

Zeitfunktionen in GMS2

Description

GameMaker Studio 2 verfügt mit der Sprache GML über zahlreiche Zeitfunktionen. Diese können für Spiele und Anwendungen äußerst nützlich sein. Dieses Tutorial zeigt eine kleine Einführung und den Umgang mit den wichtigsten Funktionen.

Zeitziel

In Spielen und Anwendungen gibt es zahlreiche Gründe Zeit und Datum zu erfassen und anzuzeigen. Ihr wollt messen, wie lange der Spieler das Spiel gespielt hat und dies in der Statistik anzeigen? Ihr wollt das Datum erfassen, an dem das Spiel zuerst gestartet wurde?

Es gibt viele andere Gründe, warum man sich mit den Funktionen befassen sollte. Vielleicht will man nach ein paar Spielminuten eine Umfrage starten oder eine offene Betaversion soll nur bis zu einem bestimmten Datum funktionieren. Daten können mit Hilfe von Zeitabfragen besser synchronisiert werden. In manchen Spielen ist eine korrekte Begrüßung des Spielers je nach Tageszeit interessant. Im Menü kann man die aktuelle Uhrzeit angeben, damit sie der Spieler im Blick hat. Das Erfassen von Wochentagen könnte eine Meldung (»Endlich Wochenende!«) hervorrufen und vielleicht habt ihr auch schon daran gedacht zu Weihnachten eine spezielle Meldung im Spiel anzuzeigen.

Datum und Uhrzeit

Der GameMaker bietet viele Funktionen und Variablen an, auf die wir zugreifen können. Sie werden in *Zeitvariablen* und *Zeitfunktionen* unterteilt.

Mit **date_set_timezone** lässt sich im Spiel eine Zeitzone bestimmen, die mit **date_get_timezone** abgefragt werden kann. Unterschieden wird lediglich zwischen *timezone_local* und *timezone_utc*. Local bezieht sich auf das Gerät, auf dem das Spiel läuft. UTC ist die koordinierte Weltzeit. Die einzelnen Zeitzonen muss man dann entsprechend selbst einrichten.

Am häufigsten werden die Zeitvariablen gebraucht, auf die ich vertieft eingehen möchte. Die

Zeitfunktionen ermöglichen es Zeiten zu definieren, zu vergleichen oder herauszulesen, etwa den Tag bei einer bestimmten Datumsangabe.

Zeitvariablen

Wir werden ein paar nützliche Skripte erstellen, anhand wir den Umgang mit den Variablen besser verstehen lernen. Da ich etwas pingelig bin, zeige ich Zeit und Datum mindestens als zweistellige Zahl an, also statt „7:6 Uhr“ lieber „07:06 Uhr«. Das heißt, dass wir einstellige Zahlen in einen zweistelligen String umwandeln müssen. Da das ziemlich oft vorkommen kann, machen wir ein eigenes Skript mit dem Namen **scr_zahl_zweistellig**.

```
/// scr_zahl_zweistellig(zahl);
// Wandelt eine einstellige Zahl
// in einen zweistelligen String um

var zahl = argument0;
var text = „";

if (zahl < 10)
{
 text = „0" + string(zahl);
} else {
 text = string(zahl);
}

return text;
```

Nun kommen wir zur aktuellen Uhrzeit. Dafür bietet GMS2 drei Variablen an. *current_hour*, *current_minute* und *current_second*. Nachfolgend das Skript **scr_draw_current_time**.

```
/// scr_draw_current_time(x, y);
// Zeigt die aktuelle Uhrzeit digital mit
// zweistelligen Zahlen an
var xx = argument0;
var yy = argument1;

var hour = scr_zahl_zweistellig(current_hour);
var minute = scr_zahl_zweistellig(current_minute);
var second = scr_zahl_zweistellig(current_second);

draw_text(xx, yy, hour + „:" + minute + „." + second);
```

Hier sehen wir auch, wie das Skript **scr_zahl_zweistellig** angewandt wird. Skripte, die wir „scr_draw…“ nennen, rufen wir im Draw-Event auf. Im Spiel / Menü zeigen wir die aktuelle Uhrzeit dann so an:

```
draw_set_color(c_white);
draw_set_halign(fa_center);
draw_set_valign(fa_middle);

draw_set_font(fnt_clock);
// Aktuelle Uhrzeit zeichnen
```

```
scr_draw_current_time(room_width/2, 60);
```

Die Formatierung (Farbe, Ausrichtung, Schriftart) kommt also nicht in das Skript sondern in den Event. So ist das Skript sehr flexibel und muss nicht bei jedem neuen Projekt angefasst werden.

Datum und Begrüßung

Für das Datum gibt es ebenfalls drei Variablen, die uns GMS zur Verfügung stellt. *current_day*, *current_month* und *current_year*. Das aktuelle Datum zeigen wir mit **scr_draw_current_date** an.

```
/// scr_draw_current_date(x, y);
// Zeigt das aktuelle Datum als
// zweistelligen Zahlen an
var xx = argument0;
var yy = argument1;

var month = scr_zahl_zweistellig(current_month);
var day = scr_zahl_zweistellig(current_day);

draw_text(xx, yy, day + „." + month + „." + string(current_year));
```

Das Jahr wandeln wir natürlich nicht um, da die Zahl noch sehr lange vierstellig bleiben wird.

Im Draw-Event kann das Skript dann wie folgt aufgerufen werden:

```
draw_set_font(fnt_date);
// Aktuelles Datum zeichnen
scr_draw_current_date(room_width/2, 100);
```

Nun wäre es toll, wenn wir den Spieler, je nach Tageszeit, richtig begrüßen würden. Dafür erstellen wir ein Skript mit dem Namen **scr_draw_begruessung**.

```
/// scr_draw_begruessung(x, y, time);
// Gibt eine Begrüßung passend zur Tageszeit aus
// Kann natürlich beliebig erweitert werden
// Im Beispiel wird die aktuelle Uhrzeit des Systems
// übergeben, im Spiel kann die anders aussehen.

var xx = argument0;
var yy = argument1;
var time = argument2;

var text = „"

if (time > 0) && (time < 10)
{
 text = „Guten Morgen!";
} else if (time >= 10) && (time < 18) {
 text = „Guten Tag!";
} else {
 text = „Guten Abend!";
}
```

```
draw_text(xx, yy, text);
```

Das Skript bekommt eine Uhrzeit mitgeteilt (*argument2*). Wenn wir die aktuelle Zeit übergeben wollen, rufen wir es mit *scr_draw_begruessung(room_width/2, 135, current_hour);* auf. Nun wird geprüft, ob die Zeit zwischen 0 und 10 liegt (»Guten Morgen!«), zwischen 10 und 18 (»Guten Tag!«) oder nach 18 Uhr (»Guten Abend!«). Das kann man gerne weiter verfeinern und die Texte durch Variablen (Thema Mehrsprachigkeit) oder lokale Begrüßungen (»Servus!«) ergänzen.

Aktueller Wochentag

Mit **scr_draw_wochentag** können wir den aktuellen Wochentag anzeigen. GMS gibt uns hierfür die Variable *current_weekday* zur Hand. Wir erhalten daraufhin eine Zahl zwischen 0 (Sonntag) und 6 (Samstag). Diese Zahl müssen wir dann umwandeln.

```
/// scr_draw_wochentag(x, y, day);
// Gibt den aktuellen Wochentag als Text zurück
var xx = argument0;
var yy = argument1;
var day = argument2;

var text = „"

switch(day)
{
 case 0: text = „Sonntag"; break;
 case 1: text = „Montag"; break;
 case 2: text = „Dienstag"; break;
 case 3: text = „Mittwoch"; break;
 case 4: text = „Donnerstag"; break;
 case 5: text = „Freitag"; break;
 case 6: text = „Samstag"; break;
}

draw_text(xx, yy, „Heute ist „ + text);
```

Das Skript rufen wir mit *scr_draw_wochentag(room_width/2, 200, current_weekday);* auf. Natürlich muss es sich dabei nicht um den wirklichen Wochentag handeln. In einer Wirtschaftssimulation kann das ein virtueller Tag sein. Das gilt selbstverständlich für alle Skripte.

Zeit seit Programmstart

In Statistiken speicher ich gerne, wie oft das Spiel gestartet wurde und wie lang das Spiel bereits läuft. Letzteres kann man mit *get_timer()* abfragen. GameMaker nimmt es hier ganz genau und gibt die Zeit als 1 Million Mikrosekunden pro Sekunde an. Um die Zeit im Spiel anzuzeigen, brauchen wir kein eigenes Skript, es reicht bereits eine Zeile:

```
draw_text(room_width/2, 170, „Das Programm laeuft seit „ + string(round(get_ti
```

Der Schlüssel liegt bei *round(get_timer()/1000000)*. Hier wird der Wert kaufmännisch gerundet, man

könnte es aber auch mit *floor(get_timer()/1000000)* anzeigen.

In einer Statistik sollte man es genau nehmen und die Zahl komplett speichern, die uns *get_timer()* liefert. In der entsprechenden INI wird der vorherige Wert ausgelesen und *get_timer()* aufaddiert.

Zeitspanne

In GMS2 kann man Zeiten vergleichen und so beispielsweise bestimmen, welches Datum früher ist oder die Differenz zwischen zwei Daten berechnen. Wenn wir den ersten Spielstart gespeichert haben, können wir im Spiel anzeigen, wie viele Tage das her ist. Das folgende Beispiel zeigt an, wie viele Minuten es her ist, dass Mitternacht war:

```
var diff = floor(date_minute_span(date_create_datetime(current_year, current_m
draw_text(room_width/2, 230, „Minuten seit Mitternacht: „ + string(diff));
```

Die Funktion **date_minute_span** gibt die Anzahl der Minuten zwischen zwei Datumsangaben wieder. Im Beispiel bezieht es sich auf den aktuellen Tag, man könnte aber so auch ermitteln, wie viele Minuten ein Mensch alt ist. *date_current_datetime()* gibt das aktuelle Datum mit Uhrzeit zurück. Allerdings ist der Rückgabewert nicht formatiert sondern lediglich eine lange Zahl. Mit entsprechenden GM-Funktionen können wir aber das herauslesen, was wir brauchen:

```
var stunde = date_get_hour(date_current_datetime());
```

Das Beispiel zeigt die aktuelle Stunde an. Also im Prinzip das Gleiche, was wir mit *current_hour* erhalten. Wenn wir das Datum speichern, wie den ersten Spielstart, dann machen wir das am besten mit *date_current_datetime()* und brauchen dafür nur eine Variable. Mit den GM-Funktionen können wir daraus die Werte herauslesen, die wir anzeigen möchten. Jahr, Monat, Tag, Stunde, Minute und Sekunde.

Wie eingangs erwähnt, bietet GMS2 zahlreiche Funktionen an. Dieses kleine Tutorial bietet nur einen Einstieg und ich kann jedem nur empfehlen, sich mit den nützlichen Funktionen intensiver zu befassen. Möglichkeiten zur Anwendung gibt es viele.

Date Created
3. Juli 2018
Author
sven