以太坊开发架构

```
$(document).ready(function() {
    $.ajax({
      url: '127.0.0.1:1337',          //不知道这个url到底应该填什么
      dataType: "jsonp",
      data: '{"data": "TEST"}',
      type: 'POST',
      jsonpCallback: 'callback',
      success: function (data) {
         var ret = jQuery.parseJSON(data);
         console.log('Success: '+ret.msg)
      },
      error: function (xhr, status, error) {
         console.log('Error: ' + error.message);},});});
```

HTML

http://ip(localhost):8080

Node.js

Express.js

Web3.js

Solidity智能合约

```
pragma solidity ^0.4.0; contract Calc{
 /*区块链存储*/ uint count;
/*执行会写入数据，所以需要`transaction`的方式执行。*/
function add(uint a, uint b) returns(uint){ count++; return a + b; }
/*执行不会写入数据，所以允许`call`的方式执行。*/
function getCount() constant returns (uint){ return count; } }
```

```
var express = require('express');

var routes = require('./routes/index');
var users = require('./routes/users');

var app = express();

app.use('/', routes);
app.post('/vote', function (req, res) { myContract.vote.sendTransaction(1, 2,{
                from: deployeAddr
            }); res.send('vote success'); })
app.get('/result', function (req, res) { var re = myContract.result.call(1, 2,{
                from: deployeAddr
            });
res.send(re); })
```

```
var Web3 = require("web3");
// 创建web3对象
var web3 = new Web3();

let source = "pragma solidity ^0.4.0;contract Calc{
     /*区块链存储*/  uint count;
     /*执行会写入数据，所以需要`transaction`的方式执行。*/
      function add(uint a, uint b) returns(uint){    count++;    return a + b;  }
     /*执行不会写入数据，所以允许`call`的方式执行。*/
       function getCount() constant returns (uint){    return count;  }}";
let calcCompiled = web3.eth.compile.solidity(source);

//得到合约对象
let abiDefinition = calcCompiled["info"]["abiDefinition"];
let calcContract = web3.eth.contract(abiDefinition);

//异步方式新部署
let myContractReturned = calcContract.new({
    data: deployCode,
    from: deployeAddr
}, function(err, myContract) {
    if (!err) {…}
})
```

```
server = http.createServer( app );
server.listen( port, function () {
    console.log( 'Express server listening on port ' + port );
});
```

```
// 连接到以太坊节点
web3.setProvider(new Web3.providers.HttpProvider("http://localhost:8545"));

// 合约ABI
var abi = [{"constant":false,"inputs":[{"name":"receiver","type":"address"},{"name":"amount","type":"uint256"}],"name":"sendCoin","outputs":
[{"name":"sufficient","type":"bool"}],"payable":false,"type":"function"},{"constant":false,"inputs":
[{"name":"addr","type":"address"}],"name":"getBalance","outputs":[{"name":"","type":"uint256"}],"payable":false,"type":"function"},
{"inputs":[],"payable":false,"type":"constructor"},{"anonymous":false,"inputs":[{"indexed":true,"name":"_from","type":"address"},
{"indexed":true,"name":"_to","type":"address"},{"indexed":false,"name":"_value","type":"uint256"}],"name":"Transfer","type":"event"}];
// 合约地址
var address = "0xb2cdd356e58280906ce53e1665697b50f88aac56";
// 通过ABI和地址获取已部署的合约对象
var mycontract = web3.eth.contract(abi).at(address);
```