**Server Multithread**

Immagine che contiene testo, diagramma, linea, schermata

Descrizione generata automaticamente

Stamm facenn sta cosa, cioè o server andrà a creare tanti thread quante sono le connessioni che vengono attiva.

**Server**  
Semplicemente metto in attesa il **processo Server** di richieste di connessioni da parte di Client.  
Quando accetta una **richiesta** il **Server** istanzia un Worker**Thread** che sarà proprio associato alla socket (rappresenta quella determinata connessione) passandogliela per parametro.

**WorkerServer**

Sono **thread** definiti per ereditarietà, il cui costruttore associa la **socket** a **dato membro**.  
Il **run**() crea il **canale di comunicazione** con la **socket Client**.

            DataInputStream fromClient = new DataInputStream ( s.getInputStream());

            DataOutputStream toClient = new DataOutputStream ( s.getOutputStream());

Ovviamente si intende per **input** **da dove prendere i dati**, e per **Output dove inviare i dati**.

Si usano le classi **DataInputStream** e **DataOutputStream** perché consentono all’applicazione di **leggere** /scrivere i **byte** dallo **stream** di **input** (cosa che fa anche InputStream) ma rendendolo indipendente dal **tipo** di **architettura** della macchina sottostante.

I metodi s.getInputStream e output restituiscono gli **stream** di **input** e **output** per la comunicazione con la socket lato client.  
(Restituiscono un tipo InputStream e OutputStream che verranno passati al costruttore DataInputStream).  
quindi praticamente ci danno la posizione da cui leggere e in cui scrivere.

In più vengono chiusi gli **stream** di **input** e **output** e ovviamente la stessa **connessione** alla socket lato client.

**Client**

Creiamo direttamente la **socket** per **richiedere** la connessione al **Server**, ricorda che il costruttore di una socket è un **metodo bloccante**.  
Una volta che il **server** ha accettato la **richiesta** del **Client**, allora il **Client** al pari di quanto fa il Server, crea gli **stream** per la comunicazione con la **socket** lato **server**.