API Specification

Events

REQUIRE

```
var events_ = require('events');
```

CREATE EMITTER

```
var emitter = events_.emitter(name_space, binder);
```

`name_space:`

The name space of the event must be well known and must conform to a known set of events. The following table lists the name spaces:

Name Spaces for Events	
'framework:layout'	All events related to layout changes
'framework:attendees'	All events related to participants
'av:connection'	All events by the AV module, related to connection states
'av:qos'	All events by the AV module, related to the QoS situation

`binder:`

A string indicating the module requesting for the emitter.

Notes

If the name_space is unrecognised, a valid emitter is still returned. This is allow to for flexibility for modules to create custom events, which can be recognised by paired modules. In general this shouldn't be done though.

If an emitter for the specified name_space and binder already exists, then NULL is returned. It should be possible to have multiple emitters for the same name_space.

EMIT EVENT

```
emitter.emit(__event, data);
```

`__event:`

The specific event to be emitted. In the global event name space, the actual event emitted will be 'name_space:event'. The events should be fixed and well known, since other modules will rely on these published events to do their work.

For example, an attendance plugin would use the events in the namespace 'framework:attendees' to keep track of the attendance. Or a signal indicator plugin could use the namespace 'av.qos' to keep track of the audio video quality.

A list of well known events is listed at the end of this section.

`data:`
Any arbitrary data, which will be passed to all the listeners.

Notes
Never fails.

BIND TO EVENT

```
events_.bind(namespace, callback, binder);
```

`namespace:`

The binding is done on a namespace, so that all events raised in that namespace will trigger the callback.

`callback:`

The signature of the callback would be as follows:

```
function callback (event, data) {}
```

`binder:`

A string indicating the module which is binding to the event name space.

Notes

No check is done to see if the namespace is valid or recognised. Again this is to allow a flexibility for modules to define custom events for mutual communication.

A duplicate for the same namespace results in an error (returns **false**).

UNBIND EVENT

```
events_.unbind(namespace, binder);
```

Note

Unbinds an event, which was bound by the 'binder'. Returns **true** on success or **false** if no bindings for 'binder' are found or if the 'namespace' does not exist.

Events

The following lists the well known events, per namespace, per module.

Framework	framework.layout
default	
av-fullscreen	Full screen AV with primary & secondary videos.
av-fullscreen-pri	Full screen AV with only primary video
av-tiled	Full screen AV with tiled (surveillance like) mode
work-fullscreen	Full screen whiteboard
work-fullscreen-av-inset	Full screen whiteboard with AV (primary only) showing as inset in a corner

Framework	framework.attendees
in	New attendee (the accompanying data needs to be defined)
out	Attendee left the session (data needs to be defined)

Audio-Video	av.connection
connected	AV connected to server
disconnected	AV disconnected from server

AV QoS anouncement	av.qos
update	QoS params update (data needs to be defined)