

LOG725 - Ingénierie et conception de jeux vidéo

Labo 10 - Conception Sonore + RPC

Gabriel C. Ullmann

École de Technologie Supérieure, Hiver 2024



Le génie pour l'industrie

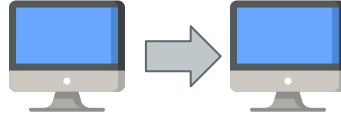
Objectifs d'apprentissage

- Apprendre à utiliser dans Godot:
 - Le noeud AudioStreamPlayer
 - Le noeud AudioStreamGenerator
 - Les Appels de Procédure à Distance (RPC)

Activités



Tutoriel: Conception
Sonore + RPC

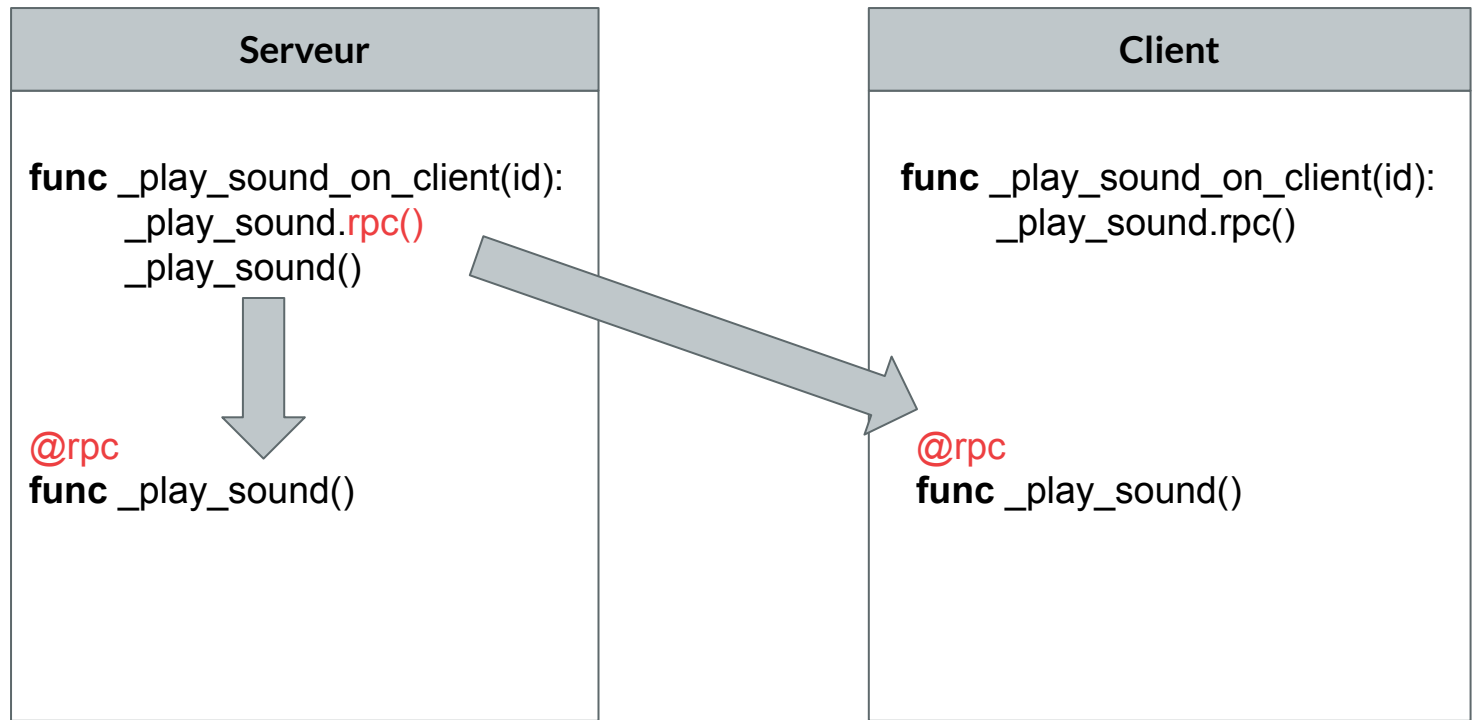


Appels de Procédure à
Distance (RPC)



Introduction
Travail Pratique 5

Appels de Procédure à Distance (RPC)



Appels de Procédure à Distance (RPC)

- **mode:** qui peut appeler les fonctions @rpc? (autorité/tous).
- **sync:** où la fonction @rpc peut-elle être appelée? (à distance/local).
- **transfer_mode:** transmission fiable/non fiable.
- **channel:** nombre de la chaîne (sub-connection).

```
@rpc("authority", "call_remote", "unreliable", 0)
```

AudioStreamPlayer vs. AudioStreamGenerator

- **AudioStreamPlayer:** jouer tous les types des sources (fichiers, streams, etc.)
- **AudioStreamGenerator:** générer un stream audio
 - `$AudioStreamPlayer.get_stream_playback()`
 - `playback.push_frame()`
 - La durée dépendra de la taille de la mémoire tampon (buffer size)

Travail Pratique 5

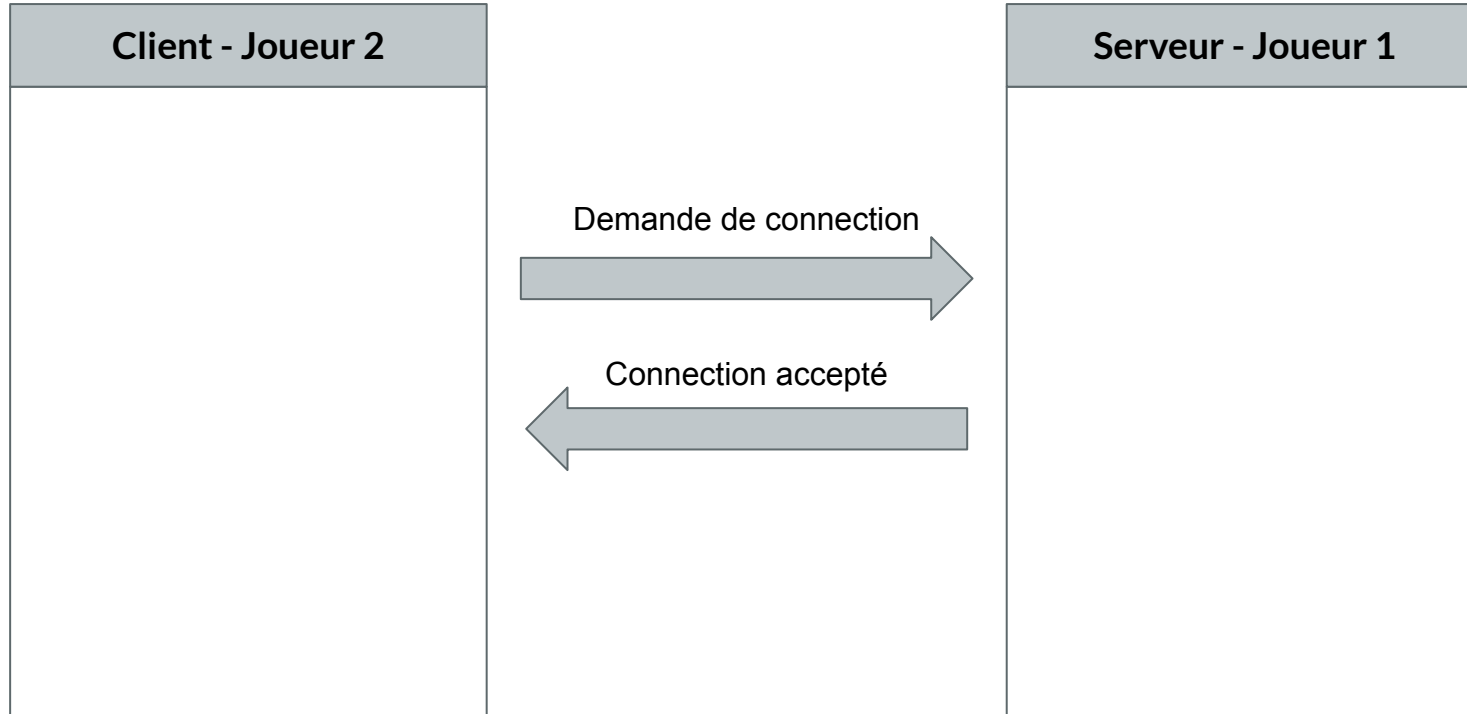
- Créer un télégraphe:
 - **Server (P1)** : envoie un message de texte à chiffrer.
 - **Client (P2)** : essaie de chiffrer le message en code morse.
 - **But du jeu**: le client doit réussir dans la conversion de texte en code morse.
 - Si le client n'est pas réussi, le serveur peut l'envoyer une nouvelle message à chiffrer.

A ● ■
B ■ ■ ● ●
C ■ ■ ■ ●
D ■ ■ ● ●
E ●
F ● ● ■ ●
G ■ ■ ■ ●
H ● ● ● ●
I ● ●
J ● ■ ■ ■ ■
K ■ ■ ● ■
L ● ■ ■ ● ●
M ■ ■ ■
N ■ ■ ●
O ■ ■ ■ ■
P ● ■ ■ ■ ●
Q ■ ■ ■ ● ■
R ● ■ ■ ●
S ● ● ●
T ■

U ● ● ■
V ● ● ● ■
W ● ■ ■ ■
X ■ ■ ● ● ■
Y ■ ■ ● ■ ■
Z ■ ■ ■ ● ●

1 ● ■ ■ ■ ■
2 ● ● ■ ■ ■
3 ● ● ● ■ ■
4 ● ● ● ● ■
5 ● ● ● ● ●
6 ■ ■ ● ● ●
7 ■ ■ ■ ● ●
8 ■ ■ ■ ■ ●
9 ■ ■ ■ ■ ■ ●
0 ■ ■ ■ ■ ■

1) Connection initiale



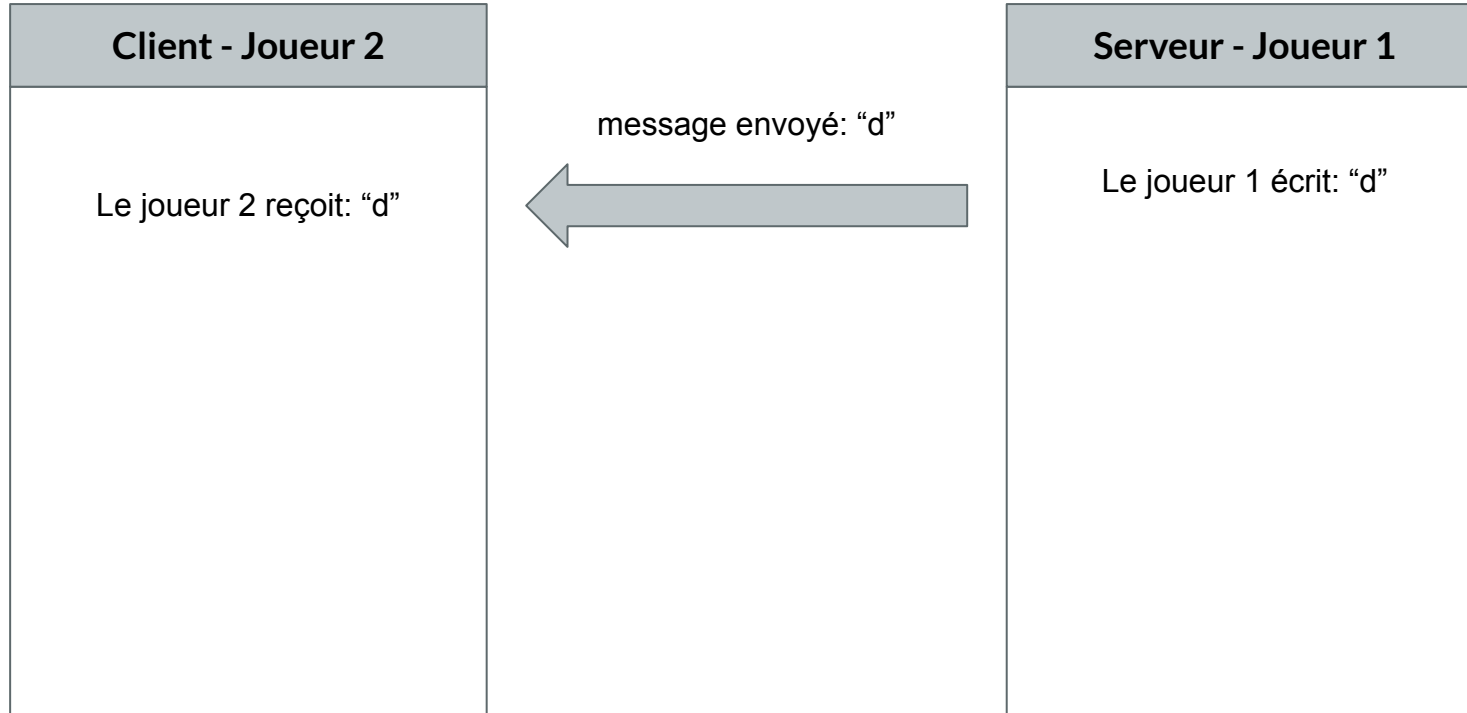
2) Attente du client

Client - Joueur 2

Le joueur 2 attend
que le joueur 1 envoyé une
message pour commencer

Serveur - Joueur 1

3) Le jeu est commencé



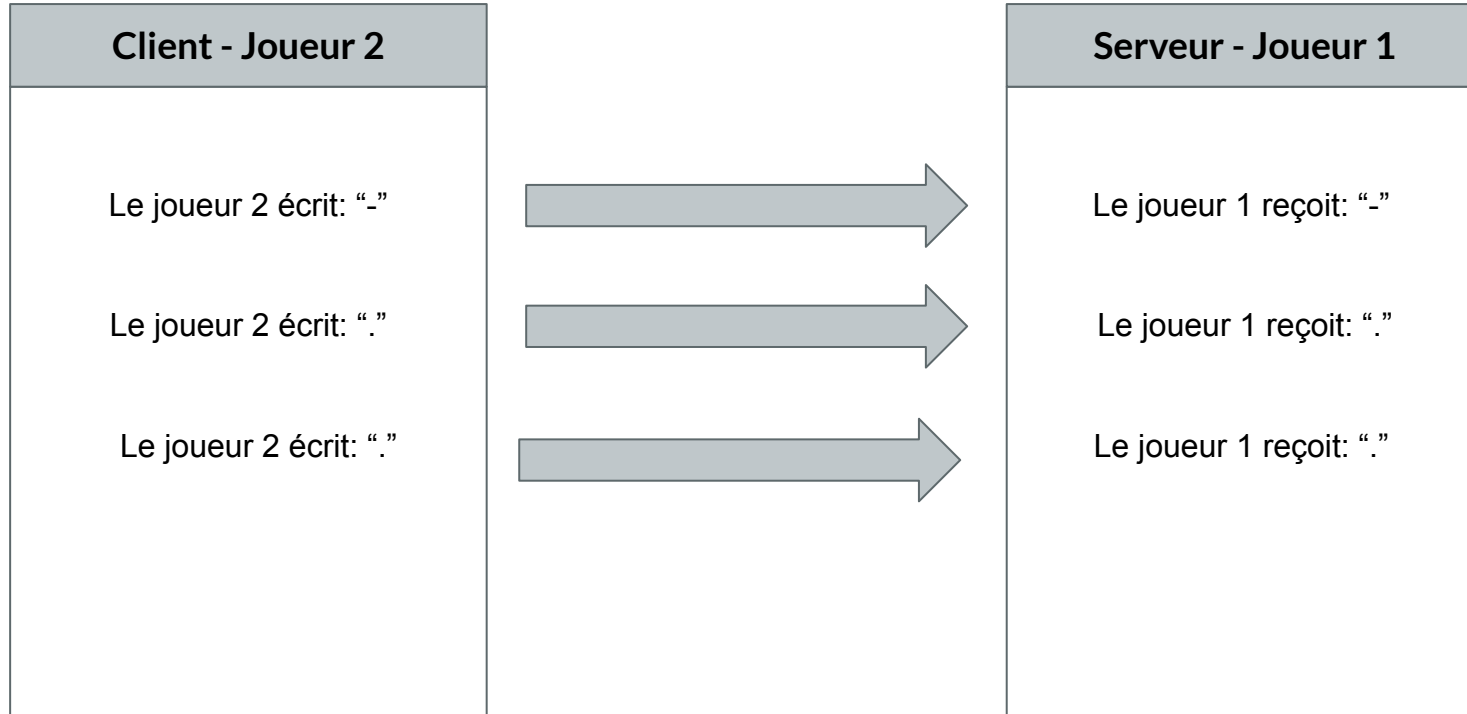
4) Attente du serveur

Client - Joueur 2

Serveur - Joueur 1

Le joueur 1 attend
que le joueur 2 envoie le
code morse

5) Le client envoie le code morse

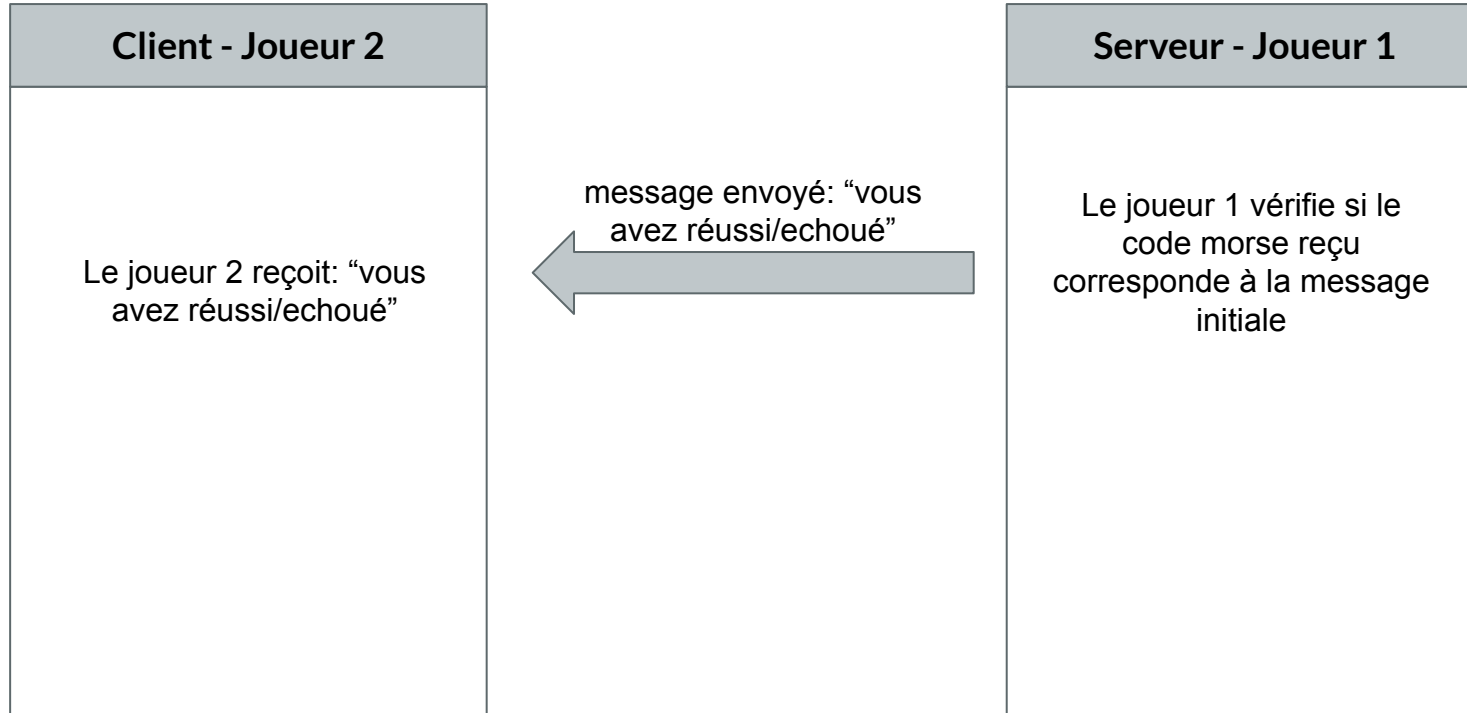


A chaque interaction, le joueur 1 vérifie si le code morse reçu corresponde à la message initiale



Nous allons jouer les sons du côté **serveur** **seulement**

6) Vérification de victoire



Conclusion

- Nous pouvons contrôler qui a la **permission** d'appeler une méthode RPC et son **mode d'envoi**.
- Nous utilisons le noeud `AudioStreamPlayer` pour jouer les sons.
- Les sons peuvent être chargés à partir d'un fichier ou créés par un `AudioStreamGenerator`.

LOG725 - Ingénierie et conception de jeux vidéo

Labo 10 - Conception Sonore + RPC

Gabriel C. Ullmann

École de Technologie Supérieure, Hiver 2024



Le génie pour l'industrie