Engineer

API

🔀 den beskrevne måten ett program kaller et annet på

Hva det er

API står for application programming interface, på norsk gjerne programmeringsgrensesnitt, og er det publiserte settet med operasjoner et program tilbyr andre programmer: hva hver operasjon heter, hva den tar imot, hva den returnerer, og hvordan den melder fra om problemer. Flertall er API-er, og bestemt form API-et. Den som kaller, lærer seg denne overflaten og ingenting bak den. Programmet på den andre siden kan derfor skrives om, flyttes til en annen maskin eller deles i tre, og den som kaller, fortsetter å virke så lenge overflaten holder.

Ordet dekker flere lag. API-et til et bibliotek er funksjonene og typene det eksporterer, og de kalles inne i én prosess. API-et til et operativsystem er settet med systemkall, og POSIX standardiserer et felles grunnsett av dem, så et program skrevet mot `open` og `read` kan bygges på mer enn én Unix-lignende systemkjerne. Et web-API tilbys over et nettverk, nesten alltid over HTTP, og det er det de fleste mener med ordet alene i dag: en server som svarer på forespørsler på adresser som `/v1/orders`, som regel med JSON i kroppen.

Web-API-er finnes i en håndfull stiler. De ressursorienterte, ofte kalt REST-stil, gir hver ting en adresse og bruker metodene i HTTP til å lese og endre den. RPC-stilen, f.eks. gRPC, publiserer navngitte prosedyrer med argumenter og resultater deklarert på forhånd. GraphQL tilbyr ett endepunkt og et skjema, og klienten skriver en spørring som navngir nøyaktig de feltene den vil ha tilbake. Webhooks snur retningen: leverandøren kaller en adresse kunden har registrert, når noe skjer.

Tre nabobegreper blandes ofte sammen med det. En protokoll, som HTTP, definerer hvordan meldinger reiser; et API er det én bestemt tjeneste sier i den protokollen. Et SDK er et bibliotek i ett programmeringsspråk som pakker inn

noens API, så en utvikler kaller en funksjon i stedet for å sette sammen forespørsler selv. Og implementasjonen er koden bak API-et, som den som kaller, aldri trenger å lese.

Bruk begrepet i en beslutning

Bruk «Hva betyr API?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Delen brukerne aldri ser: hvor dataene ligger, hvilke regler som gjelder, og hvordan andre systemer får lov til å snakke med den.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Designet og driftet 6 ETL/ELT-pipelines — 6 ETL-pipelines på tvers av tre databasemotorer

Designet og driftet 6 ETL/ELT-pipelines Ta kontakt →

Hvorfor det er viktig

Et API er den delen av et system andre bygger på, og det gjør det til et løfte som varer lenger enn koden bak det. Å skrive om en handler tar en ettermiddag. Å endre hva den samme handleren returnerer, kan ødelegge en app som allerede er installert på kundenes telefoner, en partners nattlige import eller et skript ingen husker å ha skrevet.

Den første beslutningen handler derfor om publikum. Et privat API, som bare teamets egen frontend bruker, kan endres i samme pull request som den ene klienten sin. Et offentlig API endres gjennom avvikling (deprecation) og varsel, og hvert felt det sender ut, er et felt som må støttes så lenge noen leser det.

Den andre beslutningen er hvor streng kontrakten skal være. Et strengt API deklarerer en type og en grense for hver inndataverdi, avviser det det ikke kjenner igjen, og svarer på hver feil med en presis statuskode. Det fanger feil i døra og er lettere å dokumentere, teste og generere klienter for. Et tolerant API godtar alt som kommer, og gjør så godt det kan. Det føles vennlig den første måneden, og gjør deretter hvert tilfeldig trekk ved implementasjonen til atferd noen er avhengige av. Strenghet koster designtid i starten; toleranse koster mer senere, på et tidspunkt ingen velger selv.

Den tredje beslutningen gjelder andres API-er. En betalingsleverandør, en karttjeneste eller en språkmodell er et API produktet kaller, men ikke kontrollerer: prisene flytter seg, versjoner pensjoneres, kvotene strammes inn.

Kaller du en leverandør direkte fra femti steder, blir et bytte en omskriving. Kaller du den fra én modul teamet eier, blir byttet en endring i den modulen. Innpakningen koster et ekstra lag og kjøper deg en vei ut.

Semantic Versioning gir dette løftet et vokabular: hovedversjonsnummeret går opp når det offentlige API-et endres på en måte som ikke er kompatibel, så en klient kan se av versjonen alene om en oppgradering er trygg. Over HTTP finnes det flere steder å signalisere et brudd, og RFC 9205, IETFs veiledning for protokoller bygd på HTTP, nevner bl.a. en ny lenkerelasjon, en ny medietype eller et nytt headerfelt.

Slik ser det ut i praksis

API-ene bak flere prosjekter på dette nettstedet ble formet av ett spørsmål: hva hører hjemme i API-et, og hva hører hjemme et annet sted?

På NextMariner, en maritim plattform for profesjonell nettverksbygging, selskapsomtaler og sponset opplæring, omfattet grunnleggerens arkitekturbidrag å holde Go-API-et bevisst tynt. Det er bygd med Huma over Fiber, ruter, validerer forespørselen, anvender sikkerhetsfiltreringen og orkestrerer, og inneholder ingen forretningslogikk i det hele tatt. Reglene ligger i PostgreSQL-funksjoner, slik at en regelendring skjer i ett lag, i SQL. Grensene ble skrevet ned og håndhevet i gjennomganger. Det gjør API-et til en kjedelig adapter, i god forstand.

Et beslektet prosjekt flyttet også formingen ut av API-et. PostgreSQL-funksjoner setter sammen hvert svar som JSON, og Go-handleren tar det imot som `json.RawMessage` og videresender det urørt, gjennom en liten `QueryJSON` -hjelper. DTO-ene, mappingkoden og respons-structene et lagdelt API vanligvis samler opp, falt bort, handlerne ble dramatisk kortere, og frontenden var ikke lenger avhengig av Go. Byttet fortjener å bli nevnt: formen på API-ets svar er nå definert i SQL, og det er der den som gjennomgår koden, leser kontrakten.

Et API alle kan nå, må også forsvare seg selv. En offentlig katalog over bedrifter og fagfolk ligger bak 22 hastighetsbegrensere, hver bygd for ruten den beskytter, med tilstanden i Redis så en grense holder på tvers av instanser. Når Redis ikke kan nås, feiler begrenserne lukket (fail closed) og avviser trafikken. Cloudflare Turnstile, en opprinnelseslås, en skreddersydd utfordring på katalogen og 351 linjer idempotensmellomvare, som hindrer at en skriving som

prøves på nytt, lander to ganger, fullfører forsvaret. Siden sier kostnaden rett ut: et Redis-problem blir et problem brukerne ser, og det ble foretrukket framfor at et Redis-problem blir et faktureringsproblem.

Hvor det går galt

Feil i API-er kommer gjerne sent, når klientene og dataene har vokst.

Et omdøpt felt ødelegger for en klient ingen visste om. Så snart et API er publisert, er noen avhengig av hver detalj som kan observeres: feltnavn, rekkefølgen i en liste, ordlyden i en feilmelding. Døp om `customer_name` til `name`, og integrasjonen til en partner brekker på neste forespørsel. Endre heller ved å legge til: legg det nye feltet ved siden av det gamle, merk det gamle for utfasing, følg med i loggene til ingen leser det, og fjern det først i en ny hovedversjon.

Et listeendepunkt blir tregere for hver måned. Det returnerer hver rad som matcher, noe som gikk fint med femti poster ved lansering og gir tidsavbrudd ved to hundre tusen. Bruk paginering fra første versjon, med en standard sidestørrelse og et fast maksimum klienten ikke kan overskride. For store eller travle samlinger bør du velge en markør (cursor) framfor en forskyvning (offset): med en forskyvning skyver en rad som settes inn mellom to forespørsler, en annen rad over på neste side, der klienten leser den to ganger.

```
GET /v1/orders?limit=50&after=ord_8f2c HTTP/1.1
Host: api.example.com
```

En kunde leser en annen kundes ordre. API-et sjekker at den som kaller, er logget inn, og laster deretter `/v1/orders/42` ut fra identifikatoren alene, så den som endrer tallet i adressen, kan bla seg gjennom andres data. Autentisering svarte på hvem som kaller, men det var aldri noen som spurte om den som kaller, eier ordre 42. Avgrens hvert oppslag etter eier inne i selve spørringen, og ha en test som logger inn som én konto og ber om en annen kontos poster.

Svaret bærer kolonner ingen mente å publisere. Et endepunkt serialiserer en databaserad eller et ORM-objekt som det står, så hashen av et passord, et internt flagg eller en leverandørs innkjøpspris går ut til hver klient, og en kolonne som legges til neste år, publiseres samme dag den opprettes. Den samme feilen på

inndatasiden lar en forespørsel sette `role` eller `is_admin` ved å ta feltet med i kroppen. Gi hvert endepunkt eksplisitte typer for inndata og utdata som lister opp feltene som er tillatt, så det å publisere et felt blir en beslutning noen tar.

Les mer

- RFC 9110: HTTP Semantics — metodene, statuskodene og headerne hvert web-API er bygd av, definert av IETF.
- RFC 9205: Building Protocols with HTTP — gjeldende beste praksis for å utforme en protokoll oppå HTTP, også for hvordan den videreutvikles.
- RFC 9457: Problem Details for HTTP APIs — en standard JSON-form for feilsvar, så klienter tolker hver feil på samme måte.
- RFC 6585: Additional HTTP Status Codes — der 429 Too Many Requests og hintet `Retry-After` for klienter som begrenses, er definert.
- Semantic Versioning 2.0.0 — reglene for å signalisere en inkompatibel API-endring gjennom et versjonsnummer.

Ordliste

Velg ut fra deres faktiske rammer

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva betyr API?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

welcome@engineer.company +45 50 10 61 84 København, Danmark

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/api/