

Rochard: Hard-mining with Unity

Damien Morello –
Gameplay and Graphics Programmer at Colossal Order



- **Rochard:**
 - Developed by Recoil Games
 - 2.5D side-scroller
 - Action/adventure
 - Puzzle
- Team of about 20 talented people
- Development cycle below 2 years
 - Actual production was a bit over 1 year



Rochard Begins



- **Objective: make a kick ass demo**
- Evaluating 3rd party technologies
- **Unity 2.6** stood out:
 - Fast iteration times
 - C#
 - Memory management
- Too many features, too little time
 - Fast prototyping allowed us to try a lot of concepts and eliminate bad ideas



- Pawn/Controllers
 - Controllers read Unity's **Input** and feed the pawn with an interpretation of the raw input through accessors
 - E.g pawns gameplay code see crouching as a boolean, even though the controller is checking for the vertical analog axis to be smaller than zero
 - Pawns store entities (a character e.g) states
 - states are modified by the active Movement components
 - Pawns also handle their damage effects if any
 - A HittableObject component is added if the Pawn can receive damage



Gameplay elements



- Movement controller
 - Different physics modes/animations split in different components
 - Aim (upper body animation)
 - Use object (physics and upper body animation)
 - Walk/Run forward/backward and jump (physics and animations)
 - G-Swing (physics and animations)
 - Ragdoll (physics)
 - Ladder (physics and animations)
 - Movement controller switches the components when needed



Rochard pawn



The screenshot displays the Unity Inspector for a pawn character, showing a hierarchy of scripts and their properties. The scripts listed include:

- Transform**: Position (X: 109.2141, Y: 7.564972, Z: -11.07874), Rotation (X: 0, Y: 0, Z: 0), Scale (X: 1, Y: 1, Z: 1).
- Hittable Object (Script)**: Script (HittableObject), Default Hit Points (100), Regeneration Rate (75), Regeneration Delay On Damage (4), Full Health Reached Sound (PlayerHealth_Full_Single).
- On Damage Spawner (Script)**: Script (OnDamageSpawner), Hit Sound, Hit Effect, Death Sound, Death Effect, Death Effects Spawn Offset, Reset Death Effect Spawn Rotation, Parent To Containing Object, Copy Rigidbody Velocity To Death, Hit Velocity Scale, Max Hit Force Distance, Max Hit Velocity, Hit Random Angular Velocity, Copy Scale.
- Sideview Character Player Controller (Script)**: Script (SideviewCharacterPlayer), Player ID, Controller ID, Gravity Remote Active Time, Gravity Remote Recharge Time, Delay Before Idle VO, Delay Before Idle Short VO.
- Character Pawn (Script)**: Script (CharacterPawn), Default Model (HERO_ANIMS), Head Bone Name (ORIGIN/Hips/Spine/Spine1/Spi), Team, Pawn Id.
- Inventory (Script)**: Script (Inventory), Default Inventory, Default Weapon Identifier, Default Inventory Identifiers.
- PMSideview Character Incapacitate (Script)**: Script (PMSideviewCharacterIncapacitate), Animation Recovery Acceleration, Max Recovery Acceleration.
- PMSideview Character Normal (Script)**: Script (PMSideviewCharacterNormal), Walk Acceleration, Run Input Threshold, Run Speed, Turn Speed, Walk Speed Modifier, Crouch Speed Modifier, Edge Jump Offset, Jump Height, Allow Jump After Grounded, Allow Jump After Grounded Low G, Crouched Height, Collider Height Jump, In Air Movement Modifier, Hard Landing Speed, Normal Landing Speed, Terminal Velocity, Low Gravity Animation Name Prefix (LowG_), In Air Friction, Low Gravity Jump Velocity, Low Gravity Speed Modifier, Low Gravity In Air Movement Mod, Low Gravity Jump Direction Bias.
- CMSideview Character Aim (Script)**: Script (CMSideviewCharacterAim), Turn Speed, Aim Integration Speed, Aim Transition Speed, Aim Transition Threshold, Max Focal Point Range, Low Gravity Turn Speed Modifier, Use Snap To Enemy, Snap To Enemy Max Radius, Snap To Enemy Max Angle, Use Weapon Hand IK, Gun Bone Name, Gun Arm Bone Name.
- CMSideview Character Melee (Script)**: Script (CMSideviewCharacterMelee), Hit Velocity, Melee Damage, Melee Range, Melee Damage Type, Melee Bone Name.
- Sideview Character Movement Controller (Script)**: Script (SideviewCharacterMovementController).
- PMRagdoll (Script)**: Script (PMRagdoll), Death Impulse Force, Death Impulse Multiplier, Ragdoll Gravity Multiplier Normal, Ragdoll Gravity Multiplier Low, Ragdoll Squish Min Velocity, Ragdoll SquishDamage Type.
- PMSideview Character Ladder (Script)**: Script (PMSideviewCharacterLadder), Climb Speed, Strafe Speed, Climb Acceleration, Strafe Acceleration.
- Low Gravity Switch Jump Velocity**: 6.1
- Low Gravity Turn Speed Modifier**: 0.4
- Force Z0**:
- Concussion Damage Amount**: 20
- Hard Landing Damage Amount**: 20
- Movement Effects**: Clothing Sounds (Player_ClothingSounds), Voice Over Sounds (Player_VOSounds), Stress Settings.
- CMSideview Character Use Forward Animation (Script)**: Script (CMSideviewCharacterUseForwardAnimation), Default Animation Name (Use_Generic), Turn Speed (10).
- CMSideview Character Use Hit And Run (Script)**: Script (CMSideviewCharacterUseHitAndRun), Default Animation Name (QuickUse_Generic), Turn Speed (10).
- PMSideview Character Use Advanced (Script)**: Script (PMSideviewCharacterUseAdvanced), Default Animation Name (Use_Generic), Turn Speed (10).
- PMSideview Character Animation Playback (Script)**: Script (PMSideviewCharacterAnimationPlayback).
- Capsule Collider**: Is Trigger, Material (RCLPlayerStopped), Center (X: 0, Y: 0, Z: 0), Radius (0.35), Height (1.65), Direction (Y-Axis).
- Rigidbody**: Mass (1.5), Drag (0), Angular Drag (0.05), Use Gravity, Is Kinematic, Interpolate (None), Collision Detection (Discrete).
- Physics Dummy (Script)**: Script (PhysicsDummy), Grounded, Ground Normal, Ground Point, Ground Right, Ground Velocity, Last Grounding Velocity, Last Right Velocity, Last Left Velocity, Hit Left, Hit Right, Last Time Was Grounded.
- PMSideview Character GSwing (Script)**: Script (PMSideviewCharacterGSwing), Walk Acceleration, Run Input Threshold, Run Speed, Turn Speed, Walk Speed Modifier, Crouch Speed Modifier, Edge Jump Offset, Jump Height, Crouched Height, In Air Movement Modifier, Hard Landing Speed, Normal Landing Speed, Terminal Velocity, Low Gravity Animation Name Prefix (LowG_), In Air Friction, Low Gravity Turn Speed Modifier, Force Z0, Movement Effects, Clothing Sounds, Voice Over Sounds, Swing Force, Stress Settings.
- Cause Squish Damage On Hit (Script)**: Script (CauseSquishDamageOnHit), Damage Type, Min Velocity, Only Crush, Only Affect Pawns, Damage Players, Damage Override, Damage When Grabbed.

Example of a pawn using movement components



Gameplay elements



- Inventory
 - Handles carrying objects
 - Ammunition for grenades
 - Weapons
 - Handles equipping objects
 - Rochard's hat
 - Weapon
 - Handles dropping objects
 - Different types of Dropped objects



- Weapon attachments
 - Weapons are made of attachments
 - Flashlight
 - Aiming laser
 - Implemented the weapon modes as "barrel" attachments
 - Rockblaster = projectile barrel
 - Gravity beam = beam barrel
 - Laser = beam barrel
 - Grenades = explosive barrel based on projectile barrel
 - Weapon checks the game objects in its child hierarchy and enable an attachment if valid conditions are met, input is directly handled in the attachment code

- When prototyped, it was used like this:





- In the final game, it was used like that:





- Custom physics for the characters
 - Unity's **CharacterController** unfitted
 - One way collisions
 - Implementation quite different from a rigid body
 - Used a constrained capsule rigid body instead
- Coroutines
 - Level scripting
 - Semi modular AI



- Loading scene
 - Separate scene in a root game object with a dedicated camera using its own culling layer and high rendering priority (negative depth)
 - Calls `LoadLevelAdditive()` and self-destroy when this asynchronous operation completes
- Helper methods
 - Inherited `MonoBehaviour`, implemented lots of methods and accessors to facilitate casting operations, finding objects in the scene, instantiating objects from templates
 - Used this class as the base to each of our components
 - Ease caching of some objects which remains static



Game framework



- Persistent data/GameManager
 - Create an object by script from a prefab template placed in the Resources folder and call `DontDestroyOnLoad()` on this object
- Room system
 - Optimization for culling entire areas, both graphically and logically



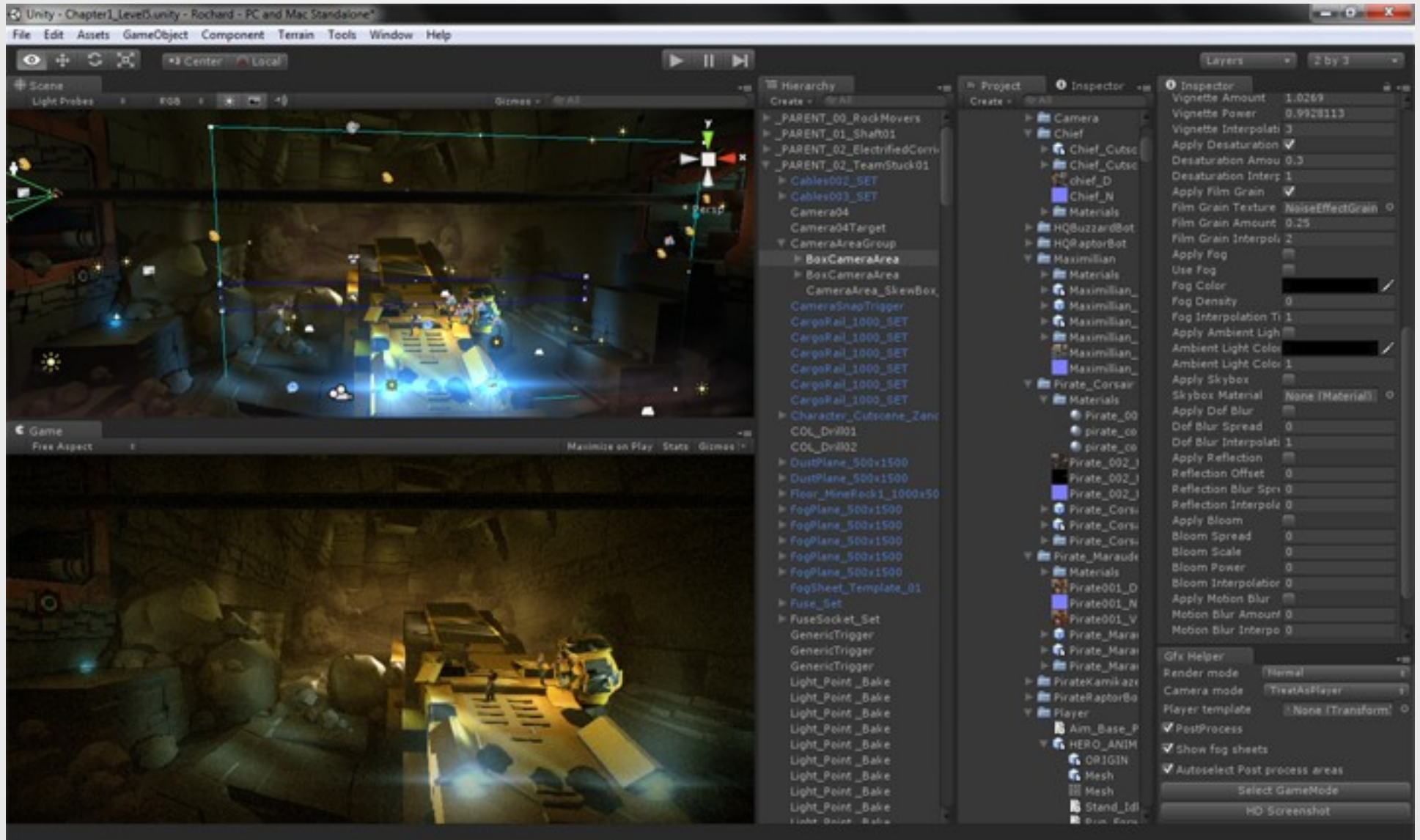
- Created a tool to populate the scene with pre-requisites game objects
- Created multiple helper tools as extension to the Unity Editor for level designers and artists using Unity's base Editor class **EditorWindow**



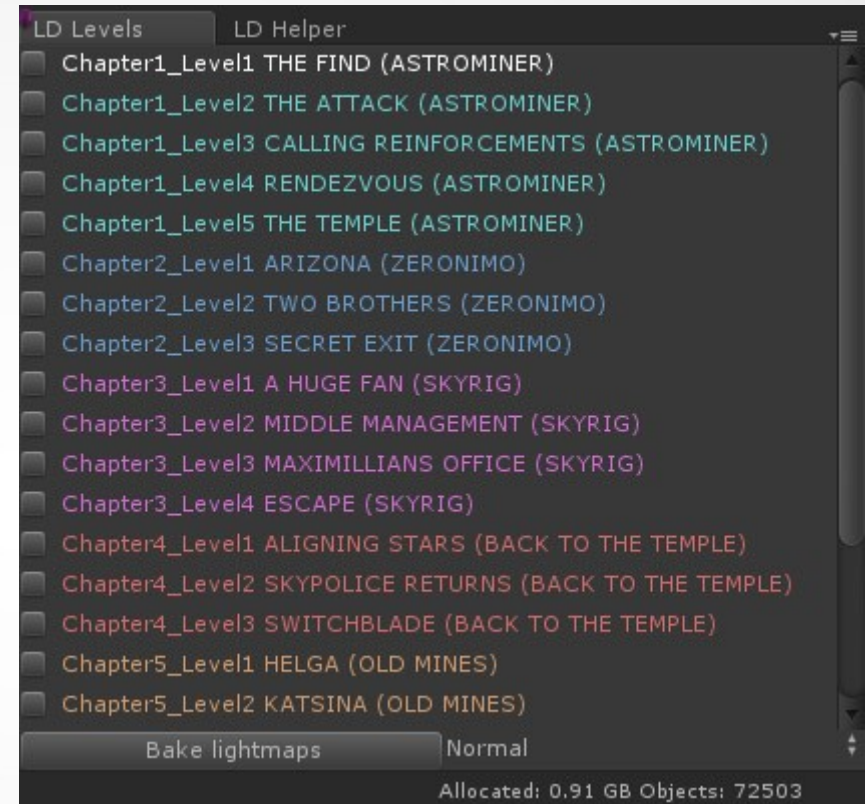
- Level Building Tool
 - Builds the paths
 - Validates certain level constraints (check for mandatory objects and components)
- Ragdolls Wizard (using Unity's [ScriptableWizard](#))
- Gfx Tool
 - Simulates the in-game camera movement according to the Camera areas constraints while still in editor mode
 - Allows to edit post processing settings while the camera is in motion and adjust camera areas
 - Allows to take marketing resolution screenshots from within the editor



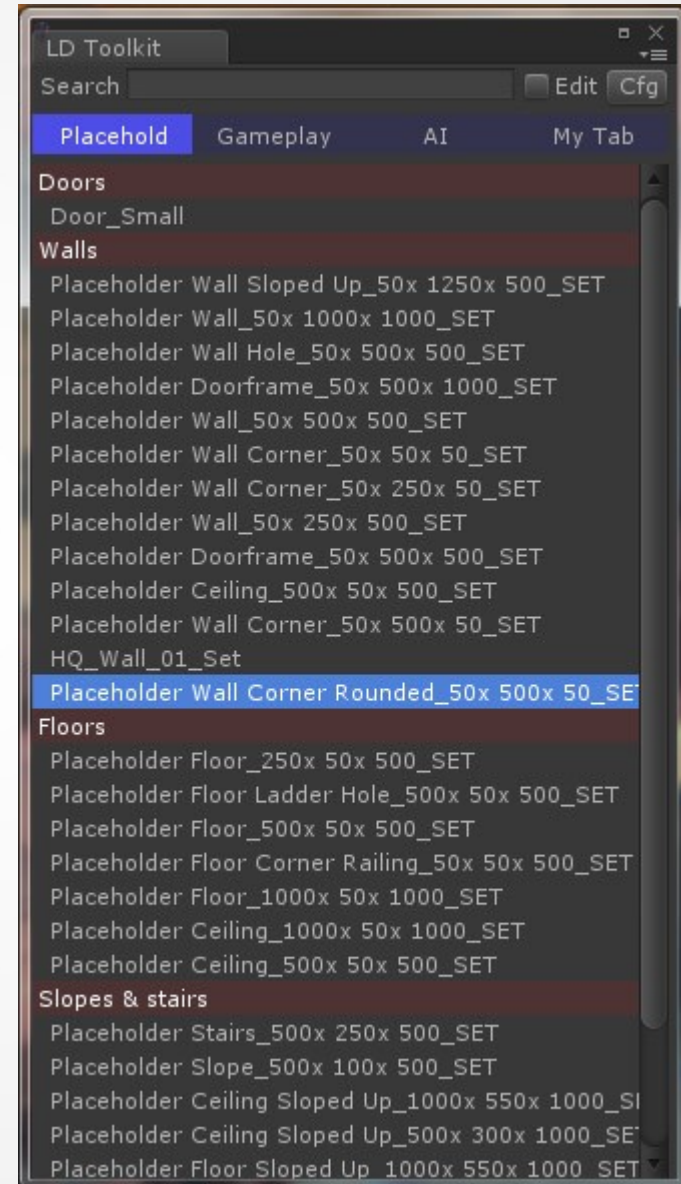
Gfx tool



- Lists all the levels in the game
- Allows to quickly browse through levels
- Allows to build the lightmaps for the selected levels



- Displays a filtered list of assets created by LDs
- Allows to quickly find an object to place on the map
- Automates the position to Z0 and uses grid snapping



- Displays the rooms in the scene
- Allows to easily move objects from a room to another



- There is lot more to say about audio, animations, rendering, visual effects, performances and profiling :)
- Unity was a great choice in reaching our goals
- We benefited from all the updates until 3.5
 - Beast integration was one of the greatest



In the end...



Any questions?

You can contact me at damsku@colossalorder.fi