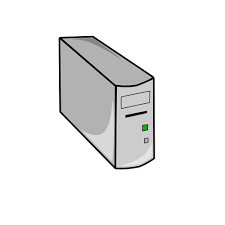
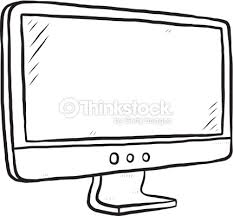
SFML #4 - User Input

Bisher sind unsere Programme mehr Video als Spiel, da sie immer gleich ablaufen und man nicht mit ihnen Interagieren kann. Das wollen wir jetzt ändern. Dazu fügen wir unseren Programmen User Input hinzu. Der Benutzer/Spieler kann also Einfluss auf den Verlauf des Programmes nehmen.

Input

Output







Der klassische User Input sind Maus und Tastatur. Deswegen werden wir uns auf diese Beschränken. Um mit User Input zu arbeiten, gibt es 2 Möglichkeiten in SFML: Über Abfrage und über Events.

## 

## User Input über Abfrage

User Input über Abfrage ist meistens einfacher als über Events, hat aber auch seine Nachteile. Wir können in SFML über die Keyboard und Maus Klassen die Zustände der Tasten bzw. der Mausposition abfragen, und demnach handeln. Dazu müssen wir in unserer **loop()-Methode** über if-Klammern die einzelnen Zustände abfragen.

Folgende Methoden sind zur Abfrage möglich:

Keyboard.IsKeyPressed(Keyboard.Key.*<Hier Taste Einfügen>*);

Mouse.IsButtonPressed(Mouse.Button.*<Hier Taste Einfügen>*);

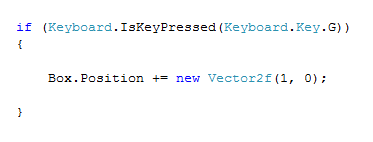
Die beiden Methoden geben "true" zurück wenn die entsprechende Taste gedrückt ist, und false, wenn sie nicht gedrückt ist.

Mouse.GetPosition(game.gameWindow);

Gibt einen Vektor2f zurück, mit den Mauskoordinaten

Wichtig! Da sonst die Mauskoordinaten auf dem Bildschirm und nicht des Fensters genommen werden!

Wenn wir also wollen, dass beim Drücken der Taste "G" ein Objekt sich nach rechts bewegt, würde dass so aussehen:



**Beachte!** User Input durch Abfrage hat den Nachteil, dass es pro Frame 1 Mal abfragt. Wenn man aber eine einmalige Veränderung bei einem Tastendruck möchte, braucht man Events.

## User Input mit Events

In C# kann man EventHandler an Events binden. SFML bietet dazu mehrere Events an, die Member der RenderWindow Klasse ist, auf die wir mit game.gameWindow zugreifen. Die wichtigsten Events sind hier aufgelistet:

game.gameWindow.KeyPressed

// Wird beim Drücken irgendeiner Taste der Tastatur ausgelöst.

(Achtung: Wenn man eine Taste länger gedrückt hält, kommt es zur KeyRepetition, dann wird die Taste schnell hintereinander gedrückt. Man kann dies mit "game.gameWindow.setKeyRepeatEnabled(false)" ausschalten)

game.gameWindow.KeyRelease

// Wird beim loslassen irgendeinerTaste der Tastatur ausgelöst.

game.gameWindow.MouseMoved

// Wird ausgelöst wenn die Maus bewegt wird

game.gameWindow.MouseButtonPressed

// Wird ausgelöst wenn eine Maustaste gedrückt wird

game.gameWindow.MouseButtonReleased

//Wird ausgelöst wenn eine Maustaste losgelassen wird

Wenn wir also auf einen Tastendruck reagieren wollen, machen wir das so: (Für Details zu Events und Eventhandlern, siehe #4.1)

game.gameWindow.KeyPressed += new EventHandler<KeyEventArgs>(OnKeyPress);

}

void OnKeyPress(object sender, KeyEventArgs e)

{

Box.Size \*= 2;

Box.Origin = Box.Size / 2;

}

Jetzt wird, wann immer wir irgendeine Taste auf der Tastatur drücken, die Größe von Box verdoppelt. Wenn wir das ganze jetzt auf eine Taste beschränken wollen, müssen wir den Parameter e benutzen.

void OnKeyPress(object sender, KeyEventArgs e)

{

if (e.Code == Keyboard.Key.G)

{

Box.Size \*= 2;

Box.Origin = Box.Size / 2;

}

}

Der Member "Code" von e, enthält den KeyCode der Taste, die gedrückt wurde. Wir müssen sie also mit einem der "Keyboard.Key" Konstansten vergleichen, um herauszufinden, welche Taste gedrückt wurde. Um genaueres über einzelne Events, deren Parameter und deren Inhalte zu erfahren, kann man sich in der Dokumentation hier:

http://www.sfml-dev.org/documentation/2.3.2/classsf\_1\_1Event.php

Alles anschauen.