

Vamos a empezar desde 0, explicando cada una de las herramientas utilizadas, realizando su instalación. Así como como unas pautas para su uso.

Introducción

¿Qué es Selenium? Selenium es una Herramienta de automatización de pruebas de código abierto. Existen varias herramientas en el mercado, pero Selenium ha conseguido ser de las más usadas debido a que es gratuita y de código abierto.

¡OJO! Selenium no está diseñado para automatizar las pruebas de escritorio o mainframe, esto se puede ver como una ventaja o como una desventaja.

Una de las mayores ventajas de Selenium es que trabaja sobre varios sistemas operativos. Es la única herramienta que trabaja sobre *Windows, OS X, Solaris y Linux*. Actualmente también se da soporte a dispositivos móviles, existen APIs adaptadas que son extensiones de Selenium que dan soporte a ambas plataformas: **IOs y Android**

Selenium se puede escribir en cualquiera de los siguientes lenguajes de programación:

- **Java → El curso está diseñado bajo JAVA**
- C#
- Ruby
- Python → Es otro de los lenguajes más utilizados con Selenium
- PHP
- Perl

Debes tener en cuenta que el lenguaje utilizado para desarrollar tu software CORE (el build) es independiente del lenguaje utilizado para automatizar las pruebas con Selenium.

Por último, Selenium soporta múltiples navegadores, proporcionando la misma estabilidad en ellos:

- **Internet Explorer (Edge) → Bastante problemático su WebDriver bajo OS X.**
- Firefox
- Chrome
- Safari

1. Instalación de la última versión disponible de JAVA

Previamente en la línea de comandos escribir `java -version` en caso de que no exista o no sea una versión 8 o superior continuar con:

Escribimos en Google «*java jdk download*». La última versión actualmente es la 12 aunque la versión 8 es la más estable. Sólo descargad la versión JDK del S.O. correspondiente.

Oracle Technology Network / Java / Java SE / Downloads:

<https://www.oracle.com/technetwork/java/javase/downloads/index.html>

1. <https://www.oracle.com/technetwork/java/javase/downloads/jdk12-downloads-5295953.html>

Se podrá descargar tras aceptar los acuerdos de licencia.

2. Configurar JAVA Path en las variables del sistema

Windows 10

Ir a Program Files > Java > jdk.1.8.0_XXX > bin → Copiar la ruta

Posteriormente vamos a Panel de Control > Sistema y Seguridad > Sistema > Pestaña “Avanzado” > Variables de Entorno > Variables de Sistema > Selecciona “Path” > New > Pega la Ruta anterior y Guarda.

OS X

Agregar la variable de entorno al PATH en OS X en el documento bash_profile agregar la línea:

export JAVA_HOME=\$(/usr/libexec/java_home)

3. Instalación de la última versión disponible de IntelliJ

A continuación, procedemos con la instalación del entorno de desarrollo (IDE) de IntelliJ:

- 1.

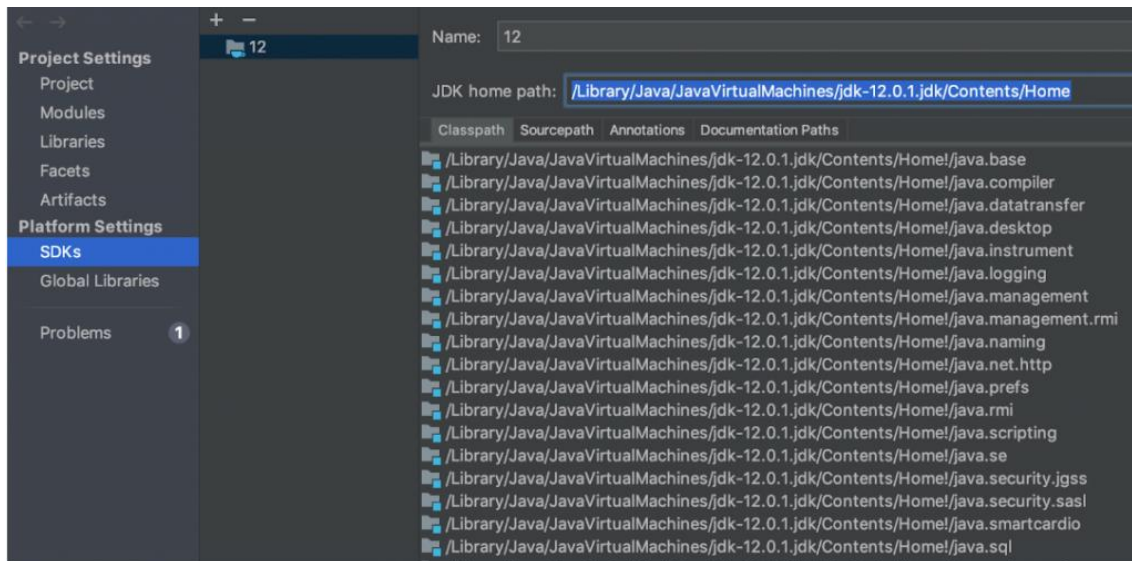
1. <https://www.jetbrains.com/toolbox/>

Seleccionamos IntelliJ IDEA → Descargamos e Instalamos



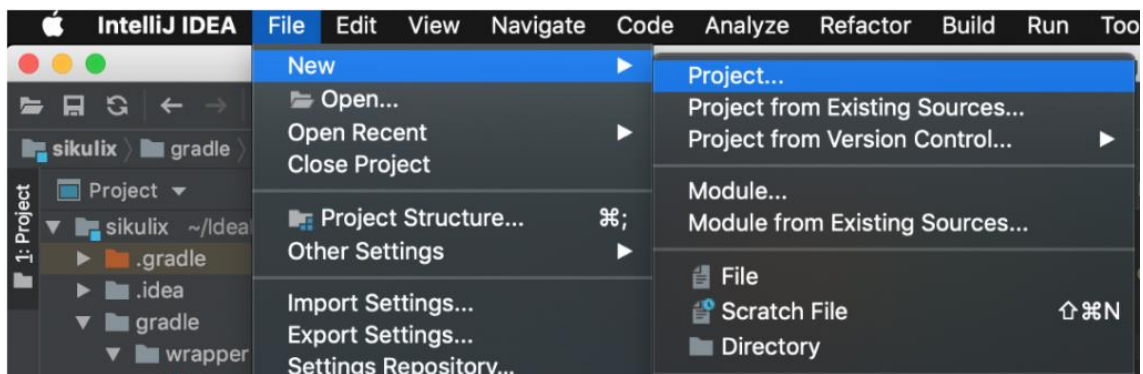
Posteriormente comprobamos que conoce el PATH de nuestra versión de Java instalada:

File > Project Structure > Platform Settings > JDK

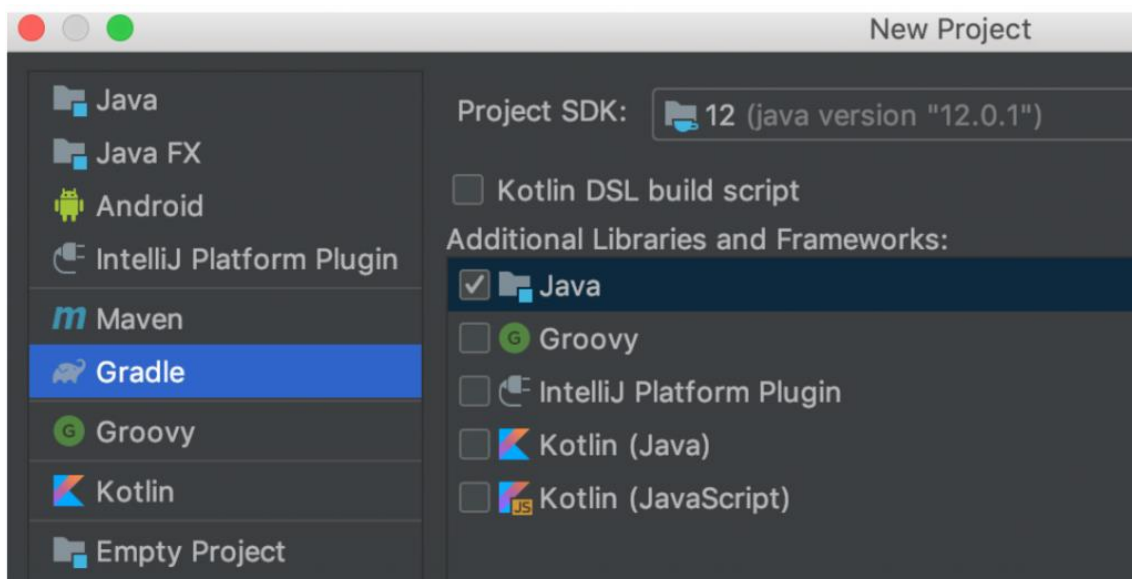


4. Crear un nuevo proyecto en IntelliJ

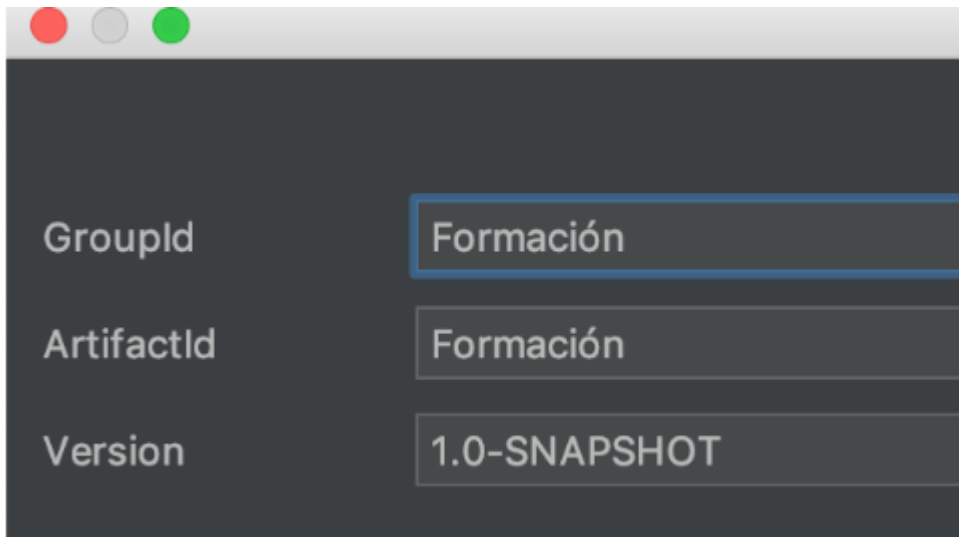
Para ello vamos a *File > New > Project*



Marcamos y Java como librería adicional:

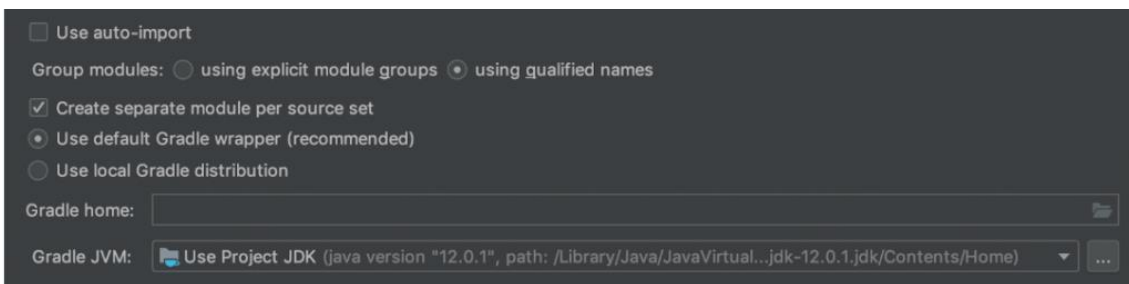


En la siguiente ventana le indicamos el Group ID y Artefact ID que queremos:



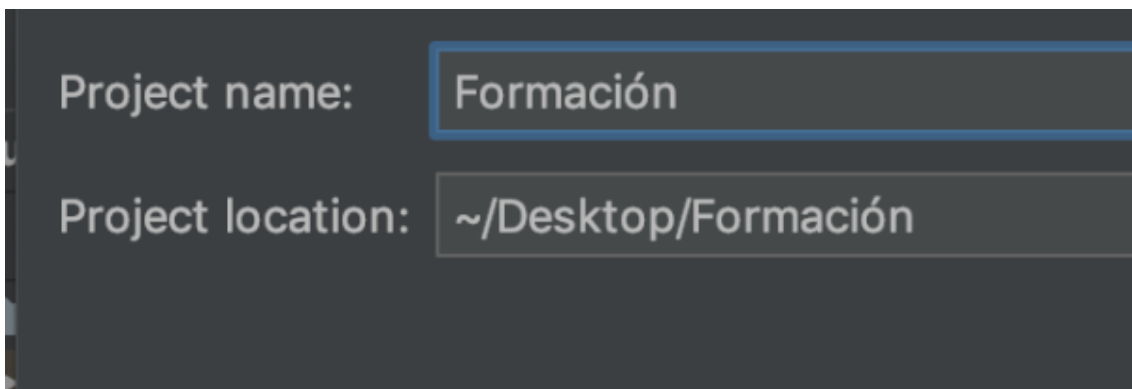
A screenshot of a dark-themed window with three input fields. The first field is labeled 'GroupId' and contains the text 'Formación'. The second field is labeled 'ArtifactId' and also contains 'Formación'. The third field is labeled 'Version' and contains '1.0-SNAPSHOT'. Each field has a blue border and a light blue highlight.

En el siguiente paso marcamos *Create separate module per source set* e **importante** seleccionar *Use default Gradle wrapper (recommended)*:



A screenshot of a dark-themed window showing configuration options. It includes a checkbox for 'Use auto-import', a radio button for 'Group modules' (selected 'using qualified names'), a checked checkbox for 'Create separate module per source set', a selected radio button for 'Use default Gradle wrapper (recommended)', and an unselected radio button for 'Use local Gradle distribution'. Below these are fields for 'Gradle home' and 'Gradle JVM' (set to 'Use Project JDK').

Para finalizar, indicamos el nombre del proyecto y la ubicación:



A screenshot of a dark-themed window showing two input fields. The first field is labeled 'Project name:' and contains the text 'Formación'. The second field is labeled 'Project location:' and contains the text '~/Desktop/Formación'. Both fields have a blue border and a light blue highlight.

Y después de aceptar se nos indicará si queremos que se cree en la ventana actual o en una nueva. Esto ya según el gusto del consumidor.

5. Selenium JARs download

¿Qué es Gradle?

Gradle, es una herramienta que permite la automatización de compilación de código abierto, la cual se encuentra centrada en la flexibilidad y el rendimiento. Los scripts de compilación de Gradle se escriben utilizando Groovy o Kotlin DSL (**Domain Specific Language**).

Gradle tiene una gran flexibilidad y nos deja hacer usos otros lenguajes y no solo de Java, también cuenta con un sistema de gestión de dependencias muy estable. Gradle es altamente personalizable y rápido ya que completa las tareas de forma rápida y precisa reutilizando las salidas de las ejecuciones anteriores, sólo procesar las entradas que presentan cambios en paralelo.

Además, es el sistema de compilación oficial para Android y cuenta con soporte para diversas tecnologías y lenguajes.

6. Configurar Selenium JARs (y Sikuli) en el Project Build Path

En build.gradle > dependencias agregamos:

```
implementation "com.sikulix:sikulixapi:1.1.4-SNAPSHOT"
```

```
implementation "org.seleniumhq.selenium:selenium-java:3.141.59"
```

NOTA para que podamos utilizar la API de Sikuli no basta con poner la Dependencia, además debemos incluir la ruta del repositorio Maven que podemos encontrar buscando en Google “Sikuli repository Url”. Adjunto la ruta donde se encuentra en la documentación de Sikuli:

<https://sikulix-2014.readthedocs.io/en/latest/sources/faq/030-java-dev.rst.txt>

```
**use this repository setting:**
| <repository>
|   <!-- OSSRH: com.sikulix -->
|   <id>com.sikulix</id>
|   <name>com.sikulix</name>
|   <url>https://oss.sonatype.org/content/groups/public</url>
|   <layout>default</layout>
|   <snapshots>
|     <enabled>true</enabled>
|     <updatePolicy>always</updatePolicy>
|   </snapshots>
| </repository>
```

De este modo tendremos:

```
repositories {
    mavenCentral()
    maven {
        url 'https://repository.mulesoft.org/nexus/content/repositories/public/'
    }
    maven {
        url 'https://oss.sonatype.org/content/repositories/snapshots/'
    }
}

dependencies {
```

```
implementation "org.seleniumhq.selenium:selenium-java:3.141.59"
implementation "com.sikulix:sikulixapi:1.1.4-SNAPSHOT"
testCompile group: 'junit', name: 'junit', version: '4.12'
}
```

Se deberán descargar los JARs files de las webs correspondientes y agregarlos como Librerías externas a nuestro proyecto.

<https://www.seleniumhq.org/download/>

7. Elegir el Navegador donde ejecutar las pruebas y su WebDriver

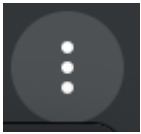
Para los ejercicios que se vana a realizar vamos a utilizar los navegadores Chrome (por su estabilidad) y Firefox.

<https://www.seleniumhq.org/download/>

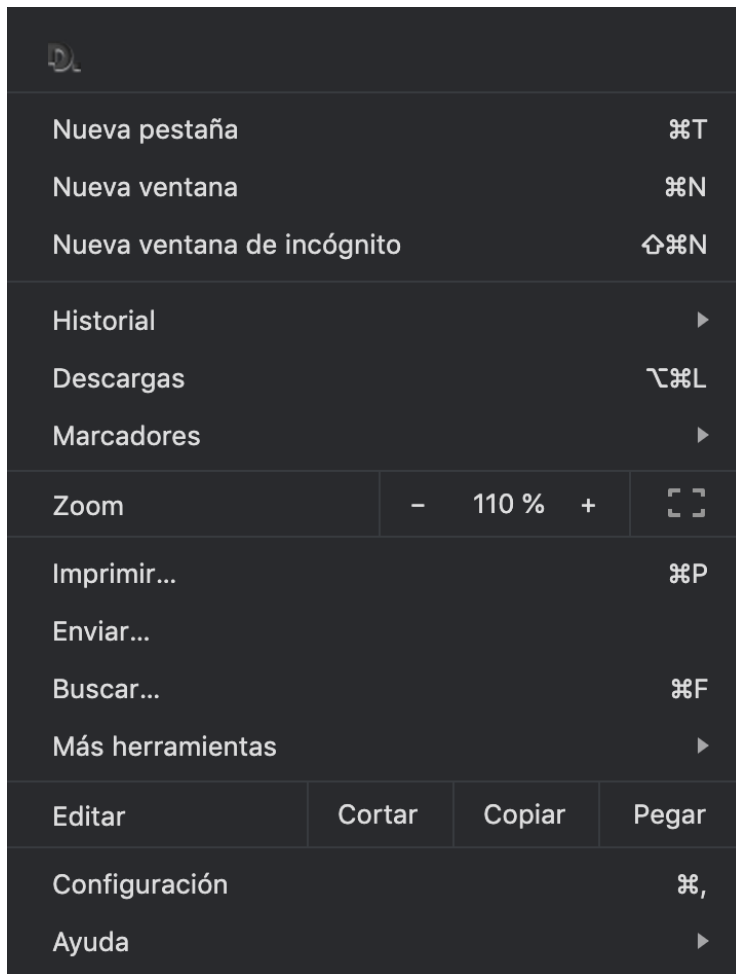
¡OJO! Antes de descargar el WebDriver es necesario conocer la versión del navegador que tenemos instalado.

WebDriver de Chrome:

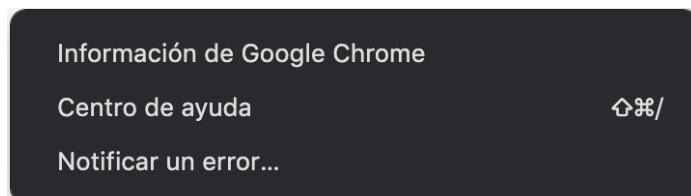
En el navegador Chrome pulsamos en:



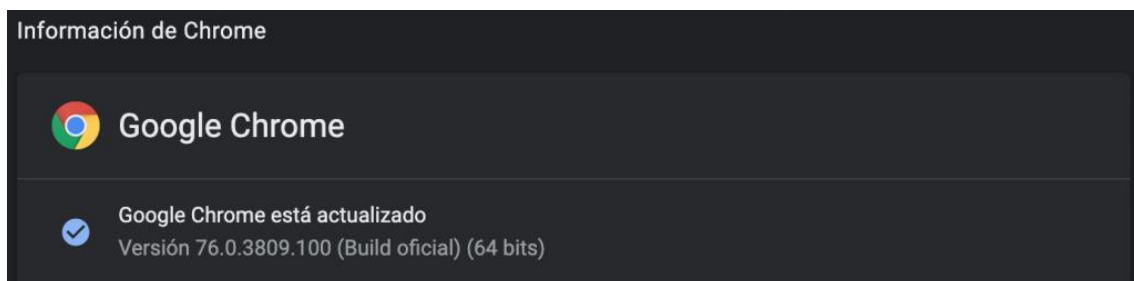
A continuación,



Pulsamos en «Ayuda»:



Y ahora en «Información de Google Chrome»:



Una vez que conozcamos la versión de nuestro navegador Chrome, descargamos el WebDriver correspondiente de:

<https://sites.google.com/a/chromium.org/chromedriver/downloads>

En nuestro caso:

Downloads

Current Releases

- If you are using Chrome version 77, please download [ChromeDriver 77.0.3865.10](#)
- If you are using Chrome version 76, please download [ChromeDriver 76.0.3809.68](#)
- If you are using Chrome version 75, please download [ChromeDriver 75.0.3770.140](#)
- For older version of Chrome, please see below for the version of ChromeDriver that supports it.

WebDriver de Firefox

Para Firefox simplemente mantén el navegador actualizado a la última versión y descarga el WebDriver de la siguiente URL:

<https://github.com/mozilla/geckodriver/releases>

Descargamos el **WebDriver** de Firefox y el de Chrome y los almacenamos en una carpeta por ejemplo en el escritorio. Posteriormente copiamos la ruta y volvemos al proyecto.

Para configurar la **Property** en JAVA, necesitamos dos parámetros. El 1ª es la KEY que viene a identificar el WebDriver del navegador que vamos a usar pudiendo ser:

- webdriver.chrome.driver → Para Chrome
- webdriver.gecko.driver → Para FireFox
- webdriver.ie.driver → Para I.E.
- webdriver.edge.driver → Para Edge

El 2º parámetro es el valor que debe ser la ubicación del WebDriver. Teniendo en cuenta de que las rutas se expresan diferentes si se trata de OS X o de Windows:

- En Mac es por ejemplo */Users/Autentia/Desktop/Sikuli/chromedriver*
- En Windows es necesario indicar la unidad y el tipo de archivo .exe

C:/Users/Autentia/Desktop/Sikuli/chromedriver.exe

Por tanto, usamos el comando en la clase Java que crees y que será tu Script para automatizar las pruebas:

```
System.setProperty("webdriver.chrome.driver","/Users/Autentia/Desktop/Sikuli/chromedriver");
```

Y listo el WebDriver del navegador Chrome en este caso.