Ajax

1.AJAX的概念及优势

\* 什么是AJAX

    \* AJAX（Asynchronous JavaScript And XML），（异步 JavaScript 和 XML），中文名：阿贾克斯。是指一种创建异步交互式网页应用的网页开发技术。

    \* AJAX 是一种在无需重新加载整个网页的情况下，能够更新部分网页的技术

    \* 前端通过与服务器进行少量数据交换，AJAX 可以使网页实现异步更新。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。传统的网页（不使用 Ajax）如果需要更新内容，必须重载整个网页页面。

\* 为什么要使用AJAX

    \* 更自然、流畅的用户体验，对用户的操作即时响应

    \* 在不中断用户操作的情况下与Web服务器进行通信

    \* 更灵敏的响应用户访问，实现近似于桌面应用程序的交互效果

    \* 通过局部更新页面降低网络流量，提高网络的使用效率

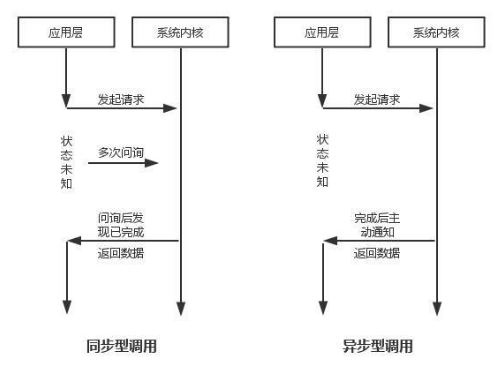
2.同步与异步

同步，可以理解为在执行完一个函数或方法之后，一直等待系统返回值或消息，这时程序是出于阻塞的，

只有接收到返回的值或消息后才往下执行其他的命令。

异步，执行完函数或方法后，不必阻塞性地等待返回值或消息，只需要向系统委托一个异步过程，

那么当系统接收到返回值或消息时，系统会自动触发委托的异步过程，从而完成一个完整的流程。



同步和异步的深入研究：

以下代码运行的结果是什么？

console.log(1);

setTimeout(function(){console.log(2);},0);

setTimeout(function(){console.log(3);},0);

setTimeout(function(){console.log(4);},0);

console.log(5);

所有任务可以分成两种，一种是同步任务（synchronous），另一种是异步任务（asynchronous）。

同步任务指的是，在主线程上排队执行的任务，只有前一个任务执行完毕，才能执行后一个任务；      //执行栈

异步任务指的是，不进入主线程、而进入"任务队列"（task queue）的任务，只有等主线程任务执行完毕，  //任务队列

"任务队列"开始通知主线程，请求执行任务，该任务才会进入主线程执行。

具体来说，异步运行机制如下：

（1）所有同步任务都在主线程上执行，形成一个执行栈（execution context stack）。

（2）主线程之外，还存在一个"任务队列"（task queue）。只要异步任务有了运行结果，就在"任务队列"之中放置一个事件。

（3）一旦"执行栈"中的所有同步任务执行完毕，系统就会读取"任务队列"，看看里面有哪些事件。那些对应的异步任务，于是结束等待状态，进入执行栈，开始执行。

（4）主线程不断重复上面的第三步。

3.XMLHttpRequest对象

\* XMLHttpRequest的理解：

  AJAX的核心对象是XMLHttpRequest，即AJAX的异步操作，和服务器 交互主要依赖该对象。

  XMLHttpRequest 对象提供了对 HTTP 协议的完全的访问，包括做出 POST 和 HEAD 请求以及普通的 GET 请求的能力

   以前浏览器负责显示和发送请求接收响应。两件事情同一时刻只能做一件，没法同时进行。这样会让用户感觉不好(友好性不好)，

   使用XMLHttpRequest对象，可以把浏览器解脱出来，可以让浏览器只负责显示，而完成请求的事情由XMLHttpRequest对象负责。

   这样两者各负其责，效率更高，效果更好，用户体验很好，用户永远不会看到浏览器空白。

4.ajax的编写步骤

1、创建XMLHttpRequest对象

  let request = new XMLHttpRequest();

2、设置请求参数

  request.open("get", "<http://10.0.152.17/AJAX/ajaxtest>", true); //接口

3、设置回调函数

    request.onreadystatechange = function(){

  if(request.readyState == 4) {

  alert(request.responseText);

  }

     }

4、发送请求

  request.send();

5、接收响应

  request.responseText或者request.responseXML

代码案例：

<script>

    //1.创建对象 掏出手机

    let xhr = new XMLHttpRequest();

    //2.设置请求参数 拨号

    xhr.open("POST","text.txt","true");

    //xhr.open("POST","content.php","true");

    //3.添加事件,设置回调函数

    xhr.onreadystatechange = function(){

        if(xhr.status == 200 && xhr.readyState == 4){

            //5.返回来的数据

            console.log(xhr.responseText);

        }

    }

    //4.发送

    xhr.send();

</script>

5.XMLHttpRequest对象的兼容

    let xhr;

    if (window.ActiveXObject) {

        //ie

        xhr = new ActiveXObject("Microsoft.XMLHttp");

    } else {

        //非ie

        xhr =  new XMLHttpRequest();

    }

6.用户重复注册案例:AjaxGet

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

    </head>

    <body>

        <input type="text" name="username" id="t1"/><span id="s"></span></br>

        <input type="text" name="pwd"/>

    </body>

</html>

<script>

    let t1 = document.getElementById("t1");

    let s = document.getElementById("s");

    t1.onblur = function(){

        //1.创建对象

        let xhr = new XMLHttpRequest();

        //2.设置连接地址   get方式发送响应

        //注意这部贼关键

        xhr.open("GET","isUserExist.php?username="+t1.value,"true");

        //3.设置响应事件

        xhr.onreadystatechange = function(){

            if(xhr.readyState == 4 && xhr.status == 200){

                //5.获取响应字符串

                //注意:前后端分离

                if(xhr.responseText == "1"){

                    s.innerHTML = "用户名已存在，请重新注册";

                }else{

                    s.innerHTML = "恭喜注册成功";

                }

            }

        }

        //4.发送

        xhr.send();

    }

</script>

<?php

    header("Content-type:text/html;charset=utf-8");

    //连接数据库

    $conn = mysql\_connect("localhost","root","root");

    //选择数据库

    mysql\_select\_db("xah51901");

    //获取请求过来的数据

    $name = $\_GET["username"];

    $reslut = mysql\_query("select \* from student where stuName = '$name'");

    $rows = mysql\_num\_rows($reslut);

    if($rows == 1){

        echo "1";

    }else{

        echo "0";

    }

?>

7.**XMLHttpRequest的属性与状态码**

onreadystatechange:每次对象状态改变所触发事件的事件处理程序

readyState:对象状态值：

            0 = 未初始化（uninitialized）对象创建完毕就是0

            1 = 正在加载（loading） 1：对象设置完成后就是1.即调用open函数后，装载，准备工作

            2 = 加载完毕（loaded） 调用send函数后，变成，因为，只要调用send说明加载完毕

            3 = 交互（interactive）到服务端了

            4 = 完成（complete） 服务器端处理完毕了

responseText:从服务器进程返回的数据的字符串形式

status:      从服务器返回的数字代码，如404（未找到）或200（就绪）

responseXML: 从服务器进程返回的数据的字符串形式  //JSON

send(content);

open(”method”,”URL”,"true");

<script>

    //1.创建对象 掏出手机

    //let xhr = new XMLHttpRequest();

    let xhr;

    if (window.ActiveXObject) {

        //ie

        xhr = new ActiveXObject("Microsoft.XMLHttp");

    } else {

        //非ie

        xhr =  new XMLHttpRequest();

    }

    console.log(xhr.readyState);

    //2.设置请求参数 拨号

    //xhr.open("POST", "text.txt", "true");

    xhr.open("POST","content.php","true");

    console.log(xhr.readyState);

    //3.添加事件,设置回调函数

    xhr.onreadystatechange = function() {

        console.log(xhr.readyState);

        if (xhr.status == 200 && xhr.readyState == 4) {

            //5.返回来的数据

            console.log(xhr.responseText);

        }

    }

    //4.发送

    xhr.send();

</script>

HTTP响应的状态码

    注意：在对错误进行处理时，只将少数常见的错误消息输出给用户了。尽管这是朝正确方向前进的一步，但是要告诉从事应用程序开发的用户和程序员究竟发生了什么问题



8.Ajax返回JSON

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

    </head>

    <body>

    </body>

</html>

<script>

    let xhr = new XMLHttpRequest();

    xhr.open("GET","testJson.php","true");

    xhr.onreadystatechange = function(){

        if(xhr.readyState == 4 && xhr.status == 200){

            let myJsonStr = xhr.responseText;

            let myJsonObj = JSON.parse(myJsonStr);

            console.log(myJsonObj.name,myJsonObj.age);

        }

    }

    xhr.send();

</script>

<?php

    echo '{"name":"laowang","age":18}';

?>

9.AjaxPost

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

    </head>

    <body>

        <input type="text" name="username" id="t1" /><span id="s"></span></br>

        <input type="text" name="pwd" />

    </body>

</html>

<script>

    let t1 = document.getElementById("t1");

    let s = document.getElementById("s");

    t1.onblur = function() {

        //1.创建对象

        let xhr = new XMLHttpRequest();

        //2.设置连接地址   get方式发送响应

        xhr.open("POST", "isUserExistPost.php", "true");

        //这句话必须抄 post的请求头

        xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

        //3.设置响应事件

        xhr.onreadystatechange = function() {

            if (xhr.readyState == 4 && xhr.status == 200) {

                //5.获取响应字符串

                if (xhr.responseText == "1") {

                    s.innerHTML = "用户名已存在，请重新注册";

                } else {

                    s.innerHTML = "恭喜注册成功";

                }

            }

        }

        //4.发送  通过send传递参数

        //多个参数依然用&组装 "user\_name="+ userName +"&user\_age="+ userAge ;

        xhr.send("username=" + t1.value);

    }

</script>

<?php

    header("Content-type:text/html;charset=utf-8");

    //连接数据库

    $conn = mysql\_connect("localhost","root","root");

    //选择数据库

    mysql\_select\_db("xah51901");

    //获取请求过来的数据

    $name = $\_POST["username"];

    $reslut = mysql\_query("select \* from student where stuName = '$name'");

    $rows = mysql\_num\_rows($reslut);

    if($rows == 1){

        echo "1";

    }else{

        echo "0";

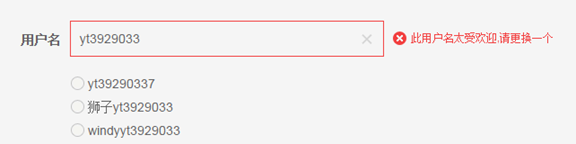
    }

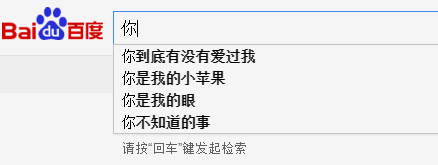
?>

作业

0.整理笔记，课堂案例

1. 验证用户名是否存在



2.搜索下拉提示

3.加载更多

4.ajax封装

10.搜索下拉提示

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

    </head>

    <body>

        搜索:<input type="text" id="t1" name="text"/></br>

        &nbsp;&nbsp;

        <ul id="list">

        </ul>

    </body>

</html>

<script>

    let t1 = document.getElementById("t1");

    let list = document.getElementById("list");

    t1.oninput = function(){

        let xhr = new XMLHttpRequest();

        xhr.open("GET","text.php?text="+t1.value,"ture");

        xhr.onreadystatechange = function(){

            if(xhr.readyState==4 && xhr.status == 200){

                showInfor(xhr.responseText);

            }

        }

        xhr.send();

    }

    function showInfor(str){

        let myJson = JSON.parse(str);

        let strHtml = "";

        for(let t in myJson){

            strHtml += "<li>" + myJson[t] + "</li>";

        }

        list.innerHTML = strHtml;

    }

</script>

<?php

    $text = $\_GET["text"];

    if($text == "我"){

        echo '{"1":"我很帅","2":"我贼帅","3":"我超级帅"}';

    }else if($text == "我爱"){

        echo '{"1":"我爱学习","2":"我爱编程","3":"我爱程序员"}';

    }

?>

11.加载更多

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

    </head>

    <body>

        <div id="box"></div></br>

        <input type="button" value="加载" id="btn"></br>

    </body>

</html>

<script>

    let box = document.getElementById("box");

    let btn = document.getElementById("btn");

    function getNews(page) {

        let xhr = new XMLHttpRequest();

        xhr.open("GET", "textnews.php?page=" + page, "true");

        xhr.onreadystatechange = function() {

            box.innerHTML += xhr.responseText;

        }

        xhr.send();

    }

    getNews(1);

    btn.onclick = function(){

        getNews(2);

        this.style.display = "none";

    }

</script>

<?php

    $page = $\_GET["page"];

    if($page==1){

        echo "1111";

    }else{

        echo "2222";

    }

?>

12.ajax的封装

<script>

    //参数

    //type:方式

    //url:请求发送地址

    //date:传参格式必须为键值对 "name=xiaoming"

    //isAsyn:是否异步

    //fun:回调函数

    function myAjax(type,url,date,isAsyn,fun){

        let xhr;

        if (window.ActiveXObject) {

            //ie

            xhr = new ActiveXObject("Microsoft.XMLHttp");

        } else {

            //非ie

            xhr =  new XMLHttpRequest();

        }

                //将字符串改成小写

        type = type.toLowerCase();

        if(type == "get"){

            let urlParam = url;

            if(date != ""){

                urlParam += "?" + date;  //test.php + "?" + "name=laowang"

            }

            xhr.open(type,urlParam,isAsyn);

            xhr.send();

        }else if(type == "post"){

            xhr.open(type,url,isAsyn);

            xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

            xhr.send(date);

        }

        xhr.onreadystatechange = function(){

            if(xhr.readyState == 4 && xhr.status == 200){

                fun(xhr.responseText);

            }

        }

    }

    //myAjax("get","textnews.php","page=2",true,showValue);

    myAjax("post","textnews.php","page=3",true,showValue);

    function showValue(str){

        console.log(str);

    }

</script>

<?php

    //$page = $\_GET["page"];

    $page = $\_POST["page"];

    if($page == 4){

        echo "123111"."</br>"."111111";

    }else{

        echo "</br>"."222222";

    }

?>

13. promise

回调函数的嵌套使用：

function fun1(f){

        console.log("fun1");

        f();

    }

    function fun2(f){

        console.log("fun2");

        f();

    }

    function fun3(f){

        console.log("fun3");

        f();

    }

    function fun4(){

        console.log("fun4");

    }

    //多个回调函数 拼接起来 可读性差

    fun1(function(){

        fun2(function(){

            fun3(fun4);

        })

    });

    //加上异步操作

    function fun1(f){

        console.log("fun1");

        setTimeout(function(){

            console.log("fun1\_异步");

            f();

        },1000);

    }

    function fun2(f){

        console.log("fun2");

        setTimeout(function(){

            console.log("fun2\_异步");

            f();

        },1000);

    }

    function fun3(f){

        console.log("fun3");

        setTimeout(function(){

            console.log("fun3\_异步");

            f();

        },1000);

    }

    function fun4(){

        console.log("fun4");

        setTimeout(function(){

            console.log("fun4\_异步");

        },1000);

    }

    fun1(function(){

        fun2(function(){

            fun3(fun4);

        })

    });

\* Promise解决的问题：回调

//如果代码可以这么实现  这样书写的代码更加容易理解

//fun1().then(fun2).then(fun3).then(fun4);

即Promise将回调模式的主从关系调换了一个位置，变成了同等的只是顺序的关系。

Promise的概念：

所谓promise，简单说是一个容器，里面保存着某个未来才会结束的事件（通常是一个异步操作）的结果，从语法上说，promise是一个对象，从它可以获取异步操作的消息，promise提供了统一的API，各种异步操作都可以用同样的方法进行处理。

promise对象的特点

（1）对象的状态不受外界影响，promise对象代表一个异步操作，有三种状态，pending（进行中）、fulfilled（已成功）、rejected（已失败）。只有异步操作的结果，可以决定当前是哪一种状态，任何其他操作都无法改变这个状态，这也是promise这个名字的由来“承若”；

（2）一旦状态改变就不会再变，任何时候都可以得到这个结果，promise对象的状态改变，只有两种可能：从pending变为fulfilled，从pending变为rejected。这时就称为resolved（已定型）。如果改变已经发生了，你再对promise对象添加回调函数，也会立即得到这个结果，这与事件（event）完全不同，事件的特点是：如果你错过了它，再去监听是得不到结果的。

有了Promise对象，就可以将异步操作以同步操作的流程表达出来，避免了层层嵌套的回调函数。此外，Promise对象提供统一的接口，使得控制异步操作更加容易。

Promise也有一些缺点。

首先，无法取消Promise，一旦新建它就会立即执行，无法中途取消。

其次，如果不设置回调函数，Promise内部抛出的错误，不会反应到外部。

第三，当处于pending状态时，无法得知目前进展到哪一个阶段（刚刚开始还是即将完成）。

promise语法:

    let p = new Promise(function(resolve,reject){

        if(条件){

           resolve();

        }else{

           reject();

        }

    });

    //切记return promise对象

    return p;

    关键的then方法:

promise对象的简单例子：

function f(param){

        let p = new Promise(function(resolve,reject){

            if(param>=3000){

                resolve(param);

            }else{

                reject(param);

            }

        });

        return p;

    }

    function f1(a){

        console.log("成功",a);

    }

    function f2(a){

        console.log("失败",a);

    }

    f(100).then(f1,f2);

f函数根据传参判断异步条件，若参数param>=3000则我们认为成功，执行reslove，反之执行reject。

then方法的参数为回调函数，根据异步的条件，决定调用哪个回调函数，参数一为reslove,参数二为reject。

通过promise修改代码：

<script>

    function fun1(){

        console.log("fun1");

        let p = new Promise(function(resolve,reject){

            setTimeout(function(){

                console.log("fun1异步");

                resolve();

            },1000);

        });

        return p;

    }

    function fun2(){

        console.log("fun2");

        let p = new Promise(function(resolve,reject){

            setTimeout(function(){

                console.log("fun2异步");

                resolve();

            },1000);

        });

        return p;

    }

    function fun3(){

        console.log("fun3");

        let p = new Promise(function(resolve,reject){

            setTimeout(function(){

                console.log("fun3异步");

                resolve();

            },1000);

        });

        return p;

    }

    function fun4(){

        console.log("fun4");

        let p = new Promise(function(resolve,reject){

            setTimeout(function(){

                console.log("fun4异步");

                resolve();

            },1000);

        });

        return p;

    }

    fun1().then(fun2).then(fun3).then(fun4);

</script>

使用Promise对象时，

1、找到异步操作的代码，放在Prmoise构造函数的参数（函数）里

2、参数（函数）的第一个参数resolve是成功时调用的函数，对应then方法（函数）的第一个参数

3、参数（函数）的第二个参数reject是失败时调用的函数，对应then方法（函数）的第二个参数

promise:修改ajax封装

<script>

        function myAjaxPromise(type,url,data,isAsyn){

        let xhr;

        if (window.ActiveXObject) {

            //ie

            xhr = new ActiveXObject("Microsoft.XMLHttp");

        } else {

            //非ie

            xhr =  new XMLHttpRequest();

        }

        type = type.toLowerCase();

        if(type=="get"){

            let urlParam = url;

            urlParam += "?"+data;

            xhr.open(type,urlParam,isAsyn);

            xhr.send();

        }else if(type=="post"){

            xhr.open(type,url,isAsyn);

            xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

            xhr.send(data);

        }

        let p = new Promise(function(resolve,reject){

            xhr.onreadystatechange = function(){

                if(xhr.status == 200 && xhr.readyState == 4){

                    resolve(xhr.responseText+"成功");

                }else if(xhr.readyState == 4){

                    reject(xhr.responseText+"失败");

                }

            }

        });

        return p;

    }

    function showValue1(str){

        console.log(str);

    }

    function showValue2(str){

        console.log("heihei");

    }

    myAjaxPromise("post","tes1t.php","name=laowangwang",true).then(showValue1,showValue2);

</script>

<?php

    header("Content-type:text/htm;charset=utf-8");

    $name = $\_POST["name"];

    echo $name;

?>

14.跨域访问

\* 什么是跨域访问

    \* 跨域访问就是跨域名访问，即A网站的网页在代码上访问了B网站的页面

    \* 由于同源策略（浏览器的安全机制），所以，AJAX不能实现跨域访问。

    同源策略：这是一种浏览器策略，浏览器的安全机制：浏览器只允许来自同一个来源的文件相互访问。（前段与后端）

    同源：同样的协议，同样的地址，同样的端口

    JavaScript或Cookie只能访问同域名下的内容同样的协议，同样的地址，同样的端口。

\* AJAX不支持跨域访问，为了达到跨域访问的目的，出现了很多的解决方案 ：JSONP,iframe,flash,xhr2等。但是比较常用的是JSONP。

\*  JSONP（JSON with Padding）可用于解决主流浏览器的跨域数据访问的问题。跟JSON没有关系。 这是一种跨域的技巧。

\*  JSONP是如何实现跨域访问的？本质上是利用HTML元素的src属性都可以跨域的思路来解决的。 因为src是可以跨域的。

如：img，script，iframe等标记的src属性的值都可以赋成其它域名的合法地址。

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

    </head>

    <body>

        <!-- 但是能通过src跨域访问 -->

        <img src= "<http://192.168.0.102/myguayu/img/1.jpg>">

    </body>

</html>

<script src="js/ajaxjs.js" type="text/javascript" charset="utf-8"></script>

<script>

    function test(value){

        console.log(value);

    }

</script>

<script src="[http://192.168.0.102/myguayu/test.php"/](http://192.168.0.102/myguayu/test.php)></script>

<script src="ttt.js"></script>

<script src="text.txt"></script>

<script src="my.php?cb=test"/></script>

<?php

    $cb = $\_GET["cb"];

    echo "$cb(123)";

?>

\* 面试题：请问jsonp是不是ajax中实现跨域访问的技术

\* jsonp不是AJAX中实现跨域访问的技术

1、jsonp没有使用XMLHttpRequest对象。

2、jsonp只是在一种跨域的技巧。

3、jsonp只支持Get方式

由于按照jsonp的这种方式跨域访问时，好像可以利用javascript和服务器端交互，能达到AJAX中XMLHttpRequest对象同样的效果。所以，人们总是把

jsonp和AJAX联系在一起。

\* 面试题：jsonp和json的区别？

1、jsonp和json根本就没有关系

2、jsonp是跨域访问的技巧

3、json是描述数据的格式

15.jsonp访问百度搜索引擎

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

    </head>

    <body>

        <input type="text" name="" id="wd" value="" />

        <ul id="list">

        </ul>

    </body>

</html>

<script>

    let oWd = document.querySelector("#wd");

    let oUl = document.querySelector("#list");

    oWd.oninput = function(){

        let scrpit = document.createElement("script");

        scrpit.src = "http://suggestion.baidu.com/su?wd="+this.value+"&cb=fun";

        document.body.appendChild(scrpit);

    }

    function fun(str){

        let strHtml = "";

        for(let i=0; i<str.s.length; i++){

            strHtml += "<li>" +str.s[i]+ "</li>";

        }

        oUl.innerHTML = strHtml;

    }

</script>

16.分页

<!DOCTYPE html>

<html>

    <head>

        <meta charset="utf-8">

        <title></title>

        <style type="text/css">

            #box{

                width: 700px;

                height: 300px;

                background-size: 500px 300px;

                position: relative;

                margin: 100px auto;

                border: 1px solid red;

            }

            #btn-previous{

                width: 30px;

                position: absolute;

                left: 25px;

                top:250px;

            }

            #btn-next{

                width: 30px;

                position: absolute;

                left: 650px;

                top:250px;

            }

            li{

                list-style: none;

                width: 120px;

                height: 200px;

                margin: 10px;

                float: left;

                border: 1px solid red;

                background: url(images/1.jpg);

            }

            ul{

                position: absolute;

                left: 25px;

                top:20px;

            }

            img{

                width: 120px;

                height: 200px;

            }

            p{

                text-align: center;

            }

        </style>

    </head>

    <body>

        <div id = "box">

        <input id="btn-previous" type="button" value="<"/>

        <input id="btn-next" type="button" value=">"/>

        <ul>

            <li>

                <img/>

                <p></p>

            </li>

            <li>

                <img/>

                <p></p>

            </li>

            <li>

                <img/>

                <p></p>

            </li>

            <li>

                <img/>

                <p></p>

            </li>

        </ul>

    </div>

    </body>

</html>

<script>

    class Paging{

        constructor(arrOli) {

            this.index = 0;

            this.arrJson = [];

            this.arrOli = arrOli;

        }

        //获取数据

        getData(){

            let xhr = new XMLHttpRequest();

            xhr.open("get","list.json",true);

            let p = new Promise(function(resolve,reject){

                xhr.onreadystatechange = function(){

                    if(xhr.status==200 && xhr.readyState==4){

                        resolve(xhr.responseText);

                    }

                }

            });

            xhr.send();

            return p;

        }

        //加载数据

        load(){

            let that = this;

            this.getData().then(function(str){

                that.arrJson = JSON.parse(str);

                console.log(str);

            });

        }

        //赋值

        Pagination(){

            for(let i=this.index\*4,j=0; i<this.index\*4+4;i++,j++){

                this.arrOli[j].firstElementChild.src =  this.arrJson[i].img;

                this.arrOli[j].lastElementChild.innerHTML = this.arrJson[i].price;

            }

        }

        previous(){

            let btnPrevious = document.getElementById("btn-previous");

            let that = this;

            btnPrevious.onclick = function(){

                that.index--;

                if(that.index<0){

                    that.index = that.arrJson.length/4-1;

                }

                that.Pagination();

            }

        }

        next(){

            let btnNext = document.getElementById("btn-next");

            let that = this;

            btnNext.onclick = function(){

                that.index++;

                if(that.index>that.arrJson.length/4-1){

                    that.index = 0;

                }

                that.Pagination();

            }

        }

    }

    let oLi = document.getElementsByTagName("li");

    let pag = new Paging(oLi);

    pag.load();

    pag.next();

    pag.previous();

</script>