

Klasse Bildschirm

Ein Bildschirm ist das Modell eines rechteckigen Bereiches auf dem angeschlossenen Computerbildschirm. Auf ihm kann mit Stiften gezeichnet werden. Zu diesem Zweck ist die Zeichenebene auf dem Bildschirm mit einem Koordinatensystem versehen, dessen Ursprung sich in der oberen linken Ecke der Zeichenebene befindet und dessen Achsen horizontal nach rechts und vertikal nach unten gerichtet sind. Die Einheit ist ein Pixel.

__init__(hintergrundbild='')

Konstruktor: Der Bildschirm wird initialisiert. 'hintergrundbild' ist ein optionaler Dateiname einer Grafikdatei als Bildschirmhintergrund.

Breite()

liefert die Breite der Zeichenebene.

Hoehe()

liefert die Hoehe der Zeichenebene.

LoescheAlles()

loescht die Zeichenebene.

setzeHintergrund(dateiname)

Lade einen Hintergrund und zeige diesen im Fenster.

GibFrei()

Das Ausgabefenster wird abgebaut.

Klasse Stift

Der Stift ist ein Zeichenwerkzeug, dass sich auf dem Bildschirm bewegen kann. Er hat eine Position (Ursprung ist links oben) und eine Richtung (0'=rechts, Drehsinn math. positiv.)

Der Stift kennt zwei Zustände: gesenkt (dann wird geschrieben) und gehoben (dann wird der Stift bewegt, ohne zu schreiben).

Zum Zeichnen kennt er zwei Modi: 'normal' zum Schreiben, und 'radieren' zum Löschen bzw. Ausradieren.

__init__(auf_bildschirm)

Konstruktor: Der Stift wird initialisiert. Der Stift befindet sich angehoben auf Position (0,0).

Als Parameter muss der zugehörige Bildschirm angegeben werden.

GibFrei()

Der Stift wird freigegeben und steht nicht mehr zur Verfügung.

setzeStandard()

Der Stift wird zurückgesetzt. Der Stift befindet sich angehoben auf Position (0,0).

bewegeBis(px, py)

Bewege den Stift nach px,py

bewegeUm(pl)

Bewege den Stift pl Einheiten in der aktuellen Richtung

dreheBis(pw)

Setze den Drehwinkel auf pw:int (0..359)

dreheUm(pw)

Erhöhe den Drehwinkel um pw:int (0..359)

schreibe(ps)

Gib einen Text an der aktuellen Position aus. Der Stift steht anschließend am Ende des Textes.

hoch()

Hebe den Stift.

runter()

senke den Stift

normal()

aktiviere den normalen Schreibmodus

radiere()

aktiviere den Radiermodus

hPosition()

liefert die horizontale X-Position

vPosition()

liefert die vertikale Y-Position

winkel()

liefert den aktuellen Winkel

zeichneRechteck(pbreite, phoehe)

Zeichnet an der aktuellen Position ein Rechteck nach rechts unten. pbreite: int, phoehe: int

zeichneKreis(pradius)

Zeichnet an der aktuellen Position einen Kreis mit dem Radius pradius.

Klasse Buntstift

Der Buntstift ist wie der Stift ein Zeichenwerkzeug, dass sich auf dem Bildschirm bewegen kann. Er hat alle Attribute und Methoden des Stiftes, zusätzlich hat er eine Farbe (Konstanten: schwarz, weiss, rot, gruen, blau, gelb); sowie einen Schriftstil.

Neue Methoden:

setzeFarbe(neue_farbe)

Setze Farbe auf den angegebenen Farbwert.

neue_farbe = (Konstanten: schwarz, weiss, rot, gruen, blau, gelb) oder ein Tupel der Form (r,g,b) mit Werten aus (0..255)

setzeFuellMuster(muster)

setze Füllung 0=aus, 1=an

setzeSchriftArt(art)

Lade Schriftart. art:string mit Werten: 'arial' oder 'cour'

setzeSchriftStil(stil)

Setze Schriftstil. stil:int mit 0=normal, 1=fett, 2=kursiv, 4=unterstrichen

setzeSchriftGroesse(groesse)

Setze Schriftgroesse. groesse: int (10 oder jede andere positive Zahl)

Klasse Maus

Die Maus realisiert die Mauseingabe des Computers. Diese ist gekennzeichnet durch Position und Tastenzustand.

__init__(auf_bildschirm)

Die Maus ist zur Nutzung aktiviert. Der zugehörige Bildschirm muss als Parameter übergeben werden.

IstGedrueckt()

Liefert 'wahr', wenn die Maustaste gedrückt ist.

DoppelKlick()

Liefert 'wahr', wenn die rechte Maustaste gedrückt ist.

HPosition()

Liefert die horizontale x-Position der Maus.

VPosition()

Liefert die vertikale y-Position der Maus.

GibFrei()

Gibt die Maus wieder frei.

Klasse Tastatur

Die Tastatur realisiert die Eingabe des verwendeten Computers.

wurdeGedrueckt()

Liefert wahr, wenn eine Taste gedrückt wurde.

zeichen()

Liefert das zuletzt gedrückte Zeichen.

GibFrei()

Die Tastatur steht nicht mehr zur Verfügung.

Klasse Sprite

Diese Klasse stammt nicht aus den ursprünglichen Stifte und Mäuse-Entwurf, wurde aber als sinnvolle Erweiterung implementiert. Sie stellt eine auf der Ausgabefläche frei bewegbare Figur (=eine kleine Grafik) dar. Sie eignet sich beispielsweise zur Darstellung von Spielcharakteren bei der Realisierung einfacher Computerspiele.

__init__(bild,fenster)

Vorbereiten des Sprites: Das Bild (jpg/bmp) wird geladen und das Sprite ist vorbereitet.

bilddatei: Dateiname als string

fenster: zugehöriger Bildschirm

auch

setzePosition(x,y)

Bewege das Sprite an die Position x,y: int

xPosition()

liefert die aktuelle x-Position

yPosition()

liefert die aktuelle y-Position

setwinkel(neuwinkel)

Drehe das Spritebild auf den angegebenen

loeschen()

Entferne das Sprite vom Bildschirm.

kollidiert(mit)

prüfe, ob das Sprite mit dem als Parameter angegebenen Sprite kollidiert. Liefert 1 (wahr) oder 0 (falsch).

abstrakte Klasse Anwendung

Anwendung ist eine abstrakte Klasse als Basis für alle weiteren Anwendungen. Sie besitzt bereits einen Bildschirm, Maus und Tastatur, die alle initialisiert sind. Bei der Realisierung von weiteren Anwendungen als Unterklasse muss insbes. die abstrakte Methode 'FuehreAus' überschrieben werden.

Die Methoden '__init__' und 'GibFrei' müssen aufgerufen werden. '__init__' sollte nicht überschrieben werden. Wird eine eigene init-Methode gebraucht, sollte ein anderer Name wie z.B. 'start' verwendet und diese explizit aufgerufen werden.

__init__()

Konstruktor: die Anwendung wird initialisiert. Sie besitzt einen Bildschirm, eine Tastatur und eine Maus, die alle initialisiert sind.

GibFrei()

Freigabe der verwendeten Objekte

Bildschirm()

liefert den Bildschirm der Anwendung

Maus()

liefert die Maus der Anwendung

Tastatur()

liefert die Tastatur der Anwendung

FuehreAus()

Abstrakte Methode, anfangs ohne Funktionalität. Die Funktion ist per Vererbung zu definieren.

abst. Klasse Ereignisanwendung

Eine Ereignis-Anwendung ist eine abstrakte Klasse als Basis einer Anwendung, die auf Ereignisse der Tastatur und Maus reagiert. Anfallende Ereignisse werden einzeln einer zugehörigen Bearbeitungsmethode übergeben.

Bei der Realisierung von Anwendungen müssen insbes. die Methoden 'BearbeiteTaste', 'BearbeiteMausDruck' überschrieben werden. Mit ihnen werden die konkreten Reaktionen auf die entsprechenden Ereignisse

realisiert. Die Methoden "FuehreAus" und "Beenden" sollte nicht überschrieben werden.

Die Methoden '__init__' und 'GibFrei' müssen aufgerufen werden. '__init__' sollte nicht überschrieben werden. Wird eine eigene init-Methode gebraucht, sollte ein anderer Name wie z.B. 'start' verwendet und diese explizit aufgerufen werden.

FuehreAus()

Es wird auf Ereignisse gewartet und ggf. die Methoden [BearbeiteTaste, BearbeiteMausDruck, BearbeiteSpezialKlick, BearbeiteMausLos, BearbeiteMausBewegt] aufgerufen.

Beenden()

Die Anwendung wird beendet.

BearbeiteTaste(zeichen)

Ein Tastatur-Ereignis ist eingetreten. Diese Methode kann neu definiert werden.

BearbeiteMausDruck(x,y)

Ein Mausdruck-Ereignis ist eingetreten. Diese Methode kann neu definiert werden.

BearbeiteSpezialKlick(x,y)

Ein Mausdruck-Ereignis ist eingetreten. Diese Methode kann neu definiert werden.

BearbeiteMausBewegt(x,y)

Ein Maus-Bewegt-Ereignis ist eingetreten. Diese Methode kann neu definiert werden.

BearbeiteMausLos(x,y)

Ein Mausdruck-Ereignis ist eingetreten. Diese Methode kann neu definiert werden.

Beispiel

""" Malen mit der Maus """

```
bild=Bildschirm()
maus=Maus()
tast=Tastatur()
stift=Stift()

stift.bewegeBis(120,20)
stift.runter()
stift.schreibe("Malen mit der Maus...")

while not maus.DoppelKlick():

    if tast.wurdeGedrueckt():
        eingabe=tast.zeichen()
        stift.schreibe(eingabe)

    if maus.IstGedrueckt()==1:
        stift.hoch()
        stift.bewegeBis(maus.HPosition(),
                        maus.VPosition())

        while maus.IstGedrueckt():
            stift.runter()
            stift.bewegeBis(maus.HPosition(),
                            maus.VPosition())

stift.GibFrei()
maus.GibFrei()
bild.GibFrei()
tast.GibFrei()
```