

Stop calling Matrix APIs! (directly)

**Alexey “Kitsune” Rusakov
Spec Core Team; Quotient/Quaternion**

`matrix:u/kitsune:matrix.org`

~~The Client-Server API is easy~~

We lied.

All these years.

(and you already know if you tried it; you can leave now)

The Client-Server API

To send a message:

```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
" -H "Authorization: $token"
```


```
{  
  "event_id": "YUwRidLecu"  
}
```

The Client-Server API

To send a message:

```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
-H "Authorization: $token"
```

```
{  
  "event_id": "YUwRidLecu"  
}
```



Look ma, I have
a bag full of
assumptions!

The Client-Server API

To send a message:

Homeserver discovery?..



```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
-H "Authorization: $token"
```

```
{  
  "event_id": "YUwRidLecu"  
}
```

The Client-Server API

To send a message:

```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
" -H "Authorization: $token"
```



Login?..

```
{  
  "event_id": "YUwRidLecu"  
}
```

The Client-Server API

To send a message:

```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
-H "Authorization: $token"
```



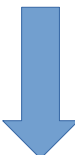
Copy-paste the endpoint
path from the spec?..

```
{  
  "event_id": "YUwRidLecu"  
}
```

The Client-Server API

To send a message:

```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
-H "Authorization: $token"
```



How do I know?..

```
{  
  "event_id": "YUwRidLecu"  
}
```


The Client-Server API

To send a message:



Normally a parameter
– escaping rules?..

```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
" -H "Authorization: $token"
```

```
{  
  "event_id": "YUwRidLecu"  
}
```

The Client-Server API

To send a message:

```
$ curl -XPOST -d '{"msgtype":"m.text", "body":"hello"}'  
"https://alice.com:8448/_matrix/client/api/v3/rooms/$ROOMID/send/m.room.message"  
-H "Authorization: $token"
```

```
{  
  "event_id": "YUwRidLecu"  
}
```



E2EE?..

The Client-Server API

It's easy to send the message, **assuming that:**

- ✓ you know the homeserver URL
- ✓ already logged in
- ✓ have done E2EE keys dance if necessary
- ✓ know the room id (room alias won't work)
- ✓ haven't made typos in metadata

The Client-Server API

To receive a message:

```
curl "https://alice.com:8448/_matrix/client/api/v3/sync" -H "Authorization: $token"
{
  "rooms": {
    "join": {
      "!SjPTfpmlqzprCNEXdF:bellerophon": {
        "timeline": {
          "events": [
            {
              "type": "m.room.message",
              "sender": "@matthew:bellerophon",
              "content": {
                "msgtype": "m.text",
                "body": "test"
              },
              "event_id": "$15582798620qPlC0:bellerophon",
              "origin_server_ts": 1558279862446,
              "unsigned": {
                "age": 5006654475
              }
            }
          ]
        }
      }
    }
  },
  "next_batch": "s281_8540_0_144_235_1_155_103_1"
}
```

Most of the assumptions for sending plus...

...pretty much never one event in one room

**It's always easier to talk in
native language**

...but you may need an interpreter

SDKs to the rescue

A good SDK gives you:

- Language-native data structures (fewer typos, less or no hand-written JSON)
- Language-native control abstractions (async execution, event loops, callbacks, algorithms, ...)
- Code/inputs validation (static/dynamic)
- Higher-level Matrix-specific abstractions (login flows, key management, event persistence, ...)
- Documentation and IDE integration

Python: matrix-nio

[matrix]

```
import matrix-nio
import getpass

async def main():
    client = AsyncClient(homeserver, user_id)
    pw = getpass.getpass()
    resp = await client.login(pw, device_name=device_name)
    # ...
    if client.should_upload_keys:
        await client.keys_upload()
    # ...
    await client.room_send(room_id, message_type="m.room.message",
                           content={"msgtype": "m.text", "body": "hello"})
    # ... (add callbacks you need)
    await client.sync_forever(timeout=30000, full_state=True)
```



:|

<https://github.com/anoadragon453/nio-template>

Rust: matrix-sdk

```
use matrix_sdk::{
    Client, config::SyncSettings,
    ruma::{user_id, events::room::message::SyncRoomMessageEvent},
};

#[tokio::main]
async fn main() -> anyhow::Result<()> {
    let alice = user_id!("@alice:example.org");
    let client = Client::builder().server_name(alice.server_name()).build().await?;

    client.matrix_auth().login_username(alice, "password").send().await?;

    let content = RoomMessageEventContent::text_plain("🎉🎊🎋🎌 let's PARTY!! 🎊🎋🎌");
    room.send(content).await.unwrap();

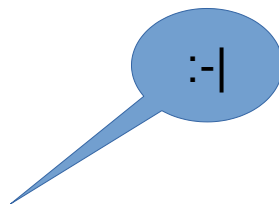
    // (add event handlers)
    client.sync(SyncSettings::default()).await?;

    Ok(())
}
```

<https://github.com/matrix-org/matrix-rust-sdk/tree/main/crates/matrix-sdk>

C++: Quotient

```
#include <connection.h>
#include <room.h>
#include <QCoreApplication>
int main(int argc, char* argv[]) {
    QCoreApplication app(argc, argv);
    const auto* userMxid = argv[1];
    const auto* password = argv[2];
    const auto* deviceName = argv[3];
    const auto* roomId = argv[4];
    using namespace Quotient;
    auto* c = new Connection(&app);
    c->loginWithPassword(userMxid, password, deviceName);
    app.connect(c, &Connection::connected, c, [c] {
        c->syncLoop();
        c->sendMessage(roomId,
            RoomMessageEvent(u"hello"_s, MessageEventType::Text));
    });
    Quotient::connectSingleShot(c, &Connection::syncDone, c, [c] {
        // event handlers go here
    });
    return app.exec();
}
```



Conclusions

- Using an HTTP API (from the command shell or otherwise) is cumbersome and error-prone
- Matrix APIs are a kind of more complex HTTP APIs – using those directly may quickly become a nightmare unless abstracted
- SDKs help to alleviate the problem
 - They usually can't eliminate complexity entirely
 - They also frequently trail behind in features by a version or two
 - But with SDKs you don't have to re-implement the same beaten patterns of low-level interactions over and over any more, and that's 80% of the cases

[matrix]

[matrix]

Thank you!

matrix:u/kitsune:matrix.org
<https://fosstodon.org/@kitsune>