Játékprogramok készítése

5. óra - entitások interakciója, öröklődés, publish-subscribe minta

Órai munka - entitások interakciója, öröklődés, publish-subscribe minta

- 4. óra folytatása, fel van töltve!
- Lehessen lőni, a lövedék semmisüljön meg a fallal való ütközéskor!
- Ne legyen felesleges getLeft(), stb metódusok.
- Az entitások tudják magukról, hogy mit kell csinálniuk a frameben, stb.

Kód duplikáció - a probléma

```
var Fal = fw.entity({
    draw: function (ctx) {
        ctx.drawImage(fal, this.x, this.y);
    },
    getLeft: function () {
        return this.x;
    },
    getTop: function () {
        return this.y;
    },
    getWidth: function () {
        return fal.width;
    },
    getHeight: function () {
        return fal.height;
    }
});
```

```
var Lovedek = fw.entity({
    draw: function (ctx) {
        ctx.drawImage(lovedek, this.x, this.y);
    },
    getLeft: function () {
        return this.x;
    },
    getTop: function () {
        return this.y;
    },
    getWidth: function () {
        return lovedek.width;
    },
    getHeight: function () {
        return lovedek.height;
    }
});
```

Kód duplikáció - a megoldás: öröklődés

```
//fw.EntityWithSprite: this.image kirajzolása, getLeft(), stb.

var Fal = fw.entity(fw.EntityWithSprite, {
    init: function () {
        this.image = fal;
    }
});

var Lovedek = fw.entity(fw.EntityWithSprite, {
    init: function () {
        this.image = lovedek;
    }
});
```

EntityWithSprite testreszabása

```
var Hos = fw.entity(fw.EntityWithSprite, {
    init: function () {
        this.irany = 'jobb';
        this.image = hos;
        this.imageColumns = 2; //a spritesheet 2 oszlopból áll
        this.imageScaleX = 0.5; //a hős szélességének felezése
        this.imageScaleY = 0.5; //a hős magasságának felezése
    }
});
```

Minden logika egy helyen - a probléma

```
//játékciklus:
for (var i = 0; i < entitasok.length; i++) {
    var entitas = entitasok[i];
    if (entitas instanceof Lovedek) {
        //lövedék mozgatása, stb.
    } else if (entitas instanceof Hos) {
        //billentyű kezelés, stb.
    }
}
```

Minden logika egy helyen - a probléma

```
//játékciklus:
for (var i = 0; i < entitasok.length; i++) {
    var entitas = entitasok[i];
    if (entitas instanceof Lovedek) {
        //lövedék mozgatása, stb.
    } else if (entitas instanceof Hos) {
        //billentyű kezelés, stb.
    } else if (entitas instanceof Fal) {
        //...
    } else if (entitas instanceof Ellenseg) {
        //...
    } else if (entitas instanceof Poti) {
        //...
```

Logika mozgatása az entitásba

```
var Lovedek = fw.entity(fw.EntityWithSprite, {
    //...

    onFrame:function(){
        //mozgatás, stb.
    }
});

var Hos = fw.entity(fw.EntityWithSprite, {
     //...

    onKeyDown_37 : function (ctx) {
     //mozgatás balra...
    },

    onKeyDown_38 : function (ctx) {
     //mozgatás felfelé...
    }
});
```

Logika mozgatása az entitásba - probléma

```
//játékciklus:
for (var i = 0; i < entitasok.length; i++) {
    var entitas = entitasok[i];
    if(entitas.onFrame){
        entitas.onFrame();
    }
    if(entitas.onDraw){
        entitas.onDraw(ctx);
    }
    for(var i in fw.pressedKeys) {
        if(fw.pressedKeys[i]) {
            var keydownHandler = 'onKeyDown_' + i;
            if (entitas[keydownHandler]) {
                entitas[keydownHandler]();
            }
        }
    }
}
```

- Sok entitást nem is érdekel az esemény!
- O(n*e):
  entitások * események
- események száma nőni fog!
  - onMouseDown
  - onBeforeFrame
  - onAfterFrame
  - onHeroDie
  - …

Logika mozgatása az entitásba - probléma

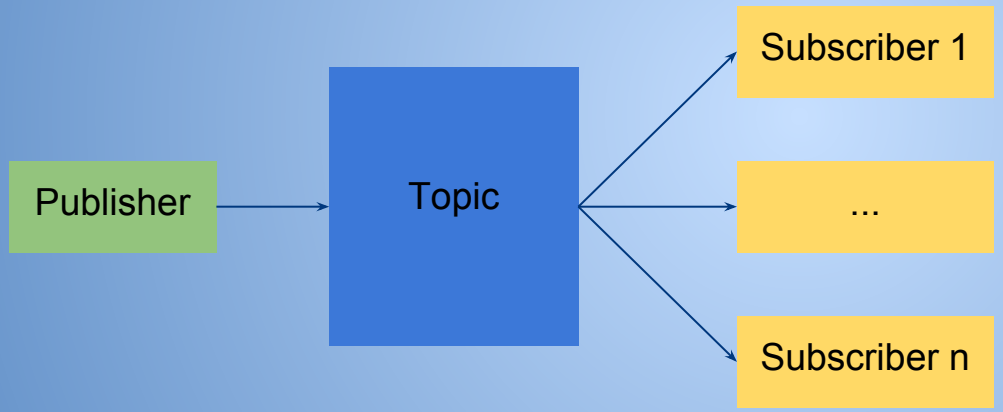
```
var Enemy = fw.entity(fw.EntityWithSprite, {
    //...
});

var Monster = fw.entity(Enemy, {
    //...
});

var Giant = fw.entity(Monster, {
    //...
});

var GiantWithAxe = fw.entity(Giant, {
    onFrame: function () {
        //balta lóbálása, stb.

        // kell ide az ős frame meghívása???
    }
});
```

- A 4. leszármazottban készítünk egy onFrame eseménykezelőt
- Vajon volt már a Giant-nak is, amit meg kéne hívni?
- És a Monster-nek?
- És az Enemy-nek?
- És ha utólag rakok egyet a Monsterbe?

Logika mozgatása az entitásba - problémák összefoglalva

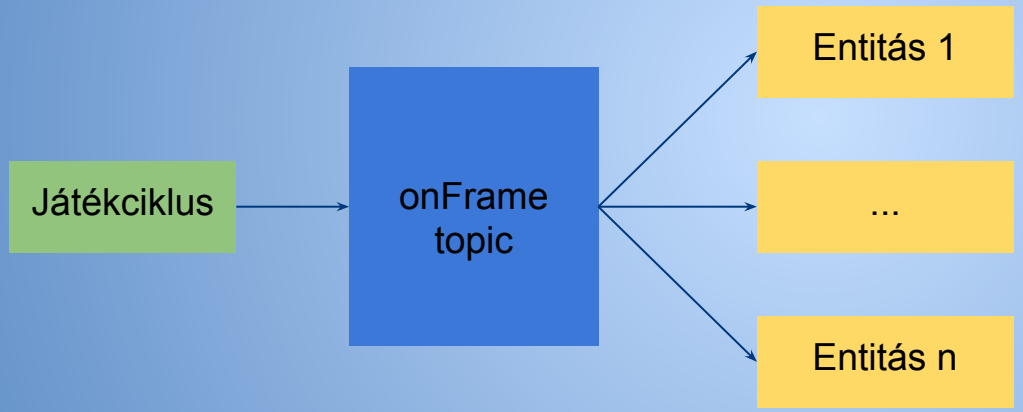
- A játékciklus nem tudja, mely entitásokat “érdekel” egy **globális** esemény!
- Az entitás sem tudja, hogy mely ősosztályokat “érdekli” a **lokális** esemény!

Megoldás: publish-subscribe minta



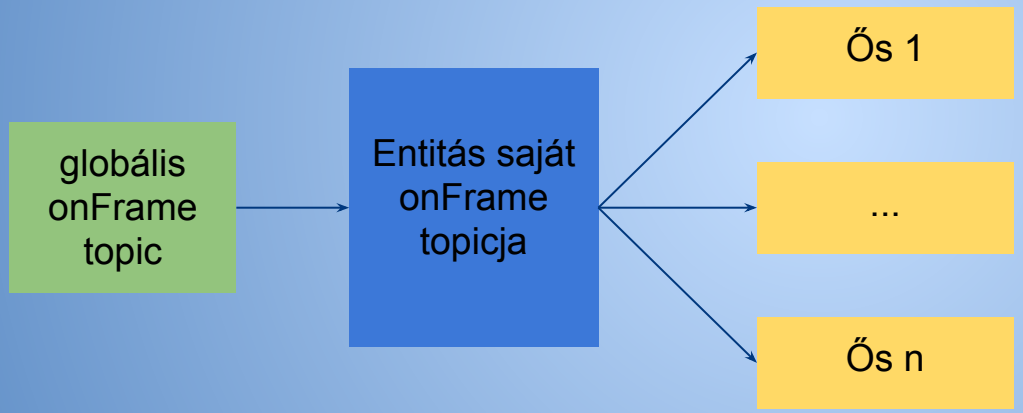
- Subscriber: feliratkozik a topic-ra
- Publisher: közzétesz egy üzenetet
- Topic: az üzenetet továbbítja a feliratkozottaknak

Megoldás: publish-subscribe minta (globális események)



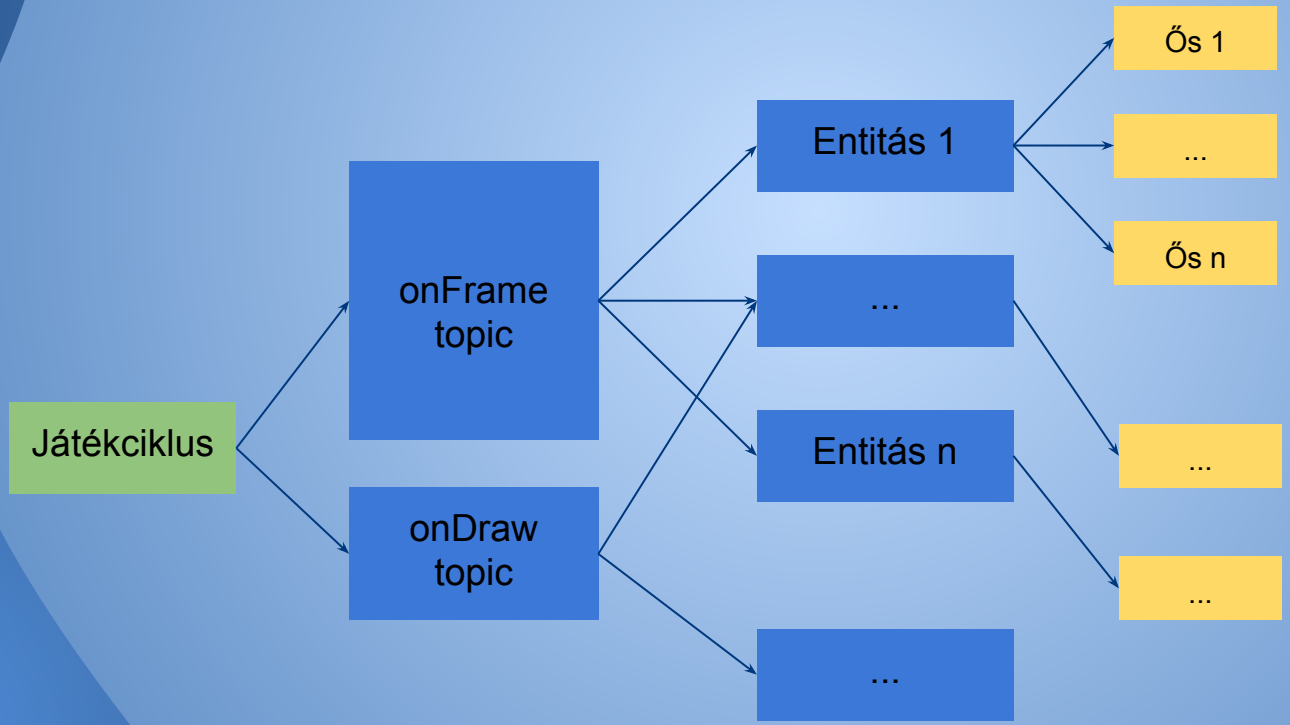
- Entitás: feliratkozik az onFrame eseményre
- Játékciklus: kiváltja az onFrame eseményt
- onFrame topic: továbbítja az entitásoknak az eseményt

Megoldás: publish-subscribe minta (lokális események)



- Ős: feliratkozik a lokális onFrame eseményre
- globális onFrame topic: kiváltja az onFrame eseményt
- lokális onFrame topic: továbbítja az ősöknek az eseményt.

Megoldás: publish-subscribe minta (teljes ábra)



Publish-subscribe megvalósítása

- Vezessünk be egy scene objektumot, mostantól ez tárolja az entitásokat!
- Ha hozzáadunk egy entitást, az magától iratkozzon fel az őt érdeklő eseményekre!
- A scene objektumnak egy eseményt jelezve, azt továbbítja az entitásoknak
- Az entitás ezt továbbítja az ősök eseménykezelőinek (ha van)

Entitások létrehozása, törlése: a Scene

```
var scene = new fw.Scene();

scene.add(new Hos(300, 300));

scene.remove(lovedek);
```

Új createIndex

```
var scene = new fw.Scene();

var index = fw.createIndex(scene);
```

Feliratkozás eseményekre

```
var Lovedek = fw.entity(fw.
EntityWithSprite, {
    //...

    $frame: function () {
        this.x += 10;
    },
    $draw: function (ctx) {
        //ctx.drawImage(...)
    }
});
```

A keretrendszer a $ kezdetű metódusokból eseménykezelőket készít.

Események kiváltása

```
function draw() {
    ctx.clearRect(0, 0, canvas.width, canvas.height);
    scene.fire('draw', ctx);
}

for(var i in fw.pressedKeys){
    scene.fire('keyDown_' + i);
}
```

Közkívánatra: hasCollision

```
//ütközik a lovedek entitással bármilyen Fal?
if (index.hasCollision(lovedek, Fal)) {
  scene.remove(lovedek);
}

//ha jobbra mozgatnánk a hős entitást 5 pixellel,
//kerülne metszésbe bármilyen Fal entitással?
if (!index.hasCollision(hos, Fal, 5, 0)) {
    this.x += 5;
}
```

Órai munka - részletesen

- Tüntessük el az ismétlődő getLeft()-et, stb.
- Helyezzük az entitásokat scene-be!
- Váltsunk ki globális eseményeket!
- Tüntessük el a globális hosEntity-t
- Gombnyomásra jöjjön létre lövedék
- A lövedék mozogjon!
- A lövedék semmisüljön meg ha falnak ütközik!