APEX Introduction

Jean Pierre Bordes
NVIDIA Corporation

NVIDIA

APEX Mission:
Solve 3 Big Game Physics Problems

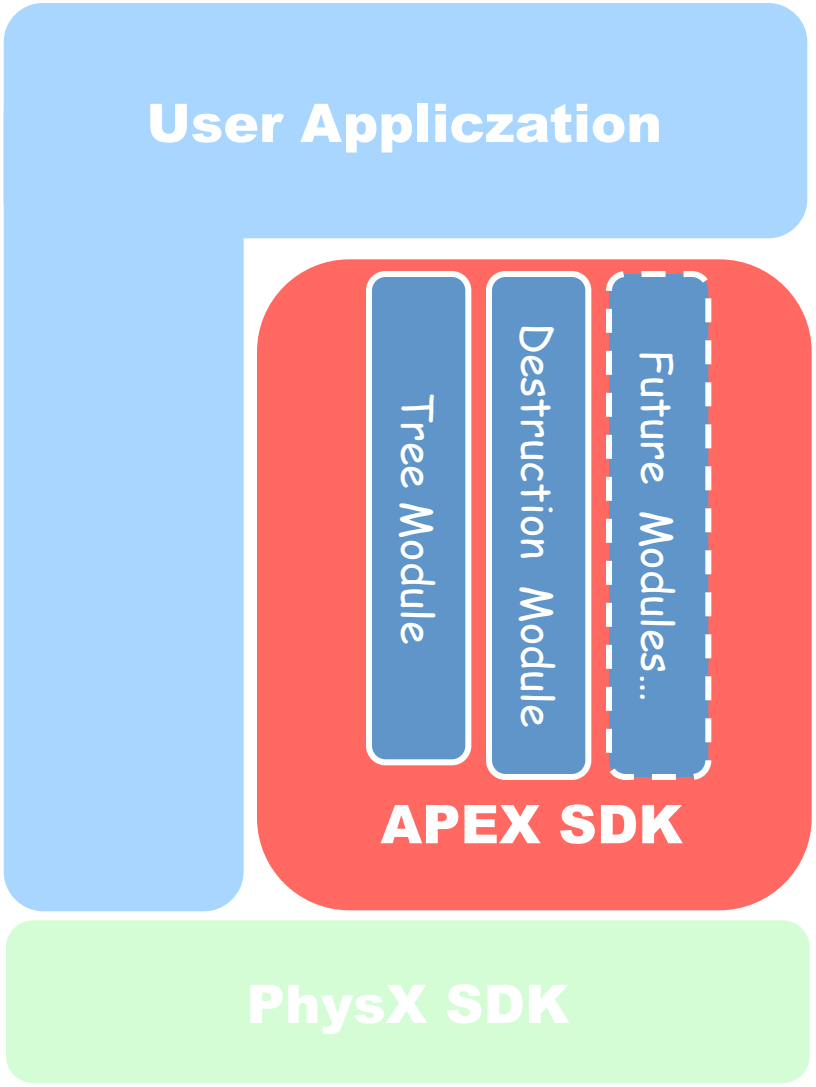
1. Significant programmer involvement

2. Content designed to “min spec”

3. Game engine performance limitations

Basic Design

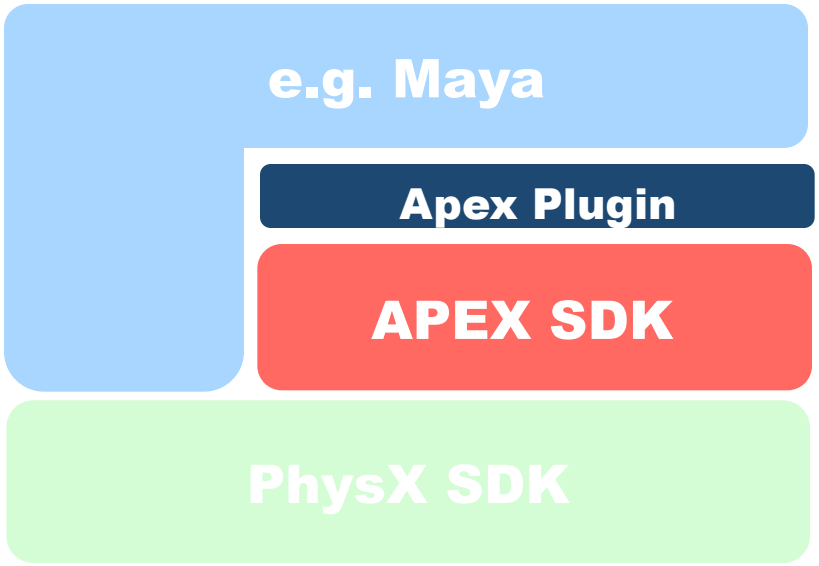
- A collection of “modules”
- Shared interfaces
- Built on top of the PhysX SDK
- User Application:
  - Game (runtime)
  - Authoring tools
  - Level editor

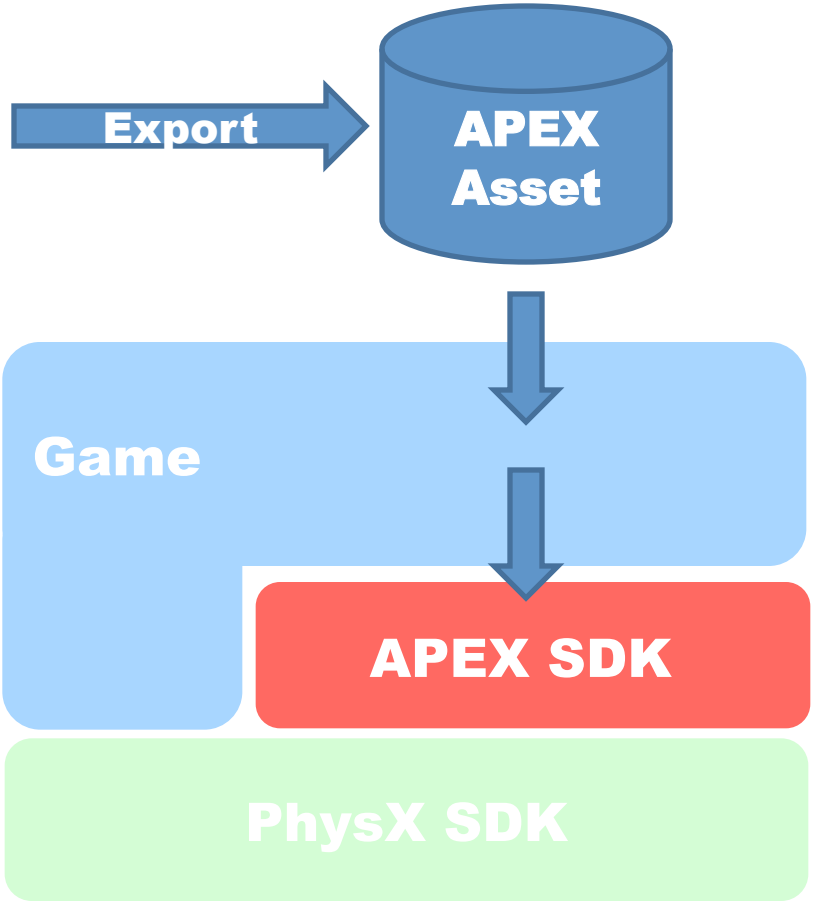


Modules

- Implement intuitive, specific purpose, high-level physics technology
- Level of abstraction appropriate for content creators
  - Low SDK level of abstraction: rigid body, joint, cloth, fluid
  - High APEX level of abstraction (modules): vegetation, character, destructible mesh
- Manage multiple physics elements and simulation types

Authoring





- Modules are provided with authoring tools
- SDK has asset file serialization support built-in (load/store)

Problem 1

- Significant programmer involvement
- APEX Solution: Provide a “high-level” interface to the artists
  - Reduces the need for programmer time
  - Add automatic physics behavior to familiar objects
  - Leverage multiple SDK features

Scaling

- Scaling graphics was easy…
  - Scaling physics, not so much!
- All modules provide load-time and/or run-time scalability mechanisms
  - E.g. Number of APEX objects to have active, total number of debris to keep around
- APEX assets are authored once
  - … to reduce work
  - But the developer still has control over scaling and LOD

Problem 2

- Content designed to “min spec”
- APEX Solution: scalable modules
  - Variable physics “quality” for each physical system
    - Static: hardware capability, player preference
    - Dynamic: visibility, distance from player, etc.

Interface to Rendering System

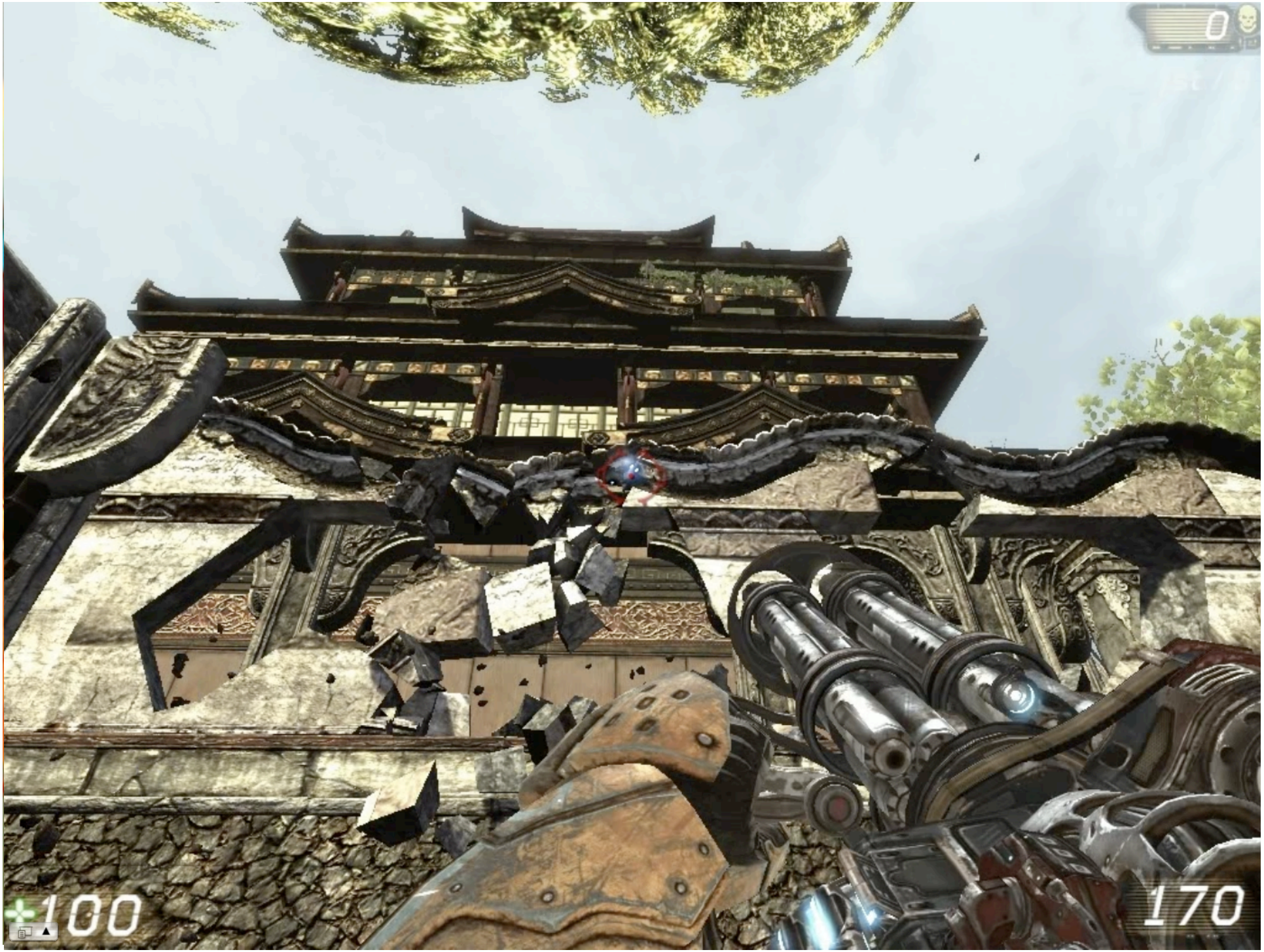
- APEX has a unified API for sending data directly to the rendering engine
  - Shared by all modules
  - Bypass game logic whenever possible
  - Efficient, but flexible
- Application implements a few interfaces
  - APEX objects ask the game to allocate buffers
  - APEX streams rendering data to the buffers
  - Application renders the buffers with appropriate materials, shaders, etc…

Problem 3

- Game engine performance limitations
- Solution: Create a rendering “fast path”
  - Bypass the inefficient, fully generic path
  - PhysX objects in an APEX asset are scriptable / networkable / etc. as a group, not individually

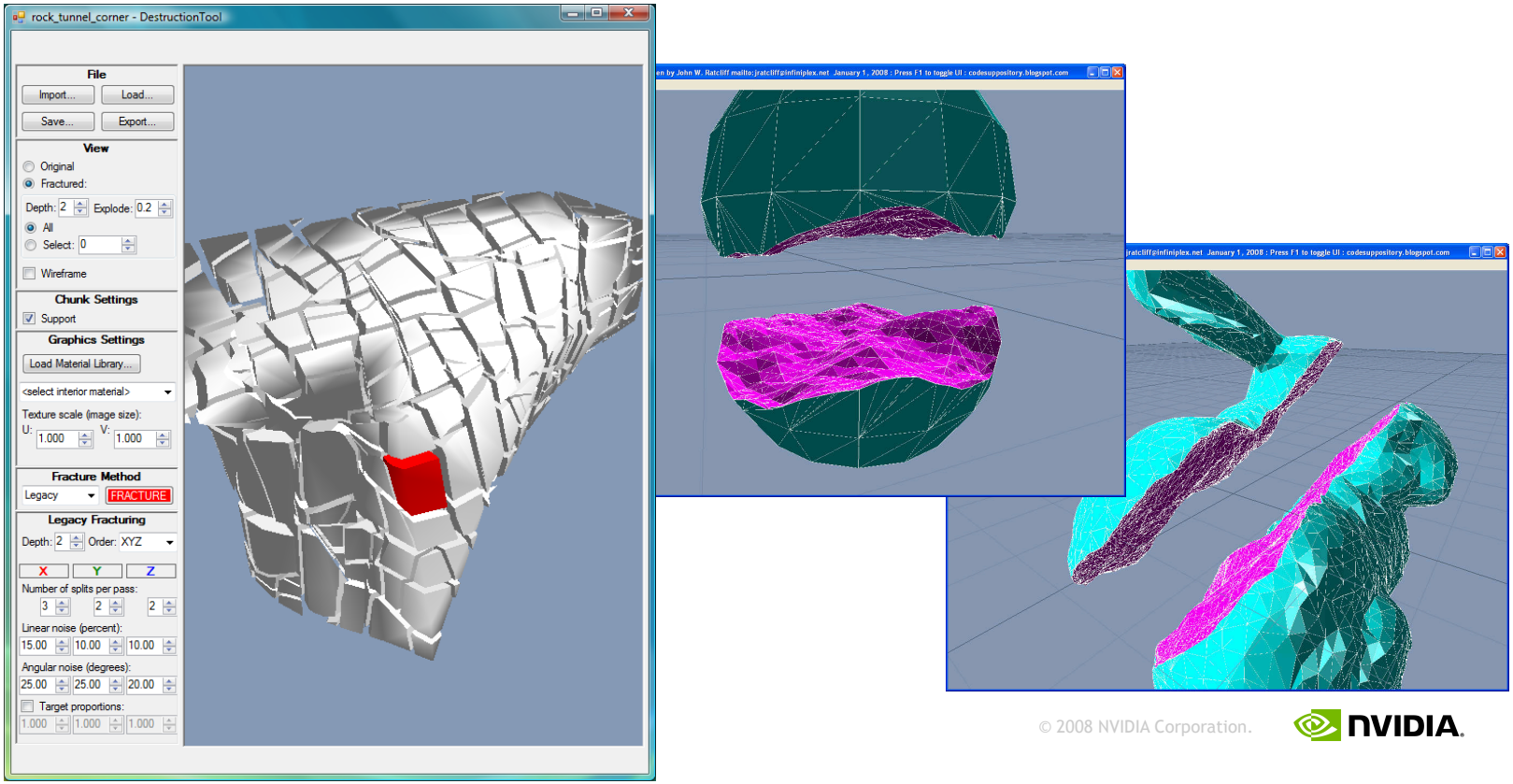
Destruction Module

- Arbitrary meshes are pre-fractured at authoring time
- In real time as they take damage, pieces get blasted away
- Meshes can be initially static or dynamic
- “Support” system for static meshes
- Automatic small debris effect using particles


0
100
170

Destruction Authoring

- Standalone tool, easily integrated into 3dsmax, Maya, or XSI
  - Hierarchical splitting with random fracture surface generation



NVIDIA.

Destruction Scalability

- At authoring time:
  - # of pieces the original asset is split into
- At runtime:
  - Whether to fracture down all the way to the finest level of pre-fracture, or only to a coarser level
  - Option to scale amount of particle meshes
  - Size adjustable global debris FIFO
- The game can change runtime APEX parameters based on LOD.

Tree Module

- Lets us create trees with physical behaviors
- Works with SpeedTree trees
- Full physical interactions
- Tree destruction
- Leaf dropping effect

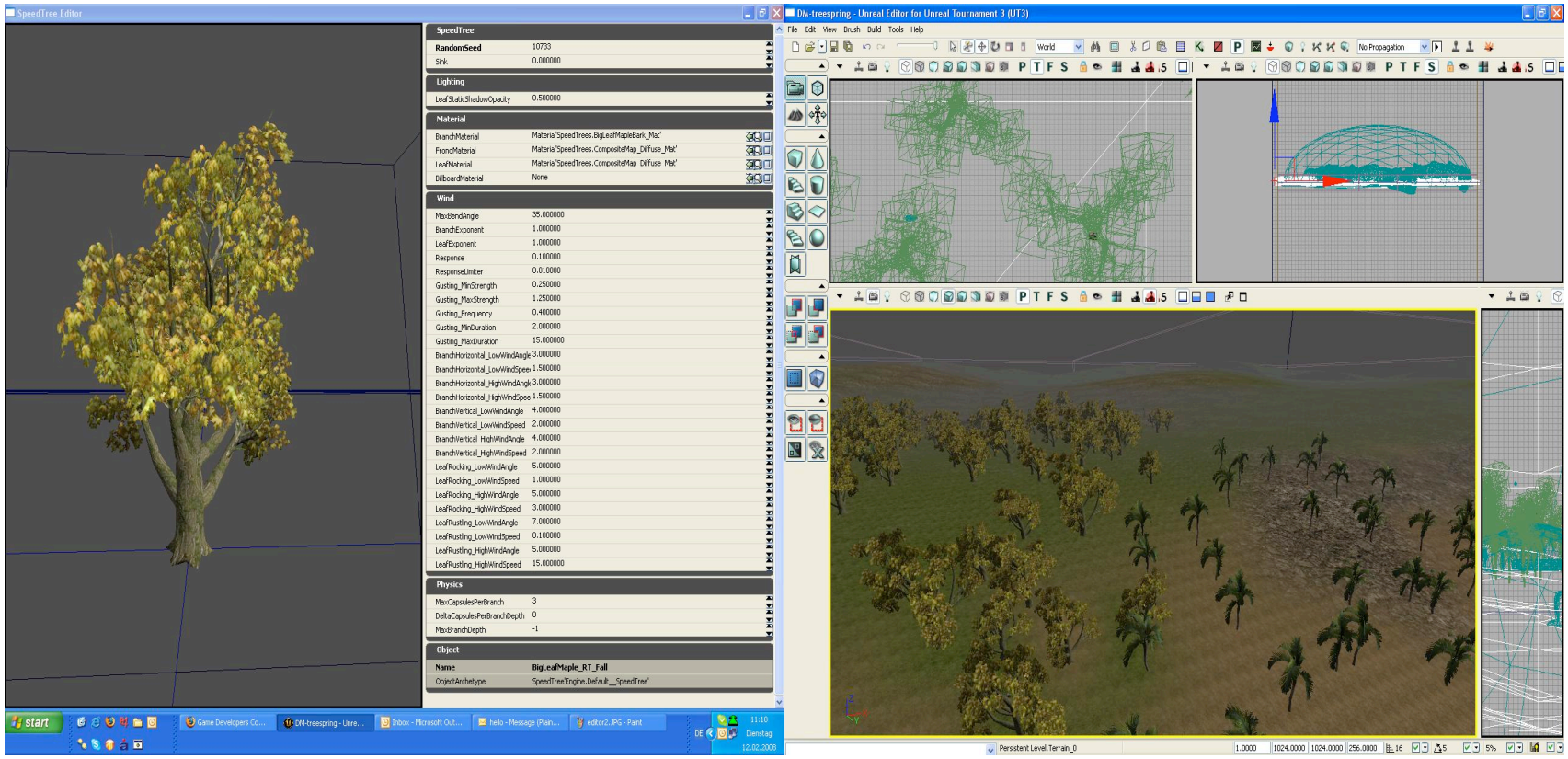
Tree Physics

- Trees skeletons are automatically generated
- Skeletons are only simulated when trees are being interacted with (LOD)
- Emitters are automatically created to spawn leaves


www.fraps.com
0
1st / 0
+100

Tree Authoring

- Authoring process same as for normal trees, except for a few simple physics parameters
  - Capable of directly loading SpeedTree format files



Tree Scalability

- At load / authoring time:
  - A particular tree asset can have variable detail RB skeleton generated
- At runtime:
  - A large forest can have more or less of its trees become physically active at any one time, in response to interactions
    - Tree actor FIFO for fast activation
  - Leaf emitters can emit more or less leaves

More modules…

- Actively seeking great ideas
  - and partners to work with…
- What would you like to see?