# WebOTX プロファイラを使用した
# メモリリーク調査方法

改版履歴

2006 年 10 月　初版

# 目次

# 1. はじめに

アプリケーションの問題のひとつとして、メモリリークがあります。メモリリークが発生し、不要となったオブジェクトがメモリ中に残り続けることによって、アプリケーションが動作に必要なメモリ量を確保できずにシステムダウンを引き起こす可能性があります。

メモリリークが発生している箇所を特定する方法としては、GC(Garbage Collection)発生時に不要となったオブジェクトが正しくメモリから削除されているかを確認する必要があります。

そこで WebOTX V6.4 では、GC で正しくオブジェクトが削除されているかを特定するためのツールとして、WebOTX プロファイラを提供しています。WebOTX プロファイラは、JavaVM 中に存在しているオブジェクトのメモリ使用量やインスタンス数、メソッドのメモリ使用量を取得することができます。WebOTX プロファイラを利用することによって、メモリリークを起こしている箇所の特定がしやすくなります。

### 対象読者

本ドキュメントは、WebOTX V6.4 上で動作させる Web アプリケーションのモニタを行うユーザを対象としています。

### 対象 WebOTX Version

WebOTX V6.4

### 関連資料

・WebOTX マニュアル運用編　プロファイラ操作ガイド
・WebOTX マニュアル運用編　運用管理ツール
・WebOTX マニュアル運用編　運用操作
・WebOTX マニュアル　運用管理コマンドリファレンス

# 2. OutOfMemoryError の確認

WebOTX に配備した Web アプリケーションが無応答になる原因が Web アプリケーションのメモリリークにあるかを調査するにあたり、WebOTX のログを確認します。確認する WebOTX のログファイルは、<WebOTX インストールディレクトリ>/domains/＜ドメイン名＞/logs 配下に出力されている server.log、webotx_agent.log です。webotx_agent.log に以下のようなログが出力されている場合は、メモリが足りずに OutOfMemoryError が発生しています。

| |
|---|
| ERROR com.nec.webotx.enterprise.system.container.web - StandardWrapperValve[HelloServlet]: サーブレット HelloServlet の Servlet.service()が例外を投げました。<br>java.lang.OutOfMemoryError |

*1　HelloServlet は配備した Web アプリケーションの名前です。

しかし、これだけでは OutOfMemoryError が発生した理由がアプリケーションのメモリリークであると確定できません。WebOTX で確保するヒープのサイズが足りなかったために、アプリケーション実行時に必要な分のメモリ量を確保できずに OutOfMemoryError が発生したということも考えられます。

次章以降で OutOfMemoryError が発生した原因の調査方法について説明します。

# 3. JavaVM のヒープ使用量の確認

まず、運用管理コマンド、または運用管理ツールを利用して、ドメインのメモリ使用量の変化を確認します。ここでは、測定、操作を行うドメイン名を domain1、ユーザ ID を admin、パスワードを adminadmin、ポートを 6212、環境は localhost とします。また、アプリケーションは予め配備されているとします。

・運用管理コマンドを利用する場合

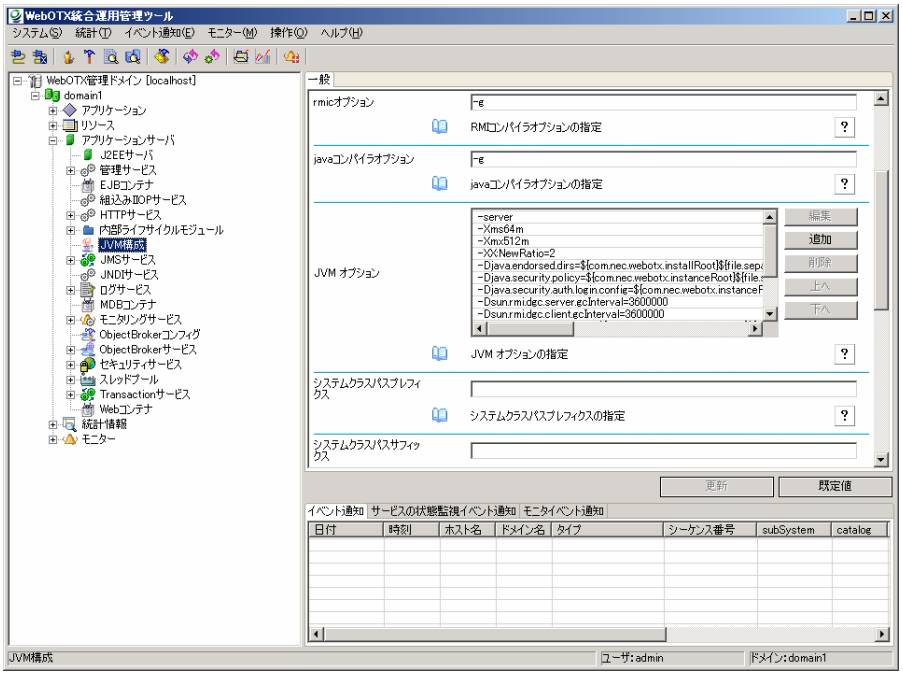
1. ドメインの起動を起動します。
   <WebOTX インストールディレクトリ>/bin/otxadmin start-domain domain1
2. ドメインにログインします。
   <WebOTX インストールディレクトリ>/bin/otxadmin
   otxadmin > login --user admin --password adminadmin --port 6212 --host localhost
3. get コマンドで JavaVM のオプションを確認します。
   otxadmin > get server.java-config.jvm-options
   JavaVM のオプションを確認し、最大ヒープサイズがアプリケーションを動作させるのに十分なメモリ量を確保しているか確認してください。
4. get コマンドで、ドメイン起動直後の JavaVM のメモリ使用量を確認します。
   otxadmin > get --monitor server.jvm. HeapSize-Current
   上記のコマンドを実行すると以下のようにドメインのプロセスで使用しているメモリの使用量が表示されます。単位は byte です。
   例：server.jvm.HeapSize-Current = 65011712

5. 配備したアプリケーションを実行します。
   アプリケーションを実行することによるメモリ使用量の変化を確認します。しばらく動作させたあと、4 の get コマンドを実行し、現在のドメインの JavaVM のヒープ使用量を確認してください。

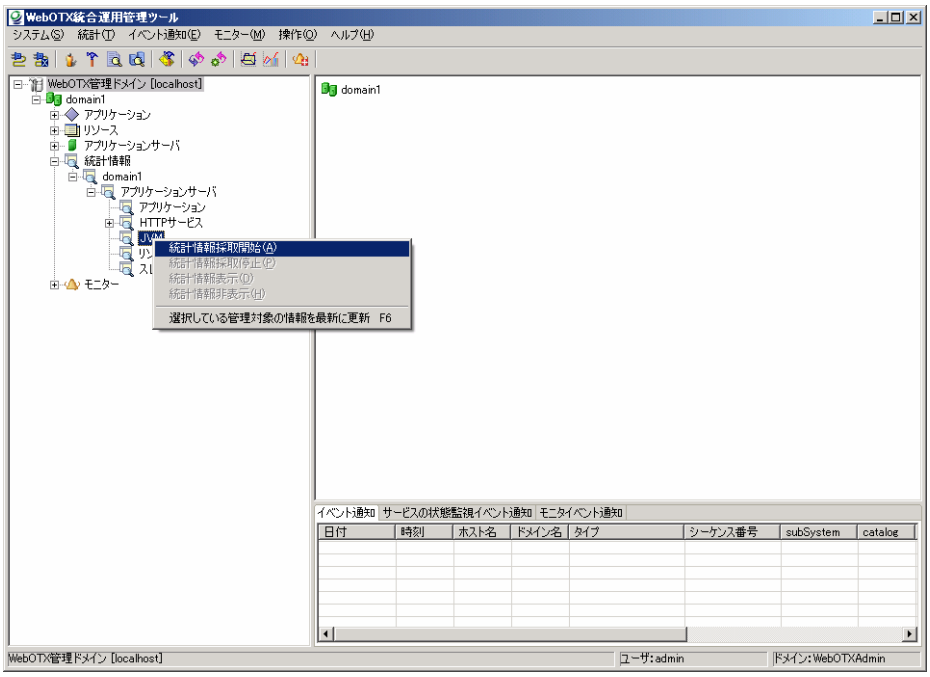
5 の手順を繰り替えし行い、HeapSize-Current の値が徐々に増加する場合は、アプリケーションがメモリリークを起こしている可能性が高くなります。

・運用管理ツールを利用する場合

1. ドメインの起動を起動します。
   <WebOTX インストールディレクトリ>/bin/otxadmin start-domain domain1
2. 運用管理ツールを起動し、ドメインに接続してください。
   ・Windows の場合
   [スタート]-[すべてのプログラム]-[WebOTX]-[運用管理ツール]
   ・Unix/Linux の場合
   /opt/WebOTX/bin/admingui
3. JavaVM が確保しているメモリ量を確認します。
   [アプリケーションサーバ]-[JVM 構成]を選択して、JVM オプションを確認してください。JavaVM のオプションを確認し、最大ヒープサイズがアプリケーションを動作させるのに十分なメモリ量を確保しているか確認してください。

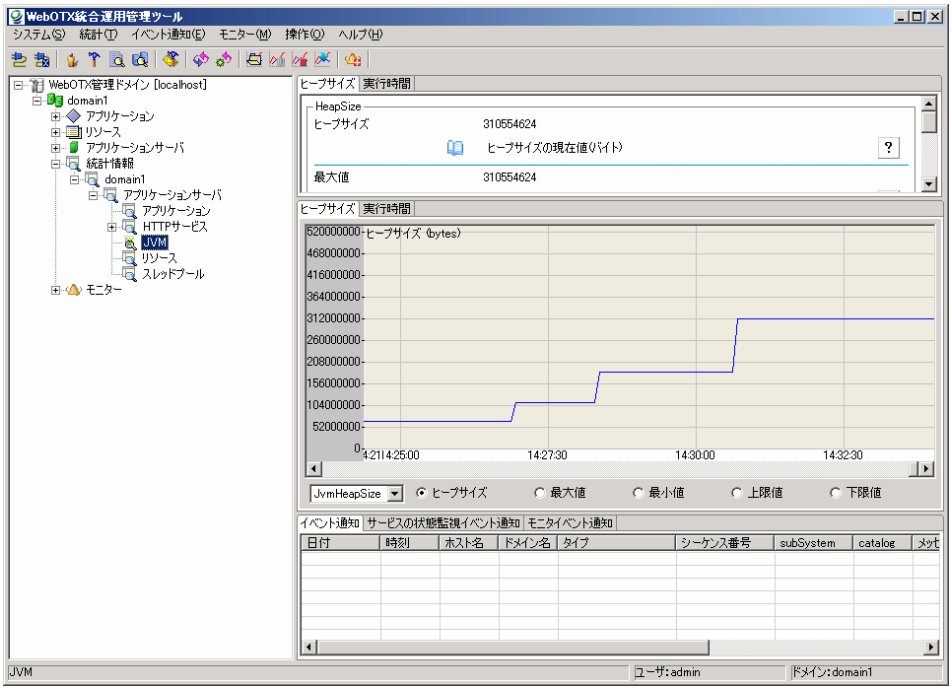



4. JavaVM のヒープ使用状況の確認をします。
   [統計情報]-[domain1]-[JVM]を右クリックし、[統計情報採取開始]を選択します。




5. 配備したアプリケーションを実行します。

   アプリケーションを実行することによるメモリ使用量の変化を測定します。しばらく動作させたあと、4 で表示させた、現在のドメインの JavaVM のヒープ使用量をグラフの変化を見て確認してください。




   アプリケーションを連続して実行し続けた場合、メモリの使用量が増加し続けるならばアプリケーションがメモリリークを起こしている可能性が高くなります。

## 4.WebOTX プロファイラの利用

3 章でアプリケーションがメモリリークを起こしている可能性が高いことを確認できました。そこで、メモリリークが起きている可能性があるクラス、メソッドを調査するために、V6.4 で提供している WebOTX プロファイラを利用します。

### 4.1.WebOTX プロファイラを利用するための設定

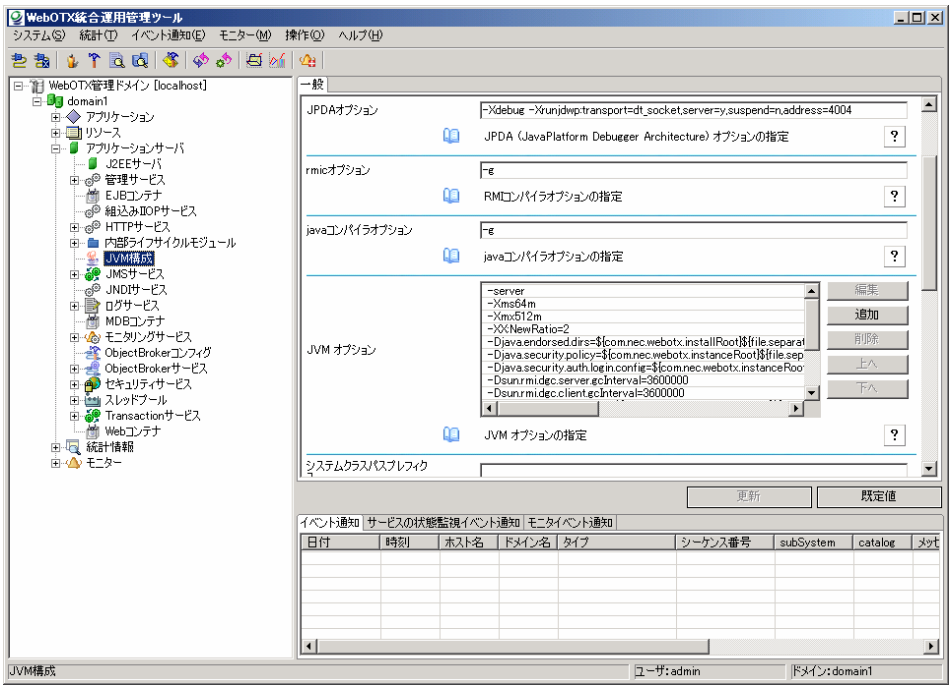
#### 4.1.1. ドメインの JavaVM オプションにプロファイラのオプションの追加

・運用管理コマンドから行う場合

1. ドメインを起動してください。
   <WebOTX インストールディレクトリ>/bin/otxadmin start-domain domain1
2. ドメインの JavaVM のオプションの設定を行います。
   <WebOTX インストールディレクトリ>/bin/otxadmin
   otxadmin > login --user admin --password adminadmin --port 6212 --host localhost
   otxadmin > create-jvm-config “-Xrunwoprofiler”
3. ドメインを停止してください。
   <WebOTX インストールディレクトリ>/bin/otxadmin stop-domain domain1

・運用管理ツールから行う場合

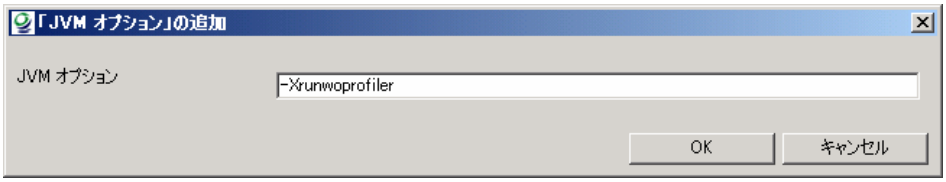
1. ドメインを起動してください。
   <WebOTX インストールディレクトリ>/bin/otxadmin start-domain domain1
2. 運用管理ツールを起動し、ドメインに接続してください。
   ・Windows の場合
   [スタート]-[すべてのプログラム]-[WebOTX]-[運用管理ツール]
   ・Unix/Linux の場合
   /opt/WebOTX/bin/admingui
3. ドメインの JavaVM のオプションの設定を行います。
   [domain1]-[アプリケーションサーバ]-[JVM 構成]を選択します。




JVM オプションの[追加]ボタンを押します。

[JVM オプションの追加]に以下のように入力してください。

-Xrunwoprofiler




運用管理ツールから入力を行う場合はエスケープ文字を入力する必要ありません。

4. ドメインを停止してください。

<WebOTX インストールディレクトリ>/bin/otxadmin stop-domain domain1

### 4.1.2.測定する Java パッケージ名の指定

WebOTX プロファイラで取得する Java のパッケージ名の指定を行います。作成したアプリケーションのパッケージ名を指定することにより必要な情報のみを取得することができるようになります。取得するパッケージ名の指定は<WebOTX インストールディレクトリ>/config/profiler/profiler_server.properties で指定することができます。取得するパッケージを指定するには Include と Exclude のどちらかを利用します。

Include : モニタしようとするパッケージ名またはクラス名です。複数指定する場合、”;”(セミコロン)で区切ってください。 Include で指定したパッケージのみモニタを行います(Exclude よりも優先)。何も指定しない場合は Exclude で指定したパッケージ以外を取得します。

例 : Include=com.nec.sample;(com.nec.sample で始まるパッケージのみモニタを行います)

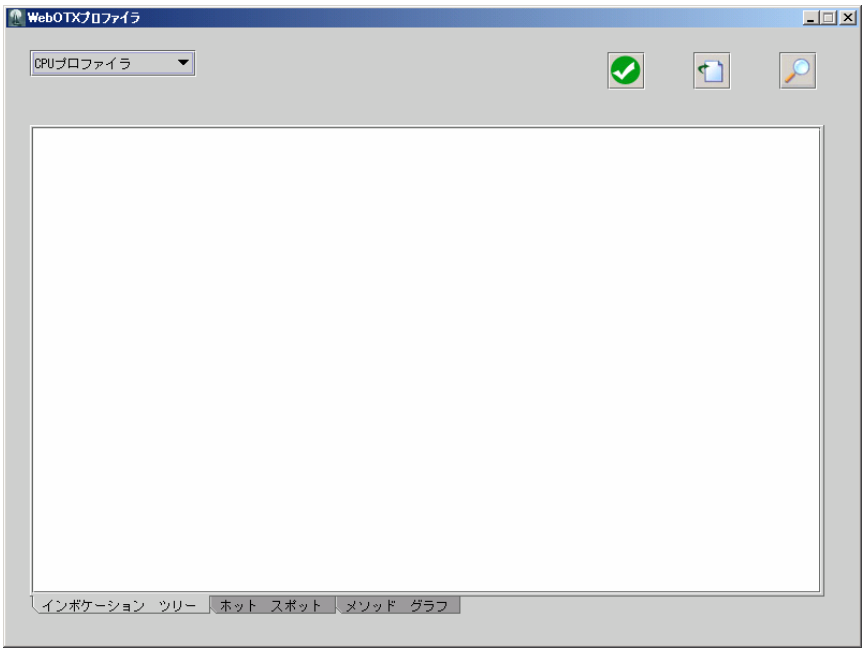
Exclude : モニタを行わないパッケージ名を指定します。複数指定する場合は、” ; ”(セミコロン)で区切って、複数指定してください。デフォルトでは、以下のパッケージが Exclude に指定されています。必要に応じて追加してください。
Exclude=java.;javax.;sun.;org.;com.nec.webotx.;jp.co.nec.orb.;jp.co.nec.WebOTX.;jp.co.nec.WebOTX_S.;jp.co.nec.wojdbc.

なお、本例では Include に com.nec.sample を指定しています。

## 4.2.アプリケーションのメモリ使用量の測定

4.1 でドメインの JavaVM のオプションにプロファイラを利用するオプションの指定と、モニタを行うパッケージの指定を server.properties に行った後、アプリケーションのメモリ使用量の測定を行います。

1. WebOTX プロファイラクライアントを起動します。
   <WebOTX インストールディレクトリ>/bin/woprofiler_client.bat




2. ドメインを起動します。
   <WebOTX インストールディレクトリ>/bin/otxadmin start-domain domain1
   ドメイン起動時に WebOTX プロファイラとプロファイラクライアントの接続を行います。プロファイラのログ(<WebOTX インストールディレクトリ>/logs/webotx_profiler_dll.log)に以下の情報が出力されることを確認してくださ

い。

| | |
|---|---|
| INFO: | connected to Socket Server. |
| INFO: | WebOTX Profiler is on. |

3. モニタを開始します。

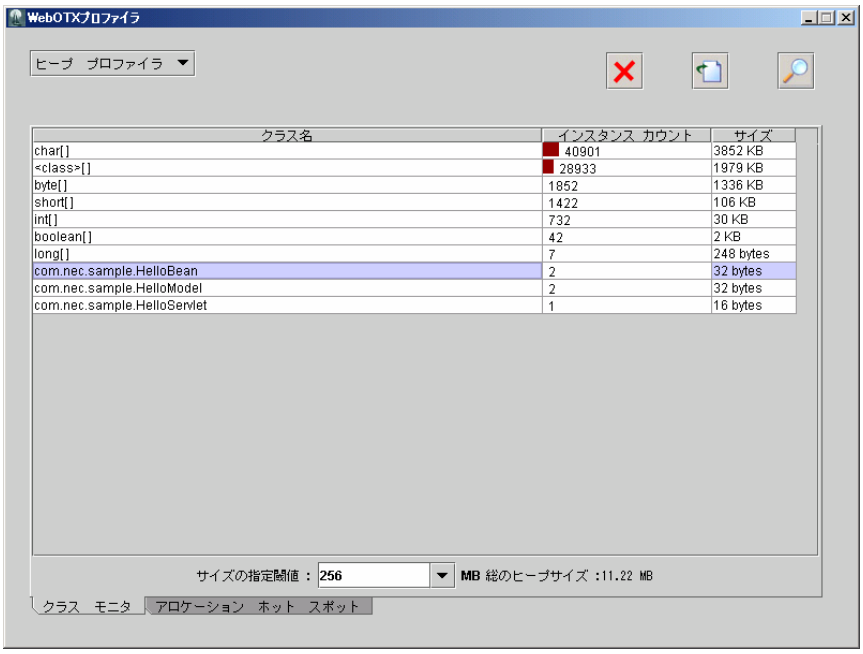
   ドメインの起動が完了した後、プロファイラクライアントの  ボタンを押してください。モニタが開始されます。

4. ヒープの情報を表示させます。
   プロファイラクライアントの左上のメニューから[ヒーププロファイラ]を選択します。

5. アプリケーション動作時の JavaVM のヒープ情報を確認します。
   アプリケーションを動作させると、プロファイラクライアントに 5.1.2 節で指定した Java パッケージの情報が表示されます。この時点で表示される情報がアプリケーション動作開始時のメモリ使用量となります。
   表示される情報は以下の通りです。

| | |
|---|---|
| クラスモニタ | クラス名 |
| | インスタンス数 |
| | サイズ(メモリ使用量) |
| アロケーションホットスポット | メソッド名 |
| | 割当比率 |
| | サイズ(メモリ使用量) |

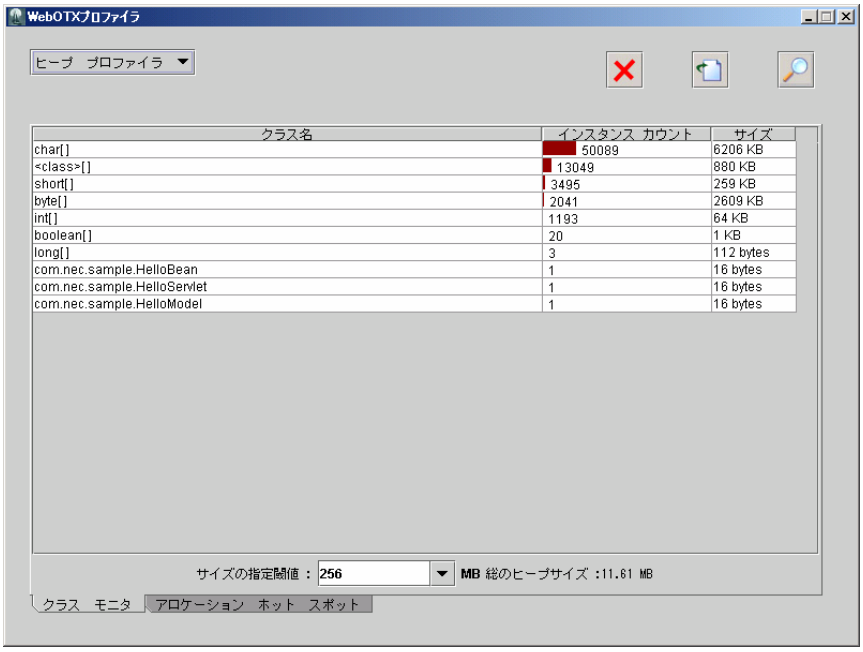
   クラスモニタ、アロケーションホットスポットともに、メモリサイズが大きい順にソートされています。




4.1.2 で Include に指定した com.nec.sample から始まるパッケージのみを表示しています。

6. アプリケーション連続動作時のインスタンス数、メモリ使用量を確認します。
メモリリークが発生している箇所を特定する場合、クラスのインスタンス数、メモリサイズ、メソッドのメモリサイズの変化を確認してください。

(a) アプリケーション実行開始時



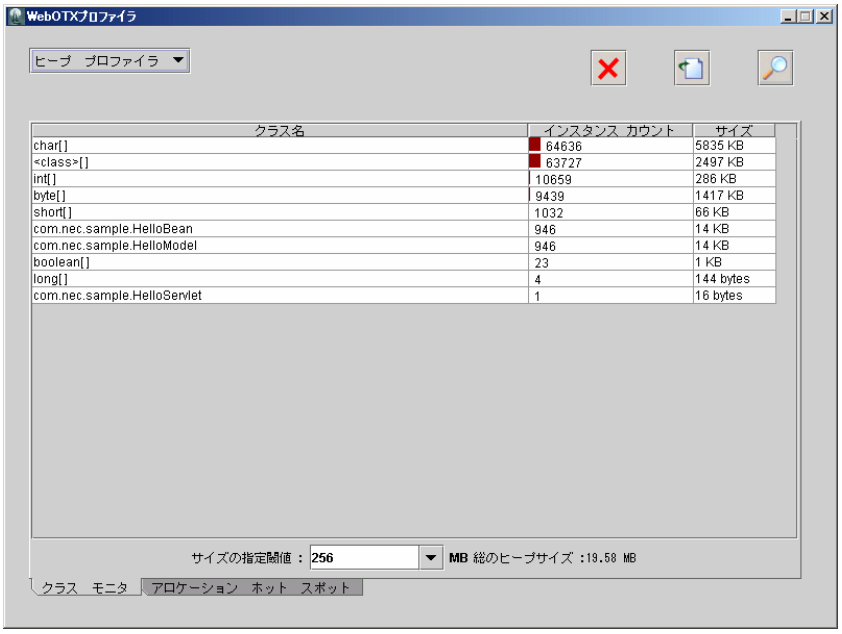

アプリケーション実行開始時にはcom.nec.sampleで始まるパッケージのメモ

リ、インスタンス数は以下のようになっています。

パッケージ名：com.nec.sample

| クラス名 | インスタンスカウント | サイズ |
|---|---|---|
| HelloBean | 1 | 16bytes |
| HelloModel | 1 | 16bytes |
| HelloServlet | 1 | 16bytes |

(b) アプリケーション実行中



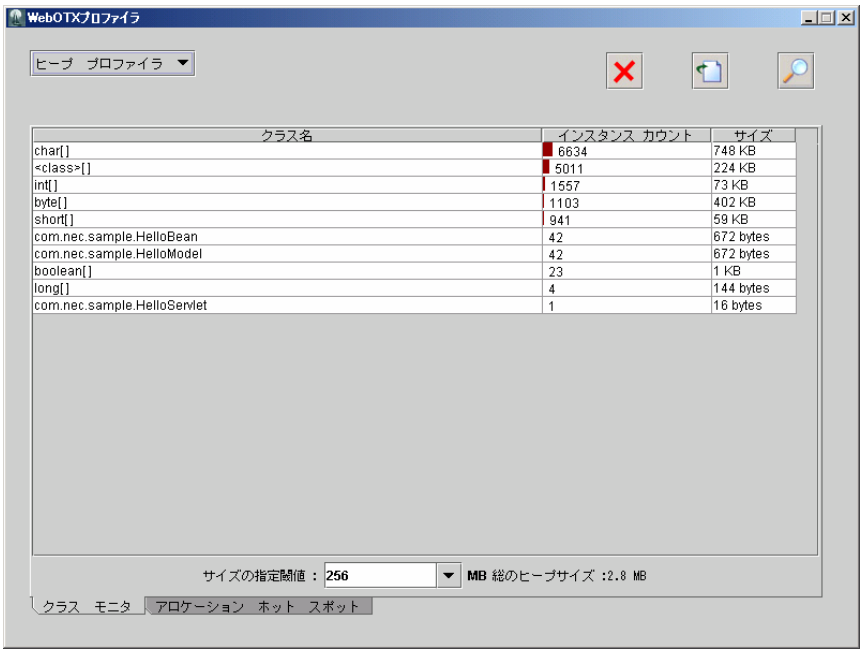

アプリケーション実行中は、HelloMode、HelloBean が HelloServlet から呼び出されているためインスタンス数、メモリサイズともに増加しています。

パッケージ名：com.nec.sample

| クラス名 | インスタンスカウント | サイズ |
|---|---|---|
| HelloBean | 946 | 14KB |
| HelloModel | 946 | 14KB |
| HelloServlet | 1 | 16bytes |

(c) GC 発生時

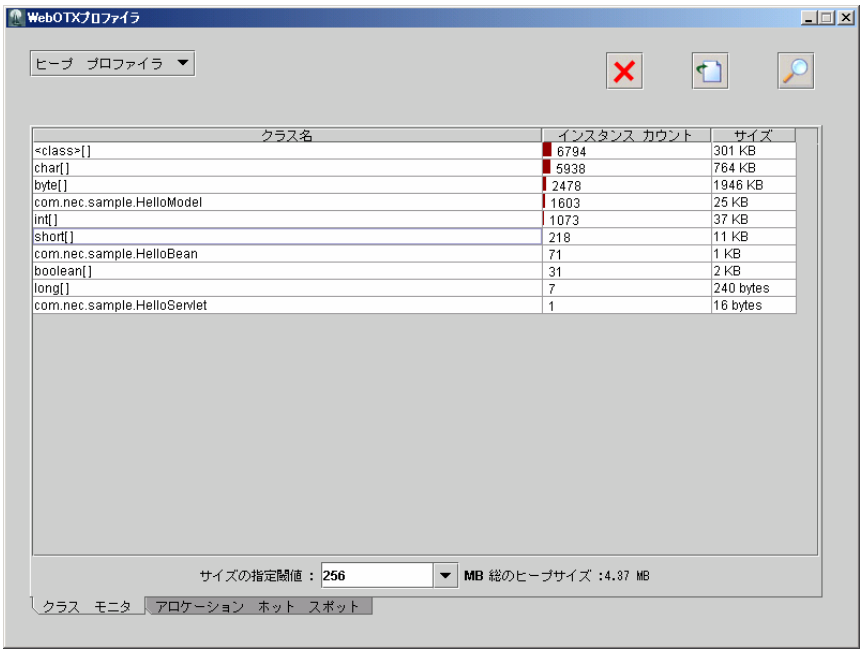
GC で正しくメモリが解放されている場合は以下の図のようにインスタンス数、メモリサイズが減少します。




GC によって、HelloBean、HelloModel のインスタンスがメモリ上から解放されるため、GC 発生前よりもインスタンス数、メモリサイズともに減少します。

| クラス名 | インスタンスカウント | サイズ |
|---|---|---|
| HelloBean | 42 | 672bytes |
| HelloModel | 42 | 672bytes |
| HelloServlet | 1 | 16bytes |

反対に、GC で正しくメモリが解放されていないクラス、メソッドは次の図のようにインスタンス数、メモリサイズが変化します。




GC が発生することによって、HelloBean のインスタンス数、メモリサイズは減少していますが、HelloModel は GC によってインスタンス数、メモリサイズの減少はありませんでした。よって、HelloModel オブジェクトを参照し続ける処理がプログラム中にあり、GC によってメモリから解放されないと考えられます。

| クラス名 | インスタンスカウント | サイズ |
|---|---|---|
| HelloBean | 1603 | 25KB |
| HelloModel | 218 | 8KB |
| HelloServlet | 1 | 16bytes |

・原因の箇所の特定

HelloModel の参照を行っている箇所の特定を行います。クラス名の com.nec.sample.HelloModel を右クリックすることによって、HelloModel を[インスタンシングする全てのメソッド]を表示することができます。


この結果によって HelloModel は com.nec.sample.HelloServlet の doPost メ

ソッド内で呼び出されていることが分かります。HelloModel が GC によってメモリ上から削除されない原因は、doPost 内の処理に原因があると考えられます。HelloServlet の doPost メソッド内の処理を見直し、HelloModel オブジェクトの参照が不正に残っていないかどうかを調査してください。

# 5.おわりに

WebOTX プロファイラは、WebOTX の Web コンテナ上で動作させるアプリケーションがメモリリークを起こしている箇所を、発見するための手段として利用可能です。メモリ使用量の測定に加えて、WebOTX プロファイラはメソッド単位で CPU 使用率を測定することも可能ですので、アプリケーションのボトルネックとなっている箇所を調査するためのツールとしても大いに役立ててください。

注意

アプリケーションがメモリリークを起こし、OutOfMemoryError が webotx_agent.log に記録されている場合は、それ以降ドメインのプロセスが応答しない可能性があります。ドメインが応答しない場合は、Windows ではマシンの再起動を行ってください。Unix/Linux の場合、ドメインを再起動する前に、ps -ef | grep domain.name=domain1 で出力されたプロセスに対し kill コマンドを実行してください。その後、ドメインの起動を行ってください。

参考

WebOTX のドメインで確保するヒープの初期サイズ(-Xms)と、ヒープの最大サイズ(-Xmx)を決定する目安は、アプリケーション実行時間に対する GC の実行時間の占める割合が 20%をきるように設定するとよいでしょう。