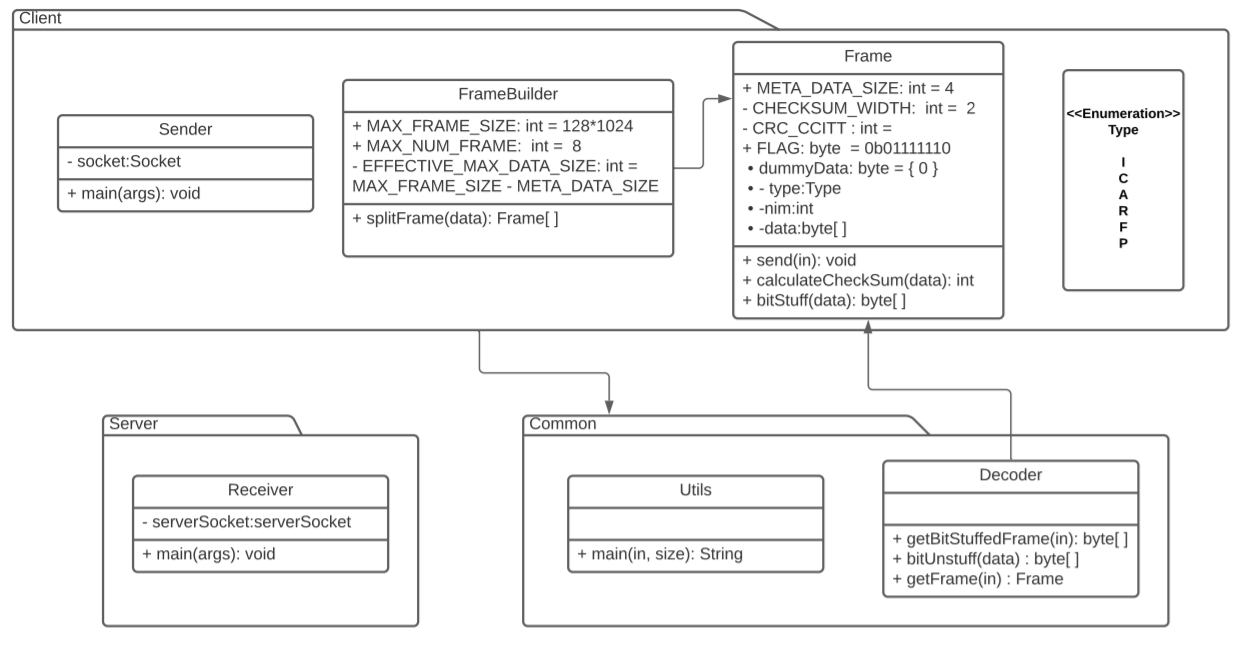
1. **Diagramme de classe:**



1. **Description de classes/méthodes:**

**Package SERVER :**

*Nom de la classe: Reveiver.java*

* Nom de la méthode : main
* Params : String [] args
* Fonction: Reçois les données
* Output: rien

**Package CLIENT :**

*Nom de la classe: Sender.java*

* Nom de la méthode : Sender
* Params : String hostname, int port, byte[] data
* Fonction: ouvre et ferme le socket du sender lorsque nécessaire
* Output: rien
* Nom de la méthode : Main
* Params : String[] args
* Fonction: vérifie que l’appel se fait correctement
* Output: rien

*Nom de la classe: FrameBuilder.java*

* Nom de la méthode : splitFrame
* Params : byte[] data
* Fonction: Divise le data en différent paramètres
* Output: Frame [] contenant les bonnes données

*Nom de la classe: Frame.java*

* Nom de la méthode : calculateChecksum
* Params : byte[] data
* Fonction: Calcule la somme CRC-CCIT
* Output: Int qui représente le résultat de la division
* Nom de la méthode : bitStuff
* Params : byte[] data
* Fonction: s’occupe de faire le bit stuffing du data donnée
* Output: byte[] avec le nouveau data

**Package COMMON :**

*Nom de la classe: Utils.java*

* Nom de la méthode : leftPad
* Params : String in, int size
* Fonction: ajoute les 0 a gauche de la string reçu en entrée
* Output: String, le nouveau nombre avec les 0

*Nom de la classe: Decoder.java*

* Nom de la méthode : getBitStuffedFrame
* Params : InputStream in
* Fonction: va ajouter les bits nécessaire dans les différentes trames
* Output: byte[] contenant les bytes le nouveau nombre
* Nom de la méthode : bitUnstuff
* Params : byte[] data
* Fonction: Retire les bits de contrôles
* Output: byte[] contenant les bytes désirés
* Nom de la méthode : getFrame
* Params : InputStream in
* Fonction: créer une nouvelle trame avec les données de bits de contrôle
* Output: Frame contenant les nouvelles données