Kommandoliste ss2d v1.0 (Build 0.6.20110108.480)

GAMEOBJECTS (inkl. Sprites)

//// Anlegen eines neuen GameObjects (OBJ =Bezeichner, unter dem das Objekt ansprechbar
//// sein soll in JavaScript)
OBJ = ss2d.NewGameObject(id: string)

//// Eigenschaften des GameObjects ändern
OBJ.width = int;
OBJ.height = int;
OBJ.speed = int;
OBJ.rate = int;

//// Position des Objekts ausgeben (Übergabe als Objekt, z.B.: var x = OBJ.GetPosition().x;
//// var y = OBJ.GetPosition().y;
OBJ.GetPosition();

//// Position des Objekts neu setzen
OBJ.SetPosition(x: int, y: int);

//// Objekt bewegen (stetige Bewegung, ohne Ziel)
OBJ.Move (x: int, y: int, [useSpeed: bool]);

//// Bewegung des Objekts anhalten
OBJ.Stop();

//// Sprite laden (wird an der Position des Objekts mit dessen Maßen angezeigt)
OBJ.LoadSprite (src: string);

//// Sprite entfernen
OBJ.RemoveSprite ();

//// Animation auf dem Objekt abspielen (überdeckt bis AnimationStop() das Sprite)
OBJ.AnimationPlay (name: string);

//// Animation beenden (Sprite wird wieder angezeigt)
OBJ.AnimationStop ();

HINWEIS
GameObjects stellen die „Haupteinheit“ des Systems dar. Ohne Sie können weder Sprites noch Animationen dargestellt werden. Des Weiteren ist es durch Sie möglich, den Code weitaus logischer aufzubauen als noch in den Versionen 0.3 oder 0.4.

ANIMATIONEN

//// Anlegen einer neuen Animation
ss2d.Graphics.Animation.Add ({
	id: string,
	steps: int,		<< Animationsschritte
	duration: int,		<< Animationsdauer, sollte ein vielfaches an steps sein!!!
	imageURL: string	<< Adresse des Animationsbildes
});

//// Animation entfernen
ss2d.Graphics.Animation.Remove (id: string);

HINWEIS
Im Vergleich zu früher gelten Animationen nicht mehr ausschließlich für Strings, sondern werden global angelegt. Dadurch kann eine Animation auf jedes vorhandene GameObject gelegt werden. Abgespielt werden kann eine angelegte Animation mittels *OBJ.AnimationPlay (id);* (OBJ steht hierbei für das jeweilige GameObject).

PANORAMA

//// Panorama festlegen
ss2d.Graphics.SetBackground (imageURL: string, [width: int, height: int, x: int, y: int]);

//// Panorama bewegen (Parallax Scrolling)
ss2d.Graphics.MovePanorama (x: int, y: int);

HINWEISE
- Die Panorama-Funktion ist vergleichbar mit beispielsweise den Backgrounds des RPG Makers.
- Ein Panorama, das bewegt wird, erscheint unendlich, da automatisch alle Bereiche außerhalb der aktuellen Position ausgefüllt werden.
- Das Panorama liegt stets in der untersten Ebene

CALLBACKS

//// Neuen Callback anlegen
ss2d.Callback.Register (callbackId: string, fn: function, rate:int, [state: int]);

//// Bestehenden Callback entfernen
ss2d.Callback.Delete (callbackId: string);

//// Anzahl aller angelegten Callbacks ausgeben
ss2d.Callback.GetCallbackCount();

//// Status eines Callbacks ändern
ss2d.Callback.SetCallbackState (callbackId: string, state: int);

//// Status eines Callbacks ausgeben
ss2d.Callback.GetCallbackState (callbackId: string);

//// Aufrufrate eines Callbacks ändern (kleinerer Wert = häufigere Aufrufe; Minimum ist 1)
ss2d.Callback.SetCallbackRate (callbackId: string, rate: int);

//// Aufrufrate eines Callbacks ausgeben
ss2d.Callback.GetCallbackRate (callbackId: string);

KOLLISION

//// Kollision ermitteln
ss2d.DetectCollision (objA: object, objB: object);

HINWEIS
Die Kollisionserkenung benötigt zwei GameObjects, die Reihenfolge, in der diese angegeben werden, ist dabei egal. Bei einer Kollision der Objekte gibt es ein TRUE, ansonsten wird FALSE übergeben.

EINGABE (TASTATUR)

Direkt aus ss2d.Input.keyList[int] abfragen. Array mit allen möglichen Tastencodes.
(Int = Tastencode)
→ gibt *TRUE* zurück sofern entsprechende Taste gedrückt ist, ansonsten *FALSE*

Wichtige Codes:

13	Enter / Return
32	Leertaste
37	Pfeiltaste links
38	Pfeiltaste oben
39	Pfeiltaste rechts
40	Pfeiltaste unten

HINWEIS
Tastencodes können unter http://zero.screensports.de/demos/030/input/ durch Drücken der jeweiligen Taste ermittelt werden.

SOUND

ACHTUNG: Die folgenden Befehle entsprechen dem HTML5-Soundwrapper PlugIn aus ss2d 0.5!

ss2d.Sound.Play(type: string (def:"sound"), id: string (def: ""), loop: bool (def: false));
ss2d.Sound.Stop();
ss2d.Sound.Pause();
ss2d.Sound.Mute();
ss2d.Sound.Load(url: string(def ""), type: string (def: "sound"), id: string(def: ""));
ss2d.Sound.Init(); << *nicht in allen PlugIns enthalten !*

FONT

ss2d.Font.SetPosition(posX: int, posY: int);
ss2d.Font.DrawText (txt: string, posX: int, posY: int);
ss2d.Font.ShowText (int: txtNo);
ss2d.Font.HideText();

//// Liest die angegebene XML-Datei ein und stellt deren Inhalte zur Verfügung
ss2d.Font.LoadXml(file: string);

SONSTIGES

//// Engine initialisieren mit oder ohne Autoload der Plugins Input und RenderCanvas
ss2d.Init([autostart: bool (def: true)]);

//// Ausführung starten und angegebene Funktion aufrufen
ss2d.Run(loadFn: string);

//// Ausführung unterbrechen
ss2d.Stop();

//// Anzahl aller bisherigen Ticks
ss2d.GetTickCount ();

//// Aktuelle Ticks/Sek ausgeben (Aktualisierungen pro Sekunde)
ss2d.GetTicksPerSecond ();

//// Ticks/Sek neu setzen (Aktualisierungen pro Sekunde)
ss2d.SetTicksPerSecond ();

//// Frameskip pruefen
ss2d.GetFrameskip ();

//// Frameskip setzen
ss2d.SetFrameskip (value: bool);

//// Aktuellen maximalen Frameskip ausgeben
ss2d.GetMaxFrameskip ();

//// Maximalen Frameskip setzen
ss2d.SetMaxFrameskip (value: int);

//// Größe des Ausgabebildschirms anpassen
ss2d.SetScreen(width: int, height: int);

//// ss2d-Plugin dynamisch nachladen
ss2d.LoadPlugin(src: string, name: string);

SONSTIGE HINWEISE:
Die Funktion LoadPlugin() muss vor dem Befehl ss2d.Init() aufgerufen werden, da diese nur zur Initialisierung zulässig ist.

Die hier aufgeführten Befehle beziehen sich auf die Standard-Plugins. Andere Plugins können ähnlich aufgebaut sein, allerdings wird hierfür keine Garantie übernommen.

MIGRATIONSHINWEIS < 0.5.x:
Eine direkte Migration von Projekten, die mit Versionen der Engine vor 0.5 erstellt wurden, ist aufgrund der GameObjects nicht mehr möglich. Die Projekte müssen angepasst werden, erfordern nach der Umstellung jedoch deutlich weniger Aufwand bei Änderungen.

MIGRATIONSHINWEIS < 0.6.x:
ss2d.Init() mit aktiviertem Autostart lädt (derzeit) keine Default-Plugins mehr.
Die Funktion *ss2d.Run()* benötigt als Parameter ab sofort den Namen einer Startfunktion des eigenen Projekts. Ein manueller Aufruf kann zu Komplikationen bei Nutzung von *LoadPlugin()* führen.
Die Standard-Plugins werden ab Version 0.6 ohne ss2d-Kürzel angesprochen (z.B. wird aus ss2d.ss2dInput ein ss2d.Input).