G. O Brien

IT Tralee

20 November 2017

**User Datagram Protocol**

**----------------------**

Introduction

------------

This is a protocol for a UDP based file transfer system. It provides information on how to use the application alongside what messages must be sent and what responses to expect. It allows for the transferring of simple text documents using datagram packets. For this protocol to take effect, a user must send a request to server. For this reason this protocol is a request/response protocol similarly to HTTP.

This protocol documents the procedure of logging in, logging out, uploading a file and downloading a file using datagram packets.

Objectives

----------

The objectives of this protocol are:

* To transfer data from client to server and vice versa
* To allow a user to login, logout, download and upload
* Show the pathway of messages from client to server when making a request

Format

------

0 127.0.0.1 7 ..! 1024

+ - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - +

| Host | Destination | Message |

| Address | Port | |

+ - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - +

|

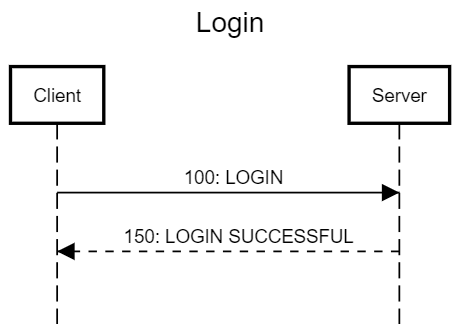
+ - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - +

UML Sequence Diagrams

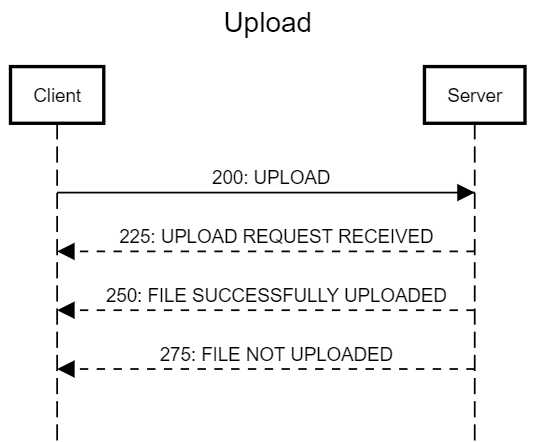
---------------------

Sequence diagrams which show the passing of messages and interaction between the Server and Client for the four functionalities of the application.

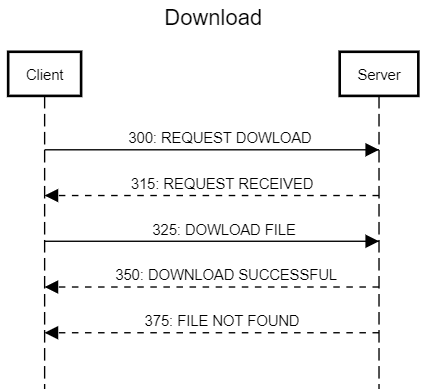
1. Login



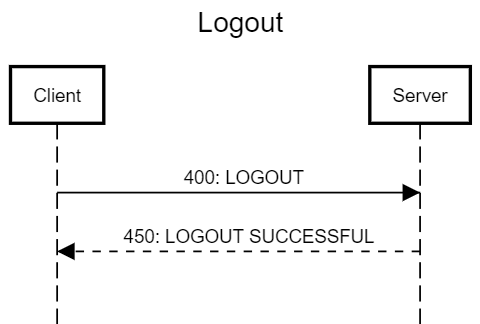
1. Upload



1. Download



1. Logout



Message Format

--------------

1. Login

**Message:** Login User

**Description:** The user may log into the server by passing in a username. Username cannot contain any special characters that are not allowed when creating a new folder on windows. The message is of the format [code] [message] e.g. 100-LOGIN John.

**Code:** 100

**Message Parameters: [**hostname, port number, message]

**Example Request:**

|  |
| --- |
| 127.0.0.1, 7, 100-LOGIN John |

**Message Response:** If the user is logged in successfully the server responds with a 150 code, else if the server cannot log in the user send a code 175.

1. Upload

**Message:** Upload file

**Description:** The user may upload a file by passing in the UPLOAD message followed by the file. The file is then converted to a byte array and sent over the network. Only the contents of a .txt is kept when transferred.

**Code:** 200

**Message Parameters: [**hostname, port number, message]

**Example Request:**

|  |
| --- |
| 127.0.0.1, 7, 200-UPLOAD File.txt |

**Message Response:** If the file is received for upload the server returns a 225, else when the file is uploaded the server returns a 250, else if the upload fails the server returns a 275.

1. Request Download

**Message:** Request Download

**Description:** The user may request to download a file from the server by passing the message 300-DOWNLOADREQUEST. This message returns a list of all the files that can be downloaded from the server.

**Code:** 300

**Message Parameters: [**hostname, port number, message]

**Example Request:**

|  |
| --- |
| 127.0.0.1, 7, 300-DOWNLOAD |

**Message Response:** If the request is accepted then the server returns a 315 request received message.

1. Download File

**Message:** Download

**Description:** The user may select a file to download by passing a message starting with 325-DOWNLOADFILE followed by the name of the file that you wish to download from the server.

**Code:** 325

**Message Parameters: [**hostname, port number, message]

**Example Request:**

|  |
| --- |
| 127.0.0.1, 7, 325-DOWNLOAD File.txt |

**Message Response:** If the file is successfully downloaded the server returns a 350 code, else if the file cannot be found on the server side then a code of 375 is returned to the client.

1. Logout

**Message:** Logout

**Description:** The user passes a message containing the LOGOUT message to the server. This will then logout the current instance of the user on the server and they will not be able to access the upload and download functionality.

**Code:** 400

**Message Parameters: [**hostname, port number, message]

**Example Request:**

|  |
| --- |
| 127.0.0.1, 7, 400-LOGOUT |

**Message Response:** When the user has been successfully logged out from the server, a code of 400 is returned.

Pseudocode of Functionalities

-----------------------------

1. Login

Try {

* Receive message from client
* Remove code from message String
* Check if a user is already logged in
  + If user isn’t logged in
    - Check if the user already has a folder
    - Make directory for user if none exists
    - If it can’t make the folder send a 175-INVALIDUSERNAME message.
    - Else log in user and send 150 success message to client

} catch {

* Catch error on folder creation

}

1. Upload

Try {

* Receive message from client
* Remove code from message String
* Set current directory to user’s directory
* Send request received message to client
* Receive file to upload from client
* Write file to the user’s directory

} catch {

* Catch file not found exception
* Send file not uploaded message back to client

}

1. Download

Try {

* Receive message from client
* Remove code from message String
* Set current directory to user’s directory
* Put list of files into an array and send to client
* Receive the name of the file to download from user
* Send the file to the client
* Send a successful download message to the client

} catch {

* Catch file not found exception
* Send file not found message back to client

}

1. Logout

* Set the currently logged in user to nobody
* Set if user is logged in to false
* Send a message to client saying logout was successful

Protocol Codes

---------------

100-LOGIN: Request to login user

150-LOGINSUCCESSFUL: Login was successful

175-INVALIDUSERNAME: The username supplied was invalid

200-UPLOAD: Request to upload file

225-UPLOADREQUESTRECEIVED: The request was received

250-UPLOADSUCCESSFUL: The upload was successful

275-FILENOTUPLOADED: The chosen file was not uploaded

300-DOWNLOADREQUEST: Request to download file

315-DOWNLOADREQUESTRECEIVED: The download request was received

325-DOWNLOADFILE: Download the chosen file

350-DOWNLOADSUCCESSFUL: The download was successful

375-FILENOTFOUND: The selected file was not found

450-LOGOUTSUCCESSFUL: The logout attempt was successful

700-INVALIDREQUEST: An unrecognisable code was sent