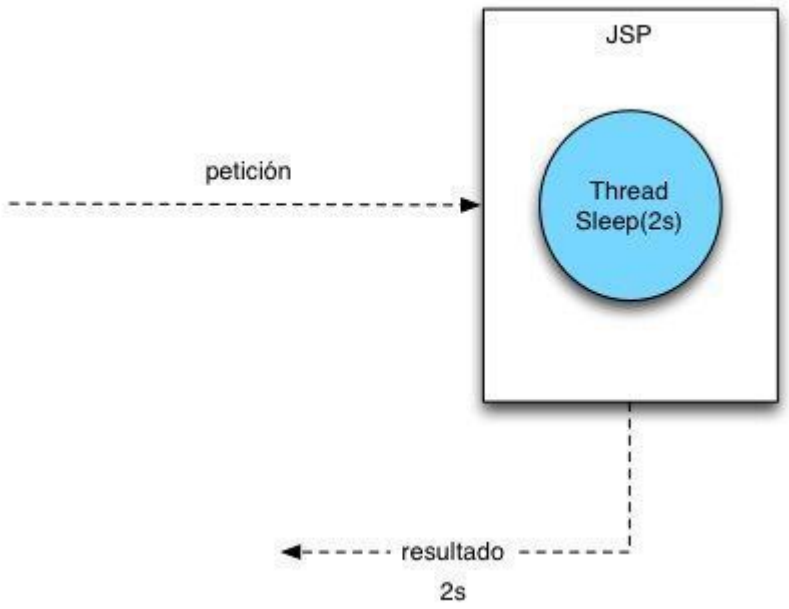
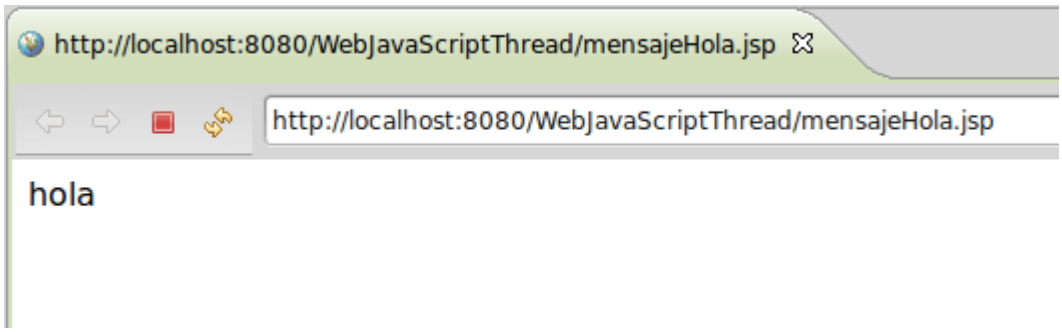
Todos utilizamos javascript a diario y quizás una de las cosas que hacemos de forma mas habitual es realizar peticiones AJAX para traernos información del servidor.Sin embargo no mucha gente entiende de forma clara como funciona Javascript Thread y su relación con el mundo de AJAX. Vamos a aprovechar este artículo para intentar aclarar ideas . Supongamos que nosotros tenemos la siguiente página JSP que nos devuelve un mensaje de “hola” y a la cual la vamos a invocar de forma normal y también vía AJAX.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
 pageEncoding="UTF-8"%>
<%
Thread.sleep(2000);
out.println("hola");
%>
```

Esta página tiene una peculiaridad y es que duerme durante 2 segundos antes de devolvernos un resultado.



Después de dos segundos la página presentará la información :



Uso de Ajax

De igual forma que podemos realizar la misma petición via AJAX :

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<script type="text/javascript" src="jquery-1.10.2.js">
</script>
<script type="text/javascript">

$(document).ready(function() {

$.get("mensajeHola.jsp",function(datos) {

alert(datos);
```

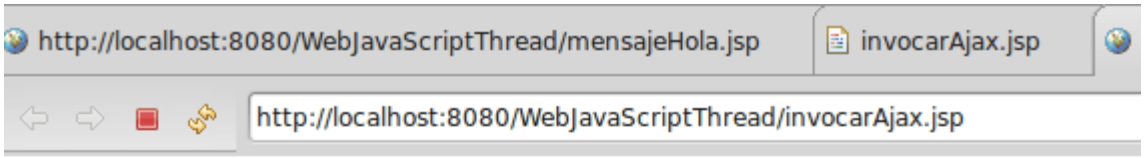
```
});

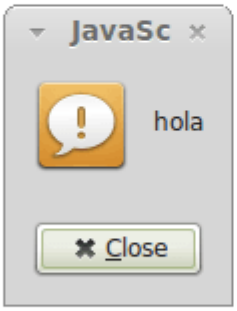
});
</script>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>

</body>
</html>
```

El resultado será muy similar y tardaremos en recibir el mensaje de “hola” 2 segundos :





Vamos a modificar el código para tener lo siguiente :

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
 pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<script type="text/javascript" src="jquery-1.10.2.js">
</script>
```

```
<script type="text/javascript">

$(document).ready(function() {

 $.get("mensajeHola.jsp",function(datos) {

 alert(datos);

 });

 var fecha= new Date();
 while (new Date() - fecha <10000){

 }

});
</script>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>
</body>
</html>
```

La modificación que hemos introduccido es muy sencilla :

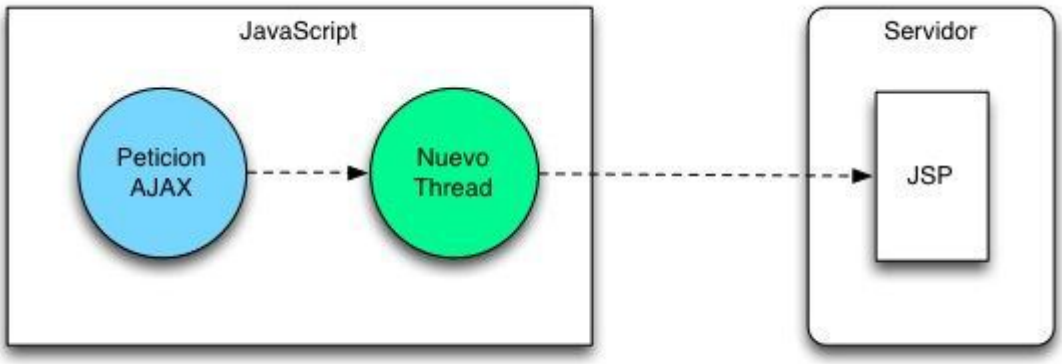
```
var fecha= new Date();
 while (new Date() - fecha <10000){
```

```
}
```

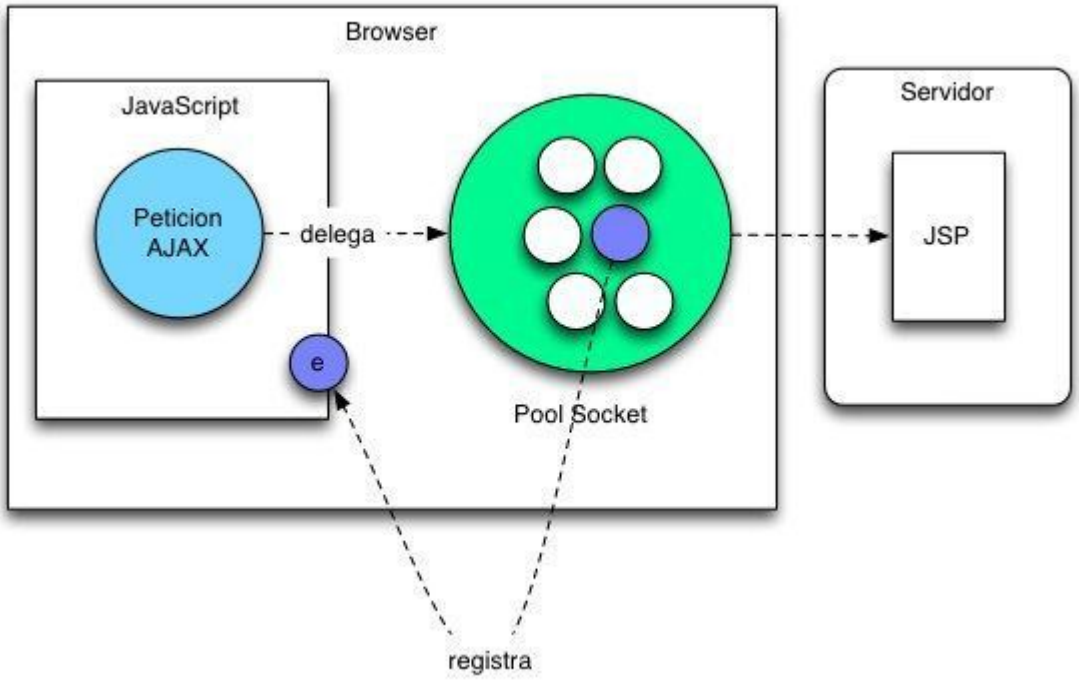
Se trata de un bucle que dura 10 segundos y que no hace nada . ¿En que nos afecta esto? en principio en NADA esta después de nuestra petición AJAX . Para nuestra sorpresa el resultado es catastrófico la petición tarda ahora 12s .¿Que esta sucediendo?.

JavaScript Thread

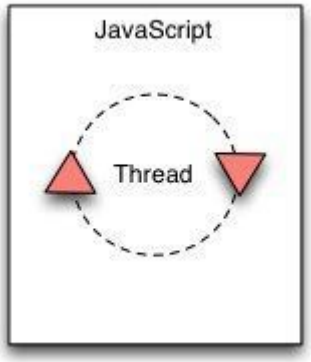
Muchos desarrolladores piensan que las peticiones AJAX funcionan de la siguiente forma :



Aunque la idea se acerca mucho a la realidad no es CORRECTA . Realmente lo que sucede es que Javascript delega en un pool de sockets que el browser tiene .La configuración y arquitectura del pool es propietaria de cada fabricante .



Ademas nos registra a nivel de JavaScript varios eventos entre ellos el evento de petición AJAX terminada. Así pues es otro proceso o thread el que se encarga de realizar la petición http desde el navegador . Es más Javascript solo tiene un Thread en el motor de Javascript (Salvo que usemos WebWorkers un concepto de HTML5) .



Así pues si mientras la petición AJAX se realiza bloqueamos durante 10 segundos el Thread

de Javascript . No será capaz de tratar el evento de petición AJAX finalizada hasta que el bucle termine. Ese es el problema que tenemos . Hay que estar muy atento a estos conceptos ya que hay muchas situaciones en las que nos pueden generar efectos colaterales y no saber que esta pasando.