Wprowadzenie

Tworzenie serwisów Web 2.0

dr inż. Robert Perliński
rperlinski@icis.pcz.pl

Politechnika Częstochowska
Instytut Informatyki Teoretycznej i Stosowanej

22 lutego 2018

Plan prezentacji

1. Sprawy organizacyjne
2. Web 2.0
   - Osiem cech Web 2.0
   - Web^2, IoT
   - Web 3.0?
3. SPA i RESTful API
   - Single Page Application
   - ECMAScript, JavaScript
   - AJAX
   - REST, URI
4. Źródła

Sprawy ogranizacyjne

- 1 godz. wykładu, 2 godz. laboratorium
- Wykład nie jest obowiązkowy ale za chodzenie są bonusy
- Przedmiot kończy się egzaminem w formie pisemnej
- Laboratoria są obowiązkowe

Kontakt:

- http://icis.pcz.pl/~rperlinski
- pokój 205, IITiS
- konsultacje: poniedziałek 11:00 - 14:00
  (najlepiej umówić się na spotkanie drogą mailową)

Plan przedmiotu

1. Wprowadzenie - Web 2.0, co będzie na przedmiocie
2. Przegląd technologii
3. Node.js (express-generator, rooting, szablony)
4. Node.js (warstwy pośrednie, obsługa sesji, uwierzytelnienie, ciasteczka)
5. RESTful API (MongoDB i mongoose)
6. Wygląd aplikacji - Bootstrap i Responsive Web Design
7. Wprowadzenie do TypeScript
8. Angular 5.2.5 - architektura
9. Angular 5.2.5 - hello world, tutorial o bohaterach
10. Angular 5.2.5 - kontrolki uzytkownika
11. Web^2 itp. - ciekawostki

Zaliczenie laboratoriów

Zasady zaliczenia laboratoriów:

- Oddanie dwóch sprawozdań, średnia z ocen, oba tak samo ważne.
- Ocena ze sprawozdań zależy od liczby użytych elementów w aplikacjach, stopnia ich zaawansowania itp.

Technologie:

- aplikacja w Node.js z silnikiem szablonów, strukturą projektu, sesją i uwierzytelnieniem, obsługą maili
- ładny i responsywny interfejs po stronie klienta Bootstrap z RWD
- funkcjonalność po stronie klienta - Angular 5.2.5 (zarządzanie danymi z wykrzystaniem bazy)

MEAN Stack (api po stronie serwera + aplikacja w Angularze)

Web 2.0

Web 2.0 - pierwsze informacje

Web 2.0

Skrótowo mianem Web 2.0 określane są serwisy internetowe, których zawartość jest generowana przez samych użytkowników.

Od kiedy mamy do czynienia z Web 2.0?

- Termin Web 2.0 został spopularyzowany w serii konferencji rozpoczętych pod koniec 2004 roku przez O'Reilly Media.
- Organizacja konferencji Web 2.0 miała na celu odbudowanie zaufania i pewności siebie w sektorze przedsiębiorstw branży internetowej bo krysysie i upadku sektora w roku 2001.
- Konferencje te dały również odpowiedź na pytanie o upadek wielu firm a jednocześnie zadziwiający sukces innych.

Bańka internetowa (ang. *dot-com bubble*)

Bańka internetowa - okres euforii na giełdach całego świata w latach 1995-2001 związany ze spółkami branży informatycznej.

- Cechą charakterystyczną było przecenianie przedsiębiorstw, które prowadziły działalność w internecie albo zamierzały ją rozpocząć.
- Była moda na spółki internetowe i te bardzo często miały mocno zawyżoną wartość.

W tym okresie (rok 1995 i najbliższy po nim czas):

- powstaje pierwsza wersja przeglądarki internetowej Netscape,
- następuje wydanie systemu operacyjnego Windows 95,
- powstaje również język Java.

W 1995 roku internet miał tylko 18 mln użytkowników, ale zaczynał bardzo szybko rosnąć - perspektywa rozwoju firm internetowych.

Bańka internetowa (ang. *dot-com bubble*)

Wykres indeksu giełdowego NASDAQ Composite w latach 1994-2005 ze szczytem na początku 2000 r. Bańka internetowa osiągnęła swój szczyt 10 marca 2000 roku, kiedy to indeks giełdowy NASDAQ Composite osiągnął rekordowy poziom 5132,52 i zamknął się tego dnia na poziomie 5048,62 punktów, notując najwyższą wartość od początku swojego istnienia, sygnalizując szczytowy punkt fascynacji akcjami spółek internetowych.

Internet przed Web 2.0

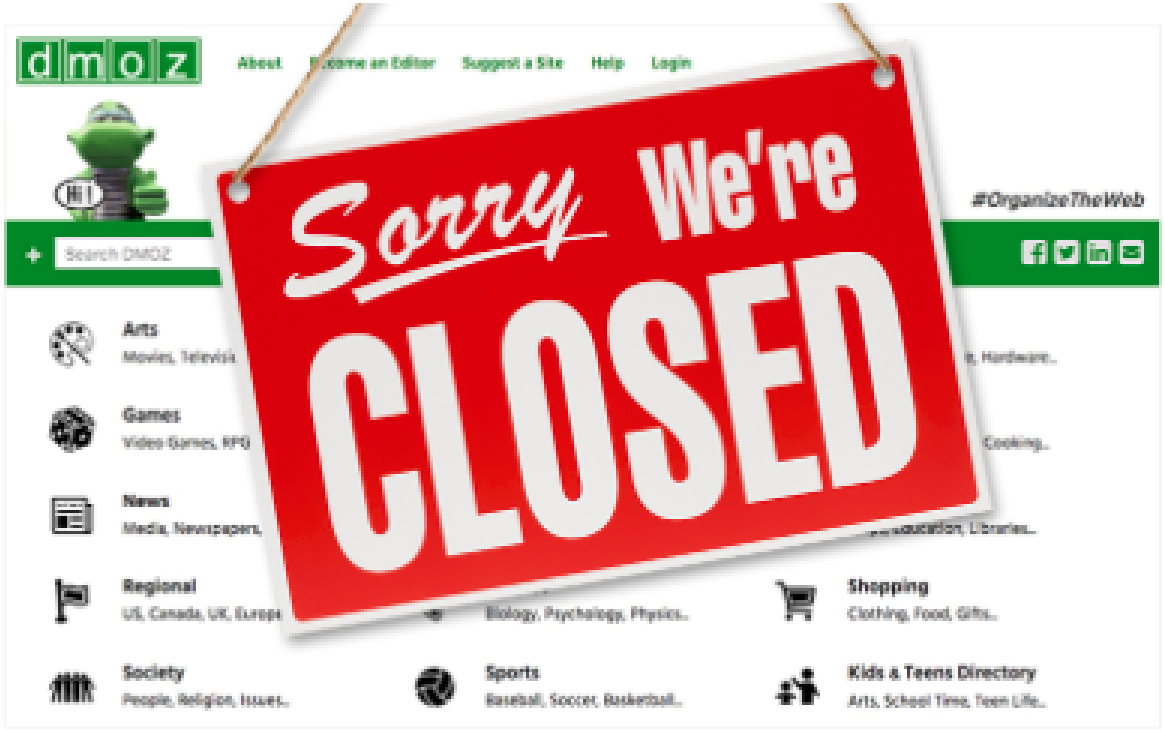
Jak wyglądał internet przed kryzysem, przed Web 2.0?
Co było wcześniej?

Web 1.0

Główne cechy serwisów Web 1.0

- statyczne strony oparte najczęściej o HTML i CSS,
- brak jakichkolwiek elementów interakcji z użytkownikiem,
- układ strony oparty na tabelach (układ wierszy i kolumn),
- stosowanie ramek (frameset), w HTML5 nie są już dostępne,
- twórca strony był jedynym kreatorem jej treści,
- księgi gości, formularze kontaktowe,
- czas pierwszej „wojny przeglądarkowej”,
- nieodzowne animowane gify...
- średnia przepustowość internetu w domu do kilkadziesiąt KiB
- taksonomia stron - kategoryzowanie zawartości (katalogi stron, np. dmoz)

DMOZ Closed



As of Mar 17, 2017, dmoz.org is no longer available.

W piątek 17 marca 2017 roku zamknięto jeden z najstarszych i największych katalogów internetowych dmoz.org. Dostępny jest jedynie mirror tego serwisu.

Mapa Web 2.0


Aggregators
Folksonomy
Wikis
Blogs
User Centered
Joy of Use
Participation
Six Degrees
Usability
Pagerank
XFN
Widgets
Recommendation
Social Software
FOAF
Browser
Simplicity
Sharing
Collaboration
Perpetual Beta
Videocasting
Podcasting
AJAX
Audio
IM
Video
Web 2.0
Design
Convergence
CSS
Pay Per Click
UMTS
Mobility
Atom
XHTML
SVG
Ruby on Rails
VC
Trust
Affiliation
OpenAPIs
RSS
Semantic
Web Standards
SEO
Economy
OpenID
Remixability
REST
Standardization
The Long Tail
DataDriven
Accessibility
XML
Microformats
Syndication
Modularity
SOAP

Web 2.0 - cechy charakterystyczne

Cechy charakterystyczne dla portali Web 2.0

- treść współtworzona przez użytkowników (nastawienie na interakcję z użytkownikiem)
- wokół portali społecznościowych tworzą się rozubdowane społeczności
- taksonomia zastąpiona folksonomią
- portale Web 2.0 budowane są w oparciu o XHTML, CSS, SOAP, AJAX, RSS,
- wykorzystanie usług sieciowych
- udostępnianie otwartego API
- wykorzystanie otwartych licencji
- prejrzystość i prostota wyglądu (gradienty, duże czcionki, zaokrąglenia, ...),
- zdecydowane zwiększenie użyteczności stron w stosunku do Web 1.0.

Ewolucja w sieci

Przemiany

Britanica online	$\Longrightarrow$	wikipedia
strony domowe	$\Longrightarrow$	blogi
publikowanie treści	$\Longrightarrow$	współudział w tworzeniu
katalogi	$\Longrightarrow$	tagowanie zawartości
mp3.com	$\Longrightarrow$	p2p (napster)
DoubleClick	$\Longrightarrow$	Adsense

Przykłady serwisów Web 2.0 : del.icio.us (początki folskonomii), Facebook, Digg, Wikipedia, YouTube...

Co to jest folksonomia?

Folksonomia - neologizm oznaczający praktykę kategoryzacji treści z wykorzystaniem dowolnie dobranych słów kluczowych.

Inne nazwy to:

- wspólne tagowanie,
- społeczna klasyfikacja,
- społeczne indeksowanie,
- społeczne tagowanie, oznaczanie,
- kumplonomia.

Potocznie folksonomia oznacza grupę ludzi współpracujących spontanicznie w celu uporządkowania informacji w kategoriach.

Oznacza ona nieformalne metody klasyfikacji. Klasyfikacją informacji zajumją się osoby, które je wykorzystuja - taka klasyfikacja lepiej oddaje model informacji.

Długi ogon (ang. *The long tail*)

Sukces takich firm jak Google zrodził się ze zrozumienia tego co Chris Anderson nazwy "długim ogonem" (ang. *the long tail*). Chodzi tutaj o zbiorową siłę małych stron, które razem tworzą ogrom treści internetowych.

Jeden z wzorców projektowych Web 2.0 głosi, że ogrom treści internetowych jest tworzonych przez małe strony, zwykłych użytkowników.

Dlatego: duży nacisk należy nakładać na łatwość i wygodę samoobsługi użytkownika i na algorytmy zarządzania danymi aby zapewnić dostęp w całej sieci, do jej krańców, a nie tylko w jej centrum, do "długiego ogona" nie tylko do głowy.

Osiem cech Web 2.0

W artykule `Osiem cech Web 2.0` Sean Seton-Rogers wybrał osiem cech, które jego zdaniem charakteryzują zjawisko Web 2.0:

1. Możliwość nawiązywania kontaktów (ang. *Connectedness*)
   - Nie ma serwisu Web 2.0 bez elementów społecznościowych.
   - Produkt Web 2.0. musi ułatwiać ludziom nawiązywanie kontaktów.
2. Łamanie istniejących zasad (ang. *Shattering the exisitng*)
   - Serwisy klasyfikowane jako Web 2.0 powinny dawać użytkownikowi nową wartość, łamać schemat, który do tej pory królował na rynku.
   - Przykłady: Zopa.com (pożyczki pieniężne między użytkownikami, bez pośredników), BlaBlaCar (wspólne podróżowanie).
3. Partycypacja (ang. *Sharing*)
   - Serwisy Web 2.0 umożliwiają łatwe dzielenie się i wymienianie informacją, a także aktywne uczestnicwto.
   - Nawet osoba, która tylko konsumuje informacje, może to robić kiedy chce i jak chce, ma możliwośc komentowania i oceniania.
   - Przykłady: Facebook, Metacafe, MySpace, ...

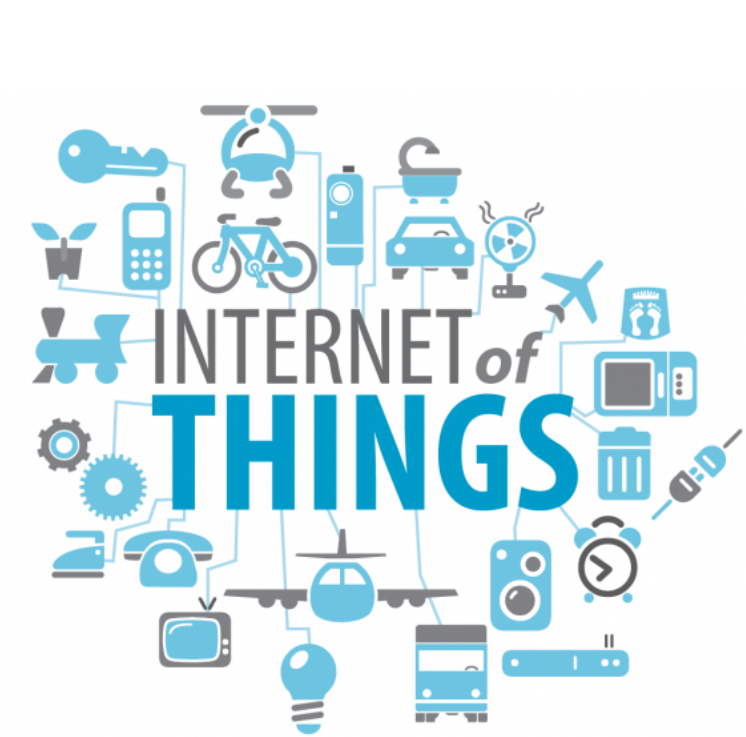
Osiem cech Web 2.0

4. Kreatwność (ang. *Creativity*)
   - Dzięki serwisom Web 2.0 użytkownicy mają dowolność twórczą i mogą dać upust swojej kreatywności.
   - W świecie Second Life właściwie wszystko od ubrań, poprzez przedmioty codziennego użytku czy nieruchomości jest tworzone przez użytkowników (`http://secondlife.com/`).
5. Niskie koszty (ang. *Low cost*)
   - Serwis Web 2.0 to dla funduszy VC (Venture Capital) wielokrotnie mniejsze inwestycje niż te sprzed kilku lat.
   - Średnia kwota przeznaczana przez VC na nowy serwis internetowy wynosi 2 mln USD, kiedyś było to ok. 20 mln USD.

Osiem cech Web 2.0

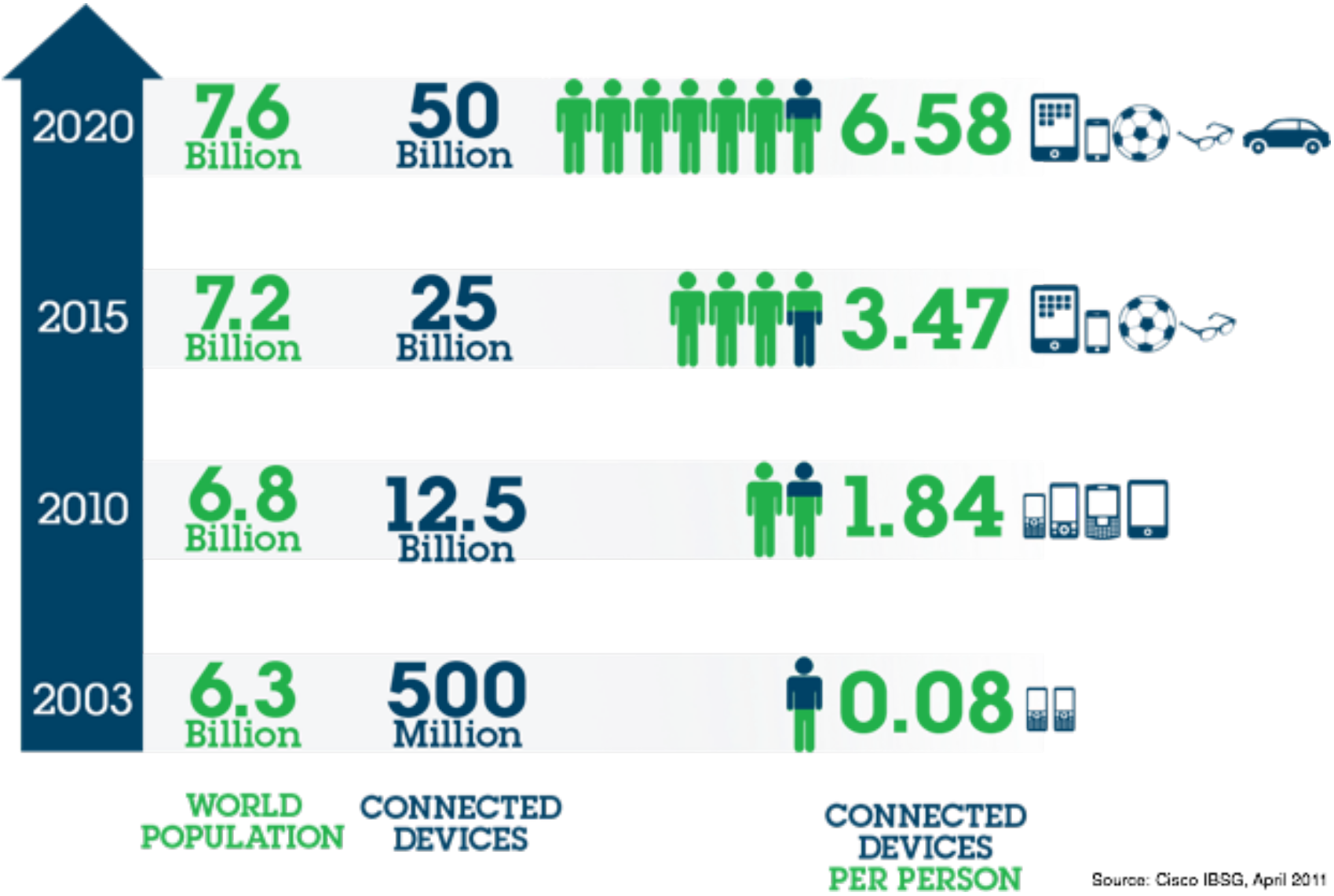
6. To czego chcę i kiedy chcę (ang. *What I want*)
   - Serwisy Web 2.0 dają użytkownikowi dowolność, dzięki technologiom takim jak RSS czy otwartym API, każdy może decydować jakie treści, kiedy i w jakiej konfiguracji chce konsumować.
7. Szybkość (ang. *Speed*)
   - Serwis Web 2.0 może powstać relatywnie szybko.
   - Jeśli pomysł jest dobry należy go intensywnie realizować, by nie wyprzedziła nas konkurencja.
8. "Śmiertelność" (ang. *Death*)
   - Znikanie wielu serwisów z rynku jest nieodłącznym elementem Web 2.0.
   - Na tak konkurencyjnym rynku przetrwają tylko produkty, które cieszą się największym powodzeniem wśród użytkowników.

Internet of Things

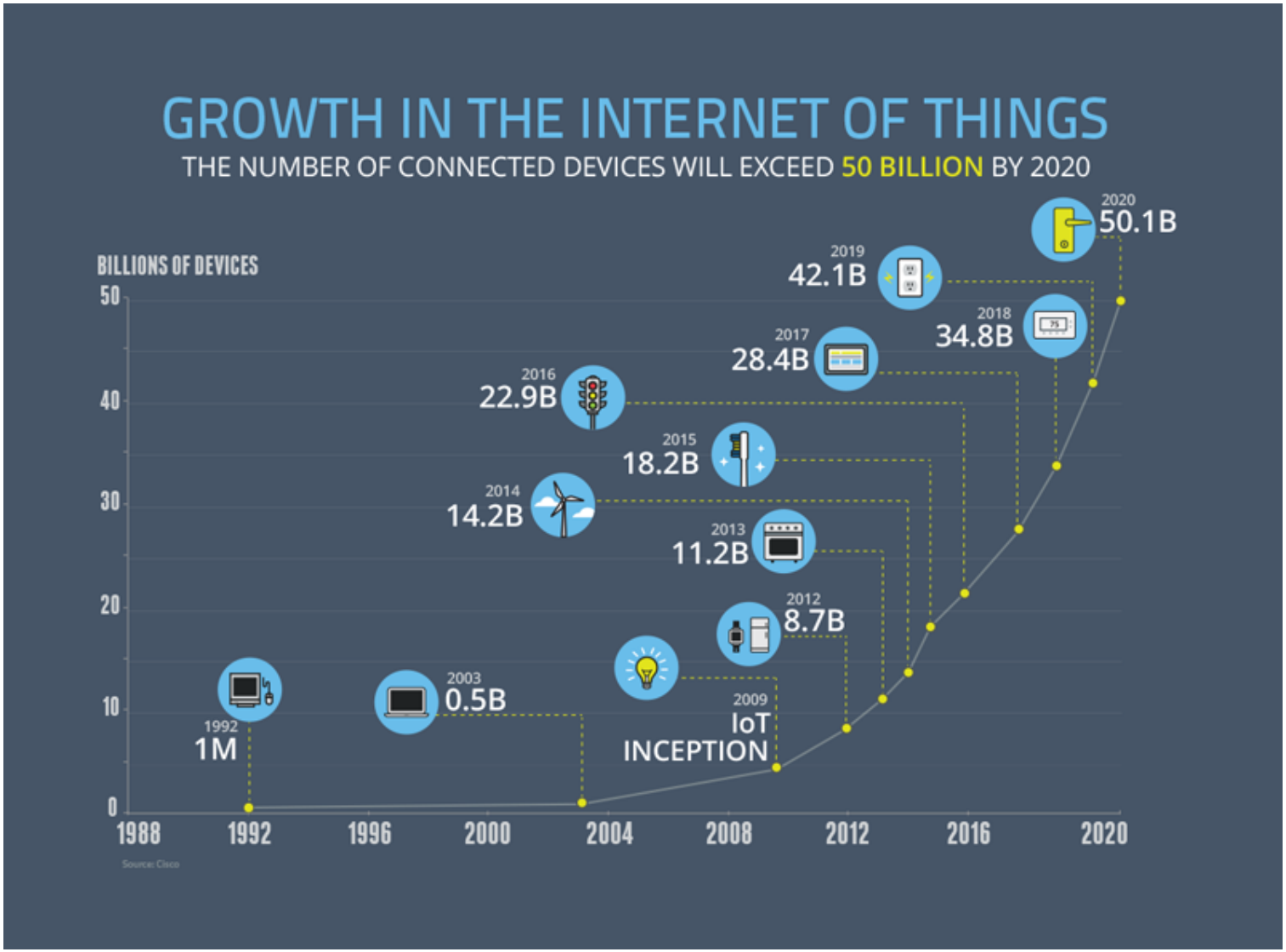


Internet rzeczy (ang. `Internet of Things`) to sieć obiektów fizycznych-urządzeń, pojazdów, budynków i innych rzeczy, wyposarzonych w elektronikę, oprogramowanie, sensory, połączenie do internetu, które pozwala tym obiektom na zbieranie i wymianę danych. Takie obiekty można zdalnie kontrolować i zarządzać.

Internet of Things - prognozy



Internet of Things - prognozy



https://www.ncta.com/whats-new/behind-the-numbers-growth-in-the-internet-of-things

Web^2 - podsumowanie

- Nowy kierunek, w którym rozwija się internet - kierunek olbrzymiej interakcji ze światem rzeczywistym.
- Dzisiaj internet nie jest już tylko platformą dla prowadzenia jakiegoś biznesu. Internet zaczyna stawać się realnym światem, zaczyna wchodzić w realne problemy światowe.
- Web^2 to internet, który wchodzi w konfrontacje ze światem rzeczywistym. Trzeba być tego świadomym.
- Daje to niesamowite możliwości kiedy będziemy chcieli dobrze je wykorzystać.

Dzisiaj sieć to nie zbiór statycznych stron HTML, które opisują coś znajdującego się świecie. Dzisiaj sieć to coraz bardziej świat. Każdy użytkownik, tysiące urządzeń zostawiają po sobie "cień informacyjny". Inteligentne wykorzystanie strumienia danych dostarczanych przez użytkownikow daje niesamowite możliwości, które mogą zmienić nasze myślenie czy postrzeganie.

Web 3.0?

Co będzie dalej?

- Web 3.0?
- Sieć semantyczna?
- Sieć odczuwająca, odbierająca świat zmysłami?
- Sieć portali społecznościowych?
- Sieć mobilna?
- Jakaś forma wirtualnej rzeczywistości?

Wszystkie powyższe i jeszcze więcej.

Web 3.0?

Web 3.0 to termin, który stworzono by opisać dalszą ewolucję Internetu oraz różnego rodzaju działań i koncepcji prowadzących do konwersji obecnego systemu przekazu wiedzy do modelu ogólnopojętej bazy danych.

Web 3.0 to koncepcja przetworzenia zawartości stron do wzorca czytanego przez różne (w tym nieprzeglądarkowe) aplikacje, systemy wykorzystujące sztuczną inteligencję, rozwiązania semantyczne oraz oprogramowanie pozwalające wizualizować oraz przetwarzać dane w trzech wymiarach. Chodzi tutaj głównie o sieci semantyczne czyli znaczeniowe.

https://www.ted.com/talks/tim_berners_lee_on_the_next_web

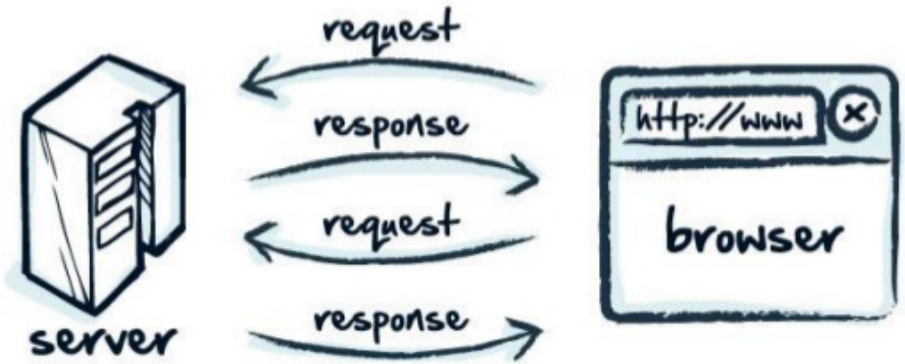
Sieci semantyczne

- Sieci semantyczne są modelem reprezentacji wiedzy.
- Bazują na założeniu, że pamięć ludzką najlepiej opisuje model asocjacyjny. **Terminy są wyjaśniane przez inne terminy**. Powstaje struktura pewnych powiązań, która może być zamknięta.
- Sieci semantyczne to **zbiory obiektów powiązane pewnymi relacjami**.
- Przykład: zdjęcie ma pozycję gdzie było zrobione, kto je robił, komu czy czemu je robił. Ile tu obiektów i ile powiązań!
- Sieci semantyczne **można przedstawiać za pomocą grafu**.
- Obiekty to węzły a relacje to krawędzie. Węzły i krawędzie mogą mieć wagi określające np. słuszność czy pewnośc danej relacji. W takich sieciach można prowadzić wnioskowanie.
- Obiekty w takim grafie mogą być obiektami rzeczywistymi, konceptualnymi (czynności, wydarzenia) albo deskryptorami (dodatkowy opis węzła, obiektu).

SPA i RESTful API

Tradycyjna strona internetowa

- Korzysta z HTML5, CSS3, JavaScript
- Bazuje na klasycznej architekturze klient-serwer
- Zawiera wiele stron



Zmiany w tworzeniu stron internetowych

Warunki:

- nowoczesne przeglądarki,
- rozwój języka JavaScript,
- większy nacisk położony na wygodę użycia.

Zmiany w aplikacjach internetowych:

- z punktu widzenia użytkownika działają jak aplikacje desktopowe,
- szybki i dużo bardziej interaktywny interfejs,
- potrafią działać nawet offline,
- działają na wielu platformach (RWD).

Single Page Application

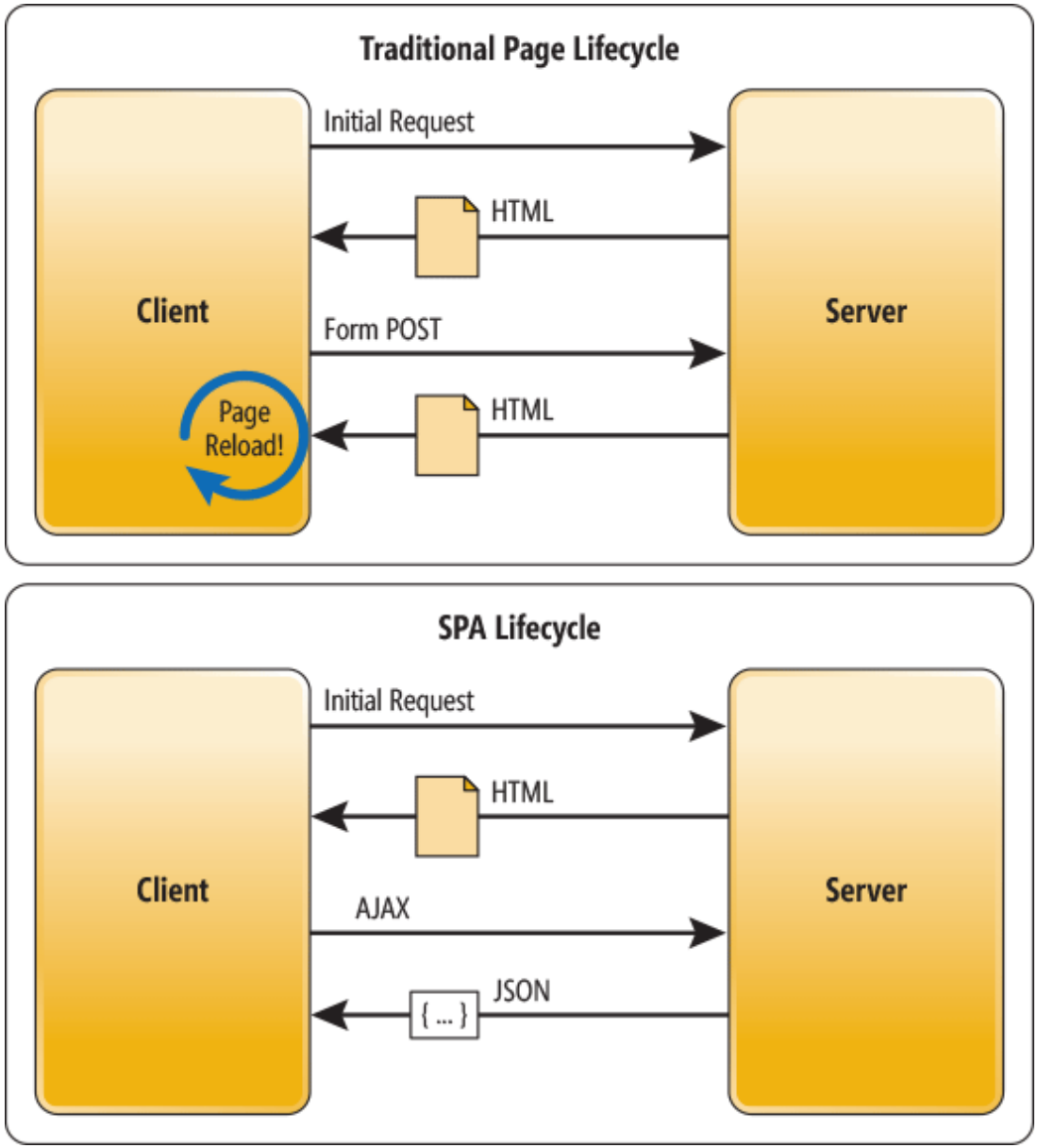
Cechy strony działającej w technologii SPA:

- jest to kolejny etap w zmianie ze stron tradycyjnych,
- wczytuje tylko jedną stronę,
- używa technologii AJAX i HTML5 - pozwala na tworzenia wygodnych, płynnych i responsywnych aplikacji internetowych bez konieczności przeładowywania strony,
- przykłady: Trello, Twitter, Facebook, GMail, ...

Zalety:

- nie ma mignięcia strony,
- treść renderowana z użyciem JavaScript,
- szybsza interakcja, lepsza przyjemność użycia.

Single Page Application



ECMAScript

ECMAScript

ECMAScript – ustandaryzowany przez ECMA skryptowy język programowania, którego najbardziej znane implementacje to JavaScript, JScript i ActionScript. Specyfikacja ta oznaczona jest jako ECMA-262 i ISO/IEC 16262.

Edycje ECMAScript I

Edycje ECMAScript - `http://www.ecma-international.org/publications/standards/Ecma-262.htm`

- 1 - czerwiec 1997 - pierwsza edycja,
- 2 - czerwiec 1998 - drobne zmiany edycyjne aby zachować zgodność ze standardem ISO/IEC 16262,
- 3 - grudzień 1999 - wyrażenia regularne, lepsza obsługa ciągów znakowych, nowe instrukcje sterujące, obsługa try/catch, lepsza definicja błędów, formatowanie wyjściowe danych numerycznych i inne ulepszenia,
- 4 - porzucona - nie została wydana, ze względu na podziały co do złożoności języka,
- 5 - grudzień 2009 - dodanie "strict mode", podzbiór pozwalający na bardziej szczegółową kontrolę błędów i unikanie konstrukcji, które często prowadziły do błędów, usunięcie niejednoznaczności z wersji 3, dostosowanie do rzeczywistych wymagań, obsługa JSON, metody dostępowe (get, set), ...
- 5.1 - czerwiec 2011 - edycja 5.1 jest w pełnej zgodności ze standardem ISO/IEC 16262:2011,

Edycje ECMAScript II

Edycje ECMAScript - `http://www.ecma-international.org/publications/standards/Ecma-262.htm`

- 6 - czerwiec 2015 - edycja szósta (ECMAScript2015), dodanie nowej składni do pisania skomplikowanych aplikacji (**klasy**, **moduły**), iteratory, pętle for/of, generatory wyrażeń w stylu języka python, arrow functions (funkcje anonimowe, wyrażenia lambda), dane binarne, tablice typowane, kolekcje, obietnice, ulepszenia matematyczne, ...
- 7 - czerwiec 2016 - wersja siódma (ECMAScript2016), kontynuacja zmian w języku: izolacja kodu, kontrola efektów i bibliotek/narzędzi z ECMAScript2015, dwie nowe cechy: operator potęgowania (**) i Array.prototype.includes,
- 8 - czerwiec 2017 - wersja ósma (ECMAScript2017, 885 stron specyfikacji), wprowadza funcje asynchroniczne (async), współdzielone bufory danych (`SharedArrayBuffer`) przy pracy współbierznej, globalną zmienną `Atomics` pozwalającą na synchronizacje czy opercje atomowe, indeksowanie zawartości obietków (metody `Object.entries` i `Object.values`) ...
- 9 - czerwiec 2018, przewidziane są asynchroniczne iteracje, rozszerzenie użycia operatora reszty/rozciągnięcia (`...`), rozszerzenie wyrażeń regularnych, ...

Implementacje ECMAScript

Niektóre implementacje ECMAScript

- JavaScript
  - SpiderMonkey - Firefox, silnik Gecko, Adobe Acrobat - ECMA-262, edycja 5.1, niektóre cechy z 6 i 7,
  - Chakra - Microsoft Edge - ECMA-262, edycja 5.1, niektóre cechy z 6.
  - V8 - Google Chrome, **Node.js**, Opera - ECMA-262, edycja 5.1, niektóre cechy z 6,
  - JavaScriptCore (Nitro) - WebKit, Safari, Qt 5 - ECMA-262, edycja 5.1, niektóre cechy z 6,
- JScript
  - JScript 9.0 - Internet Explorer, silnik Trident - ECMA-262, edycja 5.1,
  - JScript .NET - Microsoft .NET Framework - ECMA-262, edycja 3,
- ActionScript - Adobe Flash, Adobe Flex, Adobe AIR - ECMA-262, edycja 4,
- Adobe ExtendScript - produkty Adobe Creative Suite: InDesign, Illustrator, Photoshop, ... - ECMA-262, edycja 3,
- Rhino - standardowa edycja platformy Java,
- Nashorn - Java - ECMA-262, edycja 5.1.

Język JavaScript

Czym jest JavaScript?

JavaScript, JS – skryptowy język programowania, stworzony przez firmę Netscape, najczęściej stosowany na stronach internetowych. Język ten jest zorientowany obiektowo, a jego składnia jest zbliżona do języka C/C++. Odnaleźć można w nim paradygmaty programowania funkcyjnego oraz imperatywnego. Język ten wprowadza interaktywność do stron WWW.

Zastosowania

Do czego można wykorzystać język JavaScript?

- Tworzenie interaktywnych stron,
- sprawdzanie poprawności pól formularzy.
- obsługa elementów interfejsu strony,
- dynamiczne generowanie treści,
- realizacja efektów niedostępnych z poziomu HTML,
- reagowanie na zachodzące zdarzenia,
- i ...

Umieszczanie skryptów w kodzie strony.

Skrypty języka JavaScript można umieścić na stronie na dwa sposoby:

- jako skrypt zewnętrzny umieszczony w oddzielnym pliku.
- jako skrypt osadzony w kodzie HTML.

Umieszczanie bloków JavaScript:

- Liczba bloków <script> na stronie nie jest ograniczona.
- Nie jest zalecane umieszczanie bloku zawierającego skrypt JavaScript w ciele dokumentu, jednak jest to dopuszczalne.
- Kolejność umieszczania skryptów na stronie ma istotne znaczenie, są wykonywane sekwencyjnie.
- Renderowanie strony może zostać wstrzymane do momentu wykonania skryptu.

Mechanizmy udostępniane przez przeglądarki

Silniki JavaScript w przeglądarkach udostępniają programistom obiekty pozwalające między innymi na:

- modyfikacje zawartości dokumentu – obiekt: *document*,
- manipulowanie oknami oraz wywoływanie okien dialogowych – obiekt: *window*,
- pobieranie informacji o samej przegladarce – obiekt: *navigator*,
- obsługę zdarzeń.

AJAX

AJAX

AJAX (ang. `Asynchronous JavaScript and XML`) – technika tworzenia aplikacji internetowych, w której interakcja użytkownika z serwerem odbywa się bez przeładowywania całego dokumentu, w sposób asynchroniczny.

- bardziej dynamiczna interakcja z użytkownikiem niż w tradycyjnym modelu,
- uzyskanie nowych danych nie wymaga przeładowania całej strony,
- XMLHttpRequest - klasa pozwalająca na asynchroniczne przesyłanie danych (JavaScript, JScript, VBScript),
- XML, JSON, HTML, ...

XMLHttpRequest

XMLHttpRequest

XMLHttpRequest (XHR) – obiekt języków skryptowych (np. JavaScript, JScript lub VBScript) przeglądarek internetowych umożliwiający przesyłanie żądań do serwera WWW za pomocą protokołu HTTP.

- możliwość wysyłania żądań po załadowaniu się strony internetowej,
- wykorzystywane na potrzeby interakcji z użytkownikiem,
- dynamicznie zmieniająca się zawartość strony,
- jest podstawą technologij AJAX powszechnie dzisiaj wykorzystywanej...

Metody XMLHttpRequest

Najważniejsze metody:

- `abort()` - anulowanie żądania,
- `getAllResponseHeaders()` - zwraca kompletny zestaw nagłówków,
- `getResponseHeader(nazwaNaglowka)` - wartość konkretnego nagłówka,
- `open(metoda, URL)` - przygotowanie żądania,
- `open(metoda, URL, asynch, uzytkownik, haslo)` - przygotowanie żądania,
- `send(dane)` - wysyłanie żądania,
- `setRequestHeader(nazwaNaglowka, zawartoscNaglowka)` - dodanie nagłówka.

Własności XMLHttpRequest

Najważniejsze własności:

- `onreadystatechange` - referencja do funkcji wykonywanej przy zmianie `readyState`,
- `readyState` - stan obiektu, 4 oznacza zakończone zapytanie,
- `responseText` - odpowiedź jako łańcuch znaków,
- `status` - kod odpowiedzi HTTP jako numer (np. 404 albo 200).

XMLHttpRequest GET

```
<script>
function loadDoc() {
  var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (xhttp.readyState == 4) {
      document.getElementById("demo").innerHTML =
        xhttp.responseText;
    }
  };
  xhttp.open("GET", "https://localhost:3000/user", true);
  xhttp.send();
}
</script>
```

XMLHttpRequest - przykład, kod HTML

Kod HTML:

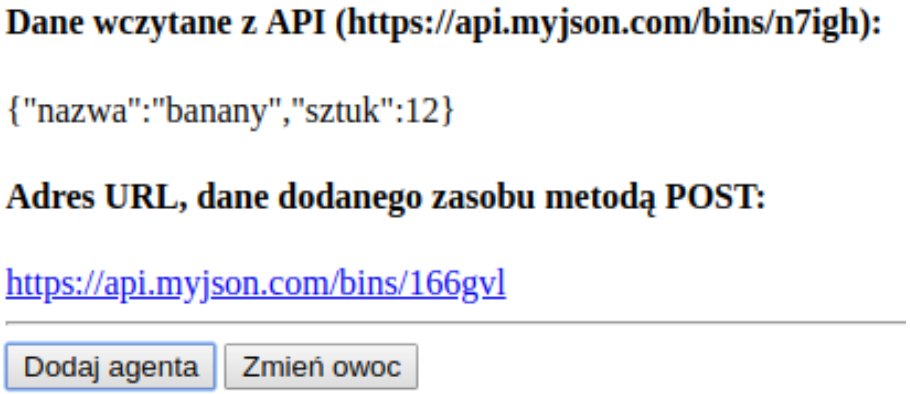
```
<h4>Dane wczytane z API (https://api.myjson.com/bins/n7igh):</h4>
<p id="demo">Tutaj będą dane wczytane z API</p>
<div>
    <h4>Adres URL, dane dodanego zasobu metodą POST:</h4>
    <a id="url">---</a>
</div>
<hr>
<button onclick="loadData()">Wczytaj dane</button>
<button onclick="addComment()">Dodaj komentarz</button>
```

XMLHttpRequest - przykład, kod JavaScript

```
function loadData() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (xhttp.readyState == 4) {
            document.getElementById("demo").innerHTML = xhttp.responseText;
        }
    };
    xhttp.open("GET", "https://api.myjson.com/bins/n7igh", true);
    xhttp.send();
}

function addComment() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (xhttp.readyState == 4) {
            var uri = JSON.parse(xhttp.responseText).uri;
            document.getElementById("url").innerHTML = uri;
            document.getElementById("url").href = uri;
        }
    };
    xhttp.open("POST", "https://api.myjson.com/bins", true);
    xhttp.setRequestHeader("Content-type", "application/json");
    xhttp.send(JSON.stringify({"userid":"2359", "content":"Moim zdaniem..."}) );
}
```

jQuery - przykład, kod HTML



Kod HTML:

```
  ...
  <script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.3.1.min.js">
  </script>
</head>
<body>
  <h4>Dane wczytane z API (https://api.myjson.com/bins/n7igh):</h4>
  <p id="demo">ASDF</p>
  <div>  <h4>Adres URL, dane dodanego zasobu metodą POST:</h4>
         <a id="url">---</a>
  </div>
  <hr>
  <button onclick="addAgent()">Dodaj agenta</button>
  <button onclick="updateFruit()">Zmień owoc</button>
  ...
```

jQuery - przykład, kod JavaScript

```
function loadAjaxData() {
  $("#demo").load("https://api.myjson.com/bins/n7igh");
}

function addAgent() {
  $.ajax( { contentType:"application/json", type:"POST",
            url:"https://api.myjson.com/bins",
            data: JSON.stringify({"agent":"James Bond","tajny_numer":007,
                                  "zdjecie":"https://.../bond.jpeg"}),
            success: function(result) {
                       $("#url").text(result.uri);
                       $("#url").attr("href", result.uri)
                     }
          } );
}

function updateFruit() {
  $.ajax( { contentType:"application/json", type:"PUT",
            url:"https://api.myjson.com/bins/n7igh",
            data: JSON.stringify( {"nazwa":"jabłka","ilość (kg)":3.5} ) } );
}

loadAjaxData();
```

Architektura REST

REST

Representational State Transfer (zmiana stanu poprzez reprezentacje) – styl architektury oprogramowania wywiedziony z doświadczeń przy pisaniu specyfikacji protokołu HTTP dla systemów rozproszonych.

Termin REST został wprowadzony w roku 2000 przez Roya'a Fieldinga w ramach jego rozprawy doktorskiej. R. Fielding jest jednym z czołowych twórców protokołu HTTP, współuczestniczył w rozwoju standardu HTML, jest jednym z założycieli projektu Apache HTTP Server.

REST w kilku słowach

Representational State Trasfer można określić jako styl architektoniczny, metaarchitektura, metodologia.

Założenia REST:

- architektura klient–serwer,
- bezstanowa komunikacja (ang. *stateless*),
- buforowanie zasobów (ang. *cachable*),
- jednolity interfejs dostępu.

REST przyjął się w sieci ponieważ jest prostszą i wygodniejszą alternatywą dla SOAP czy języka web-serwisów WSDL.

Żadania AJAX są podstawą dla usług sieciowych typu RESTful.

REST to raczej styl API niż specyfikacja.

Buforowanie zasobów - klient i serwer czy strony pośredniczące mogą buforować odpowiedzi (określa się czy ma być ona buforowana). Dobrze zarządzane buforowanie częściowo albo całkowicie redukuje część interakcji między klientem a serwerem.

Jednolity interfejs dostępu

Co to jest jednolity interfejs dostępu?

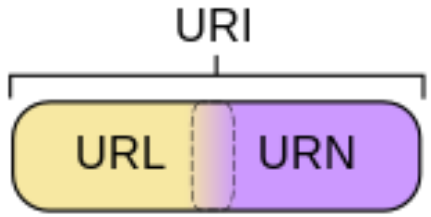
- jednolita identyfikacja/adresacja zasobow
- manipualcja zasobami poprzez ich reprezentację
- samoopisujące się wiadomości (bezstanowość)
- powiązania pomiędzy zasobami (wyrażone w reprezentacjach)

- Reprezentacja zasobów z reguły przyjmuje postać dokumentów JSON, dawniej XML.
- Zasoby posiadają jednoznaczny identyfikator np. URI.
- Serwer nie przechowuje informacji o stanie, to klient odpowiada za przejścia pomiędzy różnymi stanami.

Co to jest URI?

URI (ang. *Uniform Resource Identifier*) standard określający sposób identyfikacji zasobów w sieci. URI jest często mylone z URL, które jest specyficzną odmianą URI – określającą lokalizację konkretnego zasobu w sieci. URI może być zbudowany z dwóch członów:

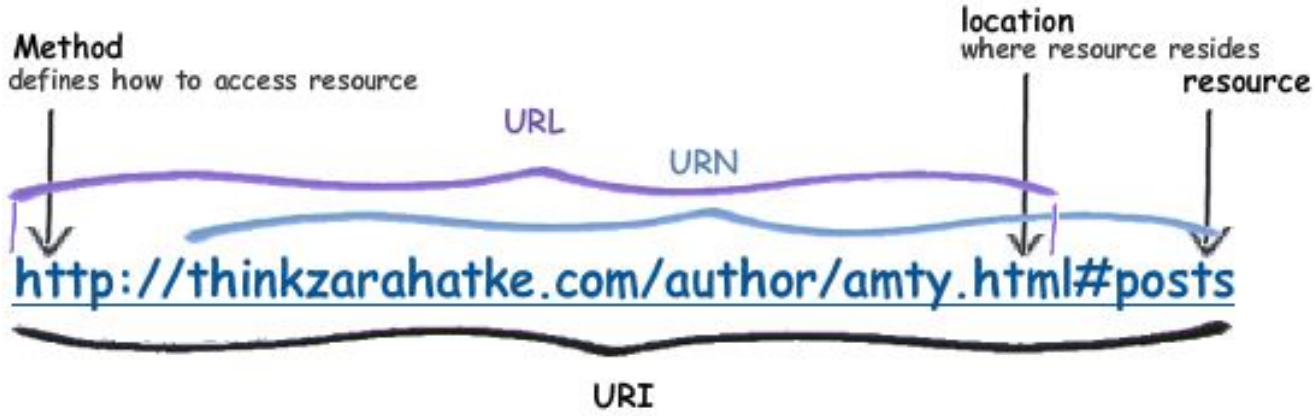
- lokatora – URL
- nazwy – URN



Standard URI został opracowany w celu identyfikacji wszystkiego: zasobów na dysku, miejsc w sieci www, kontaktów w książce adresowej.

- URL (ang. *Uniform Resource Locator*) – oznacza ujednolicony format adresowania zasobów (informacji, danych, usług) stosowany w Internecie i w sieciach lokalnych.
- Przykład:
  `http://en.wikipedia.org/wiki/Uniform_resource_locator`

Co to jest URI?



- URL z definicji wskazuje lokalizację, tj. miejsce, z którego dany zasób można ściągnąć (adres) i sposób, w jaki można to zrobić (protokół, np. http, ftp, ...).
- URI służy tylko do identyfikacji i niekoniecznie musi wskazywać miejsce skąd coś można ściągnąć.
- Standardowo URI strony www (np. http://www.pcz.pl) jest utożsamiany z jej URL. Stąd wynika fakt, że te dwa terminy są często używane zamiennie.
- URN (ang. *Uniform Resource Name*) - ujednolicony format nazw zasobów
- Przykładem URN mogą być numeracja ISBN: `urn:isbn:0451450523` albo ISSN `urn:ISSN:0167-6423`.

Cele REST

Architektura REST wyróżnia się kilkoma istotnymi cechami. Jednymi z najważniejszych są:

- skalowalność interakcji komponentów,
- ogólność interfejsów,
- niezależność wdrażania komponentów (protokół można aktualizować oraz rozszerzać),
- możliwość wprowadzania usług pośredniczących (zwiększanie bezpieczeństwa).

REST

W usłudze typu REST operacje są wyrażone w postaci połączenia:

- metody HTTP - wskazuje na konkretną operację do wykonania,
- adresu URI - wskazuje obiekt danych, na których będzie wykonana operacja.

Przykłady:

```
http://serwer.domena.pl/magazyn/towary
http://moja.firma.pl/pracownicy/tomek
http://localhost:5500/produkty
http://localhost:5500/produkty/dd4c567a0aa2a807
```

- W usłudze REST specyfikacja adresu URI leży w gestii programisty.
- Adres URL może ujawniać wewnętrzną strukturę danych - odzwierciedla to powiązanie pomiędzy komponentami.
- Mapowanie między formatem adresu URL a strukturą danych powinno odbywać się po stronie serwera.

Manipulacja zasobami

Usługi REST służą do przeprowadzania operacji tworzenia, odczytu, uaktualniania lub usunięcia (Create, Read, Update, Delete - CRUD).

Do manipulacji zasobami wykorzystywane są metody protokołu HTTP:

Metoda	Opis
GET	Pobranie danych wskazanych w adresie URI.
PUT	Uaktualnienie obiektu wskazanego w adresie URI.
POST	Utworzenie nowego obiektu danych pod wskazanym adresem URI. Zwykle korzysta się z formularza sieciowego jako źródła danych.
DELETE	Usunięcie obiektu wskazanego w adresie URI.

- CRUD – to cztery podstawowe operacje oferowane użytkownikowi aplikacji.
- Metoda POST bardzo często może służyć również jako metoda aktualizująca, jeśli obiekt pod wskazanym adresem już istnieje.
- Sposób obsługi metod protokołu HTTP operujących na danych zależy ostatecznie od programisty.

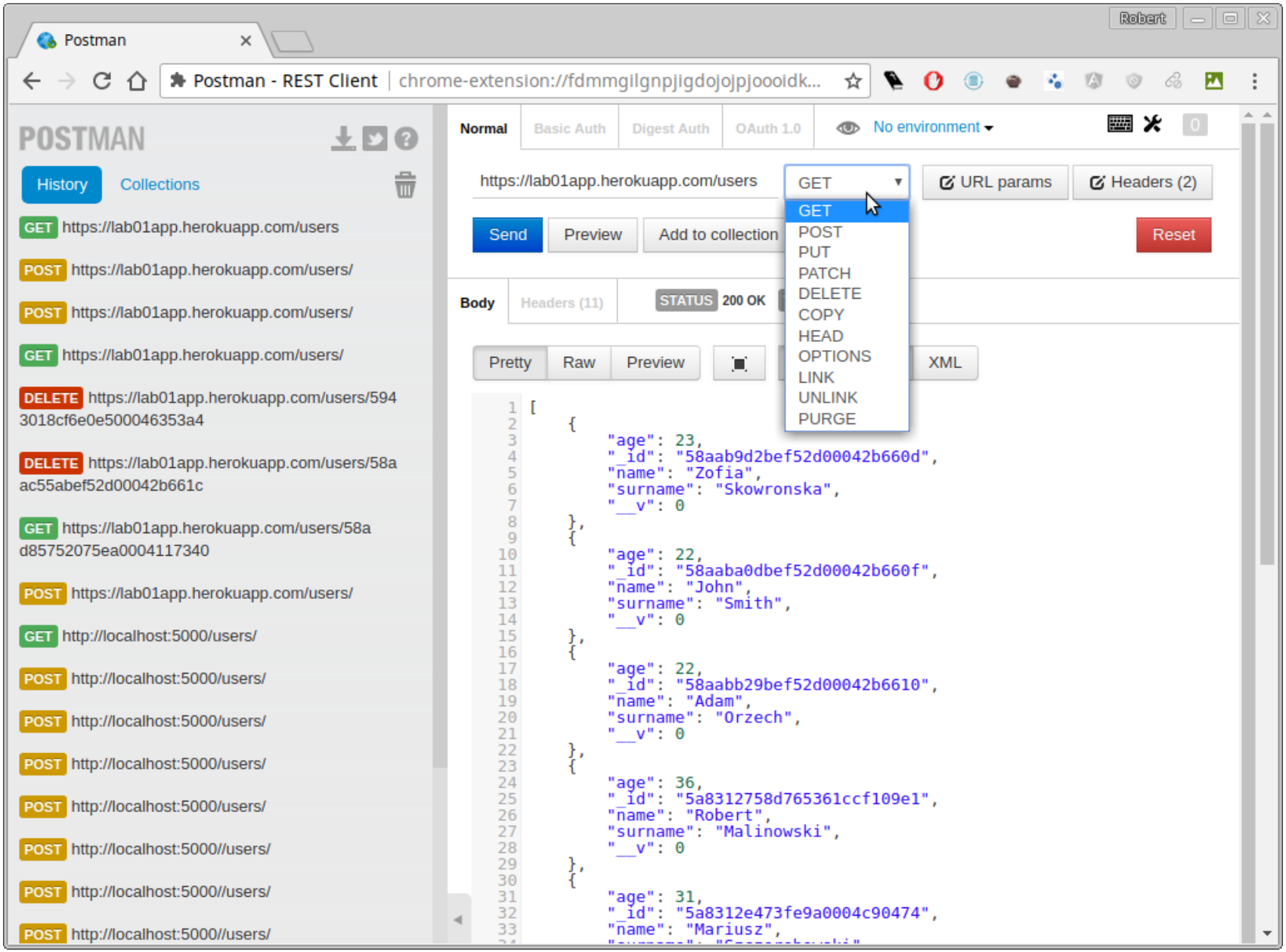
Natura metod protokołu HTTP

Wykorzystując metody protokołu HTTP staramy się wykorzystać je zgodnie z naturą ich specyfikacji:

- GET - nullipotent
  Operacje wykonywane w odpowiedzi na metodę GET powinny tylko pobierać dane, nie modyfikować ich. Wielokrotnie wykonywanie żądań GET nie powinno zmieniać stanu serwera.
- PUT i DELETE - idempotent (idem + potence : same + power)
  Wiele identycznych żądań ma taki sam efekt jak pojedyncze żądanie (raz usunięty czy zaktualizowany obiekt nie zmienia się przy kolejnej takiej samej operacji).
- POST - nie jest ani nullipotent ani idempotent
  Metoda POST jest najczęście używana do tworzenia i aktualizowania.

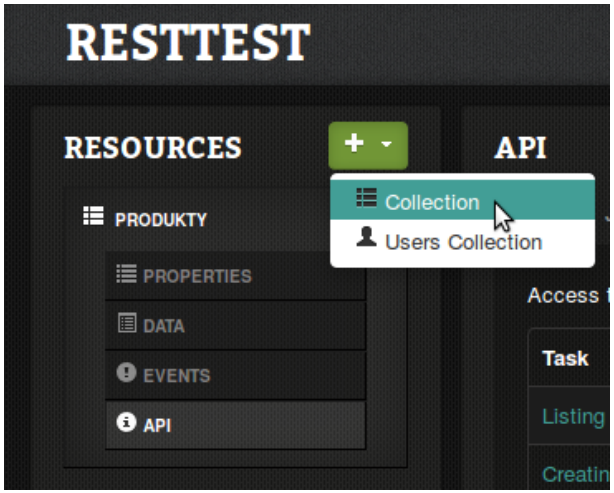
- Korzystając z dostępnego API trzeba wiedzieć jak zdefiniowano obsługę poszczególnych metod.

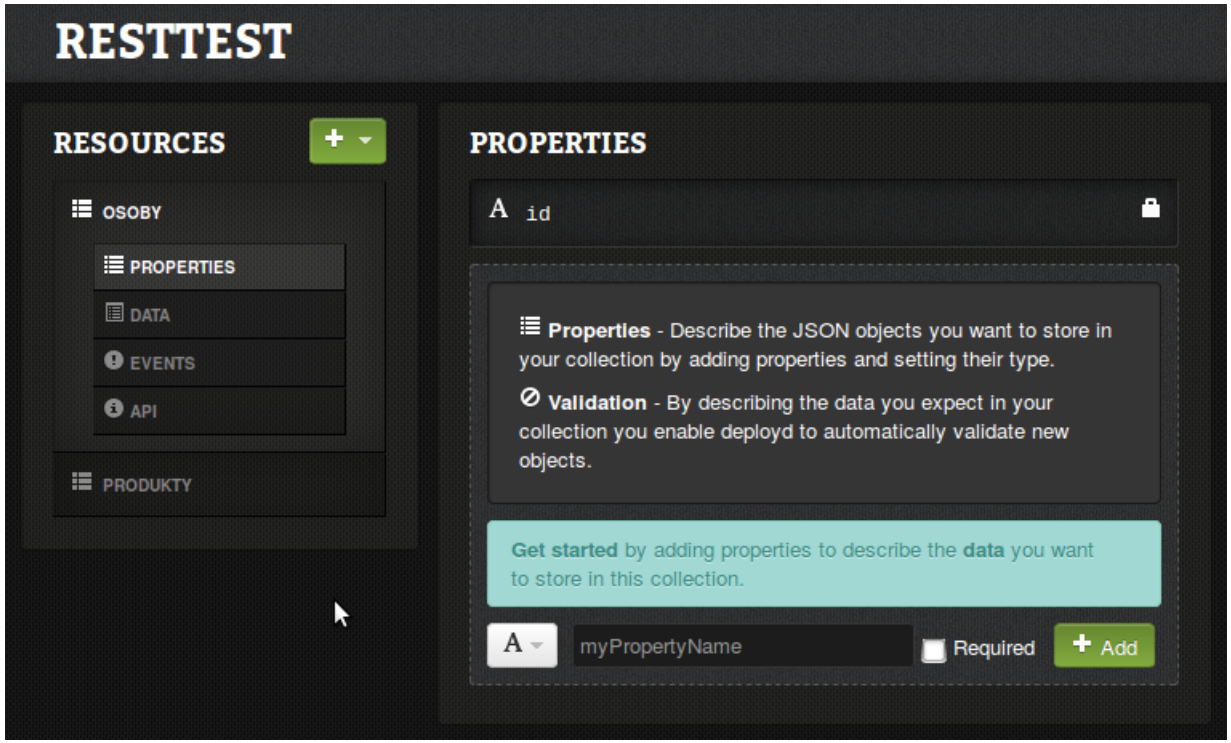
Postman


Postman
Robert
Postman - REST Client
chrome-extension://fdmmgilgnpjigdojojpjoooidk...
POSTMAN
History
Collections
GET https://lab01app.herokuapp.com/users
POST https://lab01app.herokuapp.com/users/
POST https://lab01app.herokuapp.com/users/
GET https://lab01app.herokuapp.com/users/
DELETE https://lab01app.herokuapp.com/users/5943018cf6e0e500046353a4
DELETE https://lab01app.herokuapp.com/users/58aac55abef52d00042b661c
GET https://lab01app.herokuapp.com/users/58ad85752075ea0004117340
POST https://lab01app.herokuapp.com/users/
GET http://localhost:5000/users/
POST http://localhost:5000/users/
POST http://localhost:5000/users/
POST http://localhost:5000/users/
POST http://localhost:5000/users/
POST http://localhost:5000//users/
POST http://localhost:5000//users/
POST http://localhost:5000//users/
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
https://lab01app.herokuapp.com/users
GET
URL params
Headers (2)
Send
Preview
Add to collection
Reset
GET
POST
PUT
PATCH
DELETE
COPY
HEAD
OPTIONS
LINK
UNLINK
PURGE
Body
Headers (11)
STATUS 200 OK
Pretty
Raw
Preview
XML
[
{
"age": 23,
"_id": "58aab9d2bef52d00042b660d",
"name": "Zofia",
"surname": "Skowronska",
"__v": 0
},
{
"age": 22,
"_id": "58aaba0dbef52d00042b660f",
"name": "John",
"surname": "Smith",
"__v": 0
},
{
"age": 22,
"_id": "58aabb29bef52d00042b6610",
"name": "Adam",
"surname": "Orzech",
"__v": 0
},
{
"age": 36,
"_id": "5a8312758d765361ccf109e1",
"name": "Robert",
"surname": "Malinowski",
"__v": 0
},
{
"age": 31,
"_id": "5a8312e473fe9a0004c90474",
"name": "Mariusz",

Deployd - jak szybko zbudować własne API

- Deployd (`http://deployd.com/`) - proste narzędzie pozwalające na szybkie utworzenie własnego API.





Przykładowe API dla kolekcji `produkty`

Zadanie	Metoda	Ścieżka	Przyjmuje	Zwraca
Pobieranie danych	GET	/produkty	nic	tablica obiektów
Tworzenie obiektu	POST	/produkty	pojedynczy obiekt	zapisany obiekt (albo błąd)
Pobieranie obiektu	GET	/produkty/<id>	nic	pojedynczy obiekt
Aktualizacja obiektu	PUT	/produkty/<id>	pojedynczy obiekt	zapisany obiekt (albo błąd)
Usuwanie obiektu	DELETE	/produkty/<id>	pojedynczy obiekt	nic

Inne metody protokołu HTTP do wykorzystania:
HEAD, TRACE, OPTIONS, CONNECT, PATCH.

Dane dostępne przez HTTP

http://localhost:5500/produkty/


```
[{"nazwa":"Piłka","opis":"Zatwierdzone przez FIFA rozmiar i waga.",
  "kategoria":"Piłka nożna","cena":34,"id":"f31ddf86acde5941"},
 {"nazwa":"Stadion","opis":"Składany stadion na 35 000 osób.",
  "kategoria":"Piłka nożna","cena":820000,"id":"8db5dce5281b0849"},
 {"nazwa":"Namiot","opis":"Bardzo fajny namiot dla dwóch osób",
  "kategoria":"Turystyka","cena":533,"id":"dd4c567a0aa2a807"},
 {"nazwa":"Nóż","opis":"Bardzo ostry i bardzo przydatny :)",
  "kategoria":"Turystyka","cena":89.99,"id":"825f09c954395857"},
 {"cena":932.5,"kategoria":"Sporty wodne","nazwa":"Kajak jednoosobowy",
  "opis":"Łódka przeznaczona dla jednej osoby.","id":"68ea03bf5d7d0936"}]
```


http://localhost:5500/produkty/dd4c567a0aa2a807


```
{"nazwa":"Namiot","opis":"Bardzo fajny namiot dla dwóch osób",
 "kategoria":"Turystyka","cena":533,"id":"dd4c567a0aa2a807"}
```


http://localhost:5500/robert/me


```
[{"username":"rperlinski","id":"d9a9ed20e0dec9e0"}]
```

Przydatne linki

- REST czy RESTful - http://stackoverflow.com/questions/1568834/whats-the-difference-between-rest-restful
- Co to jest MEAN Stack - http://garywoodfine.com/what-is-the-mean-stack/
- MEAN Stack 2 - http://advaitsolutions.in/services/web/mean-stack/
- SPA - http://www.benfimedia.pl/2013/12/09/single-page-applications-frontend-first-inne-podejscie-do-tworzenia-aplikacji-internetowych/
- Heroku - hosting JavaScript i MongoDB - https://www.heroku.com/nodejs
- express-generator - http://expressjs.com/en/starter/generator.html
- mongoose - http://mongoosejs.com/index.html

Źródła

W wykładzie wykorzystano informacje dostępne w internecie:

- artykuł Osiem cech Web 2.0,
- artykuł Web Squared: Web 2.0 Five Years On
- artykuł Sieci semantyczne
- Prezentacja o SPA
- artykuł ASP.NET - Single-Page Applications
- Wikipedia: Web 2.0, Semantic Web, Folksonomia, Croudsourcing, Bańka internetowa, i inne...