Programmier- und Unix-Propädeutika

Eine Einführung für Erstsemester

Carl von Ossietzky Universität Oldenburg
01. Oktober 2026

Leonard Langer

Meta

Beteiligung

1. Stellt gerne Fragen, auch während der Präsentation
2. Stellt gerne Fragen, auch während der Präsentation
3. Macht euch bemerkbar, wenn euch etwas zu schnell geht

    - Die Präsentation enthält viele potenziell neue Informationen.
    - Ihr sollt euch nicht überrannt fühlen!
    - Wenn euch etwas unklar ist, geht es anderen wahrscheinlich auch so

4. Reinrufen erlaubt, solange alle zu Wort kommen

Wiederholung

Wiederholung

UNIX

- Dateien & Verzeichnisse
- Kommandozeile
- Programme ausführen
- Hilfe finden

Python

- Variablen & Datentypen
- Bedingungen & Schleifen
- Funktionen
- Imports

Motivation

(Computer-) Spiele

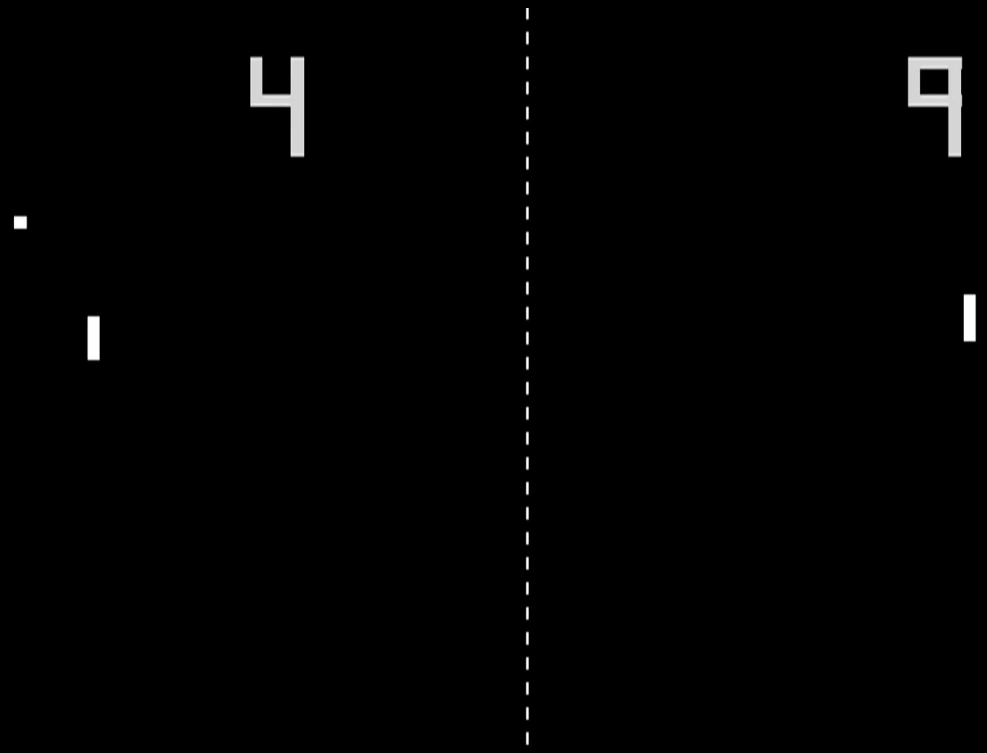
(Computer-) Spiele: Komponenten

- Grafiken
  - 2D/3D, gezeichnet/gerendert, …
- Musik
- Soziales
  - online/multiplayer, Franchise, …
- Geschichte
- Regeln
  - Puzzle, Herausforderungen
- Belohnung
  - gewinnen, sammeln, entdecken, …
- …

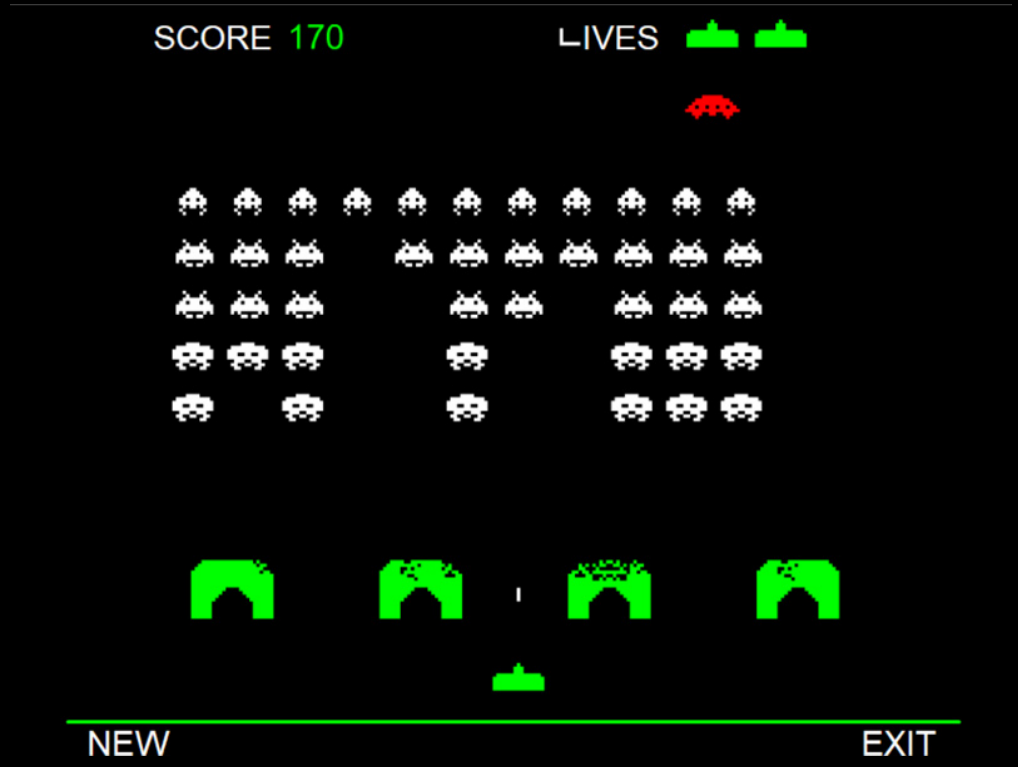
(Computer-) Spiele: Internes

- Zustand (einer, mehrere oder gemischt)
- Funktionen / Zustandsupdates
  - Interaktion
- Darstellung

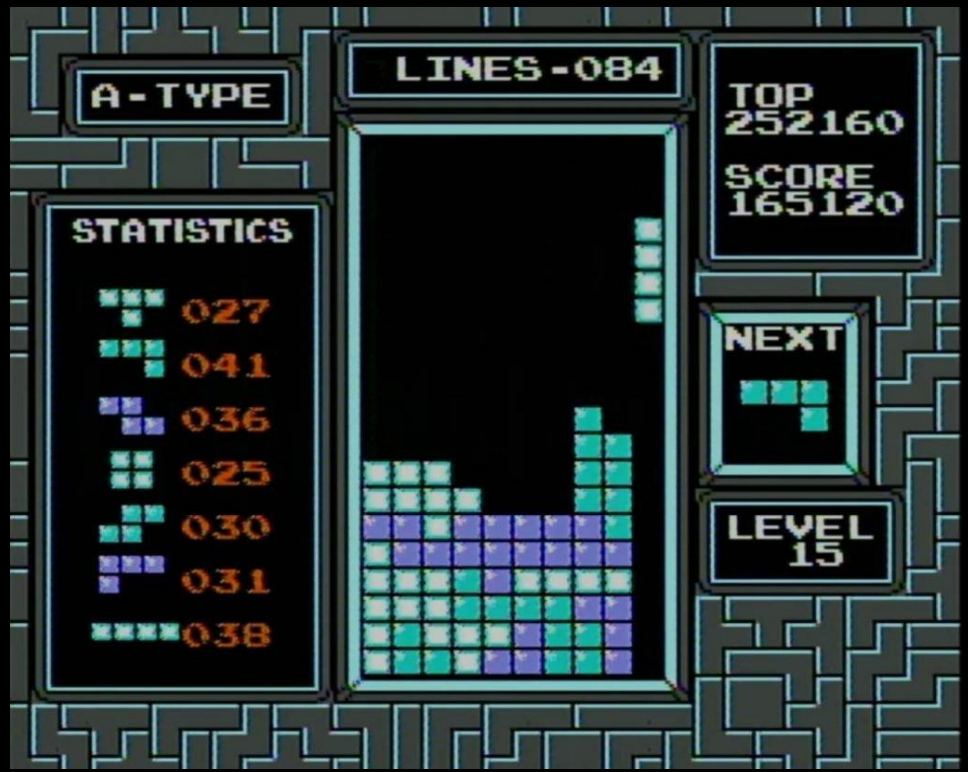
(Computer-) Spiele: Pong



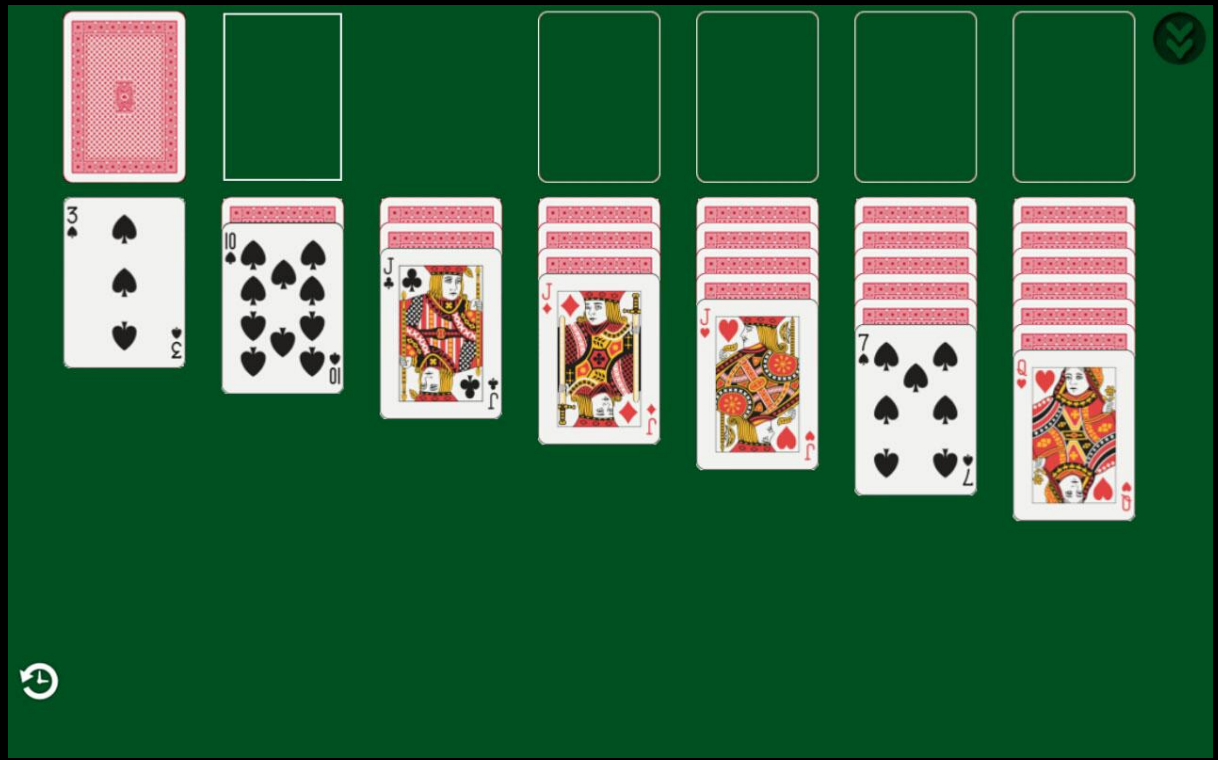
(Computer-) Spiele: Space Invaders



(Computer-) Spiele: Tetris



(Computer-) Spiele: Solitaire



(Computer-) Spiele: Space Cadet Pinball



(Computer-) Spiele: Fruit Ninja



(Computer-) Spiele: Begrenzung

- Eine zentrale Mechanik
- Wenige bewegliche Elemente
- Sofort verständliche Eingaben
- Sichtbares Ziel
- Kurze Feedbackschleife

PyGame Zero

PyGame Zero: Starten

```
touch game.py
pgzrun game.py
```

PyGame Zero: Hintergrund zeichnen

```
WIDTH = 300
HEIGHT = 300

def draw():
  screen.fill((128, 0, 0))
```

PyGame Zero: Ein Rechteck zeichnen

```
rechteck = Rect(50, 80, 100, 40)

def draw():
  screen.clear()
  screen.draw.filled_rect(rechteck, 'lightblue')
```

- `Rect(x, y, breite, höhe)` beschreibt das Rechteck
- `filled_rect()` zeichnet ein gefülltes Rechteck
- `rect()` zeichnet nur den Rand

PyGame Zero: Eine Grafik zeichnen

```
note = Actor('note')
note.pos = 100, 56

def draw():
  screen.clear()
  note.draw()
```

- Grafik in einen `images/` Verzeichnis legen
  - Das Verzeichnis muss auch dort vorhanden sein, wo unsere `game.py` liegt
- Hier: `images/note.png`

PyGame Zero: Zustand aktualisieren

```
def update():
  note.x += 2
```

- PyGame Zero ruft `update()` regelmäßig auf
- Jeder Aufruf verschiebt die Note um 2 Pixel nach rechts
- `draw()` zeichnet die Note an ihrer aktuellen Position

PyGame Zero: Bewegung und Zeit

```
def update(dt):
  note.x += 120 * dt
```

- `dt`: Zeit seit dem letzten Update in Sekunden
- `120`: Geschwindigkeit in Pixeln pro Sekunde
- Geschwindigkeit × Zeit = zurückgelegter Weg

PyGame Zero: Am Fensterrand

```
def update(dt):
  note.x += 120 * dt
  if note.left > WIDTH:
    note.right = 0
```

- Linker Rand hinter dem Fenster: Note vollständig verschwunden
- Rechten Rand auf `0` setzen: Note startet wieder links

PyGame Zero: Interaktion

```
def on_mouse_down(pos):
  if note.collidepoint(pos):
    print("Getroffen!")
  else:
    print("Daneben!")
```

PyGame Zero: Tastatur

```
def on_key_down(key):
  if key == keys.SPACE:
    print("Leertaste!")
```

- `on_key_down()` reagiert auf einen Tastendruck
- `key` enthält die gedrückte Taste
- `keys.SPACE` steht für die Leertaste

PyGame Zero: Tasten halten

```
def update(dt):
  if keyboard.up:
    note.y -= 120 * dt
  if keyboard.down:
    note.y += 120 * dt
```

- `keyboard` gibt uns den aktuellen Zustand der Tasten
- Die Note bewegt sich, solange die Taste gehalten wird

PyGame Zero: Sounds und Bilder

```
def on_mouse_down(pos):
  if note.collidepoint(pos):
    sounds.hit.play()
    note.image = 'note_hit'
```

- Weitere Grafik: `images/note_hit.png`
- Sound: `sounds/hit.wav`

PyGame Zero: Clock

```
def on_mouse_down(pos):
  if note.collidepoint(pos):
    hit_note()

def hit_note():
  sounds.hit.play()
  note.image = 'note_hit'
  clock.schedule_unique(reset_note, 1.0)

def reset_note():
  note.image = 'note'
```

PyGame Zero: Zusammenfassung

- `WIDTH` und `HEIGHT` bestimmen die Fenstergröße
- `draw()` stellt den aktuellen Zustand dar
- `update()` verändert den Zustand
- `on_mouse_down()` und `on_key_down()` reagieren auf Eingaben
- `keyboard` gibt den aktuellen Zustand der Tasten an
- `images/` und `sounds/` enthalten unsere Ressourcen
- `clock` ruft Funktionen zu einem späteren Zeitpunkt auf

Danke

Diese Präsentation wurde …

- präsentiert mit
  - **presenterm** (https://mfontanini.github.io/presenterm/)
- erstellt mit
  - **helix** (https://helix-editor.com/)
- versioniert mit
  - **git** (https://git-scm.com/)
- dargestellt in
  - **kitty** (https://sw.kovidgoyal.net/kitty/)
- zur Verfügung gestellt von
  - **nixos** (https://nixos.org/)

… und mit viel Liebe gemacht <3