

Mutating on real-time and non-real-time

- Realtime events should be processed instantly
 - We choose the realtime thread to be the mutating thread


```
struct SourceList {  
    std::array<const float*, MAX_SOURCES> buffers = {};  
    int numSources = 0;  
};
```

```
RealtimeMutatable<SourceList> sharedSourceList;  
AsyncCaller realtimeThreadCaller;
```

- Realtime audio thread which mixes audio from multiple sources
- User can add/remove sources via GUI (i.e. non realtime thread)
- Sources can also be added/removed from realtime event streams (i.e. realtime thread)



Mutating on realtime and non-realtime

```
struct SourceList {  
    std::array<const float*, MAX_SOURCES> buffers = {};  
    int numSources = 0;  
};
```

```
RealtimeMutable<SourceList> sharedSourceList;  
AsyncCaller realtimeThreadCaller;
```

- Realtime audio thread which mixes audio from multiple sources
- User can add/remove sources via GUI (i.e. non realtime thread)
- Sources can also be added/removed from realtime event streams (i.e. realtime thread)
- Realtime events should be processed instantaneously
 - We choose the realtime thread to be the mutating thread

Mutating on realtime and non-realtime

```
void addSource (const float* src) {  
    if (! isRealtimeThread()) {  
        realtimeThreadCaller.callAsync([src] () { addSource (src); });  
        return;  
    }  
    RealtimeMutable<SourceList>::ScopedAccess<true> sourceList (sharedSourceList);  
    assert (sourceList->numSources < MAX_SOURCES);  
    sourceList->buffers[sourceList->numSources++] = src;  
}
```