ARKANA JS

Piotr Martyniak

DaftCode

CZYM JEST JAVASCRIPT

Językiem programowania 😸

- Interpretowanym
- Słabo i dynamicznie typowanym
- Wieloparadygmatowym
  - Obiektowym
  - Prototypowym
  - Event-driven
  - Funkcyjnym

Stworzony przez **Brendana Eicha** w 1995

WIĘCEJ PUNKCIKÓW

- Iterpretowany przez przeglądarki internetowe
- ...oraz interpretery systemowe
- Zgodnie z **ECMAScript**
- Posiada natywną obsługę (API):
  - tekstu
  - dat
  - tablic
  - wyrażeń regularnych
  - DOM

PODSTAWY

OBIEKTY & KONSTRUKTORY

Obiekt:

```
var obiekt = {
    klucz: 'wartość',
    przefunkcja: function (a, b) {
        return a
```

```
 b - 47;
    }
}
```

Konstruktor:

```
function Animal() {
    this.sound = 'trolololo'
}
```

```
var stephen = new Animal() // stephen: Object Animal {sound: "trolololo"}
```

```
var jason = Animal() // jason: undefined
```

```
window.sound // "trolololo"
```

ROZSZERZANIE OBIEKTÓW

```
var stephen = new Animal() // stephen: Object Animal {sound: "trololololo"}
```

```
stephen.likes = 'eating' // stephen: Object Animal {sound: "trololololo", likes: "eating"}
```

PROTOTYP & DZIEDZICZENIE

```
function Elephant() {
    this.trunk = 'yes'
    this.legs = 4
}
```

```
var maggie = new Elephant() // maggie: Object Elephant {trunk: "yes", legs: 4}
```

```
Elephant.prototype = new Animal()
var jason = new Elephant() // jason: Object Elephant {sound: "trolololo", trunk: "yes", legs: 4}
```

```
Object.getPrototypeOf(jason) // Object Animal { ... }
jason.__proto__ // Object Animal { ... } (deprecated)
```

Ale...

```
Animal.isPrototypeOf(jason) // false (prototypem jest instancja "new Animal()")
Medusa.prototype = Animal // prototypem będzie "function"
```

```
var ronaldTheSquid = Object.create(new Animal, { tentacles: 'indeed' })
```

KLASY (ES6) VS. PROTOTYPOWANIE

```
class Elephant extends Animal {
    static soundMade = 'trolololo'

    constructor() {
        super()
        this.trunk = 'yes'
        this.legs = 4
    }

    //metody klasy:
    eat(fruit) {
        if (this.trunk === 'yes') {
            this.grab(fruit)
        }
    }
}
```

```
'strict mode'

function Elephant() {
    this.trunk = 'yes'
    this.legs = 4
}

Elephant.soundMade = 'trolololo' // static
Elephant.prototype.constructor = Animal // exten
Elephant.prototype.eat = function (fruit) { }
```

DELEGACJA

```
let obj = {things: 3}
let addThings = function(a, b, c){
 return this.things + a + b + c
}
```

call & apply

```
x = addThings.call(obj, 1,4,6) // x: 14
y = addThings.apply(obj, [1,4,6]) // y: 14
```

bind

```
z = addThings.bind(obj, 1,4,6) // z: function ()
let val = z() // val: 14
```

PODSTAWY II

ZASIĘG I DOMKNIĘCIE

(a.k.a. scope & closure)

ZASIĘG ZMIENNYCH

```
var x = 16
var y = 17

function changeVars() {
    x = 24
    z = 32

    var y = 18
    var p = 2

    for (var i = 0; i < y; i++) {
        // do stuffs
    }
    console.log(i)
}

changeVars()

console.log(x) // 24
console.log(y) // 17
console.log(z) // 32
console.log(p) // "ReferenceError: p is not defined"
```

`var` vs. `let` (ES6)

```
function something() {
    // v widoczne tu

    for( var v = 0; v < 5;  v++ ) {
        // v widoczne, ale żyje piętro wyżej
    }

    // v widoczne tutaj
}
```

```
function somethingElse() {
    // l nie ma tutaj

    for( let l = 0; l < 5; l++ ) {
        // l widoczne tylko tu (i głębiej)
        // każda iteracja ma nową instancję l
    }

    // l tutaj już nie ma
}
```

globalnie:

```
let me = 'go';
console.log(window.me); // undefined
```

```
var i = 'able';
console.log(window.i); // 'able'
```

```
'use strict';
let me = 'foo';
let me = 'bar'; // SyntaxError: Identifier 'me'
                // has already been declared
```

```
'use strict';
var me = 'foo';
var me = 'bar'; // No problem
```

var

```
for(var i = 1; i < 6; i++) {
  document.getElementById('my-element' + i)
    .addEventListener('click', function() { alert(i) })
}
```

let

```
for(let i = 1; i < 6; i++) {
  document.getElementById('my-element' + i)
    .addEventListener('click', function() { alert(i) })
}
```

DOMKNIĘCIE

ale najpierw... IIFE!

Immediately Invoked Function Expression

```
let kwadrat7 = (function (param) {
    console.log('robimy kwadrat')

    return param * param
})(7) // 👈 od razu wywołana
```

DOMKNIĘCIE

```
let giveMeMore = (function () {
    let i = 0 // zmienna prywatna

    const iterator = function() {
        return i++ // operacja na prywatnej zmiennej
    }
    return iterator
})()

giveMeMore() // 0
giveMeMore() // 1
giveMeMore() // 2
giveMeMore() // 3
giveMeMore() // 4
```

STRICT MODE

```
'use strict'

// Assignment to a non-writable global
var undefined = 5 // TypeError
var Infinity = 5 // TypeError

// Assignment to a non-writable property
var obj1 = {}
Object.defineProperty(obj1, 'x', { value: 42, writable: false })
obj1.x = 9 // TypeError

// Assignment to a getter-only property
var obj2 = { get x() { return 17 } }
obj2.x = 5 // TypeError

// Assignment to a new property on a non-extensible object
var fixed = {}
Object.preventExtensions(fixed)
fixed.newProp = 'ohai' // TypeError

delete Object.prototype // TypeError

var o = { p: 1, p: 2 } // syntax error

function sum(a, a, c) { // syntax error
}

eval('var x = 123')
console.log(x) // undefined
```

WZORCE PROJEKTOWE W JS

a.k.a. design patterns

MODUŁ

```
var secretClub = function (config) {
    // przestrzeń prywatna:
    var privateMembers = []
    var secretLocation = config.location

    // publiczny "interfejs" modułu
    return {
        addMember: function (person) {
            privateMembers.push(person)
        },
        getMemberCount: function () {
            return privateMembers.length
        }
    }
}

var illuminati = secretClub({ location: 'Radom' })

illuminati.addMember('Reptilian George')
illuminati.addMember('Tom Cruise')
illuminati.addMember('Magda Gessler')

illuminati.getMemberCount() // 3

console.log(illuminati.privateMembers) // undefined
console.log(illuminati.secretLocation) // undefined
```

MODUŁ - PODSTAWOWY BUDULEC APLIKACJI

```
// składnia commonJS (nodejs)

// biblioteki/inne moduły
var moment = require('moment')
var WorldOrder = require('world-order')

// przestrzeń prywatna:
var _privateMembers = []
var _secretLocation = config.location
var _newWorldOrder = new WorldOrder()

// prywatne funkcje
var _conspire () {
    if (moment().day() % 7) { _newWorldOrder.meddle() }
}

// publiczne "API" wyeksportowane
module.exports = {
    addMember: function (person) {
        privateMembers.push(person)
    },
    getMemberCount: function () {
        return privateMembers.length
    }
}
```

```
var secretClub = require('secretClubModule')
```

MODUŁ W ES5

```
// składnia ES5
// biblioteki/inne moduły
import moment from 'moment'
import WorldOrder from 'world-order'

// przestrzeń prywatna:
var _privateMembers = []
var _secretLocation = config.location
var newWorldOrder = new WorldOrder()

// prywatne funkcje
var _conspire () {
    if (moment().day() % 7) { newWorldOrder.meddle() }
}

// publiczne "API" wyeksportowane
export default {
    addMember: function (person) {
        privateMembers.push(person)
    },
    getMemberCount: function () {
        return privateMembers.length
    }
}
```

```
import secretClub from 'secretClubModule'
```

OBSERWATOR 🔭

```
class ObserverList {
    constructor() {
        this.observerList = []
    }

    count () {}
    add (obj) {}
    get (index) {}
    indexOf (obj, startIndex) {}
    removeAt (index) {}
}
```

```
class Subject {
    constructor() {
        this.observers = new ObserverList()
    }

    addObserver (observer) {
      this.observers.add( observer )
    }

    removeObserver (observer) {
        this.observers.removeAt( this.observers.indexOf( observer, 0 ) )
    }

    notify (event) {
        var observerCount = this.observers.count()
        for (let i = 0; i < observerCount; i++){
            this.observers.get(i).update(event)
        }
    }
}
```

PUBLISH/SUBSCRIBE 🗞️

```
import pubsub from 'pubsub-js'

// subskrybent
var mySubscriber = function (msg, data) {
    console.log( msg, data );
};

// dodajemy do listy subskrybentów na dany "temat"
var token = PubSub.subscribe('MY TOPIC', mySubscriber);

// ...in a galaxy far far away

// publikujemy:
PubSub.publish('MY TOPIC', { data: 'tie-fighters'});
```

FABRYKA 🏭

```
class Car {
    constructor (options) {
        this.doors = options.doors || 4
        this.state = options.state || "brand new
        this.color = options.color || "silver"
    }
}
```

```
class Truck{
    constructor (options) {
        this.state = options.state || "used"
        this.color = options.color || "blue"
        this.maxCapacity = options.maxCapacity
    }
}
```

```
class VehicleFactory() {
    constructor () {
        this.vehicleClass = Car
        this.producedCars.push(producedVehicle)
    }

    createVehicle (options) {
        if (options.vehicleType === "car") {
            this.vehicleClass = Car
        } else if (options.vehicleType === "truck") {
            this.vehicleClass = Truck
        }
        var producedVehicle = new this.vehicleClass(options)
        this.producedCars.push(producedVehicle)
        return producedVehicle
    }

    explodeAllProducedVehicles () {
        this.producedVehicles.forEach(function (vehicle) {
            this.explodeVehicle(vehicle)
        }
    }
    // ...
}
```

MIXIN I DEKORATOR

```
var moveMixin = {
    moveUp: function (amount) {
        if(this.y || this.y === 0) {
            this.y +=amount
        }
    },
    moveDown: function (amount) {},
    moveLeft: function (amount) {},
    moveRight: function (amount) {},
}
```

```
var moveDecorator = function(obj, initial) {
    obj.x = initial.x || 0
    obj.y = initial.y || 0
    var moveUp = function (amount) {}
    var moveDown = function (amount) {}
    var moveRight = function (amount) {}
    var moveLeft = function (amount) {}
    obj.moveUp = (moveUp).bind(obj)
    obj.moveDown = (moveDown).bind(obj)
    obj.moveRight = (moveRight).bind(obj)
    obj.moveLeft = (moveLeft).bind(obj)
}
```

```
thing.prototype = moveMixin
```

```
moveDecorator(thing)
```

- singleton
- mediator
- fasada
- flightweight

MV*

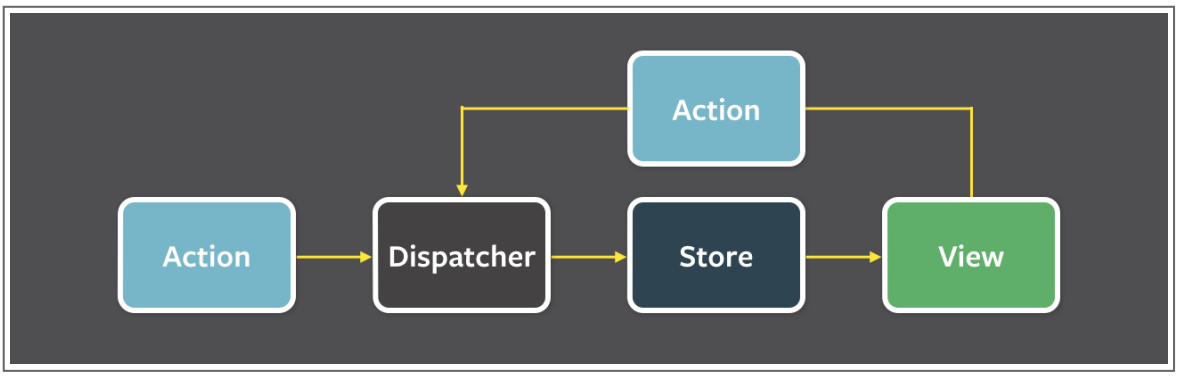
MVC: Model View Controller

MVP: Model View Presenter

MVVM: Model View ViewModel

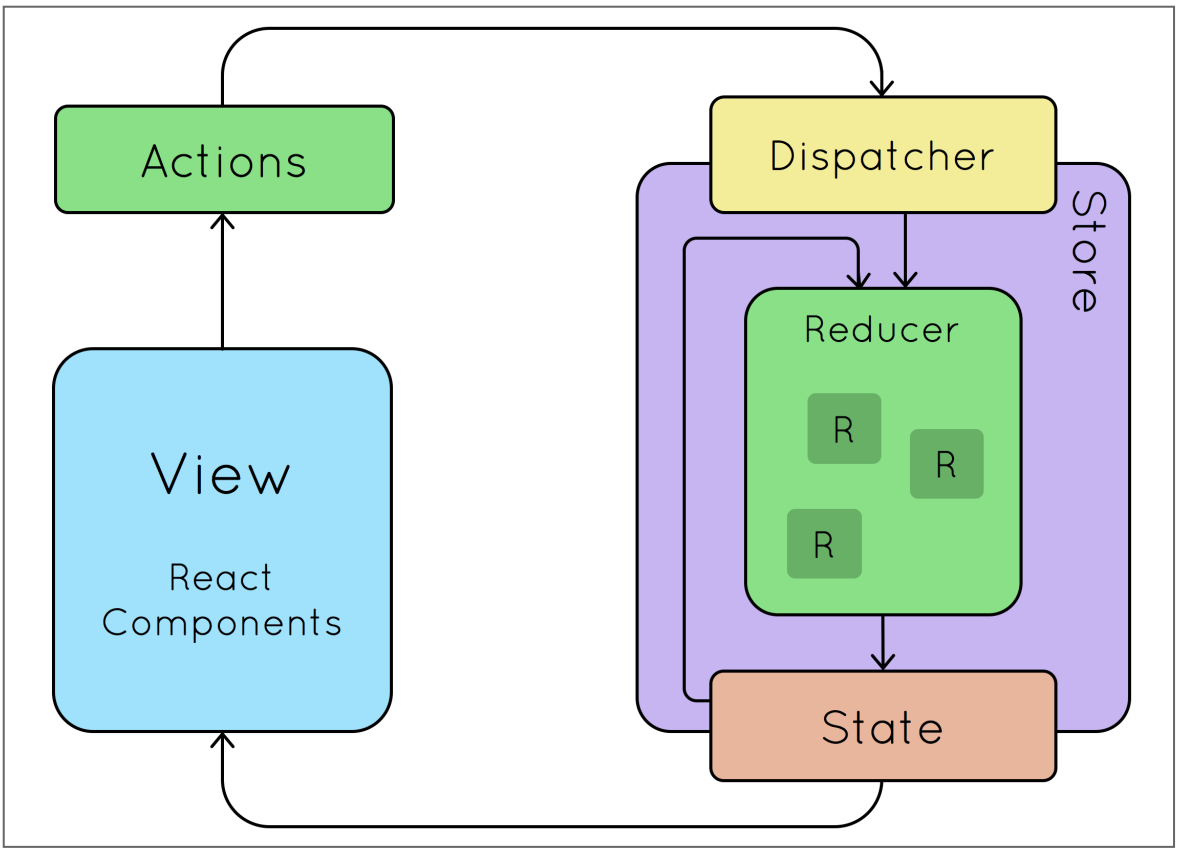
FLUX

- pochodzi z Reacta
- Widoki + Magazyny + Dispatcher + Akcje
- Jednokierunkowy przepływ zdarzeń (akcji)
- Ma kilka implementacji



REDUX

- Inspirowany fluxem
- Jednokierunkowy przepływ zdarzeń
- Widoki + 1 Magazyn + Reducery + Akcje
- Magazyn "immutable": tzw. jedno źródło prawdy
- Wcale nie ograniczony do Reacta


Actions
Dispatcher
Store
Reducer
R
R
R
View
React
Components
State

NOWOCZESNY JS

ECMAscript 2015

a.k.a. ES6

`let` & `const`

```
//const - tylko referencja jest stała!
const x = 17
x++ // no way Jose

// ale...
const thing = { x = 17 }
thing.x = 18 // no problemo
```

```
Object.freeze(thing)

this.x = 3 // nope!
```

FUNKCJE STRZAŁKOWE

arrow functions

```
// es6
const adding = (a, b) => a + b
```

```
// es5
var adding = function (a, b) { return a + b }
```

```
// es6
nums.forEach(v => {
   if (v % 5 === 0)
       fives.push(v)
})
```

```
// es5
nums.forEach(function (v) {
   if (v % 5 === 0)
       fives.push(v);
});
```

this

```
// es6
this.nums.forEach((v) => {
    if (v % 5 === 0)
        this.fives.push(v)
})
```

```
// es5
var self = this;
this.nums.forEach(function (v) {
    if (v % 5 === 0)
        self.fives.push(v);
});
```

DOMYŚLNE WARTOŚCI PARAMETRÓW

```
// es6
function f (x, y = 7, z = 42) {
    return x + y + z
}
f(1) === 50
```

```
// es5
function f (x, y, z) {
    if (y === undefined)
        y = 7;
    if (z === undefined)
        z = 42;
    return x + y + z;
};
```

PARAMETR ...REST

```
// es6
function f (x, y, ...a) {
    return (x + y) * a.length
}
f(1, 2, "hello", true, 7) === 9
```

```
// es5
function f (x, y) {
    var a = Array.prototype
        .slice
        .call(arguments, 2)
    return (x + y) * a.length
};
f(1, 2, "hello", true, 7) === 9
```

OPERACJA SPREAD

```
// es6
var params = [ "hello", true, 7 ]
var other = [ 1, 2, ...params ]
        // [ 1, 2, "hello", true, 7 ]
```

```
// es5
var params = [ "hello", true, 7 ];
var other = [ 1, 2 ].concat(params);
        // [ 1, 2, "hello", true, 7 ]
```

```
// es6
f(1, 2, ...params)
```

```
// es5
f.apply(undefined, [ 1, 2 ].concat(params))
```

```
// es6
const a = { turtle: 'pink', whale: 'yellow' }
const b = {
    sheep: 'black',
    ...a
}
```

```
// es5
var a = { turtle: 'pink', whale: 'yellow' }
var b = { sheep: 'black' }
for (var key in a) {
    if(a.hasOwnProperty(key) {
        b[key] = a[key]
    }
}
```

INTERPOLACJA TEKSTU

```
// es6
var message = `Hello ${customer.name},
want to buy ${card.amount} ${card.product}
for a total of
${card.amount * card.unitprice} bucks?`
```

```
// es5
var message = "Hello " + customer.name + ",\n" +
"want to buy " + card.amount + " " + card.produc
"\n for a total of\n" +
(card.amount * card.unitprice) + " bucks?"
```

🔑 KLUCZE OBIEKTÓW 🗝

```
// es6
var x = 0, y = 0
obj = { x, y }
```

```
// es5
var x = 0, y = 0
obj = { x: x, y: y }
```

```
// es6
let obj = {
    foo: "bar",
    [ "baz" + quux() ]: 42
}
```

```
// es5
var obj = {
    foo: "bar"
}
obj[ "baz" + quux() ] = 42
```

```
// es6
obj = {
    foo (a, b) {
        …
    },
    bar (x, y) {
        …
    }
}
```

```
// es5
obj = {
    foo: function (a, b) {
        …
    },
    bar: function (x, y) {
        …
    }
}
```

🤘 DESTRUKCJA 🤘

```
// es6
var list = [ 1, 2, 3 ]
var [ a, , b ] = list
[ b, a ] = [ a, b ]
```

```
// es5
var list = [ 1, 2, 3 ];
var a = list[0], b = list[2];
var tmp = a; a = b; b = tmp;
```

```
// es6
var { op, lhs, rhs } = getASTNode()
```

```
// es5
var tmp = getASTNode()
var op  = tmp.op
var lhs = tmp.lhs
var rhs = tmp.rhs
```

```
// es6
var obj = { a: 1 }
var list = [ 1 ]
var { a, b = 2 } = obj
var [ x, y = 2 ] = list
```

```
// es5
var a = obj.a
var b = obj.b === undefined ? 2 : obj.b
var x = list[0]
var y = list[1] === undefined ? 2 : list[1]
```

```
// es6
function f ([ name, val ]) {
    console.log(name, val)
}
function g ({ name: n, val: v }) {
    console.log(n, v)
}
function h ({ name, val }) {
    console.log(name, val)
}
f([ "bar", 42 ])
g({ name: "foo", val:  7 })
h({ name: "bar", val: 42 })
```

```
// es5
function f (arg) {
    var name = arg[0]
    var val  = arg[1]
    console.log(name, val)
}
function g (arg) {
    var n = arg.name
    var v = arg.val
    console.log(n, v)
}
function h (arg) {
    var name = arg.name
    var val  = arg.val
    console.log(name, val)
}
```

MODUŁY

```
//  lib/math.js
export function sum (x, y) { return x + y }
export var pi = 3.141593

//  someApp.js
import * as math from "lib/math"
console.log("2π = " + math.sum(math.pi, math.pi))

//  otherApp.js
import { sum, pi } from "lib/math"
console.log("2π = " + sum(pi, pi))
```

```
//  lib/mathplusplus.js
export * from "lib/math"
export var e = 2.71828182846
export default (x) => Math.exp(x)

//  someApp.js
import exp, { pi, e } from "lib/mathplusplus"
console.log("e^{π} = " + exp(pi))
```

KLASY

```
class Shape {
    constructor (id, x, y) {
        this.id = id
        this.move(x, y)
    }
    move (x, y) {
        this.x = x
        this.y = y
    }
    toString () {
        return `Shape(${this.id})`
    }
}
```

```
// dziedziczenie
class Rectangle extends Shape {
    constructor (id, x, y, width, height) {
        super(id, x, y)
        this.width  = width
        this.height = height
    }

    toString () {
        return "Rectangle > " + super.toString()
    }

    // getter i setter
    set width  (width)  { this._width = width }
    get width  ()       { return this._width }

    // metoda statyczna
    static defaultRectangle () {
        return
            new Rectangle("default", 0, 0, 10, 1
    }
}
```

DZIEDZICZENIE PO WIELU KLASACH

```
class Colored { // mixin
    initializer ()     { this._color = "white" }
    get color ()       { return this._color }
    set color (v)      { this._color = v }
}

class ZCoord { // mixin
    initializer ()     { this._z = 0 }
    get z ()           { return this._z }
    set z (v)          { this._z = v }
}

class Shape { // klasa <bazowa></bazowa>
    constructor (x, y) { this._x = x; this._y = y }
    get x ()           { return this._x }
    set x (v)          { this._x = v }
    get y ()           { return this._y }
    set y (v)          { this._y = v }
}
```

```
var aggregation = require("aggregation/es6") // https://github.com/rse/aggregation

class Rectangle extends aggregation(Shape, Colored, ZCoord) {
    // ...definicja samego Rectangle
}

const rect = new Rectangle(7, 42)
rect.z     = 1000
rect.color = "red"
```

SYMBOLE ♏

```
const obj = {
[Symbol('my_key')]  : 1,
[Symbol('my_key')]  : 2,
[Symbol('key')]     : 3,
   a                : 4,
   b                : 5
};

> Object.getOwnPropertyNames(obj)
['a', 'b']

> Object.getOwnPropertySymbols(obj)
[Symbol(my_key),Symbol(my_key),Symbol(key)]

> Reflect.ownKeys(obj)
[Symbol(my_key),Symbol(my_key),Symbol(key),'a', 'b']

> Object.keys(obj)
['a','b']

> Symbol('foo') === Symbol('foo')
false
```

"FOR ... OF" I ITERATORY

```
let fibonacci = {
    [Symbol.iterator]() {
        let pre = 0, cur = 1
        return {
           next () {
               [ pre, cur ] = [ cur, pre + cur ]
               return { done: false, value: cur }
           }
        }
    }
}
```

```
for (let n of fibonacci) {
    if (n > 1000)
        break
    console.log(n)
}
```

GENERATORY

```
function* range (start, end, step) {
    while (start < end) {
        yield start
        start += step
    }
}

for (let i of range(0, 10, 2)) {
    console.log(i) // 0, 2, 4, 6, 8
}
```

```
let fibonacci = function* (numbers) {
    let pre = 0, cur = 1
    while (numbers-- > 0) {
        [ pre, cur ] = [ cur, pre + cur ]
        yield cur
    }
}

for (let n of fibonacci(1000))
    console.log(n)

let numbers = [ ...fibonacci(1000) ]

let [ n1, n2, n3, ...others ] = fibonacci(1000)
```

📚 ZBIÓR I MAPA 🗺️

```
let s = new Set()
s.add("hello").add("goodbye").add("hello")
s.size === 2
s.has("hello") === true
for (let key of s.values())
    console.log(key)
```

```
let m = new Map()
let s = Symbol()
m.set("hello", 42)
m.set(s, 34)
m.get(s) === 34
m.size === 2
for (let [ key, val ] of m.entries())
    console.log(key + " = " + val)
```

WeakSet & WeakMap

WBUDOWANE METODY

```
// Object.assign:

var dest = { quux: 0 }
var src1 = { foo: 1, bar: 2 }
var src2 = { foo: 3, baz: 4 }
Object.assign(dest, src1, src2)

// String:

"foo-".repeat(3)               // foo-foo-foo-
"hello".startsWith("ello", 1) // true

"hello".endsWith("hell", 4)   // true

"hello".includes("ell")       // true
"hello".includes("ell", 1)    // true
"hello".includes("ell", 2)    // false
```

```
// Array:

[ 1, 3, 4, 2 ].find(x => x > 3) // 4
[ 1, 3, 4, 2 ].findIndex(x => x > 3) // 2

// Number:

Number.isNaN(42)                        // false
Number.isNaN(NaN)                       // true

Number.isFinite(Infinity)               // false
Number.isFinite(-Infinity)              // false
Number.isFinite(NaN)                    // false
Number.isFinite(123)                    // true
Number.isSafeInteger(42)                // true
Number.isSafeInteger(9007199254740992) // false

Math.trunc(42.7)                        // 42
Math.trunc( 0.1)                        // 0
Math.trunc(-0.1)                        // -0

Math.sign(7)                            // 1
Math.sign(0)                            // 0
Math.sign(-0)                           // -0
Math.sign(-7)                           // -1
Math.sign(NaN)                          // NaN
```

PROMISY

```
const msgAfterTimeout = (msg, who, timeout) => {
    return new Promise((resolve, reject) => {
        setTimeout(() => resolve(`${msg} Hello ${who}!`), timeout)
    })
}

msgAfterTimeout("", "Foo", 100).then((msg) =>
    msgAfterTimeout(msg, "Bar", 200)
).then((msg) => {
    console.log(`done after 300ms:${msg}`)
})
```

```
Promise.all([
    fetchPromised("http://backend/foo.txt"),
    fetchPromised("http://backend/bar.txt"),
    fetchPromised("http://backend/baz.txt"),
]).then(([ foo, bar, baz ]) => {
    console.log(`success: foo=${foo} bar=${bar} baz=${baz}`)
}, (err) => {
    console.log(`fetch error: ${err}`)
})
```

NOWOCZESNY JS

ES7, ES8, ES9

WBUDOWANE METODY

- Object.values
- Object.entries
- Array.prototype.includes
- String.prototype.strPad

```
'John'.strPad(7, '0') // "0000John"
```

ASYNC/AWAIT

```
async function fetchContent() {

    let content = await fetch('/api/content') // poczeka na content
    let text = await content.text() // poczeka na text

    return text // zwróci Promise, który zostanie rozwiązany (resolved) jak wykonają się await'y
}
```

PRZECINKI

```
let obj = {
    first: 'Jane',
    last: 'Doe',
}
```

...po co?

```
// git diff:
  let obj = {
    first: 'Jane',
    last: 'Doe',
+   next: 'es9',
  }
```

```
// git diff:
  let obj = {
    first: 'Jane',
-   last: 'Doe'
+   last: 'Doe',
+   next: 'es9',
  }
```

DEKORATORY

```
class Cat {
    meow () { return this.name + ' says meow' }
}
```

```
// dekorator:
const readOnly = (target, name, descriptor) => {
    descriptor.writable = false
    return descriptor
}
```

```
// udekorowana metoda
class Cat {
    @readOnly
    meow () { return this.name + ' says meow' }
}
```

```
// dekorować można klasy
const withLegs = (legCount) => (WrappedClass) => {
    class WithLegs extends WrappedClass {
        constructor(...args) {
            super(...args)
            this.legs = legCount
        }
    }
    return WithLegs
}

@withLegs(4) // i dodawać konfigurację!
class Cat { ... }
```

LEKTURY

- Douglas Crockford - **Javascript the Good Parts**
- Addi Osmani - **Learning JavaScript Design Patterns**

```
https://addyosmani.com/resources/essentialjsdesignpatterns/book/
```

- Kyle Simpson - **You Don't Know JS**

```
https://github.com/getify/You-Dont-Know-JS
```

- ECMAscript 6:

```
http://es6-features.org
```

ZADANIE DOMOWE

Stwórz stronę wyświetlającą zegar

1. Użyj generatora/ów do zwracania kolejnej sekundy/minuty/godziny. Generator(y) mają działać tak, by po otrzymaniu wartości 59 (lub 12 dla godzinowego) wyołanie generator.next() dawało 00 (iteracja cykliczna)
2. Generator sekundowy wywołuj w setInterval, pozostałe gdy generator mniejszej jednostki osiągnie wartość 59.
3. Zegar powinien przyjmować godzinę wejścia na stronę jako punkt startowy (jeżeli będziesz miał/a problem z tym punktem - postaraj się zrobić pozostałe - każdy punkt oceniamy osobno)
4. Zegar powinien być wyświetlany w formacie hh:mm:ss (np 07:15:09 - by otrzymać 07 użyj metody es7-es9)
5. Gotowy zegar wrzuć na GHP

Muchas gracias!

https://m8ms.github.io/arkana_JS/