Advancing Time

- ChucK time stands still until you "advance" it
- two semantics for advancing time
  - `chuck to now`  
    - `1 : second => now;`
  - `wait on event`  
    - `event => now;`
- you are responsible for keeping up with time
- timing embedded in program flow
- `time == sound`

Concurrency

- implemented using "shreds"
  - resemble non-preemptive threads
  - automatically synchronized by time!
- possible to easily write truly parallel, sample-synchronous audio code
- can work at low and high level
  - fine granularity == power and control
  - arbitrary granularity == flexibility and efficiency
- a solution to the control-rate issue

ChucK Virtual Machine

- On-the-fly compiler
- Process
- Shreduler
- ChucK Virtual Machine Audio Engine I/O Manager
- Code ("foo.ck", "bar.ck")

Runtime-1
Virtual Machine

- Bytecode interpreter
  - 100+ ChucK bytecode instructions
- Shreduler
  - User-level non-preemptive shreduling
  - Uses timing and event information
  - Coordinate interpreter with audio computations

Multi-Shredded Shreduling Algorithm
Audio Computation
- controlled by shreds
- computes audio outside of shreds
  - traverses the global UGen graph from well-known sinks, such as ‘dac’
- UGens and UAnae cache the latest computation

The Audicle
- visualization (audio, runtime stats, shreduling, etc.)
- insight into real-time, live programs
- different views of programs
  - syntax (code, objects)
  - concurrency (shreds)
  - time and timing (time, timing)
  - semantics (type, coming soon)
- different view of programming process
  - “Program monitoring as performance art” - Andrew Appel
- new way of thinking about real-time and live audio programming