# 搭建一个区块DApp投票系统

| | |
|---|---|
| 产品名称 | 搭建一个区块DApp投票系统 |
| 公司名称 | 河南漫云科技有限公司 |
| 价格 | 1000.00/件 |
| 规格参数 | 漫云科技:搭建一个区块DApp投票系统 |
| 公司地址 | 郑东新区升龙广场3号楼A座3202 |
| 联系电话 | 13103827627 13103827627 |

# 产品详情

第一节 概述

对于初学者，需要了解以太坊kaifa相关的基本概念，另外就是如何构建一个基于以太坊的完整去中心化应用例如一个qukuailian投票系统。

通过学习，你将掌握：

以太坊qukuailian的基本知识

kaifa和部署以太坊合约所需的软件环境

使用语言（solidity）编写以太坊合约

使用Truffle框架kaifa分布式应用

使用控制台或网页与合约进行交互

以太坊kaifa的前序知识要求

为了顺利完成本课程，好对以下技术已经有一些基本了解：

一种面向对象的kaifa语言，例如：Python，Ruby，Java...

前端kaifa语言：HTML/CSS/JavaScript

Linux命令行的使用

数据库的基本概念

第二节简介

我们将会构建一个去中心化的（Decentralized）投票应用。利用这个投票应用，用户可以在不可信（trustless）的分布环境中对特定候选人投票，每次投票都会被记录在qukuailian上：

所谓去中心化应用（DApp：DcentralizedApplication），就是一个不存在中心服务器的应用。在网络中成百上千的电脑上，都可以运行该应用的副本，这使得它几乎不可能出现宕机的情况。

基于qukuailian的投票是完全去中心化的，因此无须任何中心化机构的存在。

第三节kaifa迭代

涵盖应用kaifa的整个过程，我们将通过三次迭代来渐进地引入qukuailian应用kaifa所涉及的相关概念、语言和工具：

Vanilla：在个迭代周期，我们不借助任何kaifa框架，而仅仅使用NodeJS来进行应用kaifa，这有助于我们更好地理解qukuailian应用的核心理念。

Truffle：在第二个迭代周期，我们将使用流行的去中心化应用kaifa框架Truffle进行kaifa。使用kaifa框架有助于我们提高kaifa效率。

Token：在第三个迭代周期，我们将为投票应用引入代币（Token）——现在大家都改口称之为通证了——都是ICO惹的祸。代币是公链上bukehuoque的激励机制，也是qukuailian应用区别于传统的中心化应用的另一个显著特征。

为什么选择投票应用？

之所以选择投票作为我们的个qukuailian应用，是因为集体决策——尤其是投票机制——是以太坊的一个核心的价值主张。

另一个原因在于，投票是很多复杂的去中心化应用的基础构件，所以我们选择了投票应用作为学习qukuailian应用kaifa的个项目。

第四节初识qukuailian

如果你熟悉关系型数据库，就应该知道一张数据表里可以包含很多行数据记录。例如，下面的数据表中包含了6条交易记录：

本质上，qukuailian首先就是一个分布式（Distributed）数据库，这个数据库维护了一个不断增长的记录列表。现在，让我们对数据进行批量（batch）存储，比如每批100行，并将各存储批次连接起来，是不是就像一条链？

移除点击此处添加图片说明文字

在qukuailian里，多个数据记录组成的批次就被称为块（block），块里的每一行数据记录就被称为交易（transaction）：

开始的那个块，通常被称为创世块（genesisblock），它不指向任何其他块。

不可篡改性

qukuailian的一个显著特点是，数据一旦写入链中，就不可篡改重写。

在传统的关系型数据库中，你可以很容易地更新一条数据记录。但是，在qukuailian中，一旦数据写入就无法再更新了——因此，qukuailian是一直增长的。

那么，qukuailian是如何实现数据的不可篡改特性？

这首先得益于哈希（Hash）函数——如果你还没接触过哈希函数，不妨将它视为一个数字指纹的计算函数：输入任意长度的内容，输出定长的码流（指纹）。哈希函数的一个重要特性就是，输入的任何一点微小变化，都会导致输出的改变。因此可以将哈希值作为内容的指纹来使用。你可以点击这里进一步了解哈希函数。

由于qukuailian里的每个块都存储有前一个块内容的哈希值，因此如果有任何块的内容被篡改，被篡改的块之后所有块的哈希值也会随之改变，这样我们就很容易检测出qukuailian的各块是否被篡改了。

去中心化的挑战

一旦完全去中心化，在网络上就会存在大量的qukuailian副本（即：全节点），很多事情都会变得比之前中心化应用环境复杂的多，例如：

如何保证所有副本都已同步到新状态？

如何保证所有交易都被广播到所有运行和维护qukuailian副本的节点计算机上？

如何防止恶意参与者篡改qukuailian

在接下来的课程中，通过与经典的C/S架构的对比，我们将逐步理解去中心化应用的核心思路，并掌握如何构建以太坊上的去中心化应用。

第五节C/S架构以服务器为中心

理解去中心化应用架构的好方法，就是将它与熟悉的Client/Server架构进行对比。如果你是一个web kaifa者，应该对下图很了解，这是一个典型的Client/Server架构：

一个典型web应用的服务端通常由Java，Ruby，Python等等语言实现。前端代码由HTML/CSS/JavaScript实现。然后将整个应用托管在云端，比如AWS、GoogleCloudPlatform、Heroku....，或者放在你租用的一个VPS主机上。用户通过客户端（Client）与web应用（Server）进行交互。典型的客户端包括浏览器、命令行工具（curl、wget等）、或者是API访问代码。注意在这种架构中，总是存在一个（或一组）中心化的web服务器，所有的客户端都需要与这一（组）服务器进行交互。当一个客户端向服务器发出请求时，服务器处理该请求，与数据库/缓存进行交互，读/写/更新数据库，然后向客户端返回响应。

这是我们熟悉的中心化架构。在下一节，我们将会看到基于qukuailian的去中心化架构的一些显著区别。

第六节去中心化架构——彼此平等的节点

下图给出了基于以太坊的去中心化应用架构：

你应该已经注意到，每个客户端（浏览器）都是与各自的节点应用实例进行交互，而不是向一个中

心化的服务器请求服务。

在一个理想的去中心化环境中，每个想要跟DApp交互的人，都需要在他们的计算机或手机上面运行一个的完整qukuailian节点——简言之，每个人都运行一个全节点。这意味着，在能够真正使用一个去中心化应用之前，用户不得不下载整个qukuailian。

不过我们并非生活在一个乌托邦里，期待每个用户都先运行一个全节点，然后再使用你的应用是不现实的。但是去中心化背后的核心思想，就是不依赖于中心化的服务器。所以，qukuailian社区已经出现了一些解决方案，例如提供公共qukuailian节点的Infura,以及浏览器插件Metamask等。通过这些方案，你就不需要花费大量的硬盘、内存和时间去下载并运行完整的qukuailian节点，同时也可以利用去中心化的优点。我们将会以后的课程中对这些解决方案分别进行评测。

第七节以太坊——世界计算机

以太坊是一种qukuailian的实现。在以太坊网络中，众多的节点彼此连接，构成了以太坊网络：

以太坊节点软件提供两个核心功能：数据存储、合约代码执行。

在每个以太坊全节点中，都保存有完整的qukuailian数据。以太坊不仅将交易数据保存在链上，编译后的合约代码同样也保存在链上。

以太坊全节点中，同时还提供了一个虚拟机来执行合约代码。

交易数据

以太坊中每笔交易都存储在qukuailian上。当你部署合约时，一次部署就是一笔交易。当你为候选者投票时，一次投票又是另一笔交易。所有的这些交易都是公开的，每个人都可以看到并进行验证。这个数据永远也无法篡改。

为了确保网络中的所有节点都有着同一份数据拷贝，并且没有向数据库中写入任何无效数据，以太坊目前使用工作量证明（POW：ProofOfWork）算法来保证网络安全，即通过矿工wakuang（Mining）来达成共识（Consensus）——将数据同步到所有节点。

工作量证明不是达成共识的唯一算法，wakuang也不是qukuailian的唯一选择。现在，我们只需要了解，共识是指各节点的数据实现了一致，POW只是众多用于建立共识的算法中的一种，这种算法需要通过矿工的wakuang来实现非可信环境下的可信交易。共识是目的，POW是手段。

合约代码

以太坊不仅仅在链上存储交易数据，它还可以在链上存储合约代码。

在数据库层面，qukuailian的作用就是存储交易数据。那么给候选者投票、或者检索投票结果的逻辑放在哪儿呢？在以太坊的世界里，你可以使用

Solidity

语言来编写业务逻辑/应用代码（也就是合约：Contract），然后将合约代码编译为以太坊字节码，并将字节码部署到qukuailian上：

编写合约代码也可以使用其他的语言，不过Solidity是到目前为止流行的选择。

以太坊虚拟机

以太坊qukuailian不仅存储数据和代码，每个节点中还包含一个虚拟机（EVM：EthereumVirtualMachine）来执行合约代码——听起来就像计算机操作系统。

事实上，这一点是以太坊区别于bitebi（Bitcoin）的核心的一点：虚拟机的存在使qukuailian迈入了2.0时代，也让qukuailian次成为应用kaifa者友好的平台。

JSkaifa库

为了便于构建基于web的DApp，以太坊还提供了一个非常方便的JavaScript库web3.js，它封装了以太坊节点的API协议，从而让kaifa者可以轻松地连接到qukuailian节点而不必编写繁琐的RPC协议包。所以，我们可以在常用的JS框架（比如reactjs、angularjs等)中直接引入该库来构建去中心化应用：