

Home (<http://www.java2s.com>)  
Java  
([/Code/Java/CatalogJava.htm](#))  
2D Graphics GUI ([/Code/Java/2D-  
Graphics-GUI/Catalog2D-  
Graphics-GUI.htm](#))  
3D  
([/Code/Java/3D/Catalog3D.htm](#))  
Advanced Graphics  
([/Code/Java/Advanced-  
Graphics/CatalogAdvanced-  
Graphics.htm](#))  
Ant  
([/Code/Java/Ant/CatalogAnt.htm](#))  
Apache Common  
([/Code/Java/Apache-  
Common/CatalogApache-  
Common.htm](#))  
Chart  
([/Code/Java/Chart/CatalogChart.ht  
m](#))  
Class  
([/Code/Java/Class/CatalogClass.ht  
m](#))  
Collections Data Structure  
([/Code/Java/Collections-Data-  
Structure/CatalogCollections-Data-  
Structure.htm](#))  
Data Type ([/Code/Java/Data-  
Type/CatalogData-Type.htm](#))  
Database SQL JDBC  
([/Code/Java/Database-SQL-  
JDBC/CatalogDatabase-SQL-  
JDBC.htm](#))  
Design Pattern  
([/Code/Java/Design-  
Pattern/CatalogDesign-  
Pattern.htm](#))  
Development Class  
([/Code/Java/Development-  
Class/CatalogDevelopment-  
Class.htm](#))  
EJB3  
([/Code/Java/EJB3/CatalogEJB3.ht  
m](#))  
Email  
([/Code/Java/Email/CatalogEmail.ht  
m](#))  
Event  
([/Code/Java/Event/CatalogEvent.ht  
m](#))  
File Input Output ([/Code/Java/File-  
Input-Output/CatalogFile-Input-  
Output.htm](#))  
Game  
([/Code/Java/Game/CatalogGame.h  
tm](#))  
Generics  
([/Code/Java/Generics/CatalogGen  
erics.htm](#))  
GWT  
([/Code/Java/GWT/CatalogGWT.ht  
m](#))  
Hibernate  
([/Code/Java/Hibernate/CatalogHib  
ernate.htm](#))  
I18N  
([/Code/Java/I18N/CatalogI18N.htm](#)  
)

# Sends the specified text or file as a datagram to the specified port of the specified host : DatagramPacket « Network Protocol « Java

Java ([/Code/Java/CatalogJava.htm](#))  
/ Network Protocol ([/Code/Java/Network-Protocol/CatalogNetwork-Protocol.htm](#))  
/ DatagramPacket ([/Code/Java/Network-Protocol/DatagramPacket.htm](#)) /

Sends the specified text or file as a datagram to the specified port of the specified host

J2EE  
 (/Code/Java/J2EE/CatalogJ2EE.htm)  
 J2ME  
 (/Code/Java/J2ME/CatalogJ2ME.htm)  
 JavaFX  
 (/Code/Java/JavaFX/CatalogJavaFX.htm)  
 JDK 6 (/Code/Java/JDK-6/CatalogJDK-6.htm)  
 JDK 7 (/Code/Java/JDK-7/CatalogJDK-7.htm)  
 JNDI LDAP (/Code/Java/JNDI-LDAP/CatalogJNDI-LDAP.htm)  
 JPA  
 (/Code/Java/JPA/CatalogJPA.htm)  
 JSP  
 (/Code/Java/JSP/CatalogJSP.htm)  
 JSTL  
 (/Code/Java/JSTL/CatalogJSTL.htm)  
 Language Basics  
 (/Code/Java/Language-Basics/CatalogLanguage-Basics.htm)  
 Network Protocol ()  
 PDF RTF (/Code/Java/PDF-RTF/CatalogPDF-RTF.htm)  
 Reflection  
 (/Code/Java/Reflection/CatalogReflection.htm)  
 Regular Expressions  
 (/Code/Java/Regular-Expressions/CatalogRegular-Expressions.htm)  
 Scripting  
 (/Code/Java/Scripting/CatalogScripting.htm)  
 Security  
 (/Code/Java/Security/CatalogSecurity.htm)  
 Servlets  
 (/Code/Java/Servlets/CatalogServlets.htm)  
 Spring  
 (/Code/Java/Spring/CatalogSpring.htm)  
 Swing Components  
 (/Code/Java/Swing-Components/CatalogSwing-Components.htm)  
 Swing JFC (/Code/Java/Swing-JFC/CatalogSwing-JFC.htm)  
 SWT JFace Eclipse  
 (/Code/Java/SWT-JFace-Eclipse/CatalogSWT-JFace-Eclipse.htm)  
 Threads  
 (/Code/Java/Threads/CatalogThreads.htm)  
 Tiny Application (/Code/Java/Tiny-Application/CatalogTiny-Application.htm)  
 Velocity  
 (/Code/Java/Velocity/CatalogVelocity.htm)  
 Web Services SOA  
 (/Code/Java/Web-Services-SOA/CatalogWeb-Services-SOA.htm)

```

/*
 * Copyright (c) 2004 David Flanagan. All rights reserved.
 * This code is from the book Java Examples in a Nutshell, 3rd Edition.
 * It is provided AS-IS, WITHOUT ANY WARRANTY either expressed or implied.
 * You may study, use, and modify it for any non-commercial purpose,
 * including teaching and use in open-source projects.
 * You may distribute it non-commercially as long as you retain this notice.
 * For a commercial use license, or to purchase the book,
 * please visit http://www.davidflanagan.com/javaexamples3.
 */
//package je3.net;
import java.io.File;
import java.io.FileInputStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

/**
 * This class sends the specified text or file as a datagram to the specified
 * port of the specified host.
 */
public class UDPSend {
    public static final String usage = "Usage: java UDPSend <hostname> <port> <msg>...\n"
        + "      or: java UDPSend <hostname> <port> -f <file>";

    public static void main(String args[]) {
        try {
            // Check the number of arguments
            if (args.length < 3)
                throw new IllegalArgumentException("Wrong number of args");

            // Parse the arguments
            String host = args[0];
            int port = Integer.parseInt(args[1]);

            // Figure out the message to send.
            // If the third argument is -f, then send the contents of the file
            // specified as the fourth argument. Otherwise, concatenate the
            // third and all remaining arguments and send that.
            byte[] message;
            if (args[2].equals("-f")) {
                File f = new File(args[3]);
                int len = (int) f.length(); // figure out how big the file is
                message = new byte[len]; // create a buffer big enough
                FileInputStream in = new FileInputStream(f);
                int bytes_read = 0, n;
                do { // loop until we've read it all
                    n = in.read(message, bytes_read, len - bytes_read);
                    bytes_read += n;
                } while ((bytes_read < len) && (n != -1));
            } else { // Otherwise, just combine all the remaining arguments.
                String msg = args[2];
                for (int i = 3; i < args.length; i++)
                    msg += " " + args[i];
                // Convert the message to bytes using UTF-8 encoding
                message = msg.getBytes("UTF-8");
            }

            // Get the internet address of the specified host
            InetAddress address = InetAddress.getByName(host);

            // Initialize a datagram packet with data and address
            DatagramPacket packet = new DatagramPacket(message, message.length, address, port);

            // Create a datagram socket, send the packet through it, close it.
            DatagramSocket dsocket = new DatagramSocket();
            dsocket.send(packet);
            dsocket.close();
        } catch (Exception e) {
            System.err.println(e);
            System.err.println(usage);
        }
    }
}

```

Related examples in the same category

1. Get data from DatagramPacket (/Code/Java/Network-Protocol/GetdatafromDatagramPacket.htm)
2. Get address and port from DatagramPacket (/Code/Java/Network-Protocol/GetaddressandportfromDatagramPacket.htm)
3. Create DatagramPacket from byte array (/Code/Java/Network-Protocol/CreateDatagramPacketfrombytearray.htm)
4. Waits to receive datagrams sent the specified port (/Code/Java/Network-Protocol/Waitstoreceivedatagramssentthspecifiedport.htm)

D:\Java\_Dev\WEB\java2s>java UDPReceive  
terminate batch job (Y/N)? n

(/Code/Java/Netv