RPC client-server

DNP lab 5

Information

1. Both server and client should use Remote Procedure Call (RPC) from python's `xmlrpc` module.
2. Your submission must contain at least two files: `server.py` and `client.py`.
3. Archives are in `.zip` format.

Server

- Always ready to recieve connections from clients.
- Exits on `KeyboardInterrupt`.
- Prints `Server is stopping` on the exit.
- Has 2 command line arguments: ip address and port.

  Example: `python server.py 127.0.0.1 42000`
- Has functions:

1. send_file(filename, data)
2. list_files()
3. delete_file(filename)
4. get_file(filename)
5. calculate(expression)

send_file(filename, data)

1. Saves the `data` in the file with name `filename`.
2. If there is a file with the same name, prints `<filename> not saved` and returns `False`.
3. Otherwise prints `<filename> saved` and returns `True`

list_files()

1. Returns the list of saved files (filenames).

delete_file(filename)

1. Deletes saved file with name `filename`.
2. If there is no file with such name, prints `<filename> not deleted` and returns `False`.
3. Otherwise prints `<filename> deleted` and returns `True`.

get_file(filename)

1. Sends file with name `filename` to the client.

2. If there is no file with such name, prints `No such file: <filename>` and returns `False`.
3. Prints `File send: <filename>` otherwise.

calculate(expression)

1. Supports operations: `*`, `/`, `-`, `+`, `>`, `<`, `>=`, `<=`.
2. Determines the type (int or float).
3. Returns result of calculation of the same type.
4. Returns `False` if there is any error (wrong expression format, division by zero, etc). Probably, you want to return tuple (True/False, Error message).
5. Prints either `<expression> -- done` or `<expression> -- not done`.

Client

- Interactivly gets operations from the user.
- Prints `\nEnter the command:` before every operation.
- Stops on `KeyboardInterrupt`.
- Prints `Client is stopping` on the exit.
- Has 2 command line arguments: ip address and port of the server.

  Example: `python client.py 127.0.0.1 42000`
- At the end of each operation prints either `Completed` or `Not completed` message.
- If the operation is not completed, it prints the error message on a new line.
- If the operation is wrong, it prints `Wrong command` message.
- Has operations:

  1. `quit`
  2. `send <filename>`
  3. `list`
  4. `delete <filename>`
  5. `get <filename> <new filename>`
  6. `calc <expression>`

quit

1. Stops execution of the client.

send <filename>

1. Sends specified file to the server.
2. If there is no file with `filename` name, prints `No such file` and does not send anything.

list

1. Prints all the files saved on the server.
2. Each file is printed on a new line.

delete <filename>

1. Deletes saved file from server.
2. If there is no file with such name on the server, prints `No such file`

get <filename> <new filename>

1. `new filename` is an optional parameter.
2. Downloads the file `filename` from the server and saves it with the same name if the `new filename` is not specified, or with the `new filename` otherwise.
3. If there is already file with `filename`, prints `File already exists` and does not dowload the file.

calc <expression>

1. `expression` has the format: `operator left right`

   Example: `- 3 2`

2. Gets result of calculation from the server and prints the result.

Error messages

send_file

1. `No such file` - No file at the client side
2. `File already exists`- File already exists at the server side

list_files

1. None

delete_file

1. `No such file` - No file at the server side

get_file

1. `No such file` - No file at the server side
2. `File already exists` - File already exists at the client side

calculate

1. `Wrong expression` - Wrong format of the expression
2. `Division by zero` - Division by zero

Any other exceptions should be converted to strings and printed on the client.

Example

Client

```
> python client.py 127.0.0.1 42000

Enter the command:
> calc - 3 2
1
Completed

Enter the command:
> calc / 2 0
Not completed
Division by zero

Enter the command:
> list
Completed

Enter the command:
> send pic.jpg
Completed

Enter the command:
> send pic.jpg
Not completed
File already exists

Enter the command:
> send pic2.jpg
Completed

Enter the command:
> list
pic.jpg
pic2.jpg
Completed

Enter the command:
> get pic.jpg pic4.jpg
Completed

Enter the command:
> get pic2.jpg pic4.jpg
Not completed
File already exists

Enter the command:
> pow 2 3
Not completed
Wrong command

Enter the command:
```

```
> delete pic.jpg
Completed

Enter the command:
> delete pic.jpg
Not completed
No such file

Enter the command:
> quit
Client is stopping
```

Server

```
> python server.py 127.0.0.1 42000
- 3 2 -- done
/ 2 0 -- not done
pic.jpg saved
pic.jpg not saved
pic2.jpg saved
File send: pic.jpg
pic.jpg deleted
pic.jpg not deleted
Server is stopping
```