

## Programowanie skryptowe

Dr inż. Stanisław Polak

Grupa Systemów Komputerowych

<https://www.icsr.agh.edu.pl/~polak/>



## Plan wykładu

- ▶ Język JavaScript
- ▶ Obiektowy model dokumentu HTML
- ▶ Biblioteka React
- ▶ Podstawowe zagadnienia związane z usługą WWW
- ▶ Środowisko uruchomieniowe NodeJS
- ▶ Framework „Express.js”
- ▶ Asynchroniczne zapytania
- ▶ Podstawy języka TypeScript
- ▶ Język Ruby

Notatki

---

---

---



Notatki

---

---

---



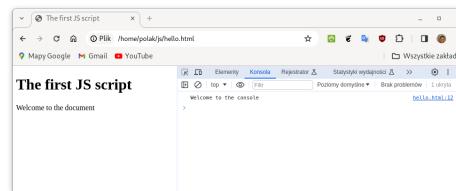
## Pierwszy skrypt JS

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>The first JS script</title>
5   </head>
6   <body>
7     <h1>The first JS script</h1>
8     <script>
9       document.write("Welcome to the document");
10      console.log("Welcome to the console");
11    </script>
12  </body>
13 </html>

```

Dokument HTML z treścią skryptu JS



### Notatki

---

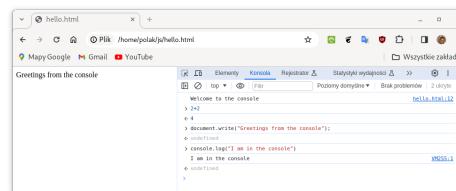


---



---

Rysunek: Wynik wykonania skryptu JS po zakończeniu renderowania strony WWW



Rysunek: Korzystanie z konsoli JS



## Osadzanie kodu JS

```

1 <script>
2   Treść skryptu JavaScript
3   The content of the JavaScript script
4 </script>
5 <noscript>
6   Twoja przeglądarka nie obsługuje JavaScript!<br>
7   Your browser does not support JavaScript!
8 </noscript>

```

Skrypt wewnętrzny osadzony w HTML

```

1 <script src="URL"></script>
2 <noscript>
3   Twoja przeglądarka nie obsługuje JavaScript!<br>
4   Your browser does not support JavaScript!
5 </noscript>

```

Odwołanie do skryptu zewnętrznego

### Notatki

---



---



---



## Kolejność wykonywania skryptów

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
5 <script>
6 function info(arg){
7   var img = document.getElementsByTagName("img");
8   var names='';
9   var img_msg = '\nThe "img" element is unavailable\n';
10  if(img.length != 0)
11    img_msg = '\nThe "img" element is available\n';
12  if(document.body){
13    x=document.body.childNodes
14    for (i=0;i<x.length;i++){
15      if(x[i].nodeType == 1)
16        names+=x[i].nodeName+", ";
17    }
18    console.log(arg+img_msg+"'document.body' contains elements: "+names);
19  }
20  else
21    console.log(arg+img_msg+"'document.body' = NULL");
22 }
23 </script>
24 <script>info("Script 1");</script>
25 <script src="a.js"></script>
26 </head>
27 <body onLoad="info('Script 4');">
28 <script>info("Script 2");</script>
29 <script src="b.js"></script>
30 
31 <script>info("Script 3");</script>
32 <script src="c.js"></script>
33 </body>
34 </html>

```

1 info('<file name>');

a.js, b.js, c.js

### Notatki

---



---



---



---



---

### Wypisze:

1. Script 1  
The "img" element is unavailable  
'document.body'=NULL
2. a.js  
The "img" element is unavailable  
'document.body'=NULL
3. Script 2  
The "img" element is unavailable  
'document.body' contains elements: SCRIPT,
4. b.js  
The "img" element is unavailable  
'document.body' contains elements: SCRIPT, SCRIPT,
5. Script 3  
The "img" element is available  
'document.body' contains elements: SCRIPT, SCRIPT, IMG, SCRIPT,
6. Script 4  
The "img" element is available  
'document.body' contains elements: SCRIPT, SCRIPT, IMG, SCRIPT, SCRIPT,
7. c.js  
The "img" element is available  
'document.body' contains elements: SCRIPT, SCRIPT, IMG, SCRIPT, SCRIPT,
8. Script 4  
The "img" element is available  
'document.body' contains elements: SCRIPT, SCRIPT, IMG, SCRIPT, SCRIPT,

## Asynchroniczne lub odroczone wykonywanie skryptów zewnętrznych



### Notatki

---



---



---



---

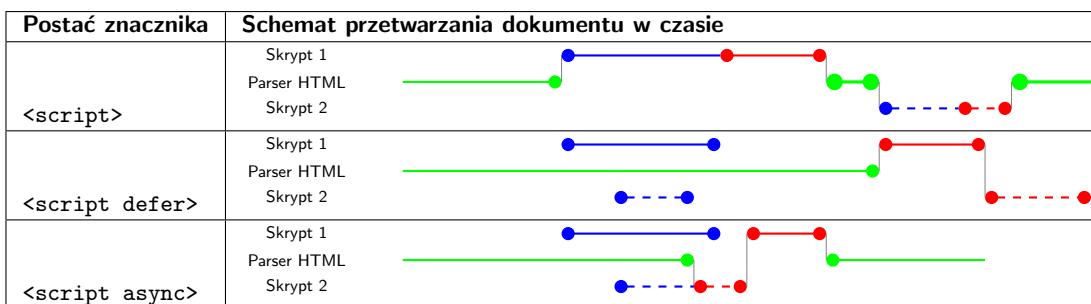


---

```

1 ...
2 <script async src="https://.../js1.js">
3 ...
4 <script async src="https://.../js2.js">
5 ...

```



Parsowanie Pobieranie Wykonywanie

Źródło: <https://www.growingwiththeweb.com/2014/02/async-vs-defer-attributes.html>



## Moduły

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta http-equiv="Content-Type" content="text/html;
5       charset=UTF-8">
6     <title>The third JS script</title>
7   </head>
8   <body>
9     <h1>The third JS script</h1>
10    <script type="module">
11      import { welcome } from './functions.js';
12      import './functions.js';
13      // import 'functions.js'; // TypeError: Failed to
14      // resolve module specifier "functions.js".
15      // Relative references must start with either "/"
16      // ", "./", or "../".
17
18      console.log(import.meta.url);
19      console.log(document.body.childNodes);
20      welcome(); // Prints "Hello World!"
21      // a = 1; // ReferenceError: a is not defined
22    </script>
23    <script>
24      console.log("Ordinary script executed");
25      console.log(document.body.childNodes);
26      b = 2;
27    </script>
28    <script nomodule>
29      console.log("Modern browsers know both type=module
30      and nomodule, so skip this")
31      console.log("Old browsers ignore script with unknown
32      type=module, but execute this.");
33    </script>
34  </body>
35 </html>

```

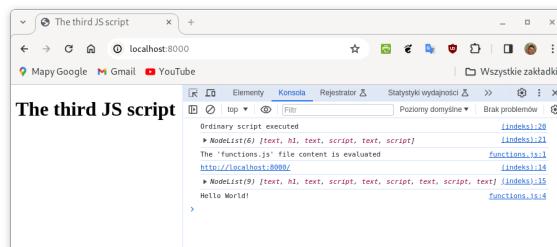
index.html

```

1 console.log("The 'functions.js' file content is evaluated");
2
3 export function welcome() {
4   console.log("Hello World!");
5 }

```

functions.js



## Notatki

## Definiowanie zmiennych

## Notatki

```

1 var x = 42;
2 //or
3 let x = 42;
4 //or
5 y = 42; // Not recommended
6 var _y = 42;
7 var $if = 42;
8 var r62a = 42;
9 var 1a = 42; // identifier starts immediately after numeric literal
10 console.log(y); // 42
11 console.log(Y); // Y is not defined
12 if = 42; //missing variable name
13 var y = 42
14 console.log(typeof(y)); // number
15 y = "42";
16 console.log(typeof(y)); // string

```

## Definiowanie stałych

Notatki

```

1 const PI = 3.1415926;
2 console.log(PI); // 3.1415926
3 console.log(typeof(PI)); // number
4 const PI = 3.14; // redeclaration of const PI
5 PI="3.24" // An attempt to overwrite a constant value
6 console.log(PI); // 3.1415926 that is, the attempt to overwrite failed
7 console.log(typeof(PI)); // number
8
9 var PI = 3.14 // redeclaration of const PI
10 var zmienna = 1;
11 const zmienna = 1; // redeclaration of var zmienna

```



## Typy specjalne

Notatki

### Typ „undefined”

```

1 console.log(typeof(abc)); // undefined
2
3 var def;
4 console.log(typeof(def)); // undefined
5
6 function f(arg){
7   console.log("arg="+arg)
8 }
9
10 var result = f() // arg=undefined
11 console.log(result) // undefined
12
13
14 if(def === undefined)
15   console.log("Undefined");
16 else
17   console.log("Defined");
18 // Undefined
19
20 if(def)
21   console.log("true");
22 else
23   console.log("false");
24 // false
25
26 console.log(def-1); // NaN

```



Materiały dla studentów Wydziału Informatyki AGH w Krakowie

## Typy proste i ich obiektowe odpowiedniki

### Notatki

---



---



---



---

Typ prosty	Obiektowy odpowiednik (Prototyp)
boolean	Boolean
number	Number
bignum	BigInt
string	String

```

1 console.log("2+2"); // "2 + 2" is of type (simple) 'string' => will write: 2 + 2
2 console.log("2+2".length); // Implicit conversion to the type (prototype) 'String' => will write: 3
3 /*
4 The above line is equivalent:
5   var objString = new String("2+2")
6   console.log(objString.length)
7 */
8 var str = "2+2";
9 console.log(str.charAt(1)); // +
10 console.log(str[1]); // +
11 console.log(str.charCodeAt(1)); // 43
12
13 var num = 1.987654
14 console.log(num.toPrecision(3)) //Implicit conversion to the type (prototype) 'Number' => will write:
   1.99

```



## Typ „boolean”

### Notatki

---



---



---



---

```

1 var dead = false;
2 var married = true;
3 console.log(typeof(dead)); // boolean
4 married = FALSE; //FALSE is not defined

```



## Typ „number”

## Notatki

```

1 var price = 10.5;
2 var num1 = 2;
3 var num2 = 2.0;
4 var binary = 0b101; //0B101;
5 var octal = 0o77; //0077
6 var hexadecimal = 0xFF; //0xFF
7
8 console.log(typeof(price)); // number
9 console.log(typeof(num1)); // number
10 console.log(typeof(num2)); // number
11 console.log(num2); // 2
12 console.log(typeof(binary)); // number
13 console.log(binary); // 5
14 console.log(typeof(octal)); // number
15 console.log(octal); // 63
16 console.log(typeof(hexadecimal)); // number
17 console.log(hexadecimal); // 255

```



## Konwersja do typu „number”

## Notatki

```

1 console.log(parseInt("3.14")); // 3
2 console.log(parseInt("3.94")); // 3
3 console.log(parseInt("3.94.1")); // 3
4 console.log(parseInt("3.94a")); // 3
5 console.log(parseInt("a3.94")); // NaN
6 console.log(parseFloat("3.14")); // 3.14
7 console.log(parseFloat("3.14.1")); // 3.14
8 console.log(parseFloat('0x10')); // 0
9 console.log(parseFloat('')); // NaN
10 console.log(parseFloat(' \r\n\t')); // NaN
11
12
13 console.log(parseInt("101",2)); // 5
14 console.log(parseInt("FF")); // NaN
15 console.log(parseInt("FF",16)); // 255
16
17 console.log(parseInt("FF - Firefox")); // NaN
18 console.log(parseInt("FF - Firefox",16)); // 255
19 console.log(parseInt("false")); // NaN
20 console.log(parseInt("false",16)); // 250 - "fa" has been changed to a number!
21
22 console.log(Number(null)); // 0
23 console.log(Number(undefined)); // NaN
24 console.log(Number(false)); // 0
25 console.log(Number(true)); // 1
26 console.log(Number("3.14")); // 3.14
27 console.log(Number("3.14.1")); // NaN
28 console.log(Number("3")); // 3
29 console.log(Number("3a")); // NaN
30 console.log(Number("0x10")); // 16
31 console.log(Number('')); // 0
32 console.log(Number(' \r\n\t')); // 0

```



**Typ „bigint”**

## Notatki

```

1 var x = Number.MAX_SAFE_INTEGER + 1;
2 var y = Number.MAX_SAFE_INTEGER + 2;
3 console.log(x == y); // true
4
5 console.log(BigInt(1)); // in
6 console.log(typeof(1n)); // bigint
7
8 // x = BigInt(Number.MAX_SAFE_INTEGER) + 1; // Error. Unable to perform operations on type 'number' and
9 // 'bigint',
10 x = BigInt(Number.MAX_SAFE_INTEGER) + 1n;
11 y = BigInt(Number.MAX_SAFE_INTEGER) + 2n;
12 console.log(x == y); //false

```

**Typ „string”**

## Notatki

```

1 var last_name = "Polak";
2 var first_name = 'Stanisław';
3
4 console.log(typeof(last_name)); // string
5 console.log(typeof(first_name)); // string
6
7 console.log(`First name=${first_name} Last name=${last_name}`); // First name=${first_name} Last name=${last_name}
8 console.log('First name=${first_name} Last=${last_name}'); // First name=${first_name} Last name=${last_name}
9
10 var a = 11 - "1" ;
11 console.log(a); // 10
12 var b = 11 + "1";
13 console.log(b); // 111
14
15 console.log(typeof(a)); // number
16 console.log(typeof(b)); // string
17
18 last_name[0]='W';
19 console.log(last_name); // "Polak" instead of "Wolak"
20
21 last_name = 'W' + last_name.substr(1);
22 console.log(last_name); // and now "Wolak"

```



**Typ „string”**

Szablony napisów

## Notatki

**Oznakowane (otagowane)****Nieoznakowane**

```

1 var a = 2;
2 var str = 'Variable `a` has value ${a}, 2+2=${2+2}\n';
3 console.log(str);
4 //*****
5 var str = 'Line 1
6 Line 2';
7 console.log(str);

```

```

1 function tag(strings, val1, val2){
2   result = "String 1:\t"+strings[0]+"\n";
3   result += "Raw string 1:\t"+strings.raw[0]+"\n";
4   result += "Value 1:\t"+val1+"\n";
5   result += "String 2:\t"+strings[1]+"\n";
6   result += "Raw string 2:\t"+strings.raw[1]+"\n";
7   result += "Value 2:\t"+val2+"\n";
8 }
9 /**
10 */
11 var a = 2;
12 var b = 3;
13 str = tag`a+b=\t${a+b}\n, a+b=${a+b}`;
14 console.log(str);

```

**Typ „string”**

Sekwencje specjalne

## Notatki

- ▶ \b
- ▶ \f
- ▶ \n
- ▶ \r
- ▶ \t
- ▶ \v
- ▶ \'
- ▶ \"
- ▶ \\
- ▶ \xXX
- ▶ \uXXXX

```

1 console.log("a`\"\\x63\\\'\\u0105") // a' "c' q

```

**Przykład użycia**

## Konwersja do typu „string”

### Notatki

```

1 var dead = true;
2 console.log(typeof(dead)); // boolean
3 var lancuch=dead.toString();
4 console.log(typeof(lancuch)); // string
5 console.log(lancuch); // true
6 var liczba = 0xFF;
7 console.log(0xFF.toString()); // "255"
8 liczba = 11;
9 console.log(liczba.toString()); // "11"
10 liczba = 11.9;
11 console.log(liczba.toString()); // "11.9"
12
13 var liczba=255;
14 console.log(liczba.toString(2)); // 11111111
15 console.log(liczba.toString(4)); // 3333
16 console.log(liczba.toString(8)); // 377
17 console.log(liczba.toString(16)); // ff
18
19 console.log(String(null)); // "null"
20 console.log(String(undefined)); // "undefined"
21 console.log(String(false)); // "false"
22 console.log(String(true)); // "true"
23 console.log(String(255)); // "255"
24 console.log(String(3.14)); // "3.14"

```



## Typ „symbol”

### Notatki

```

1 symbol1 = Symbol();
2 symbol2 = Symbol();
3 console.log(typeof(symbol1)); //symbol
4 console.log(symbol1 == symbol2) //false
5
6 symbol3 = Symbol('Symbol description');
7 symbol4 = Symbol('Symbol description');
8 console.log(symbol3); //Symbol(Symbol description)
9 console.log(symbol4); //Symbol(Symbol description)
10 console.log(symbol3 == symbol4); //false
11
12 symbol5 = Symbol.for("symbol3");
13 symbol6 = Symbol.for("symbol3");
14 console.log(symbol5); //Symbol(symbol3)
15 console.log(symbol6); //Symbol(symbol3)
16 console.log(symbol5 == symbol6); //true
17
18 var symbol7 = Symbol.for("uid");
19 console.log(Symbol.keyFor(symbol7)); // "uid"
20 var symbol8 = Symbol.for("uid");
21 console.log(Symbol.keyFor(symbo18)); // "uid"
22 var symbol9 = Symbol("uid");
23 console.log(Symbol.keyFor(symbo19)); //undefined

```



## Obiekty

Notatki

```

1 //Create an instance of the (built-in) type 'Object'
2 var object1 = new Object();
3 var object2 = {a:1, b:10};
4 console.log(typeof(object1)); // object
5 console.log(typeof(object2)); // object
6
7 //Access to object properties
8 console.log(object1.constructor); // function Object() { [native code] }
9 console.log(object1['constructor']); // function Object() { [native code] }
10 console.log(object2['constructor']); // function Object() { [native code] }
11 console.log(object2.a); // 1
12 console.log(object2['b']); // 10
13
14 const {a,b} = object2; // Object destructuring
15 console.log(a); // 1
16 console.log(b); // 10
17
18 //Use the symbol as the object's property
19 a = Symbol();
20 var object3 = {[a]:1, b:10}
21 console.log(object3[a]); // 1
22 console.log(object3.a); // undefined
23 console.log(object3['a']); // undefined
24 console.log(object3['b']); // 10
25 console.log(object3.b); // 10

```



## Hierarchia obiektów wbudowanych

Notatki



## Obiekt „Array”

Notatki

```

1 var tab1 = new Array(1,2,3); //equivalent to: var tab1=Array(1,2,3)
2 var tab2a = new Array(10); //equivalent to: var tab2a = []; tab2a.length=10;
3 var tab2b = new Array("10");
4 var tab3 = [4,'abc',6];
5 var a, rest;
6
7 console.log(tab1.length); // 3
8 console.log(tab2a.length); // 10
9 console.log(tab2b.length); // 1
10 console.log(tab3.length); // 3
11 console.log(tab1[0]); // 1
12 console.log(tab1[0]); //missing ) after argument list
13 console.log(tab2a[0]); // undefined
14 console.log(tab2b[0]); // 10
15 console.log(tab3[1]); // abc
16 console.log(tab3[5]); // undefined
17
18 [a, ...rest] = tab3; // Array destructuring
19 console.log(a); // 4
20 console.log(rest); // [ 'abc', 6 ]
21
22 var tab4 = new Array(new Array(1,null,undefined),new Array('A','B','C'),new Array('x','y','z'));
23 console.log(tab4.length); // 3
24 console.log(tab4[0]); // 1,
25 console.log(tab4[0][1]); // null
26 var tab1 = new Array(1,2,3);
27 var tab2 = [4,5,6];
28 console.log(tab1); // [1, 2, 3]
29
30 console.log(Array.from("JavaScript")); //['J','a','v','a','s','c','r','i','p','t']
31
32 var tab3 = tab1.concat(tab2);
33 console.log(tab3); // 1, 2, 3, 4, 5, 6
34
35 console.log(tab3.splice(1,2)); // 2, 3

```



## Obiekt „Map”

Notatki

```

1 var map = new Map();
2 emptyObject = {};
3 map.set("string","The value associated with the string");
4 map.set(1,{a:10});
5 map.set(emptyObject,1);
6
7 console.log(map); //Map { string: "The value associated with the string", 1: Object, Object: 1 }
8 console.log(map.get(1)); //Object { a: 10 }
9 console.log(map.get(2)); //undefined
10 console.log(map.get("string")); //The value associated with the string
11 console.log(map.get({})); //undefined
12 console.log(map.get(emptyObject)); //1
13 console.log(map.size); //3
14 map.delete("string");
15 console.log(map.size); //2
16
17 //Iteration of the hash
18 map.forEach((value,key,map) => {console.log("map["+key+"]="+value)});
19 /*
20 "map[1]=[object Object]"
21 "map[[object Object]]=1"
22 */
23 //Conversion of the array into hash
24 var tab = [["key1","String"], ["key2",5]];
25 map = new Map(tab);
26 console.log(map); //Map { key1: "String", key2: 5 }

```



## Obiekt „RegExp”

Notatki

```

1 function check(number) {
2   var re = /\d{7}/g; // 'g' - return all matching fragments, not just the first one
3   //lub
4   var re=new RegExp("\\d{7}","g");
5
6   if(re.test(number))
7     console.log("The correct phone number");
8   else
9     console.log("The telephone number should consist of seven digits");
10 }
11
12 var number1="1234567";
13 var number2="12-34";
14
15 check(number1); // The correct phone number
16 check(number2); // The telephone number should consist of seven digits

```

Sprawdzanie poprawności formatu numeru telefonu



## Obiekt „Function”

Notatki

```

1 var adder = new Function("a", "b", "return a + b");
2 var result = adder(1,2);
3 console.log(result); // 3
4 console.log(adder.length); // 2
5 var obj = {x: 1, y:2};
6 adder = new Function("message", "console.log(message+' ');return this.x + this.y");
7 console.log(adder.call(obj)); // undefined 3
8 console.log(adder.call(obj,'Value=3')) // Value=3

```



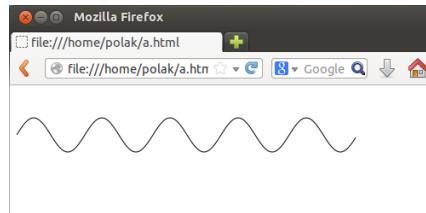
## Obiekt „Math”

```

1 ...
2 <canvas id="canvas" width="400" height="100">
3 Your browser does not support the "canvas" element
4 </canvas>
5
6 <script>
7 var canvas = document.getElementById("canvas");
8 if (canvas.getContext) {
9     var ctx = canvas.getContext('2d');
10    var ox = 0, oy = 50;
11    var t_min = 0, t_max = 10*Math.PI;
12    var scale = 20, step = 200, inc = t_max/step;
13
14    ctx.beginPath();
15    for (var t=t_min; t<=t_max; t+=inc){
16        y = scale * Math.sin(t);
17        x = (t / t_max) * canvas.width;
18        ctx.lineTo(ox+x, oy-y);
19    }
20    ctx.stroke();
21 </script>
22 ...

```

Rysowanie wykresu funkcji  $y = \sin(x)$  dla  $x \in [0, 10\pi]$



### Notatki

---



---



---

## Obiekt „Date”

```

1 function JSClock() {
2     var time = new Date()
3     var hour = time.getHours()
4     var minute = time.getMinutes()
5     var second = time.getSeconds()
6     var temp = "" + ((hour > 12) ? hour - 12 : hour)
7     if (hour == 0)
8         temp = "12";
9     temp += ((minute < 10) ? ":0" : ":") + minute
10    temp += ((second < 10) ? ":0" : ":") + second
11    temp += (hour >= 12) ? " P.M." : " A.M."
12    return temp
13 }
14 document.write(JSClock()); // 2:21:47 P.M.

```

Wypisanie aktualnego czasu w formacie 12-godzinnym



### Notatki

---



---



---

## Obiekt „navigator”

Notatki

```

1 //We assume that 'navigator.userAgent' contains the string "Mozilla/4.0 (compatible; MSIE 7.0; Windows
2   NT 5.1; .NET CLR 2.0.50727)"
3
4 if (/MSIE (\d+\.\d+)/.test(navigator.userAgent)){ //check if the browser is MSIE x.x;
5   var ieversion=new Number(RegExp.$1) // $1 contains the version number, here: 7.0
6   if (ieversion>=8)
7     document.write("You're using IE8 or above")
8   else if (ieversion>=7)
9     document.write("You're using IE7.x")
10  else if (ieversion>=6)
11    document.write("You're using IE6.x")
12  else if (ieversion>=5)
13    document.write("You're using IE5.x")
14 }
15 else
16   document.write("n/a")

```

Rozpoznawanie wersji przeglądarki IE



## Obiekt „window”

Metody prompt() oraz alert()

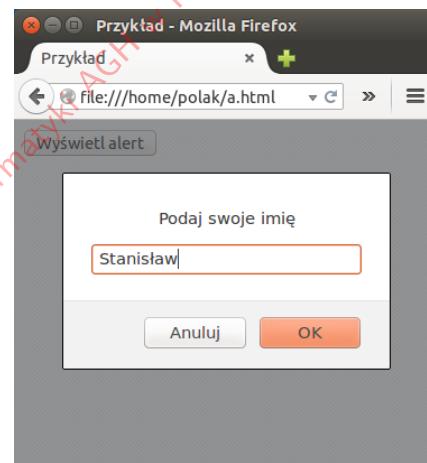
Notatki

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=UTF
4   -8">
5 <title>Przykład</title>
6 <script>
7   function load_and_display_the_name(){
8     var first_name>window.prompt('Podaj swoje imię','');
9     window.alert("Witaj "+first_name); //Display welcome text
10  }
11 </script>
12 </head>
13 <body>
14 <form>
15   <button type="button" onClick="load_and_display_the_name();">Wyświetl alert</button><br>
16 </form>
17 </body>
18 </html>

```

Otwieranie okna wprowadzania / wyprowadzania danych



## Obiekt „window”

Metody `setTimeout()` oraz `clearTimeout()`

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4 <title>An example</title>
5 <script>
6 function delayedAlert(time){
7   timeoutID = window.setTimeout(display, 2000);
8 }
9
10 function display(){
11   window.alert("Hello World!");
12   //timeoutID = window.setTimeout(display, 2000);
13 }
14
15 function stopExecution(){
16   window.clearTimeout(timeoutID);
17 }
18 </script>
19 </head>
20 <body>
21 <form>
22 <button type="button" onClick="delayedAlert();">Show alert</button><br>
23 <button type="button" onClick="stopExecution();">Cancel</button>
24 </form>
25 </body>
26 </html>
```

Po upływie 2 sekund wyświetla się okienko alertu

Notatki

---

---

---

---

---



## Obiekt „window”

Metody `setInterval()` oraz `The clearInterval()`

```
1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4 <title>An example</title>
5 <script>
6 function start_cyclic_execution() {
7   timeoutID = window.setInterval(display, 1000);
8 }
9
10 function display(){
11   console.log("Hello World!");
12 }
13
14 function stop_cyclic_execution(){
15   window.clearInterval(timeoutID);
16 }
17 </script>
18 </head>
19 <body>
20 <form>
21 <button type="button" onClick="start_cyclic_execution();">Start</button><br>
22 <button type="button" onClick="stop_cyclic_execution();">Stop</button>
23 </form>
24 </body>
25 </html>
```

Co sekundę wypisywany jest komunikat na konsoli

Notatki

---

---

---

---

---



## Obiekt „window”

Metody `requestAnimationFrame()` oraz `cancelAnimationFrame()`

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4 <title>An example</title>
5 <script>
6 function start_cyclic_execution() {
7   requestID=window.requestAnimationFrame(display);
8 }
9
10 function display(){
11   console.log("Hello World!");
12   requestID=window.requestAnimationFrame(display);
13 }
14
15 function stop_cyclic_execution(){
16   window.cancelAnimationFrame(requestID);
17 }
18 </script>
19 </head>
20 <body>
21 <form>
22 <button type="button" onClick="start_cyclic_execution()">Start</button><br>
23 <button type="button" onClick="stop_cyclic_execution()">Stop</button>
24 </form>
25 </body>
26 </html>
```

Cyklicznie, podczas odświeżania ekranu, wypisywany jest komunikat

Notatki

---



---



---



---



---



---



---



---



---



---



## Obiekt „form”

```

1 <html>
2 <head>
3 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4 <title> document.forms example</title>
5 <script>
6 function check() {
7   var form = document.forms[0];
8   //var form = document.forms.form1;
9   //var form = document.forms['form1'];
10  var element = form.elements[0];
11  //var element = form.elements.wiek;
12  //var element = form.elements['wiek'];
13  if (element.value == ""){
14    window.alert("Pole wiek musi zostać wypełnione");
15    return false;
16  }
17  else
18    return true;
19 }
20 </script>
21 </head>
22 <body>
23 <!-- <form ... onSubmit="return false;"> -->
24 <form id="form1" action="" onSubmit="return check();">
25   <input name='wiek' type='text'>
26   <input type='submit'>
27 </form>
28 </body>
29 </html>
```

Wstrzymanie wysyłki formularza

Notatki

---



---



---



---



---



---



---



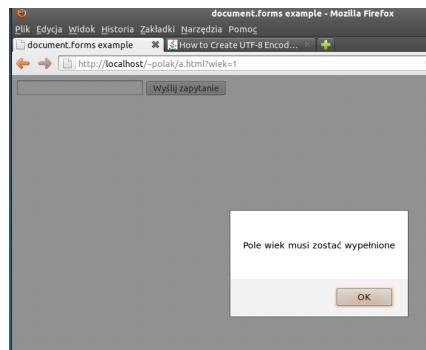
---



---



---



## Obiekt „Image”

Notatki

```

1 <html>
2 <head>
3 <script language = "JavaScript">
4
5 function preloader(){
6   bigImage = new Image();
7   bigImage.src = "bigImage.jpg";
8 }
9 </script>
10 </head>
11 <body onLoad="javascript:preloader()">
12 <a href="#" onMouseOver="javascript:document.images[0].src='bigImage.jpg'">
13 <!--
14 lub tak:
15 <a href="#" onMouseOver="javascript:document.img01.src='bigImage.jpg'">
16 -->
17 </a>
18 </body>
19 </html>
```

Wstępne ładowanie dużego obrazka



## Obiekt „location”

Notatki

```

1 <script>
2 if (window.location.protocol == "http:") {
3   var restUrl = window.location.href.substr(5);
4   location.href = "https:" + restUrl;
5 }
6 </script>
```

Przekierowanie do bezpiecznego protokołu HTTPS



## Obiekt „history”

### Notatki

---

---

---

---

---

---

```
1 history.back(); // equivalent to clicking the 'Back' button
2 history.go(-1); // equivalent to history.back();
```



## Obiekt „screen”

### Notatki

---

---

---

---

---

---

```
1 if (screen.pixelDepth < 8) {
2   // use of the "low-color" version of the page
3 } else {
4   // use of the "full-color" version of the page
5 }
```



## Funkcje

- ▶ Definiowanie (inaczej niż w C / Java) — za pomocą słowa kluczowego `function`
- ▶ Wywoływanie funkcji oraz zwracanie wartości przez funkcję — tak samo jak w C / Java
- ▶ Pozwalają definiować prototyp obiektów

```

1 function multiply(a, b=1){
2   var c = a * b;
3   a = 0;
4   b = 0;
5   return c;
6 }
7 function f() { return [1, 2, 3] }
8
9 a = 2;
10 b = 2;
11 var result = multiply(a,b);
12 console.log(result); // 4
13 var result = multiply(a);
14 console.log(result); // 2
15
16 var x, y, z;
17 [x, y, z] = f(); // Returning many values
18 [x, y, z] = (function f() { return [1, 2, 3] })(); // Simultaneous defining and calling functions
19 console.log(x); // 1
20 //*****
21 const constant = 1;
22 var variable = 2;
23 function constant(){} //Redeclaration of const constant
24 function variable(){}
25 variable(); //variable is not a function

```

### Przykładowe funkcje

```

1 function change(x,object1,object2){
2   x = 2;
3   object1.brand = "Fiat";
4   object2 = {brand: "Skoda"};
5 }
6
7 var car1 = {brand: "Ferrari"};
8 var car2 = {brand: "Ferrari"};
9 var variable = 1;
10 console.log(variable); // 1
11 console.log(car1.brand); // Ferrari
12 console.log(car2.brand); // Ferrari
13 change(variable,car1,car2);
14 console.log(variable); // 1
15 console.log(car1.brand); // Fiat
16 console.log(car2.brand); // Ferrari

```

### Przekazywanie typów prostych oraz złożonych

## Notatki

## Funkcje anonimowe

```

1 //Procedural function
2 function hello1(who) {
3   return 'Hello '+who;
4 }
5 //*****
6 console.log(hello1('world'));// "Hello world"
7 //*****
8 // Function as a variable
9 var hello2 = function (who) {return 'Hello '+who};
10 // or
11 var hello2 = (who) => {return 'Hello '+who};
12 // or
13 var hello2 = (who) => 'Hello '+who;
14 //*****
15 var hello3 = function() {
16   console.log('Hello');
17   console.log('World');
18 }
19 // or
20 var hello3 = () => {
21   console.log('Hello');
22   console.log('World');
23 }
24
25 console.log(hello2('world'));// "Hello world"
26 hello3(); // "Hello"
27           // "World"

```

Materiały dla studentów wydziału IT w Krakowie

```

1 function Person() {
2   // The Person () constructor defines 'this' as an
3   // instance of itself
4   this.age = 0;
5   this.salary = 0;
6
7   setInterval(function () {
8     //Here 'this' <=> object 'window' that is, it is
9     //different from 'this' defined in the Person
10    constructor
11    this.age++;
12    console.log("Age="+this.age);
13  }, 1000);
14
15  setInterval(() => {
16    this.salary++; //Here 'this' is a Person object
17    console.log("Salary="+this.salary);
18  }, 1000);
19
20  var person = new Person();

```

### Leksykalne this

## Notatki

## Domknięcia

```

1 function init() {
2   var name = "Polak";
3
4   function displayName() {
5     console.log(name);
6   }
7   displayName();
8 }
9 init(); // Polak

```

```

1 function createFunction() {
2   var name = "Polak";
3
4   function displayName() {
5     console.log(name);
6   }
7   return displayName;
8 }
9
10 var myFunction = createFunction();
11 myFunction(); // Polak

```

## Notatki

---



---



---



---

```

1 function multiply_by(x) {
2   return function(y) {
3     /* the function uses two variables:
4      y - available to the user
5      x - defined only inside the 'multiply_by()' function
6    */
7   return x * y;
8 };
9
10 var product_5_by = multiply_by(5); //the parameter 'x' is assigned the value 5
11
12 console.log(product_5_by(12)); // will be written 5 * 12 or 60

```

## Przykład użycia domknięcia



## Łańcuch zasięgu

```

1 function one(){
2   var a = 1;
3   two();
4
5   function two(){
6     var b = 2;
7     three();
8
9     function three() {
10       var c = 3;
11       console.log(a + b + c); //6
12     }
13   }
14 }
15 one();

```

## Notatki

---



---

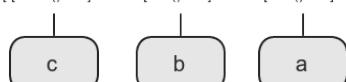


---



---

three()'s Scope Chain = [ [three() VO] + [two() VO] + [one() VO] + [Global VO] ];



## Zmienne w funkcji

```

1 function fun(x){
2   a = ++x;
3   b = 10;
4 }
5 //*****
6 a = 0; // <=> var a = 0;
7 console.log(a); // 0
8 console.log(b); //b is not defined
9 fun(a);
10 console.log(a); // 1
11 console.log(b); // 10

```

```

1 function fun(x){
2   var a = ++x;
3   var b = 10;
4 }
5 //*****
6 a = 0; // <=> var a = 0;
7 console.log(a); // 0
8 console.log(b); //b is not defined
9 fun(a);
10 console.log(a); // 0
11 console.log(b); //b is not defined

```

### Notatki

---



---



---



---

```

1 let a=1;
2 console.log(a); // 1
3 //*****
4 for (let i = 0; i<10; i++) {
5   console.log(i);
6   // 1, 2, 3, 4 ... 9
7 }
8 console.log(i); // i is not defined

```

Wyrażenie 'let'

```

1 function fun()
2 {
3   var a = 3;
4   var b = 4;
5   if (a === 3) {
6     let a = 10; // another variable 'a'. Range - interior of the 'if'
7     var b = 11; // the same variable 'b' as above. Range - interior
8     // of the 'fun' function
9     console.log(a); // 10
10    console.log(b); // 11
11  }
12  console.log(a); // 3
13  console.log(b); // 11
14 }
15 fun();

```

'let' kontra 'var'



## Funkcje ze zmienną liczbą argumentów

### Notatki

---



---



---



---

```

1 function write() {
2   // go after all arguments
3   for (var i=0; i<arguments.length; i++)
4     console.log(arguments[i]);
5 }
6
7 write("A","B","C");

```

```

1 function write(a, ...others) {
2   console.log(a);
3   console.log(Array.isArray(others));
4   for(counter in others)
5     console.log("others["+counter+"]="+others[
6       counter]);
7
8   console.log(arguments.length); //SyntaxError: '
9   arguments' object may not be used in
10  conjunction with a rest parameter
11
12 write("A","B","C");

```



## Obiekt typu Object

Definiowanie metod

Notatki

---



---



---



---



---



---



---



---



---



---



---



---

```

1 const person = {
2   firstname: 'John',
3   lastname: 'Doe',
4   printThis1: function () { console.log(this) },
5   printThis2: () => { console.log(this) },
6 }
7 console.group("printThis1()");
8 person.printThis1();
9 console.groupEnd();
10 console.group("printThis2()");
11 person.printThis2();
12 console.groupEnd();

```



## Metody call(), apply() oraz bind()

Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---

```

1 const person1 = {
2   firstname: 'John',
3   lastname: 'Doe',
4   printFullName: function (text) { console.log(`${text} ${this.firstname} ${this.lastname}`) }
5 }
6
7 const person2 = {
8   firstname: 'James',
9   lastname: 'Bond',
10 }
11 person1.printFullName('My name is'); // My name is John Doe
12
13 person1.printFullName.call(person2, 'My name is Bond,'); // My name is Bond, James Bond
14
15 const args = ['My name is Bond,'];
16 person1.printFullName.apply(person2, args); // My name is Bond, James Bond
17
18 const myNameIs = person1.printFullName.bind(person2, 'My name is Bond,');
19 myNameIs(); // My name is Bond, James Bond

```



## Definiowanie klas

### Składnia JavaScript

```

1 function Human(age, name) {
2   this.age = age;
3   this.name = name;
4 }
5 Human.prototype.getInfo = function () { return `name=${this.name} age=${this.age}`; }
6
7 var john = new Human(10, 'John');
8 console.log(john.getInfo()); // name=John age=10
9
10 /*****
11 function Infant(name) {
12   Human.call(this, 0, name);
13 }
14 Infant.prototype = Object.create(Human.prototype);
15
16 var anna = new Infant('Anna');
17 console.log(anna.getInfo()); // name=Anna age=0

```

### Składnia ES6

```

1 class Human {
2   constructor(age, name) {
3     this.age = age;
4     this.name = name;
5   }
6   getInfo() {
7     return `name=${this.name} age=${this.age}`;
8   }
9 }
10
11 var john = new Human(10, 'John');
12 console.log(john.getInfo()); // name=John age=10
13
14 class Infant extends Human {
15   constructor(name) {
16     super(0, name)
17   }
18 }
19
20 var anna = new Infant('Anna');
21 console.log(anna.getInfo()); // name=Anna age=0

```

### Notatki

---



---



---



---



---



---

## Tryb ścisły

- ▶ Eliminuje niektóre „pułapki” starszych wersji języka
- ▶ Naprawia błędy, które utrudniały silnikom JavaScriptu dokonywanie optymalizacji
- ▶ Uniemożliwia stosowanie składni, która zostanie wykorzystana w kolejnych wersjach ECMAScriptu

```

1 "use strict";
2 var v = "Cześć! Jestem skryptem trybu ścisłego!";

```

Włączenie trybu ścisłego dla całego skryptu

### Notatki

---



---



---



---



---



---

```

1 function strict() {
2   // tryb ścisły na poziomie funkcji
3   'use strict';
4   function nested() { return "Ja też"; }
5   return "Cześć! Jestem funkcją trybu ścisłego! " + nested();
6 }
7 function notStrict() { return "A ja nie.."; }
8

```

Włączenie trybu ścisłego dla funkcji

## Tryb ścisły

Przykłady działania

Notatki

---



---



---



---



---



---



---



---



---



---

```

1 "use strict";
2 var v = "Cześć! Jestem skryptem trybu ścisłego!";
3 zmienią = 17; // wyrzuca wyjątek ReferenceError
4
5 delete Object.prototype; // wyrzuca wyjątek TypeError
6
7 var x = 17;
8 with (obj) // !!! błąd składni
9 {
  // Gdyby to nie był tryb ścisły, czy byłoby to var x,
11 // czy może jednak obj.x? W ogólnym przypadku
12 // nie można tego określić przed wykonaniem kodu,
13 // co uniemożliwia optymalizację.
14 x;
15 }

```

## Magazyny danych

### Typy

- ▶ Magazyn lokalny — obiekt 'localStorage'
- ▶ Magazyn sesji — obiekt 'sessionStorage'

### Ogólna charakterystyka

- ▶ Są dostępne tylko z poziomu klienta
- ▶ Dane są dostępne dla stron, które mają to samo pochodzenie co zapisane informacje
- ▶ Pary postaci klucz-wartość
- ▶ Maksymalny rozmiar przechowywanych tam danych jest znacznie większy niż w przypadku ciasteczek



Notatki

---



---



---



---



---



---



---



---



---



---

### Strony traktowane jako różne

- ▶ <http://www.agh.edu.pl>
- ▶ <https://www.agh.edu.pl> — inny protokół
- ▶ <http://www.agh.edu.pl:8080> — inny numer portu. Domyślny port HTTP to 80
- ▶ <http://poczta.agh.edu.pl> — inny host



## Magazyny danych

Przykład użycia

```

1 if(typeof(Storage) !== "undefined") {
2   localStorage.setItem("nazwisko", "Polak");
3   //lub
4   localStorage.nazwisko = "Polak";
5   //lub
6   localStorage['nazwisko'] = "Polak";
7   var nazwisko = localStorage.getItem("nazwisko");
8   console.log(nazwisko);
9   localStorage.removeItem("nazwisko");
10  //lub
11  delete sessionStorage.nazwisko;
12  var nazwisko = sessionStorage.getItem("nazwisko");
13  console.log(nazwisko);
14  localStorage.setItem("imie", "Stanisław");
15  /*************************************************************************/
16  sessionStorage.setItem("nazwisko", "Polak");
17  //lub
18  sessionStorage.nazwisko = "Polak";
19  //lub
20  sessionStorage['nazwisko'] = "Polak";
21  var nazwisko = sessionStorage.getItem("nazwisko");
22  console.log(nazwisko);
23  sessionStorage.removeItem("nazwisko");
24  //lub
25  delete sessionStorage.nazwisko;
26  var nazwisko = sessionStorage.getItem("nazwisko");
27  console.log(nazwisko);
28  sessionStorage.setItem("imie", "Stanisław");
29 }
30 else
31 alert("Twoja przeglądarka nie obsługuje magazynów");

```

skrypt1.html

Notatki

---



---



---

```

1 <script>
2 if(typeof(Storage) !== "undefined") {
3   var imię = localStorage.getItem("imie");
4   console.log(imię);
5   var imię = sessionStorage.getItem("imie");
6   console.log(imię);
7 }
8 else
9   alert("Twoja przeglądarka nie obsługuje magazynów");
10 </script>

```

skrypt2.html



## Magazyny danych

Przechowywanie danych niebędących napisami

```

1 var liczba = 1;
2 tablica = [1,2,3];
3 obiekt = {a:10,b:20};
4
5 //Wstawianie danej typu prostego
6 localStorage.setItem("liczba", liczba);
7 liczba = Number(localStorage.getItem("liczba"));
8 console.log(liczba);
9
10 //Wstawianie danych typu złożonego
11 localStorage.setItem("tablica",JSON.stringify(tablica));
12 tablica = JSON.parse(localStorage.getItem("tablica"));
13 console.log(tablica);
14
15 localStorage.setItem("obiekt",JSON.stringify(obiekt));
16 obiekt = JSON.parse(localStorage.getItem("obiekt"));
17 console.log(obiekt);

```

Notatki

---



---



---

Na wyjściu

```

1
Array [ 1, 2, 3 ]
Object a: 10, b: 20

```



## Iteratory

```

1 function IterujPrzez(...args) {
2   let index = 0;
3   let iterable = {
4     [Symbol.iterator]() {//Protokół 'iterable'
5       return this;
6     },
7     next() { //Protokół 'iterator'
8       if (index < args.length) {
9         return {
10           value: args[index++], // wystąpienie wymagane
11           done: false // wystąpienie niewymagane
12         };
13       } else {
14         return {
15           value: undefined, // wystąpienie niewymagane
16           done: true // wystąpienie wymagane
17         };
18       }
19     } //next()
20   };
21   return iterable;
22 }
23 var iter = IterujPrzez('Stanisław', 'Polak');
24 console.log(iter.next());
25 console.log(iter.next());
26 console.log(iter.next());
27 for(var element of IterujPrzez('Stanisław', 'Polak'))
28   console.log(element);

```

### Na wyjściu

```

Object { value: "Stanisław", done: false
Object { value: "Polak", done: false
Object { value: undefined, done: true
"Stanisław"
"Polak"

```

### Notatki

## Generatory

### Pojedynczy generator

```

1 function* generatorWartości(){
2   yield "Jeden";
3   yield "Dwa";
4   for (var i = 0; i < 2; i++)
5     yield i;
6 }
7 //*****
8 //Utworzenie obiektu typu "Generator"
9 var gen = generatorWartości();
10 //*****
11 console.log(gen.next());
12 console.log(gen.next());
13 console.log(gen.next());
14 console.log(gen.next());
15 console.log(gen.next());

```

### Wiele generatorów

```

1 function* następnyGeneratorWartości(i){
2   yield i + 1;
3   yield i + 2;
4 }
5
6 function* generatorWartości(i){
7   yield i;
8   yield* następnyGeneratorWartości(i);
9   yield i + 1;
10 }
11
12 var gen = generatorWartości(1);
13 console.log(gen.next());
14 console.log(gen.next());
15 console.log(gen.next());
16 console.log(gen.next());
17 console.log(gen.next());

```

### Notatki

### Na wyjściu

```

Object { value: "Jeden", done: false
Object { value: "Dwa", done: false
Object { value: 0, done: false
Object { value: 1, done: false
Object { value: undefined, done: true

```

### Na wyjściu

```

Object { value: 1, done: false
Object { value: 2, done: false
Object { value: 3, done: false
Object { value: 2, done: false
Object { value: undefined, done: true

```



## Wyrażenia tablicowe

Notatki

```

1 var litery = ["A", "B", "C"];
2 var liczby = [1, 2, 3];
3 ****
4 var kwadraty = [for (liczba of liczby) liczba*liczba];
5 console.log(kwadraty); //Wypisze: Array [ 1, 4, 9 ]
6 ****
7 var kwadratyParzystych = [for (liczba of liczby) if(liczba%2 == 0) liczba*liczba ];
8 console.log(kwadratyParzystych); //Wypisze: Array [ 4 ]
9 ****
10 var miks1 = [for (liczba of liczby) for (litera of litery) liczba+litera];
11 console.log(miks1); //Wypisze: Array [ "1A", "1B", "1C", "2A", "2B", "2C", "3A", "3B", "3C" ]
12 ****
13 var miks2 = [for (liczba of liczby) if(liczba > 1) [for (litera of litery) liczba+litera]];
14 console.log(miks2); //Wypisze: Array [ Array[3], Array[3] ]
15 console.log(miks2[0]); //Wypisze: Array [ "2A", "2B", "2C" ]
16 console.log(miks2[1]); //Wypisze: Array [ "3A", "3B", "3C" ]

```

## Wyrażenia generujące

Notatki

```

1 var litery = ["A", "B", "C"];
2 var liczby = [1, 2, 3];
3 ****
4 var generator = (for (liczba of liczby) liczba*liczba);
5 console.log(generator); //Wypisze: Generator {}
6 console.log(generator.next()); //Wypisze: Object {value: 1, done: false}
7 console.log(generator.next()); //Wypisze: Object {value: 4, done: false}
8 console.log(generator.next()); //Wypisze: Object {value: 9, done: false}
9 console.log(generator.next()); //Wypisze: Object {value: undefined, done: true}
10 var generator = (for (liczba of liczby) liczba*liczba);
11 for(var elem of generator)
12   console.log(elem); /* Wypisze:
13   1
14   4
15   9
16 */
17 ****
18 generator=(for (liczba of liczby) if(liczba > 1) [for (litera of litery) liczba+litera]);
19 console.log(generator.next()); //Wypisze: Object {value: Array[3], done: false}
20 console.log(generator.next()); //Wypisze: Object {value: Array[3], done: false}
21 console.log(generator.next()); //Wypisze: Object {value: undefined, done: true}

```

## Pośrednicy

Notatki

```

1 var proceduraObsługi = {
2   deleteProperty(obiektDocelowy, właściwość){
3     console.log("Kasowanie właściwości '" + właściwość + "'");
4     return delete obiektDocelowy[właściwość]; //Zwraca: true lub false
5   //lub
6   //return Reflect.deleteProperty(obiektDocelowy, właściwość);
7 }
8 }
9 var p = new Proxy({}, proceduraObsługi);
10 p.a = 1;
11 delete p.a; //Wypisze: "Kasowanie właściwości 'a'"
12 console.log(p.a); //Wypisze: undefined

```



## Pośrednicy

Przykład zastosowania

Notatki

```

1 var validator = {
2   set(obiekt, właściwość, wartość) {
3     if (właściwość === 'ilość') {
4       if (!Number.isInteger(wartość)) {
5         throw new TypeError('Ilość nie jest liczbą');
6       }
7       if (wartość < 0) {
8         throw new RangeError('Ilość nie może być liczbą ujemną');
9       }
10    }
11
12    // Domyślne zachowanie - zapisanie wartości
13    obiekt[właściwość] = wartość;
14    return true; //true oznacza, że udało się przypisać wartość
15  };
16};
17
18 var produkt = new Proxy({}, validator);
19 produkt.ilosc = 10;
20 console.log(produkt.ilosc); //Wypisze: 10
21 produkt.ilosc = 'dziesięć'; //TypeError: Ilość nie jest liczbą
22 produkt.ilosc = -2; //RangeError: Ilość nie może być liczbą ujemną

```



## Wątki robocze

### Notatki

---



---



---



---



---

- ▶ Umożliwiają uruchamianie długotrwałych części skryptów JavaScript w osobnym wątku
- ▶ Mają dostęp do:
  - ▶ Obiekt 'navigator'
  - ▶ Obiekt 'location' w trybie „tylko do odczytu”
  - ▶ Obiekt 'XMLHttpRequest'
  - ▶ Metody setTimeout() / clearTimeout() oraz setInterval() / clearInterval()
  - ▶ Cache'owanie aplikacji — Application Cache API
  - ▶ Importowanie zewnętrznych skryptów za pomocą importScript()
  - ▶ Tworzenie innych wątków potomnych
- ▶ Nie mają dostępu do:
  - ▶ Obiekty DOM
  - ▶ Obiekt 'window'
  - ▶ Obiekt 'document'
  - ▶ Obiekt 'parent'

## Wątki robocze

### Przykład



```

1 var worker = new Worker('worker.js');
2 worker.onmessage = function (e) {
3   console.log('Otrzymano odpowiedź od wątku
4   roboczego:\n' + e.data);
5 };
6 worker.postMessage('Witaj wątku roboczy');
7 ...
8 worker.terminate();

```

master.js

```

1 i = 0;
2 onmessage = function(e) {
3   i++;
4   odpowiedź = '\ti=' + i + ', otrzymano odpowiedź od wątku
5   głównego: ' + e.data;
6   postMessage(odpowiedź);
7   if(i==3)
8     close();
};

```

worker.js

### Notatki

---



---



---



---



---

### Na wyjściu

Otrzymano odpowiedź od wątku roboczego:

i=1, otrzymano odpowiedź od wątku głównego: Witaj wątku roboczy

Otrzymano odpowiedź od wątku roboczego:

i=2, otrzymano odpowiedź od wątku głównego: Witaj wątku roboczy



## Document Object Model

### Ogólna charakterystyka

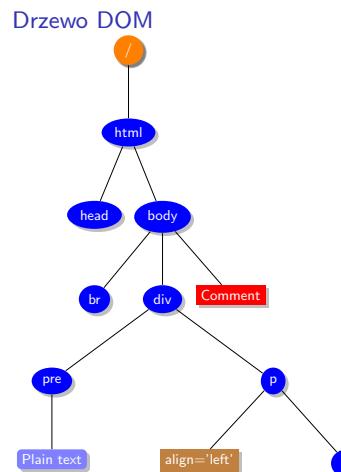
- ▶ Document Object Model (DOM) — obiektowy model dokumentu
- ▶ Dokument — drzewo obiektów
- ▶ Interfejs programowy (API) dla dokumentów HTML i XML
- ▶ Zbiór właściwości i metod do manipulacji w/w dokumentami

```

1 <!DOCTYPE html>
2 <html>
3   <head></head>
4   <body>
5     <br>
6     <div>
7       <pre>Plain text</pre>
8       <p align='left'>
9         <hr/>
10      </p>
11    </div>
12    <!-- Comment -->
13  </body>
14 </html>

```

Dokument HTML



### Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---

## Podstawowe właściwości węzłów

### Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---



---



---



---

Typ węzła	nodeType	nodeName	nodeValue
Element	Node.ELEMENT_NODE (1)	Nazwa znacznika dużymi literami	null
Atrybut	Node.ATTRIBUTE_NODE (2)	Nazwa atrybutu	Wartość atrybutu
Text	Node.TEXT_NODE (3)	#text	Tekst stanowiący jego treść
Komentarz	Node.COMMENT_NODE (8)	#comment	Treść komentarza
Dokument	Node.DOCUMENT_NODE (9)	#document	null

## Wyświetlenie informacji o pojedynczym węźle element

Metody getElementById() / querySelector()

```

1 <html>
2   <head>
3     <title>An example</title>
4     <script>
5       function start(){
6         var elem1 = document.getElementById("elem1");
7         # or
8         var elem1 = document.querySelector("#elem1");
9
10        console.log(elem1);           // [object HTMLBodyElement]
11
12        console.log(elem1.nodeType); // 1
13        console.log(elem1.nodeName); // BODY
14        console.log(elem1.nodeValue); // null
15      }
16    </script>
17  </head>
18  <body id="elem1" onLoad="start();">
19  </body>
20 </html>

```

Notatki

---

---

---

---

---

---



## Wyświetlenie informacji o kilku węzłach element

Metody getElementsByTagName() / querySelectorAll()

```

1 <html>
2   <head>
3     <title>An example</title>
4     <script>
5       function start(){
6         var tdElements = document.getElementsByTagName("td");
7         # or
8         var tdElements = document.querySelectorAll("tr td");
9         console.log(tdElements);          // [object HTMLCollection]
10        console.log(tdElements.length);   // 4
11        console.log(tdElements[0]);      // [object HTMLTableCellElement]
12        console.log(tdElements[0].nodeType); // 1
13        console.log(tdElements[0].nodeName); // TD
14        console.log(tdElements[0].nodeValue); // null
15        console.log(tdElements[1]);      // [object HTMLTableCellElement]
16        console.log(tdElements[1].nodeType); // 1
17        console.log(tdElements[1].nodeName); // TD
18        console.log(tdElements[1].nodeValue); // null
19      }
20    </script>
21  </head>
22  <body onLoad="start();">
23    <table>
24      <tr><td>a</td><td>b</td></tr>
25      <tr><td>c</td><td>d</td></tr>
26    </table>
27  </body>
28 </html>

```

Notatki

---

---

---

---

---

---



## Pobieranie potomków węzłów element oraz tekst

Własność `childNodes`

Notatki

---



---



---



---



---



---



---



---



---



---

```

1 <html>
2   <head>
3     <title>An example</title>
4     <script>
5       function start(){
6         var trElements = document.getElementsByTagName("tr");
7         for(let i=0 ; i<trElements.length ; i++){
8           var elements = trElements[i].childNodes; // the type of the 'tdElements' object is "NodeList"
9           for (let j=0;j<elements.length;j++){
10             if (elements[j].nodeType == Node.ELEMENT_NODE) {
11               var string = elements[j].nodeName + ":" + elements[j].childNodes[0].nodeValue;
12               console.log(string);
13             }
14           }
15         }
16       }
17     /* Output:
18      TD: a
19      TD: b
20      TD: c
21      TD: d
22    */
23     </script>
24   </head>
25   <body onLoad="start();">
26     <table border="1">
27       <tr><td>a</td><td>b</td></tr>
28       <tr><td>c</td><td>d</td></tr>
29     </table>
30   </body>
31 </html>
```



## Obsługa atrybutów

Własność `attributes` oraz metody `setAttribute()` i `removeAttribute()`

Notatki

---



---



---



---



---



---



---



---



---



---

```

1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
4     <title>An examples</title>
5     <script>
6       function change(thickness){
7         var elem1 = document.getElementById("elem1"); // "HTMLTableElement" type object
8         console.log(elem1.getAttribute('border'));// 1
9         console.log(elem1.getAttribute('id'));// elem1
10        elem1.setAttribute('border',thickness);
11
12        //You can do it anyway
13        var attributes = elem1.attributes; //obiekt typu "NamedNodeMap" type object
14        console.log(attributes.border.value); // 1
15        console.log(attributes.id.value); // elem1
16        attributes.border.value = thickness;
17      }
18
19      function delete(){
20        var elem1 = document.getElementById("elem1");
21        elem1.removeAttribute('border');
22      }
23
24    </script>
25  </head>
26  <body>
27    <table border="1" id="elem1">
28      <tr><td>a</td><td>b</td></tr>
29      <tr><td>c</td><td>d</td></tr>
30    </table>
31    <form>
32      <input type="button" value="Change the thickness" onClick="change(2);">
33      <input type="button" value="Delete" onClick="delete();">
34    </form>
35  </body>
36 </html>
```



## Wstawienie nowej komórki tabeli na końcu wiersza

Metody createElement(), createTextNode() oraz appendChild()

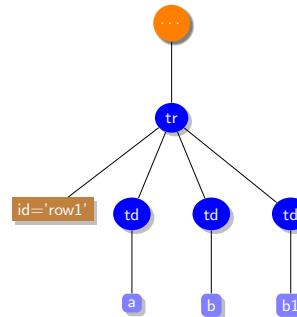
```

1 <html>
2   <head>
3     <title>An example</title>
4     <script>
5       function insert(){
6         var newTD = document.createElement("td");
7         var newTextNode = document.createTextNode("b1");
8         newTD.appendChild(newTextNode);
9         // newTD.textContent="b1"; // Instead of the above two lines
10        var row1 = document.getElementById("row1");
11        row1.appendChild(newTD);
12      }
13    </script>
14  </head>
15  <body>
16    <table border="1">
17      <tr id="row1"><td>a</td><td>b</td></tr>
18      <tr><td>c</td><td>d</td></tr>
19    </table>
20    <form>
21      <input type="button" value="Insert" onClick="insert();">
22    </form>
23  </body>
24</html>

```

a	b
c	d

a	b	b1
c	d	



Notatki



## Wstawienie nowej komórki tabeli na początku wiersza

Metoda insertBefore()

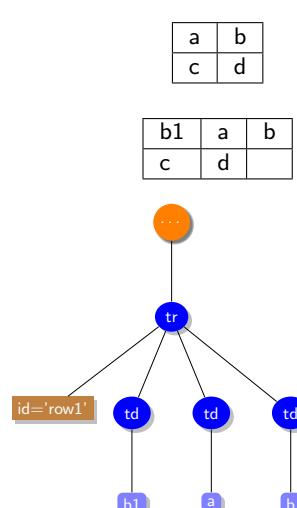
```

1 <html>
2   <head>
3     <title>An example</title>
4     <script>
5       function insert(){
6         var newTD = document.createElement("td");
7         var newTextNode = document.createTextNode("b1");
8         newTD.appendChild(newTextNode);
9         var row1 = document.getElementById('row1');
10        var cell1 = row1.getElementsByTagName("td").item(0);
11        row1.insertBefore(newTD,cell1);
12      }
13    </script>
14  </head>
15  <body>
16    <table border="1">
17      <tr id="row1"><td>a</td><td>b</td></tr>
18      <tr><td>c</td><td>d</td></tr>
19    </table>
20    <form>
21      <input type="button" value="Insert" onClick="insert();">
22    </form>
23  </body>
24</html>

```

a	b
c	d

b1	a	b
c	d	



Przykłady



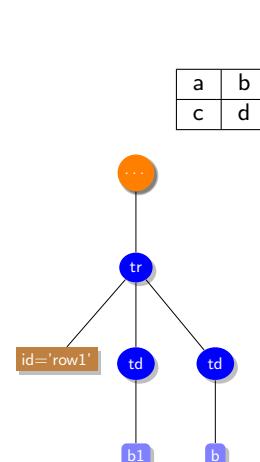
Notatki

## Zastąpienie komórki tabeli

Metoda `replaceChild()`

```

1 <html>
2   <meta http-equiv="Content-Type" content="text/html; charset=
3     UTF-8">
4   <head>
5     <title>An example</title>
6     <script>
7       function replace(){
8         var newTD = document.createElement("td");
9         var newTextNode = document.createTextNode("b1");
10        newTD.appendChild(newTextNode);
11        var row1 = document.getElementById('row1');
12        var cell1 = element.getElementsByTagName("td").item(0);
13        row1.replaceChild(newTD,cell1);
14      }
15    </script>
16  </head>
17  <body>
18    <table border="1">
19      <tr id='row1'><td>a</td><td>b</td></tr>
20      <tr><td>c</td><td>d</td></tr>
21    </table>
22    <form>
23      <input type="button" value="Replace" onClick="replace();"
24    </form>
25  </body>
</html>
```



Notatki

---



---



---



---



---

## Dostęp do stylów CSS

Obiekt `style`

- ▶ Cechy CSS → własność obiektu `style`
  - ▶ `background-color` → `style.backgroundColor`
  - ▶ `border-top-width` → `style.borderTopWidth`

a	b
c	d

```

1 <html>
2   <head>
3     <meta http-equiv="Content-Type" content="text/
4       html; charset=UTF-8" />
5     <title>An example</title>
6     <script>
7       function color(value){
8         var cell1 = document.getElementById("cell1");
9         cell1.style.backgroundColor = value;
10      }
11    </script>
12  </head>
13  <body>
14    <table border="1">
15      <tr><td id="cell1">a</td><td>b</td></tr>
16      <tr><td>c</td><td>d</td></tr>
17    </table>
18    <form>
19      <input type="button" value="Red" onClick="
20        color('#FF0000');">
21      <input type="button" value="Green" onClick="
22        color('#00FF00');">
23    </form>
24  </body>
</html>
```



Notatki

---



---



---



---



---

Materiały dla studentów Wydziału Informatycznego Uniwersytetu w Krakowie

## Obsługa zdarzeń

Zmiana koloru tła po kliknięciu komórki tabeli

```

1 <!DOCTYPE html>
2 <meta charset="UTF-8">
3 ...
4   <script>
5     function changeColor(){
6       var cell = document.getElementById("cell");
7       cell.style.backgroundColor='red';
8     }
9   ****
10    function displayAlert(){
11      alert("displayAlert()");
12    }
13  ****
14  function load() {
15   var table = document.getElementById("table");
16   table.addEventListener("click", displayAlert, false); //First, 'changeColor()' will be executed,
17   followed by 'displayAlert ()'. If the third parameter is 'true' then 'displayAlert()' will be
18   executed first, then 'changeColor ()'
19   var cell = document.getElementById("cell");
20   cell.addEventListener("click", changeColor, false);
21 }
22 ...
23 <body onload="load();">
24   <table id="table" border="1">
25     <tr><td id="cell">One</td></tr>
26     <tr><td>Two</td></tr>
27   </table>
28 </body>
29 </html>

```

Notatki



## Własne zdarzenia

```

1 ...
2 <script>
3   function load(){
4     const form = document.querySelector('form');
5
6     // Listen for the 'userLogin' event
7     form.addEventListener("userLogin", function(e) {
8       console.info("Event is: ", e);
9       console.info("Custom data is: ", e.detail);
10    });
11
12   const button = document.getElementById("button");
13   button.addEventListener('click', function() {
14
15     // Create the 'userLogin' event
16     const myEvent = new Event('userLogin');
17     const myEvent = new CustomEvent("userLogin", {
18       detail: {
19         username: document.getElementById("login").value
20       }
21     });
22
23     // Trigger it!
24     form.dispatchEvent(myEvent);
25   });
26 }
27 </script>

```

Notatki

```

28 ...
29 <body onLoad="load()">
30   <form>
31     <input
32       id="login"
33       type="text">
34     <button
35       id="button"
36       type="button">Log in</button>
37   </form>
38 </body>
39 ...

```

## Ogólna charakterystyka

Notatki

- ▶ Biblioteka do tworzenia interfejsów graficznych
- ▶ Cechy charakterystyczne:
  - ▶ Wirtualny DOM
  - ▶ JSX



## Przykład „Hello World” 1

Komponent funkcyjny o nazwie 'Hello'

Notatki

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <script src="https://unpkg.com/react@latest/umd/react.development.js" crossorigin></script>
5     <script src="https://unpkg.com/react-dom@latest/umd/react-dom.development.js" crossorigin></script>
6     <script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
7   </head>
8   <body>
9     <div id="root"></div>
10    <script type="text/babel">
11      // Create a component named 'Hello'
12      function Hello() {
13        return <h1>Hello World!</h1>; // => React.createElement('h1', null, 'Hello World!');
14      }
15      const container = document.getElementById('root');
16      const root = ReactDOM.createRoot(container);
17      root.render(<Hello />);
18    </script>
19   </body>
20 </html>

```

index.html



## Przykład „Hello World” 1

Komponent klasowy o nazwie 'Hello'

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <script src="https://unpkg.com/react@latest/umd/react.development.js" crossorigin></script>
5     <script src="https://unpkg.com/react-dom@latest/umd/react-dom.development.js" crossorigin></script>
6     <script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>
7   </head>
8   <body>
9     <div id="root"></div>
10    <script type="text/babel">
11      // Create a component named 'Hello',
12      class Hello extends React.Component {
13        render() {
14          return <h1>Hello World!</h1>;
15        }
16      }
17      const container = document.getElementById('root');
18      const root = ReactDOM.createRoot(container);
19      root.render(<Hello />);
20    </script>
21  </body>
22</html>

```

index.html

Notatki



## Przykład „Hello World” 2

Notatki

```

1 $ npx create-next-app@latest --javascript my-app
2 Would you like to use ESLint? ... No / Yes
3 Would you like to use Tailwind CSS? ... No / Yes
4 Would you like to use 'src/' directory? ... No /
  Yes
5 Would you like to use App Router? (recommended) ...
  No / Yes
6 Would you like to customize the default import
  alias (@/*)? ... No / Yes
7 ...
8 $ cd my-app
9 $ npm run dev

```

```

1 import Hello from "./hello";
2
3 export default function Home() {
4   return (
5     <main className="flex min-h-screen flex-col items-center justify-between p-24">
6       <Hello />
7     </main>
8   );
9 }

```

src/app/page.js

```

1 'use client'
2
3 import React from "react";
4
5 export default class Hello extends React.Component {
6   render() {
7     return <h1>Hello World!</h1>;
8   }
9 }

```

src/app/hello.js



## JSX

Notatki

## Przykład 2

## Przykład 1

```

1 class Hello extends React.Component {
2   render() {
3     return <div>Hello World 1+2={1+2}</div>;
4   }
5 }
```

## Wygenerowany HTML

```
<div>Hello World 1+2=3</div>
```

```

1 class Hello extends React.Component {
2   render() {
3     const elements = ["Element 1", "Element 2"];
4     const multiLine = (<ul className='name'>
5       <li>{elements[0]}</li>
6       <li>{elements[1]}</li>
7     </ul>);
8     return multiLine;
9   }
10 }
```

## Wygenerowany HTML

```
<ul className='name'>
  <li>Element 1</li>
  <li>Element 2</li>
</ul>
```

## Cykl życia komponentu

Notatki

```

1 class Hello extends React.Component {
2
3   constructor(props) {
4     super(props);
5     console.log('constructor()');
6   }
7
8   componentDidMount() {
9     console.log('componentDidMount()');
10 }
11
12   componentWillUnmount() {
13     console.log('componentWillUnmount()');
14   }
15
16   render() {
17     console.log('render()');
18
19     return <div>Hello World</div>;
20   }
21 }
22
23 const container = document.getElementById('root');
24 const root = ReactDOM.createRoot(container);
25 root.render(<Hello />);
```

## Konsola

```
constructor()
render()
componentDidMount()
componentWillUnmount()
```

Pełny wykaz metod można znaleźć na stronie <https://react.dev/reference/react/Component>



## Przekazywanie danych do komponentu

Notatki

```

1 class Hello extends React.Component {
2   render() {
3     return <h1>{this.props.welcome}</h1>;
4   }
5 }
6
7 const container = document.getElementById('root');
8 const root = ReactDOM.createRoot(container);
9 root.render(<Hello welcome="Hello World"/>);

```

### Wygenerowany HTML

<h1>Hello World</h1>



## Renderowanie warunkowe

Notatki

```

1 class Even extends React.Component {
2   render() {
3     return <div>The number is even</div>
4   }
5 }
6
7 class Odd extends React.Component {
8   render() {
9     return <div>The number is odd</div>
10  }
11 }
12 ...
13 const container = document.getElementById('root');
14 const root = ReactDOM.createRoot(container);
15 root.render(<Number value={Math.floor(Math.random() * 10)} />);

```

```

1 function Number(props) {
2   if (props.value % 2 == 0)
3     return <Even />;
4   else
5     return <Odd />;
6 }

```

Komponent funkcyjny

```

1 class Number extends React.Component {
2   constructor(props) {
3     super(props);
4   }
5
6   render() {
7     if (this.props.value % 2 == 0)
8       return <Even />;
9     else
10      return <Odd />;
11   }
12 }

```

Komponent klasowy



## Listy i klucze

```

1 class Elements extends React.Component {
2   render() {
3     const elements = [
4       { id: 'e1', name: 'Element1' },
5       { id: 'e2', name: 'Element2' }
6     ];
7     const items = elements.map((element) =>
8       <li key={element.id}>
9         {element.name}
10      </li>
11    );
12
13    return (
14      <ul>
15        {items}
16      </ul>
17    );
18  }
19}
20const container = document.getElementById('root');
21const root = ReactDOM.createRoot(container);
22root.render(<Elements />);

```

### Notatki

---



---



---



---



---

## Wygenerowany HTML

```
<ul>
<li key='e1'>Element1</li>
<li key='e2'>Element2</li>
</ul>
```



## Zdarzenia

### Notatki

---



---



---



---



---

### Pierwszy sposób

```

1 class Hello extends React.Component {
2   render() {
3     return <button onClick={(function (event) {
4       console.log(this.props.welcome);
5       console.log(event._reactName);
6     }).bind(this)}>Click me</button>
7   }
8 }
9
10 const container = document.getElementById('root');
11 const root = ReactDOM.createRoot(container);
12 root.render(<Hello welcome="Hello World"/>);

```

### Drugi sposób

```

1 class Hello extends React.Component {
2   print(event) {
3     console.log(this.props.welcome);
4     console.log(event._reactName);
5   }
6   render() {
7     return <button onClick={this.print.bind(this)}>
8       Click me</button>
9   }
10 const container = document.getElementById('root');
11 const root = ReactDOM.createRoot(container);
12 root.render(<Hello welcome="Hello World"/>);

```

## Po naciśnięciu przycisku

Hello World  
onClick



## Przechowywanie danych w komponencie

## Notatki

```

1 class Hello extends React.Component {
2   constructor(props) {
3     super(props);
4     this.state = { firstName: "John", lastName: "Doe" };
5   }
6
7   clear = () => {
8     this.setState({ firstName: '', lastName: '' })
9   }
10  render() {
11    return (<div><h1>Hello {this.state.firstName} {this.state.lastName}</h1>
12      <button onClick={this.clear}>Clear</button></div>
13    )
14  }
15}
16 const container = document.getElementById('root');
17 const root = ReactDOM.createRoot(container);
18 root.render(<Hello />);

```

## Obsługa formularza

## Notatki

```

1 class Form extends React.Component {
2   constructor(props) {
3     super(props);
4     this.state = { value: '' };
5   }
6
7   handleChange = (event) => {
8     this.setState({ value: event.target.value });
9   }
10
11  handleSubmit = (event) => {
12    alert(`A name was submitted: ${this.state.value}`);
13    event.preventDefault();
14  }
15
16  render() {
17    return (
18      <form onSubmit={this.handleSubmit}>
19        <label>
20          Name:
21          <input type="text" value={this.state.value} onChange={this.handleChange} />
22        </label>
23        <input type="submit" value="Submit" />
24      </form>
25    );
26  }
27}
28
29 const container = document.getElementById('root');
30 const root = ReactDOM.createRoot(container);
31 root.render(<Form />);

```



## Kompozycja komponentów

```

1 class Hello extends React.Component {
2   render() {
3     return (
4       <div>
5         <Welcome text="Hello " />
6         <Welcome text="World" />
7       </div>
8     );
9   }
10 }
11
12 class Welcome extends React.Component {
13   render() {
14     return <span>{this.props.text}</span>
15   }
16 }
17
18 const container = document.getElementById('root');
19 const root = ReactDOM.createRoot(container);
20 root.render(<Hello />);

```

Notatki

### Wygenerowany HTML

```
<div>
<span>Hello </span>
<span>World</span>
</div>
```



## Fragmenty

```

1 class Hello extends React.Component {
2   render() {
3     return (
4       <div>
5         <Welcome text="Hello " />
6         <Welcome text="World" />
7       </div>
8     );
9   }
10 }
11
12 class Welcome extends React.Component {
13   render() {
14     return <span>{this.props.text}</span>
15   }
16 }
17 ...
18

```

```

1 class Hello extends React.Component {
2   render() {
3     return (
4       <div>
5         <Welcome text="Hello " />
6         <Welcome text="World" />
7       </div>
8     );
9   }
10 }
11
12 class Welcome extends React.Component {
13   render() {
14     return <React.Fragment>{this.props.text}</React.
15       Fragment>
16     // return <>{this.props.text}</>
17   }
18 }
19 ...

```

Notatki

### Wygenerowany HTML

```
<div>
<span>Hello </span>
<span>World</span>
</div>
```

### Wygenerowany HTML

```
<div>
Hello
World
</div>
```



## Wynoszenie stanu w góre

```

1 class Echo extends React.Component {
2   constructor(props) {
3     super(props);
4     this.handleChange = this.handleChange.
5       bind(this);
6     this.state = { text: '' };
7   }
8
9   handleChange(newText) {
10   this.setState({ text: newText });
11 }
12
13 render() {
14   return (
15     <React.Fragment>
16       <EchoInput text={this.state.text}
17         handleChange={this.handleChange} />
18       <EchoOutput text={this.state.text} />
19     </React.Fragment>
20   );
21 }
22 const container = document.getElementById('root');
23 const root = ReactDOM.createRoot(container);
24 root.render(<Echo />);
    
```

```

1 class EchoOutput extends React.Component {
2   render() {
3     return (
4       <div>Output: {this.props.text}</div>
5     );
6   }
7 }
8
9
10 class EchoInput extends React.Component {
11   constructor(props) {
12     super(props);
13     this.handleChange = this.handleChange.
14       bind(this);
15   }
16
17 handleChange(e) {
18   this.props.handleChange(e.target.value);
19 }
20
21 render() {
22   return (
23     <input value={this.props.text}
24       onChange={this.handleChange} />
25   );
26 }
27
28 }
29
30
31
32
33
    
```

Notatki

Źródło: <https://www.geeksforgeeks.org/lifting-state-up-in-reactjs/>



## Tworzenie podstron

Biblioteka 'React Router'

```

1 ...
2 <head>
3 ...
4   <script src='https://unpkg.com/react-router-dom@5.0.0/umd/react-
5     -router-dom.min.js'></script>
6 ...
7   class Route extends React.Component {
8     render() {
9       return (<ReactRouterDOM.HashRouter>
10         <ul>
11           <li><ReactRouterDOM.Link to="/">TO HOME</ReactRouterDOM.
12             Link></li>
13           <li><ReactRouterDOM.Link to="/hello">Hello</ReactRouterDOM.
14             Link></li>
15           <li><ReactRouterDOM.Link to="/echo">Echo</ReactRouterDOM.
16             Link></li>
17         </ul>
18         <ReactRouterDOM.Switch>
19           <ReactRouterDOM.Route path="/echo">
20             <Echo />
21           </ReactRouterDOM.Route>
22           <ReactRouterDOM.Route path="/hello">
23             <Hello />
24           </ReactRouterDOM.Route>
25           <ReactRouterDOM.Route path="/">
26             <Home />
27           </ReactRouterDOM.Route>
28         </ReactRouterDOM.Switch>
29       </ReactRouterDOM.HashRouter>
    
```

```

30   function Home() {
31     return <h2>Home</h2>;
32   }
33 ...
    
```

Notatki



## Haki

Notatki

### Haki

- ▶ useState
- ▶ useReducer
- ▶ useEffect
- ▶ useRef
- ▶ useLayoutEffect
- ▶ useContext
- ▶ useImperativeHandle
- ▶ useMemo
- ▶ useCallback

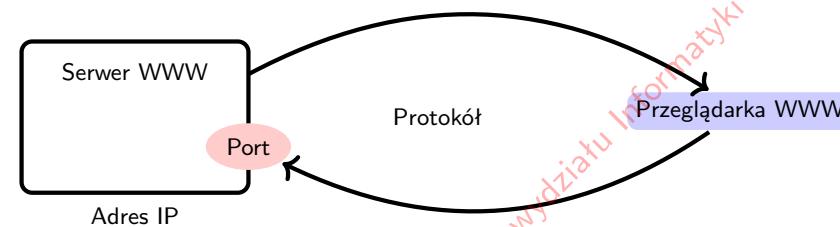
```

1 function Component() {
2   const [count, setCount] = React.useState(0);
3
4   return (
5     <div>
6       <p>Pressed {count} times</p>
7       <button onClick={() => setCount(count +
8         1)}>Press me</button>
9     );
10 }

```

## Model klient-serwer

Usługa WWW



Notatki



Materiały dla studentów wydziału Informatyki AGH w Krakowie



## Wybrane komendy (metody) protokołu HTTP

### Komenda „GET”

URL żądania:

<http://www.icsr.agh.edu.pl/index.html>

```
1 GET /index.html HTTP/1.1
2 Host: www.icsr.agh.edu.pl
3
```

Żądanie

```
1 HTTP/1.1 200 OK
2 Date: Mon, 09 Aug 2013 17:02:08 GMT
3 Server: Apache/2.4.4 (UNIX)
4 Content-Length: 1776
5 Content-Type: text/html; charset=utf-8
6
7 <!DOCTYPE html>
8 <html>
9 ...
10 </html>
```

Odpowiedź

### Komenda „POST”

URL żądania: http:

[//www.icsr.agh.edu.pl/cgi-bin/search.cgi](http://www.icsr.agh.edu.pl/cgi-bin/search.cgi)

```
1 POST /cgi-bin/search.cgi HTTP/1.1
2 Host: www.icsr.agh.edu.pl
3 Content-Length: 46
4
5 query=alpha+complex&casesens=false&cmd=submit
```

Żądanie

```
1 HTTP/1.1 200 OK
2 Date: Mon, 09 Aug 2013 17:02:20 GMT
3 Server: Apache/2.4.4 (UNIX)
4 Content-Length: 1776
5 Content-Type: text/html; charset=utf-8
6 Connection: close
7
8 <!DOCTYPE html>
9 <html>
10 ...
11 </html>
```

Odpowiedź

Notatki

---



---



---

## Wysyłanie danych z formularza HTML

Notatki

---



---



---

Zatwierdzanie danych → kodowanie → wysyłanie do serwera WWW

```
1 <form method="..." enctype="..." action="...">
2 ...
3 </form>
```

- ▶ GET
- ▶ POST

- ▶ application/x-www-form-urlencoded
- ▶ multipart/form-data



## Kodowanie „application/x-www-form-urlencoded”

Przykład

Notatki

```
1 <form action="http://www.serwer.com/script">
2   Login:   <input name="login" type="TEXT"><br>
3   Password: <input name="password" type="PASSWORD">
4 </form>
```

Dokument HTML

Login: Stanisław  
Password: Kowalski (Nowak)

```
1 login=Stanis%C5%82aw&password=Kowalski%28Nowak%29
```

Zakodowane dane



## Kodowanie „multipart/form-data”

Przykład

Notatki

```
1 <form action="..." method="POST" enctype="multipart/form-data">
2   <input name="login" type="TEXT">
3   <input name="password" type="PASSWORD">
4   <input name="file" type="FILE" accept="image/jpeg,image/gif">
5 </form>
```

Stanisław  
Kowalski (Nowak)  
image.jpg

```
1 POST /skrypt HTTP/1.0
2 Content-Length: 775
3 Content-Type: multipart/form-data; boundary=-----8152765018186645991017906692
4
5 -----8152765018186645991017906692
6 Content-Disposition: form-data; name="login"
7
8 Stanislaw
9 -----8152765018186645991017906692
10 Content-Disposition: form-data; name="password"
11
12 Kowalski (Nowak)
13 -----8152765018186645991017906692
14 Content-Disposition: form-data; name="file"; filename="image.jpg"
15 Content-Type: image/jpeg
16 Content-Transfer-Encoding: binary
17
18 The content of 'image.jpg'
19 -----8152765018186645991017906692
```



## Node.js

### Ogólna charakterystyka

- ▶ Udostępnia JavaScript po stronie serwera
- ▶ Używa V8 JavaScript Engine
- ▶ System do tworzenia serwisów sieciowych z asynchronicznym WE/WY
- ▶ Wykorzystuje paradymat programowania sterowanego zdarzeniami
- ▶ Nadaje się dobrze do pisania aplikacji, które wymagają komunikacji w czasie rzeczywistym pomiędzy przeglądarką a serwerem
- ▶ Pojedyncza instancja Node.js działa jako pojedynczy wątek

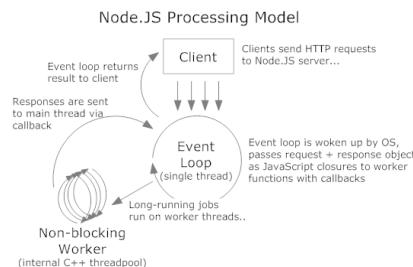
Pętla zdarzeń — podmiot, który obsługuje / przetwarza zdarzenia zewnętrzne i konwertuje je na wywołania funkcji zwrotnych

### Notatki

---

---

---



Rysunek: Schemat działania pętli zdarzeń w Node.js



## Przykład „Hello World”

```
1 #!/usr/bin/node
2 console.log("Hello World");
```

hello.js

### Notatki

---

---

---



## Wprowadzanie i wyprowadzanie danych

```

1 process.stdout.write('1');
2 process.stdout.write('2');
3 console.log(3)
4
5 /*
6  console.log = function(d) {
7   process.stdout.write(d + '\n');
8  };
9 */
10
11 process.stdin.setEncoding('utf8');
12 process.stdout.write('Enter data - pressing ^ D will finish entering them\n');
13 process.stdin.on('readable', function() {
14  let chunk;
15
16  while ((chunk = process.stdin.read()) !== null) {
17   process.stdout.write('Read: ' + chunk);
18  }
19 });
20 console.log("The end of the script has been reached");

```

script.js

### Notatki

---



---



---



---



---



---



---



---



---

## Dostęp do zmiennych środowiskowych, obsługa linii komend

### Notatki

---



---



---



---



---



---



---



---



---

```

1 //Reading the value of the 'HOME' environment variable
2 console.log("Your home directory is: "+process.env['HOME']);
3
4 //Displays the value of the command line arguments
5 console.log("The command line arguments are:");
6 process.argv.forEach(function(value, index, array) {
7  console.log('\t'+index + ': ' + value);
8 });

```

script.js



## Moduł

### Tworzenie

```

1 var myModule = require('myModule');
2 //myModule = require("./myModule");
3 /*****
4 console.log(myModule.variable1); //undefined
5 console.log(myModule.variable2); //2
6 console.log(myModule.variable3); //undefined
7 console.log(myModule.variable4); //4
8 console.log(myModule.fun1()); // "fun1"
9 console.log(myModule.fun2()); //Error
10 *****/
11 console.log(myModule); // { variable2: 2, variable4: 4, fun1: [Function] }
12 console.log(myModule()); //Error
13

```

script.js

```
1 $ export NODE_PATH='dir1:dir2:...:dirN'
```

## Moduł

### Korzystanie

#### Obsługa pakietów

```

$ npm install nazwa_pakietu
# Moduły → ./node_modules/
# Pliki wykonywalne → ./node_modules/.bin/
# Manuale → nie są instalowane

$ npm install --global nazwa_pakietu
# Moduły → {prefix}/lib/node_modules/
# Pliki wykonywalne → {prefix}/bin/
# Manuale → {prefix}/share/man/
# {prefix} = np. /usr

$ npm link nazwa_pakietu
# Wykonuje: ln -s {prefix}/lib/node_modules/nazwa_pakietu/ ./node_modules/

$ npx nazwa_programu

```

### Notatki

## Obsługa plików

```

1 var fs = require("node:fs");
2
3 fs.writeFileSync('file.txt', '1');
4 console.log('The value 1 has been saved');
5
6 let data = fs.readFileSync('file.txt');
7 console.log("Read value (sync): " + data);
8
9 fs.readFile('file.txt', 'utf-8', function (error,
10   data) {
11   if (error) throw error;
12   console.log("Read value (async): " + data);
13 });
14
15 fs.writeFile('file.txt', '2', function (error) {
16   if (error) throw error;
17   console.log('The value 2 has been saved');
18 });

```

Wersja nieprawidłowa

```

1 var fs = require("node:fs");
2
3 fs.writeFileSync('file.txt', '1');
4 console.log('The value 1 has been saved');
5 let data = fs.readFileSync('file.txt');
6 console.log("Read value (sync): " + data);
7
8 fs.readFile('file.txt', 'utf-8', function (error,
9   data) {
10   if (error) throw error;
11   console.log("Read value (async): " + data);
12
13   fs.writeFile('file.txt', '2', function (error) {
14     if (error) throw error;
15     console.log('The value 2 has been saved');
16   });
17 });

```

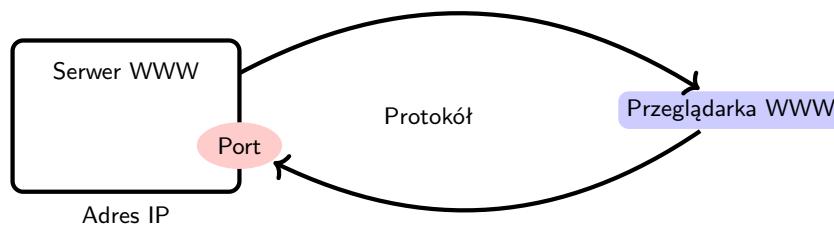
Wersja prawidłowa

## Notatki

## Model klient-serwer

## Usługa WWW

## Notatki



## Wybrane komendy (metody) protokołu HTTP

### Komenda „GET”

URL żądania:

<http://www.icsr.agh.edu.pl/index.html>

```
1 GET /index.html HTTP/1.1
2 Host: www.icsr.agh.edu.pl
3
```

Żądanie

```
1 HTTP/1.1 200 OK
2 Date: Mon, 09 Aug 2013 17:02:08 GMT
3 Server: Apache/2.4.4 (UNIX)
4 Content-Length: 1776
5 Content-Type: text/html; charset=utf-8
6
7 <!DOCTYPE html>
8 <html>
9 ...
10 </html>
```

Odpowiedź

### Komenda „POST”

URL żądania: http:

[//www.icsr.agh.edu.pl/cgi-bin/search.cgi](http://www.icsr.agh.edu.pl/cgi-bin/search.cgi)

```
1 POST /cgi-bin/search.cgi HTTP/1.1
2 Host: www.icsr.agh.edu.pl
3 Content-Length: 46
4
5 query=alpha+complex&casesens=false&cmd=submit
```

Żądanie

```
1 HTTP/1.1 200 OK
2 Date: Mon, 09 Aug 2013 17:02:20 GMT
3 Server: Apache/2.4.4 (UNIX)
4 Content-Length: 1776
5 Content-Type: text/html; charset=utf-8
6 Connection: close
7
8 <!DOCTYPE html>
9 <html>
10 ...
11 </html>
```

Odpowiedź

Notatki

---



---



---



---



---

## Wysyłanie danych z formularza HTML

Notatki

---



---



---



---



---

Zatwierdzanie danych → kodowanie → wysyłanie do serwera WWW

```
1 <form method="..." enctype="..." action="...">
2 ...
3 </form>
```

- ▶ GET
- ▶ POST

- ▶ application/x-www-form-urlencoded
- ▶ multipart/form-data

## Kodowanie „application/x-www-form-urlencoded”

Przykład

Notatki

---

---

```

1 <form action="http://www.server.com/script">
2   Login:  <input name="login" type="TEXT"><br>
3   Password: <input name="password" type="PASSWORD">
4 </form>
```

Dokument HTML

Login: Stanisław  
 Password: Kowalski (Nowak)

```
1 login=Stanis%C5%82aw&password=Kowalski%28Nowak%29
```

Zakodowane dane



## Kodowanie „multipart/form-data”

Przykład

Notatki

---

---

```

1 <form action="..." method="POST" enctype="multipart/form-data">
2   <input name="login" type="TEXT">
3   <input name="password" type="PASSWORD">
4   <input name="file" type="FILE" accept="image/jpeg,image/gif">
5 </form>
```

Stanisław  
 Kowalski (Nowak)  
 image.jpg

```

1 POST /skrypt HTTP/1.0
2 Content-Length: 775
3 Content-Type: multipart/form-data; boundary=-----8152765018186645991017906692
4
5 -----8152765018186645991017906692
6 Content-Disposition: form-data; name="login"
7
8 Stanislaw
9 -----8152765018186645991017906692
10 Content-Disposition: form-data; name="password"
11
12 Kowalski (Nowak)
13 -----8152765018186645991017906692
14 Content-Disposition: form-data; name="file"; filename="image.jpg"
15 Content-Type: image/jpeg
16 Content-Transfer-Encoding: binary
17
18 The content of 'image.jpg'
19 -----8152765018186645991017906692
```



## Obsługa protokołu HTTP

Szkielet skryptu

```

1 var http = require("node:http");
2
3 function requestListener(request, response) {
4   console.log("A request from the client has appeared");
5   response.writeHead(200, {"Content-Type": "text/plain"});
6   response.write("Hello World");
7   response.end();
8 }
9 var server = http.createServer(requestListener);
10 server.listen(8080);
11 console.log("Server started");

```

server.js

```

1 var http = require("node:http");
2
3 http.createServer(function(request, response) {
4   console.log("A request from the client has appeared");
5   response.writeHead(200, {"Content-Type": "text/plain"});
6   response.write("Hello World");
7   response.end();
8 }).listen(8080);
9 console.log("Server started");

```

Wersja alternatywna

Notatki

---



---



---



---



---

### Sprawdzenie działania



Rysunek: Z poziomu przeglądarki WWW

## Obsługa parametrów URL

```

1 ...
2 function requestListener(request, response) {
3   console.log("A request from the client has appeared");
4   var url = new URL(request.url, 'http://${request.headers.host}');
5
6   console.log(url);
7
8   response.writeHead(200, {"Content-Type": "text/plain"});
9   response.write('\n');
10  response.write(url.pathname+'\n');
11  response.write('Login: '+url.searchParams.get('login')+'\n');
12  response.write('Password: '+url.searchParams.get('password')+'\n');
13  response.end();
14 }
15 ...

```

server.js

Notatki

---



---



---



---



---

Na wyjściu

## Obsługa formularzy

Obsługa kodowania „application/x-www-form-urlencoded”

```

1 var qs = require('node:querystring');
2 ...
3 function requestListener(request, response) {
4   var url = new URL(request.url, 'http://${request.headers.host}');
5   if(url.pathname == '/') { //generating the form
6     response.writeHead(200, {"Content-Type": "text/html; charset=utf-8"});
7     response.write('<form method="POST" action="/submit">');
8     response.write('<input name="login" value="Jan">');
9     response.write('<input name="password" value="Kowalski (Nowak) qę">');
10    response.write('<input type="submit">');
11    response.write('</form>');
12    response.end();
13  }
14  if(url.pathname == '/submit') { //processing of the form content
15    if(request.method=='GET') {
16      response.writeHead(200, {"Content-Type": "text/plain; charset=utf-8"});
17      response.write(url.searchParams.get('login')+'\n'); //the browser will write: "Jan\n"
18      response.write(url.searchParams.get('password')+'\n'); //the browser will write: "Kowalski (Nowak) qę\n"
19      response.end();
20    }
21    else if(request.method=='POST') {
22      var body='';
23      request.on('data', function (data) {
24        body +=data;
25      });
26      request.on('end',function(){
27        var data = qs.parse(body); //body contains "login=Jan&password=Kowalski%28Nowak%29+%C4%85%C4%99"
28        response.writeHead(200, {"Content-Type": "text/plain; charset=utf-8"
29          });
30        response.write(data.login+'\n'); //the browser will write: "Jan\n"
31        response.write(data.password+'\n'); //the browser will write: "Kowalski (Nowak) qę\n"
32        response.end();
33      });
34    }
35  }
36 }
37 ...

```

Notatki

---



---



---



---



---



---



---



---



## Wolnostojący serwer WWW

### Instalacja i uruchamianie

```

$ npx http-server -h
usage: http-server [path] [options]

options:
  -p          Port to use [8080]
  -a          Address to use [0.0.0.0]
  -d          Show directory listings [true]
  -i          Display autoIndex [true]
  -e --ext    Default file extension if none supplied [none]
  -s --silent Suppress log messages from output
  -h --help   Print this list and exit.
  -c          Set cache time (in seconds). e.g. -c10 for 10
              seconds.
              To disable caching, use -c-.

$ npx http-server
Starting up http-server, serving ./ on port: 8080
Hit CTRL-C to stop the server

```



Notatki

---



---



---



---



---



---



---



---

Materiały dla studentów wydziału Informatyczno-Ekonomicznego w Krakowie

## Obsługa bazy danych SQLite 3

Notatki

```

1 var sqlite3 = require('sqlite3');
2
3 var db = new sqlite3.Database(':memory:'); //returns 'Database' object
4
5 db.serialize(function() {
6   db.run("CREATE TABLE products (info TEXT)");
7   var stmt = db.prepare("INSERT INTO products VALUES (?)"); //returns 'Statement' object
8   for (var i = 0; i < 2; i++) {
9     stmt.run("Product " + i);
10    stmt.finalize();
11  }
12
13  jsonData = { products: [] };
14  db.each("SELECT rowid AS id, info FROM products", function(err, row) {
15    jsonData.products.push({ id: row.id, info: row.info });
16  }, function () {
17    console.log(JSON.stringify(jsonData)); //JSON.stringify - built-in JS function
18  });
19 });
20 });
21 db.close();

```

bd.js



## Tworzenie dodatków (addons) w C++

Przykład „Hello World”

Notatki

```

1 /*
2  The following program is equivalent to the following JS code:
3  exports.hello = function() { return 'world'; };
4 */
5
6 #include <node.h>
7 using namespace v8;
8
9 void Method(const FunctionCallbackInfo<Value> &args)
10{
11  Isolate *isolate = args.GetIsolate();
12  args.GetReturnValue().Set(String::NewFromUtf8(
13    isolate, "world")
14    .ToLocalChecked());
15}
16
17 void init(Local<Object> exports) {
18  NODE_SET_METHOD(exports, "hello", Method); //Associate the name
19  // 'hello' with the above C++ method and export it
20}
21 NODE_MODULE(NODE_GYP_MODULE_NAME, init) //there is no semicolon

```

hello.cc

```

1 {
2   "targets": [
3     {
4       "target_name": "hello",
5       "sources": [ "hello.cc" ]
6     }
7   ]
8 }

```

binding.gyp

## Kompilacja i uruchomienie

```

$ npx node-gyp configure
$ npx node-gyp build
$ node hello.js
world

```

```

1 var addon = require('../build/Release/hello');
2 console.log(addon.hello());

```

hello.js



## Framework

- ▶ Dostarcza szkieletu do tworzenia aplikacji lub jej wybranych części
- ▶ Dostarcza zbioru ogólnych funkcjonalności
- ▶ Gotowe komponenty do budowy aplikacji WWW
- ▶ Ułatwia proces wdrażania nowego pracownika
- ▶ Nie jest gotową aplikacją — jest zbiorem kodu i pewnych metod postępowania, dzięki którym można efektywnie budować aplikacje
- ▶ Typowe cechy:
  - ▶ Odwrócenie sterowania
  - ▶ Domyślne zachowanie
  - ▶ Rozszerzalność
  - ▶ Zamknięta struktura wewnętrzna
- ▶ Typowe elementy:
  - ▶ Mechanizm uruchamiania i przetwarzania akcji
  - ▶ Mechanizm tworzenia logiki biznesowej aplikacji
  - ▶ Zarządzanie konfiguracją
  - ▶ Zarządzanie komunikacją z bazą danych
  - ▶ Obsługa formularzy
  - ▶ System szablonów
  - ▶ Obsługa błędów
  - ▶ Mechanizmy bezpieczeństwa, uwierzytelniania i kontroli dostępu
  - ▶ Generatory kodu

Notatki



## System szablonów

### Pug

```

1 const pug = require('pug');
2 const compiledFunction = pug.compileFile('template.
3   pug');
4 const html = compiledFunction(
5   {
6     title: 'Hello World'
7 });
8 console.log(html);

```

pug.js

```

1 html
2   head
3     title #{title}
4   body
5     h1 #{title}

```

template.pug

### EJS

```

1 const ejs = require('ejs');
2 ejs.renderFile('template.ejs',
3   {
4     title: 'Hello World' }, function (err, html) {
5       console.log(html);
6   });

```

ejs.js

```

1 <html>
2   <head>
3     <title><%= title %></title>
4   </head>
5   <body>
6     <h1><%= title %></h1>
7   </body>
8 </html>

```

template.ejs

Notatki



## Tworzenie szkieletu aplikacji

Notatki

```
1 $ mkdir MySite
2 $ cd MySite
3 $ vi package.json
4 $ npm install # Installing dependencies
```

```
1 {
2   "name": "MySite",
3   "version": "0.0.1",
4   "private": "true",
5   "dependencies": {
6     "express": "*",
7     "morgan": "*"
8   }
9 }
```

package.json



## Plik główny aplikacji

Notatki

```
1 var express = require('express'),
2     logger = require('morgan');
3
4 var app = express();
5
6 // If you uncomment the code snippet below, then
7 // the following function will be executed every
8 // time the app receives a request.
9 /*
10 app.use(function (req, res, next) {
11   console.log('Time:', Date.now());
12   next();
13 });
14 */
15 app.use(logger('dev'));
16
17 app.get('/', function (req, res) {
18   res.send('<h1>Hello World!</h1>');
19 });
20 app.listen(3000);
```

app.js

### Terminal 1

```
$ node app
GET / 200 3.137 ms - 21
```

### Terminal 2

```
$ curl http://localhost:3000/
<h1>Hello World!</h1>
```



## Plik główny aplikacji

Notatki

```
1 $ npm install #installing dependencies
```

```
1 {
2   ...
3   "dependencies": {
4     ...
5     "pug": "*",
6   }
7 }
```

package.json



## Plik główny aplikacji

Notatki

```
1 var express = require('express'),
2     logger = require('morgan');
3
4 var app = express();
5 var router = express.Router();
6
7 app.set('view engine', 'pug');
8 app.set('views', __dirname + '/views');
9
10 app.use(logger('dev'));
11 app.use(express.static(__dirname + '/public'));
12
13 router.get('/', function (req, res) {
14   res.render('index',
15   { title : 'Przykład' })
16 });
17
18 app.use('/', router);
19
20 app.listen(3000);
```

app.js



## Plik Pug

```

1 doctype html
2 html(lang='pl')
3   head
4     title #{title}
5     link(rel='stylesheet', href='/stylesheets/style.
       css')
6   body
7     header
8       h1 Moja strona
9     main
10    p
11      | Witaj Świecie
12      | Witaj Świecie
13    p
14      | Witaj Świecie
15  aside
16    h1 Nagłówek
17    p
18      | Treść ramki
19  footer
20    p Aplikacja stworzona w oparciu o framework
       Express
21
22
23
24
25
26

```

views/index.pug

```

1 <!DOCTYPE html>
2 <html lang="pl">
3   <head>
4     <title>Przykład</title>
5     <link rel="stylesheet" href="/stylesheets/style.
       css">
6   </head>
7   <body>
8     <header>
9       <h1>Moja strona</h1>
10      </header>
11      <main>
12        <p>
13          Witaj Świecie
14          Witaj Świecie
15        </p>
16        <p>Witaj Świecie</p>
17      </main>
18      <aside>
19        <h1>Nagłówek</h1>
20        <p>Treść ramki</p>
21      </aside>
22      <footer>
23        <p>Aplikacja stworzona w oparciu o framework
           Express</p>
24      </footer>
25    </body>
26  </html>

```

Notatki



## Plik CSS

```

1 aside {
2   float: right;
3   border: 5px solid blue;
4   padding: 1px;
5   margin-bottom: 14px;
6   width: 20%;
7 }
8 main {
9   float: left;
10  background-color: #44f;
11  padding: 5px;
12  width: 75%;
13 }
14 footer {
15   clear:both;
16   border-style: dotted;
17 }
18 header {
19   text-align: center;
20 }

```

public/stylesheets/style.css

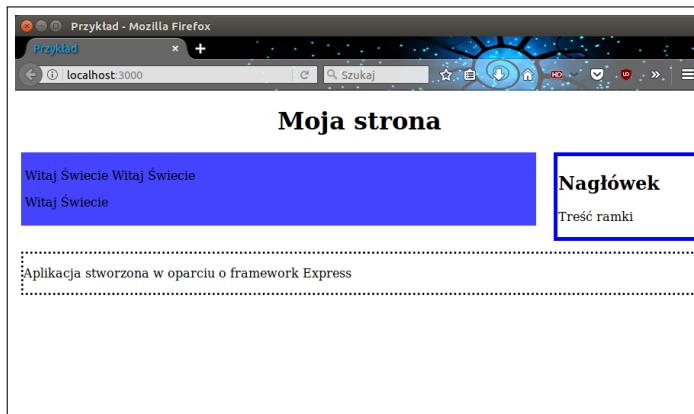
Notatki



## Uruchamianie aplikacji

Notatki

```
1 $ node app
2 GET / 304 847.588 ms -
3 GET /stylesheets/style.css 304 4.478 ms -
```



## Parametry trasy

Notatki

```
1 ...
2 app.get('/students/:studentName/departments/:departmentId', function (req, res) {
3   res.send('studentName=' + req.params.studentName, departmentId=req.params.departmentId)
4 });
5 ...
```

app.js

### Terminal 1

```
$ node app.js
GET /students/Jan%20Kowalski/departments/1 200 4.402 ms - 53
```

### Terminal 2

```
$ curl http://localhost:3000/students/Jan%20Kowalski/departments/1
studentName=Jan Kowalski, departmentId=1
```



## Obsługa formularzy

Notatki

- ▶ <http://localhost:3000/> — strona główna z formularzem
- ▶ <http://localhost:3000/submit> — dane z formularza

### Metoda „GET”

```

1 ...
2 router.get('/', function (req, res) {
3   res.send(`
4     <form method="GET" action="/submit">
5       <input name="login" value="Jan">
6       <input name="password" value="Kowalski (Nowak) àę"
7       >
8       <input type="submit">
9     </form>
10   `);
11 }
12 router.get('/submit', function (req, res) {
13   res.send(`
14     login=${req.query.login}
15     <br>
16     password=${req.query.password}
17   `);
18 });
19 ...

```

app.js

### Metoda „POST”

```

1 ...
2 app.use(express.urlencoded({ extended: false }));
3
4 router.get('/', function (req, res) {
5   res.send(`
6     <form method="POST" action="/submit">
7       <input name="login" value="Jan">
8       <input name="password" value="Kowalski (Nowak) àę"
9       >
10      <input type="submit">
11    </form>
12  `);
13
14 router.post('/submit', function (req, res) {
15   res.send(`
16     login=${req.body.login}
17     <br>
18     password=${req.body.password}
19   `);
20 });
21 ...

```

app.js



## Polecenie express

Notatki

```

1 $ npx express-generator --help
2 Usage: express [options] [dir]
3
4 Options:
5
6   --version      output the version number
7   -e, --ejs      add ejs engine support
8   --pug          add pug engine support
9   --hbs          add handlebars engine support
10  -H, --hogan    add hogan.js engine support
11  -v, --view <engine> add view <engine> support (dust|ejs|hbs|hjs|jade|pug|twig|vash) (defaults to jade)
12    --no-view    use static html instead of view engine
13  -c, --css <engine> add stylesheet <engine> support (less|stylus|compass|sass) (defaults to plain css)
14    --git        add .gitignore
15  -f, --force    force on non-empty directory
16  -h, --help     output usage information

```



## Generowanie aplikacji „Hello World”

Użycie polecenia express

Notatki

---

---

---

---

---

---

---

---

---

---

---

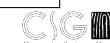
---

---

---

---

```
1 $ npx express-generator --view=pug MySite
2   create : MySite/
3   create : MySite/public/
4   create : MySite/public/javascripts/
5   create : MySite/public/images/
6   create : MySite/public/stylesheets/
7   create : MySite/public/stylesheets/style.css
8   create : MySite/routes/
9   create : MySite/routes/index.js
10  create : MySite/routes/users.js
11  create : MySite/views/
12  create : MySite/views/error.pug
13  create : MySite/views/index.pug
14  create : MySite/views/layout.pug
15  create : MySite/app.js
16  create : MySite/package.json
17  create : MySite/bin/
18  create : MySite/bin/www
19
20 change directory:
21 $ cd MySite
22
23 install dependencies:
24 $ npm install
25
26 run the app:
27 $ DEBUG=mysite:* npm start
```



## Plik 'package.json'

Notatki

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

```
1 {
2   "name": "mysite",
3   "version": "0.0.0",
4   "private": true,
5   "scripts": {
6     "start": "node ./bin/www"
7   },
8   "dependencies": {
9     "cookie-parser": "~1.4.4",
10    "debug": "~2.6.9",
11    "express": "~4.16.1",
12    "http-errors": "~1.6.3",
13    "morgan": "~1.9.1",
14    "pug": "2.0.0-beta11"
15  }
16 }
```



## Plik główny aplikacji (app.js)

Notatki

```

1 ...
2 var indexRouter = require('./routes/index');
3 var usersRouter = require('./routes/users');
4
5 var app = express();
6 ...
7 app.use('/', indexRouter);
8 app.use('/users', usersRouter);
9 ...

```

Przykładowy adres	Obiekt, który go obsługuje
http://localhost:3000/	indexRouter
http://localhost:3000/users/	usersRouter
http://localhost:3000/agh/	Żaden — pojawi się błąd
http://localhost:3000/users/agh/	Żaden — pojawi się błąd

## Pliki z definicją tras

Notatki

```

1 var express = require('express');
2 var router = express.Router();
3
4 /* GET home page. */
5 router.get('/', function(req, res, next) {
6   res.render('index', { title: 'Express' });
7 });
8
9 module.exports = router;

```

routes/index.js

```

1 var express = require('express');
2 var router = express.Router();
3
4 /* GET users listing. */
5 router.get('/', function(req, res, next) {
6   res.send('respond with a resource');
7 });
8
9 module.exports = router;

```

routes/user.js

## Pozostałe wygenerowane pliki

```
1 extends layout.pug
2
3 block content
4   h1= title
5   p Welcome to #{title}
```

views/index.pug

```
1 doctype html
2 html
3   head
4     title= title
5     link(rel='stylesheet', href='/stylesheets/style.
6       css')
7   body
8     block content
```

views/layout.pug

```
1 body {
2   padding: 50px;
3   font: 14px "Lucida Grande", Helvetica, Arial, sans
4   -serif;
5 }
6 a {
7   color: #00B7FF;
8 }
```

public/stylesheets/style.css

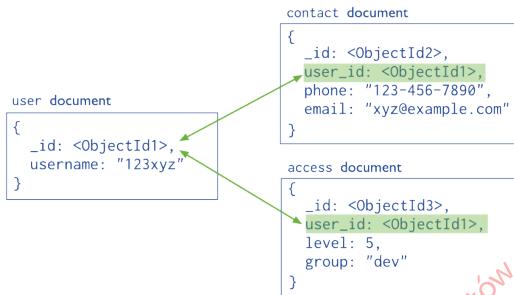
```
1 $ cd MySite && npm install
2 $ DEBUG=mysite:* npm start
3
4 > mysite@0.0.0 start
5 > node ./bin/www
6
7 mysite:server Listening on port 3000 +0ms
```

Instalowanie zależności i uruchamianie aplikacji

Notatki

## Model danych w MongoDB

### Referencje



### Osadzane dokumenty



Notatki

## Tworzenie zawartości bazy danych

Notatki

```

1 $ mongo
2 MongoDB shell version: 4.4.5
3 connecting to: test
4 > use testbase1
5 switched to db testbase1
6 > db.datacollection.insert({ "title": "Express & Mongo" })
7 > db.datacollection.insert({ "title": "Express" })
8 > db.datacollection.find()
9 { "_id" : ObjectId("52f8bb757bade7e2c4741741"), "title" : "Express & Mongo" }
10 { "_id" : ObjectId("52f8bd407bade7e2c4741742"), "title" : "Express" }
11 > db.datacollection.find({ "title": "Express" })
12 { "_id" : ObjectId("52f8bd407bade7e2c4741742"), "title" : "Express" }

```



## Instalowanie zależności

Notatki

```

1 $ vi package.json
2 $ npm install # Installing dependencies

```

```

1 ...
2   "dependencies": {
3     ...
4       "mongodb": "*"
5   }
6 ...

```

package.json



## Pobieranie i wyświetlanie danych

```

1 ...
2 router.get('/titles', async function (req, res, next) {
3   const MongoClient = require('mongodb').MongoClient;
4   const client = new MongoClient('mongodb://127.0.0.1:27017');
5   await client.connect();
6   const db = client.db('testbase1');
7   const collection = db.collection('datacollection');
8   const docs = await collection.find({}).toArray();
9   res.render('titlelist', {"titlelist": docs});
10  client.close();
11 });
12
13 module.exports = router;

```

routes/index.js

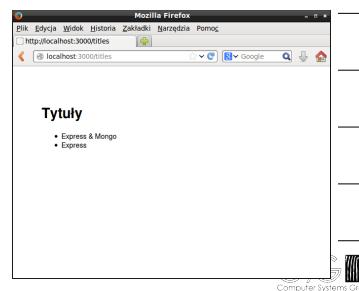
```

1 extends layout.pug
2
3 block content
4   h1 Tytuły
5   ul
6     each element in titlelist
7       li #{element.title}

```

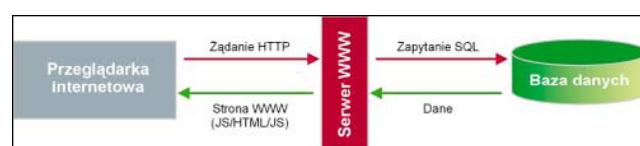
views/titlelist.pug

Notatki



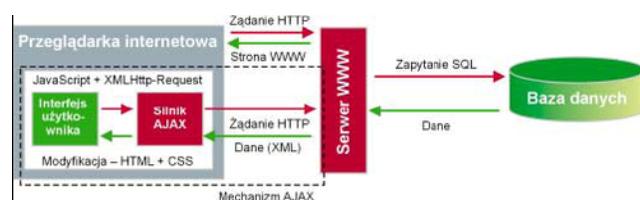
## AJAX

- ▶ **AJAX (Asynchroniczny JavaScript i XML)**
- ▶ **AJAX = HTML + CSS + DOM + XMLHttpRequest + XML + JavaScript**
- ▶ **Możliwości:**
  - ▶ Wysyłanie zapytania do serwera bez przeładowywania strony
  - ▶ Aplikacja może dokonywać szybkich, przyrostowych aktualizacji w interfejsie użytkownika bez potrzeby przeładowywania całej strony w przeglądarce
  - ▶ Parsowanie i praca z dokumentami XML
- ▶ **Czy warto go zawsze stosować?**



Notatki

Rysunek: Schemat działania typowej witryny internetowej



Rysunek: Schemat działania witryny internetowej opartej o AJAX

**AJAX**

Przykład użycia

**Zapytanie**

```

1 var xhr;
2
3 xhr = new XMLHttpRequest();
4 if (!xhr) {
5   alert('I can not create an XMLHttpRequest object instance');
6   return;
7 }
8 xhr.onreadystatechange = function() { alertContents(xhr);
9   );
10 // xhr.onreadystatechange = () => alertContents(xhr);
11 xhr.open('GET', "/requested_file.html", true);
12 xhr.send(null);

```

```

1 <table>
2 <tr><td>Hello</td></tr>
3 </table>

```

requested\_file.html

**Obsługa odpowiedzi**

```

1 function alertContents(xhr) {
2   if (xhr.readyState === XMLHttpRequest.DONE) { // XMLHttpRequest.DONE => 4
3     if (xhr.status === 200) {
4       alert(xhr.responseText);
5       var xmldoc = xhr.responseXML;
6       var root_node = xmldoc.getElementsByTagName('
7         td').item(0);
8       alert(root_node.firstChild.data);
9     }
10   else {
11     alert('There was a problem with this task.');
12   }

```

**Notatki****AJAX**

Wysyłanie danych typu „application/x-www-form-urlencoded”

**GET**

```

1 ...
2 xhr.open('GET', "/submit.php?field1=value1&field2=value2&...", true);
3 xhr.send(null);

```

**POST**

```

1 ...
2 xhr.open('POST', "/submit.php", true);
3 xhr.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
4 xhr.send('field1=value1&field2=value2&...');

```

**Notatki**

## Odbieranie danych w określonym formacie

```

1 var xhr;
2
3 xhr = new XMLHttpRequest();
4 xhr.responseType = "json";
5 ...
6
7 function alertContents(xhr) {
8   ...
9   alert(xhr.response);
10 }

```

responseType	Typ odbieranych danych
""	DOMString
"text"	DOMString
"arraybuffer"	ArrayBuffer
"blob"	Blob
"document"	Document lub XMLDocument
"json"	JSON

## Notatki

## Obsługa odpowiedzi za pomocą zdarzeń

```

1 var xhr;
2
3 xhr = new XMLHttpRequest();
4
5 // Called if 'xhr.readyState == XMLHttpRequest.DONE'
6 xhr.addEventListener("load", function (evt) {
7   if (xhr.status === 200) {
8     ...
9   }
10 });
11
12 xhr.addEventListener("error", function (evt) {
13   window.alert('There was a problem with this request.');
14 });
15
16 xhr.open('GET', "/requested_file.html", true);
17 xhr.send(null);

```



## Notatki



## Obietnice

Notatki

- ▶ Reprezentują / Przechowują wyniki operacji asynchronicznej<sup>1</sup>, np. zapytania AJAX
- ▶ Wyniki operacji mogą nie być jeszcze dostępne, ale kiedyś będą
- ▶ Obietnice można ze sobą składać

### Stany obietnic

- ▶ W oczekiwaniu (ang. pending)
- ▶ Spełniona (ang. fulfilled)
- ▶ Odrzucona (ang. rejected)
- ▶ Rozstrzygnięta (ang. settled)

<sup>1</sup>Wartość zwracaną (w przypadku sukcesu) lub błąd (w razie porażki)

## Obietnice

### Przykład użycia

```

1  /** Creating a promise ***/
2  function createPromise() {
3    return new Promise(function(resolve, reject) {
4      // Calculations performed asynchronously
5      setTimeout(function() {
6        var divider = Math.floor(Math.random() * 3);
7        if(divider != 0)
8          resolve(10/divider); // Fulfill a promise
9        else
10          reject("Attempt to divide by 0"); // Reject a promise
11      }, 2000);
12    });
13  }
14
15  /** The use of the promise ***/
16  function usePromise() {
17    createPromise() //An instance of the promise has been
18    //created
19    .then(function(result) {
20      console.log('Division result:', result);
21    })
22    .catch(function(error) {
23      console.log('An error occurred!', error);
24    });
25  }
26  usePromise();

```

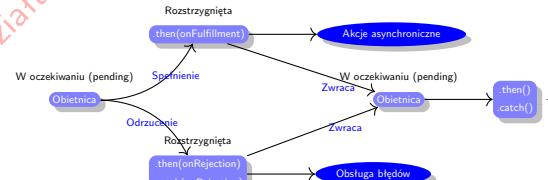
### Na wyjściu

'Division result:' 5

### Na wyjściu

"An error occurred!" "Attempt to divide by 0"

Notatki



## Obietnice

Składnia `async / await`

Notatki

---



---



---

Tak się używało obietnic dotychczas

```
1 function usePromise() {
2   createPromise()
3     .then(function (result) {
4       console.log('Division result : ', result);
5     })
6     .catch(function (error) {
7       console.log('An error occurred! ', error);
8     });
9 }
10 usePromise();
```

Użycie `async / await`

```
1 async function usePromise() {
2   try {
3     const result = await createPromise();
4     console.log('Division result : ', result);
5   } catch (error) {
6     console.log('An error occurred! ', error);
7   }
8 }
9 usePromise();
```



## Fetch API

Ogólna charakterystyka

Notatki

---



---



---

- ▶ Interfejs JS do uzyskiwania dostępu oraz manipulowania częściami potoku HTTP, takimi jak: żądanie oraz odpowiedź
- ▶ Różnice w stosunku do AJAX-a:
  - ▶ Korzysta z obietnic, a nie funkcji zwrotnych
    - ▶ Obietnica będąca rezultatem wykonania zapytania nie otrzyma stanu 'Odrzucona', nawet wtedy status odpowiedzi HTTP to 404 lub 500.
  - ▶ Domyslnie funkcja do wykonywania zapytań nie wysyła ani nie odbiera żadnych plików cookie z serwera



## Fetch API

### Przykład użycia

```

1 ...
2 <script>
3   const header = new Headers();
4   header.append('Content-Type', 'text/plain');
5   const request = new Request('/requested_file.html',
6   {
7     method: 'GET',
8     headers: header,
9   });
10  fetch(request).then(response => {
11    if (response.status !== 200)
12      return Promise.reject('The query failed');
13    else {
14      console.log(response); /*Response {
15        type: "basic",
16        url: "/requested_file.html",
17        redirected: false,
18        status: 200,
19        ok: true,
20        statusText: "OK",
21        headers: Headers,
22        bodyUsed: false
23      }*/
24      //console.log(response.arrayBuffer()); //Promise { <state>: "pending" }
25      //console.log(response.blob()); //Promise { <state>: "pending" }
26      //console.log(response.json()); //Promise { <state>: "pending" }
27      //console.log(response.text()); //Promise { <state>: "pending" }
28      //console.log(response.formData()); //Promise { <state>: "pending" }
29      response.text().then(function (text) {
30        console.log(text); //The content of the current file
31      });
32    } //if (response.status === 200)
33  }).catch(error => console.error(error))
34 </script>
35 ...

```

document.html

### Notatki

---



---



---



---



---



---



---



---



---



---



---



---



## Fetch API

### Wysyłanie danych typu „application/x-www-form-urlencoded”

#### GET

```

1 function requestGet() {
2   const response = fetch("/submit.php?field1=value1&field2=value2...", {
3     method: 'GET',
4   }).then((response) => {
5     console.log(response.text());
6   });
7 }

```

### Notatki

---



---



---



---



---



---



---



---



---



---



---



---

#### POST

```

1 function requestPost() {
2   const response = fetch("/submit.php", {
3     headers: {
4       'Content-Type': 'application/x-www-form-urlencoded',
5     },
6     method: 'POST',
7     body: 'field1=value1&field2=value2...'
8   }).then((response) => {
9     console.log(response.text());
10  });
11 }

```



## Przykład „Hello World”

Notatki

```
1 console.log("Hello World");
```

script.ts

```
1 console.log("Hello World");
```

script.js



## Obsługa błędów

Błąd składniowy

Notatki

Błąd typowania

```
1 let alive: boolean = 'abc';
2 console.log(alive);
```

script.ts

```
1 let if = 2;
2 console.log(if);
```

script.ts

```
1 let alive = 'abc';
2 console.log(alive);
```

script.js

```
1 let ;
2 if ( = 2)
3 ;
4 console.log();
5 if ()
6 ;
```

script.js



## Deklarowanie typu zmiennej

```

1 let alive:boolean = true;
2 let age:number = 48;
3 let name:string = 'Kowalski';
4
5 let names:string[] = ['Jan', 'Jerzy'];
6 let children_age:Array<number> = [1, 20, 3];
7 let parents_age:Array<number> = [40, "forty"
8     one"]; //... Type 'string' is not
9     assignable to type 'number'.
10
11 let tuple:[string,number];
12 tuple = ['1',2,true];
13 console.log(tuple[2]); // true
14 tuple = ['1',2,true,4]; // OK
15 tuple = ['1',2]; //... Type '[string, number]',
16     is not assignable to type '[string,
17     number, boolean]'. ...
18 tuple = [1,2,3]; //... Type '[number, number,
19     number]' is not assignable to type '[  

20     string, number, boolean]'.

```

Dr inż. Stanisław Polak

```

21 let anything:any = 4;
22 anything = "String"; //OK
23 anything = true; //OK
24 anything.x; //OK
25 anything(); //OK
26 new anything(); //OK
27 let string: string = anything; //OK
28
29 let something: unknown = 4;
30 something = "String"; //OK
31 something = true; //OK
32 something.x; //... Object is of type 'unknown'
33 something(); //... Object is of type 'unknown'
34 new something(); //... Object is of type '  

35     unknown'
36 let string2: string = something; // ... Type '  

37     unknown' is not assignable to type '  

38         string'.
39
40 function print(message): void {
41     console.log(message);
42 }
43
44 function exception(message): never {
45     throw new Error(message);
46 }
47
48 function loop(): never {
49     while(true){}
50 }

```

## Notatki

---



---



---



---



---



---



155

## Typowanie

```

1 let string:string;
2 string = 1; //Error, you can not assign a number to
3     a string type variable
4
5 let number = 1;
6 number = '2'; //Error, you can not assign a string
7     to a numeric type variable

```

script.ts

```

1 let string:string;
2 string = <any> 1; //Now it is OK
3 //or
4 string = 1 as any;
5 let number = 1;
6 number = <any> '2'; //Now it is OK
7 //or
8 number = '2' as any; //Now it is OK

```

script.ts

## Notatki

---



---



---



---



---



---

## Interfejsy

Tworzenie typu złożonego

```

1 interface Person {
2   first_name: string; //mandatory
3   last_name: string; //mandatory
4   age?: number;      //optional
5 }
6 ****
7 let user: Person;
8 ****
9 user = {
10   first_name: 'Jan',
11   last_name: 'Kowalski'
12 }
13 ****
14 user = {
15   first_name: 'Jan',
16   last_name: 'Kowalski',
17   age: '40' //Error: wrong value type
18 }
19 ****
20 user = {
21   // Error: 'last_name' is not specified
22   first_name: 'Jan',
23 }
24 ****
25 user = {
26   first_name: 'Jan',
27   last_name: 345, //Error: wrong value type
28   age: 'teenager',
29 }
```

script.ts

Notatki

---



---



---



---



---



---



---



---



---



---



## Interfejsy

Określanie typów funkcji

```

1 interface stringFunction {
2   (param1:string, param2:string): string;
3 }
4 ****
5 let addStrings: stringFunction;
6 let addNumbers: stringFunction;
7 ****
8 addStrings = function(string1: string, string2: string) {
9   return string1+string2;
10 } //OK
11
12 addNumbers = function(number1: number, number2: number) {
13   return number1 + number2;
14 } /*...
15 error TS2322: Type '(number1: number, number2: number) => number' is not assignable to type 'stringFunction'.
16   Types of parameters 'number1' and 'param1' are incompatible.
17   Type 'number' is not assignable to type 'string'.
18 */
19 ****
20 addStrings('Jan','Kowalski'); //OK
21 addStrings('Kowalski'); //... error TS2346: Supplied parameters do not match any signature of call target.
22 addStrings(1,2); //... error TS2345: Argument of type 'number' is not assignable to parameter of type 'string'.
23 
```

script.ts

Notatki

---



---



---



---



---



---



---



---



---



---



Materiały dla studentów Wydziału Informatyki AGH w Krakowie

## Interfejsy

Typ indeksowany

Notatki

---



---



---



---



---



---



---



---

```

1 interface hashString {
2   [index: string]: string;
3 }
4 let parameters: hashString = {};
5
6 parameters['server'] = 'HP'; // OK
7 let bar:number = parameters['server']; //... error TS2322: Type 'string' is not assignable to type 'number'.
8 parameters['server'] = 234; //... error TS2322: Type 'number' is not assignable to type 'string'.

```

script.ts



## Interfejsy

Typy klasowe

Notatki

---



---



---



---



---



---



---



---

```

1 class Person {
2   id:number;
3 }
4
5 interface Validator{
6   checkIdUniqueness(s: number): boolean;
7 }
8
9 class Employee extends Person implements Validator {}

```

script.ts



# Klasy

## Modyfikatory

```

1 class Person {
2   protected _id:number;
3   readonly first_name:string;
4   readonly last_name:string = "Anonym";
5
6   get id(): number {
7     return this._id;
8   }
9
10  set id(value: number) {
11    this._id = value
12  }
13
14  constructor(first_name:string, name:string) {
15    this.first_name = first_name;
16    this.last_name = last_name;
17  }
18}

```

```

19 class Employee extends Person {
20   constructor(id:number, first_name:string,
21             last_name:string) {
22     super(first_name, last_name);
23     this.id=id;
24   }
25
26 let Employee = new Employee(1,"Stanisław",
27                             "Polak");
27 console.log(Employee.id);           //1
28 console.log(Employee._id);          //Property '_id' is protected and only accessible
29 console.log(Employee.first_name);   //Stanisław
30 console.log(Employee.last_name);    //Polak
31 Employee.first_name = "Jan"; //...Left-hand
32                               side of assignment expression cannot be a
33                               constant or a read-only property.

```

script.ts

## Notatki

---



---



---



---



---



---



---



---



---



---

# Klasy oraz typy generyczne

```

1 class CustomCollection<T> {
2   private itemArray: Array<T>;
3
4   constructor() {
5     this.itemArray = [];
6   }
7
8   Add(item: T) {
9     this.itemArray.push(item);
10  }
11
12  GetFirst(): T {
13    return this.itemArray[0];
14  }
15}
16 ****
17 class User {
18   public Name;
19}
20 ****
21 class Message {
22   public Message;
23}

```

```

24 class MyApp {
25   constructor() {
26     let myUsers = new CustomCollection<User>();
27     let myMessages = new CustomCollection<Message>();
28
29     let user: User = myUsers.GetFirst(); // OK
30     let message: Message = myUsers.GetFirst(); // Error because of
31                                         // the Generic type validation.
32     myUsers.Add(new Message());           // Error because of
33                                         // the Generic type validation.
34   }
35}

```

script.ts

## Notatki

---



---



---



---



---



---



---



---



---



---

## Dekoratory

```

1 function f(): any {
2   console.log('f(): evaluated');
3   return function (target, propertyKey: string, descriptor: PropertyDescriptor) {
4     console.log("f(): called");
5     console.log(target);
6     console.log();
7     console.log(propertyKey);
8     console.log();
9     console.log(descriptor);
10    }
11  }
12
13 function g(value): any {
14   console.log(`g(${value}): evaluated`);
15   return function (target, propertyKey: string, descriptor: PropertyDescriptor) {
16     console.log(`g(${value}): called`);
17   }
18 }
19
20 class C {
21   @f()
22   @g('abc')
23   method() {}
24 }
```

script.ts

Źródło: <https://github.com/Microsoft/TypeScript-Handbook/blob/master/pages/Decorators.md>

## Notatki

---



---



---



---



---



---



---



---



---

## Przestrzeń nazewnicza

Jednoplikowa

```

1 namespace A {
2   var a:string = 'abc';
3   export class Twix {
4     constructor() {
5       console.log('Twix');
6     }
7   }
8
9  export class PeanutButterCup {
10    constructor() {
11      console.log('PeanutButterCup');
12    }
13  }
14
15 export class KitKat {
16   constructor() {
17     console.log('KitKat');
18   }
19 }
20
21 let o1 = new A.Twix(); // Twix
22 let o2 = new A.PeanutButterCup(); // PeanutButterCup
23 let o3 = new A.KitKat(); // KitKat
24 console.log(A.a); //...error TS2339: Property 'a' does not exist on type 'typeof A'.
```

script.ts



Źródło: <http://stackoverflow.com/questions/30357634/how-do-i-use-namespaces-with-typescript-external-modules>

## Notatki

---



---



---



---



---



---



---



---



---

## Przestrzeń nazewnicza

### Wieloplikowa

```
1 namespace A {
2   export class Twix { ... }
3 }
```

global1.ts

```
1 namespace A {
2   export class PeanutButterCup { ... }
3 }
```

global2.ts

```
1 namespace A {
2   export class KitKat { ... }
3 }
```

global3.ts



### Notatki

---



---



---

```
1 /// <reference path="global1.ts" />
2 /// <reference path="global2.ts" />
3 /// <reference path="global3.ts" />
4 let o1 = new A.Twix();
5 let o2 = new A.PeanutButterCup();
6 let o3 = new A.KitKat();
```

script.ts

## Moduły

### Eksportowanie

```
1 export class Twix {
2   constructor() {
3     console.log('Twix');
4   }
5 }
6 export {Twix as Raider};
```

Mod1.ts

```
1 class PeanutButterCup {
2   constructor() {
3     console.log('PeanutButterCup');
4   }
5 }
6 export {PeanutButterCup};
```

Mod2.ts

```
1 export class KitKat {
2   constructor() {
3     console.log('KitKat');
4   }
5 }
```

Mod3.ts



### Notatki

---



---



---

Źródło: <http://stackoverflow.com/questions/30357634/how-do-i-use-namespaces-with-typescript-external-modules>

## Moduły

### Importowanie

```
1 export class Twix {...}
2 export {Twix as Raider};
```

Mod1.ts

```
1 class PeanutButterCup {...}
2 export {PeanutButterCup};
```

Mod2.ts

```
1 export class KitKat {...}
```

Mod3.ts

```
1 import {Twix, Raider} from './Mod1';
2 import {PeanutButterCup} from './Mod2';
3 import {KitKat} from './Mod3';
4 import {KitKat as KitKatChunKy} from './Mod3';
5 import * as Mars from './Mod1';
6
7 let o1 = new Twix(); // Twix
8 let o2 = new Raider(); // Twix
9 let o3 = new PeanutButterCup(); // PeanutButterCup
10 let o4 = new KitKat(); // KitKat
11 let o5 = new KitKatChunKy(); // KitKat
12 let o6 = new Mars.Twix(); // Twix
13 let o7 = new Mars.Raider(); // Twix
```

script.ts

### Notatki

---



---



---



---



---



---



---



---

## Przestrzenie nazewnicze w modułach

```
1 export namespace A {
2   export class Twix { ... }
3 }
```

Mod1.ts

```
1 export namespace A {
2   export class PeanutButterCup { ... }
3 }
```

Mod2.ts

```
1 export namespace A {
2   export class KitKat { ... }
3 }
```

Mod3.ts



Źródło: <http://stackoverflow.com/questions/30357634/how-do-i-use-namespaces-with-typescript-external-modules>

### Notatki

---



---



---



---



---



---



---



---

## Moduły

Importowanie w stylu „NodeJS”

```
1 class Twix {
2   constructor() {
3     console.log('Twix');
4   }
5 }
6 export = Twix;
```

Mod.ts

```
1 import Twix = require('./Mod');
2 let o = new Twix();
```

script.ts

Notatki

---



---



---

## Tworzenie projektu

Notatki

---



---



---

```
1 {
2   "compilerOptions": {
3     "module": "commonjs",
4     "target": "es5",
5     "noImplicitAny": false,
6     "sourceMap": false
7   },
8   "files": [
9     "script.ts"
10   ]
11 }
```

tsconfig.json



## Pliki deklaracji

## Notatki

```
1 import area = require('./area'); //including 'area.d.ts'
2 let radius = 2;
3 console.log(`The area of the circle with
    radius ${radius} is ${area(radius)}`);
```

script.ts

```
1 function area(radius){
2     return Math.PI*Math.pow(radius,2);
3 }
4 module.exports = area;
```

area.js

```
1 declare function area(radius: number) : number
2 export = area
```

area.d.ts

```
1 "use strict";
2 var area = require("./area");
3 var radius = 2;
4 console.log("The area of the circle with
    radius " + radius + " is " + area(radius)
 );
```

script.js



## Uruchamianie kodu / skryptu Ruby

## Notatki

```
1 #!/usr/bin/env ruby
2 print "Hello World\n"
```

hello.rb

```
1 $ ruby -e 'print "hello world\n"'
2 hello world
3 $ ruby
4 print "Hello World\n"
5 Ctrl+D
6 Hello World
7 $ irb
8 irb(main):001 > print "Hello World\n"
9 Hello World
10 => nil
11 irb(main):002 > exit
12 $ ruby hello.rb
13 Hello World
14 $ chmod 755 hello.rb
15 $ ./hello.rb
16 Hello World
```

Terminal Linux / macOS



## Konwencje nazewnicze dla funkcji / metod

### Notatki

---



---



---



---



---



---



---



---

- ▶ `number.even?()`
- ▶ `string.upcase!()`
- ▶ `car.brand='Fiat'`
- ▶ `array.size()`



## Komentowanie kodu

### Notatki

---



---



---



---



---



---



---



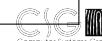
---

#### Osadzane bloki dokumentacji

```

1 =begin
2 = NAZWA
3 Przykładowy dokument RD
4
5 = OPIS
6 ...
7 =end
8
9 =begin html
10 
11 <p>
12   Oto kilka osadzonych HTML. W tym bloku mogę
13   dołączać obrazy, użyć <span style="color: green">
14   style</span> lub cokolwiek innego, co można zapisać
15   w HTML. Procesory RD, które nie wyprowadzają HTML,
16   będą to całkowicie ignorować.
17 </p>
18 =end
19 if a > 2
20   print "a jest większe od 2\n"
21 =begin
22
23   print "Ala ma kota\n"
24
25 =end
26   b = 2
27 end

```



#### Komentarze

```

1 # Ten cały wiersz jest komentarzem
2 a = "#łańcuch znakowy"      #Tu jest komentarz
3 a = /#wyrażenie regularne/ #A tu kolejny komentarz
4 #
5 # Komentarz
6 # Ciąg dalszy komentarza
7 #

```

## Wypisywanie danych

```

1 # 'to_s' is generally intended to return a human-
  readable representation of the object,
  suitable for end users.
2 #'inspect', on the other hand, is intended for
  debugging use, and should return a
  representation that is helpful to Ruby
  developers. The default 'inspect' method,
  inherited from 'Object', simply calls 'to_s'.
3 #Źródło: David Flanagan, Yukihiro Matsumoto, "The
  Ruby Programming Language"
4 ##########
5 print(1.23, [4,5.67], "8\tdziewięć\n")
6 puts
7 printf("%e %s %s\n", 1.23, [4,5.67], "8\tdziewięć")
8 puts
9 puts(1.23, [4,5.67], "8\tdziewięć\n")
10 puts
11 p(1.23, [4,5.67], "8\tdziewięć\n")
12 puts
13 p([{"pozycje": ["pomidor", "banan", "mango"],
      "sumaryczne": 200}, {"opcje_wysylki": ["dwa dni
      ", "jeden dzień"]}, {"metoda_platnosci": ["Gotó
      wka", "Karta kredytowa"]}])
14 puts
15 require "pp" #Ta linia jest wymagana dla wersji Ruby
   < 2.5.0
16 pp(1.23, [4,5.67], "8\tdziewięć\n")
17 puts
18 pp([{"pozycje": ["pomidor", "banan", "mango"],
      "sumaryczne": 200}, {"opcje_wysylki": ["dwa dni
      ", "jeden dzień"]}, {"metoda_platnosci": ["Gotó
      wka", "Karta kredytowa"]}])

```

Dr inż. Stanisław Polak

179

## Wczytywanie danych

Wczytanie pojedynczej wartości

```

1 require 'scanf'
2 puts "Podaj dwie liczby"
3 a,b = scanf("%d%f")
4 printf("a=%d b=%f\n",a,b)
5 #####
6 puts "Podaj znak"
7 znak = STDIN.getc # lub znak = $stdin.getc
8 printf("znak=%c\n",znak)

```

Dr inż. Stanisław Polak

180

## Notatki

```

1 $ irb
2 irb(main):001 > puts 123
3 123
4 => nil
5 irb(main):002 > p 123
6 123
7 => 123
8 irb(main):003 > pp 123
9 123
10 => 123
11 irb(main):004 > puts(123) ? true : false
12 123
13 => false
14 irb(main):005 > p(123) ? true : false
15 123
16 => true
17 irb(main):006 > pp(123) ? true : false
18 123
19 => true

```



## Notatki



## Wczytywanie danych

### Wczytanie pojedynczej linii tekstu

```

1 puts '#####'
2 puts "Wprowadź dane:"
3 dane = gets()
4 # lub
5 # dane = readline()
6 puts '#####'
7 puts "Wprowadzone dane:"
8 print dane
9 puts '-----'
10 p dane
11 puts '#####'

```

### Notatki

---



---



---



---



---



---



---



---



---



---



## Wczytywanie danych

### Wczytanie wielu linii tekstu

```

1 puts '#####'
2 puts "Wprowadź dane:"
3 while(dane = gets())
4   puts '#####'
5   puts "Wprowadzone dane:"
6   puts dane
7   puts '-----'
8   p dane
9   puts '#####'
10  puts "Wprowadź dane:"
11 end

```

```

1 puts '#####'
2 puts "Wprowadź dane:"
3 dane1 = STDIN.read()
4 dane2 = readline()
5 puts '#####'
6 puts "Wprowadzone dane:"
7 puts dane1
8 puts dane2
9 puts '-----'
10 p dane1
11 p dane2
12 puts '#####'

```

### Notatki

---



---



---



---



---



---



---



---



---



---



---

Materiały dla studentów wydziału Informatyki AGH w Krakowie



## Umieszczanie danych w skrypcie

### Notatki

---



---



---



---



---



---



---



---



---

```

1 puts DATA.read
2 __END__
3 puts "Ten kod nie zostanie wykonany"
4 Linia 2
5 Linia 3

```



## Umieszczanie danych w skrypcie

### Przykład zastosowania

### Notatki

---



---



---



---



---



---



---



---



---

```

1 #!/usr/bin/env ruby
2 require 'rdoc'
3 if ARGV[0] == '--man'
4   manual = DATA.read
5   # html = RDoc::Markup::ToHtml.new(RDoc::Options.new)
6   ansi = RDoc::Markup::ToAnsi.new
7   puts ansi.convert(manual)
8 end
9 __END__
10 = Name
11
12 pmh-convert.rb - converts an OAI-PMH document into the DOAJ
13   format; optionally also into the DBLP or PBN.
14
15 = SYNOPSIS
16 pmh-convert.rb [--help] [--man] [--version] [--doajLogin
17   username] [--doajPass password] [startDate] [endDate]
18
19 == Examples
20 * +pmh-convert.rb+ Generate, in the current directory, DOAJ
21   file for the entire period of time, i.e., for all (input
22   ) records
23
24 * <tt>pmh-convert.rb 2013-01-01 2013-07-30</tt> Generate, in
25   the current directory, DOAJ file for a given period of
26   time.
27
28 Author:: Stanisław Polak (mailto:polak@agh.edu.pl)

```

pmh-convert.rb

```

polak@leia:/home/polak
leia[polak]-> ./pmh-convert.rb --man
Name

pmh-convert.rb - converts an OAI-PMH document into the DOAJ format;
optionally also into the DBLP or PBN.

SYNOPSIS

pmh-convert.rb [--help] [--man] [--version] [--doajLogin username] [--doajPass
password] [startDate] [endDate]

Examples

* pmh-convert.rb+ Generate, in the current directory, DOAJ file for
the entire period of time, i.e., for all (input) records
* pmh-convert.rb 2013-01-01 2013-07-30 Generate, in the current
directory, DOAJ file for a given period of time.

Author:
Stanisław Polak (mailto:polak@agh.edu.pl)

leia[polak]->

```

Rysunek: Wyświetlanie podręcznika użytkownika z poziomu konsoli systemowej



## Blok BEGIN oraz END

```

1 puts "Kod skryptu"
2 END {
3   puts "Ten również"
4 }
5 END {
6   puts "Ten kod jest wykonywany tuż po zakończeniu wykonywania
    kodu skryptu"
7 }
8 BEGIN {
9   puts "Ten kod jest wykonywany tuż przed rozpoczęciem
    wykonywania kodu skryptu"
10 }
11 BEGIN {
12   puts "Ten też"
13 }
```

### Notatki

---



---



---



---

```

1 # Obliczanie sumy liczb wczytanych z klawiatury, aż do naciśnięcia Ctrl+D
2 # -n → while($_ = gets) do
3 #   # treść skryptu
4 #   end
5 # -p → while($_ = gets) do
6 #   # treść skryptu
7 #   puts $_
8 #   end
9 $ ruby -ne 'suma += $_.to_i; BEGIN { suma = 0 }; END { puts suma }',
10 1           #$_ = "1\n"
11 2           #$_ = "2\n"
12 10          #$_ = "10\n"
13 CTRL+D
14 13
```

### Przykład użycia w jednoliniówce



## Ładowanie skryptu

### Notatki

---



---



---



---

```

1 if __FILE__ == $PROGRAM_NAME
2   # Zmienna '__FILE__' ma wartość "skrypt.rb"
3   # Zmienna '$PROGRAM_NAME' ma wartość "skrypt.rb"
4   puts "Ten skrypt został uruchomiony z linii komend
      "
5 else
6   # Zmienna '__FILE__' ma wartość "skrypt.rb"
7   # Zmienna '$PROGRAM_NAME' ma wartość "irb"
8   puts "Ten skrypt został załadowany z poziomu
      innego skryptu"
9 end
```

skrypt.rb

```

1 $ irb
2 irb(main):001 > load './skrypt.rb'
3 Ten skrypt został załadowany z poziomu innego
  skryptu
4 => true
5 irb(main):002 > load './skrypt.rb'
6 Ten skrypt został załadowany z poziomu innego
  skryptu
7 => true
8 irb(main):003 > require './skrypt'
9 Ten skrypt został załadowany z poziomu innego
  skryptu
10 => true
11 irb(main):004 > require './skrypt'
12 => false
13 # $ export RUBYLIB=$HOME/skrypty1:$HOME/skrypty2
$LOAD_PATH
=> ['/home/polak/skrypty1", "/home/polak/skrypty2",
    "/home/polak/.rvm/rubies/ruby-2.5.0/lib/ruby/
    site_ruby/2.5.0", ...]
16
17 irb(main):005 > exit
18 $ ruby skrypt.rb
Ten skrypt został uruchomiony z linii komend
```

Konsola 'irb'



## Ruby w systemie Linux / macOS

Menedżery wersji Ruby

### Ruby Version Manager (RVM)

```
1 $ sudo apt-get update
2 $ sudo apt-get install libgdbm-dev libncurses5-dev automake
   libtool bison libffi-dev
3 $ gpg --keyserver hkp://keys.gnupg.net --recv-keys 409
   B6B1796C275462A1703113804BB82D39DC0E3
4 $ curl -sSL https://get.rvm.io | bash -s stable
5 $ source ~/.rvm/scripts/rvm
6 $ echo "source `~/.rvm/scripts/rvm`" >> `~/.bashrc`
```

#### Instalacja

### Rbenv

```
1 cd
2 git clone https://github.com/rbenv/rbenv.git `~/.rbenv
3 echo 'export PATH="$HOME/.rbenv/bin:$PATH"' >> `~/.bashrc
4 echo 'eval "$($rbenv init -)"' >> `~/.bashrc
5 exec $SHELL
6
7 git clone https://github.com/rbenv/ruby-build.git `~/.rbenv/
   plugins/ruby-build
8 echo 'export PATH="$HOME/.rbenv/plugins/ruby-build/bin:$PATH"
   >> `~/.bashrc
9 exec $SHELL
```

#### Instalacja

### Notatki

#### Interpreter Ruby

```
1 $ rvm install <nazwaInterpretera>
2 $ rbenv install <nazwaInterpretera>
```

#### Instalacja

```
1 $ rvm list known #wyświetlenie (nazw) dostępnych implementacji
   wraz z numerami ich wersji
2 $ rvm upgrade <nazwaImplementacji> # np. rvm upgrade ruby
3
4 $ rbenv install --list #wyświetlenie (nazw) dostępnych
   implementacji wraz z numerami ich wersji
5 $ rbenv install <nazwaInterpretera>
```

#### Aktualizacja

```
1 $ rvm use <nazwaInterpretera> --default #lub rvm use x.x.x --
   default
2 $ rbenv global <nazwaInterpretera>
```

#### Okręsianie domyślnego interpretera



## Ruby w systemie Linux / macOS

Ważniejsze implementacje

### Implementacje w języku C (CRuby)

- ▶ MRI (ang. Matz's Ruby Interpreter)
  - ▶ Używany w Ruby  $\leq 1.8$
  - ▶ Tworzono jest drzewo składniowe
  - ▶ Korzysta z „zielonych wątków”
- ▶ YARV (Yet Another Ruby VM)
  - ▶ Używany w Ruby  $\geq 1.9$
  - ▶ Wirtualna maszyna Ruby
  - ▶ Korzysta z wątków natywnych; użycia GIL (Global Interpreter Lock)
  - ▶ Oferuje kompilator JIT (Just-in-time) — począwszy od wersji 2.6
  - ▶ Instalacja

```
1 $ rvm install ruby # instalacja najnowszej wersji
   produkcyjnej
2 $ rvm install ruby-head # instalacja najnowszej
   wersji deweloperskiej (testowej)
3 $ rvm install x.x.x # instalacja wersji o numerze
   x.x.x
```

### Notatki

#### Inne

- ▶ JRuby
  - ▶ Implementacja w Javie
  - ▶ Używa wątków natywnych, brak GIL
  - ▶ Używa kompilatora JIT
  - ▶ Instalacja
- 1 \$ rvm install jruby
- ▶ TruffleRuby
  - ▶ Korzysta z:
    - ▶ Truffle — platforma do implementacji wysokowydajnych środowisk językowych korzystających z Javy oraz wirtualnej maszyny Javy
    - ▶ GraalVM — wirtualna maszyna Javy wyposażona w kompilator JIT o nazwie „Graal”
  - ▶ Źródła: <http://www.oracle.com/technetwork/oracle-labs/program-languages/downloads/>
- ▶ Informacje o innych implementacjach — <http://rvm.io/interpreters/>



## Ruby w systemie Linux / macOS

Porównanie wydajności implementacji

```

1 def fibonacci(n)
2   if n < 2
3     return n
4   end
5   return fibonacci(n - 2) + fibonacci(n - 1)
6 end
7 #####
8 def wątki()
9   i = 100_000_000
10  2.times.map do
11    Thread.new do
12      50_000_000.times { i -= 1 }
13    end
14  end.each(&join) #Czekamy aż wszystkie wątki zakończą pracę
15  # Znaczenie '&:nazwa' - patrz slajd nr 234BlokiDoc-Start
16  printf("Końcowa wartość i: %d\n", i)
17 end

```

funkcje.rb

```

1 require 'benchmark'
2 load 'funkcje.rb'
3 #####
4 czas_wykonania = Benchmark.realtime do |x|
5   5.times { fibonacci(35) }
6 end
7 printf "Czas wykonania: %f\n", czas_wykonania
8 #####
9 wątki()

```

skrypt.rb

```

1 $ rvm use ruby-2.6.1
2 $ time ruby skrypt.rb
3 Czas wykonania: 10.255780
4 Końcowa wartość i: 0
5
6 real 0m10.201s
7 user 0m19.179s
8 sys 0m0.013s
9 $ time ruby --jit skrypt.rb # Włączona komplikacja JIT
10 Czas wykonania: 3.352901
11 Końcowa wartość i: 0
12
13 real 0m9.019s
14 user 0m17.473s
15 sys 0m0.654s
16 #####
17 $ rvm use 9.2.6.0 # OpenJdk wersja "11.0.1"
18 $ time ruby skrypt.rb
19 Czas wykonania: 6.112136
20 Końcowa wartość i: 46414285
21
22 real 0m16.994s
23 user 0m33.553s
24 sys 0m0.313s
25 $ time ruby -X-C skrypt.rb # Wyłączona komplikacja JIT
26 Czas wykonania: 52.742840
27 Końcowa wartość i: 9910369
28
29 real 1m22.684s
30 user 1m56.812s
31 sys 0m0.340s
32 #####
33 $ rvm use truffleruby #GraalVM wersja "1.0.0-rc13"
34 $ time ruby skrypt.rb
35 Czas wykonania: 2.040543
36 Końcowa wartość i: 38358376
37
38 real 0m4.596s
39 user 0m7.361s
40 sys 0m0.157s

```

## Notatki

## RubyGems

## Notatki

- ▶ Standardowy menedżer pakietów (gemów) Rubiego — działa podobnie jak apt, yum, ...
- ▶ Dostępny wraz z Ruby  $\geq 1.9$

### Przykłady użycia

```

1 gem install rails #zainstaluj gem 'Ruby on Rails'
2 gem fetch rails #pobierz gem 'Ruby on Rails', ale go nie instaluj
3 gem uninstall rails #usuń gem 'Ruby on Rails'
4 gem list --local #wyświetl zainstalowane gemy
5 gem list --remote #wyświetl dostępne gemy
6 gem search NAPIS --remote #przeszukaj listę dostępnych gemitów
7 gem rdoc --all #stwórz dokumentację RDoc/RI dla wszystkich zainstalowanych gemitów
8 gem cleanup #usuń stare wersje zainstalowanych gemitów

```

## Tworzenie i generowanie dokumentacji

### Program „rdoc”

```

1 # The program takes an initial word or phrase from
2 # the command line (or in the absence of a
3 # parameter from the first line of standard
4 # input). It then reads successive words or
5 # phrases from standard input and reports whether
6 # they are anagrams of the first word.
7 #
8 # Author:: Dave Thomas (mailto:dave@x.y)
9 # Copyright:: Copyright (c) 2002 The Pragmatic Programmers,
10 # LLC
10 # Licensee:: Distributes under the same terms as Ruby
11 #
12 # This class holds the letters in the original
13 # word or phrase. The is_anagram? method allows us
14 # to test if subsequent words or phrases are
15 # anagrams of the original.
16 class Anagram
17
18 # Remember the letters in the initial word
19 def initialize(text)
20   @initial_letters = letters_of(text)
21 end
22
23 # Test to see if a new word contains the same
24 # letters as the original
25 def is_anagram?(text)
26   @initial_letters == letters_of(text)
26

```

```

27 # Determine the letters in a word or phrase
28 #
29 # * all letters are converted to lower case
30 # * anything not a letter is stripped out
31 # * the letters are converted into an array
32 # * the array is sorted
33 # * the letters are joined back into a string
34 def letters_of(text)
35   text.downcase.delete('a-z').split('').sort.join
36 end
37
38
39 tester = Anagram.new(ARGV.shift || gets)
40 ARGF.each do |text|
41   puts "Anagram! " if tester.is_anagram? text
42 end

```

anagram.rb

## Wyświetlanie manuala

### Program „ri”

```

1 $ ri --help
2 Usage: ri [options] [names...]
3
4 Where name can be:
5
6 Class | Module | Module::Class
7
8 Class::method | Class#method | Class.method | method
9
10 gem_name: | gem_name:README | gem_name:History
11
12 All class names may be abbreviated to their minimum unambiguous form. If a name
13 is ambiguous, all valid options will be listed.
14
15 A '.' matches either class or instance methods, while #method
16 matches only instance and ::method matches only class methods.
17
18 README and other files may be displayed by prefixing them with the gem name
19 they're contained in. If the gem name is followed by a ':' all files in the
20 gem will be shown. The file name extension may be omitted where it is
21 unambiguous.
22
23 For example:
24
25   ri Fil
26   ri File
27   ri File.new
28   ri zip
29   ri rdoc:README
30
31 Note that shell quoting or escaping may be required for method names containing
32 punctuation:
33
34   ri 'Array.[]'
35   ri compact!
36
37 To see the default directories ri will search, run:
38
39   ri --list-doc-dirs
40 ...

```

### Notatki



### Notatki



Materiały dla studentów wydziału Informatyki AGH w Krakowie

## Debugowanie skryptu

```

1 a=1
2 b=2
3 c=3
4 puts a
5 puts b
6 puts c

```

skrypt.rb

```

1 $ gem install byebug
2 $ byebug skrypt.rb
3 [1, 6] in /home/polak/skrypt.rb
4 => 1: a=1
5   2: b=2
6   3: c=3
7   4: puts a
8   5: puts b
9   6: puts c
10  (byebug) display a
11  1: a = nil
12  (byebug) next
13
14 [1, 6] in /home/polak/skrypt.rb
15   1: a=1
16 => 2: b=2
17   3: c=3
18   4: puts a
19   5: puts b
20   6: puts c
21  1: a = 1

```

Debugger „Byebug”

## Notatki



## Alternatywna konsola Ruby

### Program „pry”

#### Wybrane cechy

- ▶ Przeglądanie kodu źródłowego
- ▶ Podświetlanie składowi
- ▶ Przeglądanie dokumentacji
- ▶ System pomocy, na żywo
- ▶ Integracja powłoki poleceń (uruchamianie edytorów, uruchomianie 'git' oraz 'rake' z poziomu 'pry')
- ▶ Możliwość debugowania kodu

```

1 a=1
2 binding.pry #pierwsza pułapka
3 b=2
4 c=3
5 binding.pry #druga pułapka
6 puts a
7 puts b
8 puts c

```

skrypt.rb

```

1 $ gem install pry      #Instalacja 'pry'
2 $ gem install pry-doc #Rozszerzona obsługa dokumentacji dla '
3   pry'
4 [1] pry(main)> puts "Hello World"
5 Hello World
6 => nil
7 [2] pry(main)> exit
8 $ ruby -r pry skrypt.rb #Uruchomienie w trybie ,,debug'',
9   1: a=1
10  => 2: binding.pry #pierwsza pułapka
11  3: b=2
12  4: c=3
13  5: binding.pry #druga pułapka
14  6: puts a
15  7: puts b
16  8: puts c
[1] pry(main)> a
17 => 1
18 [2] pry(main)> b
19 => nil
20 [3] pry(main)> Ctrl+D
From: /home/polak/skrypt.rb @ line 5 :
21
22   1: a=1
23   2: binding.pry #pierwsza pułapka
24   3: b=2
25   4: c=3
26 => 5: binding.pry #druga pułapka
27   6: puts a
28   7: puts b
29   8: puts c
30   9:
[1] pry(main)>

```



## Definiowanie zmiennych

```

1 #stałe
2 STALA=1
3 =>1
4 STALA=2
5 (irb):2: warning: already initialized constant STALA
6 => 2
7 #definiowanie zmiennych
8 a = 10
9 b = 20
10 c = 30
11 d = 40
12 a, b, c, d = 10, 20, 30, 40
13 #identyfikacja typu
14 d.kind_of? Integer
15 => true
16 d.class
17 => Integer
18 #zmienna typu
19 x = 10
20 => 10
21 x.class
22 => Integer
23 x = "hello"
24 => "hello"
25 x.class
26 => String

```

### Notatki

---



---



---



---



---



---

```

1 y = 20
2 => 20
3 y.to_f
4 => 20.0
5 54321.to_s #kod dziesiątkowy
6 => "54321"
7 54321.to_s(2) #kod dwójkowy
8 => "1101010000110001"
9 54321.to_s(16) #kod szesnastkowy
10 => "d431"
11 54321.to_s(8) #kod ósemkowy
12 => "152061"

```

### Konwersja wartości

---



---



---



---



---



---

## Zasięg zmiennych

### Notatki

---



---



---



---



---



---

```

1 x = 10
2 => 10
3 defined? x
4 => "local-variable"
5 $x = 10
6 => 10
7 defined? $x
8 => "global-variable"
9 loopcounter = 10
10 _LoopCounter = 20
11 $welcome = "Welcome to Ruby"
12 gets
13 hello
14 => "hello\n"
15 $_
16 => "hello\n"
17 require 'English'
18 => true
19 $LAST_READ_LINE
20 => "hello\n"
21 $$
22 => 17403
23 @total = 0
24 @total = 10
25 Stala=1

```

## Instrukcja warunkowa if

### Wyrażenie „if”

```
1 if warunek [then]
2   kod...
3 [elsif warunek [then]
4   kod...]...
5 [else
6   kod...]
```

Składnia

```
1 if 10 < 20 then
2   print "10 jest mniejsze niż 20"
3 end
```

Przykład użycia

### Modyfikator „if”

```
1 kod if warunek
```

Składnia

```
1 print "10 jest mniejsze niż 20" if 10 < 20
```

Przykład użycia

Notatki

---



---



---



---



---

```
1 liczba = 5
2 znak = if liczba > 0
3   'dodatni'
4   elsif liczba < 0
5   'ujemny'
6   else
7     'zero'
8 end
9 znak
10 => "dodatni"
```



## Instrukcja warunkowa unless

### Wyrażenie „unless”

```
1 unless warunek [then]
2   kod
3 [else
4   kod ]
5 end
```

Składnia

### Modyfikator „unless”

```
1 kod unless warunek
```

Składnia

```
1 unless str.nil?
2   puts str.length
3 end
4 # W Ruby >= 2.3 powyższe linie można zapisać nastę-
  pująco:
5 # puts str.length
```

Przykład użycia

```
1 puts str.length unless str.nil?
```

Przykład użycia

Notatki

---



---



---



---



---



## Instrukcja wyboru

### Notatki

```

1 ##### Forma 1 #####
2 case wyrażenie
3 [when wyrażenie [, wyrażenie ...] [then]
4   kod ]...
5 [else
6   kod ]
7 end
8
9 ##### Forma 2 #####
10 case
11 [when wyrażenie_logiczne [, wyrażenie_logiczne ...] [then]
12   kod ]...
13 [else
14   kod ]
15 end

```

### Składnia

```

1 liczba = 7
2 case liczba
3 when 1,2,3 # Wykonuje: 1 === liczba || 2 === liczba || 3 ===
4   liczba
5   "mały"
6 when 4,5,6
7   "średni"
8 when 7,8,9
9   "duży"
10 else
11   "olbrzymi"
12 end
13 => "duży"
14 a = 2
15 case
16 when a == 1, a == 2
17   "a ma wartość jeden lub dwa"
18 when a == 3
19   "a ma wartość trzy"
20 else
21   "Nie potrafię określić wartości zmiennej a"
22 end
23 => "a ma wartość jeden lub dwa"
24 #####
25 case url
26 when /http:/
27   "protokół HTTP"
28 ...
29 else
30   "nieznany protokół"
31 end

```

### Przykład użycia



## Dopasowywanie wzorców

### Notatki

*Dopasowywanie wzorców polega na określeniu wzorców, z którymi powinny się zgadzać niektóre dane, a następnie sprawdzeniu, czy tak się dzieje, i dekonstrukcji danych według tych wzorców.*

Learn You a Haskell for Great Good!  
Miran Lipovaca.

```

1 case [zmienna lub wyrażenie]
2 in wzorzec [if | unless warunek]
3 ...
4 in wzorzec [if | unless warunek]
5 ...
6 else
7 ...
8 end

```

### Składnia

```

1 case [0, 1]
2 in [a,b] unless a == b
3   puts "a różne od b"
4 in [a,b] if a == b
5   puts "a równe b"
6 end
7 printf("a=%d b=%d\n", a, b)

```

### Na wyjściu

a różne od b  
a=0 b=1

### Przykład



## Dopasowywanie wzorców

Operator =>

Notatki

---



---



---

```
1 <wyrażenie> => < wzorzec >
```

Składnia

```
1 config1 = {db: {user: 'admin', password: 'abc123'}}
2 config2 = {db: {password: 'abc123'}}
3 config1 => {db: {user:}}
4 puts "Connect with user '#{user}'"
5 config2 => {db: {user:}}
```

Przykład

Na wyjściu

```
Connect with user 'admin'
script.rb:5:in '<main>': {:db=>{:password=>"abc123"}}: key not found: :user
(NoMatchingPatternKeyError)
```



## Dopasowywanie wzorców

Przykłady

Notatki

---



---



---

Dopasowywanie tablic

```
1 translation = ['pl', 'Język', 'en', 'Language']
2
3 case translation
4   in ['pl', orig_text, 'fr', trans_text]
5     puts "Tłumaczenie francuskie:\n\t#{orig_text} =>
#{trans_text}"
6   in ['pl', orig_text, 'en', trans_text]
7     # To zostanie wykonane
8     puts "Tłumaczenie angielskie:\n\t#{orig_text} =>
#{trans_text}"
9 end
```

Na wyjściu

Tłumaczenie angielskie:  
Język => Language

Dopasowywanie hasy

```
1 translation = {orig_lang: 'pl', trans_lang: 'en',
orig_txt: 'Język', trans_txt: 'Language' }
2
3 case translation
4   in {orig_lang: 'pl', trans_lang: 'en', orig_txt:
orig_txt, trans_txt: trans_txt}
5     puts "#{orig_txt} => #{trans_txt}"
6 end
```

Na wyjściu

Język => Language



## Dopasowywanie wzorców

### Przykład zastosowania

#### Z użyciem mechanizmu dopasowania wzorca

```

1 require 'json'
2
3 json = <<-EndJSON
4 {
5   "name": "Alice",
6   "age": 30,
7   "children": [
8     {
9       "name": "Bob",
10      "age": 2
11    }
12  ]
13}
14 EndJSON
15
16 person = JSON.parse(json,symbolize_names:true)
17 puts person
18 case person
19   in {name: "Alice", children:[{name: "Bob",
20     age: age}]}
21   p age
22 end

```

#### Na wyjściu

```
{:name=>"Alice", :age=>30, :children=>[{:name=>"Bob", :age=>2}]}  
2
```

## Pętle while oraz until

```

1 while warunek [do]
2   kod
3 end
4
5 begin
6   kod
7 end while warunek
8
9 kod while warunek

```

Składnia pętli „while”

```

1 until warunek [do]
2   kod
3 end
4
5 until while warunek
6 begin
7   kod
8 end until warunek

```

Składnia pętli „until”

### Notatki

#### Bez użycia mechanizmu dopasowania wzorca

```

1 require 'json'
2
3 json = <<-EndJSON
4 {
5   "name": "Alice",
6   "age": 30,
7   "children": [
8     {
9       "name": "Bob",
10      "age": 2
11    }
12  ]
13}
14 EndJSON
15
16 if person[:name] == "Alice"
17   children = person[:children]
18   if children.length == 1 && children[0][:name]
19     == "Bob"
20     p children[0][:age]
21   end
22 end

```



### Notatki

```

1 line=nil
2 while line =~ /^$/ do line = gets.chomp end
3 #####
4 i=1
5 while i < 3
6   puts i
7   i += 1 #W Ruby nie ma instrukcji: '++zmienna' / 'zmienna--', itp.
8   j = 2
9 end
10 puts j #Wypisze: 2
11 #####
12 i=1
13 begin
14   puts i
15   i += 1
16   j = 2
17 end while i < 3 #Ta konstrukcja nie jest zalecana przez Matz-a
18 puts j #Wypisze: 2
19 #####
20 line = gets.chomp while line =~ /^$/

```

```

1 line=nil
2 until line =~ /^$/ do line = gets.chomp end
3 #####
4 i=1
5 until i >= 3 do
6   puts i
7   i += 1
8   j = 2
9 end
10 puts j #Wypisze: 2
11 #####
12 i=1
13 begin
14   puts i
15   i += 1
16   j = 2
17 end until i >= 3 #Ta konstrukcja nie jest zalecana przez Matz-a
18 puts j #Wypisze: 2
19 #####
20 line = gets.chomp until line =~ /^$/
```

## Instrukcje break, redo oraz next

Notatki

```

1 i = 0
2 while p "Wyrażenie 'while'"
3   i += 1
4   printf "\ti=%d\n", i
5   case i
6   when 2
7     puts "\tWykonuję 'next'"
8     next
9   when 4
10    puts "\tWykonuję 'redo'"
11    redo
12   when 6
13    puts "\tWykonuję 'break'"
14    break
15  end
16  puts "\tJestem na końcu 'while'"
17 end
18 puts 'Koniec'

```



## Pętla loop

Notatki

```

1 #Pętla nieskończona
2 loop do
3   puts "Witaj"
4 end
5
6 #Pętla skończona
7 i = 1
8 loop do
9   puts i
10  i += 1
11  j = 2
12  break if i > 3
13 end
14 puts j #undefined local variable or method 'j'

```



## Pętla for oraz metody iteracyjne

```

1 for i in 1..8 do
2   puts i
3 end
4 # Powyższy kod jest równoważny:
5 # (1..8).each{ |i| puts i }
6 #####
7 strs = ["abc\n", "cdæ\n", "zzas\n", "aaaa\n"]
8 strs.each{ |s| s.chop! }
9 p strs
10 #####
11 5.times { |i| puts i }
12 #####
13 2.upto(4) { |i| puts i }
14 #####
15 4.downto(2) { |i| puts i }
16 #####
17 0.step(10, 2) { |i| puts i }
18 #####
19 0.step(10, 2) do |i|
20   puts i
21   redo
22 end

```

### Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---

## Typy liczbowe

### ► Integer

```

1 2 - 3 * 2
2 => -4
3 1.hour + 90.minutes + 1.minute #Dostępne w RoR lub po
   wykonaniu "require 'active_support/all'"
4 => 1 hour and 91 minutes
5 11111111111111111111111111*400
6 => 4444444444444444444444400

```

### ► Float

### ► Rational

```

1 Rational(1,3) + Rational(1,2) # $\frac{1}{3} + \frac{1}{2} = \frac{5}{6}$ 
2 => (5/6)
3 0.1r #Składnia dostępna począwszy od wersji 2.1
4 => (1/10)
5 0.1r * 3
6 => (3/10)

```

### ► Complex

```

1 Complex(1,2)+Complex(2,3) # <=> (1+2i)+(2+3i)=(3+5i)
2 => (3+5i)
3 42ri #Składnia dostępna począwszy od wersji 2.1
4 => (0+(42/1)*i)

```

### Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---

### ► Konwersje

```

1 Integer(10.898)
2 => 10
3 Integer(0b01110101) #konwersja z kodu binarnego
4 => 117
5 Integer(01231) #konwersja z kodu ósemkowego
6 => 665
7 Integer(0xaabb) #konwersja z kodu szesnastkowego
8 => 43707
9 Float(10)
10 => 10.0
11 Integer("aaa")
12 ... invalid value for Integer(): "aaa" ...
13 Float("aaa")
14 ... invalid value for Float(): "aaa" ...

```



## Wyrażenia regularne

### Notatki

---



---



---



---



---



---



---



---



---



---



---

```

1 name = "Matz"
2 name =~ /([[:lower:]]{2})/
3 => 1
4 $
5 => "M"
6 $1
7 => "at"
8 $
9 => "z"
10 # Inne sposoby konstrukcji wyrażenia regularnego
11 name =~ %r|([[:lower:]]{2})|
12 => 1
13 re = Regexp.new("([[:lower:]]{2})")
14 md = re.match(name)
15 => #<MatchData:0xb7d41054>
16 md.pre_match
17 => "M"
18 md.value_at 1
19 => "at"
20 md.post_match
21 => "z"

```



## Przedziały

### Notatki

---



---



---



---



---



---



---



---



---



---



---

```

1 .. 10    # Zakres od 1 do 10
2 1...10   # Zakres od 1 do 9
3
4 (1..10).to_a # => [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
5 (1...10).to_a # => [1, 2, 3, 4, 5, 6, 7, 8, 9]
6
7 ('a'..'l').to_a # => ["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l"]
8 ('Z'..'a').to_a # => ["Z", "!", "\\", "]", "^", "_", "c", "a"]
9
10 ('cab'...'car').to_a # => ["cab", "cac", "cad", "cae", "caf", "cag", "cah", "cai", "caj", "cak", "cal", "cam", "can", "cao", "cap", "caq", "car"]

```



## Przedziały

### Metody

#### Notatki

```

1 words = 'cab'...'car'
2 words.min
3 => "cab"
4 words.max
5 => "car"
6 words.include?('can')
7 => true
8 words.reject{|subrange| subrange < 'cal'}
9 => ["cal", "cam", "can", "cao", "cap", "caq", "car"]
10 words.each{|word| puts "Hello " + word}
11 Hello cab
12 Hello cac
13 ...
14 Hello car

```



## Przedziały

### Przykłady zastosowania

#### Notatki

```

1 while input = gets
2   puts input + " triggered" if input =~ /start/..
3 end

```

```

1 score = 70
2 result =
3 case score
4 when 0..40: "Nie zaliczył (oblał test)"
5 when 41..60: "Zaliczył"
6 when 61..70: "Zaliczył z wysokim wynikiem"
7 when 71..100: "Zaliczył z wyróżnieniem"
8 else "Nieprawidłowy wynik testu"
9 end
10 puts result

```



## Tablice zwykłe

```

1 days_of_week = Array.new(7)
2 => [nil, nil, nil, nil, nil, nil]
3 days_of_week = Array.new(7, "today")
4 => ["today", "today", "today", "today", "today", "today", "today"]
5 days_of_week = Array[ "Mon", "Tues", "Wed", "Thu", "Fri", "Sat", "Sun" ]
6 #lub
7 days_of_week = [ "Mon", "Tues", "Wed", "Thu", "Fri", "Sat", "Sun" ]
8 => [ "Mon", "Tues", "Wed", "Thu", "Fri", "Sat", "Sun" ]
9 days_of_week.empty?
10 => false
11 days_of_week.size
12 => 7
13 days_of_week.length
14 => 7
15 days_of_week[0]
16 => "Mon"
17 days_of_week.at(0)
18 => "Mon"
19 days_of_week.first
20 => "Mon"
21 days_of_week[0].class
22 => String
23 days_of_week.index("Wed")
24 => 2
25 days_of_week[1..3]
26 => [ "Tues", "Wed", "Thu" ]
27 days_of_week[1..3]
28 => [ "Tues", "Wed", "Thu" ]

```

### Notatki



## Tablice zwykłe c.d.

```

1 matrix = [[1,2,3],[3,4,5],[5,6,7]]
2 matrix[1]
3 => [3, 4, 5]
4 matrix[1][1]
5 => 4
6 matrix[1,1]
7 => [3, 4, 5]
8 #####operating_systems#####
9 operating_systems = ["Fedora", "SuSE", "RHEL", "Windows", "macOS"]
10 linux_systems = ["RHEL", "SuSE", "PCLinuxOS", "Ubuntu", "Fedora"]
11 operating_systems | linux_systems
12 => ["Fedora", "SuSE", "RHEL", "Windows", "macOS", "PCLinuxOS", "Ubuntu"]
13 operating_systems & linux_systems
14 => ["Fedora", "SuSE", "RHEL"]
15 operating_systems - linux_systems
16 => ["Windows", "macOS"]

```

### Notatki

```

17 days1 = ["Mon", "Tue", "Wed"]
18 days2 = ["Thu", "Fri", "Sat", "Sun"]
19 days = days1 + days2
20 days
21 => ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
22 days1
23 => ["Mon", "Tue", "Wed"]
24 #####
25 days1 = ["Mon", "Tue", "Wed"]
26 days2 = ["Thu", "Fri", "Sat", "Sun"]
27 days = days1.concat(days2)
28 => ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
29 days
30 => ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
31 days1
32 => ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
33 #####
34 days1 = ["Mon", "Tue", "Wed"]
35 days2 = ["Thu", "Fri", "Sat", "Sun"]
36 days = days1 << days2
37 days
38 days
39 => ["Mon", "Tue", "Wed", ["Thu", "Fri", "Sat", "Sun"]]
40 days1
41 => ["Mon", "Tue", "Wed", ["Thu", "Fri", "Sat", "Sun"]]

```



## Tablice asocjacyjne

Notatki

```

1 hash = { "jeden" => 1, "dwa" => 2, "trzy" => 3 }
2 hash["jeden"]
3 => 1
4 hash1 = {jeden:1, dwa:2, trzy:3} #Składnia dostępna począwszy od wersji 1.9
5 => {jeden=>1, :dwa=>2, :trzy=>3}
6 hash1[:jeden]
7 => 1
8 histogram = Hash.new(0)
9 => {}
10 histogram["Ala"]
11 => 0
12 histogram["Ala"] += 1
13 => 1
14 histogram["Ala"]
15 => 1
16 hash.key?("jeden")
17 => true
18 hash.value?("jeden")
19 => false

```



## Uporządkowane tablice asocjacyjne

Notatki

```

1 h = {"c" => 3, "a" => 1, "b" => 2}
2 h.each do |k, v|
3   p [k, v]
4 end
5
6 #Wynik działania:
7 #      Ruby 1.9+
8 ["c", 3]                                     Ruby 1.8
9 ["a", 1]                                       ["a", 1]
10 ["b", 2]                                      ["b", 2]
11                                            ["c", 3]

```

```

1 h = {}
2 h["a"] = "foo"
3 h["b"] = "bar"
4 p h
5 # Na wyjściu:
6 {"a"=>"foo", "b"=>"bar"}
7
8 h["a"] = nil
9 p h
10 #Na wyjściu:
11 "a"=>nil "b"=>"bar"

```



## Łańcuchy znaków

### Tworzenie i formatowanie

```

1 ##### tworzenie łańcucha #####
2 myString = String.new
3 => ""
4 myString = String.new("This is my string. Get your own string")
5 myString = String("This is also my string") #metoda 'String'
   jest metodą modułu "Kernel" - konwersja do typu
   napisowego
6 myString = "This is also my string"
7 myString = 'This is also my string'
8 ##### wypisanie łańcucha #####
9 '#{myString}#{myString}' lub %q{myString} #{i+3}& lub %&#
  => "This is also my string 4"
10 '#{myString}' lub %q{myString}& lub %q{#{myString}}
11 => "#{myString}"
12 'ls' lub %x{ls} lub %x{ls}
13 => Wynik wykonania komendy 'ls'
14 ##### składnia HereDocs #####
15 myText= <<DOC
16 1
17 #{{myString}}
18 DOC
19
20 DOC
21 => "1\nThis is also my string\n      DOC\n"
22 myText= <<DOC
23 1
24 #{{myString}}
25 DOC
26 => "1\nThis is also my string\n"

```

```

27 ##### Tworzenie tablicy napisów #####
28 %w(1 2 aaa)
29 => ["1", "2", "aaa"]
30 ##### Formatowanie łańcucha #####
31 a=1
32 b=2
33 "a=%d" % a
34 => "a=1"
35 "a=%d, b=%d" % [a,b]
36 => "a=1, b=2"
37 "c=%{c}, d=%{d}" % {c:>3,:d=>4}
38 => "c=3,d=4"
39 ##### Wybrane metody #####
40 myString = "Hello"
41 myString.length
42 => 5
43 myString.size
44 => 5
45 myString.empty?
46 => false
47 myString.sub('H','h')
48 => "hello"
49 myString
50 => "Hello"
51 myString.gsub('H','h')
52 => "hello"
53 myString
54 => "hello"
55 myString.object_id
56 => 9565120
57 myString.sub!('h','H')
58 => "Hello"
59 myString.object_id
60 => 9565120

```

### Notatki

## Łańcuchy znaków

### Łączenie, dostęp do fragmentów oraz porównywanie

```

1 ##### łączenie #####
2 myString = "Welcome " + "to " + "Ruby!" #konkatenacja (nie modyfikuje łańcucha z lewej)
3 => "Welcome to Ruby!"
4 myString = "Welcome " "to " "Ruby!" #jw.
5 => "Welcome to Ruby!"
6 myString = "Welcome ".concat("to ").concat("Ruby!") #dołącza do łańcucha z lewej (modyfikuje go)
7 => "Welcome to Ruby!"
8 myString = "Welcome " << "to " << "Ruby!" #jw.
9 => "Welcome to Ruby!"
10 ##### dostęp do fragmentów #####
11 myString["Ruby"]
12 => "Ruby"
13 myString["Perl"]
14 => nil
15 myString[3] #Ruby 1.9+
16 => "c"
17 myString[3].ord #Ruby 1.9+
18 => 99
19 myString[3].chr #Ruby 1.8
20 => "c"
21 myString[11, 4]
22 => "Ruby"
23 myString[0..6]
24 => "Welcome"
25 myString[/[A-Z]/]
26 => "W"
27 myString.index('Ruby')
28 => 11

```

### Notatki

## Łańcuchy znaków

Zastępowanie, substytucja, wstawianie

### Notatki

```

1 ##### podmiana #####
2 myString = "Welcome to JavaScript!"
3 myString["JavaScript"] = "Ruby"
4 puts myString
=> "Welcome to Ruby!"
5 myString = "Welcome to JavaScript!"
6 myString[10] = "Ruby"
7 puts myString
=> "Welcome toRubyJavaScript!"
8 myString = "Welcom to JavaScript!"
9 myString[8..20] = "Ruby"
10 puts myString
=> "Welcome Ruby!"
11 ##### substytucja #####
12 myString = "Welcome to PHP"
13 myString.gsub("PHP", "Ruby")
14 => "Welcome to Ruby"
15 myString.replace("Goodbye to PHP!")
16 => "Goodbye to PHP!"
17 ##### powielanie #####
18 myString * 3
19 => "Goodbye to PHP!Goodbye to PHP!Goodbye to PHP!"
20 ##### wstawianie #####
21 myString1 = "Paris in Spring"
22 myString2 = "Paris in Spring"
23 myString1[8] = "the" # podmiana, a nie wstawienie przed
24 => "the"
25 myString1
26 => "Paris in theSpring"
27 myString2.insert(8, " the")
28 => "Paris in the Spring"
29 myString2
30 => "Paris in the Spring"
31 myString2
32 => "Paris in the Spring"
33

```



## Łańcuchy znaków

Konwersja

### Notatki

```

1 myArray = Array("ABCDEFGHIJKLMNP")
2 => ["ABCDEFGHIJKLMNP"]
3 myArray = "ABCDEFGHIJKLMNP".split
4 => ["ABCDEFGHIJKLMNP"]
5 myArray = "ABCDEFGHIJKLMNP".split("//")
6 => ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P"]
7 "1000".to_i
8 => 1000
9 Integer("aaa")
10 ArgumentError: invalid value for Integer(): "aaa"
11 "aaa".to_i #Nie wyrzuca wyjątku
12 => 0

```



## Łańcuchy znaków

Niemodyfikowalne napisy

### Notatki

---



---



---



---



---



---



---



---

```

1 # frozen_string_literal: true
2 napis = "Stanisław"
3 puts napis << " Polak" #RuntimeError: can't modify frozen String
4 puts napis.dup << " Polak" #Wypisze: Stanisław Polak

```



## Symbole

```

1 #Tworzenie symboli
2 s1 = :Polak
3 s2 = :"Polak"
4 %s(Stanisław Polak)
5 => :"Stanisław Polak"
6 #Tworzenie tablicy symboli
7 %i(Stanisław Polak) #Dostępne w Ruby 2.0+
8 => [:Stanisław, :Polak]
9 #####
10 str1 = "Stanisław"
11 str2 = "Stanisław"
12 s1.object_id == s2.object_id
13 => true
14 str1.object_id == str2.object_id
15 => false
16 #####
17 "Stanisław" << " Polak"
18 => "Stanisław Polak"
19 :"Stanisław" << :" Polak"
20 NoMethodError: undefined method `<<' for :Stanisław:Symbol
21 #####
22 :"Stanisław Polak".to_s
23 => "Stanisław Polak"
24 :"Stanisław Polak".to_sym
25 => :"Stanisław Polak"
26 #####
27 :"Stanisław Polak".length # => :"Stanisław Polak".to_s.length
28 => 15
29 :Polak.gsub('P','p')
30 NoMethodError: undefined method `gsub' for :Polak:Symbol

```

### Notatki

---



---



---



---



---



---



---



---

```

1 key = "Jabłko"
2 => "Jabłko"
3 histogram[key] = 10
4 => 10
5 histogram[key]
6 => 10
7 key.gsub!("o", "a")
8 => "Jabłka"
9 key
10 => "Jabłka"
11 histogram[key]
12 => nil
13 histogram["Jabłko"]
14 => 10
15 #####
16 key=:Jabłko
17 => :Jabłko
18 histogram[key] = 10
19 => 10
20 key.gsub!("o", "a")
21 NoMethodError: undefined method `sub!' for :Jabłko:Symbol ...
22 histogram[key]
23 => 10

```



## Typ logiczny oraz pusty

## Notatki

```

1 a = ""
2 a.nil?
3 => false
4 a = []
5 a.nil?
6 => false
7 a = 0
8 a.nil?
9 => false
10 a = nil
11 a.nil?
12 => true
13 if 0 and [] and ''
14   "True"
15 else
16   "False"
17 end
18 => "True"
19 if false or nil
20   "True"
21 else
22   "False"
23 end # => "False"

```



## Porównywanie wartości

## Notatki

```

1 #Przykład ze strony http://stackoverflow.com/questions
2 #/7156955/whats-the-difference-between-equal-eql-and
3 def eq(a, b)
4   puts "#{{a, '==', b}} : #{a == b}"
5   puts "#{{a, '.eq?', b}} : #{a.eql?(b)}"
6   puts "#{{a, '====', b}} : #{a === b}"
7   puts "#{{a, '.equal?', b}} : #{a.equal?(b)}"
8 end
9 eq(Object.new, Object.new)
10 eq('all', 'all')
11 eq(:all, :all)
12 eq(3, 3)
13 eq(1, 1.0)

```



## Funkcje

```

1 ######
2 #Definiowanie funkcji
3 #####
4 def nazwa_funkcji(par1, par2=wartość.domyślna, ...)
5 ...
6   return wartość
7 end
8 # Składnia dostępna począwszy od Ruby 3.0
9 def metoda() = instrukcja
10 #####
11 def good_bye(*guys)
12   "Do widzenia " + guys.join(" i ")
13 end
14 # Składnia dostępna począwszy od Ruby 3.0
15 def good_bye(*guys) = "Do widzenia " + guys.join(" i ")
16
17 #####
18 #Wywoływanie funkcji
19 #####
20
21 nazwa_funkcji(arg1,arg2,...)
22 #####
23 good_bye("Stanisławie","Marcinie")
24 => "Do widzenia Stanisławie i Marcinie"

```

```

1 def funkcja(a=1,b:2,c:3)
2   puts a
3   puts b
4   puts c
5 end
6
7 funkcja(1) #Wypisze: 1 2 3 w osobnych liniach
8 funkcja(1,1) #... wrong number of arguments (2 for
0..1) (ArgumentError)
9 funkcja(c:20,b:10) #Wypisze: 1 10 20 w osobnych
liniach
10 argumenty = { c:200,b:100 }
11 funkcja(3,**argumenty) #Wypisze: 3 100 200 w
osobnych liniach
12 def moja_funkcja((a,b))
13   p a: a, b: b
14 end
15 moja_funkcja([1,2]) #Wypisze: {:a=>1, :b=>2}

```

## Notatki

## Przykłady

## Notatki

```

1 #definiowanie bloków
2 {|x,y,z| x + y + z }
3 do |x,y,z|
4   x + y + z
5 end
6 #przekazywanie bloku jako parametru wywołania metody
7 a = [1,2,3].reject{|el e.odd?}
8 #przekształcenie w obiekt
9 odd = lambda {|el e.odd?}
10 odd = ->(e){e.odd?} #Składnia dostępna począwszy od wersji 1.9. "()" można pominąć
11 even_numbers = [1,2,3].reject &odd
12 odd_number = [1,2,3].select &odd
13 odd.call(3)
14 => true
15 odd[3] # metoda '[]' - alias metody 'call()'
16 => true
17 odd.(3) #Składnia dostępna począwszy od wersji 1.9
18 => true

```



## Przykłady

```

1 def dwarazy
2   yield
3   yield
4 end
5 dwarazy { puts "Witaj" } #można również wywołać tak: dwarazy() { puts "Witaj" }
#Na wyjściu:
6   Witaj
7   Witaj
8   Witaj
9
10 def dwarazy
11   yield 1
12   yield 2
13 end
14 dwarazy { |i| puts "#{i}. raz: Witaj" }
#Na wyjściu:
15   1. raz: Witaj
16   2. raz: Witaj
17
18 def dwarazy(&blok)
19   blok.call 1 # zmienią 'blo'
20   blok.call 2
21 end
22
23 dwarazy { |i| puts "#{i}. raz: Witaj" }
#Na wyjściu:
24   1. raz: Witaj
25   2. raz: Witaj
26
27 =begin
28 Można również wywołać tak:
29   dwarazy() { |i| puts "#{i}. raz: Witaj" }
30 ale nie tak:
31   dwarazy( { |i| puts "#{i}. raz: Witaj" } )
32 =end

```

### Notatki

---



---



---



---



---



---



---



---



---



---



## Argumenty bloków

```

1 i = "foo"
2 j = "foo"
3 puts "Przed blokiem: i = #{i}"
4 puts "Przed blokiem: j = #{j}"
5 3.times do |i|
6   puts "i = #{i}"
7   j = i
8   k = 2
9 end
10 puts "Poza blokiem: i = #{i}"
11 puts "Poza blokiem: j = #{j}"
12 puts "Poza blokiem: k = #{k}"

```

### Notatki

---



---



---



---



---



---



---



---



---



---



---



## Zmienne lokalne dla bloków

### Notatki

---



---



---



---



---



---



---



---

```

1 j = 2
2 jj = 2
3 a = lambda{|;jj| j = 1; jj = 1} #Składnia dostępna począwszy od wersji 1.9
4 a.call()
5 puts j # 1
6 puts jj # 2

```



## Obiekt „Proc”

### Notatki

---



---



---



---



---



---



---



---

```

1 ##### Obiekt 'Proc' #####
2 p = Proc.new { puts "Hello World" }
3 p == proc {puts "Hello World"} #To samo co wyżej
4 # proc <=> lambda, dla Ruby < 1.9
5 # proc <=> Proc.new, dla Ruby >= 1.9
6 p.call # wypisze 'Hello World'
7 p.class # zwróci 'Proc',
8 ##### Bloki #####
9 { puts "Hello World"} #Błąd składowy
10 a = { puts "Hello World"} #Błąd składowy
11 [1,2,3].each {|x| puts x/2} #OK
12 #####
13 def metoda(proc1,proc2)
14   proc1.call
15   proc2.call
16 end
17 a = Proc.new { puts "Pierwszy proc"}
18 b = Proc.new { puts "Drugi proc"}
19 metoda(a,b) #OK
20 metoda{puts "Pierwszy proc"}{puts "Drugi proc"} #Błąd składowy

```



## Przekształcanie metody w obiekt 'Proc'

### Wybieranie liczb nieparzystych

```

1 ######
2 # Dwuetapowo
3 #####
4 odd = :odd?.to.proc # Zmienna 'odd' zawiera obiekt klasy 'Proc', reprezentujący metodę o nazwie takiej
   jak podany symbol
5 p odd #Wypisze: <Proc:0x...(&:odd)
6 p [1, 2, 3].select(&odd) #Wypisze: [1, 3]
7 #####
8 # zmienna = :nazwa_metody.to_proc
9 # obiekt.iterator(&zmienna)
10 #
11 # obiekt.iterator(&:nazwa_metody)
12 #####
13 # Jednoetapowo
14 #####
15 p [1, 2, 3].select(&odd?) #Wypisze: [1, 3]
16 #####
17 # obiekt.iterator(&:nazwa_metody) => obiekt.iterator{|arg| arg.nazwa_metody}
18 # operator '&' powoduje przekazanie nazwy metody do iteratatora
19 #####

```

Bez użycia bloku — z bezpośrednim użyciem obiektu 'Proc'

### Notatki

```
1 p [1, 2, 3].select{|e| e.odd?} #Wypisze: [1, 3]
```

Z użyciem bloku



## Domknięcia

```

1 =begin
2   funkcja wykorzystuje trzy zmienne:
3   'y' - dostepna dla użytkownika
4   'x' - dostepna tylko wewnatrz funkcji 'mnozenie_przez()'
5   'z' - dostepna tylko wewnatrz funkcji 'mnozenie_przez()'
6 =end
#####
8 def mnozenie_przez(x)
9   z=0
10  lambda{|y| z+= 1 ; printf "z=%d, %d*%d=",z,x,y; return x*y}
11 end
#####
13 iloczyn_5_przez = mnozenie_przez(5) # Parametrowi 'x' jest przypisywana wartość 5
14 iloczyn_6_przez = mnozenie_przez(6) # Parametrowi 'x' jest przypisywana wartość 6
16 puts iloczyn_5_przez #Wypisze: <Proc:0x00000000c0de38@lambda.rb:8 (lambda)>
17 x=10
18 y=20
19 z=30
20 puts iloczyn_5_przez.call(10) # Wypisze: z=1, 5*10=50
21 puts iloczyn_5_przez.call(11) # Wypisze: z=2, 5*11=55
22 puts iloczyn_6_przez.call(12) # Wypisze: z=1, 6*12=72
23 #####
24 puts x #Wypisze: 10
25 puts y #Wypisze: 20
26 puts z #Wypisze: 20

```

### Notatki



## Definiowanie klas i tworzenie obiektów

### Notatki

---



---



---



---



---



---



---

```

1 class Zwierze
2 end
3
4 puts Zwierze.class #Wypisze: Class
5 zwierze = Zwierze.new
6 p zwierze #Wypisze: #<Zwierze:0x00000009c17d8>

```



## Metody

### Notatki

---



---



---



---



---



---



---

```

1 class Zwierze
2   @obiekt_self = self
3   #####
4   ##### Metoda instancyjna #####
5   #####
6   def wypisz
7     printf "%s %s\n", self, self.class
8   end
9   #####
10  ##### Metoda klasowa #####
11  #####
12  def Zwierze.wypisz
13  #def self.wypisz
14  #  printf "%s %s %s\n", self, self.class, @obiekt_self, @obiekt_self.class
15  end
16 end
17 #####
18 z=Zwierze.new
19 z.wypisz #Wypisze: #<Zwierze:0x00000002142d58> Zwierze
20 # To co powyżej jest równe z.send('wypisz')
21 Zwierze.wypisz #Wypisze: Zwierze Class Zwierze Class
22 # To co powyżej jest równe zwierze.send('wypisz')

```



## Metoda initialize()

## Notatki

```

1 class Zwierze
2   def initialize(nazwa)
3     puts nazwa
4   end
5 end
6
7 zwierze = Zwierze.new('Pies') #Wypisze: Pies

```



## Dostęp do metody klasowej z wnętrza metody instancjowej

## Notatki

```

1 class Zwierze
2   def metodaInstancjyna
3     puts "Jestem metodą instancjijną"
4     self.class.metodaKlasowa
5   end
6
7   def self.metodaKlasowa
8     puts "Jestem metodą klasową"
9   end
10 end
11 #####
12 z = Zwierze.new
13 z.metodaKlasowa #Błąd wykonania: undefined method `metodaKlasowa' ...
14 #####
15 z.metodaInstancjyna # Wypisze:
16           # Jestem metodą instancjijną
17           # Jestem metodą klasową

```



## Dziedziczenie

```

1 class Zwierze #Klasą bazową jest Object
2 def initialize
3   puts 'Konstruktor Zwierze'
4 end
5 end
6 #####
7 p Zwierze.superclass #Wypisze: Object
8 p Zwierze.superclass.superclass #Wypisze:
  BasicObject
9 p Zwierze.superclass.superclass.superclass #Wypisze:
  nil

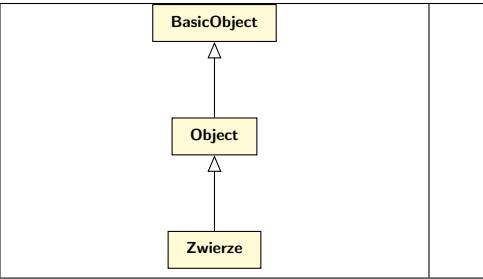
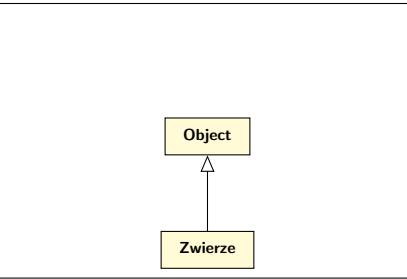
```

```

10 class Pies < Zwierze
11   def initialize
12     super
13     puts "Konstruktor Pies"
14   end
15 end
16 #####
17 Pies.new #Wypisze:
  #Konstruktor Zwierze
  #Konstruktor Pies
18
19

```

### Notatki

Hierarchia dziedziczenia w Ruby  $\geq 1.9$ Hierarchia dziedziczenia w Ruby  $\leq 1.8$ 

## Atrybuty

Instancyjne

```

1 class Zwierze
2   @ilosc = 1
3
4   def wypisz_ilosc
5     p @ilosc #Wbrew pozorom, wartością atrybutu '
  @ilosc' nie jest 1
6   end
7
8   def self.wypisz_ilosc
9     p @ilosc
10  end
11
12  # attr.writer :gatunek
13  def gatunek=(g)
14    @gatunek = g
15  end
16
17  # attr.reader :gatunek
18  def gatunek
19    @gatunek
20  end
21
22  attr_accessor :wiek # attr_accessor ⇔ attr_reader
  ⋮ attr_writer
23 end

```

```

24 z = Zwierze.new
25 z.wypisz_ilosc #Wypisze: nil
26 Zwierze.wypisz_ilosc #Wypisze: 1
27 puts z.ilosc # ... undefined method 'ilosc' for #<
  Zwierze:0x000000017923d0> (NoMethodError)
28 puts z.@ilosc # ... syntax error, unexpected tIVAR,
  expecting '('
29 puts @z.ilosc # ... undefined method 'ilosc' for nil
  NilClass (NoMethodError)
30 z.gatunek = "Ssak"
31 puts z.gatunek # Wypisze: "Ssak"
32 z.wiek = 10
33 puts z.wiek # Wypisze: 10

```

### Notatki

# Atrybuty

Klasowe

Notatki

```

1 class Zwierze
2   @@ilosc = 0
3
4   def initialize
5     @@ilosc += 1
6   end
7
8   def ilosc # metoda instancynia
9     @@ilosc
10  end
11
12  def self.ilosc # metoda klasowa
13    @@ilosc
14  end
15 end
16 ######
17 puts Zwierze.ilosc #Wypisze: 0
18 #####
19 class Pies < Zwierze
20   @@ilosc = 10
21 end
22 #####
23 puts Pies.ilosc #Wypisze: 10
24 puts Zwierze.ilosc #Wypisze: 10
25 pies = Pies.new
26 puts pies.ilosc #Wypisze: 11
27 puts Pies.ilosc #Wypisze: 11
28 puts Zwierze.ilosc #Wypisze: 11

```



## Atrybuty klasowe, a atrybuty instancynie klasy

Notatki

### Atrybut klasowy

```

1 class Zwierze
2   # '@@ilosc' jest atrybutem klasowym dzielonym przez klasę 'Zwierze' i każdą z jej podklaścami
3   # Kiedy tworzyesz instancję klasy 'Zwierze' lub jej podklasy, t.j. 'Pies',
4   # to wartość zmiennej 'ilosc' zwiększa się
5   @@ilosc = 0
6
7   def initialize
8     self.class.ilosc += 1 #Wywołanie metod 'self.ilosc=' oraz
9     'self.ilosc'
10  end
11
12  def self.ilosc
13    @@ilosc
14  end
15  def self.ilosc=(wartosc)
16    @@ilosc = wartosc
17  end
18
19  class Pies < Zwierze
20  end
21
22  8.times { Zwierze.new }
23  4.times { Pies.new }
24
25  p Zwierze.ilosc #Wypisze: 12
26  p Pies.ilosc #Wypisze: 12

```

### Atrybut instancynijny klasy

```

1 class Zwierze
2   # '@ilosc' jest atrybutem instancynijnym (wyłącznie) klasy 'Zwierze',
3   # Wartość zmiennej 'ilosc' ulega zwiększeniu, tylko wtedy gdy tworzyesz instancję klasy 'Zwierze' (nie podklasy, t.j. Pies)
4   @ilosc = 0
5
6   def initialize
7     self.class.ilosc += 1 #Wywołanie metod 'self.ilosc=' oraz
8     'self.ilosc'
9   end
10
11  def self.ilosc
12    @ilosc
13  end
14  def self.ilosc=(wartosc)
15    @ilosc = wartosc
16  end
17
18  class Pies < Zwierze
19    @ilosc = 0
20  end
21
22  8.times { Zwierze.new }
23  4.times { Pies.new }
24
25  p Zwierze.ilosc #Wypisze: 8
26  p Pies.ilosc #Wypisze: 4

```



## Moduły

```

1 module Math
2   PI = 3.14159265358979
3   def Math.sqrt(x)
4     # ...
5   end
6   ...
7 end
8 #####
9 Math.sqrt(2) # => 1.4142135623731
10 Math::sqrt(2) # => 1.4142135623731
11 Math::PI # => 3.14159265358979
12 #####
13 module Brzeczyk
14   def dzwon
15     puts "BZZZZ!BZZZZ!BZZZZ!"
16   end
17 end
18 #####
19 class Czasomierz
20   def podaj_czas
21     puts Time.now
22   end
23 end
24 #####
25 class Budzik < Czasomierz
26   include Brzeczyk
27 end
28 #####
29 b = Budzik.new
30 b.podaj_czas # Sun Aug 05 17:24:08 +0200 2007
31 b.dzwon # BZZZZ!BZZZZ!BZZZZ!
32 Budzik.dzwon #undefined method 'dzwon' for Budzik:
  Class (NoMethodError)

```

### Notatki

---



---



---

```

1 class Budzik < Czasomierz
2   extend Brzeczyk
3 end
4 #####
5 b = Budzik.new
6 b.podaj_czas # Sun Aug 05 17:24:08 +0200 2007
7 b.dzwon # undefined method 'dzwon' for #<Budzik:0
  x0000000242c0c8> (NoMethodError)
8 Budzik.dzwon #BZZZZ!BZZZZ!BZZZZ!
9 zegarek = Czasomierz.new
10 zegarek.extend(Brzeczyk)
11 zegarek.dzwon # BZZZZ!BZZZZ!BZZZZ!

```



## Metody klas oraz modułów

Sterowanie kolejnością wykonywania

```

1 module Brzeczyk
2   def podaj_czas
3     puts "Brzeczyk: " + Time.now.to_s
4   super
5   end
6 end
7 #####
8 class Czasomierz
9   def podaj_czas
10    puts "Czasomierz: " + Time.now.to_s
11  end
12 end
13 #####
14 class Budzik < Czasomierz
15   include Brzeczyk
16   def podaj_czas
17     puts "Budzik: " + Time.now.to_s
18   super
19   end
20 end
21 #####
22 Budzik.new.podaj_czas

```

```

1 module Brzeczyk
2   def podaj_czas
3     puts "Brzeczyk: " + Time.now.to_s
4   super
5   end
6 end
7 #####
8 class Czasomierz
9   def podaj_czas
10  puts "Czasomierz: " + Time.now.to_s
11 end
12 #####
13 #####
14 class Budzik < Czasomierz
15   prepend Brzeczyk #Dostępne w Ruby >= 2.0
16   def podaj_czas
17     puts "Budzik: " + Time.now.to_s
18   super
19   end
20 end
21 #####
22 Budzik.new.podaj_czas

```

### Notatki

---



---



---

Materiały dla studentów Wydziału Informatycznego AGH w Krakowie



## Atrybuty instancyjne oraz klasowe modułów

```

1 module Brzeczyk
2   @czas1 = 1
3   @@czas2 = 2
4
5   def self.czas1
6     @czas1
7   end
8
9   def self.czas2
10    @@czas2
11  end
12 end
13 #####
14 class Czasomierz
15   include Brzeczyk
16
17   def self.czas1
18     @czas1
19   end
20
21   def self.czas2
22     @@czas2
23   end
24
25   def self.czas2=(czas)
26     @@czas2=czas
27   end
28 end
29 #####
30 p Brzeczyk.czas1 # Wypisze: 1
31 p Czasomierz.czas1 # Wypisze: nil
32 p Brzeczyk.czas2 # Wypisze: 2
33 p Czasomierz.czas2 # Wypisze: 2
34 Czasomierz.czas2 = 3
35 p Brzeczyk.czas2 # Wypisze: 3
36 p Czasomierz.czas2 # Wypisze: 3

```

```

1 module Brzeczyk
2   @czas1 = 1
3   @@czas2 = 2
4
5   def self.czas1 # Tutaj 'self' oznacza moduł, więc
6     # odbiorcą komunikatu "czas1" będzie tylko
7     # moduł
8     @czas1
9   end
10
11   def self.czas2
12     @@czas2
13   end
14
15   def self.czas2=(czas)
16     @@czas2=czas
17   end
18 #####
19 p Czasomierz.czas1 # Wyjątek: undefined method 'czas1' for Czasomierz:Class
20 p Czasomierz.czas2 # Wyjątek: undefined method 'czas2' for Czasomierz:Class

```

### Notatki

---



---



---



---



---



---



---



---



---



---

## Sprawdzanie typu obiektu

### Notatki

---



---



---



---



---



---



---



---



---



---

```

1 module Man
2 end
3
4 module Woman
5 end
6
7 class Person
8   include Man
9 end
10
11 class Player < Person
12   include Woman
13 end
14
15 p = Player.new
16 p.kind_of? Player #true
17 p.kind_of? Person #true
18 p.kind_of? Man #true
19 p.kind_of? Woman #true
20 #kind_of? <=> is_a?
21
22 p.instance_of? Player #true
23 p.instance_of? Person #false
24 p.instance_of? Man #false
25 p.instance_of? Woman #false

```



## Obiekt „main”

Notatki

```

1 def funkcja
2   puts "Jestem metodą obiektu 'main'"
3 end
4
5 class Klasa
6   def metoda
7     puts "Jestem metodą obiektu 'Klasa,'"
8     funkcja()
9   end
10 end
11
12 puts self #Wypisze: main
13 puts self.class #Wypisze: Object
14
15 o=Klasa.new
16 o.metoda #Wypisze: Jestem metodą obiektu 'Klasa,
17           #                Jestem metodą obiektu 'main'
```



## Przeciążanie operatorów

Notatki

```

1 class Liczba
2   def initialize(a)
3     @a = a
4   end
5   #####
6   def +(b)
7     @a + b
8   end
9   #####
10  def -@
11    puts "zaneutowałeś liczbę"
12  end
13 end
14 liczba = Liczba.new(1)
15 puts liczba+4
16 -liczba
```



## Kontrola dostępu

```

1 class Test
2   public # każda metoda (poza initialize(), która jest prywatna) jest domyślnie publiczna
3   def pub_met1
4   end
5
6   private # metoda prywatna
7   def priv_met1
8   end
9
10  protected # wymienione niżej metody są chronione
11  def prot_met1
12  end
13
14  def prot_met2
15  end
16 end

```

```

1 class Test
2   def pub_met1
3   end
4
5   def priv_met1
6   end
7
8   def prot_met1
9   end
10
11  public :pub_met1
12  private :priv_met1
13  protected :prot_met1
14 end

```

### Notatki

---



---



---



---



---



---



---

## Kontrola dostępu

### Przykład użycia

```

1 class Foo
2   def a
3   end
4
5   # Wywołanie 'a()' z jawnym 'self' jako odbiorcą
6   def b
7     self.a
8   end
9
10  # Wywołanie metody 'a()' z niejawnym 'self' jako odbiorcą
11  def c
12    a
13  end
14
15  publicprotectedprivate :a
16
17 #####
18 class Foo1 < Foo
19
20 #####
21 def safe_send(receiver, method, message)
22   # Nie można używać 'send()', ponieważ omija zasady widoczności
23   eval "receiver.#{method}"
24 rescue => e
25   puts "#{$message}: #{$e}"
26 else
27   puts "#{$message}: wywołanie metody powiodło się"
28 end
29 #####
30 foo = Foo.new
31 foo1 = Foo1.new
32 safe_send(foo, :a, "jawnego odbiorca")
33 safe_send(foo, :b, "jawnego odbiorca 'self'")
34 safe_send(foo, :c, "niejawnego odbiorca 'self'")
35 safe_send(foo, :a, "jawnego odbiorca")
36 safe_send(foo1, :b, "jawnego odbiorca 'self'")
37 safe_send(foo1, :c, "niejawnego odbiorca 'self'")

```

```

1 $ ruby skrypt.rb
2 jawnego odbiorca : wywołanie metody powiodło się
3 jawnego odbiorca 'self' : wywołanie metody powiodło się
4 niejawnego odbiorca 'self': wywołanie metody powiodło się
5 jawnego odbiorca : wywołanie metody powiodło się
6 jawnego odbiorca 'self' : wywołanie metody powiodło się
7 niejawnego odbiorca 'self': wywołanie metody powiodło się

```

```

1 $ ruby skrypt.rb
2 jawnego odbiorca : protected method 'a' called for #<Foo:0
3 x000000028a0960>
4 jawnego odbiorca 'self' : wywołanie metody powiodło się
5 niejawnego odbiorca 'self': wywołanie metody powiodło się
6 jawnego odbiorca : protected method 'a' called for #<Foo1:0
7 x000000028a0938>
8 jawnego odbiorca 'self' : wywołanie metody powiodło się
9 niejawnego odbiorca 'self': wywołanie metody powiodło się

```

```

1 $ ruby skrypt.rb
2 jawnego odbiorca : private method 'a' called for #<Foo:0
3 x000000021b4840>
4 jawnego odbiorca 'self' : private method 'a' called for #<Foo1:0
5 x000000021b4840>
6 niejawnego odbiorca 'self': wywołanie metody powiodło się
7 jawnego odbiorca : private method 'a' called for #<Foo1:0
8 x000000021b4818>
9 jawnego odbiorca 'self' : private method 'a' called for #<Foo1:0
10 x000000021b4818>
11 niejawnego odbiorca 'self': wywołanie metody powiodło się

```

### Notatki

---



---



---



---



---



---



---

## Programowanie imperatywne a programowanie funkcyjne

Notatki

### Programowanie imperatywne

- ▶ Skupia się na „jak”
- ▶ Programy zawierają listę instrukcji, które „mówią” komputerowi, co zrobić z danymi wejściowymi

*Functional programming is like describing your problem to a mathematician. Imperative programming is like giving instructions to an idiot*

(arcus, #scheme on Freenode)

```

1 n = 1
2 num_elements = 0
3 sum = 0
4 while num_elements < 10
5   if (n**2 % 5).zero?
6     sum += n
7     num_elements += 1
8   end
9   n += 1
10 end
11 sum
12 => 275

```

Podejście imperatywne

```

1 (1...Float::INFINITY).lazy.select { |x| (x**2 % 5).
2   zero? }.take(10).inject(:+)
2 => 275

```

Podejście funkcyjne



## Nie modyfikuj zmiennych

Notatki

Nie

```

1 indeksy = [1, 2, 3]
2 indeksy << 4
3 #####
4 hasz = { :a => 1, :b => 2 }
5 hasz[:c] = 3
6 #####
7 napis = "witaj"
8 napis.gsub!(/t/, 'l')

```

Tak

```

1 indeksy = [1, 2, 3]
2 wszystkie_indeksy = indeksy + [4]
3 #####
4 hasz = { :a => 1, :b => 2 }
5 nowy_hasz = hasz.merge(:c => 3)
6 #####
7 napis = "witaj"
8 nowy_napis = napis.gsub(/t/, 'l')

```



## Bloki jako funkcje wyższego rzędu

### Notatki

---



---



---

```
1 open("/etc/passwd") { |plik| puts plik.size}
```

Funkcja, która zawiera „funkcję” na liście argumentów

```
1 def dodawacz(a)
2   proc {|b| a + b}
3 end
4 dodawacz5 = dodawacz(5)
5 puts dodawacz5.call(3) #Wypisze: 8
```

Funkcja, która zwraca „funkcję”

---



---



---

## Częściowe zastosowanie funkcji oraz rozwijanie funkcji

```
1 suma = proc { |x, y, z| (x||0) + (y||0) + (z||0) }
2 p suma.call(1, 2, 3) # 6
3 p suma[1, 2, 3] # 6
4 ######
5 # Częściowe zastosowanie funkcji (ang. partial function
  application)
6 suma_partial = proc { |x, y| proc { |z| x + y + z } }
7 p suma_partial.call(1, 2).call(3) # 6
8 p suma_partial[1, 2][3] # 6
9 #####
10 # Rozwijanie "funkcji" (ang. currying)
11 suma_curry= proc { |x| proc { |y| proc { |z| x + y + z } }
  }
12 p suma_curry.call(1).call(2).call(3) # 6
13 p suma.curry[1][2][3] # 6
14 p suma.curry[1][2][3] # 6
15 #####
16 f = proc { |x, y, z| [x, y, z] }
17 p f[4, 5, 6] # [4, 5, 6]
18 p f[4][5, 6] # nil
19 p f[4, 5][6] # nil
20 p f[4][5][6] # undefined method '[]' for nil:NilClass (
  NoMethodError)
21
22 g = f.curry
23 p g[4, 5, 6] # [4, 5, 6]
24 p g[4][5, 6] # [4, 5, 6]
25 p g[4, 5][6] # [4, 5, 6]
26 p g[4][5][6] # [4, 5, 6]
27 p g[4] # #<Proc:0x00562b3a477028>
28
29 gg = f.curry(2)
```

Materiały dla studentów wydziału IT i Kierunku Krakowie

```
30 p gg[4, 5, 6] # [4, 5, 6]
31 p gg[4][5, 6] # [4, 5, 6]
32 p gg[4, 5][6] # nil
33 p gg[4][5][6] # nil
34 p gg[4] # <Proc:0x00555ebc5edbc8>
35
36 h = g[1] # 'x' otrzymuje wartość 1
37 p h[4, 5, 6] # [1, 4, 5]
38 p h[4, 5] # [1, 4, 5]
39 p h[4][5] # [1, 4, 5]
40 p h[4] # <Proc:0x000000019404c0>
41
42 hh = gg[1] # 'x' otrzymuje wartość 1
43 p hh[4, 5, 6] # [1, 4, 6]
44 p hh[4, 5] # [1, 4, 5]
45 p hh[4][5] # nil
46 p hh[4] # [1, 4, nil]
47
48 i = h[2] # 'y' otrzymuje wartość 2
49 p i[4, 5, 6] # [1, 2, 4]
50 p i[4, 5] # [1, 2, 4]
51 p i[4] # [1, 2, 4]
52 p i # <Proc:0x00558a3d2a2b38>
53
54 ii = hh[2] # 'y' otrzymuje wartość 2
55 p ii[4, 5, 6] # wrong number of arguments (given 3,
  expected 1..2) (ArgumentError)
56 p ii[4, 5] # nil
57 p ii[4] # nil
58 p ii # [1, 2, nil]
59
60 j = i[3] # 'z' otrzymuje wartość 3
61 p j # [1, 2, 3]
62
63 jj = ii[3]
64 p jj #nil
```



### Notatki

---



---



---



## Rozwiązywanie funkcji

### Przykład użycia

```

1 ##### Normal definitions
2 # Normal definitions
3 # pattern:  $\sum f(n)$  for  $n \in [a \dots b]$ 
4 ##### pattern:  $\sum f(n)$  for  $n \in [a \dots b]$ 
5 sum_ints = lambda do |a,b|
6   s = 0 ; a.upto(b){|n| s += n} ; s
7 end
8
9 sum_of_squares = lambda do |a,b|
10  s = 0 ; a.upto(b){|n| s += n**2} ; s
11 end
12
13 sum_of_powers_of_2 = lambda do |a,b|
14   s = 0 ; a.upto(b){|n| s += 2**n} ; s
15 end
16
17 puts sum_ints.(1,5) #=> 15
18 puts sum_of_squares.(1,5) #=> 55
19 puts sum_of_powers_of_2.(1,5) #=> 62
20 ##### Some refactoring to make some abstraction
21 # Some refactoring to make some abstraction
22 # Passing the sum mechanism itself
23 #####
24 sum = lambda do |f,a,b|
25   s = 0 ; a.upto(b){|n| s += f.(n)} ; s
26 end

```

### Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---



---

```

27 puts sum[lambda{|x| x},1,5] #=> 15
28 puts sum[lambda{|x| x**2},1,5] #=> 55
29 puts sum[lambda{|x| 2**x},1,5] #=> 62
#####
30 # More refactoring using currying
31 #####
32 #####
33 #####
34 # generate the currying
35 currying = sum.curry
36
37 # Generate the partial functions
38 sum_ints = currying[lambda{|x| x}]
39 sum_of_squares = currying[lambda{|x| x**2}]
40 sum_of_powers_of_2 = currying[lambda{|x| 2**x}]
41
42 puts sum_ints[1,5] #=> 15
43 puts sum_of_squares[1,5] #=> 55
44 puts sum_of_powers_of_2[1,5] #=> 62

```

Źródło: <http://blog.khd.me/ruby/ruby-currying/>

## Unikanie skutków ubocznych

### Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---



---

### Nie

```

1 [1,2,3].each do |x|
2   #kod z efektami ubocznymi
3 end
4 for i in [1,2,3] do
5   #kod z efektami ubocznymi
6 end

```

## Metoda map()

**Notatki****Nie**

```

1 wynik = []
2 [1,2,3,4].each do |x|
3   wynik << x*2
4 end
5 wynik #[2, 4, 6, 8]

```

**Tak**

```

1 wynik=[1,2,3,4].map do |x|
2   x*2
3 end #[2, 4, 6, 8]

```



## Metoda select()

**Notatki****Nie**

```

1 wynik = []
2 [1,2,3,4].each do |x|
3   wynik << x if x > 2
4 end
5 wynik #[3,4]

```

**Tak**

```

1 wynik=[1,2,3,4].select do |x|
2   x > 2
3 end #[3,4]

```



## Metoda detect()

**Notatki****Nie**

```

1 wynik = nil
2 [1,2,3,4].each do |x|
3   if x > 2
4     wynik = x
5     break
6   end
7 end
8 wynik #3

```

**Tak**

```

1 wynik=[1,2,3,4].detect do |x|
2   x > 2
3 end #3

```



## Metoda inject()

**Notatki****Nie**

```

1 suma = 0
2 [1,2,3,4].each do |x|
3   suma += x
4 end
5 suma #10

```

**Tak**

```

1 suma = [1,2,3,4].inject(0) do |aku, x|
2   aku+x
3 end #10
4 #####
5 suma = [1,2,3,4].inject(0, :+) #10

```



## Metoda zip()

Notatki

**Nie**

```

1 a = [1,2,3,4]
2 b = [5,6,7,8]
3 wynik=[]
4 0.upto(a.length-1).each do |i|
5   wynik << [a[i], b[i]]
6 end
7 wynik #[[1, 5], [2, 6], [3, 7], [4, 8]]

```

**Tak**

```

1 a = [1,2,3,4]
2 b = [5,6,7,8]
3 wynik = a.zip(b) #[[1, 5], [2, 6], [3, 7], [4, 8]]

```

## Wybrane metody do tworzenia iteratorów

Notatki

```

1 [1,2,3].permutation(2).to_a
2 => [[1, 2], [1, 3], [2, 1], [2, 3], [3, 1], [3, 2]]
3 [1,2,3].permutation.to_a
4 => [[1, 2, 3], [1, 3, 2], [2, 1, 3], [2, 3, 1], [3, 1, 2], [3, 2, 1]]
#####
6 [1,2,3].combination(2).to_a
7 => [[1, 2], [1, 3], [2, 3]]
8 #####
9 [0,1,2,3].group_by{|x| x%2}
10 => {0=>[0, 2], 1=>[1, 3]}
11 [0,0,0,1,1,2].group_by{|x| x}.map{|k,v| v}
12 => [[0, 0, 0], [1, 1], [2]]

```

## Przykład użycia

Notatki

---



---



---



---



---



---



---



---



---



---



---

```
1 (1..Float::INFINITY).lazy.select { |x| (x**2 % 5).zero? }.take(10).inject(:+)
2 => 275
```



## Użycie „Rack”

### Przykład 1

Notatki

---



---



---



---



---



---



---



---



---



---



---

```
1 #!/usr/bin/env ruby
2 require "rack"
3 require "rackup"
4
5 class HelloWorld
6   def call(env)
7     sleep(10) #uśpij program na 10 sekund
8     return [200, {}, ["Hello world!"]]
9   end
10 end
11
12 Backup::Server.start app: HelloWorld.new, Port: 9000
```

helloworld.rb

```
1 $ chmod 755 ./helloworld.rb
2 $ ./helloworld.rb
3 [2014-04-14 10:43:16] INFO  WEBrick 1.3.1
4 [2014-04-14 10:43:16] INFO  ruby 2.1.1 (2014-02-24)
5 [x86_64-linux]
5 [2014-04-14 10:43:16] INFO  WEBrick::HTTPServer#
6 start: pid=9026 port=9000
6 localhost - - [14/Apr/2014:10:44:04 CEST] "GET /
7 HTTP/1.1" 200 12
7 -> /
```

```
1 $ curl localhost:9000
2 Hello world!
```



## Użycie „Rack”

## Przykład 2

## Notatki

```

1 class HelloWorld
2   def call(env)
3     return [
4       200,
5       {'content-type' => 'text/html'},
6       ["Hello world!"]
7     ]
8   end
9 end
10
11 run HelloWorld.new

```

config.ru

```

1 $ rackup # ⇐⇒ rackup config.ru
2 ...
3 * Listening on tcp://0.0.0.0:9292
4 $ rackup -p 1337
5 ...
6 * Listening on tcp://0.0.0.0:1337

```

```

1 $ curl localhost:9292
2 Hello world!

```



## Parametr 'env'

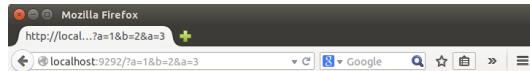
## Notatki

```

1 class HelloWorld
2   def call(env)
3     odpowiedź=env.map do |klucz,wartość|
4       klucz.to_s+"="+wartość.to_s+"<br>"
5     end
6     return [
7       200,
8       {'content-type' => 'text/html;charset='
9         "utf-8"},
10      odpowiedź
11    ]
12  end

```

helloworld.rb



```

GATEWAY_INTERFACE=CGI/1.1
PATH_INFO=
QUERY_STRING=a=1&b=2&a=3
REMOTE_ADDR=127.0.0.1
REMOTE_HOST=localhost
REQUEST_METHOD=GET
REQUEST_URI=http://localhost:9292/?a=1&b=2&a=3
SCRIPT_NAME=
SERVER_NAME=localhost
SERVER_PORT=9292
SERVER_PROTOCOL=HTTP/1.1
SERVER_SOFTWARE=WEBrick/1.3.1 (Ruby/2.1.2/2014-05-08)
HTTP_HOST=localhost:9292
HTTP_USER_AGENT=Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:30.0) Gecko/20100101 Firefox/30.0
HTTP_ACCEPT=text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
HTTP_ACCEPT_LANGUAGE=pl,en-us;q=0.7,en;q=0.3
HTTP_ACCEPT_ENCODING=gzip, deflate
HTTP_CONNECTION=keep-alive
rack.version=[1, 2]
rack.input=
rack.errors=
rack.multithread=true
rack.multiprocess=false
rack.run_once=false
rack.url_scheme=http
HTTP_VERSION=HTTP/1.1
REQUEST_PATH=

```



## Obsługa formularzy HTML

```

1 <form action="http://localhost:9292/">
2 <input type="TEXT" name="a" value="ą" />
3 <input type="TEXT" name="b" value="(" />
4 <input type="TEXT" name="a" value="1" />
5 <input type="SUBMIT">
6 </form>

```



```

1 class HelloWorld
2   def call(env)
3     odpowiedz = ""
4     #Odczytywanie danych otrzymanych od przeglądarki
5     odpowiedz << env['QUERY_STRING'] #Dane
6     otrzymane (metodą GET) od przeglądarki
7     odpowiedz << env['rack.input'].read #Dane
8     otrzymane (metodą POST) od przeglądarki
9
10    #Wysyłanie odpowiedzi do przeglądarki
11    return [
12      200,
13      {'content-type' => 'text/html'},
14      [odpowiedz]
15    ]
end

```

helloworld.rb

## Notatki



## Obsługa formularzy HTML

Obiekt Rack::Request

```

1 <form action="http://localhost:9292/">
2 <input type="TEXT" name="a[]" value="ą" />
3 <input type="TEXT" name="b" value="(" />
4 <input type="TEXT" name="a[]" value="1" />
5 <input type="SUBMIT">
6 </form>

```



```

1 class HelloWorld
2   def call(env)
3     req = Rack::Request.new(env)
4     odpowiedz = [
5       "a["+req.params['a'][0]+"]"+"<br>",
6       "a["+req.params['a'][1]+"]"+"<br>",
7       "b["+req.params['b']] # zamiast "req.params['b']"
8       można użyć "req[b]""
9     ]
10    return [
11      200,
12      {'content-type' => 'text/html; charset=utf-8'},
13      [odpowiedz]
14    ]
15  end
end

```

helloworld.rb

## Notatki



## Obsługa formularzy HTML

Obiekt Rack::Response

Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---

```

1 class HelloWorld
2   def call(env)
3     req = Rack::Request.new(env)
4     res = Rack::Response.new
5     res.write("a='"+req.params['a'][0]+"") #Dopisuje podany tekst do ciała odpowiedzi
6     res.write("a='"+req.params['a'][1]+"")
7     res.write("b='"+req.params['b'])
8     res['content-type'] = 'text/html; charset=utf-8' #Ustawienie nagłówka "content-type"
9     res.status = 200 #zbędne, gdyż jest to domyślny kod statusu
10    res.finish #zwróć odpowiedź w postaci (trzyelementowej) tablicy
11  end
12 end

```

helloworld.rb



## Tworzenie aplikacji pośredniczącej („Middleware”)

Notatki

---



---



---



---



---



---



---



---



---



---



---



---



---



---

```

1 # Źródło: http://code.tutsplus.com/articles/exploring-rack--net-32976
2 class ToUpper
3   # Our class will be initialize with another Rack app
4   def initialize(app)
5     @app = app
6   end
7
8   def call(env)
9     # First, call '@app'
10    status, headers, body = @app.call(env)
11
12    # Iterate through the body, upcasing each chunk
13    upcased_body = body.map { |chunk| chunk.upcase }
14
15    # Pass our new body on through
16    [status, headers, upcased_body]
17  end
18 end
#####
20 class Hello
21   def self.call(env)
22     [ 200, {"content-type" => "text/plain"}, ["Hello from Rack!"] ]
23   end
24 end
#####
26 use ToUpper # Tell Rack to use our newly-minted middleware
27 run Hello
28 # Odpowiednik: app = ToUpper.new(Hello); run app

```

config.ru



## Aplikacja pośrednicząca

Przekazywanie parametrów

```

1 class ToUpper
2   def initialize(app, param, &blok)
3     puts app
4     puts app.class
5     puts ',' * 10
6     puts param
7     puts param.class
8     puts ',' * 10
9     blok.call('Napis')
10    end
11
12    ...
13  end
14
15 class Hello
16  ...
17 end
18
19 # use(middleware, *args, &block)
20 use ToUpper, 'Argument' { |arg| puts arg }
21 run Hello

```

config.ru

Notatki

---



---



---



---



---

Na wyjściu

Hello

Class

---

-----

Argument

String

---

-----

Napis

---



---



---



---



---

## Ogólnodostępne aplikacje pośredniczące

Przykład użycia

```

1 require './helloworld'
2 require 'rack/contrib'
3
4 use Rack::Static, :urls => ["/css", "/images"], :root =>
5   "public"
6 use Rack::ContentType, "text/plain"
7 use Rack::BounceFavicon
8 use Rack::Reloader, 0
9 run HelloWorld.new

```

config.ru

Notatki

---



---



---



---



---

```

1 <link rel="stylesheet" href="/css/style.css">

```

Dokument HTML

```

1 h1 {color: red}

```

public/css/style.css

## Przykład „Hello World”

```
1 require 'sinatra'  
2  
3 get '/witaj' do  
4   "Witaj Świecie\n"  
5 end
```

skrypt.rb

Notatki

---

---

---

---

---

```
1 $ ruby skrypt.rb  
2 /home/polak/.local/share/gem/ruby/gems/sinatra-4.0.0/lib/sinatra/base.rb:1849: warning: Rack::Logger is  
3 deprecated and will be removed in Rack 3.2.  
4 == Sinatra (v4.0.0) has taken the stage on 4567 for development with backup from Puma  
5 * Puma starting in single mode...  
6 * Puma version: 6.4.2 (ruby 3.3.2-p78) ("The Eagle of Durango")  
7 * Min threads: 0  
8 * Max threads: 5  
9 * Environment: development  
10 * PID: 7390  
11 * Listening on http://127.0.0.1:4567  
12 * Listening on http://[::1]:4567  
12 Use Ctrl-C to stop
```

Uruchamianie aplikacji

```
1 $ curl http://localhost:4567/witaj  
2 Witaj Świecie
```

Wykonywanie żądania



## Trasy

Definiowanie

Notatki

---

---

---

---

---

```
1 get '/' do  
2   .. wyświetl coś ..  
3 end  
4  
5 post '/' do  
6   .. utwórz coś ..  
7 end  
8  
9 put '/', do  
10  .. zastąp coś ..  
11 end  
12  
13 patch '/', do  
14  .. zmodyfikuj coś ..  
15 end  
16  
17 delete '/', do  
18  .. unicestw  coś ..  
19 end  
20 ...
```

Materiały dla studentów wydziału Informatyki AGH w Krakowie



## Trasy

### Dopasowywanie

```

1 get '/witaj' do
2   "Pierwsza trasa\nhasz 'params' zawiera: #{params.inspect}\n"
3 end
4
5 get '/witaj' do #Ta trasa nie zostanie (nigdy) dopasowana
6   "Druga trasa\nhasz 'params' zawiera: #{params.inspect}\n"
7 end
8
9 get '/witaj/:nazwa' do
10   "Trzecia trasa\nhasz 'params' zawiera: #{params.inspect}\n"
11 end

```

### Notatki

```

1 $ curl http://localhost:4567/witaj
2 Pierwsza trasa
3 hasz 'params' zawiera: {}
4 $ curl 'http://localhost:4567/witaj?imie=Stanislaw&nazwisko=Polak'
5 Pierwsza trasa
6 hasz 'params' zawiera: {"imie"=>"Stanislaw", "nazwisko"=>"Polak"}
7 $ curl http://localhost:4567/witaj/Swiecie
8 Trzecia trasa
9 hasz 'params' zawiera: {"splat"=>[], "captures"=>["Swiecie"], "nazwa"=>"Swiecie"}
10 $ curl 'http://localhost:4567/witaj/Swiecie?imie=Stanislaw&nazwisko=Polak'
11 Trzecia trasa
12 hasz 'params' zawiera: {"imie"=>"Stanislaw", "nazwisko"=>"Polak", "splat"=>[], "captures"=>["Swiecie"],
13   "nazwa"=>"Swiecie"}

```



## Szablon

### Użycie szablonu ERB

```

1 get '/witaj' do
2   erb(:witaj) #Renderuj szablon 'views/witaj.erb'
3 end

```

### Notatki

skrypt.rb

```

1 <%# To jest komentarz %>
2 <%# Poniżej jest skryptlet %>
3 <% 3.downto(0) do |i| %>
4 <%# Poniżej jest wyrażenie %>
5 <%= i %>
6 <% end %>
7 Start

```

views/witaj.erb

```

1 $ curl http://localhost:4567/witaj
2 3,
3 2,
4 1,
5 0,
6 Start

```

Wykonywanie żądania



## Szablon „inline”

Notatki

```

1 require 'sinatra'
2
3 get '/witaj' do
4   erb(:witaj)
5 end
6 __END__
7
8 @@witaj
9 <%# To jest komentarz %>
10 <%# Poniżej jest skryptlet %>
11 <% 3.downto(0) do |i| %>
12 <%# Poniżej jest wyrażenie %>
13 <%= i %>
14 <% end %>
15 Start

```

skrypt.rb



## Szablon częściowy

Notatki

```

1 require 'sinatra'
2
3 get '/' do
4   erb(:index)
5 end

```

skrypt.rb

```

1 <%= erb(:witaj) %>
2 <%= erb(:swiecie) %>

```

views/index.erb

```

1 $ curl http://localhost:4567/
2 Witaj
3 Świecie

```

Wykonywanie żądania

```
1 Witaj
```

views/witaj.erb

```
1 Świecie
```

views/swiecie.erb

## Układ

Notatki

```

1 <html>
2   <head>
3     <title>Aplikacja Sinatra</title>
4   </head>
5   <body>
6     <h1>Aplikacja Sinatra</h1>
7     <!-- początek wstawki szablonu -->
8     <%= yield %>
9     <!-- koniec wstawki szablonu-->
10    </body>
11 </html>

```

views/layout.erb

```

1 <% 3.downto(0) do |i| %>
2   <%= i %>,
3   <% end %>
4 Start

```

views/witaj.erb

```

1 $ curl http://localhost:4567/witaj
2 <html>
3   <head>
4     <title>Aplikacja Sinatra</title>
5   </head>
6   <body>
7     <h1>Aplikacja Sinatra</h1>
8     <!-- początek wstawki szablonu -->
9     3,
10    2,
11    1,
12    0,
13 Start
14
15     <!-- koniec wstawki szablonu-->
16   </body>
17 </html>

```

Wykonywanie żądania



## Dostęp do zmiennych z poziomu szablonu

Notatki

```

1 require 'sinatra'
2
3 get '/witaj' do
4   @tekst1 = "Witaj" #Zmienna instancynja jest dostępna z poziomu szablonu
5   tekst2 = "Świecie" #Zwykła zmienna nie jest dostępna z poziomu szablonu
6   erb(:witaj, :locals => {:tekst3 => tekst2}) #Definiowanie, dla szablonu 'witaj', zmiennej lokalnej '
7 end

```

skrypt.rb

```

1 <%= @tekst1 %> <%= tekst3 %>

```

views/witaj.erb

```

1 $ curl http://localhost:4567/witaj
2 Witaj Świecie

```

Wykonywanie żądania

## Przykład

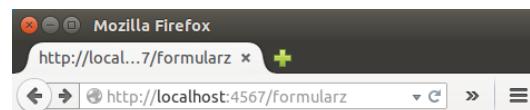
Przetwarzanie zawartości formularza

```
1 require 'sinatra'
2
3 get '/formularz' do
4   erb :formularz
5 end
6
7 post '/formularz' do
8   "Pole 'nazwa' zawiera napis '#{params[:nazwa]}'"
9 end
```

skrypt.rb

```
1 <form action="/formularz" method="post">
2   <input type="text" name="nazwa">
3   <input type="submit">
4 </form>
```

views/formularz.erb



Notatki

---

---

---



Pole 'nazwa' zawiera napis 'Stanisław Polak'

---

---

---

## Używanie aplikacji pośredniczącej

```
1 class ToUpper
2   def initialize(app)
3     @app = app
4   end
5
6   def call(env)
7     status, headers, body = @app.call(env)
8     upcased_body = body.map { |chunk| chunk.upcase }
9     [status, headers, upcased_body]
10  end
11 end
```

to\_upper.rb

```
1 require 'sinatra'
2 load 'to_upper.rb'
3
4 use ToUpper
5
6 get '/witaj' do
7   "Witaj Świecie\n"
8 end
```

skrypt.rb

```
1 $ curl http://localhost:4567/witaj
2 WITAJ ŚWIECIE
```

Wykonywanie żądania



Notatki

---

---

---

## REST API w Sinatra

Notatki

```

1 require 'sinatra'
2
3 use Rack::MethodOverride
4
5 get '/formularz' do
6   erb :formularz
7 end
8
9 delete '/products/:id' do
10   "Usuwam produkt o id=#{params['id']}"
11 end

```

skrypt.rb

```

1 <form method="POST" action="/products/1">
2 <input type="hidden" name="_method" value="DELETE">
3 <input type="submit">
4 </form>

```

views/formularz.erb



## Trasowanie plików statycznych

Notatki

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="UTF-8">
5     <link rel="stylesheet" href="/css/style.css">
6     <title>Aplikacja Sinatra</title>
7   </head>
8   <body>
9     <h1>Aplikacja Sinatra</h1>
10    <%= yield %>
11  </body>
12 </html>

```

views/layout.erb

```

1 h1 {color: red}

```

public/css/style.css



## Źródła I

- ▶ **Jamis Buck.** *Method visibility in Ruby.* URL: <http://weblog.jamisbuck.org/2007/2/23/method-visibility-in-ruby>.
- ▶ **Radosław Bułat.** *Blog.* URL: <http://radarek.jogger.pl/>.
- ▶ **Getting Started.** URL: <https://reactjs.org/docs/getting-started.html>.
- ▶ **Paweł Grzesiak.** *Ajax w kilka minut.* URL: [http://internetmaker.pl/artykul/723,1,ajax\\_w\\_kilka\\_minut.html](http://internetmaker.pl/artykul/723,1,ajax_w_kilka_minut.html).
- ▶ **Patrick Hunlock.** *Functional Javascript.* URL: [http://www.hunlock.com/blogs/Functional\\_Javascript](http://www.hunlock.com/blogs/Functional_Javascript).
- ▶ **MongoDB Inc.** *The MongoDB Manual.* URL: <http://docs.mongodb.org/manual/>.
- ▶ **Joyent.** *Node.js Manual & Documentation.* URL: <http://nodejs.org/api/>.
- ▶ **Agnieszka Matysek.** *Dopasowywanie wzorców - nowa funkcjonalność w Ruby 2.7.* URL: <https://womanonrails.com/pl/ruby-pattern-matching>.
- ▶ **Microsoft.** *TypeScript handbook.* URL: <http://www.typescriptlang.org/Handbook>.
- ▶ **Blake Mizerany.** *Sinatra - documentation.* URL: <http://www.sinatrarb.com/documentation.html>.
- ▶ **Michael Morin.** *Using Rack.* URL: <http://ruby.about.com/od/rack/a/Using-Rack.htm>.

Notatki

## Źródła II

- ▶ **Mozilla.** *AJAX.* URL: <https://developer.mozilla.org/pl/AJAX>.
- ▶ **mozilla.org.** *JavaScript Guide.* URL: <https://developer.mozilla.org/en/JavaScript/Guide>.
- ▶ **Aleksander Pohl.** *Ruby intro.* URL: <http://www.apohllo.pl/dydaktyka/ruby/intro/>.
- ▶ **Axel Rauschmayer.** *Iterables and iterators in ECMAScript 6.* URL: <http://www.2ality.com/2015/02/es6-iteration.html>.
- ▶ **Arnaud Sanchez.** *Functional Programming with Ruby.* URL: <http://www.slideshare.net/tokland/functional-programming-with-ruby-9975242>.
- ▶ **Marek Stępień.** *marcoos.techblog.* URL: <http://blog.marcoos.com/>.
- ▶ **Basarat Ali Syed.** *TypeScript Deep Dive.* URL: <http://basarat.gitbooks.io/typescript/>.
- ▶ **Techotopia.com.** *Ruby Essentials.* URL: [http://www.techotopia.com/index.php/Ruby\\_Essentials](http://www.techotopia.com/index.php/Ruby_Essentials).
- ▶ **Kazuki Tsujimoto.** *Pattern matching—New feature in Ruby 2.7.* URL: [https://speakerdeck.com/k\\_tsj/pattern-matching-new-feature-in-ruby-2-dot-7](https://speakerdeck.com/k_tsj/pattern-matching-new-feature-in-ruby-2-dot-7).
- ▶ **wikibooks.org.** *JavaScript.* URL: <http://en.wikibooks.org/wiki/JavaScript>.
- ▶ **Wikipedia.** URL: <http://pl.wikipedia.org/>.
- ▶ **Wikipedia.** *Kurs PHP.* URL: <http://pl.wikibooks.org/wiki/PHP>.



Notatki

## źródła III

Notatki

---

---

---

- ▶ Wikipedia. *Kurs Ruby*. URL: <http://pl.wikibooks.org/wiki/Ruby>.
- ▶ Noppakun Wongsrinoppakun. *An Array of Possibilities: A Guide to Ruby Pattern Matching*. URL: <https://www.toptal.com/ruby/ruby-pattern-matching-tutorial>.

---

---

---

---

---



Notatki

---

---

---

---

---

---

---

---

---

---