

Integrations at Work

Bringing Workplace Apps into Element.

Brief introductions!

Will (Half-Shot) is the Integrations Tech Lead @ Element

David Langley is Team Manager for Integrations @ Element

And shoutouts to **Amsha, Danielle, Sirius, Rick,** and **Zack** who have all contributed so much to making this happen!

**We have ~11 minutes to catch you up on a
years worth of innovation.**

Link Previews are **Pretty** and **Functional**



hsdebug

11:33

<https://cfp.2026.matrix.org/matrix-conference-2026/talk/CJXTB9/>

The Element Module Workshop The Matrix Conference 2026

Hey! You may have just seen our talk about the future of integrations, or didn't see the point and hung around the bar. No matter! You will want to come to this workshop where we will teach you how



hsdebug

11:33

<https://cfp.2026.matrix.org/matrix-conference-2026/talk/CJXTB9/>



The Element Module Workshop The Matrix ...

Hey! You may have just seen our talk about the future of ...



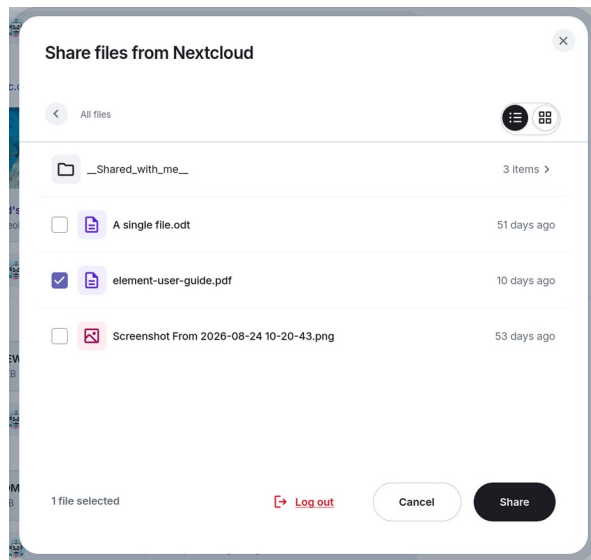
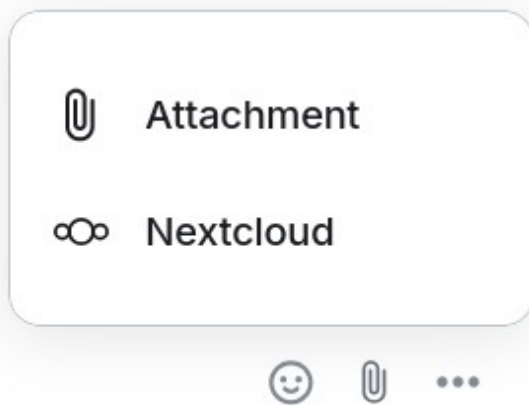
Authenticated previews? Yes we do!

View the context of a Nextcloud file without leaving your timeline!



Sharing a file from elsewhere? Easy as a click

Starting with **Nextcloud**, but who knows where next!



Doc viewers

We want you to be able to DO more in Element.

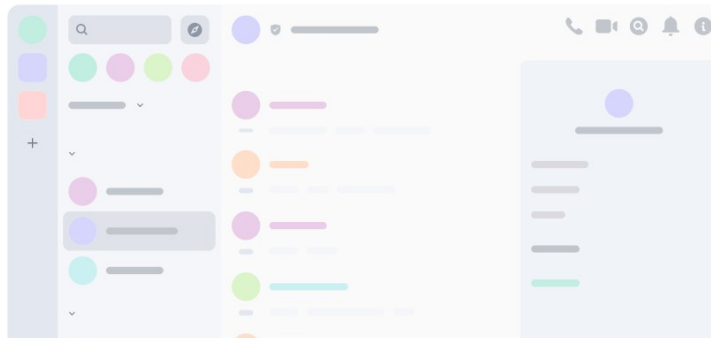
Why download a file you can easily view, securely, in your client.



Get started with Element.

Get up and running in just a few minutes!

Element is an end-to-end encrypted secure messenger and collaboration app. **Talk to friends, family and colleagues** through the open and decentralised Matrix network.



Doc viewers

Yes we have a Markdown one too :)



README.md



conference-bot

The conductor for your orchestra™ Wconference

Development

1. `yarn install`
2. Copy `config/default.yaml` to `config/development.yaml` and edit accordingly.
3. `yarn start:dev`

The project is a TypeScript bot based off the concepts of [Mjolnir](#), using [matrix-bot-sdk](#) as a base.

Production (Docker)

TODO:

- Information for what the bot does

This bot is best deployed with Docker. Note that the steps required to get the bot into production are complicated.

1. First, register a user on your homeserver for the bot to use. In Synapse, this would be `register_new_matrix_user`.
2. Set up any profile data for that user (displayname and avatar).
3. On your homeserver, prepare and install an appservice registration like so:

```
id: conference_bot # Can be any helpful identifier
hs_token: CHANGE_ME # hs_token and as_token must be unique and secret
as_token: CHANGE_ME
namespaces:
  users:
    - exclusive: true
      regex: '@yourbot:example.org' # Set this to your bot's user ID.
  aliases:
    - exclusive: false
      regex: '#.*:example.org' # Change the domain to match your server.
  rooms: []
url: null # Important! The bot doesn't receive transactions, so set this explicitly to null.
sender_localpart: not_confbot # Use something other than your bot's localpart.
rate_limited: false # Important! The bot is noisy.
```

4. Create `/etc/conference-bot` or wherever you are comfortable with mapping a volume for the bot.
5. Copy `config/default.yaml` from this repo to `/etc/conference-bot/config/production.yaml` and edit accordingly.
6. Run the container with Docker. For example: `docker run -d --rm --name conference-bot -v /etc/conference-bot:/data -p 8080:8080 matrixdotorg/conference-bot`
7. Reverse proxy port `8080` as appropriate, using SSL if needed.

and

Doc viewers

Yes it *does plaintext

*we actually need to merge this PR but it'll be in real soon now!



```
1 # conference-bot
2 The conductor for your orchestra^Wconference
3
4 ## Development
5
6 1. `yarn install`
7 2. Copy `config/default.yaml` to `config/development.yaml` and edit accordingly.
8 3. `yarn start:dev`
9
10 The project is a TypeScript bot based off the concepts of [Mjolnir](https://github.com/matrix-org/mjolnir),
11 using [matrix-bot-sdk](https://github.com/turt2live/matrix-bot-sdk) as a base.
12
13 ## Production (Docker)
14
15 TODO:
16 * Information for what the bot does
17
18 This bot is best deployed with Docker. Note that the steps required to get the bot into production are
19 complicated.
20
21 1. First, register a user on your homeserver for the bot to use. In Synapse, this would be
22 `register_new_matrix_user`.
23 2. Set up any profile data for that user (displayname and avatar).
24 3. On your homeserver, prepare and install an appservice registration like so:
25 ```yaml
26 id: conference_bot # Can be any helpful identifier
27 hs_token: CHANGE_ME # hs_token and as_token must be unique and secret
28 as_token: CHANGE_ME
29 namespaces:
30   users:
31     - exclusive: true
32       regex: '@yourbot:example.org' # Set this to your bot's user ID.
33   aliases:
34     - exclusive: false
35       regex: '#.*:example.org' # Change the domain to match your server.
36 rooms: []
37 url: null # Important! The bot doesn't receive transactions, so set this explicitly to null.
38 sender_localpart: not_confbot # Use something other than your bot's localpart.
39 rate_limited: false # Important! The bot is noisy.
40 ```
41 4. Create `/etc/conference-bot` or wherever you are comfortable with mapping a volume for the bot.
42 5. Copy `config/default.yaml` from this repo to `/etc/conference-bot/config/production.yaml` and edit
43 accordingly.
44 6. Run the container with Docker. For example: `docker run -d --rm --name conference-bot -v /etc/
45 conference-bot:/data -p 8080:8080 matrixdotorg/conference-bot`
```

xpand

Your Agenda

A

h

📅

+

Upcoming events

Today

Open In External Calendar

Mon

Tue

Wed

Thu

Fri

Sat

Sun

12th

13th

14th

15th

16th

17th

18th

Thursday 15th Today

7:30 AM
8:00 AM

Breakfast

8:30 AM
9:00 AM

Second Breakfast

9:00 AM
10:00 AM

Procrastinate writing talk

10:00 AM
11:00 AM

Team meeting
don't mention that you haven't finished the talk

12:00 PM
1:30 PM

A slightly long lunch

3:00 PM
4:00 PM

Give talk!

5:00 PM
6:00 PM

Frisbee!

6:00 PM
7:00 PM

Nurse Frisbee injury with drink

🗨️ Join room

Event details

×

Frisbee!

Thursday 15th, 5:00 PM – 6:00 PM

The park!

Organiser

Adrian Castillo

Guests (4)

Cade Jackson

Dominick Stout

John Smith

Serena Kelley

Awaiting reply

Awaiting reply

Awaiting reply

Awaiting reply

**Sounds like a lot.
Do you have *more* plans?**

OpenProject in the timeline.

Work is decided in the room and recorded somewhere else.

Step 1

"Is that done?"

Someone in the room asks about a work item.

Step 2

Another tab

Someone else opens the tracker and reads the status.

Step 3

Typed back in

They come back and type the status into the room.

Step 4

Already stale

The room now holds a copy of the truth that is going out of date.

Work items live in the timeline, up-to-date and ready to act on.

Enough context to be useful, and enough actions to be worth having.

- Status, assignee and priority stay current as work changes
- Quick Actions such as "Assign to me" and "Mark done"
- Connect a project to a room, and pick which updates it posts
- OpenProject stays the source of truth

H Member DEV user created work package 48: "Ayyhh"



OP OpenProject · [dev] Work package sharing · #48

Ayyhh

New Bug

Assigned to Project admin DEV user

Due 14 Oct

View package

OpenProject connection verified

H Member DEV user created work package 48: "Ayyhh"

Sovereign AI in your rooms.

AI is a powerful tool, but it is still the wild west.

People are seeing real productivity gains. IT, security and compliance teams have good reason to be cautious.

Data control

Where does the data go?

Who keeps it, and who can see it?

Audit and authorisation

Who asked what?

What was done on their behalf, and what came back?

Discovery and know-how

What tools exist?

What can they do, and how do teams share what works?

Matrix gives people and agents a safe place to work together.

Data control

You decide what leaves the room

Conversations stay encrypted. You choose where requests go, such as a self-hosted model or a provider you trust.

Audit and authorisation

Every action is on the record

Each request, action and response is kept in the room's history.

Discovery and know-how

What works is shared

People work with agents in shared rooms, so good workflows are visible and easy to reuse.

The tools we build for people become tools for agents.

Today a person uses a Quick Action such as “Assign to me”. Next, an agent can take the same action when asked, with that person’s permissions.

Hard problem

Authentication

How Element and other systems trust each other.

Hard problem

Identity and permissions

How room members map to access in other systems.

Hard problem

Actions

How work gets done on a user’s behalf, under your policies.

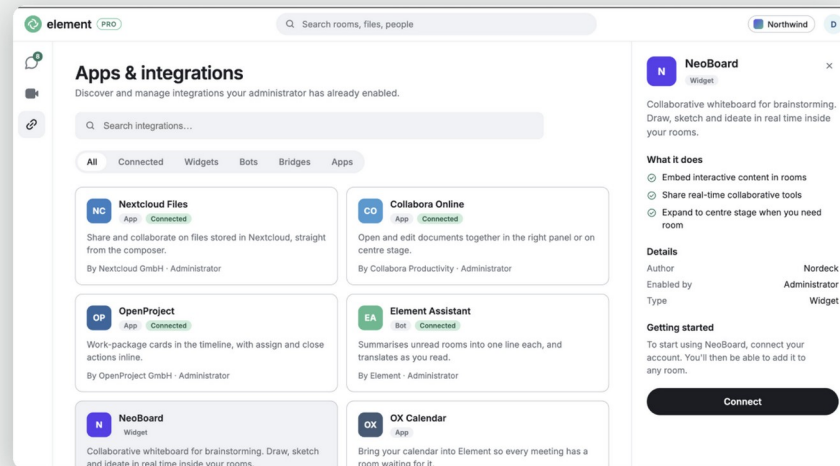
Same actions, same controls, whether a person or an agent takes them.

A home for your extensions.

An extension nobody can find is an extension nobody uses.

Discover, understand and turn on the tools you need, inside Element.

- Discover what's available to you
- Understand what each extension does and who made it
- Turn on what a room needs, where you have permission
- Admins stay in control of what's on offer



We can't do this alone.

This year we brought workplace apps and productivity features into Element, and laid groundwork that others can build on.

Technical

Shared primitives

How extensions appear, authenticate and act on a user's behalf.

Design

Consistent patterns

Cards, actions and connections that feel like Element.

Product

Discovery

Learning what customers and partners need, testing small first.

My gosh that's a lot.

How do you build these?

Can I build these?

The Element Web Module API

The v2 version is really something.

But come to the workshop to hear about that!

It's in ~20 minutes



Something missing?

Come chat to us! We'd love to hear how we could make your client work for you!

Thank you

Oh wait

DEMO TIME!

Actually thank you

Please come find us if you have questions

Or message @Half-Shot:half-shot.uk :)