

# Computer Programming For Musical Applications II

Tutorial 08

Recording in SuperCollider

21st November, 2008



# Tutorial Overview

- Review of last weeks exercise
- Recording Audio in SC
- Loading Audio in SC
- Exercises



# Review from last week

Task:

Create 5 different SynthDefs using at least 3 different Ugens in each. Try and make each SynthDef sonically unique.

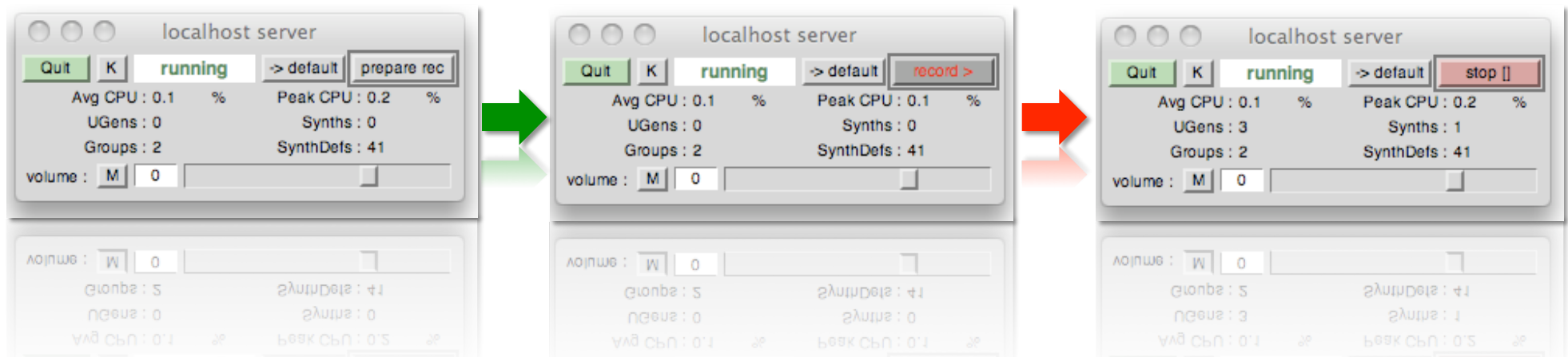
Play with the Reverb and Delay UGens to get some more interesting sounds. Remember to use arguments to make the SynthDef's more adaptable.....



# Recording Audio in SuperCollider

There are several different ways of saving your audio in SC:

The most basic is to just hit the **prepare rec** button on the server GUI, this will change the button to **rec** which will start to record any audio being played on **that** server until you hit the **stop** button



This will save the audio to a default file, it is best to setup this file prior to saving your audio....



# Recording Audio in SuperCollider

The default settings for recording are:

44.1k, 32bit, stereo file that will be save in the recordings folder in  
SC

It is better to set these however to something more specific:

```
(  
Server.default.recSampleFormat = "int16";  
Server.default.recChannels = 2;  
Server.default.prepareForRecord("TestTrack.aif");  
)
```



# Recording Audio in SuperCollider

You can also start and stop recording via the command line:

```
(  
Server.default.record;  
  ...make some beautiful music...  
Server.default.stopRecording;  
)
```

You must prepare the server for recording first however prior to using the above code

```
Server.default.prepareForRecord("TestTrack.aif");
```



# Recording Audio in SuperCollider

Task: Create a SynthDef that can create some audio, play this SynthDef and record what you play to a file



# Loading Audio in SuperCollider

The easiest way to load some audio into SC is to place the audio into a buffer, you can then playback and loop through the buffer when ever you need to. Use the read method in the Buffer class to do this -

```
Buffer.read(server, path, startFrame, numFrames, action, bufNum );
```

server – The server on which to allocate the buffer

path – A string representing the path of the soundfile to read

startFrame – The first frame of the soundfile to read. Default = 0

numFrames- The number of frames to read. Default = -1

action – A function to be evaluated once the file has loaded

bufNum – An explicitly specified buffer path



# Loading Audio in SuperCollider

You do not however have to specify all of these arguments, the server and path arguments are normally enough:

```
b = Buffer.read(s, "sounds/DrumLoop.wav" );
```



# Loading Audio in SuperCollider

You can then use the PlayBuf class to play you audio, placing this in a SynthDef is an easy and efficient way of doing this:

```
(  
x = SynthDef("myAudio",{ arg out = 0, bufNum;  
                        Out.ar( out,PlayBuf.ar(2, bufNum,  
                        BufRateScale.kr(bufNum)))  
                        });  
)
```

You can then play the audio in the buffer using play:

```
x.play(s,[\bufNum, b]);
```

As always, remember to free up any memory that has been allocated

```
x.free; b.free;
```



# Loading Audio in SuperCollider

Task: Load some audio into a buffer and play it using a SynthDef.  
Some code you may need:

```
b = Buffer.read(server, path);
```

```
(  
x = SynthDef("myAudio",{ arg out = 0, bufNum;  
                        Out.ar( out,PlayBuf.ar(2, bufNum,  
                        BufRateScale.kr(bufNum)))  
                        });  
)
```

```
x.play(s,[\bufNum, b]);
```

```
x.free; b.free;
```



# Exercises

Task: Work on some pattern tasks that can help you with your 2<sup>nd</sup> project

