Rappture Without Changing Application Source Code

Rappture is a programming tool kit to add a web browser window to a supercomputing application.

With traditional Rappture use, the tool maker changes all input and output in his or her source code to use Rappture function calls. It's unusual to use input files that are uploaded in the web browser.

This set of notes shows how you can upload script and data files within the Rappture window, wrap and run your executable tool without changing its source code. This example works for simple text files. The example also shows the use of the Submit tool. It includes a running example.

We wrapped an application called Lammps, a molecular/atomic simulator.

We created the following files to wrap Lammps and submit the job to a supercomputer:

- lmp_wrapper.tcl: creates local copies of the input files, calls submit.tcl or executes Lammps, and writes the log file to the window
- submit.tcl: execute Lammps on a remote system, possibly with parallelization
- tool.xml: the specification for the web page user interface
- new.xml: instructions for input files, kept in a subdirectory named examples
- lennardjones.xml: an example Lammps script, wrapped in xml, kept in a subdirectory named examples

tool.xml

When using Rappture, you typically create a file named tool.xml, which names the execution command. In tool.xml we instead call our tcl wrapper as follows:

```
<command>tclsh @tool/lmp_wrapper.tcl @driver</command>
```

You can see the use of the tcl shell, tcsh, which runs the tcl commands in lmp_wrapper.tcl. tool.xml also contains a file uploader.

There are two ways to get files to your application: Paste the content into a text area or use a loader to upload the file. (See
images below).

Figure 1. Paste file into text area.

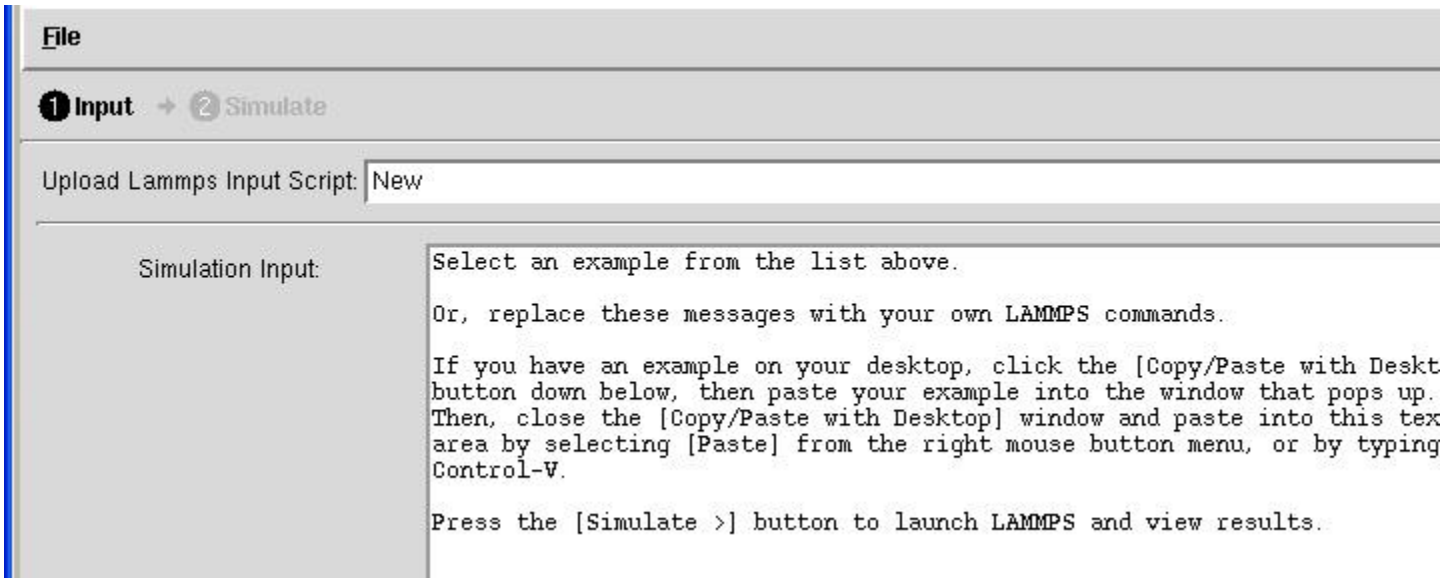
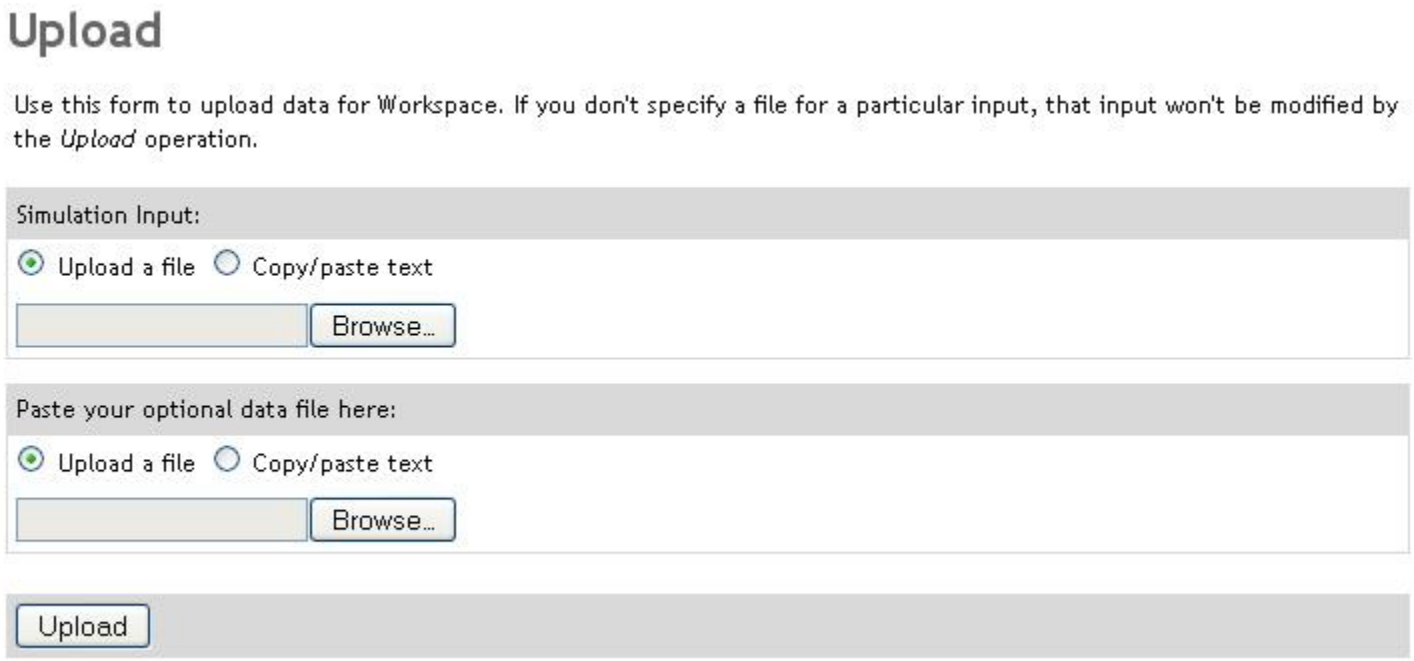


Figure 2. Upload files.



If you wish to upload multiple files, it is best to put all the files inside a single loader tag,e.g.

```
<loader>
  <about>
   <label>Upload Lammps Input Script</label>
   <description>Lammps Input Script</description>
  </about>
  <upload>
   <to>input.string(script)</to>
   <to>input.string(datafile)</to>
  </upload>
  <new>new.xml</new>
  <example>*.xml</example>
  <default>new.xml</default>
</loader>
```

"example" refers to a directory named **examples**. The directory contains new.xml and lennardjones.xml.

The uploaded files get written into input strings, in this case named script and datafile, e.g.

```
<string id="script">
  <about>
    <label>Simulation Input</label>
  </about>
  <size>80x24</size>
</string>
```

Note, only one text area can have a size specified. The others are single line.

lmp_wrapper.tcl

lmp_wrapper is a tcl script that gets input files from the Rappture web interface and writes them out locally, so the application can use it. Much of this code comes from basic Rappture string input/output and tcl file input/output. Lammps is executed
by calling the submit script, i.e.

```
source [file join $installdir submit.tcl]
```

More information about the submit command can be found at the HUBzero documentation page.

During testing, Lammps can be run locally instead:

```
set status [catch {eval Rappture::exec $lammpsExec -in script }
out]
```

submit.tcl

This file is a modification of the carefully annotated submit example found on the HUBzero documentation page.
submit.tcl literally creates a shell script and executes it.

The main difference between our file and the example is that the command arguments are the call to execute lammps, and our file allows submit to select the location for job submission instead of
us specifying it . Here you can see the shell script being written to do that.

```
# submit -v <cluster name>  specifies cluster to run on
append submitScript "submit -n $nodes -w $walltime \\\n"

# Put the command arguments below
append submitScript "          $lammpsExec -in script &\n"
```