Interaktive Computergrafik

Prof. Dr. Frank Steinicke
Human-Computer Interaction
Department of Computer Science
University of Hamburg

Interaktive Computergrafik

Übung - Woche 6

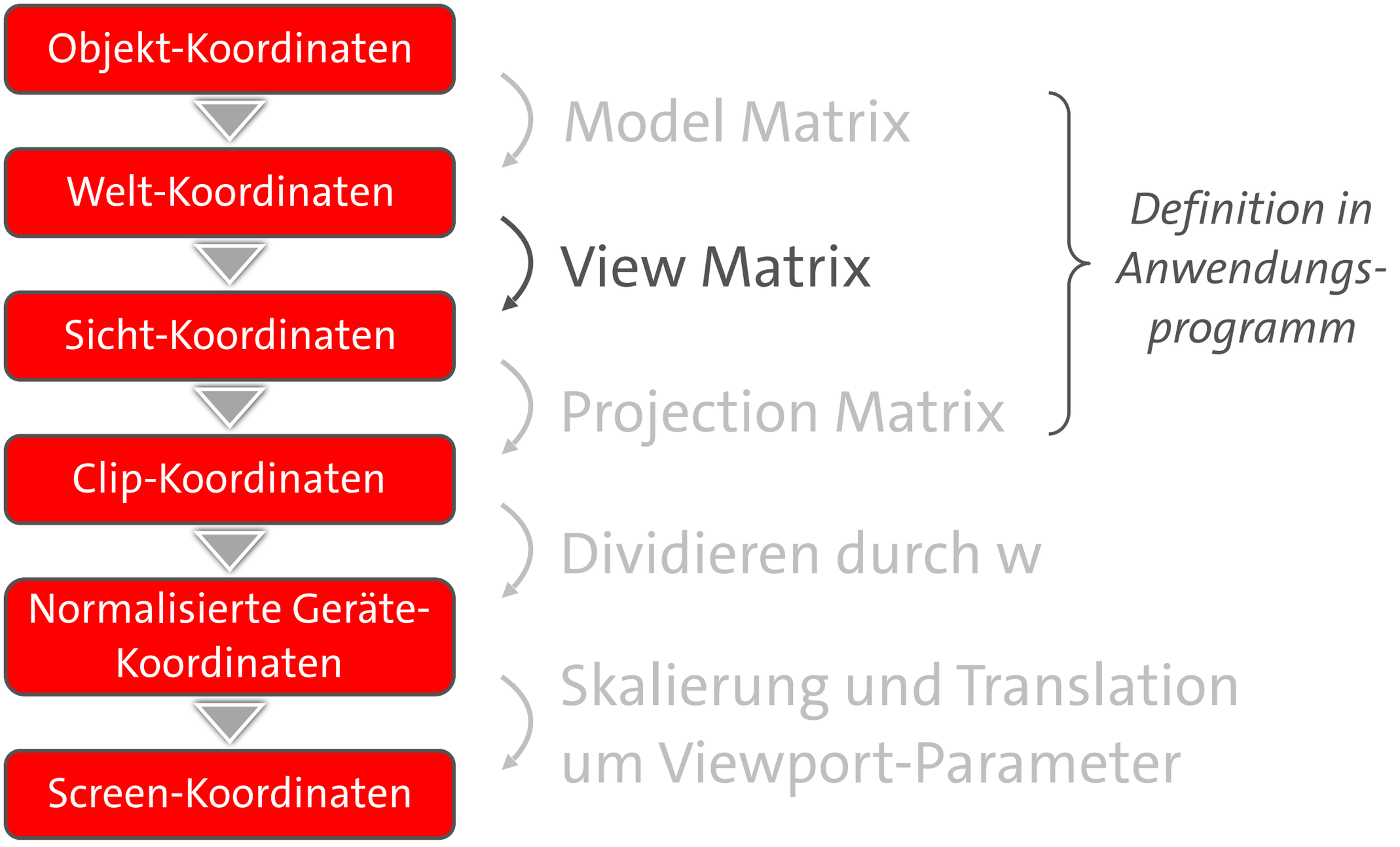
Human-Computer Interaction, University of Hamburg

Interaktive Computergrafik

Übung - Woche 6

View Matrix

Rückblick: Pipeline



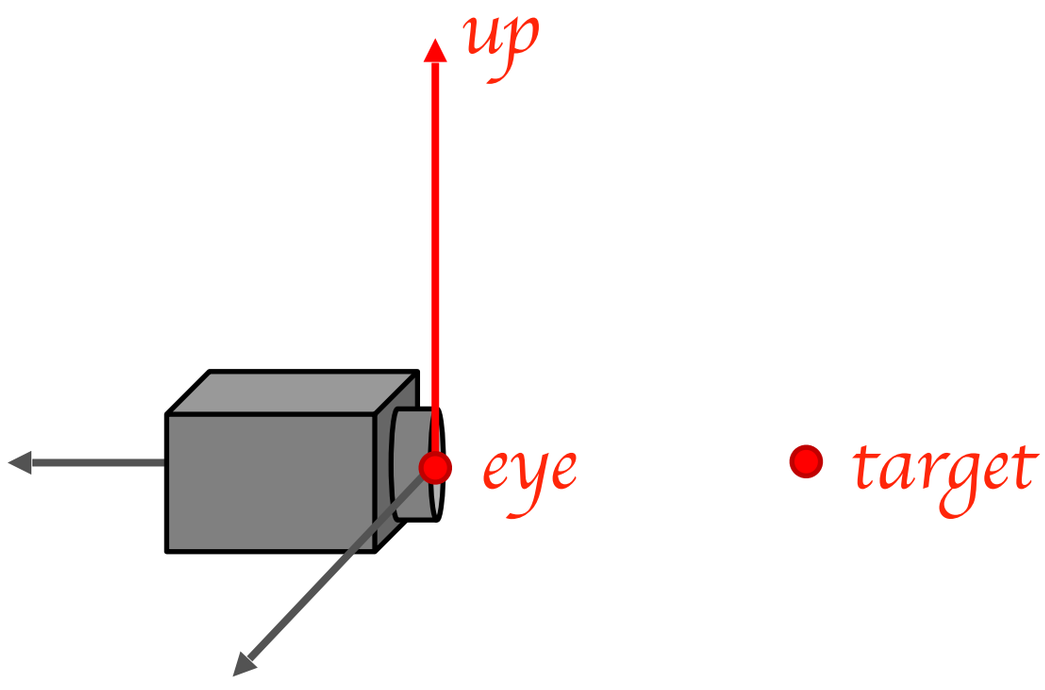
Library *glMatrix*

Rückblick

- Sammlung von JavaScript-Funktionen für Matrix- und Vektoroperationen
- Darf in Übungsaufgaben benutzt werden!
- Source + Dokumentation: *glmatrix.net*

View Matrix

```
mat4.lookAt(
    outMatrix,
    eye,
    target,
    up);
```



Interaktive Computergrafik

Übung - Woche 6

Interaktives WebGL

Interaktivität

- Nutzer kann über festgelegte Eingabe-kanäle (z.B. Maus, Tastatur, Sprache) Einfluss auf den Zustand des Systems nehmen
- System reagiert mit zeitnahem Feedback auf die Aktionen des Nutzers

Events

Arten

- **Window**-Events
- **Maus**-Events
- **Tastatur**-Events
- Touch-Events
- Medien-Events (für Video und Audio)
- Formular-Events
- …

Events

Registrierung

- Benötigt immer 3 Angaben:
  - Elementobjekt, auf dem Event registriert wird
  - Ereignistyp
  - Handler-Funktion
- Umsetzung möglich über 3 verschiedene Methoden

Events
Definition

- **Events** sind Ereignisse, die vom Nutzer oder vom Browser selbst ausgelöst werden können
- **Event-Handler** sind programmierbare Anweisungen, die beim Eintreten eines Events ausgeführt werden

Events

Registrierung

1) Direkt als Attribut eines HTML-Elements:

```
<button id="myButton"
onclick="doSomething()">OK</button>
```

2) Als Eigenschaft des Elementobjekts:

```
document.getElementById("myButton").
onclick = doSomething;
```

3) Mit Event-Listener:

```
document.getElementById("myButton").
addEventListener("click", doSomething);
```

Events

Registrierung

3) Mit **Event-Listener**:

```
document.getElementById("myButton").
addEventListener("click", doSomething);
```

- Unterstützt Prinzip des *Unobtrusive JavaScript*
- Deaktivierung des Event-Handlers möglich:
  `…("myButton").removeEventListener("click", doSomething);`
- Registrierung mehrerer Handler-Funktionen für einen Ereignistyp möglich

Events
Behandlung

```
function doSomething(e) {
// Ereignisbehandlung
}
```

- Abruf von Zusatzinformationen in Handler-Funktion über spezielle Objekte:
  - `e` - verweist auf das abgefangene Event
  - `this` - verweist auf das HTML-Element, auf dem das Event registriert wurde

Events
Behandlung

- Allgemeine Event-Informationen, z.B.:
  - `e.type` - Eventname
- Typspezifische Informationen, z.B.:
  - `e.clientX` - x-Koordinate des Mauszeigers
  - `e.button` - gedrückte Maustaste
  - `e.key` - gedrückte Taste als Unicode-Wert
  - `e.ctrlKey` - true, wenn gleichzeitig Ctrl-Taste gedrückt wurde

Events

Beispiele - Maus-Events

- onclick/ondblclick
- onmousedown/onmouseup
- onmouseenter/onmouseleave
- onmouseover
- onmousemove

Events

Beispiele - Tastatur-Events

- onkeydown/onkeyup
- onkeypress

key	Zeichen	key	Zeichen	key	Zeichen
32	Space	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	101	d
37	%	69	E	102	e
38	&	70	F	103	f

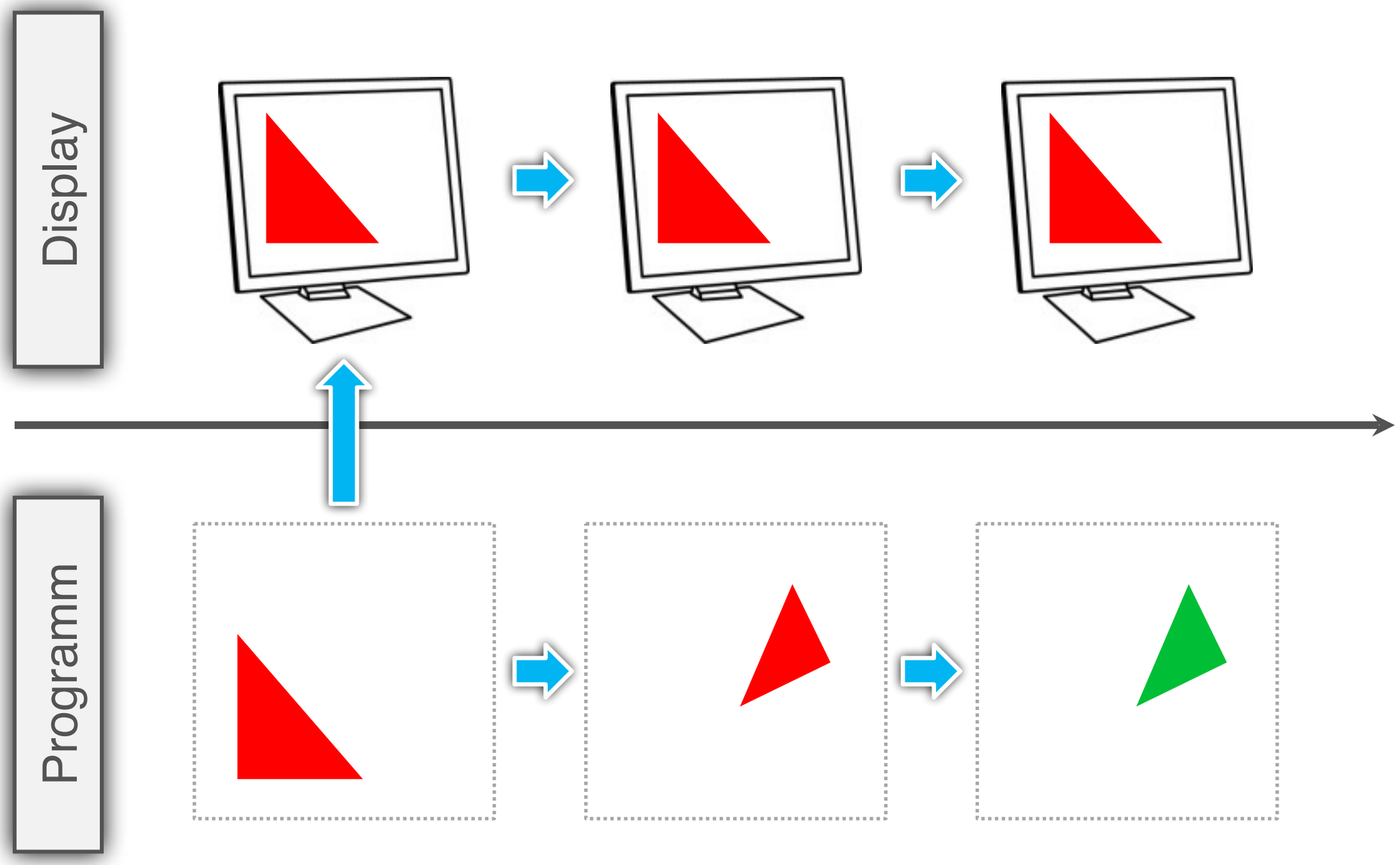
Rückblick: Interaktivität

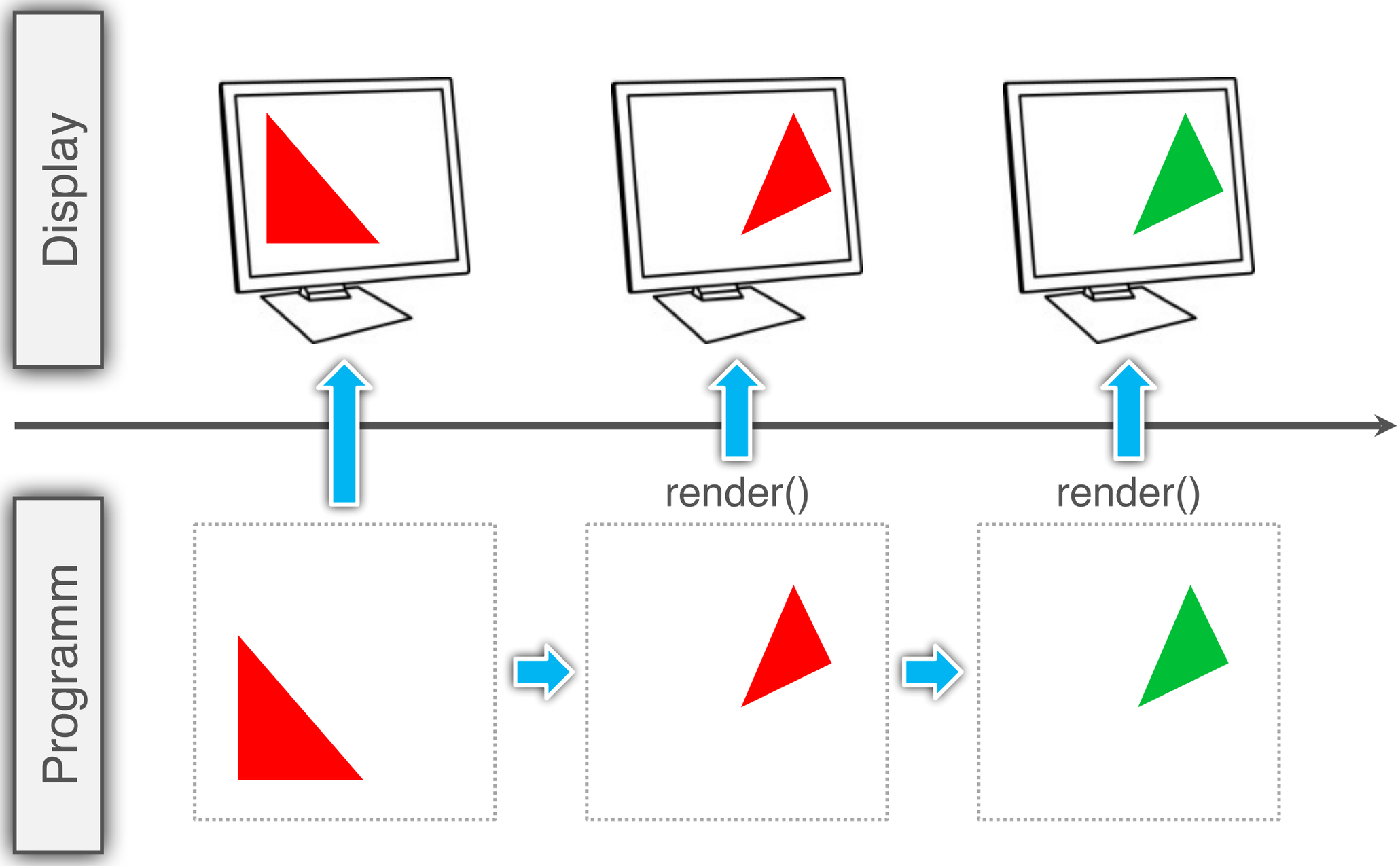
- Nutzer kann über festgelegte Eingabe-kanäle (z.B. Maus, Tastatur, Sprache) Einfluss auf den Zustand des Systems nehmen

  → **User-Events**

- System reagiert mit zeitnahem Feedback auf die Aktionen des Nutzers

  → **?**


Display
Programm


Display
render()
render()
Programm

Render-Loop

```
requestAnimationFrame(renderFunction);
```

- passt sich an Refresh-Rate des Browsers an (meist ebenfalls 60 Hz) → Browser meldet über Event, sobald neues Frame gerendert werden kann
- verringert die Aktualisierungsrate, falls Element nicht sichtbar ist