**Adding a consumer**

**Adding a consumer**

Remember how everything started from a simple and inoffensive InvalidConsumer() error? Now it's the moment we finally fix it!

Open Interactions.s.sol and create a new contract:

contract AddConsumer is Script {

function run() external {

}

}

To be able to add a consumer we need the most recent deployment of the Raffle contract. To grab it we need to install the following:

forge install Cyfrin/foundry-devops --no-commit

Import it at the top of the Interactions.s.sol:

import {DevOpsTools} from "lib/foundry-devops/src/DevOpsTools.sol";

Update the run function to get the address and call addConsumerUsingConfig(raffle):

function run() external {

address raffle = DevOpsTools.get\_most\_recent\_deployment("MyContract", block.chainid);

addConsumerUsingConfig(raffle);

}

And right about now, everything should feel extremely familiar. Let's define addConsumerUsingConfig and all the rest:

contract AddConsumer is Script {

function addConsumer(address raffle, address vrfCoordinator, uint256 subscriptionId) public {

console.log("Adding consumer contract: ", raffle);

console.log("Using VRFCoordinator: ", vrfCoordinator);

console.log("On chain id: ", block.chainid);

vm.startBroadcast();

VRFCoordinatorV2Mock(vrfCoordinator).addConsumer(subscriptionId, raffle);

vm.stopBroadcast();

}

function addConsumerUsingConfig(address raffle) public {

HelperConfig helperConfig = new HelperConfig();

(

,

,

address vrfCoordinator,

,

uint256 subscriptionId,

,

) = helperConfig.activeNetworkConfig();

addConsumer(raffle, vrfCoordinator, subscriptionId);

}

function run() external {

address raffle = DevOpsTools.get\_most\_recent\_deployment("MyContract", block.chainid);

addConsumerUsingConfig(raffle);

}

}

So... what happened here?

1. We used DevOpsTools to grab the last deployment of the Raffle contract inside the run function;
2. We also call addConsumerUsingConfig inside the run function;
3. We define addConsumerUsingConfig as a public function taking an address as an input;
4. We deploy a new HelperConfig and call activeNetworkConfig to grab the vrfCoordinate and subscriptionId addresses;
5. We call the addConsumer function;
6. We define addConsumer as a public function taking 3 input parameters: address of the raffle contract, address of vrfCoordinator and subscriptionId;
7. We log some things useful for debugging;
8. Then, inside a startBroadcast- stopBroadcast block we call the addConsumer function from the VRFCoordinatorV2Mock using the right input parameters;

Try a nice forge build and check if everything is compiling. Perfect!

Let's go back to DeployRaffle.s.sol and import the thing we added in Interactions.s.sol:

import {CreateSubscription, FundSubscription, AddConsumer} from "./Interactions.s.sol";

Now let's integrate the FundSubscription with the CreateSubscription bit:

if (subscriptionId == 0) {

CreateSubscription createSubscription = new CreateSubscription();

subscriptionId = createSubscription.createSubscription(vrfCoordinator);

FundSubscription fundSubscription = new FundSubscription();

fundSubscription.fundSubscription(vrfCoordinator, subscriptionId, link);

}

So we created a subscription and funded it. Following on the DeploymentRaffle script deploys the Raffle contract. Now, that we have its address, we can add it as a consumer.

Great work!

Remember what got us on this path. All we wanted to do was call the testDontAllowPlayersToEnterWhileRaffleIsCalculating test from RaffleTest.t.sol. Let's try that again now:

forge test --mt testDontAllowPlayersToEnterWhileRaffleIsCalculating -vv

Ran 1 test for test/unit/RaffleTest.t.sol:RaffleTest

[PASS] testDontAllowPlayersToEnterWhileRaffleIsCalculating() (gas: 151240)

Logs:

Creating subscription on ChainID: 31337

Your sub Id is: 1

Please update subscriptionId in HelperConfig!

Funding subscription: 1

Using vrfCoordinator: 0x90193C961A926261B756D1E5bb255e67ff9498A1

On ChainID: 31337

Adding consumer contract: 0x50EEf481cae4250d252Ae577A09bF514f224C6C4

Using VRFCoordinator: 0x90193C961A926261B756D1E5bb255e67ff9498A1

On chain id: 31337

Suite result: ok. 1 passed; 0 failed; 0 skipped; finished in 11.06ms (102.80µs CPU time)

Amazing work!

There is a lot more to do in this section, but you are a true hero for reaching this point, take a well-deserved break! See you in the next one!