السلام عليكم

إياد أحمد أحمد

تقرير توضيحي: تفاعل مع شبكة Ethereum واستعراض رصيد USDT

التقنيات المستخدمة:

1. TypeScript: اللغة المستخدمة لتطوير الكود. توفر TypeScript تحسينات قوية للأنواع وأدوات لفحص الأخطاء قبل وقت التشغيل.

2. Web3.js: مكتبة JavaScript للتفاعل مع شبكة Ethereum. توفر واجهة للتفاعل مع عقود الذكاء الاصطناعي والعملات والمعاملات الذكية.

3. Jest: إطار اختبار JavaScript الشائع. يستخدم لكتابة وتشغيل اختبارات وحدية للتحقق من صحة وظيفة الكود.

الدوال المستخدمة:

1. getUSDTBalance(address: string, contractABI: AbiItem[]):
تستخدم للحصول على رصيد USDT لعنوان محدد. تقوم الدالة بإنشاء مثيل لعقد USDT باستخدام web3.eth.Contract وتستدعي دالة balanceOf على العقد لاسترداد الرصيد.

2. getLastBlockNumber():

تستخدم للحصول على رقم آخر كتلة في شبكة Ethereum الرئيسية. تقوم الدالة باستدعاء web3.eth.getBlockNumber لاسترداد رقم الكتلة.

تنفيذ الكود:

1. تم تنفيذ الكود في ملف "task1.ts". يتم استيراد مكتبات "web3" و "AbiItem" وتكوين مثيل من "Web3" للتفاعل مع شبكة Ethereum.

2. تم تنفيذ دالتي getUSDTBalance و getLastBlockNumber للحصول على رصيد USDT ورقم آخر كتلة.

3. يتم تنفيذ الكود في ملف "test.ts" لاختبار وحدات الكود. يتم استخدام إطار الاختبارات Jest لإنشاء اختبارات لكل من الدوال getLastBlockNumber و getUSDTBalance.

يبدأ تنفيذ الكود من دالة main في "task1.ts". في هذه الدالة، يتم استدعاء دالة getUSDTBalance مع عنوان Ethereum ومصفوفة ABI لعقد USDT. ثم يتم استدعاء دالة getLastBlockNumber للحصول على رقم آخر كتلة. يتم طباعة الرصيد ورقم الكتلة على الإخراج القياسي.

لتنفيذ الكود، يجب كتابة المعلومات المحددة مثل معرف مشروع Infura الخاص بك وعناوين Ethereum الفعلية في "task1.ts" و "test.ts".

يمكن تشغيل الاختبارات باستخدام Jest عن طريق تشغيل الأمر jest في مجلد المشروع. ستقوم Jest بتشغيل الاختبارات وعرض النتائج على الإخراج القياسي.