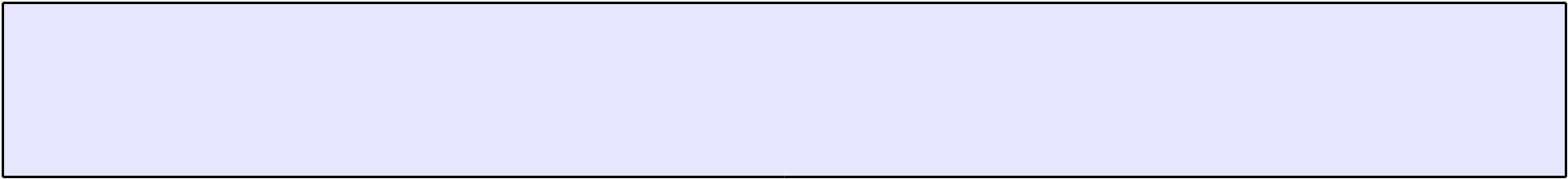
Recevoir un paquet

* On commence donc par déclarer un tableau d'octets d'une certaine taille
* On associe ensuite ce tableau d'octets à un paquet
* Au moment de la réception, le **receive** remplit le tableau d'octets

**DatagramSocket dso=new DatagramSocket(5555); byte[]data=new byte[100];**



**DatagramPacket paquet=new DatagramPacket(data,data.length); dso.receive(paquet);**

* On peut ensuite récupérer le contenu et la longueur du paquet avec les deux méthodes suivante de **DatagramPacket** :
  + **byte[] getData()**
  + **int getLength()**
* Pour passer des chaînes de caractères aux tableaux d'octets on peut utiliser :
  + **String(byte[] bytes, int offset, int length)** -> constructeur
  + **byte[] getBytes()** -> méthode de la classe String

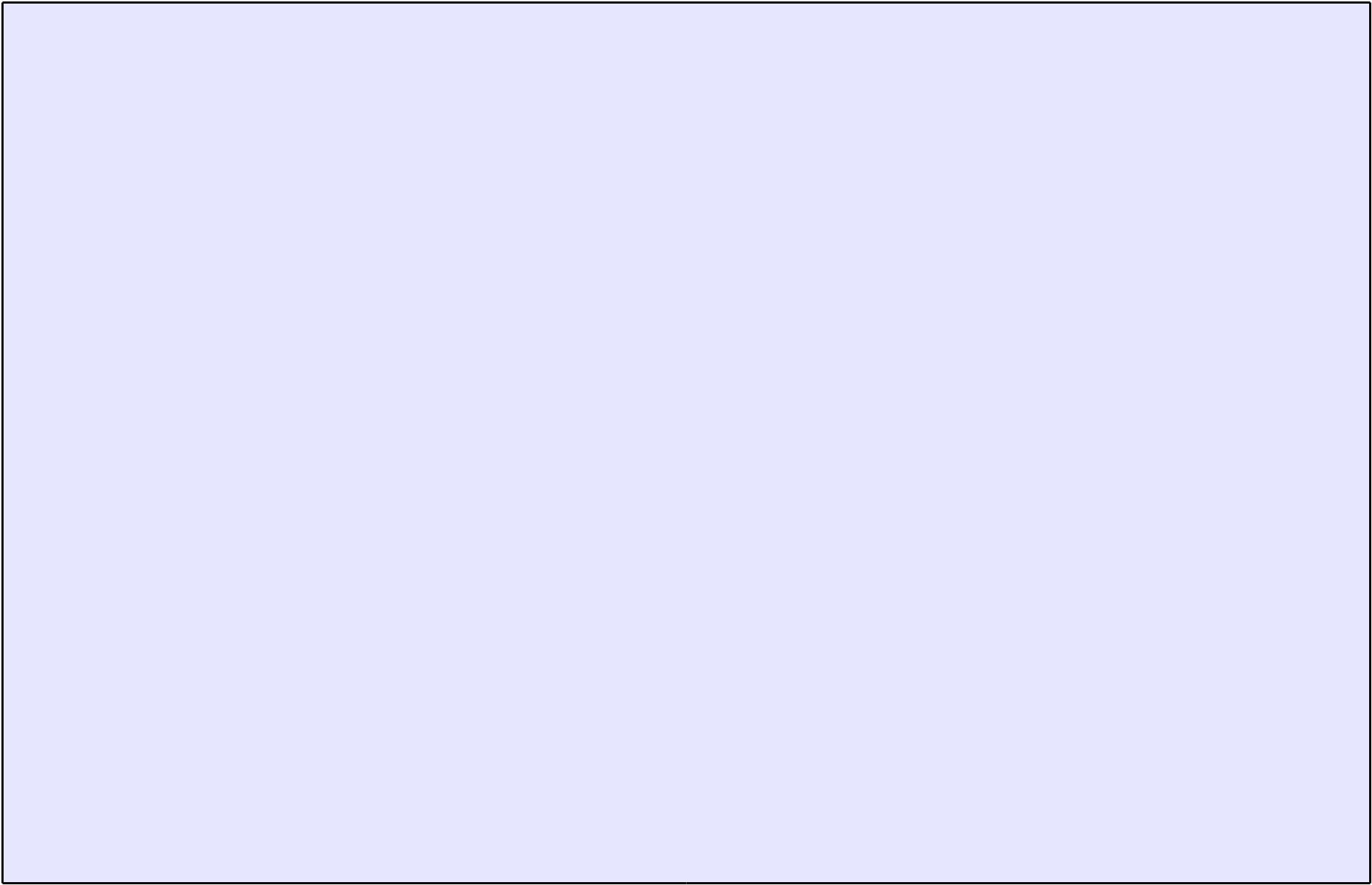
PR - UDP 13

Méthodes utiles de DatagramPacket

* **InetAddress getAddress()**
  + Renvoie l'adresse de la machine vers laquelle le paquet est envoyé ou de la machine qui a envoyé le paquet
* **int getPort()**
  + Renvoie le port vers lequel le paquet est envoyé ou depuis lequel la paquet a été envoyé
* **SocketAddress getSocketAddress()**
  + Comme les deux méthodes ci-dessus avec SocketAddress

PR - UDP 18

Exemple

**public class ReceiveUDPPlus3 {**

**public** **static void main(String[] args){**

**try{**

**DatagramSocket dso=new DatagramSocket(5555); byte[]data=new byte[100];**

**while(true){**

**DatagramPacket paquet=new**

**DatagramPacket(data,data.length);**

**dso.receive(paquet);**

**String st=new**

**String(paquet.getData(),0,paquet.getLength()); System.out.println("J'ai reçu :"+st); InetSocketAddress ia=(InetSocketAddress)**

**paquet.getSocketAddress();**

**System.out.println("De la machine "+ia.getHostName());**

**System.out.println("Depuis le port "+ia.getPort());**

**}**

**} catch(Exception e){**

**e.printStackTrace();**

**}**

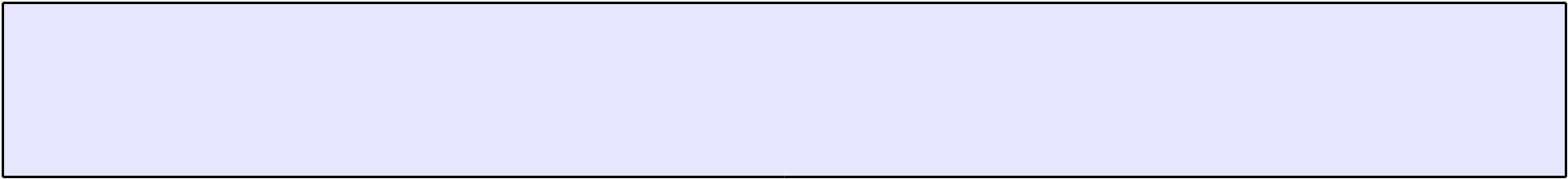
**}**

**}**

Envoi de paquets

* Pour envoyer des paquets, on n'a pas besoin d'attacher la socket à un port
* On met l'adresse et le port du destinataire dans le paquet

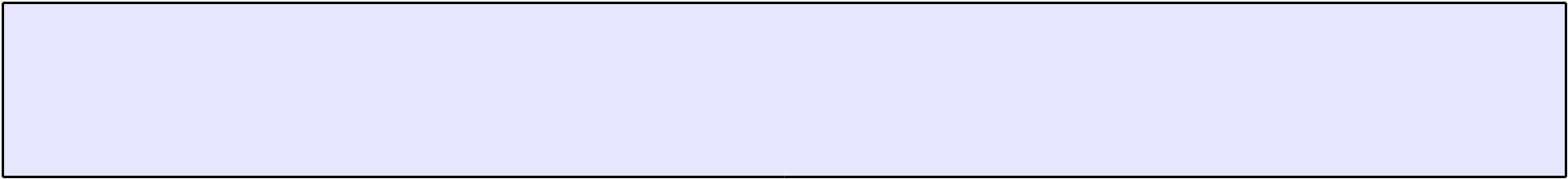
**String s="MESSAGE "+i+" \n";**



**byte[]data = s.getBytes();**

**InetSocketAddress ia=new InetSocketAddress("localhost",5555); DatagramPacket paquet=new DatagramPacket(data,data.length,ia);**

* Ou encore :

**String s="MESSAGE "+i+" \n";**

**byte[]data = s.getBytes();**

**DatagramPacket paquet=new DatagramPacket(data,data.length, InetAddress.getByName("localhost"),5555);**

* **Pensez à un colis où on écrit l'adresse sur le paquet !**
* Ensuite on fait **send** sur une **DatagramSocket**
* **ATTENTION : EN UDP, c'est pas parce que l'on envoie un paquet qu'il est reçu !!!**

PR - UDP 22

Exemple d'envoi

**public class EnvoiUDP {**

**public** **static void main(String[] args){**

**try{**

**DatagramSocket dso=new DatagramSocket(); byte[]data;**

**for(int i=0;i <= 60; i++){**

**}**

**}**

**String s="MESSAGE "+i+" \n";**

**data=s.getBytes();**

**DatagramPacket paquet=new**

**DatagramPacket(data,data.length,**

**InetAddress.getByName("localhost"),5555);**

**dso.send(paquet);**

**}**

**} catch(Exception e){**

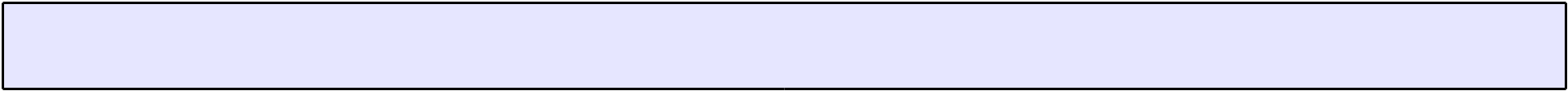
**e.printStackTrace();**

**}**

PR - UDP 23

La communication UDP en C

* Pour le C, il faut changer le type des sockets quand on appelle socket
  + **SOCK\_STREAM** : socket TCP
  + **SOCK\_DGRAM** : socket UDP



**int sock=socket(PF\_INET,SOCK\_DGRAM,0);**

* On n'a plus besoin d'utiliser **connect** (vu qu'on n'est pas en mode connecté)
* Si on veut écouter sur une socket, il faudra bien faire le **bind**
* On n'aura plus de **listen** ou d'**accept**
* On utilisera **sendto**, **recv**, **recvfrom** pour l'envoi et la réception des paquets

PR - UDP 25

Envoi en UDP

* Comme en Java, on va préciser le destinataire au moment de l'envoi du paquet
* On va utiliser la fonction suivante :
* **ssize\_t sendto(**

**int socket, // le numéro de socket**

**const void \*buffer, // le message à envoyer size\_t length, // la taille de ce message int flags, // pour les options**

**const struct sockaddr \*dest\_addr, //infos destinataire socklen\_t dest\_len); //taille de la struct sockaddr**

PR - UDP 26

Envoi en UDP (2)

* On va donc commencer par récupérer les infos du destinaire par exemple avec **getaddrinfo**
* **int getaddrinfo(const char \*hostname, const char \*servname ,const struct addrinfo \*hints, struct addrinfo \*\*res);**
* Rappel :



**struct addrinfo hints;**

**bzero(&hints, sizeof(struct addrinfo)); hints.ai\_family = AF\_INET; hints.ai\_socktype=SOCK\_DGRAM;**

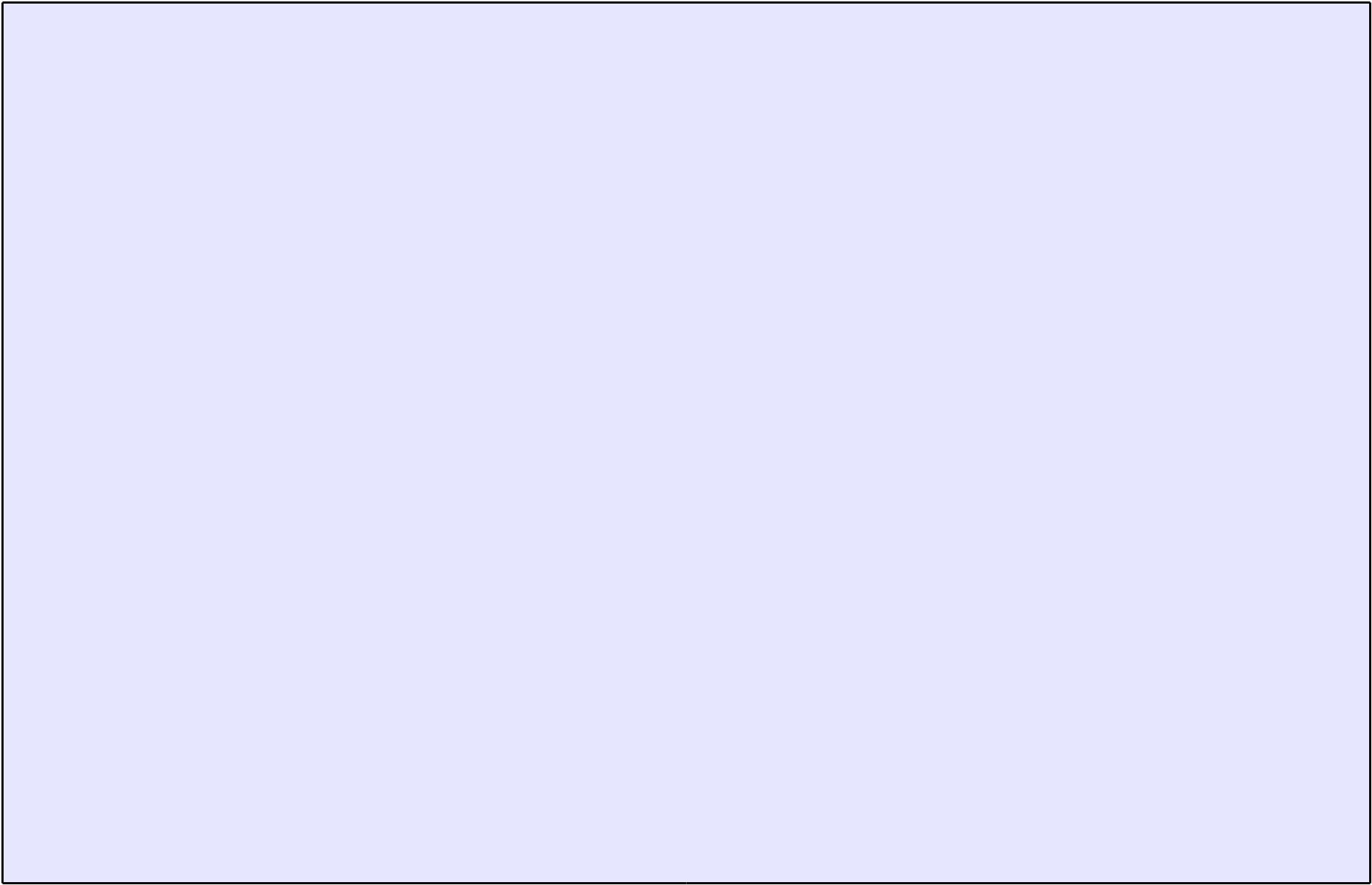
**struct addrinfo \*first\_info;**

**int r=getaddrinfo("localhost","5555",&hints,&first\_info);**

* Ici, dans les hints, on précise SOCK\_DGRAM car on veut des sockets UDP
* Ensuite le struct sockaddr correspondant est dans
  + first\_info->ai\_addr; (de type struct sockaddr\*)

PR - UDP 27

Exemple d'envoi



**int main() {**

**int sock=socket(PF\_INET,SOCK\_DGRAM,0); struct addrinfo \*first\_info;**

**struct addrinfo hints;**

**memset (&hints, 0, sizeof(struct addrinfo)); hints.ai\_family = AF\_INET; hints.ai\_socktype=SOCK\_DGRAM;**

**int r=getaddrinfo("localhost","5555",&hints,**

**&first\_info);**

**if(r==0){**

**if(first\_info!=NULL){**

**struct sockaddr \*saddr=first\_info->ai\_addr; char tampon[100];**

**for(int i=0;i<=10;i++){**

**strcpy(tampon,"MESSAGE ");**

**char entier[3];**

**sprintf(entier,"%d",i);**

**strcat(tampon,entier);**

**sendto(sock,tampon,strlen(tampon),0,saddr,**

**(socklen\_t)sizeof(struct sockaddr\_in));**

**}**

**}**

**}**

**return 0;**

**}**

28

Pour la réception

* Là encore on doit préciser que l'on utilise des socket UDP
* On fait un bind pour écouter sur le bon port
* **int bind(int sockfd, struct sockaddr \*my\_addr, int addrlen);**
* Rappel :

**sock=socket(PF\_INET,SOCK\_DGRAM,0);**



**struct sockaddr\_in address\_sock;**

**address\_sock.sin\_family=AF\_INET;**

**address\_sock.sin\_port=htons(5555);**

**address\_sock.sin\_addr.s\_addr=htonl(INADDR\_ANY);**

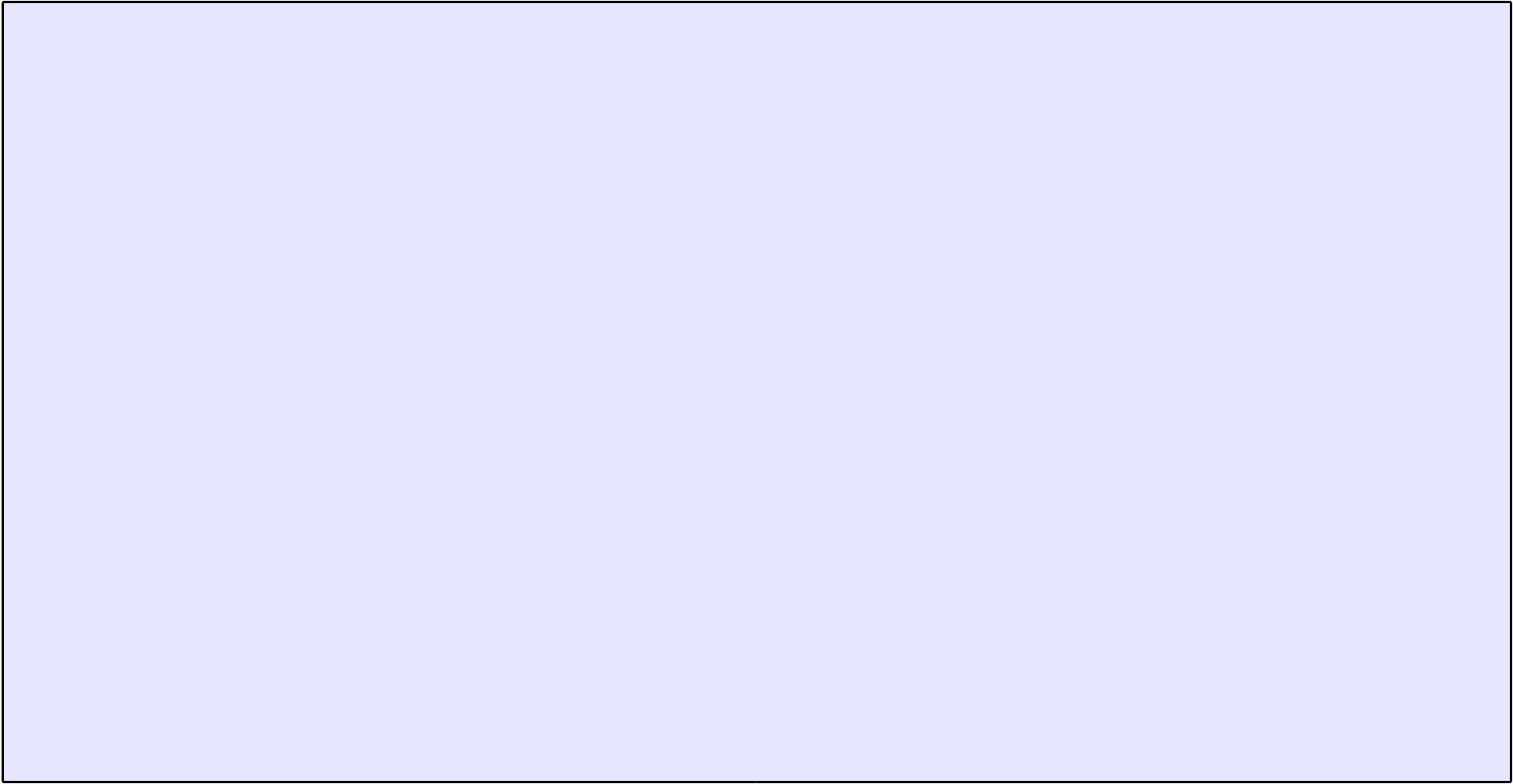
**int r=bind(sock,(struct sockaddr\*)&address\_sock,sizeof(struct sockaddr\_in);**

* On utililse la fonction suivante pour recevoir :
  + **ssize\_t recv(int socket, void \*buffer, size\_t length, int flags);**

PR - UDP 29

Exemple

**int main() {**

****

**int sock= socket(PF\_INET,SOCK\_DGRAM,0);**

**struct sockaddr\_in address\_sock;**

**address\_sock.sin\_family=AF\_INET;**

**address\_sock.sin\_port=htons(5555);**

**address\_sock.sin\_addr.s\_addr=htonl(INADDR\_ANY);**

**int r=bind(sock,(struct sockaddr \*)&address\_sock,sizeof(struct sockaddr\_in));**

**if(r==0){**

**char tampon[100];**

**while(1){**

**int rec=recv(sock,tampon,100,0);**

**tampon[rec]='\0';**

**printf("Message recu : %s\n",tampon);**

**}**

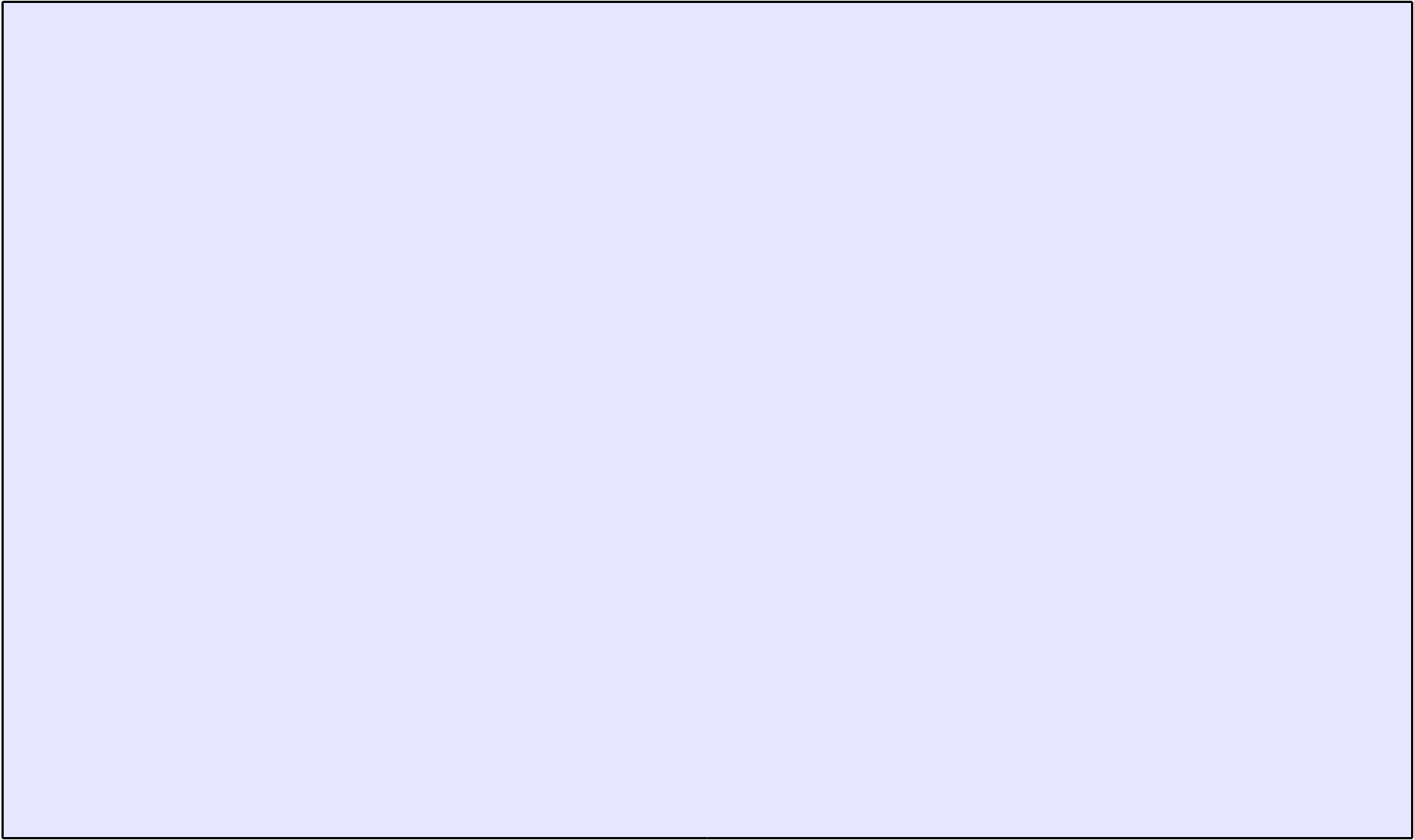
**}**

**return 0;**

**}**

PR - UDP 30

recvfrom



**int main() {**

**int sock= socket(PF\_INET,SOCK\_DGRAM,0);**

**sock=socket(PF\_INET,SOCK\_DGRAM,0);**

**struct sockaddr\_in address\_sock;**

**address\_sock.sin\_family=AF\_INET;**

**address\_sock.sin\_port=htons(5555);**

**address\_sock.sin\_addr.s\_addr=htonl(INADDR\_ANY);**

**int r=bind(sock,(struct sockaddr \*)&address\_sock,**

**sizeof(struct sockaddr\_in));**

**struct sockaddr\_in emet;**

**socklen\_t a=sizeof(emet);**

**if(r==0){**

**char tampon[100];**

**while(1){**

**int rec=recvfrom(sock,tampon,100,0,(struct sockaddr \*)&emet,&a); tampon[rec]='\0';**

**printf("Message recu : %s\n",tampon);**

**printf("Port de l'emetteur: %d\n",ntohs(emet.sin\_port));**

**printf("Adresse de l'emetteur: %s\n",**

**inet\_ntoa(emet.sin\_addr));**

**}**

**}**

**return 0;}**

PR - UDP 32