Recap:

- Twitter Notifications: auditory icon of chirping bird, text-to-speech (TTS) used for highest-priority tweets to read author and their message
- E-mail Notifications: earcon, pitch modified based on content summary score, TTS used for highest-priority emails to read author and message
- Text Message Notifications: earcon, pitch modified based on content summary score, TTS used for highest-priority texts to read author and message
- Phone Call Notifications: auditory icon of old phone ringing, TTS used to read caller's name or phone number
- Voicemail Notifications: earcon created through programmatic audio, pitch modified based on content summary score, TTS used to read author and message
- Context Adaptations:
    - Working Out: Notifications above the lowest priority have their volume boosted
    - Walking: Notifications lower than the highest priority have their volume decreased
    - Socializing: Notifications lower than the highest priority low-pass filtered to muffle them, voicemail beep completely muted
    - Presenting: All notifications including TTS muted; highest-priority urgent notifications replaced with a quiet vibrating sound

Architecture:

- Sample Files -> Low-Pass Filter -> Notification Gain -> Master Gain
    - Concurrently, Voicemail Beep Gain -> Master Gain
    - TTS completely independent of Beads processing chain using textToSpeechAudio()
- All notifications are put into 1 PriorityQueue implementing NotificationListener
    - Because TTS is independent of the processing chain, speech plays concurrently with voicemail beep and overlays any other audio currently playing
- All sonification computed within the draw() method
- All notification types activated by default, no context or .json stream activated by default
    - Recommended workflow is from left to right: pick context, configure combination of notifications, pick notification stream

Deviations:

- Twitter, E-mail, and Text message notifications switched from programmatic audio to sample files
    - Twitter notifications use an auditory icon instead of an earcon
    - Why: Sample files are much easier to work with in Processing than using WavePlayers, especially in the context of a dynamic running notification system
        - WavePlayers are much more difficult to stop prematurely as they natively play ad infinitum.
- Pitch/tone/notification type-specific volume modulation for different attributes streamlined to just pitch modulation based on assigned content summary score
    - Why: Too many moving parts and too much modulation made cases of overlapping audio far too cacophonic
- Context-based changes simplified to volume changes to notification gain
    - (Low-pass filter in the Socialized context retained)
    - Why: Subtle changes to TTS volume or different notification categories did not yield improved quality of use; too complex for the relatively little it offered.
- Reverb nixed for certain contexts
    - Why: Reverb often didn't match the background ambiance and felt too artificial; dry sound was closer to a real-life user situation in my opinion
- Priority Queue used for all contexts rather than just Working Out Context
    - Why: Much more convenient/accessible, easier to implement, easier to decipher as the end-user as opposed to risking a constant cacophony of multiple sounds