

Actor types

In the Filecoin network, an address is a unique identifier that refers to an actor in the Filecoin state. All actors in Filecoin have a corresponding address which varies from the different usages.

The Filecoin EVM runtime introduces three new actor types:

1. [Placeholder actors](#)
2. .
3. [Ethereum-style accounts](#)
4. , also called `EthAccount`
5. .
6. [EVM smart contracts](#)
7. .
8. .

Placeholder

A placeholder is a particular type of pseudo-actor that holds funds until an actual actor is deployed at a specific address. When funds are sent to an address starting with `f410f` that doesn't belong to any existing actor, a placeholder is created to hold the said funds until either an account or smart contract is deployed to that address.

A placeholder can become a real actor in one of two ways:

1. A message is sent from the account that would exist at that placeholder's address. If this happens, the placeholder is automatically upgraded into an account.
2. An EVM smart contract is deployed to the address.
3. .

Ethereum-style account

An Ethereum-style account is the Filecoin EVM runtime equivalent of an account with an `f1 or f3` address, also known as native accounts. However, there are a few key differences:

1. These accounts have `0x`
2. -style addresses and an equivalent `f`
3. -style address starting with `f410f`
4. .
5. Messages from these accounts can be sent with Ethereum wallets like MetaMask by connecting the wallet to a Filecoin client.
6. These accounts can be used to transfer funds to native or Ethereum-style.
7. They can be used to call EVM smart contracts and can be used to deploy EVM smart contracts. However, they cannot be used to call native actors such as multisig or miner actors.
8. .

EVM smart contract

An EVM smart contract actor hosts a single EVM smart contract. Every EVM smart contract will have an `0x` -style address.

Deploying

An EVM smart contract can be deployed in one of three ways:

1. An existing EVM smart contract can use the EVM's `CREATE`
2. `CREATE2`
3. opcode.
4. Ethereum-native tooling can be used in conjunction with an Ethereum-style account such as [Remix](#)
5. or [Hardhat](#)
6. .
7. A native account can call method `4`
8. on the Ethereum account manager `f010`
9. , passing the EVM init code as a CBOR-encoded byte-string (major type 2) in the message parameters.
10. .

Calling

An EVM smart contract may be called in one of three ways:

1. An EVM smart contract can use the EVM's `CALL`

2. opcode.
3. Ethereum-native tooling, like [MetaMask](#)
4. , can be used in conjunction with an Ethereum-style account.
5. Finally, a native account can call method3844450837
6. (FRC42(InvokeEVM)
7.):
 1. The input data should either be empty or encoded as a CBOR byte string.
8.
 1. The return data will either be empty or encoded as a CBOR byte string.
9. 3.
10. 3.
- 11.

[Previous Filecoin EVM-runtime Next Address types](#)

Last updated 7 months ago