Programming Paradigms

Control Abstraction (Part 5)

Prof. Dr. Michael Pradel
Software Lab, University of Stuttgart
Summer 2020
Overview

- Calling Sequences
- Parameter Passing
- Exception Handling
- Coroutines
- Events
Events

- **Event**: Something a program needs to react to at an unpredictable time
  - GUI events, e.g., mouse clicks
  - Asynchronous I/O

- **Event handler**: Routine called when a specific kind of event happens
  - Sequential handlers
  - Thread-based handlers
Sequential Handlers

- **Handle event in main thread of execution**
- **E.g., OS-level interrupt handlers**
  - Register handler for specific interrupt condition
  - Triggered at hardware level
  - OS transfers control to handler and restores state afterwards
Example: UNIX Signaling

- List of **signals** defined by the OS
- Use to
  - Abort a process, e.g., SIGKILL
  - Communicate with a process, e.g., SIGUSR1
- Program can **register a handler to overwrite** default behavior
- Signals are **delivered asynchronously**
  - Current state of program is paused immediately, wherever it is
Example: UNIX Signaling

<table>
<thead>
<tr>
<th>Signal</th>
<th>Value</th>
<th>Action</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIGHUP</td>
<td>1</td>
<td>Term</td>
<td>Hangup detected on controlling terminal or death of controlling process</td>
</tr>
<tr>
<td>SIGINT</td>
<td>2</td>
<td>Term</td>
<td>Interrupt from keyboard</td>
</tr>
<tr>
<td>SIGQUIT</td>
<td>3</td>
<td>Core</td>
<td>Quit from keyboard</td>
</tr>
<tr>
<td>SIGILL</td>
<td>4</td>
<td>Core</td>
<td>Illegal Instruction</td>
</tr>
<tr>
<td>SIGABRT</td>
<td>6</td>
<td>Core</td>
<td>Abort signal from abort(3)</td>
</tr>
<tr>
<td>SIGFPE</td>
<td>8</td>
<td>Core</td>
<td>Floating point exception</td>
</tr>
<tr>
<td>SIGKILL</td>
<td>9</td>
<td>Term</td>
<td>Kill signal</td>
</tr>
<tr>
<td>SIGSEGV</td>
<td>11</td>
<td>Core</td>
<td>Invalid memory reference</td>
</tr>
<tr>
<td>SIGPIPE</td>
<td>13</td>
<td>Term</td>
<td>Broken pipe: write to pipe with no readers</td>
</tr>
<tr>
<td>SIGALRM</td>
<td>14</td>
<td>Term</td>
<td>Timer signal from alarm(2)</td>
</tr>
<tr>
<td>SIGTERM</td>
<td>15</td>
<td>Term</td>
<td>Termination signal</td>
</tr>
<tr>
<td>SIGUSR1</td>
<td>30,10,16</td>
<td>Term</td>
<td>User-defined signal 1</td>
</tr>
<tr>
<td>SIGUSR2</td>
<td>31,12,17</td>
<td>Term</td>
<td>User-defined signal 2</td>
</tr>
<tr>
<td>SIGCHLD</td>
<td>20,17,18</td>
<td>Ign</td>
<td>Child stopped or terminated</td>
</tr>
<tr>
<td>SIGCONT</td>
<td>19,18,25</td>
<td>Cont</td>
<td>Continue if stopped</td>
</tr>
<tr>
<td>SIGSTOP</td>
<td>17,19,23</td>
<td>Stop</td>
<td>Stop process</td>
</tr>
<tr>
<td>SIGTSTP</td>
<td>18,20,24</td>
<td>Stop</td>
<td>Stop typed at tty</td>
</tr>
<tr>
<td>SIGTTIN</td>
<td>21,21,26</td>
<td>Stop</td>
<td>tty input for background process</td>
</tr>
<tr>
<td>SIGTTOU</td>
<td>22,22,27</td>
<td>Stop</td>
<td>tty output for background process</td>
</tr>
</tbody>
</table>
Signal Delivery

User application

main program execution

hardware interrupt

event handler

call

signal trampoline

[restore state]

return

[restore state]

OS kernel

interrupt handler

[save state]

return from interrupt
Thread-Based Handlers

- Specific (background) thread handles events
- Often, exactly one thread, to avoid need to synchronize
- E.g., GUI thread that reacts to user input and updates UI
  - Android: UI thread is the “main thread”
  - Only use for short-running operations
    (otherwise, app becomes unresponsive)
Quiz: Control Abstractions

Which of the following statements is true?

- Coroutines allow for preemptive multi-tasking.
- A calling sequence is the list of subroutines called during an execution.
- Finally-clauses are executed independently of whether an exception is thrown.
- Signals may interrupt the normal execution.

Please vote in Ilias.
Quiz: Control Abstractions

Which of the following statements is true?

- Coroutines allow for preemptive multi-tasking.
- A calling sequence is the list of subroutines called during an execution.
- Finally-clauses are executed independently of whether an exception is thrown.
- Signals may interrupt the normal execution.

Please vote in Ilias.