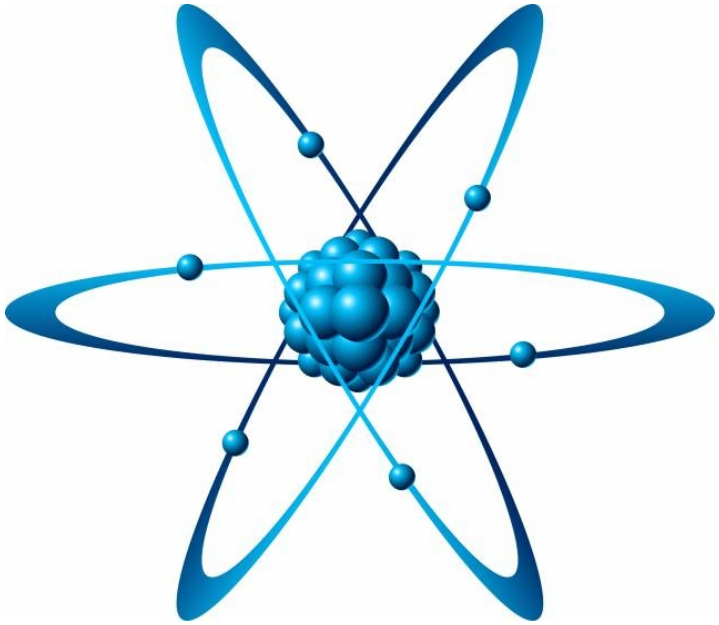
Анимируем объекты с использованием физики в JavaScript



Иванова Елена
@liveldi90
Фронтенд разработчик
Astroshock

25 июня 2015

СОДЕРЖАНИЕ

- Анимация
- Конструкторы
- Движение
- Сила трения
- Ускорение и прыжки
- Пружины и колебания

АНИМАЦИЯ

```
setTimeout(func, 1000/fps);
setInterval(func, 1000/fps);
requestAnimationFrame(func);
```

```
setTimeout(func, 1000/fps);
setInterval(func, 1000/fps);
requestAnimationFrame(func);
```

```
var t0 = 0;
var xCoord = 0;

function onTimer() {
    var t1 = new Date().getTime();
    dt = t1 — t0;
    t0 = t1;
    xCoord  += 0.001 * dt;
}
```

```
var t0 = 0;
var xCoord = 0;

function onTimer() {
    var t1 = new Date().getTime();
    dt = t1 — t0;
    t0 = t1;
    xCoord  += 0.001 * dt;
}
```

КОНСТРУКТОРЫ

ОБЪЕКТ — параметры

```
var Obj = function (ops) {
    this.$el
    this.mass

    this.x
    this.y

    this.vx
    this.vy
    … };
```

ОБЪЕКТ — методы

Obj.pos

Obj.velo

Obj.changeStyle

ВЕКТОР — параметры

```
var Vector = function (x, y) {
    this.x = x;
    this.y = y;
};
```

ВЕКТОР — методы

```
Vector.length()
Vector.add(vec) . subtract(vec)
Vector.multiply(k) . divide(k)
Vector.addScaled = function (vec, k) {
        return new Vector(
                this.x + k * vec.x,
                this.y + k * vec.y);
};
```

ДВИЖЕНИЕ

$$s = v \cdot t$$

```
obj.pos = new Vector(10, 0);
obj.velo = new Vector(60, 0);
```

```
obj.pos = new Vector(10, 0);
obj.velo = new Vector(60, 0);

function move() {
    obj.pos = obj.pos.addScaled(obj.velo, dt);
}
```

$$s = v \cdot t$$

СИЛА ТРЕНИЯ

КИНЕТИЧЕСКАЯ ЭНЕРГИЯ

$$E_k = \frac{1}{2} \cdot m \cdot v^2$$

$$v = \sqrt{\frac{2 \cdot E_k}{m}}$$

МОЩНОСТЬ

$$P = F \cdot v$$

МОЩНОСТЬ

$$P = F \cdot v$$

СИЛА ТРЕНИЯ

$$F = -k \cdot v$$

МОЩНОСТЬ

$$P = F \cdot v$$

СИЛА ТРЕНИЯ

$$F = -k \cdot v$$

$$P = -k \cdot v^2$$

НАЧАЛЬНЫЕ РАСЧЕТЫ

```
var displ = obj.pos.subtract(obj.pos0);
obj.velo = displ.divide((t1 — t0) * 0.001);
```

$$v = \frac{s}{t}$$

```
ke = new Vector(
        0.5 * obj.mass * obj.vx * obj.vx,
        0.5 * obj.mass * obj.vy * obj.vy);
```

$$E_k = \frac{1}{2} \cdot m \cdot v^2$$

ОБНОВЛЕНИЕ МОЩНОСТИ

```
function applyPower() {
    var powerLoss = new Vector(
            powerLossFactor * obj.vx * obj.vx * dt,
            powerLossFactor * obj.vy * obj.vy * dt);
    ke = ke.subtract(powerLoss);
}
```

$$P = -k \cdot v^2$$

ОБНОВЛЕНИЕ СКОРОСТИ

```
function updateVelo() {
        obj.velo = new Vector(
                Math.sqrt(2 * ke.x / obj.mass),
                Math.sqrt(2 * ke.y / obj.mass)
        );
}
```

$$v = \sqrt{\frac{2 \cdot E_k}{m}}$$

ОБНОВЛЕНИЕ ПОЗИЦИИ

```
function moveObject() {
        obj.pos = obj.pos.addScaled(obj.velo, dt);
        obj.changeStyles();
}
```

$$s = v \cdot t$$

ДОБАВЛЕНИЕ МОЩНОСТИ

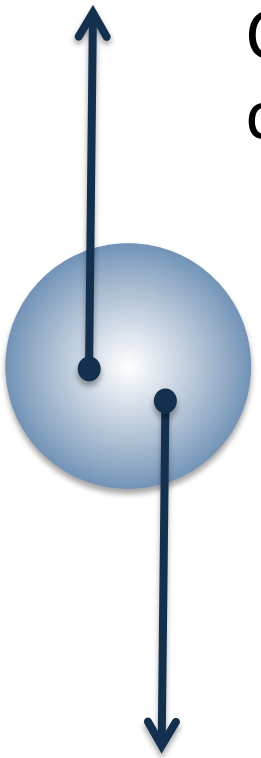
```
function applyPower() {
    if (applyThrust) {
        ke = ke.add(new Vector(
                        powerApplied * dt,
                        powerApplied * dt)
                    );
    }
    …
}
```

УСКОРЕНИЕ И ПРЫЖКИ

УСКОРЕНИЕ

$$D = -k \cdot v$$

Сила сопротивления (D)



Гравитация (W)

$$F = m \cdot g - k \cdot v$$

$$W = m \cdot g$$

УСКОРЕНИЕ

$$a = \frac{F}{m}$$

$$v = a \cdot t$$

```
function calcForce() {
    obj.force = new Vector(
        0, obj.mass * g — k * obj.vy);
}
```

$$F = m \cdot g - k \cdot v$$

```
function updateAccel() {
    obj.acc = force.divide(obj.mass);
}
```

$$a = \frac{F}{m}$$

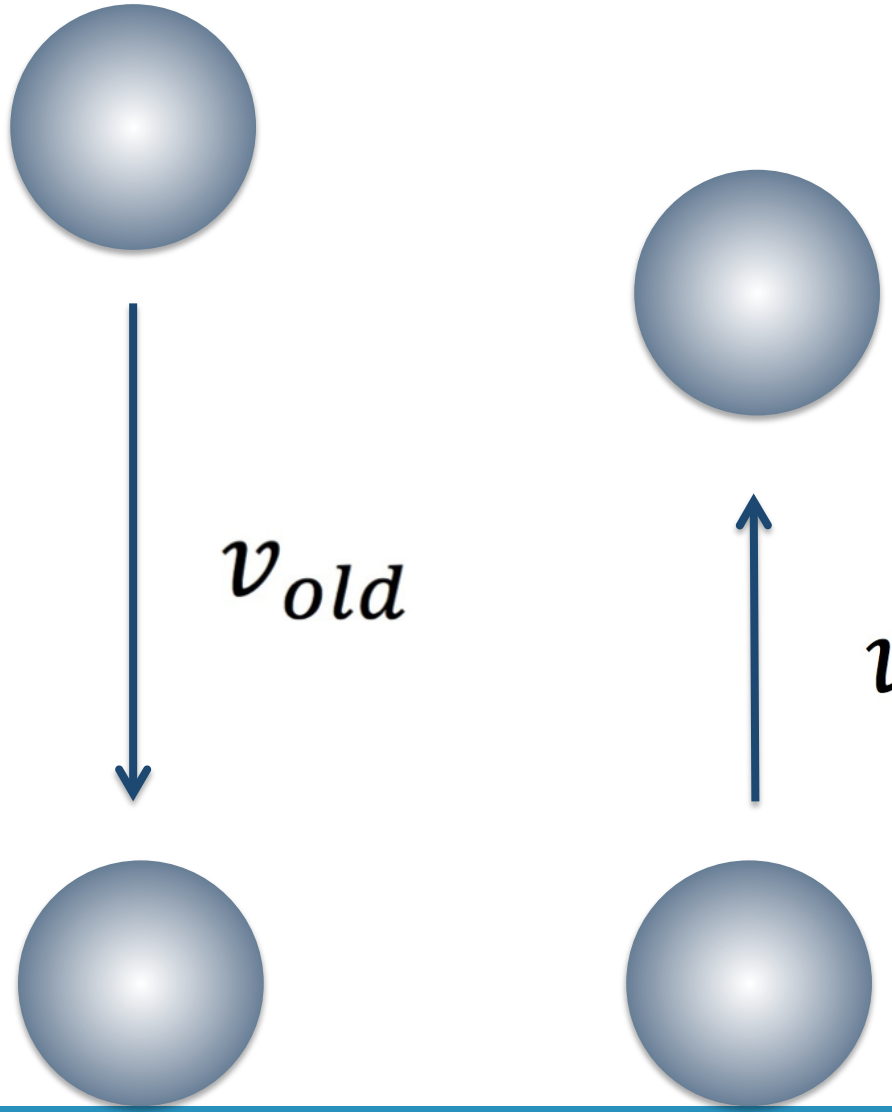
```
function updateVelo() {
    obj.velo = obj.velo.addScaled(obj.acc, dt);
}
```

$$v = a \cdot t$$

```
function moveObject() {
    obj.pos = obj.pos.addScaled(obj.velo, dt);
    obj.changeStyles();
}
```

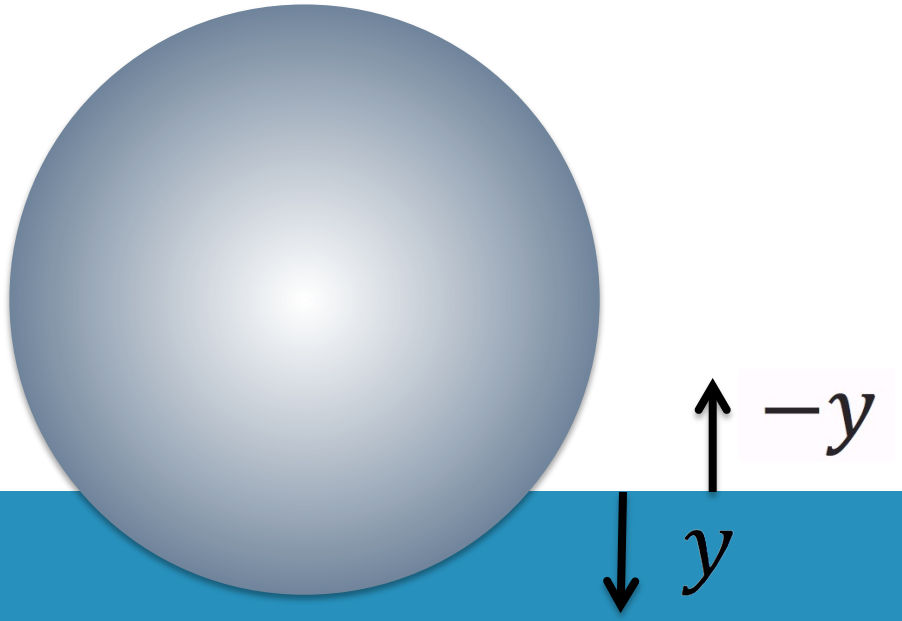
$$s = v \cdot t$$

УДАР О ПОВЕРХНОСТЬ



$$v_{new} = -k \cdot v_{old}$$

ЗАЛИПАНИЯ



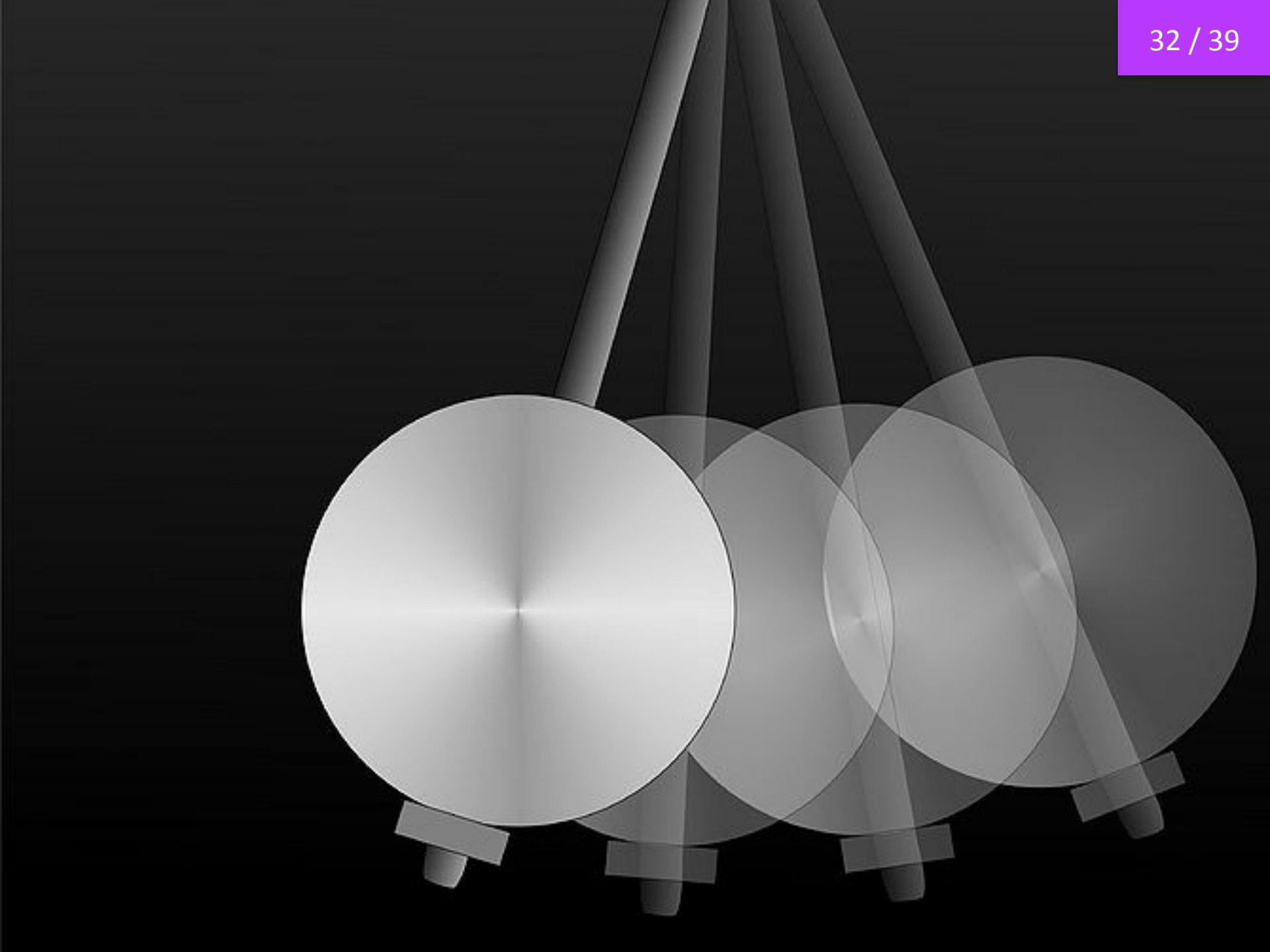
```
floor = new Vector(0, winH — floorH);


function checkBounce() {
    var displ = floor.subtract(obj.pos);
    if (displ.y — obj.radius <= 0) {
        obj.y = floor.y — obj.radius;
        obj.vy *= —vfac;
    }
}
```

```
floor = new Vector(0, winH — floorH);

function checkBounce() {
    var displ = floor.subtract(obj.pos);
    if (displ.y — obj.radius <= 0) {
        obj.y = floor.y — obj.radius;
        obj.vy *= —vfac;
    }
}
```

```
floor = new Vector(0, winH — floorH);

function checkBounce() {
    var displ = floor.subtract(obj.pos);
    if (displ.y — obj.radius <= 0) {
        obj.y = floor.y — obj.radius;
        obj.vy *= —vfac;
    }
}
```

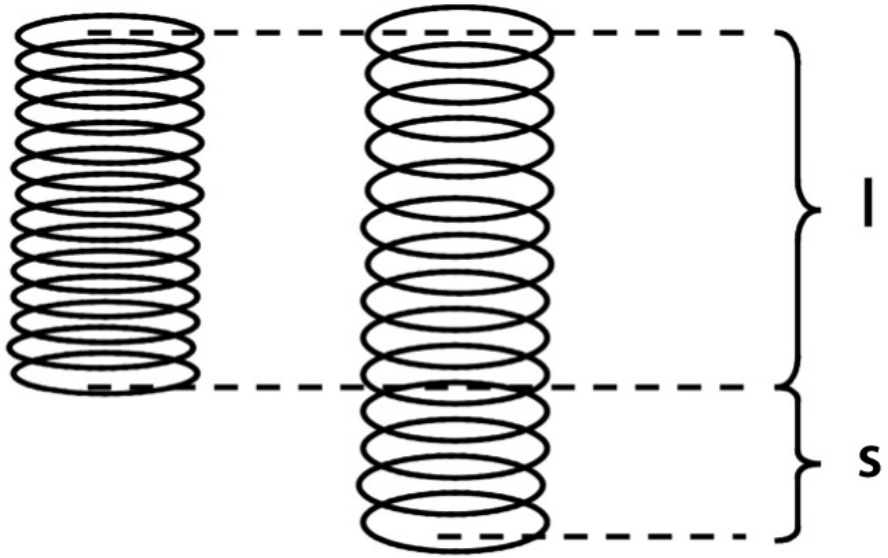
$$v_{new} = -k \cdot v_{old}$$

ПРУЖИНЫ
И КОЛЕБАНИЯ



ПРОСТОЕ ГАРМОНИЧЕСКОЕ КОЛЕБАНИЕ

Закон Роберта Гука



$$F = -k \cdot s$$

УСКОРЕНИЕ

Закон Гука

$$F = -k \cdot s$$

Второй закон Ньютона

$$F = a \cdot m$$

$$a = -\frac{k \cdot s}{m}$$

```
function calcForce() {
        var displ = obj.pos.subtract(center);
        var restoring = displ.multiply(—kSpring);
        var damping = obj.velo.multiply(—kDamping);

        obj.force = Vector.add([restoring, damping]);
}
```

```
function calcForce() {
    var displ = obj.pos.subtract(center);
    var restoring = displ.multiply(—kSpring);
    var damping = obj.velo.multiply(—kDamping);

    obj.force = Vector.add([restoring, damping]);
}
```

$$F_{sp} = -k_{sp} \cdot s$$

$$F_d = -k_d \cdot v$$

```
function calcForce() {
    ...
    cursorForce = new Vector(0, 0);
    if (cursorPos.x >= 0) {
        displCursor = obj.pos.subtract(cursorPos);
        cursorDist = displCursor.length();

        if (cursorDist > 1 &&
            cursorDist <= cursorForceZoomed) {

            var func = cursorForceZoomed / (cursorDist * cursorDist )
            cursorForce = displCursor.multiply(func);
        }
     }
    ...
}
```

```
function calcForce() {
    ...
    cursorForce = new Vector(0, 0);
    if (cursorPos.x >= 0) {
        displCursor = obj.pos.subtract(cursorPos);
        cursorDist = displCursor.length();

        if (cursorDist > 1 &&
            cursorDist <= cursorForceZoomed) {

            var func = cursorForceZoomed / (cursorDist * cursorDist )
            cursorForce = displCursor.multiply(func);
        }
    }
    ...
}
```

```
function calcForce() {
    ...
    cursorForce = new Vector(0, 0);
    if (cursorPos.x >= 0) {
        displCursor = obj.pos.subtract(cursorPos);
        cursorDist = displCursor.length();

        if (cursorDist > 1 &&
            cursorDist <= cursorForceZoomed) {

            var func = cursorForceZoomed / (cursorDist * cursorDist )
            cursorForce = displCursor.multiply(func);
        }
     }
    ...
}
```

```
function calcForce() {
    ...
    cursorForce = new Vector(0, 0);
    if (cursorPos.x >= 0) {
        displCursor = obj.pos.subtract(cursorPos);
        cursorDist = displCursor.length();

        if (cursorDist > 1 &&
            cursorDist <= cursorForceZoomed) {

            var func = cursorForceZoomed / (cursorDist * cursorDist )
            cursorForce = displCursor.multiply(func);
        }
    }
    ...
}
```

```
function calcForce() {
    ...
    cursorForce = new Vector(0, 0);
    if (cursorPos.x >= 0) {
        displCursor = obj.pos.subtract(cursorPos);
        cursorDist = displCursor.length();

        if (cursorDist > 1 &&
            cursorDist <= cursorForceZoomed) {

            var func = cursorForceZoomed / (cursorDist * cursorDist )
            cursorForce = displCursor.multiply(func);
        }
    }
    ...
}
```

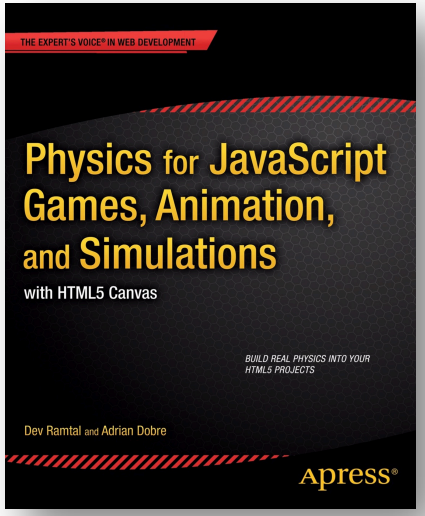
$$F = s \cdot \frac{Z}{d^2}$$

ССЫЛКИ

physicsdemos.liveldi.ru

github.com/elena90/physicsDemos

ССЫЛКИ



Physics for JavaScript Games, Animation, and Simulations (Dev Ramtal, Adrian Dobre)

Creating Animations and Interactions with Physical Models (Ralph Thomas)

СПАСИБО ЗА ВНИМАНИЕ

=

Иванова Елена

@liveldi90