1. 编写异步Servlet和过滤器：在@WebServlet和@WebFilter注解上设置asyncSupport属性为true。

或者在部署文件里面指定<async-supported>true</async-supported>.

<servlet>  
 <servlet-name>DeleteCookieServlet</servlet-name>  
 <servlet-class>com.servlet.servlet.DeleteCookieServlet</servlet-class>  
 <async-supported>true</async-supported>  
</servlet>

1. Servlet或过滤器要支持异步处理，可以通过调用ServletRequest的startAsync方法来启动一个新线程，并返回一个AsyncContext实例。、

AsyncContext实例提供各种方法并且包含ServletRequest和ServletResponse。

AsyncContext startAsync() throws IllegalStateException : 这个实例包含原生的ServletRequest和ServletResponse

AsyncContext startAsync(ServletRequest servletRequest, ServletResponse servletResponse) throws IllegalStateException ： 这个实例允许您将原来的ServletRequest和ServlerResponse重写封装后传给asyncContext。

1. AsyncContext的 start方法是非阻塞的，所以下一行代码仍将执行，即使还没调度线程启动。
2. 编写异步Servlet或者过滤器步骤：
   1. 调用ServletRequest的startAsync方法。接受返回的AsyncContext。
   2. 调用AsyncContext的setTimeout()方法，传递容器等待任务完成的超时时间的毫秒数。
   3. 调用asyncContext.start()方法，传递一个Runnable来执行一个长时间运行的任务。
   4. 调用Runnable的asyncContext.complete()或asyncContext.dispatch()方法来完成任务。

示例：

@WebServlet(urlPatterns = "/asyncDispatch", asyncSupported = true)  
public class AsyncDispatchServlet extends HttpServlet {  
 @Override  
 protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {  
 final AsyncContext asyncContext = req.getAsyncContext();  
 asyncContext.setTimeout(1000);  
 asyncContext.start(new Runnable() {  
 @Override  
 public void run() {  
 asyncContext.complete(); //or asyncContext.dispatch();  
 }  
 });  
 }  
}

1. 异步监听器AsyncListener接口

void onComplete(AsyncEvent asyncEvent) throws IOException

//在异步操作完成后调用该方法

void onTimeout(AsyncEvent asyncEvent) throws IOException

//在异步操作超时后调用该方法

void onError(AsyncEvent asyncEvent) throws IOException

//在异步操作失败后调用该方法

void onStartAsync(AsyncEvent asyncEvent) throws IOException

//在异步操作启动完毕后调用该方法

示例:

不需要@WebListener

public class MyAsyncListener implements AsyncListener {  
 @Override  
 public void onComplete(AsyncEvent asyncEvent) throws IOException {}  
 @Override  
 public void onTimeout(AsyncEvent asyncEvent) throws IOException {}  
 @Override  
 public void onError(AsyncEvent asyncEvent) throws IOException {}  
 @Override  
 public void onStartAsync(AsyncEvent asyncEvent) throws IOException }  
}

由于不需要@WebListener，所以必须在异步Servlet中的AsyncContext中显式添加异步Listener：

asyncContext.addListener(new MyAsyncListener());

1. AsyncEvent有getAsyncContext()、 getSuppliedRequest()、 getSuppliedResponse()方法获取对应AsyncContex、 ServletRequest、 ServletResponse对象