|  |  |
| --- | --- |
| **Práctica Build 2+GIT** | Entregables: Este documento, la url del repositorio público en github y el histórico bash |
| Fecha Entrega: 26/2/19 |
| Equipo: Individual |
| Herramientas: Ant, Maven y GIT |

**Introducción**

Las aplicaciones JAVA tiene un alto grado de *uniformidad* en cuanto a estructura de directorio gracias al uso mayoritario de herramientas como **Maven**. La gestión de la configuración de un entregable requiere herramientas específicas y en esta práctica se presenta la herramienta **GIT**.

**Desarrollo de la práctica**

1.-Comprueba si están instalados y, si no lo están, instala las siguientes herramientas:

**Ant**: https://ant.apache.org/manual/install.html

**Maven** : https://maven.apache.org/install.html

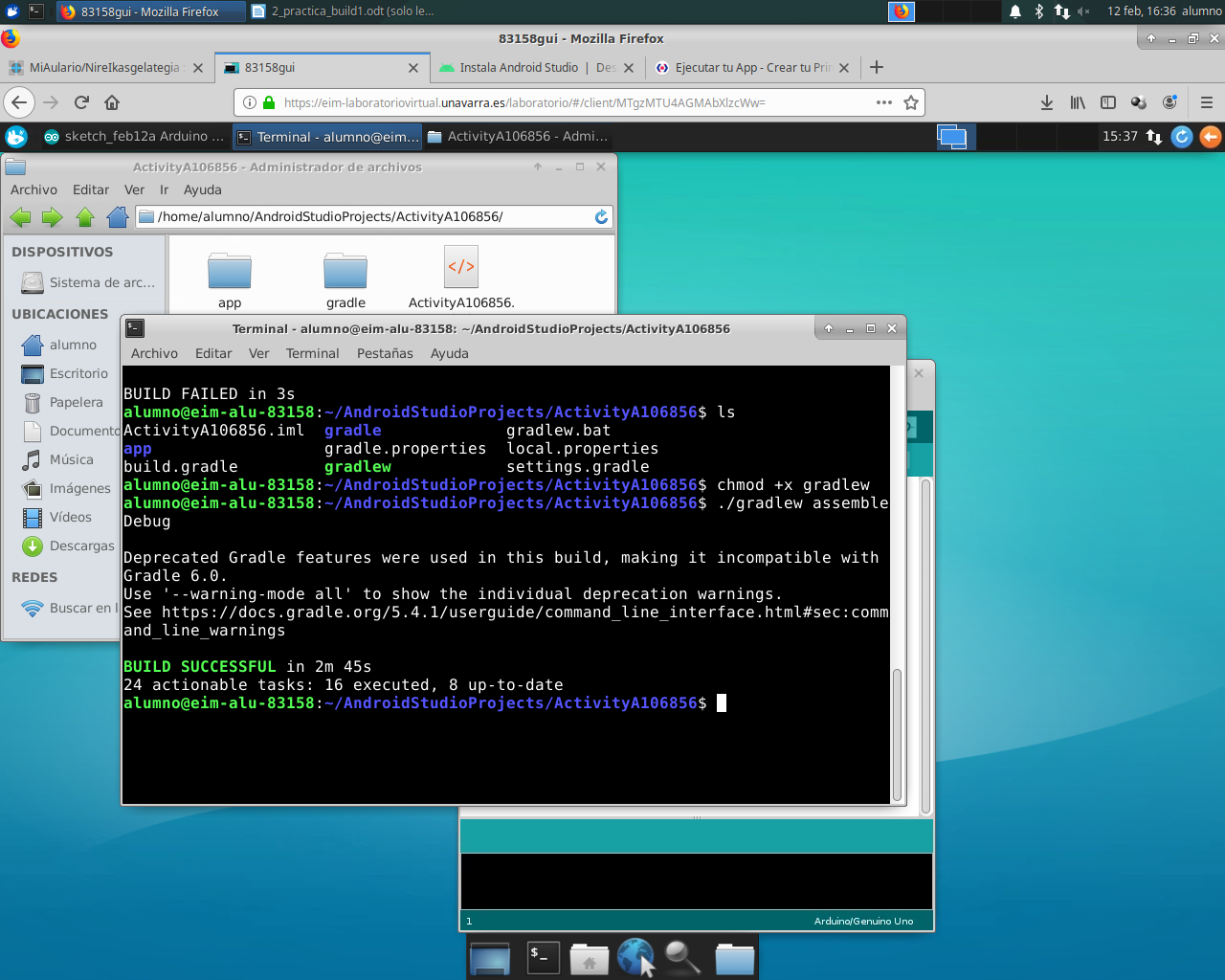
**GIT**: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

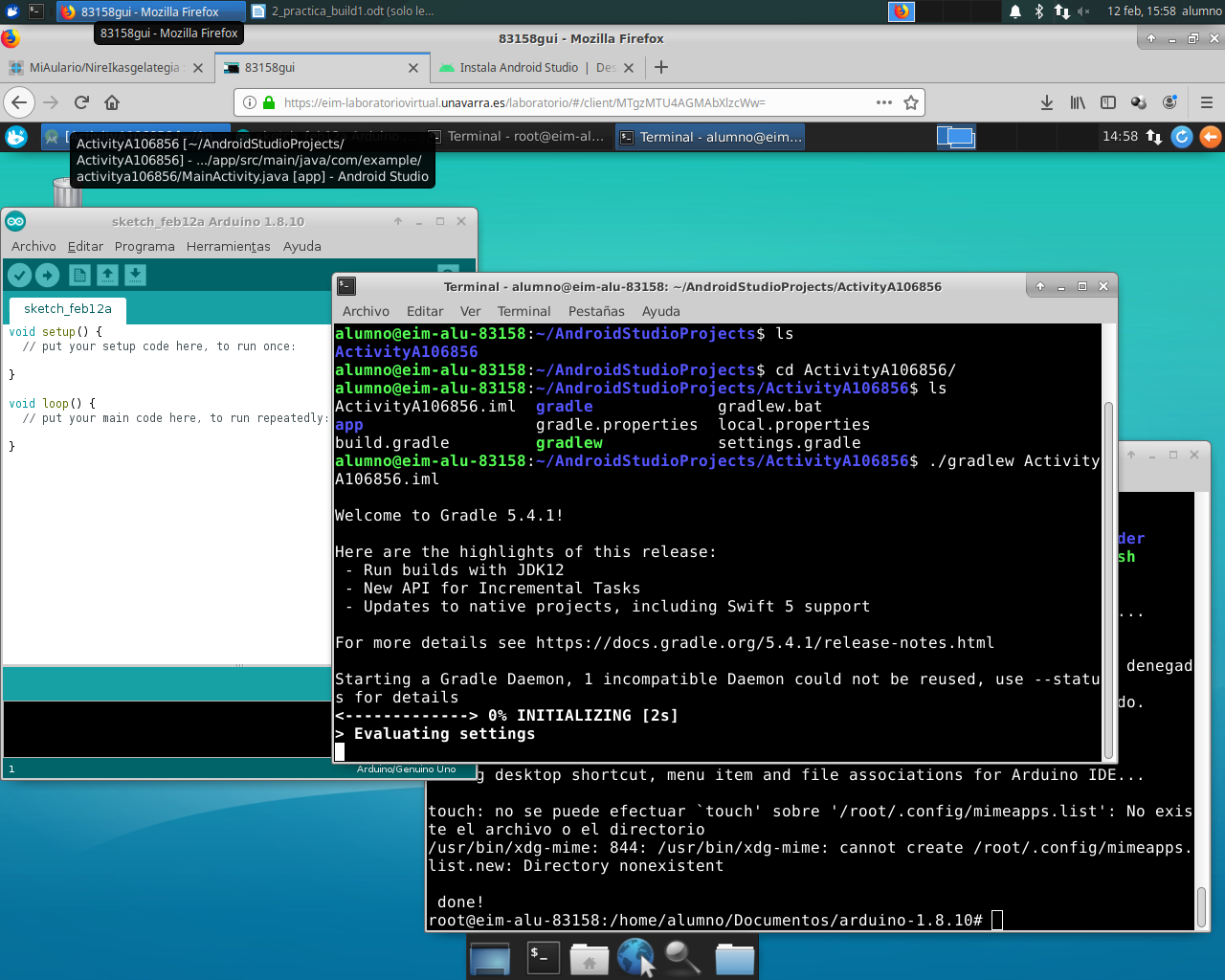
2.-Prepara la generación y compilación de dos entregables del proyecto, revisando la documentación de **Maven** y las múltiples opciones *antes* de crear la estructura:

2.1.-**Maven**. Genera dos estructuras de proyecto con las ***convenciones*** dadas (paquete y de nomenclatura de la clase principal):

* 1. -DartifactId=simple
  2. -DartifactId=simple-webapp

2.2.-**Ant**. Define un fichero de compilación, *build.xml,* para compilar la estructura del proyecto “*simple*” (src y target) generado en el apartado anterior.





3.1.- Teniendo en cuenta el WBS del proyecto, organiza la estructura de un repositorio personal público de **github** para esta asignaturadonde subir los entregables. Incluye los entregables de la práctica anterior e indica a continuación la url del **github**.

Cuenta github: https://github.com/zaratiegui106856/gpi2.git

3.2.-Realiza el siguiente ejercicio de *branching y merging* con git(por línea de comando):

a) Crea un repositorio local con el fichero “index.html”. Después crea el branch *req001* para trabajar en una nueva funcionalidad. Modifica el fichero “index.html” e intégralo en el  *repositorio local*.

b) En este punto del proyecto surge la necesidad de un *hotfix*: No se puede poner en producción los cambios de *req001* por lo que para realizar el *hotfix* se debe volver al *branch* *master*. Para resolver la incidencia hay que crear un nuevo *branch* *hotfix* donde hay que modificar “index.html”.

c) Después de hacer los cambios en *hotfix*, se haz un *merge* al *branch* *master* y borra el *branch* *hotfix*.

d) Por último, vuelve al branch *req001* para continuar el trabajo previsto inicialmente.

3.3.-Explica cómo podrías *minimizar* o *evitar* *conflictos* dentro de un equipo de desarrollo a partir de la situación que se plantea en el siguiente enlace: <https://www.atlassian.com/git/tutorials/using-branches/merge-conflicts>

Una de las soluciones sería empleando ramas para evitar la modificación paralela de los mismos archivos. Y para evitar conflictos al realizar el merge una de las soluciones sería hacer un git status para ver que archivos se han modificado. Además con git diff se observa cuales han sido estos cambios.

https://github.com/zaratiegui106856/gpi2/tree/master/p3/simple-webapp/src/test/java/106856