# 2023327100056 23计算机 李凯涛

# BTC文档

## 安装软件和网络配置

**关键要点**

* 页面内容为开发者构建比特币应用程序提供入门指导，重点介绍Bitcoin Core的安装和配置。
* 包括三个主要程序：bitcoind（编程用全节点）、bitcoin-qt（带GUI的钱包）和bitcoin-cli（命令行工具）。
* 强调安全配置bitcoin.conf文件并使用测试网络（如testnet或regtest）进行开发。

**页面内容概述**

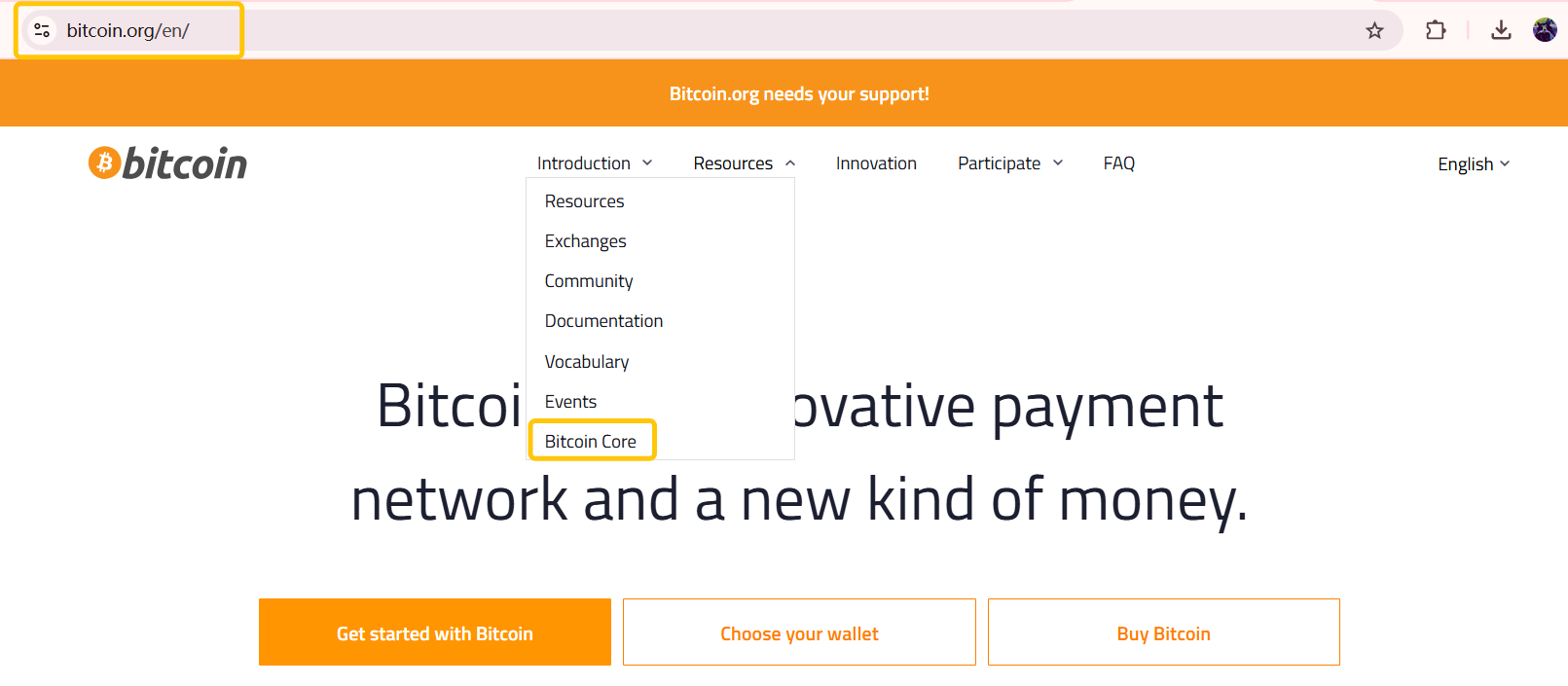
该链接 https://developer.bitcoin.org/examples/intro.html#introduction 是比特币开发者文档的一部分，旨在帮助开发者开始构建基于比特币的应用程序。页面详细介绍了如何安装和配置Bitcoin Core，包括其主要组件及其用途。

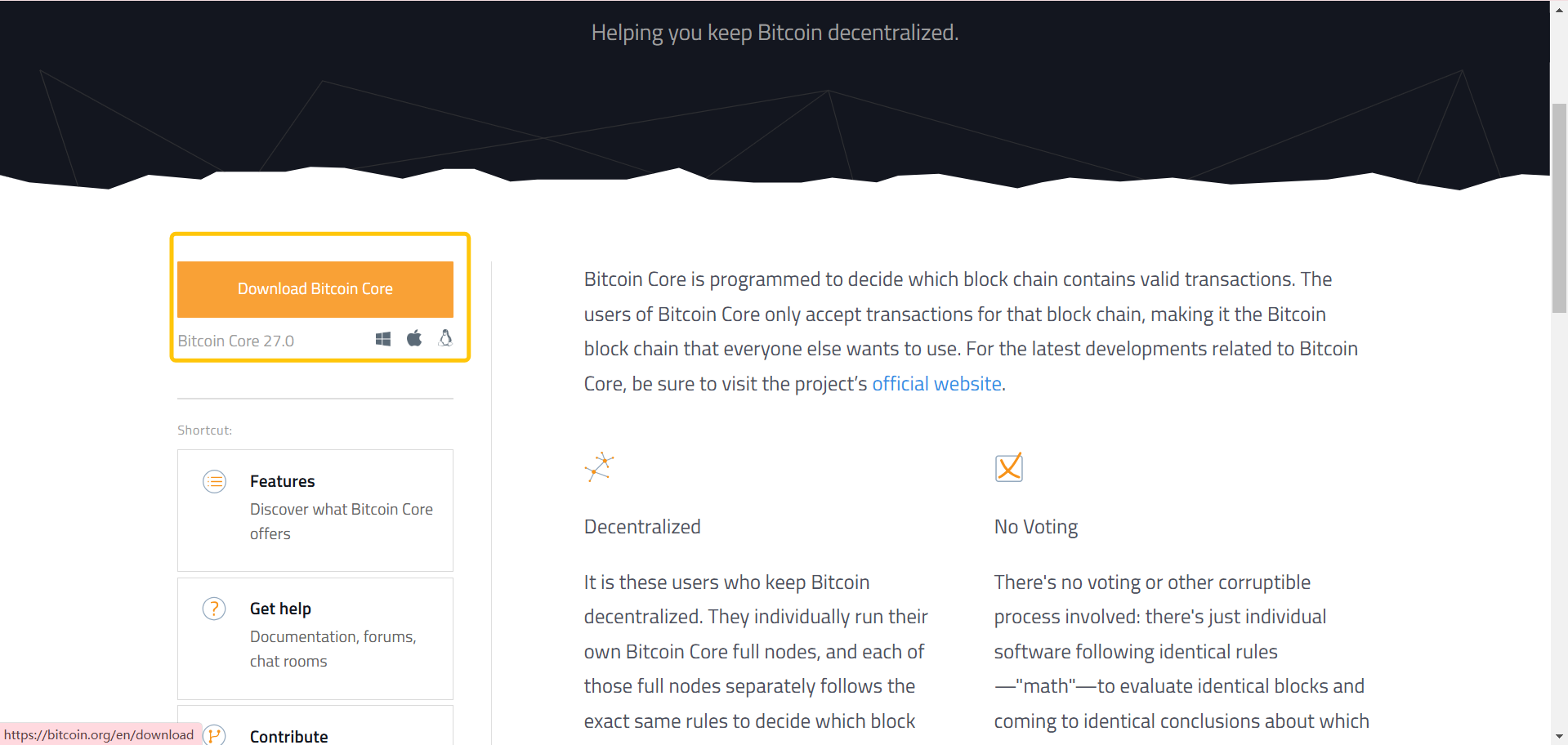
**Bitcoin Core的安装与程序**

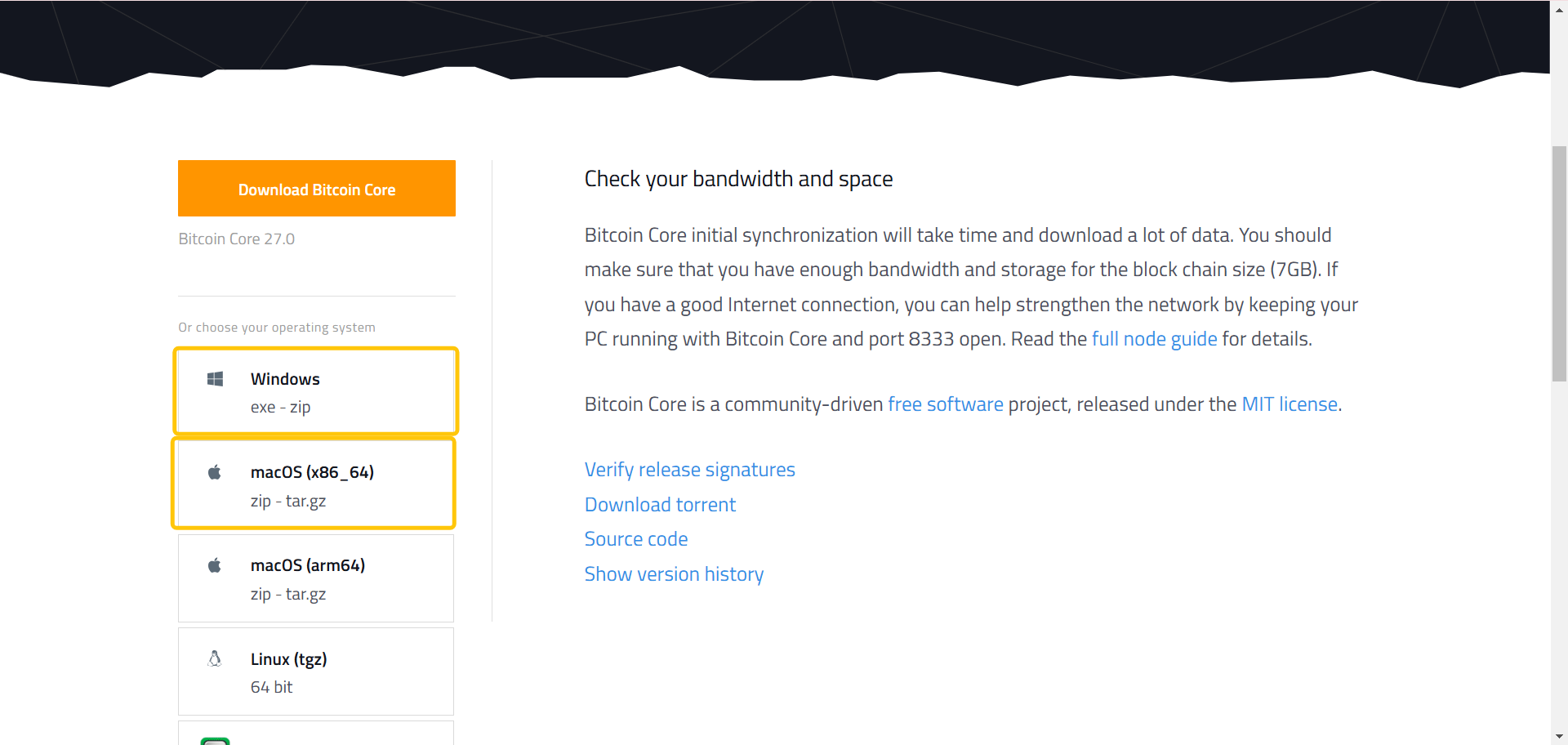
Bitcoin Core是官方比特币客户端，开发者可以通过 GitHub 或 bitcoin.org 下载安装。安装后，包含以下三个程序：

* bitcoind：全节点程序，适合编程使用，通过RPC（远程过程调用）与端口8332（主网）或18332（测试网）交互。
* bitcoin-qt：带图形界面的全节点和钱包，开发者可通过帮助菜单访问RPC控制台。
* bitcoin-cli：命令行工具，用于向bitcoind发送RPC命令，例如运行bitcoin-cli help查看可用命令。

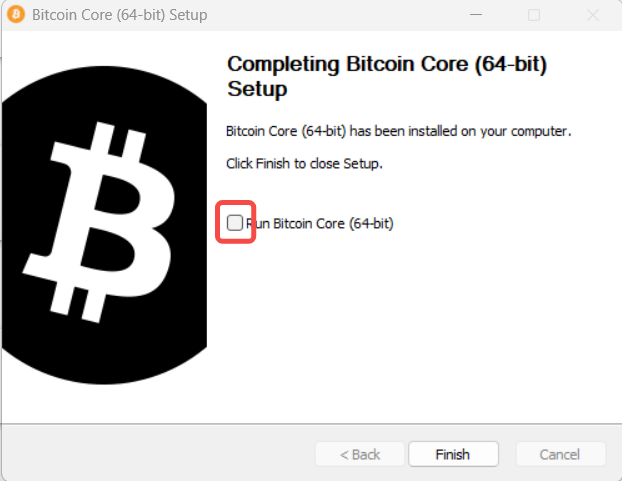
**安装**



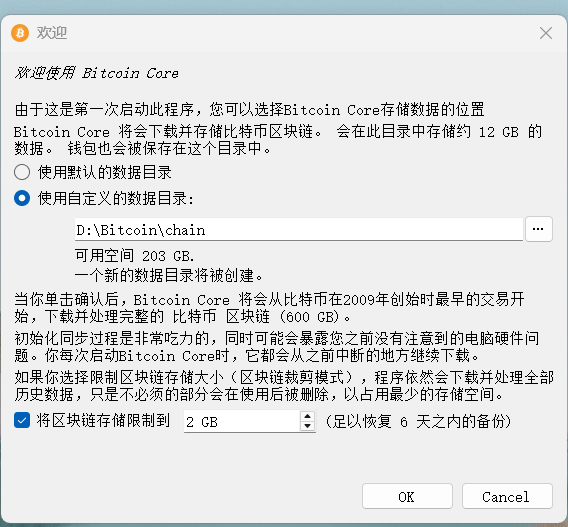




**安装过程注意：不要勾选这个，同时安装后就不要打开软件**

****

**安装过程注意：使用默认的数据目录，不要进行自定义数据目录**



**不要打开软件，它会自动更新区块链，把软件页面关闭掉**

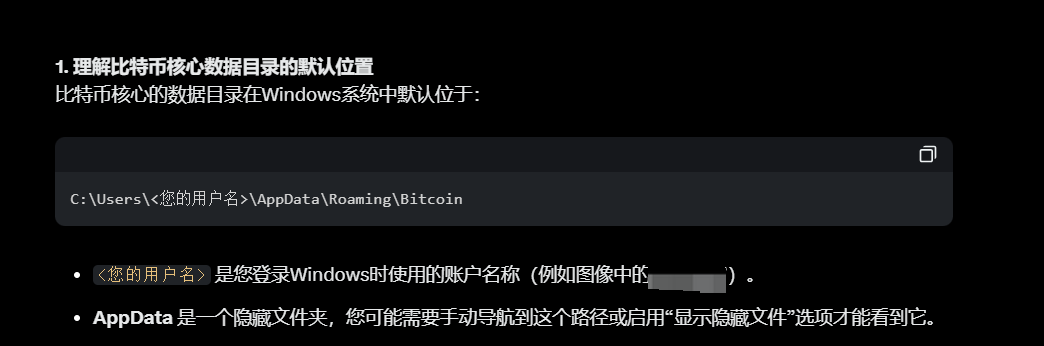


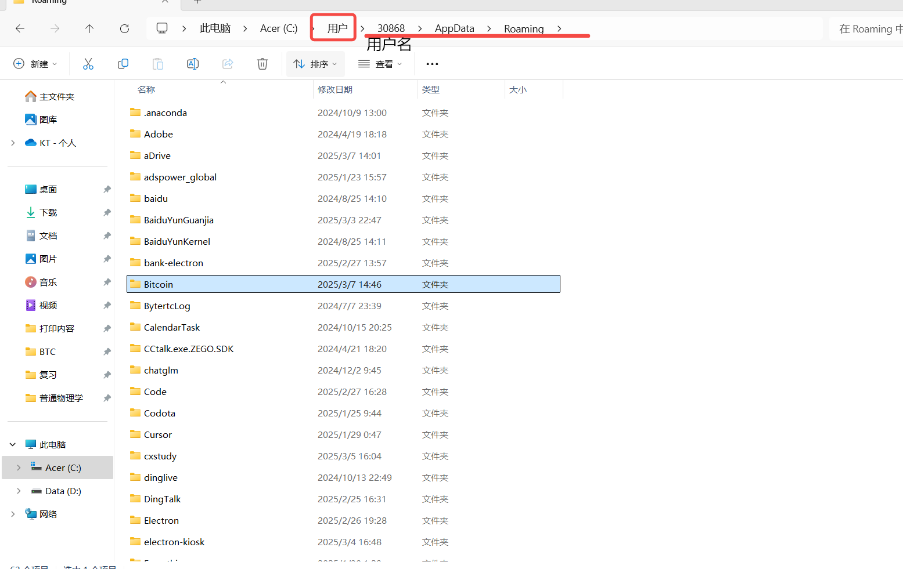
**配置与安全**

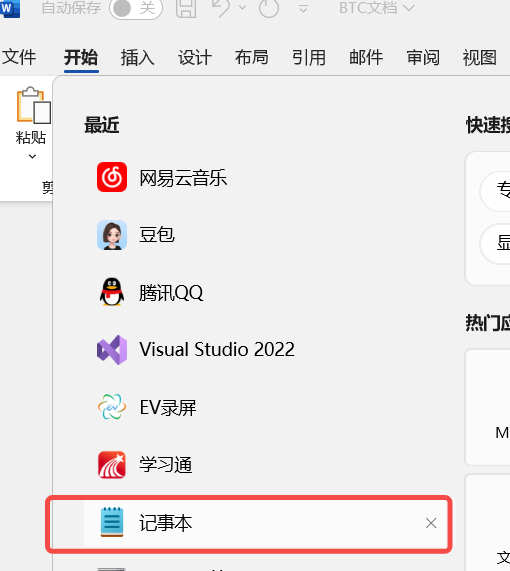
在比特币应用目录下创建bitcoin.conf文件，设置rpcpassword以保护RPC连接。文件位置因操作系统不同：

* Windows：%APPDATA%\Bitcoin\
* macOS：$HOME/Library/Application Support/Bitcoin/
* Linux：$HOME/.bitcoin/

如何找到比特币数据目录

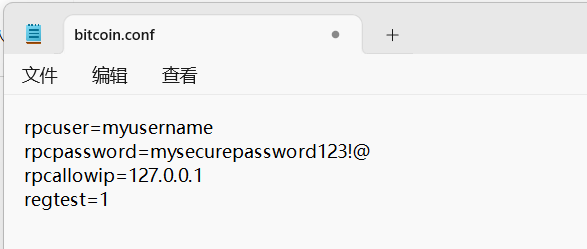


  
打开新的文本编辑器不要在文件夹中直接新建文本



然后设置rpcuser=姓名rpcpassword=密码

保持不变：rpcallowip=127.0.0.1 regtest=1

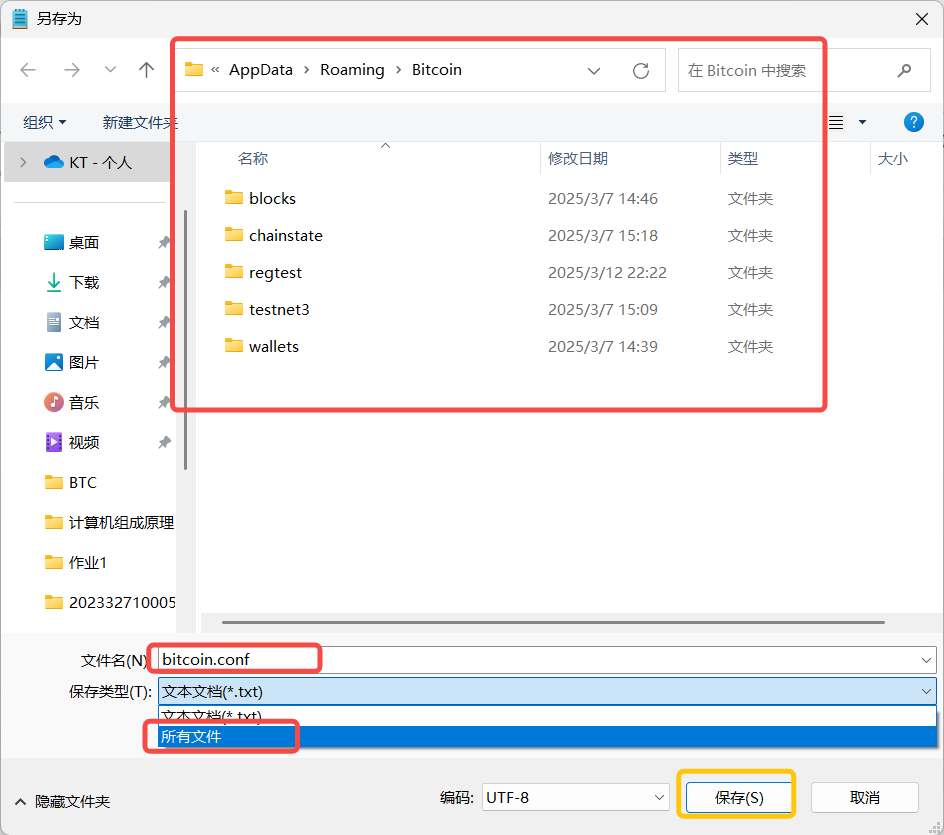


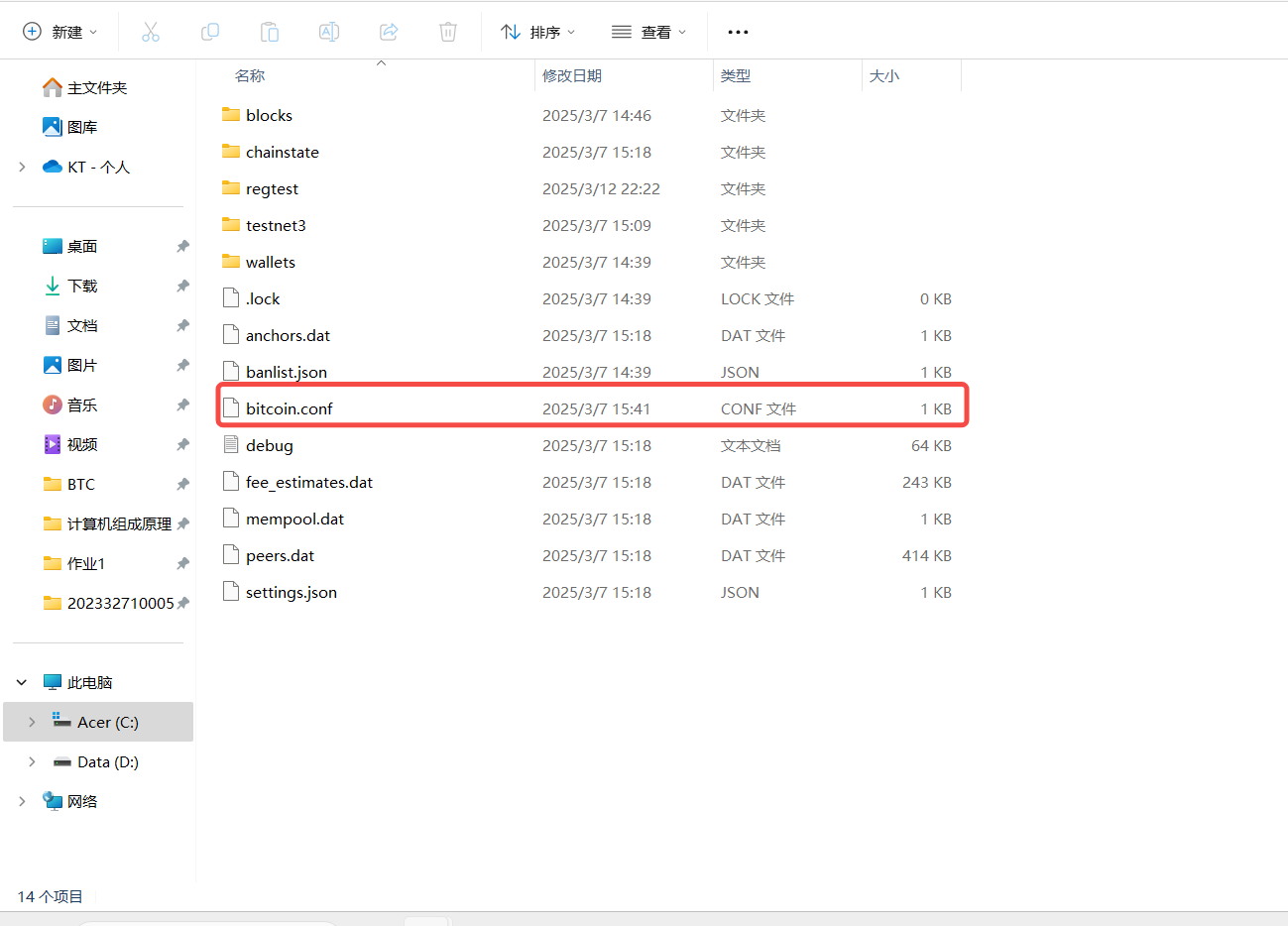
设置好点击保存



把它保存到比特币的数据目录中

命名为bitcoin.conf，然后点击类型为所有文件





**开发网络**

页面建议使用测试网络（testnet）或回归测试模式（regtest）进行开发，以避免主网的风险。更多详情可参考 网络部分。

**启动 Bitcoin Core 节点**

在配置好 bitcoin.conf 文件后，您需要启动 Bitcoin Core 节点以开始运行比特币网络。操作步骤如下：

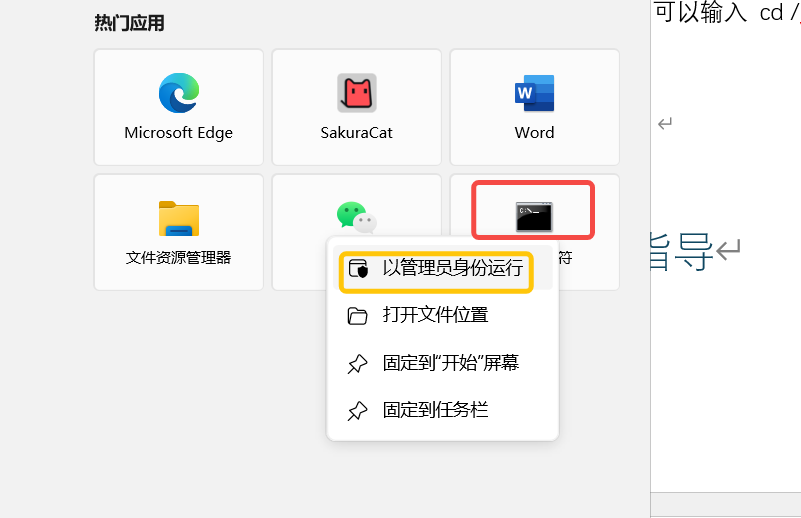
打开终端或命令提示符：

* + Linux/Mac：使用终端。
  + Windows：使用命令提示符或 PowerShell。

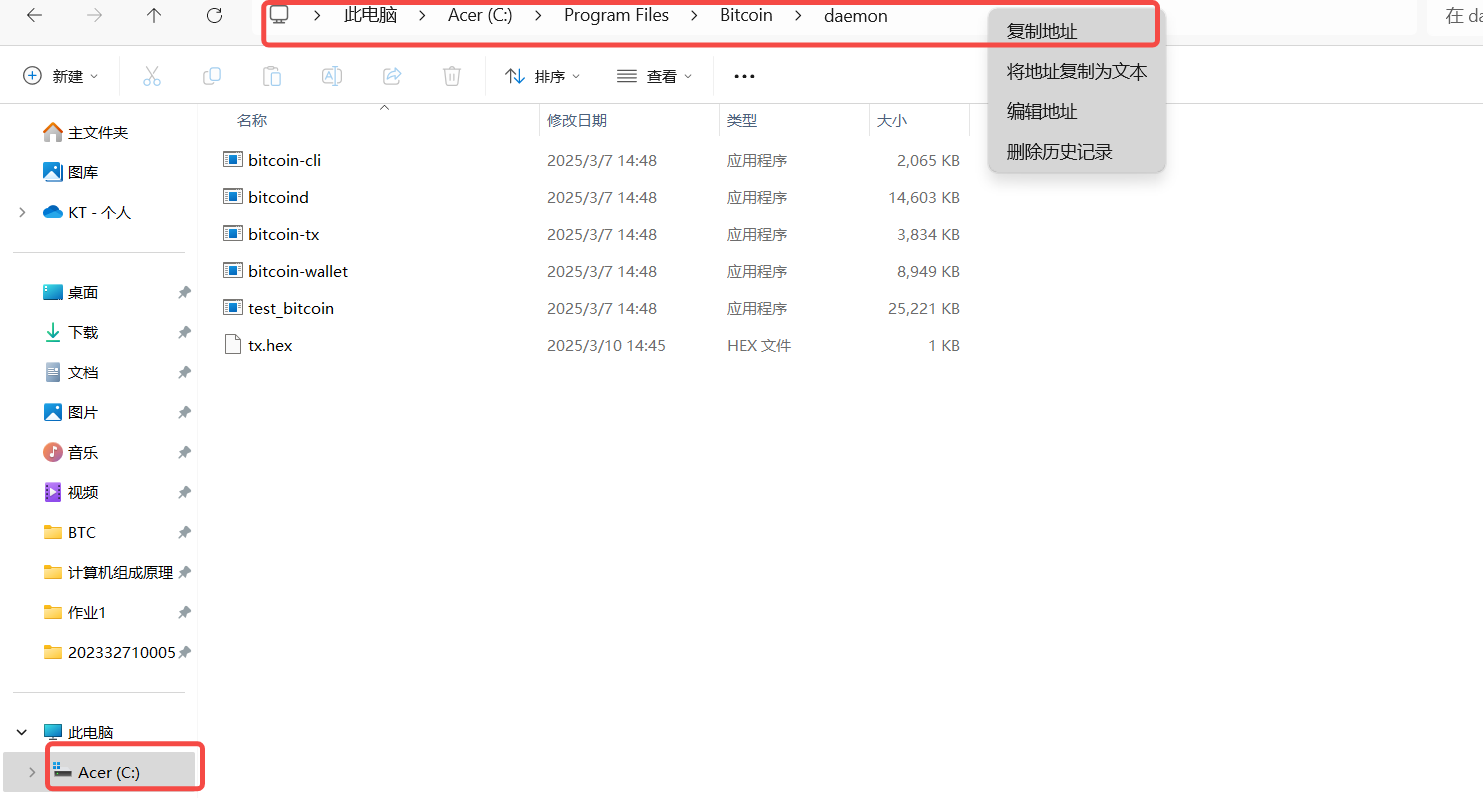
导航到 Bitcoin Core 安装目录：

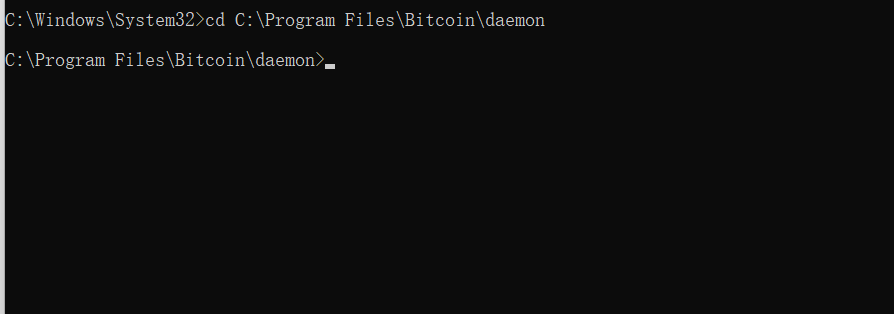
运行启动命令：

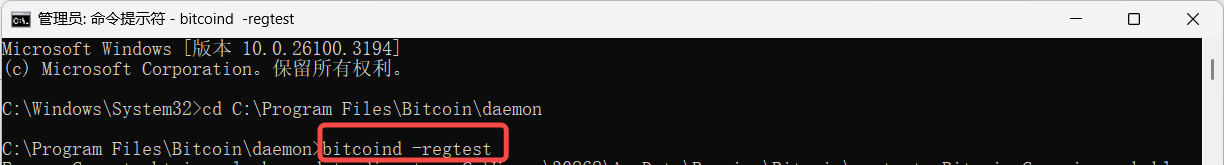
* + 输入 bitcoind 并按回车以启动节点。

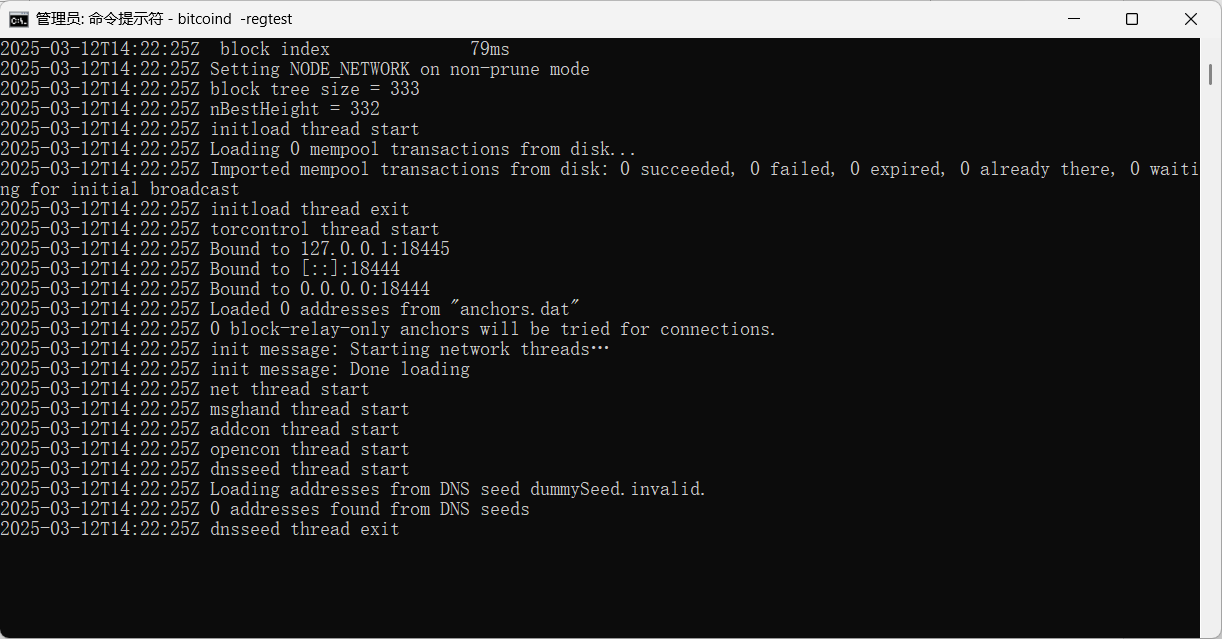


这个获取的地址是Bitcoin的C盘默认安装目录



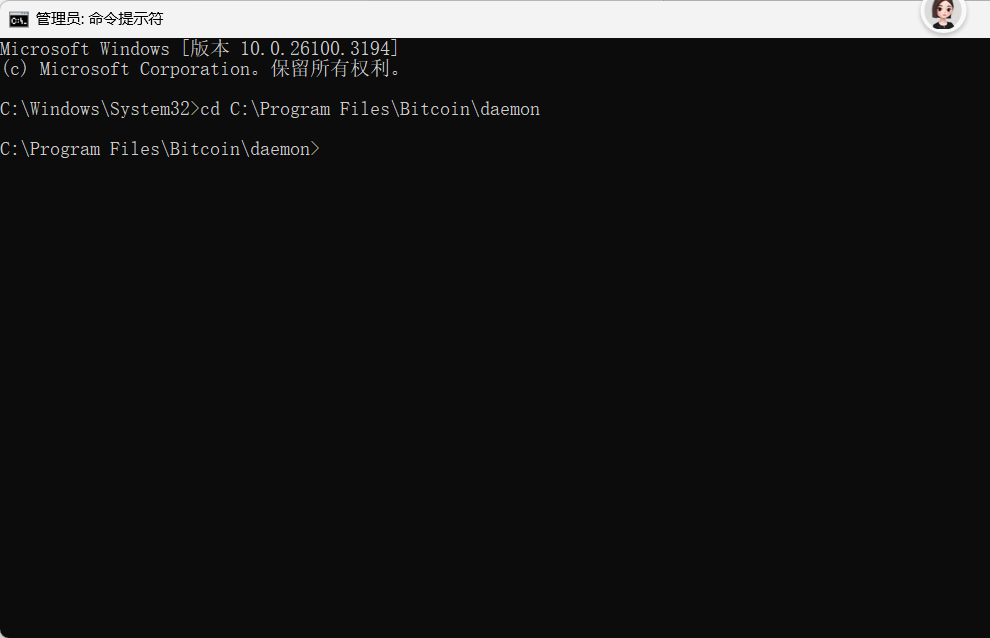






说明已经进入regtest模式

再开个新的终端



## 测试比特币应用程序的详细指导

**关键要点**

* 该页面主要介绍如何使用测试网络（testnet）和回归测试模式（regtest）来测试比特币应用程序。
* Testnet 是一个公共测试网络，使用无实际价值的比特币，适合测试网络交互；regtest 是一个私有的本地网络，开发者可以完全控制，适合开发新应用。
* 页面提供了具体命令，如启动 regtest 和生成测试块，需根据 Bitcoin Core 版本调整。

**测试网络（Testnet）（不推荐使用）**

Testnet 是一个公共测试网络，默认情况下 Bitcoin Core 使用主网（mainnet）。要切换到 testnet，开发者需要在运行 bitcoin-cli、bitcoind 或 bitcoin-qt 时添加 -testnet 参数，或者在 bitcoin.conf 文件中设置 testnet=1。Testnet 的比特币（satoshis）没有实际价值，且放宽了一些限制，如标准交易检查，这使得它适合测试交易和网络交互。开发者可以通过 Piotr Piasecki 的 testnet 水龙头 获取免费的 testnet 比特币，但需注意这是一个公共资源，请勿滥用。

**回归测试模式（Regtest）**

Regtest 模式允许开发者创建私有的本地区块链，与 testnet 具有相同的基本规则，但开发者可以控制块的创建，适合开发新应用。启动 regtest 的命令为 bitcoind -regtest -daemon。

## 比特币交易的教程

**概述**

这个链接 https://developer.bitcoin.org/examples/transactions.html 是一个比特币开发者教程，专注于如何使用Bitcoin Core的RPC接口创建和管理交易。教程假设你在测试网络（regtest）中设置了一个有50 BTC的钱包，确保不会影响真实比特币。

**简单花费**

你可以学习如何用sendtoaddress命令发送10 BTC到一个新地址，并用listunspent查看未花费的交易输出（UTXO），然后生成一个块来确认交易。这是基础的发送方式。

**原始交易**

教程展示如何手动创建交易：用createrawtransaction创建原始交易，用signrawtransaction签名，最后用sendrawtransaction广播。注意费用，比如一个例子中费用是0.0001 BTC。

**复杂原始交易**

这里涉及有两个输入和两个输出的交易，每个输入单独签名，但不发送。总金额是79.9999 BTC，其中一个输出是10 BTC，适合学习复杂场景。

**详细报告**

以下是链接 https://developer.bitcoin.org/examples/transactions.html 内容的全面分析和讲解，旨在为比特币开发者提供创建和管理交易的详细指导。该页面是比特币开发者文档“示例”部分的一部分，内容主要聚焦于使用Bitcoin Core的RPC接口进行交易操作。

**页面背景与目的**

该页面标题为“Transaction Tutorial”，明确为开发者提供一个系统化的教程，教导如何使用Bitcoin Core的RPC接口创建和管理各种类型的比特币交易。教程假设开发者已经在regtest模式下设置了一个测试钱包，初始余额为50 BTC，以确保测试环境安全且无风险。

**简单花费**

* **描述**：这一部分展示如何使用sendtoaddress RPC命令发送比特币到新地址。
* **操作步骤**：
  + 使用sendtoaddress发送10 BTC到一个新地址。
  + 使用listunspent RPC列出未花费的交易输出（UTXO），以查看可用余额。
  + 生成一个块以确认交易，确保交易被网络接受。
* **关键细节**：这一部分是基础操作，适合初学者学习如何通过钱包功能发送比特币。
* **相关命令**：
  + “sendtoaddress” RPC
  + “listunspent” RPC
* **金额**：发送10.00 BTC。

**简单原始交易**

* **描述**：这一部分教导如何手动创建、签名和发送原始交易，适合需要更多控制的开发者。
* **操作步骤**：
  + 使用createrawtransaction创建原始交易，指定输入和输出。
  + 使用signrawtransaction签名交易，确保有足够的私钥。
  + 使用sendrawtransaction广播交易到网络。
* **关键细节**：教程警告开发者注意交易费用，例如一个例子中总金额为49.9999 BTC，费用为0.0001 BTC，强调确保费用足够以被矿工确认。
* **相关命令**：
  + “createrawtransaction” RPC
  + “decoderawtransaction” RPC
  + “sendrawtransaction” RPC
* **金额与费用**：49.9999 BTC发送，0.0001 BTC费用。

**复杂原始交易**

* **描述**：这一部分扩展到更复杂的场景，涉及两个输入和两个输出的交易。
* **操作步骤**：
  + 创建一个交易，包含两个输入和两个输出。
  + 分别对每个输入进行签名。
  + 不发送交易，可能是为了展示如何构建复杂交易。
* **关键细节**：总金额为79.9999 BTC，其中一个输出为10 BTC，适合学习多输入多输出的交易结构。
* **相关命令**：未明确指定额外命令，但可能涉及createrawtransaction和signrawtransaction。
* **金额**：79.9999 BTC总计，10 BTC输出。

## 启动钱包

创建新钱包

1. 创建一个新钱包：

bitcoin-cli --regtest createwallet "mywallet"

"mywallet" 是钱包名称，可以自定义

启动钱包

检查安装路径

* 你提到 C:\Program Files\Bitcoin\daemon，说明 Bitcoin Core 已安装在该目录。
* 确保以下文件存在：
  + bitcoind.exe（后台守护进程）
  + bitcoin-cli.exe（命令行接口）
  + bitcoin-qt.exe（图形界面钱包）

步骤 2：检查现有钱包

1. 列出当前可用的钱包

bitcoin-cli --regtest listwallets

**获取一个新地址： 在 regtest 模式下，使用 bitcoin-cli 获取一个新地址**

bash

bitcoin-cli -regtest getnewaddress

**使用 generatetoaddress 命令**： 使用上一步生成的地址，运行以下命令：

bash

bitcoin-cli -regtest generatetoaddress 101 "your\_address\_here"

验证钱包功能

1. 检查余额

bitcoin-cli --regtest getbalance

## Simple Spending

1、创建新地址

bitcoin-cli --regtest getnewaddress "" "legacy"

示例输出：mfiUzJuW9Lcwng6tdaJ64U4wzwWaWCbsj1，保存为 $NEW\_ADDRESS

2、发送 10 BTC 到新地址

bitcoin-cli --regtest sendtoaddress $NEW\_ADDRESS 10.00

（注意：先设置交易费用

由于 regtest 模式下费用估算可能失败，需手动设置交易费用：

运行：bitcoin-cli --regtest -rpcwallet=（+钱包名） settxfee 0.0001，设置每千字节 0.0001 BTC 的费用。）



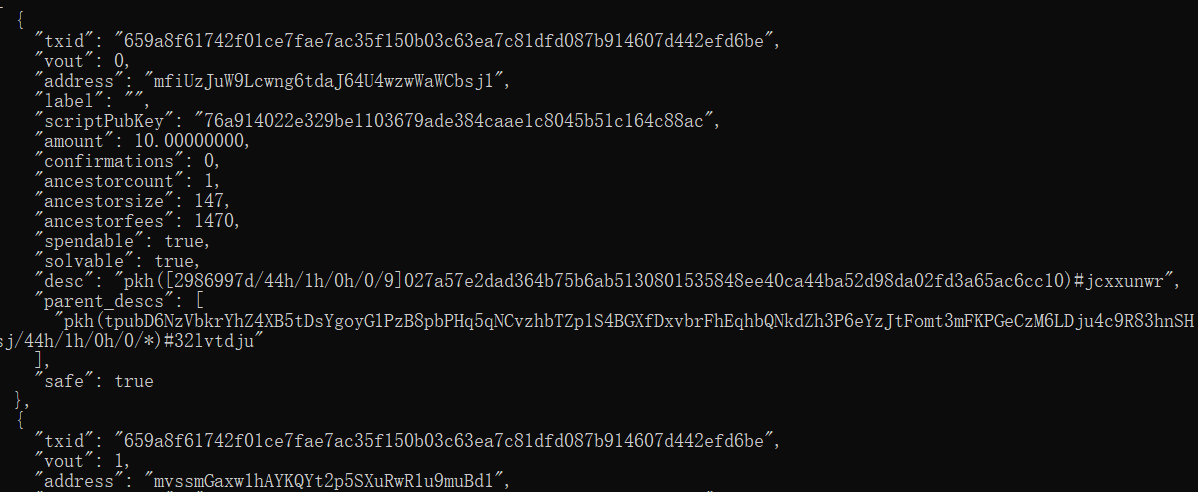
预期输出为交易 ID（txid）

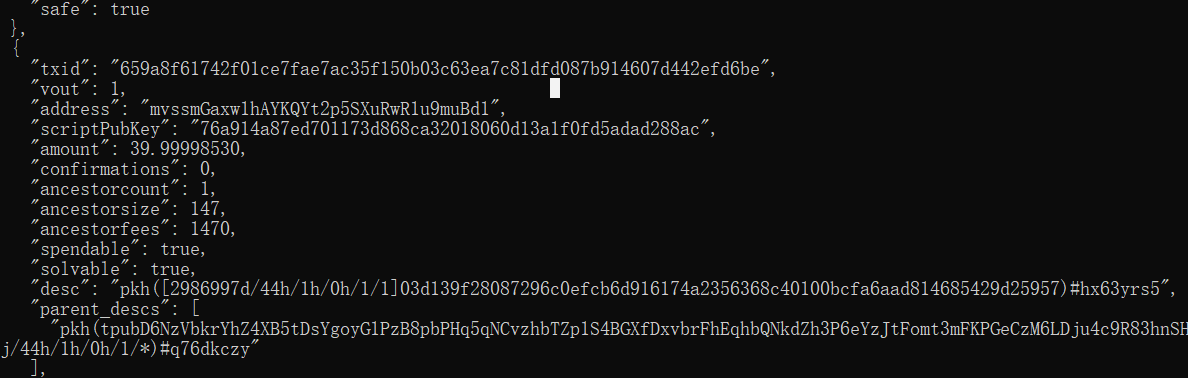
659a8f61742f01ce7fae7ac35f150b03c63ea7c81dfd087b914607d442efd6be

3、列出 UTXO

运行：bitcoin-cli --regtest listunspent 0 查看未确认的 UTXO，显示两个输出

一个是 10 BTC 到 mvbnrCX3bg1cDRUu8pkecrvP6vQkSLDSou，另一个是找零 40 BTC





4、生成区块确认

运行：bitcoin-cli --regtest -generate 1 确认交易，并清理变量。



## Simple Raw Transaction

1. 列出未花费的交易输出（UTXOs）  
   使用 listunspent 命令查看可用的 UTXOs

bitcoin-cli listunspent

这会列出你的可用 UTXOs，包括 txid 和 vout。

1. 选择一个 UTXO  
   从列表中选择一个 UTXO，记录其 txid 和 vout，例如：

UTXO\_TXID=7c7b8e46c0f4e76d13fb543822c242ac7249cb9f766ff4b4fa681917a2889e82

UTXO\_VOUT=0

1. 获取新地址  
   使用 getnewaddress 命令生成一个新地址作为输出地址，例如：

bitcoin-cli -regtest getnewaddress

地址：bcrt1q0rhkwzlwedh880zzyyld5mm7q6fnx0ptx7uwmt

1. 创建原始交易  
   使用 createrawtransaction 命令创建交易，指定输入和输出金额，例如发送 49.9999 BTC：

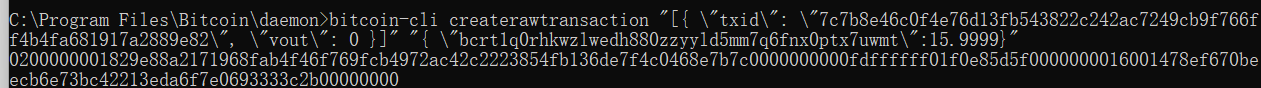
Bash

bitcoin-cli createrawtransaction "[{ \"txid\": \"$UTXO\_TXID\", \"vout\": $UTXO\_VOUT}]" "{ \"$NEW\_ADDRESS\":15.9999}"

带入内容

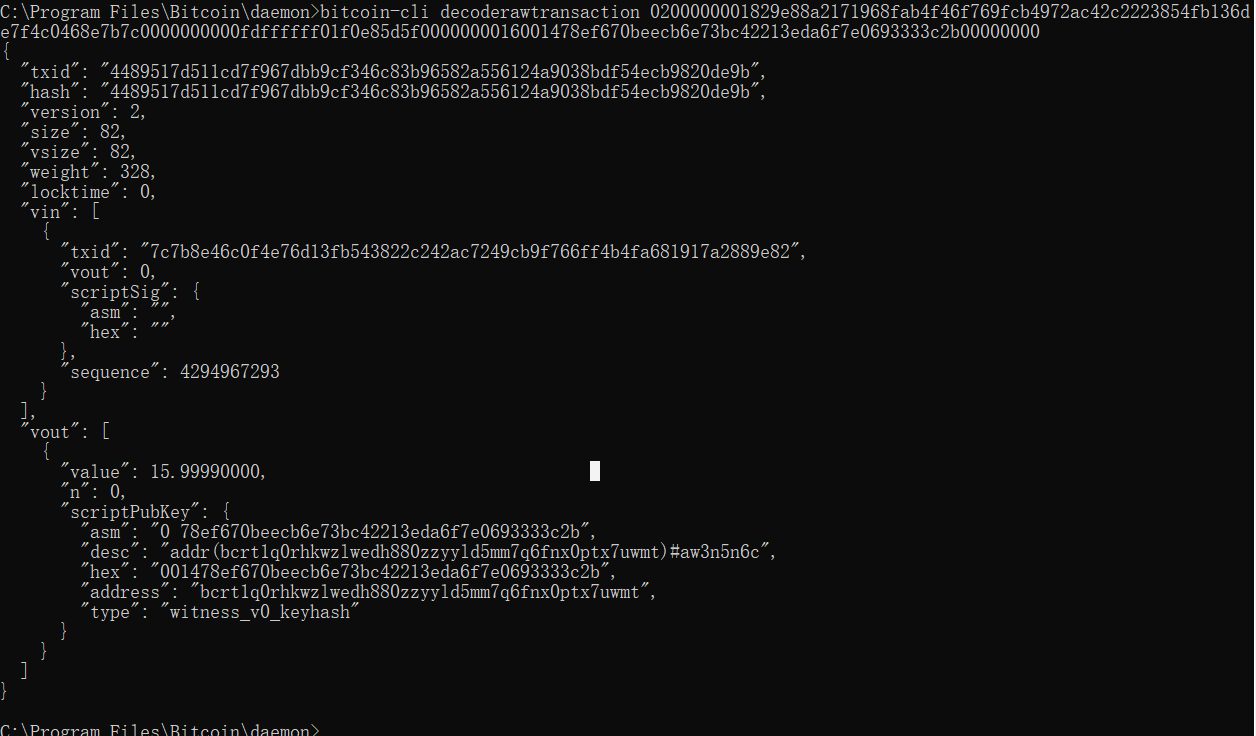
bitcoin-cli createrawtransaction "[{ \"txid\": \"7c7b8e46c0f4e76d13fb543822c242ac7249cb9f766ff4b4fa681917a2889e82\", \"vout\": 0 }]" "{ \"bcrt1q0rhkwzlwedh880zzyyld5mm7q6fnx0ptx7uwmt\":15.9999}"

RAW\_TX：  
0200000001829e88a2171968fab4f46f769fcb4972ac42c2223854fb136de7f4c0468e7b7c0000000000fdffffff01f0e85d5f0000000016001478ef670beecb6e73bc42213eda6f7e0693333c2b00000000



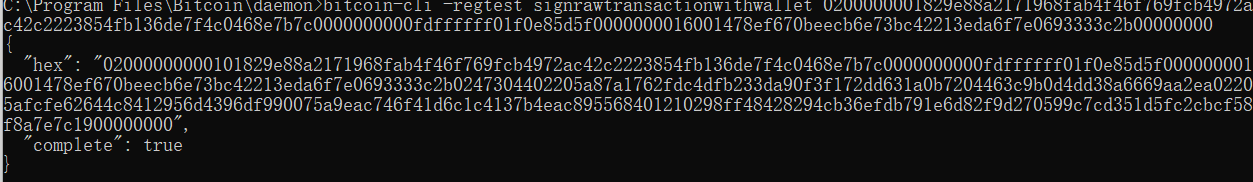
1. 解码并验证交易  
   使用 decoderawtransaction 检查交易详情：

bitcoin-cli decoderawtransaction +RAW\_TX



1. 签名交易  
   使用 signrawtransaction 签名交易：

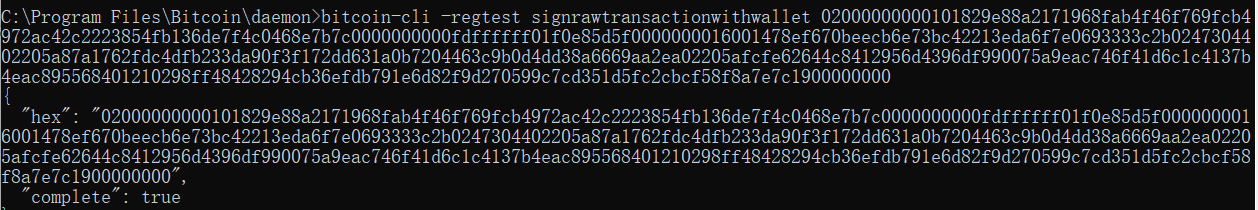
bitcoin-cli -regtest signrawtransactionwithwallet+ $RAW\_TX



SIGNED\_RAW\_TX：02000000000101829e88a2171968fab4f46f769fcb4972ac42c2223854fb136de7f4c0468e7b7c0000000000fdffffff01f0e85d5f0000000016001478ef670beecb6e73bc42213eda6f7e0693333c2b0247304402205a87a1762fdc4dfb233da90f3f172dd631a0b7204463c9b0d4dd38a6669aa2ea02205afcfe62644c8412956d4396df990075a9eac746f41d6c1c4137b4eac895568401210298ff48428294cb36efdb791e6d82f9d270599c7cd351d5fc2cbcf58f8a7e7c1900000000

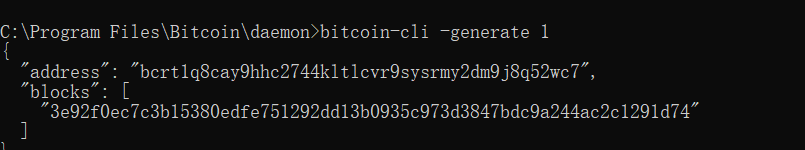
1. 广播交易  
   使用 sendrawtransaction 广播已签名的交易：

bitcoin-cli -regtest signrawtransactionwithwallet+ SIGNED\_RAW\_TX：



1. 确认交易（在 regtest 模式下）  
   如果使用 regtest 模式，生成一个新区块确认交易：

bitcoin-cli -generate 1



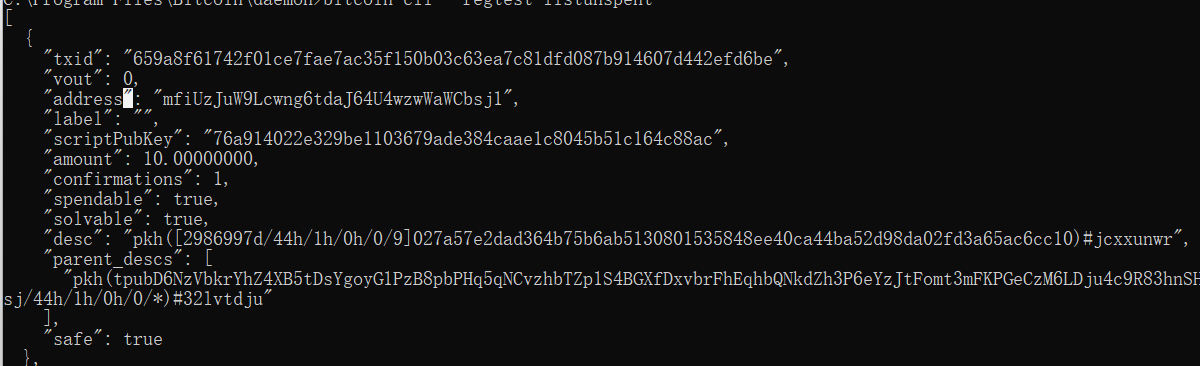
确保交易费用合理，费用是输入减去输出的差额。

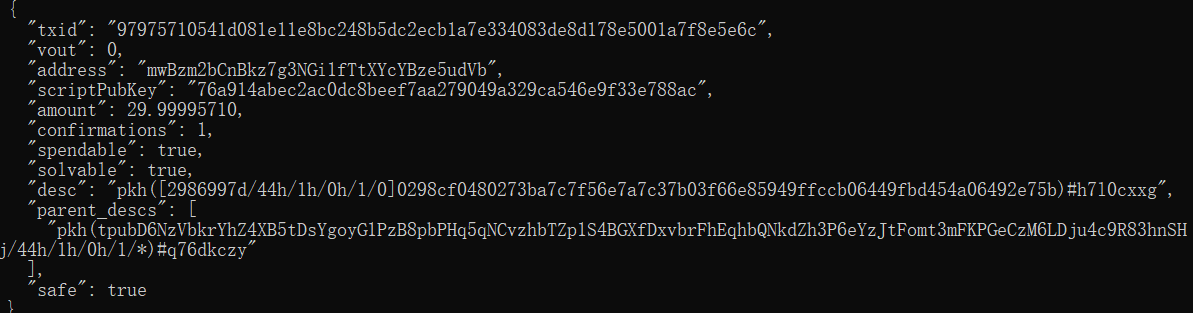
* 如果不添加找零输出，可能会导致高费用，例如输入 50 BTC，输出 10 BTC，费用可能是 40 BTC。
* 操作原始交易时需小心，错误可能导致资金损失。

## Complex Raw Transaction

1、选择 UTXO

列出 UTXO：bitcoin-cli --regtest listunspent，选择两个，例如：





UTXO1\_TXID= 659a8f61742f01ce7fae7ac35f150b03c63ea7c81dfd087b914607d442efd6be

UTXO1\_VOUT=0

UTXO1\_ADDRESS= mfiUzJuW9Lcwng6tdaJ64U4wzwWaWCbsj1

UTXO2\_TXID= 97975710541d081e11e8bc248b5dc2ecb1a7e334083de8d178e5001a7f8e5e6c

UTXO2\_VOUT=0

UTXO2\_ADDRESS= mwBzm2bCnBkz7g3NGi1fTtXYcYBze5udVb

2获取新地址

运行：bitcoin-cli --regtest getnewaddress 两次，获取

NEW\_ADDRESS1：bcrt1qv3m0u2zu7efjq773rv4wymvnjcw2nkt65pmc5h

和

NEW\_ADDRESS2：bcrt1qgd72lt8th8xr5c8wlt7mtxjv0c5xk2khktpnct

3创建原始交易

运行：黄色内容输出保存为 $RAW\_TX。

即：（直接抄下面这个格式，把相关内容更换）

bitcoin-cli --regtest -rpcwallet=mywallet createrawtransaction "[{ \"txid\": \"659a8f61742f01ce7fae7ac35f150b03c63ea7c81dfd087b914607d442efd6be\", \"vout\": 0 }, { \"txid\": \"97975710541d081e11e8bc248b5dc2ecb1a7e334083de8d178e5001a7f8e5e6c\", \"vout\": 0 }]" "{ \"bcrt1qv3m0u2zu7efjq773rv4wymvnjcw2nkt65pmc5h\":5, \"bcrt1qgd72lt8th8xr5c8wlt7mtxjv0c5xk2khktpnct\": 5 }"

检查 UTXO：bitcoin-cli --regtest -rpcwallet=mywallet listunspent，确认 txid "659a8f61742f01ce7fae7ac35f150b03c63ea7c81dfd087b914607d442efd6be" 和 "97975710541d081e11e8bc248b5dc2ecb1a7e334083de8d178e5001a7f8e5e6c" 的 UTXO 可花费，且金额足够（总和需大于 27.9999 BTC 加上交易费）。

保存输出$RAW\_TX，如

0200000002bed6ef42d40746917b08fd1dc8a73ec6030b155fc37aae7fce012f74618f9a650000000000fdffffff6c5e8e7f1a00e578d1e83d0834e3a7b1ecc25d8b24bce8111e081d54105797970000000000fdffffff02f06c3577000000001600146476fe285cf653207bd11b2ae26d93961ca9d97a0008af2f00000000160014437cafacebb9cc3a60eefafdb59a4c7e286b2ad700000000

直接进行签名的操作：  
无需获取私钥：对于复杂原始交易，你通常不需要手动获取私钥。可以使用 signrawtransactionwithwallet 命令直接用钱包的私钥签名交易，这适用于描述符钱包。

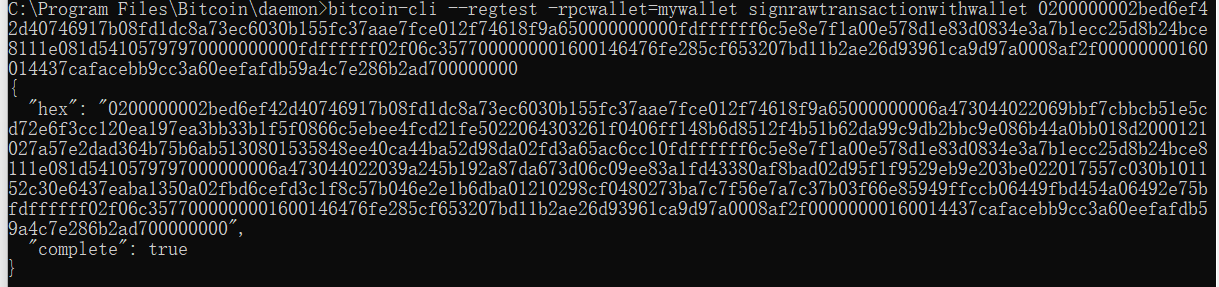
* 检查钱包类型：运行 bitcoin-cli --regtest -rpcwallet=（+钱包名） getwalletinfo 查看钱包类型。如果是描述符钱包（"descriptors": true），

下一步操作

1. 创建原始交易：bitcoin-cli --regtest -rpcwallet=mywallet createrawtransaction ...。
2. 签名：bitcoin-cli --regtest -rpcwallet=mywallet signrawtransactionwithwallet RAW\_TX。

bitcoin-cli --regtest -rpcwallet=（+钱包名） signrawtransactionwithwallet

0200000002bed6ef42d40746917b08fd1dc8a73ec6030b155fc37aae7fce012f74618f9a650000000000fdffffff6c5e8e7f1a00e578d1e83d0834e3a7b1ecc25d8b24bce8111e081d54105797970000000000fdffffff020065cd1d000000001600146476fe285cf653207bd11b2ae26d93961ca9d97a0065cd1d00000000160014437cafacebb9cc3a60eefafdb59a4c7e286b2ad700000000



$PARTLY\_SIGNED\_RAW\_TX ：

广播：bitcoin-cli -regtest signrawtransactionwithwallet $PARTLY\_SIGNED\_RAW\_TX



广播成功