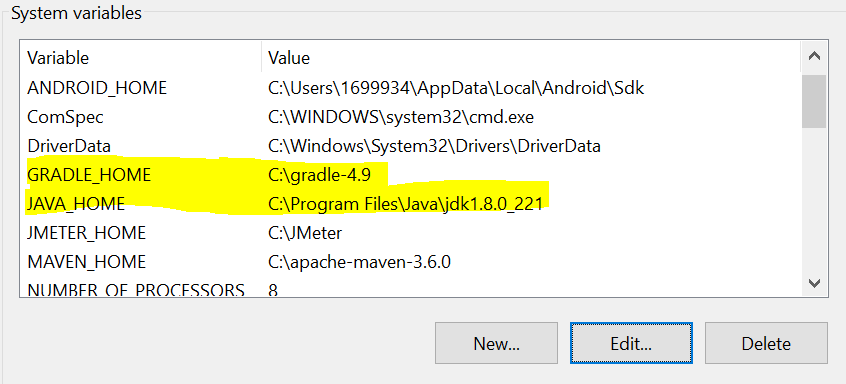
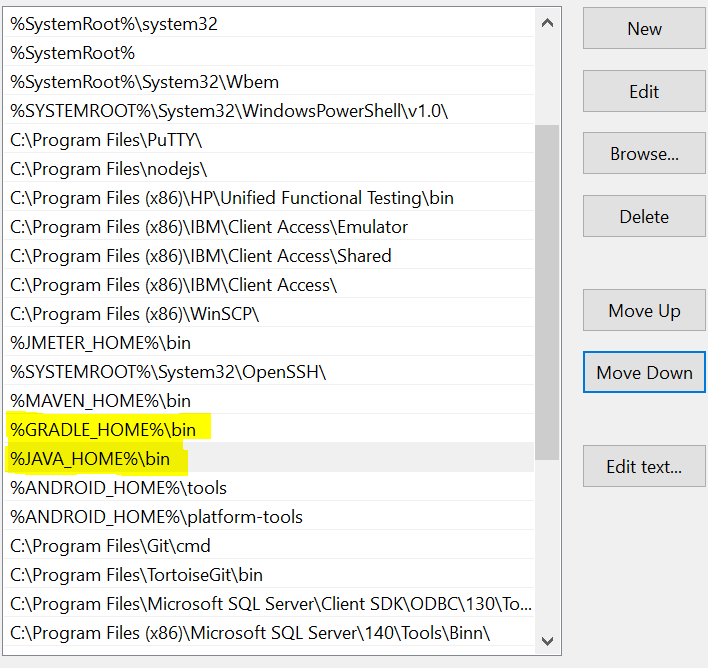
**PROGRAMAS NECESARIOS Y VARIABLES DE ENTORNO**

* **Java jdk 1.8**
* **Gradle** (Personalmente recomiendo la versión 4.9, si ya tienen un Gradle instalado no cambiarlo)

Estas son las variables de entorno que se deben crear



Junto a estas variables también se deben crear 2 nuevas variables en el path



* **GIT (Verificar que este actualizado en la última versión)**

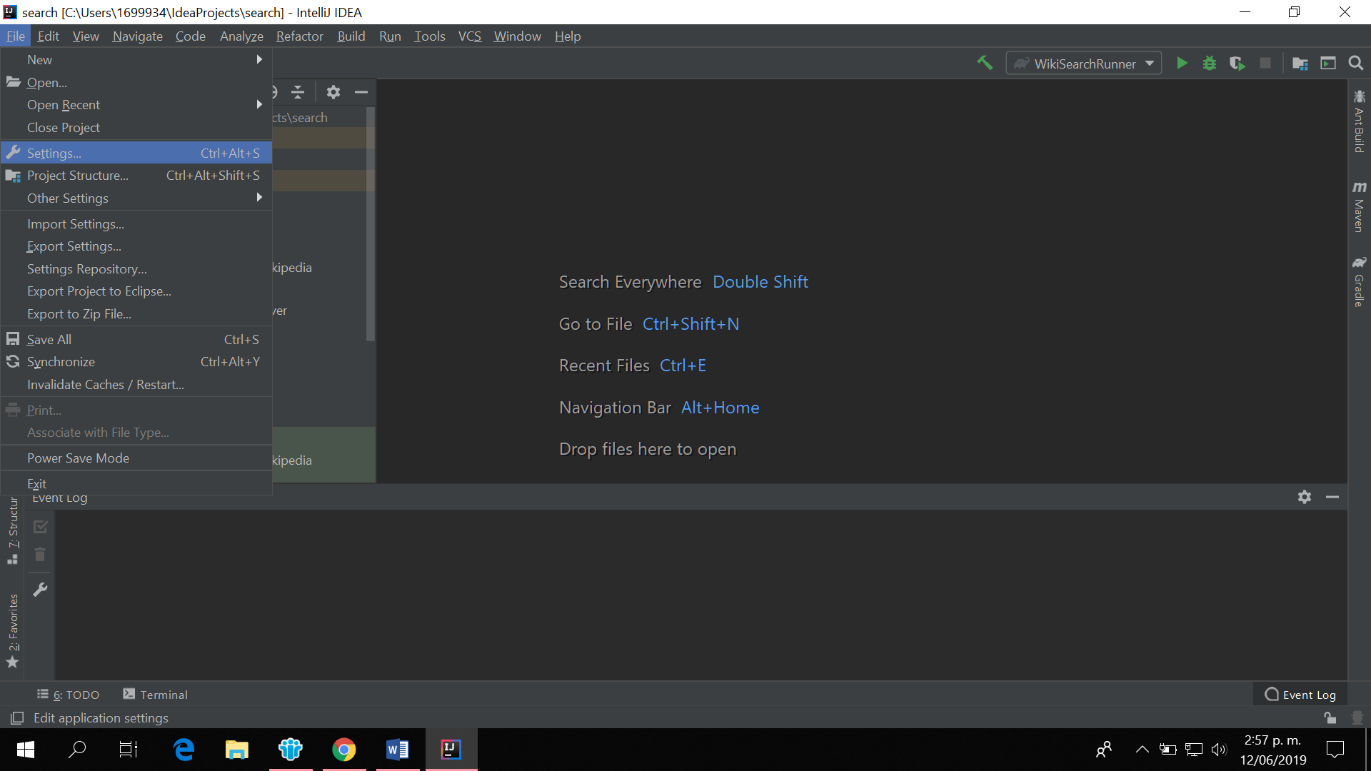
**ENTORNOS DE DESARROLLO**

* **Eclipse IDE o IntelliJ IDEA:** Se puede usar cualquiera de estos entornos, al momento de hacer integraciones de código entre un entorno y otro no habrá conflictos.

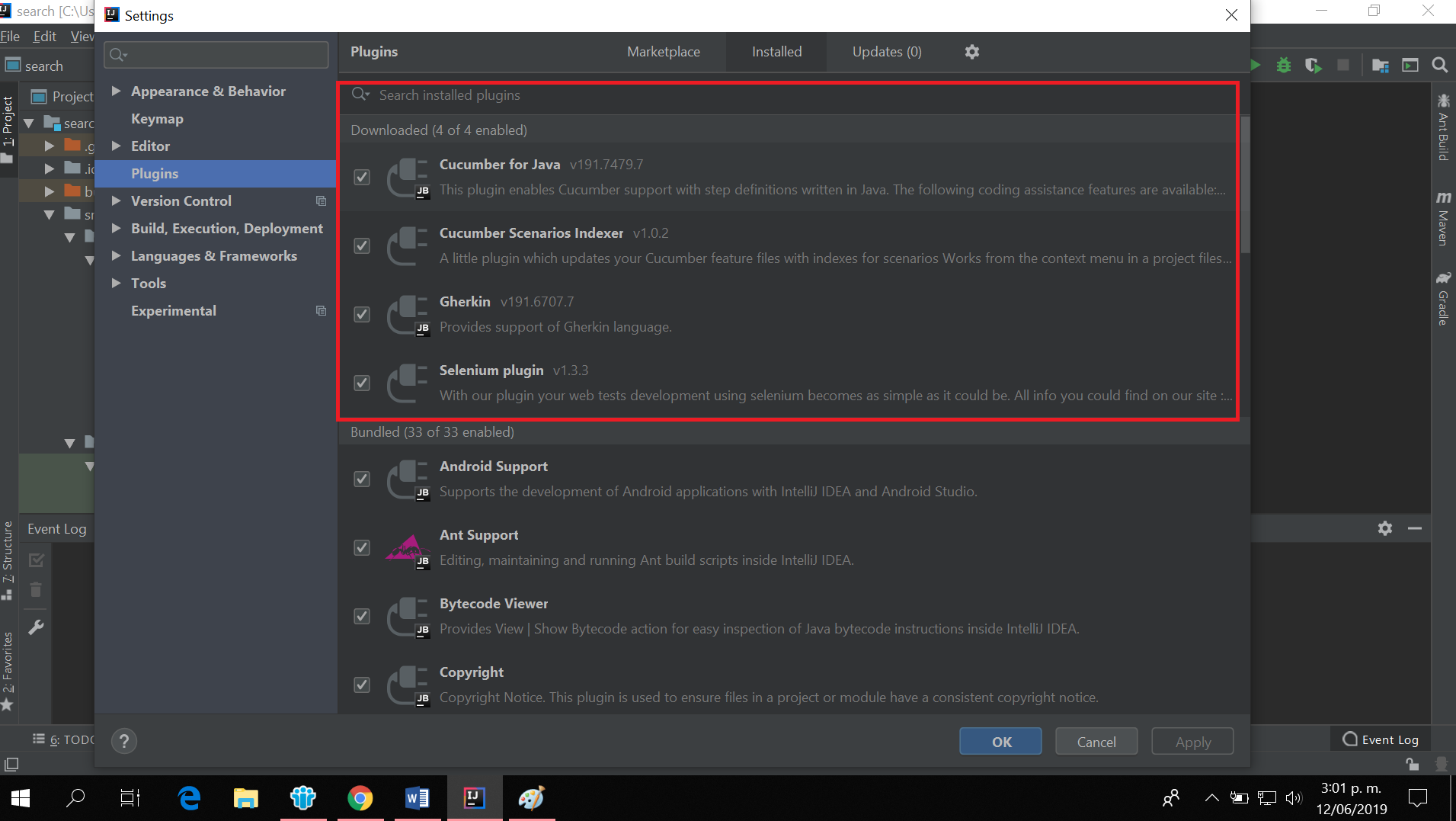
**CONFIGURAR INTELLIJ Y ECLIPSE PARA SCREENPLAY**

Configurar plugins en intellij:

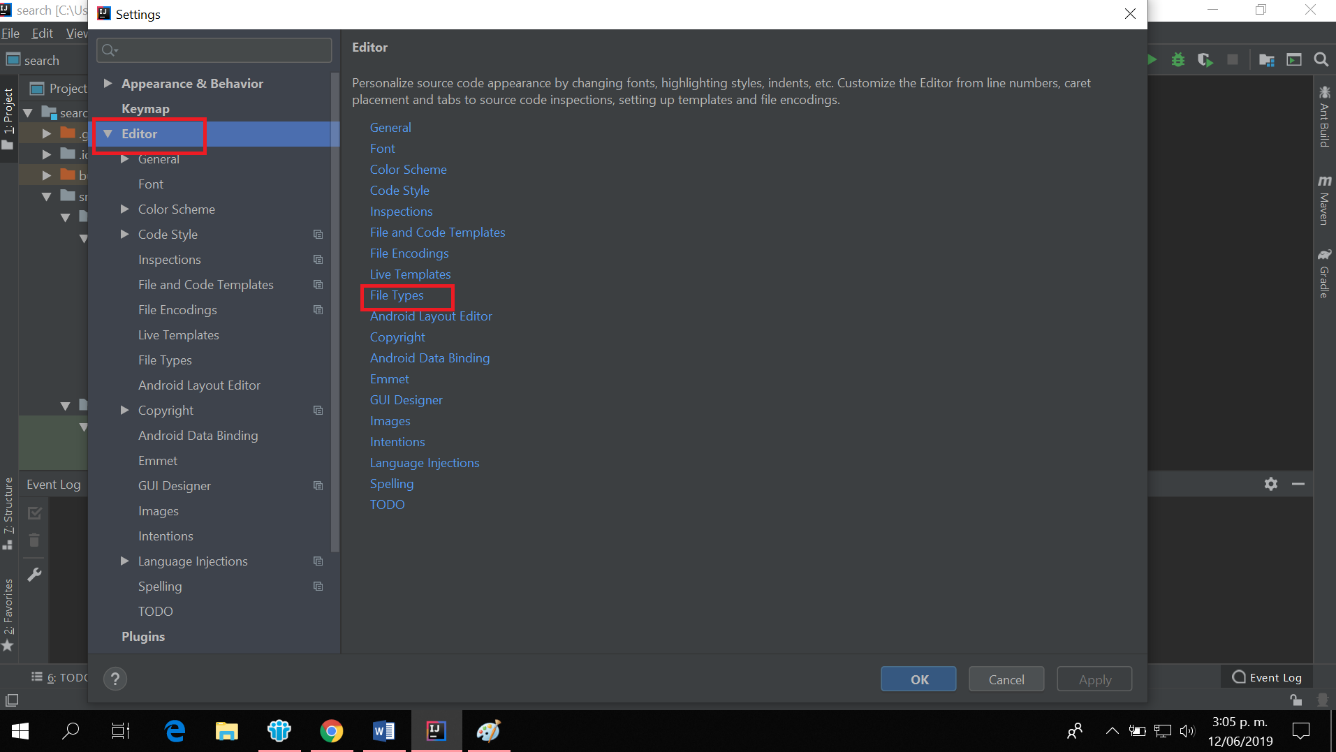
1. Ir a File(Archivo) -> Settings(Ajustes)

****

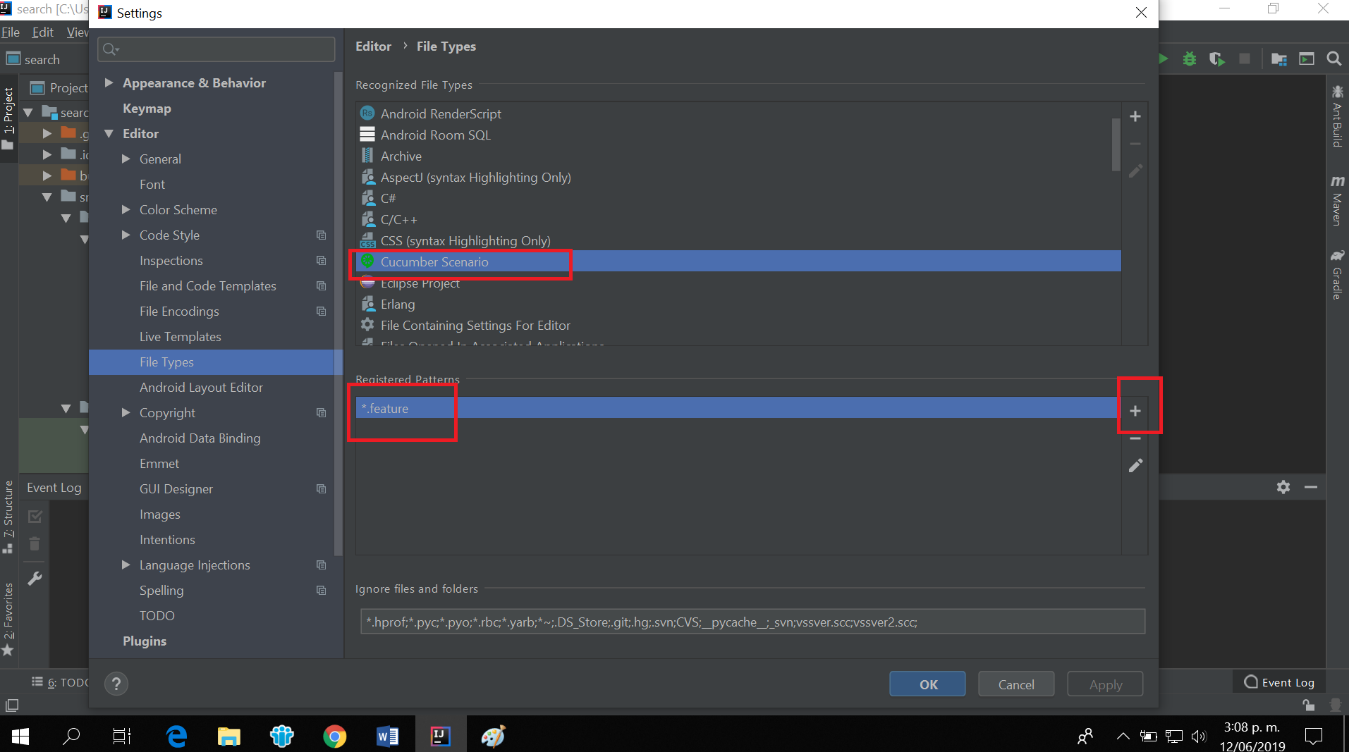
1. Seleccionar Plugins



1. Buscar en la pestaña Marketplace los plugins resaltados e instalarlos (Reiniciar el IDE) después de instalar los plugins
2. De nuevo File(Archivo) -> Settings (Ajustes)
3. Ir a la opción Editor -> File Types (Tipos de Archivo)



1. Buscar la opción Cucumber Scenario (Escenario cucumber), Dar click en el botón + para añadir una extensión de archivo al registro, y escribir el tipo de archivo “\*.feature”



1. Click en Apply(Aplicar) y en OK

Al tener los plugins de cucumber y la extensión de .feature registrada, al momento de crear un .feature para la automatización el IDE reconocerá su extensión y permitirá la creación de features al igual que en Eclipse IDE(Scenario, Background, Examples, Scenario Outline, Given, When, Then, And, @tags, etc)

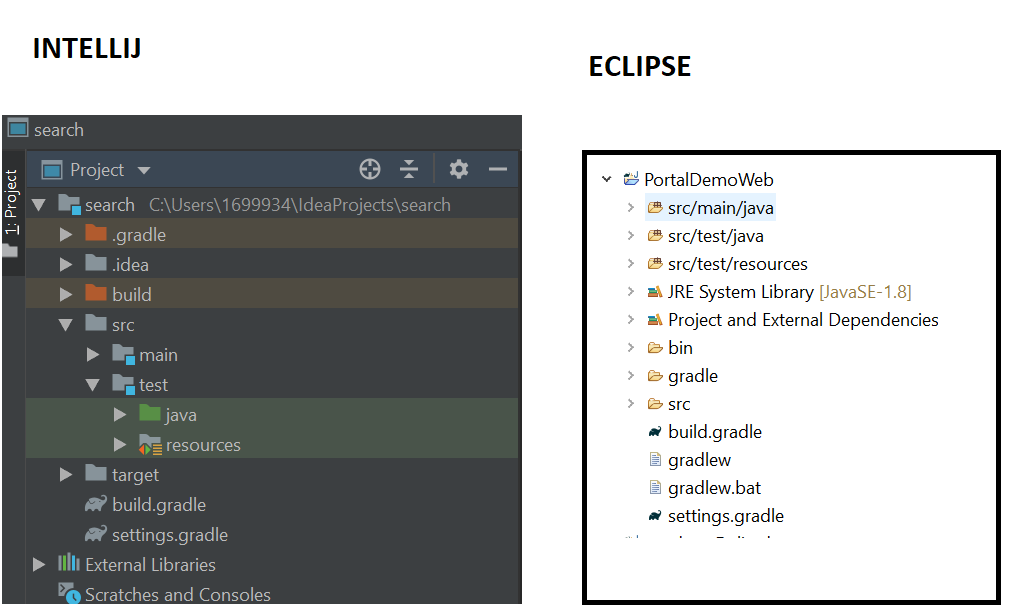
Configurar en eclipse

1. Click en la pestaña Help -> eclipse Marketplace, buscar cucumber y descargar cucumber plugin for eclipse

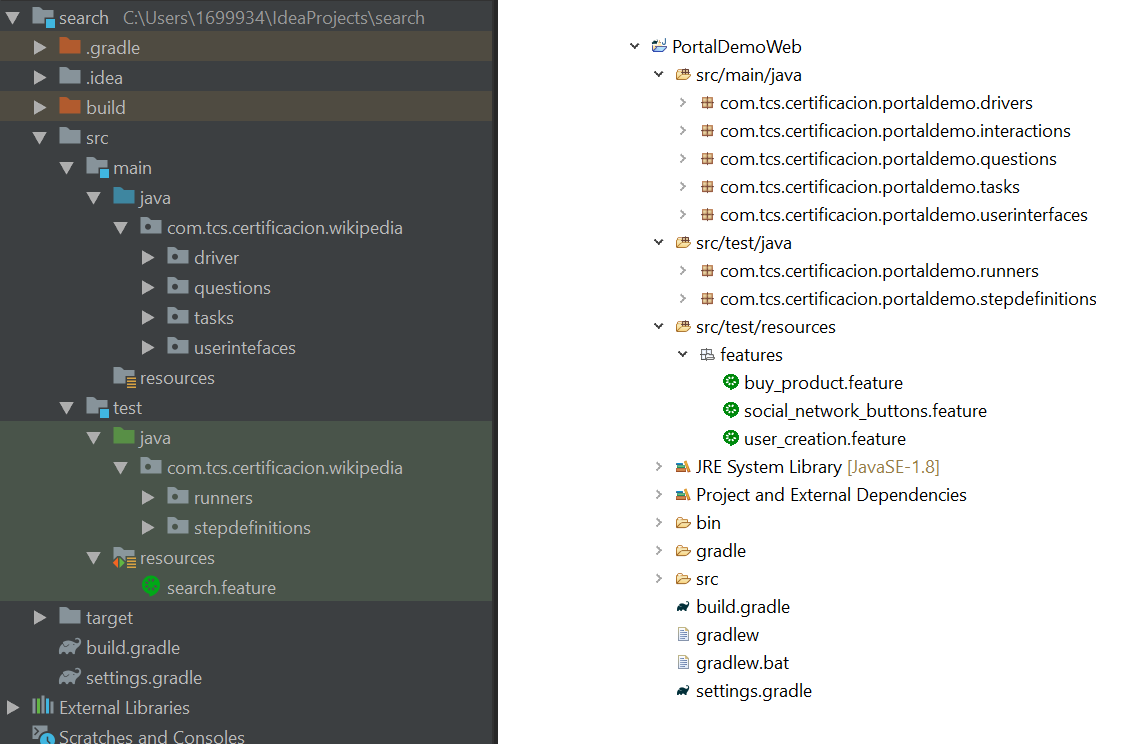
**ESTRUCTURA DE SCREENPLAY**

La arquitectura manejada por screenplay es la siguiente para proyectos gradle en eclipse o intellij

Source folders: src/main/java, src/test/java, src/test/resources



Paquetes:

****

drivers: El paquete donde se crearán las clases que manejarán los drivers usados en la automatización (Web: Chrome, explorer, mozilla, etc, Movil: Android, Escritorio (Excel, word))

questions: El paquete donde se crearán las clases que manejarán las comparaciones y validaciones (Asserts, Matchers)

tasks: Paquete donde se crearán las clases con las acciones que realizara el actor

(Inicia sesión, compra un producto, hace una transaccion)

interactions: Paquete donde se crearán las clases con las interacciones que hace el actor (click, escribir, etc)

userinterface(ui): Paquete donde se crearán las clases donde se mapearán los objetos (botones, campos de texto, etc). Se debe crear una clase por cada cambio de página(web), pantalla(móvil), interfaz(escritorio) que suceda en la automatización

runners: Paquete donde se crearán las clases que correrán las automatizaciones

existir un runner que lo maneje

stepdefinitions: Paquete donde se crearán las clases que tendrán los métodos que llaman los pasos (steps) que realiza el usuario para la automatización (inicia el navegador, se loguea, verifica su logueo exitoso)

**BUILD.GRADLE**

Es el archivo donde se definirá que librerías se utilizaran para el proyecto, estas cambian dependiendo de las necesidades y van organizadas de la siguiente manera:



Este no cambia al usarse en intellij o eclipse, en el encabezado dependencies es donde van las dependencias que se van a utilizar, en el encabezado repositories se define el repositorio desde el que se descargaran las dependencias elegidas (jcenter() es el repositorio por defecto y descargara las dependencias desde internet

Maven{url} es el repositorio interno de TCS)