



Herramientas para trabajar con Ionic

Yhoan Andrés Galeano Urrea

Desarrollador por pasión

Medellín - 2018



Herramientas necesarias para trabajar con Node.js, Angular y Ionic

En este documento abordaremos las diferentes herramientas necesarias para trabajar durante el proceso de formación. Se dará una breve introducción a cada una de las tecnologías y cómo se instalan, comencemos.

Qué es y cómo se instala Node.js

Node.js (<https://nodejs.org/>) te permite compilar aplicaciones del lado del servidor y aplicaciones de red en V8 Javascript. Espere... ¿qué? ¿JavaScript en el servidor? Sí, leyó correctamente. JavaScript del lado del servidor puede ser un concepto nuevo para cualquiera que haya trabajado exclusivamente con JavaScript del lado del cliente, pero la idea en sí no es tan inverosímil — ¿por qué no utilizar el mismo lenguaje de programación que usted usa en el cliente del lado del servidor?

¿Qué es el V8? El motor V8 JavaScript es el motor JavaScript subyacente que Google usa con su navegador Chrome. Pocas personas piensan en lo que en realidad sucede con JavaScript en el cliente. Bien, un motor JavaScript en realidad interpreta el código y lo ejecuta. Con el V8, Google creó un intérprete ultra-rápido escrito en C++, con otro aspecto único: usted puede descargar el motor e incorporarlo a cualquier aplicación que desee. No está restringido a ejecutarse en un navegador. Así, Node en realidad usa el motor V8 JavaScript escrito por Google y le da otro propósito para usarlo en el servidor.

Aunque para nuestros fines, lo usaremos para correr herramientas de líneas de comando escritas en Javascript, como Ionic CLI y que nos servirán para crear algunas tareas y ver nuestra aplicación en acción.

Node.js incluye **npm** (Node Package Manager) una herramienta que facilita instalar y actualizar paquetes. De esta manera no tendremos que ir a la página de las librerías, descargar, descomprimir e instalar en nuestros proyectos, pues en realidad esta tarea es un poco tediosa y nos hace ser menos productivos.

Para instalar Node.js en Windows, simplemente descarga el instalador MSI desde <https://nodejs.org/es/download/>, después corre el instalador y sigue las instrucciones.

Node.js® es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema más grande de librerías de código abierto en el mundo.

Important March 2018 security upgrades now available

Descargar para Windows (x64)

8.11.1 LTS
Recomendado para la mayoría

9.10.1 Actual
Últimas características

Otras Descargas | Cambios | Documentación del API Otras Descargas | Cambios | Documentación del API

Ó revise la Agenda de LTS.

<https://nodejs.org/en/blog/vulnerability/march-2018-security-releases/>

Welcome to the Node.js Setup Wizard

The Setup Wizard will install Node.js on your computer.

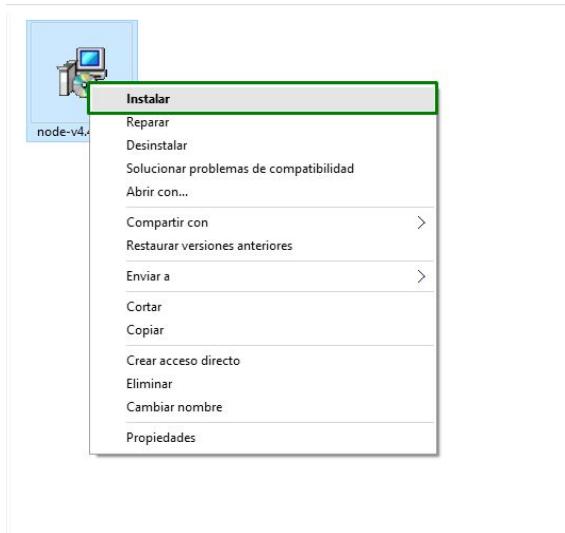
Back Next Cancel

Recomendado para la mayoría Últimas características

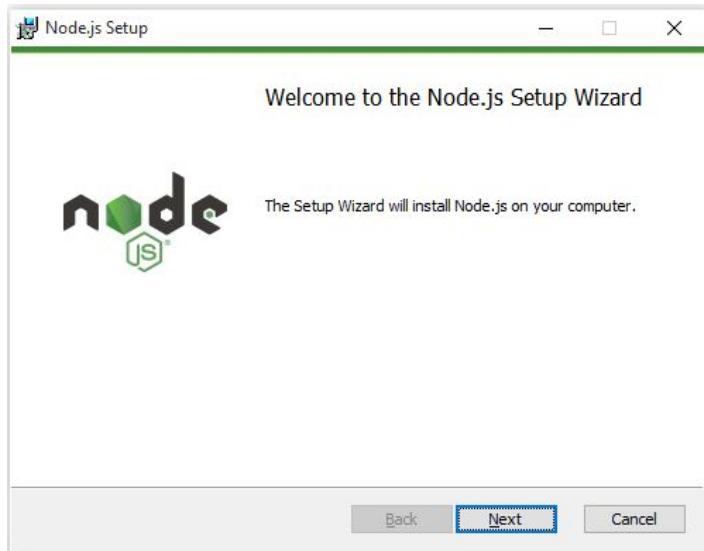
Otras Descargas | Cambios | Documentación del API Otras Descargas | Cambios | Documentación del API

Ó revise la Agenda de LTS.

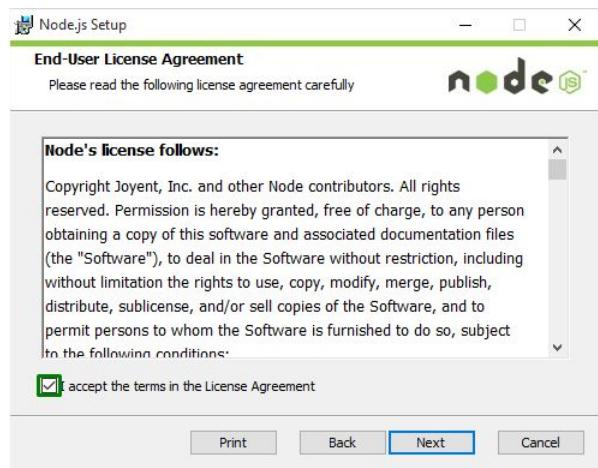
Una vez descargado el archivo haz clic derecho y escoge la opción Instalar



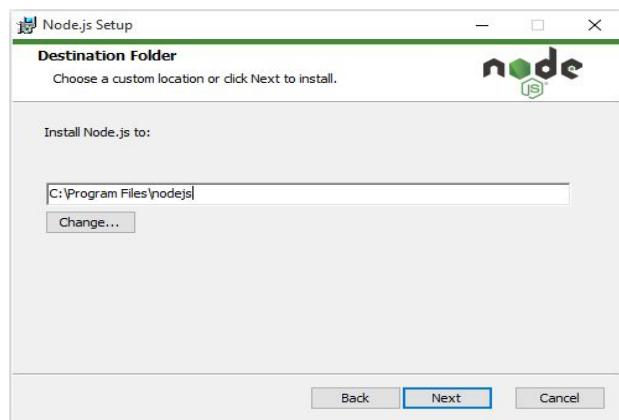
Presiona en el botón next



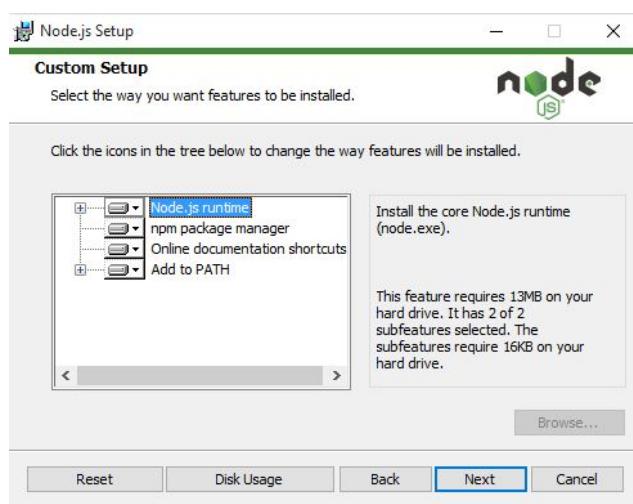
Ahora selecciona la casilla donde dice (I accept the terms in the License Agreement) como en la imagen y luego clic en el botón Next.



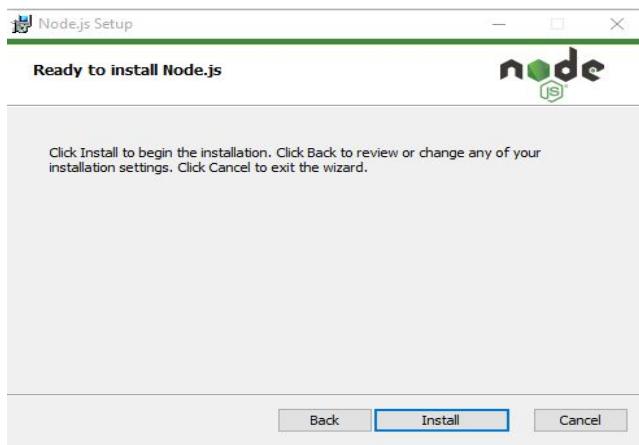
Aquí puedes cambiar la ruta de instalación si lo deseas y si no lo dejas igual. Para continuar haz clic en el botón Next



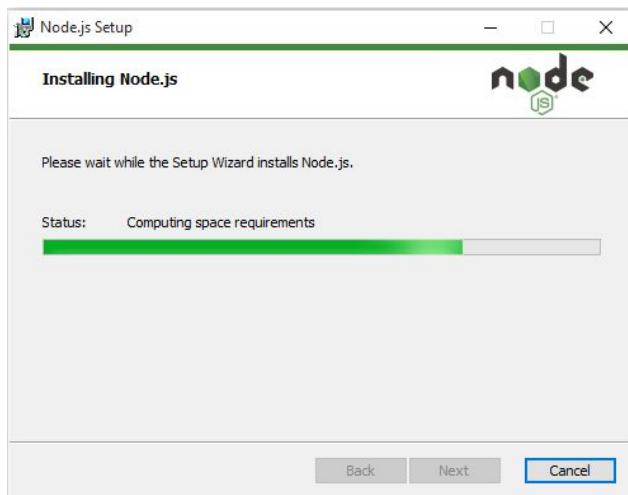
Aquí tienes la opción de personalizar la instalación, una vez culminado haz clic en el botón Next.



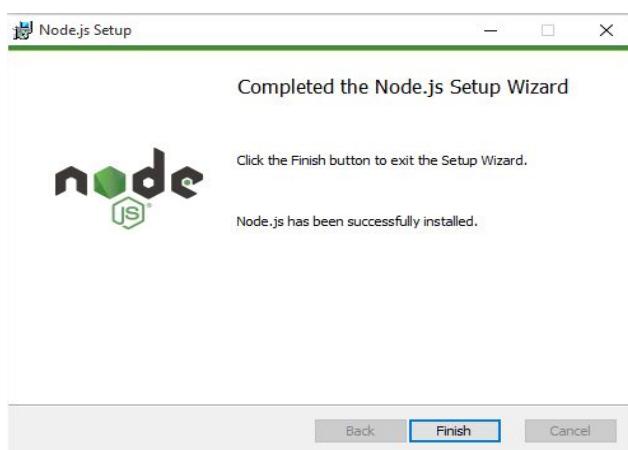
Ahora hacemos clic en el botón Install



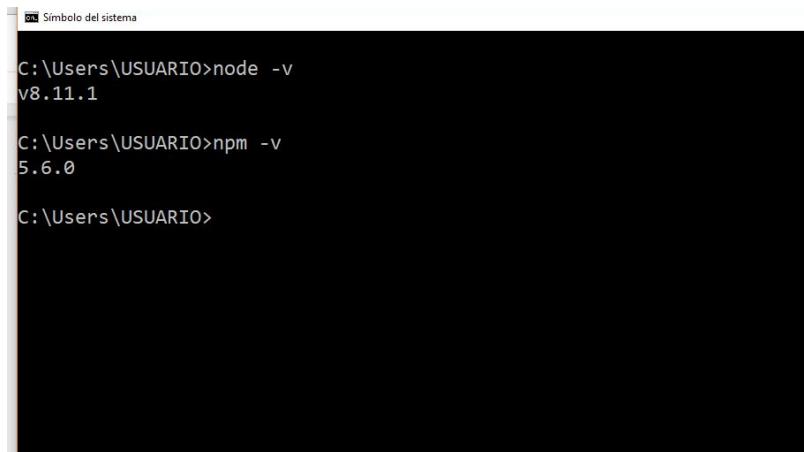
Node.js está en proceso de instalación por lo general no lleva mucho tiempo.



En esta ventana muestra un mensaje Node.js ha sido instalado exitosamente.



Para verificar la correcta instalación de Node.js, de npm, vamos a pulsar los siguientes comandos en tu consola:



```
C:\Users\USUARIO>node -v
v8.11.1
C:\Users\USUARIO>npm -v
5.6.0
C:\Users\USUARIO>
```

Que es GIT y como se instala

Git es un sistema de control de versiones usado en el desarrollo de software para guardar los cambios realizados en un proyecto, mantener un historial y navegar a través del mismo en cualquier momento.

Esta herramienta es necesaria para el curso, puesto que cuando se descarga un proyecto en ionic, sus plantillas bases están ubicadas en repositorios de Github y requieren de que tengamos configurado nuestro usuario y correo electrónico. Veamos el paso a paso para la correcta instalación.

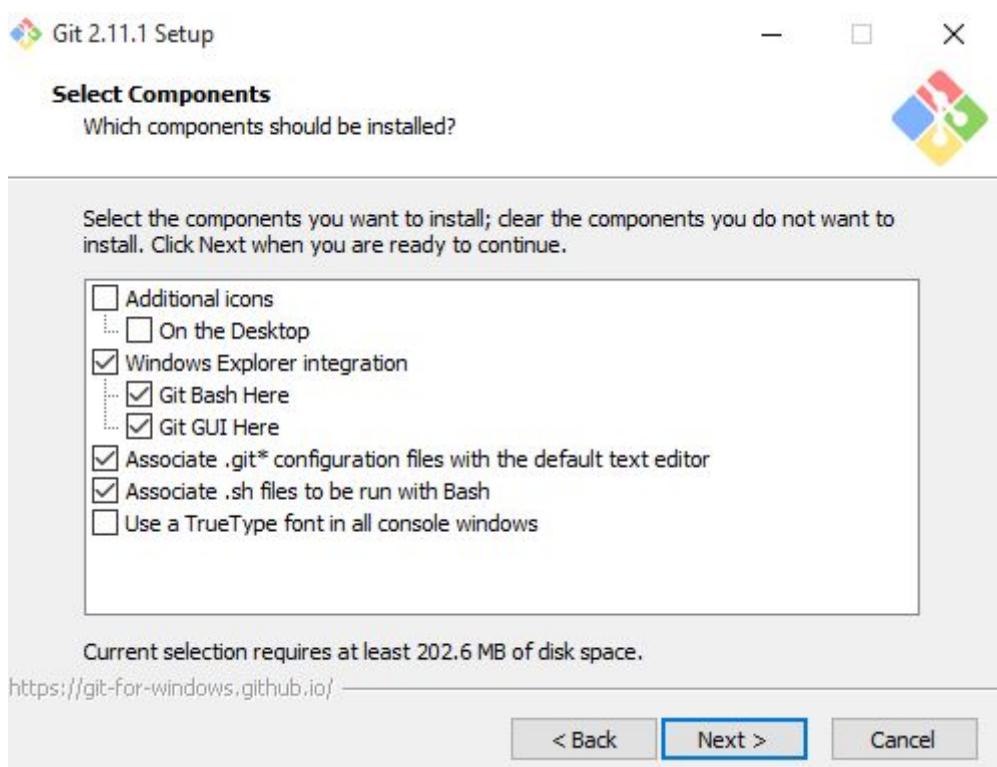
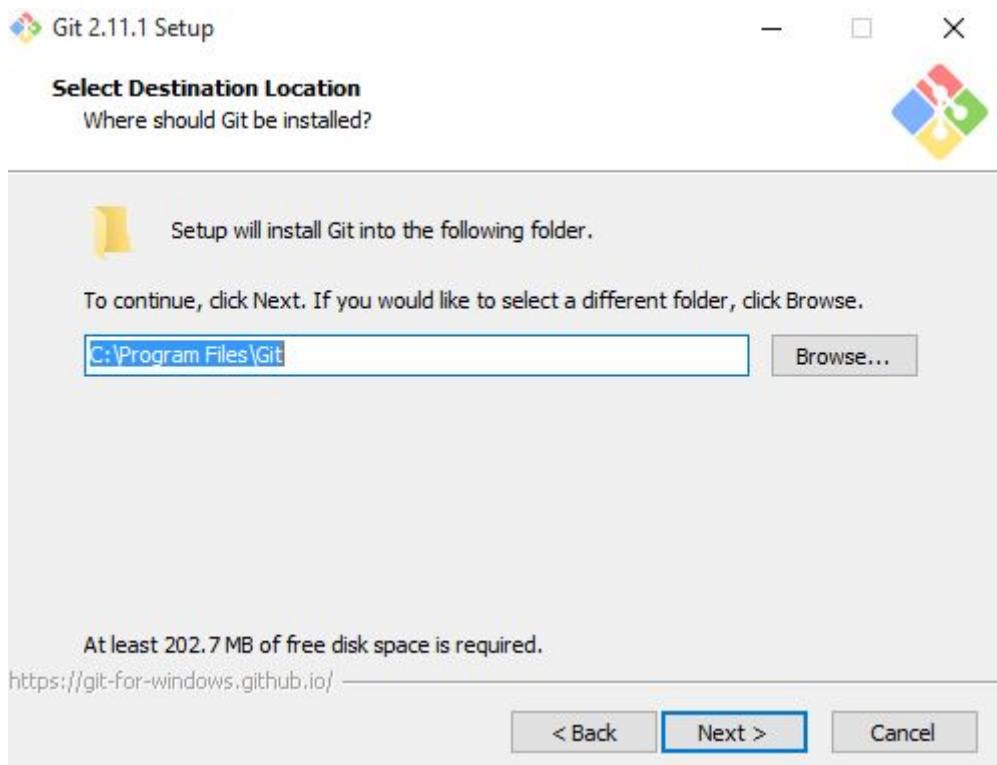
Ingresa a la página oficial (<https://git-scm.com/downloads>) y descarga la versión actualizada:

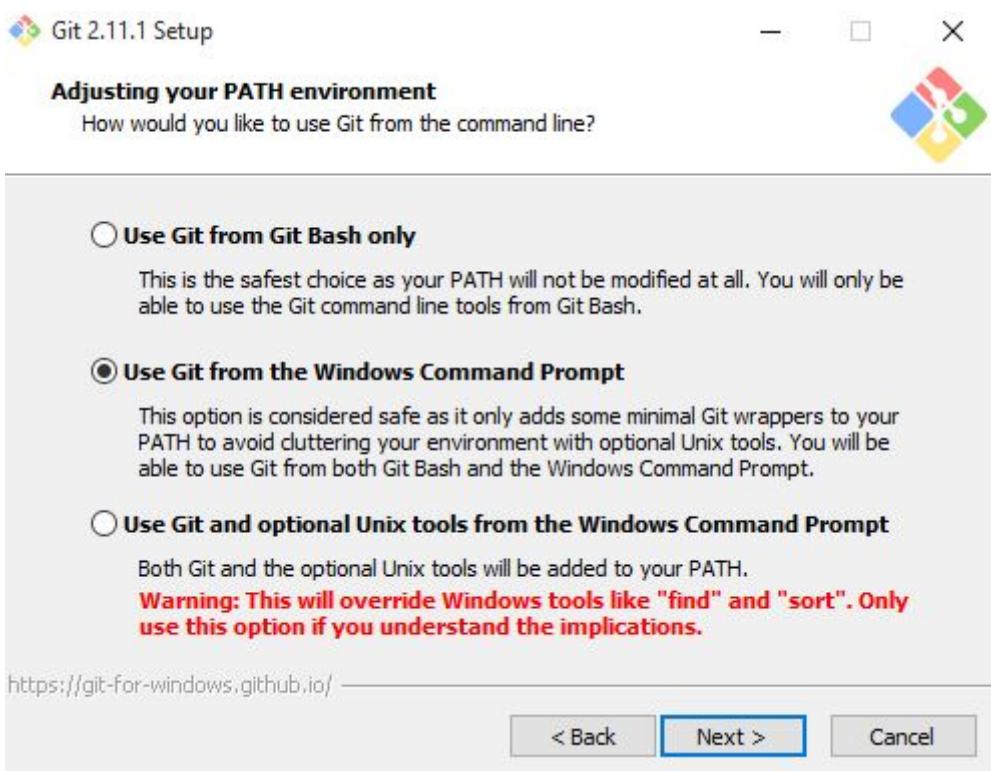
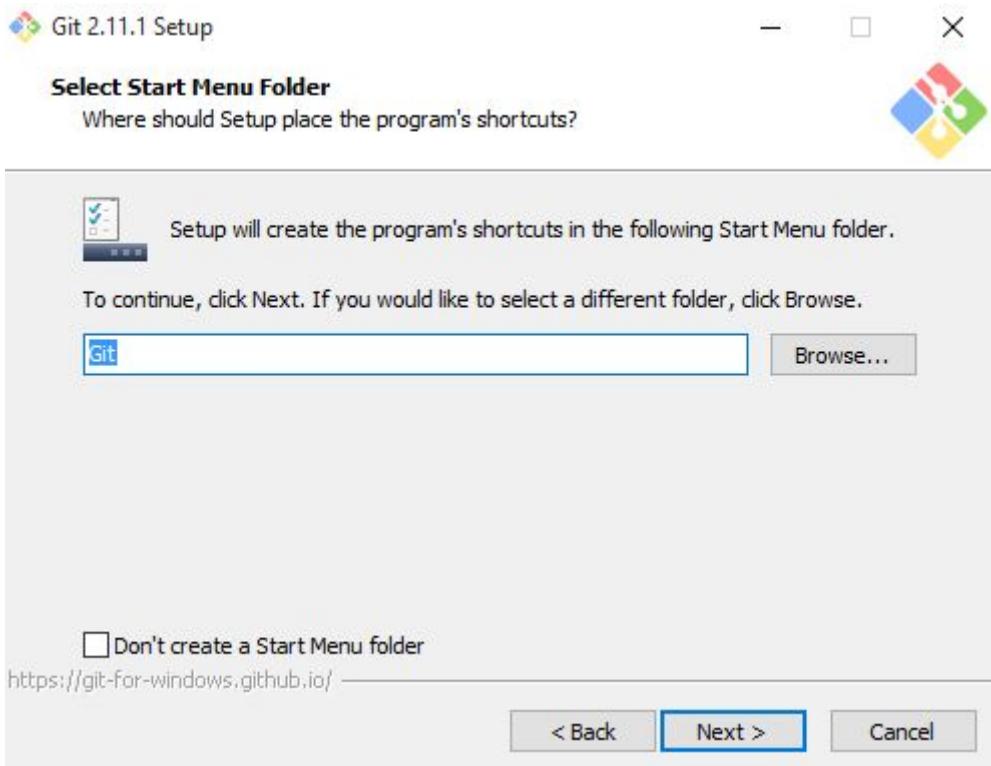
The screenshot shows the official Git website. At the top left is the Git logo with the word "git". To its right is the tagline "local-branching-on-the-cheap". A search bar is located at the top right. Below the header, there's a section about Git's features: "Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency." Another section highlights its performance: "Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows." A "Try Git" button with a GitHub icon is present. To the right, there's a diagram illustrating branching and merging in Git. Below these sections are links for "About", "Documentation", "Downloads", and "Community". A "Pro Git" book cover is shown. On the right side, there's a large image of a computer monitor displaying the "Latest source Release 2.16.3" with a "Download 2.16.3 for Mac" button. Below the monitor are links for "Mac GUIs", "Tarballs", "Windows Build", and "Source Code".

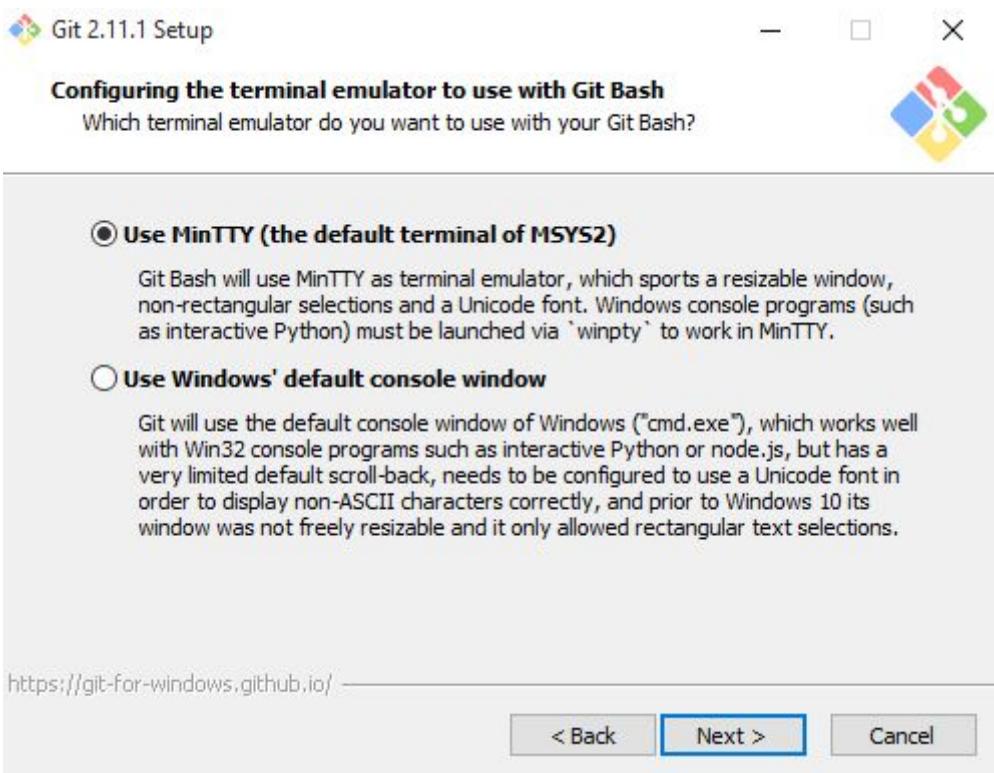
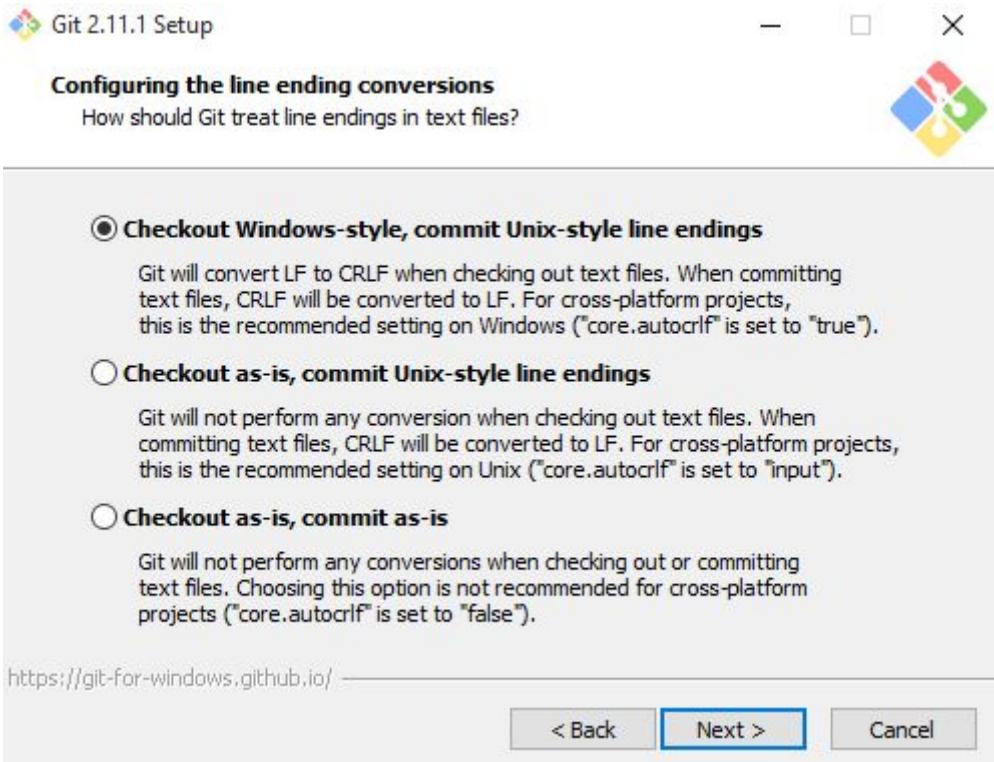
Una vez descargada, dale click al archivo con el formato **Git-version.exe**. Al inicio preguntará por permisos para ejecutar el instalador, a lo cual debemos responder que sí.

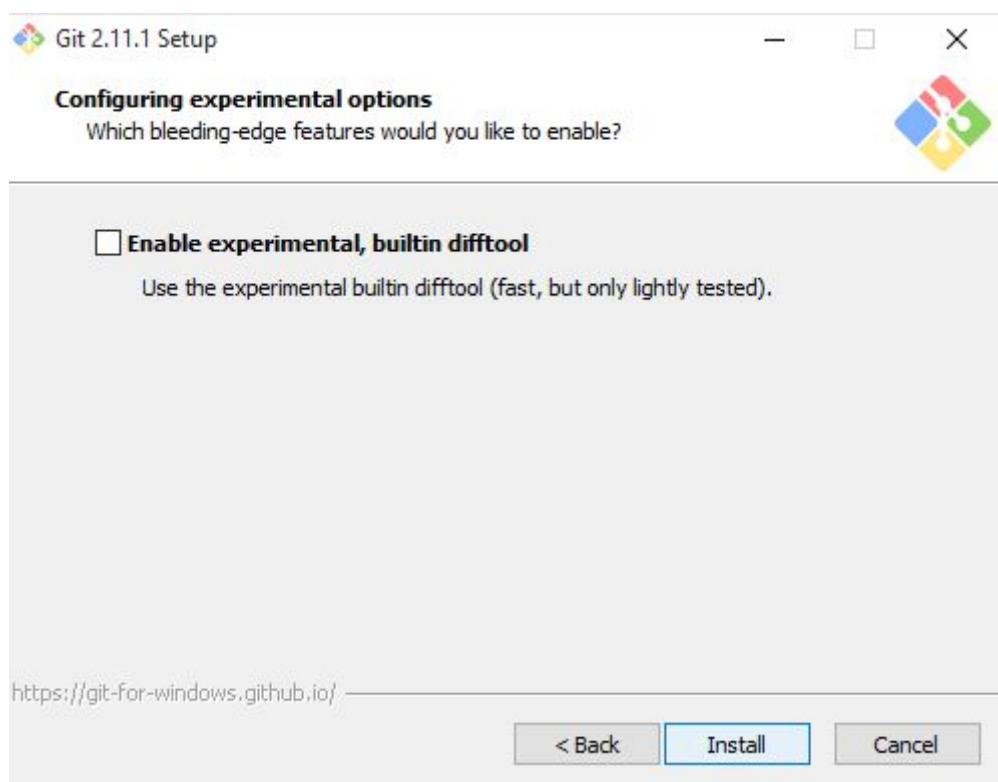
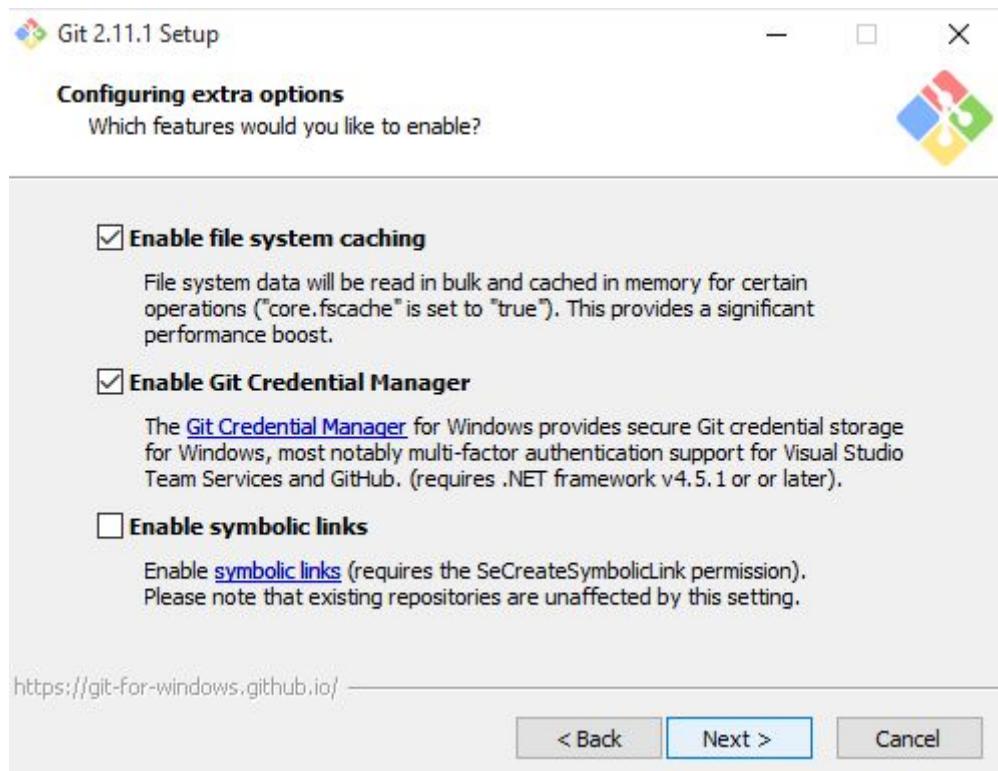


Como muchos de los instaladores en Windows, debemos de aceptar las opciones por defecto y darle Next (siguiente) a todo hasta que nos salga el botón de instalar.

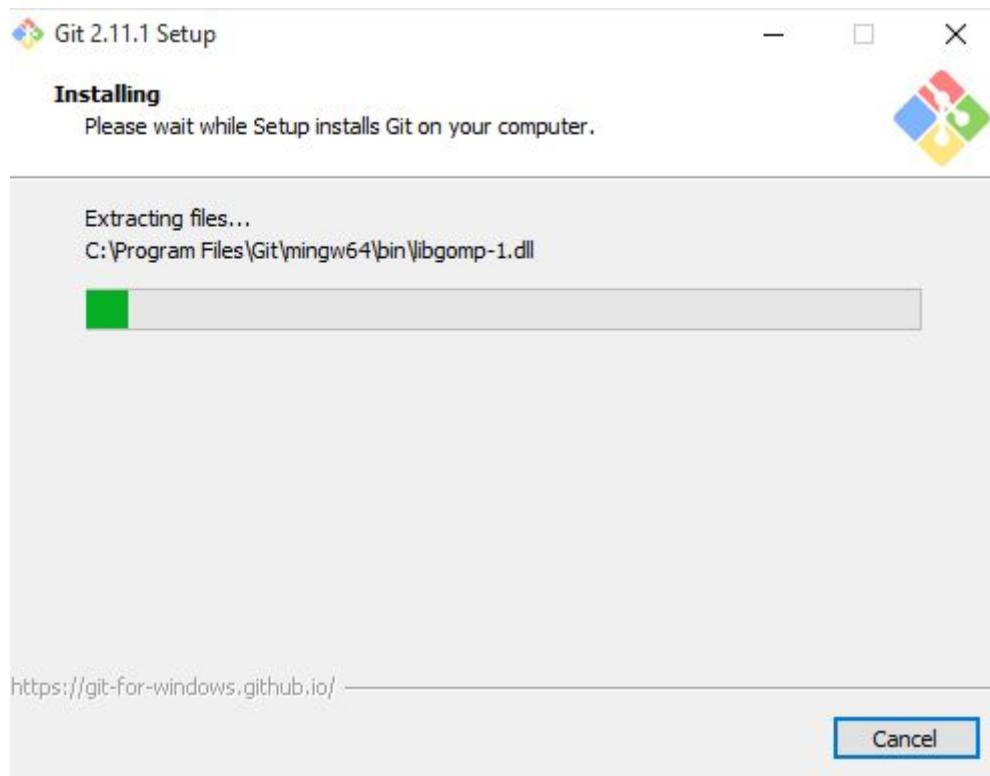




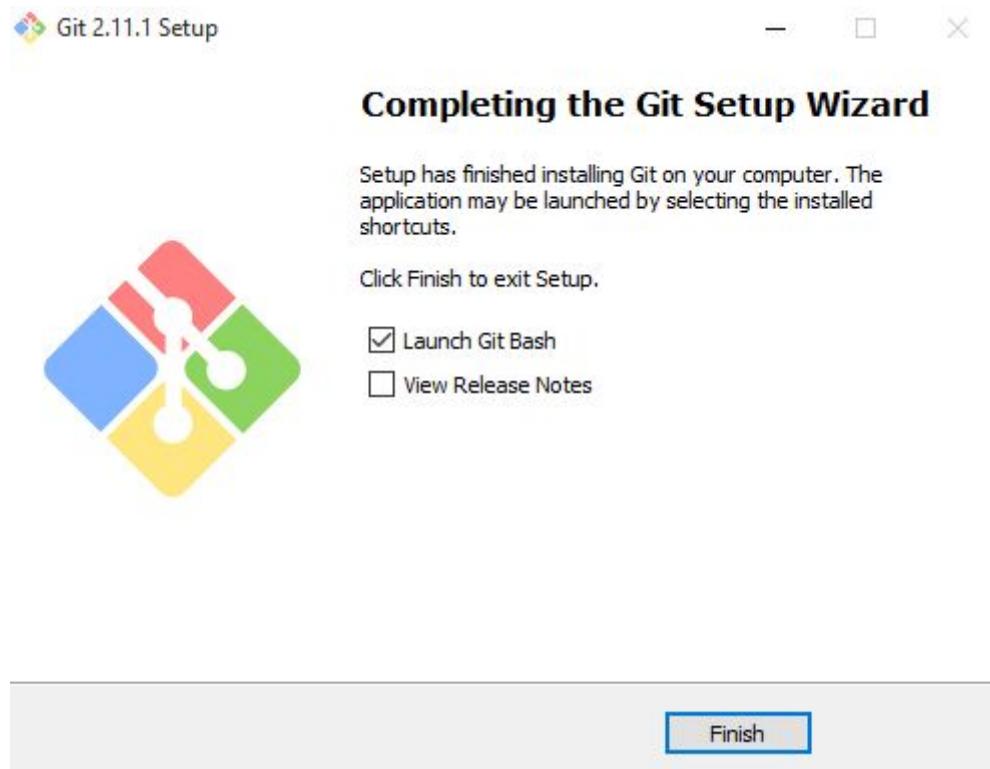




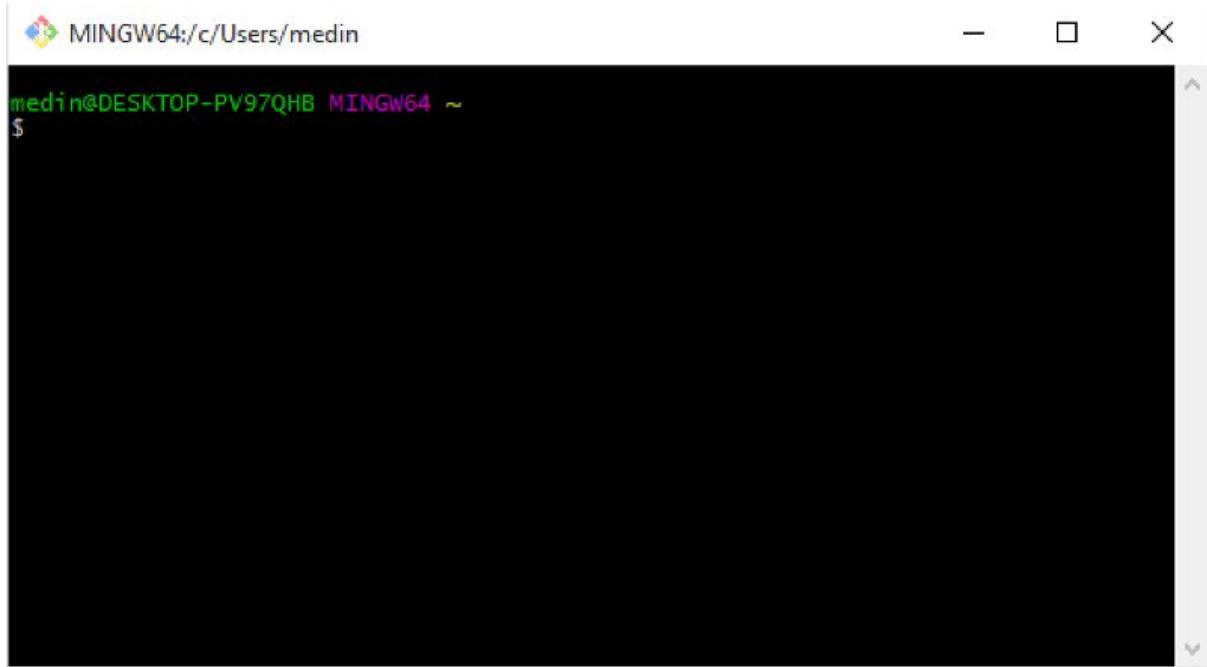
Una vez terminada la configuración del instalador de Git, comenzará a instalar todos los archivos necesarios en la carpeta indicada al inicio.



Al finalizar la instalación, nos dará la opción de abrir Git Bash para finalizar.



Y ésta será la terminal que usaremos para ingresar comandos Unix y de Git sobre todo:

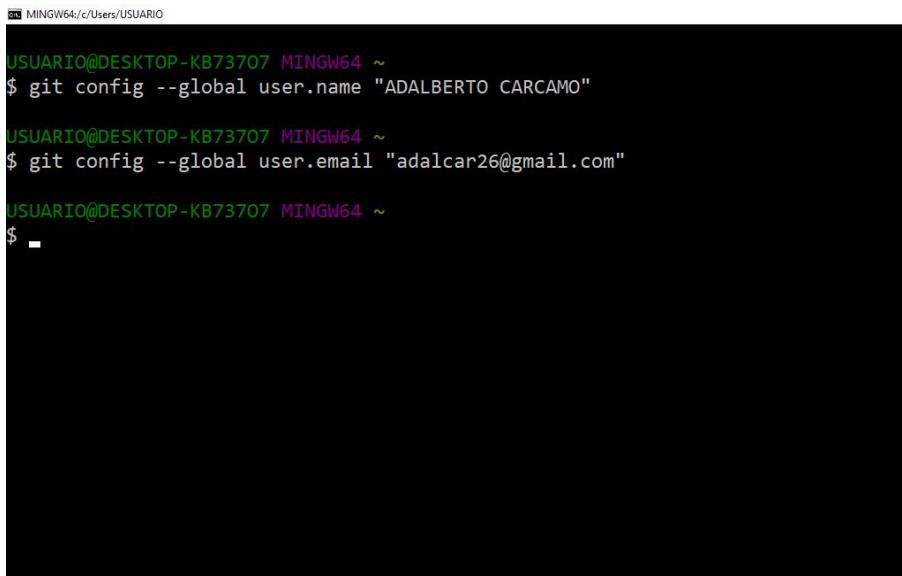


MINGW64:/c/Users/medin

medin@DESKTOP-PV97QHB MINGW64 ~

\$

Para finalizar, debes configurar el **usuario** y el **correo electrónico**, de esta manera cuando estemos trabajando con alguna de las plantillas bases de IONIC no sacará problemas.



```
MINGW64:/c/Users/USUARIO
USUARIO@DESKTOP-KB73707 MINGW64 ~
$ git config --global user.name "ADALBERTO CARCAMO"
USUARIO@DESKTOP-KB73707 MINGW64 ~
$ git config --global user.email "adalcar26@gmail.com"
USUARIO@DESKTOP-KB73707 MINGW64 ~
$ -
```

GIT es una herramienta supremamente poderosa y muy trabajada en el día a día en las empresas y proyectos de Software. Te invito a darle una mirada a este enlace que les dejo, donde podrán encontrar más información acerca de GIT, su funcionamiento, comandos y herramientas. http://librosweb.es/libro/pro_git/

Android Studio

Para trabajar con Ionic es necesario tener instalado Android Studio, pues a la hora de compilar la aplicación para que se ejecute en el dispositivo móvil, necesitaremos de los SDK, del Gradle y otras herramientas que ya vienen configuradas para esta plataforma. Lo mismo sucede cuando queremos subir el apk a la Play Store y firmarla.

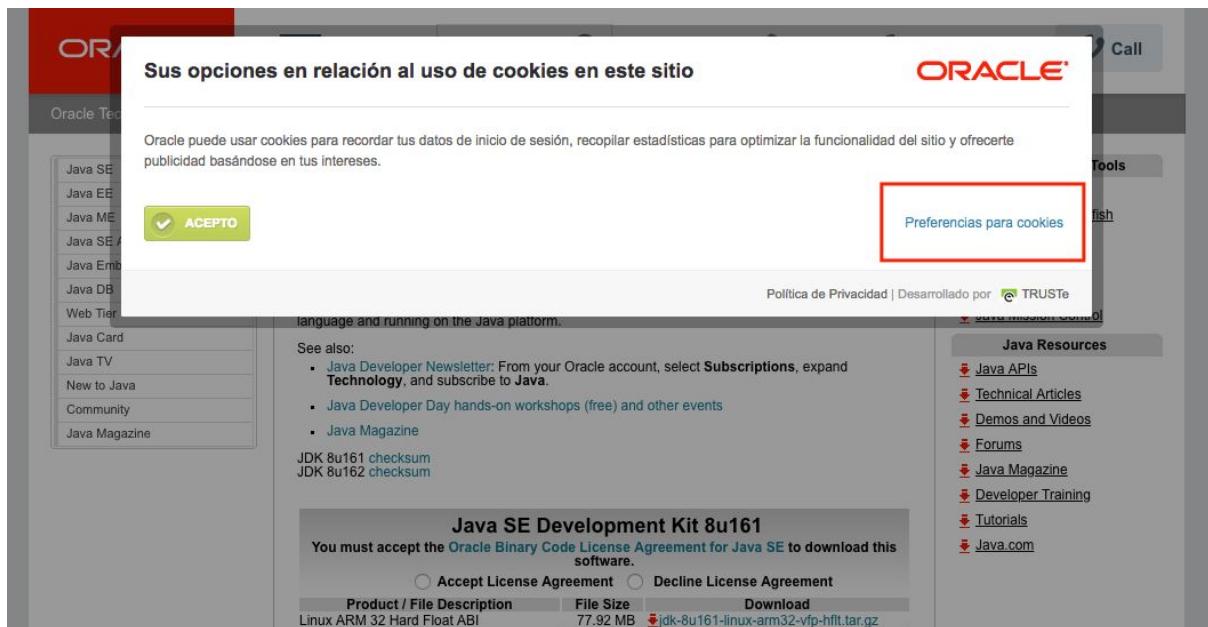
Para su correcta instalación sigamos los siguientes pasos:

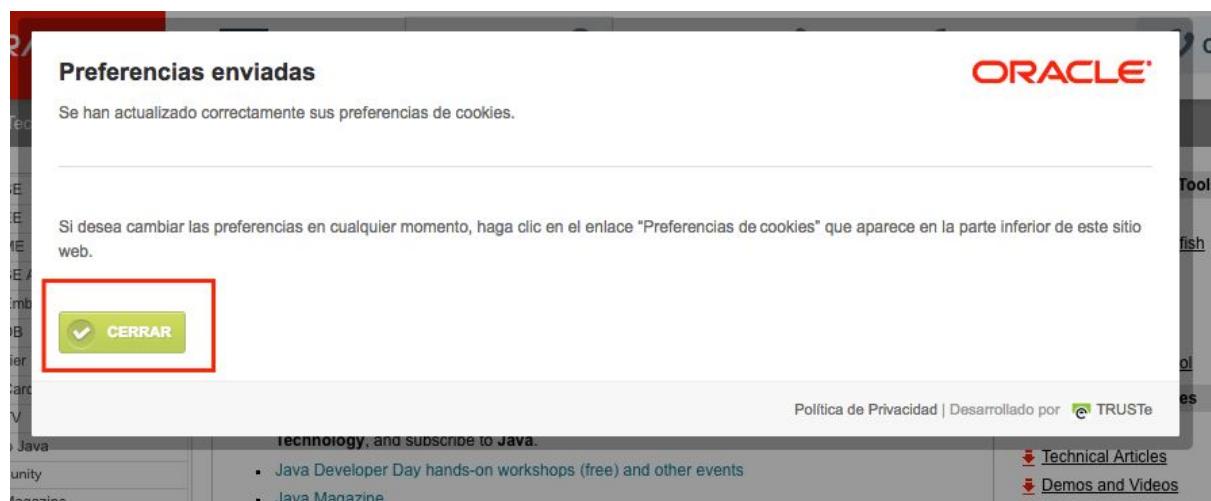
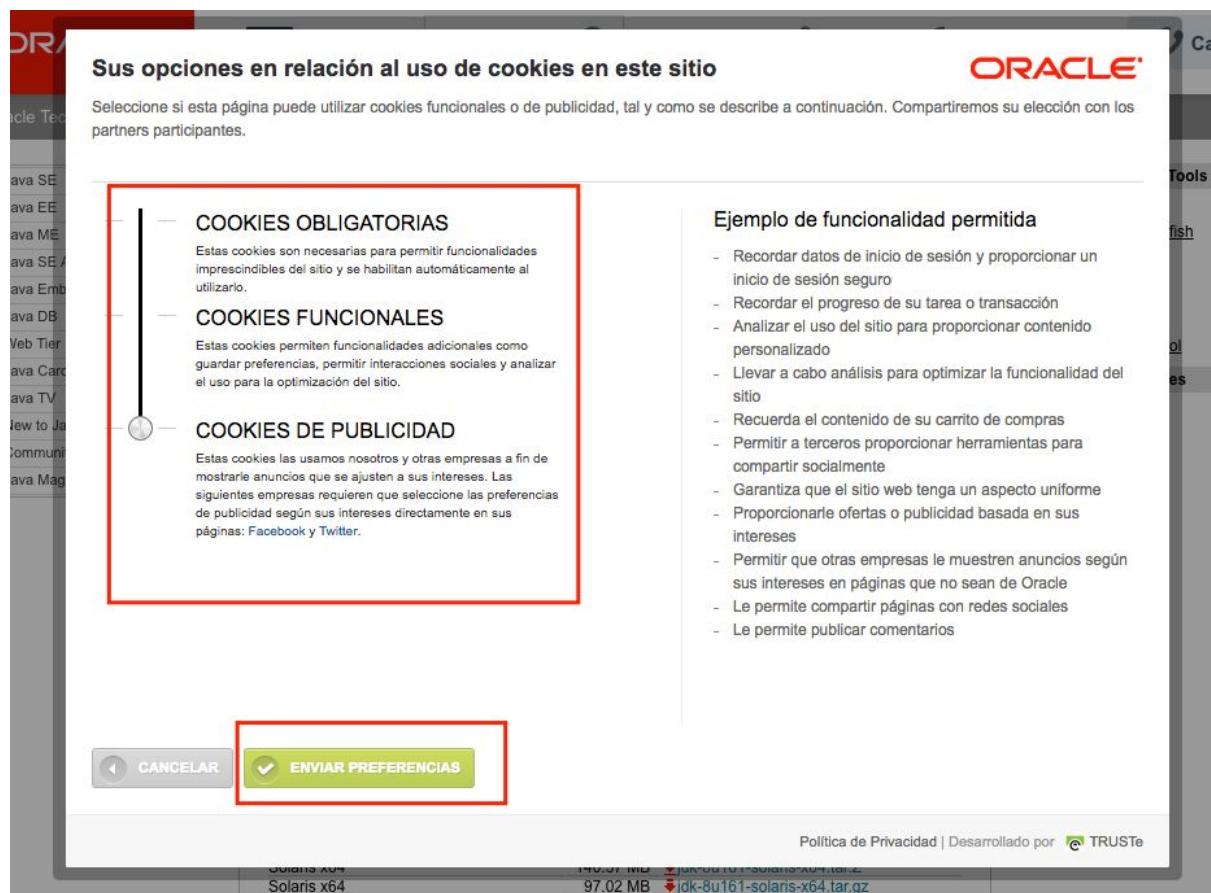
Instalar el JDK

El Java Development Kit (JDK por sus siglas en inglés) se trata de un conjunto de herramientas que permiten desarrollar programas en lenguaje Java. Cómo Android se basa en este lenguaje, se hace necesario de esta herramienta. Para ello deberás dirigirte al siguiente enlace.

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>.

Aquí te aparecen unos permisos del uso Cookies que solicita la página para poder entrar y descargar la herramienta. Para esto por favor seguir los siguientes pasos:





Luego de realizar la verificación de las cookies, seleccionamos la opción que dice **Aceptar el acuerdo de licencia** en la parte superior de la ventana. Debajo verás una lista de enlaces justo al lado de los sistemas operativos. En nuestro caso debes buscar donde dice 64 bits de windows.

Seguido dale clic al enlace que está a su lado.

Nota: Si estás trabajando con macOS descarga el instalador dmg que aparece en la imagen.

The screenshot shows the Oracle Technology Network Java SE Downloads page. On the left, there's a sidebar with links like Java SE, Java EE, Java ME, etc. The main content area has tabs for Overview, Downloads (which is selected), Documentation, Community, Technologies, and Training. Below the tabs, it says "Java SE Development Kit 8 Downloads". It thanks users for downloading the Java Platform, Standard Edition Development Kit (JDK). It states that the JDK is a development environment for building applications, applets, and components using the Java programming language. It includes a note about tools for developing and testing programs. A "See also:" section lists Java Developer Newsletter, Java Developer Day workshops, and Java Magazine. At the bottom, there are checksum links for JDK 8u161 and 8u162.

Java SE Development Kit 8u161 Downloads

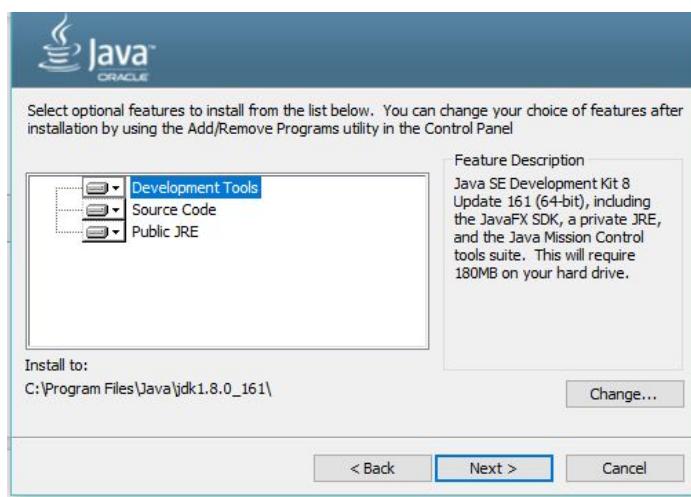
You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.92 MB	jdk-8u161-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.88 MB	jdk-8u161-linux-arm64-vfp-hflt.tar.gz
Linux x86	168.96 MB	jdk-8u161-linux-i586.rpm
Linux x86	183.76 MB	jdk-8u161-linux-i586.tar.gz
Linux x64	166.09 MB	jdk-8u161-linux-x64.rpm
Linux x64	180.97 MB	jdk-8u161-linux-x64.tar.gz
macOS	247.12 MB	jdk-8u161-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.99 MB	jdk-8u161-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.29 MB	jdk-8u161-solaris-sparcv9.tar.gz
Solaris x64	140.57 MB	jdk-8u161-solaris-x64.tar.Z
Solaris x64	97.02 MB	jdk-8u161-solaris-x64.tar.gz
Windows x86	198.54 MB	jdk-8u161-windows-i586.exe
Windows x64	206.51 MB	jdk-8u161-windows-x64.exe

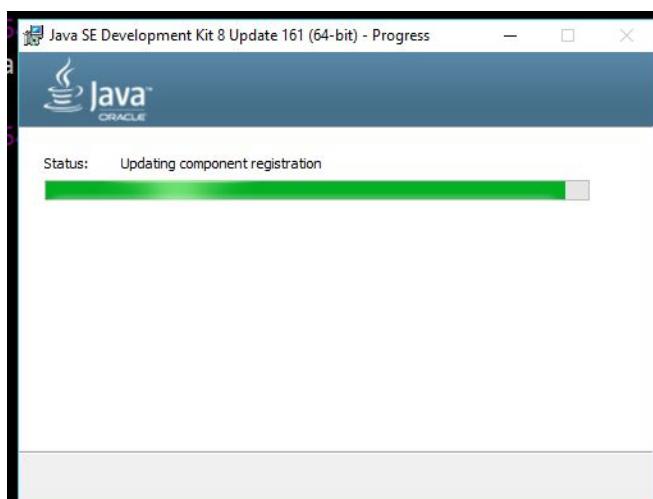
En breves instantes comenzará la descarga. Una vez terminada la descarga, abrirás el archivo. Al hacerlo verás la ventana de bienvenida. Para comenzar la instalación debes dar clic en el botón **Next**, como se muestra la imagen.



Luego debes continuar dando clic en los botones **Next** como se ve en la imagen



Verás que se muestra el progreso de la instalación.



El Java JDK incluye también el Java Runtime Environment (JRE por sus siglas en inglés). Éste es el programa que se usa para ejecutar aplicaciones escritas en el lenguaje Java. En otras palabras con el JDK las desarrollamos, con el JRE las ejecutamos. Entonces también debes instalarlo y por ello te saldrá una ventana preguntándote donde quieres instalarlo. Te recomendamos que dejes la carpeta por defecto para que funcionen bien el JDK con el JRE. Luego de seleccionar la carpeta donde deseas que se instale el JRE dale clic al botón Siguiente.



Automáticamente comenzará la instalación del Java JRE también. Verás el progreso de la instalación en una ventana similar a la que se muestra en la siguiente imagen

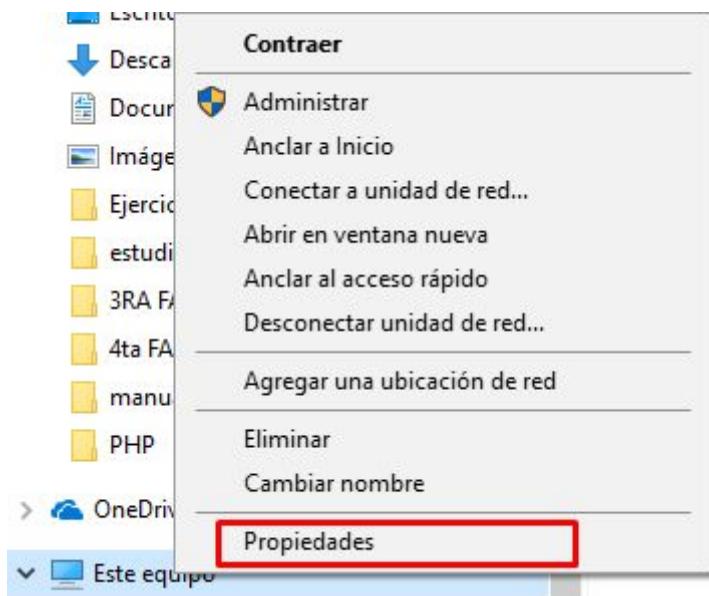


Verás que la instalación está completada correctamente cuando veas la ventana como la que se muestra a continuación. Para terminar con la misma dar clic en el botón **Close**.

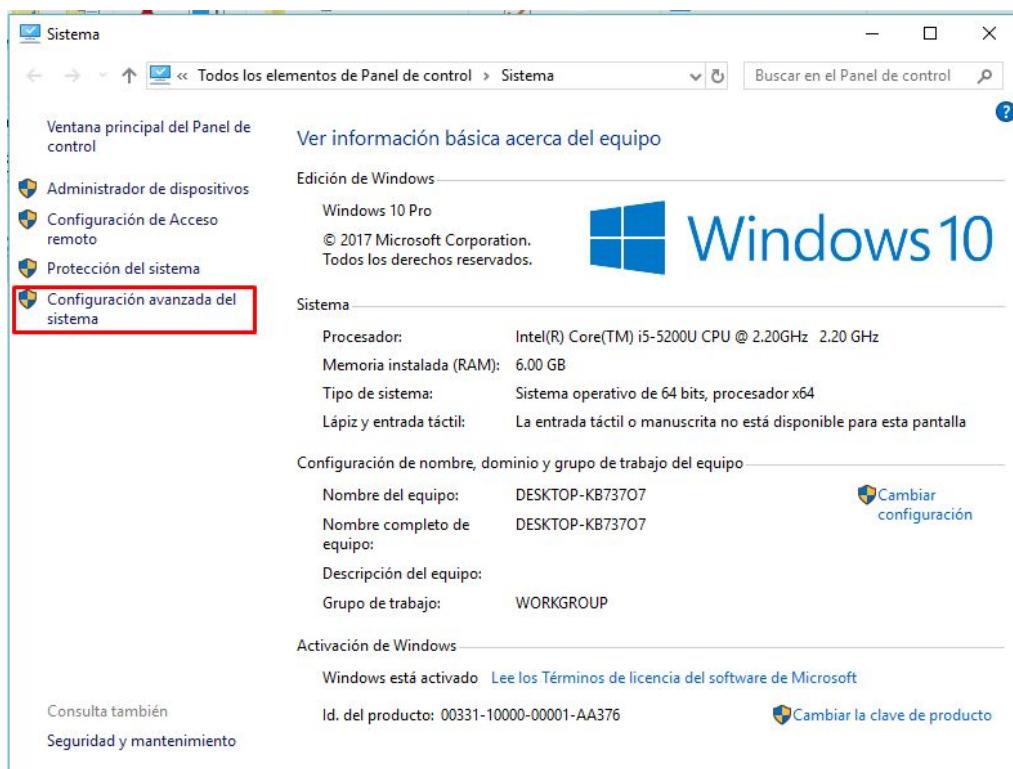


Luego de su instalación, es necesario crear las variables de entorno para que el sistema operativo pueda buscar los ejecutables necesarios desde la línea de comandos o la ventana Terminal. Para esto entonces realicemos los siguientes paso:

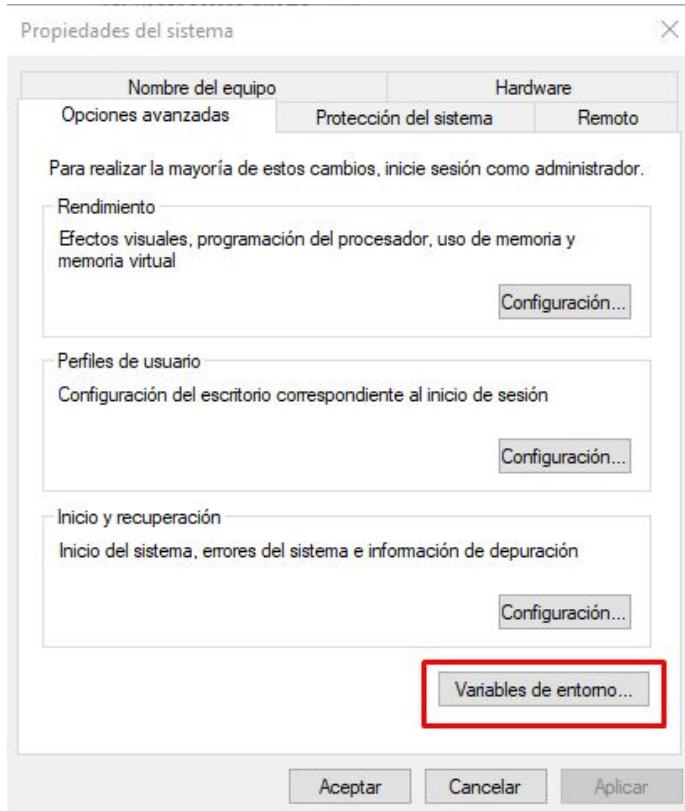
Abre un explorador de archivos y da clic derecho en **Este Equipo**. Luego seleccionamos la opción de propiedades.



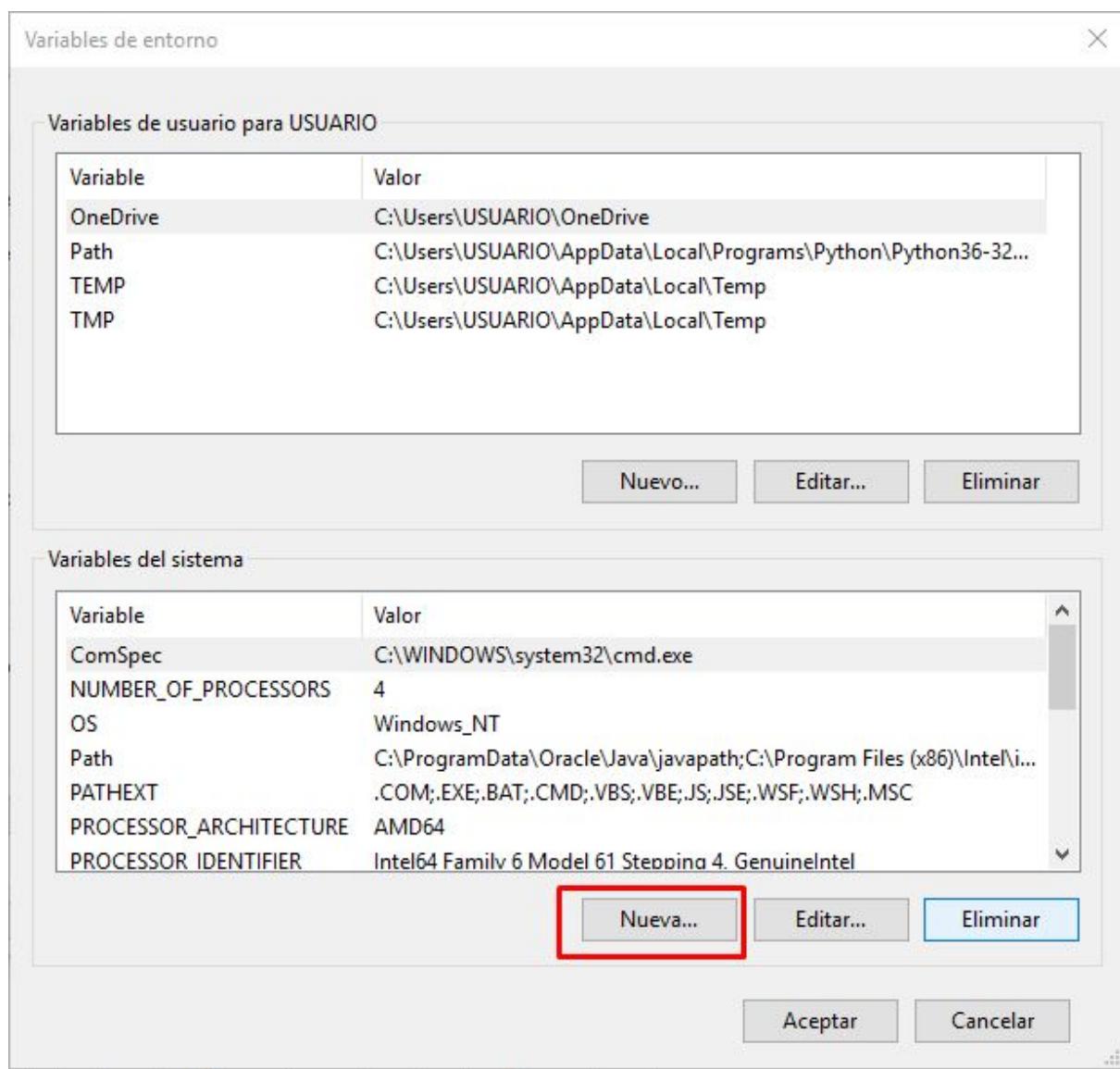
Luego en la pantalla que se nos habilita, seleccionamos **Configuración avanzada del sistema**.



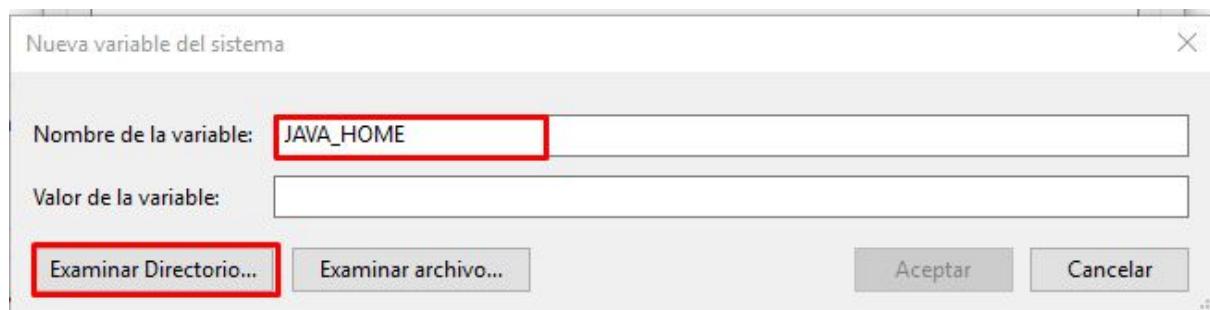
Haga clic en Variables de entorno.

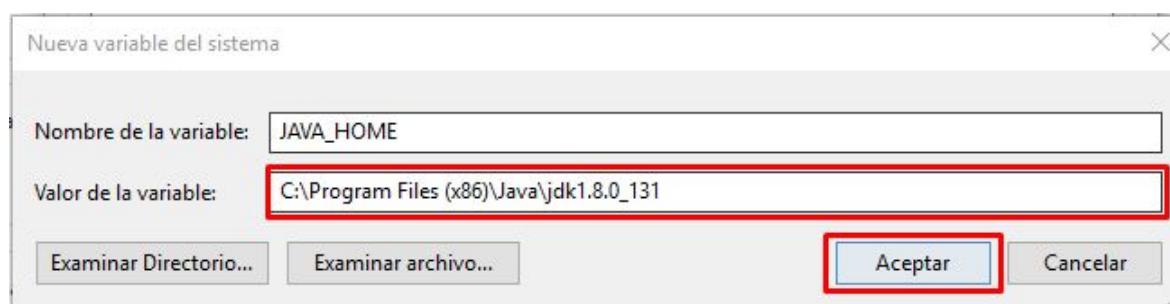
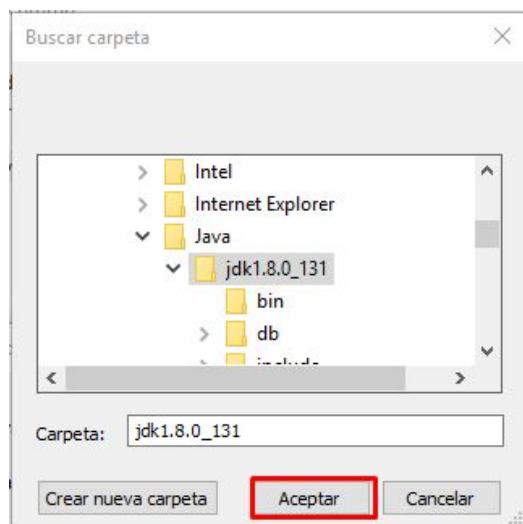


En la sección Variables del sistema de clic en Nueva..

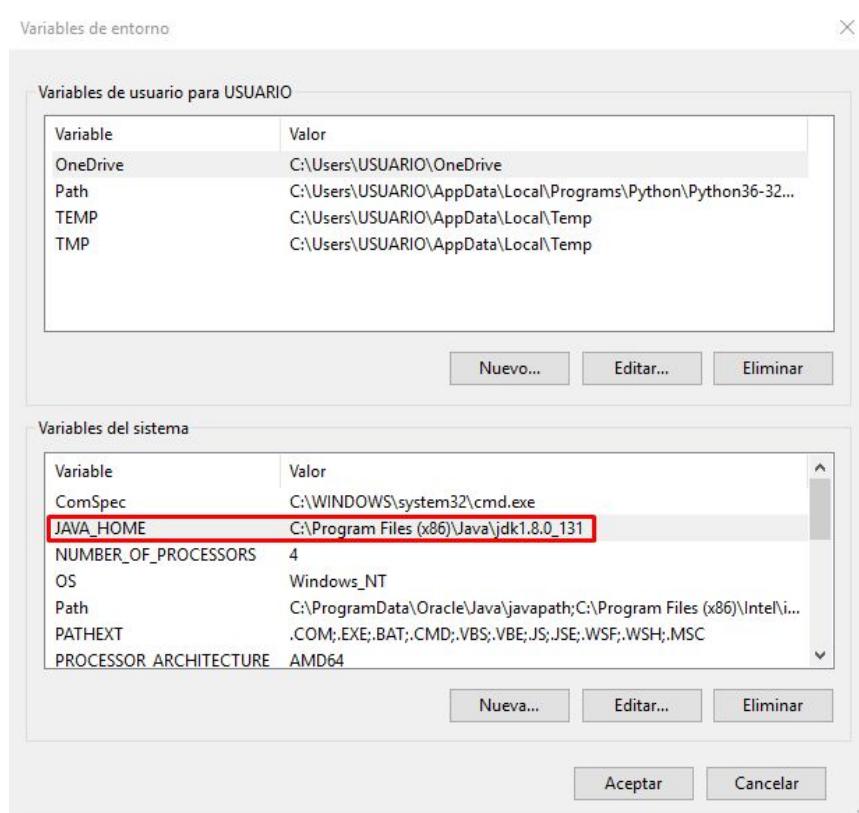


Ingresamos en el campo de Nombre de la Variable el valor de **JAVA_HOME** que será el nombre de nuestra variable. Posteriormente seleccionamos el botón de examinar directorio y buscamos la carpeta del JDK que normalmente está instalada en **C:/Program Files (x86)/Java/jdk1.8.0_131** y damos clic en aceptar.

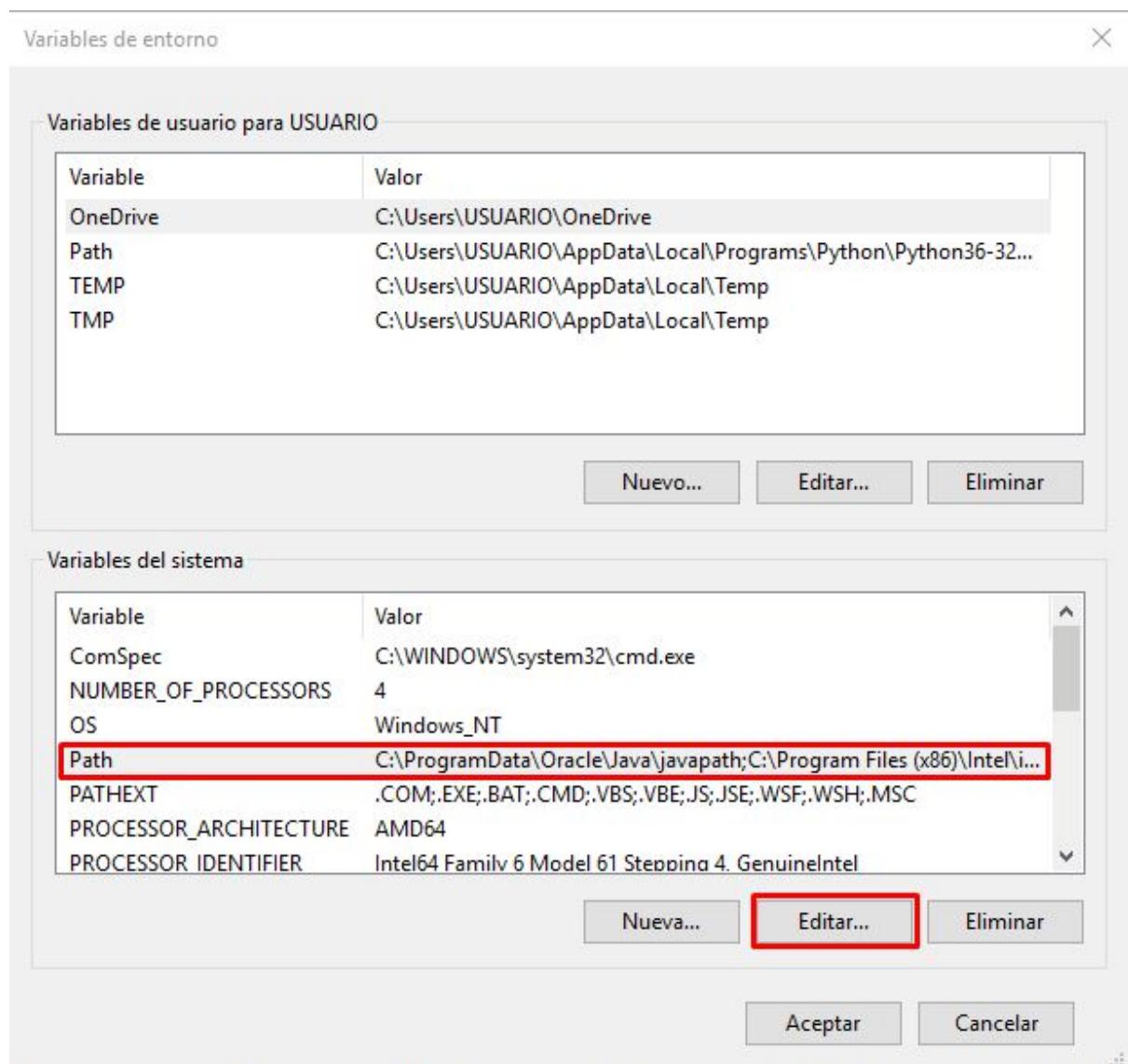




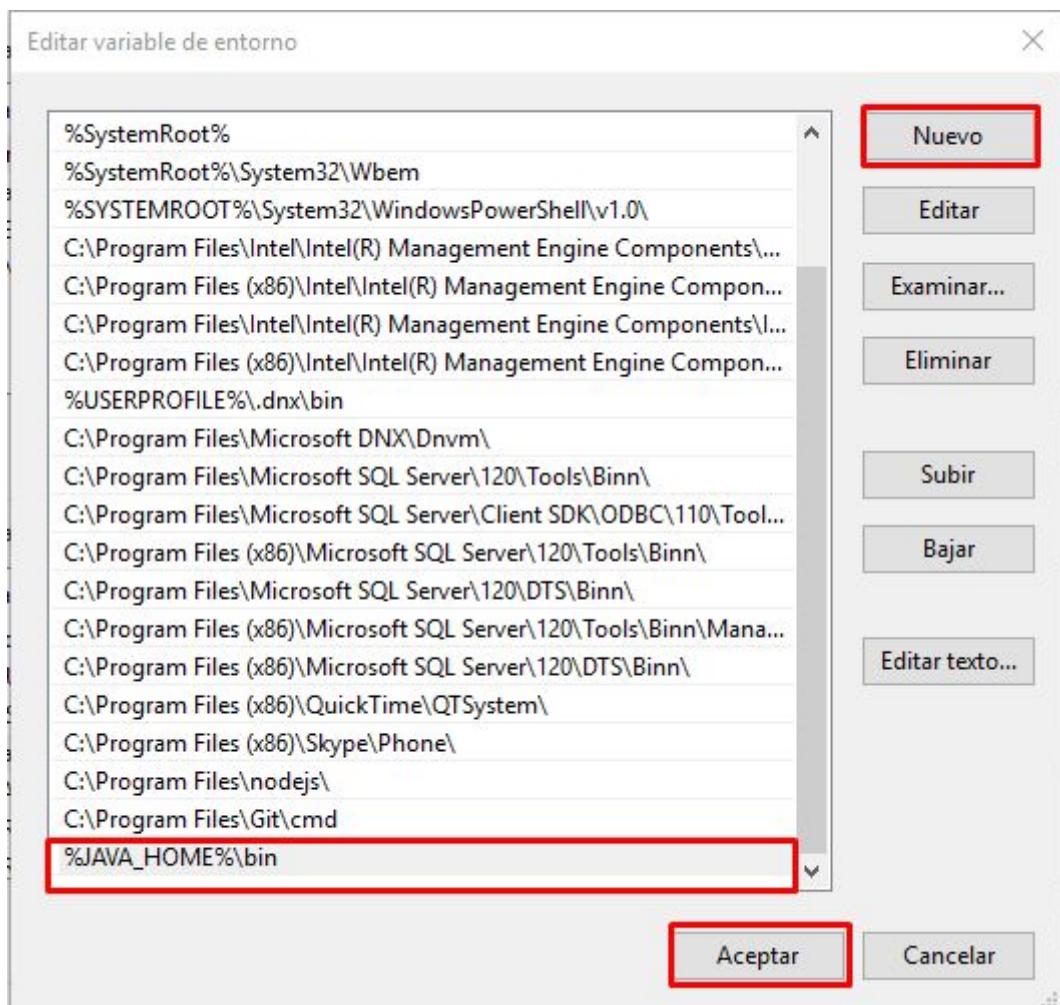
Al darle aceptar confirmamos de que nuestra variable haya quedado configurada correctamente.



De nuevo nos ubicamos en la sección de Variables del sistema y seleccionamos la variable de entorno PATH y hacemos clic en Editar



Luego de que se nos abra la ventana, damos clic en Nuevo e ingresamos el valor correspondiente para la carpeta bin de Java (**%JAVA_HOME%\bin**) y damos clic en aceptar.



Luego damos clic en aceptar en las ventanas que teníamos abiertas para que se apliquen los cambios a las variables del sistema que modificamos.

Instalación de Android Studio

Ya con JDK instalado y configurado, podemos proceder a la instalación de Android Studio en su última versión. Para eso ingresamos a la página oficial (<https://developer.android.com/studio/index.html?hl=es-419>) y damos clic en el botón de descarga.

Los requisitos mínimos para la instalación de Android son:

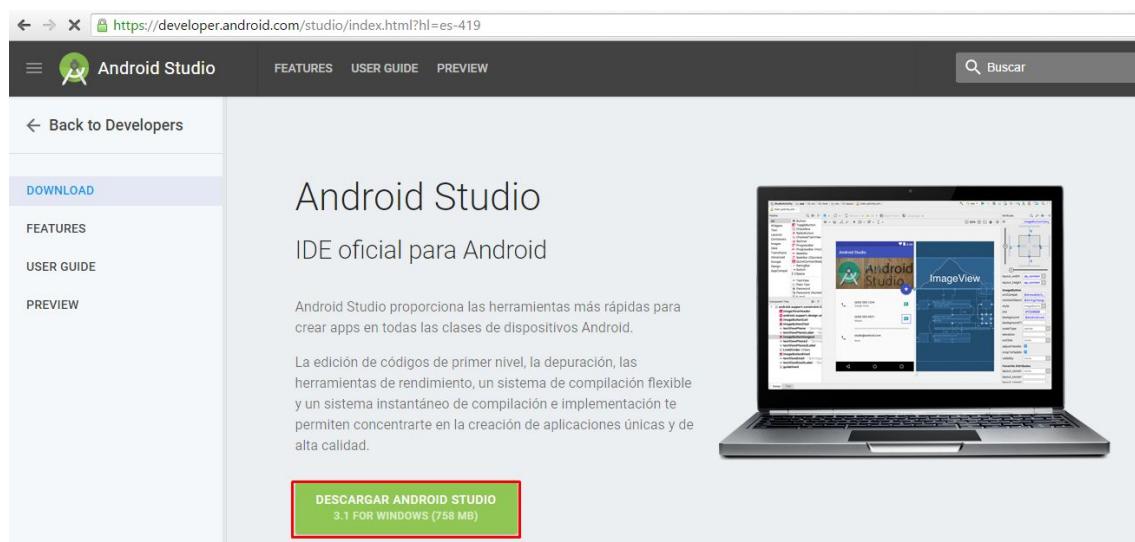
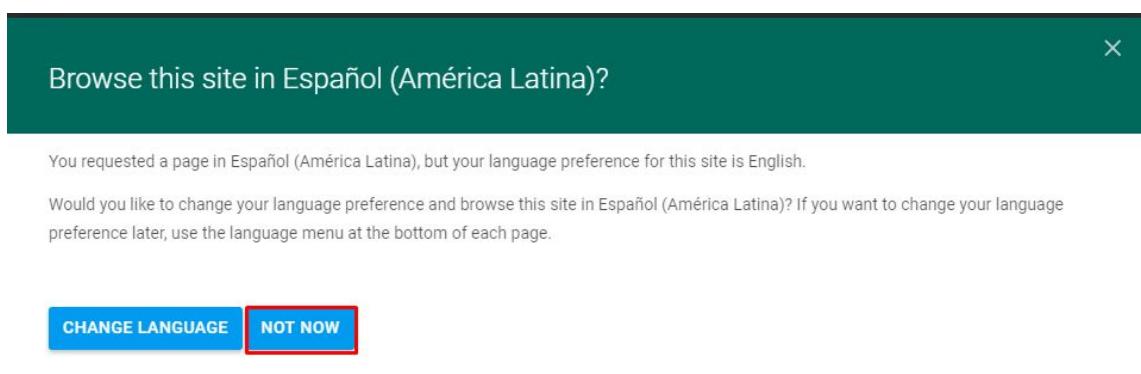
- 2 GB de memoria RAM como mínimo (8 recomendado)
- Resolución de pantalla mínima de 1280 x 800
- Windows 7/8/10 (32 o 64 bits)

Para el emulador acelerado: Sistema operativo de 64 bits y procesador Intel® compatible con Intel® VT-x, Intel® EM64T (Intel® 64) y la funcionalidad Execute Disable (XD) Bit.

Nota: Puede que se nos abra una ventana que nos pide cambiar de lenguaje. Si deseas tener el Android Studio en español, puedes seleccionar cambiar el lenguaje, de lo contrario (nuestro caso) le pueden dar Not Now.

Si tuviéramos problemas con el JDK o Java al ejecutar Android Studio luego de haberlo instalado debemos realizar lo siguiente:

- Ir a: Clic derecho en Equipo > Propiedades > Menú: Configuración avanzada del sistema > Botón: Variables de entorno (en la pestaña opciones avanzadas)
- Adicionamos una nueva variable de sistema llamada JAVA_HOME y que apunte donde se instaló el JDK, por ejemplo C:\Program Files\Java\jdk_version



Luego de dar clic, nos sale una ventana para aceptar los términos y condiciones del uso de la aplicación, chequeamos los ítems de condiciones y luego descargar.

Before downloading, you must agree to the following terms and conditions.

Terms and Conditions

This is the Android Software Development Kit License Agreement

1. Introduction

1.1 The Android Software Development Kit (referred to in the License Agreement as the "SDK" and specifically including the Android system files, packaged APIs, and Google APIs add-ons) is licensed to you subject to the terms of the License Agreement. The License Agreement forms a legally binding contract between you and Google in relation to your use of the SDK.

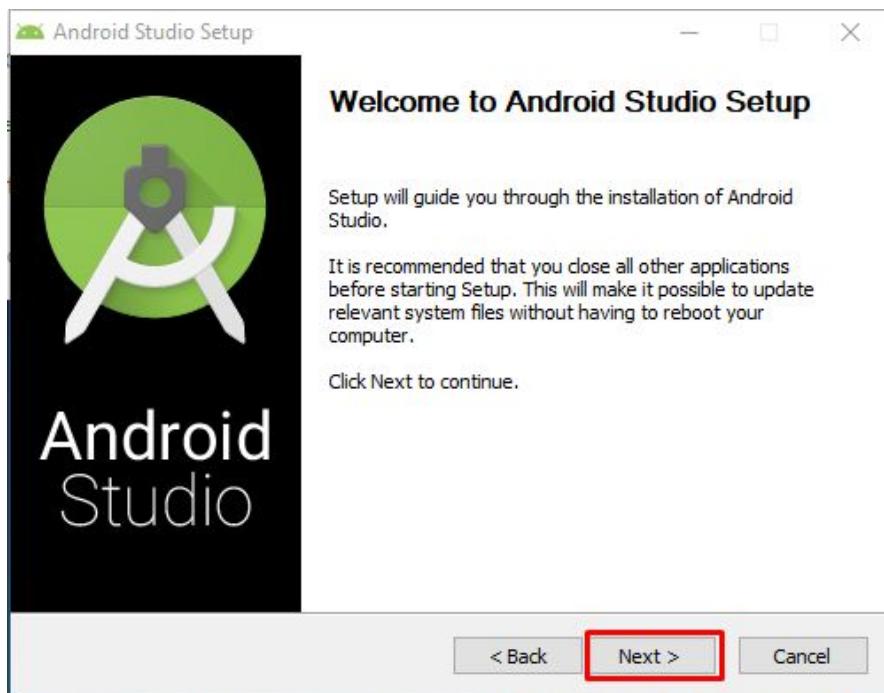
1.2 "Android" means the Android software stack for devices, as made available under the Android Open Source Project, which is located at the following URL: <http://source.android.com/>, as updated from time to time.

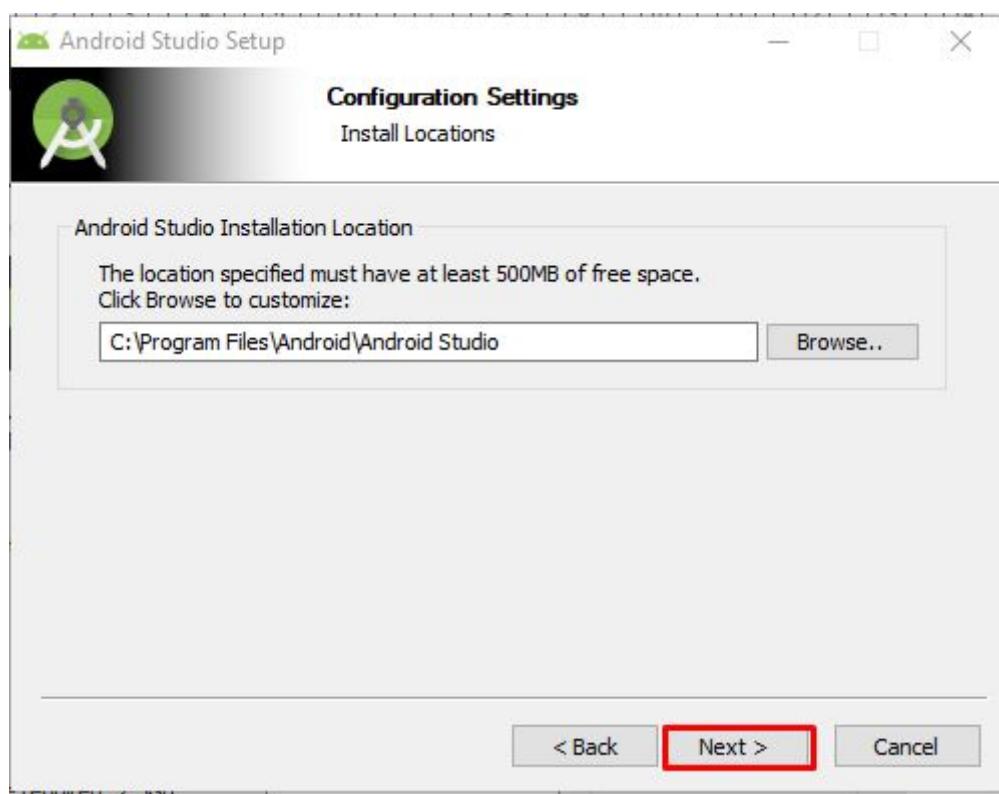
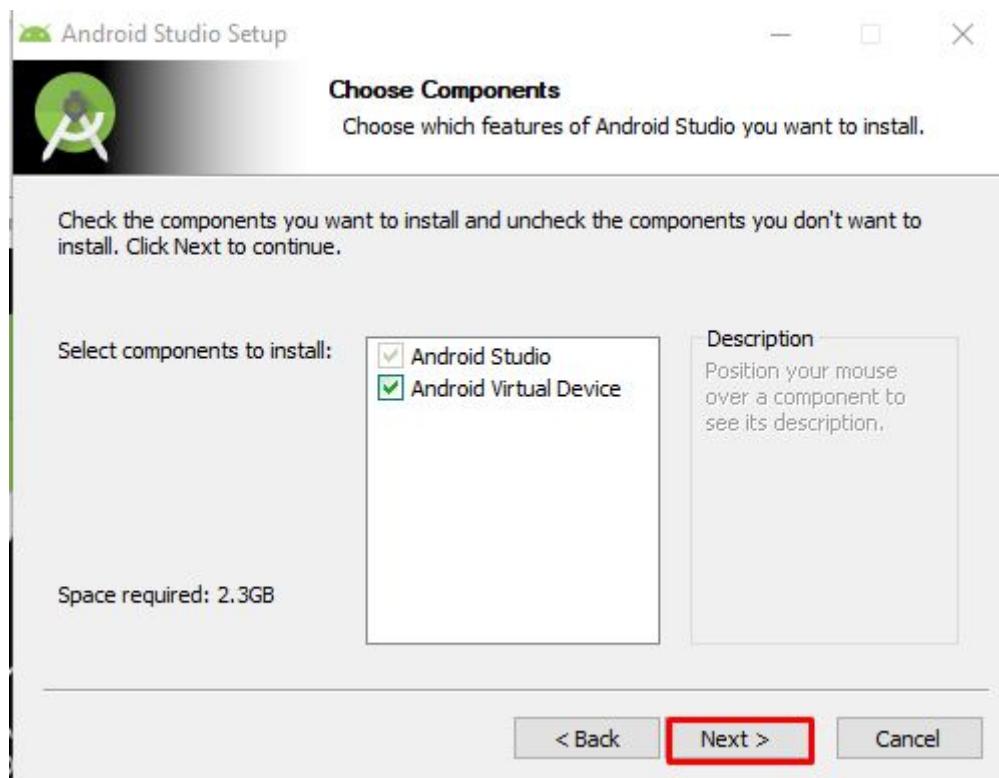
1.3 A "compatible implementation" means any Android device that (i) complies with the Android Compatibility Definition document, which

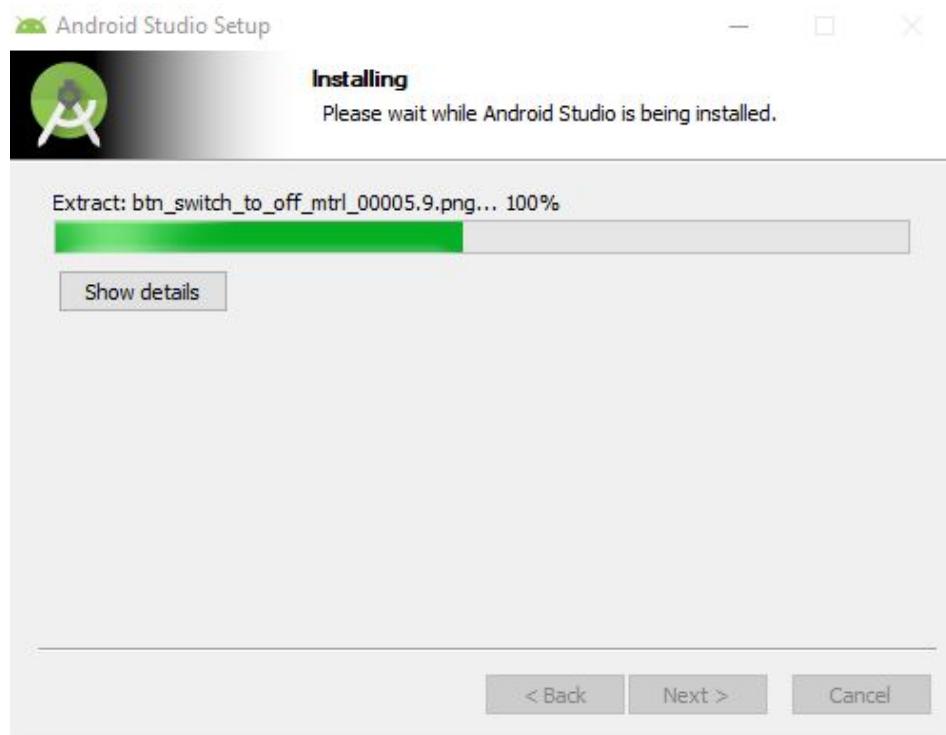
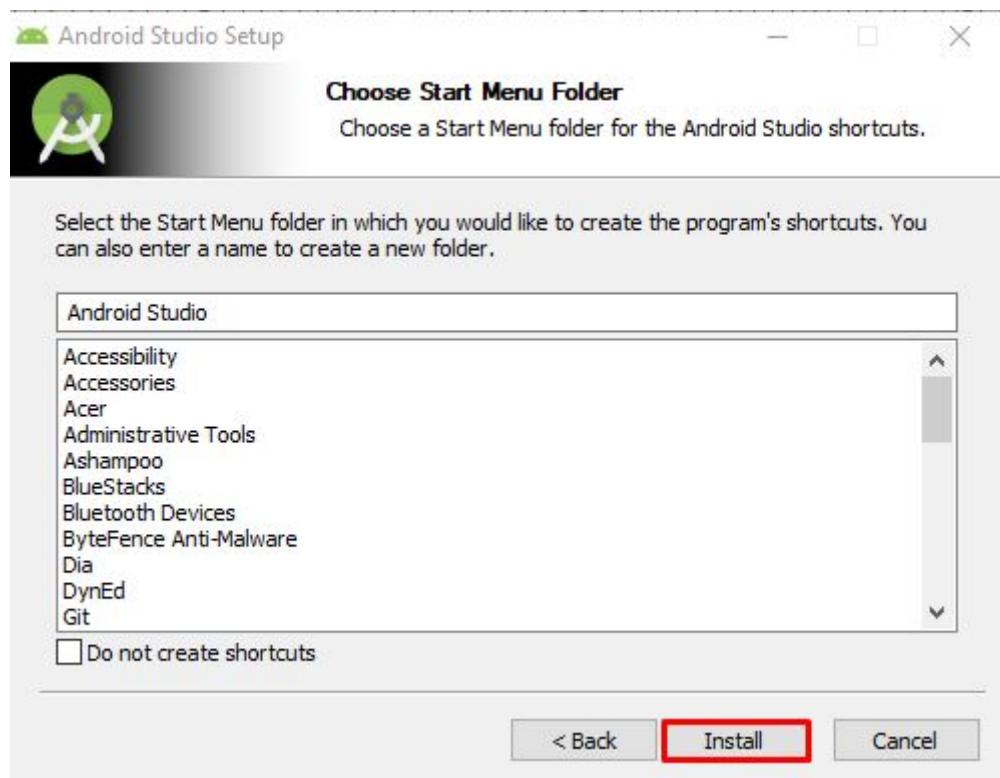
I have read and agree with the above terms and conditions

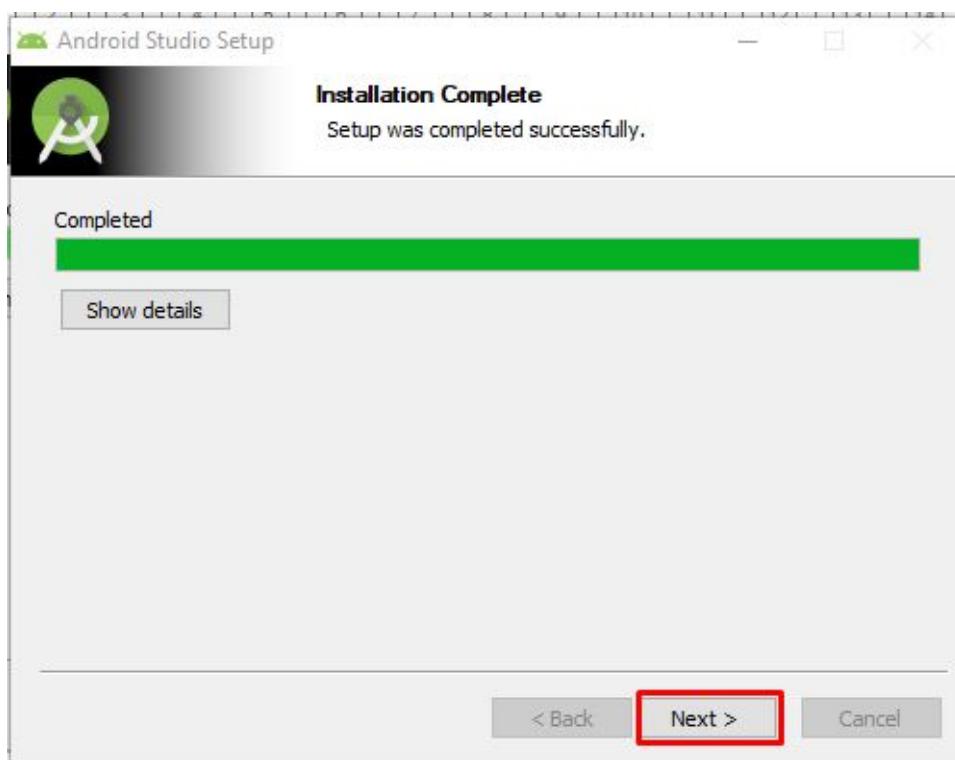
DOWNLOAD ANDROID STUDIO FOR WINDOWS

Luego de la descarga, procedemos a la instalación del android studio para lo que hacemos Next (Siguiente) que es lo normal en cualquier instalador de Windows.

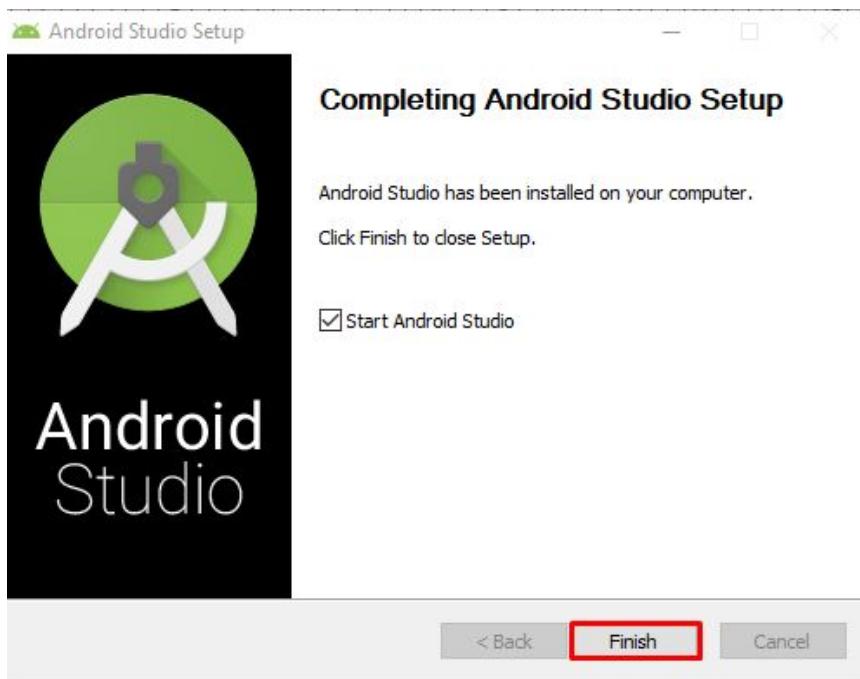




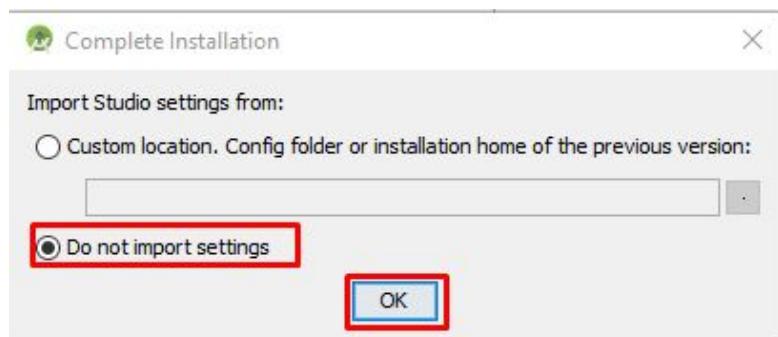




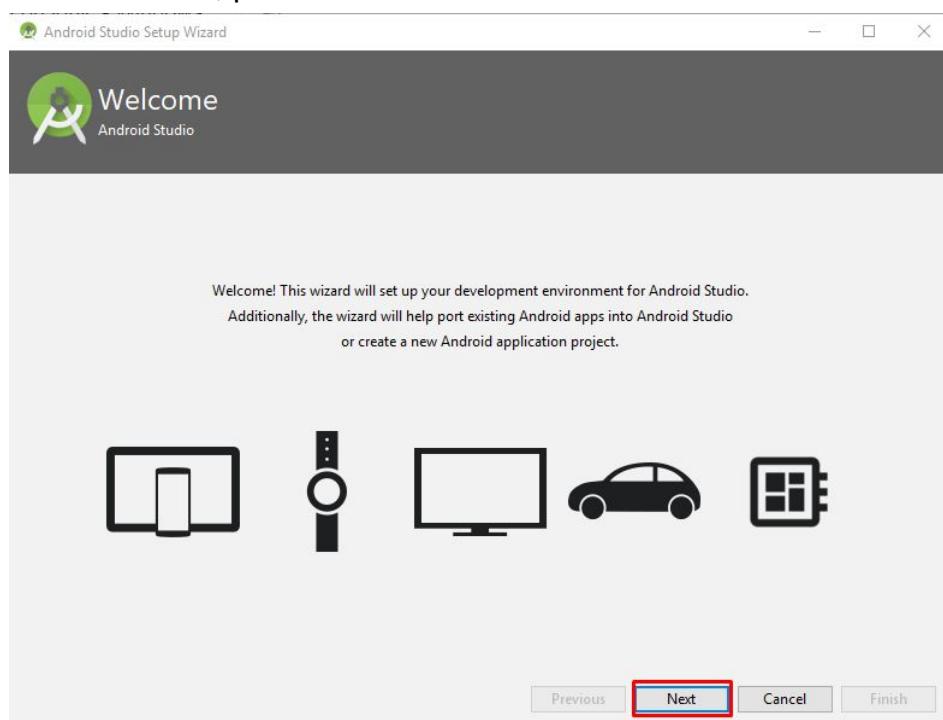
Ya para finalizar la instalación, presionamos el botón de **finish** y esperamos que abra el programa.



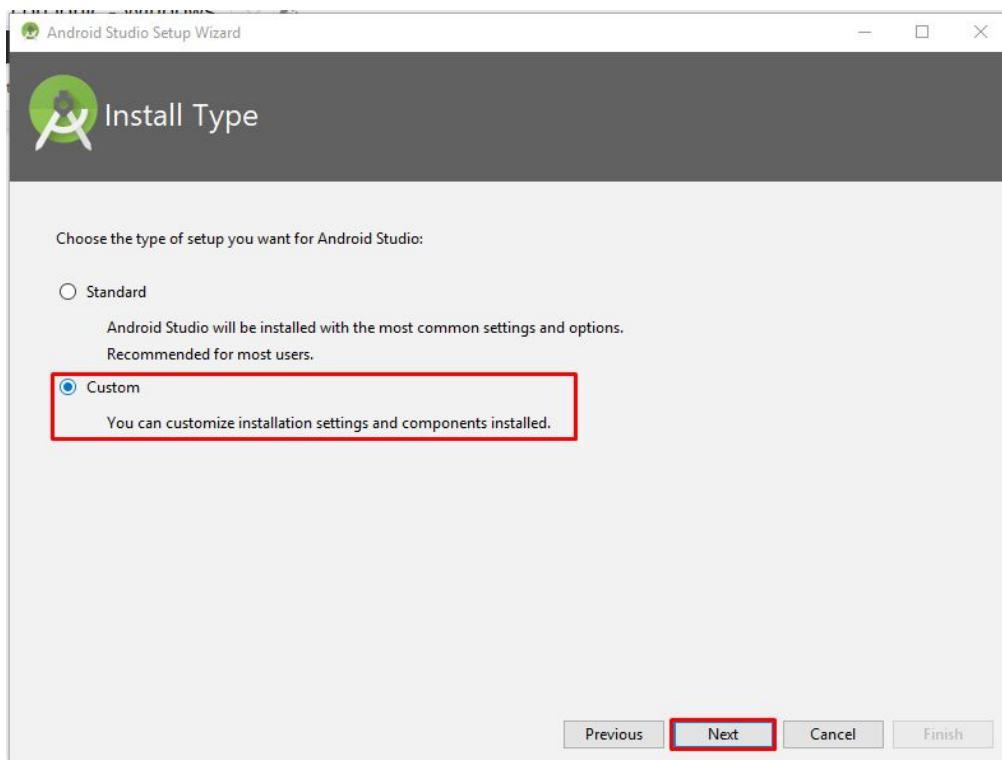
Cuando abre, nos pide si deseamos importar las configuraciones de otra instalación. Si no cuentas con estas configuraciones de otra instalación, seleccionamos No importar las configuraciones.



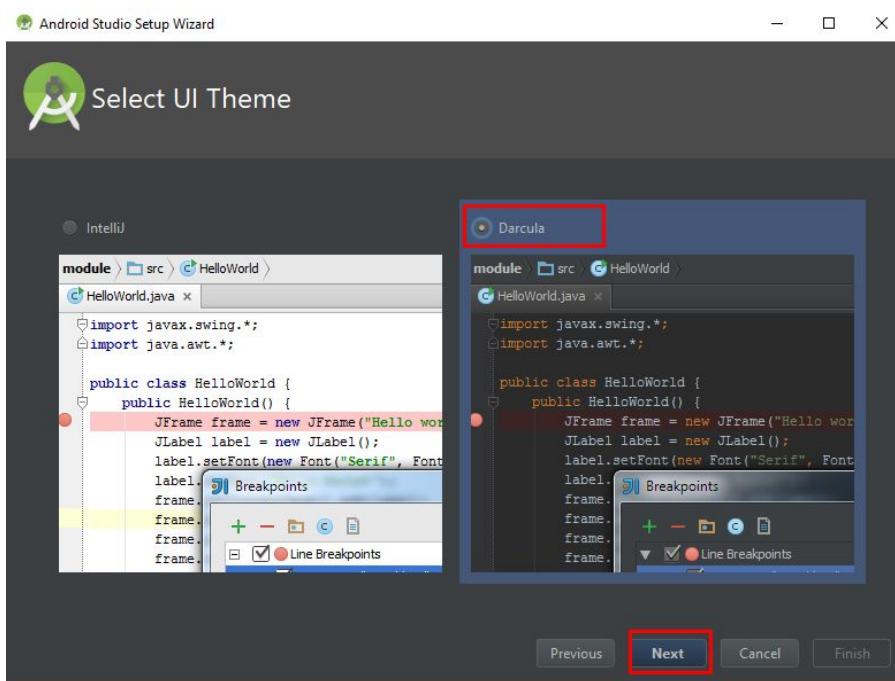
A continuación, presionamos continuar.



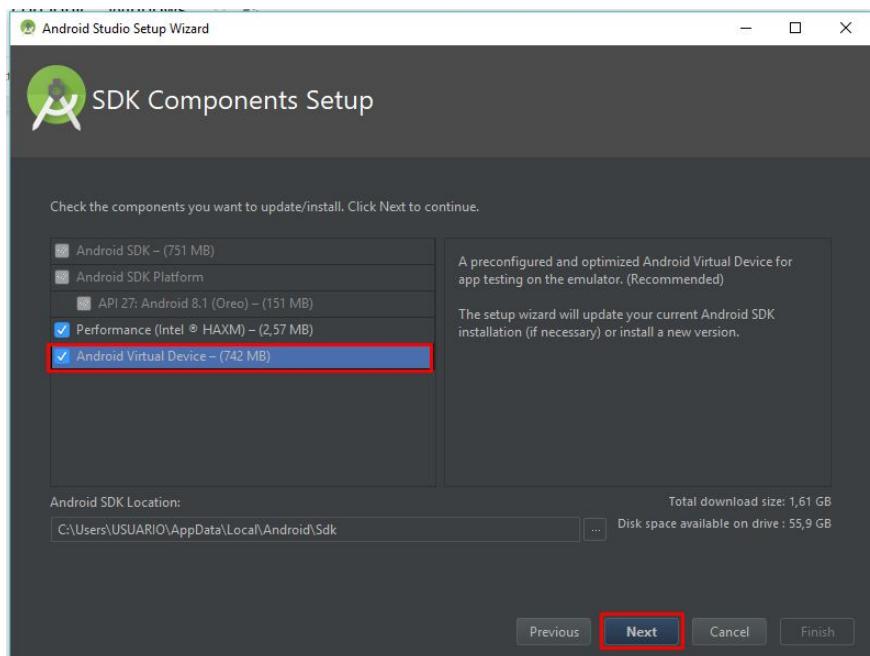
A continuación, se presentan dos opciones, si deseamos las configuraciones estándar o personalizadas. En nuestro caso, seleccionamos personalizadas.



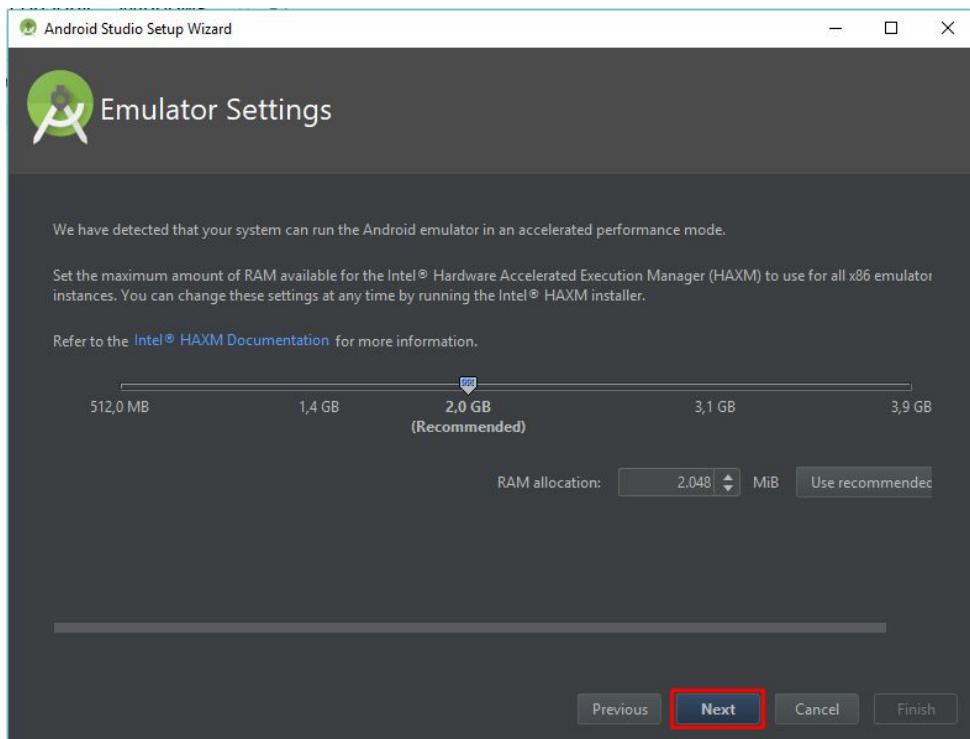
Para ayudarle un poco a nuestra vista, seleccionamos el tema de visualización oscuro que nos provee Android Studio. Si es de tu agrado el tema blanco, lo puedes seleccionar sin ningún problema.



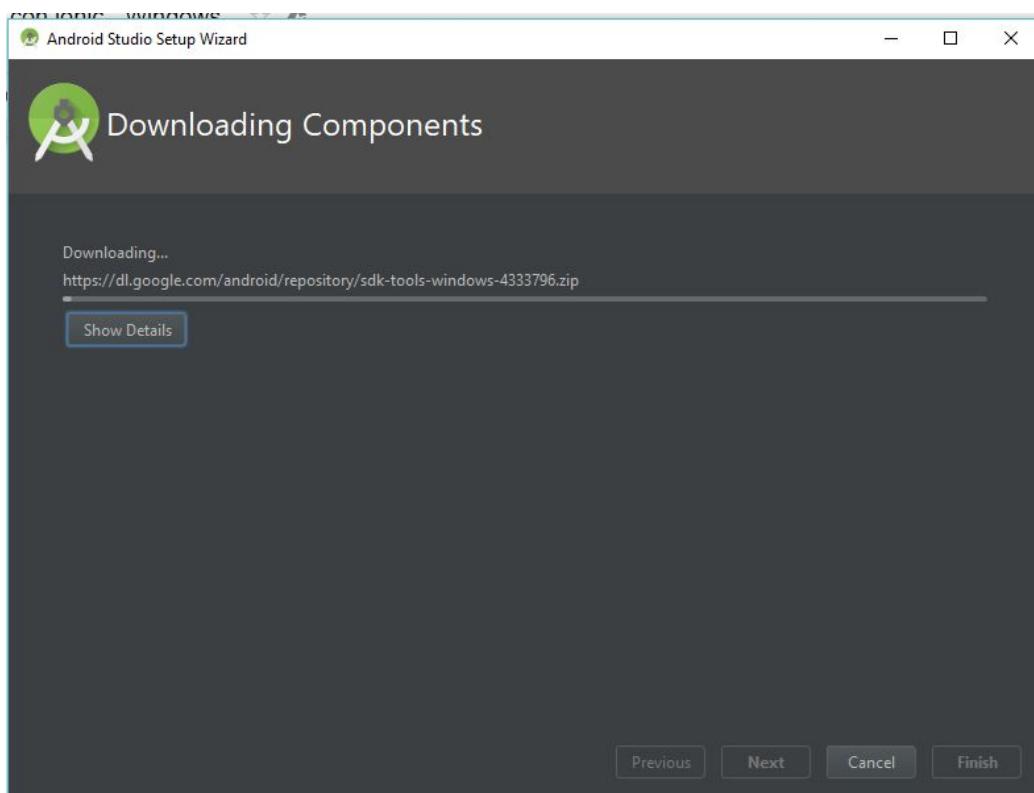
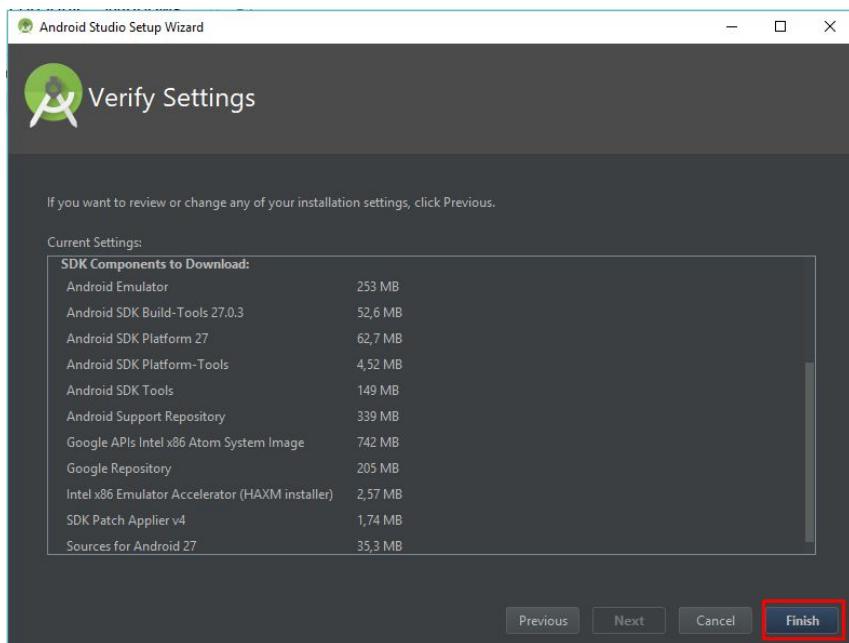
Para continuar, seleccionamos el dispositivo virtual si queremos realizar pruebas en nuestra máquina con emuladores de diferentes dispositivos. Es recomendado utilizar el dispositivo físico por temas de rendimiento con nuestra máquina.



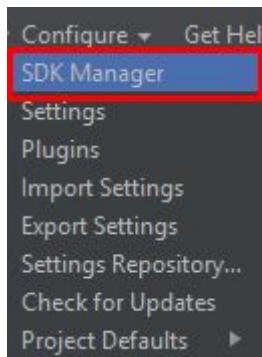
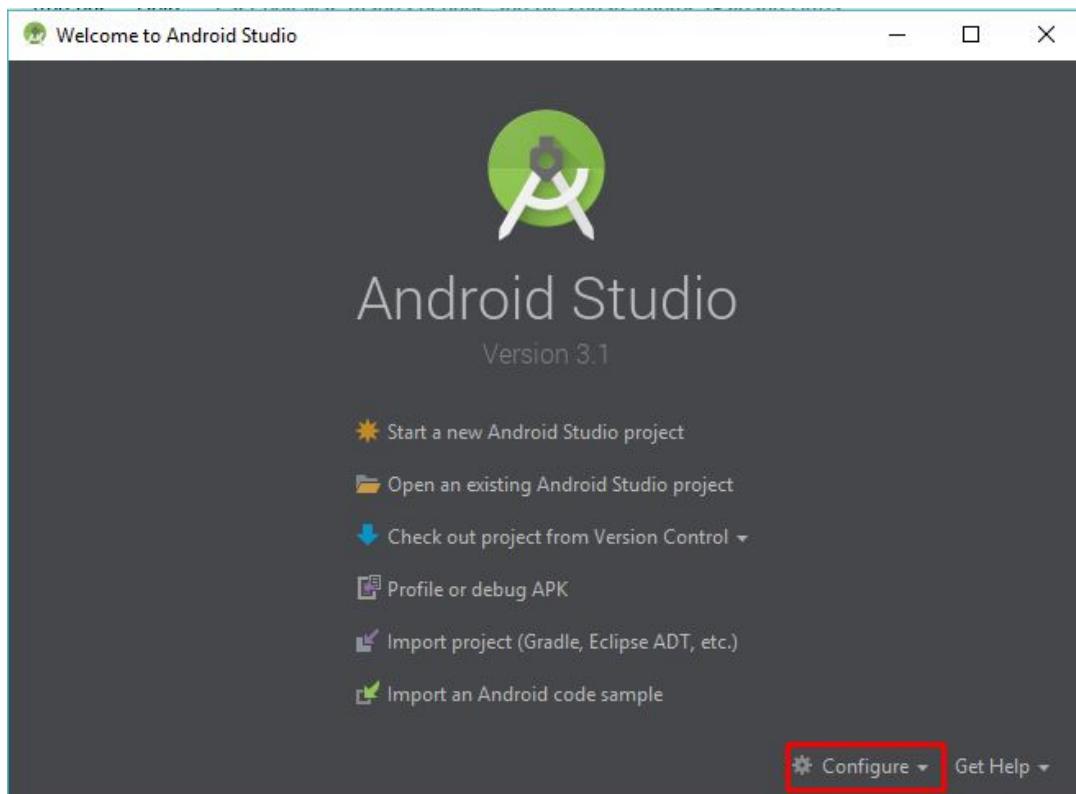
Para utilizar la memoria RAM para el dispositivo virtual, Android Studio nos recomienda utilizar 2.0GB. Puedes subirlo dependiendo de la cantidad de RAM con la que cuentes. No es recomendado menor cantidad.



Damos clic en finalizar y esperamos que culmine el proceso de descarga de los componentes iniciales que necesito Android Studio para lanzarse.

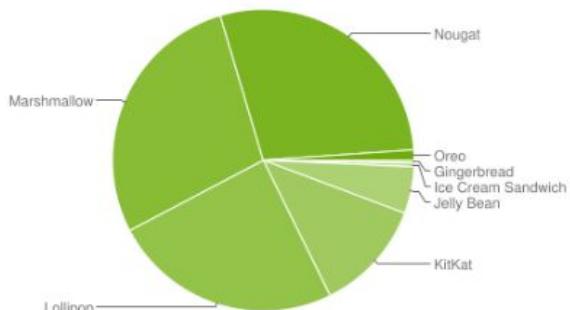


Ya con la configuración inicial, es necesario instalar varios SDK para que la compilación de nuestra aplicación se pueda escoger el más adecuado.



A continuación debes seleccionar las versiones del SDK para las cuales vas a trabajar. Es importante que conozcas la cantidad de dispositivos que cuentan con esa versión y puedas alcanzar el máximo de usuarios disponibles. Para escoger la plataforma específica en la cual desarrollarás, pudieses mirar esta gráfica y tomar la decisión con tu equipo de trabajo o con tu cliente:

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.4%
4.1.x	Jelly Bean	16	1.7%
4.2.x		17	2.6%
4.3		18	0.7%
4.4	KitKat	19	12.0%
5.0	Lollipop	21	5.4%
5.1		22	19.2%
6.0	Marshmallow	23	28.1%
7.0	Nougat	24	22.3%
7.1		25	6.2%
8.0	Oreo	26	0.8%
8.1		27	0.3%



Datos recopilados durante un periodo de 7 días hasta 8/1/2018.

No se muestran versiones con una distribución inferior al 0,1%.

Tomado de <https://developer.android.com/about/dashboards/index.html>

Te recomiendo que instales los SDK para los cuales específicamente quieras desarrollar y el del dispositivo en donde vayas a Emular. No olvides incluir la “Support Library” y los “Google Play Services” para tareas que realizaremos durante todo el curso.

Actualmente Ionic es soportado por las siguientes versiones de los sistemas operativos:

- Android 4.4+
 - Si deseas que soporte 4.1 o inferior, debes utilizar un plugin de terceros llamado CrossWalk (<https://crosswalk-project.org/>)
- iOS 8+
- Windows 10 Universal App

Cordova que es el Framework que permite correr aplicaciones hechas con Ionic soporta actualmente diferentes versiones, según la versión que se instale (Actualmente nos encontramos en la versión 7.X.X aunque la documentación aparece 6.X.X)

Veamos la tabla de la página oficial:

Android Platform Guide

This guide shows how to set up your SDK environment to deploy Cordova apps for Android devices, and how to optionally use Android-centered command-line tools in your development workflow. You need to install the Android SDK regardless of whether you want to use these platform-centered shell tools or cross-platform Cordova CLI for development. For a comparison of the two development paths, see the [Overview](#). For details on the CLI, see [Cordova CLI Reference](#).

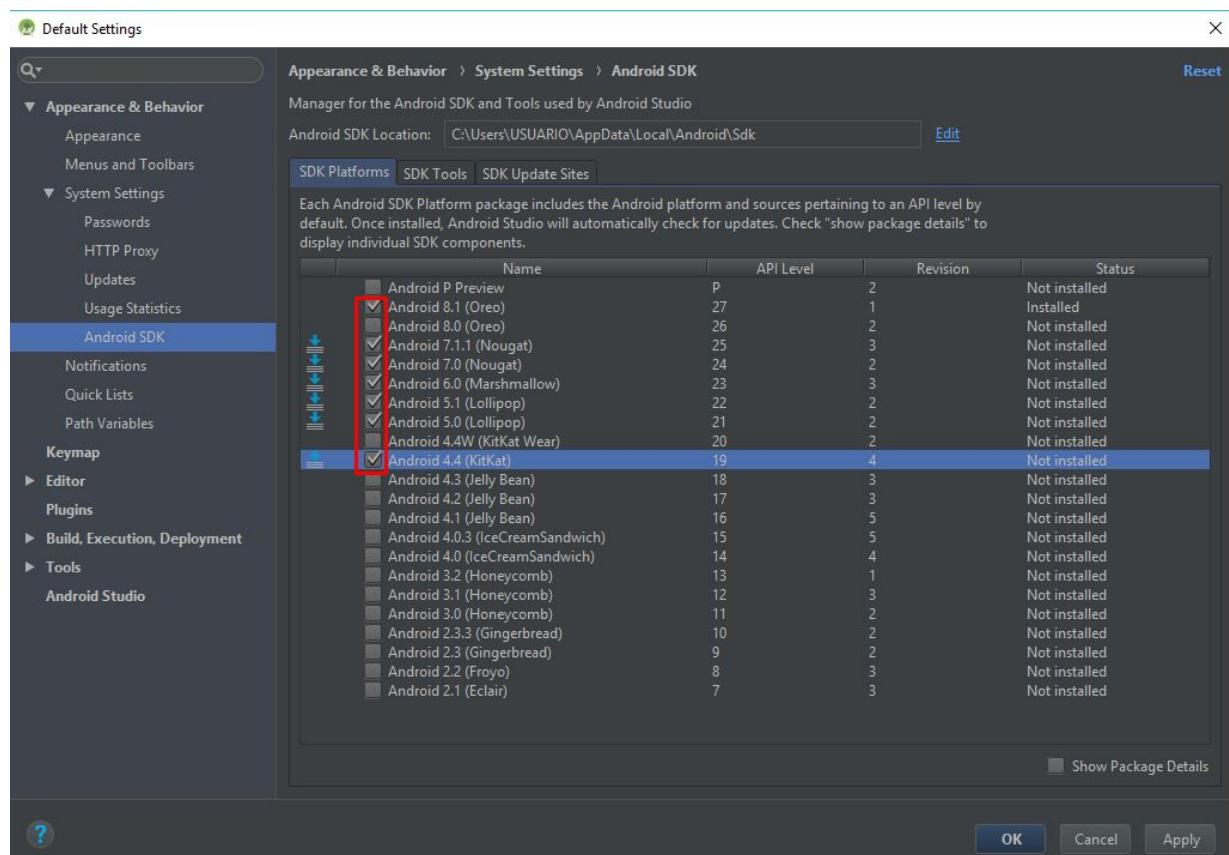
Requirements and Support

Cordova for Android requires the Android SDK which can be installed on OS X, Linux or Windows. See the [Android SDK's System Requirements](#). Cordova's latest Android package supports up to Android API Level 25. The supported Android API Levels and Android Versions for the past few cordova-android releases can be found in this table:

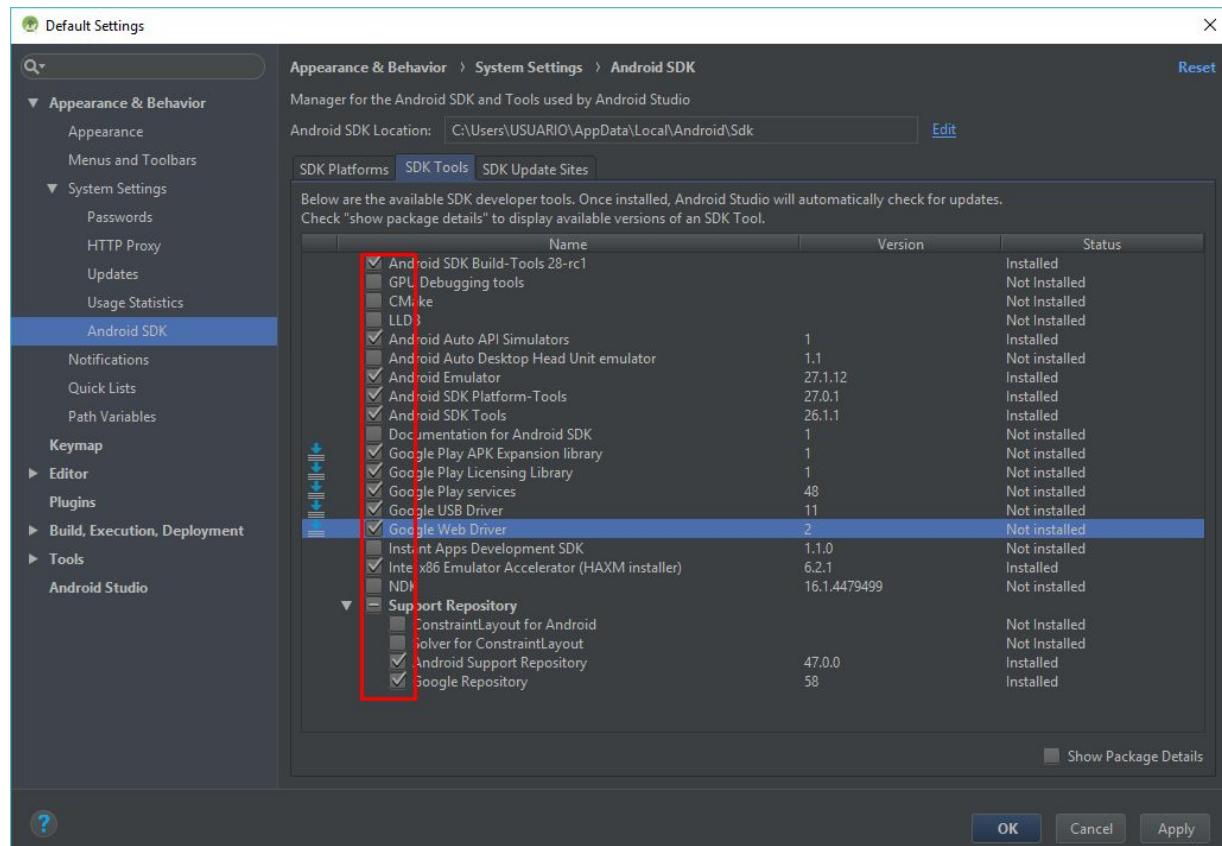
cordova-android Version	Supported Android API-Levels	Equivalent Android Version
6.XX	16 - 25	4.1 - 7.1.1
5XX	14 - 23	4.0 - 6.0.1
4.1X	14 - 22	4.0 - 5.1
4.0X	10 - 22	2.3.3 - 5.1
3.7X	10 - 21	2.3.3 - 5.0.2

En nuestro caso necesitamos las siguientes configuraciones:

SDK Platforms

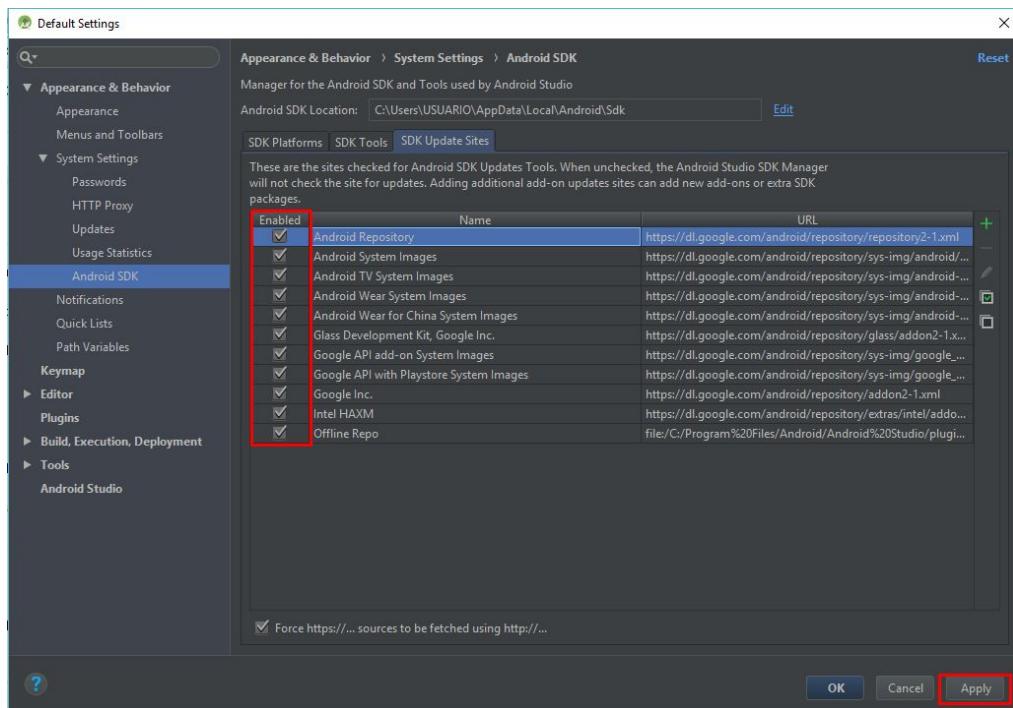


SDK Tools

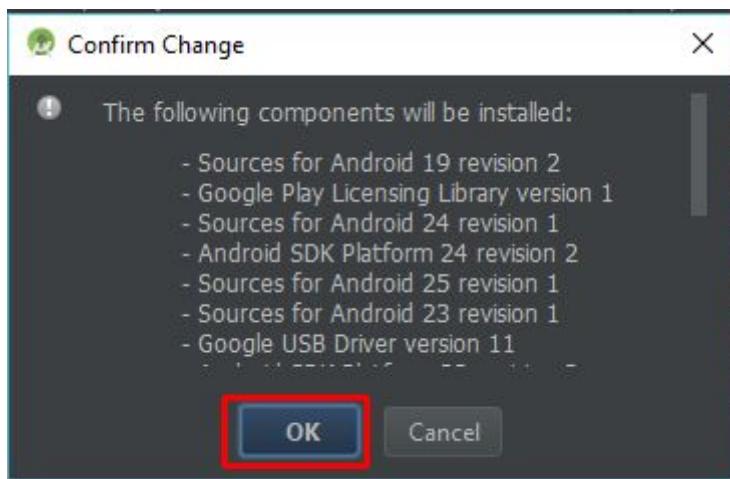


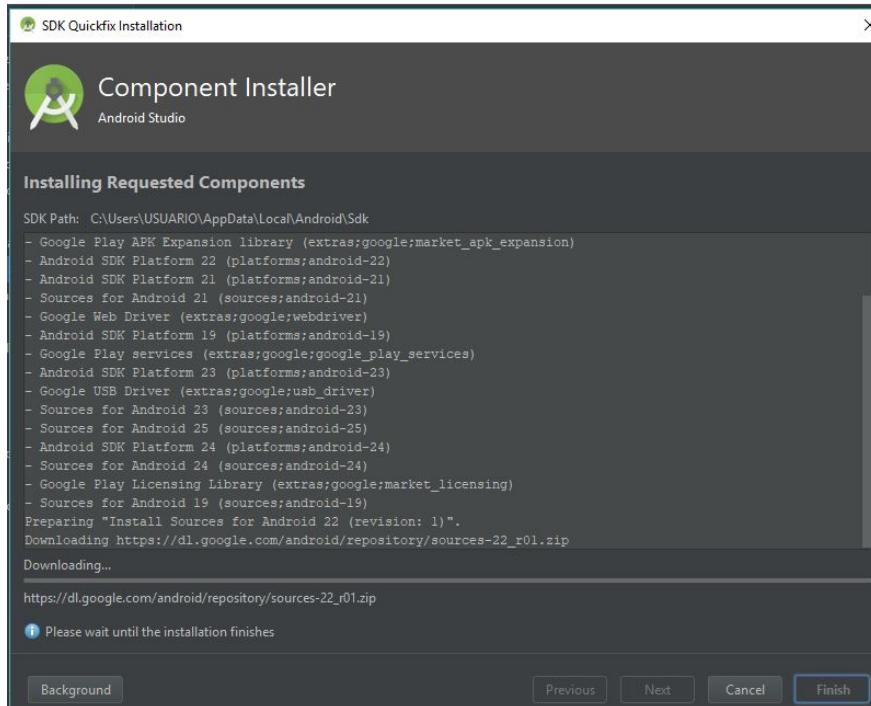
Si se fijan en las herramientas instaladas, incluye a Google Repository para el soporte de algunas funcionalidades de Android como Material, etc. Además se tienen varios servicios de la Play ya que en algún momento serán utilizados. Se recomienda tener esta configuración.

SDK Updates

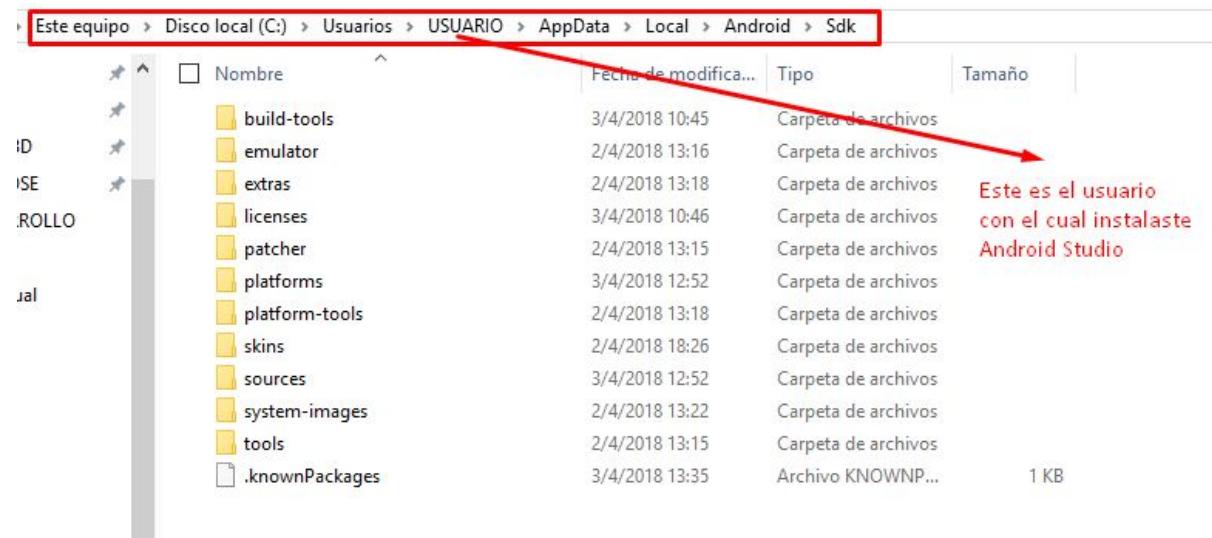


Es importante tener seleccionadas todas las opciones para que los SDK puedan estar verificando constantes actualizaciones.

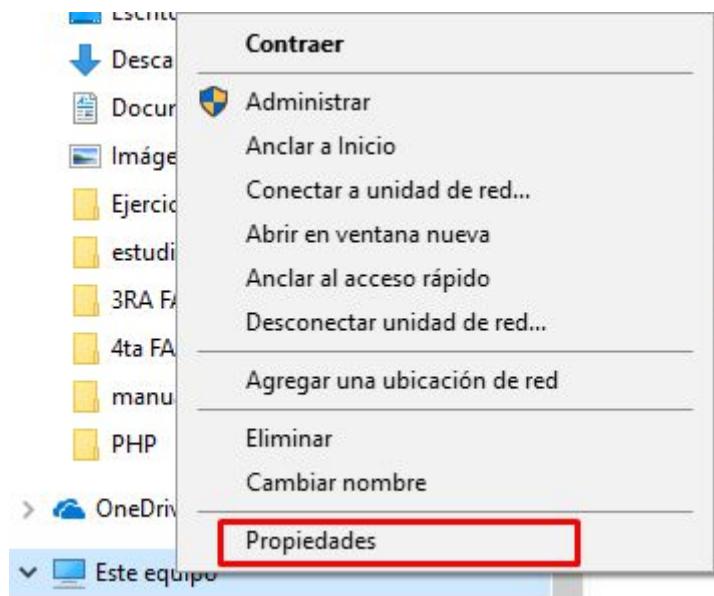




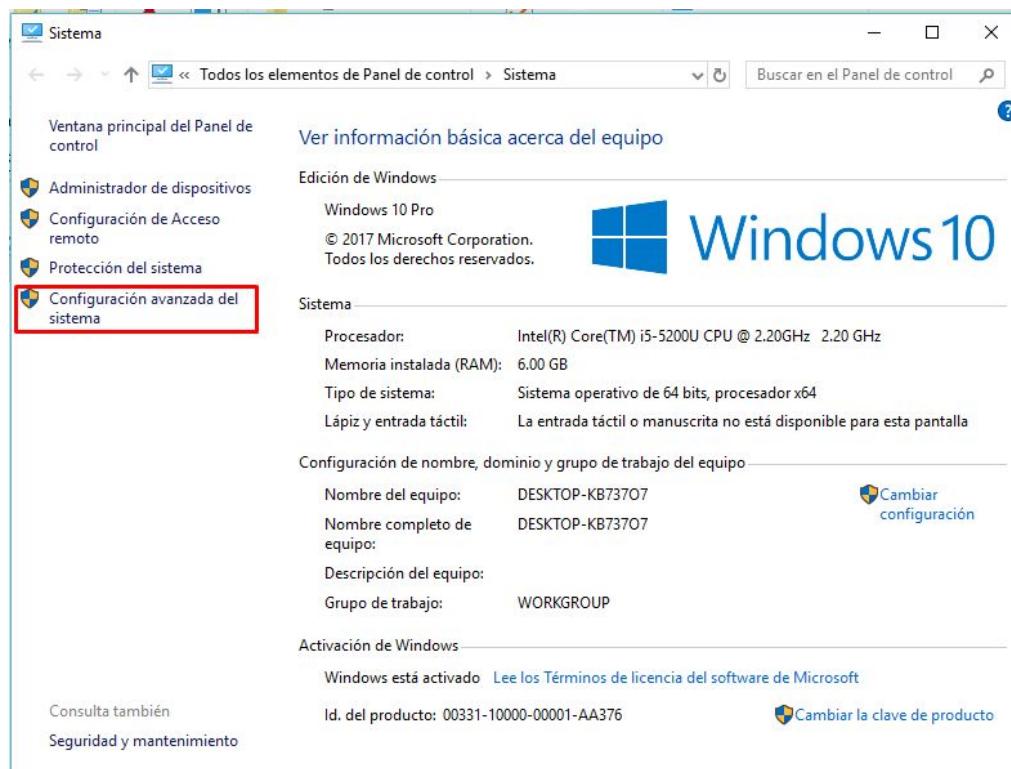
Al igual que JDK para Android también debemos generar la variable de entorno. Para esto vamos a buscar la ruta de instalación de nuestro SDK y abrir la configuración avanzada del sistema.



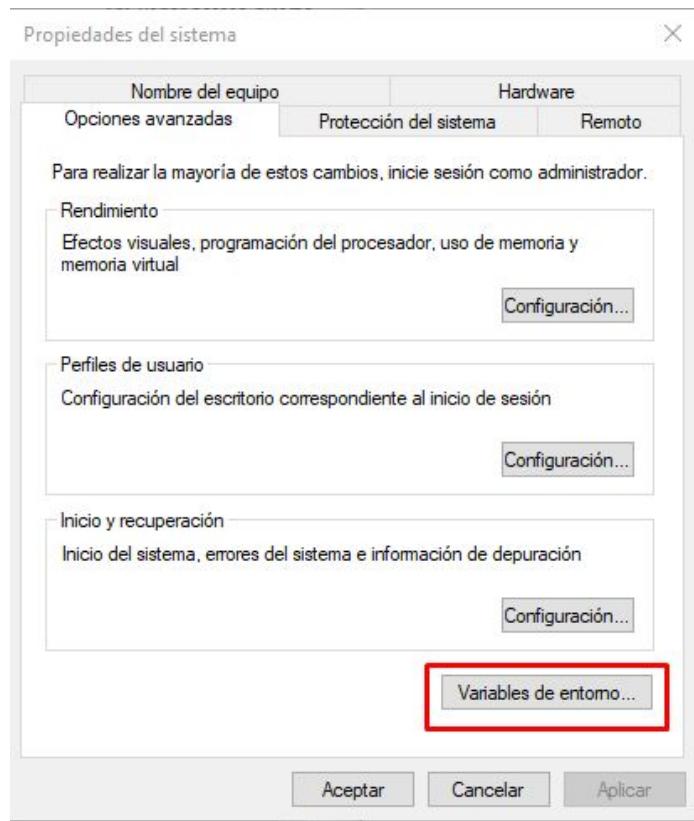
Abre un explorador de archivos y da clic derecho en **Este Equipo**. Luego seleccionamos la opción de propiedades.



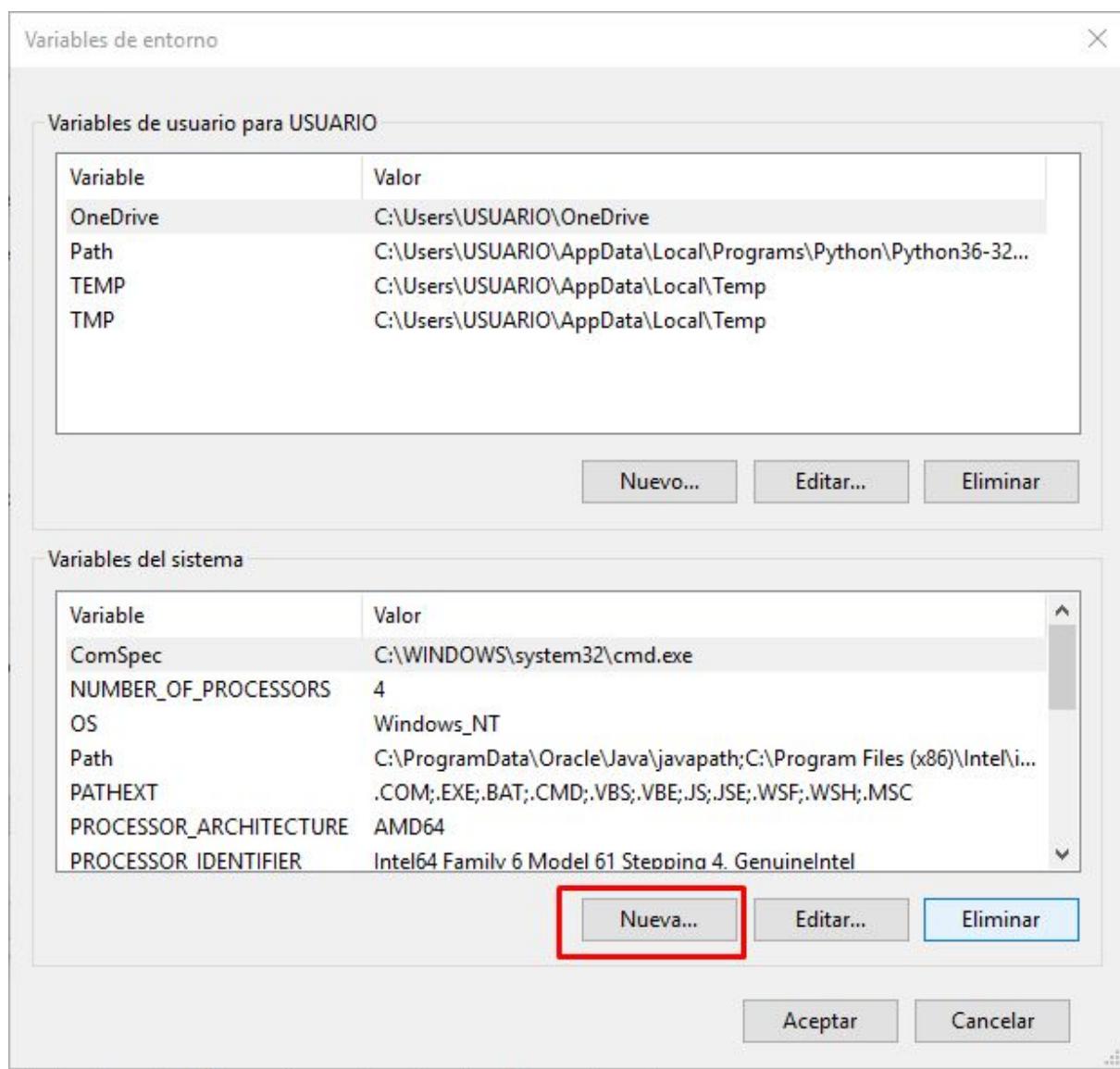
Luego en la pantalla que se nos habilita, seleccionamos **Configuración avanzada del sistema**.



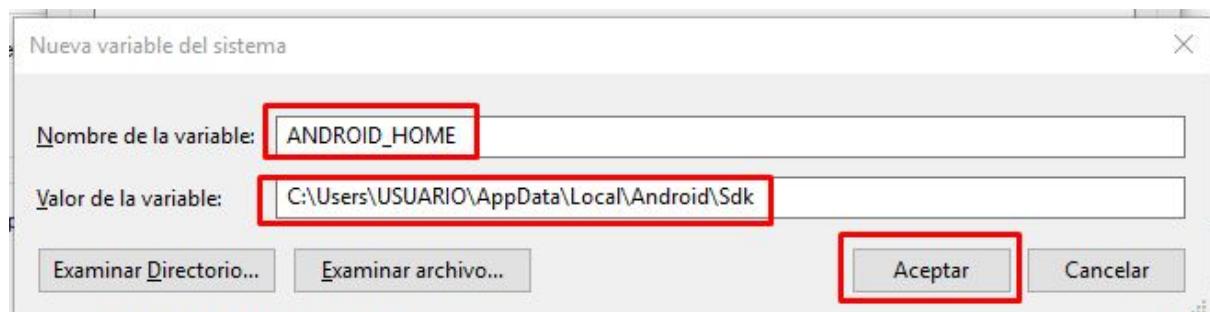
Haga clic en Variables de entorno.



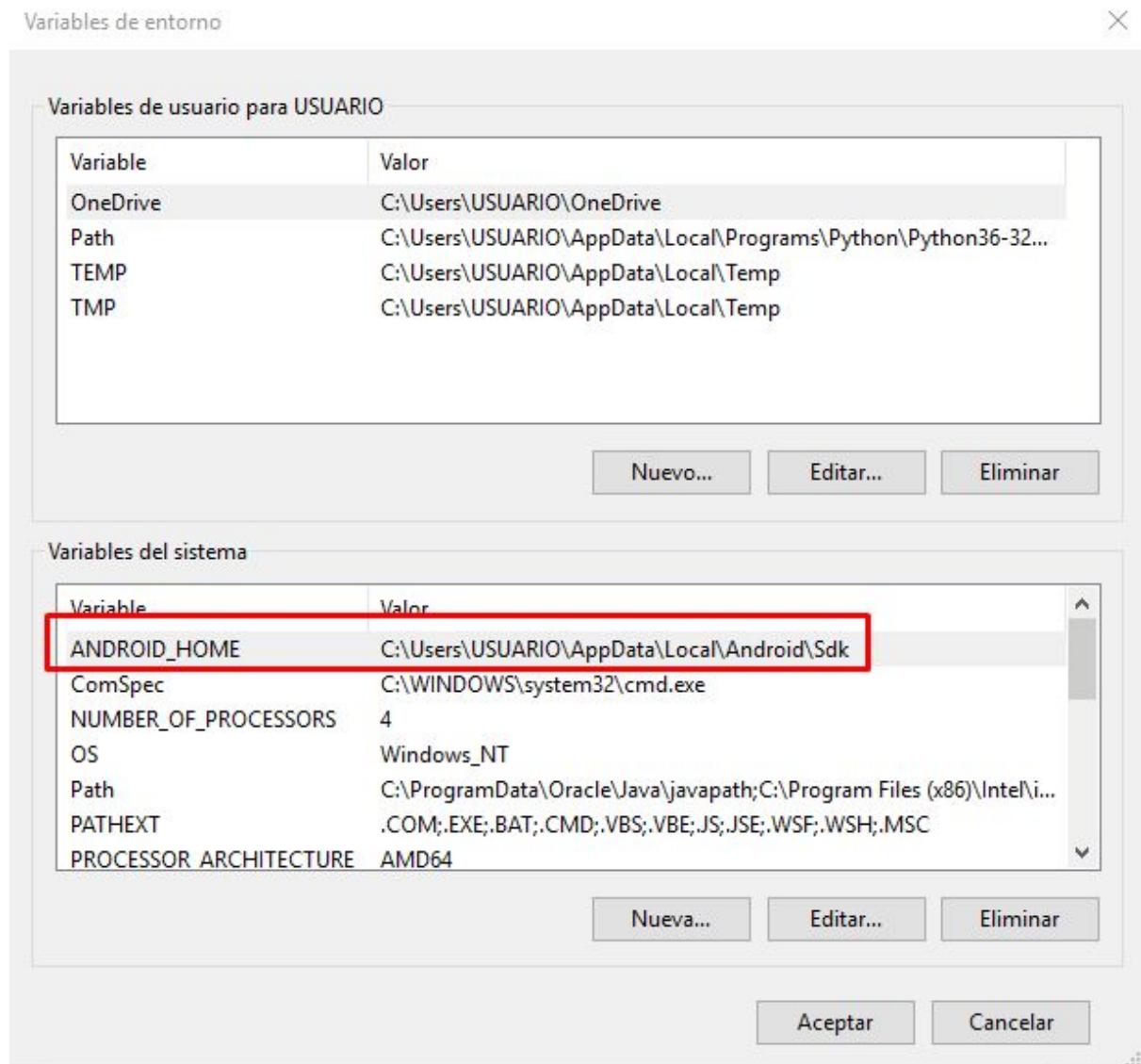
En la sección Variables del sistema de clic en Nueva..



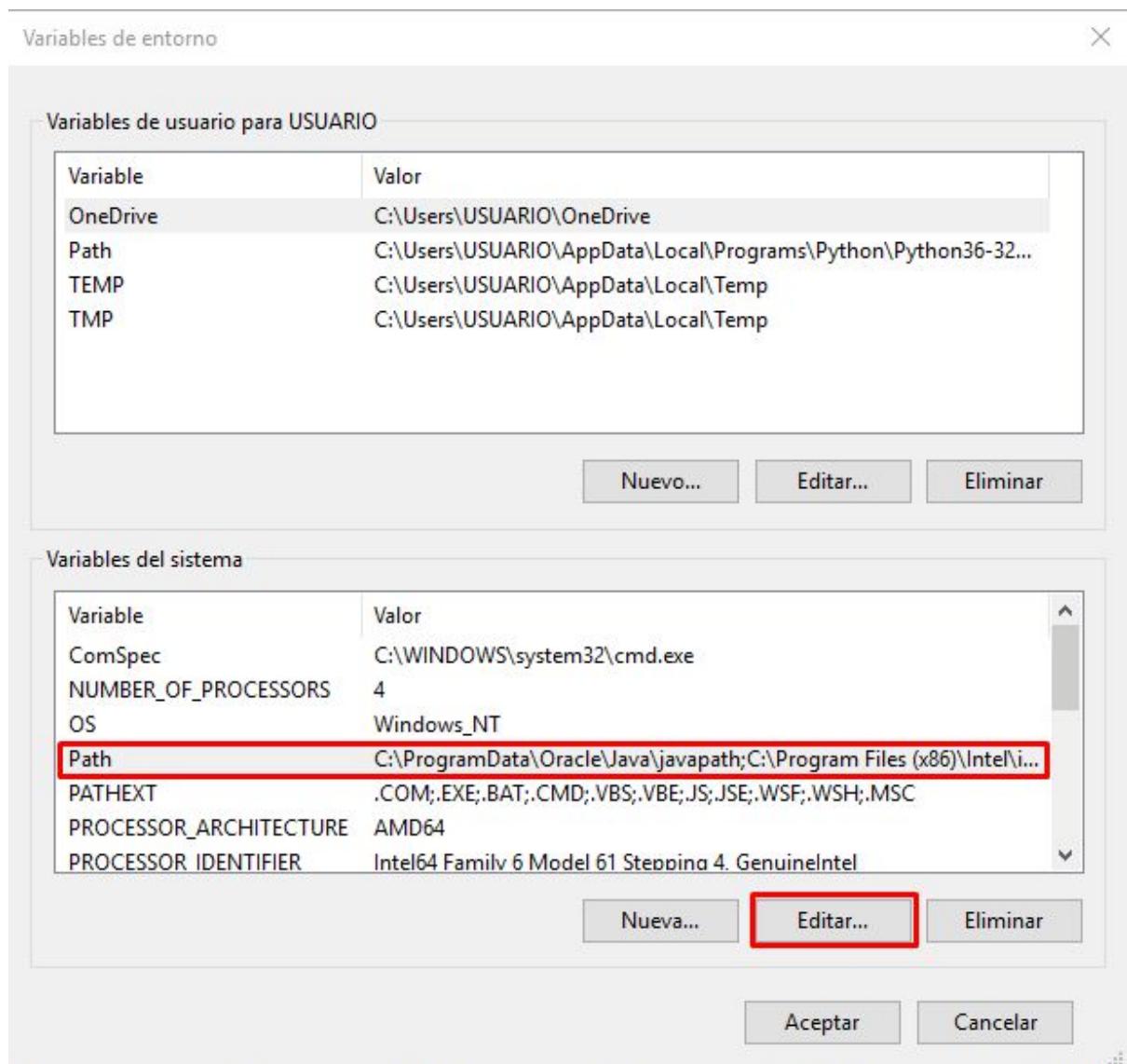
Ingresamos en el campo de Nombre de la Variable el valor de **ANDROID_HOME** que será el nombre de nuestra variable. Posteriormente seleccionamos el botón de examinar directorio y buscamos la carpeta del SDK que normalmente está instalada en **C:\Users\USUARIO\AppData\Local\Android\Sdk** y damos clic en aceptar.



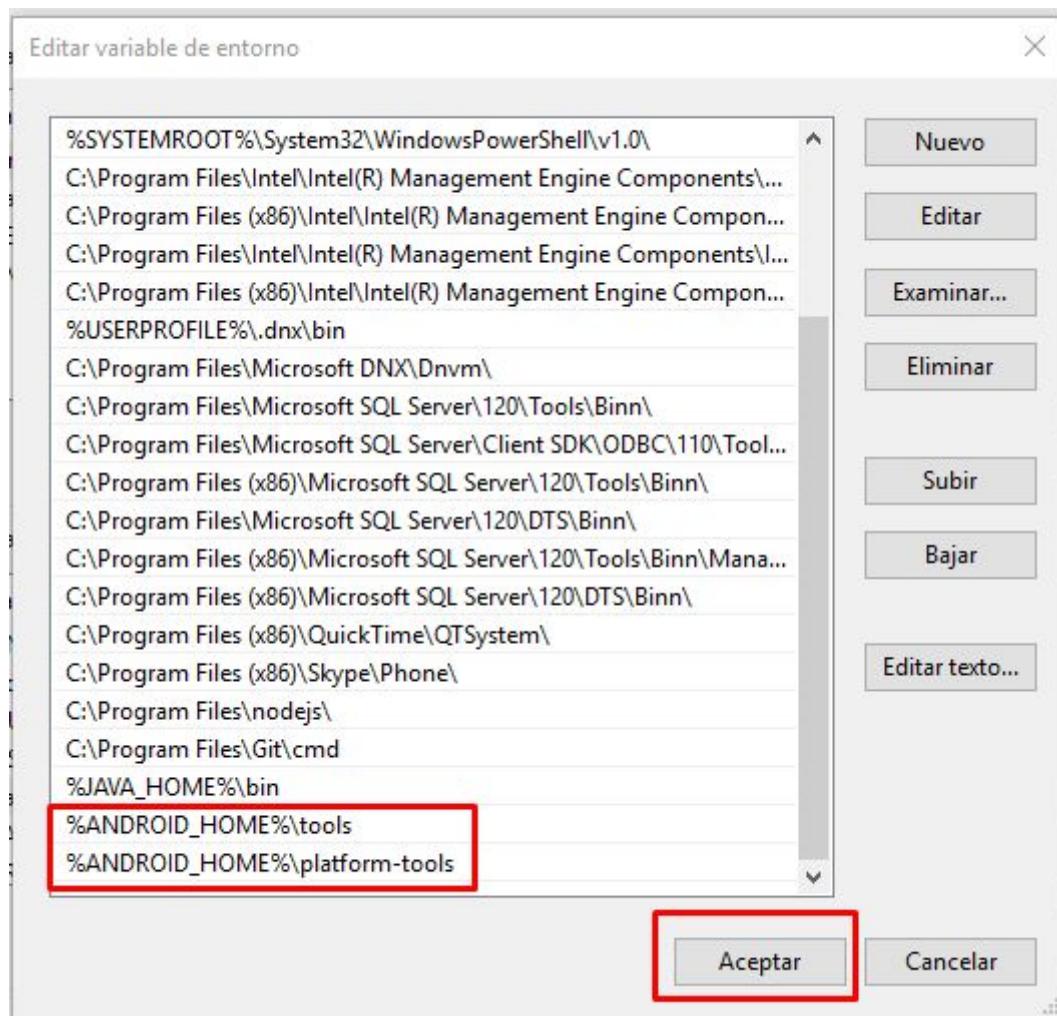
Al darle aceptar confirmamos de que nuestra variable haya quedado configurada correctamente.



De nuevo nos ubicamos en la sección de Variables del sistema y seleccionamos la variable de entorno PATH y hacemos clic en Editar



Luego de que se nos abra la ventana, damos clic en Nuevo e ingresamos dos configuraciones para los **tools** y las **platform-tools**, como se muestra a continuación.



Luego damos clic en aceptar en las ventanas que teníamos abiertas para que se apliquen los cambios a las variables del sistema que modificamos.

Variables de Sistema para macOS

En macOS y Linux también debes definir la variable de entorno llamada ANDROID_HOME y la famosa JAVA_HOME que apunte a donde instalaste el SDK y el JDK respectivamente. Puedes hacer esto agregando estas líneas a tu archivo .bash_profile que se encuentra en tu directorio home (puedes editar lo desde la terminal pulsando el siguiente comando: `open -e .bash_profile`).

NOTA: Si no tienes un archivo `.bash_profile` creado recuerda realizar estos pasos:

1. Abrir la Terminal
2. Pulsar "cd ~/" para ir al directorio de home
3. Pulsar "touch .bash_profile" para crear el archivo.
4. Pulsar "open -e .bash_profile"

De esta manera ya puedes realizar los siguientes pasos para crear las variables de entorno.

```
export ANDROID_HOME=/Users/yhoanandresgaleanourrea/Library/Android/sdk  
export PATH=$ANDROID_HOME/platform-tools:$PATH  
export PATH=$ANDROID_HOME/tools:$PATH  
export JAVA_HOME=$(/usr/libexec/java_home)
```

Reemplazando por supuesto /Users/yhoanandresgaleanourrea/Library/Android/sdk con la ruta donde instalaste tu SDK de Android (En mi caso /Users/yhoanandresgaleanourrea/Library/Android/sdk).

Para finalizar esta configuración debes pulsar el siguiente comando:

source ~/.bash_profile

Si por alguna razón tenías una versión diferente a la 1.8 del JDK, debes correr el siguiente comando:

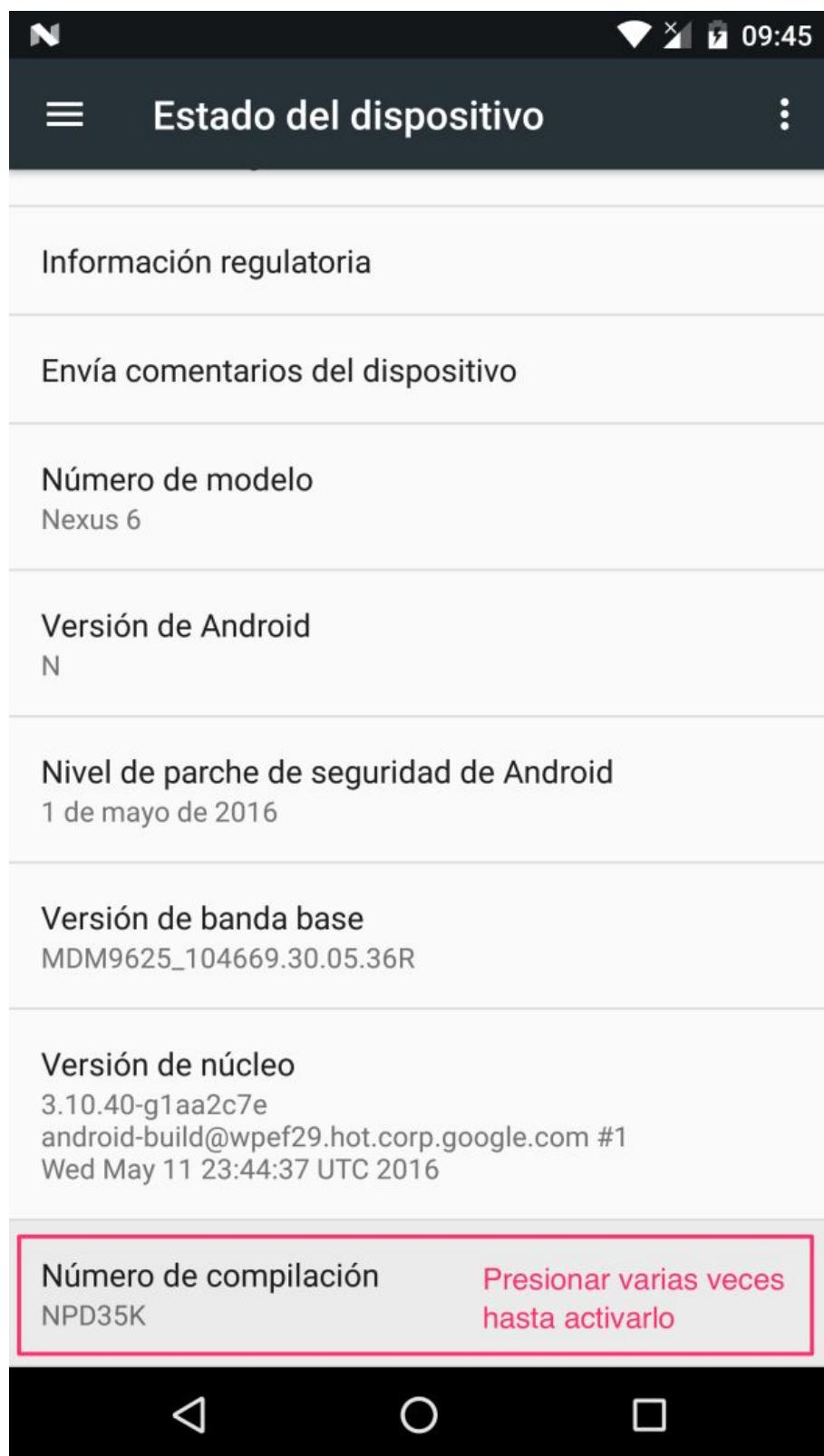
```
export JAVA_HOME=`/usr/libexec/java_home -v 1.8`
```

Depuración para Android

Al desarrollar aplicaciones en Android nos encontramos con la tarea de probar que funcione nuestra magia para saber si hemos sido unos campeones o si debemos volver a nuestro código fuente a realizar algunas mejoras. La forma más común entre los developers (lo digo en nuestro caso) es probar directamente en nuestro dispositivo, pues es más rápido, sencillo y no demora mucho si se tiene un pc tostadito (que también nos pasa). Para trabajar de esta manera es necesario habilitar el modo desarrollador y conseguir los drivers de nuestro dispositivo; De esta manera Android Studio lo reconocerá y enviará la aplicación pre-compilada para realizar las pruebas correspondientes.

Para habilitar el modo desarrollador debemos ir a configuraciones o ajustes del dispositivo y luego nos vamos a **Acerca del dispositivo** o **Estado del dispositivo**.

Damos clic en Número de compilación varias veces como se muestra en la imagen:

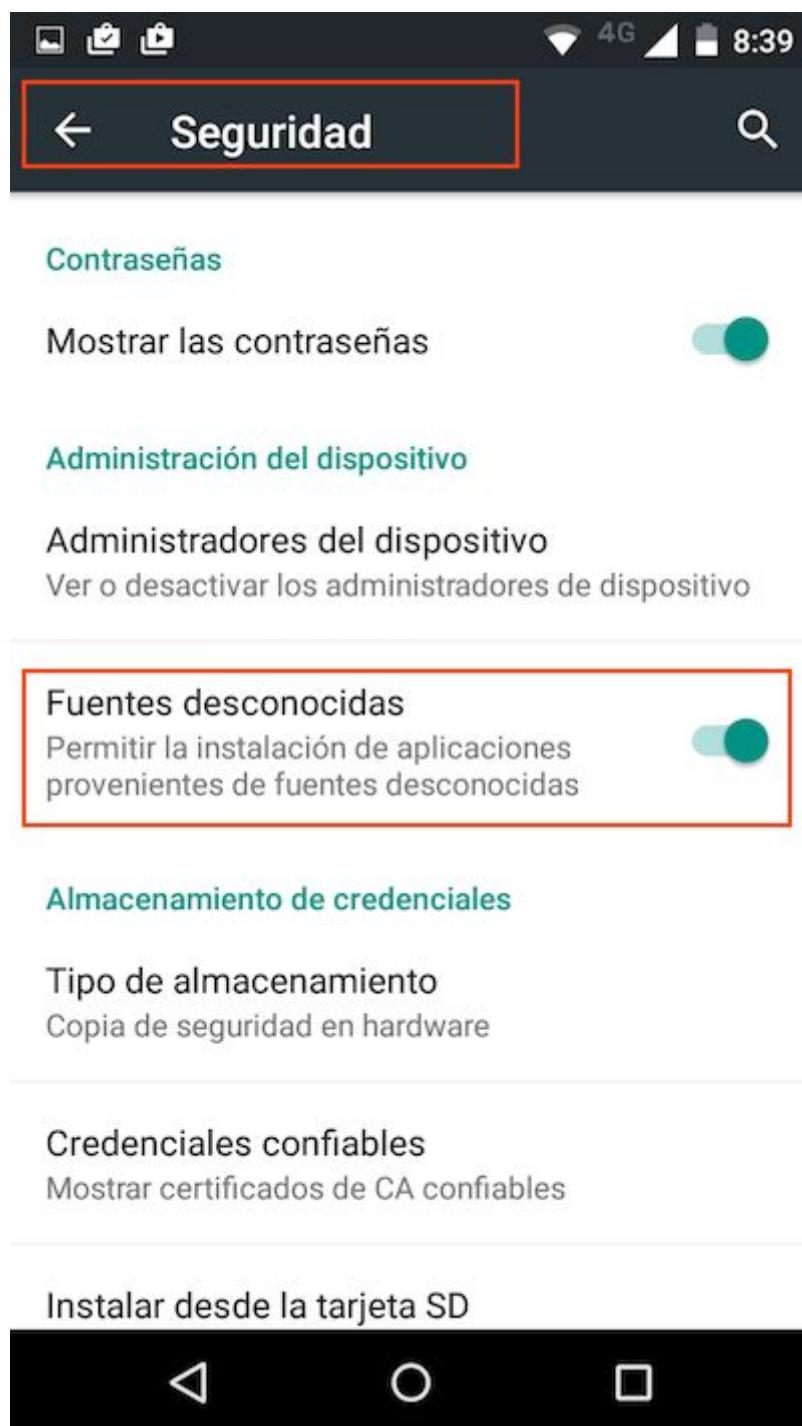


Luego en el menú de nuestro dispositivo, nos aparecerá una opción **Opciones de desarrollador**, en la cual debemos habilitar la depuración USB como se muestra a continuación:



Luego de configuradas las opciones como desarrollador, podremos habilitar la instalación de fuentes desconocidas. Esto se debe realizar debido a que cuando desarollamos aplicaciones y queremos instalarla en nuestro dispositivo no se hace

desde la app oficial (Play Store). Para esto entramos en la opción de seguridad y procedemos habilitar esta característica como se aprecia en la imagen:



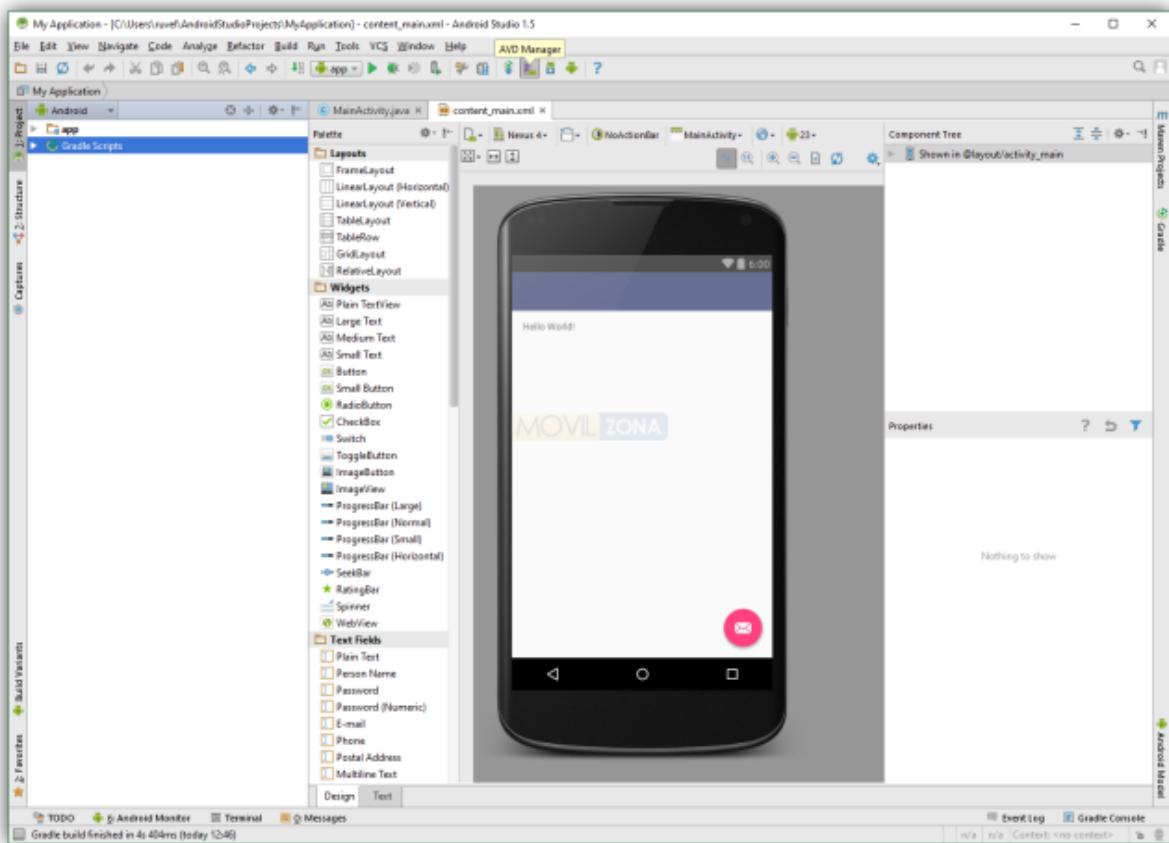
De esta manera cuando ejecutemos nuestra aplicación no tendremos problemas para visualizar y trabajar con todas las características de nuestro dispositivo y que son necesarias para nuestra app.

Otra de las formas para visualizar y depurar nuestra app, es crear un emulador para realizar pruebas en nuestro equipo de desarrollo sin necesidad de tener algo físico. Esta opción es supremamente útil cuando queremos probar una misma aplicación en diferentes versiones, dispositivos y tamaños (Pues físicos podría salirnos un poco costoso la adquisición de los dispositivos).

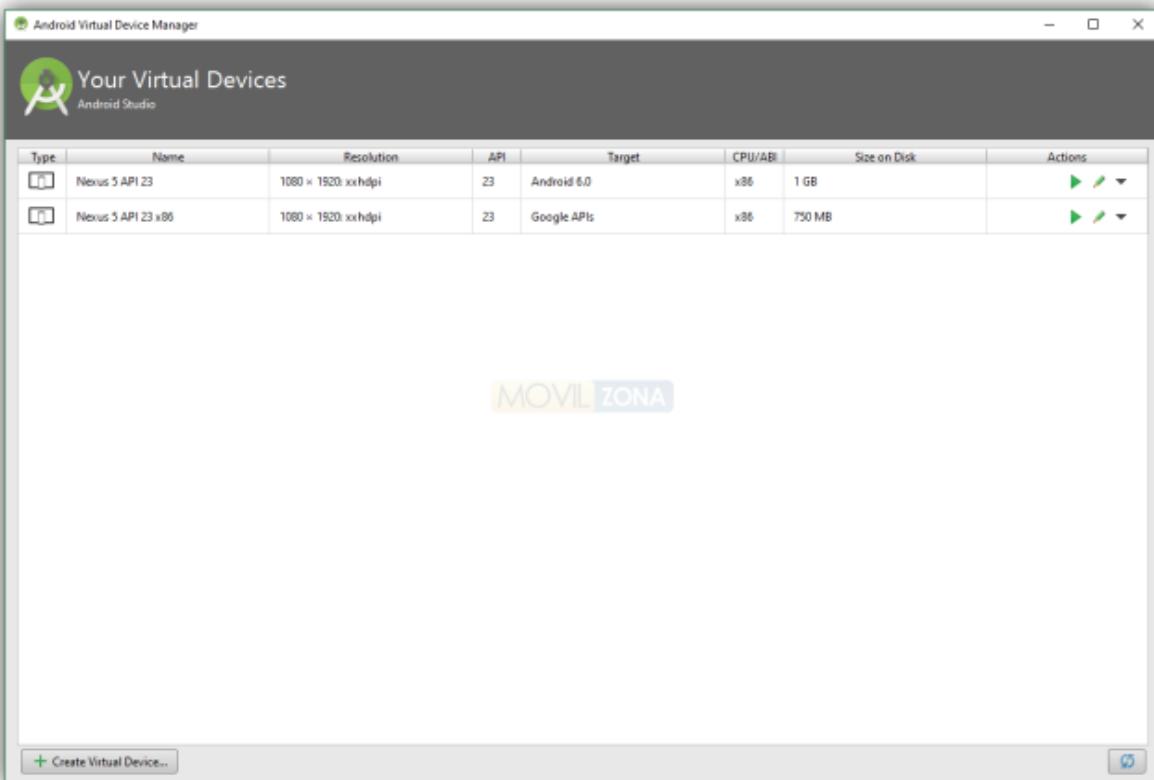
Android Studio nos permite crear y configurar diferentes máquinas virtuales, cada una de diferentes especificaciones, de manera que podamos probar nuestras aplicaciones en varias configuraciones (diferente memoria, diferente tamaño, resolución, etc) sin tener que tener decenas de dispositivos físicos a nuestro alcance. La página oficial de Android, nos provee una guía muy completa para crear un emulador. Puedes hacerlo siguiendo estas instrucciones: <https://developer.android.com/studio/run/managing-avds.html>. El emulador corre más rápido si usas una imagen “Intel x86 Atom” en lugar de una con “ARM EABI v7a”.

De igual manera, en este manual abordaremos los pasos para crear este AVD, veamos:

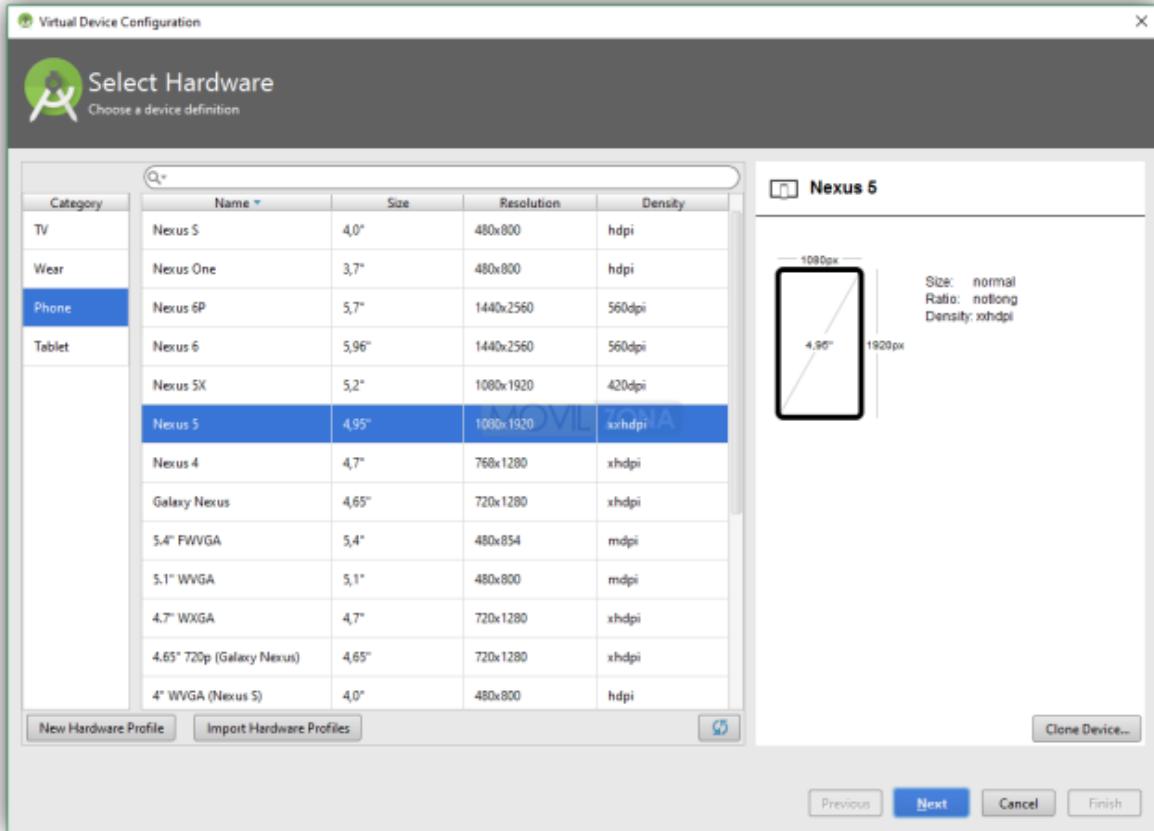
Aunque Android Studio configura automáticamente una máquina virtual, nosotros mismos podemos crear y configurar otros entornos personalizados de manera que se adapten a la situación que queremos analizar realmente. Para ello, lo primero que debemos hacer es ejecutar Android Studio en nuestro ordenador y seleccionar el botón **AVD** (Android Virtual Device).



Se nos abrirá una nueva ventana donde veremos todas las máquinas virtuales que tenemos configuradas.



En la parte inferior veremos un botón llamado “**Create Virtual Device**“. Pulsamos sobre él y se nos abrirá un sencillo asistente de configuración del hardware del dispositivo que vamos a virtualizar.

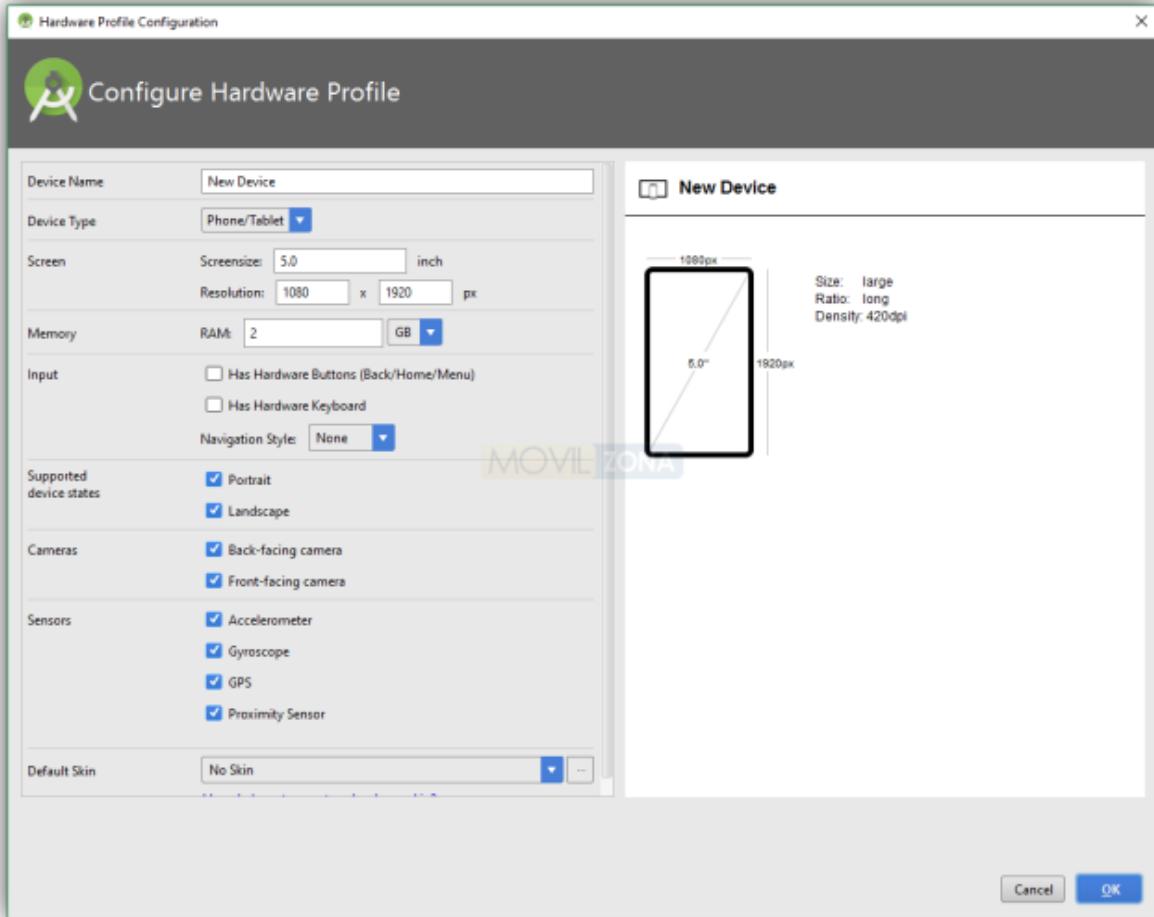


Aquí vamos a poder emular los 4 principales tipos de dispositivos que ejecutan Android:

- TV
- Wear
- Smartphone
- Tablet

En nuestro caso vamos a crear un emulador de Smartphone personalizado. Aunque nos vienen varios modelos ya previamente creados (generalmente dispositivos Nexus y otras configuraciones menos comunes), nosotros vamos a ver cómo crear un smartphone diferente a los que nos aparecen en la lista, y este será el que utilicemos en este tutorial.

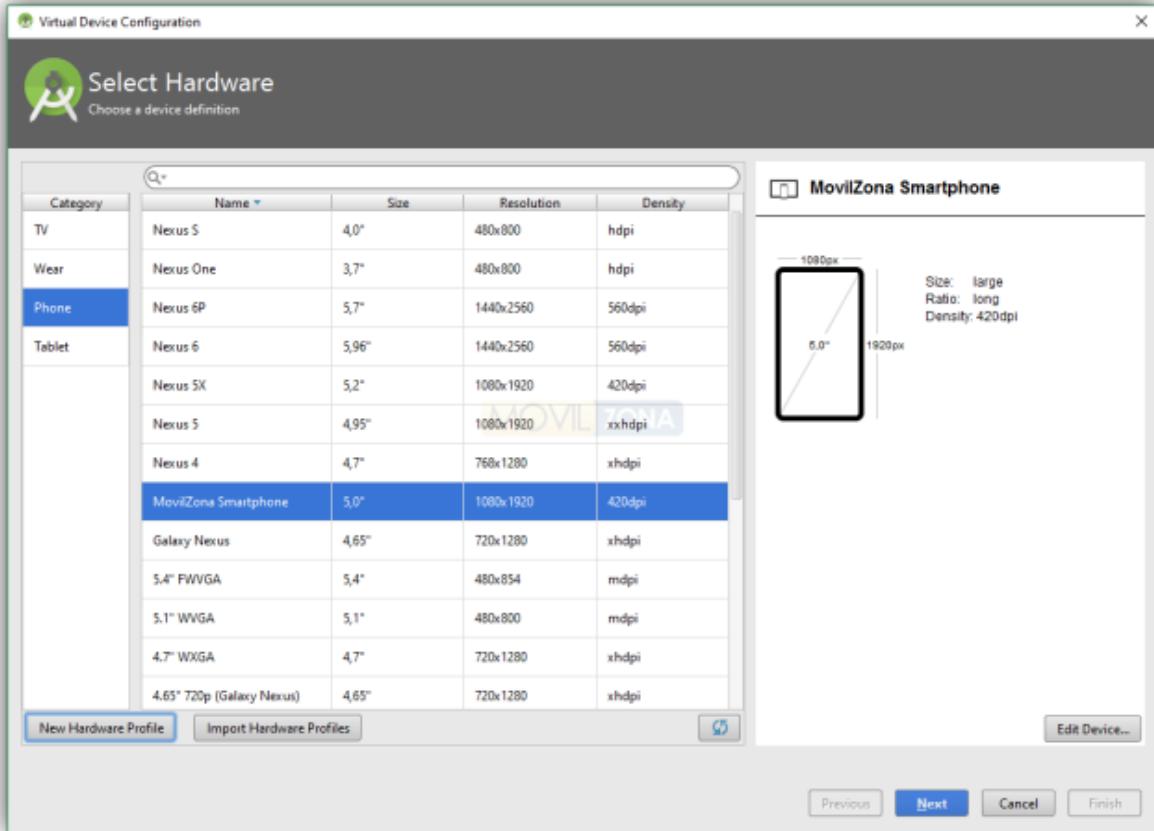
Para crear una nueva configuración de hardware debemos pulsar sobre “**New hardware Profile**” y veremos un asistente como el siguiente.



Aquí podemos configurar las características que tendrá nuestro smartphone virtual:

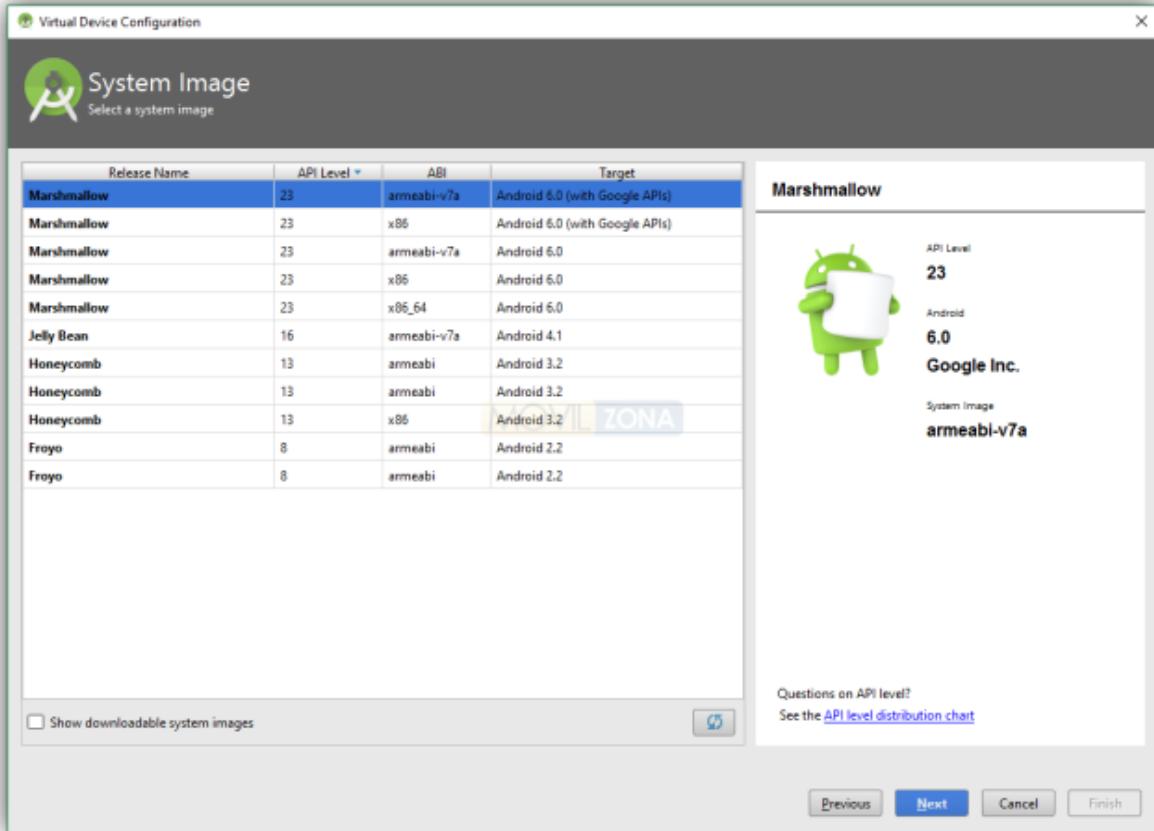
- Nombre.
- Tipo (smartphone, tablet, wear, tv).
- Tamaño y resolución de la pantalla.
- Memoria RAM.
- Botones y teclado físico.
- Tipo de pantalla (vertical u horizontal)
- Cámaras.
- Sensores.

Una vez configurado el dispositivo a nuestro gusto guardamos los cambios y nuestro nuevo móvil aparecerá en la lista del programa.



Si no queremos complicarnos también podemos elegir cargar un smartphone preconfigurado, por ejemplo, un Nexus 5.

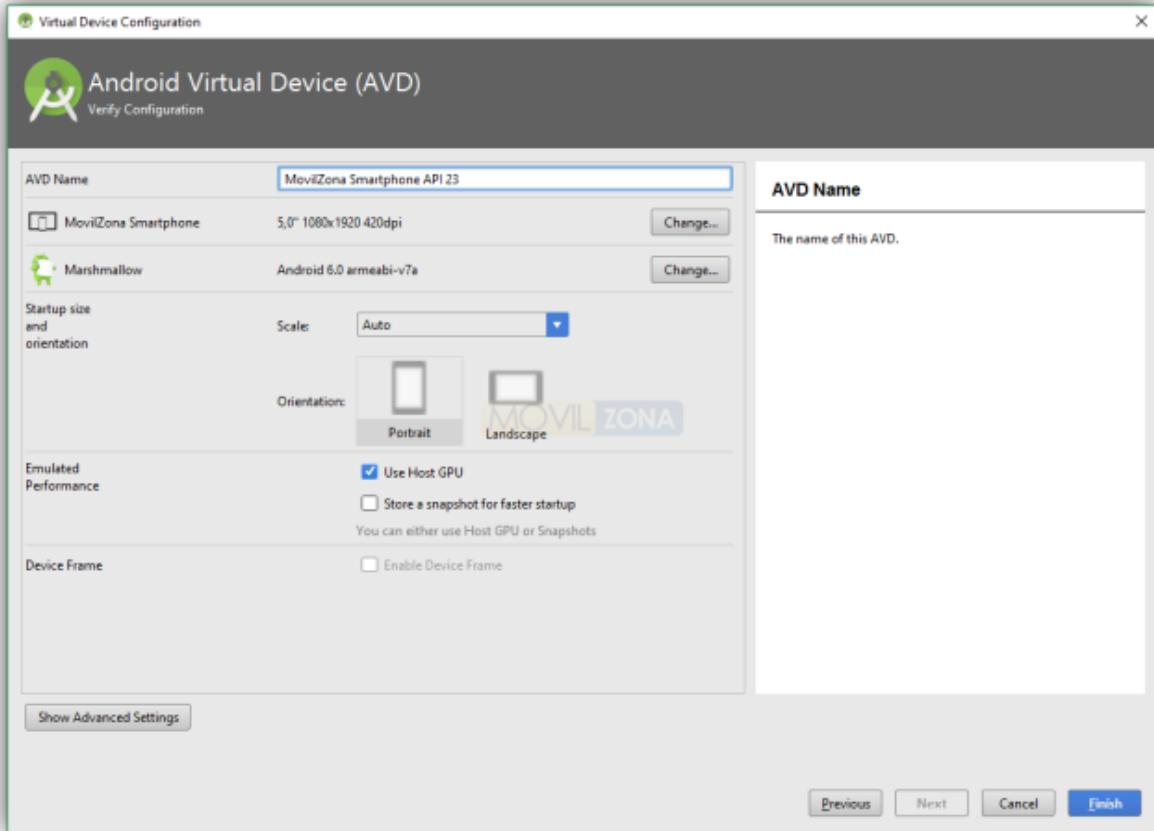
Pulsamos sobre “**Next**” y a continuación elegiremos el sistema operativo que vamos a instalar a nuestro smartphone virtual.



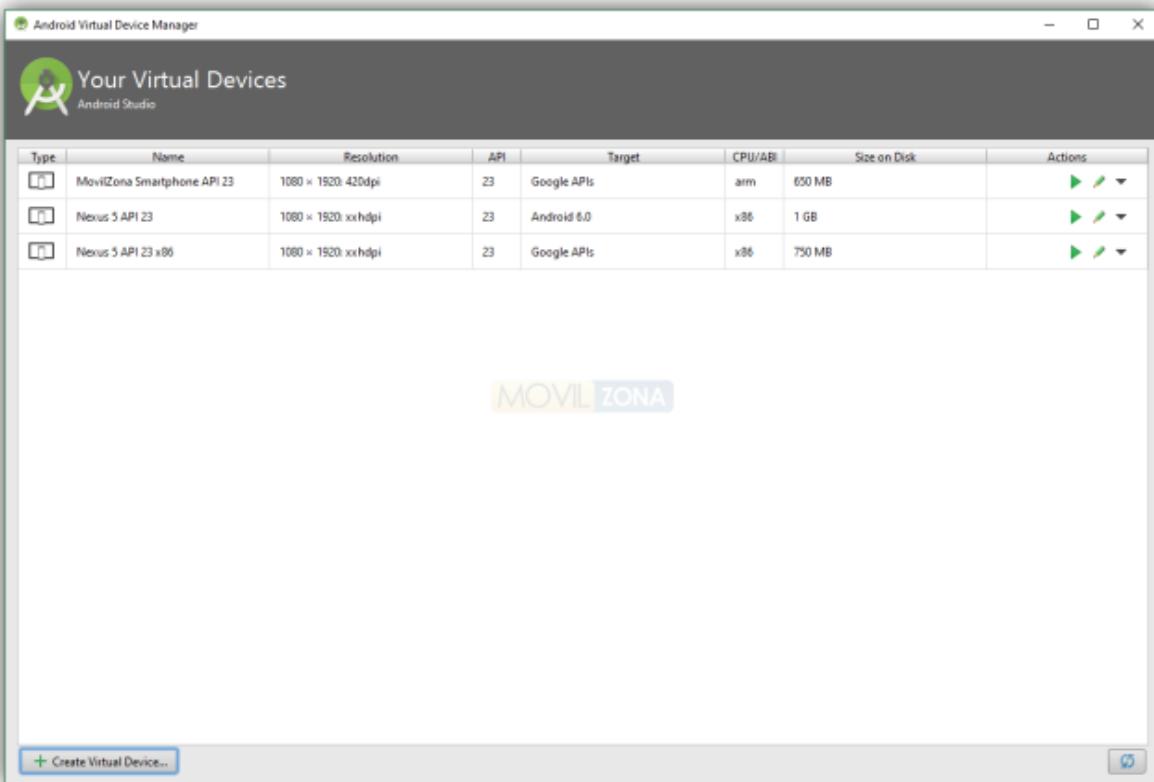
Es recomendable elegir una versión de Android con las Google Apps. De esta manera podremos hacer un mejor uso del sistema y de la máquina virtual, pudiendo incluso probar su comportamiento junto a otras aplicaciones descargadas desde la Play Store o probar la integración con Google Play Services.

En el caso de que no aparezcan demasiadas máquinas virtuales podemos marcar la opción “**Show downloadable system images**” para ver una lista más grande de las versiones de Android disponibles y descargar la que mejor se adapte a nuestro proyecto.

Seguimos con el asistente y veremos un resumen de cómo será nuestro smartphone virtual.



Si todo está correcto guardamos los cambios y finalmente nos aparecerá, como uno más, en la lista.



Para arrancar el dispositivo virtual solo tenemos que pulsar sobre el botón “Play” y esperar a que este cargue en nuestro sistema.



Ya con estas herramientas instaladas, necesitamos un editor que nos permite crear nuestras apps y el cual nos ayude un poco en nuestro desarrollo. Para esto instalaremos **Visual Studio Code** un editor de código muy potente y que ha dado mucho de qué hablar.

En primer lugar nos dirigimos a la web: <https://code.visualstudio.com/>

The screenshot shows the Visual Studio Code website. At the top, there's a navigation bar with links for Docs, Updates, Blog, Community, Extensions, and FAQ. A search icon and a 'Download' button are also present. Below the navigation, a message says 'Version 1.22 is now available! Read about the new features and fixes from March.' The main content area features a large banner with the text 'Code editing. Redefined.' and 'Free. Open source. Runs everywhere.' Below the banner, there's a 'Download for Mac' button. Underneath it, there are sections for 'macOS' (Package and Stable/Insiders), 'Windows x64' (Installer and .zip, highlighted with a red border), and 'Linux x64' (deb, rpm, tar.gz). To the right, a code editor window displays some TypeScript code. At the bottom, there are icons for Intellisense, Debugging, Built-in Git, and Extensions.

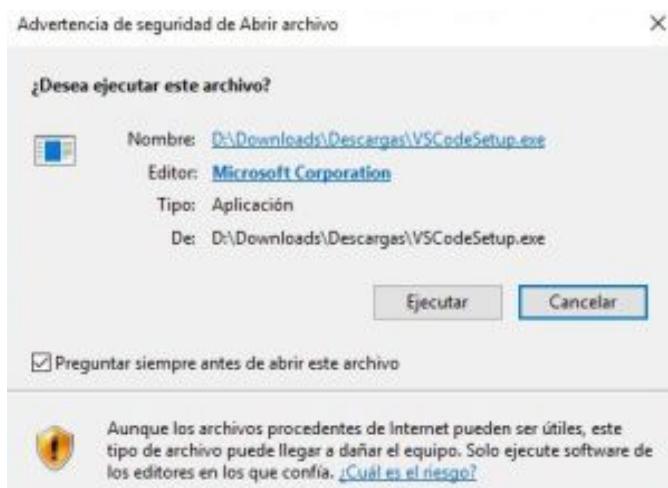
Podemos hacer clic en instalador (**installer**) de la versión estable (**Stable**)

The screenshot shows the Visual Studio Code website after a download. The top navigation bar and message are the same as the previous screenshot. The main content area has a green header box that says 'Thanks for downloading VS Code for Windows!' It includes a link to a direct download and a survey. Below this, there's a 'Getting Started' section with a brief introduction and a link to introductory videos. To the left, a sidebar lists various documentation categories like Overview, SETUP, GET STARTED, USER GUIDE, LANGUAGES, NODEJS / JAVASCRIPT, PYTHON, JAVA, EXTENSION AUTHORING, EXTENSIBILITY REFERENCE, and OTHER. On the right, there's a 'GETTING STARTED' sidebar with links to VS Code in Action, Top Extensions, First Steps, Keyboard Shortcuts, Downloads, Privacy, and several community links: Subscribe, Ask questions, Follow @code, Request features, Report issues, and Watch videos. At the bottom, there's a code editor window showing a snippet of JavaScript code related to a server setup.

Se descarga el fichero "VSCodeSetup.exe" en nuestro computador



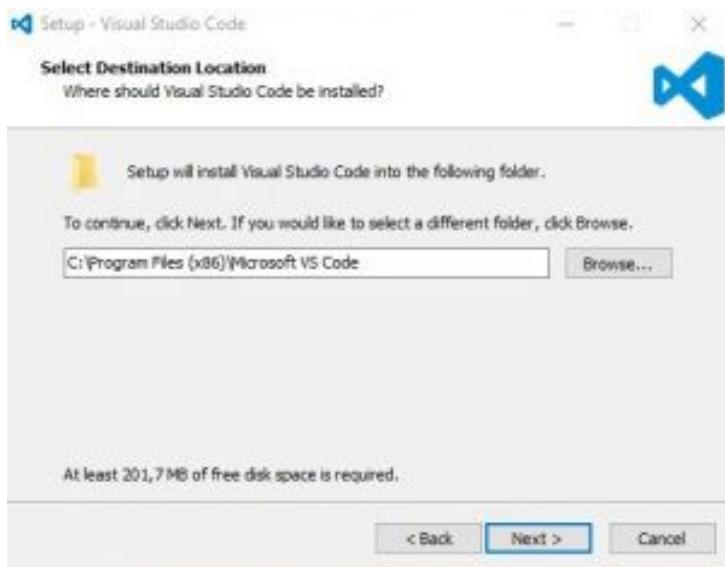
Hacemos doble clic sobre él y procedemos a la instalación, es posible que nos pida permiso para ejecutar este archivo, para ello debemos seleccionar Ejecutar en dicha ventana



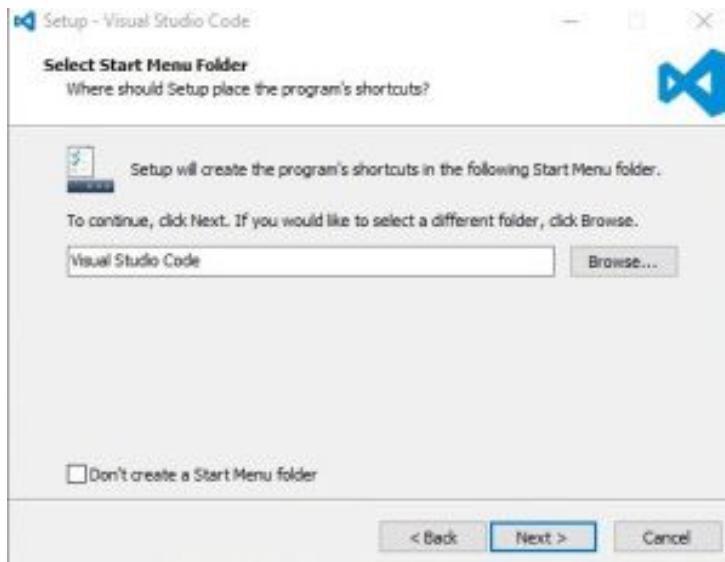
Debemos seguir los pasos que nos van apareciendo en las siguientes ventanas, comenzando por la pantalla de bienvenida:



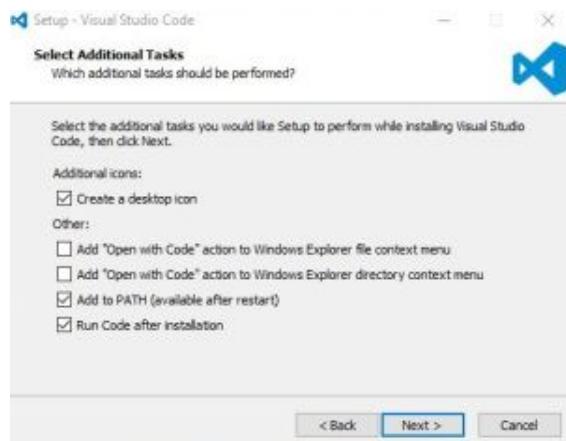
Nos aparece una ventana para seleccionar la carpeta de instalación del programa, por defecto, “C:\Program Files(x86)\Microsoft VS Code”. Podemos dejar esta o elegir una diferente y hacer clic en “Next”.



Debemos seleccionar la carpeta del Menú de Inicio en la que se instalará Visual Studio Code.



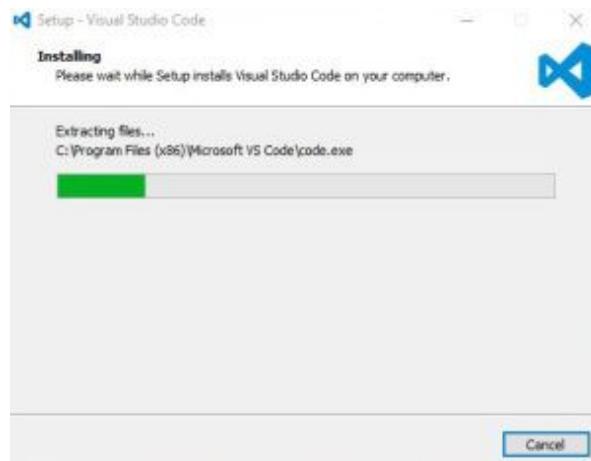
Podemos seleccionar tareas adicionales, como crear un ícono en el Escritorio, ejecutar Code después de la instalación.



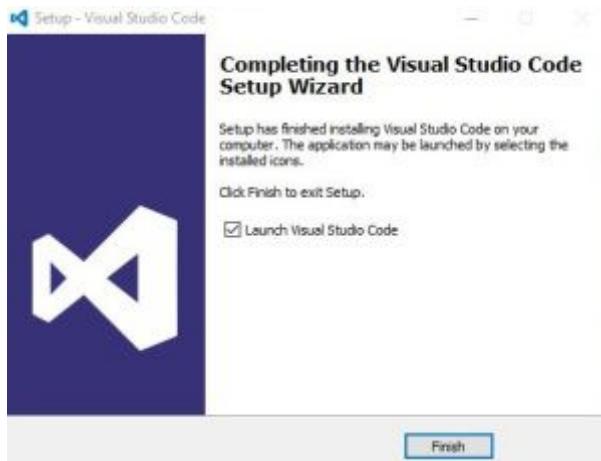
Ya estamos preparados para iniciar la instalación. Debemos hacer clic en “Install” para comenzar la misma.



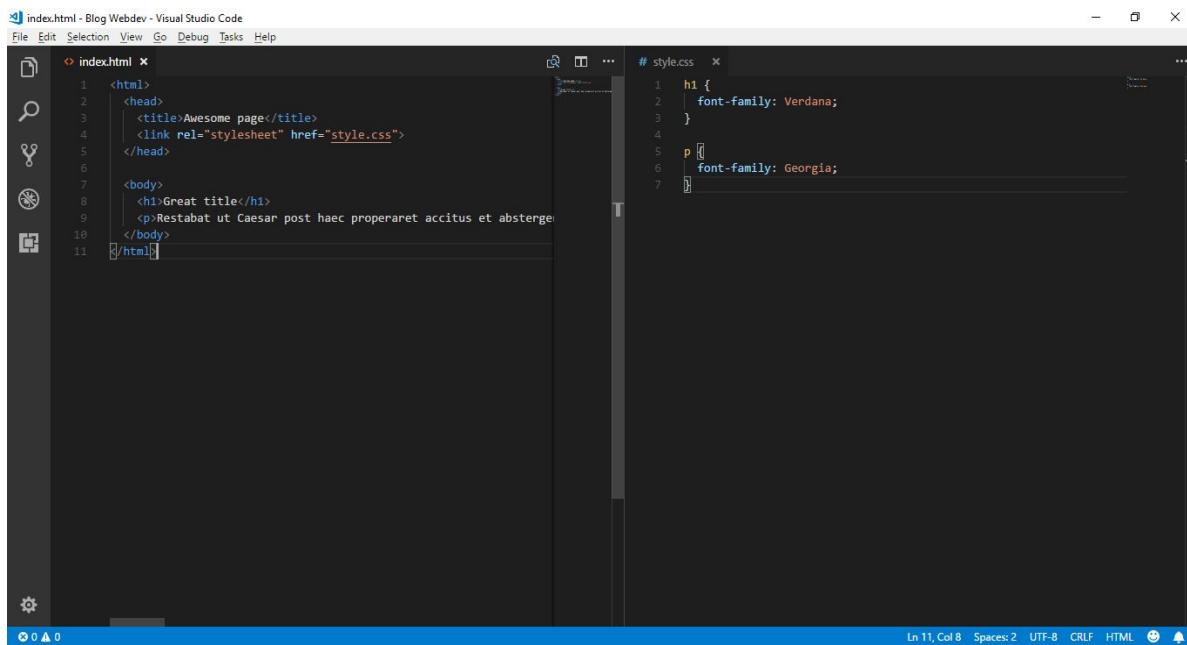
Se instala dicha aplicación en el sistema, la instalación llevará unos pocos minutos.



Una vez finalizada la instalación nos aparecerá una ventana indicando que se ha completado y permitiendo ejecutar la misma cuando hagamos clic en Finalizar (Finish).



Ya tenemos nuestra aplicación instalada:



Instalación de Ionic

Con Node.js ya instalado, podemos comenzar a trabajar con nuestros proyectos de Ionic y Cordova sin ningún problema, pero antes debemos instalar esto por medio del npm. Para esto vamos a abrir la consola de Windows y vamos a ejecutar los siguientes comandos:

Ionic

```
C:\> npm
C:\Users\Yhoan-Galeano>npm install -g ionic@latest
[.....] \ retreiving metadata.  artemis C:\Users\Yhoan-Galeano\AppData\Roaming\npm
```

Cordova

```
C:\>
C:\> npm
C:\Users\Yhoan-Galeano>npm install -g cordova
[.....] \ normalizeTree.  SHIM install loadCurrentTree
```

De esta manera ya tenemos nuestro entorno para trabajar creando apps híbridas con Ionic.

```
C:\>ionic
○ ○
IONIC: CLI 3.18.3
Usage:
$ ionic [command] [args] [-h|--help] [--verbose] [-q|--quiet] [-n|--interactive] [-c|--config] [options]
Global Commands:
config <subcommand> ..... Manage CLI and project config values (subcommands: get, set)
docs ..... Open the Ionic documentation website
info ..... Print system/environment info
login ..... Login with your Ionic ID
signup ..... Create an Ionic account
start ..... Create a new project
telemetry ..... Opt in and out of telemetry
Project Commands:
You are not in a project directory.

C:\>cordova
Synopsis
cordova command [options]
Global Commands
create ..... Create a project
help ..... Get help for a command
telemetry ..... Turn telemetry collection on or off
config ..... Set, get, delete, edit, and list global cordova options
Unrecognized Commands
```

Comenzando con IONIC

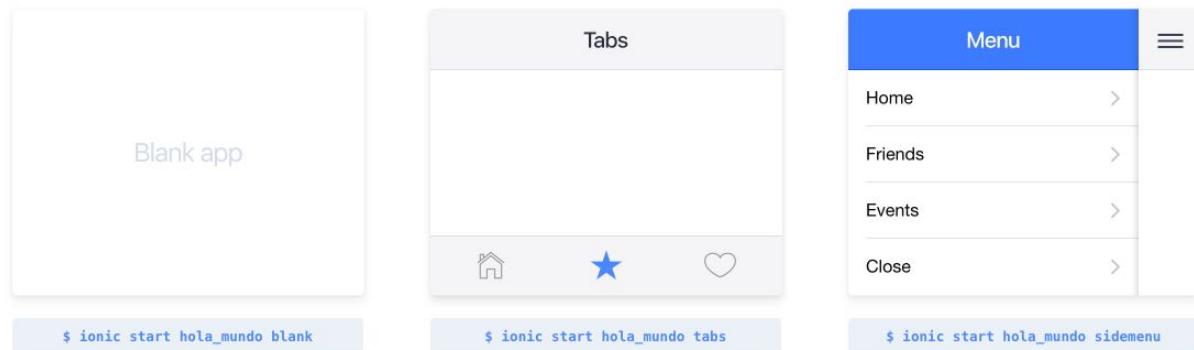
Ahora si podemos iniciar nuestro primer proyecto en IONIC, para esto pulsaremos el siguiente comando:

ionic start myApp tabs

```
C:\>
C:\>ionic start myApp tabs
Creating directory .\myApp - done!
[INFO] Fetching app base <https://github.com/ionic-team/ionic2-app-base/archive/master.tar.gz>
\ Downloading - done!
[INFO] Fetching starter template tabs <https://github.com/ionic-team/ionic2-starter-tabs/archive/master.tar.gz>
\ Downloading - done!
Updating package.json with app details - done!
Creating configuration file ionic.config.json - done!
[INFO] Installing dependencies may take several minutes!
> npm install
\ Running command
```

Donde **myApp** es el nombre que le damos a nuestro proyecto y **tabs** es el template con el cual trabajaremos en nuestra app.

Ionic permite trabajar con algunos templates para iniciar una aplicación de cero, estos son:

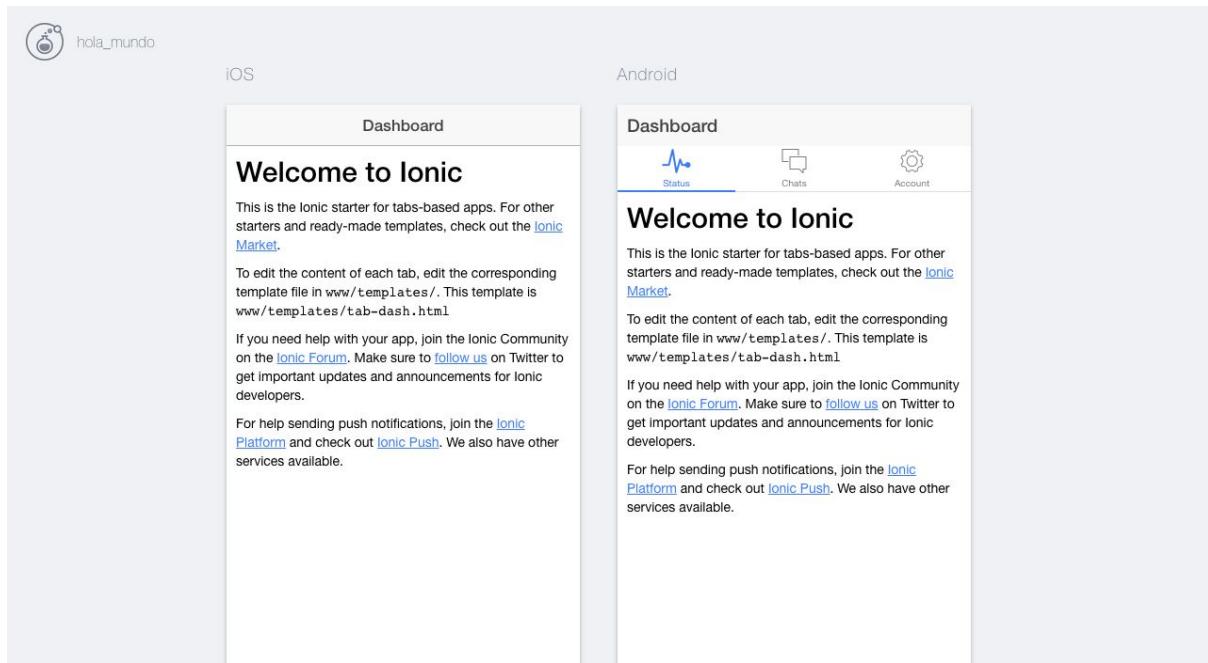


El siguiente paso es poner a correr nuestra app en el navegador. Para eso debemos navegar hasta la ruta de **myApp** con el comando **cd myApp** en la consola de windows y pulsar los siguientes comandos:

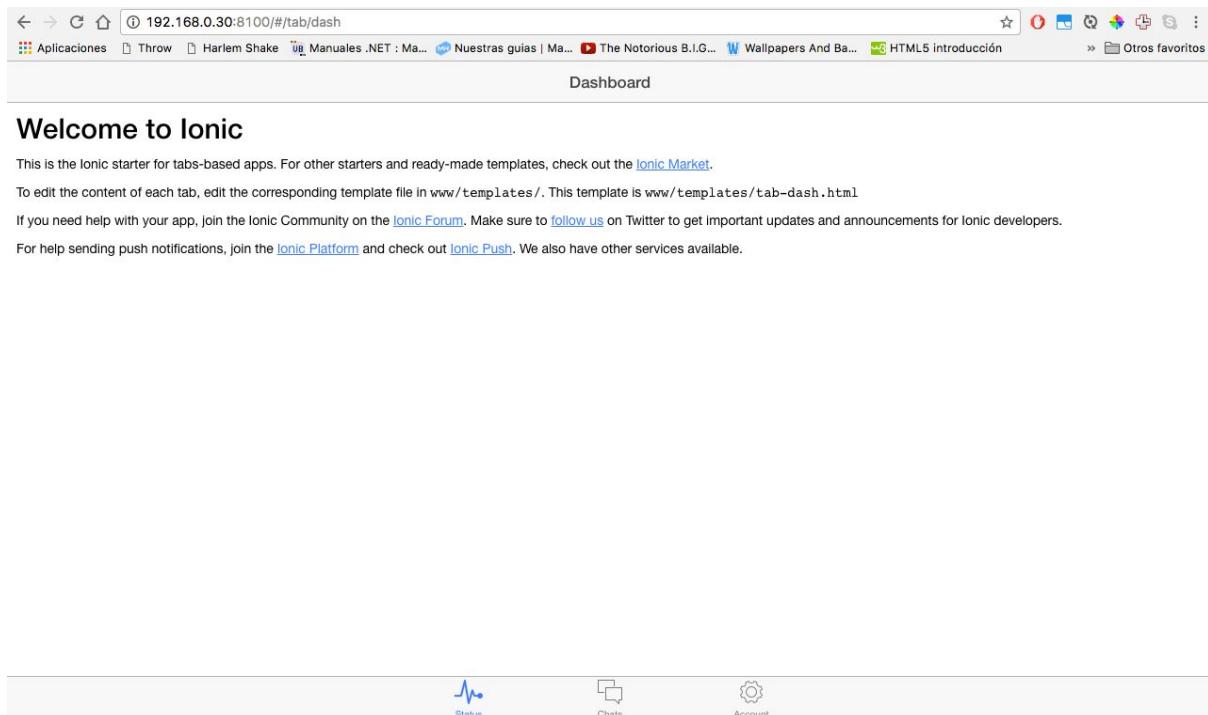
```
C:\MyProjects\myApp>ionic serve --lab
[INFO] Starting app-scripts server: --address 0.0.0.0 --port 8100
  - Ctrl+C to cancel
[00:02:34] watch started ...
[00:02:34] build dev started ...
[00:02:34] clean started ...
[00:02:34] clean finished in 5 ms
[00:02:34] copy started ...
[00:02:34] transpile started ...
[00:02:51] transpile finished in 17.27 s
[00:02:51] preprocess started ...
[00:02:51] deeplinks started ...
[00:02:51] deeplinks finished in 39 ms
[00:02:51] preprocess finished in 44 ms
[00:02:51] webpack started ...
[00:02:54] copy finished in 20.31 s
[00:03:30] webpack finished in 38.80 s
[00:03:30] sass started ...
[00:03:49] sass finished in 19.14 s
[00:03:49] postprocess started ...
[00:03:49] postprocess finished in 248 ms
[00:03:49] lint started ...
[00:03:50] build dev finished in 75.83 s
[00:03:55] dev server running: http://localhost:8100/
[INFO] Development server running!
  Local: http://localhost:8100
  External: http://192.168.43.152:8100
[00:06:11] lint finished in 141.20 s
```

De esta manera la consola nos informa porque puerto y porque url se esta corriendo nuestra app y además la ejecuta en el navegador.

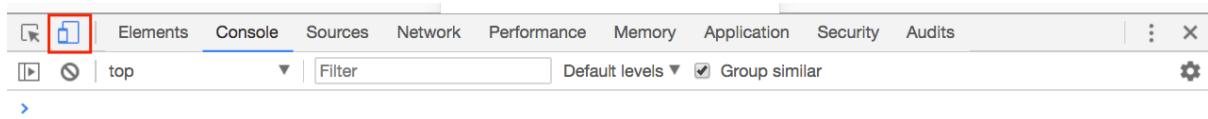
Para verla de una manera más bonita y sin necesidad de abrir el inspeccionar elemento y ver cómo se vería en los diferentes sistemas operativos, utilizamos el comando **ionic serve --lab**. Entonces nuestra app se vería así en el navegador:



Si solo ejecutamos el comando **ionic serve** (sin el --lab) se visualizará nuestra app en la web de la siguiente manera:



Para verlo como si fuese un dispositivo debemos hacer uso de las herramientas de desarrollador de google chrome como se muestra a continuación.



En la parte superior podríamos seleccionar el dispositivo de visualización:



Así terminamos nuestra pequeña guía de configuración para trabajar con ionic y apps híbridas.

Espero les haya servido.

Bibliografía

Instalación de Node.js	https://01luisrene.com/instalar-nodejs-en-windows/
Instalación de Git	https://medium.com/laboratoria-how-to/c%C3%B3mo-instalar-git-368c78187b51
Instalación de ionic	http://www.programmingworldtech.com/2017/09/set-up-mobile-app-developmen.html