# MAF前端通信协议设计方案 version 3.3

## 客户端发起交易请求

- 实例 LoginCtrl.js

```
huaxiaModule.controller('LoginCtrl', ['$scope', 'MafService',
    function ($scope, mafService) {
        // 正确返回之后的回调
        var doSJC300001Success = function(data){};

        // 在 datacenter 中设置值
        var dc = new unieap.ds.DataCenter();
        dc.setParameter('username', $('#username').val());
        dc.setParameter('password', $('#password').val());
        dc.setParameter('verifycode', $('#security-code').val());

        //发送请求
        $scope.loginAction = function () {
            mafService.doSJC300001(dc).
                then(doSJC300001Success); //回调
        };
    }]);
```

## 需要自动生成的部分 MafService.js

- service 命名

  MafService

- 接口命名规则

  do + 交易号，比如： doSJC300001

- 接口调用成功和失败的回调函数命名规则

  业务名称 +Success 或者 业务名称 +Error ，比如：

  登录成功的回调函数命名为： doSJC300001Success

  登录失败的回调命名为： doSJC300001Error

- service 报文格式

报文分为上行报文和下行报文。

- 生成的 service 格式，里面包含了客户选择的交易，比如登录服务：

```
huaxiaModule.factory('MafService', ['Action',
    function(action) {
        return {
            // 登录服务接口，假设：SJC300001表示登录交易
            doSJC300001: function(dc) {
                dc.addHeaderAttribute("transCode", "security_login");
                action.send({
                    url: "/techcomp/security/inbound/securityIbd!commonMethod.action",
                    dc: dc,
                    //上行报文
                    requestMapping: {
                        headTag: "head",
                        bodyTag: "body",
                        head: {
                            "transCode": {
```

```
                            "label": " 交易码 ",
                            "type": "string"
                        }
                    },
                    body: {
                        "username": {
                            "label": " 用户名 ",
                            "type": "string"
                        },
                        "password": {
                            "label": " 密码 ",
                            "type": "string"
                        },
                        "verifycode": {
                            "label": " 验证码 ",
                            "type": "string"
                        }
                    }
                },
                //下行报文
                responseMapping: {
                    "returnCode": "#head.returnCode",
                    "successCode": "000000",
                    "systemExceptionCode": "999999",
                    "exceptionMessage": "#head.exceptionMsg",
                    headTag: "head",
                    bodyTag: "body",
                    head: {
                        "transCode": {
                            "label": " 交易码 ",
                            "type": "string"
                        },
                        "returnCode": {
                            "label": " 返回码 ",
                            "type": "string"
                        },
                        "exceptionMsg": {
                            "label": " 异常信息 ",
                            "type": "string"
                        }
                    },
                    body: {}
                },
                loadSuccessed: this.view.loginSuccess,
                loadFailed: this.view.loginError,
                token: false
            });
        },
        // 登出服务接口
        doSJC300002: function(dc, name){
            //...
        },
        // 其他交易接口
    };
}]);
```

- 生成的 form validation格式，里面包含了每个字段的校验项和提示

```
huaxiaModule.factory('MafService.formValidate.doSJC300001', function(){
    var columns = function(){};
    columns.username = {
        "prompts": {
            "required": " 请输入用户名 ",
            "maxLength": " 最大长度不能超过 255 个字符。 " ,
            "pattern": " 用户名必须以字母开始 "
        },
```

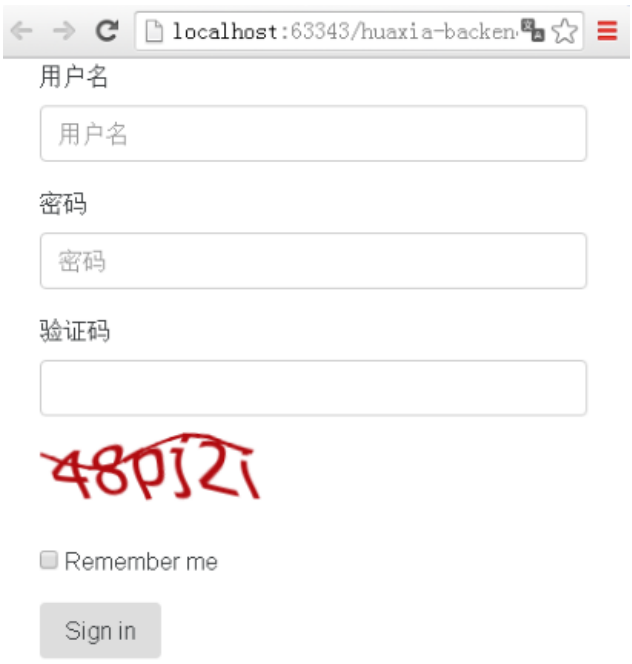
```
        "required": "required",
        "maxLength": 255,
        "pattern": /^[a-z]+.*$/i
    };

    columns.password = { //... };

    columns.verifycode = {
        "prompts": {
            "required": " 请输入验证码 ,
            "pattern": " 用户名必须以字母开始 "
        },
        "required": true,
        "maxLength": 10,
        "minLength": 4,
        "pattern": /^[a-z1-9]+$/i
        };
    return columns;
});

huaxiaModule.factory('MafService.formValidate.doSJC300002', function(){
    //...
});
```

# 报文签名

```
    //从rootScope中取值_signature
    $rootScope._signature
```

# 开发流程

以登录为例，进行详细说明




- 新建一个 login.html ， 在头部引入maf.js, angularValidateWithToast.js, MafService.js ， Action.js 和 LoginCtrl.js

```
    <script type="text/javascript" src="../app/services/common/MafService.js"></script>
    <script type="text/javascript" src="../app/services/common/Action.js"></script>
    <script type="text/javascript" src="../app/controllers/LoginCtrl.js"></script>
</head>
```

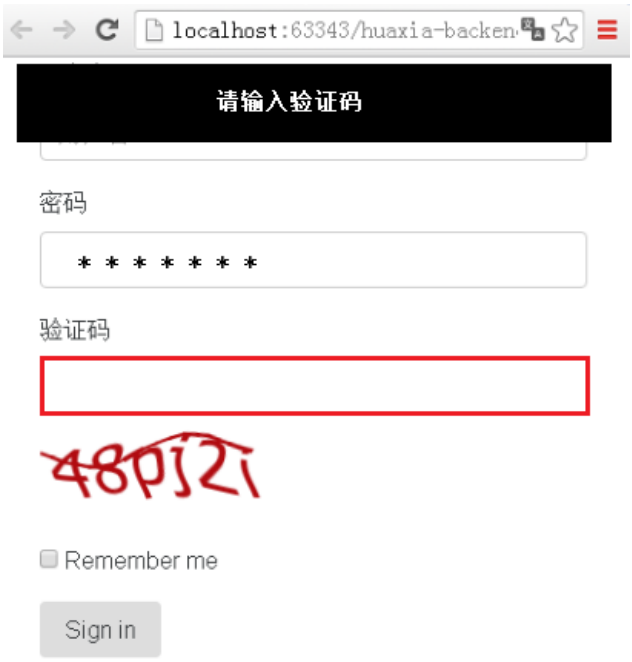
- 在form中加入表单验证 `formValidate指令`（例如：formValidate-doSJC300002）
- 在各个表单中添加`数据模型`（例如：ng-model="username"）
- 指令的格式：formValidate-为前缀，后面为do+交易号

```
<form name="login" action="" formValidate-doSJC300002>
    <input type="text" ng-model="username" />
    <input type="password" ng-model="password" />
    <!-- 其他表单 -- >
</form>
```

- 打开 login.html ， 把 logAction 注册在按钮的点击事件中即可：

```
<div class="col-sm-offset-2 col-sm-10">
    <button type="submit" class="btn btn-default" ng-click="loginAction();">Sign in</button>
</div>
```

- 验证表单，输入用户名和密码后，点击sign in，弹出错误提示




# 防重复提交处理

- 根据用户的需要， 把需要做 `防重复提交处理` 的交易码，放在main.js中TOKEN_SERVICES这个数组中

```
huaxiaModule.constant('TOKEN_SERVICES', ['SJC300002']);
```

- 实例化huaxiaModule的时候，首先会把需要做 `防重复提交处理` 的交易码发送到后台

```
huaxiaModule.run(function(TOKEN_SERVICES){
    _getServiceToken(TOKEN_SERVICES);
});
```

- 将请求返回的token放在$rootScope中
- 安全检查处理会根据send方法中传入的对象的token字段是否为真。 如果为true，就会从$rootScope中取得_serviceToke[head.transCode]，并放在securityContext['token']作为防重复提交的依据

```
// 2、防重复提交处理
if (args.token) {
    var tokenValue = $rootScope._serviceToke[head.transCode];
    securityContext['token'] = tokenValue.toString();
}
```

- 在一次登录的生命周期内，一个交易可能会做多次，那么需要多次获取同一个交易的Token情况

  在需要的时候调用`ServiceToken`服务的get方法，这样可以针对这类交易在重新请求新的token。