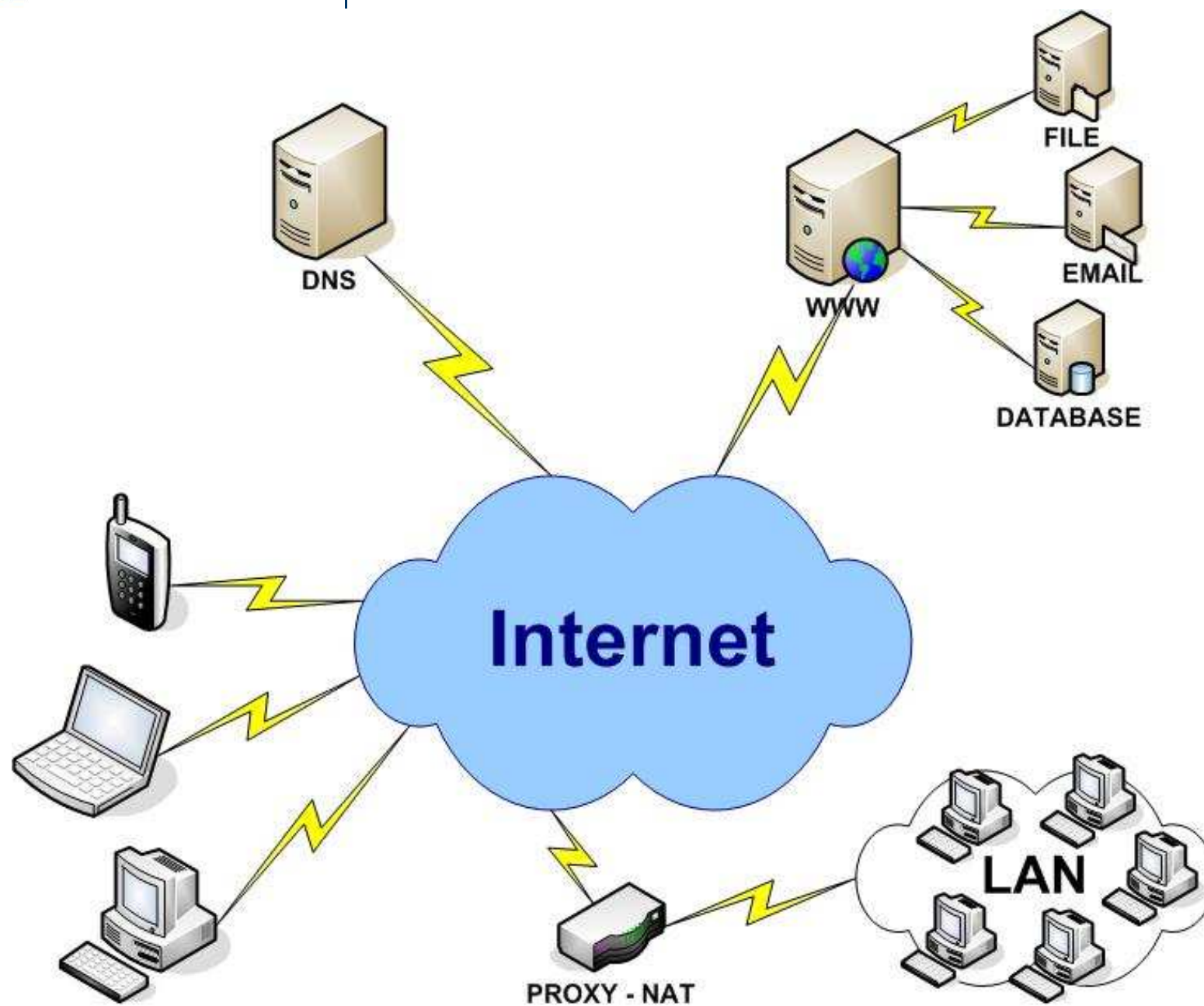




Technologie internetowe w aplikacjach mobilnych

Autor Wojciech Gumiński





Spis treści

- [Zasady zaliczenia](#)
- [Wprowadzenie](#)
- [Standardy webowe](#)
- [Języki programowania internetowego](#)
- [Technologie - Ajax](#)
- [JSON - JavaScript Object Notation](#)
- [Biblioteka jQuery](#)
- [Biblioteki Bootstrap, jQueryUI, AngularJS i Angular Material](#)
- [Model MVC](#)
- [Bezpieczeństwo – Uwierzytelnianie](#)
- [Bezpieczeństwo – Formularze i SQL Injection](#)
- [Wydajność – Web Caching](#)
- [Wydajność – Web Switching](#)



Zasady zaliczenia

- Laboratorium
 - Ćwiczenia laboratoryjne
 - Realizacja 5 miniprojektów w czasie zajęć (25pkt.).
 - Wykład
 - Test (15 pkt.).
- Ocena
 - Zaliczenie sumą punktów z liniową skalą ocen tzn.:
>50% 3.0, >60% 3.5, >70% 4.0, >80% 4.5, >90 5.0
 - Oceny pozytywne nie podlegają poprawie.
 - Punkty zdobyte w terminach poprawkowych uśredniają się z wcześniej zdobytymi.



Literatura

- <http://www.w3.org>
- <http://www.w3schools.com>
- <https://jquery.org>
- <https://angularjs.org>
- <http://getbootstrap.com/>
- Podręczniki do HTLM, CSS, XML, PHP, SQL, Javascript itp.

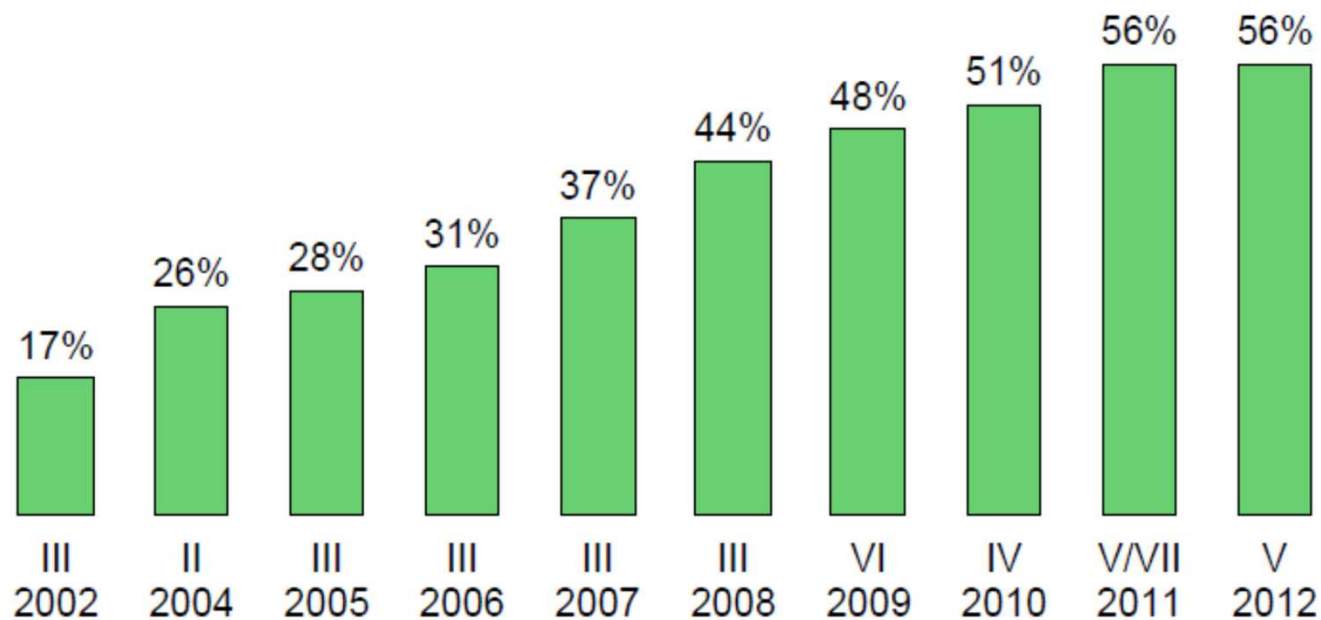


Internet w Polsce

CBOS

RYS. 1. CZY KORZYSTA PAN(I) Z INTERNETU (STRON INTERNETOWYCH, POCZTY E-MAIL, KOMUNIKATORA INTERNETOWEGO ITP.) PRZYNAJMNIEJ RAZ W TYGODNIU?*

Odpowiedzi twierdzące



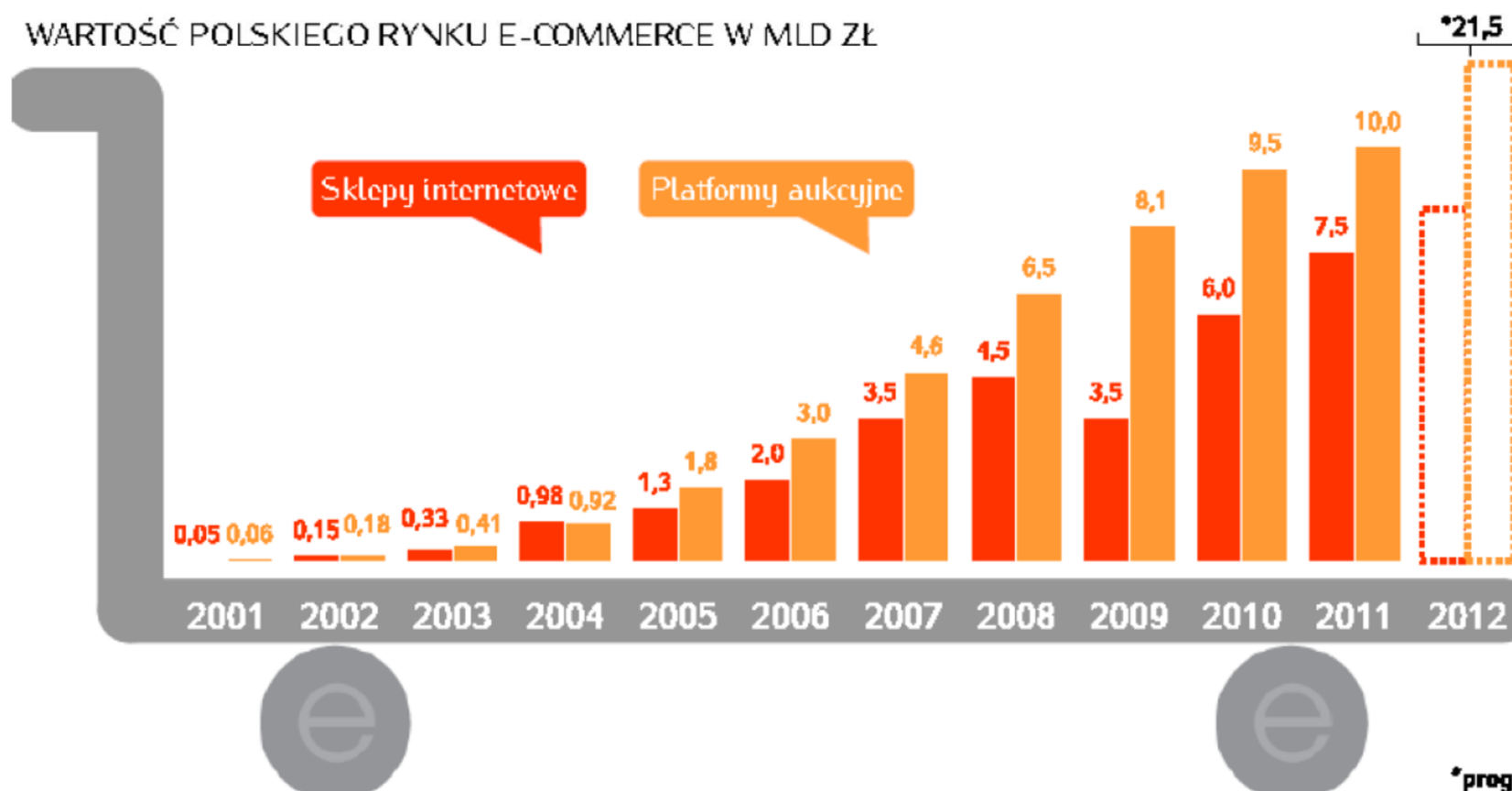


WARTOŚĆ RYNKU E-COMMERCE W POLSCE

Wartość rynku e-commerce w Polsce [2001-2012]

Według szacunków SMB, Kelkoo, Forrester Research łączna wartość rynku e-commerce w roku 2012 to około **21,5 mld PLN**

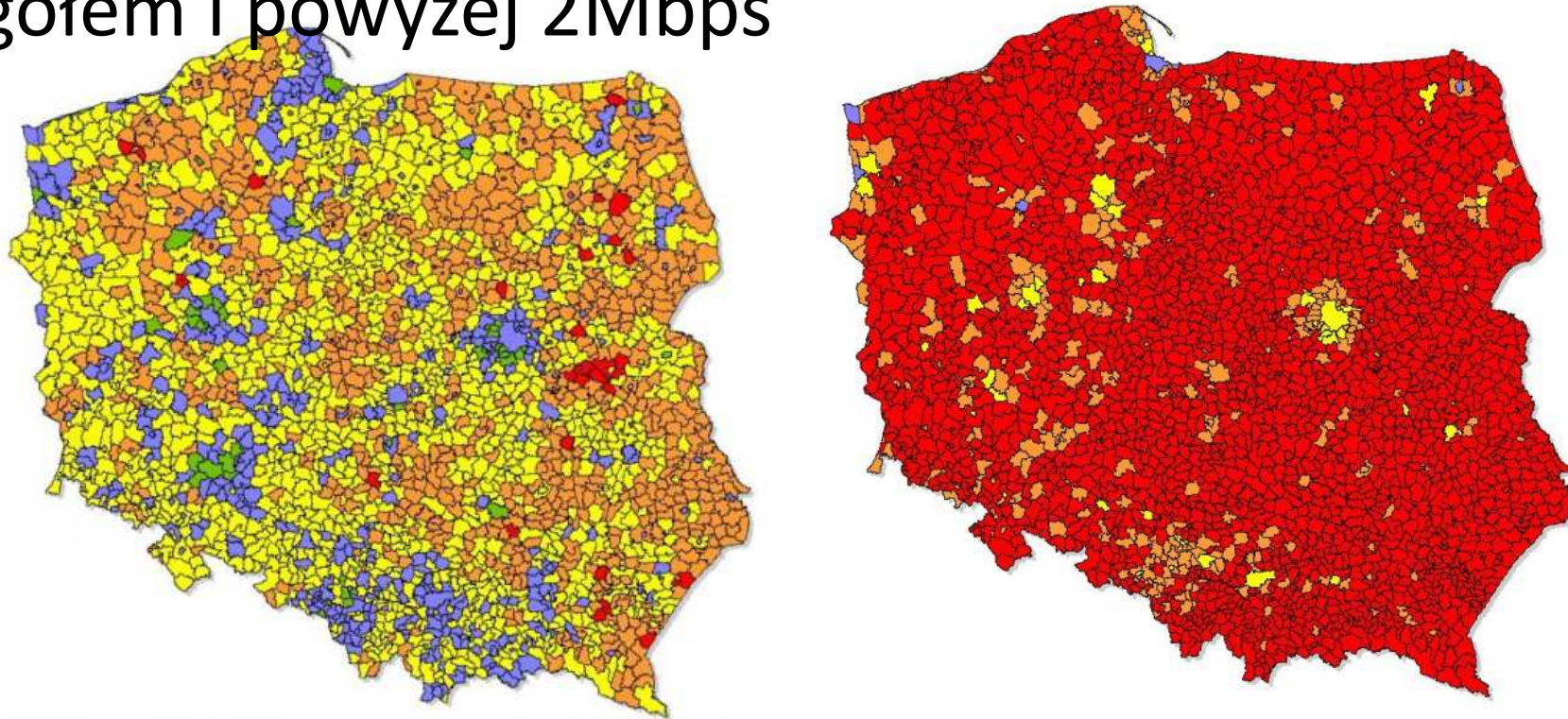
WARTOŚĆ POLSKIEGO RYNKU E-COMMERCE W MLD ZŁ



*prognoza

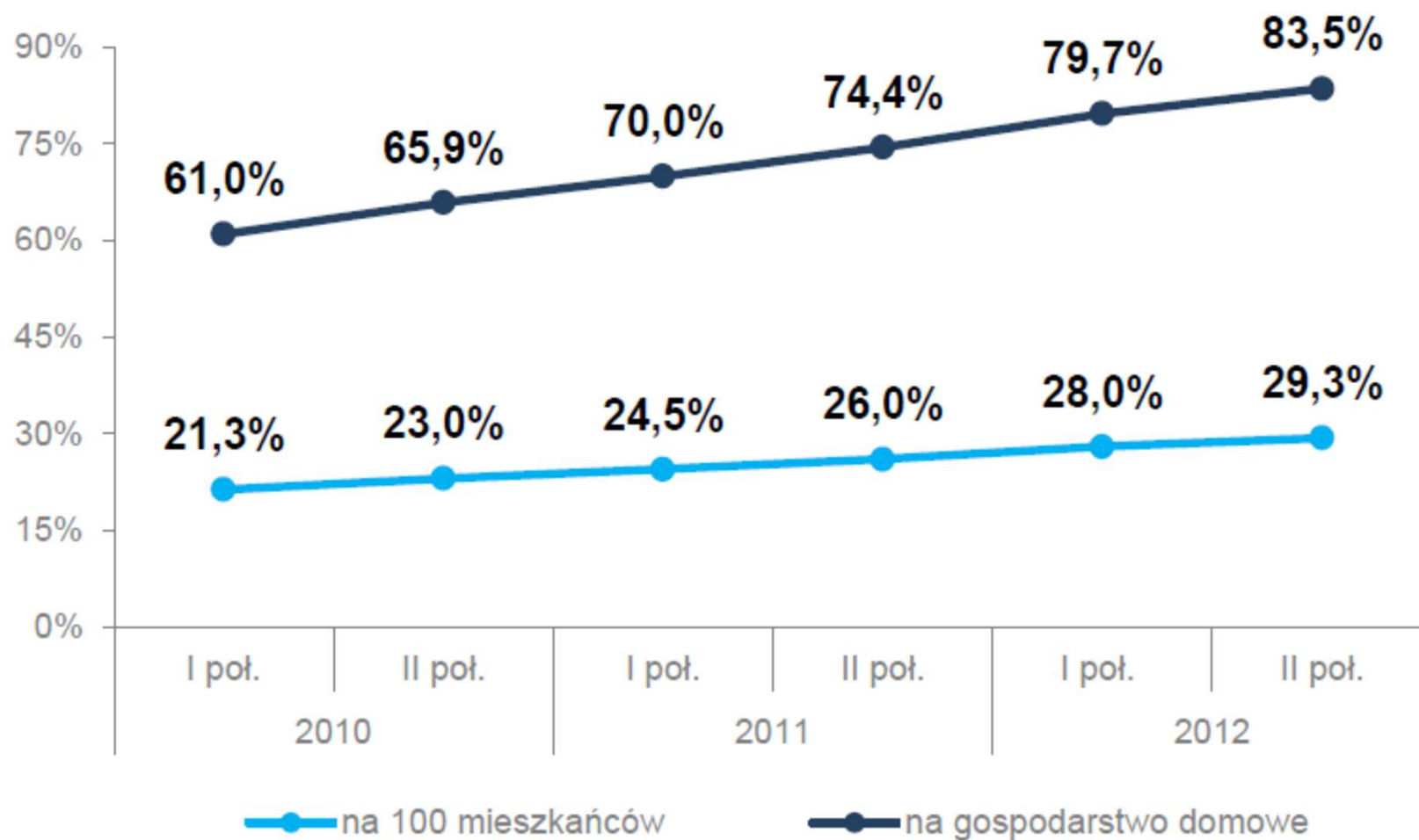
Dostępność Internetu w Polsce (UKE IX 2010)

Ogółem i powyżej 2Mbps



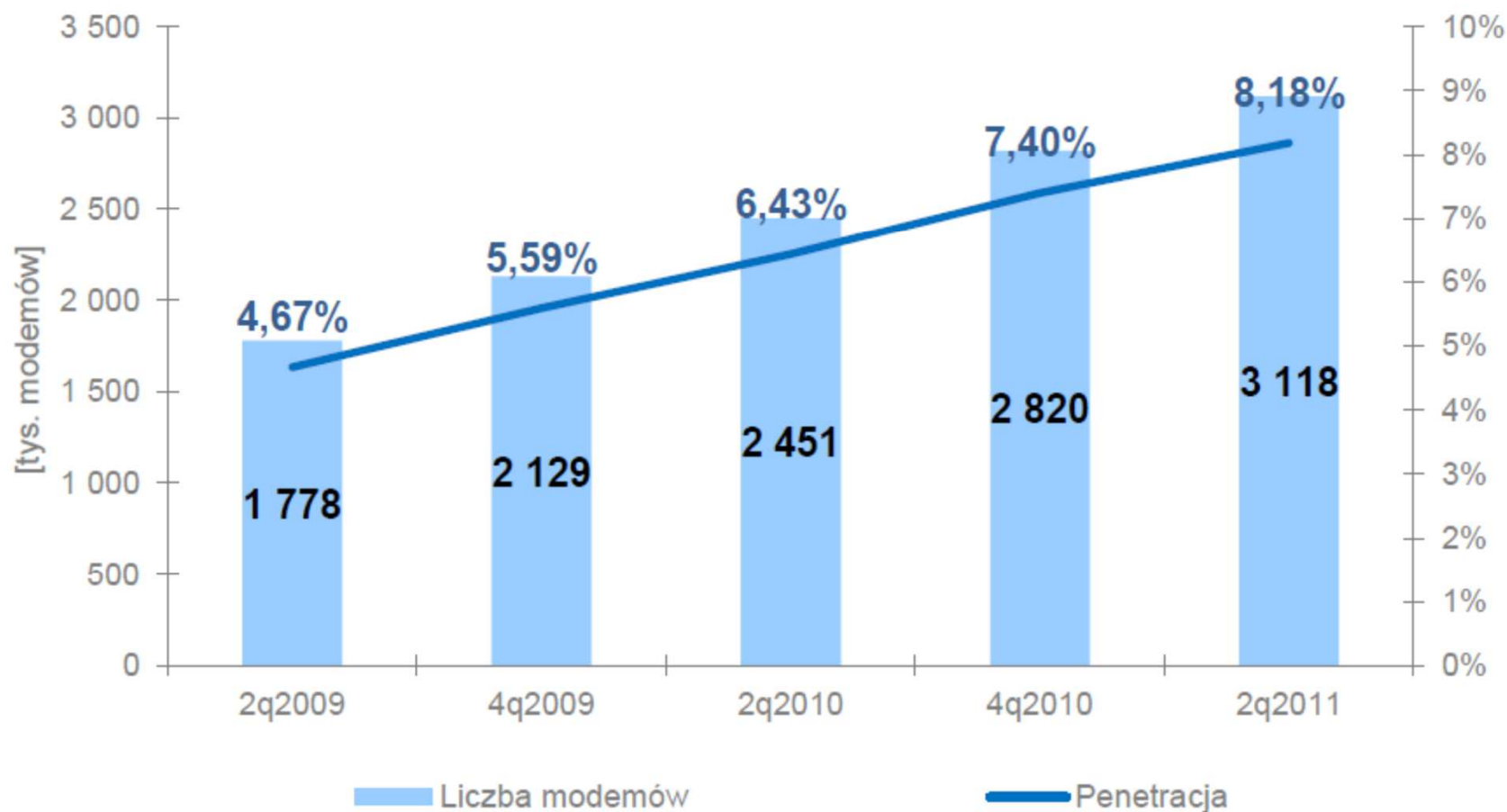
- kolor czerwony – gminy o najniższym wskaźniku dostępności od 0 do 10%,
- kolor pomarańczowy – gminy o wskaźniku dostępności od 10 do 30%,
- kolor żółty – gminy o wskaźniku dostępności od 30 do 50%,
- kolor niebieski – gminy o wskaźniku od 50 do 70%,
- kolor zielony – gminy o wskaźniku dostępności od 70 do 100%.

Wykres 1. Wskaźniki penetracji Internetu szerokopasmowego



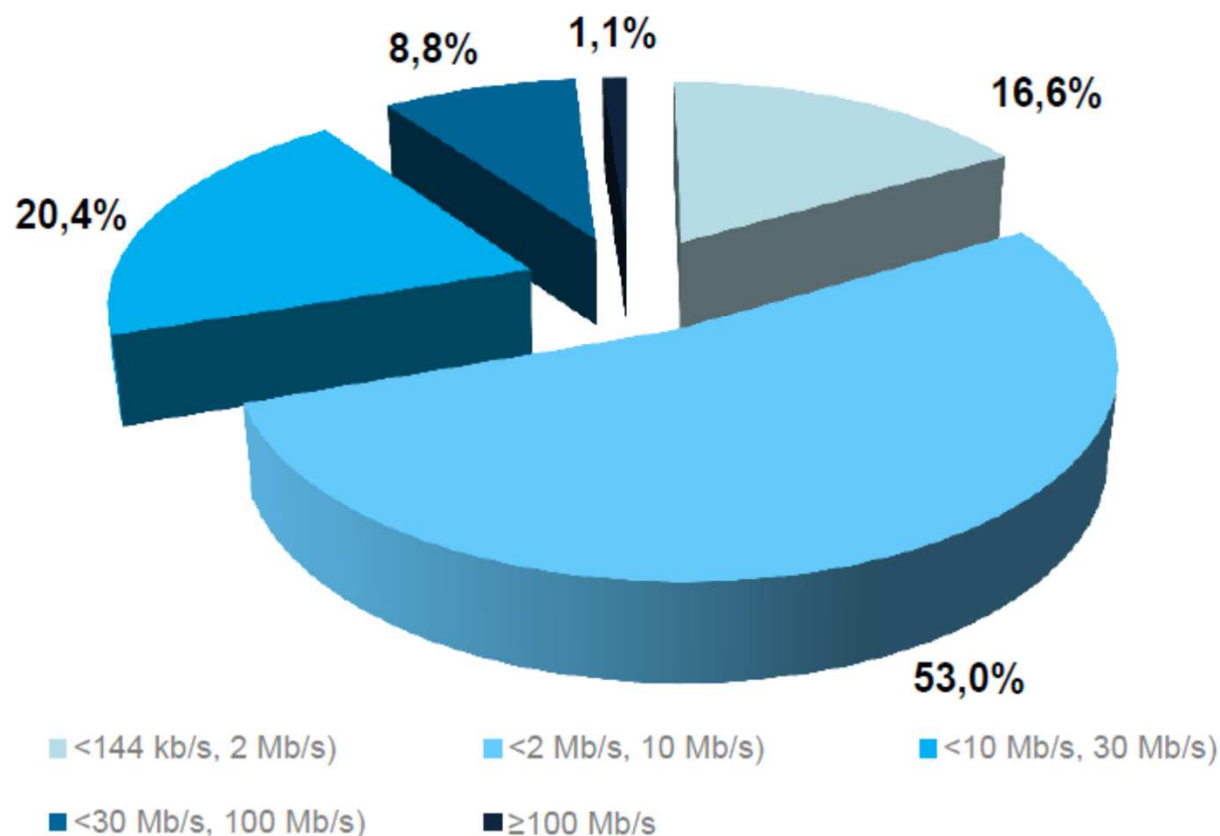
Źródło: UKE.

Wykres 1. Liczba modemów oraz penetracja rynku usługami mobilnego Internetu w Polsce



Źródło: UKE

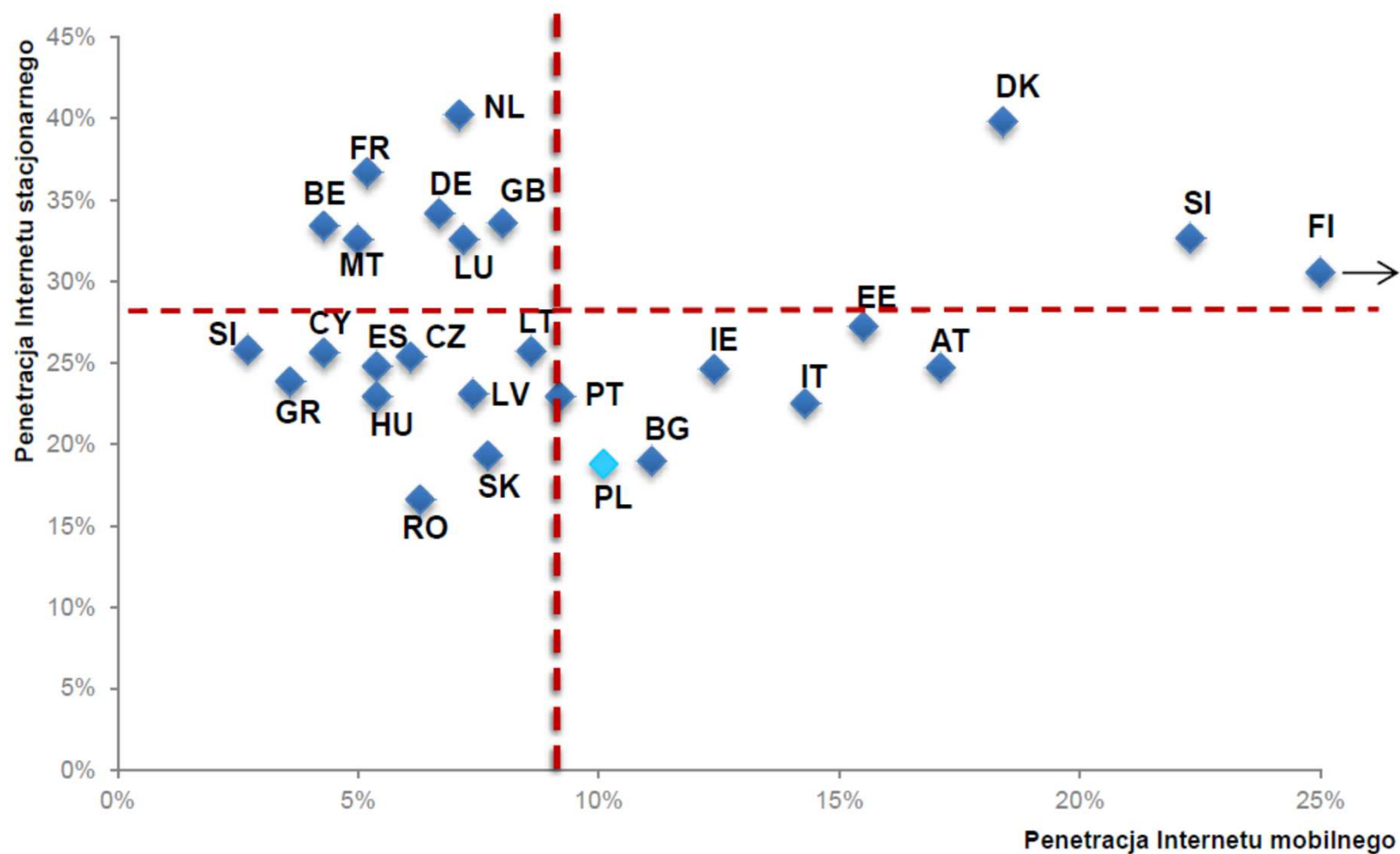
Wykres 12. Udział łączy według przepływności



Źródło: UKE.

Uwaga: uwzględniono łącza, których przepływność została określona przez operatora. Zestawienie obejmuje: xDSL (w tym również LLU i BSA), TVK, FWA, LAN-Ethernet, WiFi (WLAN), WiMax, stacjonarne CDMA, FTTH oraz łącza satelitarne.

Wykres 13. Penetracja na 100 mieszkańców dostępu do Internetu szerokopasmowego w krajach UE



Źródło: UKE na podstawie Digital Agenda Scoreboard 2012.

Uwaga: wskaźnik dla Finlandii poza skalą wykresu, penetracja mobilnego Internetu dla tego kraju wyniosła 70,9%.



Standardy

- HTML
- XML
- XHTML
- CSS, CSS2, CSS3
- HTML5

HTML

```
<HTML>
<HEAD>
  <TITLE>Strona</HTML>
</HEAD>
<BODY>
  <H3>Tytuł</H3>
  <P>Treść strony</P>
  <IMG SRC=obrazek.jpg>
</BODY>
</HTML>
```

XML

```
<?xml version="1.0" encoding="utf-8"?>
<towary>
  <towar>
    <id>1</id>
    <nazwa>butka</nazwa>
    <kod>59000000000010</kod>
    <cena>0.25</cena>
  </towar>
  <towar>
    <id>2</id>
    <nazwa>rogal</nazwa>
    <kod>59000000000020</kod>
    <cena>1.00</cena>
  </towar>
</towary>
```

xHTML

```
<?xml version="1.0" encoding="iso-8859-2"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pl">
<head>
  <meta http-equiv="content-type" content="text/html; charset=iso-8859-2" />
  <title>XHTML</title>
  <link href="style.css" type="text/css" rel="stylesheet" />
</head>
<body>
  <p>Akapit tekstu treści dokumentu</p>
  <hr />
  
</body>
</html>
```



CSS3 wprowadza nowe możliwości w zakresie prezentacji i transformacji obiektów HTML.

- **Obramowania**

- border-radius
- box-shadow
- border-image

- **Teksty**

- text-shadow
- word-wrap
- @font-face

- **Transformacje na płaszczyźnie**

- translate
- rotate
- scale
- skew



- Transformacje przestrzenne

- translate3D
- rotate3D
- scale3D
- perspective

- Przejścia

- transition-property
- transition-duration
- transition-timing-function
- transition-delay

- Animacje

- animation
- @keyframes

- Kolumny

- column-count
- column-width
- column-gap



CSS3 - Kompatybilność, czy jej brak?

- Prefiksy

- -o-
- -moz-
- -ms-
- -webkit-

```
transition-duration: 5s;  
-o-transition-duration: 5s;  
-moz-transition-duration: 5s;  
-ms-transition-duration: 5s;  
-webkit-transition-duration: 5s;
```

- Najnowsze wersje przeglądarek



Propozycja HTML5 jest odpowiedzią na brak akceptacji dla standardu XHTML 2.

HTML5 wykorzystuje CSS3, nowe znaczniki i atrybuty HTML oraz Javascript w celu wyeliminowania wtyczek i niezależności od platformy.

HTML5 wprowadza natywną obsługę dźwięku i video oraz umożliwia tworzenie animacji bez wtyczek typu Flash.





- Proste nagłówki

```
<!DOCTYPE html>  
<html lang="pl">  
<head>  
  <meta charset="utf-8" />  
  <link rel="stylesheet" href="styl.css" />  
  <title>HTML5</title>  
</head>
```

- Obsługa audio i video

```
<audio controls="controls">  
  <source src="muzyka.mp3" type="audio/mpeg" />  
</audio>
```

```
<video width="320" height="240" controls="controls">  
  <source src="film.mp4" type="video/mp4" />  
</video>
```





- Rozszerzona obsługa formularzy – nowe typy
 - color
 - date
 - datetime
 - datetime-local
 - email
 - month
 - number
 - range
 - search
 - tel
 - time
 - url
 - week





- Obsługa grafiki wektorowej SVG
- Możliwość tworzenia grafiki wprost na ekranie przeglądarki – canvas
- Obsługa geolokacji
- Obsługa Drag and Drop
- Lokalne przechowywanie danych aplikacji webowych
- Webcaching aplikacji i ich danych umożliwiający pracę bez połączenia internetowego

Przykłady.





Języki programowania internetowego

- PHP
- ASP, C#, (.Net)
- Java (JSP)
- Javascript (DOM)
- ~~Actionscript (Flash)~~
- Perl
- Inne (CGI)



XML

Extensible Markup Language



XML - EXtensible Markup Language

XML to uniwersalny standard dla potrzeb przechowywania i przesyłania informacji.

- Prosty
- Niezależny od platformy
- Wolny od typów
- Dobrze zdefiniowany
- Walidowalny
- Rodzic dla innych, nawet starszych języków znacznikowych
- Transformowalny



```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE osoby SYSTEM "kadry.dtd">
<?xml-stylesheet type="text/xsl" href="kadry.xsl" ?>
<osoby>
  <osoba>
    <id>1</id>
    <nazwisko>Nowak</nazwisko>
    <imie>Teresa</imie>
    <numer>202</numer>
  </osoba>
  <osoba>
    <id type="integer">2</id>
    <nazwisko>Kowalski</nazwisko>
    <imie>Jan</imie>
    <numer />
  </osoba>
</osoby>
```



DTD – Document Type Definition

```
<?xml version="1.0"?>  
<!DOCTYPE osoby [  
  <!ELEMENT osoba (id, nazwisko, imie+, numer)*>  
  <!ELEMENT id (#PCDATA)>  
  <!ELEMENT nazwisko (#PCDATA)>  
  <!ELEMENT imie (#PCDATA)>  
  <!ELEMENT numer (#PCDATA)>  

```



XSLT – EXtensible Stylesheet Language Transformation

```
<?xml version="1.0" encoding="utf-8" ?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html>
  <body>
    <xsl:for-each select="osoby/osoba">
      <xsl:sort select="nazwisko" />
      <p><xsl:value-of select="nazwisko" /></p>
    </xsl:for-each>
  </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```




Najprostszy export do XML

```
$db = new PDO('sqlite:baza.db');  
$res = $db->query('SELECT * FROM osoby');  
$xml = fopen('kadry.xml','w');  
fwrite($xml, '<?xml version="1.0" ?><osoby>');  
while($row = $res->fetch())  
{  
    fwrite($xml, '<osoba><imie>'.$row['imie']);  
    fwrite($xml, '</imie><nazwisko>'.$row['nazwisko']);  
    fwrite($xml, '</nazwisko></osoba>');  
}  
fwrite($xml, '</osoby>');  
fclose($xml);
```



Import danych XML w PHP

```
$osoby=new SimpleXMLElement('kadry.xml', 0, true);  
foreach($osoby->osoba as $o)  
{  
    print $o->nazwisko.' ' . $o->imie.'<br />'.PHP_EOL;  
}
```



Import danych XML w JavaScript

```
var xmlhttp;  
if(window.XMLHttpRequest)  
    xmlhttp = new XMLHttpRequest();  
else  
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");  
xmlhttp.open("GET", "kadry.xml", false);  
xmlhttp.send();  
var xml = xmlhttp.responseXML;
```



Import danych XML w JavaScript

```
txt = '';
o = xml.documentElement.getElementsByTagName('osoba');
for(i=0; i<o.length(); i++)
{
    txt=txt+o[i].getElementsByTagName('nazwisko')[0].childNodes[0].nodeValue;
    txt=txt+'    ';
    txt=txt+o[i].getElementsByTagName('imie')[0].childNodes[0].nodeValue;
    txt=txt+'<br />';
}
document.getElementById('tresc').innerHTML = txt;
```



JSON

Javascript Object Notation



JSON -JavaScript Object Notation

JSON to uniwersalny standard tekstowego kodowania danych dla potrzeb przesyłania informacji.

- Prosty
- Niezależny od platformy - stosuje składnię JavaScript, ale to zwykły plik tekstowy
- Czytelny i samokomentowalny
- Szeroko stosowany
- Niezwykle wygodny w połączeniu z językiem JavaScript, technologiami bazującymi na JS np. Ajax i bibliotekami tego języka jak np. jQuery



JSON

Przykład reprezentacji danych:

```
{"osoby":[  
  {"id":1, "nazwisko":"Nowak", "imie":"Teresa", "numer":202 },  
  {"id":2, "nazwisko":"Kowalski", "imie":"Jan", "numer":204 },  
  {"id":3, "nazwisko":"Smith", "imie":"John", "numer":504 }  
]}
```

Typy danych w JSON:

- Liczy - całkowite i zmiennopozycyjne
- Teksty - ujęte w podwójne cudzysłowy
- Wartości logiczne - **true** albo **false**
- Tablice - ujęte w nawiasy kwadratowe
- Obiekty - ujęte w nawiasy klamrowe



JSON

Przetwarzanie w Javascript:

```
var txt = '{"osoby":[
  {"id":1, "nazwisko":"Nowak", "imie":"Teresa", "numer":"202" },
  {"id":2, "nazwisko":"Kowalski", "imie":"Jan", "numer":"204" },
  {"id":3, "nazwisko":"Smith", "imie":"John", "numer":"504" }
]}';
```

```
var obj = JSON.parse(txt);
```

```
document.getElementById("moje_id").innerHTML =
  obj.osoby[1].nazwisko + " " + obj.osoby[1].imie;
```

Uwaga:

Zamiast JSON.parse() można, ale nie należy, używać mniej bezpiecznej funkcji eval():

```
var obj = eval("(" + txt + ")");
```



AJAX

Asynchronous Javascript and XML



Ajax - (*Asynchronous JavaScript and XML*)

Technologia (technika programistyczna) umożliwiająca tworzenie interaktywnych aplikacji webowych bez przeładowywania całej strony.

Działanie techniki AJAX:

- **1** – normalne załadowanie i wyświetlenie strony
- **2** – reakcja na zdarzenie np. naciśnięcie przycisku
- **3** – utworzenie połączenia z serwerem i asynchroniczna wymiana danych
- **4** – odebranie i przetworzenie danych w JavaScript
- **5** – modyfikacja zawartości strony



AJAX elementy składowe:

- **XMLHttpRequest** - obiekt umożliwiający asynchroniczne przesyłanie danych
- **Javascript** – albo dowolny inny język skryptowy działający w przeglądarce i obsługujący obiektowy model dokumentu DOM
- **XML** – albo HTML, albo fragmenty kodu Javascript (JSON) albo inne specyficzne dane odpowiedzialne za zmianę zawartości strony
- **CSS** – odpowiada za zmianę wyglądu strony



Przykład - zmiana zawartości fragmentu strony:

```
<html>
<head>
...
  <script type="text/javascript" src="myFun.js">
  </script>
</head>
<body>
...
<div id="myDiv"></div>
<button type="button" onclick="myAjax()">Zmień</button>
...
```

Przykład:

1. utworzenie obiektu XMLHttpRequest:

```
var xmlhttp;  
if (window.XMLHttpRequest) //IE7+, FF, O, Chrome, Safari  
    xmlhttp = new XMLHttpRequest();  
else // IE6, IE5  
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
```

2. wysłanie żądania:

```
xmlhttp.open("GET", "myS.php?p=1&id=Math.random()", true);  
xmlhttp.send();
```



Przykład:

3. obsługa odpowiedzi HTML:

```
xmlhttp.onreadystatechange = function()  
{  
  if (xmlhttp.readyState==4 && xmlhttp.status==200)  
  {  
    elem=document.getElementById("myDiv");  
    elem.innerHTML=xmlhttp.responseText;  
  }  
}
```

Uwaga: Obsługę należy zdefiniować przed wysłaniem żądania.

Przykład:

3. obsługa odpowiedzi XML:

```
xmlhttp.onreadystatechange = function()  
{  
  if (xmlhttp.readyState==4 && xmlhttp.status==200)  
  {  
    xmlDoc=xmlhttp.responseXML;  
    txt="";  
    x = xmlDoc.getElementsByTagName("Nazwa");  
    for (i=0;i<x.length;i++)  
    {  
      txt=txt+x[i].childNodes[0].nodeValue+"<br />";  
    }  
    document.getElementById("myDiv").innerHTML=txt;  
  }  
}
```

Uwaga: Obsługę należy zdefiniować przed wysłaniem żądania.



Najważniejsze właściwości i metody klasy XMLHttpRequest

Właściwość	Wartość	Opis
onreadystatechange	Nazwa funkcji	Funkcja obsługująca połączenie
status	200	OK
	404	Nie znaleziono
readyState	0	Nie zainicjowano
	1	Połączenie nawiązane
	2	Żądanie przesłane
	3	Żądanie przetwarzane
	4	Otrzymana odpowiedź



Najważniejsze właściwości i metody klasy XMLHttpRequest

Właściwość	Argumenty	Opis
open()	metoda, url, async	Otwarcie połączenia
send()	[danePost]	OK
responseText		Text odpowiedzi
responseXML		XML odpowiedzi



JQuery



jQuery

jQuery szeroko stosowana uniwersalna biblioteka ułatwiająca pisanie aplikacji webowych w języku JavaScript.

Motto: "write less, do more".

- Prosta w stosowaniu
- Wykorzystująca selektory CSS
- Działająca we wszystkich przeglądarkach
- Wykorzystywana w wielu aplikacjach
- Ułatwiająca kodowanie JavaScript i skracająca kod
- Obsługuje: HTML, CSS, zdarzenia, efekty, animacje, AJAX
- Napisano do niej wiele użytecznych wtyczek



jQuery

Dołączenie biblioteki:

```
<script src="jquery-3.2.1.js"></script>
```

Uwaga:

Wygodniejszym od utrzymywania lokalnej kopii biblioteki jest jej ładowanie z sieci CDN firm Google albo Microsoft. Powszechność stosowania biblioteki powoduje, że najczęściej jest już ona w pamięci podręcznej przeglądarki.

```
<script  
  src="http://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js">  
</script>
```

albo

```
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.2.1.min.js">  
</script>
```

albo

```
<script src="http://code.jquery.com/jquery-3.2.1.min.js"></script>
```

Aktualną numerację wersji można sprawdzić na stronie:

<https://code.jquery.com/jquery/>



jQuery - Składnia

Podstawowa składnia:

`$(selektor).metoda();`

Wywołanie metody.

`$(selektor).zdarzenie(function(){...});`

Zdefiniowane metody obsługi zdarzenia.

`$(selektor).właściwość();`

Odczytanie wartości.

`$(selektor).właściwość(wartość);`

Przypisanie nowej wartości.

Przykład:

`$("p").click(function() { $(this).css("color", "red"); });`

Przypisanie obsługi zdarzenia kliknięcia myszką do wszystkich akapitów tekstu z akcją polegającą na zmianie koloru znaków klikniętego akapitu na czerwony.



jQuery - Selektory

Selektory biblioteki jQuery są zgodne ze składnią selektorów CCS.

Przykłady:

`$("table")...`

Wszystkie tabele.

`$("td.lewy")...`

Wszystkie komórki tabeli klasy lewy.

`$("#mojeid")...`

Element dokumentu identyfikowany id="mojeid".

`$("ol li:first-child")...`

Każdy pierwszy element li, każdej listy numerowanej ol.

`$("input[type='password']")...`

Wszystkie pola formularzy typu password.

`$(this)...`

Element, który właśnie przetwarzamy.



jQuery - Zdarzenia

Zdarzenia pozwalają na wywoływanie i obsługę akcji w aplikacji.

Podstawowe zdarzenia:

- Myszy:
click, dblclick, mouseenter, mouseleave, hover, mousedown, mouseup, mousemove
- Klawiatury:
keypress, keydown, keyup
- Formularzy:
submit, focus, blur, change
- Dokumentu:
ready, resize, scroll

W szczególności metoda obsługi zdarzenia `ready` dla dokumentu jest wykorzystywana do wykonania wszystkich akcji przypisania właściwej obsługi zdarzeń elementów.

```
$(document).ready(function()  
{  
  $("#id1").click(function()  
  {  
    //...implementacja  
  });  
});
```



jQuery - Właściwości

Właściwości pozwalają na odczyt i zmianę zawartości atrybutów elementów dokumentu.

Podstawowe właściwości:

- **html()**
wartość elementu jako HTML
- **text()**
wartość elementu jako tekst
- **val()**
wartość elementu formularza
- **attr()**
wartość atrybutu elementu

Przykład:

```
$("#id1").val( $("#id2").val() );
```

Szczególnym rodzajem atrybutu elementów są style CSS.

```
$("#id1").removeClass("prawy");  
$("#id1").addClass("lewy");  
$("p").css("color", "red");  
$(".lewy").css( {"color": "red", "font-size": "small"} );
```



jQuery - AJAX

Biblioteka jQuery pozwala na bezpośrednie korzystanie z technologii AJAX bez konieczności obsługi poszczególnych faz żądania.

- **`$(...).load(url, param)`**
zmiana właściwości elementu przez załadowanie jej z serwera z wykorzystaniem asynchronicznej komunikacji AJAX metodą POST
- **`$.get(url, param, function(data,status)), $.getJSON()`**
załadowanie danych z serwera z wykorzystaniem asynchronicznej komunikacji AJAX metodą GET
- **`$.post(url, param, function(data,status))`**
załadowanie danych z serwera z wykorzystaniem asynchronicznej komunikacji AJAX metodą POST

Przykłady:

```
$("#zegarek").load("czas.php");  
$.post("skrypt.php",  
    { "jeden": 1, "napis": "napis", "t[]": [ "a", "b" ] },  
    function(data, status) { $("#wynik").html(data); } );
```



jQuery - Efekty

Biblioteka jQuery pozwala na realizację wizualnych efektów na elementach dokumentu.

- **hide(), show()**
ukrywanie i pokazywanie elementów
- **fadeOut(), fadeIn(), fadeTo()**
płynne zanikanie, pojawianie się albo ustawienie półprzeźroczystości
- **slideDown(), slideUp()**
zwijanie i rozwijanie elementów
- **animate()**
animacja przez transformacje położenia, wielkości i innych właściwości elementów

Przykłady:

```
$("#id1").html("Czekaj...").show().fadeTo("fast", 0.1).fadeTo("slow", 1);  
$("#id2").hide();  
$("#id3").slideUp("slow");
```

Efekty można łączyć w łańcuchy. Realizacja następuje kolejno.

W animacjach nazwy atrybutów CSS muszą być pisane w notacji "wielbłądowej" JS tj. `marginTop`, a nie `margin-top` jak w CSS.



Biblioteki interfejsu użytkownika

Zaawansowane biblioteki złożone z gotowych stylów CSS, elementów graficznych oraz skryptów JavaScript (najczęściej z wykorzystaniem jQuery) upraszczających i przyspieszających wytwarzanie funkcjonalnych i estetycznych interfejsów graficznych.

Bootstrap

- szybkie wytwarzanie aplikacji internetowych, których interfejs użytkownika dopasowuje się do rozdzielczości urządzenia.

jQuery UI

- interfejs użytkownika zorientowany na interakcję z wygodną obsługą ekranów dotykowych

Angular Material

- implementacja Material Design - języka komunikacji wizualnej łączącego zasady dobrego projektowania z innowacyjnością oraz możliwościami technologicznymi zapewniającego jednolitą obsługę na urządzeniach różnych platform i rozmiarów.



Bootstrap

Bootstrap

Biblioteka Bootstrap jest zorientowana na szybkie wytwarzanie aplikacji internetowych, których interfejs użytkownika dopasowuje się do rozdzielczości urządzenia. Oferuje obsługę ekranów od bardzo małej (telefony-360px) do ultra wysokiej (desktopy-4K) rozdzielczości.

Bootstrap składa się z dwóch części:

- Zestawu stylów CSS3
- Skryptu Javascript napisanego z wykorzystaniem jQuery

Trzecią częścią musi zatem być biblioteka jQuery.

Wykorzystanie biblioteki Bootstrap sprowadza się do przypisywania elementom HTML właściwych klas CSS. Czasami wymagane jest użycie elementów w określonej kolejności.



Bootstrap

Dołączanie biblioteki:

```
<link rel="stylesheet"  
  href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"  
>
```

```
<script src="//code.jquery.com/jquery-3.2.1.min.js"></script>
```

```
<script  
  src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js">  
</script>
```

Aktualną numerację wersji można sprawdzić na stronach:

<https://getbootstrap.com/>

<https://code.jquery.com/jquery/>



Wymiarowanie ekranu oparte na predefiniowanych rozmiarach:

- **-xs-** - bardzo mały 34em (544px) np. telefon
- **-sm-** - mały 48em (768px) np. tablet
- **-md-** - średni 62em (992px) np. laptop
- **-lg-** - duży 75em (1200px) np. desktop
- **-xl-** - bardzo duży >75em np. FullHD (od Bootstrap 4.0)

oraz układzie szerokości kolumn sumującym się do 12:

```
<div class="container">  
  <div class="row">  
    <div class="col-xs-12 col-sm-4 col-md-6"> ... </div>  
    <div class="col-xs-12 col-sm-8 col-md-6"> ... </div>  
  </div>  
  ...  
</div>
```



Kolorowanie z wykorzystaniem odcieni kilku kolorów:

- default - biały
- muted - szary
- primary - niebieski
- success - zielony
- info - błękitny
- warning - pomarańczowy
- danger - czerwony

```
<div class="bg-info"> ... </div>  
<p class="text-primary"> ... </p>  
<tr class="success"> ... </tr>
```



Klasy dla układu tekstu:

`.text-left .text-center .text-right .text-justify`
`.text-lowercase .text-uppercase .text-capitalize`

Style dla wszystkich standardowych znaczników m.in.:

`h1 h2 h3 h4 h5 h6 p mark del ins s u em small strong ul ol li`

Style dla znaczników prezentacji kodów:

`code pre kbd var`

Klasy dla określenie trybu widoczności elementów:

`.show .hidden .invisible`
`.hidden-xs .hidden-sm .hidden-md .hidden-lg`
`.visible-*-block .visible-*-inline`
`.visible-print-block .visible-print-inline`
`.sr-only` (tylko dla czytników ekranu)



Klasy do szybkiego formatowania tabel:

- **.table** - tabela standardowa
- **.table-responsive** - tabela przewijana w poziomie na małych ekranach
- **.table-striped** - "paski" przyciemniony co drugi wiersz
- **.table-bordered** - obramowania komórek
- **.table-hover** - podświetlanie wiersza z kursorem
- **.table-condensed** - gęstsze ułożenie wierszy

```
<div class="table-responsive">  
  <table class="table table-striped table-bordered">  
    <tbody>  
      <tr><td> ... </td><td> ... </td></tr>  
    </tbody>  
  </table>  
</div>
```



Przyciski:

- `.btn`
- `.btn-default .btn-primary .btn-success .btn-info .btn-warning .btn-danger`
- `.btn-group .btn-group-lg .btn-group-sm .btn-group-xs`

```
<div class="btn-group btn-group-lg">  
  <button class="btn btn-primary"> ... </button>  
  <button class="btn btn-success"> ... </button>  
  <a href="..." class="btn btn-info" /> ... </a>  
</div>
```



Formularze:

- **.form** - formularz standardowy
- **.form-inline** - formularz z elementami w linii
- **.form-horizontal** - formularz z etykietami z boku

```
<form class="form-horizontal">
  <div class="form-group">
    <label for="imie" class="col-sm-2 control-label">
      Imię
    </label>
    <div class="col-sm-10">
      <input type="text" id="imie" class="form-control" />
    </div>
  </div>
</form>
```

Uwaga na pola typu radio i checkbox – powinny być częścią znacznika label.



Alerty i panele:

- **.well** - wyróżniony fragment tekstu (jasnoszare tło)
- **.alert** - wyróżniony kolorowy fragment tekstu
 - **.alert-success .alert-info .alert-warning .alert-danger**
- **.panel** - wydzielone obramowane okno z nagłówkiem i stopką

```
<div class="panel panel-primary">  
  <div class="panel-heading">Tytuł</div>  
  <div class="panel-body">  
    ...  
  </div>  
  <div class="panel-footer">Podpis</div>  
</div>
```

Uwaga! W bootstrap 4.0 well, alert i panel zostaną zastąpione przez card.



Menu i nawigacja:

- `.nav.nav-pills` - nawigacja w formie przycisków
- `.nav.nav-pills .nav-stacked` - nawigacja pionowa
- `.nav.nav-tabs` - nawigacja w formie zakładek
- `.nav.nav-tabs .nav-justified` - nawigacja na całą szerokość
- `.navbar .navbar-default` - pasek nawigacyjny, który może zawierać elementy złożone
- `.dropdown-menu` - rozwijane menu np. do przycisku

```
<ul class="nav nav-tabs nav-justified">  
  <li><a href="#">Pozycja pierwsza</a></li>  
  <li><a href="#">Pozycja druga</a></li>  
  <li><a href="#">Pozycja trzecia</a></li>  
</ul>
```


Podpowiedzi do przycisków:

```
<button class="btn"  
  data-toggle="tooltip" data-placement="top"  
  title="Podpowieź do działania przycisku">  
Przycisk z podpowiedzią typu tooltip  
</button>
```

```
<button class="btn"  
  data-toggle="popover" data-placement="left"  
  title="Podpowieź do działania przycisku"  
  data-content="Tekst podpowiedzi dla przycisku">  
Przycisk z podpowiedzią typu popover  
</button>
```

Elementy tooltip i popover wymagają uruchomienia w jQuery:

```
$(document).ready( function() {  
  $('[data-toggle="tooltip"]').tooltip();  
  $('[data-toggle="popover"]').popover();  
});
```



Przyciski odsłaniające treść:

```
<button class="btn"  
  data-toggle="collapse" data-target="#opis"  
Pokaż opis  
</button>
```

```
<div class="collapse" id="opis">  
Treść opisu, która ukaże się dopiero po naciśnięciu przycisku  
</div>
```

Zakładki przełączające widoczną treść:

```
<ul class="nav nav-tabs">
  <li><a href="#tab1" data-toggle="tab">Błękitne</a></li>
  <li><a href="#tab2" data-toggle="tab">Zielone</a></li>
  <li><a href="#tab3" data-toggle="tab">Żółte</a></li>
</ul>

<div class="tab-content">
  <div class="tab-pane fade in active" id="tab1"> ... </div>
  <div class="tab-pane fade" id="tab2"> ... </div>
  <div class="tab-pane fade" id="tab3"> ... </div>
</div>
```



jQueryUI – <http://jqueryui.com>

Biblioteka jQueryUI umożliwia wykorzystanie predefiniowanych komponentów wykorzystujących Javascript (jQuery) i style CSS.

Dołączanie biblioteki:

```
<link rel="stylesheet"
href="//code.jquery.com/ui/1.12.1/themes/smoothness/jquery-
ui.css">
<script src="//code.jquery.com/jquery-3.2.1.js"></script>
<script src="//code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
```

Aktualną numerację wersji można sprawdzić na stronie:

<https://code.jquery.com/ui/>



jQueryUI - Komponenty interaktywne:

- Draggable – przesuwanie po oknie aplikacji
- Droppable – obsługa Drag and drop
- Resizable – zmiana wielkości
- Sortable – zmiiana kolejności

Widżety:

- Accordion – sekcje zmieniające treść
- Autocomplete – podpowiadanie zawartości
- Datepicker – okno wyboru daty
- Dialog – okno dialogowe
- Menu – rozwijane wielopoziomowe menu
- Slider – suwak
- Tabs – zakładki zmieniające treść
- Tooltip – podpowiedzi

Efekty przejść oraz pozycjonowanie względne elementów itp.



AngularJS

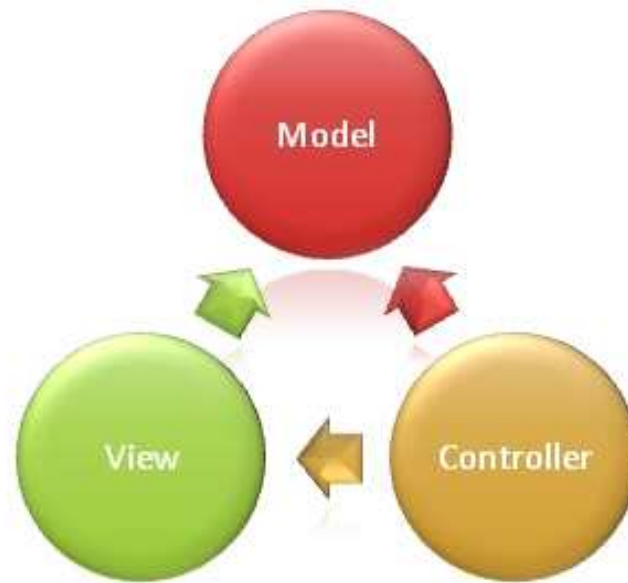
MVC - Model View Controller

Architektura oprogramowania oparta o trzy elementy

Model - stanowi klasyczny back-end aplikacji. Reprezentuje wszystkie struktury danych, bazy danych i całą logikę aplikacji.

Widok - jest interfejsem użytkownika. Opisuje metody prezentowania i edycji danych przez użytkownika. Widok wykorzystuje dane pobrane z Modelu.

Kontroler - wykonuje reakcje na dane wprowadzone przez użytkownika. Obsługuje wprowadzanie i edycję danych wpływając na model i widok.





AngularJS

AngularJS to biblioteka (framework) języka JavaScript rozszerzająca statyczne znaczniki HTML o możliwość modelowania treści i zachowań elementów strony. Szczególnie wygodna do implementacji aplikacji jednoekranowych.

AngularJS bazuje na trzech podstawowych elementach

ng-app

- Definicja aplikacji AngularJS i określenie jej zasięgu do zasięgu wybranego znacznika HTML

ng-model

- Model obsługujący przetwarzanie danych

ng-bind

- Odzwierciedlenie danych modelu na widok aplikacji



AngularJS

Dołączanie biblioteki

```
<script  
src="//ajax.googleapis.com/ajax/libs/angularjs/1.6.6/angular.min.js  
>  
</script>
```

Aktualną numerację wersji można sprawdzić na stronie:
<https://developers.google.com/speed/libraries/#angularjs>



AngularJS

ng-model

- Przywiązanie wartości znacznika HTML do danych modelu aplikacji

ng-bind

- Przywiązanie wartości kontrolki HTML do danych modelu aplikacji

ng-repeat

- Iterator po wszystkich danych do wypełniania wzorca widoku

ng-click ng-keyup ng-change ...

- Obsługa zdarzeń

ng-hide ng-show ng-disabled

- Zmiana stanu elementu HTML



AngularJS

Przykład wzorca widoku w aplikacji AngularJS

```
<div ng-app="ajsApp" ng-controller="ajsAppCtrl">
<table>
  <tbody>
    <tr ng-repeat="x in imiona">
      <td>{{ x.pozycja }}</td>
      <td>{{ x.imie }}</td>
    </tr>
  </tbody>
</table>
</div>
```



AngularJS

Przykład definicji kontrolera aplikacji AngularJS

- pobranie danych techniką AJAX z webservice'u REST API zwracającego wynik w formacie JSON

```
var app = angular.module("ajsApp", []);

app.controller("ajsAppCtrl", function($scope, $http) {

    $scope.liczba = 5;
    $http.get("model.php").success( function(response) {
        $scope.imiona = response;
    });
}

});
```



AngularJS

Przykład definicji modelu aplikacji AngularJS

- Webservice REST API zwracający wynik w formacie JSON

```
<?php
```

```
header("Access-Control-Allow-Origin: *");
```

```
header("Content-Type: application/json; charset=UTF-8");
```

```
//połączenie z bazą danych
```

```
$db = new PDO('sqlite:baza.db');
```

```
$sql="SELECT * FROM imiona ORDER BY pozycja LIMIT 20";
```

```
//przesłanie wyniku
```

```
print json_encode( $db->query($sql)->fetchAll(PDO::FETCH_ASSOC) );
```

```
?>
```



Web Services

Usługi webowe realizują koncepcję budowy systemów rozproszonych ukierunkowanych na dostarczanie usług.

W architekturze Web Services są usługami dostarczającymi określoną funkcjonalność za pomocą ogólnie dostępnych i powszechnie wykorzystywanych standardów. Najczęściej wykorzystywane są HTTP albo HTTPS i XML.

Przewodnią ideą jest niezależność sposobu implementacji usług i sposobu implementacji aplikacji klientów usług.

SOA Service Oriented Architecture

Do realizacji SOA wykorzystuje się różne popularne standardy:

- REST
- CORBA
- DCOM
- WDSL
- RDF
- itp..



REST Representational State Transfer

REST API to wzorzec realizacji usług sieciowych z wykorzystaniem adresów URL i metod protokołu HTTP.

- **GET** - odczytaj wartości
- **PUT** - zapisz (zmodyfikuj) wartości
- **POST** - zapisz (dodaj nowe) wartości
- **DELETE** - usuń wartości

Przykłady:

Odczytaj listę wartości

GET <http://serwer.domena/katalog/usluga>

Odczytaj wartość numer 123

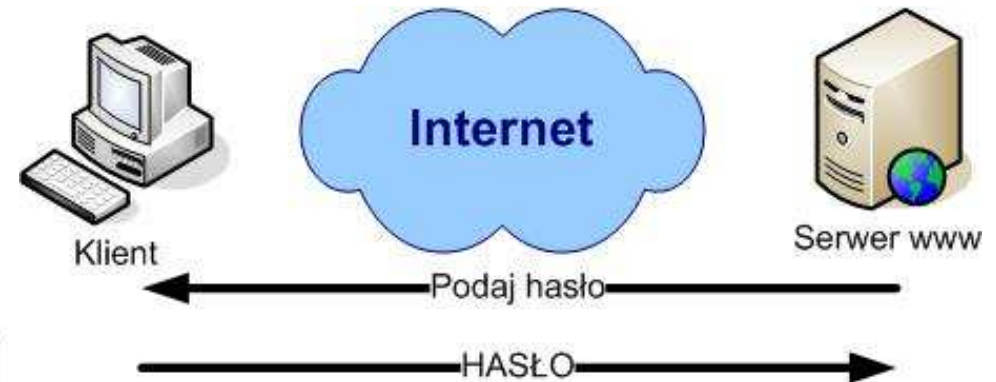
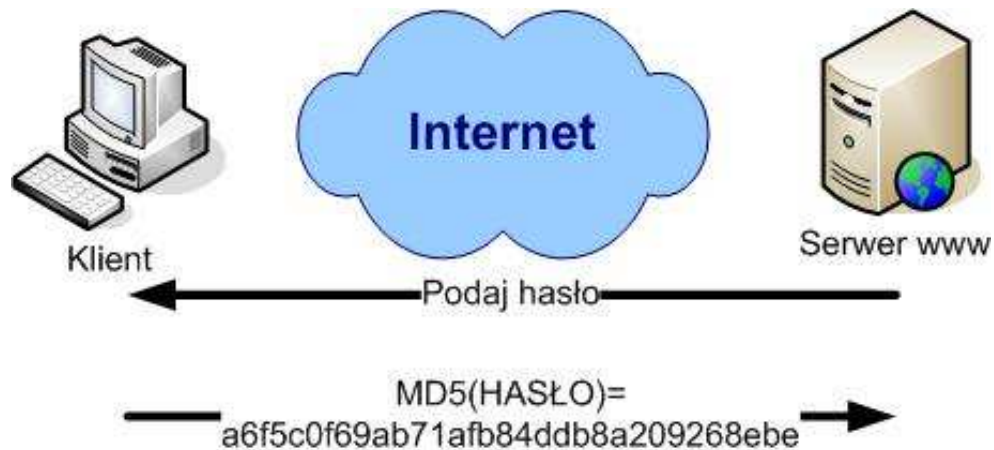
GET <http://serwer.domena/katalog/usluga?id=123>

Usuń wartość numer 123

DELETE <http://serwer.domena/katalog/usluga?id=123>

Uwierzytelnianie czyli identyfikacja, weryfikacja i autoryzacja.

Weryfikacja klasyczna,
szyfrowana
i z kluczem zmiennym





- Formularze a bezpieczeństwo
- HTML, CSS, Javascript Code Injection

- Przykładowe dane

```
<h1><big><big>Jasio<br /><br /><br /><br /><br />  
<script>window.href=http://mojastrona.com</script>
```

- SQL Injection

- Przykładowe ataki:

```
' OR '1'='1
```

```
' OR '1'='1' -- ,
```

```
0 OR 1=1
```

```
admin') --
```

```
'); DROP TABLE users
```

```
\''; DROP TABLE users
```

```
'); DROP DATABASE sklep
```

```
x' AND BENCHMARK(2000000000,MD5(NOW()))=0 OR '1'='1
```



– Wersja 1 – niebezpieczna

```
$db=new PDO('sqlite:baza.db');  
$sql='SELECT * FROM tabela WHERE login='.$_POST['user'];  
$res=$db->query($sql);
```

– Wersja 2 – bezpieczniejsza

```
$db=new PDO('sqlite:baza.db');  
$param=substr(strip_tags($_POST['user']),0,32);  
$param=$db->quote($param);  
$sql='SELECT * FROM tabela WHERE login='.$param;  
$res=$db->query($sql);
```



— Wersja 3 – najbezpieczniejsza

```
$db=new PDO('sqlite:baza.db');  
$param=substr(strip_tags($_POST['user']), 0, 32);  
$sql='SELECT * FROM tabela WHERE login=:arg';  
$stmt=$db->prepare($sql);  
$stmt->bindParam(':arg', $param, PDO::PARAM_STR, 32);  
$stmt->execute($sql);
```

Formularz uwierzytelniania użytkownika:

<head>

...

<script type="text/javascript" src="md5.js"></script>

<script type="text/javascript" src="funkcje.js"></script>

</head>

<body>

...

<form method="post" action="" onsubmit="zabezpiecz()">

<input type="hidden" name="key" id="key"

value="<?php print \$_SESSION['challenge_key']; ?>" />

Login: <input type="text" name="user" id="user" />

Hasło: <input type="password" name="pass" id="pass" />

<input type="submit" />

</form>

Wartość pola key powinna zmieniać się przy każdym przeładowaniu formularza.

Np. `$_SESSION['challenge_key'] = sha1(uniqid());`



Funkcja skrótu z kluczem zmiennym:

```
function zabezpiecz()  
{  
  var key = document.getElementById('key');  
  var pass = document.getElementById('pass');  
  pass.value = hex_md5(hex_md5(pass.value) + key.value);  
}
```

albo w jQuery:

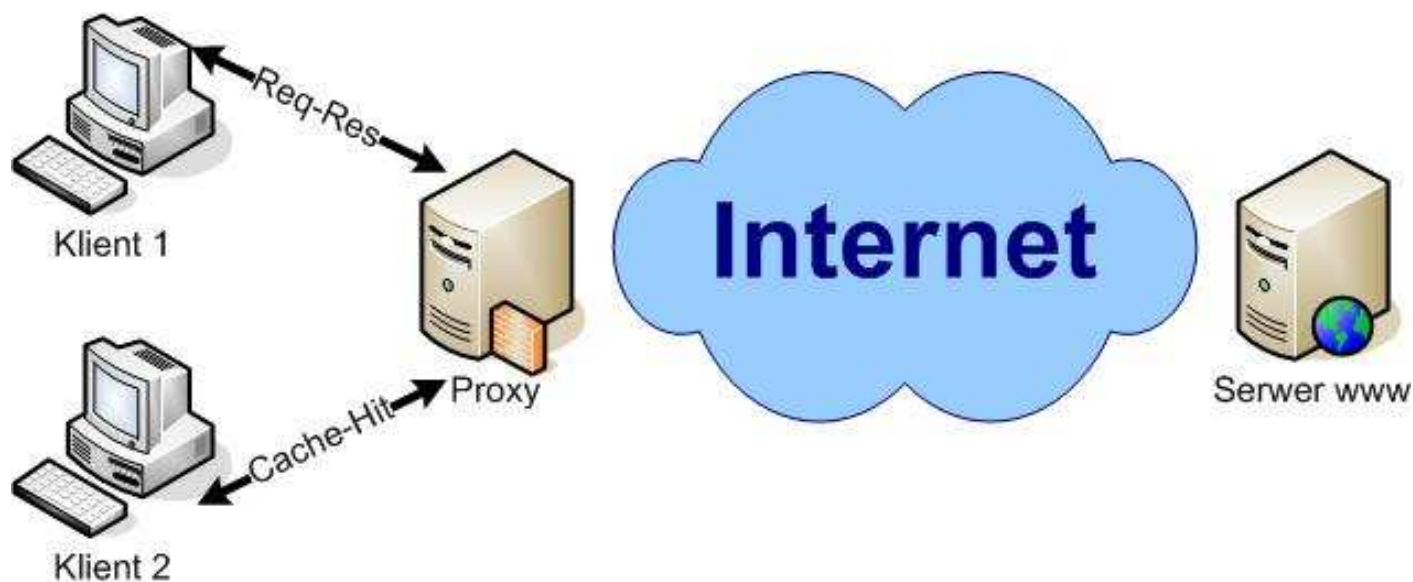
```
$("#form").submit( function() {  
  $("#pass").val(hex_md5(hex_md5($("#pass").val())+$("#key").val()));  
});
```

Implementację funkcji skrótów MD5, SHA1 itp. w JavaScript można znaleźć np. na stronie <http://pajhome.org.uk/crypt/md5/>.

W miejsce MD5 można użyć SHA-1 albo innej funkcji skrótu, a zamiast prostego sklejana łańcuchów znaków można zastosować algorytm HMAC.

WEB CACHING

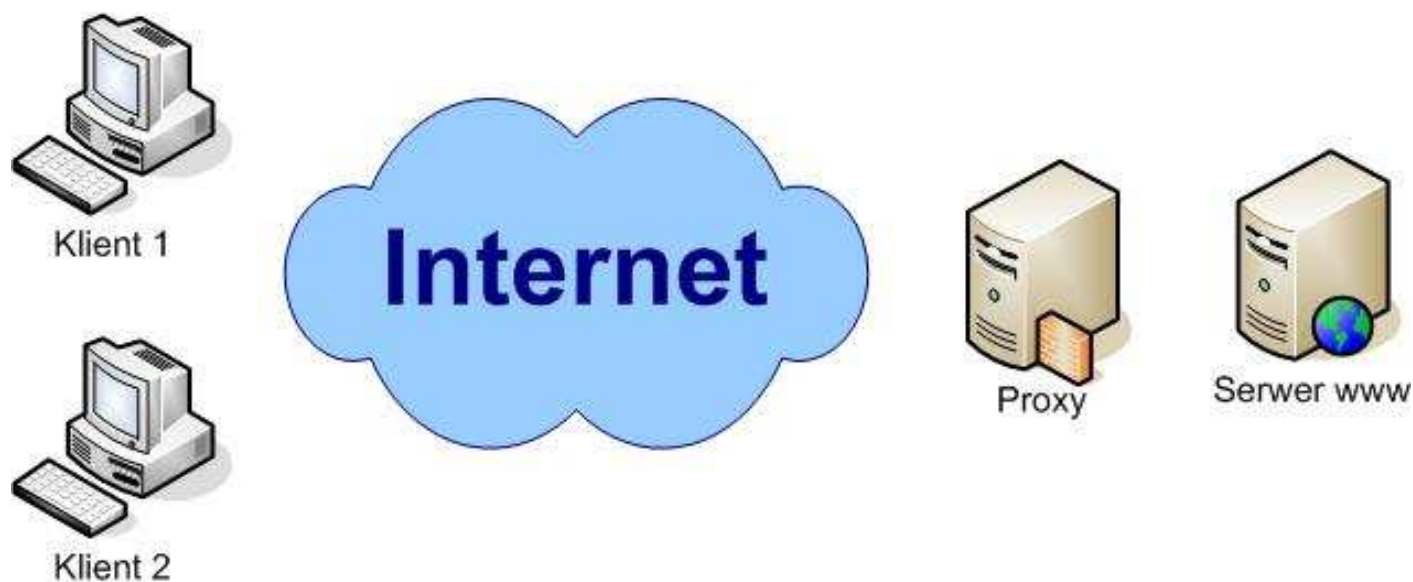
Tradycyjne Proxy - odciążanie wspólnego łącza





WEB CACHING

Odwrotne Proxy - odciążanie serwera webowego





WEB CACHING

HTTP (ang. Hypertext Transfer Protocol) RFC2616

Nagłówki HTTP dotyczące przechowywania dokumentów:

Last-modified: Thu, 01 Dec 1994 11:00:00 GMT

Expires: Thu, 01 Dec 1994 16:00:00 GMT

Cache-control: public

Cache-control: no-cache

Cache-control: no-store

Cache-control: must-revalidate

Cache-control: max-age=0

Uwaga: max-age wyklucza się z no-cache.



WEB CACHING

HTTP (ang. Hypertext Transfer Protocol)
RFC2616

Nagłówki HTML dotyczące przechowywania
dokumentów:

```
<meta content-type="Expires"  
content="Thu, 01 Dec 1994 16:00:00 GMT" />
```

Uwaga: Serwery Proxy nie czytają treści HTML i
znacznik meta z atrybutem cache-control nie
ma sensu, a powyższy przykład dotyczy tylko
pamięci cache w przeglądarce.



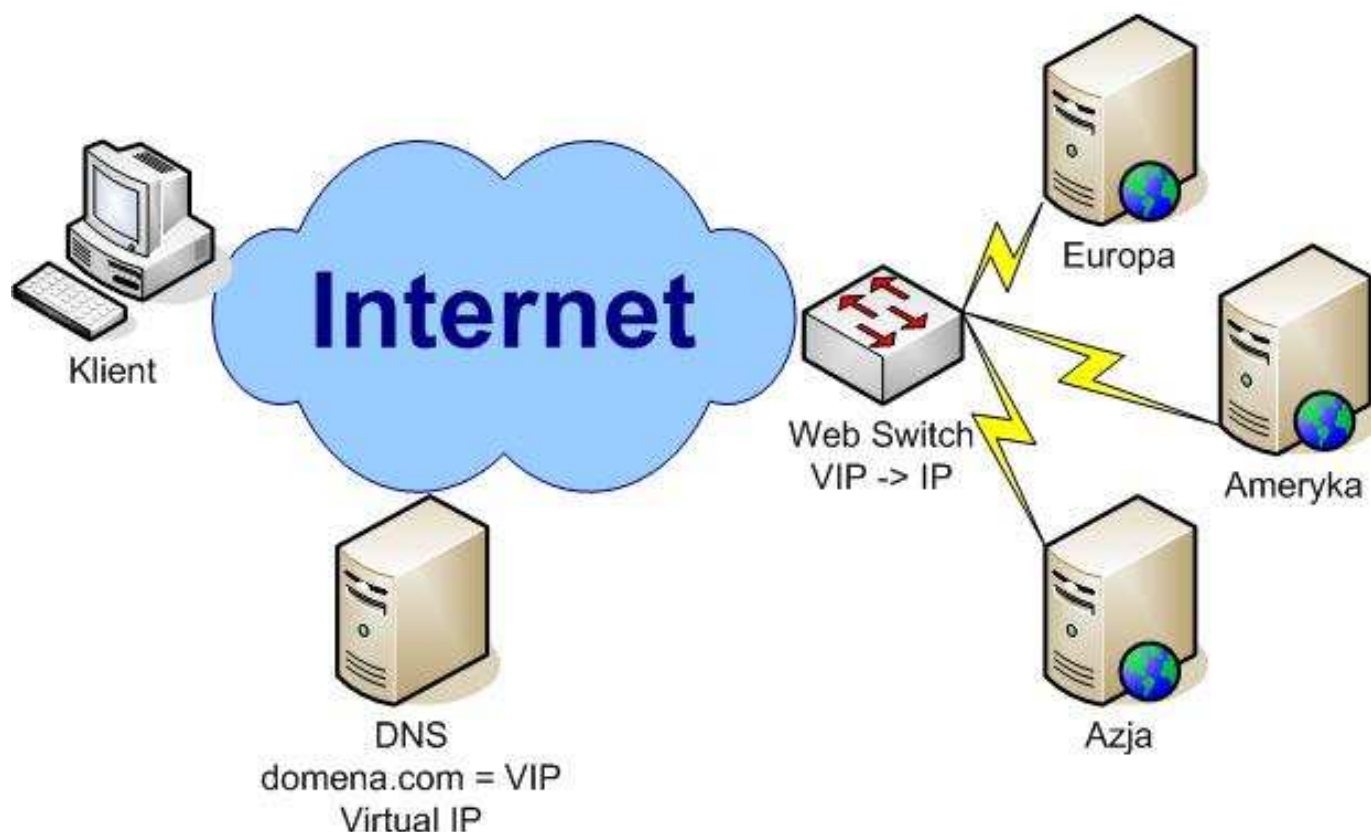
Prosty Web Caching w aplikacji:

```
if( !file_exists("cache") || (time()-filemtime("cache"))>60)
{
    //zapisz zawartość pliku cache

}
readfile("cache");
```

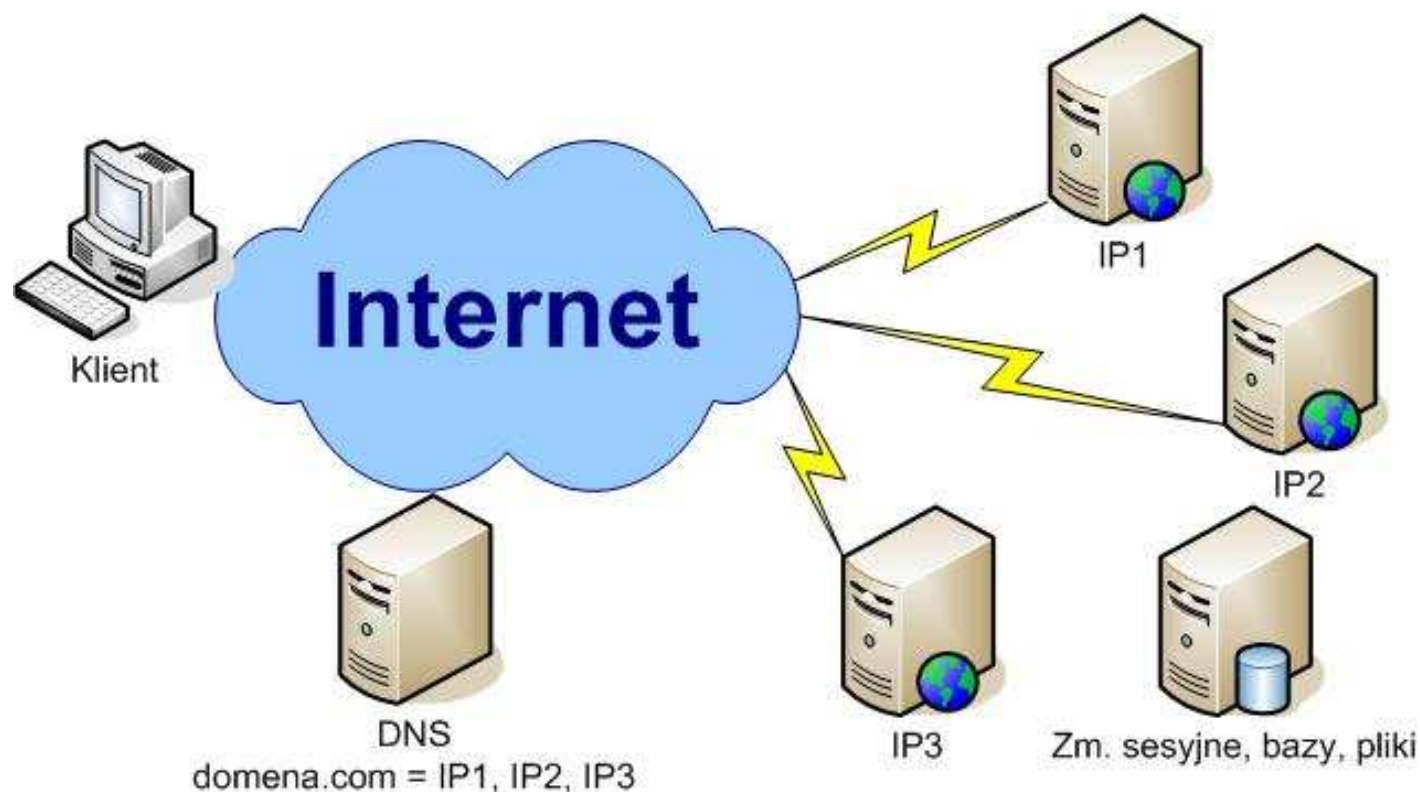
WEB SWITCHING

Rozpraszanie ruchu dla domeny globalnej



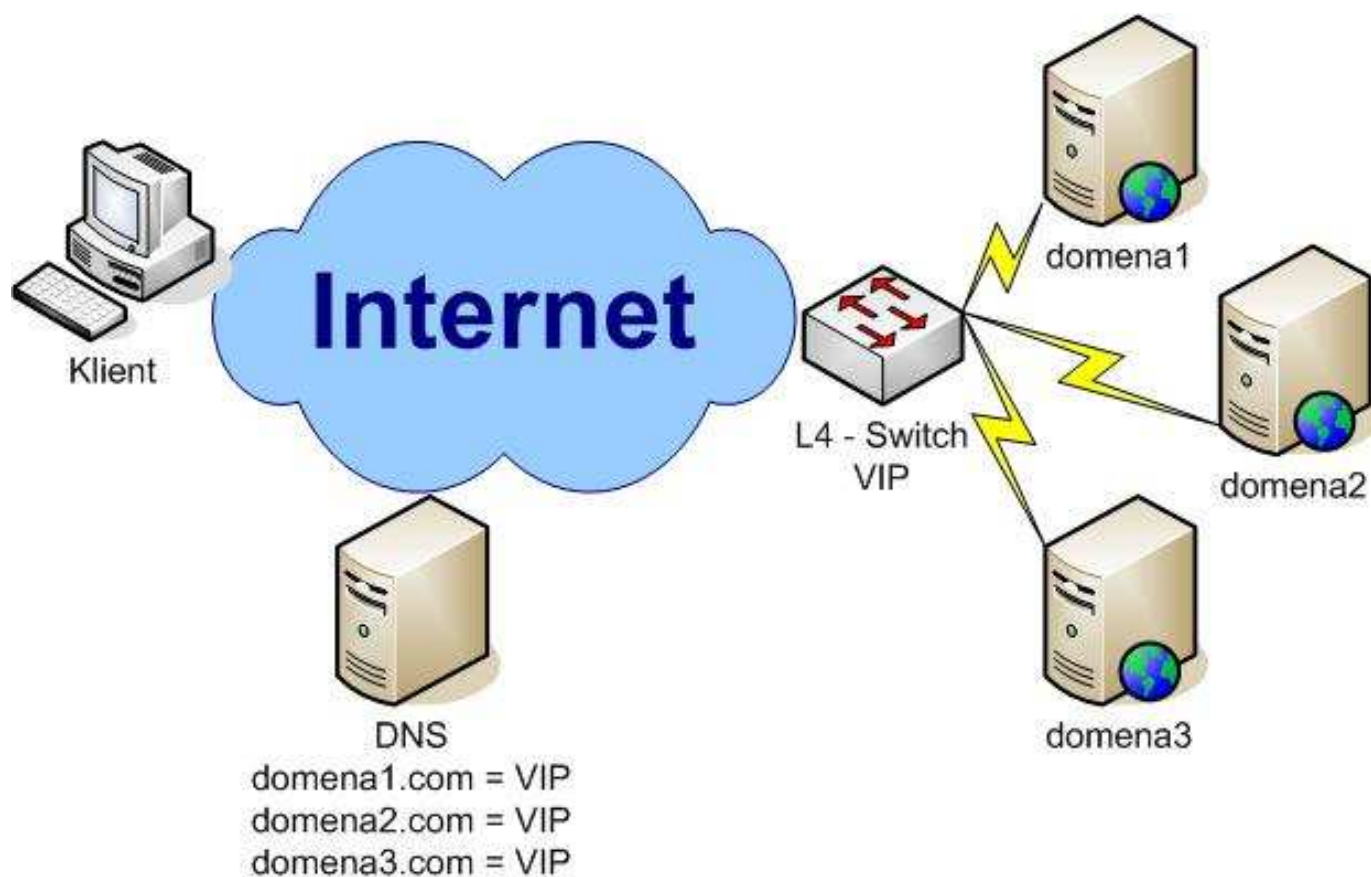
WEB SWITCHING

Rozpraszanie ruchu dla farmy serwerów



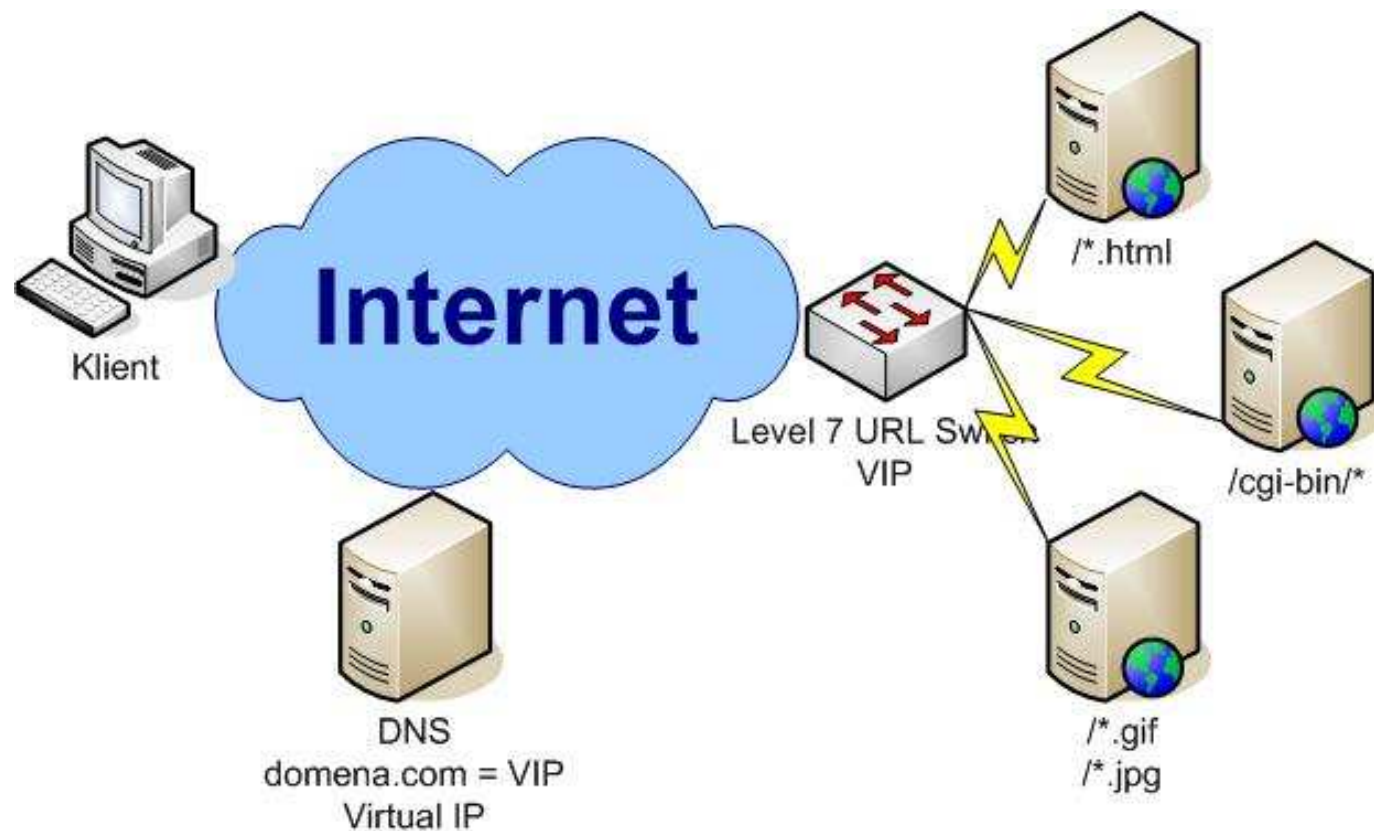
WEB SWITCHING

Serwery wirtualne



WEB SWITCHING

Przełączanie treści



Proste, losowe rozpraszanie ruchu pomiędzy kilka serwerów:

```
$serwery=array('gdansk.pl', 'sopot.pl', 'gdynia.pl');  
$nr=mt_rand(0,count($serwery)-1);  
header('Location: http://'.$serwery[$nr]);
```

Rozpraszanie algorytmem Round-Robin:

```
$serwery=array('gdansk.pl', 'sopot.pl', 'gdynia.pl');  
if(file_exists('licznik'))  
    $nr = file_get_contents('licznik');  
else  
    $nr = 0;  
$nr = ($nr+1) % count($serwery);  
file_put_contents('licznik', $nr);  
header('Location: http://'.$serwery[$nr]);
```



Rozpraszanie wagowe ze stałymi wagami:

```
$serwery=array('gdansk.pl', 'sopot.pl', 'gdynia.pl');  
$wagi=array(50, 30, 20);  
$los=mt_rand(0, array_sum($wagi));  
$nr=0;  
$suma=$wagi[0];  
while($los>$suma)  
{  
    $nr=$nr+1;  
    $suma=$suma+$wagi[$nr];  
}  
header('Location: http://'.$serwery[$nr]);
```




Rozpraszanie wagowe z wagami zależnymi od czasu ładowania strony głównej, jako miary zależnej od obciążenia łączy i serwerów:

```
function http_ping($serwer)
{
    $czas = microtime(true);
    $str = file('http://'.$serwer);
    return microtime(true)-$czas;
}
```

Ten przykład należy zrealizować samodzielnie.



- Identyfikacja, weryfikacja i autoryzacja
 - System uwierzytelniania użytkowników z szyfrowaniem z kluczem zmiennym. (Dodatki: ograniczeniem czasowe, kody jednorazowe).
- SQL Injection
 - Rozdzielenie obsługi bazy danych na kompilację SQL i wykonanie SQL z danymi
- Web Switching
 - Rozpraszanie ruchu dla farmy serwerów. (Różne algorytmy: round-robin, wagowy, itp.. Dodatek: algorytm wagowy z dynamicznymi wagami).
- Web Caching
 - Odciążanie serwera webowego w bazodanowej aplikacji internetowej. (Np. galeria obrazów, aplikacja bazodanowa).
- Standardowe biblioteki i technologie
 - Ajax
 - jQuery
 - Bootstrap
 - AngularJS
 - jQueryUI
 - Angular Material