3. How to structure tasks

Two commands are available for structuring a $\kappa\lambda\epsilon\iota\omega$ program as a series of tasks.

3.1 The `continue` command

The general form of a `continue` command is:

TARGet=<name of file on your computer> **(default: none)**
OVERwrite=No | Yes **(default: No)**

The `continue` command indicates that a defined task is to be performed, after which the system should expect to find further instructions.

3.1.1 The `target=` parameter

The value of the `target=` parameter normally consists of a constant corresponding to the name – or part of the name, identified more exactly by the appended parameter – of a file which has been correctly named according to the conventions of the operating system you are using.

All the results of the last task you formulated – i.e. without preceding commands – are written to the named file. If you specify an invalid filename (from the operating system's point of view), $\kappa\lambda\epsilon\iota\omega$ will *not* perform the task. The filename's full length (including path) is almost unlimited (4095 characters in $\kappa\lambda\epsilon\iota\omega$ as delivered). This means you can define long and complex paths in MS-DOS, UNIX, VMS and similar systems. However, we recommend that you only ever define *absolute* paths, as future versions of $\kappa\lambda\epsilon\iota\omega$ running on these systems will manage their own directories.

3.1.2 The `overwrite=` parameter

The `overwrite=` parameter expects one of the keywords `yes` or `no` as a value. Both keywords can be shortened to their respective first character.

- `no` instructs the system not to perform the task if the output file you specified already exists (default value).
- `yes` instructs the system to overwrite any existing file which has the same name as the output file you specified.

3.1.3 Example

A correctly written `continue` command looks like this:

```
continue target="result.5";overwrite=yes
```

3.2 The `stop` command

The `stop` command instructs $\kappa\lambda\epsilon\iota\omega$ to perform a given task and then exit.

The command accepts the same parameters as the `continue` command and interprets them in exactly the same way.

3.2.1 Example

A correctly written `stop` command looks like this:

```
stop target="result.3"
```