通过创建 Promise 对象开启一个异步操作的过程，一般用几步完成多次异步操作：

1. **new Promise(fn) 返回一个Promise 对象**
2. **在fn 中指定异步等处理**
3. 处理结果正常的话，调用resolve(处理结果值)
4. 处理结果错误的话，调用reject(Error对象)

我们根据这个语法尝试自己封装一个 Ajax 的方法：

```
function getURL (URL) {
  return new Promise(function (resolve, reject) {
    var req = new XMLHttpRequest()
    req.open('GET', URL, true)
    req.onload = function () {
      if (req.status === 200) {
        resolve(req.responseText)
      } else {
        reject(new Error(req.statusText))
      }
    }
    req.onerror = function () {
      reject(new Error(req.statusText))
    }
    req.send()
  })
}
// 运行示例
var URL = 'http://httpbin.org/get'
getURL(URL).then(function onFulfilled (value) {
  console.log(value)
}).catch(function onRejected (error) {
```

静态方法：resolve,reject,all,race

原型方法：then,catch