Demo_Animation_Explosion

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;



namespace Demo_Animation_Explosion
{
    /// <summary>
    /// MonoGame to demonstrate how to implement an animation
    /// </summary>
    public class DemoExplosion : Game
    {
        private GraphicsDeviceManager _graphics;
        private SpriteBatch _spriteBatch;

        private int _gameWindowWidth;
        private int _gameWindowHeight;

        private KeyboardState _keyboardOldState;
        private KeyboardState _keyboardNewState;

        private Vector2 _applePos;
        private Vector2 _explosionPos;
        private Vector2 _screenTextPos;

        private Apple _apple;
        private Explosion _appleExplosion;
        private ScreenText _screenText;

        /// <summary>
        /// game constructor initializes the window
        /// </summary>
        public DemoExplosion()
        {
            _graphics = new GraphicsDeviceManager(this);

            // initialize game window settings
            _gameWindowWidth = 400;
            _gameWindowHeight = 400;

            _graphics.PreferredBackBufferWidth = _gameWindowWidth;
            _graphics.PreferredBackBufferHeight = _gameWindowHeight;

            // game content location
            Content.RootDirectory = "Content";
        }

        /// <summary>
        /// initialize the starting locations for each game object
        /// </summary>
        protected override void Initialize()
        {
            // set the position of all objects
            _applePos.X = 200;
            _applePos.Y = 200;

            _explosionPos.X = 200;
            _explosionPos.Y = 200;

            _screenTextPos.X = 100;
            _screenTextPos.Y = 100;

            base.Initialize();
        }

```

```
67          /// <summary>
68          /// load all of the game content into the Content object
69          /// </summary>
70          protected override void LoadContent()
71          {
72              // Create a new SpriteBatch, which can be used to draw textures.
73              _spriteBatch = new SpriteBatch(GraphicsDevice);
74
75              // create (instantiate) objects
76              _apple = new Apple(Content);
77              _appleExplosion = new Explosion(Content);
78              _screenText = new ScreenText(Content);
79
80              // make apple and text visible
81              _apple.Active = true;
82              _screenText.Active = true;
83          }
84
85          /// <summary>
86          /// unload all of the content from the Content object
87          /// game-specific content.
88          /// </summary>
89          protected override void UnloadContent()
90          {
91              Content.Unload();
92          }
93
94          /// <summary>
95          /// this method is call once for each game "click"
96          /// checking for collisions, gathering input, and playing audio.
97          /// </summary>
98          /// <param name="gameTime">Provides a snapshot of timing values.</param>
99          protected override void Update(GameTime gameTime)
100         {
101             // handle any new keyboard events
102             HandleKeyboardEvents();
103
104             // if explosion is active update frame
105             if (_appleExplosion.Active) _appleExplosion.Update(gameTime);
106
107             base.Update(gameTime);
108         }
109
```

```
110         /// <summary>
111         /// this method is call once for each game "click"
112         /// and draws all active game objects
113         /// </summary>
114         /// <param name="gameTime">provides a snapshot of timing values.</param>
115         protected override void Draw(GameTime gameTime)
116         {
117             GraphicsDevice.Clear(Color.GreenYellow);
118
119             _spriteBatch.Begin();
120
121             string message =
                "Press the 'e' to explode the apple.\n Press the Escape key to quit.";
122
123             if (_screenText.Active) _screenText.Draw(_spriteBatch, new Vector2(200
                , 200), message);
124
125             if (_apple.Active) _apple.Draw(_spriteBatch, new Vector2(200, 200));
126
127             if (_appleExplosion.Active) _appleExplosion.Draw(_spriteBatch, new
                Vector2(200, 200));
128
129             _spriteBatch.End();
130
131             base.Draw(gameTime);
132         }
133
134         private void HandleKeyboardEvents()
135         {
136             // get new state of keyboard
137             _keyboardNewState = Keyboard.GetState();  // get the newest state
138
139             // handle the keyboard input
140
141             // player chooses explosion
142             if (_keyboardOldState.IsKeyUp(Keys.E) & _keyboardNewState.IsKeyDown(
                Keys.E))
143             {
144                 _apple.Active = false;
145                 _appleExplosion.Active = true;
146             }
147
148             //player chooses to quit
149             if (_keyboardOldState.IsKeyUp(Keys.Escape) & _keyboardNewState.
                IsKeyDown(Keys.Escape))
150             {
151                 Exit();
152             }
153
154             // set the new state of the keyboard as the old state
155             _keyboardOldState = _keyboardNewState;
156         }
157     }
158 }
```