PeerConnection errors

at 2015 f2f

Something goes wrong with DTLS but not ICE

If something goes wrong with DTLS, but not ICE, then ICE is connected but DTLS is not, and the JS will get the wrong idea about the connectivity of PeerConnection. Let's fix it:

```
partial interface PeerConnection {
  readonly attribute PeerConnectionState connectionState;
  attribute EventHandler onconnectionstatechange;
}

enum PeerConnectionState {
  disconnected,
  connecting,
  connected,
  failed
}
```

Rules for DTLS state aggregation

1. if any failed => failed
2. else if any connecting/checking/disconnected => connecting
3. else if all new (or closed) => new
4. else if all connected/completed (or closed) => connected
5. else => connecting (mix of new and connected and checking and disconnected)

But what went wrong?

There are lots of things that can go wrong. Some are fatal, some are not. The application developer needs to know 1. Something bad happened, 2. Roughly what it was. 3. How severe it was. Something like this:

```
partial interface PeerConnection {
  // The PeerConnection can't continue.  You get a DOMString message.
  attribute EventHandler onfatalerror;
  // The PeerConnection can continue.  You get a DOMString message.
  attribute EventHandler onwarning;
}
```

onfatalerror/onwarning examples

```
pc.onwarning = function(evt) {
  // Maybe something like "Hey, your TURN server isn't responding"
  // Or perhaps "Resolution lowered because the CPU is pegged"
  console.log(evt.message);
}

pc.onfatalerror = function(evt) {
  // Maybe something like "Couldn't generate a crypto cert: ..."
  // Or "Failed to initialize SRTP: ..."
  console.log(evt.message);
  goToBrokedUI();
}
```

Questions

- Should we have two events or one?
- Should we have error codes?
- Should we have error categories?

Or... just get rid of it altogether?