Übung 2 – Multimedia-Programmierung

Inhalt

Einführung in ActionScript

Aufgaben

Aufgabe 1: Sekundenzähler

a) Es soll ein einfacher Sekundenzähler erstellt werden. Erstellen Sie dazu eine Klasse *ClockCounter*. Diese soll folgende Methoden enthalten:

- *getAngle()* gibt den aktuellen Winkel des Sekundenzeigers zurück (Winkel in Grad)
- *getSeconds()* gibt die aktuelle Sekundenzahl zurück (zwischen 0 und 59)
- *getMinutes()* gibt die aktuelle Minutenzahl zurück.
- *incSeconds()* erhöht die Zahl der Sekunden um 1 (falls > 59, dann auf 0 zurücksetzen und Minuten erhöhen)

Zu Beginn sind Sekunden und Minuten auf 0 gesetzt.

Erstellen Sie eine Uhr mit einem Sekundenzeiger. Der Sekundenzeiger soll ein Movieclip sein und einen Instanznamen erhalten. Erstellen Sie eine weitere Ebene *actions*. Fügen Sie dort ActionScript-Code ein der eine Instanz von *ClockCounter* erstellt. Jede Sekunde soll *incSeconds()* aufgerufen werden. Die Drehung des Uhrzeigers soll jeweils auf den Winkel *getAngle()* gesetzt werden. Verwenden Sie dafür die Eigenschaft *_rotation* der Klasse MovieClip (Sie können den Zeiger einfach über seinen Instanznamen ansprechen). Beachten Sie, dass sich MovieClips um den durch ein Kreuz markierten Anfasspunkt drehen.

b) Erstellen Sie ein Textfeld mit dynamischem Text, das die vergangenen Sekunden und Minuten anzeigt. Ordnen Sie dazu dem Textfeld im Eigenschaftsfenster die Eigenschaft *Dynamischer Text* und im Feld *Var* eine Variable zu. Die Variable können Sie dann in ActionScript wie eine normale String-Variable verwenden.

Aufgabe 2: Game of Life

Regeln: Das Spielfeld ist in quadratische Zellen eingeteilt. Jede Zelle kann entweder tot oder lebendig sein. In jeder Runde verändert sich der Zustand der Zellen gemäß folgender Regeln:

- Eine Zelle mit weniger als 2 Nachbarn stirbt (Einsamkeit)
- Eine Zelle mit mehr als 3 Nachbarn stirbt (Überpopulation)
- Eine tote Zelle mit 3 Nachbarn wird lebendig
- Eine lebendige Zelle mit 2 oder 3 Nachbarn bleibt am Leben

Als Nachbarn gelten die 8 Zellen, die senkrecht, waagerecht oder diagonal angrenzen. Es ist zu beachten, dass als Berechnungsgrundlage für alle Zellen der Zustand der letzten Runde gilt, d. h. in jeder Runde werden zunächst die neuen Zustände aller Zellen berechnet und erst dann werden etwaige Zustandsänderungen ausgeführt.

Realisieren Sie eine einfache Implementierung des Game Of Life, bei der die Zellen in der ersten Runde zufällig initialisiert werden und anschließend das Spiel beliebig lange abläuft (auf Benutzerinteraktion soll hier zunächst verzichtet werden).

Erstellen Sie dazu die zwei folgenden ActionScript-Klassen:

Die Klasse *Cell* repräsentiert eine einzelne Zelle. Sie spezialisiert die Klasse *MovieClip*.
Eigenschaften:

- *alive:Boolean* - Zustand der Zelle (tot oder lebendig)
- *hasChanged:Boolean* – eine Zustandsänderung wurde berechnet (und noch nicht ausgeführt)

Methoden:

- *setChanged():Void* – benachrichtigt die Zelle, dass eine Zustandsänderung für die Zelle berechnet wurde
- *performChange():Void* – wird zu Beginn der neuen Runde aufgerufen und führt die Zustandsänderung der Zelle durch, sofern eine Änderung gesetzt wurde
- *isAlive():Boolean* – gibt den Zustand der Zelle zurück

Die Klasse *CellArea* repräsentiert das Spielfeld und verwaltet die einzelnen Zellen. Die Größe des Spielfeldes soll in Konstanten festgelegt werden.
Klassenvariablen (Konstanten):

- *MAX_CELL_X:Number* – Anzahl Zellen in X-Richtung
- *MAX_CELL_Y:Number* – Anzahl Zellen in Y-Richtung
- *CELL_SIZE:Number* – Seitenlänge einer Zelle in Pixel

Eigenschaften:

- *Cells:Array* – Zweidimensionales Feld, das die Zellen enthält (Array of Array of Cell)

Methoden:

- *initArea():Void* – initialisiert das Feld mit Zellen, die sich in einem zufälligen Zustand befinden
- *calculateChanges():Void* – berechnet, welche Zellen ihren Zustand ändern werden und benachrichtigt diese
- *performChanges():Void* – benachrichtigt alle Zellen, dass etwaige Zustandsänderungen nun ausgeführt werden sollen
- *isAlive(x:Number, y:Number):Void* – Hilfsmethode, die den Zustand einer Zelle liefert und dabei überprüft, ob die Indizes *x* und *y* gültig sind. Zellen, die ausserhalb des Spielfeldes liegen, sollen der Einfachkeit wegen stets als tote Zellen betrachtet werden.

Erstellen Sie einen MovieClip der die Ansicht einer Zelle repräsentiert. Erstellen Sie Schlüsselbilder, um die beiden Zustände (tot, lebendig) repräsentieren (z. B. unterschiedliche Farben) und Übergänge zwischen den Zuständen (z. B. Tweening). Bei der Erstellung des MovieClips geben Sie im Dialogfenster *Export nach ActionScript* und den Klassennamen *Cell* an (im Dialog, in dem der Name des Movieclips eingegeben wird, unter der Schaltfläche *Erweitert*). Dadurch wird der Movieclip mit der Klasse *Cell* assoziiert und bei der Erstellung einer Instanz automatisch eine Instanz von *Cell* erstellt.
Um in der Methode *initArea()* der Klasse *CellArea* die Zellen zu erstellen, verwenden Sie z. B. die Methode *attachMovie()* der Klasse *MovieClip* (siehe Hilfe => ActonScript-Lexikon => MovieClip). Die Hauptzeitleiste kann mit *_root* angesprochen werden, d. h. mit *_root.attachMovie(...)* fügen Sie die Zellen der Hauptzeitleiste hinzu. Hilfreich ist die Methode *getNextHighestDepth()*, um den MovieClip-Instanzen eine Tiefe zuzuordnen.

Um den Zellen einen zufälligen Zustand zuzuweisen, verwenden Sie die Methode *random* der Klasse *Math*. Um die Zellen bei Zustandsänderung zu animieren, rufen Sie die Methode *gotoAndPlay(...)* der Klasse *Cell* auf (Cell soll als eine Unterklasse der Klasse *MovieClip* definiert sein).