Fiche Technique de Projet

SimpleEQ - Plugin Audio Temps Réel

BOUCHET Romain (IPSA - Majeure TIE)
Dossier technique récapitulatif

Type : Plugin Audio VST3 / AU	**Framework :** JUCE 7 (Module `juce_dsp`)
Langage : C++	**Architecture :** Traitement temps réel stéréophonique

Description Fonctionnelle

Développement d'un égaliseur paramétrique 3 bandes fonctionnant en temps réel, conçu pour s'intégrer dans une Station Audio Numérique (DAW).

- **Bande 1 (Low Cut) :** Filtre passe-haut à pente variable (12, 24, 36, 48 dB/oct) via cascading de filtres biquad.
- **Bande 2 (Peak) :** Filtre en cloche paramétrique (Gain, Fréquence, Facteur Q).
- **Bande 3 (High Cut) :** Filtre passe-bas à pente variable.

Architecture DSP (Digital Signal Processing)

L'implémentation repose sur le module `juce::dsp` pour optimiser les performances CPU et la facilité d'implémentation.

Pipeline de Traitement

Utilisation de `juce::dsp::ProcessorChain` pour créer une chaîne de traitement statique, typée à la compilation, garantissant une exécution rapide.

— Structure de chaîne : `MonoChain = ProcessorChain<CutFilter, Filter, CutFilter>`.

— Les filtres de coupure (`CutFilter`) sont composés d'une chaîne de 4 filtres IIR d'ordre 2, permettant de cumuler les pentes jusqu'à 48 dB/oct.

— Duplication de l'instance pour le traitement stéréo (`leftChain`, `rightChain`).

Algorithme de Filtrage et Optimisation

— **Coefficients IIR :** Calcul dynamique via `makePeakFilter` et `ButterworthMethod`.

— **Traitement "In-Place" :** Utilisation de `juce::dsp::ProcessContextReplacing` pour éviter les allocations mémoire inutiles pendant le callback audio critique (`processBlock`).

— **Zero-Copy :** Utilisation de `juce::dsp::AudioBlock` pour manipuler les pointeurs de canaux sans copie de données brute.

Gestion de l'État (State Management)

Synchronisation thread-safe entre l'interface utilisateur (GUI) et le thread audio haute priorité.

— **APVTS (AudioProcessorValueTreeState) :** Centralisation de la gestion des paramètres et de la sérialisation (sauvegarde/chargement de presets XML).

— **NormalisableRange :** Définition programmatique des plages de valeurs (mapping linéaire pour les dB, ressenti logarithmique des fréquences fidèle à la compréhension du son par l'oreille humaine).

— **Structure Atomique :** Implémentation d'une structure `ChainSetting` pour extraire les valeurs des paramètres avant le traitement DSP, assurant une séparation claire entre la logique de contrôle et le traitement du signal.

Compétences Techniques Mises en Œuvre

— **C++ :** utilisation des template JUCE, lecture et compréhension de la documentation.

— **Contraintes Temps Réel :** Respect strict du thread audio (aucune allocation dynamique dans `processBlock`).

— **Mathématiques Appliquées :** Implémentation de filtres IIR biquadratiques.