

Manual Técnico

Ürban services



Versión: 1.0

Elaborado Por

Alexander Gomez Enriquez y Hugo Enrique Cecenes Morales

Universidad del Noreste

Implementación de Sistemas

Contenido

MATERIAL NECESARIO	3
INSTALACION DEL SERVIDOR	3
CONFIGURACION DE LA BASE DE DATOS.....	8
URBAN WEB.....	14
REGISTRO DE USUARIOS.....	14
HTML DEL REGISTRO DE USUARIOS	14
JS DEL REGISTRO DE USUARIOS	14
PHP DEL REGISTRO DE USUARIOS	15
ACCESO A USUARIOS	16
HTML DEL ACCESO A USUARIOS.....	16
JS DEL ACCESO A USUARIOS	16
PHP DEL ACCESO A USUARIOS	17
CONEXIÓNUNIVERSAL.....	17
TRAZADOR DERUTAS	18
HTML DE TRAZADOR DERUTAS	18
JAVA DE TRAZADOR DERUTAS	18
INSTALACION DE LOS COMPONENTES EN ANDROID.....	23
INSTALACION DE JAVA JDK.....	24
Instalación de Android Studio	30
Elaboración de la Aplicación Android	37
Elementos de la Aplicacion.....	39
Inicio de Sesion	40
Registro	42
Menu.....	44

MATERIAL NECESARIO

Una PC con cualquiera de los siguientes sistemas operativos:

- Windows 2000
- Windows XP
- Windows Vista
- Windows 7
- Windows 8

A partir de la versión 2.1e.

También se necesitan conocimientos básicos en lenguaje html, css, php, MySQL, y Java Script (tanto en Web y Android).

- Instalador de WampServer.
- Instalador de Android Studio.
- Editor de Código (Se recomienda uno universal: Notepad++, SimpleText, etc).

INSTALACION DEL SERVIDOR

Para implementar el sistema del sitio WEB y la App de Urban es necesario configurar principalmente la Base de datos la cual fue elaborada en lenguaje MySQL y administrada mediante PHP, se utilizar un servidor local para ejecutar los archivos PHP y administrarla desde los mismo, se utilizó un programa externo llamado WampServer el cual es un WebService gratuito que instala en el LocalHost (Servidor local) los servicios de PHP, Apache y MySQL que es lo que necesita el sistema tanto Web como App Android, a continuación mostramos la instalación de WampServer.

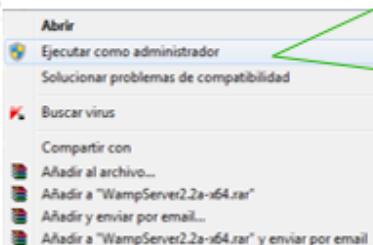
Accedemos al sitio Web oficial de WampServer: <http://www.wampserver.com/en/>



Se descargar un archivo de instalación.



WampServer2.2a-x64



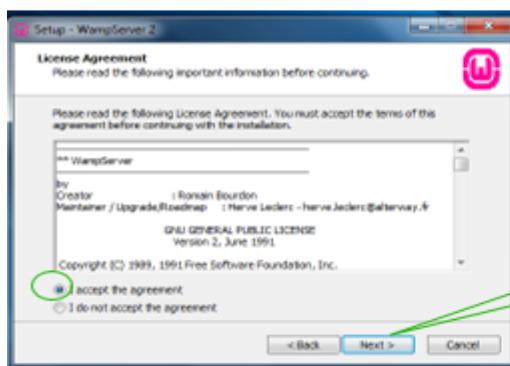
Logo y nombre del archivo de la aplicación
Con el botón derecho del ratón elige “ejecutar como Administrador” en lugar de hacer doble clic



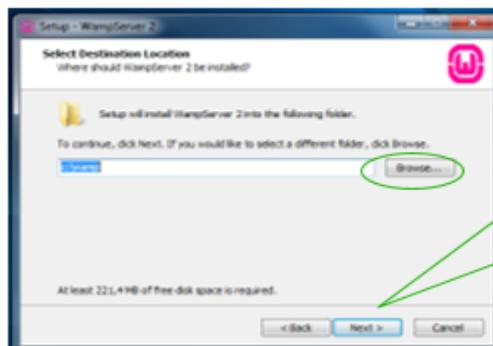
Aparecerá el programa de instalación.

A la izquierda te indica las versiones de Apache, MySQL, PHP, etc.. que se van a instalar

Cierra los programas que tengas abiertos y haz clic en **Next**

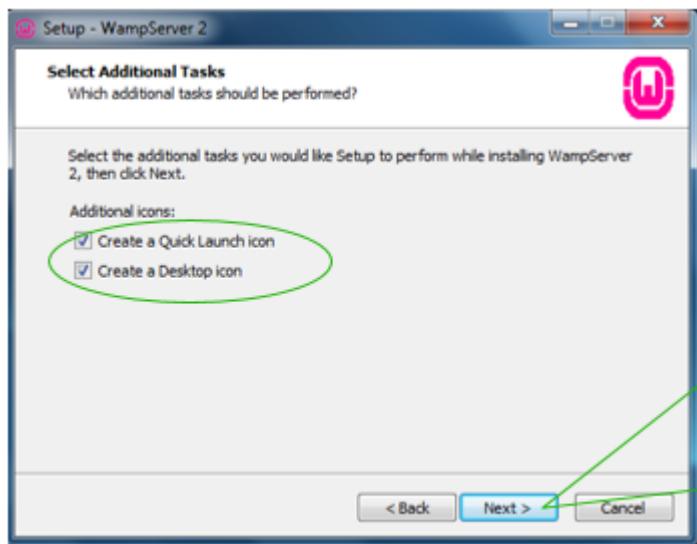


Aceptar el acuerdo de licencia y clic en **Next**



¿Dónde quieres instalar WampServer?
Se recomienda en c: y la carpeta donde se instalarán todos los archivos se llamará “wamp”

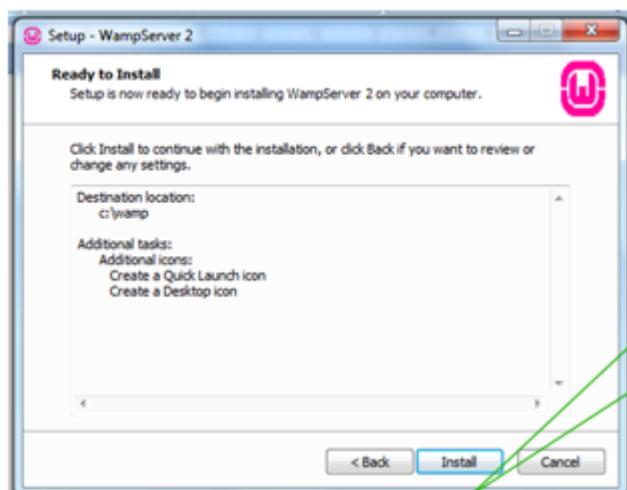
Puedes elegir otra carpeta si lo deseas
Clic en **Next**



Si validas las 2 opciones te creará un ícono de acceso directo en el escritorio y otro ícono en la barra de acceso rápido

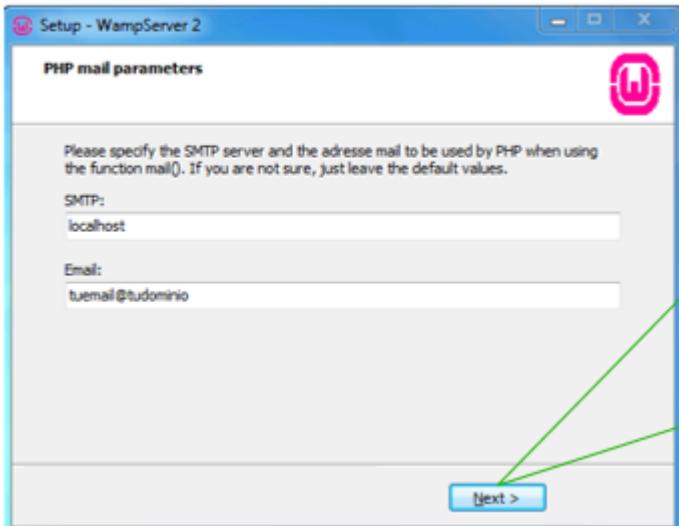
Si tienes Windows 7 es posible que no tengas la barra de acceso rápido, por lo que te recomiendo marques "Create a desktop icon" para crear el acceso en el escritorio

Después clic en **Next**

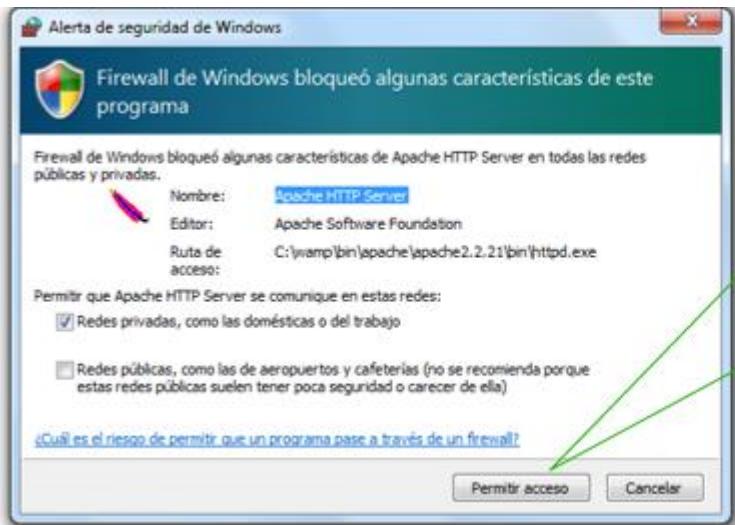


Clic en **Install** para instalar el programa, aparecerá otra pantalla con una barra de estado, espera unos minutos..

Durante el proceso de instalación te pedirá que selecciones el **navegador** que quieres utilizar por defecto. Por defecto WampServer utilizará el navegador **Internet Explorer**, si deseas utilizar otro debes buscar el ejecutable del navegador que quieras usar y pulsa el botón – **Abrir** – para seleccionarlo.

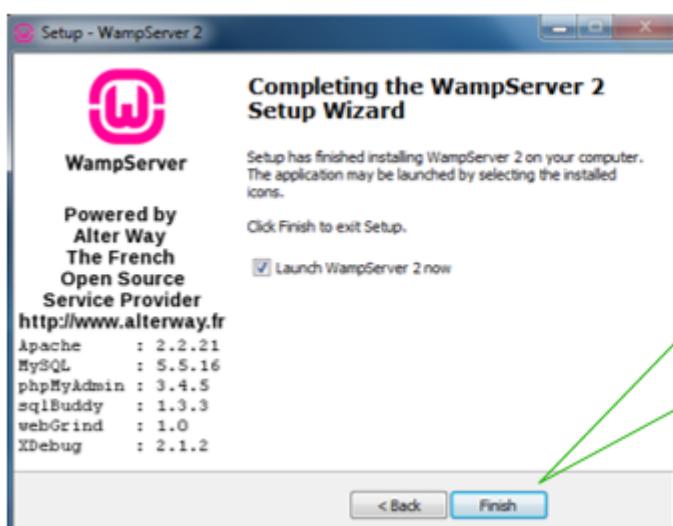


Si tienes un servidor de correo (servidor SMTP) introduce el nombre del servidor en el campo SMTP y una cuenta de correo en el campo Email, esto será usado por la función mail() de PHP como remitente para correos de salida. Si no estás seguro de tener un servidor de correo instalado, deja los valores que aparecen por defecto y pulsa el botón Next



Permite el acceso de Apache a través del Firewall de windows para redes privadas.

Si no lo haces el Firewall de windows bloqueará el programa y no funcionará

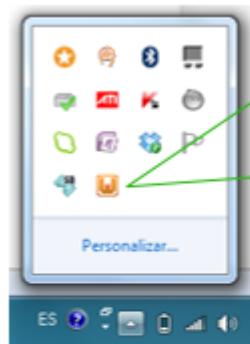


La última pantalla, clic en finish

¡ya está hecho!

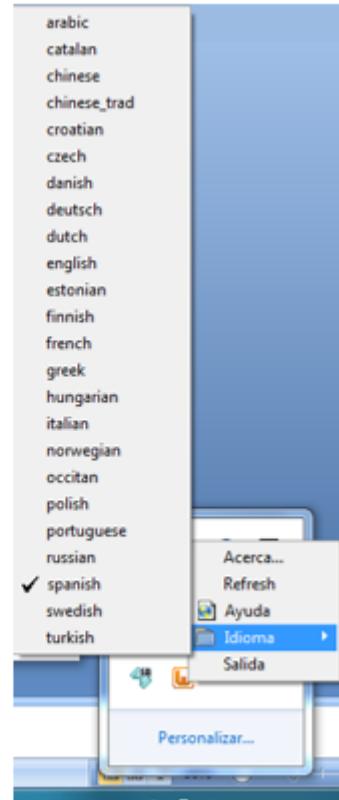
¿Ha sido fácil verdad?

Ahora vamos a ponerlo en marcha...



Localiza el Icono de WampServer en la bandeja del sistema en la barra inferior derecha

Selecciona el idioma con el botón derecho del ratón para seleccionar el idioma “Spanish”



Panel de administración de WampServer

Para acceder a él sólo tienes que pulsar en el ícono del programa con el botón izquierdo del ratón

1º - El primer paso es pulsar en Encender y comenzarán a correr todos los servicios de Apache, PHP y MySQL.

2º Instala o crea tus proyectos en la carpeta C:/wamp/www

Acceso al panel localhost:

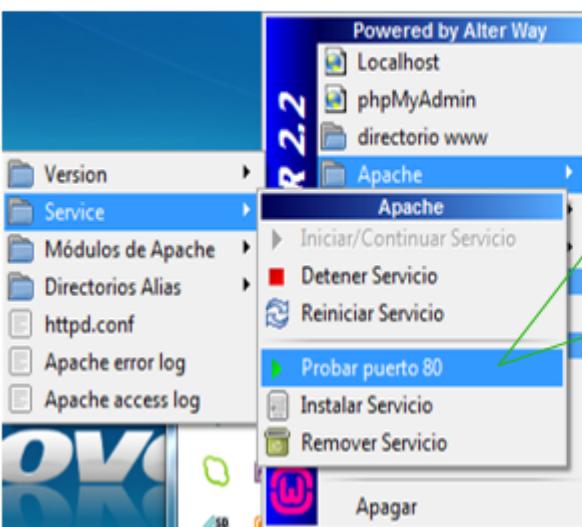
- Podrás ver todas las versiones y extensiones instaladas
- Acceder en "Tools" directamente a phpMyAdmin
- Acceder a los proyectos de la carpeta www directamente

Acceso directo a phpMyAdmin

Acceso directo a la carpeta www mediante el explorador de windows



Con WampServer podemos configurar cada servicio. Entra en el menú de cada programa para configurarlo si tienes conocimientos avanzados.

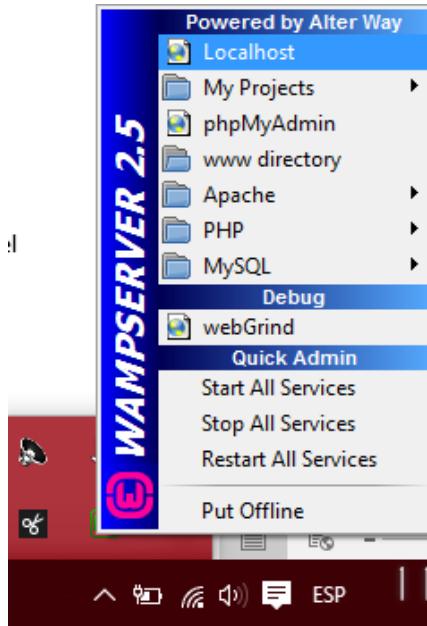


Importante: Comprueba el puerto 80 para que no esté en uso por otra aplicación
Si tienes una aplicación funcionando en el puerto 80, ciérrala, cierra WampServer y vuelve a abrirlo. Si es necesario reinicia los servicios

¡Ya tienes tu servidor local!

CONFIGURACION DE LA BASE DE DATOS

Después del proceso de instalación de WampServer entramos al panel del Servidor, el cual se entra con clic en el icono y elegimos la opción de LocalHost.



No abrirá el navegador que tengamos como pre terminado y nos mostrará el panel de opciones del servidor, accedemos al link de PHPMyAmin.

Server Configuration

Apache Version : 2.4.9 - [Documentation](#)

PHP Version : 5.5.12 - [Documentation](#)

Server Software: Apache/2.4.9 (Win64) PHP/5.5.12

Loaded Extensions :

- apache2handler
- bcmath
- bz2
- calendar
- com_dotnet
- Core
- ctype
- curl
- date
- ereg
- exif
- fileinfo
- filter
- gd
- gettext
- gmp
- hash
- imap
- json
- libxml
- mbstring
- mcrypt
- mhash
- mysql
- mysqli
- odbc
- openssl
- pcntl
- PDO
- pdo_mysql
- Phar
- Reflection
- session
- shmop
- soap
- sockets
- SPL
- sqlite3
- standard
- tokenizer
- wddx
- xml
- xmlreader
- xmlwriter
- xsl
- zip

MySQL Version : 5.6.17 - [Documentation](#)

Tools

[phpinfo\(\)](#) [phpmyadmin](#)

Your Projects

- android-test
- Proyecto
- UrbanWEB
- _notes

Your Aliases

- [phpmyadmin](#)
- [phpsysinfo](#)
- [sqlbuddy](#)
- [webgrind](#)

[WampServer](#) [Donate](#) [Alter Way](#)

Nos direccionara al panel de phpmyadmin, la cual es la manera más fácil de administrar e insertar la base de datos que se almacenara en el servidor, en la imagen se muestra un explorador que muestra muchas opciones la cual procederemos a crear una Nueva base de datos llamada “Urban”.

Bases de datos

Crear base de datos

Urban Cotejamiento Crear

Nota: Activar aquí las estadísticas de la base de datos podría causar tráfico pesado entre el servidor web y el servidor MySQL.

Base de datos	Cotejamiento
android_test_db	latin1_swedish_ci
information_schema	utf8_general_ci
myapp	latin1_swedish_ci
mysql	latin1_swedish_ci
performance_schema	utf8_general_ci
test	latin1_swedish_ci
urban	latin1_swedish_ci
Total: 7	latin1_swedish_ci

Marcar todos Para los elementos que están marcados: Eliminar

Activar las estadísticas

Creamos una nueva tabla con sus respectivas columnas:

Base de datos: urbann

Tabla: Usuarios

Acción

1 tabla Número de filas

Vista de impresión Diccionario de datos

Crear tabla

Nombre: Usuarios Número de columnas: 5

Continuar

Rellenamos con los siguientes campos, estos campos son los datos del usuario, ya que estén llenados los campos presionamos guardar, cabe mencionar que hay dos maneras de crear una tabla en la base de datos, una es visual por medio del asistente de phpMyAdmin y otra importando un archivo de MySQL que contenga el código para crear la tabla (el cual proporcionamos):

Base de datos: urbann

Tabla: Usuarios

Estructura

Nombre Tipo Longitud/Valores Predeterminado Cotejamiento Atributos Nulo Índice

IDUsuario INT Ninguno PRIMARY

NombreUsuario VARCHAR(50) Ninguno

Contraseña VARCHAR(50) Ninguno

Nombre VARCHAR(50) Ninguno

Email INT(50) Ninguno

 INT Ninguno

 INT Ninguno

Comentarios de la tabla: Motor de almacenamiento: InnoDB Cotejamiento:

definición de la PARTICIÓN:

```

1  create table Usuarios
2  (
3    IdUsuario int not null identity,
4    NombreUsuario varchar(50) not null,
5    Contraseña varchar(50) not null,
6    ApPaterno varchar(50) not null,
7    ApMaterno varchar(50) not null,
8    Nombre varchar(50) not null,
9    Email varchar(50) not null,
10   Constraint PK_Usuarios primary key (IdUsuario)
11  )
12
13  create table AutoBus
14  (
15    IdAutoBus int not null,
16    TipoAutoBus varchar(50)not null,
17    Constraint PK_AutoBus primary key (IdAutoBus)
18  )
19
20  create table Ruta
21  (
22    IdRuta int not null,
23    TipoRuta varchar(50) not null,
24    Lugar varchar(50) null,
25    Latitud float null,
26    Longitud float null,
27    Status int null,
28    Counter int null,
29    Constraint PK_Ruta primary key (IdRuta)
30  )
31
32  create table UrbanApp
33  (
34    Usuario int not null,
35    Rutas int not null,
36    Autobuses int not null,
37    Constraint PK_UrbanApp primary key (Usuario),
38    Constraint FK_UrbanApp_Usuarios foreign key (Usuario) references Usuarios (IdUsuario),
39    Constraint FK_UrbanApp_Ruta foreign key (Rutas) references Ruta (IdRuta),
40    Constraint FK_UrbanApp_Autobus foreign key (Autobuses) references Autobus (IdAutoBus)

```

El procedimiento para importar los scripts hechos en un archivo (el archivo que se desee importar a la base de datos debe estar en formato .sql) en el menú de opciones superior seleccionamos “Importar”, y después seleccionamos el botón de “Seleccionar Archivo...”, y nos lanzara un explorador para elegir el archivo a insertar en la base de datos.

phpMyAdmin

Servidor: mysql wampserver

- Bases de datos
- SQL
- Estado actual
- Usuarios
- Exportar
- Importar** 1
- Configuración
- Replicación
- Variables
- Más

Importando al servidor actual

Archivo a importar:
 El archivo puede ser comprimido (gzip, bzip2, zip) o descomprimido.
 Un archivo comprimido tiene que terminar en **.[formato].[compresión]**. Por ejemplo: **.sql.zip**

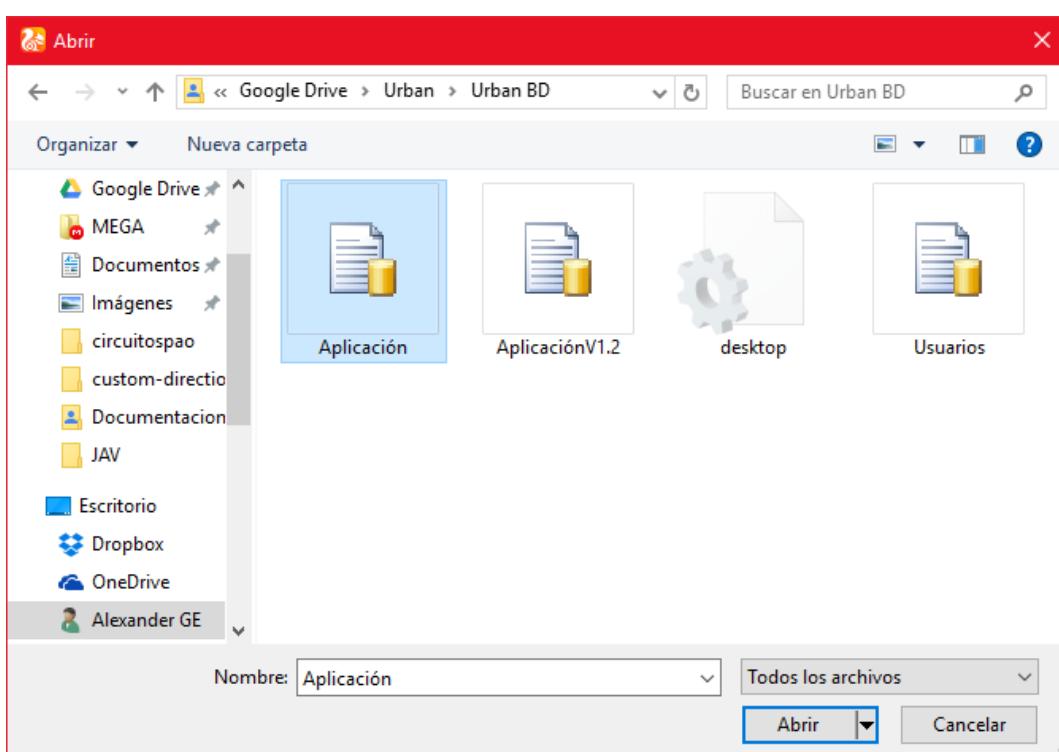
Buscar en su ordenador 2 Seleccionar archivo No se eligió archivo (Máximo: 128MB)

Conjunto de caracteres del archivo: utf-8

Importación parcial:
 Permitir la interrupción de una importación en caso que el script detecte que se ha acercado al límite de tiempo PHP. (Esto podría ser un buen método para importar archivos grandes; sin embargo, puede dañar las transacciones.)
 Omitir esta cantidad de consultas (en SQL) o líneas (en otros formatos) desde la primera: 0

Formato:
SQL Aquí se selecciona el formato que se desea importar(de preferencia SQL).

Opciones específicas al formato:
 Modalidad SQL compatible: NONE



Ya que tengamos seleccionada la base de dato presionamos el botón de abrir, y nos aparecerá automáticamente alado el botón, el nombre del archivo y su terminación (Si tiene problemas con esta cuestión véase en el apartado de Preguntas frecuentes), y en la parte inferior de administrador del PHP se encuentra el botón de “Continuar” y lo seleccionamos.

El archivo puede ser comprimido (.gzip, .bz2, .zip) o descomprimido.
Un archivo comprimido tiene que terminar en **[formato].[compresión]**. Por ejemplo: **.sql.zip**

Buscar en su ordenador: (Máximo: 128MB)

Conjunto de caracteres del archivo: utf-8

Importación parcial:

Permitir la interrupción de una importación en caso que el script detecte que se ha acercado al límite de tiempo PHP. (Esto podría ser un buen método para importar archivos grandes; sin embargo, puede dañar las transacciones.)

Omitir esta cantidad de consultas (en SQL) o líneas (en otros formatos) desde la primera:

Formato:

Opciones específicas al formato:

Modalidad SQL compatible:

No utilizar AUTO_INCREMENT con el valor 0

Continuar

Si el archivo que se deseaba importar se creó correctamente en la base de datos, nos re direccionara a la siguiente página la cual es la confirmación de que su base de datos está configurada correctamente.

La importación se ejecutó exitosamente, se ejecutaron 11 consultas. (myapp.sql)

```
-- phpMyAdmin SQL Dump
-- version 4.1.14
-- http://www.phpmyadmin.net
--
-- Servidor: 127.0.0.1
-- Tiempo de generación: 15-07-2016 a las 22:49:17
-- Versión del servidor: 5.6.17
-- Versión de PHP: 5.5.12

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";# MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas).

SET time_zone = "+00:00";# MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas).

/*I40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;# MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas).
/*I40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;# MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas).
/*I40101 SET @OLD_COLLATION_CONNECTION=@@@COLLATION_CONNECTION */;# MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas).
/*I40101 SET NAMES utf8 */;# MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas).

-- Base de datos: `myapp`
[...]
```

Para confirmar que las tablas necesarias para las aplicaciones de urban están realizada no dirigimos al panel lateral izquierdo donde se encuentran las bases de datos y seleccionamos sobre la que estamos trabajando, al seleccionar nos mostrara un panel con sus tablas creadas.

Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Residuo a depurar
user	Examinar Estructura Buscar Insertar Vaciar Eliminar	~0	InnoDB	latin1_swedish_ci	48 KB	-
1 tabla	Número de filas	0	InnoDB	latin1_swedish_ci	48 KB	0 B

↑ Marcar todos Para los elementos que están marcados: ▾

REGISTRO DE USUARIOS

La base de datos es necesaria para el registro de los usuarios y estos obtengan los servicios que Urban ofrece, el registro de usuarios se realiza mediante un registro de PHP y este mismo manda a llamar un Js (Java Script) de acceso en la Web, a continuación, se explican las acciones del código PHP para registrar usuarios.

HTML DEL REGISTRO DE USUARIOS

Este el código que realiza el registro visual para el registro de usuario, nótese que al final se manda a llamar un script el cual se encarga de insertar a los usuarios en la base de datos.

```
34 <div class="container">
35 <div class="row">
36 <div class="col-md-6">
37   <h2>Registro</h2>
38
39   <form role="form" name="registro" action="php/registro.php" method="post">
40     <div class="form-group">
41       <label for="username">Nombre de usuario</label>
42       <input type="text" class="form-control" id="username" name="username" placeholder="Nombre de usuario">
43     </div>
44     <div class="form-group">
45       <label for="fullname">Nombre Completo</label>
46       <input type="text" class="form-control" id="fullname" name="fullname" placeholder="Nombre Completo">
47     </div>
48     <div class="form-group">
49       <label for="email">Correo Electronico</label>
50       <input type="email" class="form-control" id="email" name="email" placeholder="Correo Electronico">
51     </div>
52     <div class="form-group">
53       <label for="password">Contraseña</label>
54       <input type="password" class="form-control" id="password" name="password" placeholder="Contraseña">
55     </div>
56     <div class="form-group">
57       <label for="confirm_password">Confirmar Contraseña</label>
58       <input type="password" class="form-control" id="confirm_password" name="confirm_password" placeholder="Confirmar Contraseña">
59     </div>
60
61     <button type="submit" class="btn btn-default">Registrar</button>
62   </form>
63 </div>
64 </div>
65 </div>
66
67   <script src="js/valida_registro.js"></script>
```

JS DEL REGISTRO DE USUARIOS

A continuación, se explica el Java script de valida_registro.js, este se encarga de validar los campos correctamente y mandar una alerta en caso de que los usuarios no realizaron su registro adecuadamente, las especificaciones son las siguientes:

- Todos los campos deben estar llenados o contener caracteres.
- El correo electrónico debe tener @ y un dominio (@xxx.xxx).
- La contraseña no se debe mostrar al escribirla en los campos (para medidas de seguridad).
- Los dos campos de contraseña deben coincidir (es para evitar errores en la redacción del usuario).

```

1 ▼ with(document.registro){
2 ▼   onsubmit = function(e){
3     e.preventDefault();
4     ok = true;
5 ▼       if(ok && username.value==""){
6         ok=false;
7         alert("Debe escribir un nombre de usuario");
8         username.focus();
9       }
10 ▼      if(ok && fullname.value==""){
11        ok=false;
12        alert("Debe escribir su nombre");
13        fullname.focus();
14      }
15 ▼      if(ok && email.value==""){
16        ok=false;
17        alert("Debe escribir su email");
18        email.focus();
19      }
20 ▼      if(ok && password.value==""){
21        ok=false;
22        alert("Debe escribir su password");
23        password.focus();
24      }
25 ▼      if(ok && confirm_password.value==""){
26        ok=false;
27        alert("Debe escribir su confirmacion de password");
28        confirm_password.focus();
29      }
30
31 ▼      if(ok && password.value!= confirm_password.value){
32        ok=false;
33        alert("Los passwords no coinciden");
34        confirm_password.focus();
35      }
36
37
38      if(ok){ submit(); }
39    }
40  }

```

PHP DEL REGISTRO DE USUARIOS

El siguiente código se enlaza con el php de registro visual, este archivo registro.php es el que se encarga de realizar crear los usuarios en la base de datos, este lo primero que hace es localizar los campos del registro visual en php en la base de datos, ya que localiza los campos realiza una consulta (select) con los campos de email y usuario para verificar que estos no estén registrados ya en la base de datos y si lo están mandar un aviso y no realizar la creación del usuario. En cambio, si el email y usuario no están en la base de datos este procede a insertar un registro(insert) con los campos que se llenaron (para proceder a realizar la validación del registro el archivo validar_registro.js debió de mandar confirmación) y acomodándolos en sus respectivos campos en la base de datos (es para eso que se realiza primero la localización de los campos) y manda un mensaje de registro exitoso y nos re direccionara al registro de iniciar sesión el cual significa que el usuario creo una cuenta para los servicios de Urban Web.

```

1  <?php
2
3  if(empty($_POST)){
4    if(isset($_POST["username"]) && !isset($_POST["fullname"]) && !isset($_POST["email"]) && !isset($_POST["password"]) && !isset($_POST["confirm_password"])){
5      if($_POST["username"]!="" && $_POST["fullname"]!="" && $_POST["email"]!="" && $_POST["password"]!="" && $_POST["password"]==$POST["confirm_password"]){
6          include "conexion.php";
7
8          $found=false;
9          $sql1= "select * from user where username='".$_POST[username]."' or email='".$_POST[email]."'";
10         $query = $con->query($sql1);
11         while ($r=$query->fetch_array()) {
12             $found=true;
13             break;
14         }
15         if($found){
16             print "<script>alert(\"Nombre de usuario o email ya estan registrados.\");window.location='..registro.php';</script>";
17         }
18         $sql = "insert into user(username,fullname,email,password,created_at) value ('".$_POST[username]."','".$_POST[fullname]."','".$_POST[email]."','".$_POST[password]."','NOW()')";
19         $query = $con->query($sql);
20         if($query!=null){
21             print "<script>alert(\"Registro exitoso. Proceda a logearse\");window.location='..login.html';</script>";
22         }
23     }
24 }
25 ?>

```

ACCESO A USUARIOS

Procederemos a explicar el procedimiento del login del usuario, para que el usuario pueda acceder a los servicios tiene que iniciar sesión con su cuenta que previamente, a continuación, explicaremos el proceso del acceso a la cuenta, de igual manera que el registro de usuario el primero paso es el código visual del registro, pero en este caso es un registro de acceso.

Nombre de usuario o email

Contraseña

HTML DEL ACCESO A USUARIOS

Este código muestra el contenido visual del registro en html incluyendo sus campos para la contraseña y el email o usuario.

```

70  <div class="container">
71  <div class="row">
72  <div class="col-md-6">
73      <h2>Iniciar Sesión</h2>
74
75      <form role="form" name="login" action="php/login.php" method="post">
76          <div class="form-group">
77              <label for="username">Nombre de usuario o email</label>
78              <input type="text" class="form-control" id="username" name="username" placeholder="Nombre de usuario">
79          </div>
80          <div class="form-group">
81              <label for="password">Contraseña</label>
82              <input type="password" class="form-control" id="password" name="password" placeholder="Contraseña">
83          </div>
84
85          <button type="submit" class="btn btn-default">Acceder</button>
86      </form>
87  </div>
88 </div>
89 </div>
90     <script src="js/valida_login.js"></script>

```

Nótese que al final se manda a llamar el un js el cual se explica a continuación:

JS DEL ACCESO A USUARIOS

Este código de js se encarga de validar los campos que estén llenos correctamente para el acceso al usuario, el cual también desde el inicio manda a llamar un archivo llamado “Login” en caso de que las condiciones no se cumplan:

```

1▼ with(document.login){
2▼     onsubmit = function(e){
3         e.preventDefault();
4         ok = true;
5▼         if(ok && username.value==""){
6             ok=false;
7             alert("Debe escribir un nombre de usuario");
8             username.focus();
9         }
10▼        if(ok && password.value==""){
11            ok=false;
12            alert("Debe escribir su password");
13            password.focus();
14        }
15        if(ok){ submit(); }
16    }
17 }

```

PHP DEL ACCESO A USUARIOS

El archivo que el js manda llamar en su condición es el login.php este de encarga de hacer la consulta a la base de datos con la información de los campos llenados, si la base de datos devuelve un valor con los datos que el usuario introdujo, será redireccionado a la página principal, pero si no recibe respuesta de la base de datos solo recarga la página de login.

```

1 <?php
2
3▼ if(!empty($_POST)){
4▼     if(isset($_POST["username"]) && isset($_POST["password"])){
5▼         if($_POST["username"]!=""&&$_POST["password"]!=""){
6             include "conexion.php";
7
8             $user_id=null;
9             $sql1= "select * from user where (username='".$_POST[username]."' or email='".$_POST[username]."' ) and
10                |password='".$_POST[password]."' ";
11             $query = $con->query($sql1);
12▼             while ($r=$query->fetch_array()) {
13                 $user_id=$r["id"];
14                 break;
15             }
16             if($user_id==null){
17                 print "<script>alert(\"Acceso invalido.\");window.location='..../login1.html';</script>";
18▼             }else{
19                 session_start();
20                 $_SESSION["user_id"]=$user_id;
21                 print "<script>window.location='..../acceso.php';</script>";
22             }
23         }
24     }
25 }
26 ?>

```

CONEXIÓN UNIVERSAL

Este mismo código manda a llamar a otro archivo (es muy importante) el cual se encarga simplemente la conexión de la página Web con la base de datos, esta conexión se mete a una función para que sea más fácil conectar a la base de datos, este archivo no solamente se puede utilizar para registro, también para cualquier razón que se necesite conectar a la base de datos.

```

1 <?php
2 $host="localhost";
3 $user="root";
4 $password="";
5 $db="myapp";
6 $con = new mysqli($host,$user,$password,$db);
7
8 ?>

```

TRAZADOR DE RUTAS

Este programa se encarga de trazar las rutas que los usuarios deseen estableciendo dos puntos, el de partida y el de destino, este programa se conforma de dos códigos, el html y el js.

HTML DE TRAZADOR DE RUTAS

a continuación, explicaremos el código html, esta es la parte visual de los formularios para el trazado de las rutas, este se conforma de 3 botones, uno es para señalar la ubicación actual, otro es para trazar las rutas cuando los campos están llenos y el ultimo es una pequeña pestaña que sirve para ocultar el panel de los datos.

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Google Maps API V3: Custom Directions Panel</title>
5     <meta name="viewport" content="initial-scale=1.0, user-scalable=no" />
6
7     <meta name="description" content="Google Maps API V3: Custom Directions Panel" />
8     <meta name="keywords" content="Google Maps API V3, Custom Directions Panel, jQuery, jquery tutorials, thewebstorebyg wordpress code blog" />
9     <meta name="author" content="Giri Jeedigunta - thewebstorebyg" />
10
11    <script type="text/javascript" src="https://ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"></script>
12    <script type="text/javascript" src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=true&libraries=places"></script>
13    <link href="style.css" type="text/css" rel="stylesheet" />
14
15
16  </head>
17  <body>
18    <div id="mapCanvas">&#160;</div>
19    <div id="directionsPanel">
20      <a href="#" id="useGPS">Use My Location</a>
21      <p class="or">[OR]</p>
22      <div class="directionInputs">
23        <form>
24          <p><label>A</label><input type="text" value="" id="dirSource" /></p>
25          <p><label>B</label><input type="text" value="" id="dirDestination" /></p>
26          <a href="#" id="getDirections">Get Directions</a>
27          <a href="#" id="reset">Reset</a>
28        </form>
29      </div>
30      <div id="directionSteps">
31        <p class="msg">Direction Steps Will Render Here</p>
32      </div>
33      <a href="#" id="toggleBtn" id="paneToggle" class="out">&lt;&lt;&gt;&lt;&gt;</a>
34    </div>
35    <script type="text/javascript" src="sample.js"></script>
36  </body>
37 </html>

```

JAVA DE TRAZADOR DE RUTAS

Este java se manda a llamar en el script final del html, este lo que hace es poner recomendaciones de ubicaciones al llenar los campos (con las que tiene registrada google), y al trazar la ruta (presionar el botón) rellene el panel donde se señalan las calles por las que se pueden trazar las rutas, además de que muestre el mapa con los puntos señalados y paradas. Cabe mencionar que los campos remarcados son los cuales se señalan los marcadores y también con este mismo se señalan las paradas por las que pasa el vehículo.

```
5 ▼ (function(mapDemo, $, undefined) {
6 ▼   mapDemo.Directions = (function() {
7 ▼     function Directions() {
8 ▼       var map,
9         directionsService, directionsDisplay,
10        autoSrc, autoDest, pinA, pinB,
11
12      markerA = new google.maps.MarkerImage('m1.png',
13        new google.maps.Size(24, 27),
14        new google.maps.Point(0, 0),
15        new google.maps.Point(12, 27)),
16      markerB = new google.maps.MarkerImage('m2.png',
17        new google.maps.Size(24, 28),
18        new google.maps.Point(0, 0),
19        new google.maps.Point(12, 28)),
20
21      // Caching the Selectors
22      $Selectors = {
23        mapCanvas: jQuery('#mapCanvas')[0],
24        dirPanel: jQuery('#directionsPanel'),
25        dirInputs: jQuery('.directionInputs'),
26        dirSrc: jQuery('#dirSource'),
27        dirDst: jQuery('#dirDestination'),
28        getDirBtn: jQuery('#getDirections'),
29        dirSteps: jQuery('#directionSteps'),
30        paneToggle: jQuery('#paneToggle'),
31        useGPSBtn: jQuery('#useGPS'),
32        paneResetBtn: jQuery('#paneReset')
33      },
34
35      autoCompleteSetup = function() {
36        autoSrc = new google.maps.places.Autocomplete($Selectors.dirSrc[0]);
37        autoDest = new google.maps.places.Autocomplete($Selectors.dirDst[0]);
38      }, // autoCompleteSetup Ends
39
40      directionsSetup = function() {
41        directionsService = new google.maps.DirectionsService();
42        directionsDisplay = new google.maps.DirectionsRenderer({
43          suppressMarkers: true
44        });
45
46        directionsDisplay.setPanel($Selectors.dirSteps[0]);
47      }, // directionsSetup Ends
```

Aquí se declaran las variables con las cuales se establecen los puntos de partida y destino el marcador(unas imágenes) y el tamaño del mismo.

```
49▼
50
51▼
52
53
54
55
56
57
58
59
60▼
61▼
62
63
64▼
65
66
67
68
69
70
71
72▼
73▼
74
75
76
77
78▼
79
80
81
82
83
84
85
86
87
88▼
89▼
90
91
92
```

```
trafficSetup = function() {
    // Creating a Custom Control and appending it to the map
    var controlDiv = document.createElement('div'),
        controlUI = document.createElement('div'),
        trafficLayer = new google.maps.TrafficLayer();

    jQuery(controlDiv).addClass('gmap-control-container').addClass('gmnoprint');
    jQuery(controlUI).text('Traffic').addClass('gmap-control');
    jQuery(controlDiv).append(controlUI);

    // Traffic Btn Click Event
    google.maps.event.addListener(controlUI, 'click', function() {
        if (typeof trafficLayer.getMap() == 'undefined' || trafficLayer.getMap() === null) {
            jQuery(controlUI).addClass('gmap-control-active');
            trafficLayer.setMap(map);
        } else {
            trafficLayer.setMap(null);
            jQuery(controlUI).removeClass('gmap-control-active');
        }
    });
    map.controls[google.maps.ControlPosition.TOP_RIGHT].push(controlDiv);
}, // trafficSetup Ends

mapSetup = function() {
    map = new google.maps.Map($Selectors.mapCanvas, {
        zoom: 9,
        center: new google.maps.LatLng(22.219361, -97.862272),

        mapTypeControl: true,
        mapTypeControlOptions: {
            style: google.maps.MapTypeControlStyle.DEFAULT,
            position: google.maps.ControlPosition.TOP_RIGHT
        },

        panControl: true,
        panControlOptions: {
            position: google.maps.ControlPosition.RIGHT_TOP
        },

        zoomControl: true,
        zoomControlOptions: {
            style: google.maps.ZoomControlStyle.LARGE,
            position: google.maps.ControlPosition.RIGHT_TOP
        },
    });
}
```

```

94         scaleControl: true,
95         streetViewControl: true,
96         overviewMapControl: true,
97
98         mapTypeId: google.maps.MapTypeId.ROADMAP
99     });
100
101     autoCompleteSetup();
102     directionsSetup();
103     trafficSetup();
104 }, // mapSetup Ends
105
106 ▼ directionsRender = function(source, destination) {
107     $selectors.dirSteps.find('.msg').hide();
108     directionsDisplay.setMap(map);
109
110     var request = {
111         origin: source,
112         destination: destination,
113         provideRouteAlternatives: false,
114         travelMode: google.maps.DirectionsTravelMode.DRIVING
115     };
116
117 ▼ directionsService.route(request, function(response, status) {
118     if (status == google.maps.DirectionsStatus.OK) {
119
120         directionsDisplay.setDirections(response);
121
122         var _route = response.routes[0].legs[0];
123
124         pinA = new google.maps.Marker({position: _route.start_location, map: map, icon: markerA}),
125         pinB = new google.maps.Marker({position: _route.end_location, map: map, icon: markerB});
126     }
127 });
128 }, // directionsRender Ends
129
130 ▼ fetchAddress = function(p) {
131     var Position = new google.maps.LatLng(p.coords.latitude, p.coords.longitude),
132     Locater = new google.maps.Geocoder();
133
134 ▼ Locater.geocode({'latLng': Position}, function(results, status) {
135     if (status == google.maps.GeocoderStatus.OK) {
136         var _r = results[0];
137         $selectors.dirSrc.val(_r.formatted_address);
138     }
139 });
140 }, // fetchAddress Ends

```

En esta parte es donde se proyectan en el mapa los marcadores que previamente declaramos.

```

142 ▼           invokeEvents = function() {
143             // Get Directions
144             $Selectors.getDirBtn.on('click', function(e) {
145               e.preventDefault();
146               var src = $Selectors.dirSrc.val(),
147                   dst = $Selectors.dirDst.val();
148
149               directionsRender(src, dst);
150             });
151
152             // Reset Btn
153             $Selectors.paneResetBtn.on('click', function(e) {
154               $Selectors.dirSteps.html('');
155               $Selectors.dirSrc.val('');
156               $Selectors.dirDst.val('');
157
158               if(pinA) pinA.setMap(null);
159               if(pinB) pinB.setMap(null);
160
161               directionsDisplay.setMap(null);
162             });
163
164             // Toggle Btn
165             $Selectors.paneToggle.toggle(function(e) {
166               $Selectors.dirPanel.animate({'left': '-=305px'});
167               jquery(this).html('>');
168             }, function() {
169               $Selectors.dirPanel.animate({'left': '+=305px'});
170               jquery(this).html('<');
171             });
172
173             // Use My Location / Geo Location Btn
174             $Selectors.useGPSBtn.on('click', function(e) {
175               if (navigator.geolocation) {
176                 navigator.geolocation.getCurrentPosition(function(position) {
177                   fetchAddress(position);
178                 });
179               }
180             });
181           }, //invokeEvents Ends
182
183             _init = function() {
184               mapSetup();
185               invokeEvents();
186             }; // _init Ends
187
188             this.init = function() {
189               init();
190
191               return this; // Refers to: mapDemo.Directions
192             }
193             return this.init(); // Refers to: mapDemo.Directions.init()
194           } // _Directions Ends
195           return new _Directions(); // Creating a new object of _Directions rather than a function
196         }()); // mapDemo.Directions Ends
197       })(window.mapDemo = window.mapDemo || {}, jquery);

```

MANUAL DE INSTALACIÓN



- A continuación mostraremos los pasos a seguir para Instalar el Java JDK y el Android SDK.
- Con ésto podremos desarrollar aplicaciones móviles con el Android Studio.

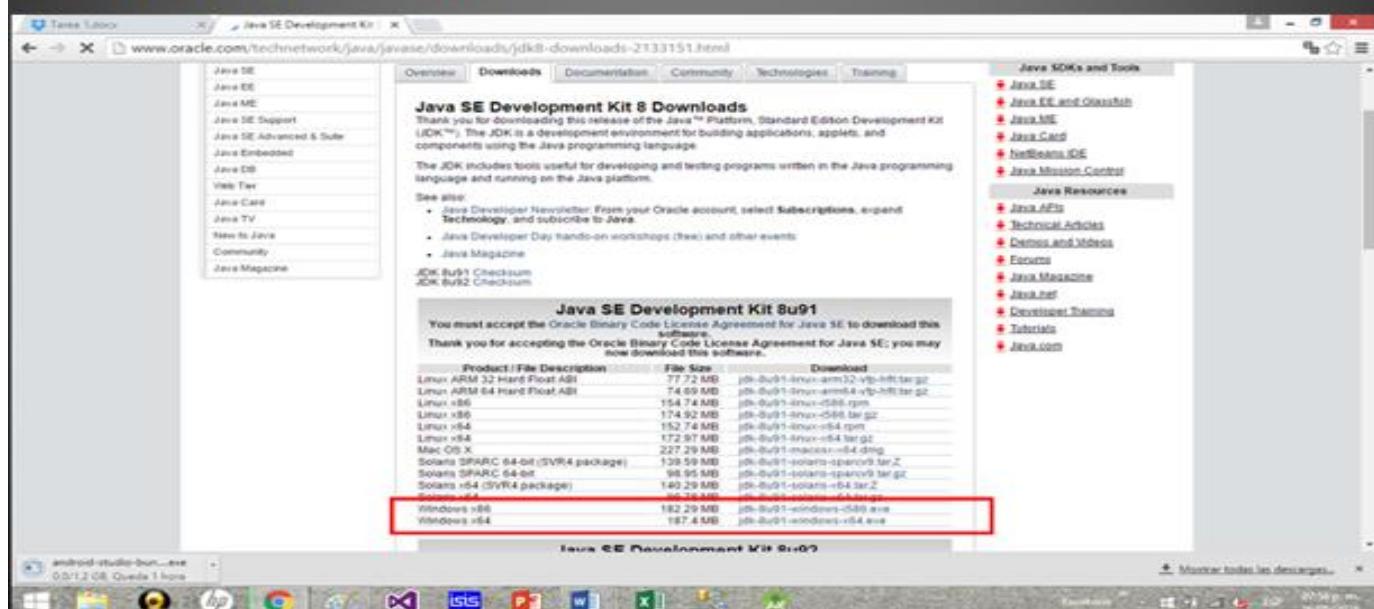
Instalación de Java JDK



Primero debemos instalar el Java JDK (Java Development Kit) antes que el SDK (Software Development Kit) debido a que es nuestro lenguaje de programación.

Éste es el enlace para descargar el Java JDK:

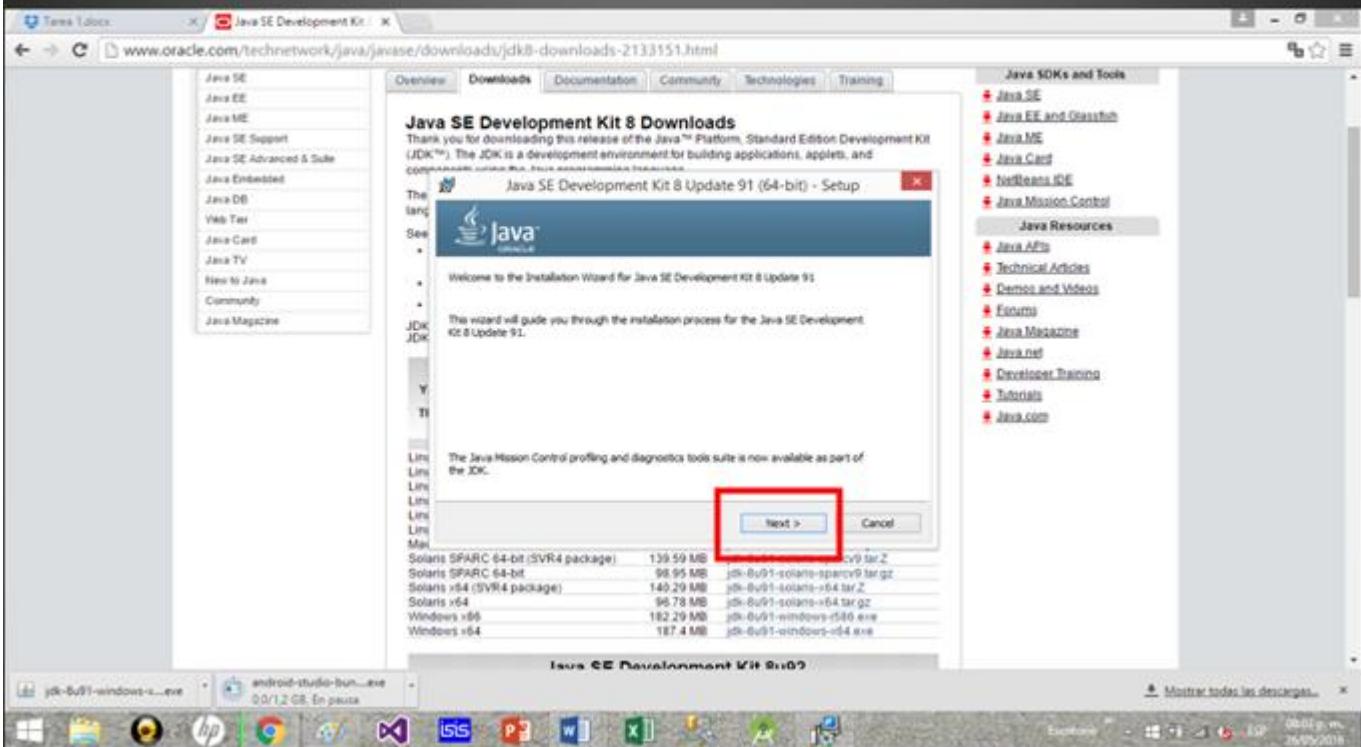
<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>



The screenshot shows the Java SE Development Kit 8 Downloads page from Oracle's website. The page has a sidebar with links like Java SE, Java EE, Java ME, Java Support, Java SE Advanced & Suite, Java Embedded, Java DB, View Test, Java Card, Java TV, News to Java, Community, and Java Magazine. The main content area has tabs for Overview, Downloads, Documentation, Community, Technologies, and Training. The Downloads tab is selected. It contains a section for Java SE Development Kit 8u91, which includes a note about accepting the Oracle Binary Code License Agreement. Below this is a table of download links for various platforms:

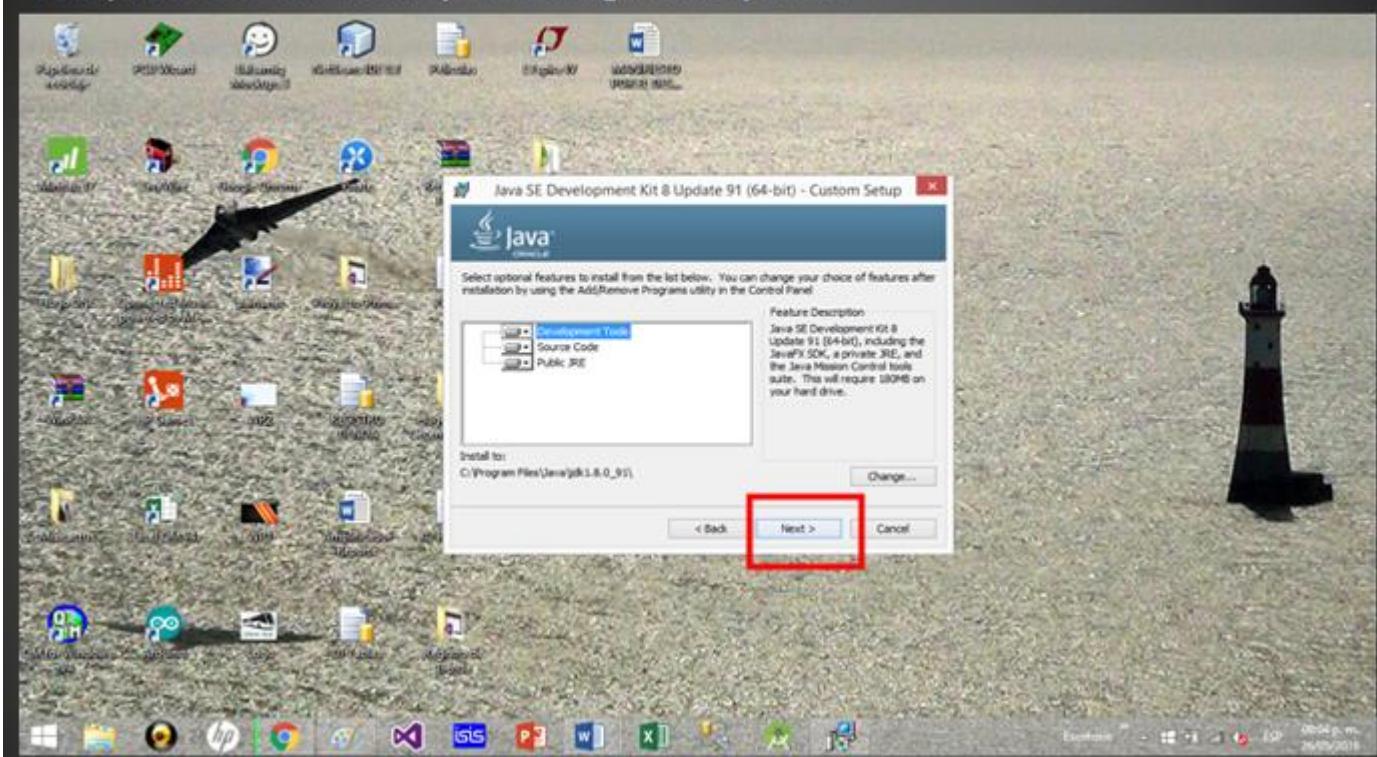
Product / File Description	File Size	Download
Linux ARM 32-Hard Float ABI	77.72 MB	jdk-8u91-linux-arm32-vfp-hf.tar.gz
Linux ARM 64-Hard Float ABI	74.69 MB	jdk-8u91-linux-arm64-vfp-hf.tar.gz
Linux x86	154.74 MB	jdk-8u91-linux-i586.rpm
Linux x86	174.92 MB	jdk-8u91-linux-i686.tar.gz
Linux x86	152.74 MB	jdk-8u91-linux-i686.rpm
Linux x64	172.29 MB	jdk-8u91-linux-x64.rpm
Mac OS X	227.29 MB	jdk-8u91-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	129.59 MB	jdk-8u91-solaris-sparcv9.tar.gz
Solaris SPARC 64-bit	98.95 MB	jdk-8u91-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.29 MB	jdk-8u91-solaris-x64.tar.gz
Solaris x64	95.78 MB	jdk-8u91-solaris-x64.tar.gz
Windows x86	182.21 MB	jdk-8u91-windows-i586.exe
Windows x64	187.4 MB	jdk-8u91-windows-x64.exe

Le damos clic en “Next” para el siguiente paso.



Aquí elegimos las opciones que queramos instalar.

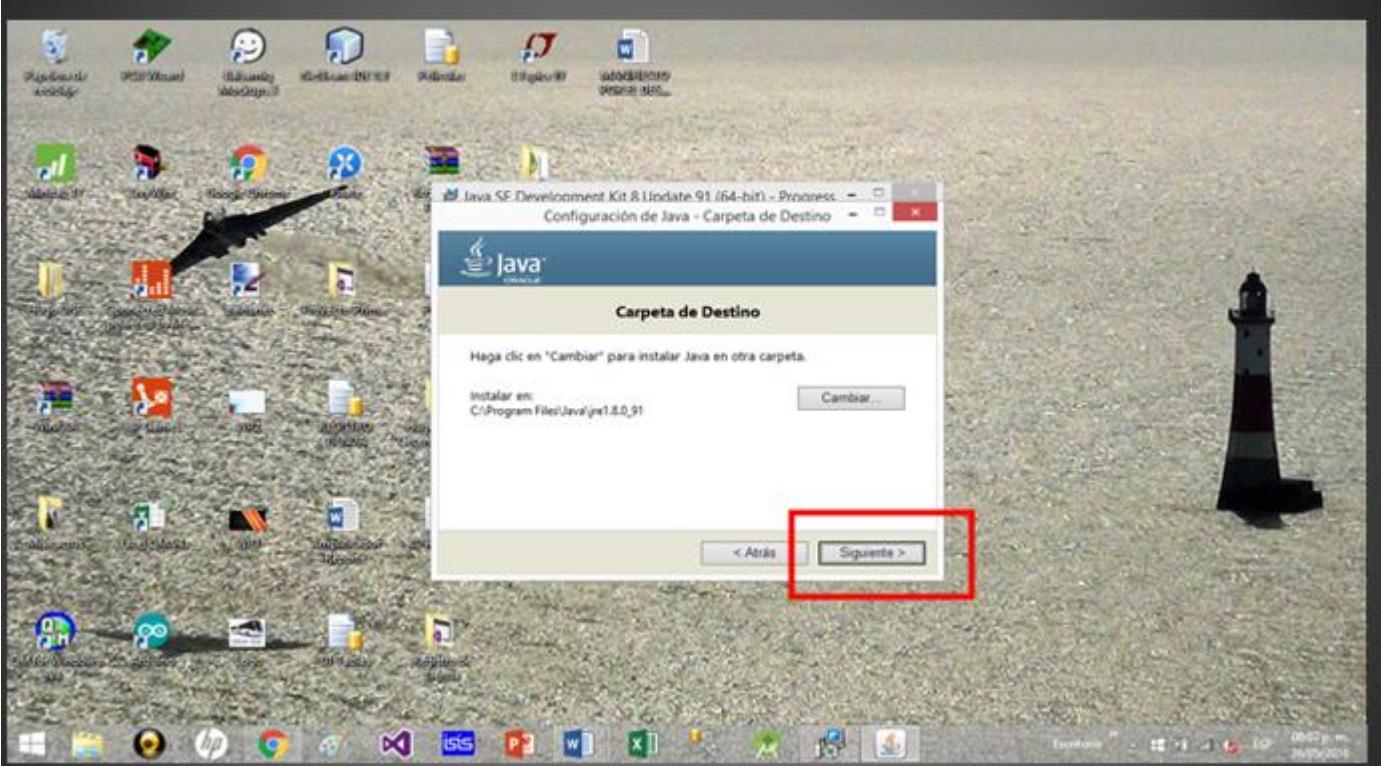
Después clic en “Next” para el siguiente paso.



Esperamos a que el software haga el procedimiento de descarga.



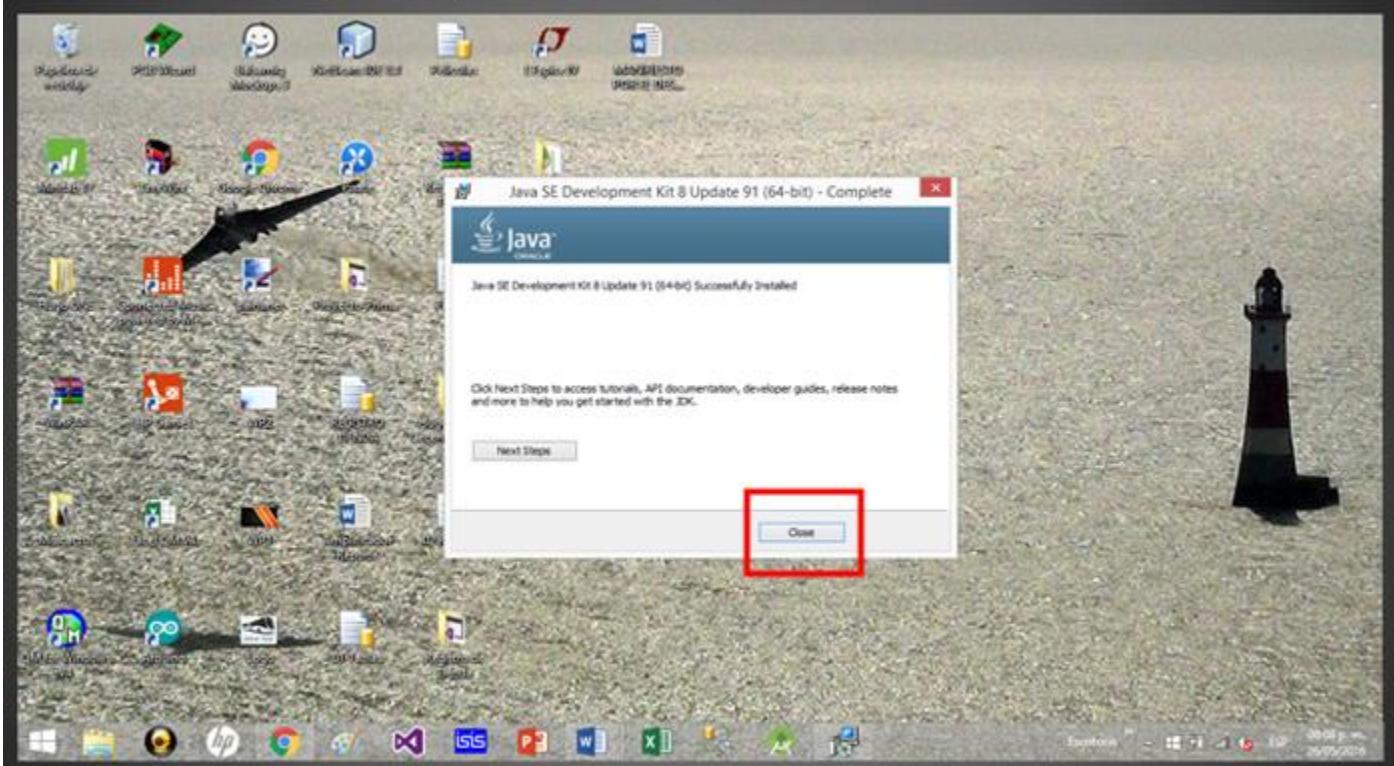
Terminada la descarga, damos clic en “Siguiente” para que inicie la instalación.



Esperamos a que se instale el JDK.

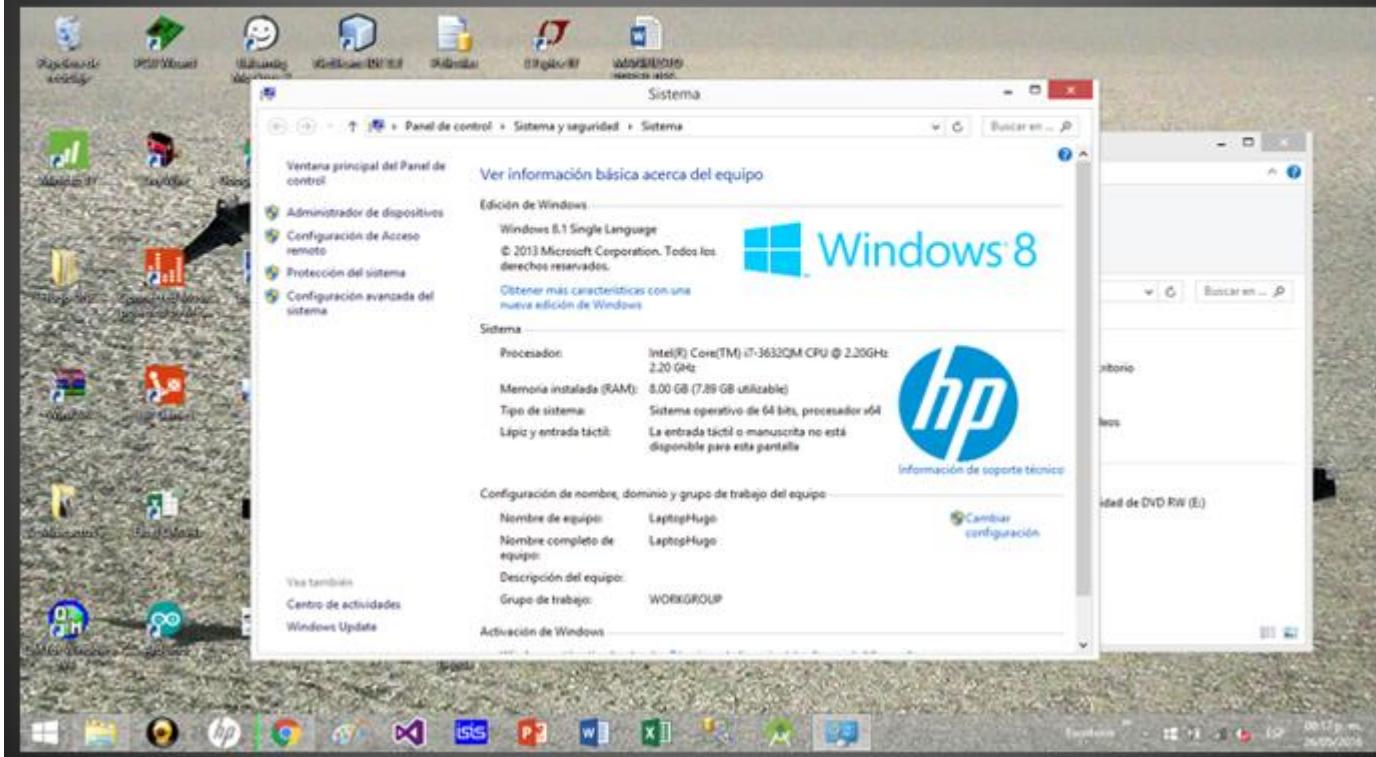


Ya con el JDK instalado, damos clic en "Close"

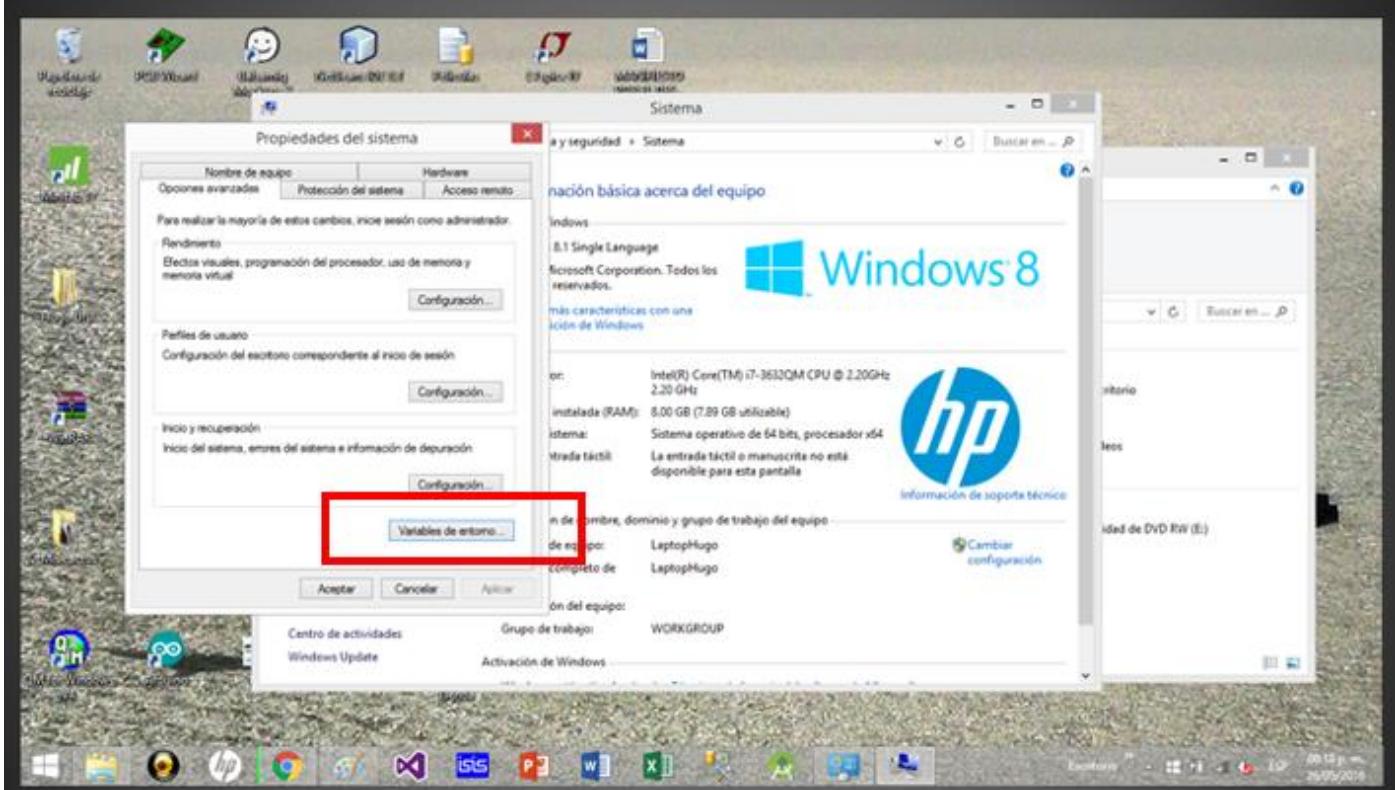


El siguiente paso es definir la variable de entorno para que el sistema pueda usar el lenguaje Java.

Abrimos la ventana de “Sistema” y le damos clic en “Configuración avanzada del Sistema”.



En ésta nueva ventana, le damos clic en “Variables de Entorno”.

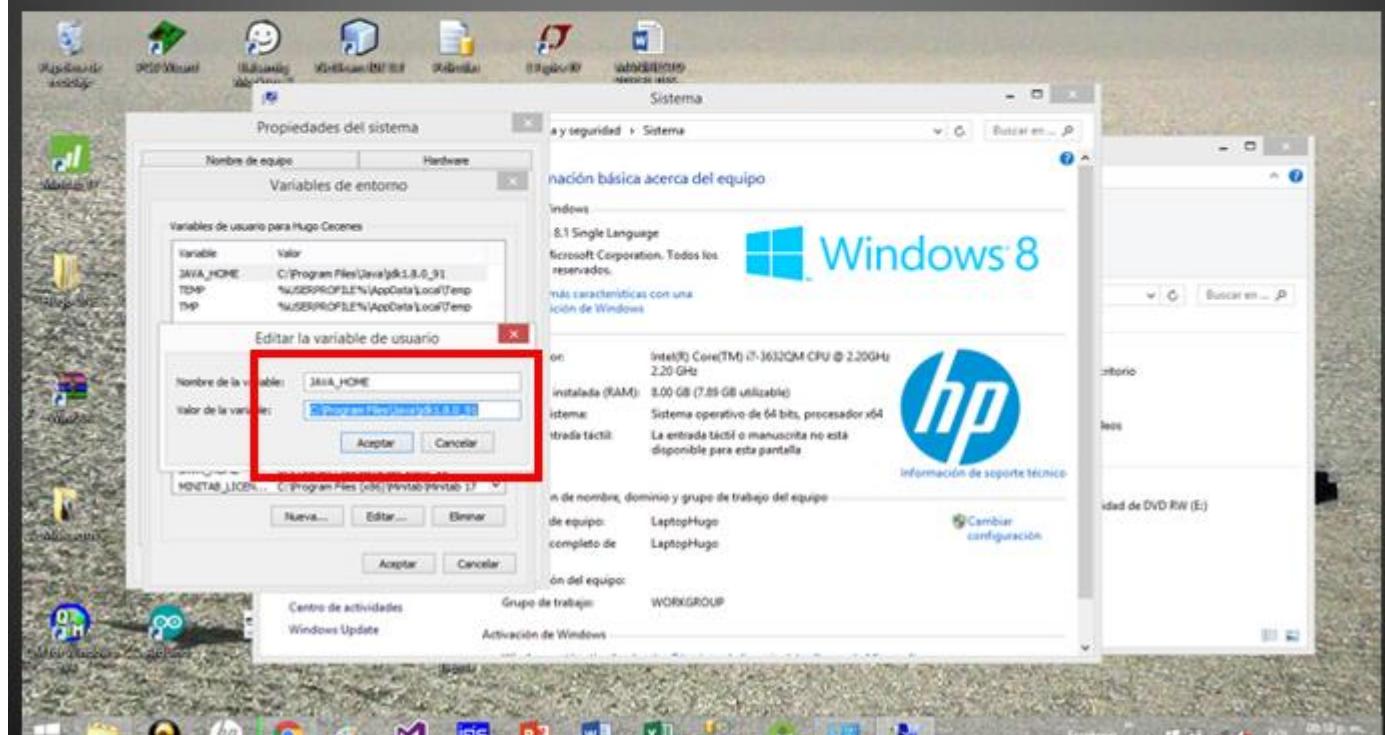


Aquí tenemos que crear una nueva variable de entorno.

Nos van a pedir el Nombre de la nueva variable, la cual va a ser: "JAVA_HOME"

También el valor de la variable, la cual va a ser la ruta donde se encuentra la carpeta del JDK.

Ahora damos paso a la instalación del Android JDK.



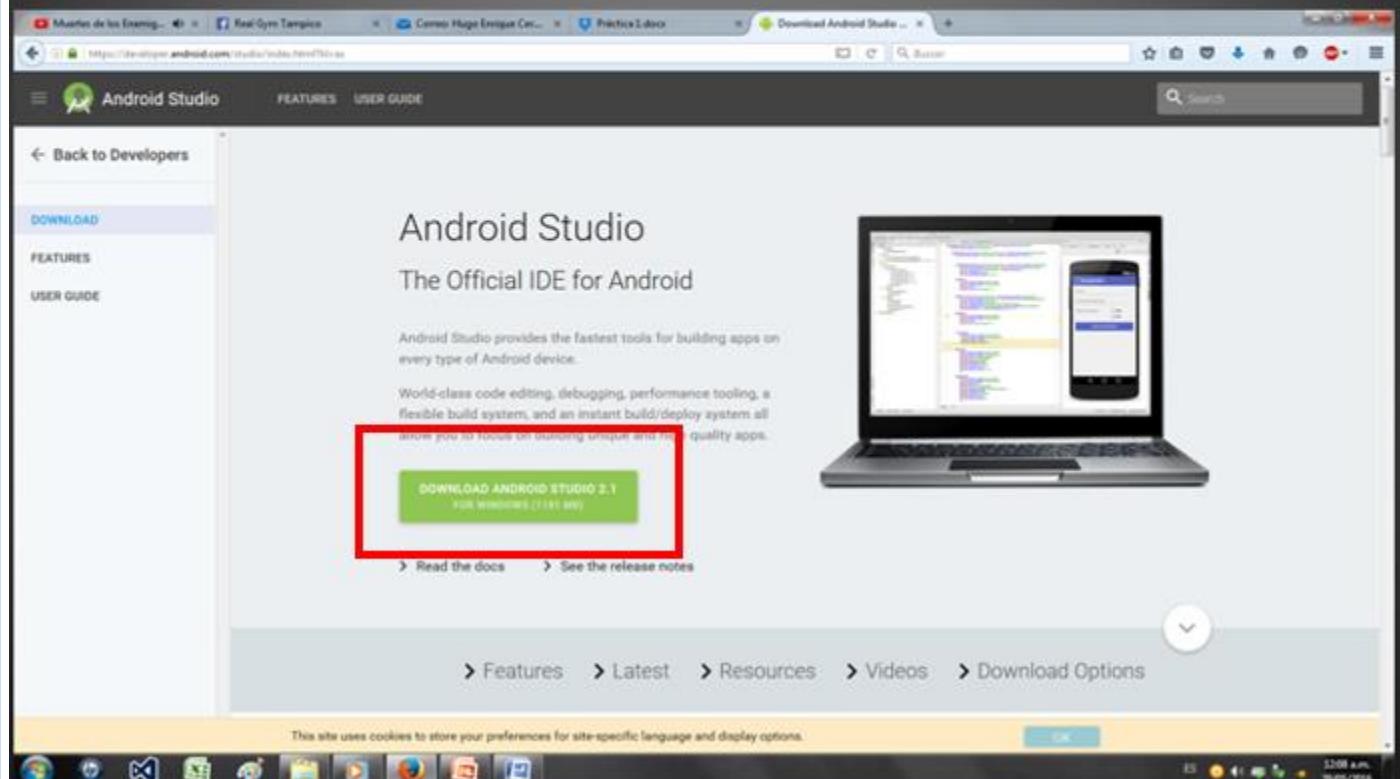
Instalación Android Studio



A continuación, vamos a instalar en Android Studio.

El enlace para la descarga es:

<https://developer.android.com/studio/index.html?hl=es#tos-header>



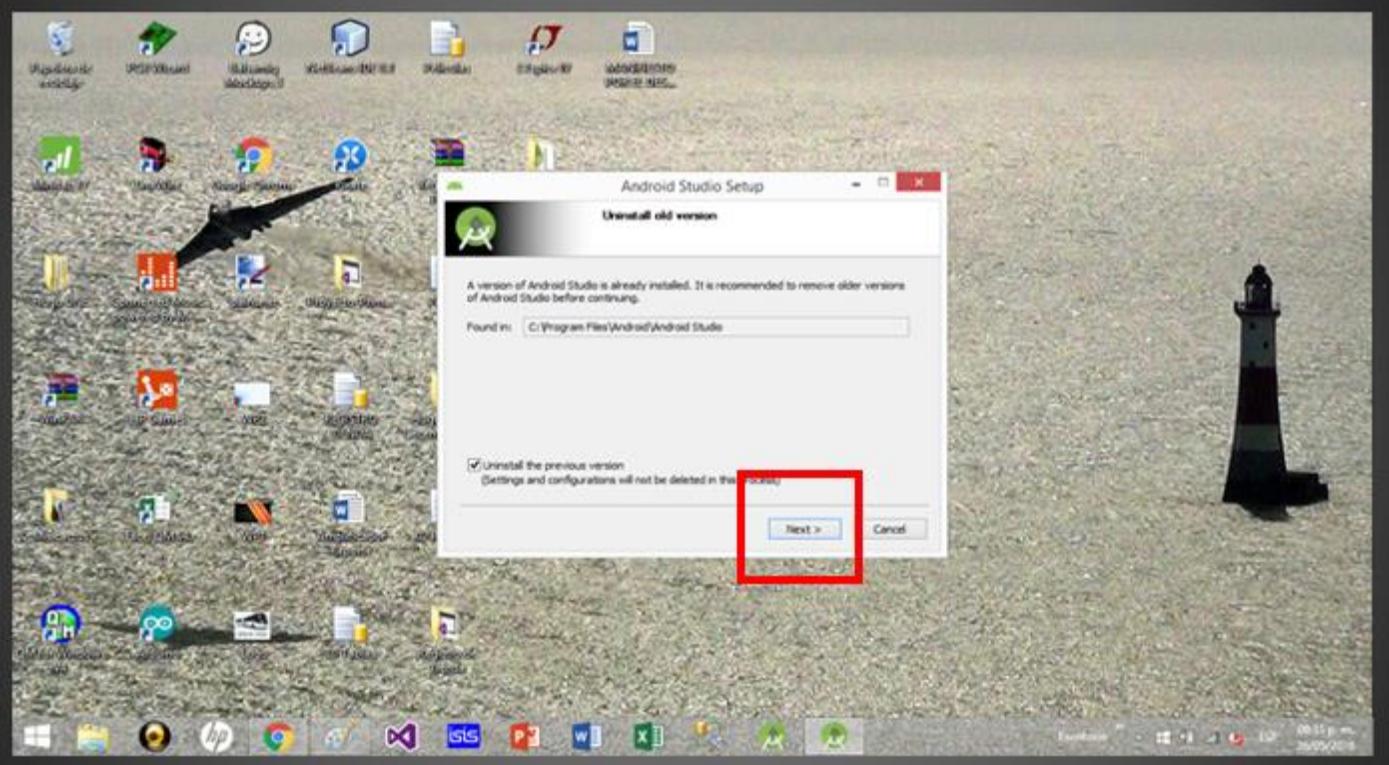
The screenshot displays the official Android Studio download page. At the top, there's a navigation bar with links for "Android Studio", "FEATURES", and "USER GUIDE". Below this, a sidebar on the left offers links to "Back to Developers", "DOWNLOAD" (which is highlighted in blue), "FEATURES", and "USER GUIDE". The main content area features the title "Android Studio" and the subtitle "The Official IDE for Android". It highlights that "Android Studio provides the fastest tools for building apps on every type of Android device." A prominent green button labeled "DOWNLOAD ANDROID STUDIO 3.1 FOR WINDOWS (1.1GB)" is centered, with a red rectangle drawn around it. Below the button are links for "Read the docs" and "See the release notes". To the right, there's an image of a laptop displaying the Android Studio interface, showing code and a connected smartphone. At the bottom of the page, there are links for "Features", "Latest", "Resources", "Videos", and "Download Options". A note at the very bottom states, "This site uses cookies to store your preferences for site-specific language and display options.", with a "OK" button. The browser's address bar shows the URL "https://developer.android.com/studio/index.html?hl=es#tos-header".

Comenzamos con el proceso de instalación.



Aquí nos muestra la dirección donde será instalado el programa.

Damos clic en “Next”.

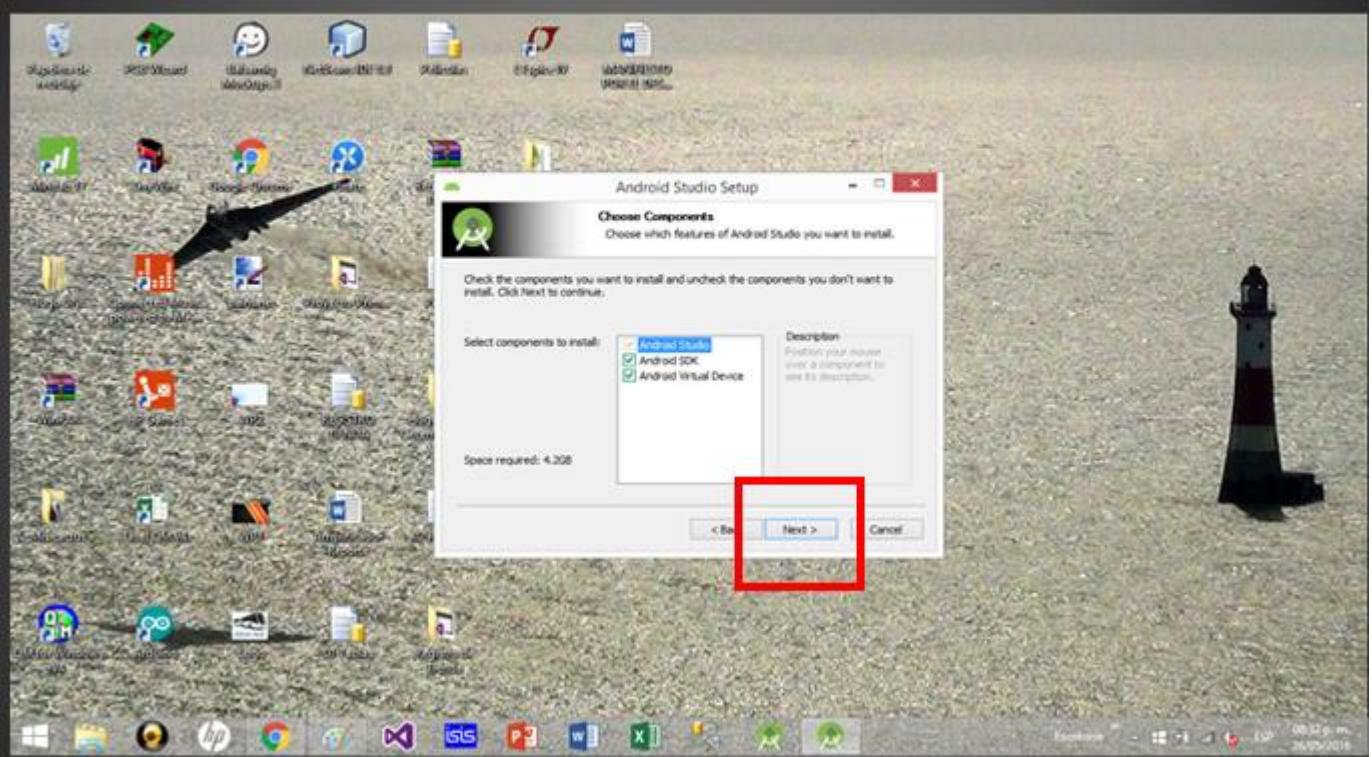


Damos otra vez clic en “Next”.



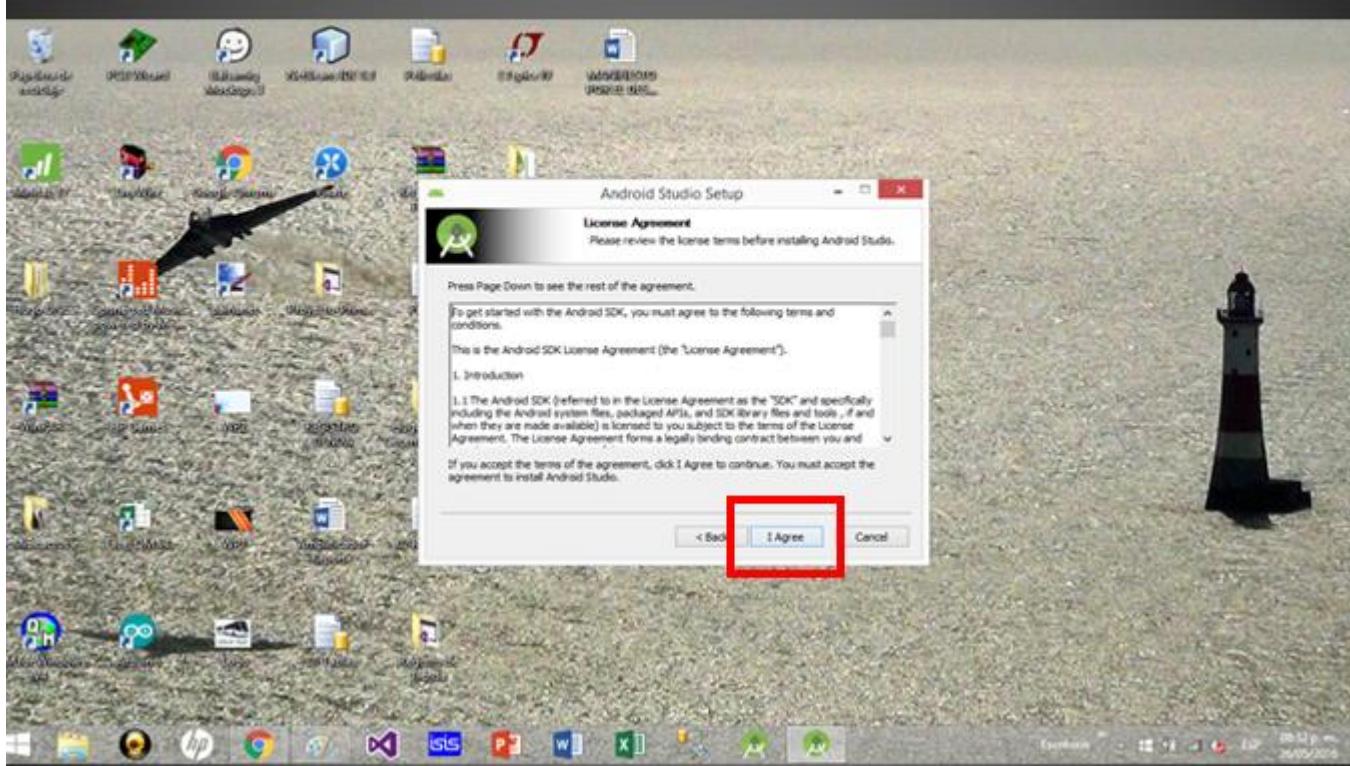
Aquí nos da varias opciones a instalar.

Después de seleccionar las opciones le damos clic en “Next”.



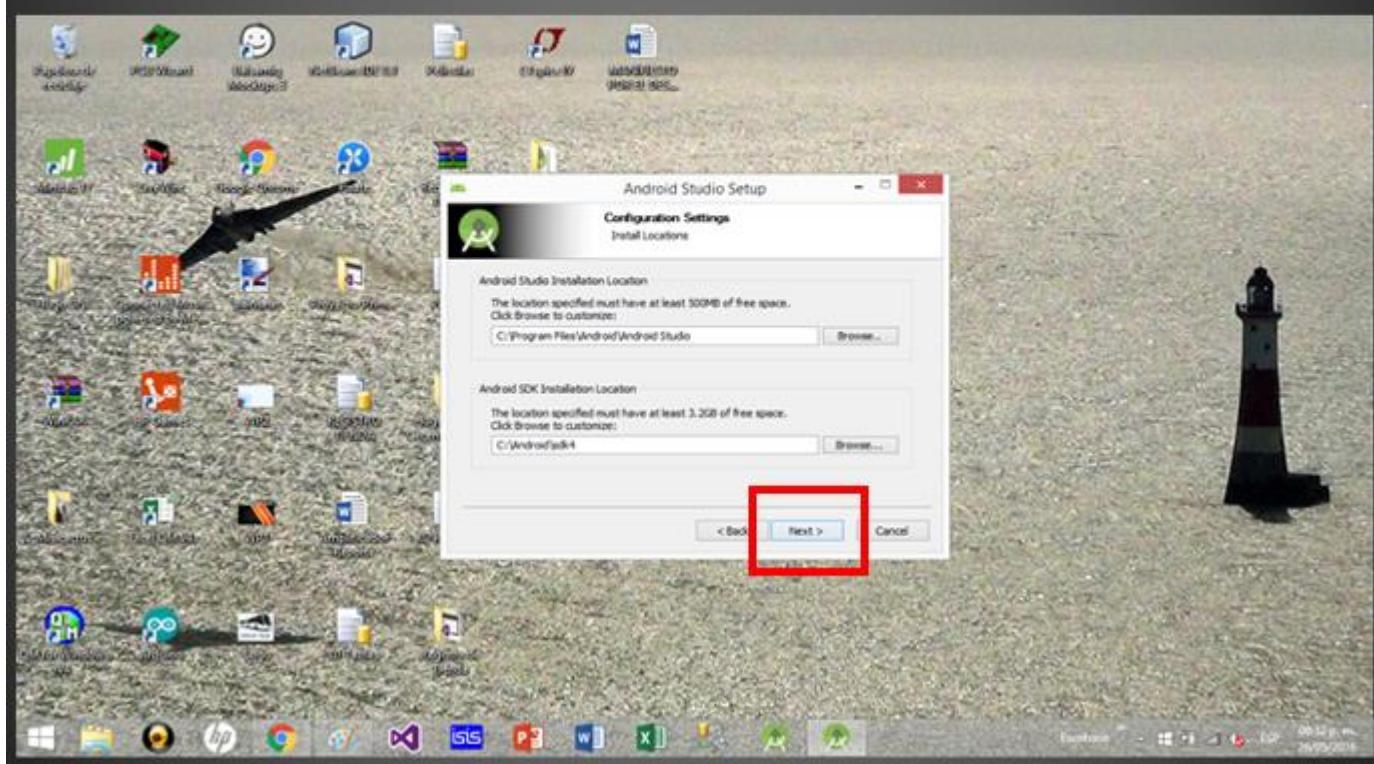
Leemos y aceptamos los términos de la licencia.

Le damos clic en “I Agree”.



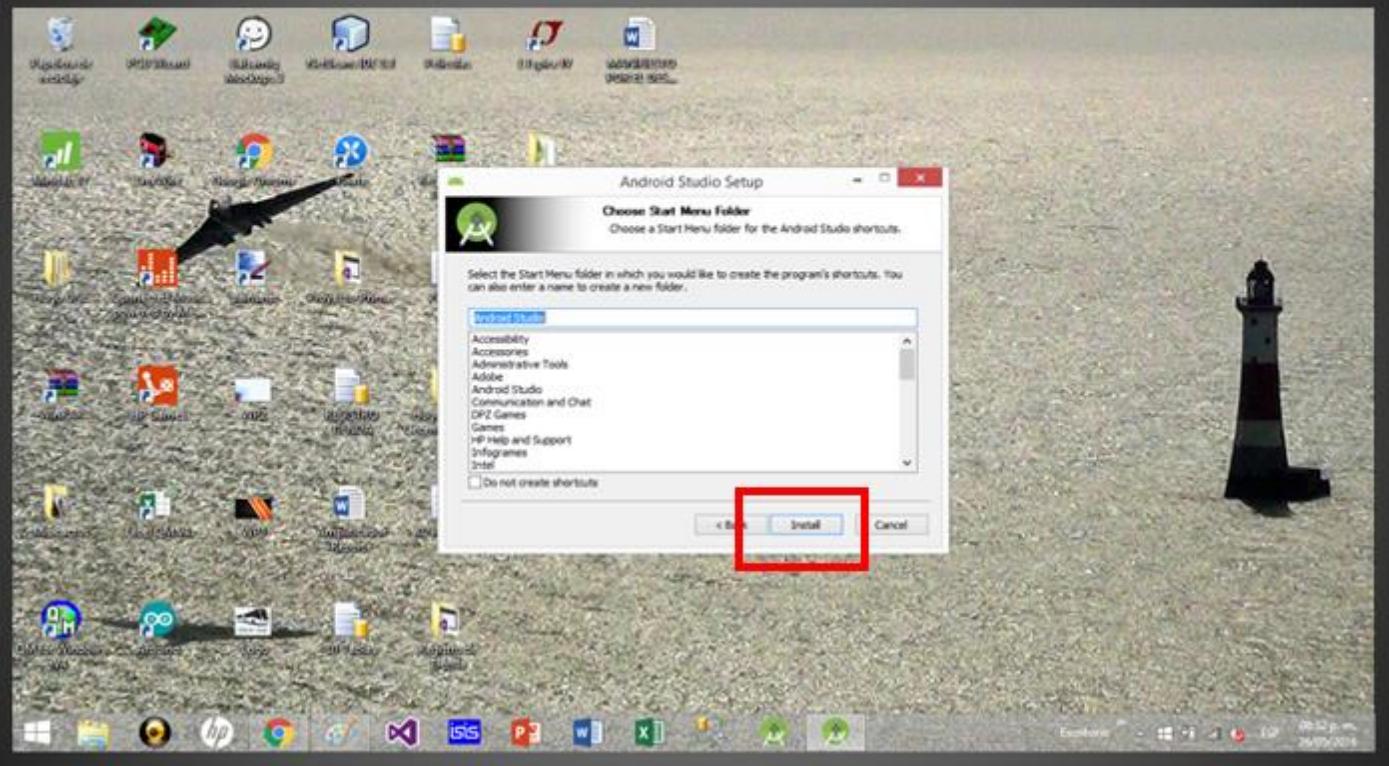
Aquí nos muestra las direcciones donde se van a instalar tanto Android Studio como el Android SDK.

Le damos clic en “Next”.



Aquí tenemos que elegir la carpeta para crear los “shortcuts” del programa.

Después le damos clic en “Install” para comenzar con la instalación.

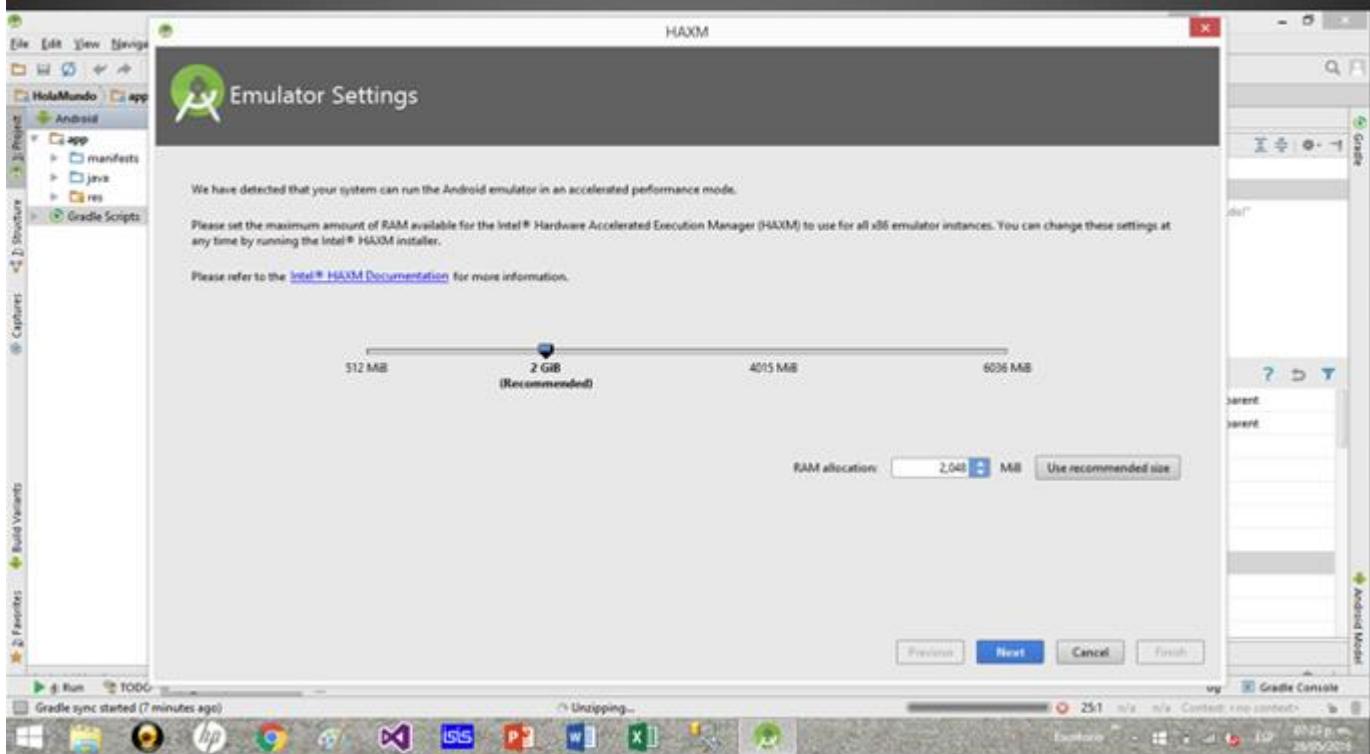


Esperamos a que termine la Instalación de programa.

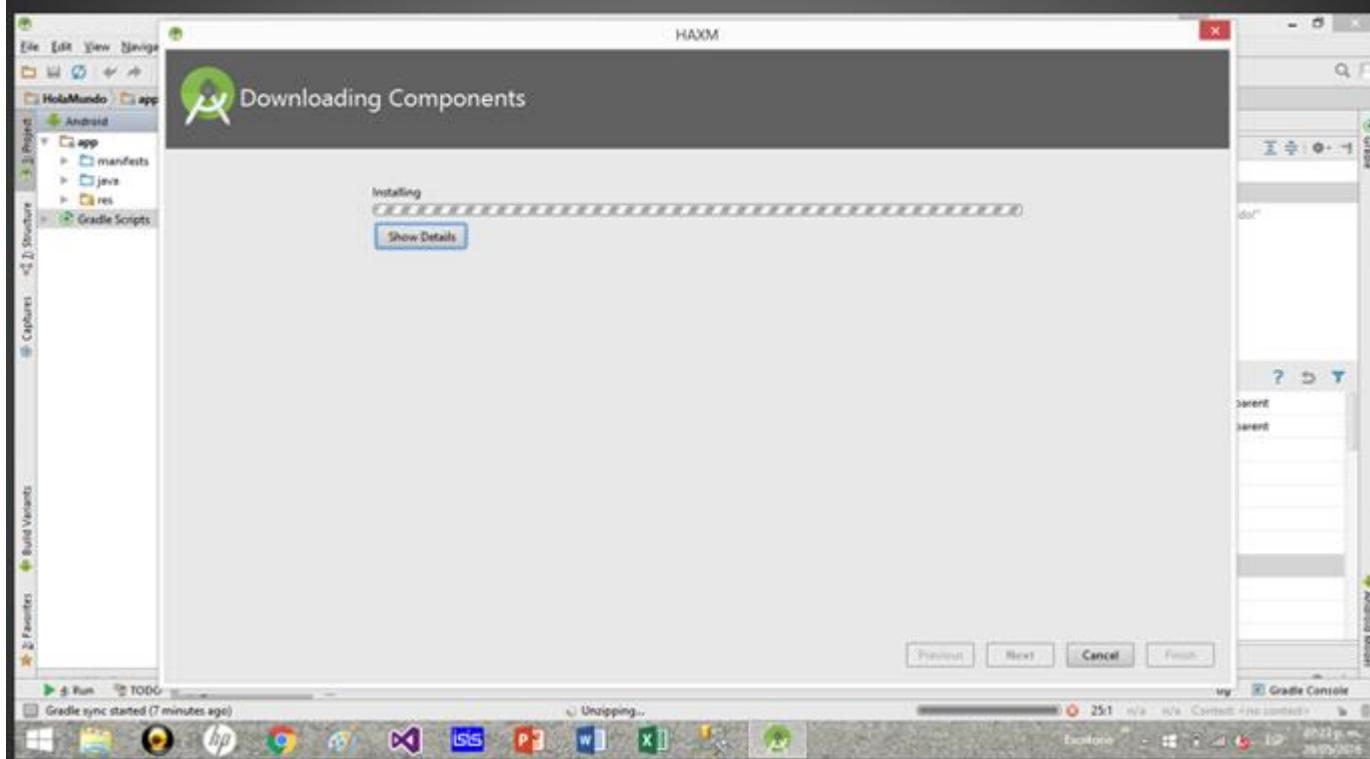


Aquí nos da a elegir la cantidad de RAM destinada para el emulador.

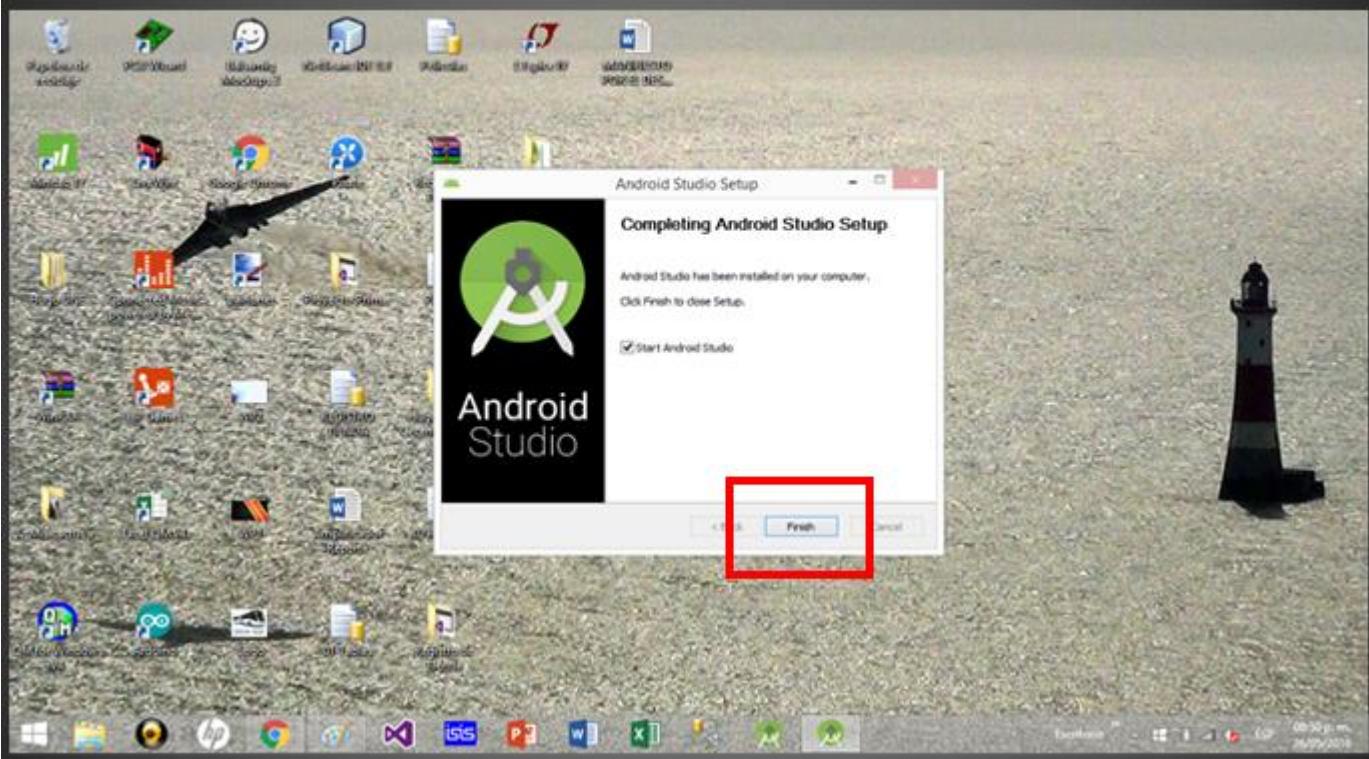
Se recomienda 2GB de RAM, dependiendo del equipo que se tenga.



Esperamos a que se instalen los componentes del SDK para finalizar con el proceso.

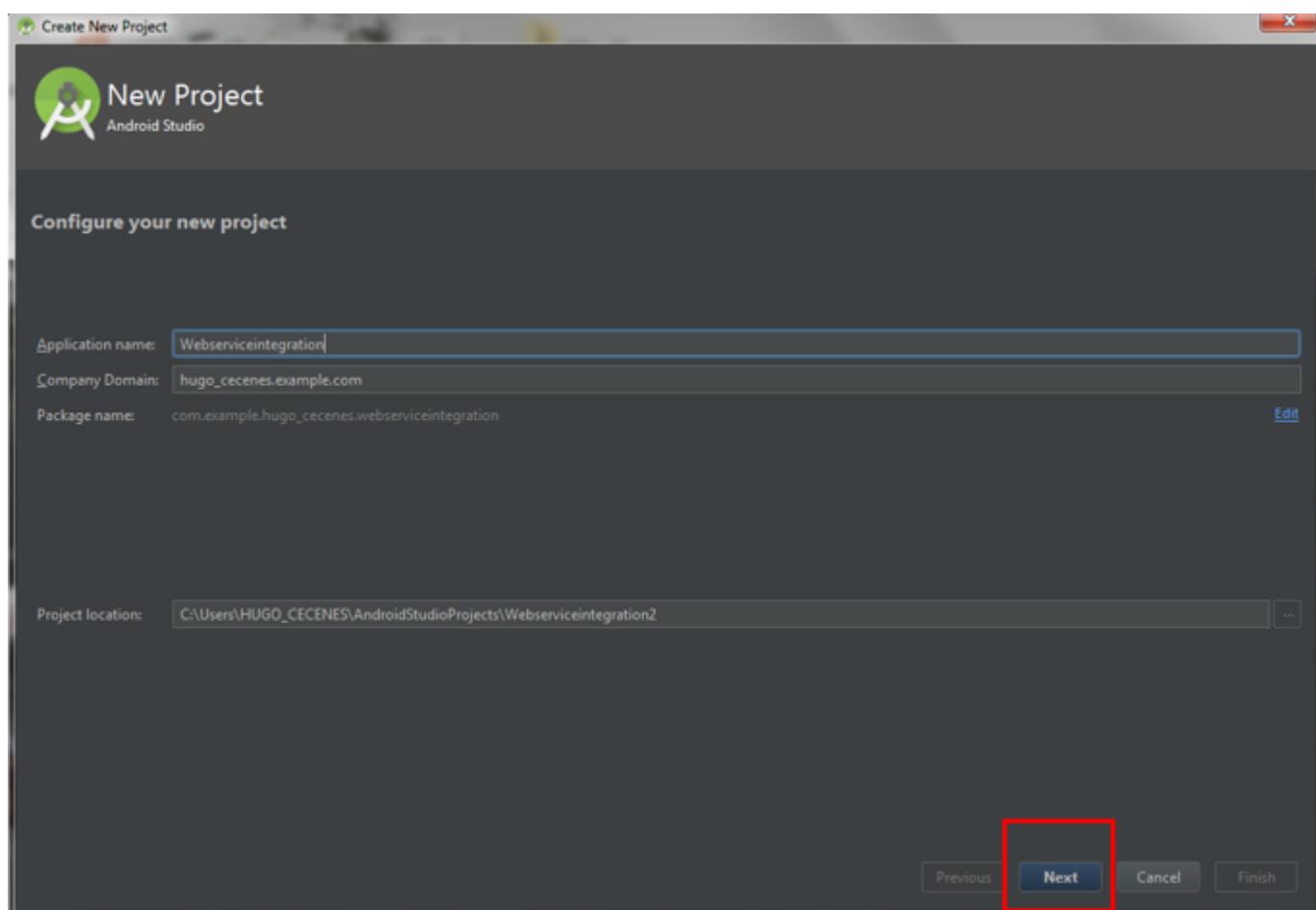
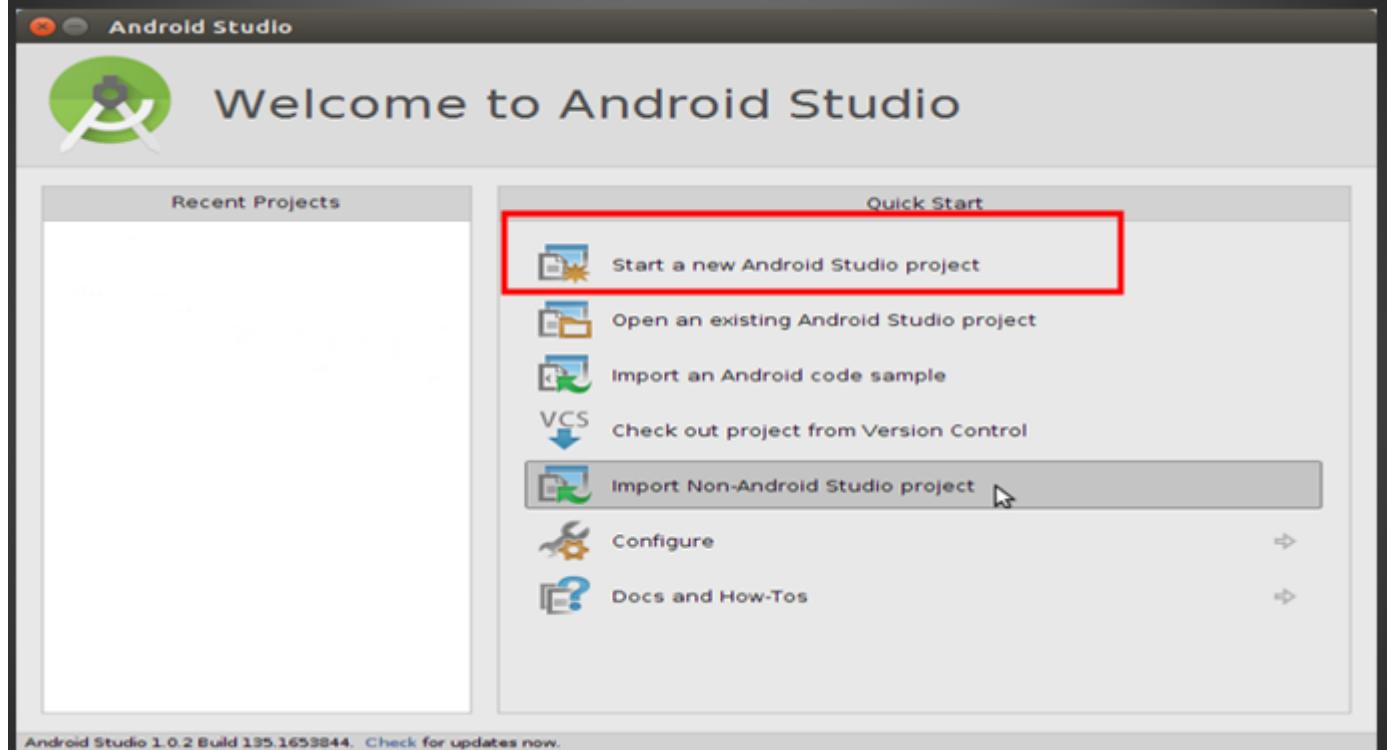


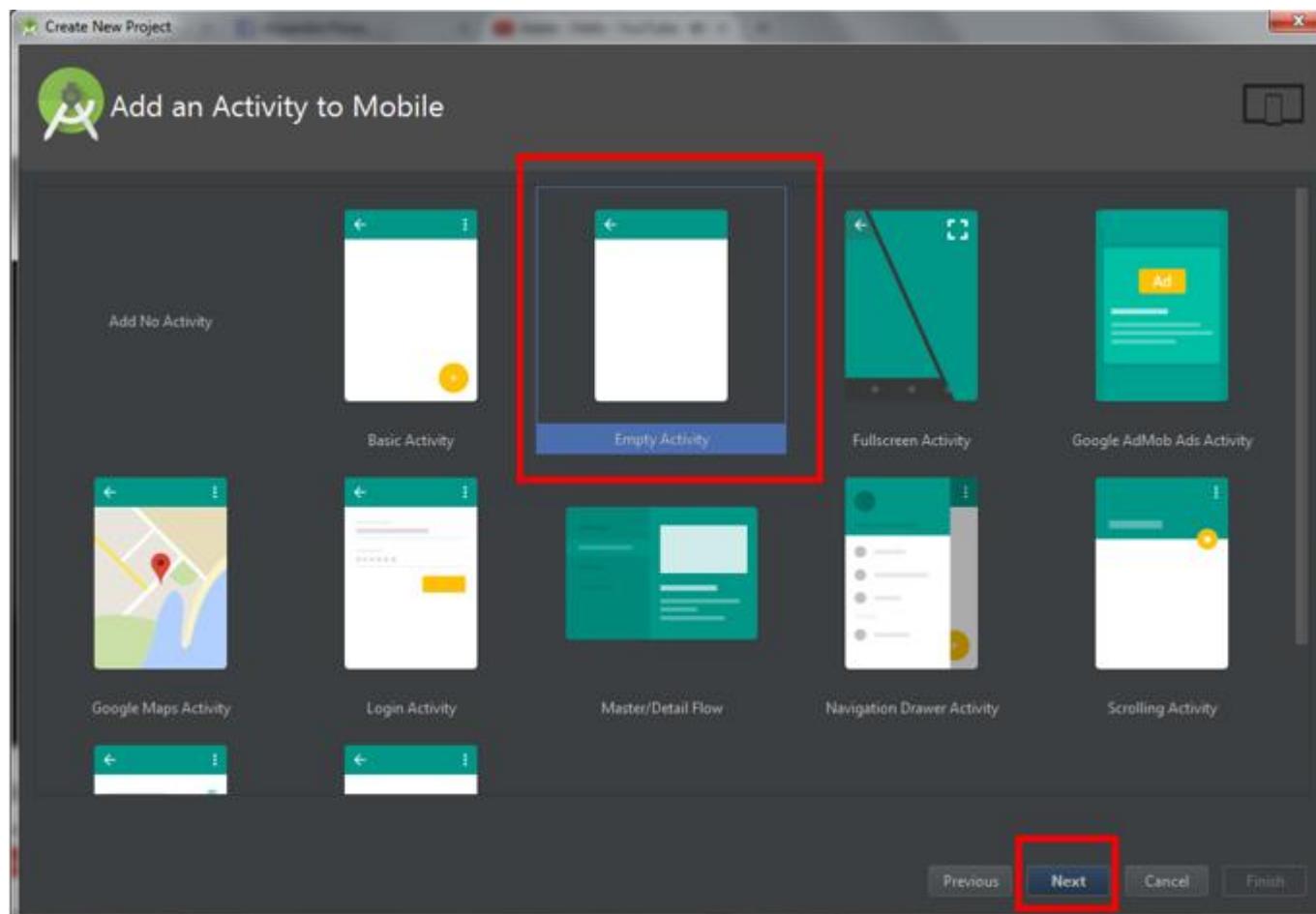
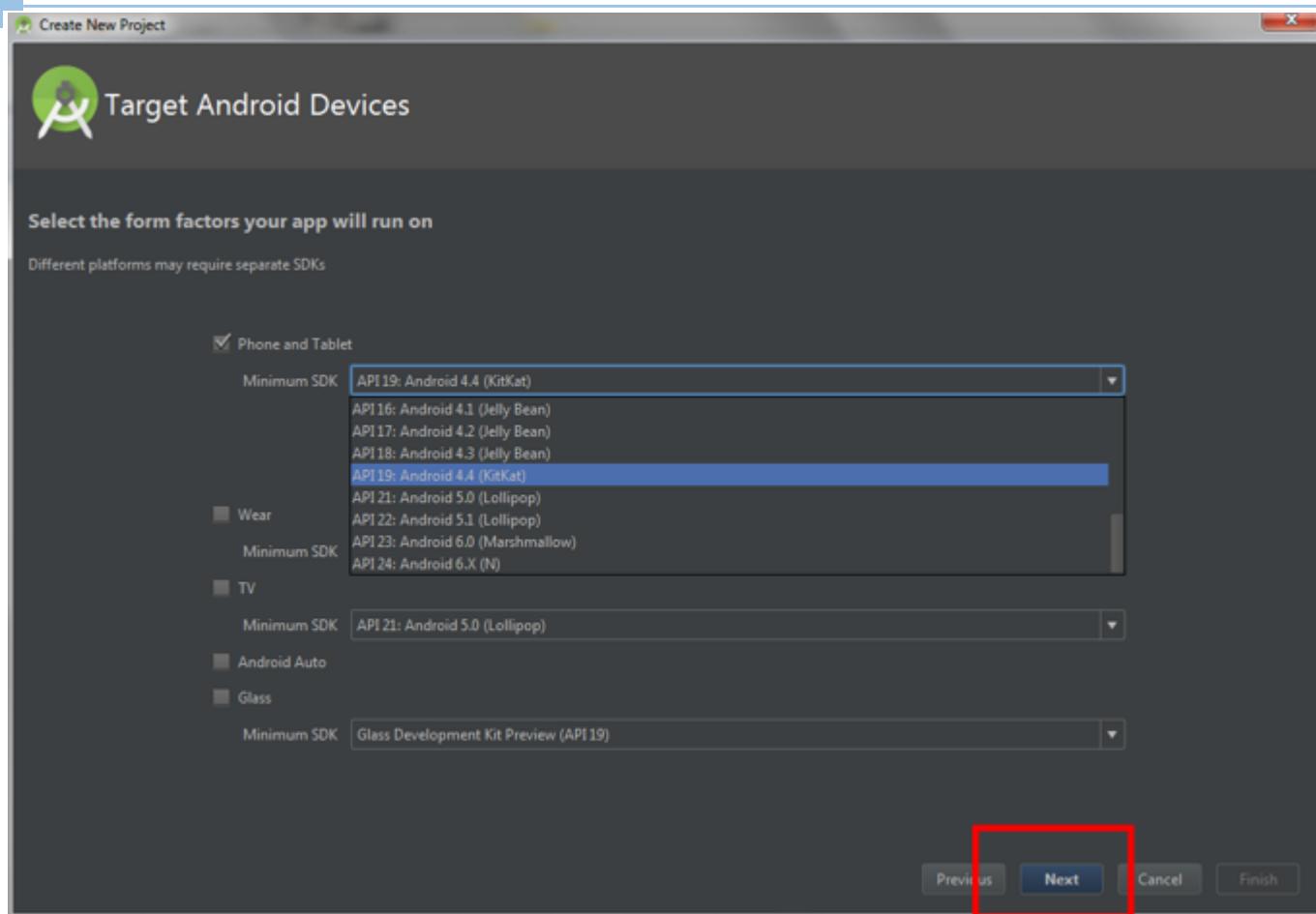
Al finalizar la instalación del Android Studio, le damos clic en “Finish” e iniciamos el programa.

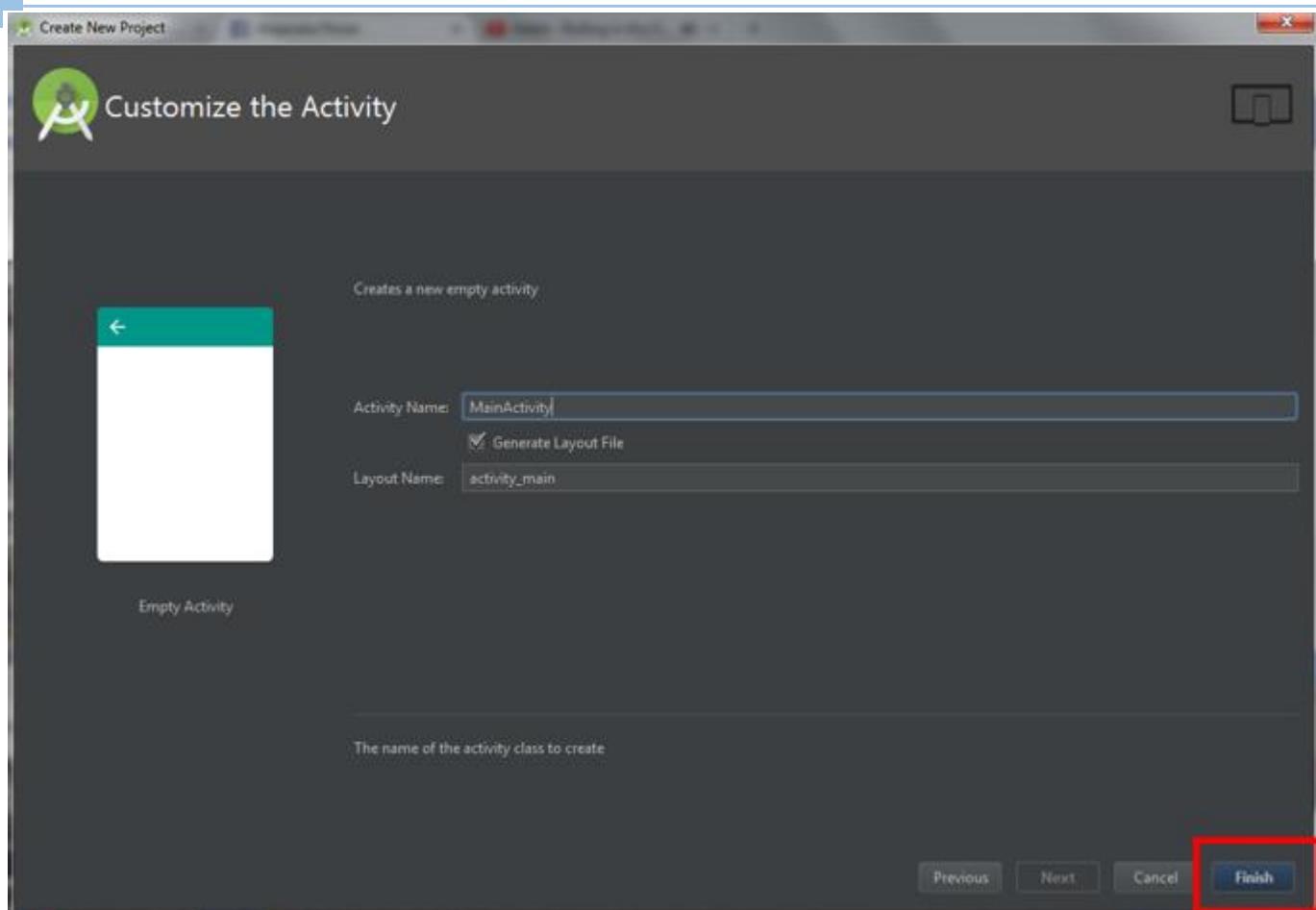


Elaboración de la Aplicación Android

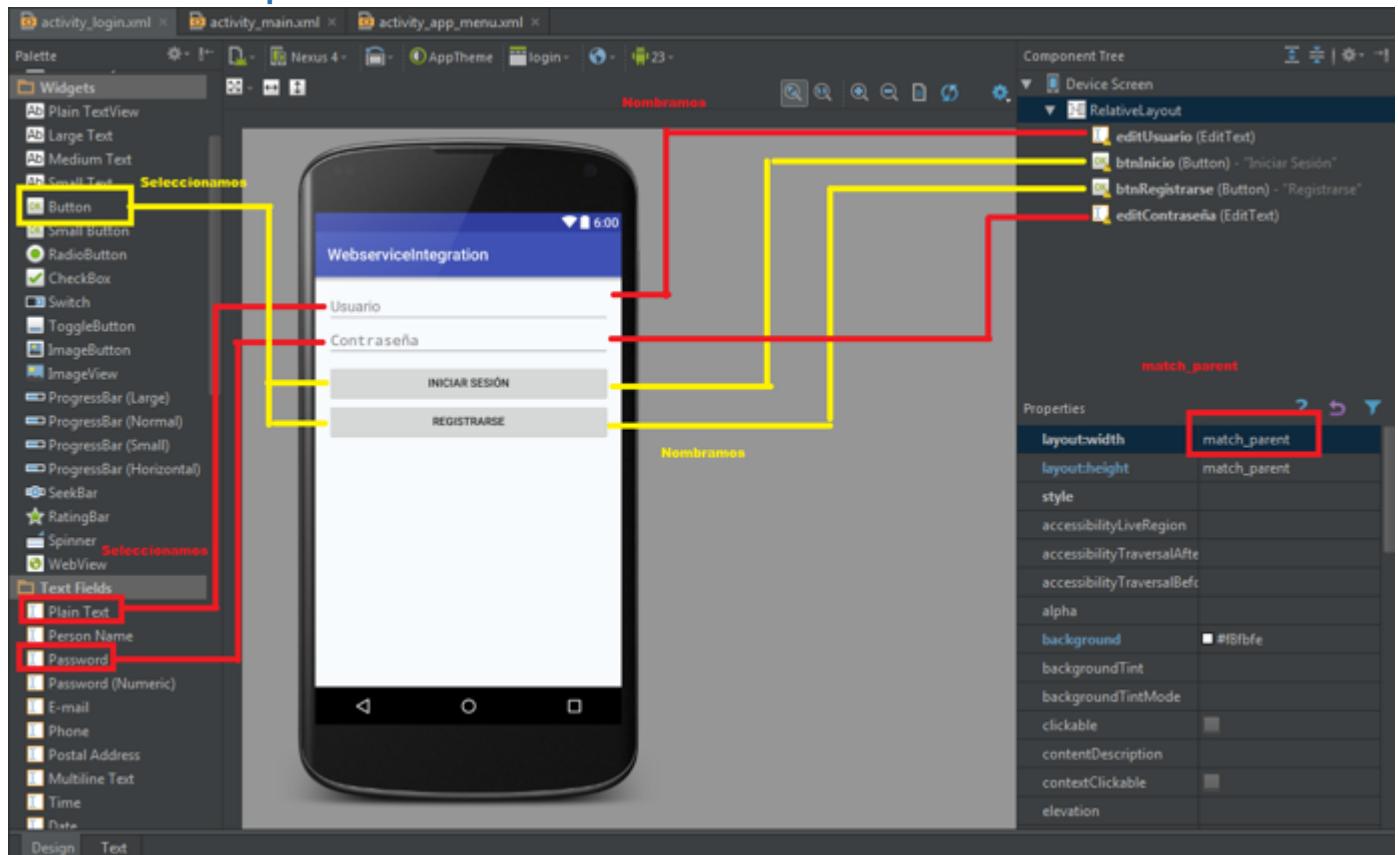
Ya con todo lo necesario instalado, podemos iniciar nuestro proyecto.







Elementos de la Aplicación



Inicio de Sesión

```
activity_login.xml × appMenu.java × login.java × MainActivity.java × activity_main.xml × activity_app_menu.xml ×
package castaneda.com.webserviceintegration;

import ...

public class login extends AppCompatActivity {

    private EditText user;
    private EditText pass;
    private Button login;
    private Button register;
    ProgressDialog progressDialog;

    private String END_POINT_URL = "http://10.0.12.154/android-urban/Registro/login.php"; Este va a ser nuestro enlace al localhost para iniciar sesión.

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        user = (EditText)findViewById(R.id.editUsuario);
        pass = (EditText)findViewById(R.id.editContrasenna); Identificamos los componentes declarados con los de la interfaz.

        progressDialog = new ProgressDialog(this);
        progressDialog.setMessage("Por favor espere...");
        progressDialog.setCancelable(false); Nuestro mensaje de espera en lo que se inicia la sesión.

        register = (Button) findViewById(R.id.btnRegisterarse);
        register.setOnClickListener(v) -> {
            Intent registro = new Intent(getApplicationContext(),MainActivity.class);
            startActivity(registro);
        }

        login = (Button) findViewById(R.id.btnIniciar);
        login.setOnClickListener(v) -> {
            String usuario = user.getText().toString();
            String contrasena = pass.getText().toString();
            if (!isValidPassword(contrasena)) {
                pass.setError("Invalid Password");
                return;
            }
        }
    }

    user = (EditText)findViewById(R.id.editUsuario);
    pass = (EditText)findViewById(R.id.editContrasenna);

    progressDialog = new ProgressDialog(this);
    progressDialog.setMessage("Por favor espere...");
    progressDialog.setCancelable(false);

    register = (Button) findViewById(R.id.btnRegisterarse);
    register.setOnClickListener(v) -> {
        Intent registro = new Intent(getApplicationContext(),MainActivity.class);
        startActivity(registro);
    }

    login = (Button) findViewById(R.id.btnIniciar);
    login.setOnClickListener(v) -> {
        String usuario = user.getText().toString();
        String contrasena = pass.getText().toString(); En esta ocasión sólo se crean 2 cadenas(usuario, contrasena) al Request params y unirlas al código php para seleccionar los datos en el localhost y así iniciar sesión.
        if (!isValidPassword(contrasena)) {
            pass.setError("Invalid Password");
            return;
        }

        RequestParams params = new RequestParams();
        params.put("usuario", usuario);
        params.put("contrasena", contrasena);
        callRegisterWebservice(params); Declaramos el Request params para traer los datos de las cadenas anteriormente declaradas.
    }
}

private void callRegisterWebservice(RequestParams params) {
    progressDialog.show();

    AsyncHttpClient client = new AsyncHttpClient();
    client.post(END_POINT_URL, params, new AsyncHttpResponseHandler());
}

@Override
```

```

private void callRegisterWebservice(RequestParams params) {
    progressDialog.show();
    AsyncHttpClient client = new AsyncHttpClient();
    client.post(END_POINTS_URL, params, new AsyncHttpResponseHandler() {
        @Override
        public void onSuccess(int statusCode, String content) {
            progressDialog.hide();
            try {
                JSONObject jsonResponse = new JSONObject(content);
                String msg = "?";
                if(jsonResponse.getInt("status") == 1){
                    msg = jsonResponse.getString("msg");
                    String user = jsonResponse.getString("user");
                    Intent hiActivity = new Intent(getApplicationContext(), appMenu.class);
                    hiActivity.putExtra("Name", user);
                    startActivity(hiActivity);
                }else{
                    msg = jsonResponse.getString("msg");
                }
                Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_LONG).show();
            }catch(JSONException e){
                Toast.makeText(getApplicationContext(), "Error occurred [Server's JSON response might be invalid]", Toast.LENGTH_LONG).show();
            }
        }

        @Override
        public void onFailure(int statusCode, Throwable error, String content) {
            progressDialog.hide();
            if(statusCode == 404){
                Toast.makeText(getApplicationContext(), "Requested resource not found", Toast.LENGTH_LONG).show();
            }else{
                msg = jsonResponse.getString("msg");
            }
            Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_LONG).show();
        }
    });
}

@Override
public void onFailure(int statusCode, Throwable error, String content) {
    progressDialog.hide();
    if(statusCode == 404){
        Toast.makeText(getApplicationContext(), "Requested resource not found", Toast.LENGTH_LONG).show();
    }else if(statusCode == 500){
        Toast.makeText(getApplicationContext(), "Something went wrong at server end", Toast.LENGTH_LONG).show();
    }else{
        Toast.makeText(getApplicationContext(), "Unexpected Error occurred! [Most common Error: Device might not be connected to Internet or remote server is not up and running]", Toast.LENGTH_LONG).show();
    }
}
};

private boolean isValidPassword(String contraseña) {
    if (contraseña != null && contraseña.length() > 2) {
        return true;
    }
    return false;
}

```

Aquí mandamos a llamar al mensaje de espera que declaramos.

Con ésto sincronizamos el código con el enlace al localhost.

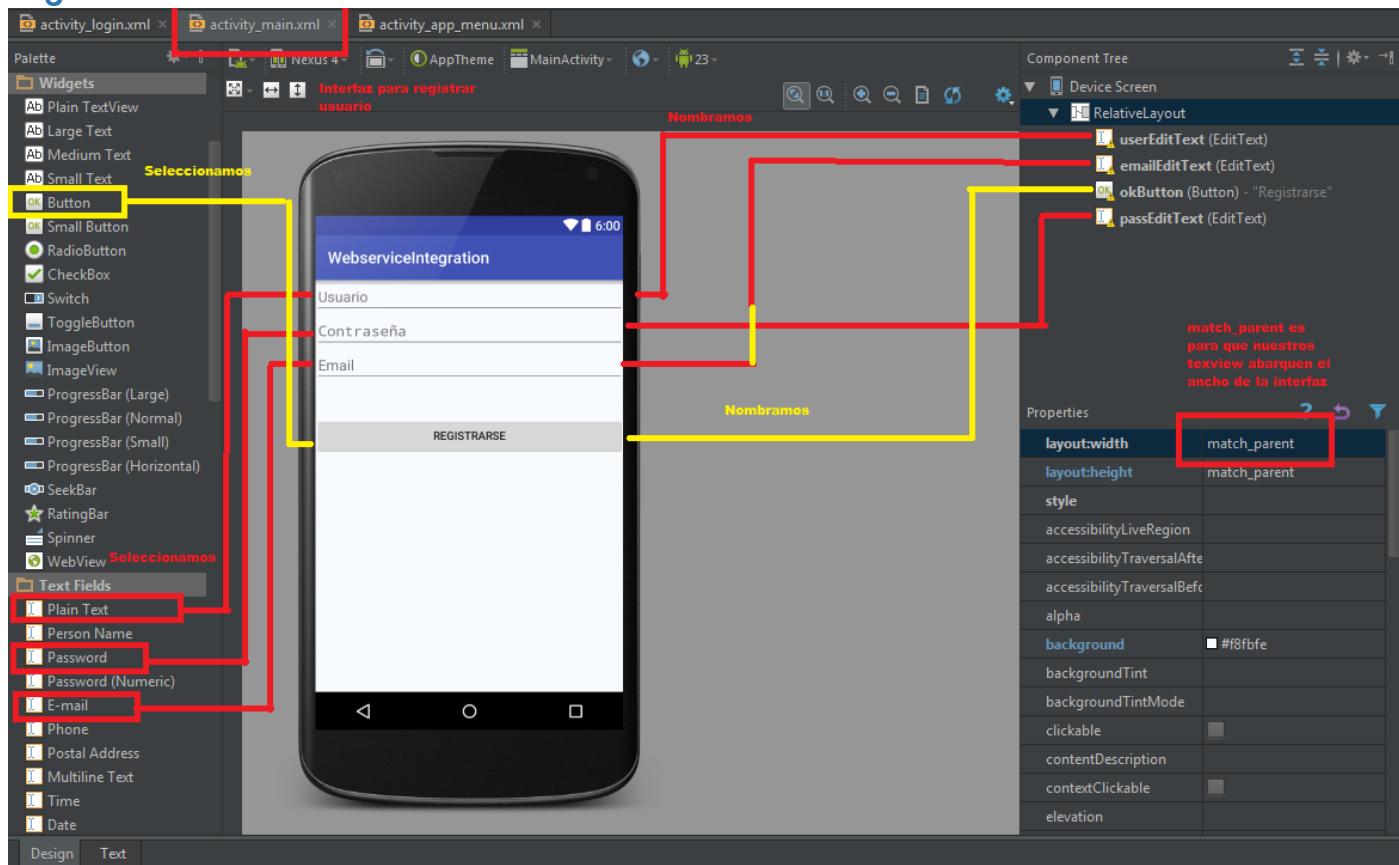
Aquí el código se conecta con el php para seleccionar las columnas que se encuentran en el localhost y así poder iniciar sesión.

Además que se hace una cadena que toma el nombre d eusuario y reemplaza el simbolo "?" en el TextView de appMenu.

Por el contrario, éste código es cuando la selección de los datos en el localhost fallan ya sea por falta de conexión o datos inexistentes.

Éste es el método para validar que la contraseña no pueda ser menor que 2 dígitos.

Registro



```
import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;
import java.util.regex.Pattern;

public class MainActivity extends AppCompatActivity {

    private EditText userEditText;
    private EditText passEditText;
    private EditText emailEditText;
    private Button okButton;
    ProgressDialog progressDialog;

    private String URL_REGISTER = "http://192.168.1.10/webservices/ultimo/registrar.php";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        userEditText = (EditText) findViewById(R.id.userEditText);
        passEditText = (EditText) findViewById(R.id.passEditText);
        emailEditText = (EditText) findViewById(R.id.emailEditText);

        progressDialog = new ProgressDialog(this);
        progressDialog.setMessage("Por favor espere...");
        progressDialog.setCancelable(false);

        okButton = (Button) findViewById(R.id.okButton);
        okButton.setOnClickListener(new View.OnClickListener() {
            String userText = userEditText.getText().toString();
            String contraseña = passEditText.getText().toString();
            if (!isValidPassword(contraseña)) {
                passEditText.setError("Invalid Password");
                return;
            }

            String email = emailEditText.getText().toString();
            if (!isValidEmail(email)) {
                emailEditText.setError("Invalid Email");
                return;
            }
        });
    }

    private boolean isValidEmail(String email) {
        String emailRegex = "^[\\w-_\\.]+@[\\w-_\\.]+\\.[\\w]{2,3}$";
        Pattern pattern = Pattern.compile(emailRegex);
        if (pattern.matcher(email).matches())
            return true;
        else
            return false;
    }

    private boolean isValidPassword(String password) {
        String passwordRegex = "^(?=.*[A-Z])(?=.*[a-z])(?=.*[0-9]).{8,}$";
        Pattern pattern = Pattern.compile(passwordRegex);
        if (pattern.matcher(password).matches())
            return true;
        else
            return false;
    }
}
```

The code highlights several sections with red boxes:

- Variable declarations: userEditText, passEditText, emailEditText, okButton, progressDialog.
- URL definition: URL_REGISTER.
- ProgressDialog setup: progressDialog = new ProgressDialog(this); progressDialog.setMessage("Por favor espere..."); progressDialog.setCancelable(false);
- OkButton click listener: okButton.setOnClickListener(new View.OnClickListener() { ... }).
- Text input validation logic: isValidEmail and isValidPassword methods.

Annotations in the code:

- Se declaran las variables para obtener el resultado de los tres edittext.
- Se identifican las componentes con las que se creará la interfaz.
- Se agregan datos adicionales para que aparezca un mensaje de espera mientras se realizan las peticiones.

```

        userEditText = (EditText) findViewById(R.id.userEditText);
        passEditText = (EditText) findViewById(R.id.passEditText);
        emailEditText = (EditText) findViewById(R.id.emailEditText);

        progressDialog = new ProgressDialog(this);
        progressDialog.setMessage("Por favor espere...");
        progressDialog.setCancelable(false);

        okButton = (Button) findViewById(R.id.okButton);
        okButton.setOnClickListener(v) -> {
            String usuario = userEditText.getText().toString();
            String contraseña = passEditText.getText().toString();
            if (!isValidPassword(contraseña)) {
                passEditText.setError("Invalid Password");
                return;
            }

            String email = emailEditText.getText().toString();
            if (!isValidEmail(email)) {
                emailEditText.setError("Invalid Email");
                return;
            }

            RequestParams params = new RequestParams();
            params.put("usuario", usuario);
            params.put("contraseña", contraseña);
            params.put("email", email);
            callRegisterWebservice(params);

            Intent back = new Intent(getApplicationContext(), login.class);
            startActivity(back);
        };
    };

private void callRegisterWebservice(RequestParams params)
{
    progressDialog.show();
}

```

Proceso para insertar los parámetros al momento de dar click en el botón okButton.

Creamos 3 cadenas (usuario, contraseña, email) para mandarlas al Request params y así unirlas con el código php para insertar en el localhost. Para las cadenas de contraseña y email se mandan a llamar 2 métodos que se encuentran al final del código para validar los Editext.

Declaramos el Request params para traer los datos que están en las cadenas anteriormente declaradas.

Código para regresar a la interfaz principal al momento de hacer click en el botón okButton.

```

private void callRegisterWebservice(RequestParams params)
{
    progressDialog.show(); Aquí mandamos a llamar al mensaje de espera que declaramos.
    AsyncHttpClient client = new AsyncHttpClient(); Con esto sincronizamos el código con el enlace al localhost.
    client.post(EMO_POINT_URL, params, new AsyncHttpResponseHandler()
    {
        @Override
        public void onSuccess(int statusCode, String content) {
            progressDialog.hide();
            try {
                JSONObject jsonResponse = new JSONObject(content);
                String msg = "";
                if(jsonResponse.getInt("status") == 1){
                    msg = jsonResponse.getString("msg");
                }else{
                    msg = jsonResponse.getString("msg");
                }
                Toast.makeText(getApplicationContext(), msg, Toast.LENGTH_LONG).show();
            }catch(JSONException e){
                Toast.makeText(getApplicationContext(), "Error occurred [Server's JSON response might be invalid]", Toast.LENGTH_LONG).show();
            }
        }

        @Override
        public void onFailure(int statusCode, Throwable error, String content) {
            progressDialog.hide();
            if(statusCode == 404){
                Toast.makeText(getApplicationContext(), "Requested resource not found", Toast.LENGTH_LONG).show();
            } else if(statusCode == 500){
                Toast.makeText(getApplicationContext(), "Something went wrong at server end", Toast.LENGTH_LONG).show();
            } else{
                Toast.makeText(getApplicationContext(), "Unexpected Error occurred! [Most common Error: Device might not be connected to Internet or remote server is not up and running]", Toast.LENGTH_LONG).show();
            }
        }
    });
}

public void onFailure(int statusCode, Throwable error, String content) {
    progressDialog.hide();
    if(statusCode == 404){
        Toast.makeText(getApplicationContext(), "Requested resource not found", Toast.LENGTH_LONG).show();
    } else if(statusCode == 500){
        Toast.makeText(getApplicationContext(), "Something went wrong at server end", Toast.LENGTH_LONG).show();
    } else{
        Toast.makeText(getApplicationContext(), "Unexpected Error occurred! [Most common Error: Device might not be connected to Internet or remote server is not up and running]", Toast.LENGTH_LONG).show();
    }
}

private boolean isValidEmail(String email) {
    String EMAIL_PATTERN = "[a-zA-Z0-9+\\n]+([\\.,][a-zA-Z0-9+\\n]+)*@[a-zA-Z0-9+\\n]+([\\.,][a-zA-Z0-9+\\n]+)*\\.[a-zA-Z]{2,}[\\n]*";
    Pattern pattern = Pattern.compile(EMAIL_PATTERN);
    Matcher matcher = pattern.matcher(email);
    return matcher.matches();
}

private boolean isValidPassword(String contraseña) {
    if (contraseña != null & contraseña.length() > 2) {
        return true;
    }
    return false;
}

```

Este es el código completo para conectarse entre el Android Studio con el PHP y el localhost.

Aquí el código se conecta con el php para insertar los parámetros por medio de cadenas.

Per el contrario, este código es cuando el registro falla ya sea por algún error de conexión o de código y manda mensajes para el usuario.

Este es el método para validar el correo y acepte distintos símbolos que se utilizan.

Este es el método para validar que la contraseña no pueda ser menor de 2 dígitos.

Menu

