

***NEW FEATURES OF
ZAGREUS 1.5.6.0***

Introduction

Zagreus 1.5.6.0 was released at the end of November, 2023. This version contains exciting new features and some updates of already existing actions.

With **z:execute** action Zagreus can execute scripts of various programming languages, like *Python*, *R*, *Powershell*, etc. The Zagreus worker module can be started as a **standalone command line application** and execute one specific script.

For the new features as mentioned above, a new feature is designed, the so called **worker-filesystem**, that allows the user to store files on the worker side filesystem. A dedicated **wfile action group** is created for managing file operations in the worker filesystem.

New Features of Zagreus 1.5.6.0

Worker filesystem and WFILE action group

Worker filesystem allows the user to store files on the worker side filesystem. The root of the worker-filesystem is located in the worker module root folder, `zagreus/worker-controller/worker/filesystem` subfolder by default. This location can be configured in the `worker.properties` configuration file. Worker filesystem is used also by the new **z:execute** action (see below).

WFILE action group was implemented for managing file operations in the worker filesystem, like the well known **file action group**, that operates in the filesystem of the Zagreus server.

Actions of the wfile action group:

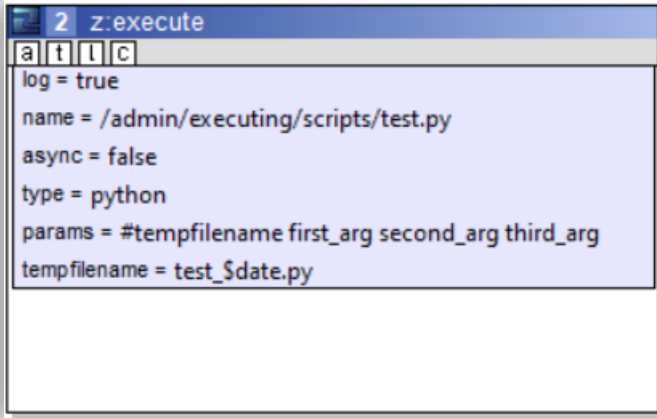
- **wfile:copy** - Copies a resource to a specified location in the worker file system.

- **wfile:delete** - Deletes the resource at the specified location in the worker file system.
- **wfile:dir** - Return the list of files in the folder at a specified worker file system folder or returns details if the target is a file on the worker file system.
- **wfile:exists** - Returns true value if the file or folder exists in the specified location in the worker file system.
- **wfile:move** - Moves a resource to a specified location in the worker file system.
- **wfile:read** - Reads data from a specified file on the worker file system.
- **wfile:write** - Writes its contents to a specified file on the worker file system.

Besides the new worker filesystem actions, the **worker-output** common attribute was created and it provides the same functionality as the **output** attribute, but it saves the result into the worker-filesystem.

z:execute

Executes the specified external script (e.g. python, R, Powershell) with the predefined executor binary.



```

2 z:execute
atl c
log = true
name = /admin/executing/scripts/test.py
async = false
type = python
params = #tempfilename first_arg second_arg third_arg
tempfilename = test_$date.py

```

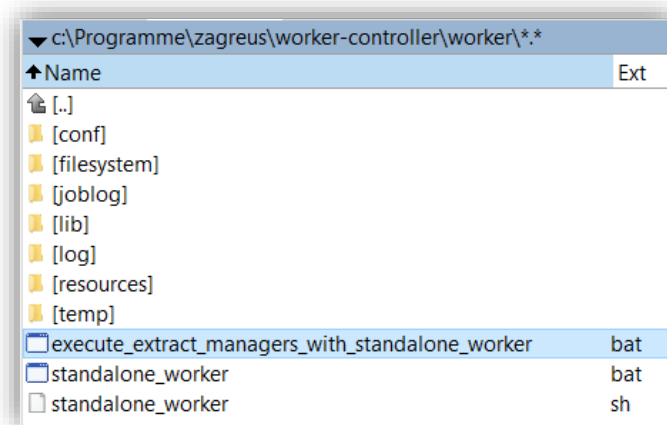
The content of the external script can be specified as a child element / child text

content, or by filling the id / name attributes. The external script is first saved in the worker filesystem temporary folder. Its name can be customized with the "tempfilename" attribute, otherwise it will be generated automatically with the current timestamp. For further detail, please visit our webpage and watch the demo video:

<https://support.etixpert.com/zagreus/version.php>

Standalone worker

The Zagreus worker module can be started as a **standalone command line application** and execute one specific script. The script that needs to be executed is defined as an xml file in the worker-filesystem.

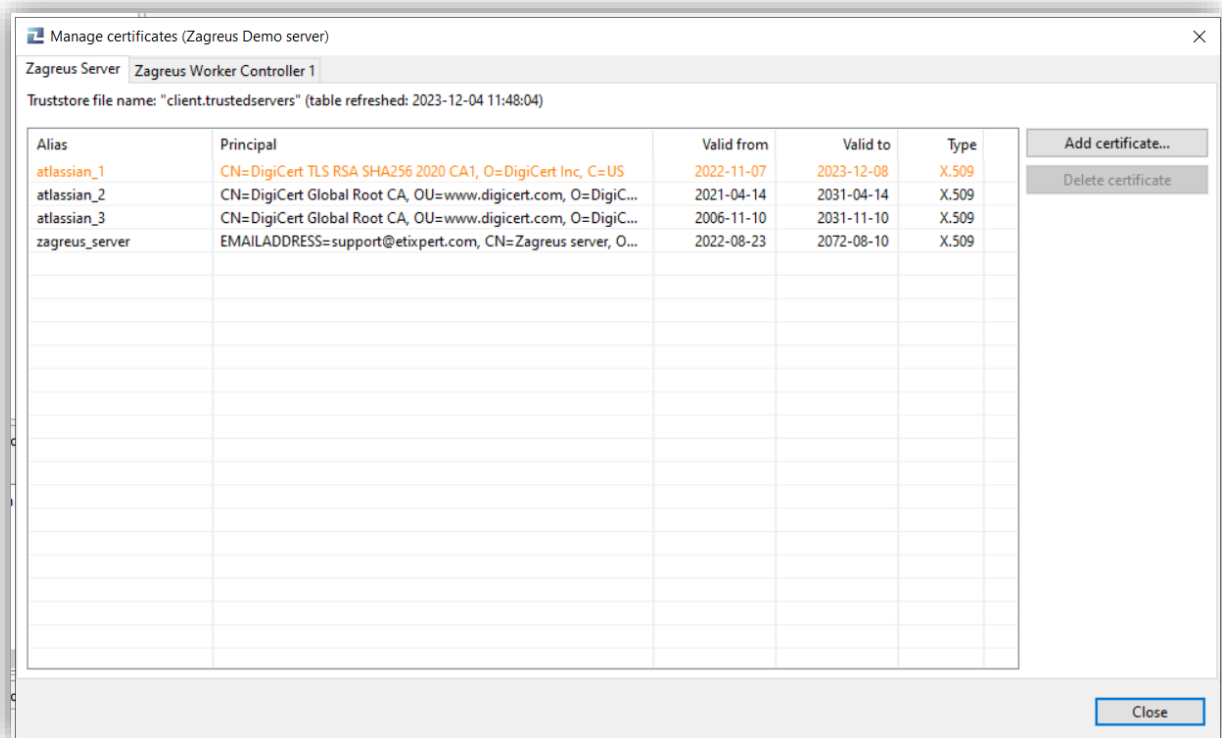
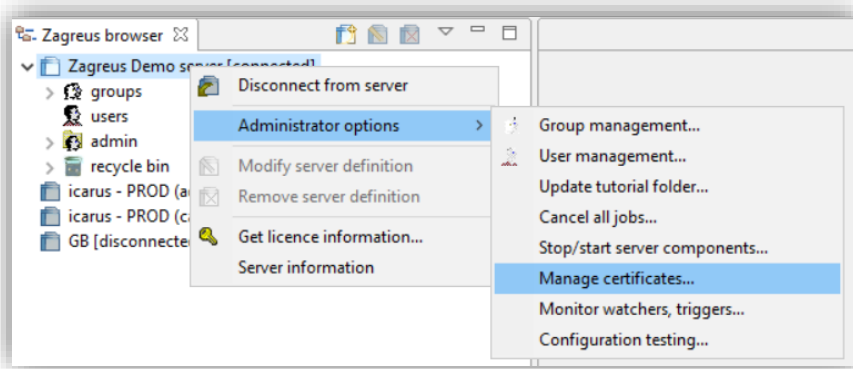


The example **execute_extract_managers_with_standalone_worker.bat** file refers to a Zagreus script called *extract_managers.xml* (stored in worker filesystem). When executing the .bat file, the job will create an Excel file in the filesystem that contains the managers listed in the input csv file. For further details about the example script, please visit our webpage and watch the demo video:

<https://support.etixpert.com/zagreus/version.php>

Certification maintenance in the GUI

The server and worker keystores can be maintained directly from the GUI (Server definition node context menu/Administrator options/Manage certifications...). After inserting new certificates, no restart is needed!



Further new features

- **exit and exit-message common attributes**
 - `exit=true/false` - when set to true, the execution will be stopped after that specific action - this simulates using the `z:exit` action
 - `exit-message`: the same as "message" attribute in `z:exit` action
- **result-encoding attribute** - It specifies the encoding of the server response is implemented for `rest:call` and `mstrrest:call` actions:
- **JSON beautifying** - When the result of an action (`rest:call`, `mstrrest:call`, `jira:call`, `confluence:call` and `json` action group actions) is json, it will be beautified.
- **action label default values** - When opening the label of a script, the action name is used by default.
- **result messages** are shown in the *Finished jobs* panel by default