Promise用链式调用、状态机等机制优化了传统的异步嵌套（回调地狱）问题。

Promise主要成员有status(用来记录当前操作的状态)、value(当前操作成功后的数据)、then、resolve、reject、catch

then是Prominse的核心方法，它的接收两个可选参数但是参数必须是函数，用于异步操作失败和成功的函数回调，当异步操作状态为成功并且链式调用时then会把当前对象成员变量（由resolve设置的上一次异步操作结果）传给下一个异步操作，如果当前异步操作为失败则会执行失败回调函数并将value值传入并返回一个新的Promise对象。

resolve和reject分别是设置当前状态为成功和失败，并将传来的值赋给当前对象的value。

catch是在Promise链中专门用来错误处理的方法，它能捕获前面then中抛出的错误和resolve的拒绝状态并返回一个新的Promise对象。

Promise对异步代码的优化，假如有一个登录并返回用户信息的接口A、获取当前用户好友列表的接口B，根据数组里所有用户id获取对应用户基本信息的接口C。现在需要完成在用户登录后获取列表里所有好友的基本信息的功能。

普通异步请求需要进行三重的请求嵌套，即在A接口请求成功的回调函数里的用户ID向B接口请求。当用户好友列表请求成功后需要查询列表里所有用户的基本信息，即在B请求成功的回调函数里再次请求接口C，最后把C获取到的数据渲染到页面上。

普通异步调用的模拟代码

login((err, user) => {…

getFriends(user.userId, (err, data) => { …

getFriendsInfo(data.friends, (err, friendsInfo) => {…

console.log(friendsInfo); // 最终数据

…});…

});…});

Promise优化后的代码

login()

.then(user => getFriends(user.userId))

.then(data => getFriendsInfo(data.friends))

.then(friendsInfo => console.log(friendsInfo))

.catch(err => console.error(err));