

# 网络请求与数据渲染

# REQUESTS ON DATA




# 后端渲染 与 前端渲染




# 后端渲染

前端组  
**HTML, CSS, JS**  
假数据

 **project**

 **img**

 **css**

 **js**

**home.html**

**forum.html**

**article.html**

**about-us.html**

**contact.html**



程序组  
处理后上线



```
<h1>Hello!</h1>
```

```
<p>
```

Welcome to my homepage.

You are the 20th visitor.

```
</p>
```

```

```

# HTML

```
<h1>Hello!</h1>
```

```
<p>
```

Welcome to my homepage.

You are the <?=\$count?> visitor.

```
</p>
```

```

```

# PHP

# HTML

```
<div class="main">  
  <li class="item">public</li>  
  <li class="item">admin</li>  
  <li class="item">backend</li>  
  <li class="item">frontend</li>  
  <li class="item">mobile</li>  
</div>
```

# PHP

```
<div class="main">  
  <?php foreach ($team as $t) { ?>  
    <li class="item"><?=$t['name']?></li>  
  <?php } ?>  
</div>
```

后端渲染  
很省事儿  
也有很多麻烦



前端渲染？



# 2004 AJAX

单页内数据部分地、实时地更新  
**Web** 开始具有构建桌面级应用的能力





```
<h1>Hello!</h1>
```

```
<p>
```

```
  Welcome to my homepage.
```

```
  You are the <span id="count"></span> visitor.
```

```
</p>
```

```

```

# HTML

```
var field = document.getElementById("count");  
var fetchUrl = "https://api.myjson.com/bins/q30nk";  
var fetchSettings = {  
  'method': 'GET'  
};
```

```
fetch(fetchUrl, fetchSettings)  
  .then(function(response) {  
    return response.json();  
  })  
  .then(function(data) {  
    field.innerText = data.count;  
  });
```

# JS

能 **get** 到服务器发来的数据  
也得能向服务器 **post** 数据



JS

```
var field = document.getElementById("count");
var fetchUrl = "https://api.myjson.com/bins/q30nk";
var fetchSettings = {
  'method': 'GET'
};

fetch(fetchUrl, fetchSettings)
  .then(function(response) {
    return response.json();
  })
  .then(function(data) {
    field.innerText = data.count;
  });
```

JS

```
var field = document.getElementById("count");
var fetchUrl = "https://api.myjson.com/bins/q30nk";
var fetchSettings = {
  'method': 'POST',
  'data': {...}
};

fetch(fetchUrl, fetchSettings)
  .then(function(response) {
    return response.json();
  })
  .then(function(data) {
    field.innerText = data.count;
  });
```

网络请求  
是一个异步操作



JS

```
var field = document.getElementById("count");  
var fetchUrl = "https://api.myjson.com/bins/q30nk";  
var fetchSettings = {  
  'method': 'GET'  
};
```

```
fetch(fetchUrl, fetchSettings)  
  .then(function(response) {  
    return response.json();  
  })  
  .then(function(data) {  
    field.innerText = data.count;
```

所有  
需要在  
数据被渲染出之后  
才能执行的  
页面逻辑  
都放在这

```
});
```

能放在这儿吗？



```
var...  
for...  
while...  
do()...  
fetch!  
var...  
for...
```



```
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
loading...  
got data  
resolved!  
do()...  
for...  
while...  
var...  
for...
```



在 **caniuse** 上  
我们得知  
**IE** 不支持 **fetch**

<https://caniuse.com/#search=fetch>





别急，我们可以

**polyfill**

<https://github.com/github/fetch>

或者用其它的库

它们的操作方式和 **fetch** 大同小异

**axios**

**jQuery AJAX**



# 实战：获取今天知乎日报首个文章标题

## **To Playground**



# 跨域问题

## Cross-Origin-Resource-Sharing (CORS) Control

- 浏览器为了安全考虑，通常不会允许网页**向另一个域名下**发起网络请求
- 在实际项目上线之后，因为**项目所在地址和请求地址一致**，一般不会有
- 在写项目的测试过程中，是一个**很烦人**的问题...
- 使用 **Chrome** 扩展程序（插件）就地解决
- <https://chrome.google.com/webstore/detail/allow-control-allow-origi/nlfbmbojpeacfgkhkpbjhddihlkkiljbi>



# Exercise 03



# 投票系统搜索

只需显示符合条件的投票个数

北洋园|

搜索



共2条符合条件的结果



# Remember:

- 本次实现功能即可，无需考虑设计、交互、响应式与浏览器兼容性
- 所需的 **fetch** 参数
  - URL: <https://vote.twtstudio.com/search?keyword=输入的关键词>
  - method: GET



# REFERENCES

## MDN Fetch

[https://developer.mozilla.org/en-US/docs/Web/API/Fetch\\_API/Using\\_Fetch](https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch)

## Github

搜索 几乎所有 你需要使用的库和说明文档



截至时间  
下周日晚 **18:30**

提交方式  
邮件至 **i@tzingtao.com**  
或者你喜欢的其它方式







# CHEERS