# 206.https://stackoverflow.com/questions/72370984/on-mainnet-beta-signmetadata-returns-a-program-error-whereas-it-works-on-dev

**T:**On mainnet-beta, signMetadata() returns a program error, whereas it works on devnet

**Q:**The odd part is, the transactions both look completely successful. It's just that signMetadata throws an exception when using mainnet-beta.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Two txns for comparison, using identical code:  
  
Mainnet-beta: https://solscan.io/tx/4zJ9qHRr3kDbhm7vMw4KmBmM4EZ8onx9ozbgKxrfw5wrvMTzaLfC1iDiYY1sQKbYKmYqVnDn4kntAwkwkG8US5YyDevnet: https://solscan.io/tx/3SZpHSEw3xhLKcBNiy6iPgkJvVnJNCCUjMfGHnmfvh7KWxd6zgnuDoQo1fbTCB4Uc6DqrsWfmDKAbyNR8g5wSCeK?cluster=devnet  
  
The general shape of the code that is creating those txns:  
  
const ret = await actions .mintNFT({ connection, wallet, uri: `${apiHost}/metadata/${uuid}.json`, maxSupply: 0, }) .then((res) => res); await sleep(20000); // Sign the txn const signTx = await actions.signMetadata({ connection, wallet, editionMint: ret.mint, signer: undefined, });  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Calling mintNFT then signMetadata after a 20 second delay, which I'd seen in some example doc or another.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
The specific exception: Error: failed to send transaction: Transaction simulation failed: Error processing Instruction 0: custom program error: 0x39  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
Even when I look at the related NFT mint, everything appears correct. They show up in my wallet and look right on Solscan.  
  
Mainnet-beta: https://solscan.io/token/KzFd3E5M1uZrYknzUmMnw7Z2xL11opQ24zmCBY9mPgD#txsDevnet: https://solscan.io/token/7v3kbyjQeEyP4Ubw73kmfHcvwK5Jo63S4fqiHSFCS3Lt?cluster=devnet#txs  
  
Package.json  
  
{ "private": true, "scripts": { "dev": "next dev", "build": "next build", "start": "next start", "lint": "next lint" }, "dependencies": { "@solana/web3.js": "^1.4", "bignumber.js": "^9", "@metaplex-foundation/mpl-core": "^0.0.4", "@metaplex-foundation/mpl-token-metadata": "^1.2.4", "@metaplex/js": "^4.11.2", "@pinata/sdk": "^1.1.23", "@types/node": "^17.0.12", "@types/react": "18.0.1", "@types/sharp": "^0.30.2", "@types/uuid": "^8.3.4", "dotenv": "^11.0.0", "isomorphic-fetch": "^3.0.0", "next": "12.1.4", "react": "18.0.0", "react-dom": "18.0.0", "sharp": "^0.29.3", "uuid": "^8.3.2" }, "devDependencies": { "eslint": "8.12.0", "eslint-config-next": "12.1.4", "ts-node": "^10.4.0", "typescript": "^4.5.4" }}  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]

1 **Answer**

**A1:**The preflight confirmation level of your connection may be set to finalized, which means that the transaction is simulated against the most recent finalized block, which is 32 slots ago.  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
If the cluster is going slowly, which mainnet-beta has been recently, with 732ms slots compared to 390ms slots in devnet, then it'll take longer to produce the 32 blocks.  
  
That way, when you try to simulate your second transaction 20 seconds later, your first one hasn't landed in a finalized block yet, so the cluster rejects it.  
  
To get around this, you can use a commitment level of confirmed on your connection, by creating it with:  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
const connection = new Connection('https://api.mainnet-beta.solana.com', 'confirmed');  
  
WARN: THIS PARAGRAPH CONTAINS TAG: [CODE]   
  
More information about confirmation levels: https://docs.solana.com/developing/clients/jsonrpc-api#configuring-state-commitment

**C1:**No sooner than I'd thought this was resolved than I'm seeing the same issue again, although this time only sometimes vs. every time. Could this be chalked up to mainnet network issues do you think?

**C2:**Most likely yes