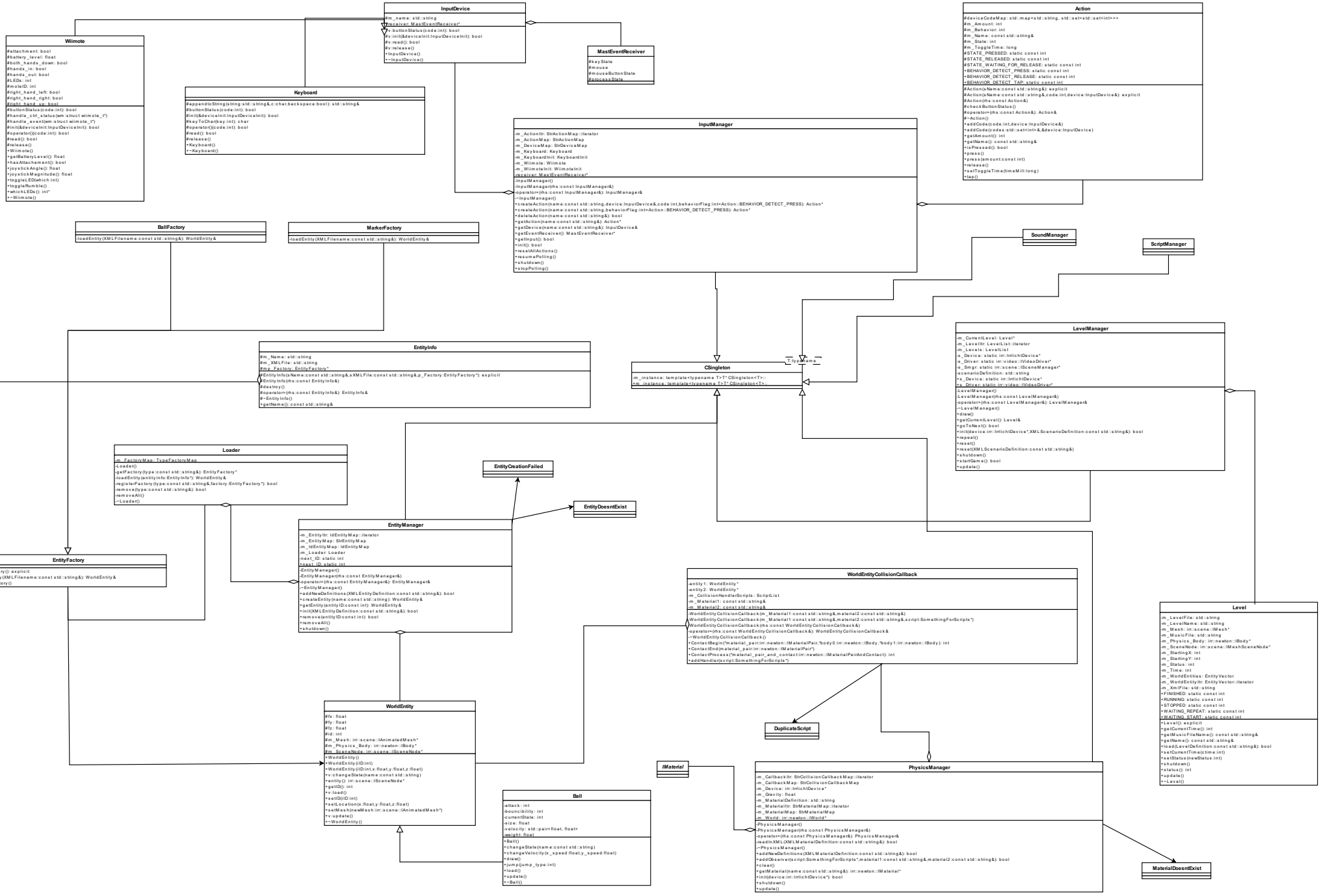
InputDevice
#m_name: std::string
#receiver: MastEventReceiver*
#v:buttonStatus(code:int): bool
#v:init(&deviceInit:InputDeviceInit): bool
#v:read(): bool
#v:release()
+InputDevice()
+~InputDevice()
MastEventReceiver
#keyState
#mouse
#mouseButtonState
#processState
Wiimote
#attachment: bool
#battery_level: float
#both_hands_down: bool
#hands_in: bool
#hands_out: bool
#LEDs: int
#moteID: int
#right_hand_left: bool
#right_hand_right: bool
#right_hand_up: bool
#buttonStatus(code:int): bool
#handle_ctrl_status(wm:struct wiimote_t*)
#handle_event(wm:struct wiimote_t*)
#init(&deviceInit:InputDeviceInit): bool
#operator()(code:int): bool
#read(): bool
#release()
+Wiimote()
+getBatteryLevel(): float
+hasAttachement(): bool
+joystickAngle(): float
+joystickMagnitude(): float
+toggleLED(which:int)
+toggleRumble()
+whichLEDs(): int*
+~Wiimote()
Keyboard
#appendToString(string:std::string&,c:char,backspace:bool): std::string&
#buttonStatus(code:int): bool
#init(&deviceInit:InputDeviceInit): bool
#keyToChar(key:int): char
#operator()(code:int): bool
#read(): bool
#release()
+Keyboard()
+~Keyboard()
Action
#deviceCodeMap: std::map<std::string, std::set<std::set<int>>>
#m_Amount: int
#m_Behavior: int
#m_Name: const std::string&
#m_State: int
#m_ToggleTime: long
#STATE_PRESSED: static const int
#STATE_RELEASED: static const int
#STATE_WAITING_FOR_RELEASE: static const int
+BEHAVIOR_DETECT_PRESS: static const int
+BEHAVIOR_DETECT_RELEASE: static const int
+BEHAVIOR_DETECT_TAP: static const int
#Action(sName:const std::string&): explicit
#Action(sName:const std::string&,code:int,device:InputDevice&): explicit
#Action(rhs:const Action&)
#checkButtonStatus()
#operator=(rhs:const Action&): Action&
#~Action()
+addCode(code:int,device:InputDevice&)
+addCode(codes:std::set<int>&,&device:InputDevice)
+getAmount(): int
+getName(): const std::string&
+isPressed(): bool
+press()
+press(amount:const int)
+release()
+setToggleTime(timeMill:long)
+tap()
InputManager
-m_ActionItr: StrActionMap::iterator
-m_ActionMap: StrActionMap
-m_DeviceMap: StrDeviceMap
-m_Keyboard: Keyboard
-m_KeyboardInit: KeyboardInit
-m_Wiimote: Wiimote
-m_WiimoteInit: WiimoteInit
-receiver: MastEventReceiver*
-InputManager()
-InputManager(rhs:const InputManager&)
-operator=(rhs:const InputManager&): InputManager&
-~InputManager()
+createAction(name:const std::string,device:InputDevice&,code:int,behaviorFlag:int=Action::BEHAVIOR_DETECT_PRESS): Action*
+createAction(name:const std::string,behaviorFlag:int=Action::BEHAVIOR_DETECT_PRESS): Action*
+deleteAction(name:const std::string&): bool
+getAction(name:const std::string&): Action*
+getDevice(name:const std::string&): InputDevice&
+getEventReceiver(): MastEventReceiver*
+getInput(): bool
+init(): bool
+resetAllActions()
+resumePolling()
+shutdown()
+stopPolling()
BallFactory
-loadEntity(XMLFilename:const std::string&): WorldEntity&
MarkerFactory
-loadEntity(XMLFilename:const std::string&): WorldEntity&
SoundManager
ScriptManager
EntityInfo
#m_Name: std::string
#m_XMLFile: std::string
#mp_Factory: EntityFactory*
#EntityInfo(sName:const std::string&,sXMLFile:const std::string&,p_Factory:EntityFactory*): explicit
#EntityInfo(rhs:const EntityInfo&)
#destroy()
#operator=(rhs:const EntityInfo&): EntityInfo&
#~EntityInfo()
+getName(): const std::string&
CSingleton
-m_instance: template<typename T>T* CSingleton<T>::
+m_instance: template<typename T>T* CSingleton<T>::
T:typename
LevelManager
-m_CurrentLevel: Level*
-m_LevelItr: LevelList::iterator
-m_Levels: LevelList
-s_Device: static irr::IrrlichtDevice*
-s_Driver: static irr::video::IVideoDriver*
-s_Smgr: static irr::scene::ISceneManager*
-scenarioDefinition: std::string
+s_Device: static irr::IrrlichtDevice*
+s_Driver: static irr::video::IVideoDriver*
-LevelManager()
-LevelManager(rhs:const LevelManager&)
-operator=(rhs:const LevelManager&): LevelManager&
-~LevelManager()
+draw()
+getCurrentLevel(): Level&
+goToNext(): bool
+init(device:irr::IrrlichtDevice*,XMLScenarioDefinition:const std::string&): bool
+repeat()
+reset()
+reset(XMLScenarioDefinition:const std::string&)
+shutdown()
+startGame(): bool
+update()
Loader
-m_FactoryMap: TypeFactoryMap
-Loader()
-getFactory(type:const std::string&): EntityFactory*
-loadEntity(entityInfo:EntityInfo*): WorldEntity&
-registerFactory(type:const std::string&,factory:EntityFactory*): bool
-remove(type:const std::string&): bool
-removeAll()
-~Loader()
EntityCreationFailed
EntityDoesntExist
EntityManager
-m_EntityItr: IdEntityMap::iterator
-m_EntityMap: StrEntityMap
-m_IdEntityMap: IdEntityMap
-m_Loader: Loader
-next_ID: static int
+next_ID: static int
-EntityManager()
-EntityManager(rhs:const EntityManager&)
-operator=(rhs:const EntityManager&): EntityManager&
-~EntityManager()
+addNewDefinitions(XMLEntityDefinition:const std::string&): bool
+createEntity(name:const std::string): WorldEntity&
+getEntity(entityID:const int): WorldEntity&
+init(XMLEntityDefinition:const std::string&): bool
+remove(entityID:const int): bool
+removeAll()
+shutdown()
EntityFactory
#EntityFactory(): explicit
#v:loadEntity(XMLFilename:const std::string&): WorldEntity&
#~EntityFactory()
WorldEntityCollisionCallback
-entity1: WorldEntity*
-entity2: WorldEntity*
-m_CollisionHandlerScripts: ScriptList
-m_Material1: const std::string&
-m_Material2: const std::string&
-WorldEntityCollisionCallback(m_Material1:const std::string&,material2:const std::string&)
-WorldEntityCollisionCallback(m_Material1:const std::string&,material2:const std::string&,script:SomethingForScripts*)
-WorldEntityCollisionCallback(rhs:const WorldEntityCollisionCallback&)
-operator=(rhs:const WorldEntityCollisionCallback&): WorldEntityCollisionCallback&
-~WorldEntityCollisionCallback()
+ContactBegin(*material_pair:irr::newton::IMaterialPair,*body0:irr::newton::IBody,*body1:irr::newton::IBody): int
+ContactEnd(material_pair:irr::newton::IMaterialPair*)
+ContactProcess(*material_pair_and_contact:irr::newton::IMaterialPairAndContact): int
+addHandler(script:SomethingForScripts*)
Level
-m_LevelFile: std::string
-m_LevelName: std::string
-m_Mesh: irr::scene::IMesh*
-m_MusicFile: std::string
-m_Physics_Body: irr::newton::IBody*
-m_SceneNode: irr::scene::IMeshSceneNode*
-m_StartingX: int
-m_StartingY: int
-m_Status: int
-m_Time: int
-m_WorldEntities: EntityVector
-m_WorldEntityItr: EntityVector::iterator
-m_XmlFile: std::string
+FINISHED: static const int
+RUNNING: static const int
+STOPPED: static const int
+WAITING_REPEAT: static const int
+WAITING_START: static const int
+Level(): explicit
+getCurrentTime(): int
+getMusicFileName(): const std::string&
+getName(): const std::string&
+load(LevelDefinition:const std::string&): bool
+setCurrentTime(ctime:int)
+setStatus(newStatus:int)
+shutdown()
+status(): int
+update()
+~Level()
WorldEntity
#fx: float
#fy: float
#fz: float
#id: int
#m_Mesh: irr::scene::IAnimatedMesh*
#m_Physics_Body: irr::newton::IBody*
#m_SceneNode: irr::scene::ISceneNode*
+WorldEntity()
+WorldEntity(iID:int)
+WorldEntity(iID:int,x:float,y:float,z:float)
+v:changeState(name:const std::string)
+entity(): irr::scene::ISceneNode*
+getID(): int
+v:load()
+setID(iID:int)
+setLocation(x:float,y:float,z:float)
+setMesh(newMesh:irr::scene::IAnimatedMesh*)
+v:update()
+~WorldEntity()
DuplicateScript
IMaterial
PhysicsManager
-m_CallbackItr: StrCollisionCallbackMap::iterator
-m_CallbackMap: StrCollisionCallbackMap
-m_Device: irr::IrrlichtDevice*
-m_Gravity: float
-m_MaterialDefinition: std::string
-m_MaterialItr: StrMaterialMap::iterator
-m_MaterialMap: StrMaterialMap
-m_World: irr::newton::IWorld*
-PhysicsManager()
-PhysicsManager(rhs:const PhysicsManager&)
-operator=(rhs:const PhysicsManager&): PhysicsManager&
-readInXML(XMLMaterialDefinition:const std::string&): bool
-~PhysicsManager()
+addNewDefinitions(XMLMaterialDefinition:const std::string&): bool
+addObserver(script:SomethingForScripts*,material1:const std::string&,material2:const std::string&): bool
+clear()
+getMaterial(name:const std::string&): irr::newton::IMaterial*
+init(device:irr::IrrlichtDevice*): bool
+shutdown()
+update()
MaterialDoesntExist
Ball
-attack: int
-bouncibility: int
-currentState: int
-size: float
-velocity: std::pair<float, float>
-weight: float
+Ball()
+changeState(name:const std::string)
+changeVelocity(x_speed:float,y_speed:float)
+draw()
+jump(jump_type:int)
+load()
+update()
+~Ball()