

## Ajuda on-line do JavaMelody monitoring

O JavaMelody monitoring é uma ferramenta para medir e calcular estatísticas sobre o funcionamento real de uma aplicação através da utilização do aplicativo pelos usuários. O monitoramento é baseado principalmente nas estatísticas das requisições realizadas e nos gráficos de evolução gerados. Esta ferramenta permite melhorar as aplicações em processo de QA (controle de qualidade) e em produção, e ajuda a:

- fornecer fatos reais sobre os tempos médios de resposta e número de execuções
- tomar decisões quando as tendências são ruins, antes que os problemas se tornem sérios demais
- otimizar a aplicação baseando-se nos tempos de resposta mais limitantes
- localizar a causa raiz dos tempos de resposta
- verificar o ganho real após as otimizações

O relatório em html do JavaMelody é melhor visualizado com [Firefox](#), [Chrome](#) ou MS-IE9 (MS-IE7 não recomendado).

### JavaMelody Project






[Home page](#)  
[Downloads](#)  
[Release notes](#)

### Documentações

[Java SE documentations](#)  
[Java SE Platform at a Glance](#)  
[Troubleshooting Guide for Java SE 8](#)  
[Monitoring and Management for Java SE 8](#)  
[Java SE 6 Performance](#)

### Sumário

Na página de monitoração, o sumário apresenta gráficos de evolução para diversos valores de medições.

Essas medições são tomadas em um tempo T, por exemplo, a cada dois minutos. Cada gráfico acompanha a evolução de um valor medido no curto ou longo prazo, escolhido com links sobre o sumário:  dia a dia,  semana a semana,  mês a mês,  ano a ano ou  por um período escolhido. Em cada gráfico, o valor médio é exibido em verde e o valor máximo, em azul, com números abaixo do mesmo. The units in the charts are chosen based on the values: "m" is for milli (1 / 1000), "k" is for kilo/thousands, "M" is for mega/millions, "G" is for giga/billions and "u" is for micro (1 / 1000000). Os gráficos são persistidos: um reinício do servidor de aplicação não tem influência sobre eles, exceto por um intervalo nas medidas. Cada gráfico pode ser ampliado e redimensionado, clicando sobre eles no sumário.

Os gráficos exibidos são:

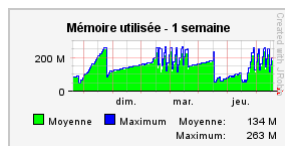
- A memória java usada, entre 0 e o máximo definido por Xmx na configuração do servidor
- O percentual de CPU usado pelo processo java, entre 0 e 100, mesmo que existam vários núcleos e/ou vários processadores
- O número de sessões HTTP (ou número de usuários conectados)
- O número de threads ativas (ou número de requisições HTTP correntes)
- O número de conexões JDBC ativas (ou número de requisições SQL correntes)
- O número de conexões JDBC usadas (ou número de transações SQL abertas); no caso onde não há fonte de dados (datasource com JDBC sem pool), ele é, de fato, o número de conexões JDBC abertas no pool de conexões
- Para cada contador de requisições (HTTP, SQL e possivelmente EJB, Spring, Guice, ações de jsf, ações de struts, páginas JSP):
  - O tempo médio em milissegundos
  - O percentual de erros do sistema (Este valor, para um contador, é a média para o último período de medição)
- E também em 'Outros gráficos', alguns dados que são globais para a JVM ou para o Sistema Operacional:
  - Se for Tomcat ou JBoss, o número de threads ativas em todo o servidor, o número de bytes recebidos por minuto e o número de bytes enviados por minuto pelo servidor (o número de bytes recebidos é geralmente 0 para a maioria das webapps)
  - O percentual de CPU para o coletor de lixo da JVM
  - O número de threads para a JVM
  - O número de classes carregadas pela JVM
  - O uso de memória não heap para a JVM
  - A memória física utilizada pelo Sistema Operacional
  - A memória swap utilizada pelo Sistema Operacional
  - A idade média para sessões HTTP atualmente válidas em minutos
  - Se Linux: a carga do sistema e o número de arquivos abertos pelo Sistema Operacional
  - The transactions per minute (more precisely, the number of opened jdbc connections per minute)

As horas nos gráficos para a dia corrente dependem da hora do servidor e do fuso horário (como definido pelo Sistema Operacional ou o especificado para o servidor de aplicação).



O link ' atualizar' atualiza a página e os gráficos. O link ' PDF' exibe o relatório no formato PDF para o Adobe Reader.

Se um servidor de coleta é usado para exibir a monitoração para diversos servidores em uma Farm ou em um Cluster, o gráfico da memória é o somatório das memórias de cada servidor, mas o percentual de CPU é entre 0 e 100 para todos os servidores, e o número de sessões HTTP é o somatório de sessões em cada servidor, bem como o número de threads ativas, o número de conexões JDBC ativas ou usadas, e o número de acessos por minuto ou o tempo médio em milissegundos.

#### Gráfico de Memória: caso de uso de um aplicativo sendo utilizado de forma intermitente (dia / noite, por exemplo)



A memória no gráfico aumenta rapidamente quando a aplicação é usada e aumenta lentamente, mas ainda aumenta, quando o aplicativo não é usado. Um gráfico com inclinações, como no exemplo acima, é normal mesmo se o aplicativo não for usado.

Em todos os casos, e se necessário, a memória é finalmente liberada, de uma só vez, pelo coletor de lixo (CL principal) e vai para um nível baixo, antes de voltar a subir rapidamente ou lentamente. Se o conjunto de ações é suficientemente fino, o gráfico também mostra os CLs secundários como pequenas variações no gráfico entre os coletores de lixo principais (CL). Estes coletores de lixo secundários liberam, também alguma memória, mas em menor quantidade que os coletores de lixo principais (CLs principais). É possível forçar um coletor de lixo principal com a ação  'Executar o coletor de lixo' na parte  'Informações do sistema' do relatório.

#### Gráfico de memória: caso de uso de uma saturação da memória

Se a memória no gráfico chega ao máximo e se a aplicação não pode liberar um pouco de memória com o coletor de lixo, o servidor possivelmente causará algum "OutOfMemoryErrors: Java heap space" para interromper a execução e liberar alguma memória se possível.

Até que a memória seja suficiente, a CPU permanece em 100% porque o coletor de lixo é executado de forma permanente. Isso pode causar extrema lentidão ou um bloqueio do servidor de aplicativos. A solução pode ser incrementar o valor máximo para a memória java (parâmetro Xmx na configuração do servidor), e, possivelmente, a memória física do servidor ou otimizar, se possível, a memória utilizada pela aplicação.

#### Memória Perm Gen: caso de uso de uma saturação da memória Perm Gen

Se o valor da memória Perm Gen aumenta ao máximo e se a aplicação não consegue liberar um pouco da mesma, o servidor, possivelmente, causará alguns "OutOfMemoryErrors: Perm Gen space" interrompendo a execução da aplicação. Isso pode bloquear todos os recursos em sua aplicação.

As soluções podem ser incrementar a memória Perm Gen máxima (parâmetro XX:MaxPermSize na configuração do servidor, 64 MB por padrão), e, possivelmente, a memória física do servidor, ou reduzir, se possível, o número de classes carregadas pela aplicação.



#### Gráficos de threads ativas e conexões JDBC ativas: Caso de uso de nível contínuo plano (requisições longas)





Quando um aplicativo é pouco utilizado ou não é utilizado, os gráficos de threads ativas e o de conexões JDBC ativas devem ficar em 0. Na verdade, a maioria das requisições são, felizmente, rápidas e curtas e a maioria das medidas (realizadas a cada 2 minutos, por exemplo) são tomadas sem requisições ativas, exceto 1 ou 2 vezes ocasionalmente mostradas como um pico.

Mas, quando há um nível estável plano no gráfico de threads ativas, como no exemplo acima, isto significa que há uma ou várias requisições que são longas (vários minutos ou várias horas) e que podem, potencialmente, saturar o servidor de aplicativos ou banco de dados.

Para descobrir a causa:

- Se o nível estável de threads ativas é reproduzido exatamente no de conexões JDBC ativas, isso mostra que a causa está em uma ou várias requisições SQL e não na execução de código java.
- Se o nível plano continua correntemente, você pode procurar as requisições HTTP (e SQL) envolvidas na parte ' Requisições correntes' do relatório.
- Se o nível estável terminou, você pode procurar as estatísticas de requisições HTTP (e SQL) envolvidas, na tabela de detalhada de ' HTTP (e SQL)', exceto se o servidor de aplicação foi parado antes do fim destas requisições.
- Os dados exibidos no detalhe de uma solicitação HTTP podem ajudá-lo a encontrar a causa raiz nas façades ou nas requisições SQL.
- Se um banco de dados Oracle é usado, a exibição do plano de execução no detalhe da requisição SQL pode ajudá-lo a procurar a causa para o tempo gasto nesta execução. (se a exibição do plano de execução diz que uma tabela "plan\_table" deve ser criada: você pode usar o script @\$ORACLE\_HOME/rdbms/admin/catplan.sql como um usuário SYS, se v10g ou acima e caso contrário @\$ORACLE\_HOME/rdbms/admin/utlxplan.sql e "grant all on plan\_table to public" se v9i ou anterior).
- Pode ser útil para verificar com a monitoração o estado da memória e da CPU usado pelo servidor de aplicativos. Também pode ser útil para verificar o estado da memória, da CPU e do acesso ao disco do seu banco de dados.










#### Gráficos de conexões JDBC utilizadas: caso de uso de um vazamento de conexões JDBC


Em geral, as conexões JDBC são compartilhadas em um pool de conexões com uma fonte de dados (Data Source) JDBC, que permite, em particular, não reabrir as conexões à base de dados continuamente e que também define um número máximo de transações que podem ser executadas simultaneamente por instância do servidor de aplicativos. É possível, no JavaMelody, ver 2 gráficos de conexões JDBC ativas e conexões JDBC usadas. O gráfico de conexões JDBC ativas exibe o número de conexões executando consultas SQL. E o gráfico de conexões JDBC usadas exibe o número de conexões abertas que são usadas em transações com o banco de dados, mas não necessariamente executando consultas SQL. Se o aplicativo monitorado contém um vazamento de conexões JDBC, em alguma condição particular, as conexões são usadas desnecessariamente e permanecem indisponíveis fora do pool de conexões. O gráfico de conexões utilizadas pode mostrar que seu número aumenta sem diminuir ao mínimo configurado no pool de conexão da fonte de dados (datasource) mesmo se o servidor não tiver nenhuma atividade, à noite, por exemplo. Atingindo o máximo, os erros são registrados no pool de conexões, porque não há mais conexão disponível. Se há suspeita de vazamento de conexões JDBC, tempos e rastreamentos de pilha mostrando onde as conexões foram abertas podem ser visualizados pelo link ' Conexões JDBC abertas' em ' Informações do sistema' no relatório. (Este link estará disponível se o parâmetro 'system-actions-enabled' tiver sido configurado como 'true'). Uma leitura atenta do código para esses rastreamentos de pilha irá confirmar ou não o vazamento de conexões JDBC.

O gráfico de conexões JDBC utilizadas e os rastreamentos de pilha de conexões são de pouca utilidade se um pool de conexão é utilizado diretamente através de um driver JDBC e sem uma fonte de dados (datasource), porque, neste caso, você visualizará neste gráfico e nestes rastreamentos de pilha apenas as aberturas realizadas através do pool de conexões e não o uso real de conexões pela aplicação.

#### Estatísticas

As estatísticas de um contador mostram as estatísticas das requisições executadas no servidor de aplicativos. Hoje, existem 7 contadores de requisições:

-  Estatísticas Http
-  Estatísticas SQL (JDBC)
-  Estatísticas EJB (se EJB3 façades em JavaEE 5+)
-  Estatísticas Spring (se Spring façades)
-  Estatísticas Guice (se Guice façades)
-  Estatísticas de interfaces façades (sem EJB3, Spring ou Guice)
-  Estatísticas de ações jsf
-  Estatísticas de ações struts
-  Estatísticas de renderização de páginas JSP (ou, mais precisamente, de execuções de inclusões ou repasses em HttpServletRequest.getRequestDispatcher, que são as páginas JSP, em geral)

Para cada contador, uma requisição aparece nas estatísticas quando ela termina; para conhecer as requisições inacabadas veja a sessão ' Requisições correntes' no relatório. Quanto aos gráficos, as estatísticas das requisições são para um período pequeno ou grande, escolhido através dos links acima do gráfico: o dia (desde 0h), a semana, o mês ou o ano. Então, é possível ver as estatísticas somente para o dia atual, ou por exemplo, para uma versão do aplicativo implantado há um mês sem as estatísticas das versões anteriores. E para cada contador, o relatório exibe o resumo de estatísticas para todas as requisições (global), das estatísticas que vão além do limiar de alerta de nível 1 (warning) daqueles que vão para além do limiar de alerta de nível 2 (severe). Os detalhes das estatísticas para cada requisição são exibidas clicando no link '+ Detalhes'.

Cada estatística da requisição exibe:

- o conteúdo da requisição
- um gráfico de evolução para os tempos médios dessa requisição como uma tooltip (dica) para o mesmo período das estatísticas; podemos ampliar este gráfico clicando no conteúdo da requisição
- o percentual de tempo acumulado(tempo médio \* número de acessos) comparado com outras requisições
- o número de acessos (ou número de execuções)
- o tempo médio em milissegundos
- o tempo máximo em milissegundos
- o desvio padrão entre os tempos de execução: se ele é alto, os tempos de execução são variados, e se ele é baixo, os tempos de execução são próximos uns dos outros
- o percentual acumulado de tempo de CPU (tempo médio CPU \* número de acessos) comparado com outras requisições.
- o tempo médio de CPU em milissegundos
- o percentual de erros de sistema

e se é um HTTP, EJB, Spring, Guice interfaces, ações jsf, ações struts ou estatística JSP:

- o tamanho médio em kilo-bytes para a o fluxo de resposta (somente para HTTP)
- o número médio de acessos SQL , por execução da requisição HTTP(, EJB...)
- o tempo médio de execução de requisições SQL, por execução da requisição de HTTP(, EJB...)

Os tempos médios, os tempos máximos e os tempos médios de CPU são cumulativos: por exemplo, os tempos médios de HTTP incluem o tempo médio de EJB e o tempo médio de SQL; além disso, os tempos médios de CPU para o HTTP incluem os tempos médios de CPU para o EJB e os tempos médios de CPU para o SQL. Mas os tempos de espera como paradas ou aguardando à espera de I/O não consomem CPU, assim, eles são incluídos nos tempos médios, mas não em tempos médios de CPU.

Como em todas as tabelas do relatório, as tabelas de estatísticas pode ser ordenadas em ordem ascendente ou descendente, clicando no cabeçalho das colunas.

As estatísticas são mantidas para cada contador: um reinício do servidor da aplicação não tem qualquer influência sobre eles.

Se um servidor coletor é usado para exibir o monitoramento de vários servidores em uma Farm ou em um Cluster, as estatísticas dos contadores são globais para todos os servidores.

Nota: No MS Windows XP (antes Vista), a resolução do clock utilizado está em torno de 16 ms (ou 0, ou 16 ou 32 ms). Assim, o tempo de uma execução não é preciso abaixo de 16 ms com o Windows XP. Isto é em parte corrigido pelo uso de uma média quando há execuções suficientes. Mas isso não é grave porque, em geral, as requisições problemáticas terão um prazo de execução muito maior do que 16 ms.

#### Estatísticas de erros HTTP no sistema

As estatísticas de erros HTTP no sistema mostram os erros que voltaram para o filtro HTTP do monitoramento, seja por exceções geradas pela aplicação, através de um servlet, seja por um código de erro HTTP (404 "não encontrado" ou "500 Erro Interno do Servidor", por exemplo, [Lista completa](#)) Estas estatísticas exibem a lista dos 250 erros mais frequentes do sistema com o número de casos de cada erro no período escolhido, e com o tempo médio da requisição para cada erro. Somente o erro mais frequente é apresentado inicialmente, os outros erros serão exibidos clicando no link '+ Detalhes'. As datas, horários, os usuários e as solicitações HTTP completas dos últimos 100 erros são exibidos clicando no link '+ Últimos erros'. Ao clicar sobre o nome de um erro, é possível ver para este erro, o rastreamento da pilha do java, quando ele é uma exceção java.

Estas estatísticas de erros permitem melhorar a confiabilidade do aplicativo com base em seu uso real na produção.

#### Estatísticas de erros do sistema

As estatísticas dos logs de erros do sistema exibem 'alertas' e 'erros' produzidos pelo aplicativo (as bibliotecas log4j do Apache, logback quanto a java.util.logging do JDK são suportadas). Estas estatísticas exibem a lista dos 500 erros mais frequentes registrados com o número de casos de cada erro ocorrido no período escolhido. Apenas o log do erro mais frequente é exibido inicialmente, os outros registros no log serão exibidos, clicando no link '+ Detalhes'. As datas, horários, os usuários e, eventualmente, as requisições completas dos últimos 100 erros HTTP dos logs são exibidas clicando no link '+ Últimos erros'. Ao clicar sobre o nome de um log de erro, é possível ver, para este log, o rastreamento da pilha do erro quando uma exceção java tiver sido escrita no log.

Estas estatísticas de erros registradas no log permitem melhorar a confiabilidade da aplicação com base no seu uso real na produção.

#### Requisições correntes

As requisições correntes exibem, no momento da geração do relatório, as execuções das requisições que não terminaram.

A árvore de solicitações HTTP, EJB possivelmente, Spring e/ou SQL é exibida, para cada uma das requisições, juntamente com o tempo já decorrido, o tempo gasto de CPU, o número de requisições SQL já executadas e o tempo SQL destas requisições.












As estatísticas globais das requisições são exibidas novamente para as requisições correntes: tempo médio, a média de tempo de CPU, média de acessos SQL e tempo médio de SQL. Isso permite

comparar as requisições correntes com as estatísticas globais.  
O rastreamento da pilha do erro atual do java é exibido como uma dica na thread.  
Somente a maior requisição é apresentada inicialmente, as outras são exibidas clicando no link '+ Detalhes'.

## Informações sobre o sistema

As informações sobre o sistema são exibidas no momento da geração do relatório com algumas informações sobre o servidor de java, seu estado e do sistema operacional do servidor.  
As principais informações apresentadas, inicialmente, são: memória java usada, número de sessões HTTP, número de threads ativas, número de conexões JDBC ativas e número de conexões JDBC usadas. Estes são os mesmos valores que são usados para os gráficos.  
As outras informações, tais como a versão do servidor, do sistema operacional ou do banco de dados ou a memória do sistema operacional, são exibidas clicando no link '+ Detalhes'.

Nesta parte das informações do sistema, alguns links permitem executar ações do sistema:

-  'Executar o coletor de lixo', para forçar a liberação de memória
-  'Gerar um heap dump', para despejar todo o conteúdo da memória em um arquivo do diretório temporário no servidor, que pode ser aberto com o [VisualVM](#) do JDK ou com o [Eclipse MAT](#)
-  'Ver histograma da memória', para mostrar o número de instâncias na memória para cada classe java
-  'Invalidar sessões HTTP', para forçar todos os usuários a desligar
-  'Ver sessões HTTP', para ver os atributos, os tamanhos serializados (geralmente maiores que os tamanhos em memória), o país e, possivelmente, o usuário de cada sessão (se houver autenticação por JavaEE)
-  'Ver descritor de publicação': arquivo web.xml da aplicação
-  'MBeans': gerenciamento de beans, como no console ou [jvisualvm](#), com alguma configuração e sobre dados técnicos do servidor de aplicação e a JVM. Os valores são visíveis, mas não alteráveis e as operações não podem ser executadas.
-  'Ver os processos do SO': lista de processos do sistema operacional (Linux com [ps](#) ou Windows com tasklist) e para cada processo: usuário, memória e CPU
-  'Árvore JNDI': Procura o contexto JNDI, por exemplo, para encontrar a localização e o nome de uma fonte de dados JDBC.
-  Para ajudar a localizar possíveis vazamentos de conexões JDBC, o 'Conexões JDBC abertas' exibe a lista de rastreamentos da pilha de onde foram abertas as conexões JDBC.
-  'Banco de dados': Informações e estatísticas sobre o banco de dados (se for PostgreSQL, MySQL, Oracle, DB2, Sybase ou H2), como, por exemplo, as requisições SQL correntes ou se for Oracle, as requisições mais longas em tempo acumulado com exibição do tempo de CPU e do custo básico (de buffer gets). Esta função irá mostrar um erro se o usuário configurado no aplicativo para se conectar ao banco de dados não tiver os direitos necessários. No Oracle, a seguinte SQL concede esse direito: "grant select any dictionary to myuser".

Se um servidor coletor é usado para exibir a monitoração de vários servidores em uma Farm ou em um Cluster, as informações do sistema de cada servidor são exibidas (agregadas para a lista de sessões e histograma de memória) e as ações são executadas em cada servidor, um de cada vez.

## Threads

Esta tabela exibe a lista de todas as threads no servidor, com a prioridade de cada uma, o estado e a pilha de rastreamento em uma tooltip, entre outras informações.

## Caches de dados

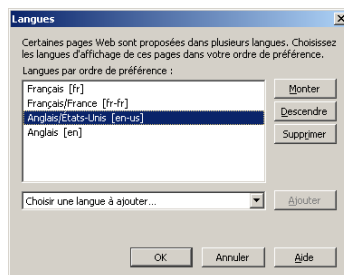
Os caches de dados exibem a lista de caches no aplicativo java, desde que a biblioteca [ehcache](#) seja utilizada por eles.  
Para cada um dos caches, algumas estatísticas são exibidas: o número de objetos em memória e em disco no momento, a eficiência da memória cache (percentual desde o início do servidor de acessos ao cache de memória em comparação com os acessos ao cache em disco), a eficiência global do cache (percentagem do cache de memória ou cache de disco comparado com todos os acessos ao cache, incluindo dados não presentes no cache). A configuração de cada cache também é exibida para obter informações.  
Since ehcache v2.1, the caches statistics are not enabled by default. So you need to enable them by adding statistics="true" in all the configurations of the caches, in order to display the efficiency values of the caches.  
Um botão está disponível para limpar todos os caches.

## Jobs

Os jobs exibem a lista de tarefas (ou lotes) no aplicativo java, desde que a biblioteca [quartz](#) seja utilizada por eles.  
Para cada job, o tempo médio, o último erro com rastreamento da pilha para um período escolhido, e os tempos da execução anterior e da próxima execução são exibidos. Como nos pedidos HTTP, as estatísticas de trabalhos são exibidas com tempos médios, o tempo médio de CPU, acessos SQL e tempos de SQL para o período selecionado.  
Botões estão disponíveis para fazer uma pausa ou retomar um ou todos os jobs.  
Se você tem jobs Quartz agendados com Spring, mas você não vê job algum no relatório, você adicionou a propriedade "exposeSchedulerInRepository", como diz o guia do usuário?

## Linguagem

O monitoramento é exibido em Francês, Inglês ou Português Brasileiro dependendo da linguagem preferida em seu browser. Se o monitoramento está em língua francesa ao invés de estar no idioma Português, você deve modificar o idioma de preferência. Por exemplo, em [Firefox](#), abra o menu Editar e "Ferramentas, Opções, Conteúdo, Idiomas, ... Selecionar" e colocar "Português/Brasil [pt-br]" em primeiro lugar:



## Créditos de ícones

[Silk, mini and flag icons \(Creative Commons\)](#)

[Tango icons \(GPL\)](#)