Directly Calling CrocSwapDex

The most gas efficient approach is for users to directly call theCrocSwapDex contract. However with the swap path upgrade, swappers must use theuserCmd() function.

This method takes the same arguments asswap() but requires pre-formatting the arguments to the call into an ABI byte array.userCmd() should be called with a proxy index of 1 (the swap callapth) and the standard swap arguments ABI encoded.

Solidity

```

Copy // Ethereum address dex=0xAaAaAAAaA24eEeb8d57D431224f73832bC34f688

uint16SWAP_PROXY=1

CrocSwapDex(dex).userCmd(SWAP_PROXY,abi.encode( base, quote, poolIdx, isBuy, inBaseQty, qty, tip, limitPrice, minOut, settleFlags));

---

// Scroll address dex=0xAaAaAAAaA24eEeb8d57D431224f73832bC34f688

uint16SWAP_PROXY=1

CrocSwapDex(dex).userCmd(SWAP_PROXY,abi.encode( base, quote, poolIdx, isBuy, inBaseQty, qty, tip, limitPrice, minOut, settleFlags));

```

Javascript

```

Copy const{ethers}=require('ethers'); import{ AbiCoder }from"ethers/lib/utils";

constprovider=newethers.providers.JsonRpcProvider(...);

constrouterAbi=...// See below

constetherDexAddr="0xAaAaAAAaA24eEeb8d57D431224f73832bC34f688"
constscrollDexAddr="0xAaAaAAAaA24eEeb8d57D431224f73832bC34f688"

constcontract=newethers.Contract(etherRouterAddr,routerAbi,provider);

constabi=newethers.utils.AbiCoder() constcmd=abi.encode([ "address", "address", "uint256", "bool", "bool", "uint128", "uint16", "uint128", "uint128", "uint8"], [ base, quote, 420, isBuy, inBaseQty, qty, 0, limitPrice, minOut, settleFlags])

contract.userCmd(1,cmd)

```

ABI

```

Copy [{ "inputs":[ { "internalType":"uint16", "name":"callpath", "type":"uint16" }, { "internalType":"bytes", "name":"cmd", "type":"bytes" } ], "name":"userCmd", "outputs":[ { "internalType":"bytes", "name":"", "type":"bytes" } ], "stateMutability":"payable", "type":"function" }]

```

Previous Hot Path Swap Migration Next External Swap Router Last updated19 days ago On this page *Solidity * Javascript