git-hg-export

Description

`git-hg-export` exports a commit to standard out on a format suitable for `hg import`. git-hg-export will also reformat the commit message to the format that is used by OpenJDK Mercurial repositories. These two properties makes it easy to export a commit from an OpenJDK Git repository and import the patch into an OpenJDK Mercurial repository (with correct commit metadata such as author, timestamp and commit message preserved).

Usage

```
$ git hg-export -h
usage: git-hg-export [options] <REV>
        --verbose   Turn on verbose output
        --debug     Turn on debugging output
        --version   Print the version of this tool
    -h, --help      Show this help text
```

Examples

To export commit 6df465de7309e90bc4de8da66c7059035ffc9bef from the jdk repository:

```
$ git hg-export 6df465d
# HG changeset patch
# User stuefe
# Date 1596957602 -7200
#      Sun Aug 09:20:02 2020 +0200
8251257: NMT: jcmd VM.native_memory scale=1 crashes target VM
Reviewed-by: zgu, dholmes

diff --git a/src/hotspot/share/services/nmtCommon.cpp b/src/hotspot/share/services/nmtCommon.cpp
index 10a6190d783..b90981f5cd2 100644
--- a/src/hotspot/share/services/nmtCommon.cpp
+++ b/src/hotspot/share/services/nmtCommon.cpp
@@ -35,6 +35,7 @@ const char* NMTUtil::_memory_type_names[] = {

 const char* NMTUtil::scale_name(size_t scale) {
   switch(scale) {
+    case 1: return "";
     case K: return "KB";
     case M: return "MB";
     case G: return "GB";
diff --git a/test/hotspot/jtreg/runtime/NMT/JcmdScale.java b/test/hotspot/jtreg/runtime/NMT/JcmdScale.java
index 12a7f649e02..c0c08be0967 100644
--- a/test/hotspot/jtreg/runtime/NMT/JcmdScale.java
+++ b/test/hotspot/jtreg/runtime/NMT/JcmdScale.java
@@ -42,6 +42,14 @@ public static void main(String args[]) throws Exception {
     // Grab my own PID
     String pid = Long.toString(ProcessTools.getProcessId());

+    pb.command(new String[] { JDKToolFinder.getJDKTool("jcmd"), pid, "VM.native_memory", "scale=1"});
+    output = new OutputAnalyzer(pb.start());
+    output.shouldContain(", committed=");
+
+    pb.command(new String[] { JDKToolFinder.getJDKTool("jcmd"), pid, "VM.native_memory", "scale=b"});
+    output = new OutputAnalyzer(pb.start());
+    output.shouldContain(", committed=");
+
     pb.command(new String[] { JDKToolFinder.getJDKTool("jcmd"), pid, "VM.native_memory", "scale=KB"});
     output = new OutputAnalyzer(pb.start());
     output.shouldContain("KB, committed=");
```

Source

See GitHgExport.java.