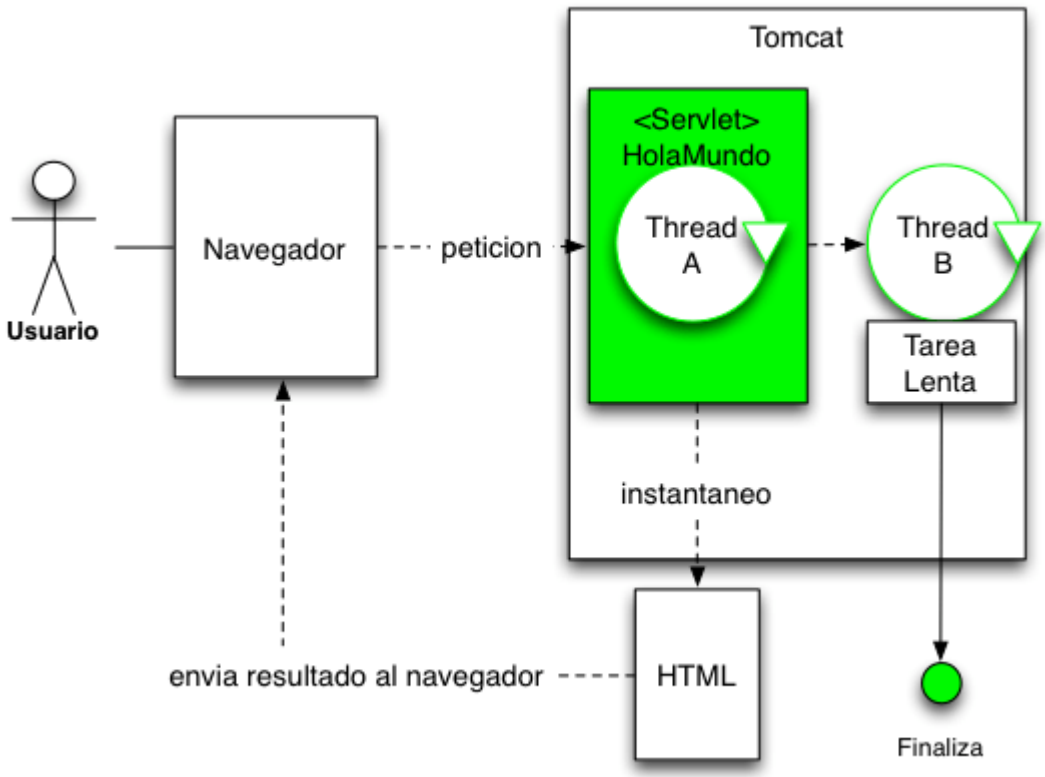
En el post anterior hemos visto como crear un servlet asincrono que realiza una tarea en background . Sin embargo el usuario una vez lanzada la petición y obtenido el mensaje de que el proceso se ejecutara de forma asincrona no recibe ningún aviso de que la tarea ha finalizado .Ya que el fichero html se envio mucho antes de que la tarea en background finalizase.



Para solventar este problema y que el usuario pueda recibir un mensaje de aviso advirtiéndole de que la tarea ha finalizado debemos añadir al servlet que hemos construido previamente un listener .

```
package com.arquitecturajava;
```

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.AsyncContext;
import javax.servlet.AsyncEvent;
import javax.servlet.AsyncListener;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(name = "HolaMundo", urlPatterns = { "/HolaMundo" },
asyncSupported = true)
public class HolaMundo extends HttpServlet {

private static final long serialVersionUID = 1L;

protected void doGet(HttpServletRequest request,
 HttpServletResponse response) throws ServletException, IOException {

final PrintWriter pw = response.getWriter();
 pw.println("<html>");
 pw.println("<body>");
 pw.println("hola mundo asincrono tarea realizandose en background");
 System.out
 .println("Hilo Principal:" + Thread.currentThread().getName());
 final AsyncContext contextoAsincrono = request.startAsync();
 contextoAsincrono.setTimeout(12000);

contextoAsincrono.addListener(new AsyncListener() {
```

```
@Override
 public void onTimeout(AsyncEvent arg0) throws IOException {
 }

@Override
 public void onStartAsync(AsyncEvent arg0) throws IOException {

}

@Override
 public void onError(AsyncEvent arg0) throws IOException {

}

@Override
 public void onComplete(AsyncEvent arg0) throws IOException {

System.out
 .println(" enviando un correo al usuario (tarea terminada)");

}
 });

contextoAsincrono.start(new Runnable() {
 @Override
 public void run() {
 for (int i = 0; i <= 10; i++) {
 System.out.println("Hilo Tarea Asincrona :"
 + Thread.currentThread().getName());
 try {
 Thread.sleep(1000);
```

```
 } catch (InterruptedException e) {
 }
 }
 contextoAsincrono.complete();

}
 });
 pw.println("</body>");
 pw.println("</html>");
 pw.close();

}

}
```

Como podemos ver hemos añadido un AsyncListener a nuestro contexto asincrono

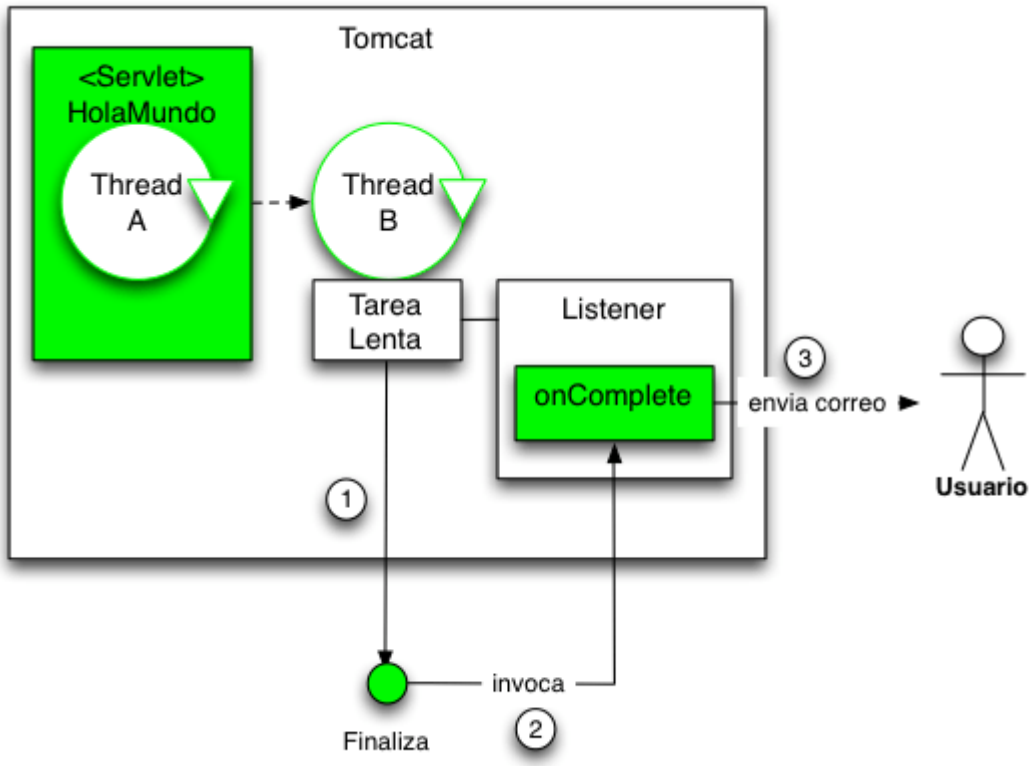
```
contextoAsincrono.addListener(new AsyncListener() {
```

Realizada esta operación debemos implementar todos los metodos de este interface . Nosotros en concreto nos centraremos en el método onComplete que es el encargado de enviar el correo (pseudocódigo) una vez que nuestra tarea asincrona haya finalizado

```
 public void onComplete(AsyncEvent arg0) throws IOException {

System.out.println(" enviando un correo al usuario (tarea
terminada)");
```

A continuación se muestra una figura aclaratoria :



Una vez implementados todos los pasos invocamos el servlet y en la consola de Tomcat nos aparecera algo como lo siguiente :

```
Hilo Principal:http-bio-8080-exec-3
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
Hilo Tarea Asincrona :http-bio-8080-exec-4
 enviando un correo al usuario (tarea terminada)
```

Al finalizar la tarea el usuario será avisado por correo de que la tarea ha finalizado

correctamente .Por ejemplo este servlet podría encargarse de procesar PDFs pesados y cuando hayan sido construidos enviar un correo al usuario que lo solicito.