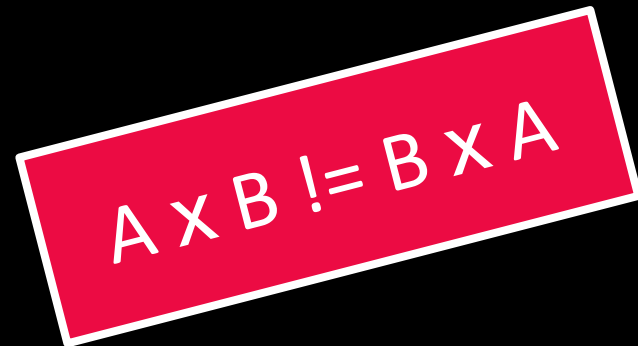




**GAMECORE**

# Линейная алгебра

- Vector2
- Vector3
- Vector4
- Matrix3x3
- Matrix
- BoundingBox
- BoundingFrustum
- BoundingSphere
- Ray


$$A \times B \neq B \times A$$

# Игровая сущность — поля

- Позиция
- Ориентация в пространстве
- Кол-во жизней
- Боезапас
- Кол-во очков
- Флаги состояний
- Размер
- ...
- Все что угодно

```
public class Entity {  
    public Vector2 Position;  
    public float Angle  
    public int Health;  
    public int Ammo;  
    public int Score;  
    public EntityState State;  
    public float Size;  
}
```

```
[Flags]  
public enum EntityState {  
    None = 0x00,  
    Dead = 0x01,  
    Invisible = 0x02;  
    // add moar!  
}
```

# Игровая сущность — методы

- Создать
- Обновить  
внутреннее  
состояние
- Нарисовать
- Нанести урон
- Уничтожить
- Касание

```
public class Entity {  
    // continue...  
    readonly Game game;  
    public Entity (Game game){ this.game = game; ...}  
    public void virtual Update(GameTime gameTime){}  
    public void virtual Draw(SpriteBatch spriteBatch){}  
    public void virtual Damage(Entity attacker, int dmg){}  
    public void virtual Kill (Entity killer){}  
    public void virtual Touch(Entity other){}  
}
```

# Игровая сущность — наследники

```
public class Monster : Entity {
    Texture2D texture;
    public Monster (Game game) : base(game) {
        texture = game.Content.Load<Texture2D>("mySuperMonsterImage");
    }
    public void override Update(GameTime gameTime) {
        // ...
    }
    public void override Draw(SpriteBatch spriteBatch) {
        spriteBatch.Draw( texture, position, ... );
    }
    public void override Damage(Entity attacker, int dmg){}
    public void override Kill (Entity killer){}
    public void override Touch(Entity other, Vector2 p){}
}
```

# Игровой мир

- Глобальное состояние
- Вместители все существей
- Порождение существей
- Обновление мира и существей
- Рисование мира и существей

```
public class World {  
    List<Entity> entities = new ...  
    List<Entity> toKill  
    public TimeSpan worldTime;  
  
    public void Update(GameTime gameTime){}  
    public void Draw(GameTime gt, SpriteBatch sb){}  
    public Entity Spawn( string className, ... ){}  
    public void Kill(Entity ent){}  
}
```

# Игровой мир — жизненный цикл сущностей

```
public class World {  
  
    public void Update(GameTime gameTime){  
        foreach ( var ent in entities) {  
            ent.Update(gameTime);  
        }  
        foreach ( var ent in entities) {  
            ent.Update(gameTime);  
        }  
        entities.RemoveAll( e => toKill.Contains(e); );  
    }  
  
    public void Draw(GameTime gt, SpriteBatch sb) {  
        foreach ( var ent in entities) {  
            ent.Draw(sb);  
        }  
    }  
}
```

# Игровой мир — жизненный цикл сущностей

```
public class World {  
  
    public Entity Spawn( string className, ... ){  
        return (Entity)Activator.CreateInstance(...);  
    }  
  
    public void Kill(Entity ent) {  
        toKill.Add(ent);  
    }  
  
}
```

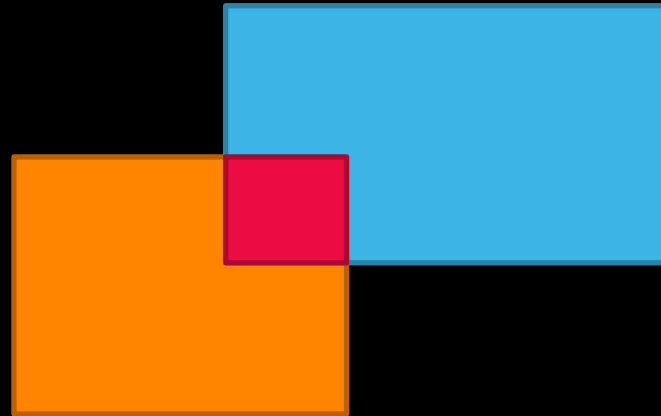


# Игровой мир — взаимодействие сущностей

Псеводкод:

```
foreach ( var ent in entities) {  
    foreach ( var other in entities) {  
        if ( ent intersects other ) {  
            ent.Touch( other );  
        }  
    }  
}
```

- Axis-aligned bounding box
  - Google It ;)
  - class BoundingBox



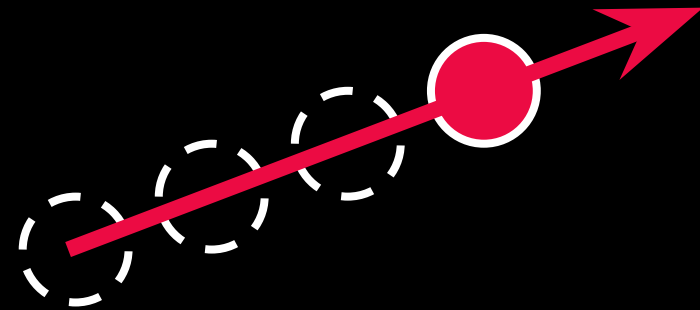
# Интегрирование игрового мира

- Time step:

- $dt$
- gameTime

- Fixed time step
- Variable time step

$$X_{n+1} = X_n + V * dt$$



**Участвуем в конкурсе! ;)**







**Вопросы? В бой!**

Алексей Безгодов