System Documentation

ChatSiGira v 1.0 – 24/05/2020

Team:

- Allari Edoardo
- Bologna Abreham
- Nicolis Pietro
- Pollinari Jonathan
- **Zerbato Nicolò**

Version	Date	Name	Description
1	3/3/20	Allari Edoardo Bologna Abreham Pollinari Jonathan Nicolis Pietro Zerbato Nicolò	Initial Document

Introduction

ChatSiGira is an application created for communication between different clients through a centralized server. This document will provide instructions for accessing the application code and installing it into a compatible device.

Client-side configuration

For the client application you need to have a development environment based on your needs and preferences. The ChatSiGira project was based on the Java programming language because it was less complex for the team and we already had a solid preparation. This is the link of the repository to download the code and test it to see all the implemented features:

https://github.com/teddyedo/2020_5EI_team1_Allari.git.

Install on Simulator or Device

Required Components

To test the application you need to use a computer with a Windows or IOS operating system, the NetBeans development environment, the process of running a centralized server for communication between clients. This application can only be used locally and is not supported for mobile devices.

Install Code

- Use the GitHub service to download the project locally,
- Create your own Github account,
- Click on the "Clone or download" button and download the project,
- Start the Netbeans platform,
- Go to the "File" window and click "Open file", select the correct application,
- Start the Server and connect it to the application via its IP address,
- Click on "Run" to start the application.

System maintenance

In the ChatSiGira application we have created three interfaces. The first "Login Interface" was used to send the request to the server using the InputStream and OutputStream channels to access the public room and communicate with other clients, the second "MainInterface" interface was created for public communication, for the change of aliases and for disconnection. Finally we have the "Chat Interface" to communicate privately with a single client. All these iterations are possible thanks to the created classes that allow to analyze the packet that arrives from the server, so the client will respond correctly and functionally.