Contenido

[Objetivo 2](#_Toc48677926)

[Prerequisitos 2](#_Toc48677927)

[Instalación 3](#_Toc48677928)

[Usar IDE para aatDSL 10](#_Toc48677929)

# Objetivo

En este documento se describe el proceso de instalación y uso del marco de trabajo para usar attDSL.

# Prerequisitos

Antes de instalar la solución se requiere previamente los siguientes componentes:

* JDK 1.8 o superior

<https://www.oracle.com/java/technologies/javase-downloads.html>

* Android SDK – requerido incluir SDK Tools

<https://developer.android.com/studio/install>

* Maven 3.6.3 o superior

<https://maven.apache.org/download.cgi>

* AppiumServer

<https://github.com/appium/appium-desktop>

* Emulador Android (blueStack)

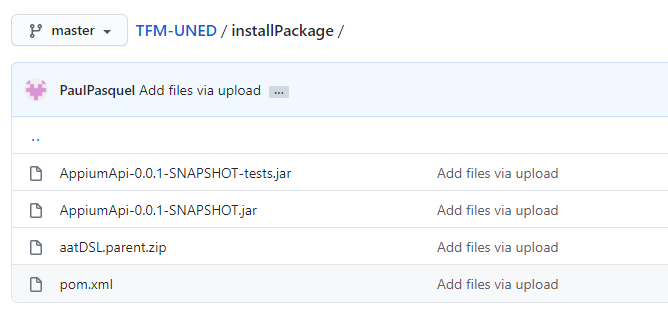
<https://www.bluestacks.com/download.html>

* Eclipse IDE (Java and DSL developers)

<https://www.eclipse.org/downloads/packages/release/2020-06/r/eclipse-ide-java-and-dsl-developers>

* Obtener componentes para instalación

<https://github.com/PaulPasquel/TFM-UNED/tree/master/installPackage>



* + AppiumApi-0.0.1-SNAPSHOT.jar librería Appium Api
  + AppiumApi-0.0.1-SNAPSHOT-tests.jar librería Appium Api
  + aatDSL.parent.zip proyectos eclipse Xtext
  + pom.xml Archivo Maven para proyecto aatDSL

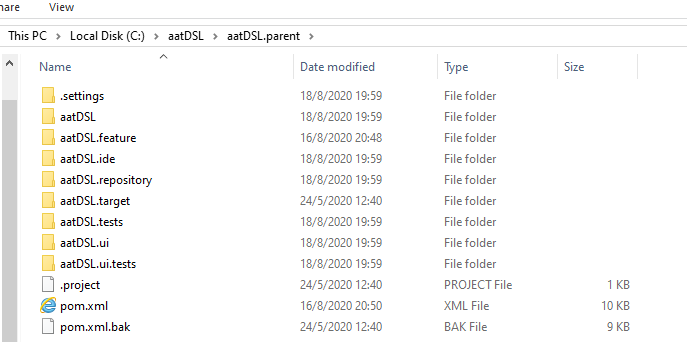
# Instalación

Para instalar los siguientes elementos, sigas las instrucciones que cada uno de estas aplicaciones proveen en sus respectivos sitios.

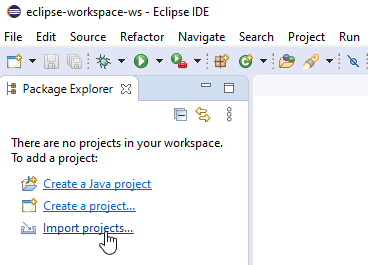
* JDK 1.8
* Maven 3.6.3
* Appium Server (Desktop)
* SDK Android
* Emulador Android - Bluestack
* Eclipse IDE (Java and DSL developers)

Posteriormente seguir los siguientes pasos descritos:

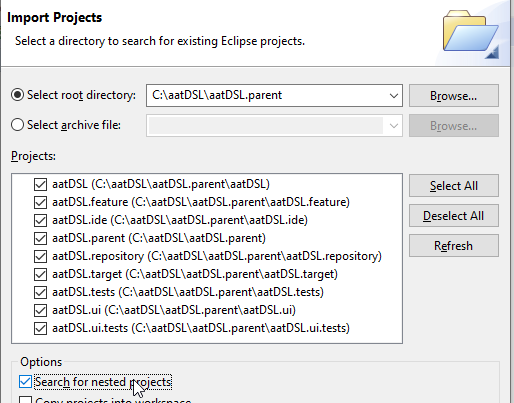
1. Descomprimir el archivo aatDSL.parent.zip



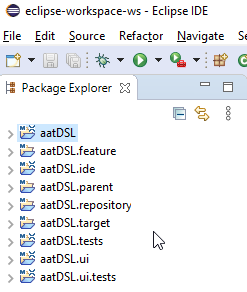
1. Ejecutar la instancia Eclipse IDE, e importar los proyectos de aatDSL.parent



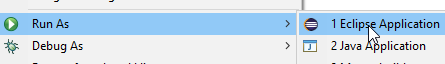
1. No olvidar seleccionar la opción “Search for nested Project”



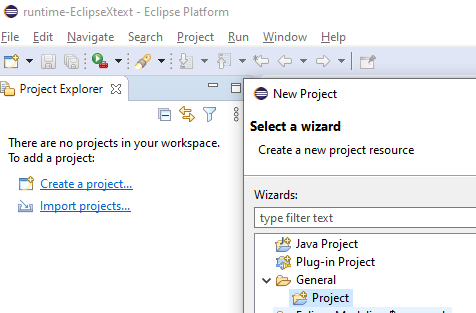
1. Seleccionar todos los proyectos internos, y presionar “Finish”



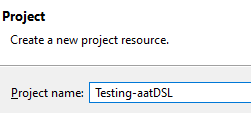
1. A continuación, seleccionar el proyecto aatDSL, clic derecho y seleccionar la opción “Eclipse Application” dentro de la opción de menú “Run As”



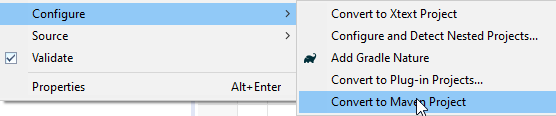
1. Una nueva instancia de “Eclipse IDE” se muestra.
2. En esta nueva instancia se debe crear un proyecto “General”



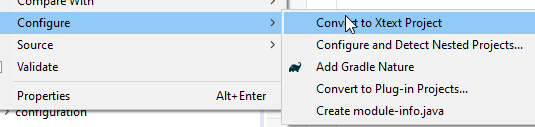
1. Colocar el nombre del proyecto por ejemplo Testing-aatDSL



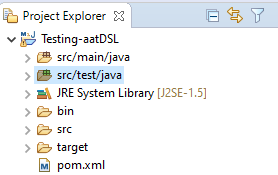
1. Se tiene que transformar el proyecto en un proyecto Maven. Para esto clic derecho sobre el nuevo proyecto y aplicar la opción “Convert to Maven Project” dentro de “Configure”



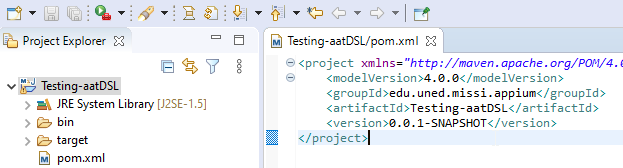
1. Se tiene que transformar el proyecto a un proyecto Xtext. Para esto clic derecho sobre el nuevo proyecto y aplicar la opción “Convert to Xtext Project” dentro de “Configure”



1. Crear la carpeta de fuentes src/main/java y src/test/java y añadirles como carpeta de fuentes en el proyecto



1. Reemplazar el contenido del nuevo archivo pom.xml con el provisto. Mantener el contenido de los elementos modelVersion, groupId, artifactId y version



1. El pom.xml final debe ser similar:

<project xmlns=**"http://maven.apache.org/POM/4.0.0"**

xmlns:xsi=**"http://www.w3.org/2001/XMLSchema-instance"**

xsi:schemaLocation=**"http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"**>

<modelVersion>**4.0.0**</modelVersion>

<groupId>**edu.uned.missi.appium**</groupId>

<artifactId>**Testing-aatDSL**</artifactId>

<version>**0.0.1-SNAPSHOT**</version>

<properties>

<project.build.sourceEncoding>**UTF-8**</project.build.sourceEncoding>

<maven.compiler.source>**1.7**</maven.compiler.source>

<maven.compiler.target>**1.7**</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>**io.appium**</groupId>

<artifactId>**java-client**</artifactId>

<version>**7.2.0**</version>

</dependency>

<dependency>

<groupId>**junit**</groupId>

<artifactId>**junit**</artifactId>

<version>**4.13**</version>

<scope>**test**</scope>

</dependency>

<dependency>

<groupId>**org.apache.logging.log4j**</groupId>

<artifactId>**log4j-api**</artifactId>

<version>**2.11.2**</version>

</dependency>

<dependency>

<groupId>**org.apache.logging.log4j**</groupId>

<artifactId>**log4j-core**</artifactId>

<version>**2.11.2**</version>

</dependency>

</dependencies>

<build>

<sourceDirectory>**src**</sourceDirectory>

<resources>

<resource>

<directory>**src**</directory>

<excludes>

<exclude>**\*\*/\*.java**</exclude>

</excludes>

</resource>

</resources>

<plugins>

<plugin>

<artifactId>**maven-compiler-plugin**</artifactId>

<version>**3.8.0**</version>

<configuration>

<source>**1.8**</source>

<target>**1.8**</target>

</configuration>

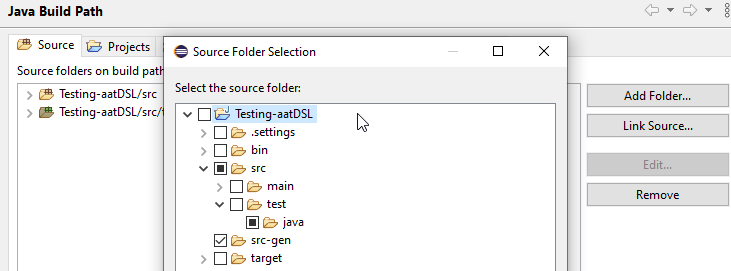
</plugin>

</plugins>

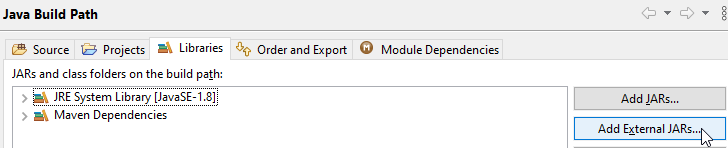
</build>

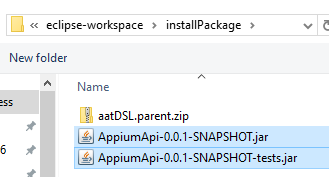
</project>

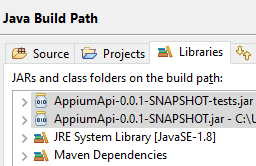
1. Crear folder src-gen y añadirlo a carpeta de fuentes del proyecto



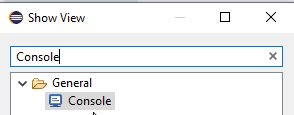
1. Finalmente agregar al “Build Path” los .jar de la librería Appium Api provistos.







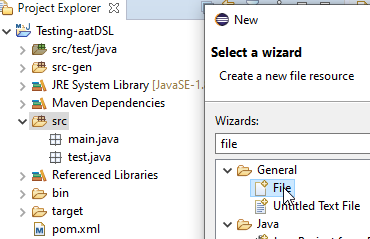
1. Como parte de la configuración requerida, mostrar la View Console

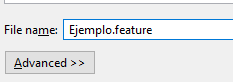


# Usar IDE para aatDSL

Verificar que la aplicación Android correspondiente este en ejecución, además que el servidor APPIUM se este ejecutando.

1. Dentro de la carpeta src, crear el nuevo archivo .feature en donde se colocan las instrucciones a ejecutar





1. Colocar dentro el script de pasos a ejecutar. Por ejemplo:

**Feature:**

"Crear una lista"

**Scenario:** 1

"El usuario presiona el botón Agregar, e ingresa el nombre de la lista"

**Given:**

// Verificar que el botón agregar Gasto se encuentra presente

Agregar.Lista **is** **visible**

**When:**

// Hacer clic sobre el botón agregar

**I** **press** **over** Agregar.Lista

// Ingresar el Nombre de la Lista

**I** **type** \"Víveres" **into** Nombre.Lista

// Presional botón Aceptar

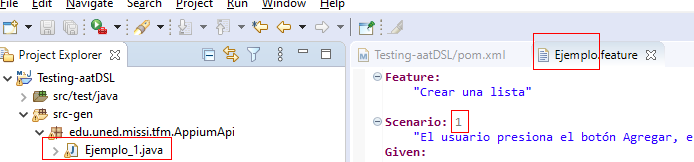
**I** **press** **over** Aceptar

**Then:**

// Se despliega el mensaje, Lista Vívires agregada

**Message** \"Lista Víveres agregada!" **is** **showed**

1. Al grabar el archivo (ctrl+S), se puede notar que se crea una nueva clase Java, cuyo nombre es la concatenación del nombre del archivo feature y el número de escenario



1. Crear en el director src-gen el archivo aatDSL.properties. Este debe describir los datos de conectividad hacia la aplicación a ejecutar a través de Appium. Por ejemplo:

deviceName=emulator-5554

platformName=android

appPackage=edu.uned.missi.listamercado

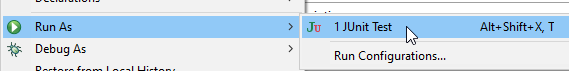
appActivity=.MainActivity

noReset=true

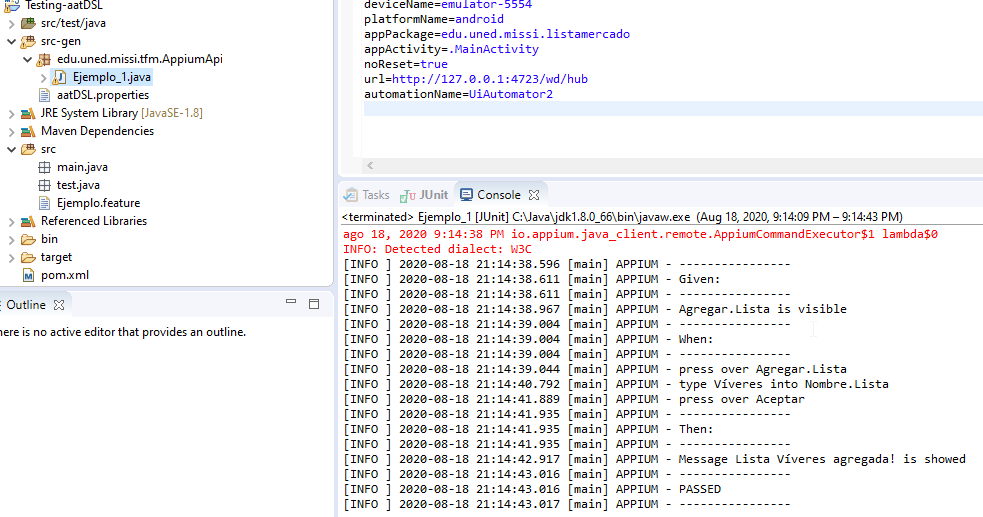
url=http://127.0.0.1:4723/wd/hub

automationName=UiAutomator2

1. Para ejecutar el escenario, clic derecho sobre la clase java, y seleccionar la opción “Junit Test” dentro de “Run As”



1. Al ejecutar en la vista Console se mostrará el resultado de la ejecución



FIN