# React Native之Bmob为后端实现登录注册



**什么是Bmob：**

bmob是一个后端云服务平台，通过它，Bmob可以为你提供实时数据与文件存储功能，轻松实现应用“云与端”的数据连通。数据存储除了常规应用文本信息的存储，还可以存储图片、视频、音频、地理位置等信息。除了这些，还可以做消息推送，短信验证，支付（可以自己集成，不需要企业账号），即时通讯等功能。如果我们不会服务器相关技术，我们可以直接使用bmob，再也不用担心后端的问题。

通过简单的对Bmob的介绍，能大致理解为什么我们会使用Bmob作为猴急端后台了吧。

献上网站<https://www.bmob.cn/>

**怎么集成项目呢**

实际上和很多的第三方平台差不多。

**第一步：注册账户**

很easy，微信就可以扫二维码。

**第二步：注册应用**

也很简单，进入控制台，点击左上角的创建应用，按照符合自己应用的套餐，日常学习的话，选择免费版本就好了。创建完就是下面的样子：

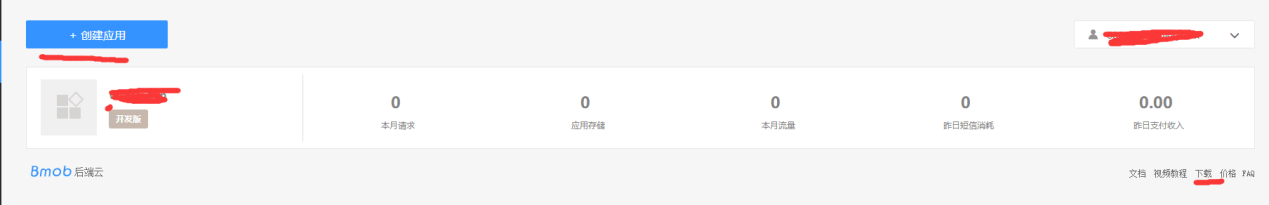


图1-1 创建应用

**第三部：下载sdk**

点击上图的下载，选择自己想要的功能：



图1-2 下载sdk

用在RN上的话，选择js版本。下载下来就是四个js文件。右上角还有快速入门，文档以后使用可以来看的。

资源我们到手了，接下来就是非常重要的步骤了。

**第四步：表操作**

进入控制台，点击创建的项目，中间栏目有所有的表，傻瓜式操作就可以啦。

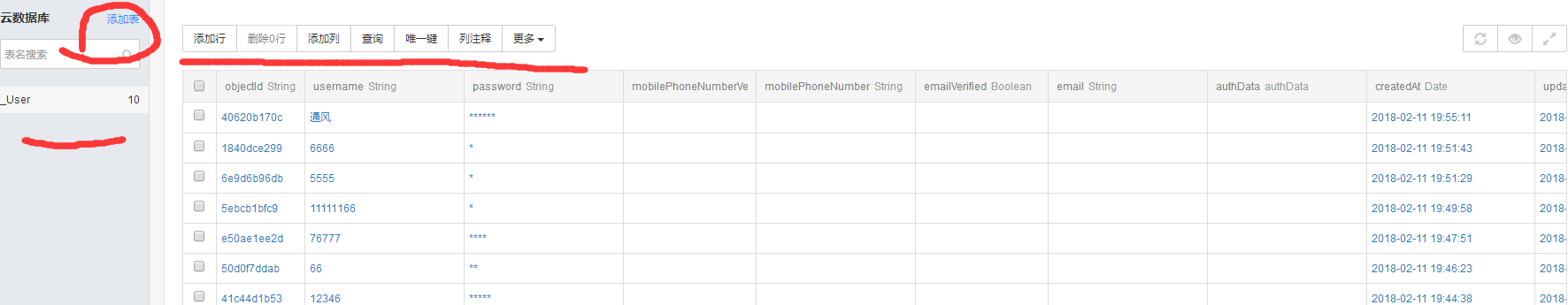


图 数据库

是不是很傻瓜式操作。

**第五步：集成导入RN**

在此之前，你得有个初始化好的RN项目，没有的话，先去创一个吧。

在目录下创建个文件夹，保存四个js文件。

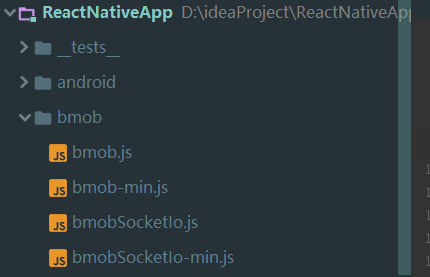


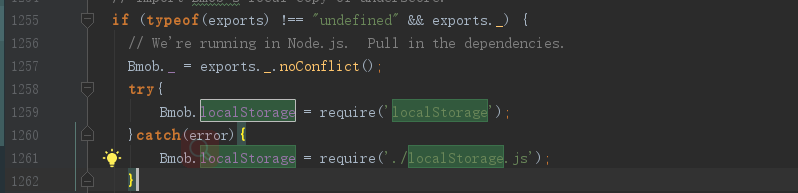
图 1-3 导入sdk

**填坑**

在编写代码前，得先说一下我踩的坑。

坑1：bmob中找不到localStorage的module，

问题位置在bmob.js文件中1259行的样子：



1259写的是返回内置模块。

1261写的是返回绝对路径模块。

看不懂require的话看看这个吧

<http://www.ruanyifeng.com/blog/2015/05/require.html> 阮一峰的

很棒，那我使用npm导入一下吧。

npm install localStorage

项目中有了

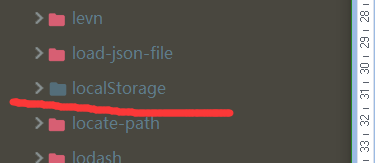


图1-3 localstroage位置

但是它还是没找到，是不是版本问题?

既然内部模块没找到，我给它来个绝对路径，把我们的localStorage.js复制到bmob目录下面。

成功了（心痛,知道原因的可以告诉我呀 我没找到原因）

坑2：XMLHttpRequest 没找到（mmp 闹哪样代码没法用啊）

XMLHttpRequest这个名字如果不熟悉的话，ajax这个名字熟悉吧，就是一个网络请求类,RN推荐使用的网络请求类是Fetch，ajax它不推荐使用在RN中，但是RN还算是内置有这个东西（幸好幸好）路径:

.\node\_modules\react-native\Libraries\Network\XMLHttpRequest.js

发现要的是XMLHttpRequest的实例对象，那好办我给它创建一个呗。在坑一的正下方：

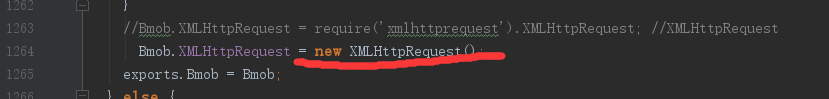


图1-4 填坑

还有一处一样的在1490行一样改掉就好了。

终于，坑没了，至少编译通过了。

**第六步：编写代码拉**

**快速入门可以查看：**

**http://doc.bmob.cn/data/javascript/**

**首先，导入Bmob到页面吧。**

**import** bmob **from** './bmob/bmob.js';

**和普通导入组件差不多的，就是写的变成了绝对路径。**

**然后嘞，初始化呗**

bmob.Bmob.initialize(, ,);

**bmob内部有个Bmob实例对象。看着累的话写个变量保存bmob.Bmob就看着舒服了。**

**来个登录呗，说白了就是查询用户存在性。**

**看具体代码吧;**

loginCheck(username, password) {  
 **var** list = bmob.Bmob.Object.extend("\_User");//获取user表的类  
 **var** qury = **new** bmob.Bmob.Query(list);  
 **var** q = qury.equalTo("username", username).equalTo("password", password);//条件  
 q.find({  
 success: (result)=>{//查询注册（不管有没有存在都是成功的）  
 alert("查询成功: "+result.length);  
 **if**(result.length==1){  
 **this**.props.navigation.navigate("APP",**null**);//跳转页面  
 }  
 },  
 error: **function** (error) {//发生了异常（服务炸了 没网啥的）  
 alert("查询失败: " + error.code + " " + error.message);  
 }  
 })  
}

**代码很简单，提一点正常情况下不管存不存在该用户都是返回成功的，但是result实际上是个数组，里面如果没有数据的话，就不存在；反之，存在。**

**仔细发现是不是有点不一样，为什么success后面用的箭头函数，error后面用的是function函数，**

**因为如果用function this保存的是Qury内部的上下文，不是当前页面的上下文，是不是很坑啊。所以要改变当前的属性或者啥操作的话，还是用箭头函数吧,或者是在回调函数后头使用.bind（this）.**

**再来看个注册代码呗。**

*/\*\*  
 \* 查询用户存在性  
 \** ***@param*** *username  
 \*/*checkUser(username) {  
 **var** User = bmob.Bmob.Object.extend("\_User");  
 **var** qury = **new** bmob.Bmob.Query(User);  
 qury.equalTo("username", username).find({  
 success: (result) => {  
 **if** (result.length == 0) {//不存在  
 **this**.insertUser(**this**.user, **this**.pass);//插入  
 } **else** {  
 alert("用户名已存在");  
 }  
 }, error: (error) => {  
 alert("异常");  
 }  
 });  
}  
*/\*\*  
 \* 插入表操作  
 \** ***@param*** *username  
 \** ***@param*** *password  
 \*/*insertUser(username, password) {  
 **var** User = bmob.Bmob.Object.extend("\_User");  
 **var** \_user = **new** User();  
 \_user.set("username", username).set("password", password);  
 \_user.save(**null**, {  
 success: (object) => {  
 },  
 error: **function** (error) {  
 alert("异常：" + error.message);  
 }  
 })  
}

**实际上就是个先查询存在性，然后插入表的操作。**

**献上详细开发文档网址；**

[**http://doc.bmob.cn/data/javascript/develop\_doc/**](http://doc.bmob.cn/data/javascript/develop_doc/)

**献上上述代码：**

[**https://github.com/xiaogoudandan/StudyReactNative**](https://github.com/xiaogoudandan/StudyReactNative)

**感谢查看文章，如果可以关注微信公众号，一起学习。**

**作者：william\_zhang**

**时间：20180211**

****