```
1   -- This file is part of a Liberty Eiffel library.
2   -- See the full copyright at the end.
3   --
4   expanded class PROCESS_FACTORY
5      --
6      -- This class allows one to spawn an external process and make it
       execute some file.
7      --
8      -- The standard streams of the process are available: `input',
       `output' and `error'.
9      --
10     -- '''Note:''' This class is in a beta stage. POSIX and Windows
       versions are available but there may be
11     -- resource leaks or other bugs left.
12     --
13
14  insert
15     ANY
16        redefine
17           default_create
18        end
19
20  create {ANY}
21     default_create
22
23  feature {ANY}
24     execute (program: STRING; arguments: TRAVERSABLE[STRING]): PROCESS
25           -- Execute the given `program' (how the program is
             discovered is OS-dependent), passing to it the
26           -- `arguments'. The environment is cleared if
             `keep_environment' is False.
27        require
28           program /= Void
29        do
30           Result := create_process
31           Result.execute(program, arguments, keep_environment)
32        end
33
34
35     execute_command_line (command_line: STRING): PROCESS
36           -- Execute the given `program' (how the program is
             discovered is OS-dependent), passing to it the
37           -- `arguments'. The environment is cleared if
             `keep_environment' is False.
38        require
39           command_line /= Void
40           not command_line.is_empty
41        do
42           Result := create_process
43           Result.execute_command_line(command_line, keep_environment)
44        end
45
46     create_process: PROCESS
47        do
48           if basic_exec_system = basic_exec_system_posix then
49              create {PROCESS_POSIX} Result.make
50           elseif basic_exec_system = basic_exec_system_win32 then
51              create {PROCESS_WIN32} Result.make
52           elseif basic_exec_system = basic_exec_system_none then
```

```
53              create {PROCESS_NONE} Result.make
54           else
55              not_yet_implemented
56           end
57           Result.set_direct_input(direct_input)
58           Result.set_direct_output(direct_output)
59           Result.set_direct_error(direct_error)
60           Result.set_group(group)
61        end
62
63  feature {ANY}
64     keep_environment: BOOLEAN
65
66     set_keep_environment (keep_environment_: like keep_environment)
67        do
68           keep_environment := keep_environment_
69        ensure
70           keep_environment = keep_environment_
71        end
72
73  feature {ANY}
74     group: PROCESS_GROUP
75
76     create_group: PROCESS_GROUP
77        do
78           if basic_exec_system = basic_exec_system_posix then
79              create {PROCESS_GROUP_POSIX} Result.make
80           elseif basic_exec_system = basic_exec_system_win32 then
81              create {PROCESS_GROUP_WIN32} Result.make
82           elseif basic_exec_system = basic_exec_system_none then
83              create {PROCESS_GROUP_NONE} Result.make
84           else
85              not_yet_implemented
86           end
87        end
88
89     default_group: PROCESS_GROUP
90        once
91           Result := create_group
92        end
93
94     set_group (a_group: like group)
95        require
96           a_group /= Void
97        do
98           group := a_group
99        ensure
100          group = a_group
101       end
102
103 feature {ANY}
104    direct_input: BOOLEAN
105          -- Is the program's input stream read directly from the
106          -- standard input stream rather than from `input'?
107
108    set_direct_input (direct_input_: like direct_input)
109       do
110          direct_input := direct_input_
111       ensure
```

```
112             direct_input = direct_input_
113          end
114
115       direct_output: BOOLEAN
116             -- Is the program's output stream sent directly to the
117             -- standard output stream rather than to `output'?
118
119       set_direct_output (direct_output_: like direct_output)
120          do
121             direct_output := direct_output_
122          ensure
123             direct_output = direct_output_
124          end
125
126       direct_error: BOOLEAN
127             -- Is the program's error stream sent directly to the
128             -- standard error stream rather than to `error'?
129
130       set_direct_error (direct_error_: like direct_error)
131          do
132             direct_error := direct_error_
133          ensure
134             direct_error = direct_error_
135          end
136
137    feature {}
138       default_create
139          do
140             group := default_group
141             keep_environment := True
142          end
143
144    feature {}
145       -- plugin features
146
147       basic_exec_system: INTEGER
148          external "plug_in"
149          alias "{
150             location: "${sys}/plugins"
151             module_name: "exec"
152             feature_name: "basic_exec_system"
153             }"
154          end
155
156       basic_exec_system_posix: INTEGER
157          external "plug_in"
158          alias "{
159             location: "${sys}/plugins"
160             module_name: "exec"
161             feature_name: "basic_exec_system_posix"
162             }"
163          end
164
165       basic_exec_system_win32: INTEGER
166          external "plug_in"
167          alias "{
168             location: "${sys}/plugins"
169             module_name: "exec"
170             feature_name: "basic_exec_system_win32"
```

```
171             }"
172          end
173
174       basic_exec_system_none: INTEGER
175          external "plug_in"
176          alias "{
177             location: "${sys}/plugins"
178             module_name: "exec"
179             feature_name: "basic_exec_system_none"
180             }"
181          end
182
183    end -- class PROCESS_FACTORY
184    --
185    -- Copyright (C) 2009-2018: by all the people cited in the AUTHORS
       file.
186    --
187    -- Permission is hereby granted, free of charge, to any person
       obtaining a copy
188    -- of this software and associated documentation files (the
       "Software"), to deal
189    -- in the Software without restriction, including without limitation
       the rights
190    -- to use, copy, modify, merge, publish, distribute, sublicense,
       and/or sell
191    -- copies of the Software, and to permit persons to whom the
       Software is
192    -- furnished to do so, subject to the following conditions:
193    --
194    -- The above copyright notice and this permission notice shall be
       included in
195    -- all copies or substantial portions of the Software.
196    --
197    -- THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
       EXPRESS OR
198    -- IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
       MERCHANTABILITY,
199    -- FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT
       SHALL THE
200    -- AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR
       OTHER
201    -- LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE,
       ARISING FROM,
202    -- OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
       DEALINGS IN
203    -- THE SOFTWARE.
204
```