Das Cashu Protokoll

Going Nuts

(soft) requirements

- part 1: grundlegendes verständnis von Bitcoin / Lightning
- part 2: JavaScript basics

über mich

- class of 21
- cashu contributor
- npub.cash
- cashu-ts co-lead / maintainer

get in touch

purple bird (nostr): egge@npub.cash

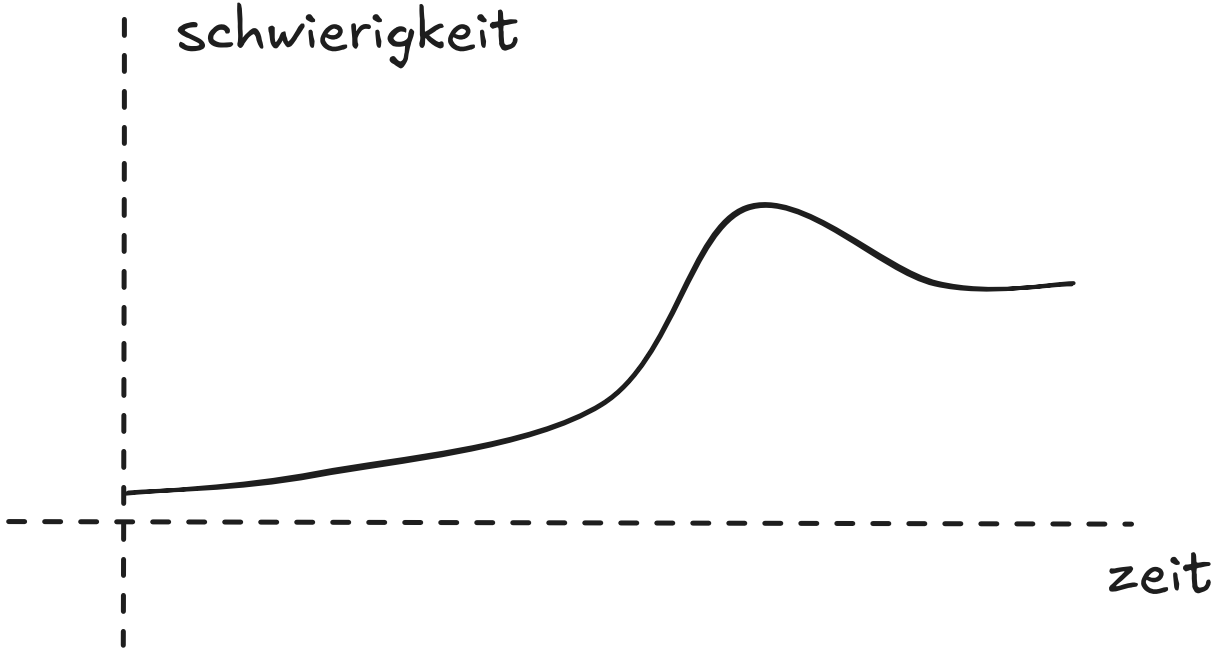
blue bird (twitter): Egge21M

über diesen workshop

ziele:

- lernen was ecash ist und wofür es nützlich ist
- verstehen wie das cashu protokoll funktioniert
- optional: gemeinsam implementierungen beginnen

über diesen workshop



agenda

1. intro
2. chaumian ecash
3. cashu und seine anwendungsfälle
4. blinde signaturen
5. die cashu API
4. Q&A 1
5. optional: live coding mit cashu-ts

chaumian ecash

David Chaum

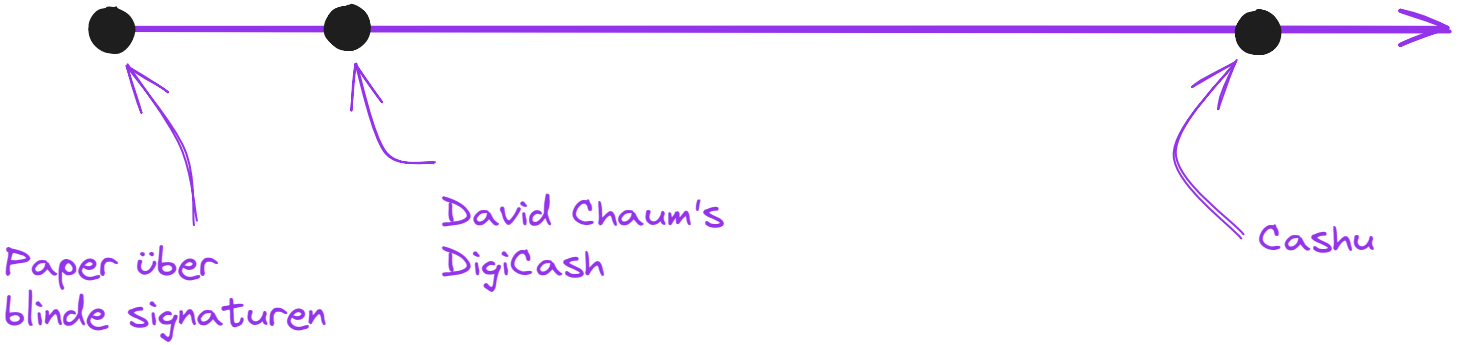
cypherpunk und
crypto-urgestein

chaumian ecash

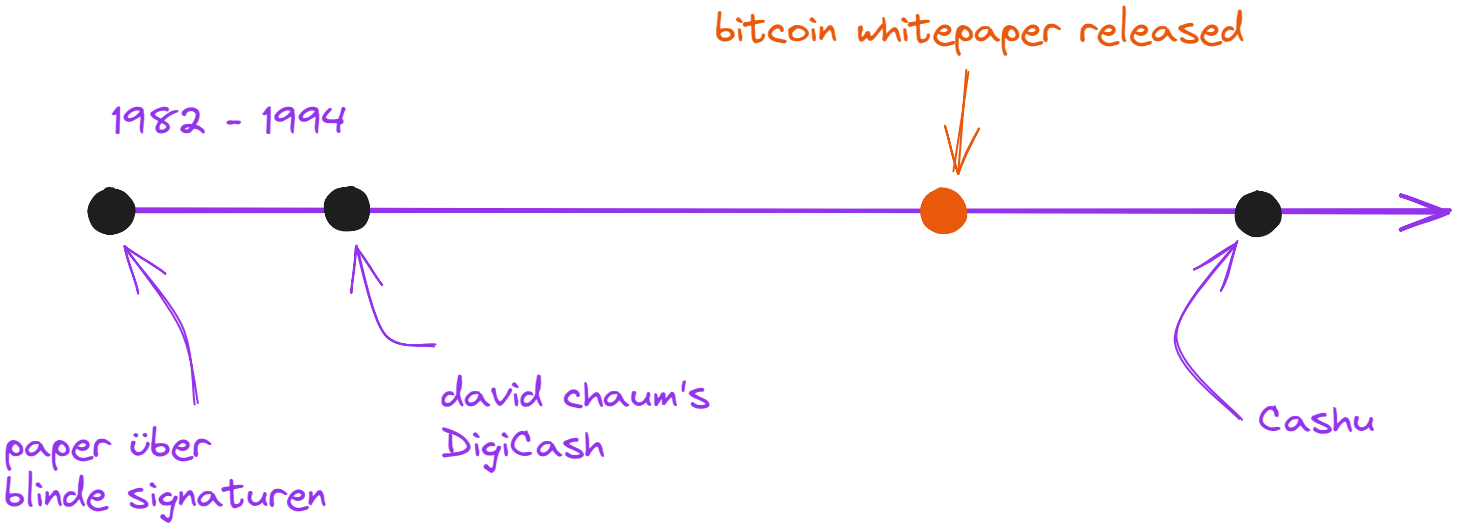
an anonymous cryptographic electronic money or electronic cash system

- digitales bargeld
- durch digitale signaturen gesichert
- offline verwahrbar
- erlaubt anonyme zahlungen im Internet ohne accounts

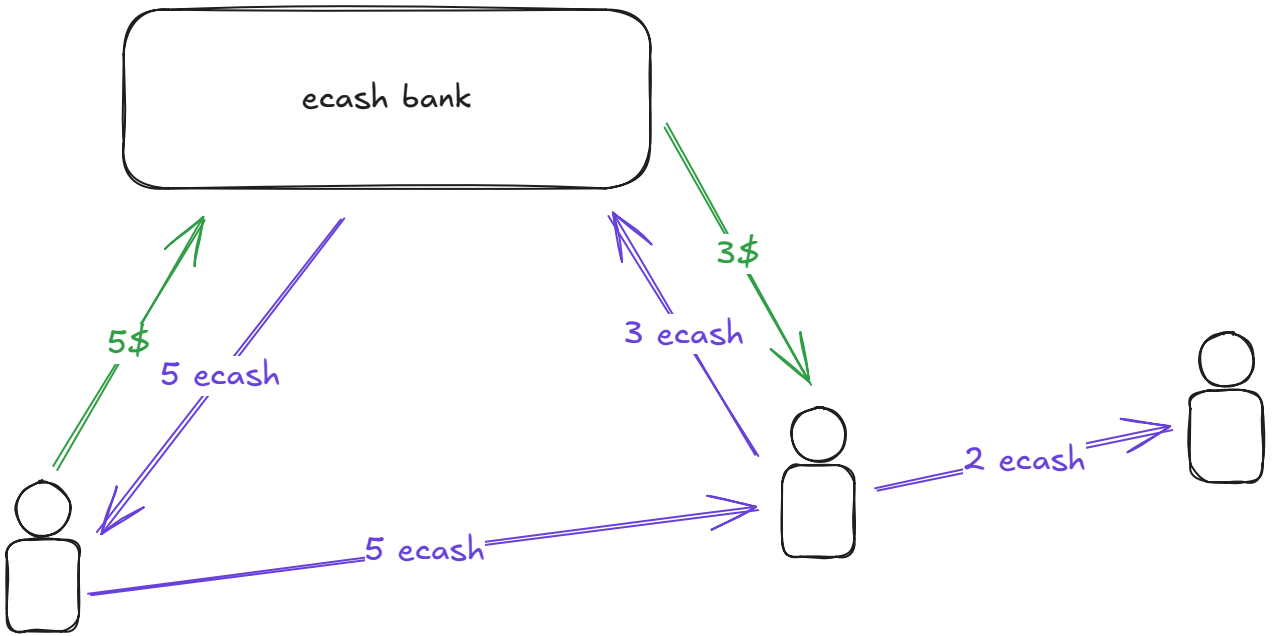
ecash zeitstrahl



ecash zeitstrahl



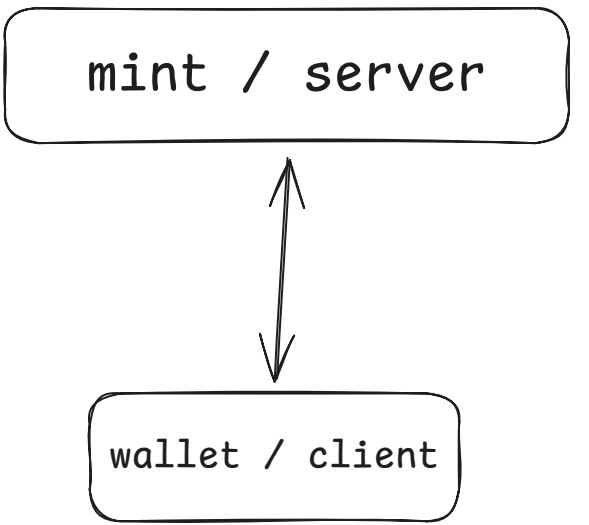
chaumian ecash



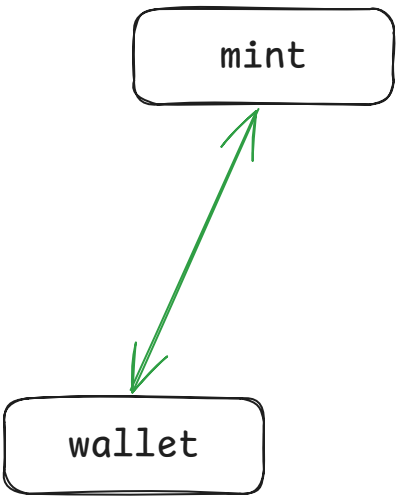
cashu in a nutshell

cashu in a nutshell

ist ein crypto und kommunikations
protokoll zwischen servern und clients



cashu in a nutshell

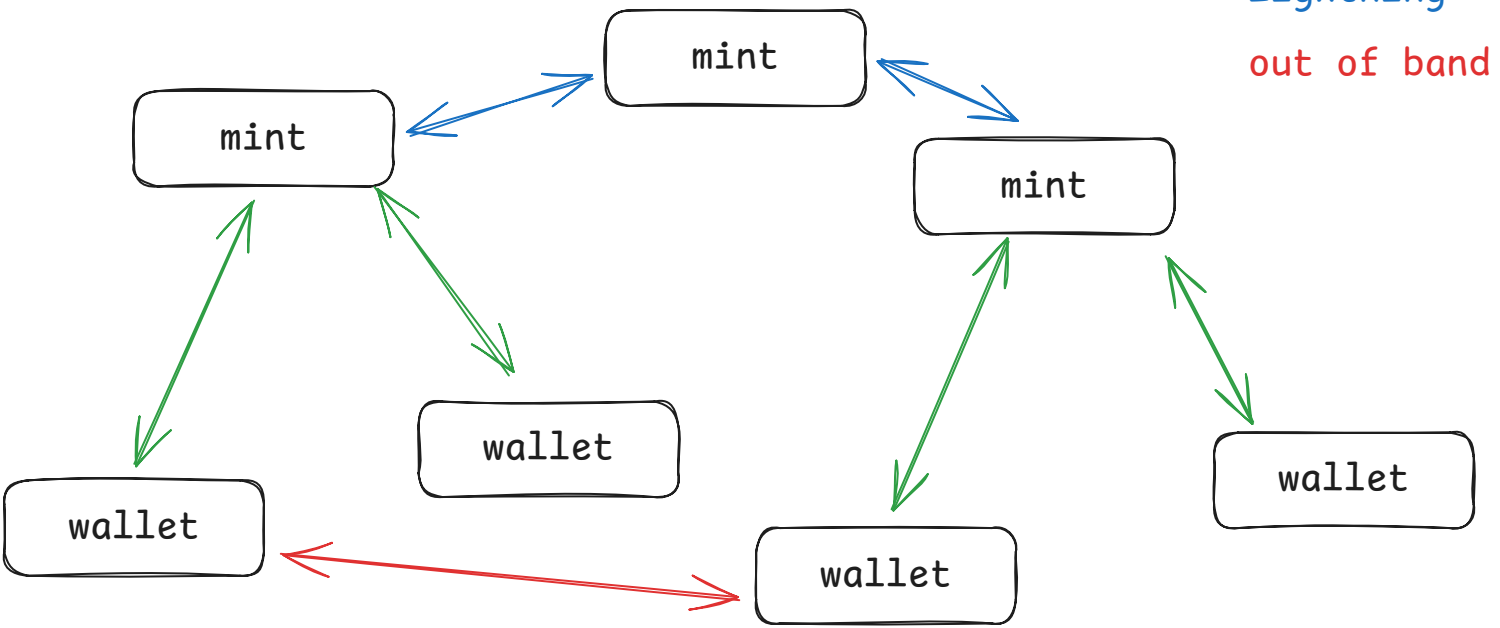


cashu in a nutshell

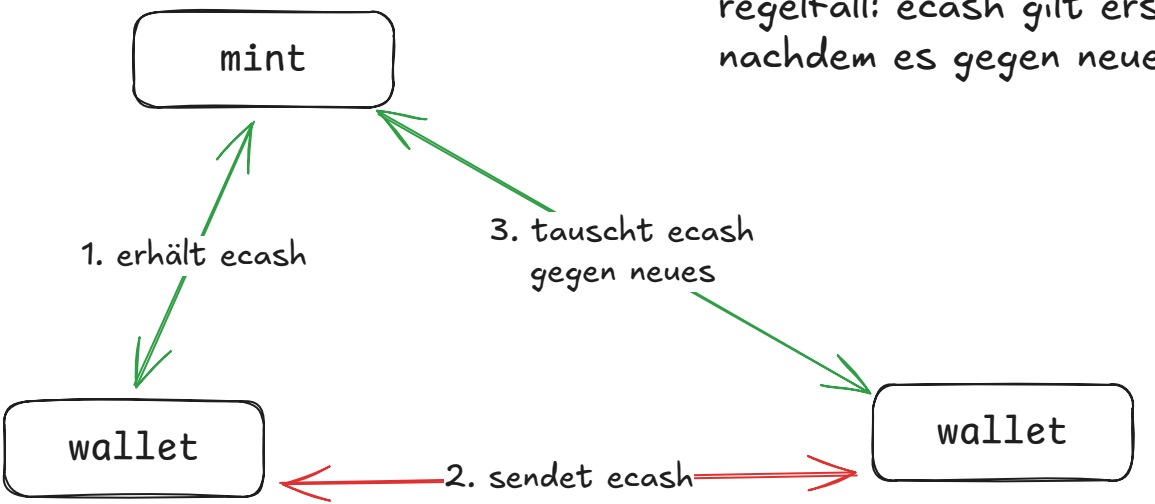
Cashu

Lightning

out of band



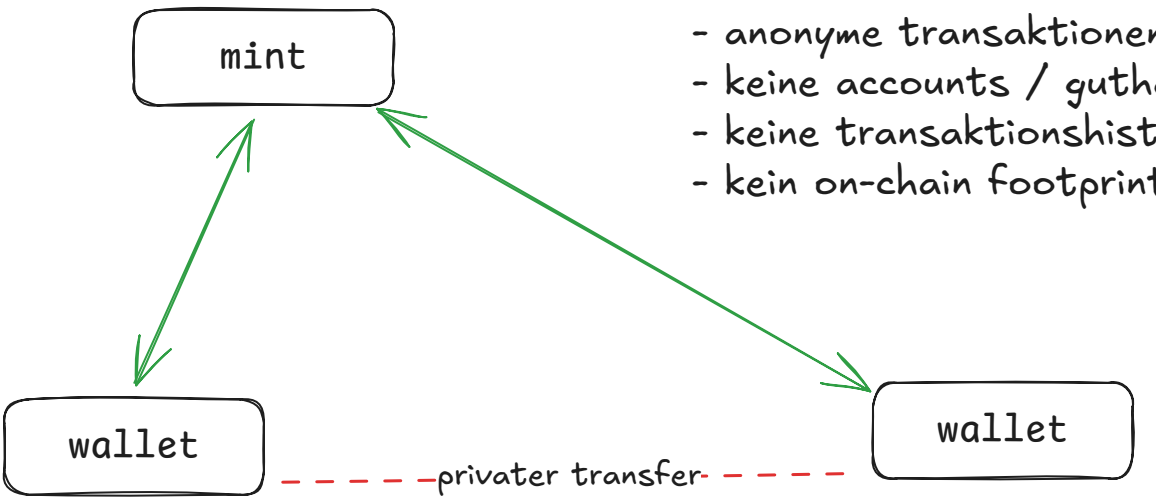
cashu in a nutshell: doublespending



regelfall: ecash gilt erst als "erhalten"
nachdem es gegen neues getauscht wurde

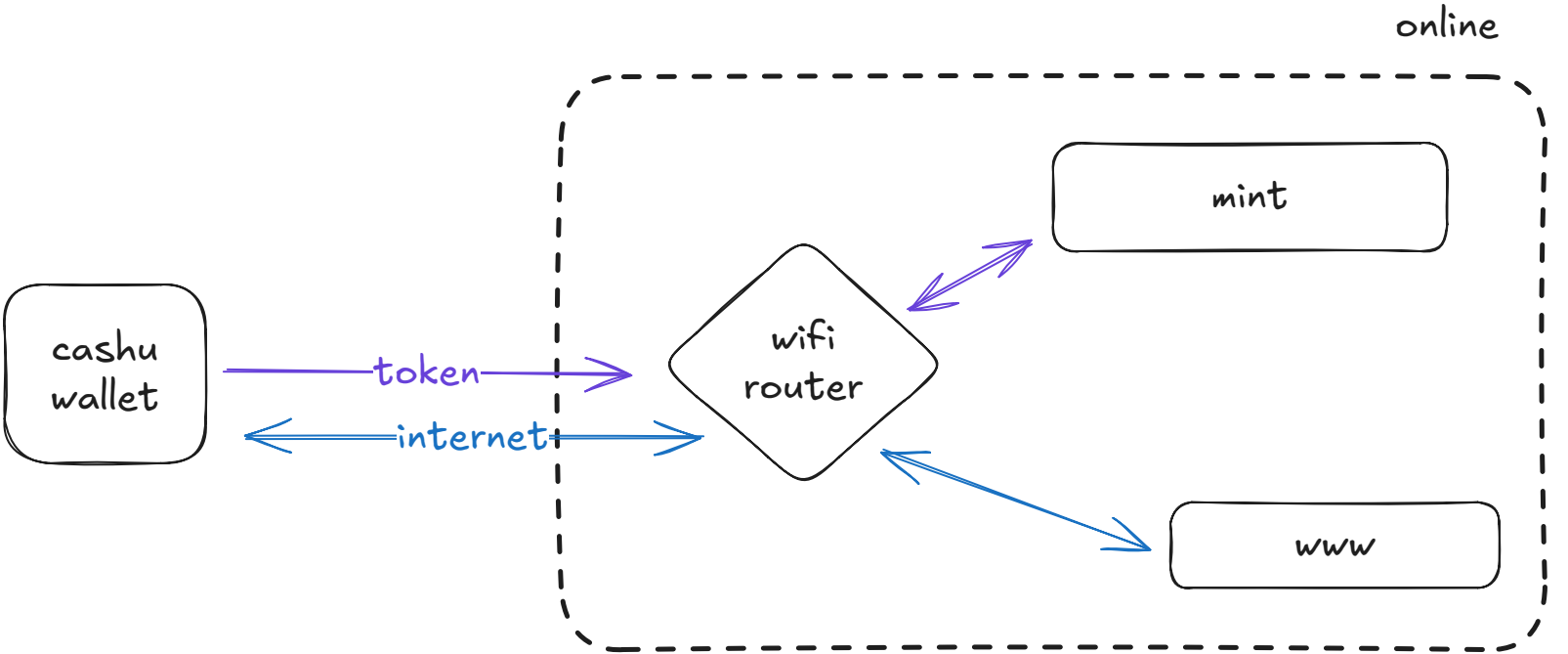
warum cashu?

warum cashu? anonymität!

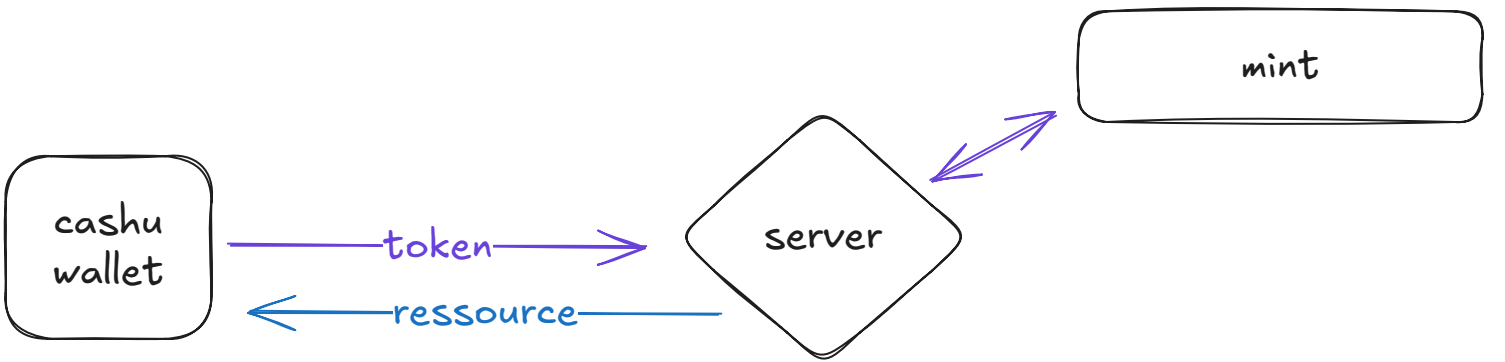


- anonyme transaktionen zwischen nutzern
- keine accounts / guthaben
- keine transaktionshistorie
- kein on-chain footprint

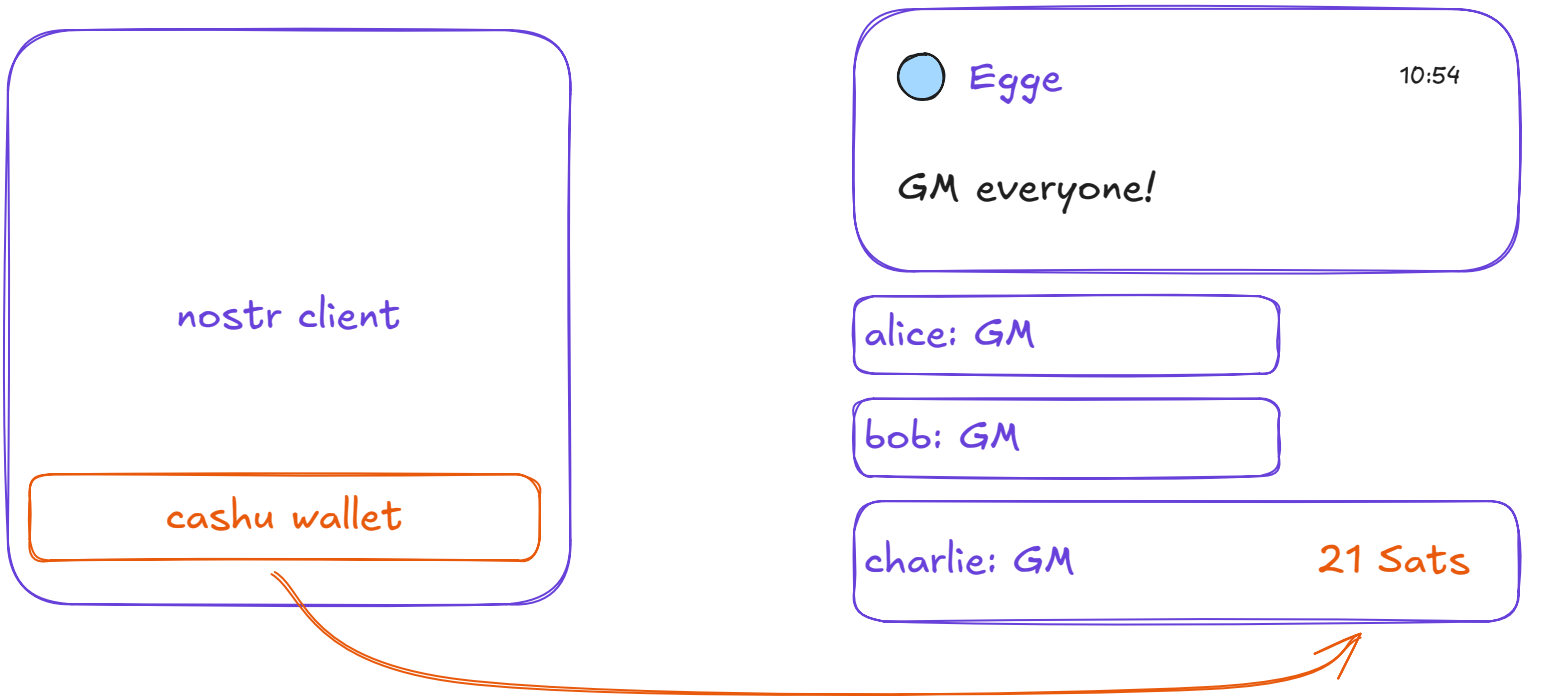
warum cashu? neue usecases! tollgate



warum cashu? neue usecases! Cashu 402

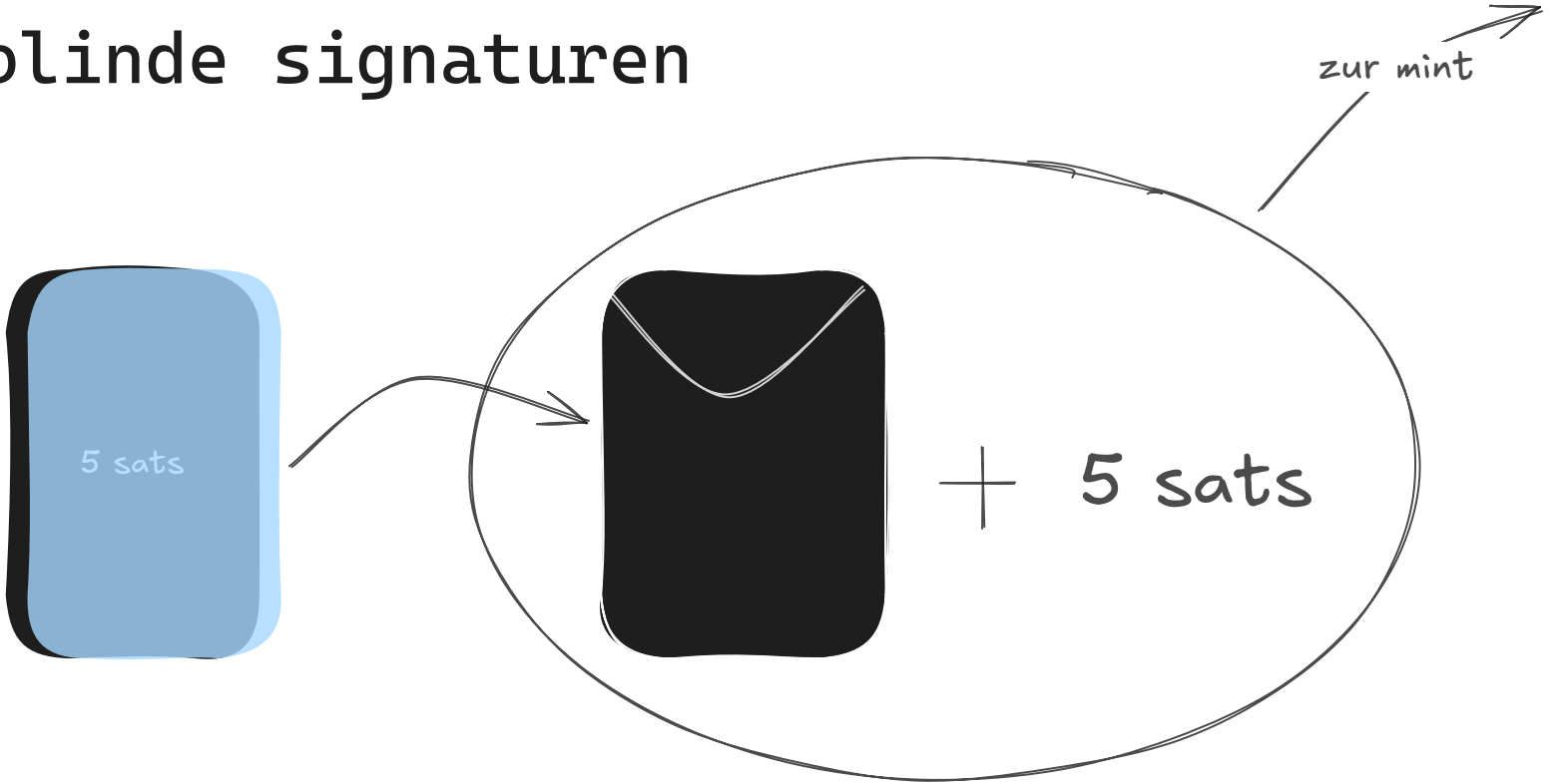


warum cashu? neue usecases! nutzaps

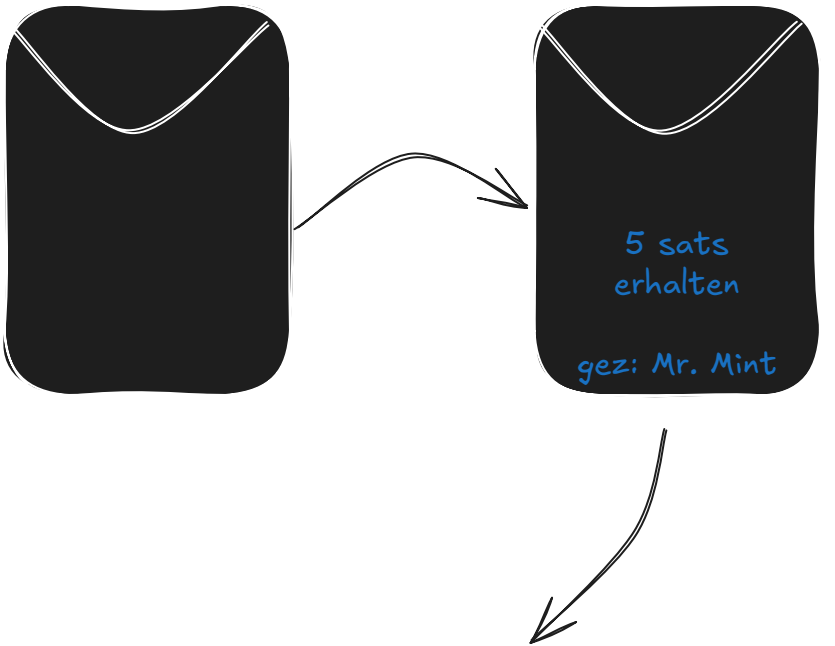


blinde signaturen

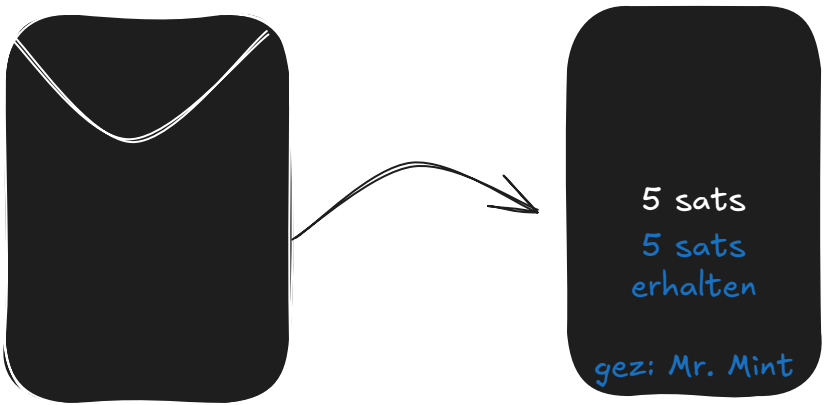
blinde signaturen



blinde signaturen



blinde signaturen



blinde signaturen

bob wählt schlüssel k und veröffentlicht kG = K

alice wählt geheimnis x und berechnet Y = hash_to_curve(x)

alice wählt Blinding Factor r, berechnet B_ = Y + rG (blind message) ⟶ bob

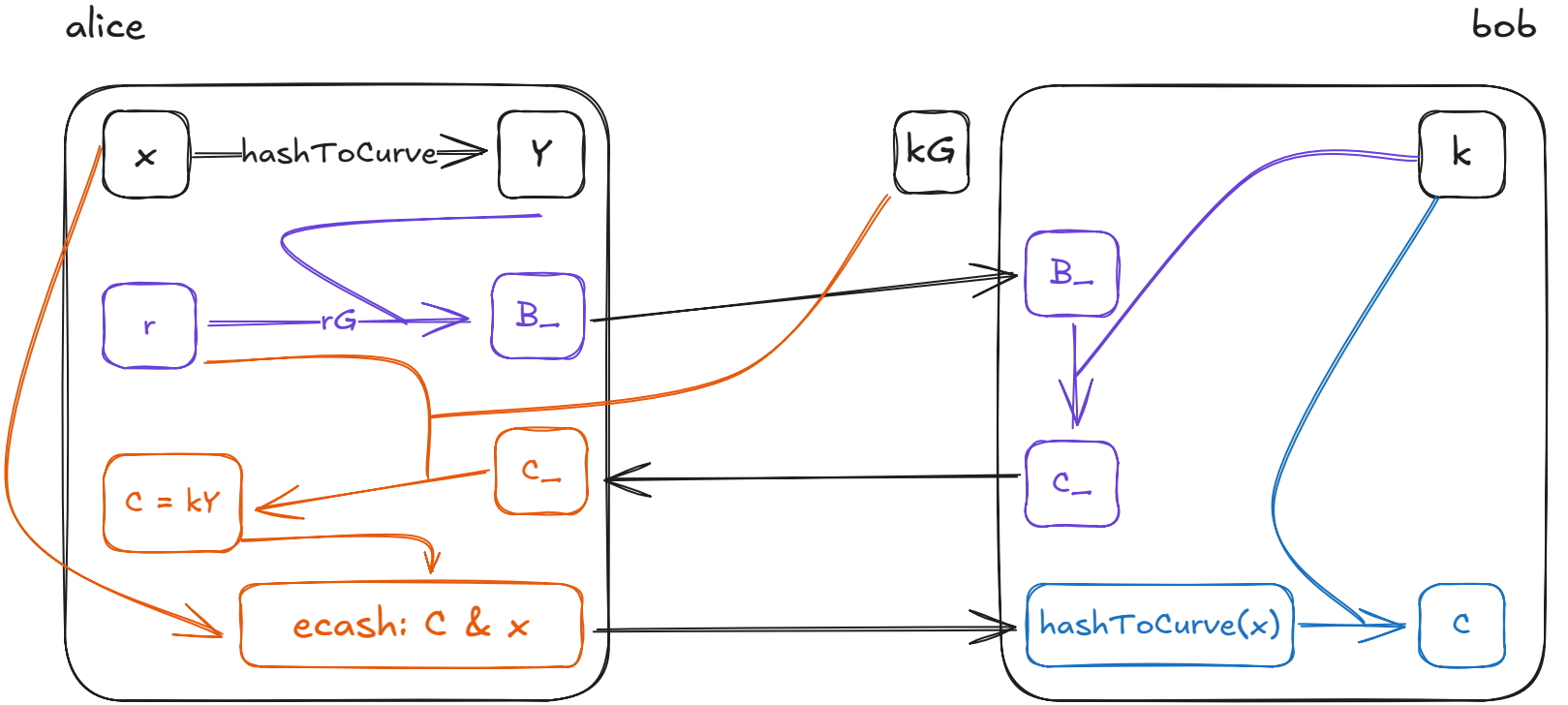
bob erstellt blind signatur C_ = kB_ = kY + krG ⟶ alice

alice berechnet die unblided signature C = C_ - rK = kY + krG - krG = kY

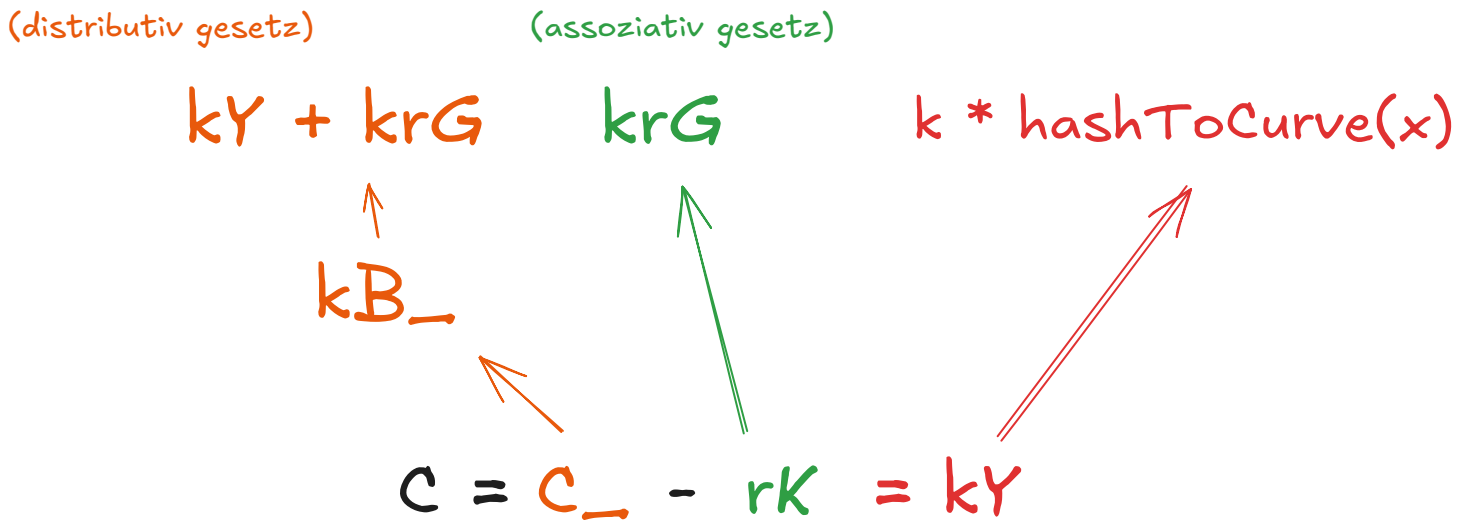
⟶ ecash = C & x

bob kann ecash verifizieren: k*hash_to_curve(x) == C

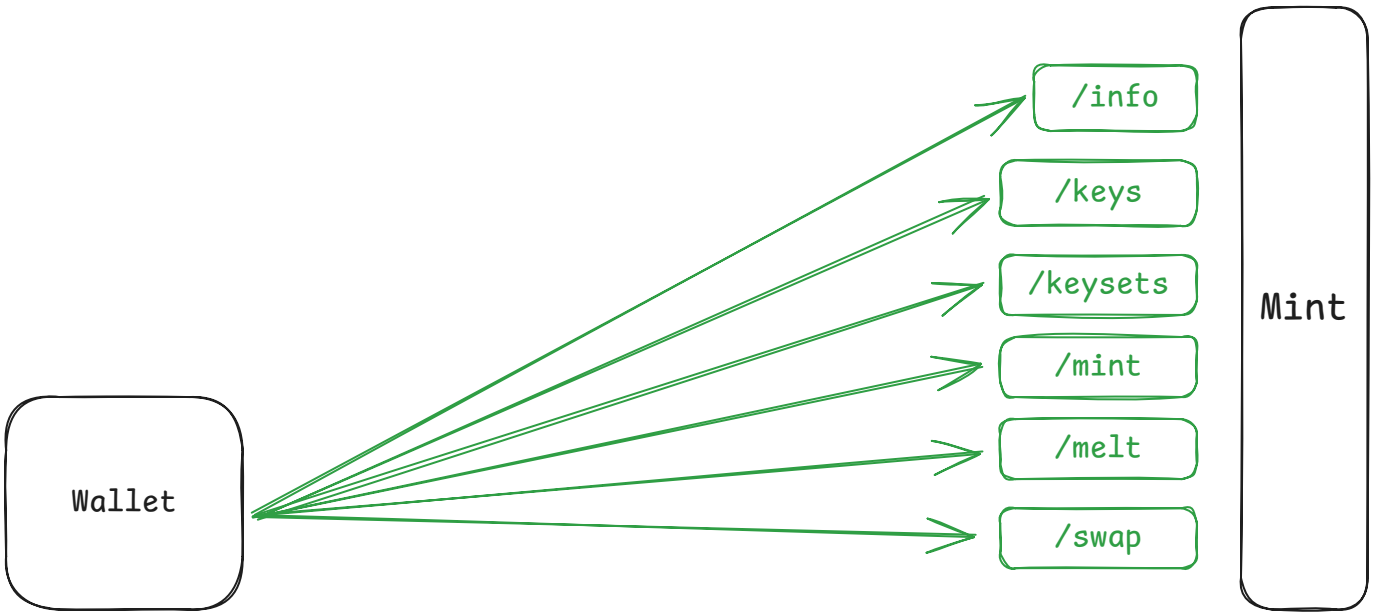
blinde signaturen



blinde signaturen



cashu in a nutshell



cashu in a nutshell: /info


```
{
"name": "Bob's Cashu mint",
"pubkey":
"0283bf290884eed3a7ca2663fc0260de2e20
64d6b355ea13f98dec004b7a7ead99",
"version": "Nutshell/0.15.0",
"description": "The short mint description",
"contact": [...],
"icon_url": "https://mint.host/icon.jpg",
"time": 1725304480,
"nuts":...
}
```

```
"nuts": {
"4": {
"methods": [
{
"method": "bolt11",
...
}
],
"disabled": false
},
"5": {
"methods": [...],
"disabled": true
},
"7": {
"supported": true
}
}
```

cashu in a nutshell: /keys


```
{
"keysets": [
{
"id": "009a1f293253e41e",
"unit": "sat",
"keys": {
"1": "02194603ffa36356f4a56b7df9371fc31924723514 53ec7398b8da8117e7c3e104",
"2": "03b0f36d6d47ce14df8a7be9137712c42bcdd960b19dd02f1d4a9703b1f31d7513",
"4": "0366be6e026e42852498efb82014ca91e89da2e7a5bd3761bdad699fa2aec9fe09",
"8": "0253de5237f189606f29d8a690ea719f74d65f617bb1cb6fbea34f2bc4f930016d",
...
}
}
]
}
```

cashu in a nutshell: /keysets


```
{
"keysets": [
{
"id": "009a1f293253e41e",
"unit": "sat",
"active": True,
"input_fee_ppk": 100
},
{
"id": "0042ade98b2a370a",
"unit": "sat",
"active": False,
"input_fee_ppk": 100
}
]
}
```

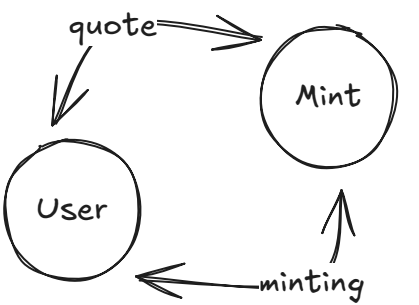
cashu in a nutshell: /mint

endpunkt zum erstellen von ecash

1. mint quote mit POST /mint/<method> erstellen
-> mint gibt MintQuoteResponse zurück

2. mint quote bezahlen und passendes ecash erstellen

3. ecash mit POST /mint/<method> signieren lassen
-> mint gibt MintResponse zurück



cashu in a nutshell: /melt

endpunkt zum einlösen von ecash

1. melt quote mit POST /melt/quote/<method> erstellen
-> wallet sendet zu zahlende invoice
-> mint gibt PostMeltQuoteBolt11Response zurück (enthält gebühr)

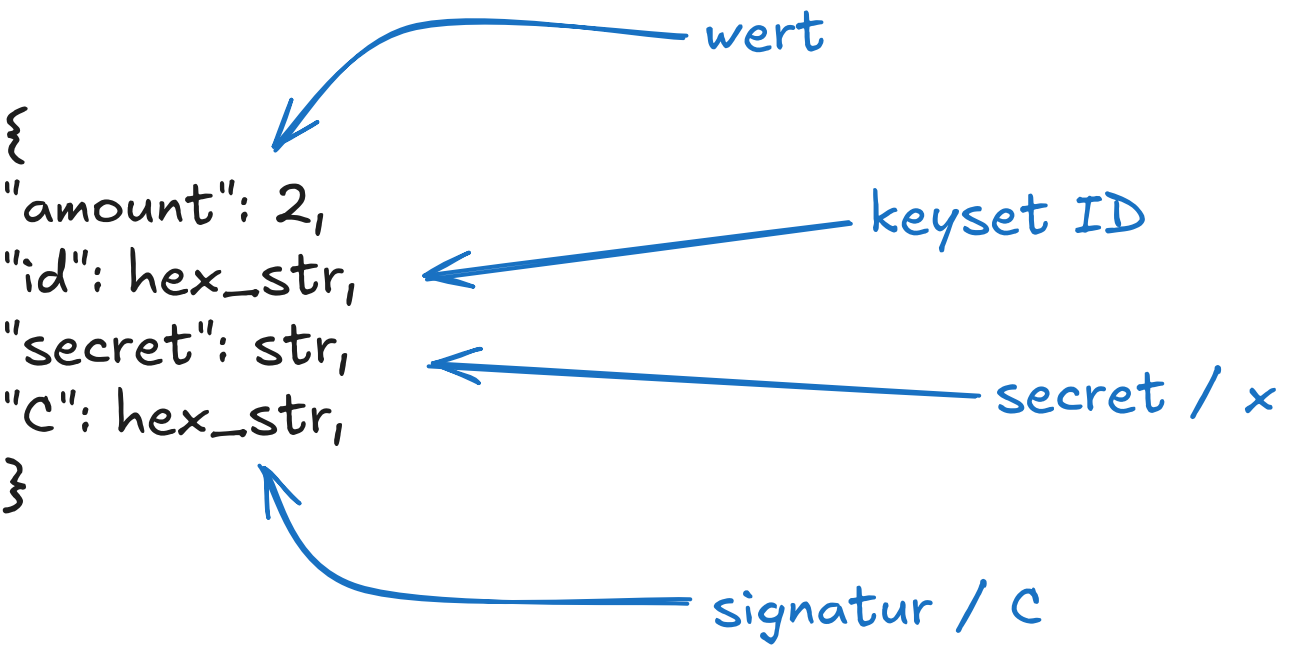
3. ecash in entsprechender höhe mit POST /melt/<method> an mint senden
-> mint gibt PostMeltQuoteBolt11Response zurück

cashu in a nutshell: /swap

endpunkt zum tauschen von ecash

1. zu tauschende proofs wählen und mit neuen blind messages an /v1/swap senden
-> mint gibt PostSwapResponse zurück, welche neue signaturen enthält.
-> das "alte" ecash ist jetzt verbraucht / ungültig.

cashu in a nutshell: proof



cashu in a nutshell: token