Colors

The phosphor color is: R = 85, G = 212, B = 0. : Hex: 0x55D400FF

# Zoom level

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***Device*** | ***Good zoom*** | ***Zoom (W)*** | ***width*** | ***height*** | ***Diagonal*** | ***dpi*** |
| Windows (100%) | 0.4 | 0.4 | 1536 | 1024 | 1846 | 96 |
| Windows (150%) | 0.4 | 0.4 | 1536 | 1024 | 1846 | 144 |
| Windows (150%) | 0.25 | 0.25 | 960 |  |  | 144 |
| Tab | 0.53 | 0.53 | 2048 | 1536 | 2560 | 265 |
| Phone | 0.74 | 0.57 | 2220 | 1080 | 2468 | 422 |

Zoom = width / 3840

MainActor

The MainActor is the top level actor that has any programmatically behavior. It is the only descendant to the stage. It initiates the global functions such as: Head down display, Message display, LUA interface and game status. It changes scenes by removing old objects from the MainActor and creating new ones. In some cases minor scene changing animations are done (fade to white and back, although this is not complete).It creates buttons and handles all events regarding the touch buttons (which is not further described here). It calls LUA scripts to determine next scene and to handle the mission state of the current scene. It also creates and holds the status bars.

Events sent by all MainActor

None.

Events subscribed to by all LandingActor

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| LandingActorTransitToDeepSpaceEvent | When received, the MainActor changes creates a fade-to-white animation and then changes the scene to LUA-defined deep space scene. |
| DeepSpaceSceneTransitToOrbitEvent | When received, the MainActor changes the scene to LUA-defined orbit scene. |
| OrbitSceneLandingComplete | When received, the MainActor changes the scene to LUA-defined landing scene. |
| StatusResourceDepletedEvent | When received this checks what resources are depleted, and if it is the fuel, it stops the flame emitters. |

Common for all CompoundObjects

Events sent by all CompoundObject

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| CompoundObjectCreatedEvent | Event sent when a CompoundObject is created |
| CompoundObjectDeathEvent | Event sent when a CompoundObject is killed |

Behaviours

A behaviour is an attachment to a CompoundObject that provides behaviour for the object. It can be an interface from the user (as in the case of the Leapfrog), an AI controlled entity or a way to react to events like collisions or death.

Scene

A scene is a required, special behaviour of the top most CompoundObject. It handles pan and zoom of the scene view, it handles spawning and death of objects, it handles the user control of the leapfrog.

All the objects of the scene are descendants to the scene object. The z-level of each object is in regard to the scene's actor children.

The scene also holds the Box2D world object and performs sprite position and rotation according to the physical simulation.

Events subscribed to by all Scene

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| SetPanningSettingsEvent | Event to command a certain panning mode and lock on to a specific target. |

The scene behaviour exists different types. The following paragraphs discusses these types.

FreeSpaceActor

The free space scene is executed when in deep space. It provides some bouncing boundaries to keep everything within the scene borders. It also monitors when the leapfrog holds itself within any corner of the boundaries and fires events after a while in that condition.

Required objects

|  |  |  |
| --- | --- | --- |
| ***Object*** | ***Name*** | ***Description*** |
| Leapfrog object | leapfrog1 | The Leapfrog |

Events sent by all FreeSpaceActor

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| DeepSpaceSceneTransitToOrbitEvent | Event sent when the leapfrog have stayed in the bottom right corner of the scene for more than 2.5 seconds. |
| DeepSpaceSceneTransitToHyperEvent | Event sent when the leapfrog have stayed in the top right or top left corner of the scene for more than 2.5 seconds. |

LandingActor

The landing scene is executed when on the ground. It requires access to the "leapfrog1" and "launchsite1" objects to start the sequence but perhaps this should be removed. It does not seem necessary for the scene to decide if a launch sequence should start. Perhaps the mission handling system need to take this responsibility.

Required objects

|  |  |  |
| --- | --- | --- |
| ***Object*** | ***Name*** | ***Description*** |
| Leapfrog object | leapfrog1 | The Leapfrog |
| Launch Site | launchsite1 | A launch site. How does this work if there are multiple launch sites? Can this dependency be removed? TODO: A way for launch sites to register themselves to the scene may be needed. During the registration, the scene can add the event handler. |

Events sent by all LandingActor

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| LandingActorTranstToDeepSpaceEvent | Event sent when the LaunchSequenceCompleteEvent event has been received. |

Events subscribed to by all LandingActor

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| LaunchSiteLeapfrogLandedEvent | LandingScene commands the launch sequence when this is received. This should be done by the mission handler. |
| LaunchSequenceCompleteEvent | When this is received, LandingScene sends the LandingActorTranstToDeepSpaceEvent event. This should be done by the mission system. |

Orbit scenes

The orbit scene is really made up of two scenes: The OrbitScene is the top most object and it is not a descendant to the SceneActor, it inherits directly from the CompoundObject. It instantiate a PlanetActor and creates a window in which the OrbitSpaceScene is shown. The OrbitSpaceScene is an descendant to the SceneActor and works as a close up of the Leapfrog. It contains the leapfrog object.

OrbitScene

Events sent by all OrbitScene

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| OrbitSceneLandingComplete |  |

Events subscribed to by all OrbitScene

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| OrbitSpaceOrbitEstablished |  |

OrbitSpaceScene

Events sent by all OrbitSpaceScene

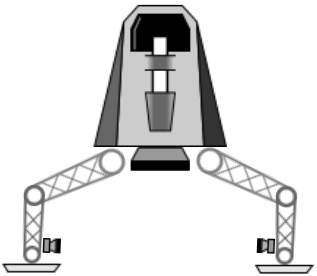
|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| OrbitSpaceOrbitEstablished |  |

Events subscribed to by all OrbitSpaceScene

None.

Leapfrog

The Leapfrog is the main player vehicle.



Required Bodies

|  |  |  |
| --- | --- | --- |
| ***Body*** | ***Name*** | ***Description*** |
| Main Body | lfMainBody | The main body upon where forces are applied |
| Main Booster | lfMainBooster | The booster engine used to attach a FlameEmitter |
| Right Steer Body | lfRightSteer | The right steering engine used to attach a FlameEmitter |
| Left Steer Body | lfLeftSteer | The left steering engine used to attach a FlameEmitter |
| Booster body | lfBooster | The booster body. It is required by the LaunchSite behaviour to attach to |

Required Joints

|  |  |  |
| --- | --- | --- |
| ***Joint*** | ***Name*** | ***Description*** |
| Right Steer/leg Joint | rightSteerJoint | Hold steering engine to leg. Used to destroy the joint |
| Left Steer/leg Joint | leftSteerJoint | Hold steering engine to leg. Used to destroy the joint |
| Right leg joint | rightLegJoint | Hold Big leg to main body. Used to control the mode of the vehicle and to destroy the joint |
| Left leg joint | leftLegJoint | Hold Big leg to main body. Used to control the mode of the vehicle and to destroy the joint |
| Right small leg joint | rightSmallLegJoint | Hold small and big leg together. Used to control the mode of the vehicle and to destroy the joint |
| Left small leg joint | leftSmallLegJoint | Hold small and big leg together. Used to control the mode of the vehicle and to destroy the joint |
| Right foot joint | rightFootLegJoint | Hold small leg and foot together. Used to control the mode of the vehicle and to destroy the joint |
| Left foot joint | leftFootLegJoint | Hold small leg and foot together. Used to control the mode of the vehicle and to destroy the joint |

Required Systems

|  |  |  |
| --- | --- | --- |
| ***System*** | ***Name*** | ***Description*** |
| Gun | lfGun | The gun of the leapfrog. Has ammo. Start/Stop fire interface. |
| Booster Flame Emitter | lfBoosterFlame | The Main Booster Flame particle system. Parameters of it change with the environment. Start/Stop interface. |
| Left Steering Flame Emitter | lfLeftSteerFlame | Steering flame particle system. Parameters of it change with the environment. Start/Stop interface. |
| Shield | lfShield | Attached to main body. Rotates to the hit direction. Not used in Landing environment. |
| Center Reentry Heat Flames | lfCenterReentryFlame | Reentry flame particle system. Only used during reentry. Heat amount interface. |
| Left Reentry Heat Flames | lfLeftReentryFlame | Reentry flame particle system. Only used during reentry. Heat amount interface. |
| Right Reentry Heat Flames | lfRightReentryFlame | Reentry flame particle system. Only used during reentry. Heat amount interface. |

Optional Systems

None

Properties

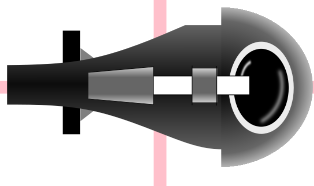
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Property*** | ***Event to set property*** | ***Id*** | ***Read Only*** | ***Description*** |
| Set Mode | LeapfrogExtSetModeEvent | 0 | False | Set the mode of the Leapfrog, i.e. how the legs are rotated.  0 = Reset  1 = Deep Space  2 = Landing  3 = Orbit  4 = Reentry |
| Set Environm. | LeapfrogExtSetEnvEvent | 1 | False | Set the environment of the Leapfrog, i.e. what forces are applied and how the user control affects the Leapfrog.  0 = Deep Space  1 = Ground  2 = Orbit  3 = Launch |
| Read State | N/A | 2 | True | Read the state of the Leapfrog  0 = Normal  1 = Hold Angle |
| Read X-pos | N/A | 3 | True | Read the x-pos |
| Read Y-pos | N/A | 4 | True | Read the y-pos |

Events

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| LeapfrogModeReachedEvent | Event sent when a command to change mode has been completed. |

SteerableObject

The SteerableObject is a deep space vehicle controlled by steering manuevers.



Required Bodies

|  |  |  |
| --- | --- | --- |
| ***Body*** | ***Name*** | ***Description*** |
| Main Body | lfMainBody | The main body upon where forces are applied |

Required Joints

None joints are used.

Required Systems

|  |  |  |
| --- | --- | --- |
| ***System*** | ***Name*** | ***Description*** |
| Gun | lfGun | The gun of the leapfrog. Has ammo. Start/Stop fire interface. |
| Booster Flame Emitter | lfBoosterFlame | The Main Booster Flame particle system. Parameters of it change with the environment. Start/Stop interface. |

Optional Systems

None

Properties

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Property*** | ***Event to set property*** | ***Id*** | ***Read Only*** | ***Description*** |
| Set Steering State | SteerableExtSetSteeringStateEvent | 0 | False | Set the mode of the Leapfrog.  0 = Wander  1 = Seek  2 = Pursuit  3 = Flee  4 = Wander Hunt  5 = Fix |
| Read X-pos | N/A | 1 | True | Read the x-pos |
| Read Y-pos | N/A | 2 | True | Read the y-pos |

Events

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| SteerableHitByBulletEvent | Event sent a bullet hit the steerable object. |
| SteerableHitImpulseEvent | Evetn sent when something bumped the steerable object |

BreakableObject

A BreakableObject is a CompoundObject that takes damage when shot by bullets. The total amount of damage it can take is configurable. Once the damage exceeds the total amount, the object is destroyed. It will then spawn new objects according to its configuration.

Required Bodies

None.

Required Joints

None.

Required Systems

None.

Optional Systems

None.

Properties

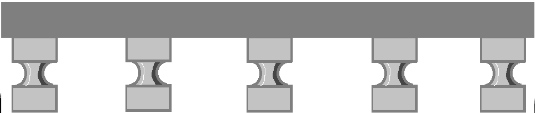
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Property*** | ***Event to set property*** | ***Id*** | ***Read Only*** | ***Description*** |
| Damage left | N/A | 0 | True | Read how much more damage the object can take before breaking. |

Events

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| BreakableCollisionEvent | Event sent a something (incl bullets) hit the breakable object. |
| BreakableSpawnedObjectEvent | Event sent when the Breakable broke and spawned new object. Not called if no objects were spawned. |

Landing Pad

A landing pad is a pad to land on. It can give (or sell) resources or be a trading post. Special handling of resources and trading is TBD.



Required Bodies

None.

Required Joints

None.

Required Systems

None.

Optional Systems

None.

Properties

None.

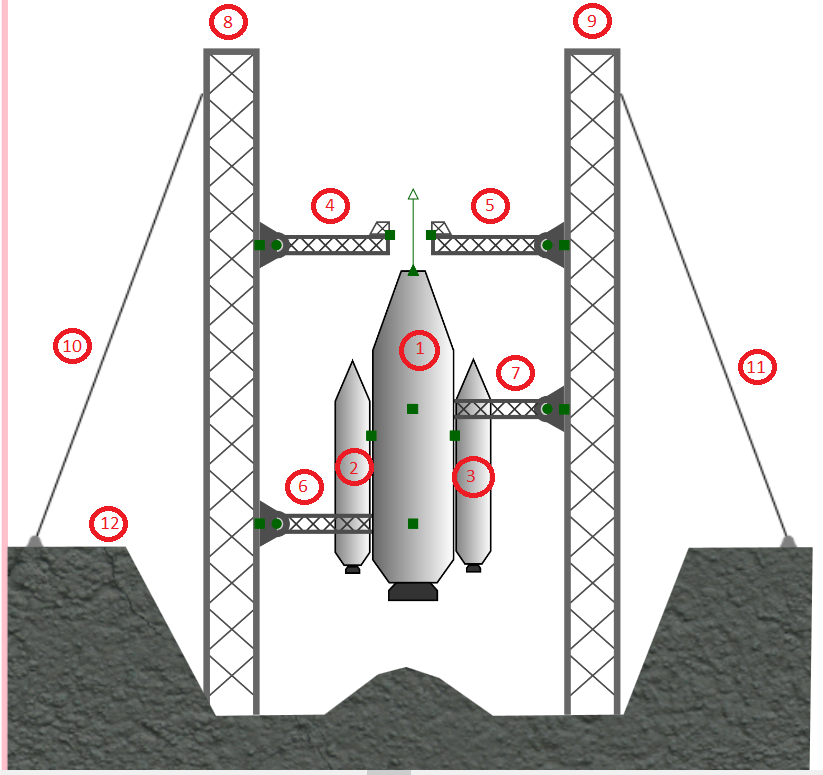
Events

|  |  |
| --- | --- |
| Event | Description |
| ObjectLandedEvent | Event sent when some Object (other than Leapfrog) landed on the pad. |
| ObjectTookOffEvent | Event sent when some Object (other than Leapfrog) left the the pad. |
| LandingPadLeapfrogLandedEvent | Event sent when the Leapfrog landed on the pad, and is stable. |
| LandingPadLeapfrogTakeOffEvent | Event sent when the Leapfrog left the pad. |

§

Launch Site

Installation for launching objects into deep space.



Required Bodies

|  |  |  |
| --- | --- | --- |
| ***Body*** | ***Name*** | ***Description*** |
| Main Tank | mainTank | The main tank (and rocket) |
| Left support booster | leftSupportBooster | The booster rocket on the left side of the main tank |
| Right support booster | rightSupportBooster | The booster rocket on the right side of the main tank |
| The passanger grabber | grabber | The extendable arm that attaches to the passanger object and pulls it to the main rocket. |
|  |  |  |

Required Joints

|  |  |  |
| --- | --- | --- |
| ***Joint*** | ***Name*** | ***Description*** |
| Left rest revolute joint | leftRestJoint | The revolute joint aournd wich the left foot rest rotates |
| Right rest revolute joint | rightRestJoint | The revolute joint aournd wich the right foot rest rotates |
| Left tank holder joint | leftHolderTankJoint | The joint that attaches the main tank to the left (lower) tank holder |
| Right tank holder joint | rightHolderTankJoint | The joint that attaches the main tank to the right (upper) tank holder |
| Left holder trolley revolute joint | leftHolderJoint | The joint around where the left holder rotates once it has detached from the main tank |
| Right holder trolley revolute joint | rightHolderJoint | The joint around where the right holder rotates once it has detached from the main tank |
| Prismatic grabber joint | grabberSpringJoint | The joint that makes the grabber extend and retract. |

Required Systems

|  |  |  |
| --- | --- | --- |
| ***System*** | ***Name*** | ***Description*** |
| Booster flame emitter | boosterFlame | The Main Booster Flame particle system |
| Left booster Flame Emitter | leftFlame | The left booster flame particle system |
| Right booster flame emitter | rightFlame | The right booster flame particle system |

Events

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| LaunchSiteLeapfrogLandedEvent | Event sent when the leapfrog has landed on the launch site |
| LaunchSequenceCompleteEvent | Event sent when the whole launch sequence is complete and the Leapfrog has detached from the main booster |
| LaunchSiteIsPreparedEvent | Event sent when the launch site has prepared itself for launch, attached the vehicle and is about to start its countdown. |

Properties

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Property*** | ***Event to set property*** | ***Id*** | ***Read Only*** | ***Description*** |
| Read State | N/A | 0 | True | Read the state of the Leapfrog  0 = idle  1 = leapfrogLanded  2 = extendedGrabber  3 = lowerFootRest  4 = attachLeapfrgo  5 = countdown  6 = ignite  7 = time zero  8 = release  9 = support booster burnout  10 = drop support boosters  11 = main booster burnout  12 = drop main booster  13 = finish sequence |

Available states:

* idle, boosters and tank attached, Leapfrog has not landed
* extendGrabber, grabber is extended to just below Leapfrog (if there is a prompt for not launching, this is the place)
* grabLeapfrog, Leapfrog is grabbed, i.e. welded to the extended grabber
* lowerFootRests, Foot rests are rotated down
* attachLeapfrog, grabber is retacted pulling Leapfrog with it, Leapfrog goes to Deep Space Mode
* countdown, count down from ten to three
* ignite, start engine flame emitters, countdown to zero
* tZero, apply force to main booster, stay for 0.5 s
* release, release joints holding main tank booster
* prepare, leg rests rotate down, leapfrog is attached to spring which is draged into main tank, (leapfrog goes to deep space mode)
* launch sequence - Stoppable countdown, smoke and small engine fire,
* launch execute - Unstoppable countdown, booster fire, is released and all goes up

ExplosiveObject

An ExplosiveObject is a CompoundObject that can explode sending lots of invisible bullets around as well as an explosion animation. It can be commanded, hit by a bullet or bumped by a high enough impact to explode. The object dies at the explosion.



Required Bodies

None.

Required Joints

None.

Required Systems

None.

Optional Systems

None.

Properties

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Property*** | ***Event to set property*** | ***Id*** | ***Read Only*** | ***Description*** |
| Impact Threhold | N/A | 0 | False | Sets the threshold of impact for when the object explodes. This is initialized from XML but can change by this property. |

Events sent by the object

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| ExplosiveObjectExplodedEvent | Event sent when the object has exploded. |

Events subscribed to by the object

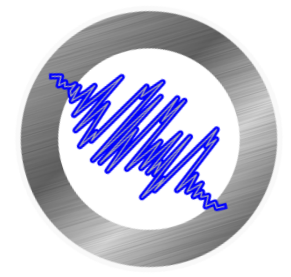
|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| ExplodeCommandEvent | Event to command the explosion independently to being hit or bumped. |

MagneticMine

The MagneticMine is a descendent of ExplosiveObject and can thus explode by command, being hit by a bullet or being bumped. However, it also has the ability to draw itself towards the selected target. It shares the events of the ExplosiveObject.

PickupObject

A PickupObject is an object that can be picked-up by vehicles. When collided with, It gives a resource item or an inventory item and then is destroyed.



Required Bodies

None.

Required Joints

None.

Required Systems

None.

Optional Systems

None.

Properties

None.

Events sent by the object

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| PickedUpEvent | Event sent when something picked the object up. |

Events subscribed to by the object

None.

Systems

Systems are things like particle systems or object factories. They can be stationary or attached to a body. They are initialized by a GameStatus that holds the resources required by the system.

The following systems exists:

* ObjectFactory
* FlameEmitter
* Gun
* Shield
* ReentryFlame

ObjectFactory

An ObjectFactory will spawn new objects (defined by its configuration) at a configurable rate for a configurable lifetime.

Required Bodies

None.

Required Joints

None.

Required Systems

None.

Optional Systems

None.

Properties

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Property*** | ***Event to set property*** | ***Id*** | ***Read Only*** | ***Description*** |
| Num of alive spawns | N/A | 0 | True | Holds the number of currently live spawn objects. |

Events sent by the object

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| FactoryLifetimeExpiredEvent | Event sent when the lifetime of the factory is exceeded. |

Events subscribed to by the object

None.

FlameEmitter

TBD.

Gun

TBD.

Shield

TBD.

ReentryFlame

TBD.

GameStatus

The GameStatus is an object that holds resources for the Leapfrog or other entities. The following resources exists:

* Ammo
* Shield
* Fuel
* Credits
* Damage

Properties

There is an property for each resources type

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Property*** | ***Event to set property*** | ***Id*** | ***Read Only*** | ***Description*** |
| Ammo | N/A | 0 | True | The number of bullets left |
| Shield | N/A | 1 | True | The amount of shield left |
| Fuel | N/A | 2 | True | The amount of fuel left |
| Credits | N/A | 3 | True | The amount of credits accumulated |
| Damage | N/A | 4 | True | The amount of damage left until death |

Events sent by the object

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| StatusChangedEvent | Event sent whenever there is a change in any of the resources. Event contains the resource type and the amount of change. |
| StatusResourceDepletedEvent | Event sent whenever a resource has been depleted. |

Events subscribed to by the object

|  |  |
| --- | --- |
| ***Event*** | ***Description*** |
| TransactResourceCommandEvent | Event to command a specific change in a resource. Used to give or take resource based on mission events. |