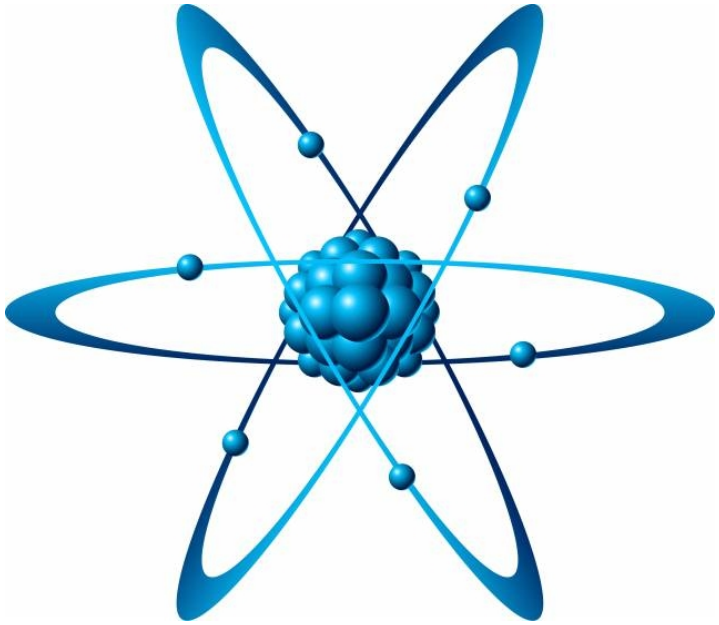
Анимируем объекты с использованием физики в JavaScript



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, dt;

function onTimer() {
    var t1 = new Date().getTime();
    dt = t1 — t0;
    t0 = t1;
    // Меняем кадры
}
```

КОНСТРУКТОРЫ

ОБЪЕКТ — параметры

```
var Obj = function (ops) {
    this.$el     // Доступ к элементу
    this.mass  // Масса

    this.x       // Позиция x, y
    this.y

    this.vx      // Скорость по x, y
    this.vy
    ... };
```

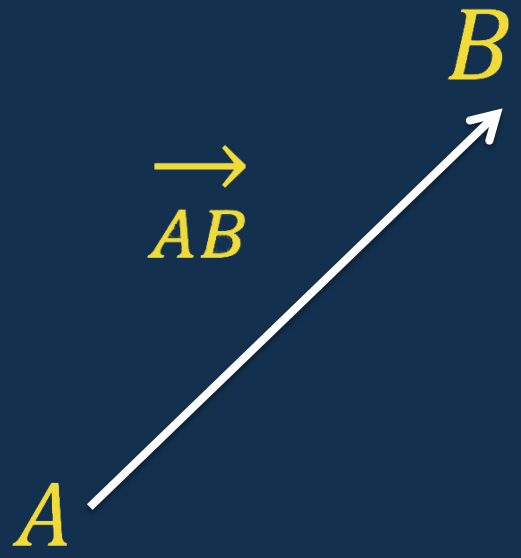
ОБЪЕКТ — методы

Obj.prototype.pos

Obj.prototype.velo

ВЕКТОР — параметры

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



ВЕКТОР — методы

Vector.prototype.length()

Vector.prototype.add(vec)

Vector.prototype.subtract(vec)

Vector.prototype.multiply(k)

Vector.prototype.divide(k)

ВЕКТОР — методы

```
Vector.prototype.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

$$F = -k \cdot v$$

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

ИЗМЕНЕНИЕ КИНЕТИЧЕСКОЙ ЭНЕРГИИ

$$\Delta E_k = P \cdot t$$

ИЗМЕНЕНИЕ КИНЕТИЧЕСКОЙ ЭНЕРГИИ

$$\Delta E_k = P \cdot t \qquad P = -k \cdot v^2$$

$$\Delta E_k = -k \cdot v^2 \cdot t$$

НАЧАЛЬНЫЕ РАСЧЕТЫ

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

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

```
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 КИНЕТИЧЕСКОЙ ЭНЕРГИИ

```
var powerLoss = new Vector(
                powerLossFactor * obj.vx * obj.vx * dt,
                powerLossFactor * obj.vy * obj.vy * dt);

ke = ke.subtract(powerLoss);
```

$$\Delta E_k = -k \cdot v^2 \cdot t$$

ОБНОВЛЕНИЕ СКОРОСТИ

```
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

```
obj.pos = obj.pos.addScaled(obj.velo, dt);
```

$$s = v \cdot t$$


100
120
140
160
180
200
220
240
260
280
300

ДОБАВЛЕНИЕ МОЩНОСТИ

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

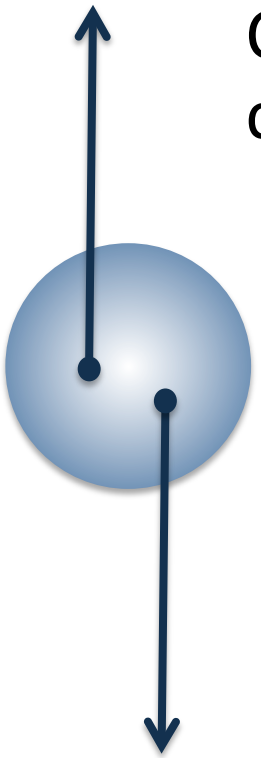
$$\Delta E_k = P \cdot t$$

УСКОРЕНИЕ И ПРЫЖКИ

УСКОРЕНИЕ

$$D = -k \cdot v$$

Сила сопротивления (D)



Гравитация (W)

$$W = m \cdot g$$

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

УСКОРЕНИЕ

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

$$v = a \cdot t$$

РАСЧЕТ СИЛЫ

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

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

ОБНОВЛЕНИЕ УСКОРЕНИЯ

```
obj.acc = force.divide(obj.mass);
```

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

ОБНОВЛЕНИЕ СКОРОСТИ

```
obj.velo = obj.velo.addScaled(obj.acc, dt);
```

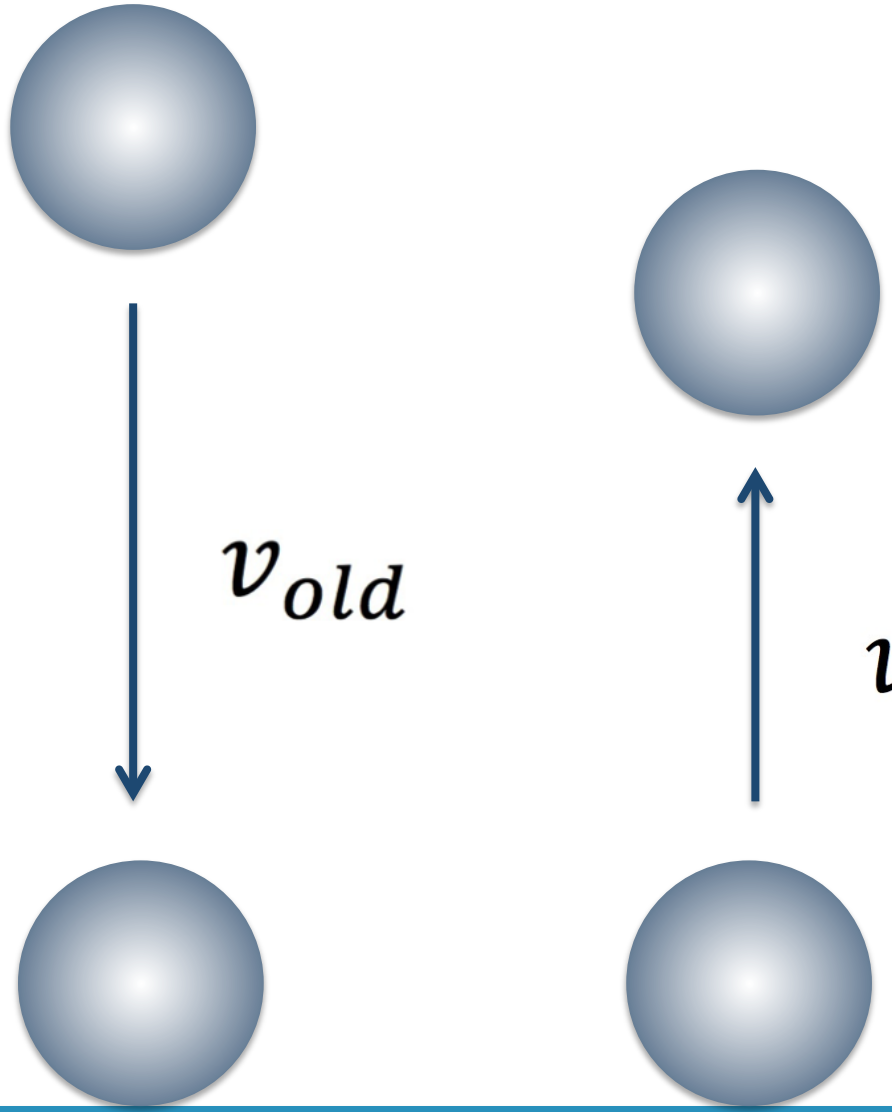
$$v = a \cdot t$$

ОБНОВЛЕНИЕ ПОЗИЦИИ

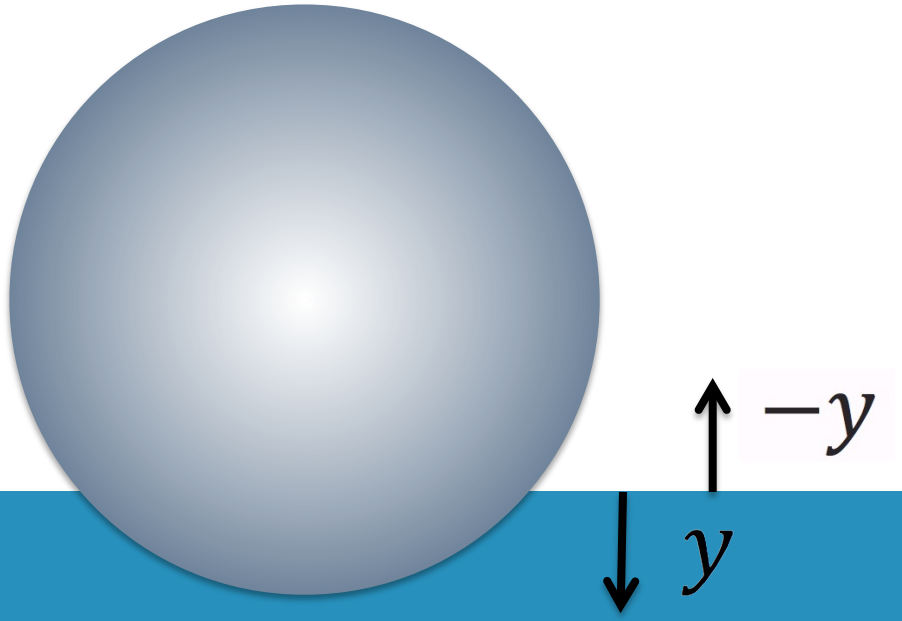
```
obj.pos = obj.pos.addScaled(obj.velo, dt);
```

$$s = v \cdot t$$

УДАР О ПОВЕРХНОСТЬ



ЗАЛИПАНИЯ



ПРОВЕРКА УДАРА

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


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

ПРОВЕРКА УДАРА

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


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

ПРОВЕРКА УДАРА

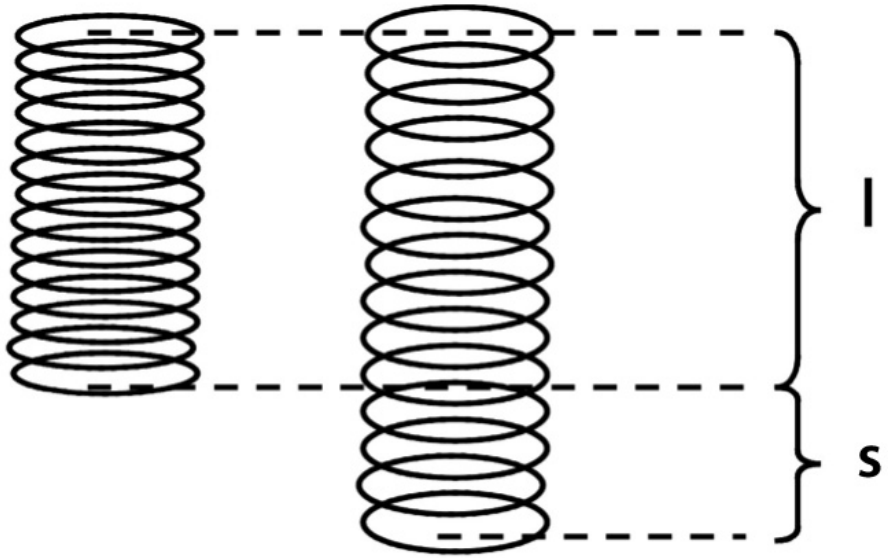
```
floor = new Vector(0, winH — floorH);

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

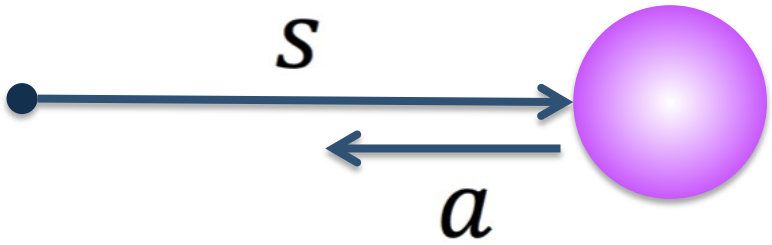
Закон Гука

$$F = -k \cdot s$$

Второй закон Ньютона

$$F = a \cdot m$$

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



РАСЧЕТ СИЛЫ

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

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

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

$$F_{damping} = -k_{damping} \cdot v$$


WHIT

РАСЧЕТ СИЛЫ КУРСОРА

```
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);
    }
}
```

РАСЧЕТ СИЛЫ КУРСОРА

```
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);
    }
}
```

РАСЧЕТ СИЛЫ КУРСОРА

```
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);
    }
}
```

РАСЧЕТ СИЛЫ КУРСОРА

```
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);
    }
}
```

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

РАСЧЕТ СИЛЫ КУРСОРА

```
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);
    }
}
```

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

Иванова Елена

@liveldi90

physicsdemos.liveldi.ru