Message Passing

Data Synchronization

An Interlude by Quentin Cheshire

Proudly Representing West RTC

Firebase

mozilla

What's all this fuss about?

AJAX

was all the rage...

...which evolved into

Long-polling

lovingly known as "Comet"

Lo and Behold!

HTML

WebSockets

Introducing...

WebRTC

Data Channels

Why Bother?

Users don't like waiting

Are obsolete

The Realtime Web is Here!

How will you use the force?

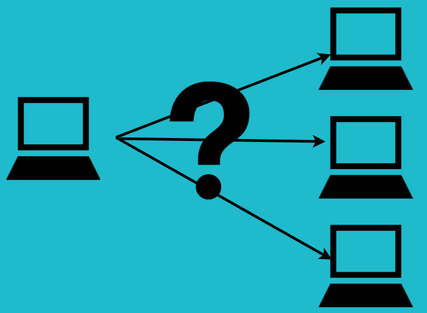
It's just a messaging channel!

Have you thought this through?

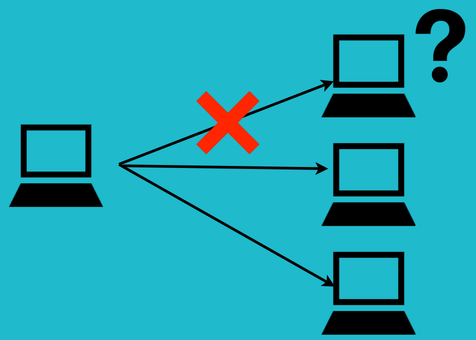
Persistence

Fault Tolerance

Scaling



Consistency



Security

Message Passing is a Primitive

Are better tools in order?

"All problems in computer science can be solved by another level of ~~indirection~~ abstraction"

Data Synchronization

Most apps observe and modify *data*

That data reflects *state*

Your API should be built around this!

An Example: Chat

```
var channel = new MessageChannel();

channel.subscribe(function(msg) {
  receivedNewMessage(msg);
});

function sendMsg(msg) {
  channel.send(msg);
}
```

```
var dataStore = new DataStore();

dataStore.on("new_row", function(msg) {
  receivedNewMessage(msg);
});

function sendMsg(msg) {
  dataStore.addRow(msg);
}
```

An Example: Chat

Archival

```
var dataStore = new DataStore().limit(10);

dataStore.on("new_row", function(msg) {
  receivedNewMessage(msg);
});

function sendMsg(msg) {
  dataStore.addRow(msg);
}
```

An Example: Chat

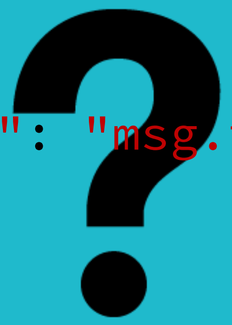
Fault Tolerance

```
var dataStore = new DataStore();

dataStore.on("new_row", function(msg) {
  receivedNewMessage(msg);
});

function sendMsg(msg) {
  dataStore.addRow(msg);
}
```

An Example: Chat

Security

```
{
  ".read": "msg.to == auth.id"
}
```



```
var dataStore = new DataStore();

dataStore.on("new_row", function(msg) {
  receivedNewMessage(msg);
});

function sendMsg(msg) {
  dataStore.addRow(msg);
}
```

Conceptually Simple

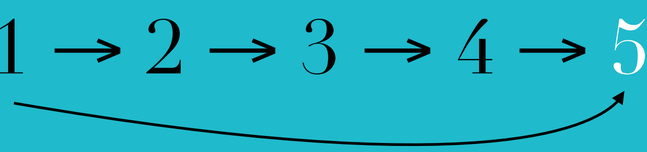
There's a lot of complexity in turning a stream of messages into usable state

→ ? → ? → ?

Why not just directly store state?

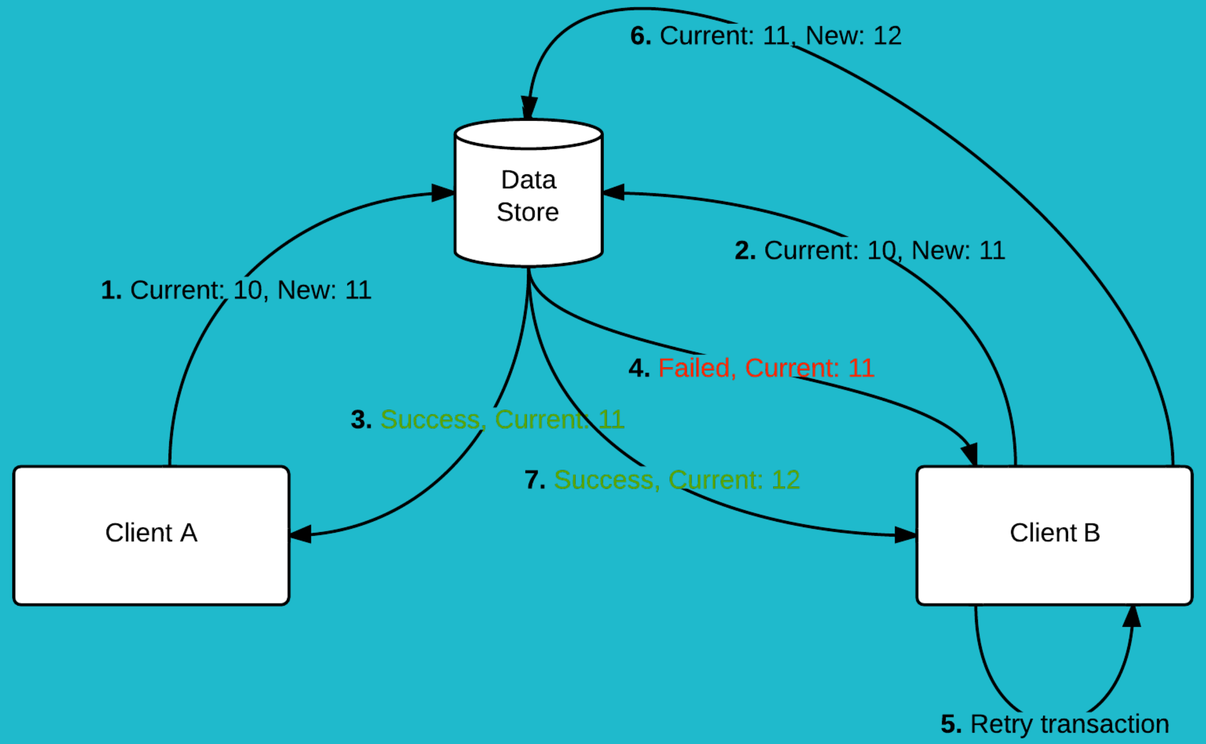
More Efficient

You have the flexibility to combine operations
New clients only care about the latest state



Automatic Merge Behavior

Conflicts will typically require several messages to resolve
With a data abstraction, it can be a core primitive



Data Sync **not** Message Passing

Don't let the primitives dictate how your application code is structured.
Build the abstractions you need (Or use one of the available ones!)

Store state - don't pass messages

(Except when that's really what you want to do)

www kix.in
github anantn
twitter @anantn
email anant@firebase.com