

Linear Integration Project Structure

Linear OAuth integration with webhooks, GraphQL API client, and issue automation.

#linear #integration #graphql #typescript #oauth

PNG

PDF

Copy

Prompt

Project Directory

linear-integration/

- package.json

Dependencies
- tsconfig.json
- .env.example

API keys and se...
- .gitignore
- README.md
- >

src/

index.ts

App entry point

>

linear/

Linear API clie...

client.ts

LinearClient wr...

queries.ts

GraphQL queries

mutations.ts

GraphQL mutatio...

types.ts

Linear types

>

auth/

OAuth flow

oauth.ts

OAuth handlers

tokens.ts

Token storage

>

webhooks/

Webhook handlers

handler.ts

Webhook router

issue.ts

Issue events

comment.ts

Comment events

verify.ts

Signature verif...

>

actions/

Business logic

sync-issues.ts

create-issue.ts

update-status.ts

>

routes/

HTTP endpoints

index.ts

oauth.ts

OAuth routes

webhooks.ts

Webhook endpoint

api.ts

Your API routes

>

utils/

logger.ts

config.ts

>

tests/

webhook.test.ts

actions.test.ts

Why This Structure?

Linear uses OAuth 2.0 for authentication and GraphQL for its API. This structure separates OAuth flow, webhook handling, and business logic. The official `@linear/sdk` provides typed GraphQL operations. Webhooks let you react to issue and project changes.

Key Directories

linear/ - GraphQL client, queries, and mutations

auth/ - OAuth 2.0 flow and token management

webhooks/ - Event handlers for Linear events

actions/ - Business logic triggered by events

Linear SDK Usage

```
// src/linear/client.ts
import { LinearClient } from "@linear/sdk";

export function createLinearClient(accessToken: string) {
  return new LinearClient({ accessToken });
}

// Usage
const linear = createLinearClient(token);

const issues = await linear.issues({
  filter: { state: { name: { eq: "In Progress" } } },
});

await linear.createIssue({
  teamId: "TEAM_ID",
  title: "New Issue",
  description: "Created via API",
});
```

Getting Started

- Create OAuth app at [linear.app/settings/api](#)
- Set redirect URI and copy Client ID/Secret
- Install SDK: `npm install @linear/sdk`
- Implement OAuth flow in `auth/oauth.ts`
- Set up webhook endpoint and verify signatures

When To Use This

- Syncing issues with external systems
- Automating issue workflows
- Building dashboards with Linear data
- Slack/Discord notifications for Linear events
- Custom integrations for your team

Webhook Events

Issue - create, update, remove events

Comment - New comments on issues

Project - Project updates and changes

Cycle - Sprint/cycle events

Best Practices

- Verify webhook signatures with `linear.webhookSignature`
- Use GraphQL fragments for reusable query parts
- Implement token refresh for long-running integrations
- Rate limit your API calls (1,500 req/hour)
- Use pagination for large issue lists

Trade-offs

GraphQL complexity - Steeper learning curve than REST

Rate limits - 1,500 requests per hour per user

OAuth required - Personal API keys limited to read-only