

# Testnet Wallets

How to create and fund a testnet wallets to use for developing Celo dApps.

## Getting Started

While developing and deploying dApps on Celo, it's helpful to have a wallet prepared with funds to pay for any transactions you make on the blockchain. These can be set up using either real or test funds, and this allows application developers and users to interact Celo applications more easily.

tip This guide will focus on funding an account on MetaMask with Celo [Alfajores testnet](#) tokens). You can also use the [Celo Extension Wallet](#) if you prefer. Additionally, you can fund your wallet with real Celo if you would like to deploy to the Celo mainnet.

## Download a Wallet

When deploying a dApp, you will need to pay for transactions that write data to the Celo blockchain. This is done using a wallet funded with Celo on your preferred network.

- Download [MetaMask](#)
- for your browser
- If using MetaMask, [Connect MetaMask to Celo](#)
- ([video](#)
- ) to access Celo network options from your wallet

## Create your Account

You are now able to fund an account that you can access using your browser extension wallet. You can do this using accounts you create with the wallet, ganache or from the [Celo CLI](#) .

### Using a wallet

MetaMask allows you to easily create and manage accounts. This provides you with a quick way to create and fund an account to use when interacting with your dApp.

- From your MetaMask extension, select Celo (Alfajores Tesnet)
- Choose wallet Settings
- , select Create Account
- , name your account, and select Create
- View your account details and copy your account address from the top center of the user interface

tip Skip to [this step](#) to fund your account.

### Using Ganache

Deploying locally requires that you have a local blockchain setup. Install and start a local blockchain using the Celo ganache-cli.

#### Install Ganache

```
npm
install
-g @celo/ganache-cli
```

#### Start Ganache

```
ganache-cli --port
```

7545 Ganache will return a list of available accounts, private keys, and additional wallet details that you can access on port 7545.

Note that @celo/ganache-cli does not currently allow you to send transactions and pay fees in stable currencies, nor does it include the core protocol contracts. To run a development chain with the protocol contracts consider using the [celo-devchain package](#).

tip Skip to [this step](#) to connect your accounts to your wallet.

## Using Celo CLI

Install celocli by running

```
npm
```

```
install
```

```
-g @celo/celocli
```

## Create an Account

celocli account:new Creating an account will return its account details as shown below (details will be specific to your account).

mnemonic: turtle cash neutral drift brisk young swallow raw payment drill mail wear penalty vibrant entire adjust near chapter mistake size angry planet slam demand accountAddress: 0x5986ac413fA0C4A0379A674Cb986A59a962FC84e privateKey: 8cab22c2bb08f0d20bd9e1109a156e87219d63a2c0b40b027483decf194bd787 publicKey: 024baaae61bab2a6e16ccb008c78dddb7132fc48d082e2a6166f8cc52d8d7a5289 address: 0x5986ac413fA0C4A0379A674Cb986A59a962FC84e tip Skip to [this step](#) to connect your account to your wallet.

## Import Account to Wallet (for Options 2 & 3)

If you created an account using option 2 or 3, you can now import these accounts to your wallet.

- Open your wallet browser extension from your browser
- Select Settings
- Import Account
- Copy the private key
- from your local account, paste it into the window provided, and select Import

## Fund your Account

No matter where you created your accounts, you can send them testnet funds using your Account Address and the testnet faucet.

- Navigate to the [Alfajores Testnet Faucet](#)
- Copy your address from your terminal or wallet
- Paste this address into the Testnet Faucet
- , complete the Captcha
- , and click Get Started

Wait for the transaction to process to view your funds in your account [Edit this page](#) [Previous Deploy with Replit](#) [Next Celo Local Development Chain with Protocol Contracts](#)