# MATRIX 智慧合约

# MATRIX 智慧合约支持多语言编程，支持离线形式智慧合约安全审计和形式化验证工具；

# MATRIX在线钱包支持智慧合约的部署，并内置一键token、博彩等常用合约；

# 可以执行代码中的缺陷检测和形式化验证，并对风险结果友好提示；支持根据训练结果得到错误推断；

# 如何在我们的在线钱包上部署和使用合约

# 通过MATRIX在线钱包，可以通过交易的形式来发布和执行合约，从而避免了繁琐的步骤

# 第一步：使用 remix 对写好的合约代码文件进行编译，把编译结果保存到本地文件

# 

# 第二步：查看编译结果，如下所示

# 

# 第三步：登录在线钱包，进入“合同”，点击‘部署合约’

# 

# 第四步：将上面Bytecode中的object结果拷贝到如下Byte Code中，首部添加0x，因为是合约交易附带的数据很多，Gas Limit尽量设置大一些

# 

# 第五步：执行和部署，在执行结果中记下 contract address（合约地址）；也可以在网页钱包源码中添加智慧合约（添加前做好备份），nginx重启后，合约就会出现在合约列表中

# 第六步：使用部署好的智慧合约：点击 Interact with Contract

# 第七步：选择一个合约，点击access，例如：

# 

# 第八步：选择一个函数，例如getDepositList，来获取哪些节点参与了抵押（展示地址）

# 

# 第九步：复制一个地址，然后选择另外一个函数：getDepositInfo

# 

# 如何通过第三方工具truffle 框架进行智慧合约的开发、部署和编译。

# 什么是TRUFFLE

# Truffle是针对Solidity语言的一套开发框架，本身基于Javascript。Truffle整合了开发环境、测试框架和资产管道(pipeline)，使用Truffle可以得到：

# 内置智能合约编译、链接、部署和二进制字节码管理。

# 针对快速迭代开发的自动化合约测试。

# 可脚本化，可扩展的部署和迁移框架。

# 网络管理，用于部署到任意数量的公共和私有网络。

# 使用ETHPM和NPM进行包安装管理。

# 用于直接合约通信的交互式控制台。

# 支持持续集成的可配置构建管道。

# 外部脚本运行程序可以在Truffle环境中执行脚本。

# 提供了合约抽象接口，可以直接通过var instance = Storage.deployed();拿到合约对象后，在Javascript中直接操作对应的合约函数。

# 原理是使用了基于web3.js封装的Ether Pudding工具包。简化开发流程。

# 提供了控制台，使用框架构建后，可以直接在命令行调用输出结果，可极大方便开发调试。

# 利用 truffle 框架编写、编译和部署智慧合约

# 第一步：安装truffle框架

# 第三步： 创建并初始化项目

* $ mkdir myproject—新建一个 myproject 文件夹
* $ cd myproject—进入
* $ truffle init—执行初始化

# 初始化后目录结构类似于：

# 

# 其中，contracts/ - 存放我们编写的合约。

# migrations/ - 存放迁移部署脚本。

# test/ - 存放合约测试脚本

# truffle.js - Truffle的配置文件

# truffle init会给我们创建一个名叫MetaCoin的代币应用。我们将这个默认的应用删除，来编写自己的合约（有些版本，自动删除了metacoin这个demo）

# 第四步：创建合约

# 在文件夹contracts上右键，选择新建文件，输入文件名：storage.sol，写入以下内容：

pragmasolidity^0.4.8;

contractStorage{

uint256storedData;

functionset(uint256data) {

storedData=data;

}

functionget() constantreturns(uint256) {

returnstoredData;

}

}

# 第五步：编译合约

# 使用truffle compile命令编译刚刚完成的合约

$ truffle compile

Compiling ./contracts/Migrations.sol...

Compiling ./contracts/Storage.sol...

Writing artifacts to ./build/contracts

# 从控制台的输出中，我们可以看到合约编译后的文件（artifacts）会写入./build/contracts目录中，这些合约编译后的文件对于Truffle框架能否正常工作至关重要。请不要手动修改这些文件，因为即使修改了，再次执行编译命令时又会被覆盖掉。

# Truffle默认只编译自上次编译后被修改过的合约，目的是为了减少不必要的编译。如果你想编译全部合约 ，可以使用--all选项。

# 合约编译完成后，我们需要部署合约到区块链上。

# 第六步：安装和启动客户端gman（要打开一个新的终端操作）

# 第七步：执行迁移命令：truffle migrate

# 智慧合约交互

# 一般来说，交互有读数据（调用）和写数据（交易）两种。通过交易调用合约函数时，无法立即获得返回值，因为交易还没有发送、打包和执行。而调用只是单纯读取数据，不会做任何更改，通过调用执行合约函数时，可以立即获得返回值

# 合约抽象

# 简言之，合约抽象就是帮我们封装好代码，可以让你和合约之间的交互变得简单，不必关心合约调用细节。Truffle通过truffle-contract模块来使用自己的合约抽象。合约抽象中的函数和我们合约中的函数是一样的。

# 为了使用合约抽象和合约交互，我们需要通过npm安装truffle-contract模块