history.pushState()

在 HTML 文件中, history.pushState() 方法向浏览器历史添加了一个状态。

pushState() 带有三个参数：一个状态对象，一个标题（现在被忽略了），以及一个可选的URL地址。

1. state object — 状态对象是一个由 pushState()方法创建的、与历史纪录相关的JS对象。当用户定向到一个新的状态时，会触发popstate事件。事件的state属性包含了历史纪录的state对象。（译者注：总而言之，它存储JSON字符串，可以用在popstate事件中。）state 对象可以是任何可以序列化的东西。由于 火狐 会将这些对象存储在用户的磁盘上，所以用户在重启浏览器之后这些state对象会恢复，我们施加一个最大640k 的字符串在state对象的序列化表示上。如果你像pushState() 方法传递了一个序列化表示大于640k 的state对象，这个方法将扔出一个异常。如果你需要更多的空间，推荐使用sessionStorage或者localStorage。
2. title — 火狐浏览器现在已经忽略此参数，将来也许可能被使用。考虑到将来有可能的改变，传递一个空字符串是安全的做法。当然，你可以传递一个短标题给你要转变成的状态。（译者注：现在大多数浏览器不支持或者忽略这个参数，最好用null代替）
3. URL — 这个参数提供了新历史纪录的地址。请注意，浏览器在调用pushState()方法后不会去加载这个URL，但有可能在之后会这样做，比如用户重启浏览器之后。新的URL不一定要是绝对地址，如果它是相对的，它一定是相对于当前的URL。新URL必须和当前URL在同一个源下;否则，pushState() 将丢出异常。这个参数可选，如果它没有被特别标注，会被设置为文档的当前URL。

一些情况下，调用pushState和设置 window.location = "#foo"相当，这种状况下，这两种行为都会创建和激活另一个和当前页面有关的历史纪录。但是pushState()有其他优势：

新URL可以是当前URL同源下的任意地址。相反的，设置window.location会让你保持在相同页面，除非你只修改hash.

如果不必要，你可以不改变URL，相反的，将window.location设定为“#foo”；只会创建一个新的历史纪录，如果当前hash不为#foo.

你可以关联任意的数据到你的新历史纪录中。使用基于hash的方法，你需要将所有相关 的数据编码为一个短字符串。

请注意pushState()方法绝不会导致hashchange 事件被激活，即使新的URL和旧的只在hash上有区别。

在 XUL 中，它会创建一个特殊的XUL元素。

在其他文档中，则会创建一个空的URI命名空间。

代码示例：

var state = { 'page\_id': 1, 'user\_id': 5 };

var title = 'Hello World';

var url = 'hello-world.html';

history.pushState(state, title, url);