SAPO

User Manual

Tracing

GIS Team
01-08-2013

Index

Tracing

O *tracing* é um conjunto de registos de informação sobre a execução de um programa. Cada registo é designado de entrada.

Cada entrada contem informação acerca de um acontecimento, registando uma mensagem explicativa da ocorrência, um nível de severidade, a duração e o desfasamento do acontecimento em relação ao início do *tracing*, e um conjunto de propriedades específicas do acontecimento. Cada entrada pode ainda conter subentradas.

A configuração do *tracing* é especificada em ficheiro XML.

O ficheiro de configuração começa com a tag **<configuration>**.

```
<configuration>
…
</configuration>
```

Appenders

Os *Appenders* são utilizados para processar os eventos do *tracing*.

Tipos de eventos:

Tipo	descrição
entry	criação de uma entrada
trace	finalização do *tracing*.

Para serem adicionados *appenders* deve ser adicionada a tag **<appenders>** à configuração, indicando o tipo (type). sendo adicionadas a esta os *appenders* pretendidos.

Em cada *appender* é configurado quais os eventos que queremos que este processe, acrescentando a tag **<event type="xxx">** com o tipo de evento entry ou trace, para os eventos de entry e trace respetivamente na secção **<events>**.

```
<configuration>
<appenders>
  <appender type="pt.sapo.gis.trace.appender.ConsoleAppender">
        <events>
              <event type="entry"></event>
              <event type="trace"></event>
        </events>
  </appender>
</appenders>
</configuration>
```

Filtro

Para cada um dos eventos pode ser definido um conjunto de filtros, por forma a filtrar que tipos de eventos serão processados.

Para acrescentar um filtro tem de ser adicionada a tag **<filters>** ao elemento que define o evento, ao qual queremos aplicar o filtro.

SeverityFilter

O filtro **severityFilter** server para indicar quais os níveis de severidade que queremos processar em cada um dos eventos definidos.

O filtro quando aplicado num evento de **entry**, indica que apenas serão processados os eventos, originados pela criação de entradas, que tenham como severidade os valores indicados.

O filtro quando aplicado num evento de **trace**, indica que apenas serão processados os eventos, originados pela finalização de um trace, que tenham pelo menos uma entrada que tenha como severidade os valores indicados.

Exemplo de filtros aplicados aos eventos:

```
<event type="entry">
      <filters>
            <severityFilter>ERROR</severityFilter>
            <severityFilter>DEBUG</severityFilter>
      </filters>
</event>
<event type="trace">
      <filters>
            <severityFilter>WARN</severityFilter>
            <severityFilter>INFO</severityFilter>
            <severityFilter>DEBUG</severityFilter>
      </filters>
</event>
```

O appender processa os eventos de **entry** que tenham como severidade **ERROR** ou **DEBUG**, assim como os eventos de **trace** que contenham pelo menos uma entrada com severidade **WARN**, **INFO** ou **DEBUG**.

ConsoleAppender

O *ConsoleAppender* é o appender que escreve a informação dos eventos de tracing para a consola de sistema.

Para adicionar um *ConsoleAppender* tem de ser adicionada à tag **<appenders>** a tag **<appender>**, com o type “pt.sapo.gis.trace.appender.ConsoleAppender”.

Exemplo de configuração de um ConsoleAppender:

```
<appender type="pt.sapo.gis.trace.appender.ConsoleAppender">
            <events>
                <event type="entry"><event>
                <event type="trace"></event>
            </events>
        </appender>
```

Adiciona um *appender* de *ConsoleAppender* que processa todos os eventos de **entry** e **trace**.

SMILoggerAppender

O *SMILoggerAppender* é o appender que envia a informação dos eventos de tracing para o SDB do SAPO segundo a norma de SMI definida pelo TMForum.

Este Appender envia um ***ManagmentReport*** com uma ***Failure*** , caso seja um evento de entry, contendo a informação da entrada que originou o evento, ou um conjunto de ***Failure***, contendo a informação das entradas, caso seja um evento de trace.

Nota: Para mais detalhes ler documentação do SMI.

BrokerSMILoggerAppender

SMILoggerAppender que envia a informação para o SDB através do broker do SAPO.

Para adicionar um *BrokerSMILoggerAppender* tem de ser adicionada à tag **<appenders>** a tag **<appender>**, com o type "pt.sapo.gis.trace.appender.SMI.BrokerSMILoggerAppender".

Configurações:

Nome	Descrição
host	url base para o servidor de broker do SAPO.
port	porto do servidor de broker do SAPO.
topic	tópico para o qual o relatório vai ser publicado.

Exemplo de configuração de um *BrokerSMILoggerAppender*:

```
<appender
type="pt.sapo.gis.trace.appender.SMI.BrokerSMILoggerAppender">
  <events>
        <event type="entry">
              <filters>
                    <severityFilter>ERROR</severityFilter>
                    <severityFilter> DEBUG </severityFilter>
              </filters>
        </event>
   </events>
   <property name="host">broker.bk.sapo.pt</property>
   <property name="port">1111</property>
```

```
    <property name="topic">sapo/sdb/smi/json/XPTO </property>
</appender>
```

Adiciona um *appender* de *BrokerSMILogger* que processa os eventos de **entry** que tenham como severidade **error** ou **debug**, sendo publicados os relatórios no tópico **sapo/sdb/smi/json/XPTO** no servidor de **broker broker.bk.sapo.pt** no porto **1111**.

SDBTraceAppender

O *SDBTraceAppender* é o appender que envia a informação de tracing para o SDB do SAPO através do serviço *Trace*.

Este Appender cria um Trace, segundo a definição de dados do serviço, seriado em JSON e enviado para o serviço por HTTP POST.

Configurações:

Nome	Descrição
host	url base para o servidor.
port	porto do servidor.
url	url do serviço de trace
protocol	protocolo a utilizar http ou https.
token	token de autenticação

Exemplo de configuração de um *SDBTraceAppender*:

```
<appender type="pt.sapo.gis.trace.appender.SMI.SDBTraceAppender">
   <events>
         <event type="trace"></event>
   </events>
   <property name="host">services.bk.sapo.pt</property>
   <property name="protocol">https</property>
   <property name="url">/SDBSS/Trace</property>
   <property name="token">32ffd807584f0</property>
</appender>
```

Adiciona um *appender* de *SDBTrace* que processa os todos eventos de trace, sendo enviados para o serviço **`/SDBSS/Trace`** no **host services.bk.sapo.pt** atavez do protocolo **https** com o token **32ffd807584f0** `de autenticação.`

Adaptadores

Os adaptadores têm como função automatizar a criação e finalização do *tracing*.

Java

TraceServletFilter

Filtro aplicável a servlets que inicia um novo trace no início de cada pedido e finaliza-o no fim desse pedido.

O trace é iniciado com o contexto extraído do header "**ESBTraceContext**", caso este seja enviado.

Adiciona igualmente uma propriedade ao trace para guardar o valor do "activity_id", extraído do header "**ESBActivityId**", caso este tenha sido enviado pelo SDB.

.NET

Para utilização do trace com processamento parallelo existe o tipo `TraceEnabledTask` que captura o trace corrente para a thread que irá executar a tarefa e assegura um contexto de sub-entradas específico a cada thread. Todas as entries geradas serão integradas no mesmo trace.

NAnt

Para configurar o tracing em ambiente NAnt, é usado o tipo `NantListenerTrace` configurado como listener. Este inicia um trace no início do build e termina-o no final.

Inclui ainda trace automático de entradas a cada início de objetivo (target)e suas sub-entradas a cada início de tarefa (task). Este comportamento pode ser desligado através da definição da propriedade NAnt `trace.ant.disableAutoTrace`.

WCF

Para configurar o tracing em ambiente WCF, é usado o endpoint behavior `TraceEndpointBehavior`, que injeta um message inpector que inicia o trace aquando da chegada de um novo pedido e o termina antes da resposta res enviada para o cliente.

ASP.NET

Para configurar o tracing em ambiente ASP.NET, é usado o módulo HTTP `TraceAspNetHttpModule`, que inicia um trace no ínício do pedido, adiciona entries de erro quando ocorrem erros não tratados e termina o trace no final do pedido.

Gestor de traces

O gestor de traces (*TraceManager*) é responsável por guardar e disponibilizar o *trace* atual no contexto da thread corrente, e treads filhas, em execução.

O *TraceManager* carregar a configuração, a partir do ficheiro especificado e inicia os *appenders* com as configurações especificadas.

Na versão em JAVA a localização e nome do ficheiro de configuração são indicados pela propriedade "trace.configFile". Em caso de omissão é considerado o ficheiro "trace.xml" na pasta de execução.

Interface

Operação	Descrição
BeginTrace	Iniciar um trace para a thread corrente.
EndTrace	Finalizar o trace corrente.
GetTrace	Obter o objeto *Trace* corrente.