2D Spiele in Java

Inhalt

- Full Screen Mode vs. Window Mode
- Actives vs. Passives Rendering
- 2D- Hardwarebeschleunigung
- Steuerung
- Game-Loop

Fullscreen vs Windowmode

- Spiele laufen üblicherweise Fullscreen
- Hardwarebeschleunigung teilw. nur im Fullscreenmode
- „Zeichnen“ wird vom Programmierer übernommen

Active vs. Passive Rendering

- in AWT wird das Zeichnen des Fensters vom OS ausgelöst
  - Programm weiß nicht, wann der Benutzer das Fenster vergrößert, verschiebt, ...

```
public void paint(Graphics g) {
    // Use g to draw my Component
}
```

- sogenanntes „Passives Rendering“

Actives vs. Passives Rendering

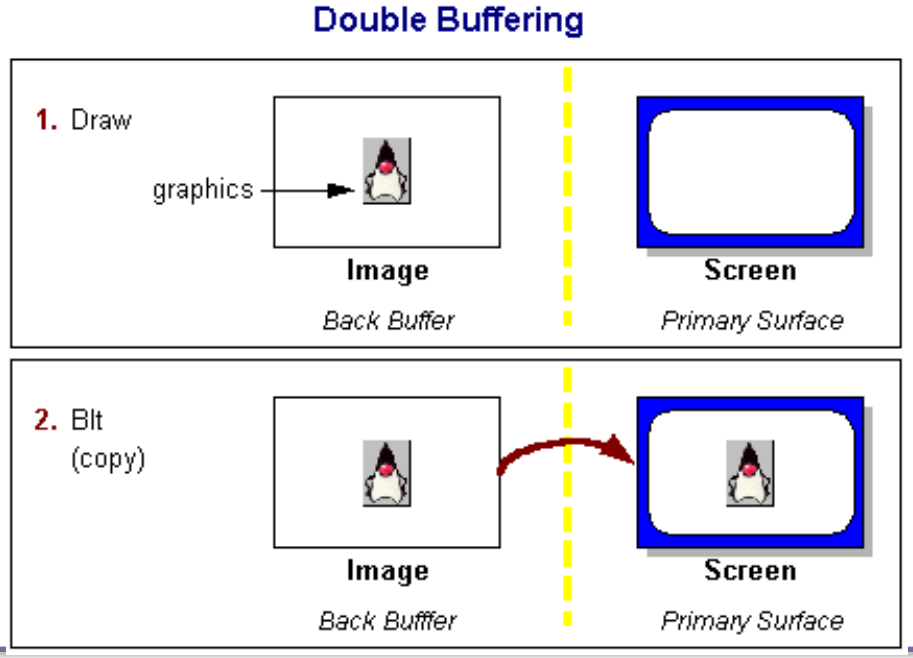
- bei der Spieleentwicklung wird das Rendern aktiv betrieben
- d.h. wir bestimmen selbst
  - wann etwas gezeichnet wird
  - was gezeichnet wird
- Automatismus wird „deaktiviert“

Hardwarebeschleunigung

- Bewegung am Bildschirm
  - neu Zeichnen des gesamten Bildes mit geänderter Szene
  - Vorher: altes Bild löschen
  - Problem:
    - im schlimmsten Fall: Zeichnen kann beobachtet werden
    - Löschen führt zu Flackern!

Hardwarebeschleunigung

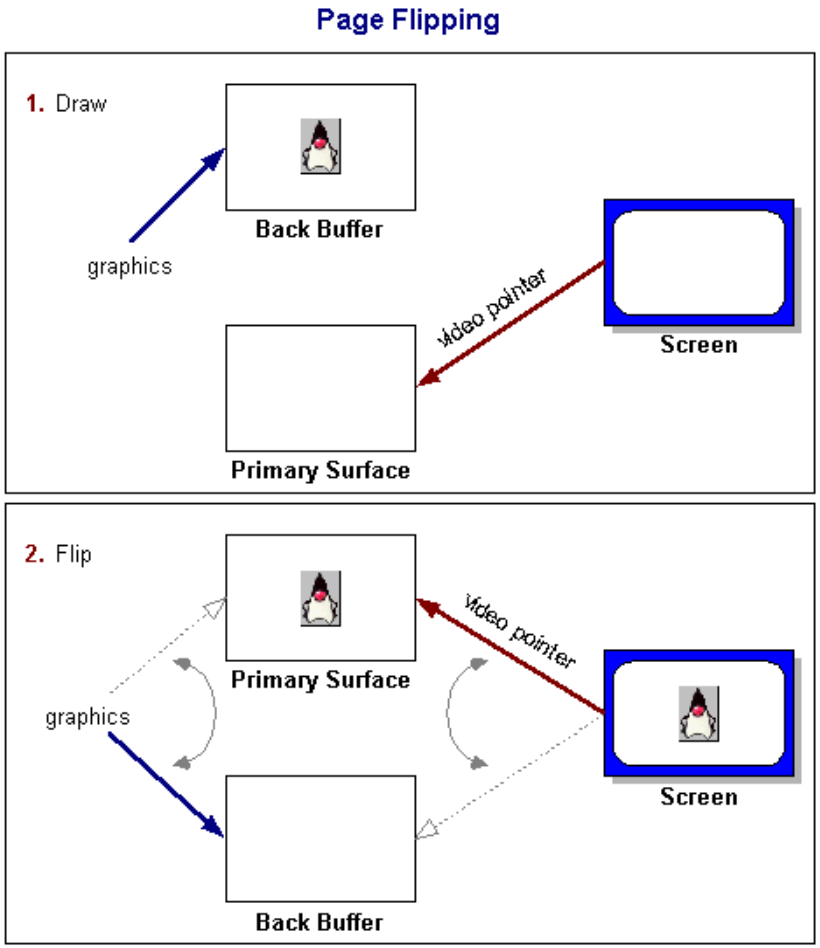
- Double Buffering
  - das Zeichnen des Bildes erfolgt vollständig
  - erst wenn vollständig, wird das Bild in den GrafikRAM kopiert
- Blit = kopieren



Hardwarebeschleunigung

- Page Flipping
  - manche GraKa haben einen sog. „Video Pointer“
  - dieser zeigt auf eine Adresse im VideoRAM der angezeigt wird
  - Video Pointer kann einfach auf eine neue Adresse gesetzt werden
  - es können mehrere Bilder im Vorhinein berechnet werden

Hardwarebeschleunigung



Hardwarebeschleunigung

- Rechenbeispiel:
  - 60 fps
  - Auflösung 1280x1204, 32Bit Fabtiefe
  - ca. 5MB pro Frame
  - => 60 x 5MB = 300MB Datentransfer am Bus (RAM -> VRAM)

Hardwarebeschleunigung

- ab Java 1.5: VolatileImage
- Bilder werden direkt im VideoRAM geladen
  - Vorteil: Bus wird entlastet!
  - Nachteil: sie können „verloren“ gehen
    - andere App geht Fullscreen
    - Taskmanager wird geöffnet
    - ...

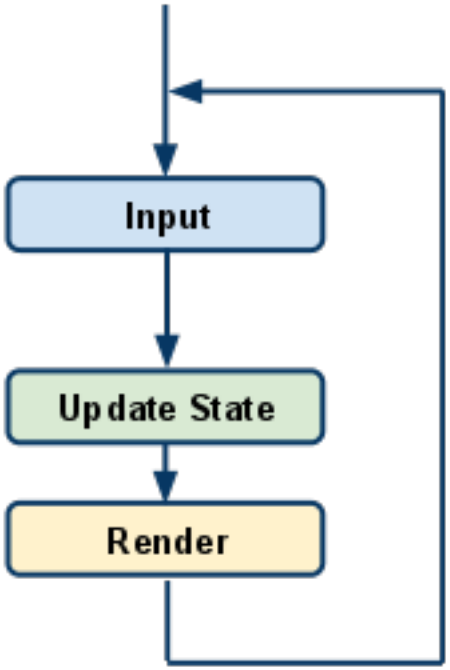
Tastenabfrage

- für die Steuerung müssen die Keyevents abgefragt werden: Event bei
  - Taste wird gedrückt
  - Taste wird losgelassen
  - Taste wird „eingegeben“

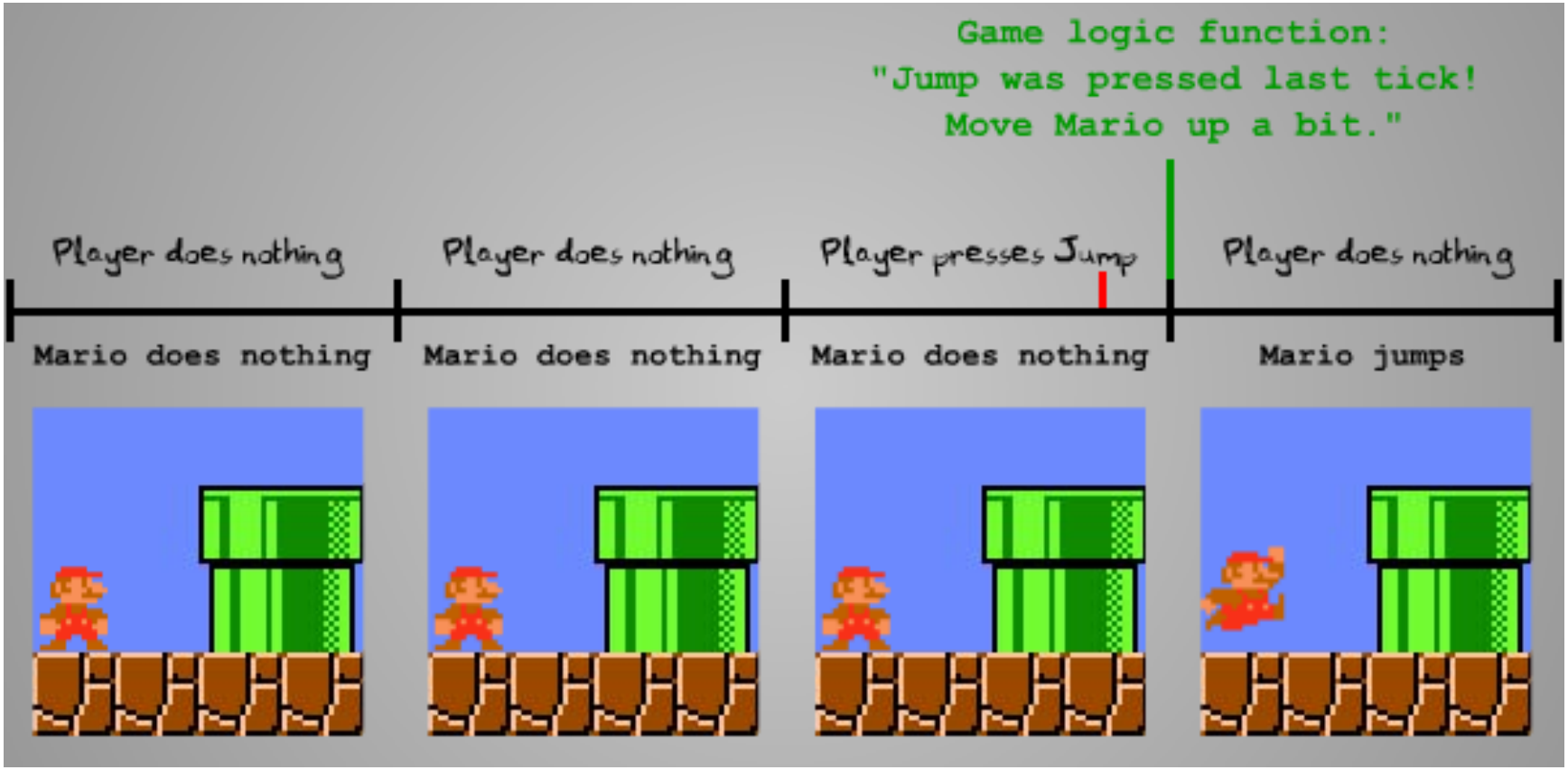
```
public class GameKeyListener extends KeyAdapter
{
    public void keyPressed(KeyEvent e)
    {
        if (e.getKeyCode() == KeyEvent.VK_LEFT)  left = true;
        if (e.getKeyCode() == KeyEvent.VK_RIGHT) right = true;
        if (e.getKeyCode() == KeyEvent.VK_SPACE) fire = true;
        if (e.getKeyChar() == 27) esc = true;
    }
}
```

Game Loop

- die Game Loop ist zuständig für
  - Zeichnen der Szene
  - Abfragen der Tasten
  - Spiellogik
  - Kontrolle der Spielgeschwindigkeit



Game Loop


Game logic function:
"Jump was pressed last tick!
Move Mario up a bit."
Player does nothing
Player does nothing
Player presses Jump
Player does nothing
Mario does nothing
Mario does nothing
Mario does nothing
Mario jumps

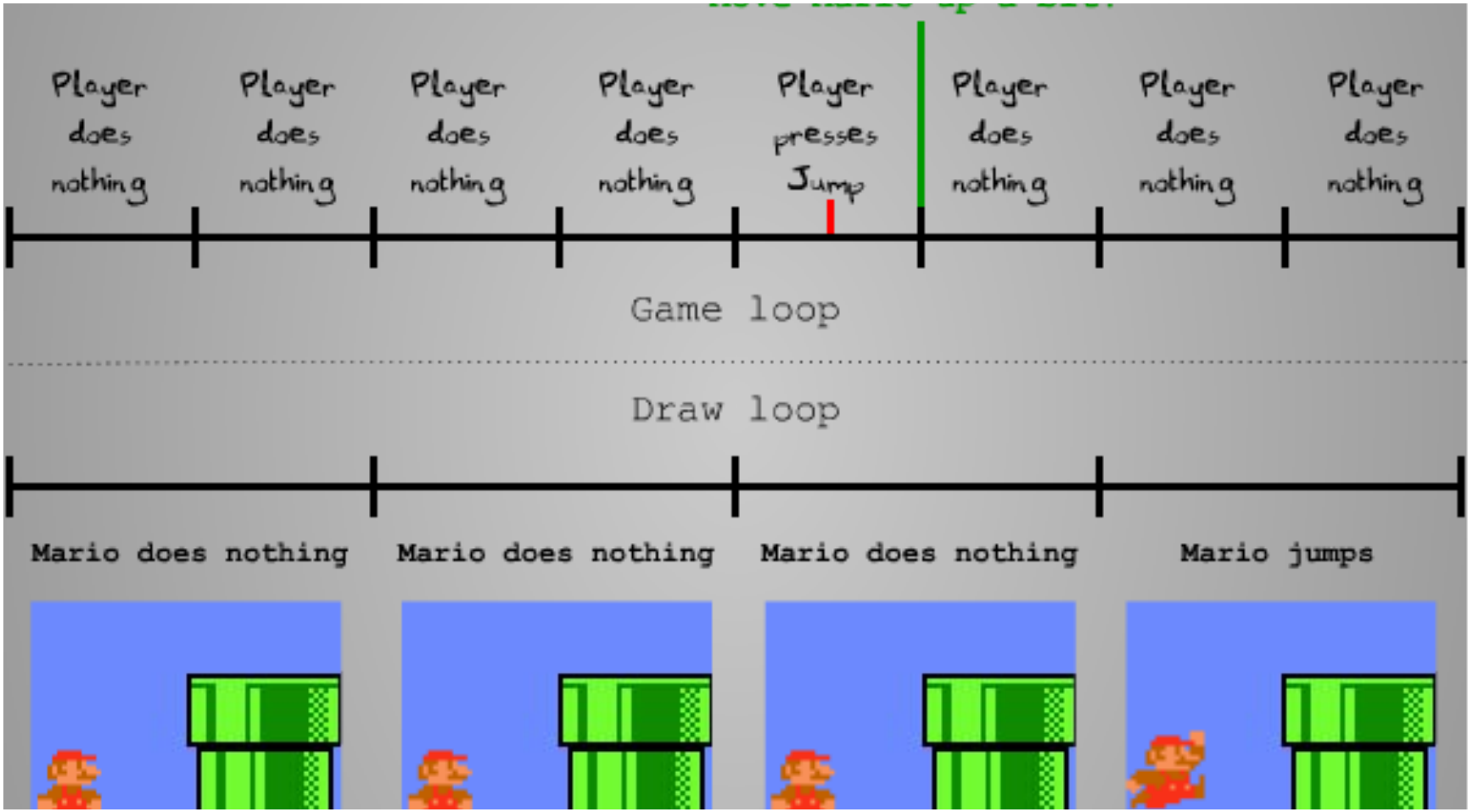
Game Loop

- Annahme: Action Spiel mit 60FPS
  - innerhalb von 60FPS muss
    - schnell auf Benutzeraktionen reagiert werden (Tastenabfragen auswerten)
    - Gravitation berechnen
    - KI berechnen
    - Grafik zeichnen
    - ...
  - Was ist, wenn Rechner zu schwach für diese Zeitvorgabe ist?

Game Loop

- Lösung:
  - FPS müssen weniger werden
  - sprich: es wird nicht so oft gezeichnet
  - ABER:
    - Spiellogik läuft weiterhin schneller
  - -> Game Loop + Draw Loop!

Game Loop


Player does nothing
Player does nothing
Player does nothing
Player does nothing
Player presses Jump
Player does nothing
Player does nothing
Player does nothing
Game loop
Draw loop
Mario does nothing
Mario does nothing
Mario does nothing
Mario jumps

Referenzen

- Java Tutorials:
  - http://docs.oracle.com/javase/tutorial/extra/fullscreen/index.html
  - http://www.petrastumpf.de/michael/Wissen/SpieleprogrammierungInJava.pdf
  - http://www.javacodegeeks.com/2011/06/android-game-development-tutorials.html
- Volatile Image:
  - http://content.gpwiki.org/index.php/Java:Tutorials:VolatileImage
- Game Loop:
  - http://active.tutsplus.com/tutorials/games/understanding-the-game-loop-basix/