и запушены(зафиксированы и отправлены) в удаленный репозиторий в ближайшее время;

Давайте сделаем коммит, и отправим проект в удаленный репозиторий?)

* Для этого выполним команду следующего вида:

nkravchenko@ws-nkravchenko MINGW64 /c/JavaCourse/firstProject (master)

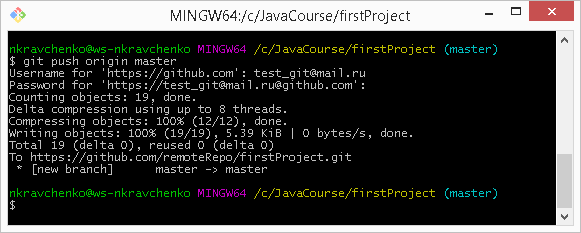
$ git commit -m "first commit"



* Все готово, для того, чтобы теперь сделать так называемый **push,** если дословно – отправить данные в удаленный репозиторий(**GitHub**)

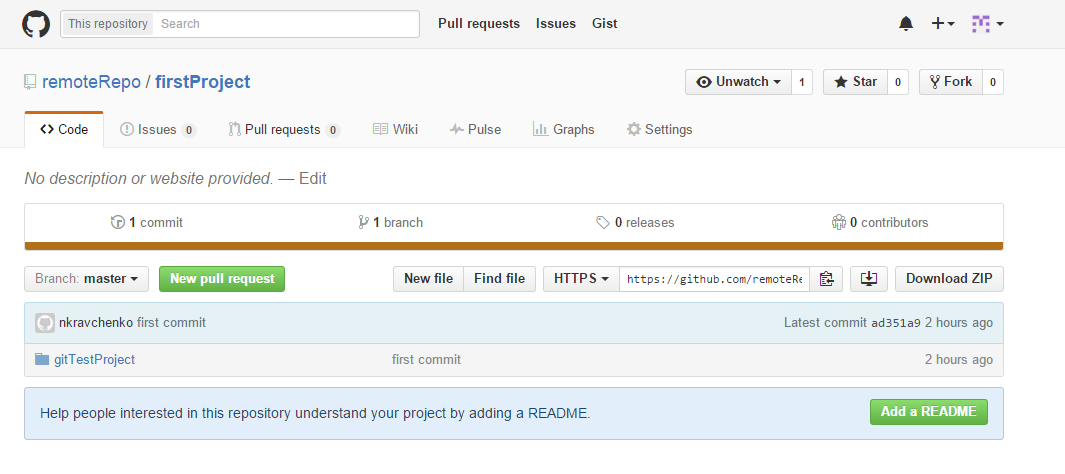
nkravchenko@ws-nkravchenko MINGW64 /c/JavaCourse/firstProject (master)

$ git push origin master



Давайте теперь посмотрим, что происходит на стороне remote репозитория(**gitHub**(a))

<https://github.com/remoteRepo/firstProject>



Видим, что в пределах нашей папки **firstProject** появился еще один проект **gitTestProject**