Module 11
Class Design and Implementation

Lecture
XNA Class:
Fields and Properties

Module 11 Learning Objectives

Bloom Level	Number	Name	Description	Course Learning Objectives
2: Understand	1	Fields and Properties	Describe the relationship between fields and properties	Basic OO Concepts
3: Apply	2	Console Application Fields and Properties	Design and implement the fields and properties for a console application class	Basic OO Concepts
3: Apply	3	Console Application Methods	Design and implement the constructors and methods for a console application class	Basic OO Concepts
3: Apply	4	XNA Fields and Properties	Design and implement the fields and properties for an XNA class	Basic OO Concepts,Basic XNA Concepts
3: Apply	5	XNA Methods	Design and implement the constructor and methods for an XNA class	Basic OO Concepts,Basic XNA Concepts

Last time, we finished our implementation of a class for a console application.

In this lecture, we'll start designing and implementing a class to use in an XNA game.

In-Lecture Quiz

Designing and implementing our rubber chicken class would be

- A: awesome
- B: great
- C: fantastic
- D: wonderful

- We’ll just use our rubber chicken as a ranged weapon
- Simplified state information
  - Active
  - Sprite
  - Draw/Collision rectangle
  - Velocity
  - Damage

In-Lecture Quiz

We should make our fields public when

- A: we feel like it
- B: we want to write bad code
- C: the moon is full
- D: the tide is high

RubberChicken
Class

Fields

- active : bool
- damage : int
- drawRectangle : Rectangle
- sprite : Texture2D
- velocity : Vector2

Properties

Methods

In-Lecture Quiz

Properties can have

- A: read access
- B: write access
- C: read-write access
- D: this answer is wrong

RubberChicken
Class

Fields

- active : bool
- damage : int
- drawRectangle : Rectangle
- sprite : Texture2D
- velocity : Vector2

Properties

- Active { get; set; } : bool
- CollisionRectangle { get; } : Rectangle
- Damage { get; } : int

Methods

- Simplified behavior
  - Update
    - Change position based on velocity
    - Detect click for launch
  - Draw

RubberChicken
Class

Fields

- active : bool
- damage : int
- drawRectangle : Rectangle
- sprite : Texture2D
- velocity : Vector2

Properties

- Active { get; set; } : bool
- CollisionRectangle { get; } : Rectangle
- Damage { get; } : int

Methods

- Draw(SpriteBatch spriteBatch) : void
- Update(GameTime gameTime, MouseState mouse) : void

- Recap
  - Designed rubber chicken class and implemented its state using fields and properties
- Next Time
  - We’ll implement the constructor and methods