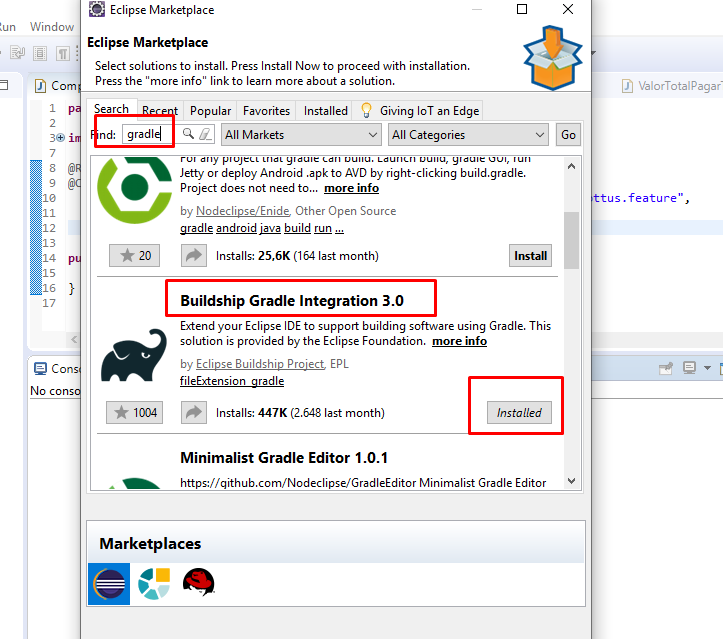
Documentación Reto Tech and Solve

Desarrollado por Edison Correa

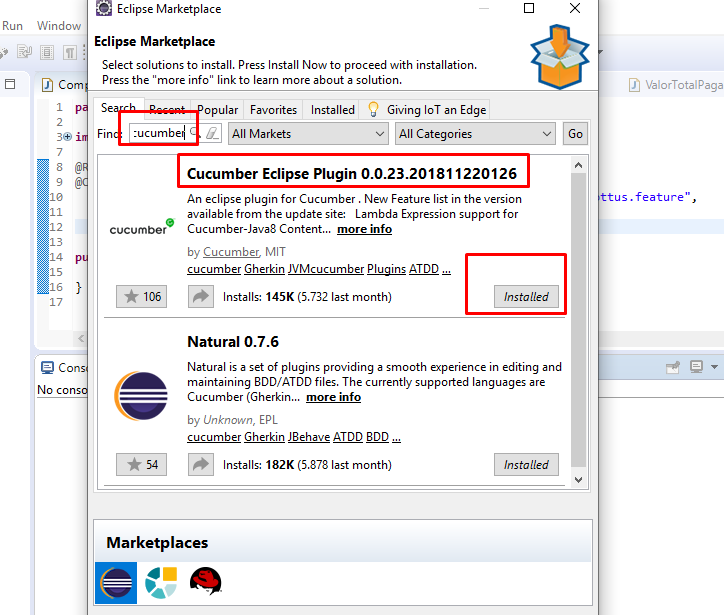
Email: [correa.edison@gmail.com](mailto:correa.edison@gmail.com)

Cel: 3016116908

1. Validar que Gradle se encuentre instalado

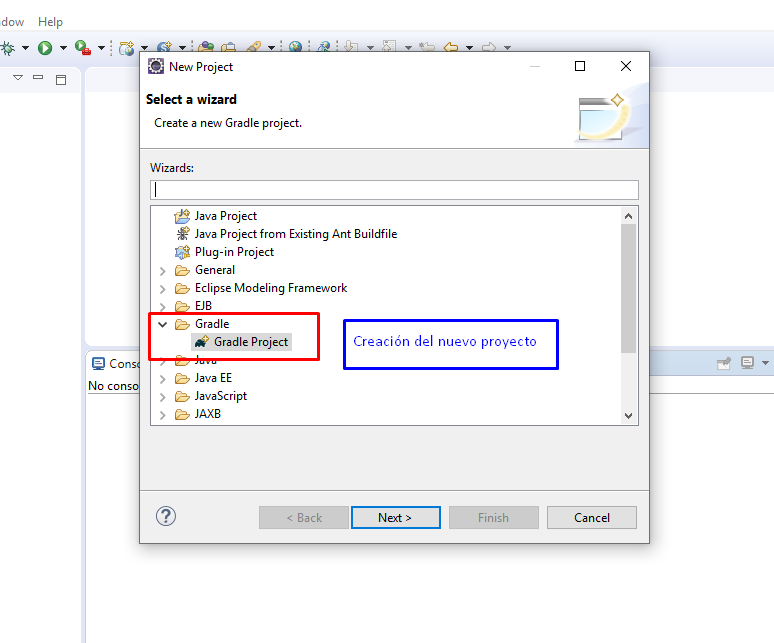


1. Validar que Cucumber se encuentre instalado

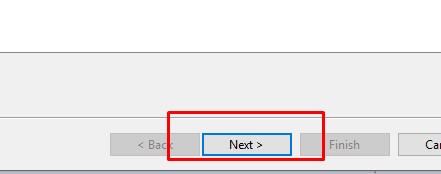


1. Creación de un nuevo proyecto

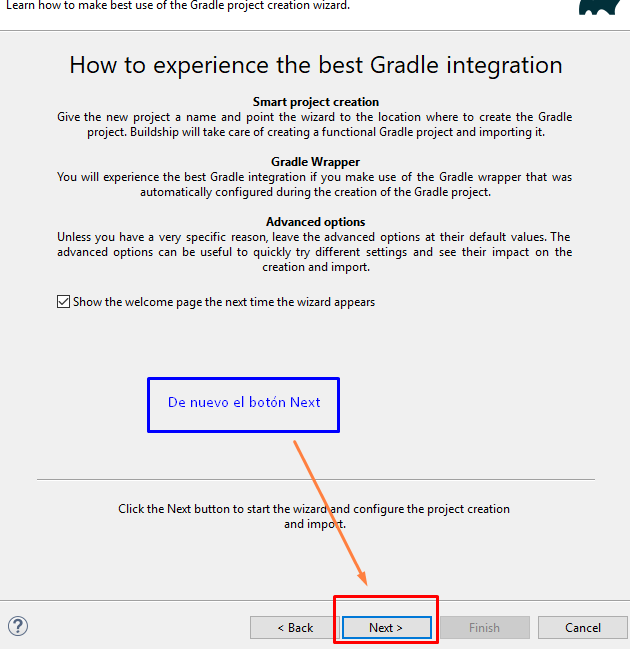
Ruta: File/New/Other/Gradle/Gradle Project



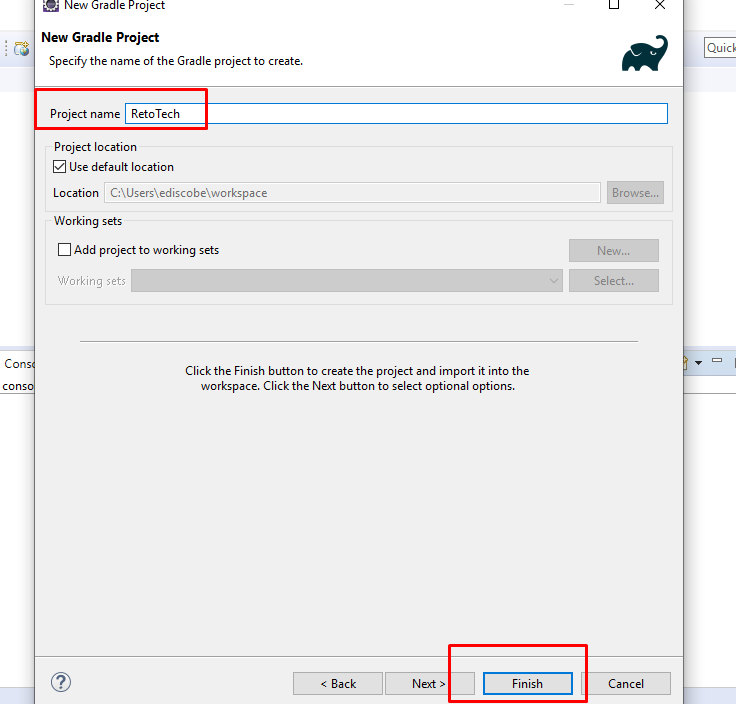
Se presiona el botón Next



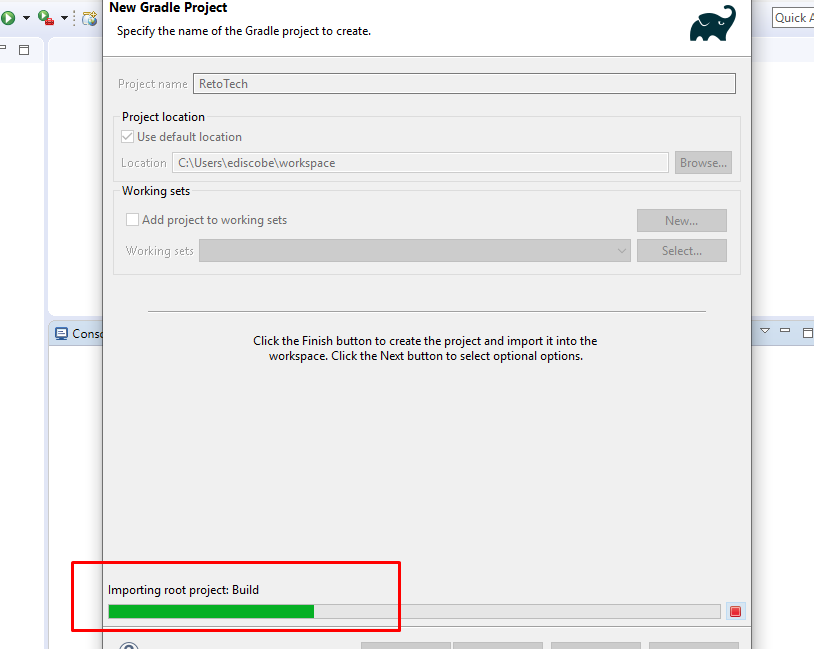
De nuevo se presiona el botón Next



Se le da un nombre al proyecto y se presiona el botón de Finish



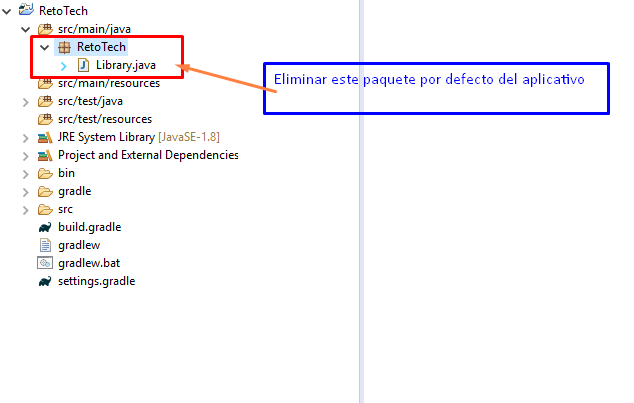
Se espera la carga de la generación del Archivo

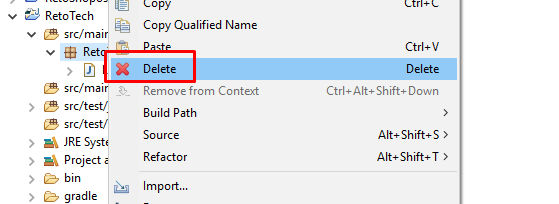




1. El aplicativo por defecto crea dos paquetes, los cuales serán borrados para procesos de creación del proyecto. El primero ubicado en la ruta

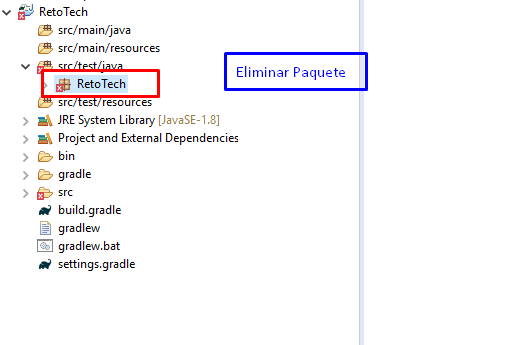
Src/main/java



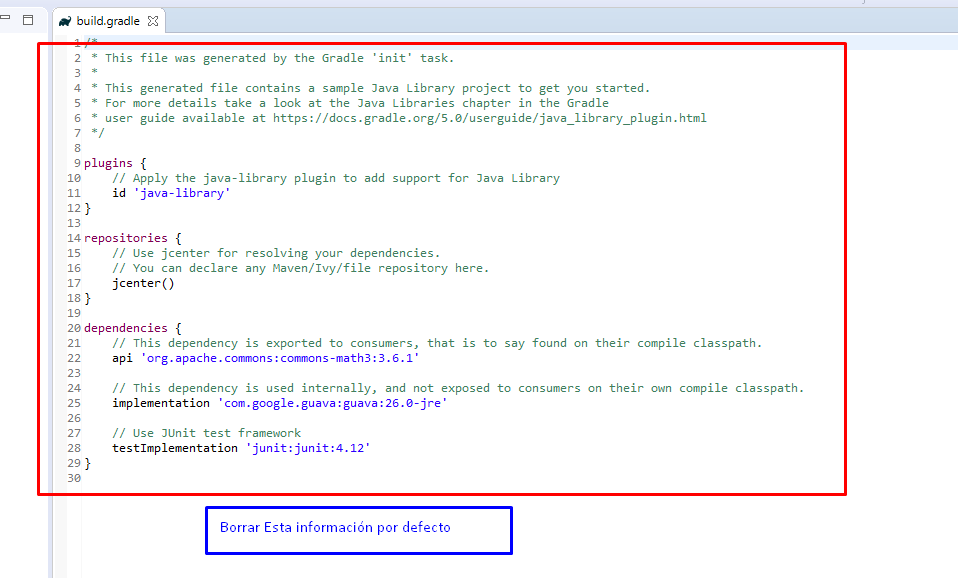


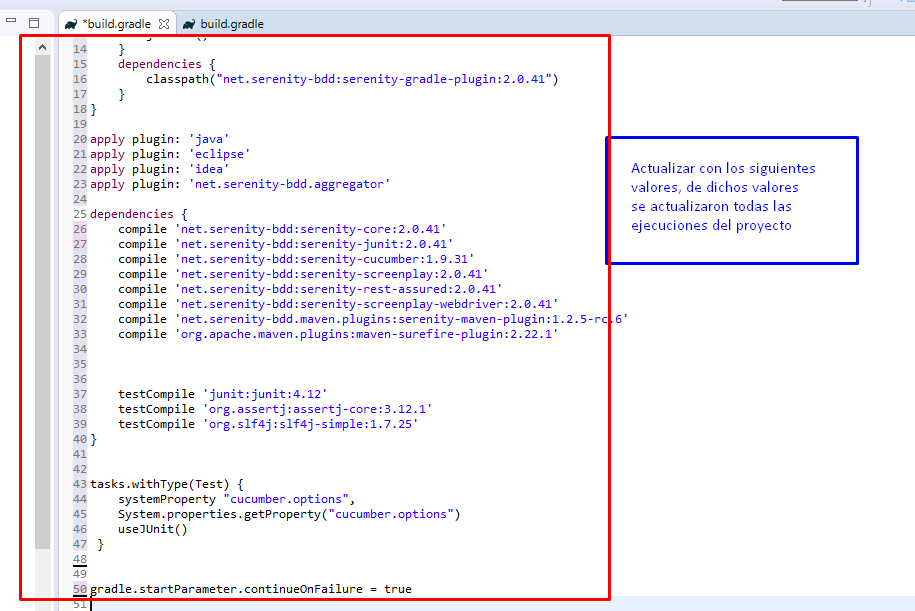
El segundo paquete se encuentra en la ruta:

src/test/java

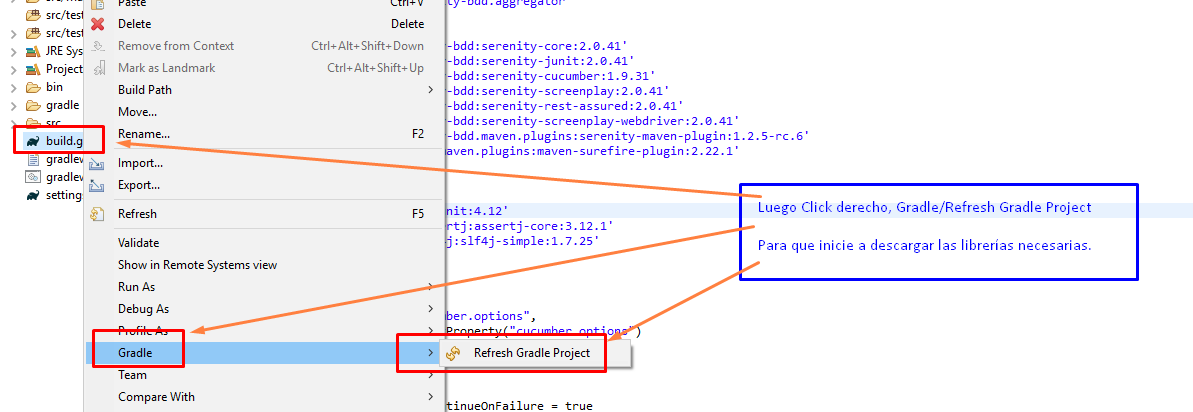


1. Actualizar el archivo build.gradle

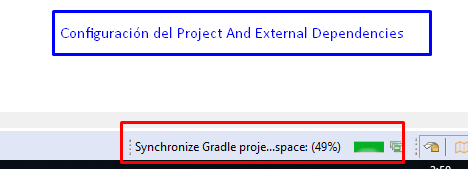




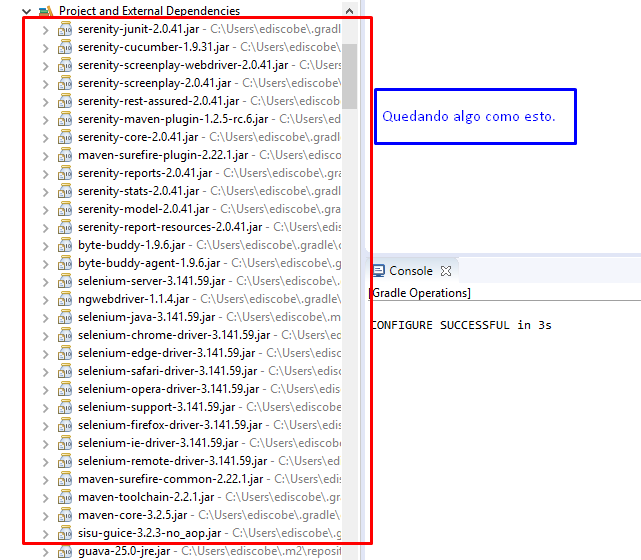
Luego de haber pegado las instrucciones, se procede a actualizar para las descarga de las dependencias



Validamos que las dependencias si se hayan actualizado en la carpeta de Project and External Dependencies



Nota: a veces carga de una la instrucción. En caso contrario cerrar todo y volver abrir la aplicación. Y continuar el proceso.



1. Creación de las carpetas para el arquetipo

Deben para esto debe de existir tres carpetas principales

Src/main/java

Src/test/java

Src/test/resources

**En la carpeta de Src/main/java va ir:**

co.com.tutorialsninja.www.exceptions

co.com.tutorialsninja.www.interactions

co.com.tutorialsninja.www.models

co.com.tutorialsninja.www.questions

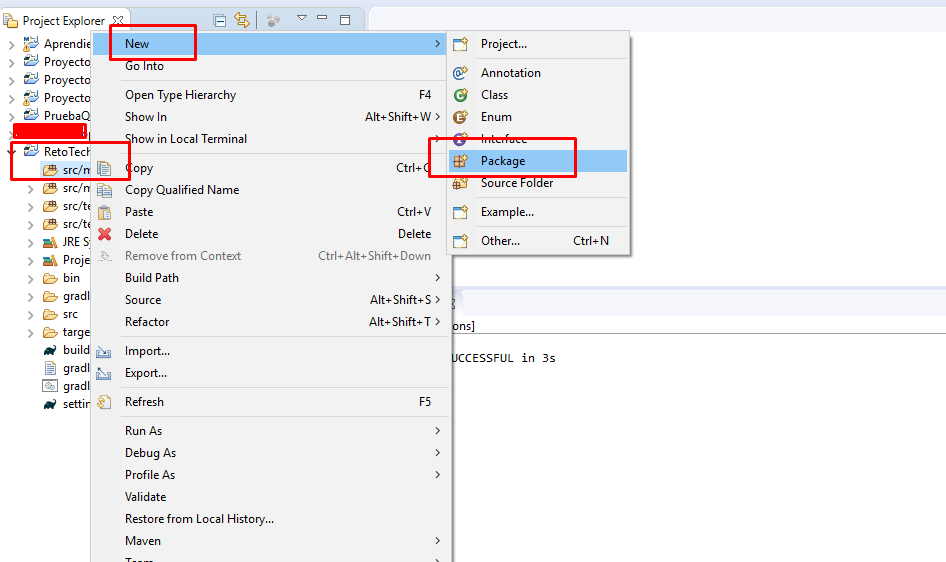
co.com.tutorialsninja.www.tasks

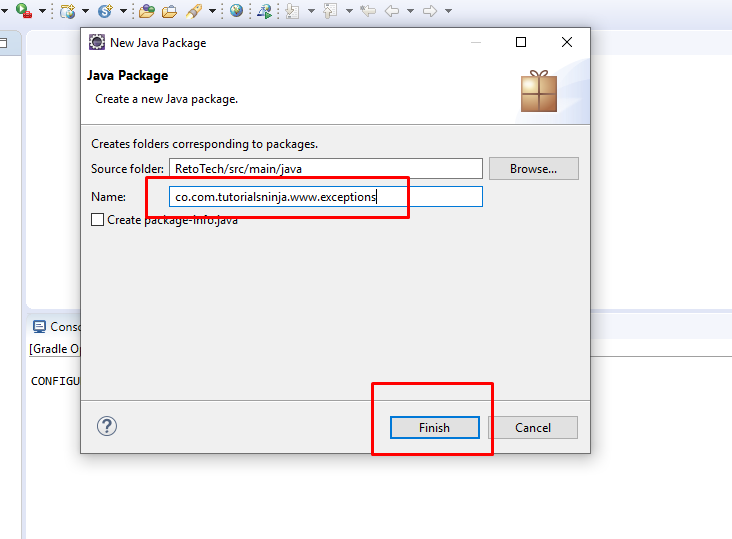
co.com.tutorialsninja.www.userinterfases

co.com.tutorialsninja.www.utils

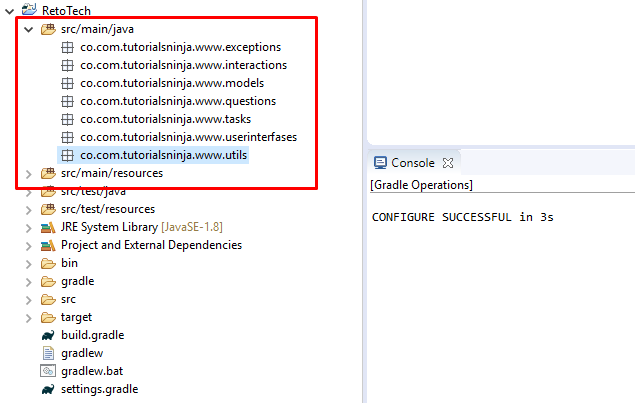
Nota importante: En mí ejemplo, quise llamarlo de esta forma, con el cliente bajo la estructura de nombramiento que él tenga.

Quedando algo como esto.





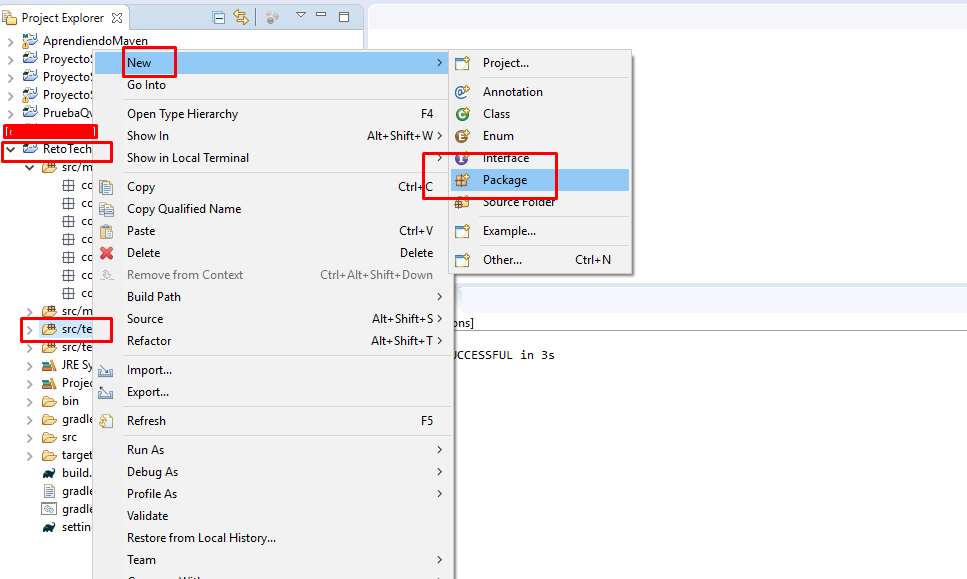
Realizar con cada uno de los 7 nombres los mismo pasos, quedando algo como lo siguiente:

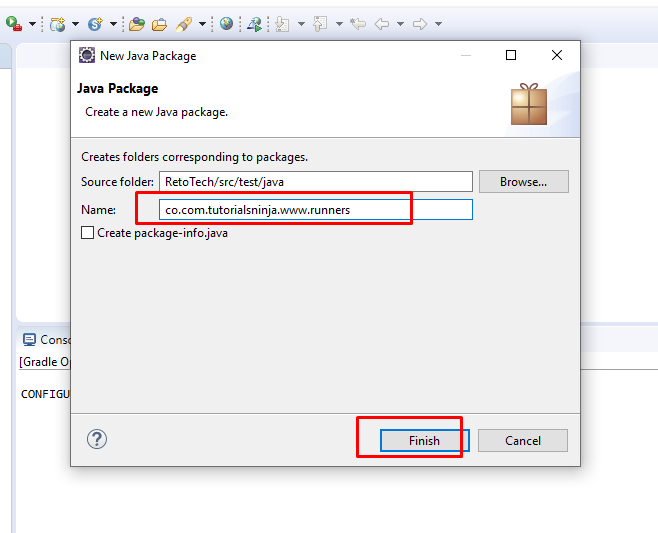


**En la carpeta de Src/test/java va ir:**

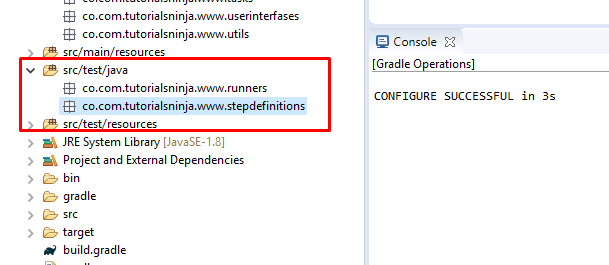
co.com.tutorialsninja.www.runners

co.com.tutorialsninja.www.stepdefinitions



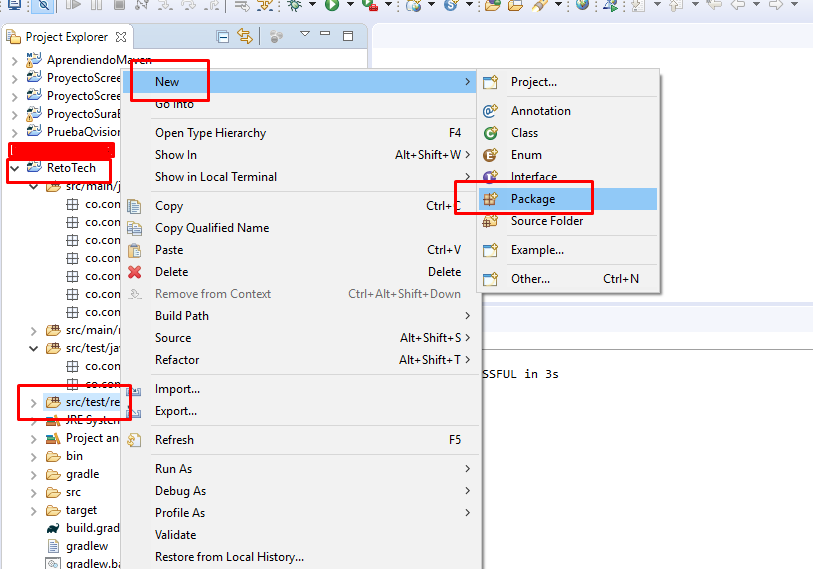


Quedando algo como esto.

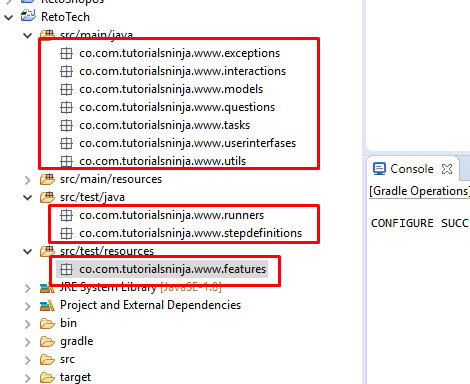


**En la carpeta de Src/test/resources va ir:**

co.com.tutorialsninja.www.features



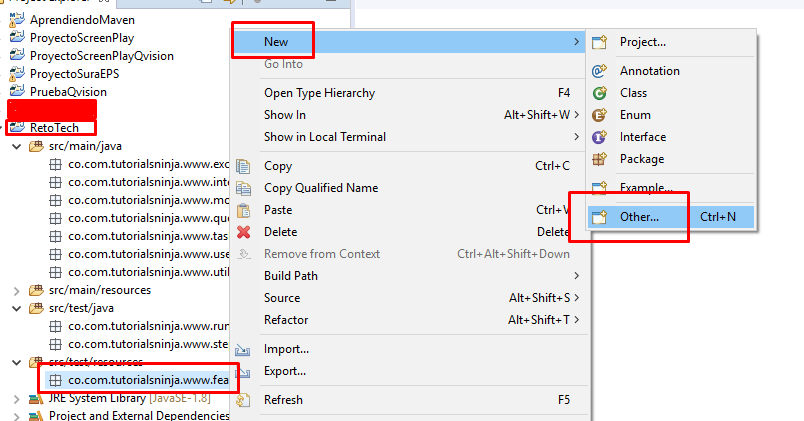
Quedando totalmente construido el arquetipo de la siguiente forma.

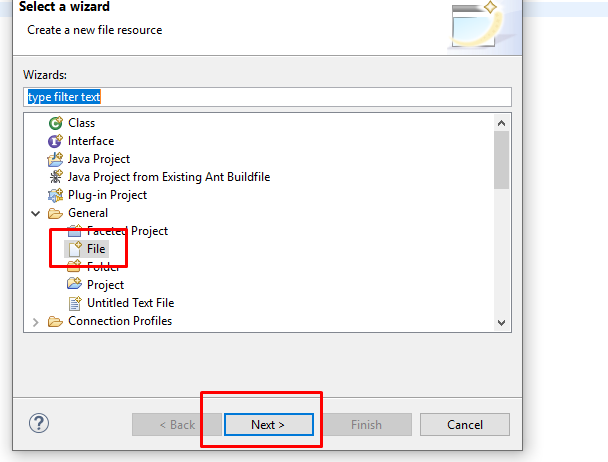


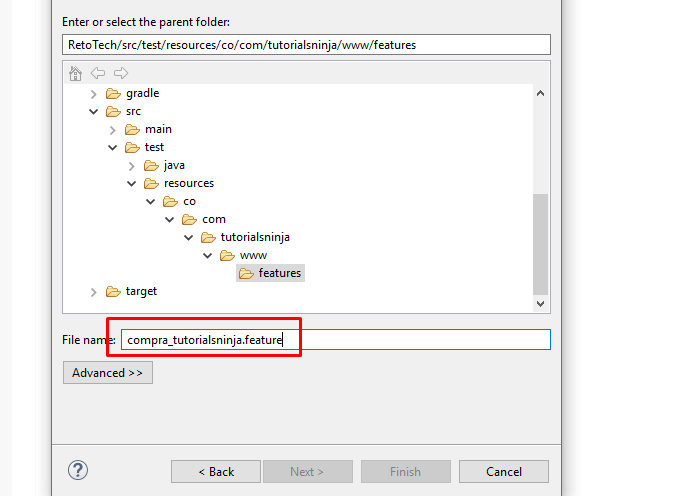
1. Creación del Features

En la carpeta de *src/test/resources* en el paquete que se acaba de crear *co.com.tutorialsninja.www.features* se creara un archivo FILE, en donde se escribirá todos los pasos necesarios para la ejecución del proyecto.

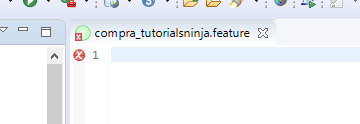
Ese archivo de tipo FILE tendrá como nombre compra\_tutorialsninja.feature

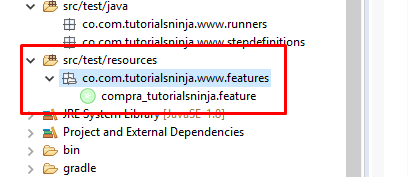






Quedando algo como esto.





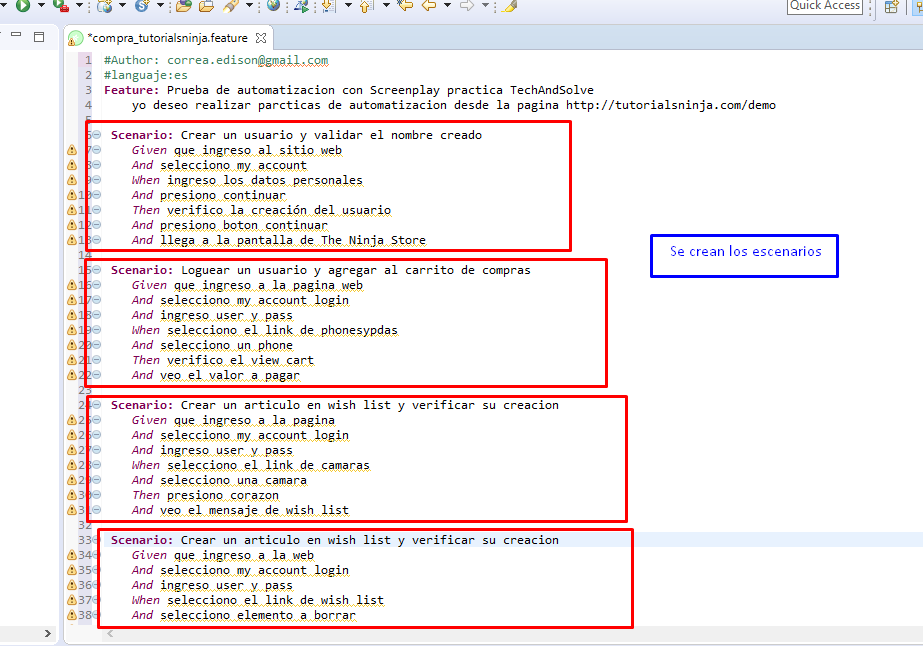
Para este ejemplo se plantean 4 escenarios los cuales son:

* ingresar ingresar a la aplicación y crear un usuario para compras.Verificar que el nombre usuario corresponde con el usuario que ingresó.
* Loguearse en la aplicación y agregar un artículo de la sección de “ phones &

PDAs”.verificar que el artículo fue agregado al carrito exitosamente.

* Agregar el artículo a la wish list y verificar que se haya agregado correctamente
* Eliminar el artículo del carrito y de la wish list y verificar que hayan quedado vacío.

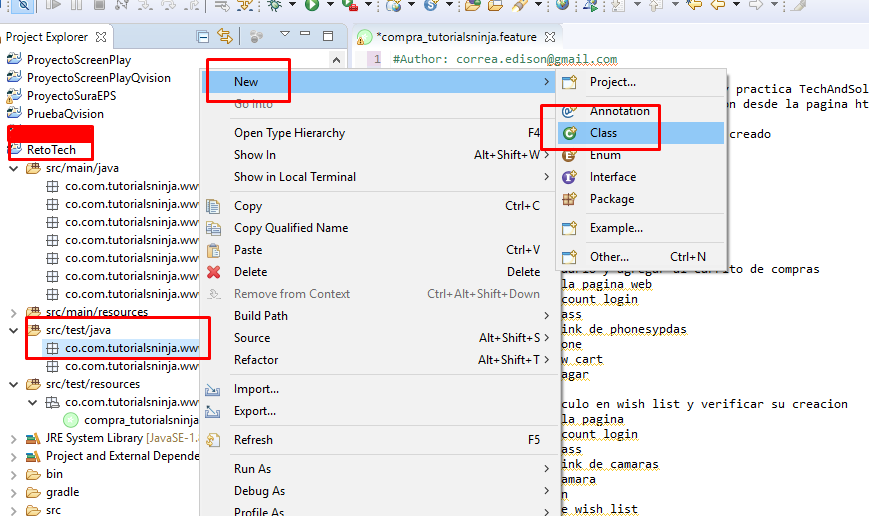
Quedando de la siguiente manera

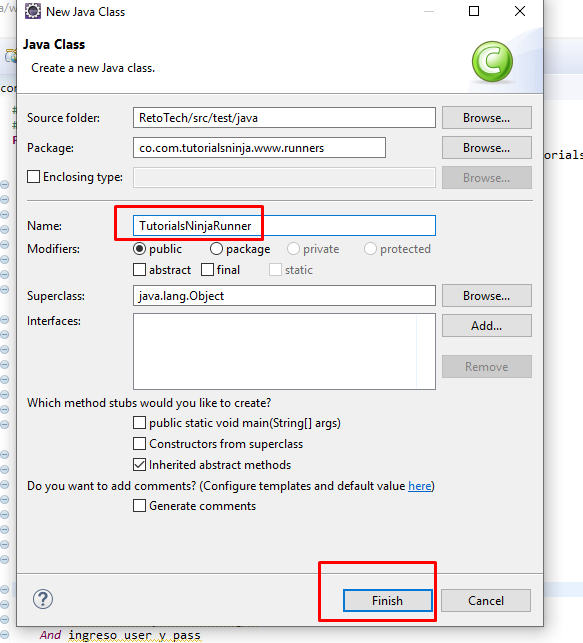


1. Creación de la clase ejecutará el proyecto

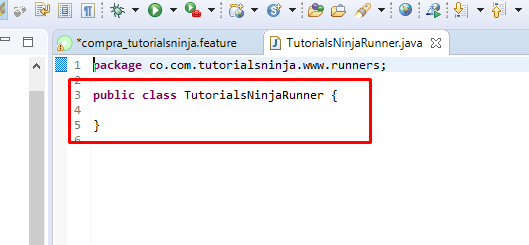
Desde la ruta src/test/java en el paquete co.com.tutorialsninja.www.runners

Se creara una nueva clase llama TutorialsNinjaRunner y desde este sitio se realizará la ejecución de cada uno de los escenarios



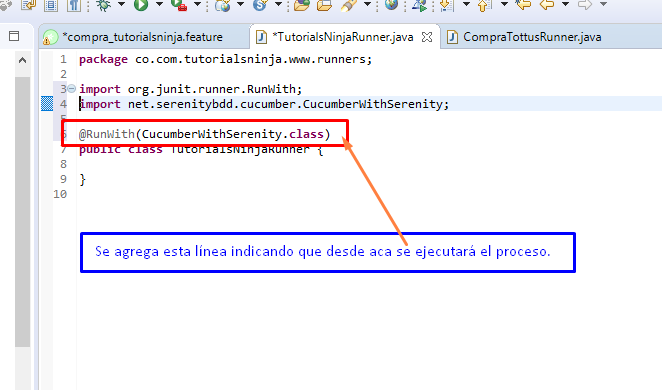


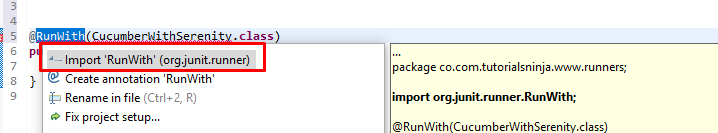
Quedando algo así:



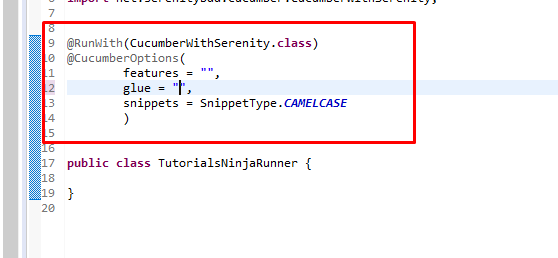
Se agrega la línea de ejecución

Y el aplicativo sacará los mensaje de importar librerias necesarias

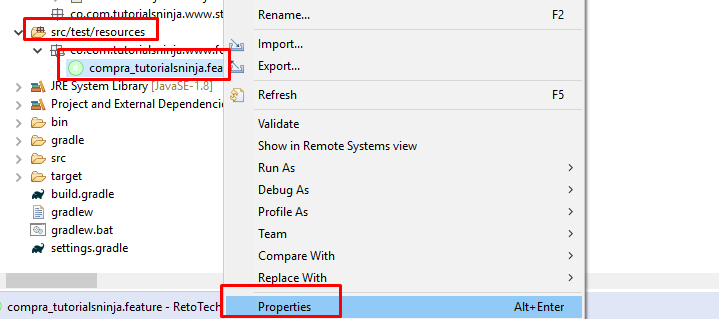


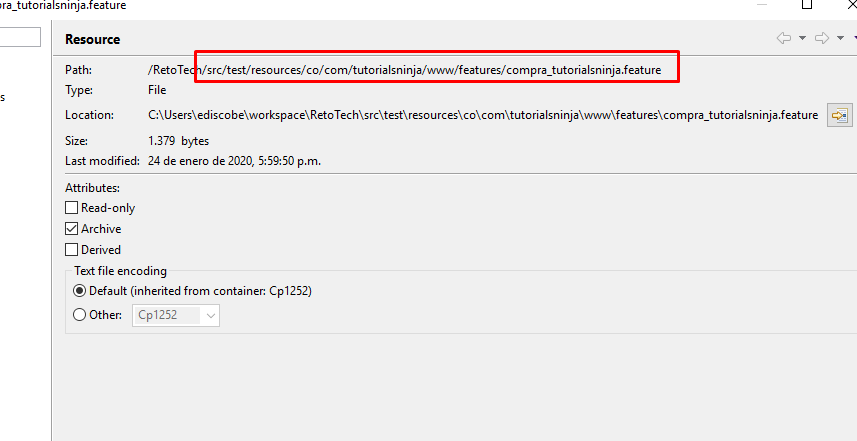


Se escribe las líneas de los features, glue y de los snippets, de la siguiente forma

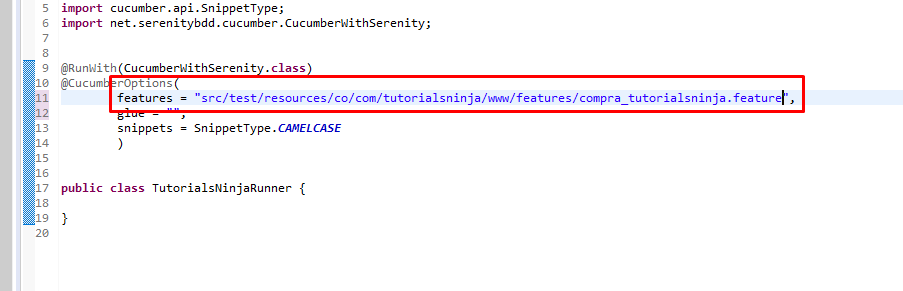


El features se saca de la siguiente dirección.

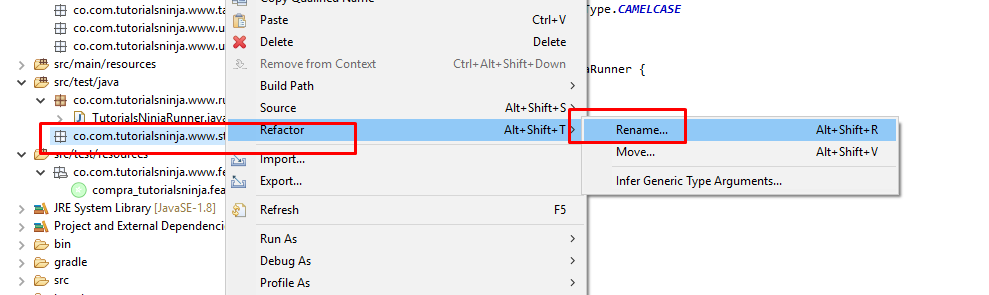


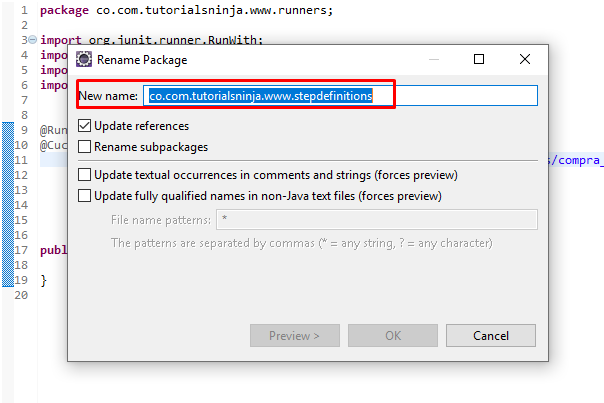


Quedando algo como esto:

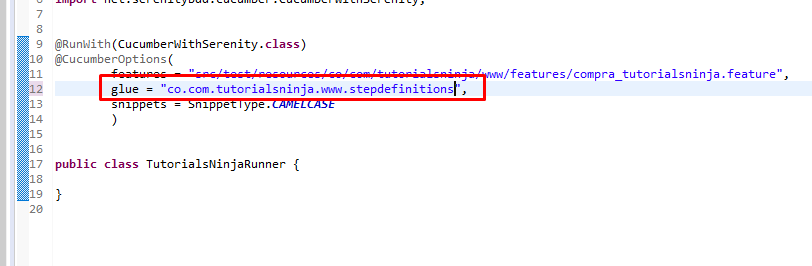


Y el glue del stepdefinitions

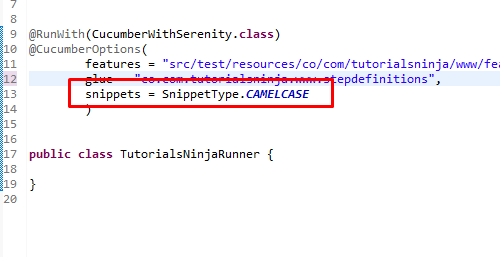




Quedando algo asi

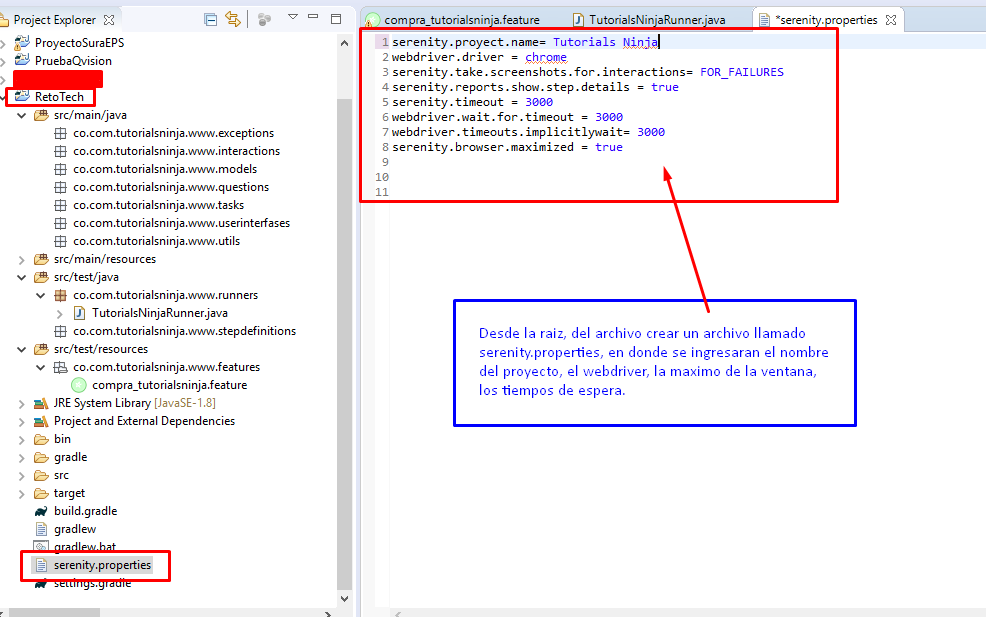


Con anotación en CAMELCASE

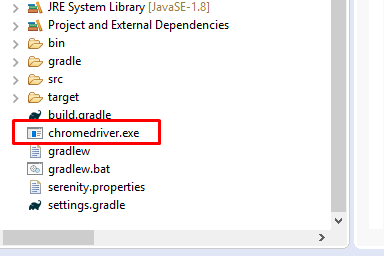


Configuraciones adicionales.

La creación del archivo serenity.properties



Desde la raíz adjuntar el ChromeDriver, el cual es el encargado de visualizar la secuencia de pasos de las pruebas automatizadas

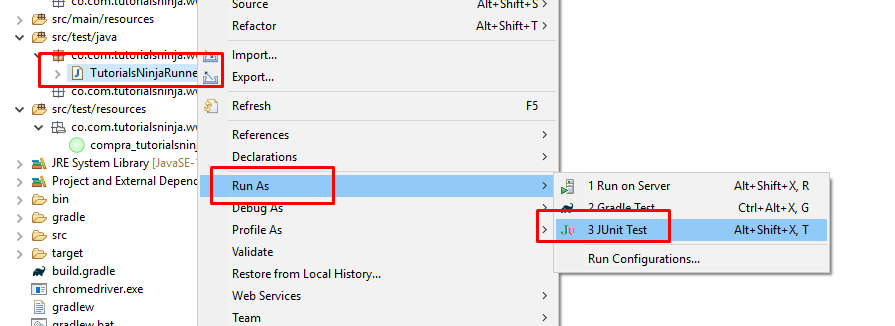


Nota: se baja la versión desde la página de chromedriver

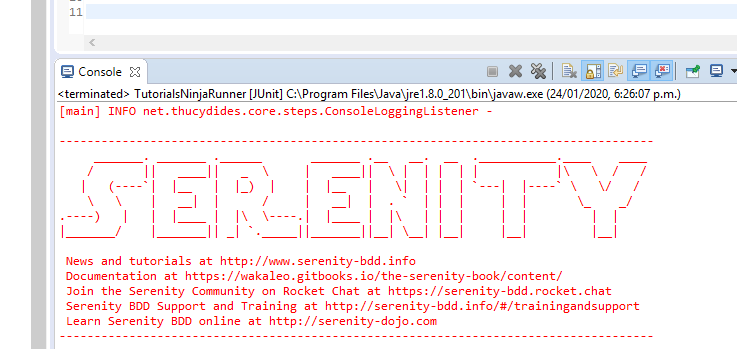
1. Ejecución de la Clase

Luego de haber realizado todos los pasos anteriores se ejecuta el proyecto desde el Runner.

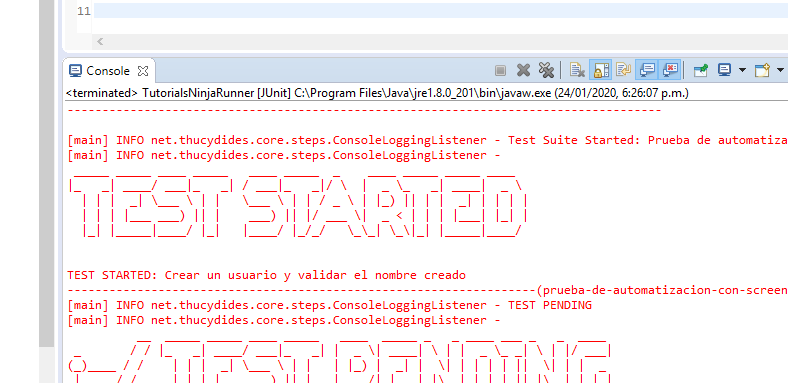
Click derecho a la clase, Run As, Junit Test



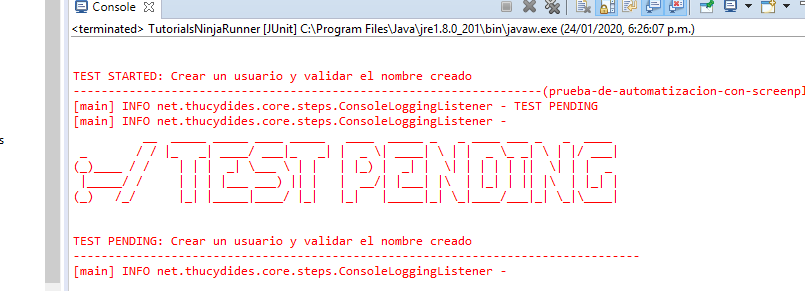
El aplicativo inicia generando los snippets para la ejecución de la pruebas y escenarios



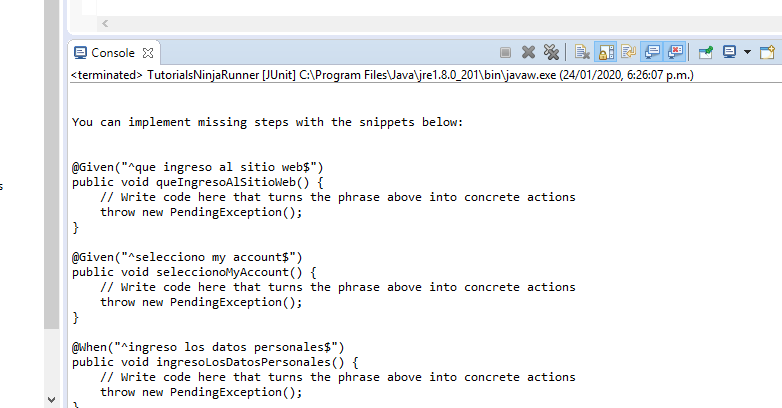
Indicando cada uno de los escenarios planteados inicialmente



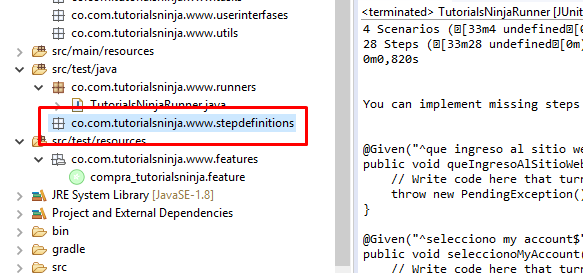
He informa que esta pendiente de ejecución

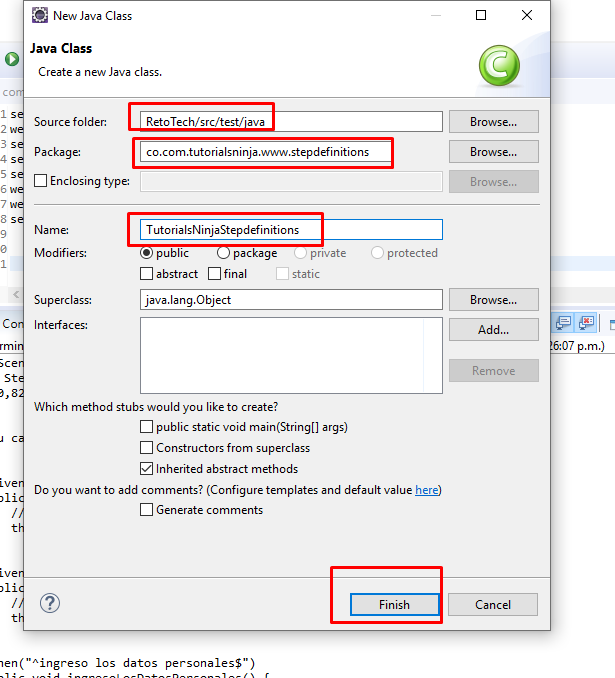


Se sacan los snippets que el aplicativo aconseja

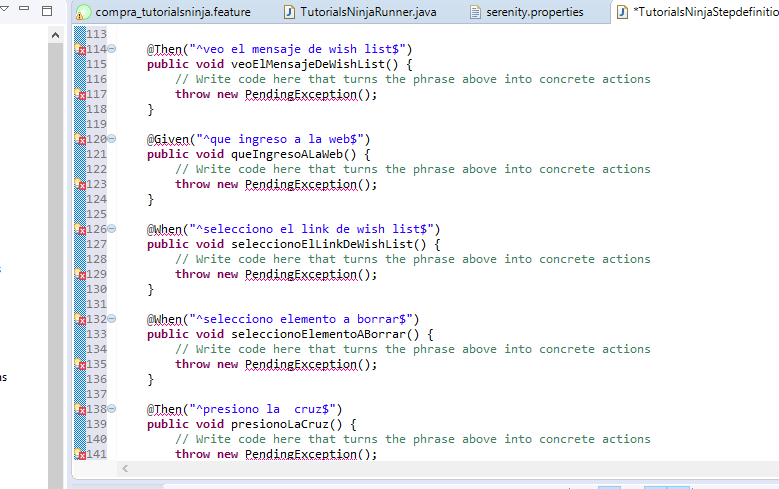


En el paquete de stepdefinitions, creo una clase llamada TutorialsNinjaStepdefinitions y se pega los snippets generados previamente.

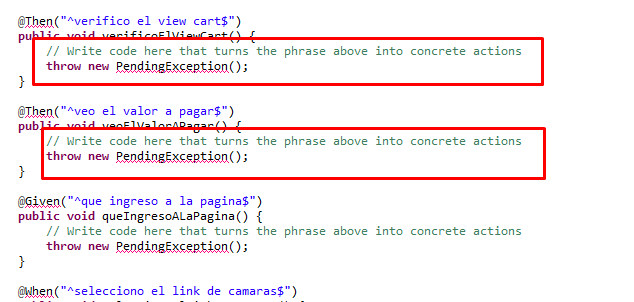




Quedando algo como esto

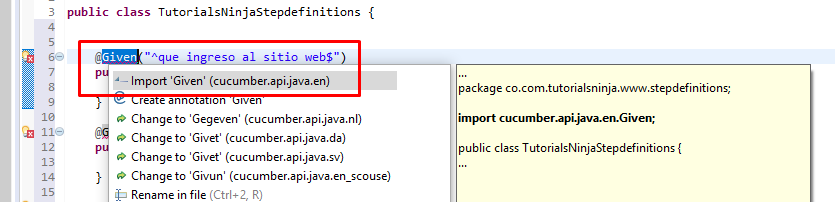


Elimino las líneas sobrantes

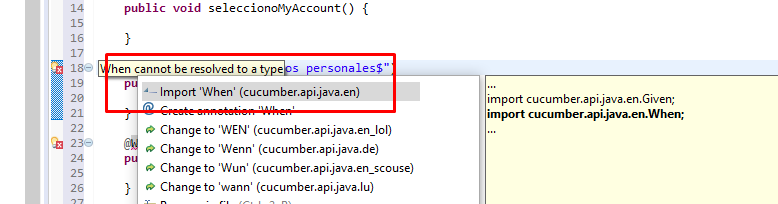


Se importan las librerias correspondientes

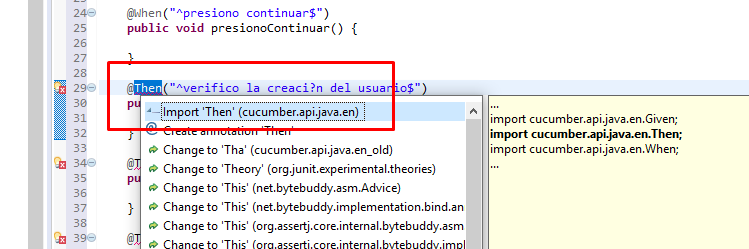
Given



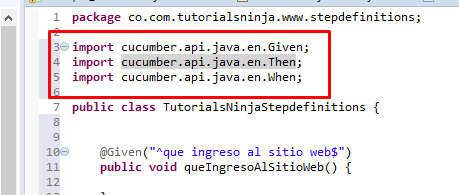
When



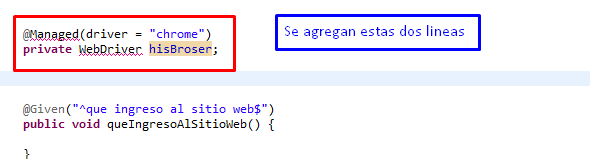
Then



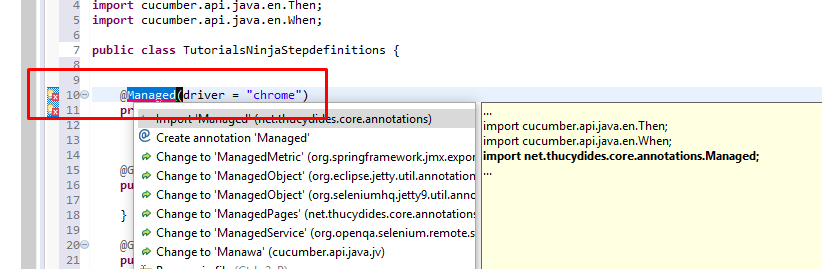
Quedando algo como esto

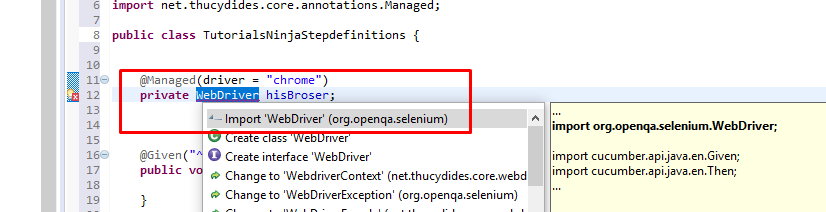


Se agregan estas dos líneas a continuación las cuales indican que utilizare a Chrome como visualizador de las pruebas

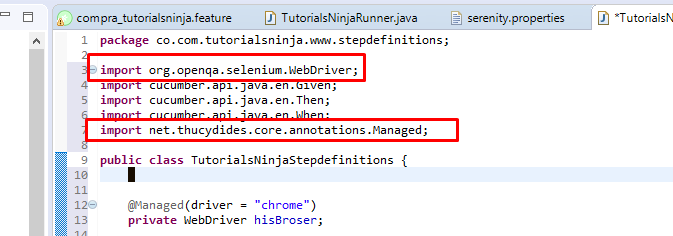


Se agragan las librerias necesarias





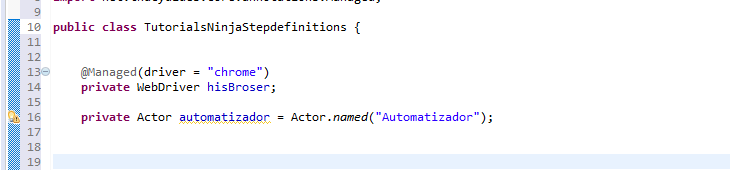
Quedando algo asi

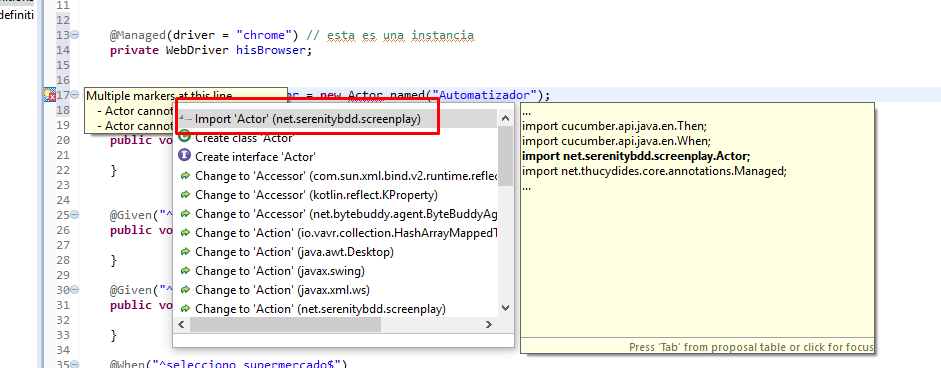


1. La creación del actor del proyecto

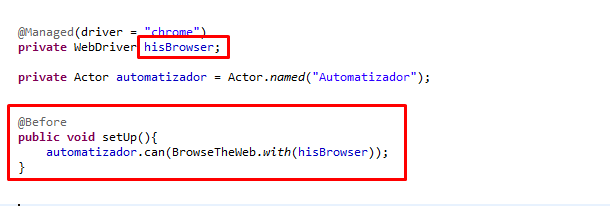
Este es el rol que la aplicación entenderá que ejecutará los pasos a realizar

private Actor automatizador = Actor.named("Automatizador");



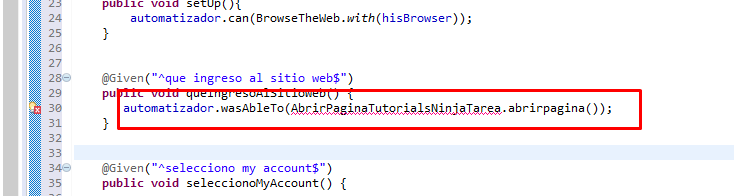


Se hace una nueva línea de código indicando quien es el que ejecutara el código, para mi ejemplo automatizador.



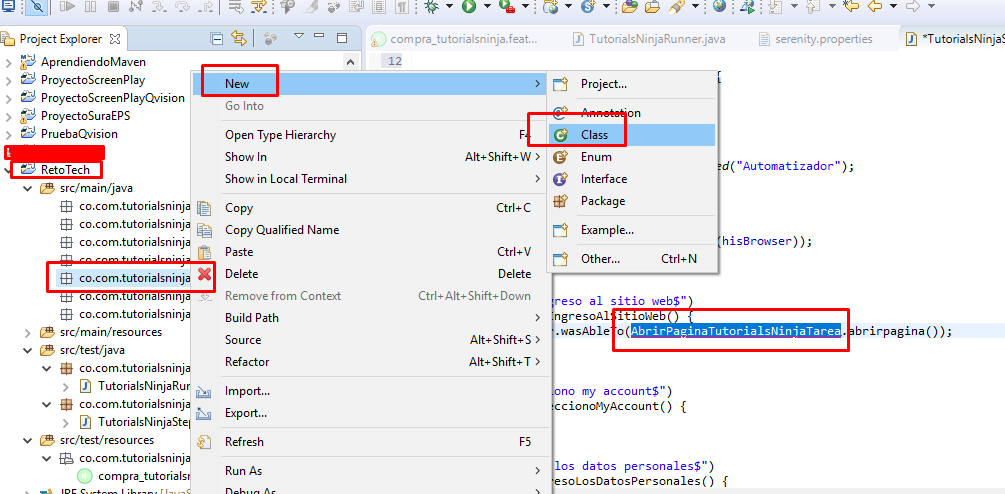
1. La creación de los pasos y las clases

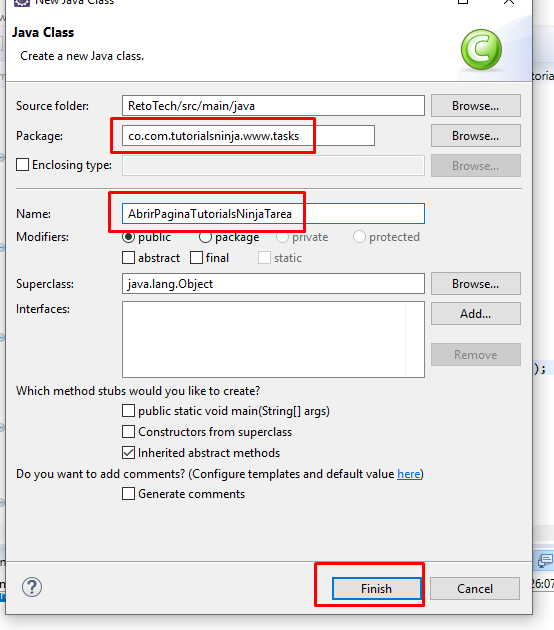
El primer paso de la ejecución es ingresar a la pagina web



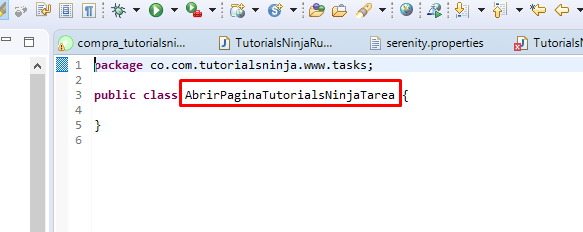
En la ruta src/main/java en el paquete co.com.tutorialsninja.www.tasks crear una clase llamada

AbrirPaginaTutorialsNinjaTarea esta Clase será la encargada de abrir la pagina web inicial.

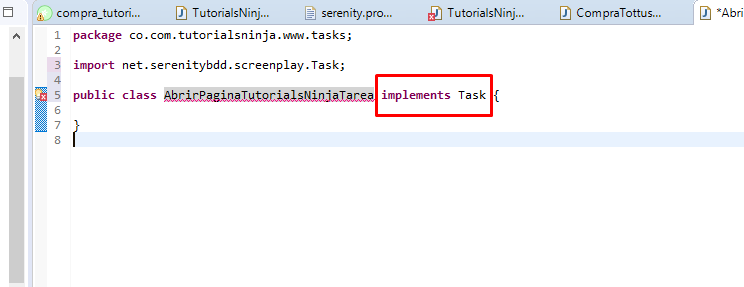




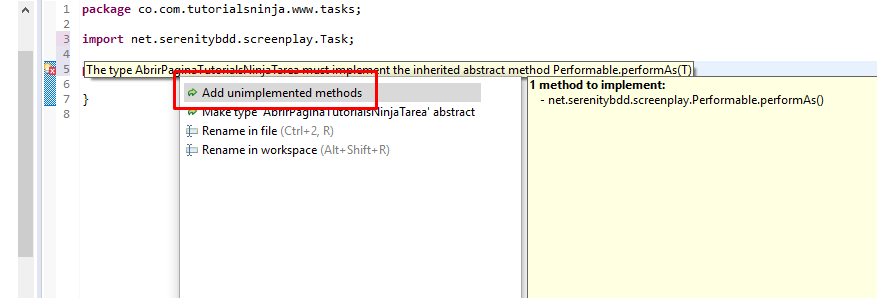
Quedando algo como esto



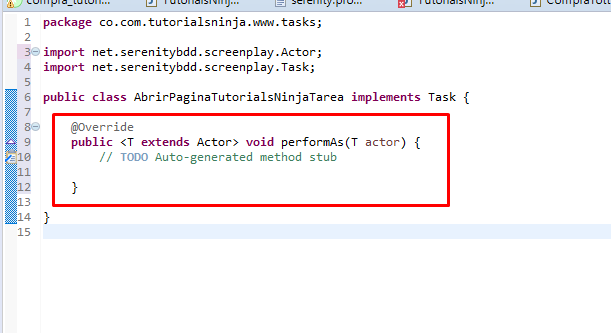
Se agrega la siguiente line adicional, la cual indica que es una implementación de una tarea



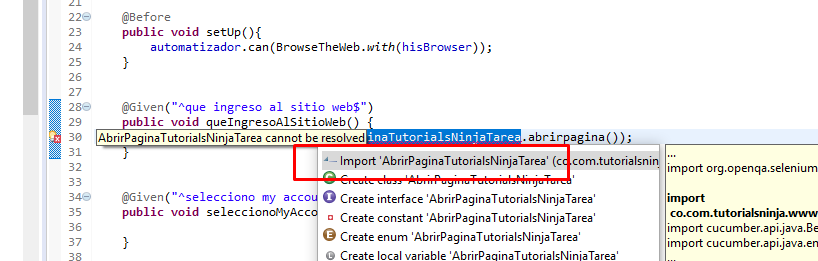
Se agregan los métodos de la implementación de la clase



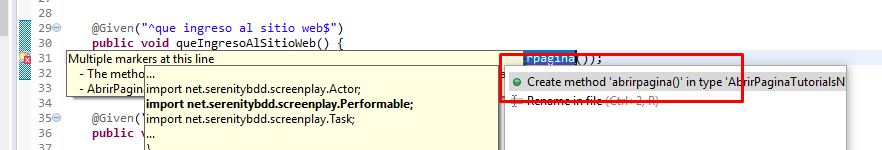
Quedando algo como esto



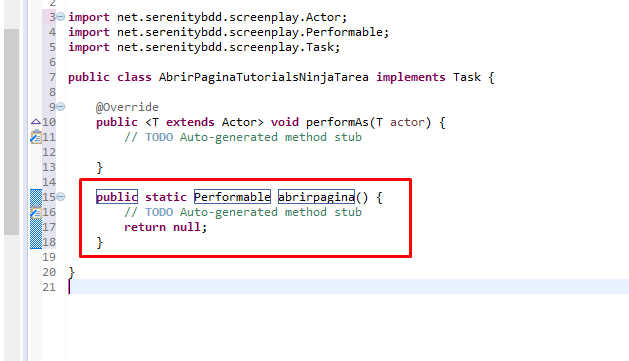
Nos devolvemos al stepdefinitions, se importa la clase AbrirPaginaTutorialsNinjaTarea



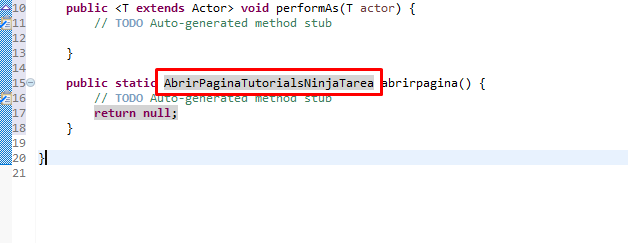
Si se crea el método de abrirpagina()



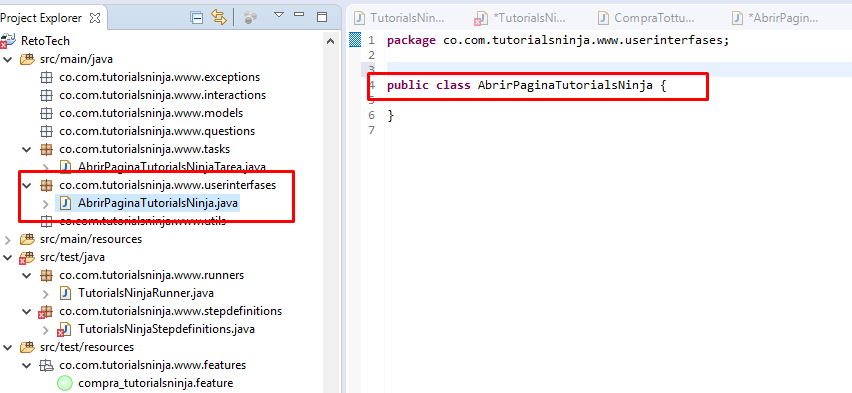
El aplicativo crea este método



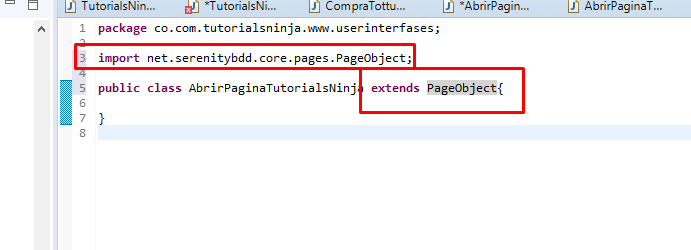
Pero cambiamos el nombre de Performable por el de AbrirPaginaTutorialsNinjaTarea, quedando algo como esto



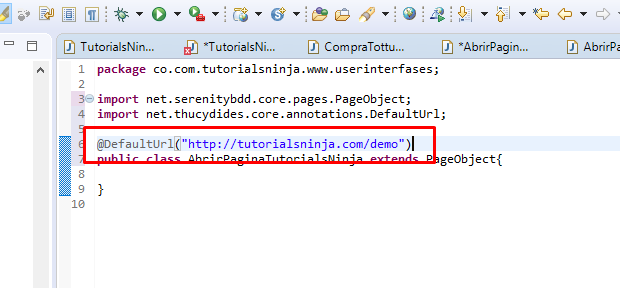
Se crea una nueva clase que será la encargada de la interfase ( donde este la dirección de la página)



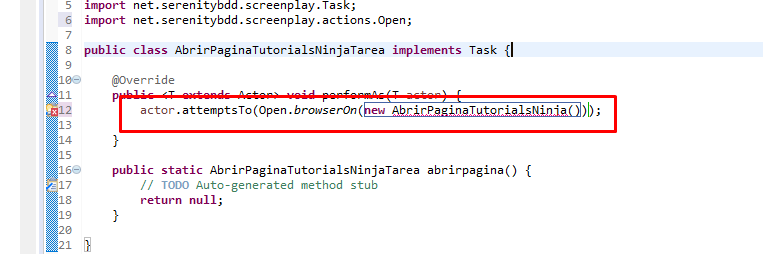
Agrego la siguiente línea e importo las librerías



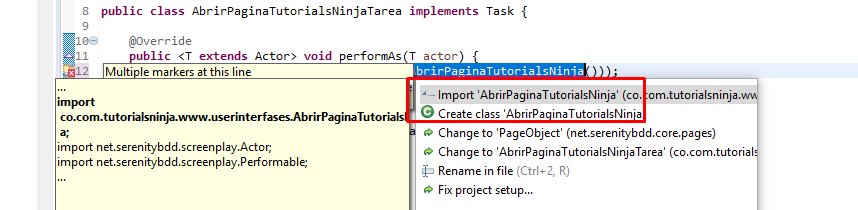
Se agrega la siguiente línea, la cual es la que indica a donde debe de ir la instrucción.



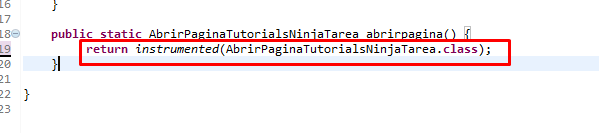
De la siguiente línea indica de donde va a extraer la información para abrir la página



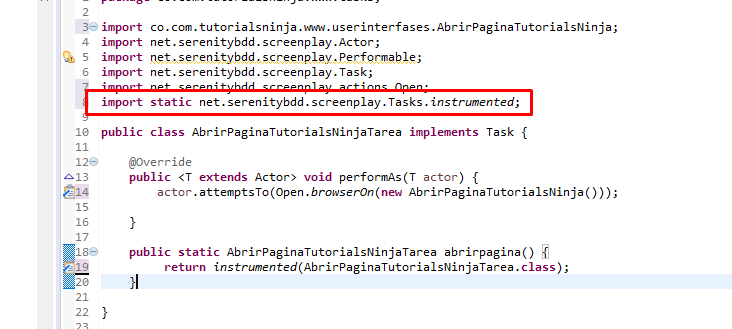
Se importa la librería necesaria de la clase



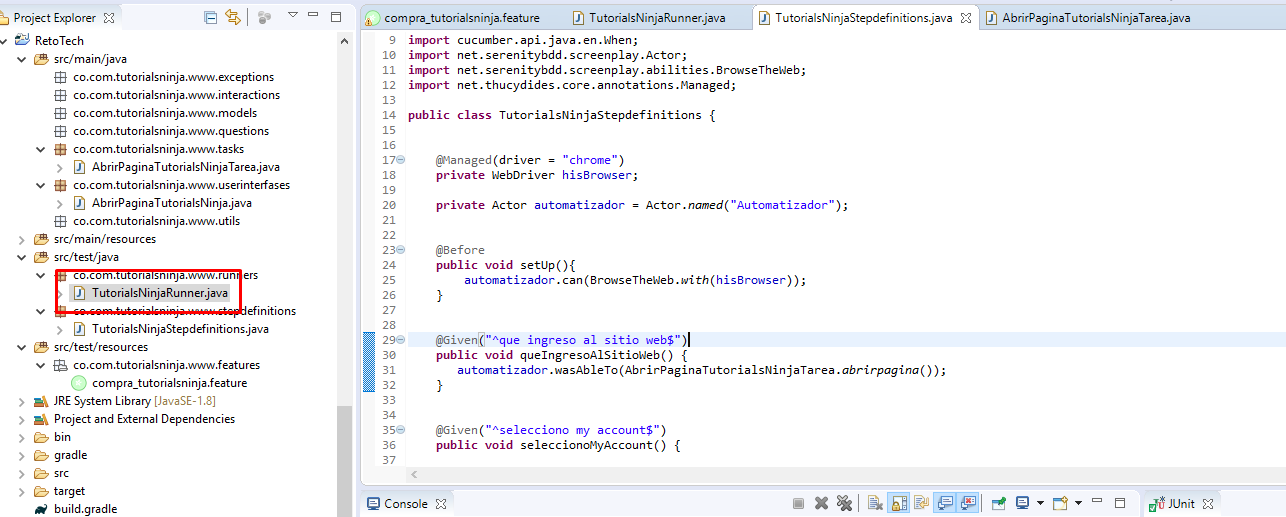
Se agrega la siguiente línea, la cual es la encargada de retornar



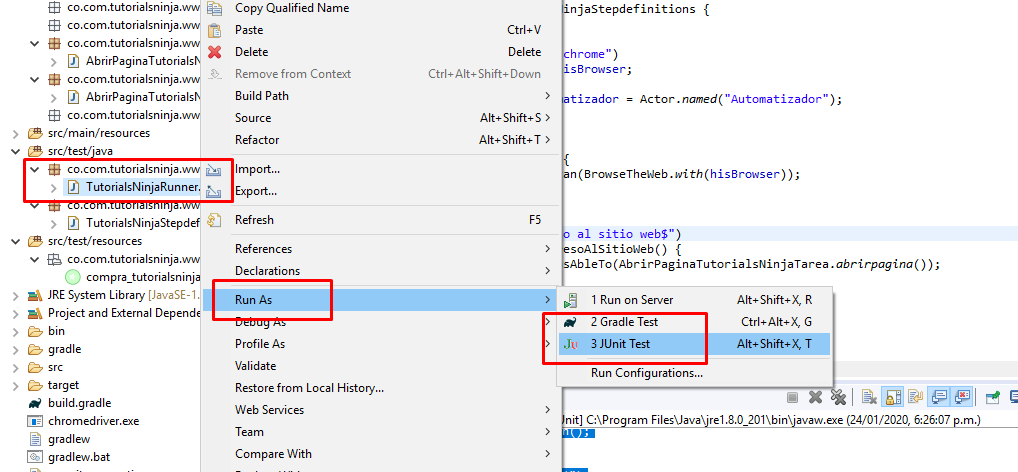
Y se importa la librería

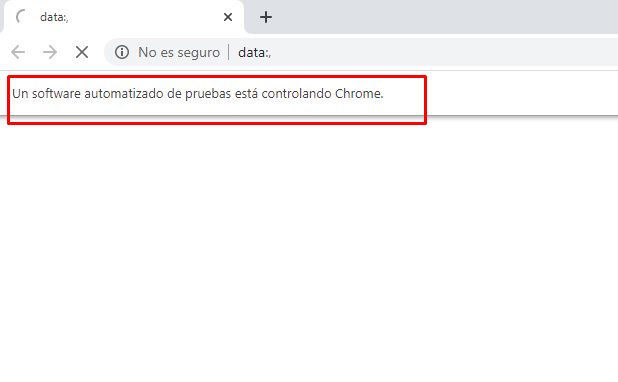


Luego de haber realizado cada uno de los pasos anteriormente desarrollados se inicia la primera ejecución del proyecto



En la clase Runner click derecho / Run As / Junit Test

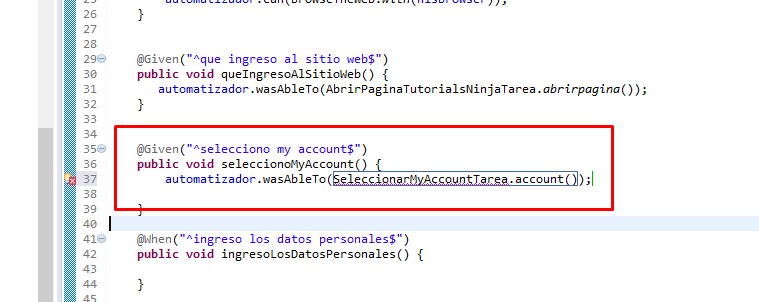




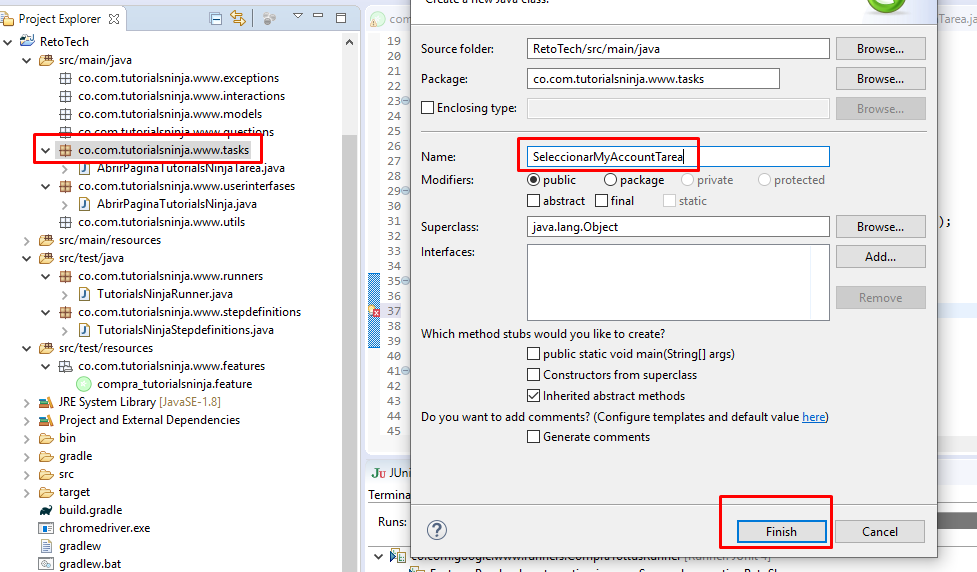


Creación del segundo paso

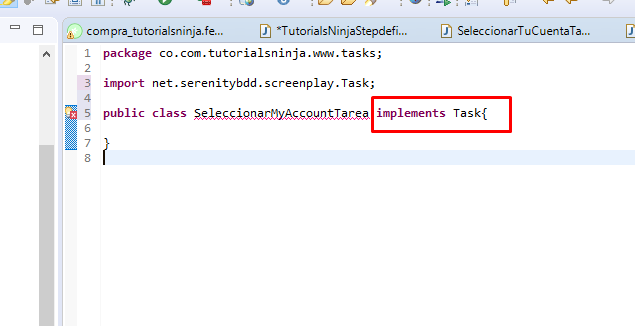
En el stepdefinitions agrego la nueva instrucción.



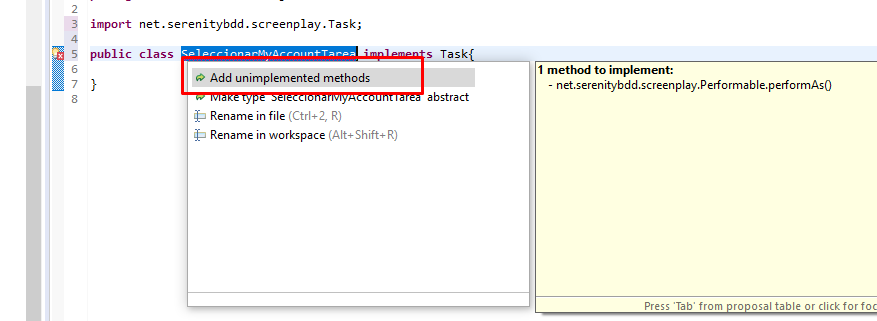
Igual que el paso anterior en el paquete de tasks creo la nueva clase llamada SeleccionarMyAccountTarea

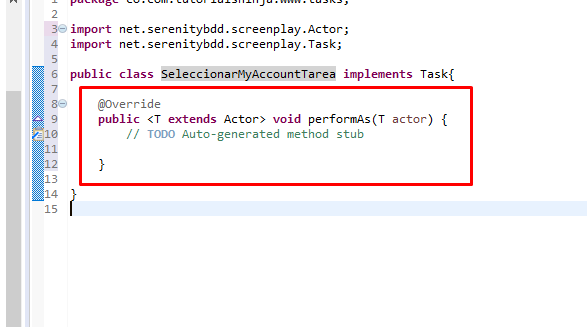


Igualmente que en la clase anterior agrego el texto implents Task

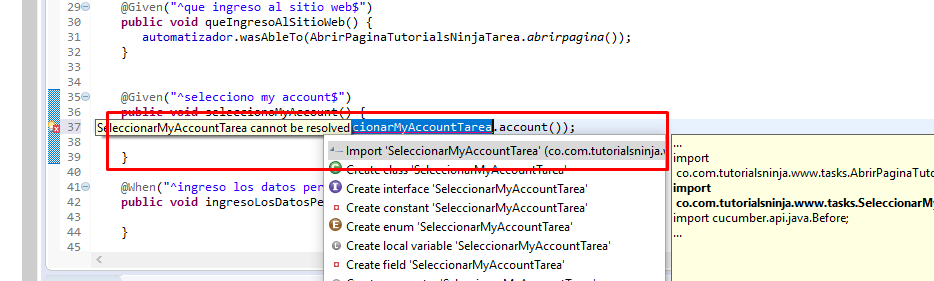


Implemento el nuevo metodo

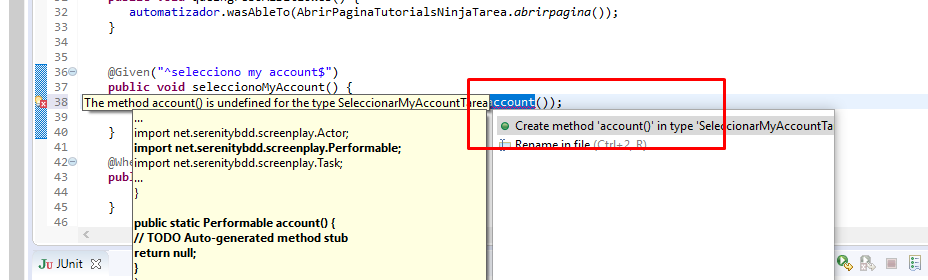


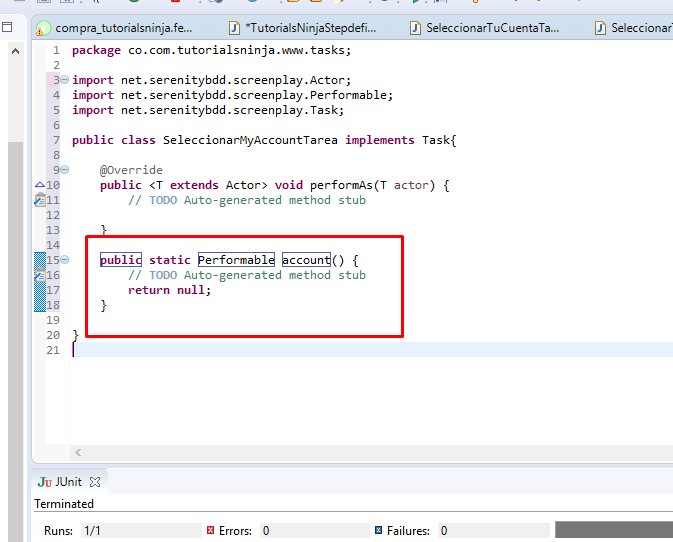


Me dirijo al stepdefinitions y cargo los métodos y librerias faltantes

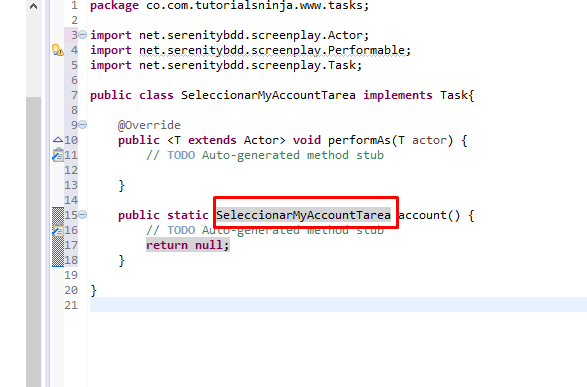


Creación del método

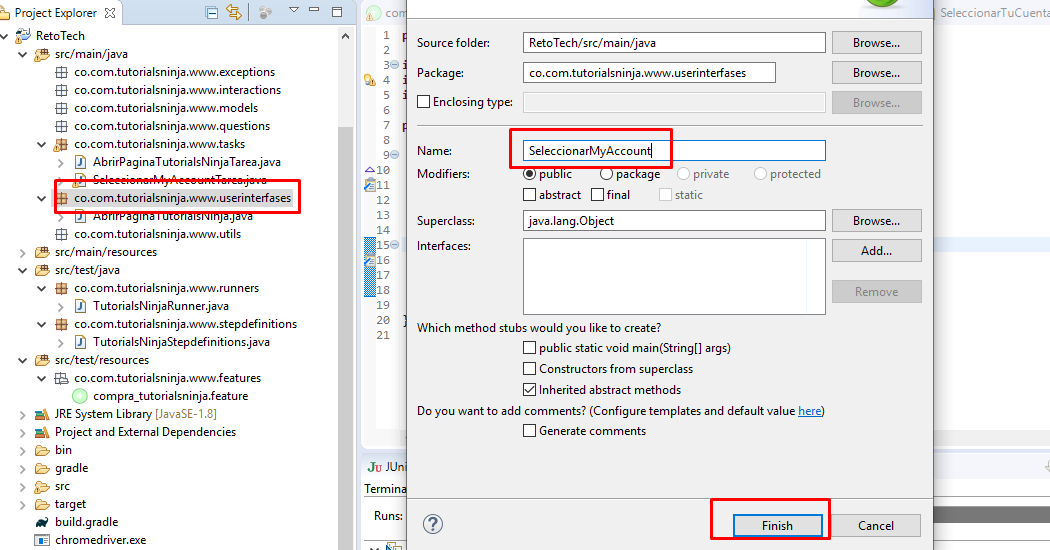




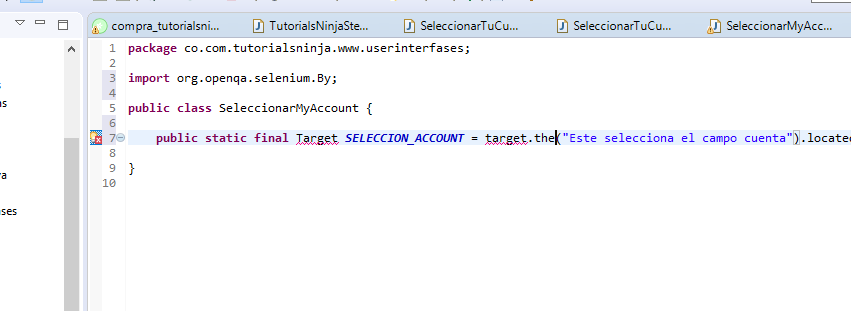
Se cambia el nombre



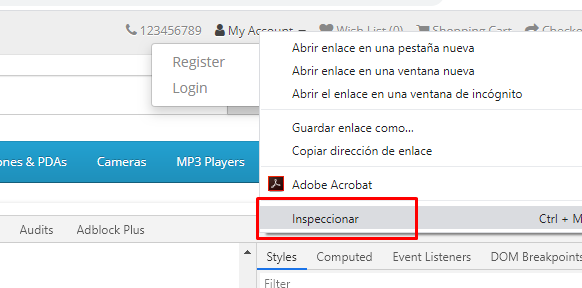
Se crea la clase en userinterfases, llamada SeleccionarMyAccount la cual tendrá toda la información de localización del link de My Account

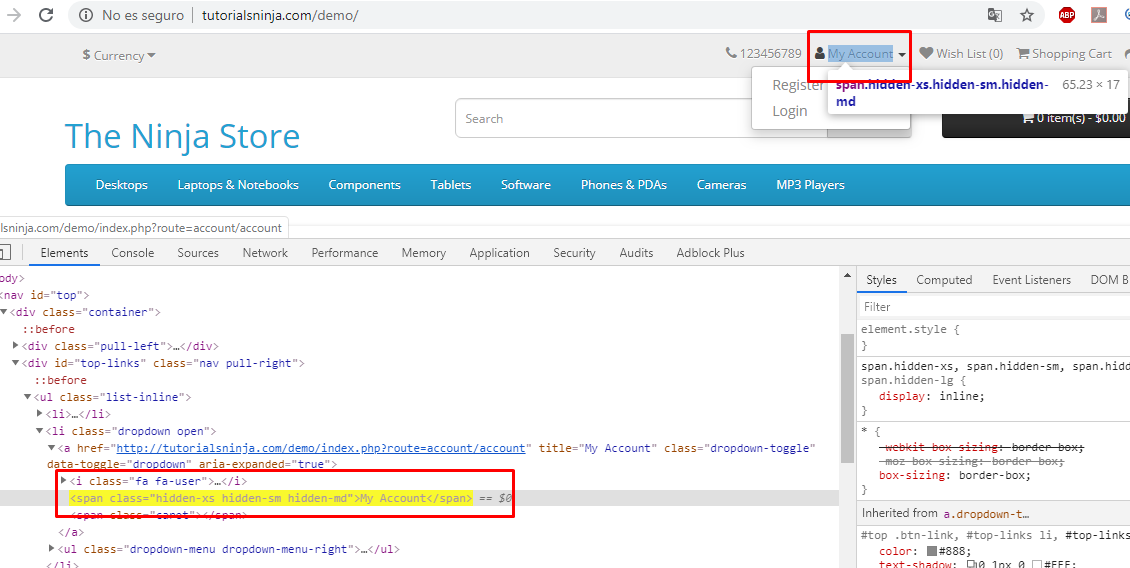


Se crea la clase con la siguiente información:



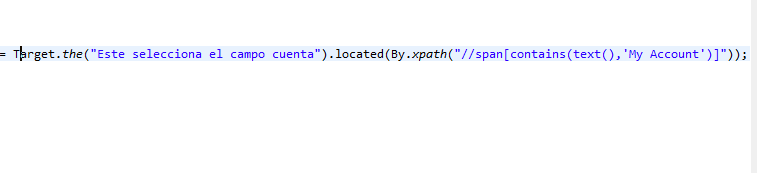
Se inspecciona el Xpath de localización del elemento



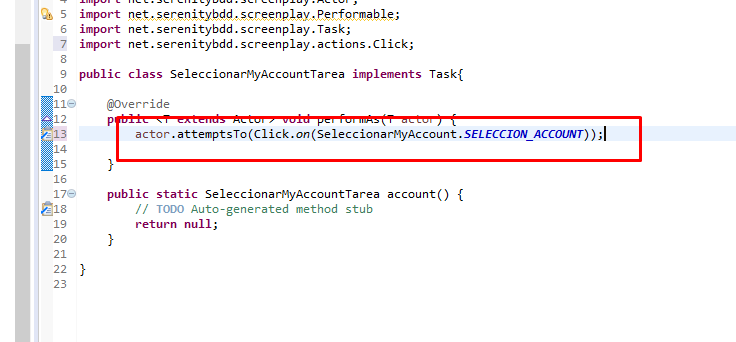


El Xpath culminado seria el siguiente:

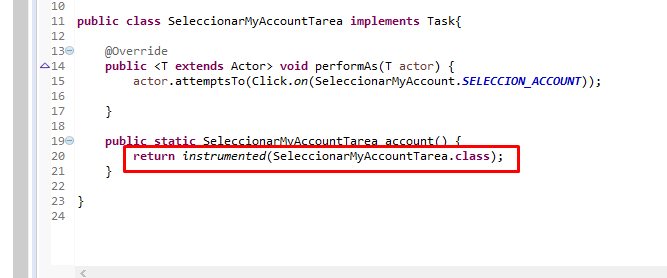
public static final Target SELECCION\_ACCOUNT = Target.the("Este selecciona el campo cuenta").located(By.xpath("//span[contains(text(),'My Account')]"));



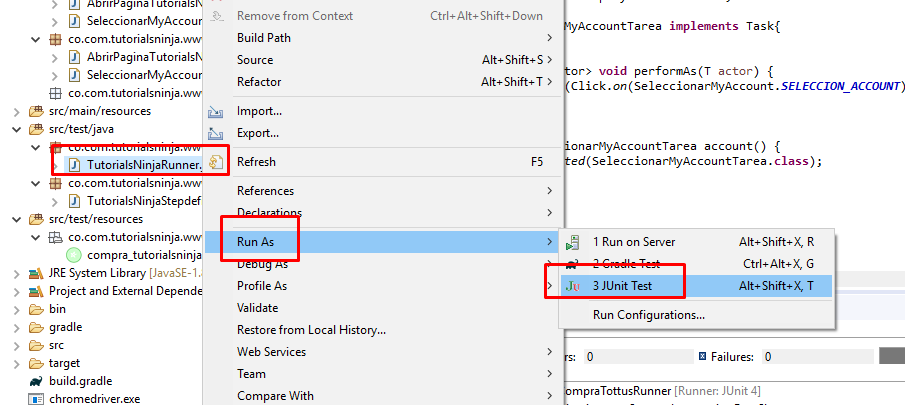
Se agrega la extensión en la clase de tarea

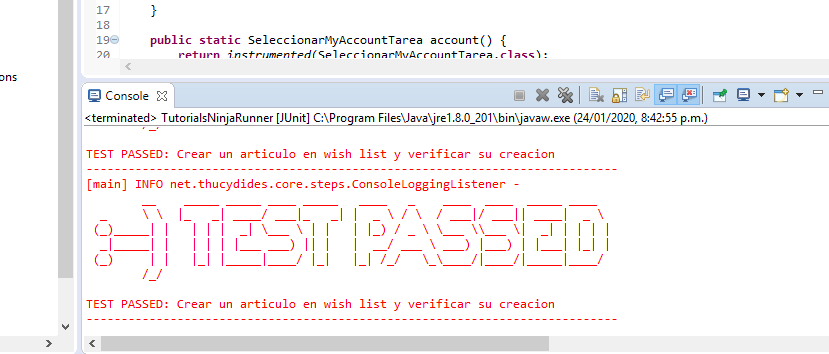


Y se retorna la clase principal

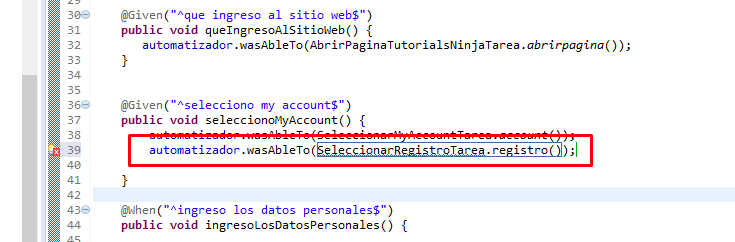


Se inicia la segunda ejecución del proceso

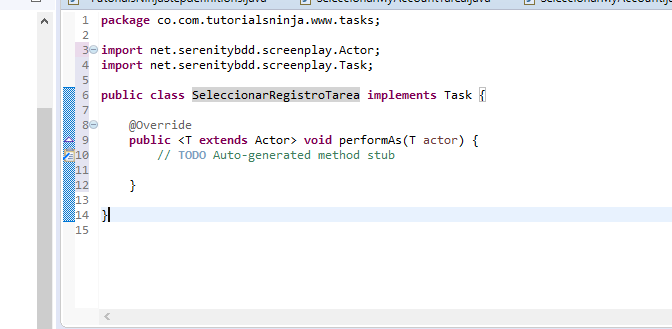




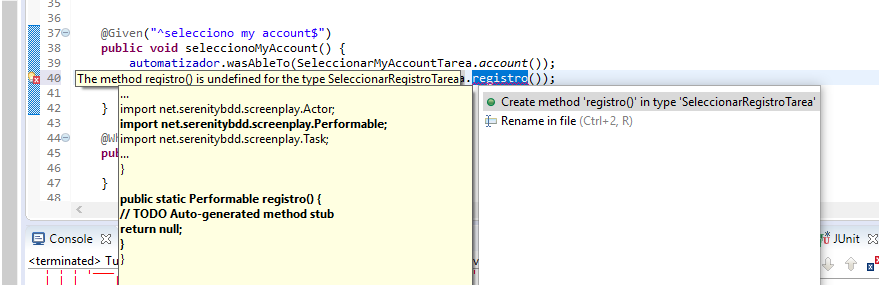
Seleccionar el link de registro, desde el stepdefinitions, creo la instrucción de ejecución.

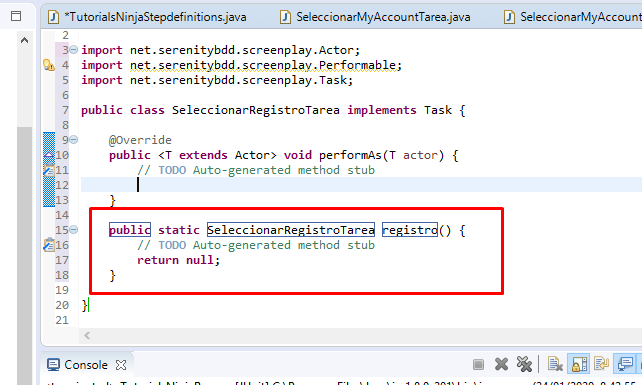


Se crea la clase en el paquete tarea.



Se crea el método desde el stepdefinitions





Se crea la clase en userinterfases



**POR TIEMPO NO PUDE CULMINAR EN LA DOCUMENTACION.**

**¡¡¡DISCULPEN!!!**