在开发过程中,我们向服务端发送请求,一般会使用三种方式

XMLHttpRequest(XHR)

Fetch

jQuery实现的AJAX

其中, XMLHttpRequest(XHR)和Fetch是浏览器的原生API，jquery的ajax其实是封装了XHR。

语法： https://www.cnblogs.com/libin-1/p/6853677.html

```
fetch(...).then(fun2)
          .then(fun3) //各依赖有序执行
          .....
          .catch(fun)
```

注意：由于Fetch API是基于Promise设计，旧浏览器不支持Promise，需要使用pollyfill es6-promise

Fetch获取数据的方式

```
fetch('请求的地址')
  .then(function (resp) { //response参数是固定的
      return resp.json() //响应回来的数据格式
  })
  .then(function (data) { //真正响应回来的数据,参数不一定
     // 处理数据
  // data就是我们请求的repos
          console.log(data)
  })
  .catch(function (error） { //异常处理
      console.log('error is', error)
  });
```

重点是：响应的数据格式

fetch 提供了一个 json 方法将数据转换为 json 格式

```
return response.json();
```

fetch 提供了一个 text 方法用于获取数据，返回的是 string 格式数据

```
return response.text();
```

获取的是一个图像，需要先设置头信息，然后 fetch 会正常处理本次请求，最终使用 blob 方法获取本次请求的结果

```
return response.blob();
```

获取数据的处理

```
fetch('请求的地址')
.then(function (response) {
  let contentType = response.headers.get('content-type') //获取请求信息
  if (contentType.includes('application/json')) { //对不同的请求信息判断处理
      return response.json()
   }
   else if (contentType.includes('text/html')) {
      return response.text()
   }
});
```

Fetch发送数据

```
let content = {some: 'content'}; //提前准备要发送的数据
fetch('地址', {
  method: '请求发送方式（GET/POST）',
```

```
  headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify(content) //发送的数据
}).then(function(response) {
  // 响应回来的结果处理
});
```