Overview

- Physics
  - Rigid Body
  - Wheels
  - Cloth
    - Fracture (not covered in live demo)
- Particle Systems
- Navigation (not covered in live demo)
Physics

JUST CAUSE 248
Rigid Body

Stereotypical Barrel Drop Physics Test (UE3)
Rigid Body

Cow Boosting (Just Cause 3)
Rigid Body

- Use Unity Rigid Bodies ([http://docs.unity3d.com/ScriptReference/Rigidbody.html](http://docs.unity3d.com/ScriptReference/Rigidbody.html))
  - Your GameObject should have some collider attached (box, capsule, mesh, sphere, etc.)
  - Give it a PhysicMaterial ([http://docs.unity3d.com/Manual/class-PhysicMaterial.html](http://docs.unity3d.com/Manual/class-PhysicMaterial.html)) to change how rigid bodies interact.

- How to move your rigid bodies? With forces! (Don’t forget: \( F = ma \))
  - RigidBody also has ‘AddForce/Torque’ functions:

- Don’t use the C++ plugin rigid bodies (unless you really, really, really want to).
Rigid Body Constraints

- Use Unity joints (character joints, fixed joints, hinge joints, etc.).

Euphoria Engine Ragdoll
Wheels

- Use wheel collider: http://docs.unity3d.com/Manual/class-WheelCollider.html
- You can now move your vehicle by applying forces (or braking) to your wheels

Need For Speed 2015 Racing
Cloth

- Use Unity’s provided cloth component: [http://docs.unity3d.com/Manual/class-Cloth.html](http://docs.unity3d.com/Manual/class-Cloth.html)
- Your cloth won’t be perfect – it will intersect with things in the world and that’s okay.
Fracture

- Unity does not come with a built-in fracture system (UE4 does). Plugins exist (mainly non-free ones).
- But the concept is fairly simple (implementation not so much):
  - Start off with a static mesh (i.e. a pillar).
  - Upon hitting the mesh with a force (i.e. with a ball), split the entire mesh (or just part of the mesh) using an algorithm (pre-generated or on the spot).
  - Apply forces to the fractured mesh pieces as necessary (i.e using the rigid body techniques from earlier)
Particle Systems

WHAT GOOD ARE EXPLOSIONS IF THEY'RE NOT PRETTY
Particle Systems

Metal Gear Solid 5 Explosion
Particle Systems

Counter Strike Global Offensive Muzzle Flash
Particle Systems

- Particle systems transform according to rules that you give it:
  - Initial velocity, velocity over time, size over time, etc.
  - You also control number of particles to spawn, when they spawn, when they stop spawning, etc.
- Particle system is just a bunch of points moving in space.
- Rendered using textures (each particle gets the same texture).
  - Can use a texture atlas to give particle system the illusion of changing over time (i.e. going from fire to smoke).
- Free particle textures available on the asset store (you will probably find it hard to make your own)!
Navigation

usually easier than drawing paths
NavMesh

- NavMesh concept
  - As a preprocess step, figure out where a character could potentially move to/can not move to (can have different “agents” that represent different character types).
  - At runtime, use an “agent” that queries the NavMesh to get the path to get from point A to point B.
  - You can use the path to do more than just moving a AI character.

Skyrim Clairvoyance Spell (Show Path to Objective)
Live Demo

CREATING
JUST CAUSE 248