**Tarea 7**

|  |  |
| --- | --- |
| **Nombre:** Univ. Mamani Chavez Carla Vanesa | **CI:** 9124602 LP  **Paralelo:** Martes |
| **Docente :** Lic. Gallardo Portanda Franz Ramiro | **Fecha :** 14/04/2020 |

1. **SERVICIO DE TIEMPO**

**U**sando el API de sockets de Java desarrolle una aplicación cliente-servidor para proporcionar la fecha y hora. El servidor atenderá peticiones en el puerto 8888 y por cada petición recibida devolverá como respuesta: la fecha y hora junto con la información del socket del cliente al cual fue enviada la respuesta (dirección IP y numero de puerto). El programa cliente deberá imprimir la respuesta del servidor.

Deberá usar el protocolo UDP para la interacción entre su programa cliente y su programa servidor.

* **EL SERVIDOR**

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.nio.charset.StandardCharsets;

**import** java.util.Date;

**public class** ServidorUdp {

**private** DatagramSocket socketUDP;

**private** DatagramPacket paqrecibido;

**private** DatagramPacket paqparaenviar;

**private int** puerto;

**private byte** msg[] = **new byte**[1024]; **public** ServidorUdp(**int** p) {

puerto = p;

}

**public void** iniciar() {

## try {

1024);

socketUDP = **new** DatagramSocket(puerto); System.*out*.println(" --- SERVIDOR UDP INICIADO --- "); System.*out*.println(" --- Esperando Cliente --- "); **while** (**true**) {

paqrecibido = **new** DatagramPacket(**new byte**[1024],

socketUDP.receive(paqrecibido); String recibidos[]=(**new**

String(paqrecibido.getData(),StandardCharsets.*UTF\_8*)).split(" ");

System.*out*.println(**new** String(paqrecibido.getData(),StandardCharsets.*UTF\_8*));

**if**(recibidos[0]!=**null**){ **if**(recibidos[0].equals("servicio")){

// Se prepara el mensaje a enviar con el dia de la semana ,fecha y hora del servidor

String message = **new** String("Dia de la semana , fecha , hora : " + **new** Date());

msg = message.getBytes();

}

**else**{ mensajeRespuesta="Solicitud\_invalida";

}

}

String msg=mensajeRespuesta.getBytes();

System.*out*.println("Ha llegado una peticion \n"); System.*out*.println("Procedente de : "+

paqrecibido.getAddress());

System.*out*.println("En el puerto : " +

paqrecibido.getPort());

}

System.*out*.println("Sirviendo la petición"); paqparaenviar = **new** DatagramPacket(msg, msg.length, paqrecibido.getAddress(), paqrecibido.getPort()); socketUDP.send(paqparaenviar);

} **catch** (Exception e) { e.printStackTrace();

} **finally** {

finalizar();

}

}

**public void** finalizar() {

## try {

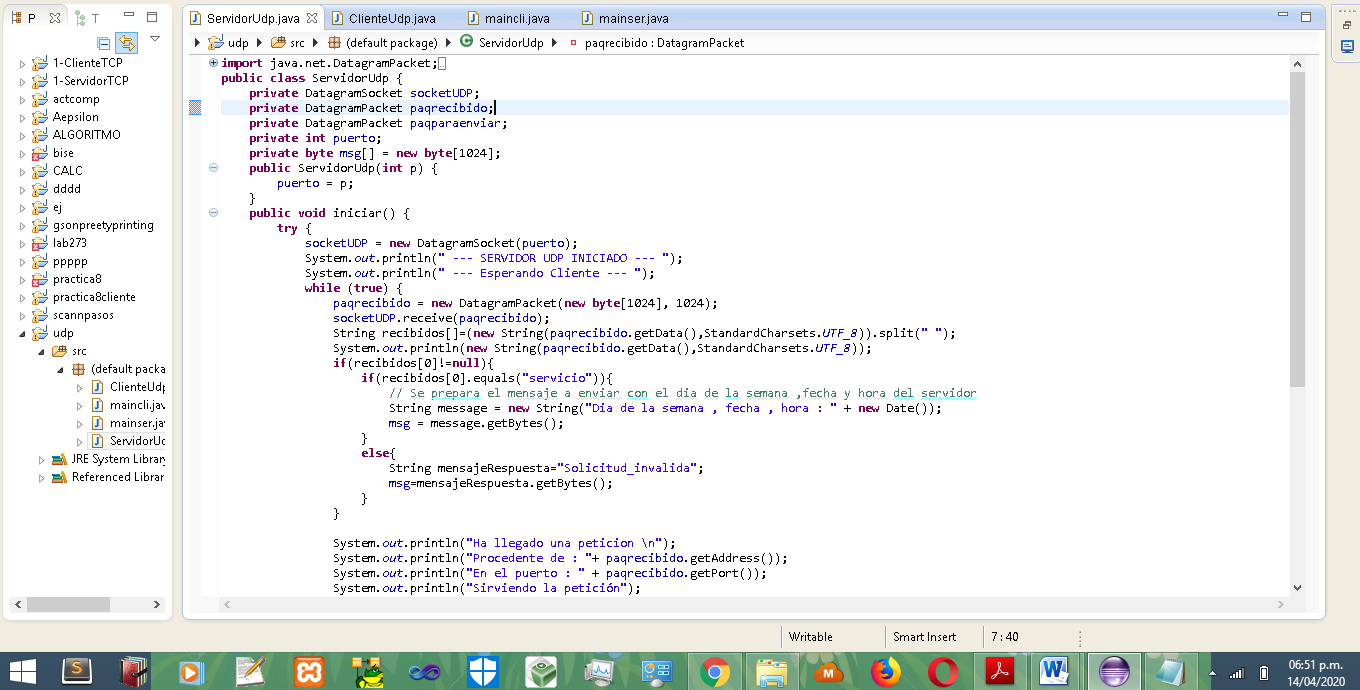
socket0UDP.close(); System.*out*.print("Conexion Finalizada!!");

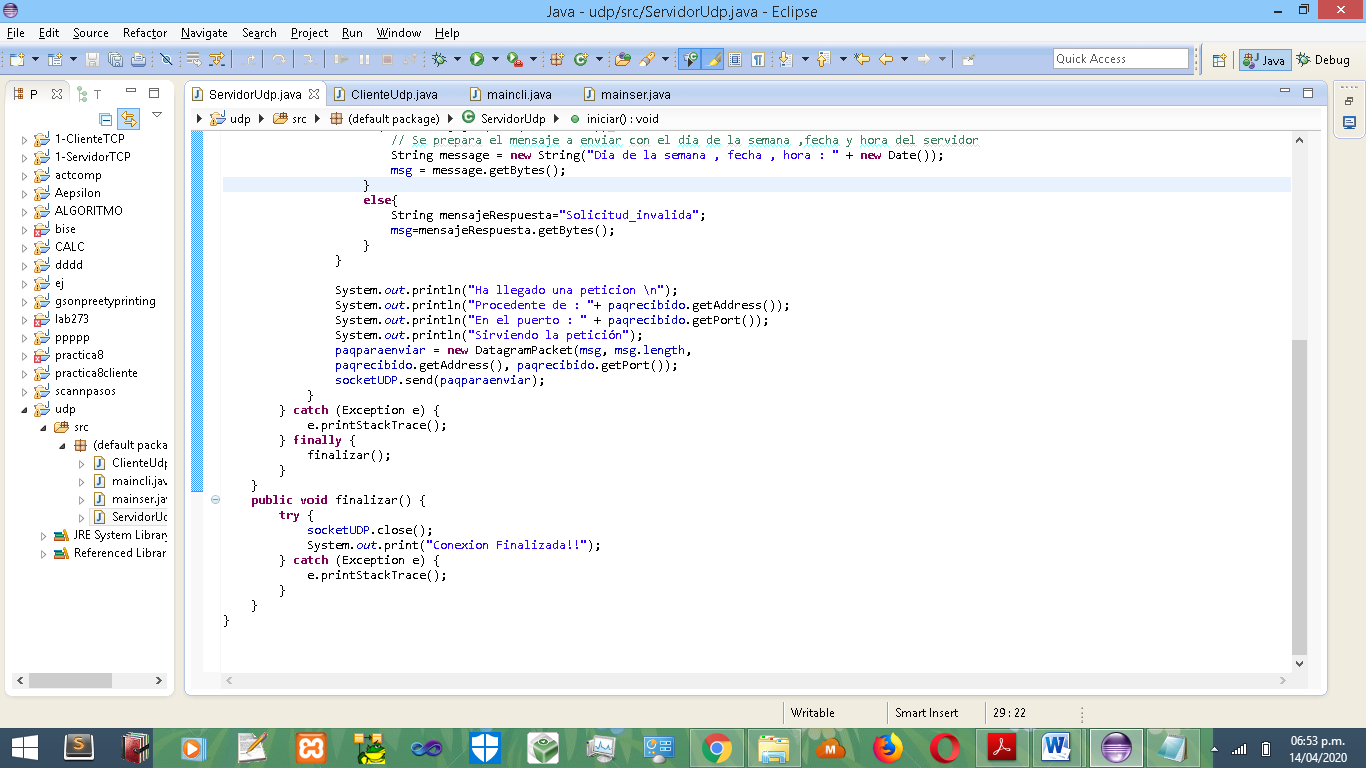
} **catch** (Exception e) { e.printStackTrace();

}

}

}





* **CLASE QUE EJECUTA EL SERVIDOR**

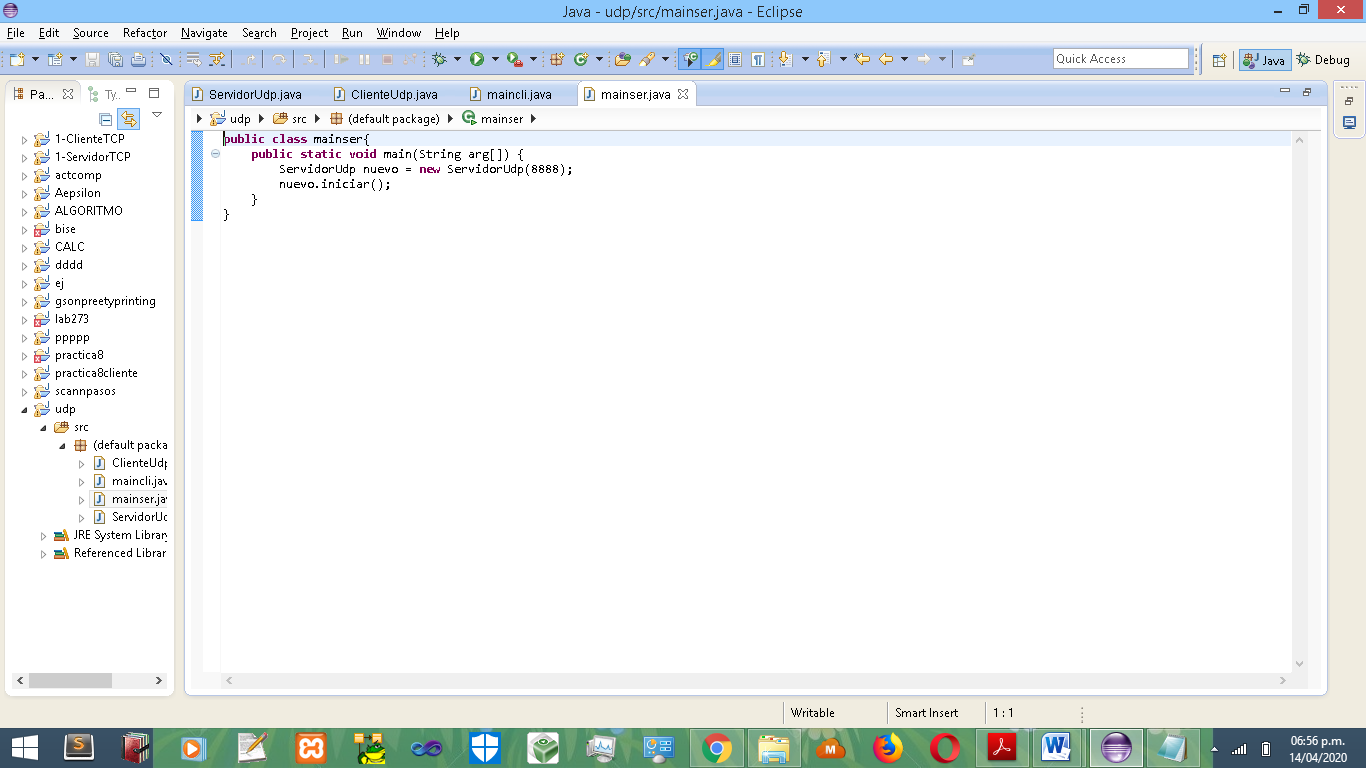
**public class** mainser{

**public static void** main(String arg[]) { ServidorUdp nuevo = **new** ServidorUdp(8888);

nuevo.iniciar();

}

}



* **EL CLIENTE**

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetAddress;

**import** java.net.UnknownHostException;

**import** java.util.Scanner;

**public class** ClienteUdp {

**private int** puerto = 8888;

**private** InetAddress servidorDestino;

**private** Scanner lector=**new** Scanner(System.*in*);

**private** DatagramSocket socketUdp;

**private** DatagramPacket paqrecibido;

**private** DatagramPacket paqparaenviar; **private byte** mensaje[] = **new byte**[1024];

**public** ClienteUdp(String a, **int** p) **throws** UnknownHostException { servidorDestino = InetAddress.*getByName*(a);

puerto = p;

}

**public void** iniciar() **throws** Exception { socketUdp = **new** DatagramSocket();

System.*out*.println("Introduzca su peticion : "); String peticion=**new** String(lector.nextLine()); mensaje=peticion.getBytes();

paqparaenviar = **new** DatagramPacket( mensaje, mensaje.length, servidorDestino,

puerto); socketUdp.send(paqparaenviar);

paqrecibido = **new** DatagramPacket(**new byte**[1024], 1024); socketUdp.receive(paqrecibido);

String received = **new** String(paqrecibido.getData()); System.*out*.println("\nDATOS DEL DATAGRAMA: " + received); finalizar();

}

**public void** finalizar() {

## try {

socketUdp.close(); System.*out*.print("Conexion Finalizada!!");

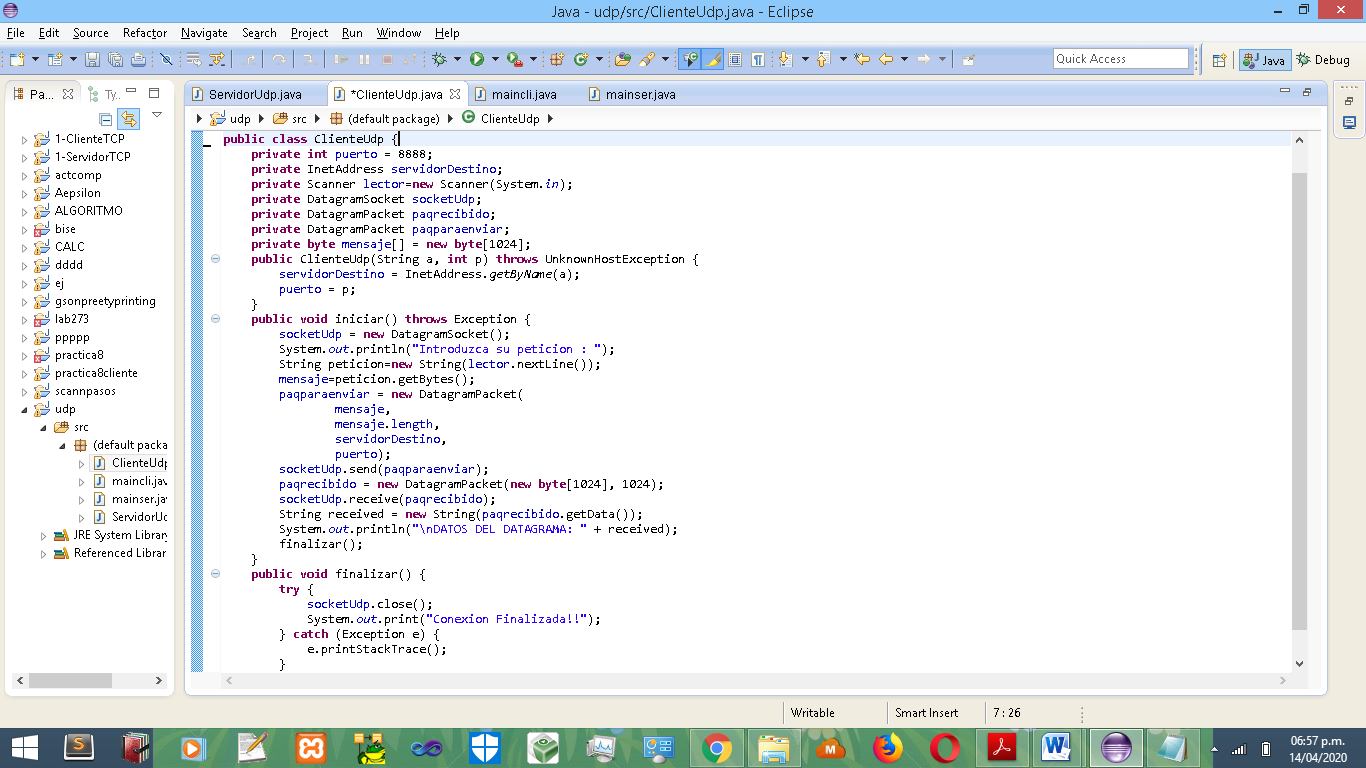
} **catch** (Exception\_e)

{ e.printStackTrace();

}

}

}



* **CLASE QUE EJECUTA EL CLIENTE**

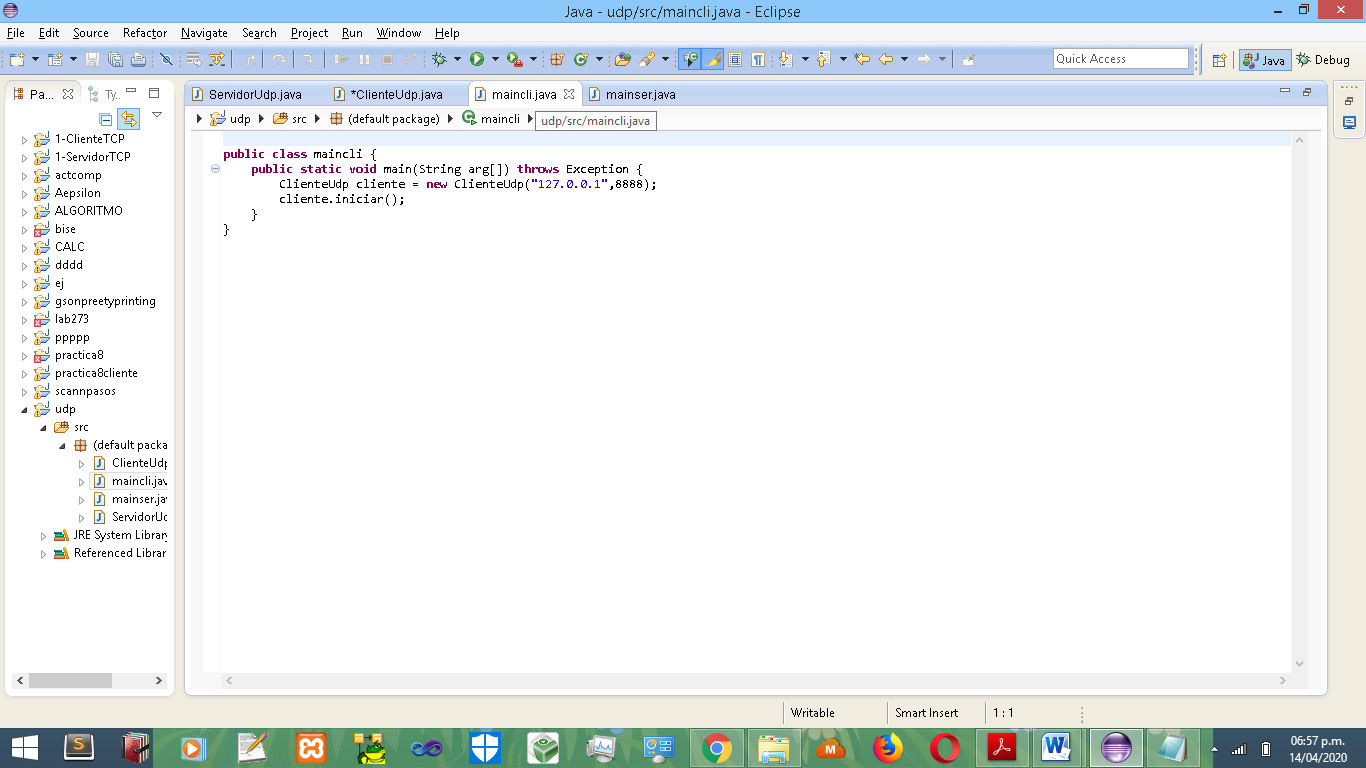
**public class** maincli {

**public static void** main(String arg[]) **throws** Exception { ClienteUdp cliente = **new** ClienteUdp("127.0.0.1",8888);

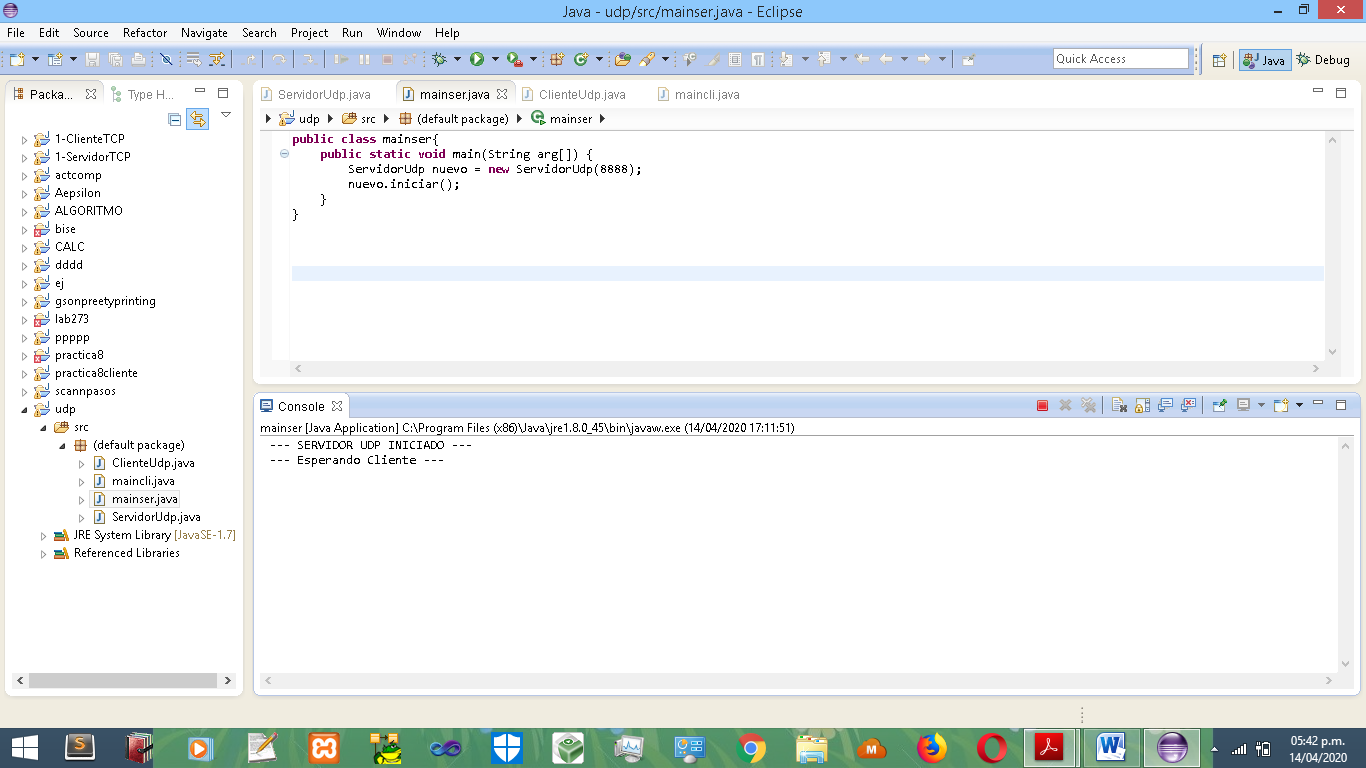
cliente.iniciar();

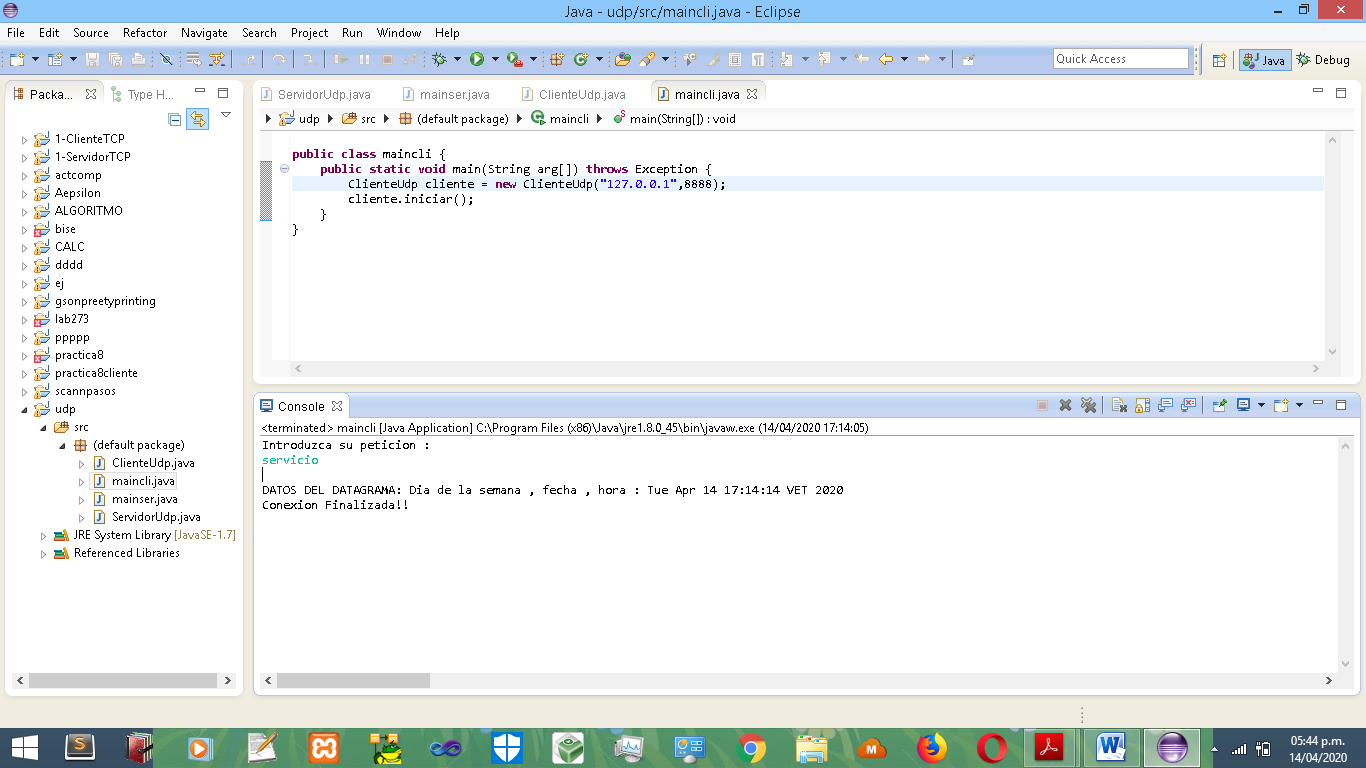
}

}

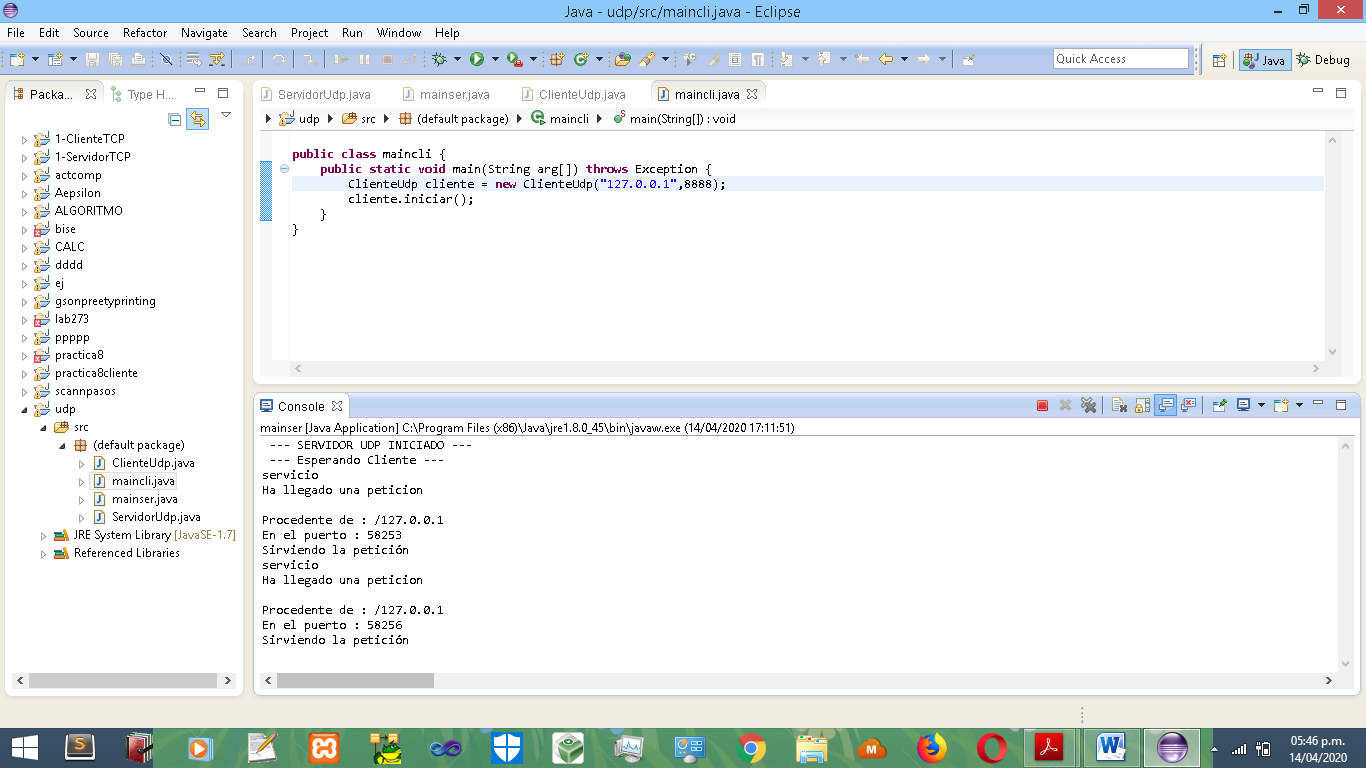


Ejecución de los programas

* **Servidor : Se muestra en consola que el servidor udp iniciado** 
* Cliente : Se envía el datagrama para conectarse con el servidor



* Verificación que llego el mensaje del cliente por parte del servidor, por lo tanto se demuestra el objetivo del ejercicio.



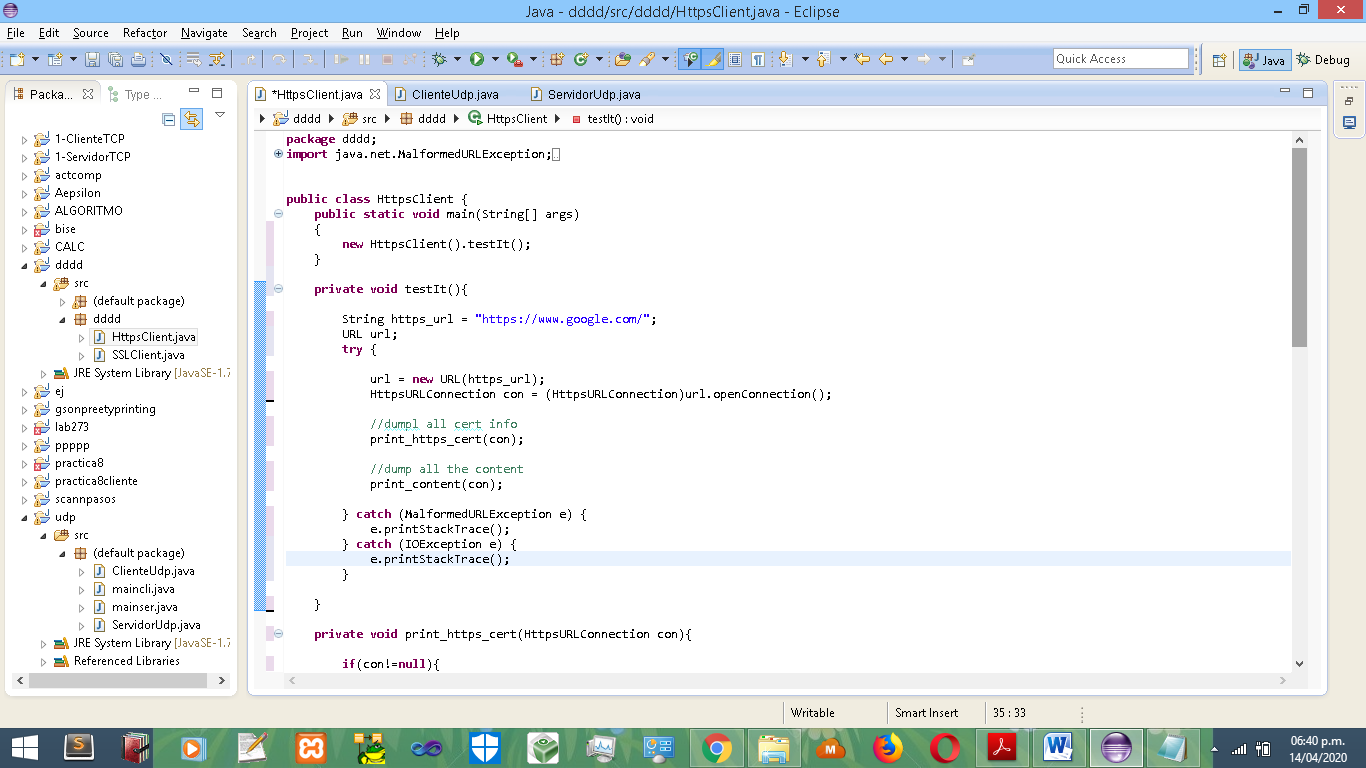
1. **CLIENTE HTTP**

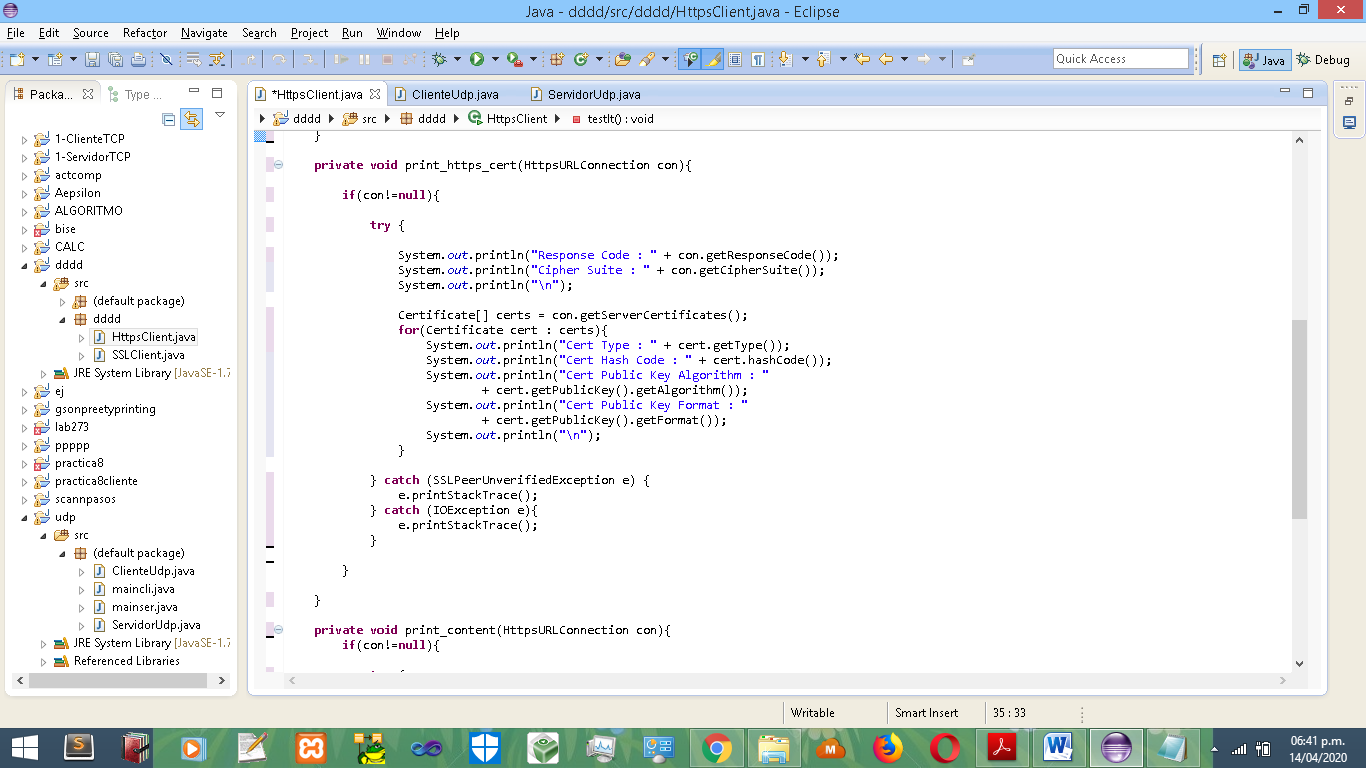
Desarrolle un cliente que se conecte a un sitio web y despliegue la respuesta que retorna el servidor (encabezados HTTP y las 15 primeras líneas de datos recibidos). Para este ejercicio use la clase URL de Java. La clase URL representa una abstracción de mayor nivel que el de los sockets y provee métodos asociados que le permitirán obtener la información que se le solicita.

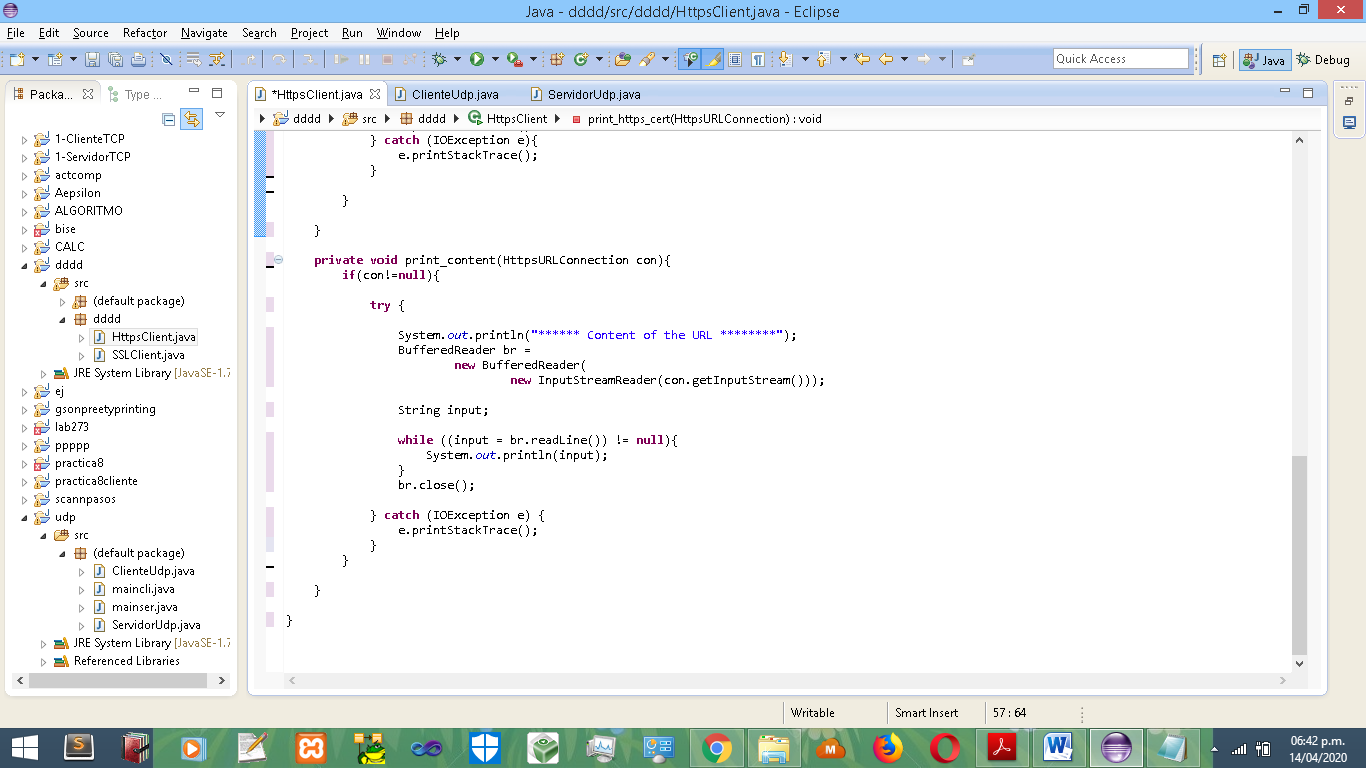
**COMO SE UTILIZA HTTPSURLCONNECTION**

HttpURLConnection sirve para interactuar con formularios web y para acceder y manipular URLs, para definir una URL en Java usamos la clase URL lo cual por igual tiene algunos métodos que podrían ser útiles.

* DOCUMENTANDO CODIGO FUENTE

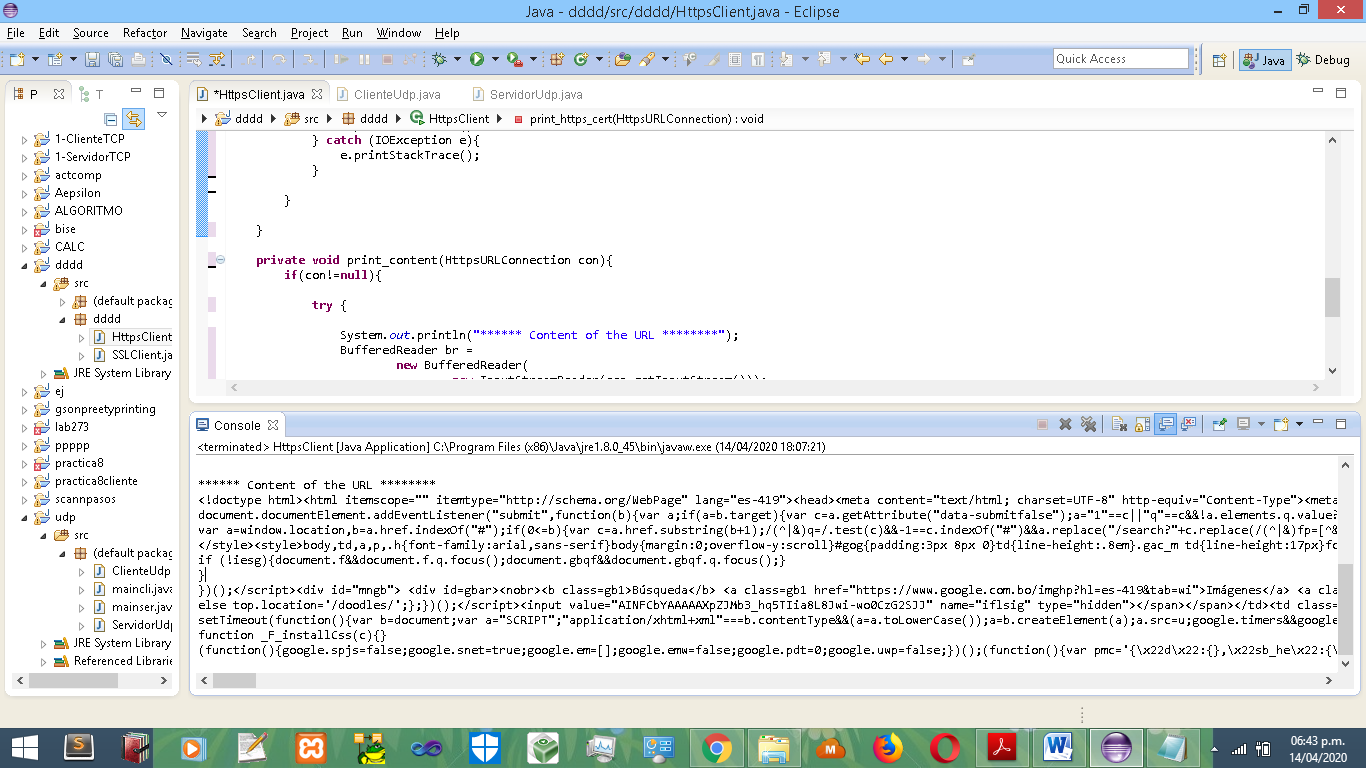


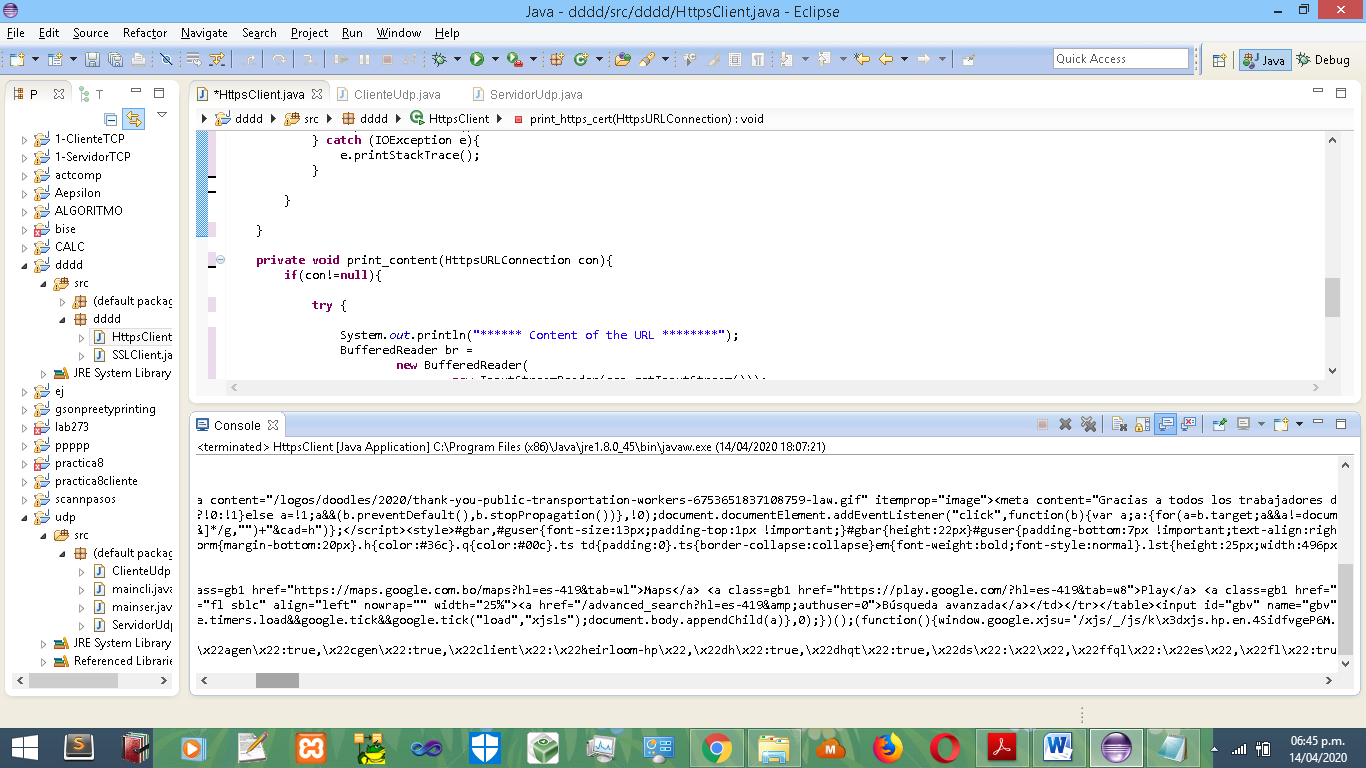


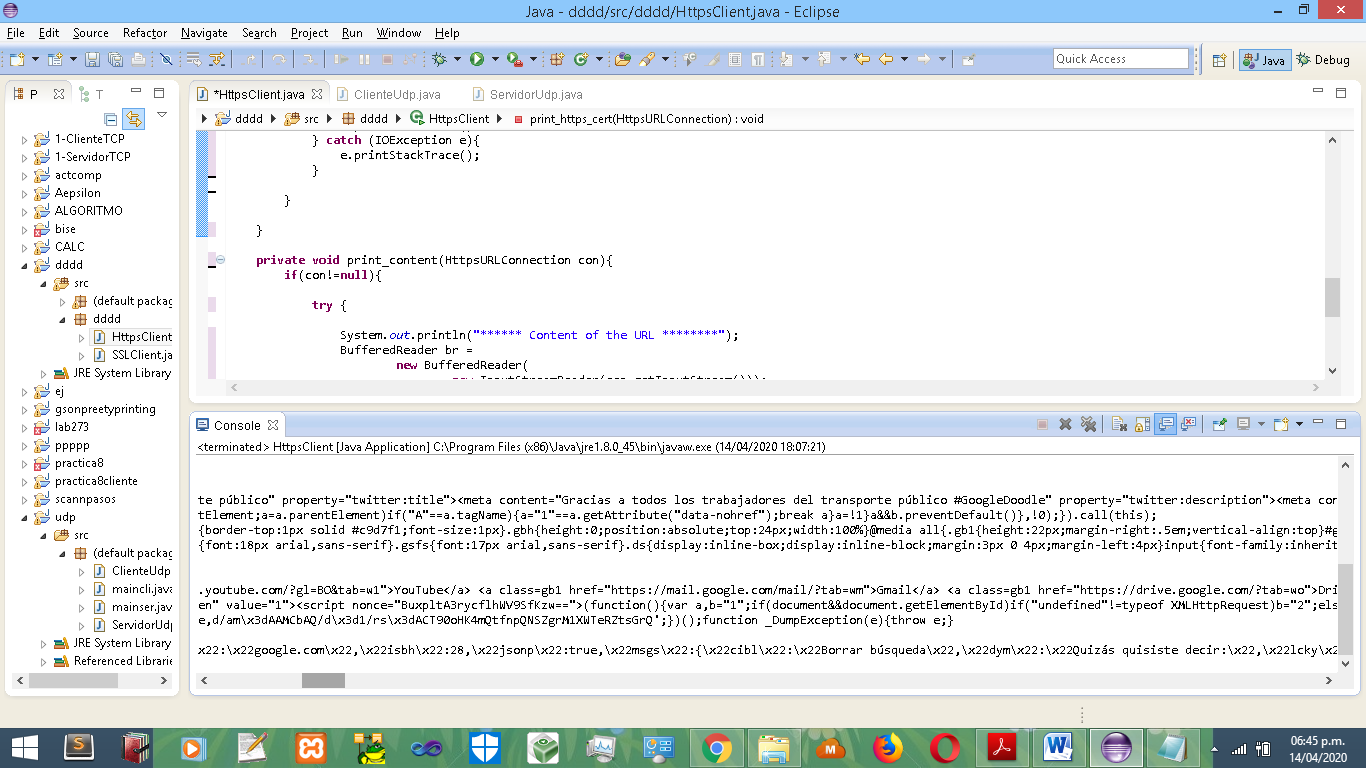


* EJECUCION DEL PROGRAMA

Se muestra en consola todos los datos obtenidos del sitio web googele.com como se muestra en las anteriores imágenes.







* EJECUCION DEL PROGRAMA

Verificación de la url del contenido.

En nuestro caso. El sitio web, para verificar la conexión y se obtuvo el resultado como mostramos en ejecución del programa.

