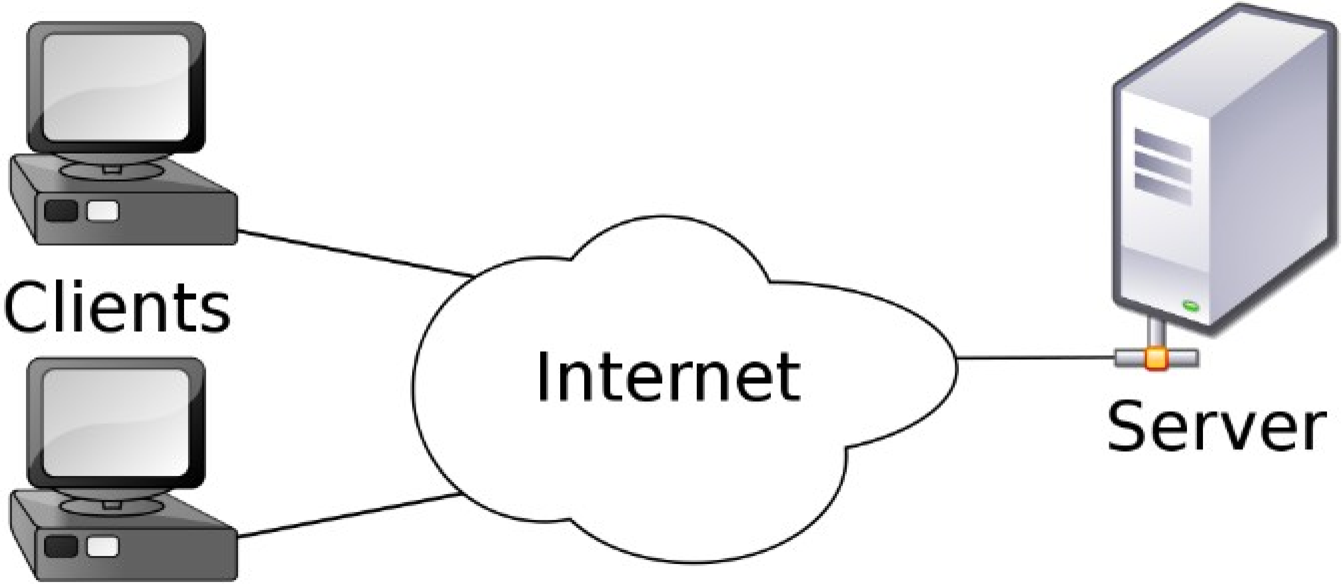
HTTP & AJAX

Networking using the client-server model



HTTP Request Header

Example 1 – Requesting introjsiap.com

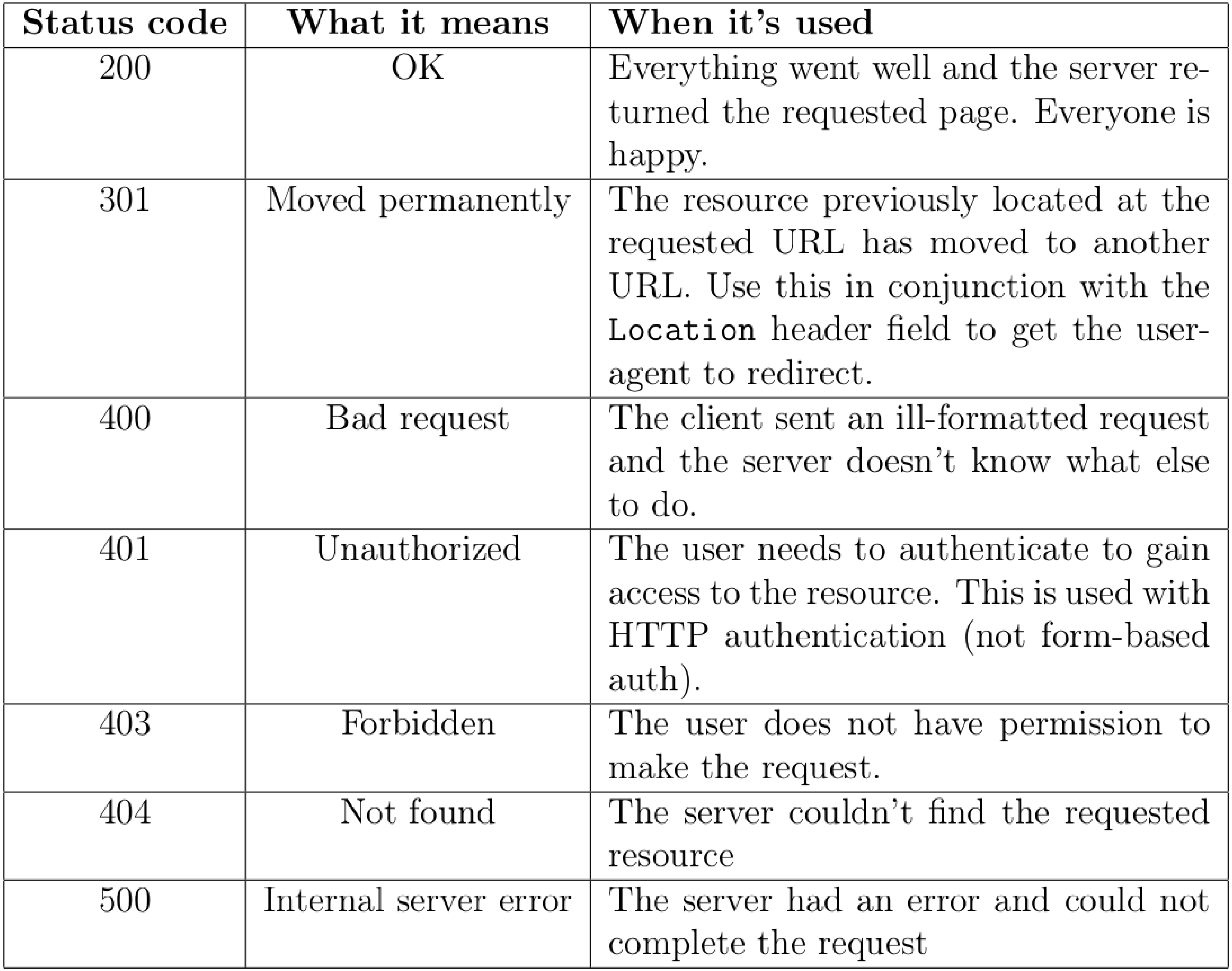
```
GET / HTTP/1.1
Host: introjsiap.com
Connection: keep-alive
User-Agent: Mozilla/5.0 Gecko/20100101 Firefox/26.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
If-None-Match: "80b3c-dd3-4f03251d489e7"
If-Modified-Since: "Fri, 17 Jan 2014 22:46:58 GMT"
Cache-Control: "max-age=0"
```

HTTP Response Header

Example 2 – Here you go: introjsiap.com

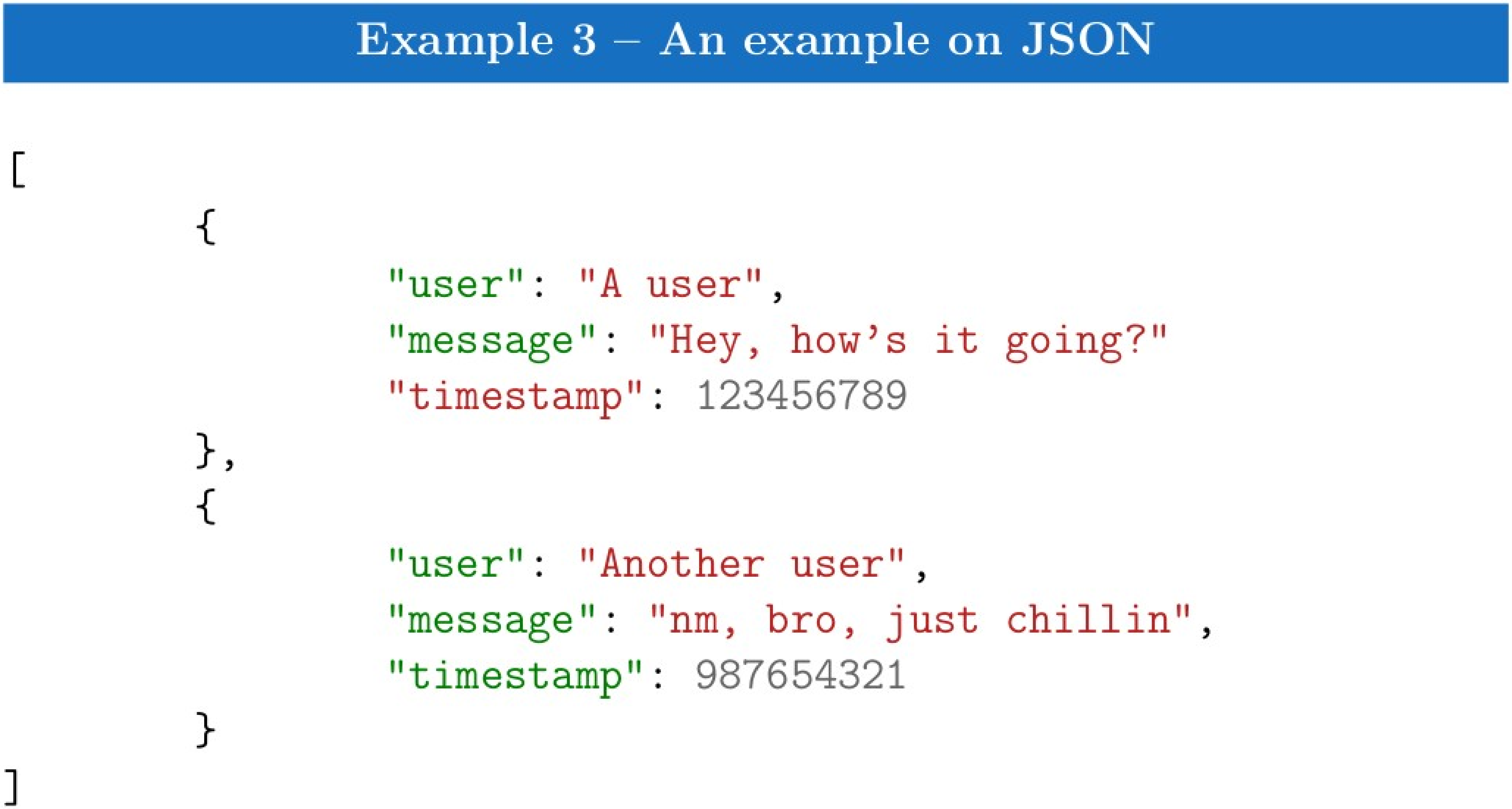
```
HTTP/1.1 200 OK
Connection: keep-alive
Date: Tue, 21 Jan 2014 11:11:11 GMT
Etag: "80b3c-dd3-4f03251d489e7"
Keep-Alive: timeout=5
Server: Apache/2.2.16 (Debian)
Vary: Accept-Encoding
Content-Length: 3450

<!-- page contents here (has a length of 3450 bytes) -->
```

HTTP Status Codes



Status code	What it means	When it's used
200	OK	Everything went well and the server returned the requested page. Everyone is happy.
301	Moved permanently	The resource previously located at the requested URL has moved to another URL. Use this in conjunction with the `Location` header field to get the user-agent to redirect.
400	Bad request	The client sent an ill-formatted request and the server doesn't know what else to do.
401	Unauthorized	The user needs to authenticate to gain access to the resource. This is used with HTTP authentication (not form-based auth).
403	Forbidden	The user does not have permission to make the request.
404	Not found	The server couldn't find the requested resource
500	Internal server error	The server had an error and could not complete the request

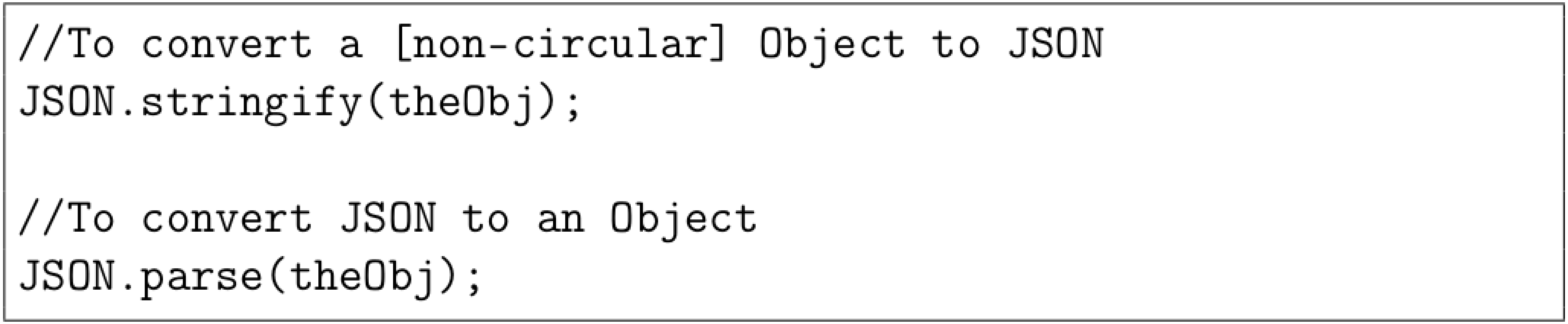
Bring it on, JSON



Example 3 – An example on JSON

```
[
        {
                "user": "A user",
                "message": "Hey, how’s it going?"
                "timestamp": 123456789
        },
        {
                "user": "Another user",
                "message": "nm, bro, just chillin",
                "timestamp": 987654321
        }
]
```

JSON, JSOFF



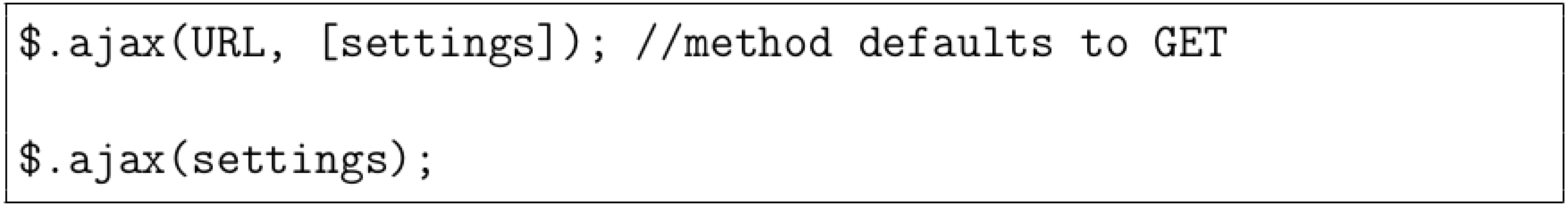
```
//To convert a [non-circular] Object to JSON
JSON.stringify(theObj);

//To convert JSON to an Object
JSON.parse(theObj);
```

AJAX Using the JS XHR Object

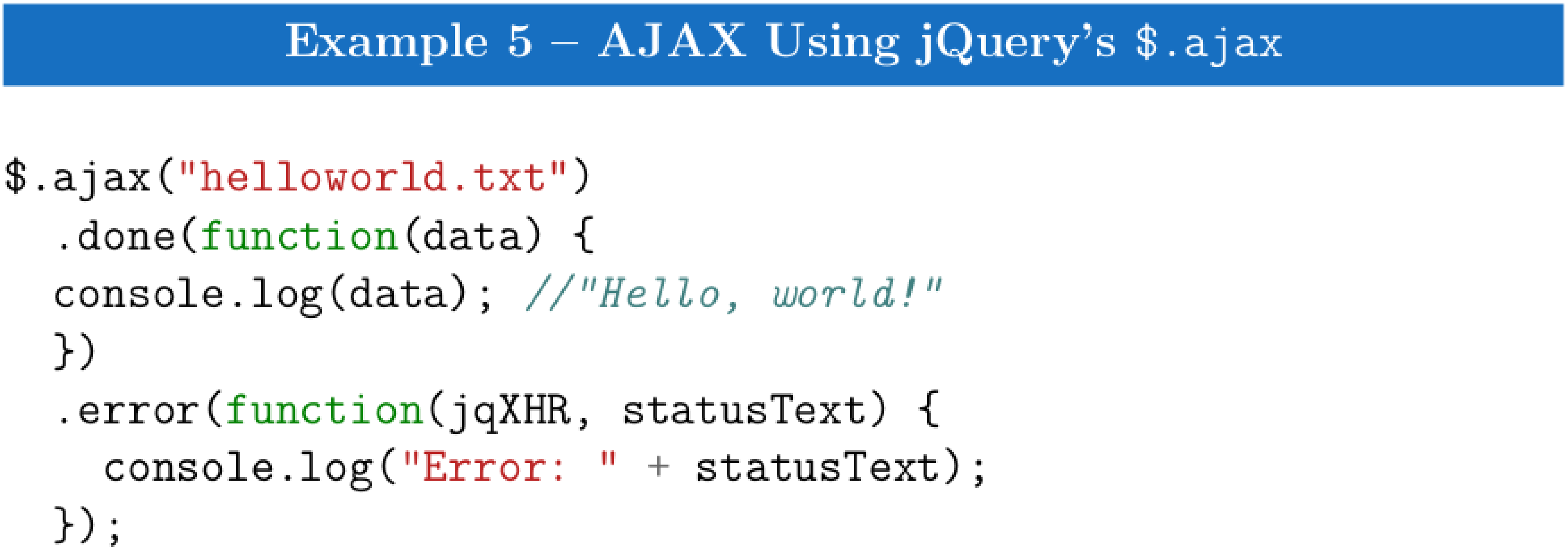
Example 4 – AJAX Using the XHR Object

```
var req = new XMLHttpRequest();

var contentLoaded = function() {
  //this is the response object
  if(this.status === 200) {
    console.log(this.responseText); //"Hello, world!"
  } else {
  console.log("Error: "+this.status);
  }
}

req.addEventListener("onload", contentLoaded);
//could also use req.onload = contentLoaded

req.open("GET", "helloworld.txt", true);
req.send();
console.log("Request sent"); //"Request sent"
```

JQuery $.ajax



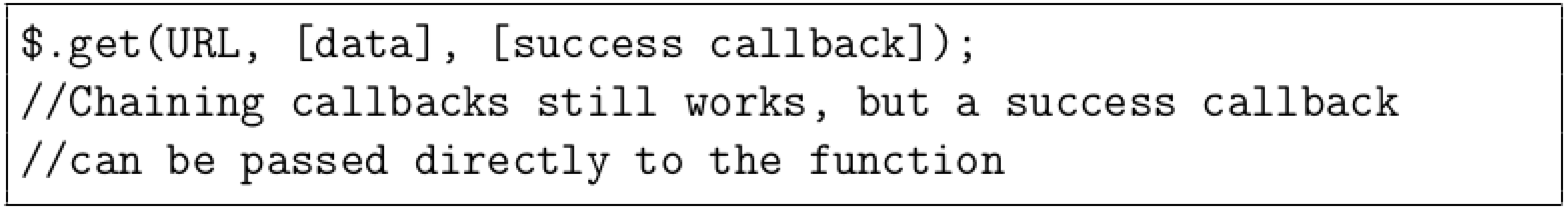
```
$.ajax(URL, [settings]); //method defaults to GET

$.ajax(settings);
```



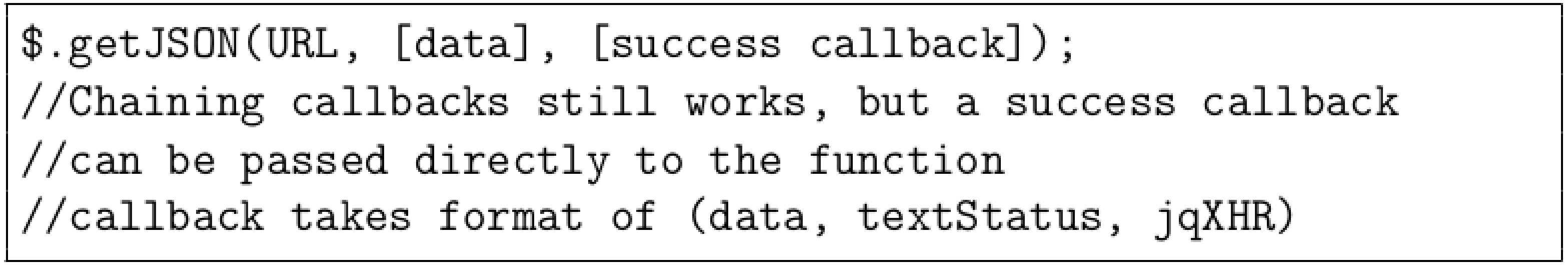
Example 5 – AJAX Using jQuery's $.ajax

```
$.ajax("helloworld.txt")
  .done(function(data) {
  console.log(data); //"Hello, world!"
  })
  .error(function(jqXHR, statusText) {
    console.log("Error: " + statusText);
  });
```

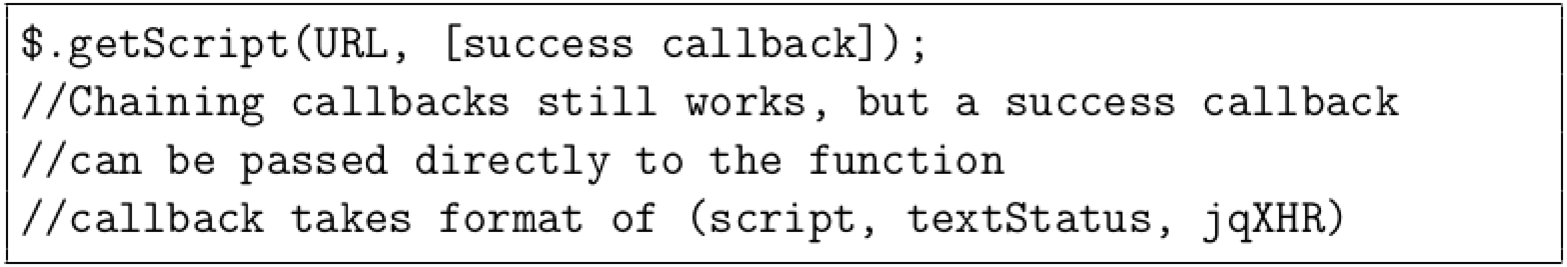
Convenient $.ajax



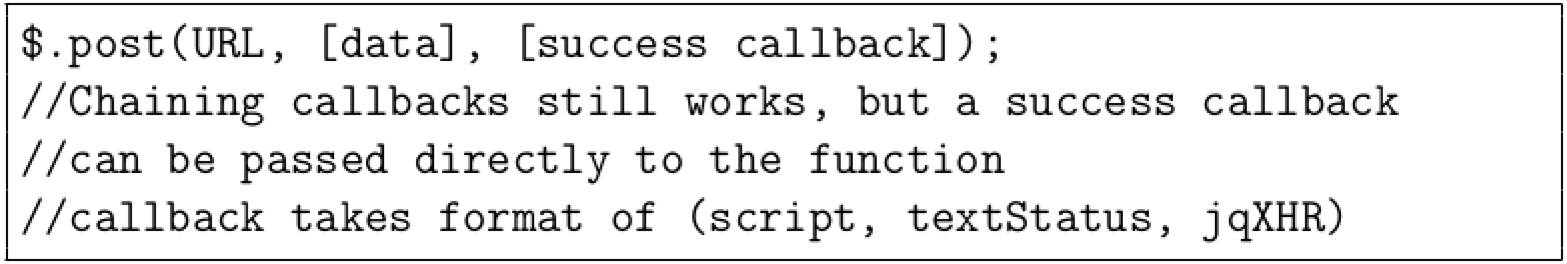
```
$.get(URL, [data], [success callback]);
//Chaining callbacks still works, but a success callback
//can be passed directly to the function
```



```
$.getJSON(URL, [data], [success callback]);
//Chaining callbacks still works, but a success callback
//can be passed directly to the function
//callback takes format of (data, textStatus, jqXHR)
```



```
$.getScript(URL, [success callback]);
//Chaining callbacks still works, but a success callback
//can be passed directly to the function
//callback takes format of (script, textStatus, jqXHR)
```



```
$.post(URL, [data], [success callback]);
//Chaining callbacks still works, but a success callback
//can be passed directly to the function
//callback takes format of (script, textStatus, jqXHR)
```