Development of Games

Lecture 1

Introduction

Outline

- About course
- What is computer game
- Game loop
- Structure of Game
- What is engine
- Topics around Games
- Career connecting with Games
- Kinds of Games

Purpose of course

- The objective of the course is to introduce the students to a principles, methods and algorithms using in development of games, in particular, in special parts of graphics and AI in games.

What do you need to know

- Be able to write programs in C++ or Delphi or Java
- Basic AI

Course Evaluation

- Midterm exam: 1 exam 20%
  - Obligatory condition for attendance and passing:
    - Attendance of lectures (no less than 70%)
    - Evidence of successful work under project – if project is game then at least concept of project and functional specification
- Final Exam: 1 exam 40%
  - Obligatory condition for attendance and passing:
    - Attendance of lectures (no less than 70%)
    - Presentation on completed project
- Term Project: 1 project 40%

- Total 100%

Schedule

- Lectures
  - Monday, 9-00, room 103
- Consultations on lectures and project
  - Monday, 14-00, room B08
  - Or short questions by email

avg@oslab.khu.ac.kr

Information about ftp-server will be later

More details in syllabus

Possible themes of projects

- Implementation of determined fragment of any game using animation
- Development of technical project of game
- Development of game by team of students with different roles
- Development of game without animation and strong graphic
- Development of simulation of face with emotions
- Development of game for mobile phone
- Development of game in Internet

Terms

- Game
  - Goal
  - Devices
  - Gameplay
  - Usually opponent
  - Criterion of won
- Video Game
- **Computer Game**
- Mobile Game
- Digital Game

Different views on computer game?

- Program product
- Custom
- Art
- Source of training
- Source of knowledge
- Entertainment

What is a Game? (1 of 3)

- Movie? (why not?)
  - → no *interaction*, outcome fixed)
- Toy? (why not?)
  - → no *goal*, but still fun)
- Puzzle? (goal + interaction … why not?)
  - → strategy and outcome is the *same* each time
- Definition:
- “A computer game is a software program in which one or more players make decisions through the control of game objects and resources, in pursuit of a goal.”

What is a Game (2 of 3)

- A Computer Game is a Software Program
  - Not a board game or sports
  - Lose: 1) *physical pieces*, 2) *social interaction*
  - Gain: 1) *real-time*, 2) *more immerse*, 3) *more complexity*
  - Ex: chess vs. soccer vs. warcraft
- A Computer Game involves Players
  - The game is not for *you* but for *them.*
  - Ex: complicated flight simulator but audience is beginner

What is a Game (3 of 3)

- Playing a Game is About Making Decisions
  - Ex: what weapon to use, what resource to build
  - Can be frustrating if decision does not matter
- Playing a Game is About Control
  - Player wants to impact outcome
  - Uncontrolled sequences can still happen, but be sparing and made logical
  - Ex: *Riven* uses train system between worlds
- A Game Needs a Goal
  - Ex: Defeat Ganandorf in Zelda
  - Ex. Survive in Age of Empire
  - Long games may have sub-goals

What a Game is Not (1 of 2)

- *A bunch of cool features*
  - Necessary, but not sufficient
  - May even detract, if not careful, by concentrating on features not game
- *A lot of fancy graphics*
  - Games need graphics just as hit movie needs special effect … but neither will save weak idea
  - Again, may detract
  - Game must work without fancy graphics
  - Suggestion: should be fun with simple objects
- "When a designer is asked how his game is going to make a difference, I hope he … talks about gameplay, fun and creativity – as opposed to an answer that simply focuses on how good it looks" – Sid Meier *(Civilizations, Railroad Tycoon, Pirates)*

What a Game is Not (2 of 2)

- *A series of puzzles*
  - All games have them
  - But not gameplay in themselves
  - Puzzles are specific, game systems spawn more generic problems
- *An intriguing story*
  - Good story encourages immersion
  - But will mean nothing without gameplay
  - Example: Baldur's Gate, linear story. Going wrong way gets you killed. But not interactive. Interaction in world all leads to same end.

Games are Not Everything

- Most important … *is it fun*?
- Computers are good at interactivity
  - Allow for interactive fun
  - *Interactive Media* and Game Development ☺
- Examples:
  - *SimCity*
  - *Grim Fandango*, good visuals, story, etc. But need to do puzzles to proceed. Could have skipped to just watch story. Would still have been *fun* without game.

Overview of Gameplay

- *Game theory* – branch of economics in which systems governed by rules are mathematically analyzed to determine payoffs of various end points.
- *Gameplay* – collective strategies to reach end points
- Note, gameplay is not everything
  - Choice of car in GTA is not always about payoff, but about what is *fun*
  - Software doesn't have to have gameplay to be entertaining … consider SimCity
- No one expects gameplay in movies or plays
  - "Hey, where is the gameplay in Hamlet?"
  - Rule 1: It should be fun (entertainment)
  - Rule 2: It should be interactive (make use of computer, else perhaps use film)
  - Rule 3: It can have gameplay (but that is choice)

Gameplay Example (1 of 2)

- Adventure game: knight, dwarf, priest, thief
- During combat, knight and dwarf in front, thief fires arrows
- Priest casts spells (all cost the same)
  - E-bolts (do damage equal to sword)
  - Band-aids (heal equal to sword)
- Which to cast?
  - Ask: against single opponent (they are equal)
  - Ask: against opponent with 6 arms (bolts)
  - Ask: against many opponents with weak attacks (band-aids)
  - → Can always decide which is better
    - Not so interesting

Gameplay Example (2 of 2)

- Now, suppose
  - Band-aids still affect single target but e-bolts are area affect in radius
  - E-bolts do less damage, but armor doesn't make a difference
- Now, which to cast?
  - Answer isn't as easy. Interesting choices. Good gameplay.

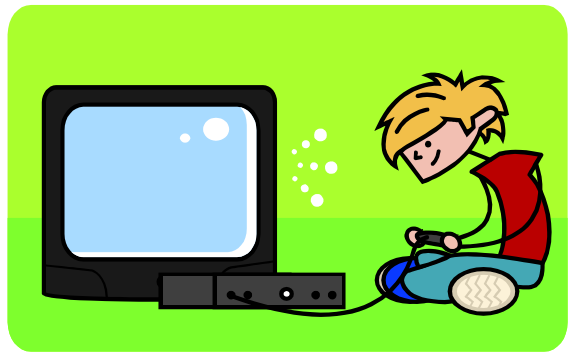
"A game is a series of interesting choices."
- Sid Meier (pirates, civilization…)

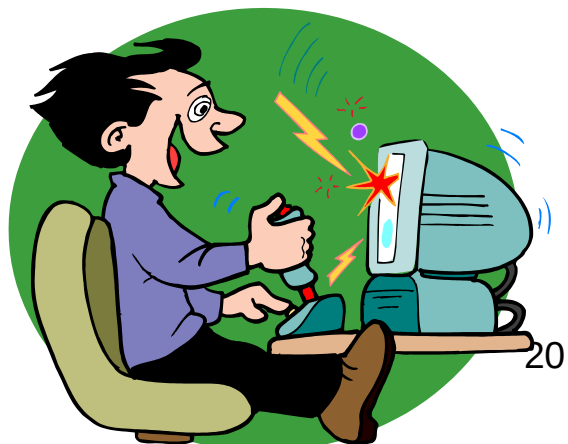
Game loop

- Starting the Game
- Player Input
- Updating Game Internals
- Main playing process
  - Displaying of screen
    - One time with updating during playing
    - Many times for different processes
  - Interaction with user
- Ending the Game
- Conclusion

Typical Game Sections

- Game startup
  - Initialize variables
  - Set up data structures
  - Allocate memory
  - Load graphics and sound files
- Game enters main loop or exits to OS
- User is prompted for input
- User input retrieve



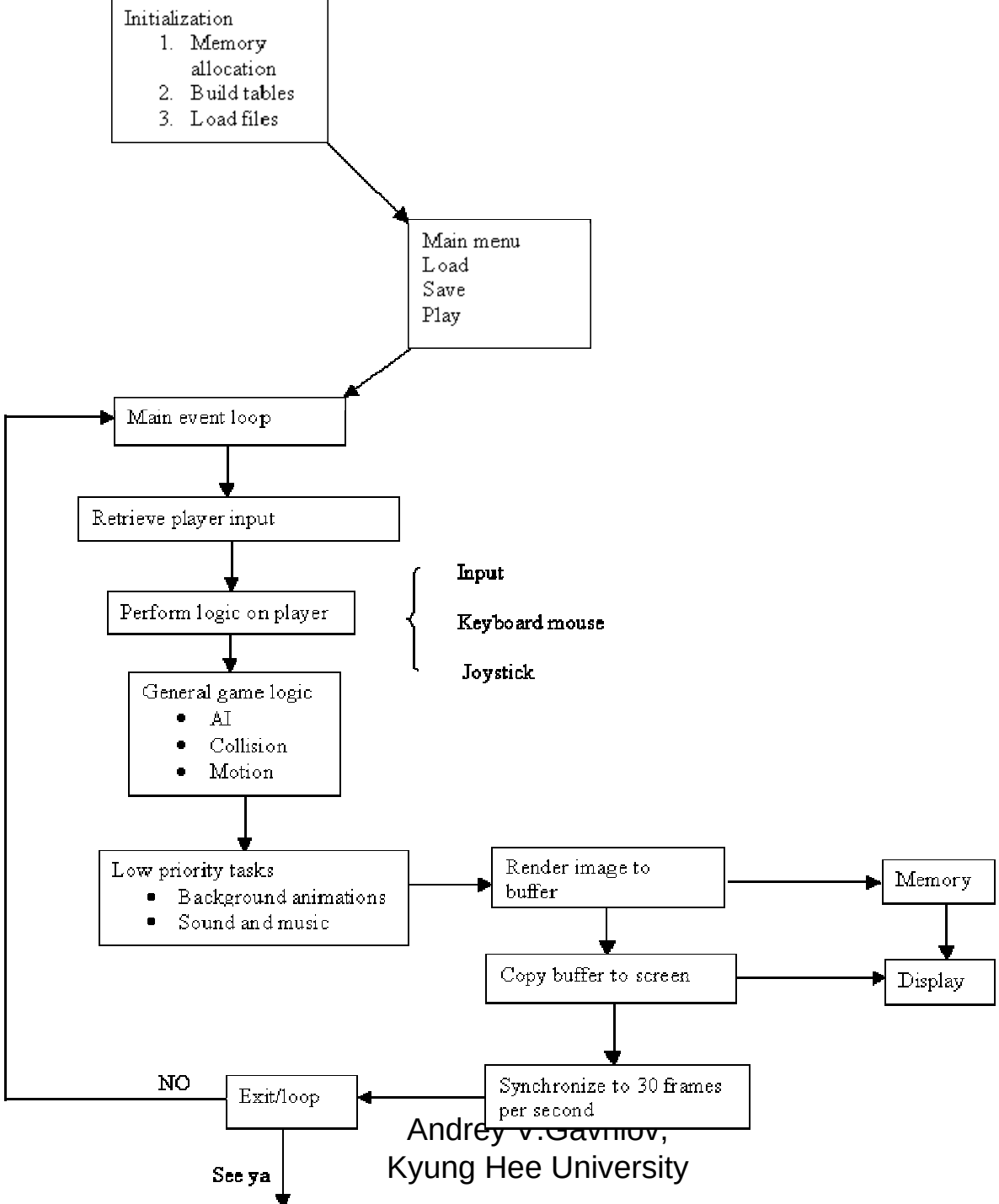


Game Sections - 2

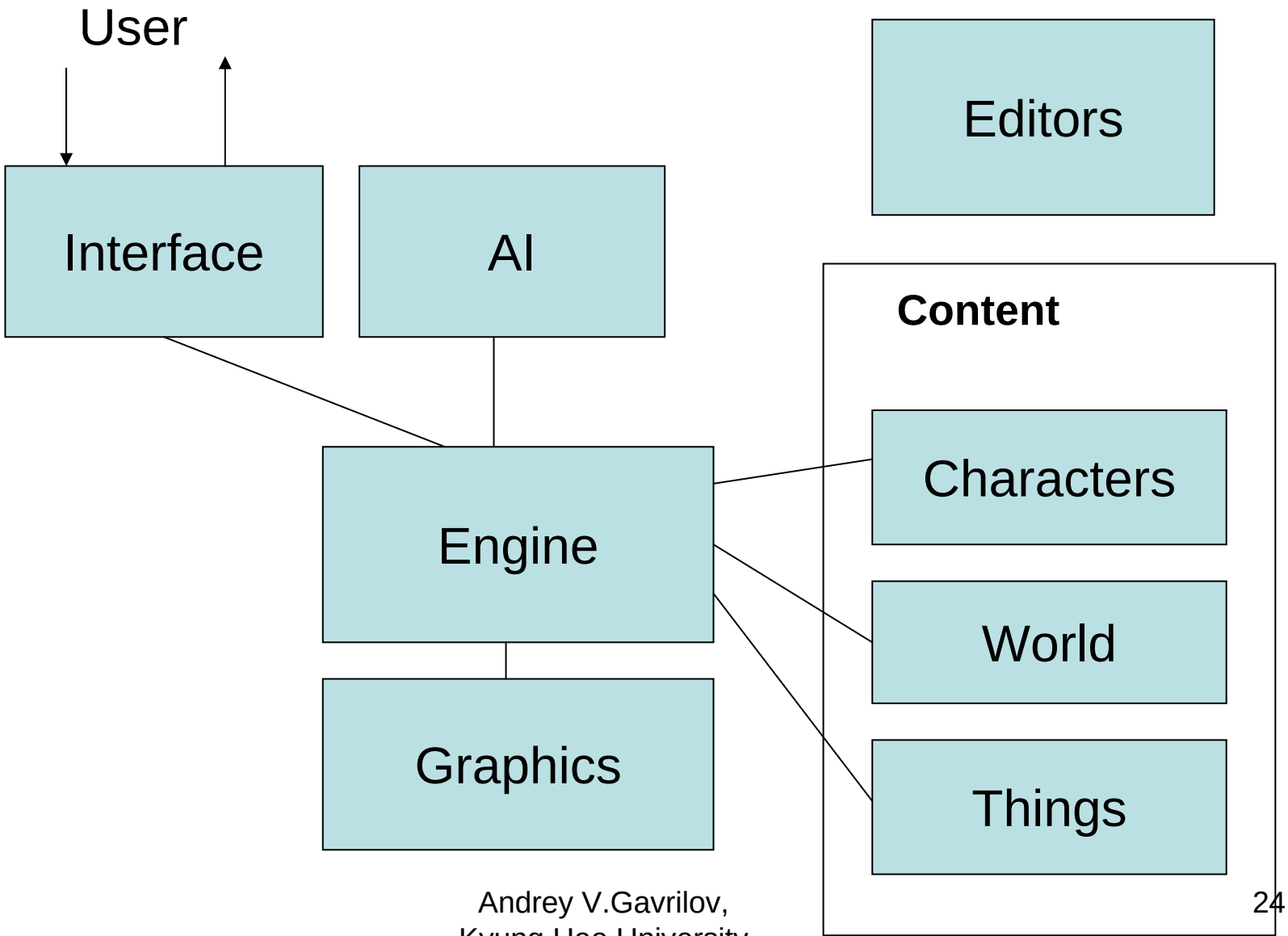
- Game state updated based on user's last input
- Based on last player action AI is applied, collisions processed, objects move
- Once player logic processing is complete, background animation performed, music, sound effects,and housekeeping performed

Game Sections - 3

- Current animation frame is rendered (drawn to virtual buffer)
- Program displays frame by copying buffer to screen
- Frame display rate locked to 30 fps
- Exit section (game over)
  - Release resources
  - Restore system settings
  - Exit to OS


Initialization
1. Memory allocation
2. Build tables
3. Load files
Main menu
Load
Save
Play
Main event loop
Retrieve player input
Perform logic on player
Input
Keyboard mouse
Joystick
General game logic
AI
Collision
Motion
Low priority tasks
Background animations
Sound and music
Render image to buffer
Memory
Copy buffer to screen
Display
Synchronize to 30 frames per second
NO
Exit/loop
See ya

Typical structure of computer game



The Parts

- It’s often hard to break up a game into distinct parts, because there is usually too much overlap to separate them. But, here are four broad components:
- Game Engine
- Rules and Mechanics
- User Interface
- Content and Challenges

Game Engines

Sometimes when a developer or player uses the term “engine” they really mean “graphics engine”. But a game engine encompasses much more. Game engines:

- Power the graphics and sound
- Power the AI
- Power the physics and interactions in the game
- Describe the nature of the game space
- Define the parameters of game objects
- Define the space of possibilities in the game world

Game Engines: Graphics

- Includes the low level computational instructions for how things are drawn on the screen.
- Contains routines for manipulating images.
- Defines the graphical capabilities of the game.

Game Engines: Physics

A (idealized) physics engine defines what physical attributes objects and the world itself can have, but not the precise values or effects of those attributes.

A physics engine may specify that:

- There is a gravity force.
- Objects have friction constants.
- The ways in which water can deform.
- The computation routines by which objects interact.

A physics engine does not (necessarily) specify that:

- Gravity is G, or g, or even inverse square.
- The specific friction constants of objects.
- The specific result of dropping a ball into water.
- The specific routines called by particular interactions.

General Game Engines

A game engine specifies the *space of possibilities* for a game, but not the *specific parameters* of elements of that game.

Some components of the *Super Mario Bros.* “engine”:

- Levels are fixed height scrolling maps.
- Levels are populated by blocks and enemies.
- Mario (and Luigi) can be small, big, or fiery.
- Blocks are affected by being bumped from below.
- Enemies are affected by being stomped, bumped from below, or hit by enemies or projectiles.
- Enemies have different movement/AI schemes.
- Enemies can spawn projectiles or other enemies.

Characteristics of an Engine

- Is broad, adaptable, and extensible.
- Firmly encodes all non-mutable design decisions.
- Allows parameters for all mutable design decisions.
- Should outline the gameplay and challenge possibilities.
- Determines the overall game architecture.
- Is coded so that new design decisions leave it unchanged.

Rules and Mechanics

Specific decisions about game parameters, obstacles, and abilities determine the rules and mechanics of the game. This includes things like:

- Player abilities
- Enemy stats
- Enemy behaviour
- Spell details
- Jumping height
- Gravity strength
- Point values
- Interplay between game objects

While the overall challenges aren’t determined here, the heart of gameplay is in mechanics.

Rules: Super Mario Bros.

Some rules from Super Mario Bros:

- One kind of block is the “question” block. A question block, when bumped, yields either a coin, 10 coins, a power-up, or a star.
- If Mario triggers a power-up when small, it is a mushroom. When big or fiery, it is a fire flower.
- Goombas die when stomped.
- Turtles become shells when stomped or bumped.
- 100 coins yields an extra life.
- Spinys damage Mario when stomped.
- Piranha Plants aim fireballs towards Mario.

Rules and Mechanics (cont'd)

- If we continue the D&D analogy, then engine + mechanics = core rulebooks.
- Engine and mechanics still doesn't make a whole game.
- AI is part of the mechanics.
- If you have the engine and the mechanics, you should be able to make a level editor or game toolset.
- Takes the space of possibilities, and makes *decisions* for all parameters

Interfaces

- The engine and mechanics tells us what the player and other objects in the game can do.
- The interface tells us how the player does things, and how she knows what's happening in the game.
- Interfaces thus have two parts:
  - Player-to-Computer
  - Computer-to-Player
- The interface is the center of the user experience.
- In the D&D analogy, the interface is character sheets, maps, dice, pencils, and the voices of the players and the Dungeon Master.

Interface Tips

- On the PC, your inputs are mouse and keyboard. This affects not just the interface, but the design itself.
- Carefully consider the depth and width of your interface.
- Details are best processed at the center of vision.
- Peripheral vision mostly detects motion.
- Enhance your interface with sounds.
- Familiarity is better than innovation in interface.
- Strive for an “invisible” interface, but metaphorically.

Content and Challenges

Content is everything we haven't discussed yet. We can dived it into two types: gameplay and non-gameplay.

Non-gameplay content includes:

- Graphics
- Sound Effects
- Background Music
- Cut Scenes
- Story
- Flavor Text
- Dialogue

To be fair, many of these have deep gameplay implications, and should be considered at other stages.

Gameplay Content

When developers speak of content, they often mean gameplay content:

- Goals and victory conditions
- Missions and quests
- Level design
  - Pacing and Atmosphere
  - Difficulty curves and Balance
  - Reward structure
  - Atmosphere and Harmony
- In the D&D analogy, “modules” (adventures), and the DM’s imagination are the content.

Why the division?

- These four components - *Engine, Mechanics, Interface*, and *Content* – are **not** created sequentially, or separately. But thinking about them will keep your organized.

- Understanding the *Engine* tells you what decisions must be made early, and what should be hard-coded.

- Understanding the *Mechanics* tells you what design decisions may need changing and should be mutable.

- Understanding the *Interface* allows you to shape the user experience to fit your game vision.

- Understanding the *Content* ensures that your create the right world and gameplay for your game.

Areas related with Games

Three Major Areas

- Humanistic Study (Art, Entertainment, Source of training, Source of knowledge)
- Game Technology (Program product, Art)
- Game Business (Custom, Program product)

Topics in Study of Games

- Humanistic Study:
  - Critical Game Studies
    - Criticism, Analysis and History of electronic and non-electronic games
  - Games and Society
    - Understanding how games reflect and construct individuals and groups

Technical Study of Games

- Game Design
- Game Programming
- Visual Design
- Audio Design
- Interactive Storytelling

Process & Management

Game Production

- Practical challenges of managing the development of games

Game Business

- Economic, legal and policy aspects of games

Career in Game Industry

- Scholarly/Academic
  - Game Studies Scholar/Educator
  - Game Technology Educator
  - Game Journalist
- Applied
  - Game Artist
  - Game Programmer
  - Game Designer
  - Game Producer

Requirements to Game Studies Scholar and Educator

- Trained in History, Analysis, Criticism
- Experienced Gamer
  - Knows Genres, Designs
  - Understands Technology
- Familiar with Industry
  - Understands Dev. Process
  - Knows gist of Business & Legal

Requirements to Game Technology Educator

- Trained in Design and Development
- Experienced Programmer
  - Knows Mechanics, Dynamics
  - Hardware Strengths & Limitations
- Emphasizes Good Process
  - Software Dev. Best & Worst Practices
  - Group Work, Creativity Management

Requirements to Game Journalist

- Trained in Design, Analysis, Criticism
- Expert Communicator
- Investigator of Game Culture
  - Non-Digital, PC, Console, Online
  - Visual Aesthetics, Narrative Theory
  - Social Issues (Gender, Violence)
  - Technical trends, research, novel implementations

Requirements to Game Programmer/Artist

- Trained in Design, Analysis, Tech
- Experienced Procedural Thinker
- Specialization Expert
  - Graphics Programming
  - Audio Design & Implementation
  - Concept Art, 3D design & Rendering
  - Level Design and Game Mechanics
  - Character Design, Behavior, Artificial Intelligence

Requirements to Game Designer

- Trained in Design, Analysis, Tech
- Experienced Procedural Thinker
- Expert Communicator
  - Narrative and Experience goals
  - Visual & Audio Aesthetics
  - Practical Nuts & Bolts
    - Example: Thief

Requirements to Game Producer

- Trained in Biz & Management
- Experienced Procedural Thinker
- Expert Communicator
  - Team structure and goals
  - Time, Budget and Design constraints
  - Markets, Promotion, Publication
  - Legal issues

Bit of history of games

- 1962: *Spacewar* for the DEC PDP-1
- 1972: *Pong*, Magnivox Odessy
- 1985: Nintendo
- 1990: 3D (First Person Shooters)
- 2000: Games = $$$
  - Over 30 million consoles in homes
  - Over 20 million PC gamers

New game categories over time

- 1981: Dungeons and Dragons
- 1982: Flight Sims
- 1986: Chess
- 1988: Sports simulations
- 1989: God games
- 1993: Shooters
- 1994: Interactive movies
- 1997: MMO’s
- 1999: Dance games
- 2000: Dollhouse games
- 2001: Living city games
- 2002: “Casual” games
- 2005: Pet games
- 2005: Music games

Kinds of games (tasks)

- Shooter
  - Doom, Quake, Unreal Tournament, Mortal Combat,
- Strategy
  - Civilization, Simcity, Tycoon, Warcraft, Starcraft, Capitalism, Europe, Master of Orion, Sudden Strike, Empire of Earth, Airport
- Quest

- Intelligent game
  - Chess, Go, Manjong, Playing cards, Games with words, Puzzles
- Simulator
  - F-18, F-117, Battle for Britain, Airplane, Billiard,
- Simulator without participation of user
  - Robots

Kinds of games (using of time)

- Turn-based
  - Civilization, Capitalism, intelligent games
- Real time
  - Sudden strike, warcraft, spacecraft, simulators

Kinds of games (using of communication)

- Autonomous games
  - For PC
  - For Play Stations
- On-line games with computer in Internet
- On-line games with other users in local network/intranet
- Mobile Games
- Mobile Games in Internet

For More Information

- Resources
  - www.igda.org
  - www.gamasutra.com
- Journalism
  - www.edgeonline.com
  - www.gamegirladvance.com