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 | |

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

|

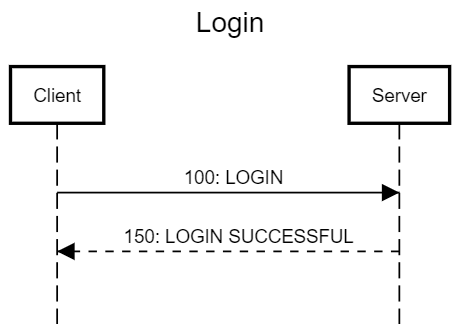
+ - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - + - - - - - - - - +

UML Sequence Diagrams

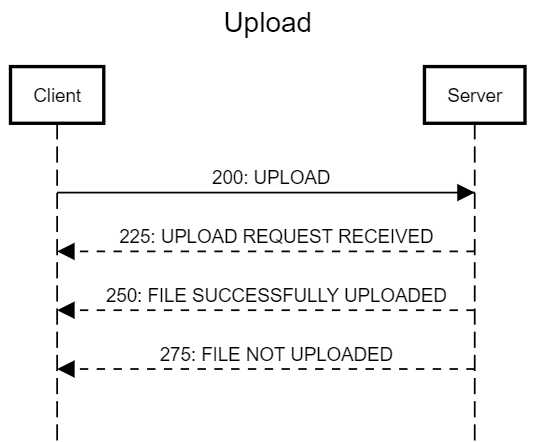
---------------------

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

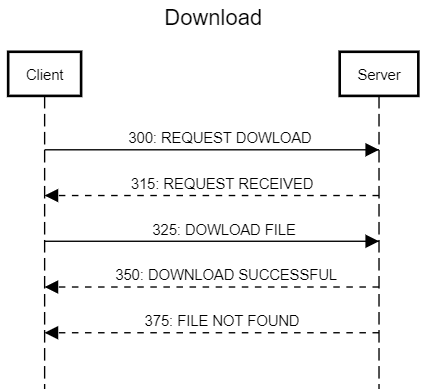
1. Login



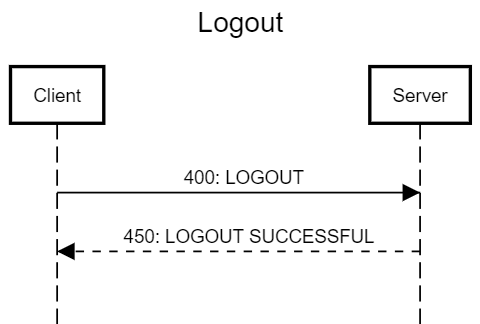
1. Upload



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.

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. Download

**Message:** Download file

**Description:** The user may upload a file by passing in the DOWNLOAD message. All the available files that may be downloaded are returned in an array. The user must then select a file to be downloaded and send another request with this file

**Code:** 300

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

**Example Request:**

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

**Message Response:** If the request for a download is received, 300 is returned, else when the file is downloaded the server returns a 350, else if the download fails or the file doesn’t exist, the server returns a 275.

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