250927-1300

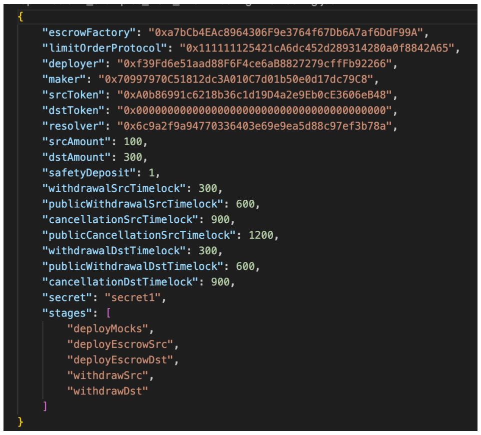
Saturday, September 27, 2025 12:54 PM

Attempt

Albert Hsueh at 2025-09-27 6:21 AM
Installation is simple enough...

> Foundryup
>
> Download jq and make sure the cli runs

The config has some addresses



```
{
    "escrowFactory": "0xa7bCb4EAc8964306F9e3764f67Db6A7af6DdF99A",
    "limitOrderProtocol": "0x111111125421cA6dc452d289314280a0f8842A65",
    "deployer": "0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266",
    "maker": "0x70997970C51812dc3A010C7d01b50e0d17dc79C8",
    "srcToken": "0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48",
    "dstToken": "0x0000000000000000000000000000000000000000",
    "resolver": "0x6c9a2f9a94770336403e69e9ea5d88c97ef3b78a",
    "srcAmount": 100,
    "dstAmount": 300,
    "safetyDeposit": 1,
    "withdrawalSrcTimelock": 300,
    "publicWithdrawalSrcTimelock": 600,
    "cancellationSrcTimelock": 900,
    "publicCancellationSrcTimelock": 1200,
    "withdrawalDstTimelock": 300,
    "publicWithdrawalDstTimelock": 600,
    "cancellationDstTimelock": 900,
    "secret": "secret1",
    "stages": [
        "deployMocks",
        "deployEscrowSrc",
        "deployEscrowDst",
        "withdrawSrc",
        "withdrawDst"
    ]
}
```

From the notion doc I got my access to the dev portal https://carnelian-raft-206.notion.site/Building-with-1inch-132af144f6708092be0ee25ec80cec4d#132af144f670809d8537dfe3d22e85f6:~:text=Since%20this%20is%20a%20hackathon%2C%20we%20are%20skipping%20our%20normal%20KYC%20requirements, best part is no kyc if I signup via this link.

API key is there. It goes in the developer_private_key.

RPC_URL= http://localhost:8545 as I'm just testing locally for now.

Chain_id=31337 the local chain

Maker_private_key= got to this airdop and follow the instructions to get a wallet, private key and some 1inch tokens https://carnelian-raft-206.notion.site/Claim-Your-1inch-Airdrop-279af144f6708077a37ae93c58757980

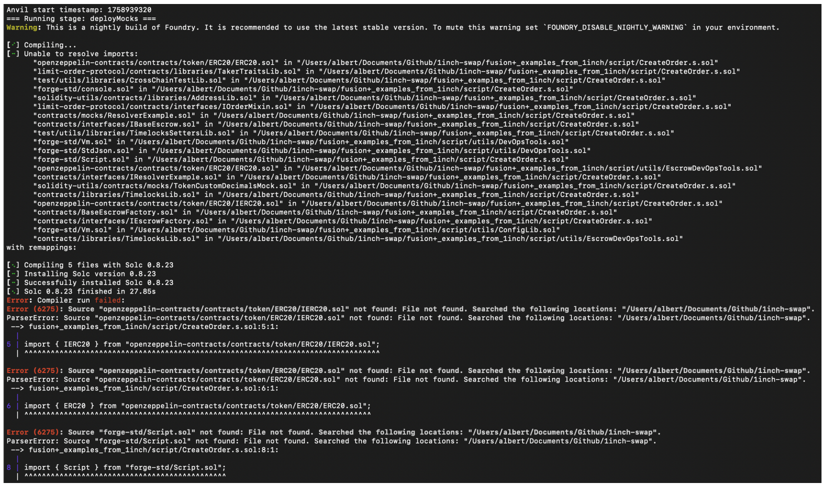
Albert Hsueh at 2025-09-27 7:33 AM
I'm quite stuck... I didn't get a redeem code for the airdrop so I'm not actually sure what to run here.

The script is confusing that it uses some rcp to copy some contracts over or something but I'm not sure what I'm actually supposed to do after. There were mentions of the contracts already deployed on sepolia

> **Aanand S — 1:00 AM**
> Hey guys, is it necessary to redeploy the fusion+ escrow contracts again on sepolia, is there any existing deployment we could use?
> **Tanner — 1:01 AM**
> There are contracts deployed already. You will find those addresses in this example:
> https://gist.github.com/belactriple9/8f0a029961981952ca418224c0b7c0eb

So... questions:

1) Do I put sepolia as my rpc so that it can copy it over to my local chain? 31337?
   a. Looks promising
2) What private key should I use?
   a. Used the 1inch wallet but I still don't have airdrop



```
Anvil start timestamp: 1758939320
=== Running stage: deployMocks ===
Warning: This is a nightly build of Foundry. It is recommended to use the latest stable version. To mute this warning set `FOUNDRY_DISABLE_NIGHTLY_WARNING` in your environment.

[·] Compiling...
[-] Unable to resolve imports:
      "openzeppelin-contracts/contracts/token/ERC20/ERC20.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "limit-order-protocol/contracts/libraries/TakerTraitsLib.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "test/utils/libraries/CrossChainTestLib.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "forge-std/console.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "solidity-utils/contracts/libraries/AddressLib.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "limit-order-protocol/contracts/interfaces/IOrderMixin.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "contracts/mocks/ResolverExample.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "contracts/interfaces/IBaseEscrow.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "test/utils/libraries/TimelocksSettersLib.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "forge-std/Vm.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/utils/DevOpsTools.sol"
      "forge-std/StdJson.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/utils/DevOpsTools.sol"
      "forge-std/Script.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "openzeppelin-contracts/contracts/token/ERC20/ERC20.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/utils/EscrowDevOpsTools.sol"
      "contracts/interfaces/IResolverExample.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "solidity-utils/contracts/mocks/TokenCustomDecimalsMock.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "contracts/libraries/TimelocksLib.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "openzeppelin-contracts/contracts/token/ERC20/IERC20.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "contracts/BaseEscrowFactory.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "contracts/interfaces/IEscrowFactory.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/CreateOrder.s.sol"
      "forge-std/Vm.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/utils/ConfigLib.sol"
      "contracts/libraries/TimelocksLib.sol" in "/Users/albert/Documents/Github/1inch-swap/fusion+_examples_from_1inch/script/utils/EscrowDevOpsTools.sol"
with remappings:

[·] Compiling 5 files with Solc 0.8.23
[-] Installing Solc version 0.8.23
[-] Successfully installed Solc 0.8.23
[·] Solc 0.8.23 finished in 27.85s
Error: Compiler run failed:
Error (6275): Source "openzeppelin-contracts/contracts/token/ERC20/IERC20.sol" not found: File not found. Searched the following locations: "/Users/albert/Documents/Github/1inch-swap".
ParserError: Source "openzeppelin-contracts/contracts/token/ERC20/IERC20.sol" not found: File not found. Searched the following locations: "/Users/albert/Documents/Github/1inch-swap".
 --> fusion+_examples_from_1inch/script/CreateOrder.s.sol:5:1:
  |
5 | import { IERC20 } from "openzeppelin-contracts/contracts/token/ERC20/IERC20.sol";
  | ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Error (6275): Source "openzeppelin-contracts/contracts/token/ERC20/ERC20.sol" not found: File not found. Searched the following locations: "/Users/albert/Documents/Github/1inch-swap".
ParserError: Source "openzeppelin-contracts/contracts/token/ERC20/ERC20.sol" not found: File not found. Searched the following locations: "/Users/albert/Documents/Github/1inch-swap".
 --> fusion+_examples_from_1inch/script/CreateOrder.s.sol:6:1:
  |
6 | import { ERC20 } from "openzeppelin-contracts/contracts/token/ERC20/ERC20.sol";
  | ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Error (6275): Source "forge-std/Script.sol" not found: File not found. Searched the following locations: "/Users/albert/Documents/Github/1inch-swap".
ParserError: Source "forge-std/Script.sol" not found: File not found. Searched the following locations: "/Users/albert/Documents/Github/1inch-swap".
 --> fusion+_examples_from_1inch/script/CreateOrder.s.sol:8:1:
  |
8 | import { Script } from "forge-std/Script.sol";
  | ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

Some back and forth with gpt... I need to set up the project lib properly with:

- Foundry.toml
- Remappings.txt

Okok I guess before I run example I need to set a lot more things up... I guess I skipped some steps... damn it.

Proper steps to get running examples/scripts/

1. Install rust, install foundry

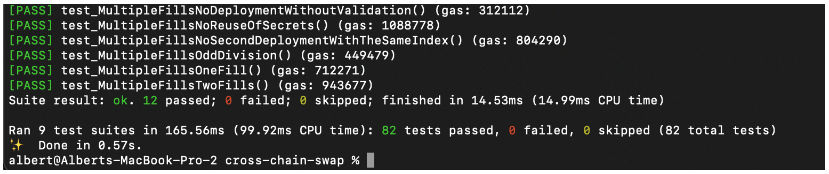
   `curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh`

   `foundryup`

2. Git clone the https://github.com/1inch/cross-chain-swap.git as a subrepo. The libs has multiple linked repos. If I just do the lazy way and download zip, it's going to be missing lib files for the next step
3. Forge build at the root folder of the repo cloned in step 2

   `forge build`

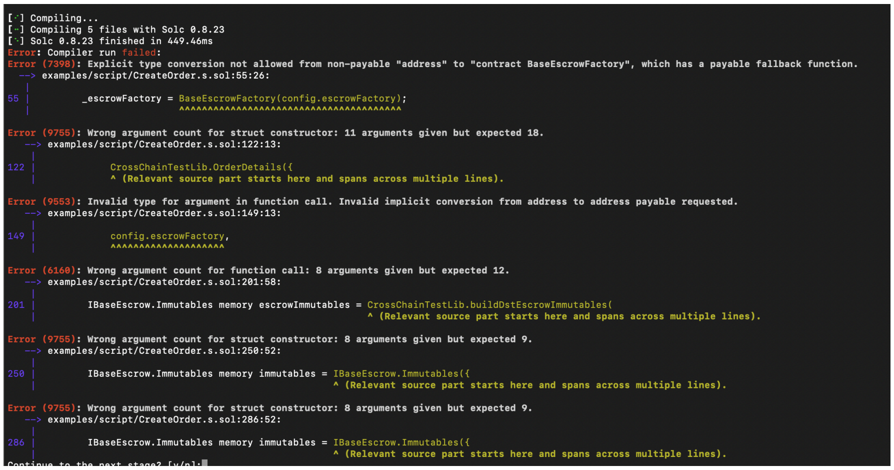
4. Apply my config and .env changes
5. Cd <into the scripts folder>
6. Chmod +x create_order.sh
7. ./create_order.sh



```
[PASS] test_MultipleFillsNoDeploymentWithoutValidation() (gas: 312112)
[PASS] test_MultipleFillsNoReuseOfSecrets() (gas: 1088778)
[PASS] test_MultipleFillsNoSecondDeploymentWithTheSameIndex() (gas: 804290)
[PASS] test_MultipleFillsOddDivision() (gas: 449479)
[PASS] test_MultipleFillsOneFill() (gas: 712271)
[PASS] test_MultipleFillsTwoFills() (gas: 943677)
Suite result: ok. 12 passed; 0 failed; 0 skipped; finished in 14.53ms (14.99ms CPU time)

Ran 9 test suites in 165.56ms (99.92ms CPU time): 82 tests passed, 0 failed, 0 skipped (82 total tests)
✨  Done in 0.57s.
albert@Alberts-MacBook-Pro-2 cross-chain-swap %
```

I did the test after the build and it passed.

Now I try run examples. But it give error. Big time.

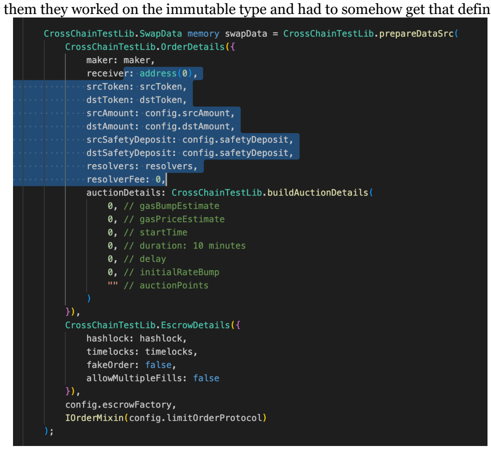


```
[⠊] Compiling...
[⠒] Compiling 5 files with Solc 0.8.23
[⠢] Solc 0.8.23 finished in 449.46ms
Error: Compiler run failed:
Error (7398): Explicit type conversion not allowed from non-payable "address" to "contract BaseEscrowFactory", which has a payable fallback function.
  --> examples/script/CreateOrder.s.sol:55:26:
   |
55 |         _escrowFactory = BaseEscrowFactory(config.escrowFactory);
   |                          ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Error (9755): Wrong argument count for struct constructor: 11 arguments given but expected 18.
   --> examples/script/CreateOrder.s.sol:122:13:
    |
122 |             CrossChainTestLib.OrderDetails({
    |             ^ (Relevant source part starts here and spans across multiple lines).

Error (9553): Invalid type for argument in function call. Invalid implicit conversion from address to address payable requested.
   --> examples/script/CreateOrder.s.sol:149:13:
    |
149 |             config.escrowFactory,
    |             ^^^^^^^^^^^^^^^^^^^^

Error (6160): Wrong argument count for function call: 8 arguments given but expected 12.
   --> examples/script/CreateOrder.s.sol:201:58:
    |
201 |         IBaseEscrow.Immutables memory escrowImmutables = CrossChainTestLib.buildDstEscrowImmutables(
    |                                                          ^ (Relevant source part starts here and spans across multiple lines).

Error (9755): Wrong argument count for struct constructor: 8 arguments given but expected 9.
   --> examples/script/CreateOrder.s.sol:250:52:
    |
250 |         IBaseEscrow.Immutables memory immutables = IBaseEscrow.Immutables({
    |                                                    ^ (Relevant source part starts here and spans across multiple lines).

Error (9755): Wrong argument count for struct constructor: 8 arguments given but expected 9.
   --> examples/script/CreateOrder.s.sol:286:52:
    |
286 |         IBaseEscrow.Immutables memory immutables = IBaseEscrow.Immutables({
    |                                                    ^ (Relevant source part starts here and spans across multiple lines).
Continue to the next stage? [y/n]:
```

Albert Hsueh at 2025-09-27 12:23 PM
Could be a .sol code error says the mentor. Tried to fix it but there's too much to fix. If the fix were isolated in just the createorder.s.sol, then ok. But nope.

Albert Hsueh at 2025-09-27 12:28 PM
Ideas from talking to another team and mentor that had experience building with extensions:

- Look at 1inch fusion+ paper: https://1inch.io/assets/1inch-fusion-plus.pdf
- For them they worked on the immutable type and had to somehow get that definition to the other chain

  ```
  CrossChainTestLib.SwapData memory swapData = CrossChainTestLib.prepareDataSrc(
      CrossChainTestLib.OrderDetails({
          maker: maker,
          receiver: address(0),
          srcToken: srcToken,
          dstToken: dstToken,
          srcAmount: config.srcAmount,
          dstAmount: config.dstAmount,
          srcSafetyDeposit: config.safetyDeposit,
          dstSafetyDeposit: config.safetyDeposit,
          resolvers: resolvers,
          resolverFee: 0,
          auctionDetails: CrossChainTestLib.buildAuctionDetails(
              0, // gasBumpEstimate
              0, // gasPriceEstimate
              0, // startTime
              0, // duration: 10 minutes
              0, // delay
              0, // initialRateBump
              "" // auctionPoints
          )
      }),
      CrossChainTestLib.EscrowDetails({
          hashlock: hashlock,
          timelocks: timelocks,
          fakeOrder: false,
          allowMultipleFills: false
      }),
      config.escrowFactory,
      IOrderMixin(config.limitOrderProtocol)
  );
  ```

  I've seen multiple usage of it in CreateOrder.s.sol.
- Check 1inch github for any mention of auction (we're trying hard to get it cuz the previous implementation doesn't seem to be using auction, it interacted with Limit order protocol directly). Connecting with above snippet, I'd want to search for:

  `CrossChainTestLib`

  Yields

https://github.com/1inch/cross-chain-swap/blob/d0a59ab2c4b6be5c9769d5775769681873fcf162/test/utils/libraries/CrossChainTestLib.sol#L4

Key hints:

- OrderDetails contains aution details
- SwapData.order contains the above OrderDetails
- Which comes back to OrderCreate.s.sol like this:

  ```
  CrossChainTestLib.SwapData memory swapData = CrossChainTestLib.prepareDataSrc(
  ```

  And

  ```
   IResolverExample(resolver).deploySrc(
              swapData.immutables,
              swapData.order,
              r,
              vs,
              config.srcAmount,
              takerTraits,
              args
          );
  ```

  Where

  ```
  // SPDX-License-Identifier: MIT
  pragma solidity 0.8.23;
  import { IOrderMixin } from "limit-order-
  protocol/contracts/interfaces/IOrderMixin.sol";
  import { TakerTraits } from "limit-order-
  protocol/contracts/libraries/TakerTraitsLib.sol";
  import { IBaseEscrow } from "../interfaces/IBaseEscrow.sol";
  /**
   * @title Interface for the sample implementation of a Resolver contract for
  cross-chain swap.
   * @custom:security-contact security@1inch.io
   */
  interface IResolverExample {
      error InvalidLength();
      error LengthMismatch();
      /**
       * @notice Deploys a new escrow contract for maker on the source chain.
       * @param immutables The immutables of the escrow contract that are used in
  deployment.
       * @param order Order quote to fill.
       * @param r R component of signature.
       * @param vs VS component of signature.
       * @param amount Taker amount to fill
       * @param takerTraits Specifies threshold as maximum allowed takingAmount
  when takingAmount is zero, otherwise specifies
       * minimum allowed makingAmount. The 2nd (0 based index) highest bit
  specifies whether taker wants to skip maker's permit.
       * @param args Arguments that are used by the taker (target, extension,
  interaction, permit).
       */
      function deploySrc(
          IBaseEscrow.Immutables calldata immutables,
          IOrderMixin.Order calldata order,
          bytes32 r,
          bytes32 vs,
          uint256 amount,
          TakerTraits takerTraits,
          bytes calldata args
      ) external;
      /**
       * @notice Deploys a new escrow contract for taker on the destination
  chain.
       * @param dstImmutables The immutables of the escrow contract that are used
  in deployment.
       * @param srcCancellationTimestamp The start of the cancellation period for
  the source chain.
       */
      function deployDst(IBaseEscrow.Immutables calldata dstImmutables, uint256
  srcCancellationTimestamp) external payable;
      /**
       * @notice Allows the owner to make arbitrary calls to other contracts on
  behalf of this contract.
       * @param targets The addresses of the contracts to call.
       * @param arguments The arguments to pass to the contract calls.
       */
      function arbitraryCalls(address[] calldata targets, bytes[] calldata
  arguments) external;
  }
  ```

  this is the Iresolverexample.. it's an interface.. what? and so if i want to write my own resolver that inherits the dutch auction capability what would i minimally need?

  > The Dutch auction itself **is not implemented inside the resolver**. It is already encoded inside the order (auctionDetails → gettersAmountData → passed into buildOrder) and enforced by the **Limit Order Protocol** when the resolver calls fillOrder.
  >
  > Albert Hsueh at 2025-09-27 12:53 PM
  > ... and now then I guess I need to fill in my knowledge about this...

- They confirmed that for them the next best hint was looking at https://github.com/1inch/cross-chain-resolver-example rather than the one I've been playing with