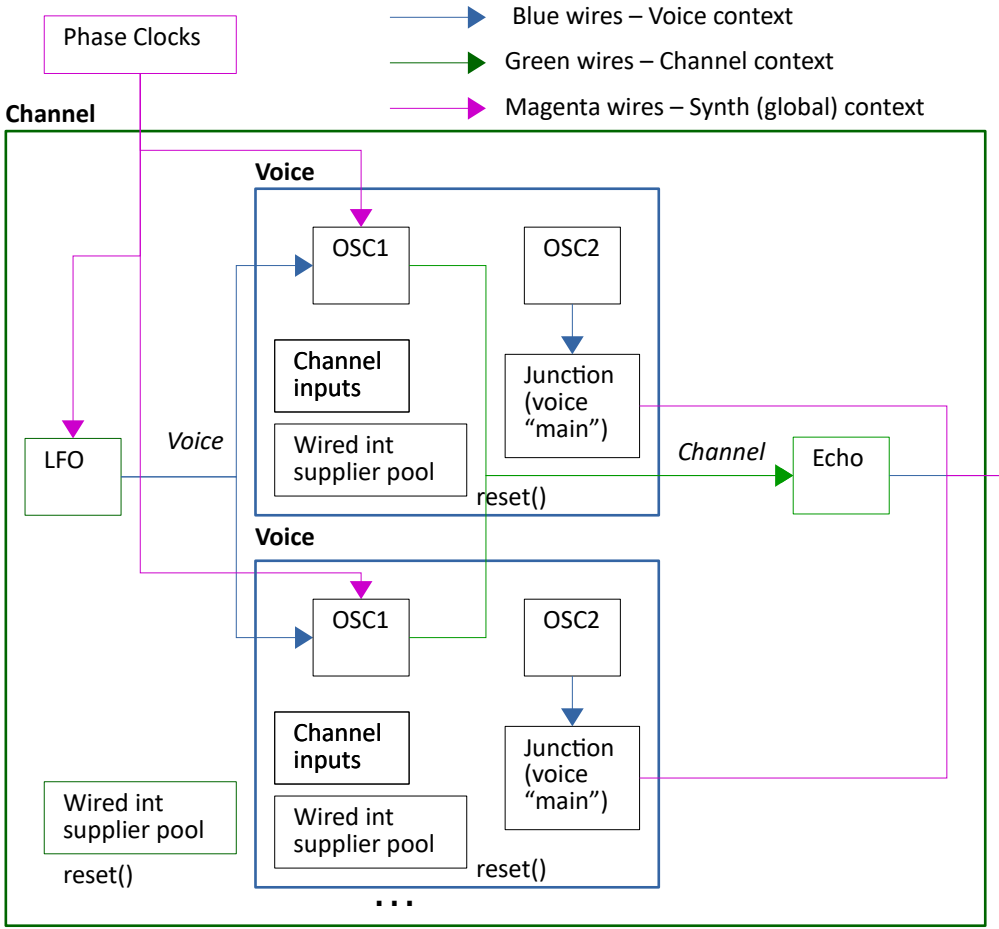
OndeSynth

Ondes – close-up of Voice and Channel Components



Components may exist in both the **Voice** and **Channel** contexts. The Channel context manages all of the voices on a given MIDI channel.

The "wires" are technically drawn backwards (see next slide).

They each have a 'visited' flag that needs to be reset for every sample, by calling

WiredIntSupplierPool.reset()

These pools are kept in both the **Voice** and **Channel** contexts.

The Phase Clocks live in the **Synth** context. The main loop advances these for each sample. When an oscillator goes dormant, it removes its phase clock from the Global collection, and on reactivation must add it in again.

"**Wires**" internally to other Voice Components and to outputs from Channel components to Voice inputs are kept in the **Voice**.

Wires from Channel→Channel Components, and from Voice → Channel Components are kept in the **Channel**.

Voice → Channel connections must be removed when the voice goes inactive, and re-added when it activates again. The **pause()** and **resume()** functions are in charge of that.

The Voice tracks the Voice→Channel connections in a list of **ChannelInputs**, which pair from the Voice Component's output to a Channel Component.

Ondes – close-up of Connections

The arrow is technically backwards on the previous slide. When the "output" of LFO is connected to the "input" of OSC, it means that OSC has a Lambda in its List (**inputs**) that will return the current value of the LFO.

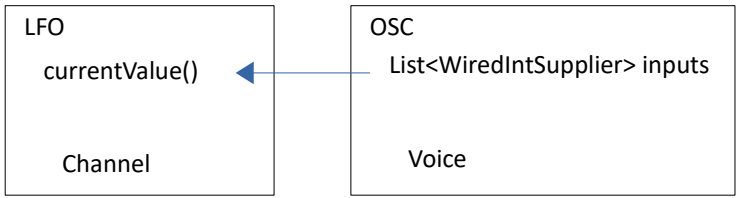
The voice is thus a directed (possibly cyclic) Graph starting with the Main Mix, going backwards from the perspective of common audio circuitry. The Mixer pulls rather than the sound generators pushing.

Because it can be cyclic (for FM) the 'visited' flag on each WiredIntSupplier must be reset for each sample.

LFO output connected with **OSC** input

When the voice is inactive, OSC's output never is polled, so it never calls currentValue()

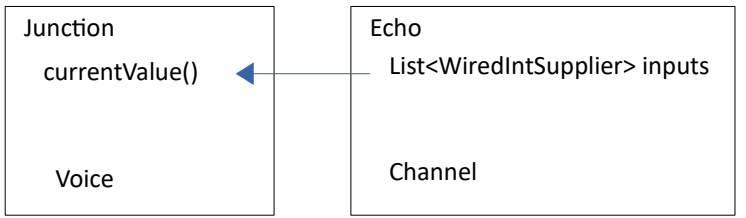
Therefore, these connections (at the Voice level, from a Channel output to a Voice input) can remain when the voice is inactive.



Junction output connected with **Echo** input

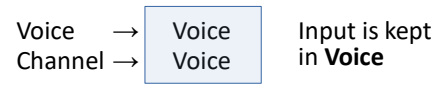
This connection activates the voice, so it must be removed to deactivate the voice. The Voice remembers it in **channelnputs**, and must:

- disconnect on **pause()** and
- reconnect on **resume**()

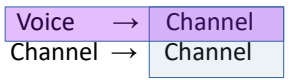


Voice → **Channel** connections need to be disconnected on **pause()** and reconnected on **resume().**

The other combinations should be OK without requiring to be unplugged.



Input is kept in **Voice**

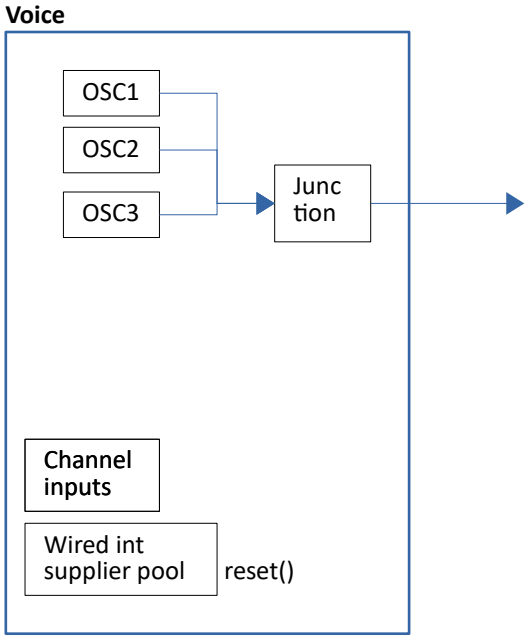


Input is kept in **Channel**

Connections to a Voice-level component will be unused when the voice is inactive, so they can remain connected.

Channel-to-Channel connections only ever **resume()**. They do not **pause()**.

Ondes – Junction output



Each **Voice** has a single output in the form of a Junction, that all the sub-outputs of the Voice are inputs to. So, other than the channel-level components, the only output to disconnect on **pause()** and reconnect on **resume()** is the Junction.

For output to **Channel**-context components, they connect to the actual "main" rather than back to the junction, because otherwise all of the Voice Junctions would get the output from all the other ones.

Channel-context components only ever **resume()**. They never **pause()**