Abdullah Tarawneh

Technical Writing Portfolio

Notable work

Mastodon documentation

Currently live at https://docs.joinmastodon.org

Pretty much everything except the installation guide was written by me. Selected samples are available in this portfolio as well.

Senior Technical Writer / Product Knowledge Curator, Trilogy

Maintained private knowledge-bases for a company that owns and operates numerous products.

This portfolio includes two sample articles produced during the interview process – one solution article, and one troubleshooting article.

Assorted sample writings

I am also attaching a blog post that was written in the course of interviewing or contract-for-hire work.

Table of Contents

Sample 1 – What is Mastodon?

https://docs.joinmastodon.org

Audience: Someone who wants to find out more about Mastodon. They may be a user, admin, developer, or contributor.

Occasion: Landing on the home page via a direct link or search result.

Purpose: Orient the reader with Mastodon’s values, and then direct them to the section that is relevant to them.

What is a microblog?

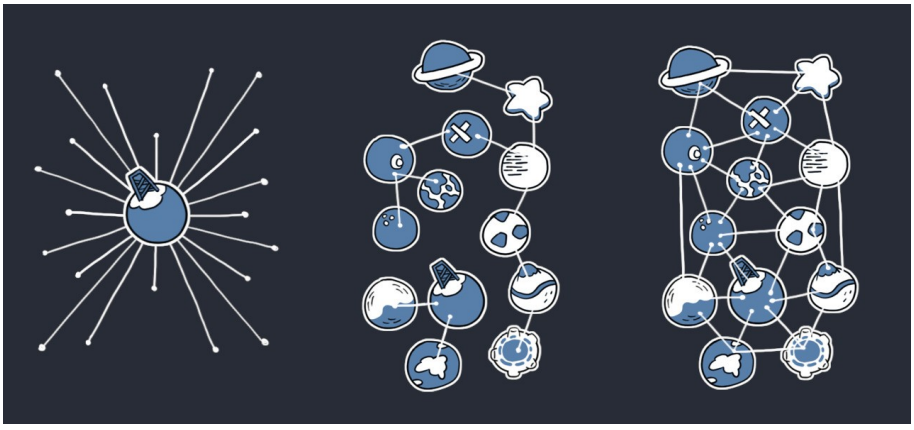
Similar to how blogging is the act of publishing updates to a website, **microblogging** is the act of publishing small updates to a stream of updates on your profile. You can publish text posts and optionally attach media such as pictures, audio, video, or polls. Mastodon lets you follow friends and discover new ones.

What is federation?

Federation is a form of decentralization. Instead of a single central service that all people use, there are multiple services, that any number of people can use.

Grade of centralization	Examples
Centralized	Twitter, Facebook, Instagram
Federated	Email, XMPP, phone networks, physical mail
Distributed	BitTorrent, IPFS, Scuttlebutt

A Mastodon website can operate alone. Just like a traditional website, people sign up on it, post messages, upload pictures and talk to each other. *Unlike* a traditional website, Mastodon websites can interoperate, letting their users communicate with each other; just like you can send an email from your Gmail account to someone from Outlook, Fastmail, Protonmail, or any other email provider, as long as you know their email address, **you can mention or message anyone on any website using their address**.



From left to right: Centralized, Federated, Distributed

What is ActivityPub?

Mastodon uses a standardized, open protocol to implement federation. It is called **ActivityPub**. Any software that likewise implements federation via ActivityPub can seamlessly communicate with Mastodon, just like Mastodon websites communicate with one another.

The **fediverse** ("federated universe") is the name for all websites that can communicate with each other over ActivityPub and the World Wide Web. That includes all Mastodon servers, but also other implementations:

- Pleroma, a modular microblogging engine
- Pixelfed, federated image sharing platform, which lets you share and consume media posts
- Misskey, which includes microblogging alongside a customizable dashboard
- PeerTube, which lets you upload videos to channels
- Plume, which lets you publish longer-form articles
- And many more, including individual and personal websites!

The fediverse does not have its own brand, so you might more often hear "follow me on Mastodon" than "follow me on the fediverse", but technically the latter is more correct.

Practical implications

Choice of server provider and policy

Because Mastodon is just software that can be used to power any website, potential users of Mastodon have the option of choosing a service provider from already-existing Mastodon websites, or to create their own Mastodon website if they wish. The Mastodon project maintains a list of recommended service providers at joinmastodon.org, sortable by category and/or language. Some websites may have moderation policies that go beyond this, such as requiring the use of certain tags on potentially sensitive content, and some websites may have more relaxed moderation policies, but websites listed in the picker all agree to adopt the Mastodon Server Covenant, meaning that they pledge to actively moderate against hate speech, to take daily backups, to have at least one emergency admin, and to provide at least 3 months advance notice in case of shutdown.

> *Maintaining communities that feel safe for all of its members is not easy. Mastodon provides a lot of foundational framework and tools for doing it, and shifts the power to effect change from one commercial entity to the communities themselves.*

– Eugen Rochko, Jul 6 2018, "Cage the Mastodon"

> *A centralized social media platform has a hierarchical structure where rules and their enforcement, as well as the development and direction of the platform, are decided by the CEO […] A decentralized network deliberately relinquishes control of the platform owner, by essentially not having one.*

– Eugen Rochko, Dec 30 2018, "Why does decentralization matter?"

Funding and monetization

Mastodon websites are operated by different people or organizations completely independently. Mastodon does not implement any monetization strategies in the software.

Some server operators choose to offer paid accounts, some server operators are companies who can utilize their existing infrastructure, some server operators rely on crowdfunding from their users via Patreon and similar services, and some server operators are just paying out-of-pocket for a personal server for themselves and maybe some friends. So if you want to support the server hosting your account, check if it offers a way to donate.

Mastodon development is likewise crowdfunded via Patreon and via OpenCollective. **No venture capital is involved.**

> *In my opinion, "instant, public, global messaging and conversation" should, in fact, be global. Distributed between independent organizations and actors who can self-govern. A public utility, without incentives to exploit the conversations for profit.*

– Eugen Rochko, Mar 3 2018, "Twitter is not a public utility"

Interoperability between different software

In practical terms: Imagine if you could follow an Instagram user from your Twitter account and comment on their photos without leaving your account. If Twitter and Instagram were federated services that used the same protocol, that would be possible. With a Mastodon account, **you can communicate with any other compatible website, *even if it is not running on Mastodon***. All that is necessary is that the software support the same subset of the ActivityPub protocol that allows for creating and interacting with status updates. To find out more about the technical specifications required to interoperate with Mastodon, see ActivityPub, WebFinger, and Security. To read more about what ActivityPub allows us to do, see Why ActivityPub is the future.

> *All of these platforms are different and they focus on different needs. And yet, the foundation is all the same: people subscribing to receive posts from other people. And so, they are all compatible.*

– Eugen Rochko, Jun 27 2018, "Why ActivityPub is the future"

Free/libre software

Unlike proprietary services, **anyone has the complete freedom to run, examine, inspect, copy, modify, distribute, and reuse the Mastodon source code, provided they guarantee the same freedoms for any derivative work.** Just like how users of Mastodon can choose their service provider, you as an individual are free to contribute features to Mastodon or publish a modified version of Mastodon that includes different features. These modified versions, also known as software forks, are required to also uphold the same freedoms as the original Mastodon project. For example, glitch-soc is a software distribution that adds various experimental features. Many individual forks exist as well, perhaps themed slightly differently or including small modifications to the codebase. Because

Mastodon is libre software that respects your freedom, personalizations like these are not only allowed but encouraged.

> The ultimate power is in giving people the ability to create their own spaces, their own communities, to modify the software as they see fit, but without sacrificing the ability of people from different communities to interact with each other.

– Eugen Rochko, Feb 20 2017, "The power to build communities: A response to Mark Zuckerberg"

> Decentralization is biodiversity of the digital world, the hallmark of a healthy ecosystem. A decentralized network like the fediverse allows different user interfaces, different software, different forms of government to co-exist and cooperate.

– Eugen Rochko, Dec 30 2018, "Why does decentralization matter?"

Choose your path

Learn how to use Mastodon:

Signing up for an account

Learn how to install Mastodon:

Preparing your machine

Learn how to write an app for Mastodon:

Getting started with the API

Learn about the Mastodon backend and how to contribute:

Technical overview

Sample 2 – Getting started with the Mastodon API

https://docs.joinmastodon.org/client/intro

Audience: Someone who wants to play around with the Mastodon REST API, but doesn't necessarily know much about REST APIs. This is probably the first API they have ever tried to use.

Occasion: A potential client developer has opened the API client developer guide.

Purpose: Explain the fundamentals of HTTP, REST, and JSON.

An introduction to REST

Mastodon provides access to its data over a REST API. REST stands for REpresentational State Transfer, but for our purposes, just think of it as sending and receiving information about various resources based on the request. The Mastodon REST API uses HTTP for its requests and JSON for its payloads.

Understanding HTTP requests and responses

REST API endpoints can be called with certain HTTP methods, and more than one method can be used on the same endpoint. The Mastodon API will generally use the following HTTP methods:

- **GET**: Read or view a resource.
- **POST**: Send information to the server.
- **PUT** | **PATCH**: Update a resource.
- **DELETE**: Removes a resource.

Your favorite programming language probably has a utility or library to make HTTP requests. For the purposes of this section, the cURL utility will be used for examples, which is a command-line utility included with many operating systems by default (as `curl`).

With cURL, the default HTTP method is GET, but you can specify the type of request to make by using the `--request` or `-X` flag; for example, `curl -X POST` will send a POST request instead of a GET request. You may also want to use the `-i` flag to include additional HTTP headers that may be returned as part of the response where relevant.

Providing parameters

HTTP requests can include additional parameters in various different ways, but most notably, the Mastodon API understands query strings, form data, and JSON.

Query strings, form data, and JSON submitted via POST body are equally understood by the API. It is expected that query strings are used for GET requests, and form data or JSON is used for all other requests.

Query strings

Request the URL, but append query strings to the end. Query strings can be appended by first typing ? and then appending them in the form of parameter=value. Multiple query strings can be appended by separating them with &. For example:

```
curl https://mastodon.example/endpoint?q=test&n=0
```

Form data

Instead of mutating the URL with query strings, you can send the data separately. With cURL, this is done by passing it with the `--data` or `-d` flag. Data may be sent together similar to query strings, or it

may be sent separately as key-value pairs with multiple data flags. You may also use the `--form` or `-F` flag for key-value pairs, which also allows sending multipart data such as files. For example:

```
# send raw data as query strings
curl -X POST \
     -d 'q=test&n=0' \
     https://mastodon.example/endpoint
# send raw data separately
curl -X POST \
     -d 'q=test' \
     -d 'n=0' \
     https://mastodon.example/endpoint
# explicit form-encoded; allows for multipart data
curl -X POST \
     -F 'q=test' \
     -F 'n=0' \
     -F 'file=@filename.jpg' \
     https://mastodon.example/endpoint
```

JSON

JavaScript Object Notation as defined in ECMA-404. Quick one-page overview: https://www.json.org/

Similar to sending form data, but with an additional header to specify that the data is in JSON format. To send a JSON request with cURL, specify the JSON content type with a header, then send the JSON data as form data:

```
curl -X POST \
     -H 'Content-Type: application/json' \
     -d '{"parameter": "value"}' \
     https://mastodon.example/endpoint
```

Data types

Multiple values (array)

An array parameter must be encoded using bracket notation. For example, `array[]=foo&array[]=bar` would be translated into the following:

```
array = [
  'foo',
  'bar',
]
```

As JSON, arrays are formatted like so:

```
{
  "array": ["foo", "bar"]
}
```

Nested parameters (Hash)

Some parameters need to be nested. For that, bracket notation must also be used. For example, `source[privacy]=public&source[language]=en` would be translated into:

```
source = {
  privacy: 'public',
  language: 'en',
}
```

As JSON, hashes are formatted like so:

```
{
  "source": {
    "privacy": "public",
    "language": "en"
  }
}
```

True-or-false (Booleans)

A boolean value is considered false for the values `0`, `f`, `F`, `false`, `FALSE`, `off`, `OFF;` considered to not be provided for empty strings; and considered to be true for all other values. When using JSON data, use the literals `true`, `false`, and `null` instead.

Files

File uploads must be encoded using `multipart/form-data`.

This can be combined with arrays as well.

How to use API response data

The Mastodon REST API will return JSON as the response text. It also returns HTTP headers which may be useful in handling the response, as well as an HTTP status code which should let you know how the server handled the request. The following HTTP status codes may be expected:

- 200 = OK. The request was handled successfully.
- 4xx = Client error. Your request was not correct. Most commonly, you may see 401 Unauthorized, 404 Not Found, 410 Gone, or 422 Unprocessed.
- 5xx = Server error. Something went wrong while handling the request. Most commonly, you may see 503 Unavailable.

Sample 3 – WebFinger

https://docs.joinmastodon.org/spec/webfinger/

Audience: Another social web implementer wants to be compatible with Mastodon.

Occasion: The implementer is ready to implement WebFinger, and wants more information about it.

Purpose: Explain why and how WebFinger is used.

Translate `user @ domain` mentions to actor profile URIs.

What is WebFinger, and why is it used?

On Mastodon, user profiles can be hosted either locally on the same website as yours, or remotely on a completely different website. The same username may be used on a different domain. Therefore, a Mastodon user's full mention consists of both the username and the domain, in the form `@username@domain`. In practical terms, `@user@example.com` is not the same as `@user@example.org`. If the domain is not included, Mastodon will try to find a local user named `@username`. However, in order to deliver to someone over ActivityPub, the `@username@domain` mention is not enough – **mentions must be translated to an HTTPS URI first**, so that the remote actor's inbox and outbox can be found.

Enter WebFinger. WebFinger as described in RFC 7033 is a spec that defines **a method for resolving links to a resource**, given only a URI on a particular server. This allows anyone to look up where a resource is located without having to know its exact location beforehand; for example, by email or phone number. This lookup is directed at the endpoint `/.well-known/webfinger`, and a `resource` query parameter is passed along with the lookup. The resource URI used with Mastodon is the `acct:` URI as described in RFC 7565, with the username of a profile that is hosted on a particular domain.

Because Mastodon heavily relies on mentions for addressing other profiles, WebFinger is required for fully interoperating with Mastodon. Users can generally load profiles by searching for the direct HTTPS URI if they know it, or for the `username@domain` address, but Mastodon's internal logic depends almost completely on `acct`: URIs or `username@domain` representations. Searching for any objects or profiles from an ActivityPub implementation without WebFinger will fail because the author cannot be converted to a user in the local database.

Sample WebFinger flow

Suppose we want to lookup the user `@Gargron` hosted on the `mastodon.social` website.

Just make a request to that domain's `/.well-known/webfinger` endpoint, with the `resource` query parameter set to an `acct:` URI.

```
{
  "subject": "acct:Gargron@mastodon.social",
  "aliases": [
    "https://mastodon.social/@Gargron",
    "https://mastodon.social/users/Gargron"
  ],
  "links": [
    {
      "rel": "http://webfinger.net/rel/profile-page",
      "type": "text/html",
      "href": "https://mastodon.social/@Gargron"
    },
    {
      "rel": "self",
```

```
      "type": "application/activity+json",
      "href": "https://mastodon.social/users/Gargron"
    },
    {
      "rel": "http://ostatus.org/schema/1.0/subscribe",
      "template": "https://mastodon.social/authorize_interaction?uri={uri}"
    }
  ]
}
```

https://mastodon.social/.well-known/webfinger?resource=acct:gargron@mastodon.social

You can parse this JSON response to find a link with your desired type. For ActivityPub `id`, we are interested in finding `application/activity+json` specifically.

This way, we have translated `@Gargron@mastodon.social` to `https://mastodon.social/users/Gargron` and we can now interact over ActivityPub by referring to this URI as `id` where appropriate.

```
{
"id": "https://social.example/activities/1",
"type": "Create",
"actor": "https://social.example/actors/1",
"object": {
    "id": "https://social.example/objects/1",
    "type": "Note",
    "content": "Hello, Gargron!"
},
"to": "https://mastodon.social/users/Gargron"
}
```

Sample activity

Note in the above example that `social.example` does not use the same URI structure as Mastodon. Thus, we cannot guess the actor `id` given only the username and domain. However, if `social.example` supports WebFinger, then we can get this `id` by requesting `https://social.example/.well-known/webfinger?resource=acct:username@social.example`and parsing the response for a link with the `application/ld+json; profile="https://www.w3.org/ns/activitystreams"` or `application/activity+json` type. This link should also have the link relation `rel="self"`.

Mastodon’s requirements for WebFinger

When given an account in the form `username@domain` or `@username@domain`, Mastodon will do the following:

- Construct an `acct:` URI using that username and domain
- Make a WebFinger request for that `resource`

Using that WebFinger response, Mastodon will check the following:

- The `subject` is present
- The `links` array contains a link with `rel` of `self` and `type` of either `application/ld+json; profile="https://www.w3.org/ns/activitystreams"` or `application/activity+json`
  - The `href` for this link resolves to an ActivityPub actor

Using that ActivityPub actor representation (which may be provided directly, without the initial WebFinger request), Mastodon will do the following:

- Take `preferredUsername` and the hostname of the actor's server
- Construct an `acct:` URI using that username and domain
- Make a Webfinger request for that `resource`

If the `subject` matches the `resource`, then the process stops here. Otherwise, if the `subject` contains a different canonical account URI, then Mastodon will perform an additional Webfinger request for that canonical account URI in order to ensure that this new `resource` links to the same ActivityPub actor with the same criteria being checked.

In other words, the following cases are valid:

- Asking `example.com` for the resource `acct:alice@example.com` yields a link to an actor on the domain `example.com` with a `preferredUsername` of `alice`, and the `subject` matches the requested resource `acct:alice@example.com`
- Asking `example.com` for the resource `acct:alice@example.com` yields a link to an actor on the domain `ap.example.com` with a `preferredUsername` of `alice`
  - …then, asking `ap.example.com` for the resource `acct:alice@ap.example.com` yields a `subject` of `acct:alice@example.com` and a link to the same actor

See also

app/services/activitypub/fetch_remote_actor_service.rb
app/services/resolve_account_service.rb
app/lib/webfinger.rb

Sample 4 – Why does the Maven repository for Jive Core use Tomcat 7, while the release version of Jive Core instead uses Tomcat 9? (Solution KB Article)

Audience: An enterprise customer has a question while developing a custom plugin for a certain product.

Occasion: The customer is searching through the KB to find a relevant article before filing a new ticket.

Purpose: Answer the customer’s question so that they don’t have to file a new ticket.

Additional stipulation: A time limit of 1 hour was imposed

Overview

Up to Jive Core version 9.2.0, the jive-parent-pom points to Tomcat version 7 in cargo.tomcat.id and cargo.tomcat.home. Meanwhile, the released RPM file points to Tomcat version 9.

Glossary of relevant terms

Apache **Maven** is a tool that allows building and managing Java-based projects.

POM is short for Project Object Model. It is an XML file that defines information that will be used by Maven to build the project.

Codehaus **Cargo** is a thin wrapper for Java application containers. It is used to configure and manage contained environments where a Java application can be deployed.

Cause

The Jive Core project is managed by Cargo, so the Tomcat version is simply the version that will be used in the container. When developing custom plugins for Jive Core, **the exact version of Tomcat doesn't actually matter** -- it's just a development server.

The technical reason that the development environment uses version 7 of Tomcat is that there is a dependency that requires Tomcat 7.x and will not work with later Tomcat versions. This dependency is cargo-maven2-plugin:1.1.4, the Maven 2 plugin for Cargo.

Solution

While this isn't technically a problem, it is possible to upgrade the developer environment to use a Tomcat 9.x container.

The Cargo plugin for Maven 2 added support for Tomcat 9.x in version 1.5.1. Therefore, any version 1.5.1 or later should support Tomcat 9.x. You can view a table of which plugin versions support which Tomcat containers here.

Now that you've changed the dependency's version to something that supports a later version of Tomcat, you can change the version of Tomcat that will be used for the Cargo container.

Step 1: Change the dependency's version

Edit the POM XML file to change the version of cargo-maven2-plugin from 1.1.4 to any version 1.5.1 or later.

Step 2: Change the Cargo container configuration

Edit the POM XML to change cargo.tomcat.id and cargo.tomcat.home to version 9.

Alternative

The upcoming Jive Core 9.3.0 will include an updated reference to a newer version of Tomcat. Again, the version of Tomcat used for development doesn’t matter, but the above steps should not be necessary once Jive Core 9.3.0 is released.

Sample 5 – Troubleshooting issues related to publishing APKs on the Google Play Store (Troubleshooting KB Article)

Audience: A Central Support agent is trying to help a customer solve an issue.

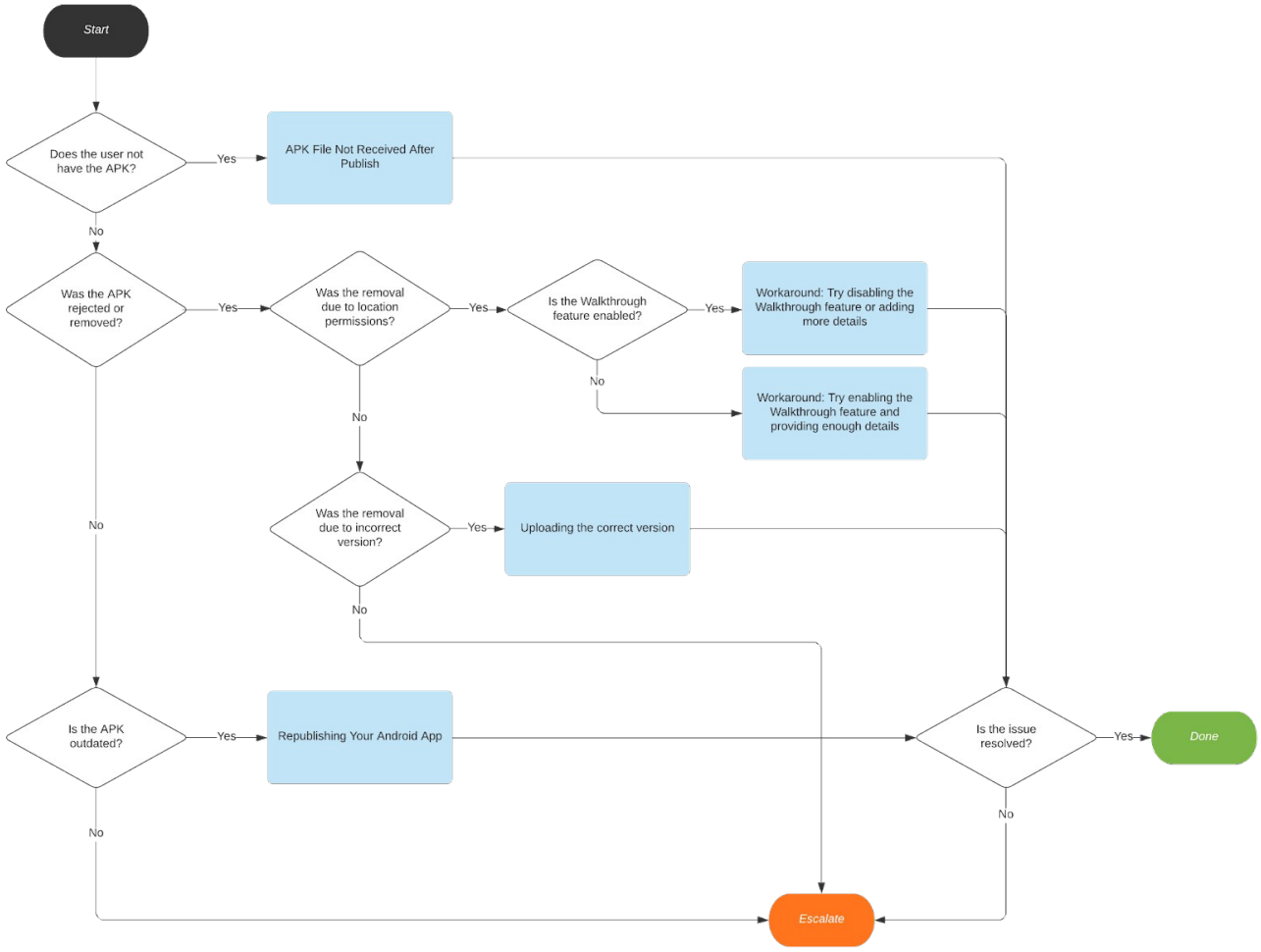
Occasion: The support agent has just received a support ticket.

Purpose: Help the agent disambiguate and identify the customer’s exact issue.

Additional stipulation: A time limit of 1 hour was imposed

Overview

Use the following flowchart to help solve the customer’s problem.



Troubleshooting procedures

Does the user not have the APK?

Occasionally, the user will not have the APK file they need to upload to the Google Play developer console. This may be because they either do not know where to download the APK, or because the APK generation failed.

Relevant links:

- APK File Not Received After Publish
- APK Request Form
- The APK will typically be uploaded to a URL of the form https://www.mobileappco.org/uploads/apk/myappcodename-release.apk which can be accessed directly.

Possible next steps:

- Upload the Android App's APK Onto The Google Play Store

Direct the user to the relevant links so that they may obtain the APK successfully, which should solve their issue. You may then direct the user to possible next steps.

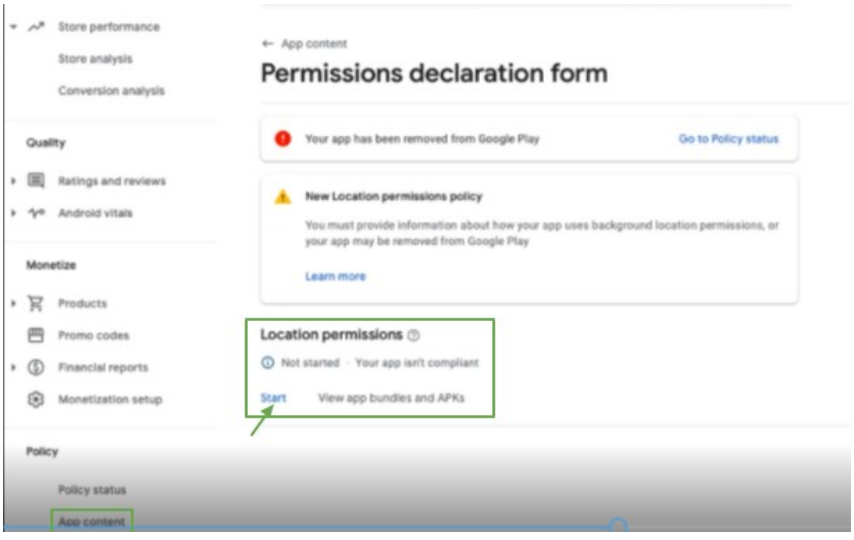
Was the APK rejected or removed from the Play Store?

Location permissions violation

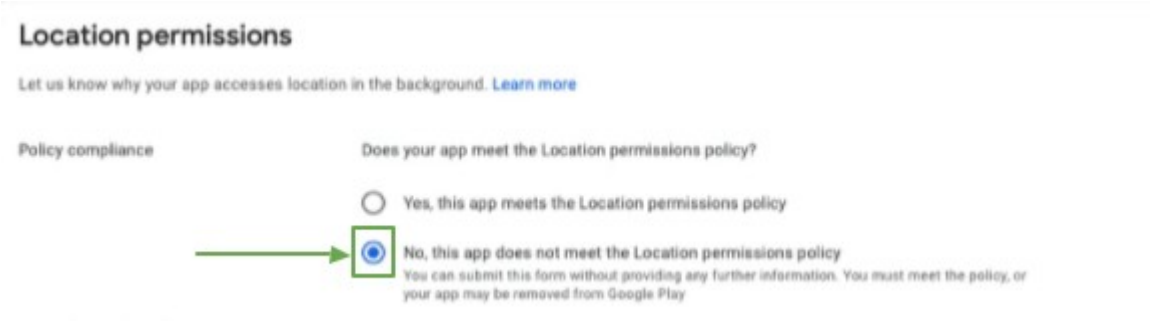
Apps being in violation of the location permissions policy is a common issue.

1. Direct the user to visit the Google Play developer console and navigate to “Policy and Compliance” > “App content”

2. Under “Location permissions”, click “Start”



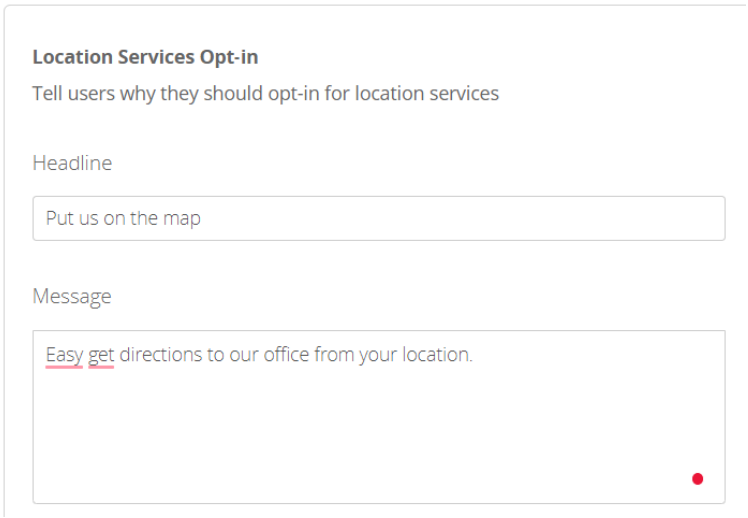
3. Select “No, this app does not meet the location permissions policy”. This will allow you to file an appeal.



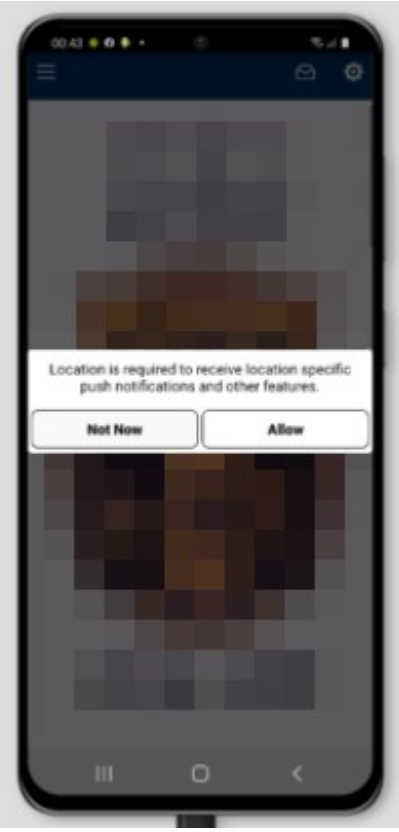
4. Submit an Appeal to the developer support team. You may use the following template:

The <app> utilizes geofencing which allows us to send important push notifications such as once-in-a-lifetime offers, rewards for visiting multiple business locations, and loyalty program announcements, etc. not to mention features such as <indicate them here> which is why we have the location monitoring enabled at all times, even when the app is not in use. Users will also have the option to override this permission by setting the initial launch permissions to 'Only while using the app' (yes, they can miss out on our push notifications but the opt-out option is there). (Optional line: We can also include this clause in our Privacy Policy)

This problem may be avoidable by either enabling or disabling the Walkthrough feature within BiznessApps. The Walkthrough feature will display a prompt for the user to enable location permissions, but the text should be detailed and informative. Here is an example of text that led to a rejection:



There is no guarantee whether this will be enough of a prompt to be accepted by Google, especially if the text is not detailed or informative enough. If this is the case, then the Walkthrough feature can be disabled so that a standard prompt is shown instead:



Relevant links:

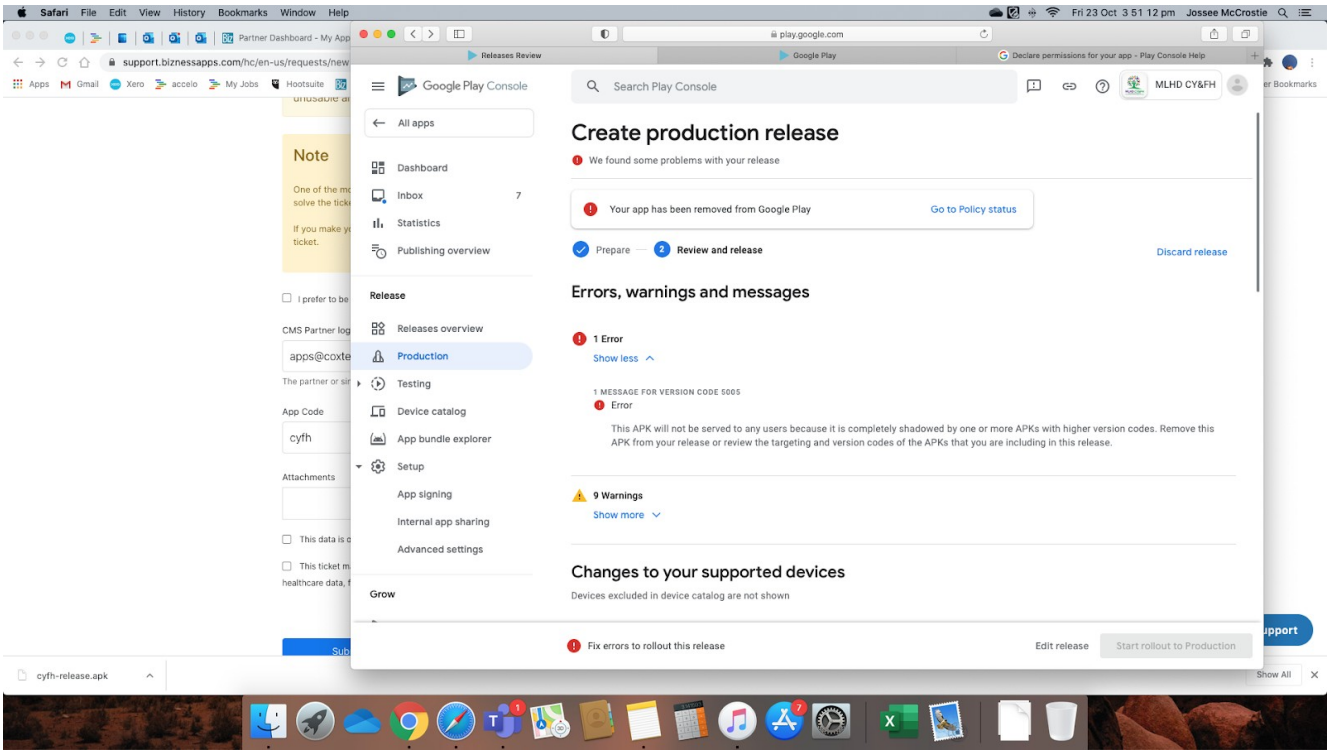
- Submit an Appeal to the Google Play developer support team
- Configuring the Walkthrough feature
- Republishing/Updating Your App

Upload the Android App's APK Onto The Google Play Store -- direct the customer to pay special attention to make sure they upload the correct APK version, in order to avoid the issue described in the Incorrect version uploaded troubleshooting procedure.

Incorrect version uploaded

It is possible that the user has uploaded an older or incorrect version to the Google Play developer console.

For example, the user may have previously uploaded or published version 5006, but upon attempting to create a release via the Google Play console, the user selected the old APK file for version 5005. In this case, Google will throw an error:

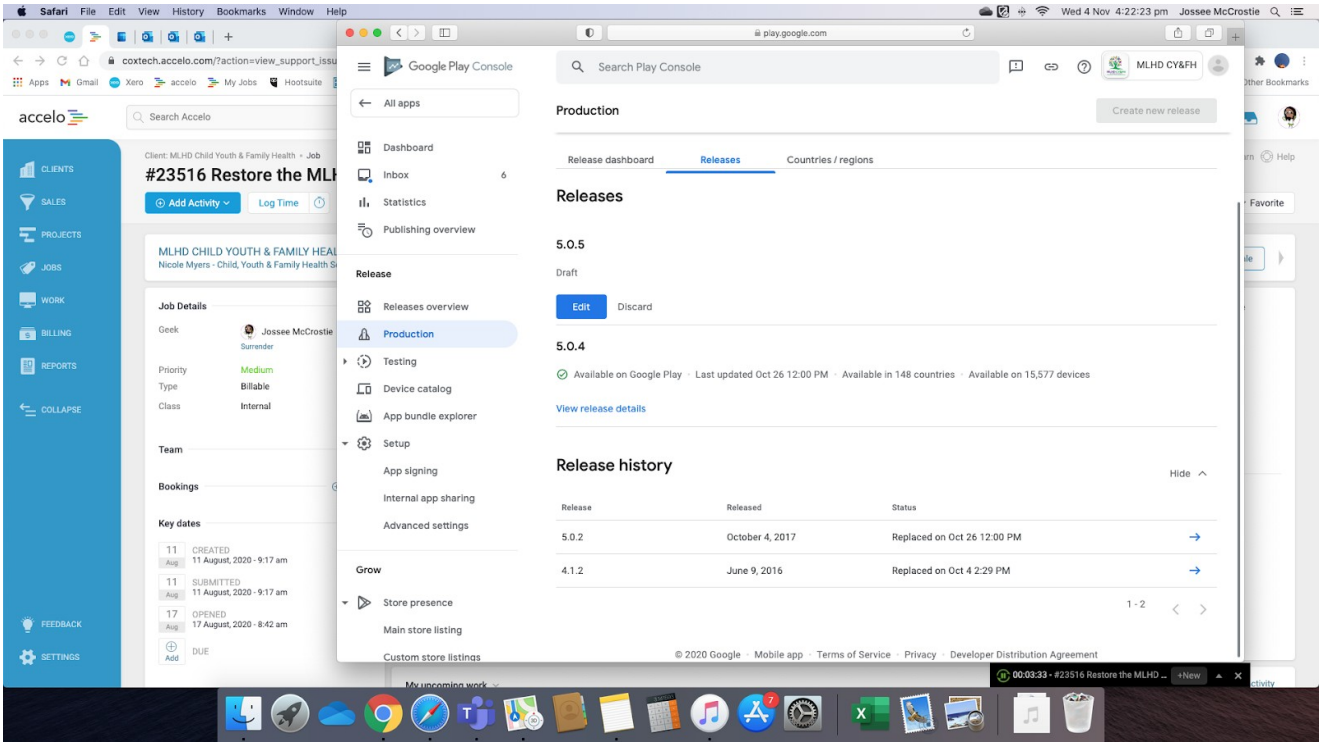


Explain to the user that two things might have happened:

- You uploaded a new APK without removing the previous submission (before it was accepted)
- The APK was submitted with a wrong version code

The user can solve this issue by performing the following steps:

- Kindly check your Releases page (you can access it from the left menu in your Google Play Dashboard) for a double submission or anything that might look strange (e.g. pending reviews)
- Delete the (older) pending submission(s) with version code lower and leave just the last one



At this point, the user may still have an issue with non-compliance with Google's policies. Direct them to "Policy and Compliance" > "App content" so that they can check to see where the app is noncompliant. If the non-compliance is related to location permissions, see the above procedure for Location permissions violation.

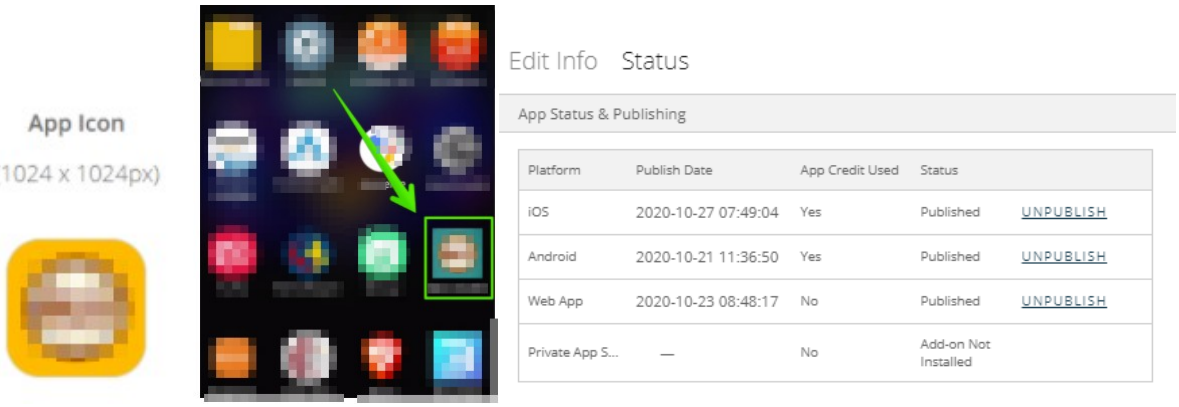
Relevant links:

- Upload the Android App's APK Onto The Google Play Store

Is the APK outdated?

The user may have no issues with getting the app published, but may have issues with the published APK.

For example, the app icon may be outdated in the APK while it is correct in the listing. In this case, check to see if the app has been updated after being published from within BiznessApps. If so, then the app will need to be republished.



Relevant links:

- Republishing Your Android App
- Upload the Android App's APK Onto The Google Play Store -- direct the customer to pay special attention to make sure they upload the correct APK version, in order to avoid the issue described in the Incorrect version uploaded troubleshooting procedure.

Escalation

If the above troubleshooting procedures have all been exhausted and the user still has an issue, then the issue will need to be investigated further or escalated so that it can be solved. This article may be updated as new issues are discovered and solved.

Sample 6 – Introducing ChatGPT: A primer on OpenAI’s new conversational AI product

Audience: Someone familiar with technology, but who has never used ChatGPT before. They’re curious about how to get the best results and what happens behind the scenes.

Occasion: The reader has landed on this blog article via a Web search.

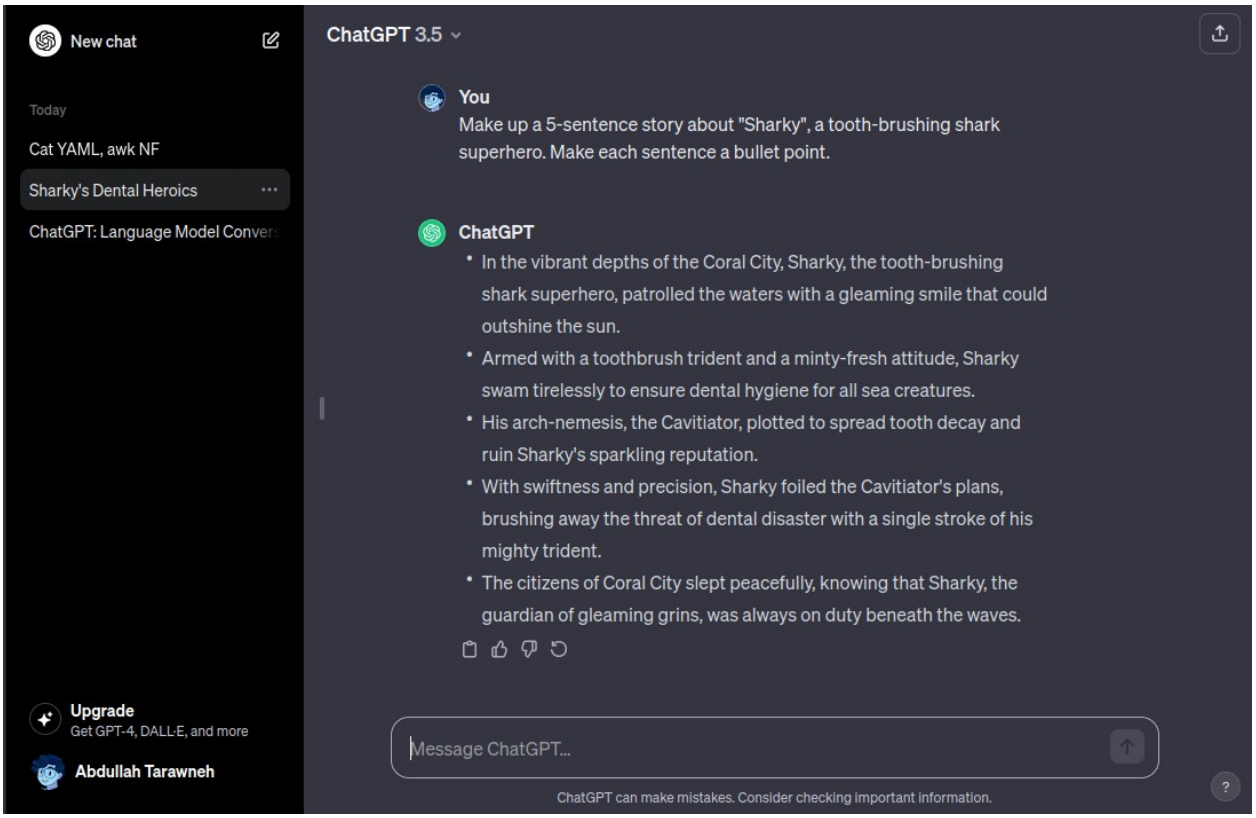
Purpose: Explain how ChatGPT works.

Additional stipulation: Spend no more than 2 hours on this assignment.

With all the buzz around artificial intelligence and machine learning, you may have heard about ChatGPT recently. In this article, you will learn about ChatGPT – what it is, how it works, and how to use it effectively.

What is ChatGPT?

ChatGPT is a chatbot service built by OpenAI that takes a chat message as input and then uses that text to generate and output a convincing human reply based on a body of previously written human material. Most notably, ChatGPT allows the user to continue chatting about a topic while retaining some context of previous messages, so you can guide the conversation in the middle of it. The impressive thing about ChatGPT is the size and diversity of its prior training data, sourced from the internet.



Pictured above: The interface of ChatGPT. In the current conversation, ChatGPT makes up a story about a tooth-brushing shark superhero. As instructed, it did so while making each sentence a bullet point.

How does ChatGPT work?

Behind the scenes, ChatGPT is powered by a **large language model (LLM)**, a machine learning model which uses a lot of data and parameters – far more than usual. Training and operating this LLM also requires a lot of computing resources.

In particular, ChatGPT is built on OpenAI's family of models known as **generative pre-trained transformers (GPT)**. A neural network encodes and decodes information according to an "attention

mechanism” which weighs the relative importance of each part of your input, known as a “token”. Each token is transformed into a vector, which preserves and embeds context in the model and its output. Unlike other models, GPT predicts the next token based on both directions of the current context – left and right. During generation, GPT will generate the output word-by-word, according to the current “context window”. Pre-training for the GPT model used by ChatGPT was performed on a large and diverse source of internet documents, websites, and messages, with a cutoff of January 2022.

How can I use ChatGPT effectively?

To get the best results with ChatGPT, it’s important to keep in mind the limitations of such a product. Although OpenAI built ChatGPT to process and generate natural human language, you can make its job easier by formulating your inputs in a structured format. Try to provide necessary context in your input, and keep the scope of your input small if you can.

If you’re interested in some strategies you can use to get better results, then consider the following:

- Specify some limitations explicitly in your input. Telling ChatGPT that you expect a certain style or format of output helps it formulate its response in that style or format.
- Break the problem down into a chain of thought. ChatGPT can work through step-by-step instructions more efficiently than working through a large problem all at once. This is because of GPT’s context window having a limited size. Newer and more advanced versions of GPT have larger context windows.
- Prepare to validate or correct the output. ChatGPT’s responses are only human-sounding, not human-checked. If ChatGPT says something incorrect, you can guide it into correcting itself mid-conversation.

Conclusion

By now, you should have some understanding of ChatGPT, how it works, and how you can use it effectively. Whether for assistance in writing, brainstorming ideas, or simply enjoying a chat, users can engage in dynamic conversations with ChatGPT. As AI continues to evolve, ChatGPT stands as a powerful tool for natural language interaction.