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

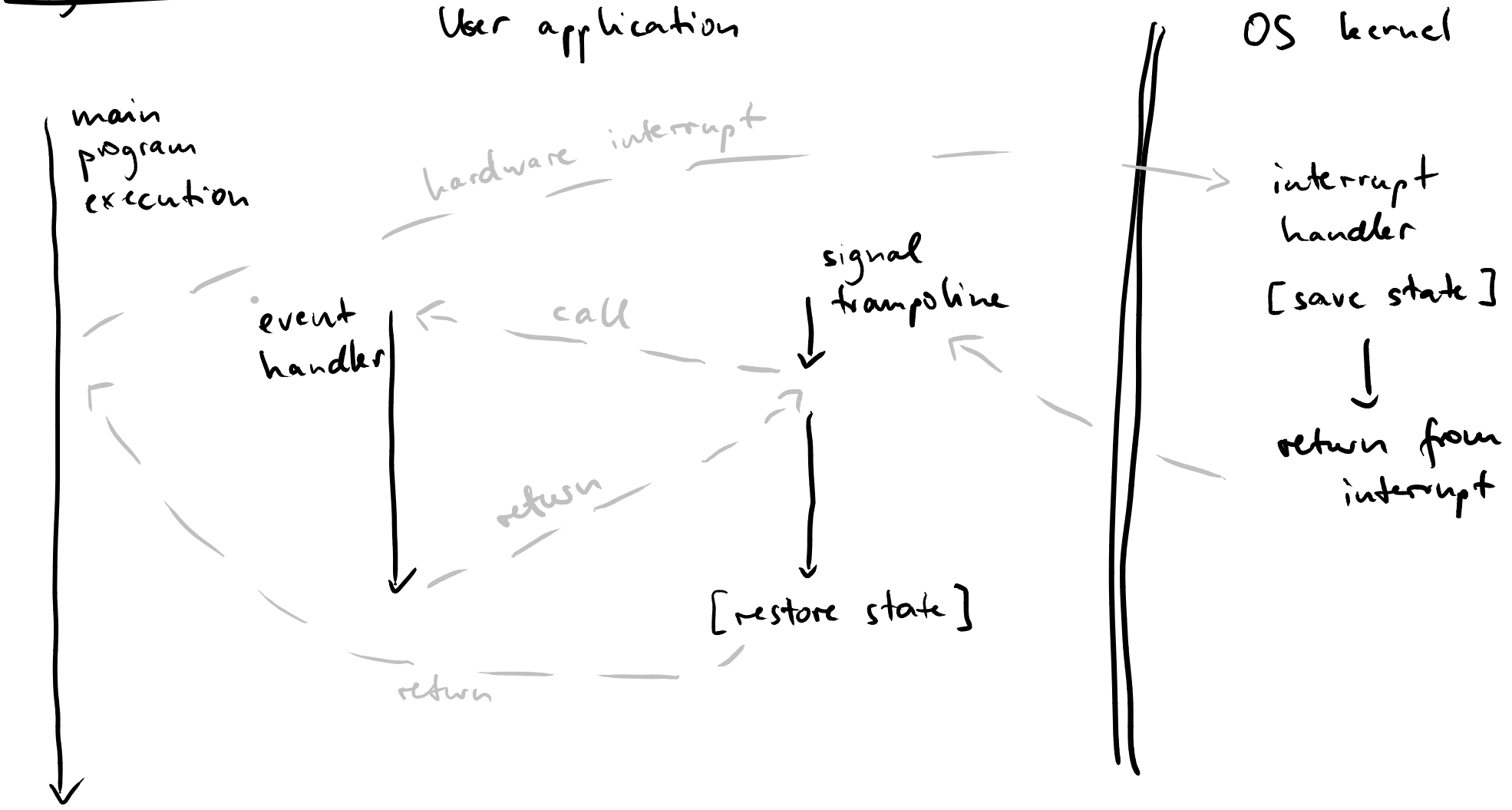
- **List of signals defined by the OS**
- **Use to**
  - Abort a process, e.g., SIGKILL
  - Communicate with a process, e.g., SIGUSR1
- **Program can register a handler to overwrite default behavior**
- **Signals are delivered asynchronously**
  - Current state of program is paused immediately, whereever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

Example: UNIX Signaling

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

immediately, whereever it is

Signal Delivery



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.