TapestryJmx

NOTE: This is outdated information that applies only to Tapestry 3 & 4. For current information see

http://tapestry.apache.org/current/tapestry-jmx/project-summary.html

Here is my preliminary code that enables JMX monitoring of Tapestry. It is based on the IMonitor interface. It requires 4 classes and a declaration inside the .application file:
<extension name="org.apache.tapestry.monitor-factory" class="actualis.web.tapestry.jmx.MonitorFactory" />

The main issue with this code is that the MBeans can not be released when the application is undeployed or redeployed. A solution could be to extend the ApplicationServlet class and use the destroy() method to release them. Maybe with Tapestry 4 it would be possible to plug the MBeans into the hivemind registry and let it release them?

```
package actualis.web.tapestry.jmx;

public interface ServiceMBean {

  /**
   * @return Returns the serviceTime.
   */
  public abstract double getServiceTime();

}
```

```
package actualis.web.tapestry.jmx;

public class Service implements ServiceMBean {

  private String serviceName;

  public Service(String serviceName) {
    this.serviceName = serviceName;
  }

  double serviceStart;

  double serviceTime;

  /* (non-Javadoc)
   * @see actualis.web.tapestry.jmx.ServiceMBean#getServiceTime()
   */
  public final double getServiceTime() {
    return serviceTime;
  }

}
```

```
package actualis.web.tapestry.jmx;

public interface PageMBean {

  /**
   * @return Returns the createTime.
   */
  public abstract double getCreateTime();

  /**
   * @return Returns the createTime.
   */
  public abstract double getLoadTime();

  /**
   * @return Returns the renderTime.
   */
  public abstract double getRenderTime();

  /**
   * @return Returns the rewindTime.
   */
  public abstract double getRewindTime();

}
```

```
package actualis.web.tapestry.jmx;

public class Page implements PageMBean {

  private String pageName;

  double createTime;

  double loadTime;

  double renderTime;

  double rewindTime;

  double createStart;

  double loadStart;

  double renderStart;

  double rewindStart;

  /* (non-Javadoc)
   * @see actualis.web.tapestry.jmx.PageMBean#getCreateTime()
   */
  public final double getCreateTime() {
    return createTime;
  }

  /* (non-Javadoc)
   * @see actualis.web.tapestry.jmx.PageMBean#getLoadTime()
   */
  public final double getLoadTime() {
    return loadTime;
  }

  /* (non-Javadoc)
   * @see actualis.web.tapestry.jmx.PageMBean#getPageName()
   */
  public final String getPageName() {
    return pageName;
  }

  /* (non-Javadoc)
   * @see actualis.web.tapestry.jmx.PageMBean#getRenderTime()
   */
  public final double getRenderTime() {
    return renderTime;
  }

  /* (non-Javadoc)
   * @see actualis.web.tapestry.jmx.PageMBean#getRewindTime()
   */
  public final double getRewindTime() {
    return rewindTime;
  }

  public Page(String pageName) {
    this.pageName = pageName;
  }

}
```

```
package actualis.web.tapestry.jmx;

import org.apache.tapestry.engine.IMonitor;
import org.apache.tapestry.engine.IMonitorFactory;
import org.apache.tapestry.request.RequestContext;

public class MonitorFactory implements IMonitorFactory {

  private static IMonitor MONITOR = new Monitor();

  public IMonitor createMonitor(RequestContext context) {
      return MONITOR;
  }
}
```

```
package actualis.web.tapestry.jmx;

import java.lang.management.ManagementFactory;
import java.util.HashMap;

import javax.management.MBeanServer;
import javax.management.MalformedObjectNameException;
import javax.management.ObjectName;

import org.apache.log4j.Logger;

import org.apache.tapestry.engine.IMonitor;

public class Monitor implements IMonitor {
  /**
   * Logger for this class
   */
  private static final Logger s_logger = Logger.getLogger(Monitor.class);

  private HashMap<String, Page> page_mbeans = new HashMap<String, Page>();
  private HashMap<String, Service> service_mbeans = new HashMap<String, Service>();

  public Page getPageMBean(String pageName) {
    Page res = page_mbeans.get(pageName);
    if (res == null) {
      res = new Page(pageName);
      MBeanServer mbs = ManagementFactory.getPlatformMBeanServer();
      ObjectName name;
      try {
        name = new ObjectName("Tapestry:type=Pages,name=" + pageName);
        mbs.registerMBean(res, name);
      } catch (Exception e) {
        s_logger.warn("Impossible to register mbean", e);
      }
      page_mbeans.put(pageName, res);
    }
    return res;
  }

  public Service getServiceMBean(String serviceName) {
    Service res = service_mbeans.get(serviceName);
    if (res == null) {
      res = new Service(serviceName);
      MBeanServer mbs = ManagementFactory.getPlatformMBeanServer();
      ObjectName name;
      try {
        name = new ObjectName("Tapestry:type=Services,name=" + serviceName);
        mbs.registerMBean(res, name);
      } catch (Exception e) {
        s_logger.warn("Impossible to register mbean", e);
      }
      service_mbeans.put(serviceName, res);
    }
```

```
    return res;
  }


  public void pageCreateBegin(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.createStart = System.currentTimeMillis();
  }

  public void pageCreateEnd(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.createTime = System.currentTimeMillis() - pmb.createStart;  }

  public void pageLoadBegin(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.loadStart = System.currentTimeMillis();
  }

  public void pageLoadEnd(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.loadTime = System.currentTimeMillis() - pmb.loadStart;
  }

  public void pageRenderBegin(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.renderStart = System.currentTimeMillis();
  }

  public void pageRenderEnd(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.renderTime = System.currentTimeMillis() - pmb.renderStart;
  }

  public void pageRewindBegin(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.rewindStart = System.currentTimeMillis();
  }

  public void pageRewindEnd(String pageName) {
    Page pmb = getPageMBean(pageName);
    pmb.rewindTime = System.currentTimeMillis() - pmb.rewindStart;
  }

  public void serviceBegin(String serviceName, String detailMessage) {
    Service serv = getServiceMBean(serviceName);
    serv.serviceStart = System.currentTimeMillis();
  }

  public void serviceEnd(String serviceName) {
    Service serv = getServiceMBean(serviceName);
    serv.serviceTime = System.currentTimeMillis() - serv.serviceStart;
  }

  public void serviceException(Throwable exception) {
      // Intentional
  }

  public void sessionBegin() {
      // Intentional
  }
}
```