

Deploy a dapp on your Arbitrum rollup devnet

First, review the [Arbitrum integration](#), [Deploy an Arbitrum rollup devnet](#), and [Deploy a smart contract to your Arbitrum rollup](#) pages.

Dependencies

- a funded account to deploy your smart contract
- an [Arbitrum rollup devnet](#)
- running

Setup and contract deployment

1. Clone thegm-portal
2. from Github and start the frontend:
3. bash
4. cd
5. HOME
6. git
7. clone
8. <https://github.com/jcstein/gm-portal.git>
9. cd
10. gm-portal
11. &&
12. git
13. checkout
14. arbitrum
15. cd
16. frontend
17. &&
18. yarn
19. &&
20. yarn
21. dev
22. cd
23. HOME
24. git
25. clone
26. <https://github.com/jcstein/gm-portal.git>
27. cd
28. gm-portal
29. &&
30. git
31. checkout
32. arbitrum
33. cd
34. frontend
35. &&
36. yarn
37. &&
38. yarn
39. dev
40. In a new terminal instance, set your private key for the faucet as a variable and the RPC URL you're using:
41. bash
42. export
43. PRIVATE_KEY
44. =
45. 0xb6b15c8cb491557369f3c7d2c287b053eb229daa9c22138887752191c9520659
46. export
47. ARB_RPC_URL
48. =
49. <http://localhost:8547>
50. export
51. PRIVATE_KEY
52. =
53. 0xb6b15c8cb491557369f3c7d2c287b053eb229daa9c22138887752191c9520659

```

54. export
55. ARB_RPC_URL
56. =
57. http://localhost:8547
58. Change into thegm-portal/contracts
59. directory in the same terminal and deploy the contract using Foundry:
60. bash
61. cd
62. HOME
63. /gm-portal/contracts
64. forge
65. script
66. script/GmPortal.s.sol:GmPortalScript
67. --rpc-url
68. ARB_RPC_URL
69. --private-key
70. PRIVATE_KEY
71. --broadcast
72. cd
73. HOME
74. /gm-portal/contracts
75. forge
76. script
77. script/GmPortal.s.sol:GmPortalScript
78. --rpc-url
79. ARB_RPC_URL
80. --private-key
81. PRIVATE_KEY
82. --broadcast
83. In the output of the deployment, find the contract address and set it as a variable:
84. bash
85. export
86. CONTRACT_ADDRESS
87. =<
88. your-contract-address-from-the-output-abov
89. e
90.

91. export
92. CONTRACT_ADDRESS
93. =<
94. your-contract-address-from-the-output-abov
95. e
96.

```

Interact with the contract

Next, you're ready to interact with the contract from your terminal!

```

1. Send a "gm" to the contract:
2. bash
3. cast
4. send
5. CONTRACT_ADDRESS
6. \
7. "gm(string)"
8. "gm"
9. \
10. --private-key PRIVATE_KEY
11. \
12. --rpc-url ARB_RPC_URL
13. cast
14. send
15. CONTRACT_ADDRESS
16. \
17. "gm(string)"
18. "gm"
19. \
20. --private-key PRIVATE_KEY

```

```
21. \  
22. --rpc-url ARB_RPC_URL  
23. Now that you've posted to the contract, you can read all "gms" (GMs) from the contract with this command:  
24. bash  
25. cast  
26. call  
27. CONTRACT_ADDRESS  
28. "getAllGms()"  
29. --rpc-url  
30. ARB_RPC_URL  
31. cast  
32. call  
33. CONTRACT_ADDRESS  
34. "getAllGms()"  
35. --rpc-url  
36. ARB_RPC_URL  
37. Next, query the total number of gms, which will be returned as a hex value:  
38. bash  
39. cast  
40. call  
41. CONTRACT_ADDRESS  
42. "getTotalGms()"  
43. --rpc-url  
44. ARB_RPC_URL  
45. cast  
46. call  
47. CONTRACT_ADDRESS  
48. "getTotalGms()"  
49. --rpc-url  
50. ARB_RPC_URL  
51. (Optional) In order to interact with the contract on the frontend, you'll need to fund an account that you have in your  
    Ethereum wallet. Transfer to an external account with this command:  
52. bash  
53. export  
54. RECEIVER  
55. =<  
56. receiver  
57. ETH  
58. addres  
59. s  
60.  
  
61. cast  
62. send  
63. --private-key  
64. PRIVATE_KEY RECEIVER  
65. --value  
66. 1  
67. ether  
68. --rpc-url  
69. ARB_RPC_URL  
70. export  
71. RECEIVER  
72. =<  
73. receiver  
74. ETH  
75. addres  
76. s  
77.  
  
78. cast  
79. send  
80. --private-key  
81. PRIVATE_KEY RECEIVER  
82. --value  
83. 1  
84. ether  
85. --rpc-url  
86. ARB_RPC_URL
```

87. TIP

88. If you are in a different terminal than the one you set the private key in, you may need to set it again.

Update the frontend

Next, you will need to update a few things before you can interact with the contract on the frontend:

1. Change the contract address ongm-portal/frontend/src/App.tsx
2. to your contract address
3. Match the chain info ongm-portal/frontend/src/main.tsx
4. with the chain config of your L2
5. If you changed the contract, update the ABI ingm-portal/frontend/GmPortal.json
6. fromgm-portal/contracts/out/GmPortal.sol/GmPortal.json
7. . This can be done with:

```
bash cd HOME cp
```

```
dev/gm-portal/contracts/out/GmPortal.sol/GmPortal.json
```

```
dev/gm-portal/frontend cd HOME cp
```

```
dev/gm-portal/contracts/out/GmPortal.sol/GmPortal.json
```

```
dev/gm-portal/frontend
```

Interact with the frontend

Now, login with your wallet that you funded, and post a GM on your GM portal! [\[Edit this page on GitHub\]](#) Last updated:
[Previous page Deploy a smart contract on Arbitrum rollup](#)[Next page Intro to OP Stack integration](#) []