**异步函数 async await**

// 异步函数，它调用后，返回的是一个promise对象

async function fn() {

return 1;

}

fn().then(function(data) {

console.log(data);

})

所以，上面改成更好的方法

function p(num) {

return new Promise(function(resolve, reject) {

setTimeout(function() {

resolve(num);

}, 1000);

});

};

// 异步函数，它调用后，返回的是一个promise对象

async function fn() {

let d1 = await p(1); // await的意思是等待这个函数执行完了再执行下一个，函数的返回值d1是p函数成功后resolve(num)中的num

let d2 = await p(2);

let d3 = await p(3);

return [d1, d2, d3];

}

fn().then(function(data) {

console.log(data); // [1, 2, 3]

})

如果有失败

function p(num) {

return new Promise(function(resolve, reject) {

setTimeout(function() {

// 判断是不是数字，如果是数字，走成功

if (!isNaN(num)) {

resolve(num);

} else {

reject(new Error('"' + num + '" 不是个数字'));

}

}, 1000);

});

};

// 加async后，就是异步函数，它调用后，返回的是一个promise对象

async function fn() {

let d1 = await p(1); // await的意思是等待这个函数执行完了再执行下一个，函数的返回值d1是成功后resolve(num)中的num

let d2 = await p('abc');

let d3 = await p(3);

return [d1, d2, d3];

};

fn().then(function(data) {

console.log(data); // [1, 2, 3]

}).catch(function(err) {

console.log(err); // 如果失败了，则打印失败的内容

});