# Protokoły HTTP

HTTP jest obecnie najpopularniejszym i najbardziej rozpowszechnionym protokołem służącym do komunikacji klientów z serwerami. Jego początki nie zakładały tak wszechstronnego użycia jak ma to miejsce obecnie. Niemal wszystkie aplikacje i urządzenia korzystające z sieci internet korzystają z tego rozwiązania.

Pierwszą wersję protokołu HTTP stworzył Tim Berners-Lee, i zakładała ona jak najprostszą komunikację. Został zaprojektowany do przesyłania dokumentów hipertekstowych i oparty na protokołach TCP/IP i oznaczony numerem wersji 0.9. Funkcjonalności jakie zaimplementowano w protokole przedstawiają się następująco:

* Żądanie klienta to prostu ciąg znaków ASCII
* Żądanie klienta zakończone było znakiem nowego wiersza (CRLF)
* Odpowiedź z serwera była prostym strumieniem tekstowym
* Odpowiedź była sformatowana w języku HyperText Markup Language (HTML)
* Po zakończeniu przesyłania dokumentu połączenie było zamykane

W popularnych serwerach takich jak Apache czy Nginx wciąż dostępna jest obsługa protokołu HTML 0.9. Jego działanie jest traktowane raczej jako ciekawostka i wykorzystanie do niekonwencjonalnych zadań.

|  |
| --- |
| $> telnet website.org 80  Connected to xxx.xxx.xxx.xxx  GET /rfc/rfc1945.txt HTTP/1.0  (plain-text response)  (connection closed) |

Powyższe żądanie składa się z metody GET oraz ścieżki do żądanego dokumentu, natomiast odpowiedź jest zwykłym dokumentem HTML. Najprostsze połączenie jakie można stworzyć.

## HTTP 1.0

Największy rozwój specyfikacji protokołu HTML nastąpił w latach 1991 – 1995. W tym czasie powstawało takie oprogramowanie jak przeglądarki internetowe, narzędzia do komunikacji. To był także czas rozwoju infrastruktury całego internetu. Pożądanie nowych funkcjonalności rozwijającego się internetu wymagało czegoś więcej niż HTTP 0.9, który nie zadowalał programistów tworzących aplikacje inne niż klient-serwer = żądanie-odpowiedź.

W Maju 1996 roku zbiór „dobrych praktyk i wzorców” został opublikowany przez „HTTP Working Group” (HTTP-WG) pod nazwą RFC 1945 i była to pierwsza specyfikacja protokołu HTTP 1.0. Specyfikacja ta była jednak jedynie informacyjna, a sam protokół do chwili obecnej nie doczekał się oficjalnej, formalnej dokumentacji i nie jest także oficjalnym standardem, który obowiązuje w internecie.

|  |
| --- |
| $> telnet website.org 80  Connected to xxx.xxx.xxx.xxx  GET /rfc/rfc1945.txt HTTP/1.0 ❶  User-Agent: CERN-LineMode/2.15 libwww/2.17b3  Accept: \*/\*  HTTP/1.0 200 OK❷  Content-Type: text/plain  Content-Length: 137582  Expires: Thu, 01 Dec 1997 16:00:00 GMT  Last-Modified: Wed, 1 May 1996 12:45:26 GMT  Server: Apache 0.84  (plain-text response)  (connection closed) |

❶ - Numer wersji protokołu HTTP oraz następujący po nim nagłówek

❷ - Status odpowiedzi oraz następujący po nim nagłówek

* Przykładowe klasy odpowiedzi Status-Code(PRZYPIS):  
  "200"   ; OK   
  "201"   ; Created    
  "202"   ; Accepted    
  "204"   ; No Content  
  "302"   ; Found  
  "403"   ; Forbidden  
  "404"   ; Not Found

W przeciwieństwie do poprzedniej wersji nastąpiło kilka zmian:

* Żądanie może się składać z kilku pól nagłówka oddzielonych znakami nowego wiersza
* Obiekt z odpowiedzią jest poprzedzony wierszem ze statusem odpowiedzi
* Obiekt zawierający odpowiedź zawiera własny zestaw pól nagłówka oddzielonych znakami nowego wiersza
* Po każdym żądaniu połączenie między klientem a serwerem jest zamykane

Nagłówki żądań oraz odpowiedzi są zapisane w kodzie ASCII, lecz obiekt odpowiedzi może być każdego innego typu np. plikiem tekstowym, plikiem HTML, obrazem itp. Specyfikacja RFC opisuje nie tylko negocjację typu mediów ale także kodowanie treści, autoryzację, autoryzację serwera proxy, obsługę zestawów znaków, formaty daty, typ komunikatów wieloczęściowych, zapis do pamięci podręcznej itd.

## HTTP 1.1

Praca nad dokumentowaniem protokołu w wersji 1.0 trwała około pięciu lat (do 1999 roku). Równolegle trwały prace nad przekształceniem technologii w nowszą wersję oznaczoną jako HTTP 1.1 i to ona jako pierwsza została oficjalnie opublikowana w styczniu 1997 roku w specyfikacji RFC 2068. Dwa lata po tej publikacji zostało wprowadzonych kilka usprawnień w wydajności lecz kod protokołu pozostał taki sam, jednak nowsza wersja specyfikacji to RFC 2616.

Podobnie jak to miało miejsce podczas przechodzenia z wersji 0.9 na nowszą, również teraz znaleziono wiele problemów i wprowadzono usprawnienia w stosunku do wersji 1.0. Przykładowe optymalizacje to między innymi kolejkowanie żądań, transmisja danych w blokach, żądania przesyłania zakresu bajtów, kodowanie przesyłanych danych czy podtrzymywanie połączenia.

Poniżej znajduje się przykładowa sesja HTTP 1.1:

|  |
| --- |
| $> telnet website.org 80  Connected to xxx.xxx.xxx.xxx  GET /index.html HTTP/1.1 ❶  Host: website.org  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_7\_4)... (snip)  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8  Accept-Encoding: gzip,deflate,sdch  Accept-Language: en-US,en;q=0.8  Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.3  Cookie: \_\_qca=P0-800083390... (snip)  HTTP/1.1 200 OK ❷  Server: nginx/1.0.11  Connection: keep-alive  Content-Type: text/html; charset=utf-8  Via: HTTP/1.1 GWA  Date: Wed, 25 Jul 2012 20:23:35 GMT  Expires: Wed, 25 Jul 2012 20:23:35 GMT  Cache-Control: max-age=0, no-cache  Transfer-Encoding: chunked  100 ❸  <!doctype html>  (snip)  100  (snip)  0 ❹  GET /favicon.ico HTTP/1.1 ❺  Host: www.website.org  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_7\_4)... (snip)  Accept: \*/\*  Referer: http://website.org/  Connection: close ❻  Accept-Encoding: gzip,deflate,sdch  Accept-Language: en-US,en;q=0.8  Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.3  Cookie: \_\_qca=P0-800083390... (snip)  HTTP/1.1 200 OK ❼  Server: nginx/1.0.11  Content-Type: image/x-icon  Content-Length: 3638  Connection: close  Last-Modified: Thu, 19 Jul 2012 17:51:44 GMT  Cache-Control: max-age=315360000  Accept-Ranges: bytes  Via: HTTP/1.1 GWA  Date: Sat, 21 Jul 2012 21:35:22 GMT  Expires: Thu, 31 Dec 2037 23:55:55 GMT  Etag: W/PSA-GAu26oXbDi  (icon data)  (connection closed) |

❶ - Żądanie pliku HTML wraz z metadanymi opisującymi kodowanie, zestaw znaków i ciasteczka.

❷ - Blok z odpowiedzią na pierwsze żądanie HTML.

❸ - Liczba bajtów w bloku danych wyrażona jako liczba szesnastkowa w kodzie ASCII (256 bajtów)

❹ - Koniec strumienia bloków danych z odpowiedzią.

❺ - Żądanie pliku ikony przesłany tym samym połączeniem TCP

❻ - Informacja do serwera, że połączenia nie będzie ponownie wykorzystane.

❼ - Odpowiedź z plikiem ikony i zamknięcie połączenie

Od razu rzucają się w oczy dwa obiekty żądań TCP, co wcześniej było niemożliwe poprzez zamykane połączenia po zakończeniu żądania. Podtrzymywanie i ponowne wykorzystanie tego samego połączenia znacznie umożliwia znacznie szybsze dostarczenie mu odpowiedzi. W celu zamknięcia połączenia wykorzystany jest nagłówek (Connection: Close).

W ulepszonej wersji protokołu HTTP 1.0 też została wprowadzona funkcja podtrzymywania połączenia poprzez użycie nagłówka (Connection: Keep-Alive). Nagłówek ten jednak nie jest wymagany jeśli serwer aplikacji webowej korzysta z nowszej wersji protokołu.

## HTTP 2.0

Coraz powszechniejsze i bardziej zaawansowane aplikacje webowe, już nawet nie statyczne czy interaktywne strony WWW, wymagają także większej wydajności. W odpowiedzi na to z pozoru banalne zapotrzebowanie powstaje protokół HTTP w wersji 2.0. Zakłada on zmniejszenie opóźnień, zwiększenie wydajności transmisji danych oraz uzyskanie większej przepływności. W tym momencie nie planuje się zmieniać takich cech protokołu jak nagłówki HTTP oraz ich wartości i funkcje.

Wszystkie aplikacje webowe, które w tej chwili już zostały uruchomione na serwerach w momencie pełnego wdrożenia nowego protokołu sieciowego będą z niego korzystały bez konieczności wprowadzania zmian w kodzie aplikacji. Ważne jest aby serwer oraz klient obsługiwały to połączenie. Zmiany w działaniu aplikacji powinny być jednak zauważalne jeśli prócz serwera, aplikacja jest dobrze skonfigurowana i wolna od błędów jakie może wskazać np. „Snail Project” przygotowany specjalnie w celu wykonania testu wydajnościowego aplikacji. Warto także pamiętać, że tworząc najbardziej nowoczesną aplikację warto zadbać o kompatybilność wsteczną protokołu. Serwer Apache (jeśli programista nie skonfiguruje połączenia inaczej) zawsze wykorzystuje najwyższą możliwą wersję protokołu HTTP.

http://www.drzewo-wiedzy.pl/?page=artykul&id=70

# Wydajność części front-endu i back-endu

Front end to część strony czy aplikacji webowej, którą użytkownik może zobaczyć lub „doświadczyć”. Ponieważ ludzi uważa się za wzrokowców, doświadczenia użytkownika są bardzo ważne, aby przyciągnąć uwagę odbiorcy.

Na „front end” składa się między innymi hipertekstowy język znaczników – HTML. Kaskadowe arkusze styli w plikach z rozszerzeniem .CSS, mające na celu „ozdobienie” wcześniej wspomnianych znaczników czy zawartych w nich informacji. Współczesnym użytkownikom często serwuje się także odrobinę „magii” w postaci skryptów języka JavaScript czy biblioteki jQuery. Wszelkie pojawiające się i znikające elementy, ruszające się przyciski czy wyskakujące okienka, można stworzyć pisząc skrypty, które później są dołączane do plików .html.

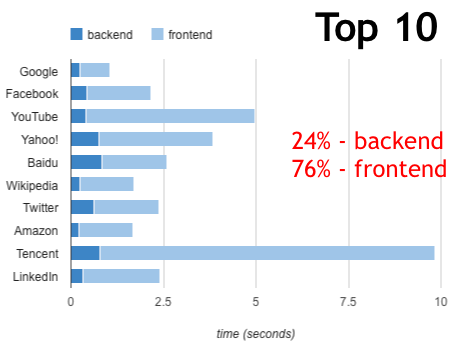
Prócz dostarczania User eXperience zadaniem front endu jest także komunikacja z back endem aplikacji przekazywanie jej interakcji użytkownika lub wyświetlanie mu informacji od serwera. Back end zazwyczaj jest dzielony na trzy części: serwer, aplikację (działającą na serwerze) oraz bazę danych. Każda z nich jest w pewien sposób uniwersalna np. gdyby aplikację (a raczej logikę aplikacji) przenieść z jednego serwera na inny oraz podpiąć do bazy danych z innymi (ale spójnymi) danymi, aplikacja też by działała. Żeby zaprogramować aplikację działającą po tzw. stronie serwera najczęściej korzysta się z takich języków programowania jak: PHP, Java, Ruby czy Python.

Wybór konkretnej technologii to bardzo ważny element tworzenia aplikacji webowej, jednak dyskusja na temat, który z nich jest najszybszy, najwydajniejszy, czy którego kod jest najłatwiejszy w utrzymaniu może pochłonąć kilka lat. Przede wszystkim należy dobrać odpowiednio język do tworzonej aplikacji. Dla wszystkich jednak można korzystać ze wspólnych koncepcji backendowych, wymienionych między innymi w opisanym projekcie „Snail Project”. Sugestie zawarte w wynikach stanowią uniwersalne rozwiązania, które każdy programista może zastosować w swoim projekcie aby jego aplikacja działała bardziej wydajnie.

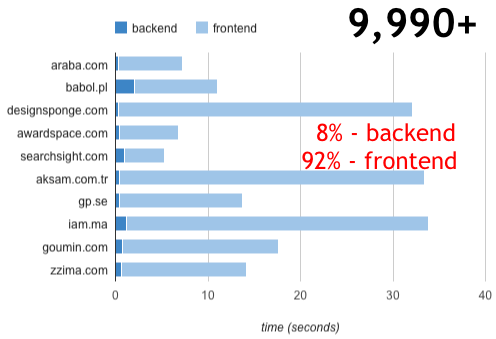
Działania części określanej jako front end oraz back end mimo różnych zadań są ze sobą połączone. Przykładowo osoba chcąca zarezerwować bilet lotniczy, po wpisaniu odpowiedniego adresu w swojej przeglądarce internetowej w pierwszej kolejności doświadczy kontaktu z wyglądem strony. Wszystkie informacje, które można od razu przeczytać na pierwszej stronie zazwyczaj nie wymagają od razu połączenia z bazą danych. Dopiero w momencie sprawdzania aktualnej informacji i ilości miejsc dla danego lotu, następuje komunikacja z serwerem, wyszukanie odpowiednich informacji w bazie danych, ewentualna filtracja ich przez kontrolery i zwrócenie ich do części wizualnej, gdzie użytkownik sam dokona wyboru czy dokonać kolejnej akcji np. rezerwacji biletu. Ciężko by było stworzyć taką aplikację gdyby nie wykorzystać bazy danych znajdującej się po stronie serwera czy nie używać arkusza styli lub skryptu pobierającego wartość maksymalnej ceny biletu podanej przez użytkownika.

Steve Souders na swoim blogu(PRZYPIS) opublikował testy dotyczące czasu jaki jest potrzebny do załadowania się strony internetowej pod względem podziału na front end oraz back end. Wyniki te dowodzą, że front end to nawet 92% czasu jaki jest potrzebny na gotowość aplikacji.

Poniżej przytoczono wyniki jakie uzyskał Steven Souders. Sprawdzian został przeprowadzony dla serwisów znajdujących się w rankingu TOP 10 najczęściej odwiedzanych stron internetowych na świecie.



Top 10 najpopularniejszych stron WWW



Ostatnie 10 stron WWW rankingu top 10000

Powyższy wykres przedstawia sytuację dla serwisów internetowych dla dziesięciu ostatnich z rankingu TOP 10 000.

Opublikowane rezultaty są jednoznaczne, back end to część pracy programisty, który w najmniejszym stopniu pływa na wydajność działania aplikacji webowej. Kiedy nie zastosuje się rad, które sugeruje wykonany i opisany „Snail project” nawet najwydajniejszy na świecie serwer nie poradzi sobie z odpowiedziami na „niepotrzebne” żądania czy przesyłaniem do klienta „ciężkich” plików graficznych lub niezoptymalizowanych skryptów blokujących ładowanie się pozostałej części strony.

<http://www.stevesouders.com/blog/2012/02/10/the-performance-golden-rule/>

# Wydajność sieci bezprzewodowych

Początek sieci bezprzewodowych sięga lat 70 XX wieku kiedy to powstała pierwsza sieć o nazwie AlochaNET. Była to pierwsza publiczna prezentacja sieci bezprzewodowej opracowanej w 1971 roku na Uniwersytecie Hawajskim. Sposób działania był bardzo prosty nadawca zaczynał transmisję informacji zawartych w ramkach, a następnie oczekiwał potwierdzenia odebrania ramki od odbiorcy. Po tak prostej komunikacji cała transmisja zaczynała się od nowa. To rozwiązanie miało bardzo dużo wad gdyż sposób działania przypominał komunikację „walkie talkie” komunikacja mogła następować z jednej strony jednocześnie. To jeden z największych problemów tego rozwiązania eliminował je z wykorzystania go do zaawansowanych rozwiązań technologicznych. Rewolucją okazało się opracowanie przez Robert Metcalfe i David Boggs'a, Etherntu opartego na kablu koncentrycznym. Ostateczna specyfikacja komunikacji przez Ethernet została opracowana przez takie firmy jak Xerox, Intel i Digital Equipment i opublikowana w 1980 roku. W 1985 roku dokument został zaakceptowany przez "Institute of Electrical and Electronics Engineers", ustanawiając normę IEEE 802.3. Od tego momentu dzięki szerokiemu zastosowaniu w przemyśle, rachunkowości, administraji sieć Ethernet zyskiwała na popularności. Pierwsze karty bezprzewodowe zostały wyprodukowane w latach 90, jednak ich cena oraz dostępność w sklepach blokowały rozwój technologii sieciowo-radiowych. Organizacja Institute of Electrical and Electronics Engineers w 1997 roku organizacja stworzyła standard sieciowy na częstotliwości radiowej 2,4 GHz, uzyskał on oznaczenie 802.11. Standard posiadał bardzo niską przepustowość około 2 Mb/s. Dopiero rozwój tego rozwiązania oznaczone 802.11b umożliwiał korzystanie z prędkości do 11Mb/s, a zastosowanie częstotliwości 2,4 GHz umożliwiło działanie sieci radiowej bez zakłóceń przez urządzenia wykorzystujące mikrofale. Kolejnym krokiem było stworzenie standardu 802.11a, który działał przy wykorzystaniu częstotliwości 5GHz i przepustowości 54Mb/s. Drugi standard sieci nie zyskał poparcia przez brak kompatybilności ze standardem 802.11b oraz mniejszy zasięg efektywny a także wysoką ceną urządzeń obsługujących ten standard. Jedynym atutem tego standardu było to że posiadał osiem nie pokrywających się kanałów dla częstotliwości fal radiowych. Następnie w roku 2002 na targach Comdex zaprezentowano nowe urządzenie pracujące w standardzie IEEE 802.11g. Standard bardzo szybko został zaakceptowany przez organizacje IEEE i to spowodowało iż znalazł byt u większości producentów.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Protokół 802.11 | Rok publikacji | Częstotliwość (GHz) | Szerokość pasma (MHz) | Prędkość transmisji danych na strumień (Mb/s) |
| g | 1999 | 2,4 | 20 | 1, 2, 5.5, 11 |
| b | 2003 | 2,4 | 20 | 6, 9, 12, 18, 24, 36, 48, 54 |
| n | 2009 | 2,4 | 20 | 7.2, 14.4, 21.7, 28.9, 43.3, 57.8, 65, 72.2 |
| n | 2009 | 5 | 40 | 15, 30, 45, 60, 90, 120, 135, 150 |
| ac | 2014 | 5 | 20,40,50,160 | ~866,7 |

Obecnie najbardziej rozpowszechnionymi standardami są „b” oraz „g” korzystające z szerokości pasma 2,4 GHz. Jeśli chce się przyspieszyć działanie sieci najlepiej wykorzystać standardy 802.11 „n” oraz wdrażany „ac”. Najnowsze trendy podwajają szerokość pasma z 20 MHz do 40 MHz na każdy kanał. Wykorzystują modulację wyższego rzędu oraz zwielokrotniają moduły radiowe tak aby transmitować wiele strumieni jednocześnie.

Sieci bezprzewodowe przeżywają w ostatnich latach niekończącą się ewolucję. Powszechny dostęp do internetu staje się wręcz standardem dla ludzkości. Dostęp do poczty elektronicznej, obsługa połączeń telefonicznych, przeglądanie stron www, z każdego miejsca na ziemi niezależnie od położenia to oczekiwania obecnej społeczności.

Siecią bezprzewodową jest każda sieć, która nie łączy urządzeń wykorzystywanych do komunikacji za pomocą kabla. Sieć LTE jest w stanie transmitować dane z danego serwera do urządzenia mobilnego użytkownika, tak aby mógł on obejrzeć daną stronę internetową bez użycia kabla typu skrętka. Aplikacje internetowe mogą być dostarczane nie tylko przy użyciu sieci mobilnych jak 4G, systemy webowe korzystające z płatności mogą używać sieci NFC, Bluetooth do komunikacji P2P czy Wi-Fi ale nie do przesyłania danych po drodze serwer-klient, tylko dla przykładu używania aplikacji strumieniującej obraz wideo.

Mechanizmy dostarczania danych za pomocą komunikacji radiowej bardzo się różnią od architektury „kablowej” jednak doświadczenia użytkownika powinny być identyczne. Zarówno jeśli chodzi o wydajność jak i oczekiwane rezultaty takiego połączenia.

Obecnie najpopularniejszymi sieciami bezprzewodowymi są technologie takie jak LTE, HSPA, Wi-Fi, WiMax, Bluetooth czy standardy 3G. Żeby zrozumieć podstawowe zasady działania i założenia takich sieci opisane zostaną te, z których najczęściej korzystają użytkownicy aplikacji internetowych. Ponieważ programiści optymalizując swoje aplikacje internetowe pod względem wydajności i sposobu działania sieci bezprzewodowych mogą jedynie zyskać w oczach użytkowników, nie należy pomijać tych informacji.

Matematyczny model Twierdzenia Shannona-Hartleya (PRZYPIS), pomaga w określeniu pojemności kanału niezależnie od zastosowanej technologii.

C - oznacza pojemność kanału w bitach na sekundę  
BW – dostępna szerokość pasma w hercach  
S – sygnał w watach  
N – szum w watach

Powyższa bardzo uproszczona formuła przedstawia najważniejsze aspekty połączenia bezprzewodowego. Prócz wykonania testów wydajnościowych np. za pomocą projektu(PRZYPIS) opisanego we wcześniejszej części pracy, oraz optymalizacji części front-endowej czy back-endowej należy także mieć świadomość jak działa większość bezprzewodowych sieci bez znaczenia jaki jej rodzaj, wersja, czy akronim został akurat zastosowany. Dostępna szerokość pasma, siła sygnału czy odległość między nadajnikiem i odbiornikiem też w istotny sposób wpływają na prędkość przesyłania danych do np. urządzeń mobilnych.

Nie wszystkie zakresy częstotliwości są tak samo wydajne. Im niższa częstotliwość nadawania sygnału tym większy zasięg i pokrycie lecz wymaga to budowy większej infrastruktury np. długich anten. Odwrotnie jest w przypadku użycia sygnału o wysokiej częstotliwości, można wtedy przesyłać większą ilość danych ale zasięg jest dużo mniejszy. To znaczy pokrycie jest mniejsze oraz wymaga większej ilości nadajników tak aby klient łączył się z najsilniejszym z nich.

Korzystanie z niektórych zakresów częstotliwości jest bardziej wydajne od innych. Na przykład aplikacje, które wykorzystują tylko nadawanie (np. gdyby użyto infrastruktury i aplikacji wysyłającej tylko powiadomienia push), będą działać najlepiej na niskich częstotliwościach. Natomiast aplikacje wykorzystujące komunikację dwukierunkową, będą działały dużo wydajniej kiedy zastosuje się mniejsze komórki, zapewniające wyższą przepustowość i jednocześnie mniejszą konkurencję.

Siła sygnału jest, po szerokości pasma, drugim najważniejszym czynnikiem, który może stanowić pewne ograniczenia w całej sieci bezprzewodowej. Jak wspomniano wcześniej siła sygnału to stosunek sygnału (mierzonego w watach) do szumu (również mierzonego w watach) . Sygnał musi być tym silniejszy im większe zakłócenia występują na jego drodze. Zakłócenia mogą być emitowane przez różne urządzenia emitujące fale radiowe o częstotliwości zbliżonej do 2,5 GHz np. kuchenka mikrofalowa. Najczęściej są to jednak inne urządzenia Wi-Fi działające na tym samym kanale lub częstotliwości i szerokości wykorzystywanego pasma, takie jak punkty dostępu na przystankach autobusowych, czy routery sąsiadów.

Czynniki wpływające na wydajność sieci bezprzewodowych to nie różnica odległości między odbiorcą i nadawcą. Ważny jest poziom szumów w danych lokalizacjach. Moc przetwarzania oraz schemat modulacji sygnału. Siła nadawanego sygnału oraz moc jego odbierania. Zakłócenia generowane przez użytkowników w tych samych sieciach czy lokalizacjach oraz użytkowników znajdujących się w podobnej lokalizacji lecz innych sieciach.

W świecie idealnym każda sieć bezprzewodowa byłaby nadawana w innej szerokości pasma i na innej częstotliwości tak aby sieci nie zakłócały się wzajemnie. Jednak jest to niemożliwe do zastosowania, szczególnie w dużych miastach i aglomeracjach takich jak Wrocław, gdzie bezprzewodowa komunikacja jest bogato wykorzystywana. Aby aplikacja webowa była wydajna i działała w „każdych warunkach”, należy mieć na uwadze wykorzystanie i sposób działania oraz wydajność właśnie takich sieci.

<http://pl.wikipedia.org/wiki/Twierdzenie_Shannona-Hartleya>

# Mobilne witryny internetowe

Obsługa stron internetowych z puntu widzenia telefonów komórkowych niewiele się różni od obsługi tych samych witryn za pomocą komputera stacjonarnego czy laptopa. Mobilna przeglądarka internetowa po wywołaniu adresu odpowiedniej strony, szuka serwera o określonym adresie IP, wysyła mu żądanie wyświetlenia zawartości strony internetowej i czeka na odpowiedź generując użytkownikowi telefonu gotową stronę www. Wszystko wydaje się być proste a sposób działania znany od kilkudziesięciu lat. Dlaczego w takim w razie większość programistów dba o to aby ich strony internetowe były przystosowane do obsługi na urządzeniach takich jak smartfony czy tablety ? Największy problem to przepustowość oraz opóźnienia sieci takich jak 2G, 3G, czy 4G, z których korzystają urządzenia mobilne.

Według dokumentacji Google Insights (PRZYPIS), badania przeprowadzone przez Google określają, że uwaga skupiona przez osobę obsługującą np. tablet na ładującej się stronie internetowej nie przekracza jednej sekundy. Rozkojarzenie spowodowane zbyt długim oczekiwaniem na pojawienie się zawartości strony może doprowadzić do porzucenia strony przez użytkownika na rzecz innej, która wyrenderuje się szybciej.

Oczywiście obecne rozwiązania technologiczne nie pozwalają na pełnie wyświetlenie się strony w określonym przez Google czasie. Należy zadbać jednak aby część widoczna na ekranie pojawiła się możliwe jak najszybciej tak, aby skupić uwagę odbiorcy i umożliwić mu korzystanie ze strony, a pozostałe elementy systematycznie przesyłać do urządzenia.

Kryterium renderowania widocznej części witryny w czasie krótszym niż jedna sekunda to pewne wyzwanie nie tylko dla developerów ale także dla mobilnych sieci, ponieważ niektóre problemy nie występują w innych sieciach kablowych czy światłowodowych. Obecnie najpopularniejszą siecią na świecie jak i w Polsce jest tzw. 3G bazująca na standardzie UMTS. Jej następca czyli 4G jest dopiero w pełni wdrażana i rozpowszechniana na świecie, przez producentów jak i dostawców telefonii komórkowej. Dotyczy to nie tylko budowanej infrastruktury, ale także najnowszej wersji telefonów obsługujących daną wersję standardu sieci telekomunikacyjnej.

Powyższy wykres przedstawia typową sekwencję komunikacji między przeglądarką a serwerem. 600 milisekund zostało wykorzystane na wyszukiwanie DNS i identyfikację nazwy hosta (np. pwr.edu.pl) powiązanej z adresem IP, wymianę danych w związku z uzgodnieniami TCP oraz przesłanie żądania HTTP. W pozostałych 400ms serwer musi dać odpowiedź klientowi, aplikacja kliencka musi wykonać kod a przeglądarka wyrenderować treść i pokazać ją użytkownikowi urządzenia mobilnego.

Aby zmieścić się w ciągu jednej sekundy serwer powinien zwrócić początkową część HTML-a w czasie 200ms lub nawet, krótszym aby mieć cenny zapas czasu na przesłanie danych.

Należy się przyjrzeć przekierowaniom na swojej stronie tak aby ich liczba była jak najmniejsza. Ponieważ każde dodatkowe przekierowanie HTTP to dodatkowy czas na wymianę danych lub wyszukanie serwera DNS, co przekłada się na większe opóźnienia w sieciach 3G i wyższych.

Protokół TCP szacuje wydajność połączenie według „Strategii powolnego startu”. Według założeń ma zapobiegać spowolnieniom w protokole transmisji danych. „Slow-start” (and. Powolny start) zazwyczaj jest używany równolegle z innymi algorytmami, które zapobiegają wysłaniu do sieci większej ilości danych niż jest ona w stanie odebrać.

Działanie algorytmu polega na tym, że okno TCP początkowo mieści jeden maksymalny segment. Okno to jest zwiększane o jeden segment wraz z każdym odebranym pakietem ACK to znaczy, że okno podwaja się co RTT. Powolny start kończy działanie, kiedy nastąpi utrata pakietu lub okno osiągnie maksymalną szerokość.

Na podstawie powyższego opisu działania algorytmu powolnego startu TCP, należy wywnioskować, że każde nowe połączenie klient- serwer nie korzysta od razu z pełnej przepustowości łącza. Biorąc pod uwagę fakt, że w ramach nowego połączenia serwer jest w stanie wysłać maksymalnie 10 pakietów czyli ok 14 KB, a następnie żeby zwiększyć transfer musi poczekać na odpowiedź klienta, należy ograniczyć liczbę sesji wymiany danych. Można tego dokonać poprzez zminimalizowanie rozmiaru pliku z widoczną częścią strony, aby ten nie przekraczał wspomnianych 14 KB. Zabieg ten pozwoli na to żeby przeglądarka w ciągu jednej sesji wymiany danych była w stanie odrysować szablon strony.

Tak jak we wcześniej opisanych testach „pełnych stron www”, również w wersjach mobilnych należy unikać kodów JavaScript umieszczonych w osobnych blikach, blokujących zasoby strony w części widocznej na ekranie użytkownika. Jeśli przeglądarka renderując widok strony napotka w kodzie link do pliku ze skryptem HTML zacznie go interpretować. Dodatkowo jeśli będzie to skrypt zewnętrzny lub niesynchroniczny, program zatrzyma wczytywanie następnych elementów strony, dopóki nie przetworzy obecnych zasobów. Zarówno kod CSS jak i JavaScript, jeśli są niezbędne do działania widocznej dla odbiorcy części strony, powinny zostać umieszczone bezpośrednio w kodzie HTML. Rozwiązaniem pośrednim może być umieszczenie odpowiednich skryptów czy arkuszy styli na końcu kody HTML. W ten sposób użytkownik zacznie analizować widoczny dla niego tekst, czy szablon strony a dodatkowe wizualne efekty zostaną załadowane w odpowiedniej kolejności. Często w kodzie używana jest biblioteka jQuery, która wzbogaca statyczne strony o animacje i dodatkowe funkcjonalności. Bez najmniejszych obaw jej opcje można wywoływać już po załadowaniu się strony, tak aby nie blokowała wczytywania się pozostałych elementów. Wpłynie to pozytywnie na skrócenie czasu renderowania witryny.

Interpretacja kodu JavaScript jak i CSS powinna zająć ostatnie 200ms dopełniające pełną sekundę, lecz jest to dość problematyczne do określenia ze względu na różnorodność urządzeń mobilnych. Tablety czy telefony różnią się od siebie nie tylko pamięcią, mocą procesora, ilością rdzeni i szybkością działania aplikacji jak i samego mobilnego systemu operacyjnego. W związku z tą różnorodnością, łatwiej jest zadbać o kod strony, jej poziom skomplikowania oraz całkowitą spójność i kompresję punktów składowych, tak aby jej szybkość i poprawność działania pozytywnie zaskakiwała odbiorców wszystkich możliwych urządzeń wyświetlających konkretną witrynę internetową.

Sieć 4G to następca systemu trzeciej generacji. Założenia nowszego standardu są następujące:

* Komutacja pakietów na protokole IP
* Rzadsze występowanie przestojów i błędów transferu
* Istotne przyspieszenie możliwości transferowych
* Szybszy czas reakcji w porównaniu do sieci 3G
* Uproszczenie struktury sieci szkieletowej

W rezultacie dostęp do strony jest szybszy i mniej wadliwy. Nie należy jednak w pełni ufać nowemu standardowi i zaniedbać optymalizację własnego serwisu www. Zdecydowana większość ludzi na świecie nie posiada najnowszych urządzeń obsługujących sieci LTE i wciąż korzystają z technologii przesyłania danych 3G.

Nad najnowszą technologią przesyłu danych prócz takich potęg jak Nokia, Huawei(PRZYPIS), Samsung, Ericsson(PRZYPIS) pracuje także Unia Europejska(PRZYPIS).

Korzyści wynikające z założeń sieci 5G mogą być przełomowe. Nowy system ma charakteryzować się odpornością spadki wydajności związane z liczbą jednocześnie korzystających z sieci użytkowników. Prędkość przysyłania danych ma osiągać nawet 100 Gbps (obecnie LTE osiąga przepustowość do 300 Mbps).

Twórcy najnowszej technologii (PRZYPIS) szacują pojawienie się najnowszej edycji standardu telefonii komórkowej dopiero w roku 2025.

Bardziej efektywne wykorzystanie TCP można uzyskać poprzez użycie protokołu HTTP 2.0 czy rozszerzenia HTTP 1.1 – SPDY. Nowy standard protokołu przesyłania dokumentów hipertekstowych nie jest jeszcze w pełni ukończony, jednak można z nich korzystać w wersjach testowych modułów w serwerów takich jak Nginx czy Apache. Pozwoli to między innymi na kompresję nagłówków, używanie priorytetów czy użycie mechanizmu Server Push, który poprzez eliminację opóźnień sieci zwiększa wydajność połączenia.

<https://developers.google.com/speed/docs/insights/mobile?hl=pl>

<http://www.huawei.eu/files/publications/pdf/huawei_5g_white_paper_en_20140129.pdf>

<http://www.ericsson.com/res/docs/whitepapers/wp-5g.pdf>

http://www.komputerswiat.pl/nowosci/internet/2014/25/ue-i-korea-poludniowa-beda-wspolnie-rozwijac-internet-5g.aspx

# Opis narzędzia Snail Project

Agile  
W trakcie tworzenia Projektu wykorzystane zostało programowanie zwinne (ang. Agile software development). Ponieważ założeniem było jak najszybsze otrzymywanie wyników poprzez wykonywanie skryptów, layout był tworzony „w trakcie” programowania innych zadań. Programowanie iteracyjno-przyrostowe, powstałe jako alternatywa do tradycyjnych metod typu waterfall, świetnie się sprawdziło podczas implementowania różnego rodzaju skryptów np. Insight. Jednym z założeń rozwiązania Agile jest praca w zespołach, które mogą niezależnie od siebie tworzyć pewne części danego projektu. W tym przypadku zespół składał się z jednej osoby oraz „odbiorcy”. Współpraca owocowała wieloma pomysłami oraz zmianami w trakcie tworzenia projektu, podejście typu Waterfall mogłoby się zakończyć niepowodzeniem projektu, lub brakiem możliwości wprowadzania jakichkolwiek zmian po fazie projektowej.

Agile bardzo często znajduje zastosowanie w małych zespołach programistycznych, w których jest dobra komunikacja, tak by nie tworzyć rozbudowanej dokumentacji kodu. Kolejne etapy wytwarzania oprogramowania zamknięte są w iteracjach, w których za każdym razem przeprowadza się testowanie wytworzonego kodu, zebranie wymagań, planowanie rozwiązań itd. Metoda nastawiona jest na szybkie wytwarzanie oprogramowania wysokiej jakości oraz regularna adaptacja do zmieniających się wymagań.

## Pierwsza iteracja Snail Project

Zanim zaimplementowano skrypty badające wydajność serwisów webowych, stworzony został *layout* Projektu. Po wpisaniu odpowiedniego adresu URL w przeglądarce wyświetli się górny pasek szerokości okna przeglądarki, na którym znajduje się nazwa projektu oraz adres IP użytkownika, który odwiedził stronę. Pozostałą część okna zajmuje obraz ślimaka, stanowiący symbol projektu, a na jego środku widnieje miejsce, w którym użytkownik powinien wprowadzić adres URL w postaci „[www.pwr.edu.pl](http://www.pwr.edu.pl)” bez poprzedzającego adres prefiksu http:// oraz przycisk „Test”, który po wciśnięciu uruchomi wszystkie skrypty dokonujące odpowiednich analiz, parsując wyniki i wyświetli je w określony odpowiednimi stylami sposób, aby były czytelne i zrozumiałe dla odbiorcy. Podczas oczekiwania na wykonanie się skryptów testujących do momentu wyświetlenia wyników widoczny jest „loader” w postaci obracających się kropek sugerujący, że właśnie trwa wykonywanie testu i należy poczekać na jego wynik. Rozmiar obrazu ze ślimakiem jest przystosowany to szerokości i wysokości okna przeglądarki a wyniki prezentowane są na białym półprzezroczystym tle, które jest nałożone na estetyczny gradient, tło całej witryny.

Do nadania Projektowi ostatecznego kształtu wykorzystane zostały takie technologie i frameworki jak PHP, JavaScript, Ajax, Bootstrap, Mustache.js, jQuery oraz oczywiście HTML oraz CSS. Wszystko służyło odpowiedniemu ułożeniu i pozycjonowaniu elementów, oraz odpowiedniemu parsowaniu wyników zwracanych przez skrypty.

Cała aplikacja webowa jest anglojęzyczna ponieważ czyni to ją bardziej uniwersalną i dostępną również dla programistów z zza granicy, którzy chcą dokonywać testów wydajności i optymalizacji stron internetowych.

## PageSpeed Insight

Kolejną iteracją, zgodnie z założeniami programowania zwinnego, lecz pierwszym zaimplementowanym skryptem w projekcie został Page Speed Insights, który mierzy wydajność stron na komputerach. Skrypt w projekcie pobiera adres strony internetowej podanej przez użytkownika, parsuje go a następnie przesyła do serwera Google, gdzie następuje analiza strony. Po wykonaniu testu wyniki są zwracane do skryptu zaimplementowanego w Snail. Są to wyniki w postaci obiektów, następnie są one parsowane i wyświetlone użytkownikowi w sposób czytelny i zrozumiały w postaci tabeli.

Rezultat działania PageSpeed mieści się w zakresie od 0 do 100 punktów. Im wyższy wynik, tym lepszy. Wynik na poziomie co najmniej 85 punktów oznacza, że strona działa dobrze. Jednak na potrzeby Projektu, wynik jest wyświetlany w postaci procentowej. Po krótkim badaniu na grupie około 30 studentów, określono, że procent zoptymalizowania serwisu webowego jest dla odbiorców bardziej znaczącym wynikiem niż punkty. Google ostrzega, że narzędzie PageSpeed Insights jest nieustannie usprawniane, więc wynik może się zmieniać, gdy zostaną dodane nowe reguły oceny serwisów.

Insights sprawdza, jak można poprawić wydajność strony w następujących aspektach:

* czas wczytywania części strony widocznej na ekranie: czas, który upływa od momentu wysłania żądania nowej strony do momentu wyrenderowania części strony widocznej na ekranie przez przeglądarkę.
* czas pełnego wczytania strony: czas, który upływa od momentu, gdy użytkownik wysyła żądanie nowej strony do momentu, gdy strona zostanie w pełni wyrenderowana przez przeglądarkę.

Ze względu na dużą zmienność wydajności połączenia sieciowego PageSpeed Insights uwzględnia wyłącznie aspekty wydajności renderowania strony niezależne od sieci: konfigurację serwera, strukturę kodu HTML strony, a także korzystanie z zasobów zewnętrznych, takich jak obrazy, pliki JavaScript i CSS. Zastosowanie się do sugestii powinno poprawić względną wydajność strony, jednak wydajność bezwzględna wciąż pozostanie zależna od połączenia sieciowego użytkownika.

Skrypt zwraca wynik działania zawsze tych samych wartości:

* Unikanie zbędnych przekierowań
* Włączenie kompresji danych
* Wykorzystanie pamięci podręcznej przeglądarki
* Skrócenie czasu odpowiedzi serwera
* Minimalizowanie plików CSS
* Minimalizowanie plików HTML
* Minimalizowanie plików JavaScript
* Eliminowanie blokujących skryptów
* Optymalizacja obrazów
* Priorytetyzacja widocznej treści

Insights w Snail Project określa stopień wydajności strony w procentach. Rezultat „This site is optimized in 70%” należy czytać jako „Strona jest zoptymalizowana w 70 procentach”. Im wyższy wynik osiąga testowana aplikacja, tym mniej „kroków” należy wykonać aby zoptymalizować jej wydajność i jakość działania. PageSpeed dokonuje analizy stron internetowych za pomocą reguł ustalonych przez Google. Każda reguła opiera się na ogólnych zasadach działania strony internetowej, takie jak buforowanie zasobów, przekazywanie danych i wielkości pobierania danych, odpowiednią minimalizację plików ze stylami, skryptami itp.

Każda reguła generuje zmiennoprzecinkową wagę, priorytet, z jakim należy uwzględnić pierwszeństwo wprowadzania zmian aby osiągnąć lepszą wydajność. Na przykład, jeśli optymalizacja obrazów na testowanej stronie internetowej pomoże zmniejszyć ich rozmiar o 1 MB, to ta reguła otrzyma większą przykładowo o 5,5845 wagę niż zminimalizowanie rozmiaru plików CSS, które np. zmniejszą rozmiar przesyłanych danych o 300 KB. Ważne jest jednak aby zastosować wszystkie sugestie, które przy ponownym wykonaniu mogą dać lepszy rezultat punktowy. Innym przykładem może być sugestia włączenia kompresji danych w ustawieniach serwera, która będzie dwa razy ważniejsza od optymalizacji samych obrazów na stronie. Tabela wyników jest ustalona względem najważniejszych cech jakie należy wprowadzić (według Google), lecz w drugiej kolejności należy zwracać uwagę na odpowiednie wagi, które wyświetlają się po wykonaniu testu.

## GTMetrix

Zgodnie z zasadą programowania zwinnego, kolejną iteracją w projekcie Snail było zaimplementowanie skryptu, który niezależnie od miejsca przebywania użytkownika (w tym przypadku, osoby, która dokonuje testu) wykona analizę strony pod względem długości ładowania się elementów, czasu odpowiedzi, dokona pomiaru rozmiaru strony oraz utworzy wykres typu *waterfall*, z wynikami aby były one zrozumiałe i czytelne dla odbiorcy.

GTmetrix jest projektem rozwijanym przez wyspecjalizowanych programistów w firmie „Gossamer Threads” w Vancouver. Firma posiada ponad 19 lat doświadczenia w pracy z technologiami webowymi. Serwis ten nie posiada swojego własnego systemu oceny strony, korzysta z PageSpeed Insight oraz narzędzia Yslow do prezentowania wyników użytkownikowi. Serwis ten udostępnia jednak API do skryptu, który spełnia założenia analizy strony, oraz umożliwia implementację na własnej stronie internetowej, co sprawiło, że został on wykorzystany w Snail Project. Aby móc skorzystać z tego rozwiązania należało dokonać rejestracji na stronie [www.gtmetrix.com](http://www.gtmetrix.com) oraz wygenerować indywidualny „API Key” dla developera.

Skrypt pobiera adres z paska na głównej stronie Projektu, a następnie przesyła go na własny serwer i dokonuje weryfikacji, przed nieuprawnionym użyciem. Następnie żądanie serwera WWW jest przesyłane przez obiekt XMLHttpRequest do skryptu na serwerze Projektu Snail. JavaScript przetwarza odpowiednio ten obiekt prezentując zwrócony wynik w formie graficznej w odpowiednim miejscu na stronie. Cały test jest wykonywany przez „user-agent: Mozilla/5.0 (X11; Linux i686 on x86\_64; rv:25.0) Gecko/20100101 Firefox/25.0”

## Moderniz

Korzystanie z nowych technologii internetowych może wydawać się świetną zabawą, dopóki nie muszą wspierać przeglądarek, które często nie obsługują najnowszych trendów, celowo bądź do kolejnej aktualizacji przeglądarki. Modernizr to biblioteka, która ułatwia pisanie warunkowego kodu JavaScript i CSS do obsługi użycia w każdej sytuacji. Modernizr dokonuje sprawdzenia obsługi nowych trendów (np. obsługi cieni, responsywności, HTML Audio/Video) przez przeglądarkę a następnie tworzy obiekt JavaScript z wynikami, i dodaje klasy do elementu HTML.

Biblioteka może testować obsługę, funkcjonalności HTML5, CSS3 dla przeglądarki gdzie została uruchomiona, dlatego wynik jaki jest zwracany na stronie Snail Project dotyczy przeglądarki przez, którą użytkownik wszedł na stronę projektu.

## Walidacja HTML przez W3C

Zgodnie z duchem Agile, po konsultacjach z Profesorem Leszkiem Borzemskim ustalono następną iterację powstawania projektu o nazwie Snail jaką jest sprawdzenie poprawności kodu HTML testowanej strony z wykorzystaniem Validatora konsorcjum W3C.

W3C to skrót od World Wide Web Consortium, które zajmuje się ustanawianiem standardów sieci WWW takich jak skrypty, style, znaczniki HTML itp.

Walidator, jest dostępny na stronie internetowej konsorcjum jednak możliwe jest także wykorzystanie go w projektach bez implementowania całego systemu testującego. Działa to na zasadzie zadania zapytania AJAX serwera konsorcjum, a następnie odpowiednie sparsowanie wyniku, który jest zwracany w formacie JSON. Po pobraniu adresu testowanej strony z odpowiedniego miejsca na stronie głównej projektu Snail, wysyłane jest odpowiednie żądanie a po jego przetworzeniu wynik jest wyświetlany w odpowiednim miejscu na stronie.

Wynikiem jest lista błędów (o ile takie wystąpią), z krótkim komentarzem do problemu jaki nastąpił, oraz linia i kolumna w kodzie testowanej strony, w której dany błąd się znajduje.

# Narzędzie Snail Project w porównaniu z innymi

Aby móc ocenić wydajność serwisu webowego nie wystarczy być świadomym, jakie techniczne zmiany można wprowadzić w serwisie internetowym. Nie można do oceny każdego projektu podejść w dokładnie taki sam sposób. Na początku próby określenia wydajności właściciel serwisu webowego, lub programista mający za zadania dokonać optymalizacji powinien postawić sobie kilka znaczących pytań: Co właściwie rozumiem poprzez wydajność serwisu ? Co dokładnie chciałbym osiągnąć optymalizując serwis ? Gdzie najwięcej czasu traci użytkownik oczekując na interakcję ze stroną www ? Najważniejsze pytanie powinno brzmieć: Jaki jest cel poprawy wydajności ?

Dla niektórych osób najważniejszym osiągnięciem i czynnikiem optymalizacyjnym jest skrócenie czasu ładowania się strony internetowej. Inni przywiązują największą wagę do poprawy „jakości” kodu, tak aby był semantyczny.

„Semantyka - dział semiotyki zajmujący się badaniem związków, jakie zachodzą między wyrażeniami języka a przedmiotami, do których się one odnoszą” – Słownik Języka Polskiego

Powyższy cytat odnosi się raczej do języka polskiego aczkolwiek kod pisany przez programistów również jest określany poprawnym czy też semantycznym jeśli jego odniesienia, wywołania, funkcje, klasy, metody są napisane prawidłowo, zwracają oczekiwane wyniki i są interpretowane bezbłędnie przez przeglądarki internetowe czy serwery.

Prócz czasu, który jest dominującą skalą oceny wydajności stron www przez użytkowników sieci, można poddać optymalizacji wydajności praktycznie każdą część aplikacji. Dlatego warto sobie odpowiedzieć na zadane wcześniej pytanie „Jaki jest cel poprawy wydajności ?”. Celem może być zwiększona wydajność serwera, przesyłania danych, wykorzystanie nowszych technologii sieciowych, np. zmiana protokołów HTTP, przykładowo zmiana serwera z Apache na Nginx, poprawna validacja strony, SEO itp.

„Gdzie najwięcej czasu traci użytkownik oczekując na interakcję ze stroną www ?” Odpowiedzi może być tak wiele jak celów, które sobie stawia osoba chcąca zoptymalizować daną aplikację. Czas odpowiedzi z serwera może być bardzo długi. Przesyłanie dokumentu HTML może stanowić ponad 20% czasu udzielania odpowiedzi. Skrypty na stronie blokują wczytywanie się pozostałych elementów. Zbyt długie wczytywanie się obrazów itp.

Jak widać wyznaczenie celów działań optymalizacyjnych nie jest jednoznaczne dla każdej osoby, jest wręcz subiektywnym zdaniem, które prowadzi do określenia pewnych działań, które mają osiągnąć zamierzony rezultat. Być może nie jest to łatwa w interpretacji definicja, jednak wszelkie wątpliwości zostaną rozwiane poniżej.

W odpowiedzi na sformułowane wyżej zapytania powstała aplikacja webowa o nazwie „Snail Project – ang. Projekt Ślimak” Jest to aplikacja webowa, która testuje czy w innych aplikacjach, stronach internetowych wdrożone zostały najważniejsze zasady optymalizacyjne. Wynik jak otrzyma użytkownik na pewno pomoże w wyborze celów optymalizacji, skonkretyzuje do czego przyłożyć największą wagę podczas działań polepszających działanie innego serwisu oraz oceni aktualny stan danego serwisu czy aplikacji.

## Test porównujący narzędzia

W sieci internet, która jest pełna różnych ciekawych projektów, istnieją rozwiązania podobne do Projektu Ślimak. Różnią się od siebie szybkością działania, niektóre wykonują w ekspresowym tempie testy a inne dłuższy czas. Nie wszystkie serwisy zwracają ten sam wynik, a co gorsza różnice między nimi są znaczne i bardzo często niezgodne z prawdą.

Aby lepiej poznać różnice między serwisami, które wykonują pomiary wydajnościowe to właśnie one zostały poddane testowi. Serwisy badały różne czynniki dlatego specjalnie przygotowano test tak żeby w miarę możliwości każdy z nich miał szansę przejść go pozytywnie.

Zadaniem każdego serwisu było:

* Pozytywne wykonanie testu
* Zwrócenie wyniku, który zawiera najbliższy prawdziwemu rozmiarowi/wagi strony
* Podanie ilości zapytań do strony

Na potrzeby tego eksperymentu przygotowano specjalną stronę internetową, która będzie służyła jako materiał do badań i późniejszej oceny.   
Strona HTML zawiera w sobie elementy drzewa DOM, których atrybutami są 4 obrazy o wymiarach odpowiednio: 2074x1383 piksele 1954x1301 pikseli, 1954x1301 pikseli, 1815x1031 pikseli o łącznej wadze 1,71 megabajta.  
Sam plik z o rozszerzeniu HTML zajmuje powierzchnię o wadze 28 kilobajtów.  
Szablon HTML wczytuje również bibliotekę Bootstrap oraz wszystkie jej style o łącznej wadze 169 kilobajtów.  
Plik ze skryptami JavaScript dla frameworka Bootstrap 32 kilobajty.  
3 foldery, w celu uporządkowania plików tworzących stronę.  
Przewijany slider ze zdjęciami o łącznej wadze 1.12 megabajta.  
Dodatkowo na stronie zostały celowo wygenerowane błędy aby sprawdzić jak wpływają one na wyniki testów. Błędy to odwołanie do pliku favicon.ico, który nie istnieje. Odwołanie w kodzie HTML do pliku .CSS, który także nie istnieje.

Rozmiar całej strony testowej na dysku twardym to **3,23 megabajta**. (Rozmiar został podany przez explorator systemu Microsoft Windows 8)

Strona została następnie opublikowana w internecie , tak aby można było przeprowadzić eksperyment w łatwy sposób posługując się jedynie docelowym adresem URL.

PageSpeed Insights (PRZYPIS)

Opis: Jest to narzędzie do testowania wydajności stron internetowych, którego autorem jest firma Google. Test jest przeprowadzany i oceniany zarówno pod kątem telefonów komórkowych jak i komputerów stacjonarnych (włączając w to laptopy). Strona została też oceniona w skali 0-100 jednak bez określenia konkretnej jednostki jaką można by było się posługiwać.  
Wyniki są ciekawe, dobrze opisane lecz to wszystko co pokazuje narzędzie Google.

Wynik: Udało się wykonać test, jednak zwracane wyniki sprowadzają się jedynie do sugerowania użytkownikowi jakie poprawki może on zastosować w swoim projekcie aby był wydajniejszy.

GTmetrix (PRZYPIS)

Opis: GTmetrix jest projektem rozwijanym przez wyspecjalizowanych programistów w firmie „Gossamer Threads” w Vancouver. Firma posiada ponad 19 lat doświadczenia w pracy z technologiami webowymi. Serwis ten nie posiada swojego własnego systemu oceny strony, korzysta z PageSpeed Insight oraz narzędzia Yslow do prezentowania wyników użytkownikowi. Rezultaty z obu dostępne są w osobnych zakładkach i trzeba się między nimi przełączać aby dostrzec różnice w ocenie. Trzecią zakładką jest linia czasu, czyli wykres czasu ładowania się kolejnych elementów w zależności od zapytania do serwera o konkretny element. Testy przeprowadzane są przez serwer w Vancouver w Kanadzie oraz przez agenta internetowego Mozilla.

Wynik: Test został wykonany pozytywnie. Waga strony według serwisu to 2.90 megabajta, a ilość zapytań do serwera to dziesięć.

Load Impact (PRZYPIS)

Opis: Serwis zajmujący się odpłatną optymalizacją stron i serwisów internetowych. Bez rejestracji i bez dodatkowych płatności można jednak wykonać test wydajności strony www. Test jest wykonywany przez serwery w różnych miejscach na całym świecie. Rezultat zawiera ilość zapytań do serwera, średni czas ładowania się strony (dodatkowy wykres przedstawia czas ładowania w zależności od serwera). Na stronie znajdują się także wykresy kołowe przedstawiające jaką część czas ładowania się przedstawiają poszczególne grupy elementów na stronie np. skrypty, obrazki, HTML oraz style CSS.

Wynik: Test został wykonany pozytywnie. Wynik nie pokazuje jaki jest ciężar strony. Ilość zapytań do serwera to dziesięć.

WebPageTest (PRZYPIS)

Opis: WebPagetest jest projektem open source, który jest przede wszystkim tworzony i wspierany przez Google w ramach poprawy szybkości obiektów w internetcie. WebPagetest jest narzędziem, który został pierwotnie opracowany przez AOL do stosowania wewnątrz firmy. Platforma jest w stanie aktywnego rozwoju na GitHub a okresowo bywa pakowany do plików ZIP i udostępniany na oficjalnej stronie dla użytkowników. Wersja online jest prowadzona przez Fundację WPO. Firmy wspierają projekt poprzez udostępnianie hostingu, pomoc w ulepszaniu narzędzi testujących itp. Każdy programista może zwrócić się z prośbą do serwisu o udostępnienie specjalnego klucza dla programistów „API key”, za pomocą którego można wykorzystać WebPageTest do implementowania we własnej aplikacji webowej. Wyniki są bardzo wnikliwe, zawierają dużo wykresów i analiz. Jedynym mankamentem jest User eXperience. Zakładki, służące do przechodzenia pomiędzy wynikami i sam sposób przedstawienia rezultatów nie jest estetyczny i nie przyciąga wzroku. Przez to dużo ważnych szczegółów można pominąć.

Wynik. Test został wykonany pozytywnie. Waga strony 2.70 megabajta. Liczba zapytań do serwera wynosi dziesięć.

WebSiteOptimization (PRZYPIS)

Opis: Narzędzie do analizowania stron www, którego celem jest sugerowania użytkownikowi co mógłby poprawić aby jego strona wczytywała się szybciej. Ciekawą informacją jest podanie zależności pomiędzy różną prędkością łącza internetowego a czasem wczytywania się strony. Ponieważ eksperyment zakładał podstawowe testy, nie sprawdzono poprawności innych wyników niż zakładane.

Wynik: Serwis uważa, że test został wykonany prawidłowo, jednak wyniki całkowicie odbiegają od zakładanych. Według serwisu waga pobranych elementów wynosi 1,27 megabajta. Liczba zapytań do serwera 7.

Neustar (PRZYPIS)

Opis: Serwis oferuje test przy pomocy narzędzia, które wykonuje test wydajnościowy innego serwisu internetowego a następnie przedstawia wyniki w ładny graficznie sposób. Test jest wykonywany także przez serwery w różnych miastach np. Dublin, Singapur, Washington oraz San Francisco . Rezultaty jakie zostały wygenerowane najbardziej odbiegają od zakładanych.

Wynik: Test został wykonany. Bez względu na położenie serwera test wygenerował te same wyniki dla wagi strony – 60,3 kilobajty ! Pokazane zostały również tylko obiekty strony, na które serwer odpowiedział statusem 200 i było ich 7.

Pingdom (PRZYPIS)

Opis: Pingdom jest narzędziem, które może być wykorzystywane przez użytkowników w celu ustalenia czasu ładowania strony internetowej poprzez generowanie licznych raportów, takich jak rozmiar strony, pamięci podręcznej przeglądarki, klas wydajności i wiele innych. Pozwala także na śledzenie historii wydajności i przeprowadzenia testów z różnych miejsc. Narzędzie to cechuje się czytelnością, przejrzystością oraz szybkością działania.

Wynik: Test został wykonany pozytywnie. Waga strony to 2.90 megabajta. Liczba zapytań do serwera to dziesięć.

### Snail Project

Opis: Narzędzie stworzone z myślą o pomocy w optymalizacji stron internetowych. Po wykonaniu testu prezentowane są wyniki sugerujące jakie zmiany należy wprowadzić tak aby testowana aplikacja webowa działała szybciej i bardziej wydajnie. Narzędzie podobnie jak WebPageSpeed oraz GTmetrix wykorzystuje silnik testujący Insight stworzony przez Google. Snail Project generuje także wykres typu Waterfall, po to aby można było się dowiedzieć, które z obiektów na stronie działają nieprawidłowo. Widać na nim także ilość zapytań do serwera, zwracany jest rozmiar pobieranej strony a po kliknięciu odpowiedniego przycisku można zobaczyć wszystkie nagłówki odpowiedzi. Dodatkową informacją jaką jest w stanie pokazać opisywany projekt jest, informacja o tym jakie możliwości HTML i CSS wykorzystuje przeglądarka internetowa, z której użytkownik wszedł na stronę Projektu. Snail jako jedyny potrafi wykorzystać walidator konsorcjum W3C aby przetestować stronę pod kątem błędów w kodzie HTML, XHTML, SMIL, MathML itp.

Wynik: Test został wykonany w całości. Waga całej strony internetowej wynosi 3,02 megabajta. Liczba połączeń z serwerem wynosi dziesięć.

Podsumowanie wykonanych testów

Ocena wykonania testu w całości była oceniana w sposób poniekąd binarny, to znaczy albo test się wykonał i pokazał wynik albo nie można było wykonać testów i żaden wynik ani informacja na temat testowanej strony nie został zaprezentowany. Wszystkie narzędzia zwracały mniej lub bardziej wiarygodne wyniki dlatego każdemu należy się plus za poprawną implementację skryptów testujących.

Jakość skryptów pozostaje jednak wiele do życzenia. Najbliższy prawdy okazało się narzędzie „Snail Project”, które działało na niezależnym serwerze. Większość opisanych wcześniej narzędzi korzystało z silnika testującego firmy Google jednak na wyniki mogło mieć wpływ wiele różnych czynników.

Niektóre z narzędzi np. Neustar pokazywały tylko te zapytania, które otrzymały pozytywną odpowiedź z serwera, mimo wszystko nie jest to prawidłowe prezentowanie wyników, ponieważ użytkownik nie wie wtedy jakich dokonać zmian w kodzie aby wszystkie zapytania posiadały status odpowiedzi 200.

## Wnioski

Różne firmy tworzące narzędzia do wykonywania testów i oceny aplikacji webowych, na swój sposób wykorzystują już istniejące skrypty bądź swoje autorskie rozwiązania, jednak wszystkie pokazują rozbieżności w wynikach działa tych narzędzi. Celem eksperymentu nie było sprawdzanie poprawności zaimplementowanych skryptów czy też sposobu działania, lecz szybkie zweryfikowanie czy projekt, który powstał jako część pracy magisterskiej może być z powodzeniem wykorzystywany przez programistów oraz wszystkich ludzi chcących przetestować swoje strony internetowe, aplikacje webowe czy po prostu testować strony już istniejące w sieci celem wyciągnięcia wniosków czy sugerowania ich autorom zmian jakie można wprowadzić. Rezultaty jednoznacznie wskazały, że Snail Project nie odstaje od reszty łatwo dostępnych w internecie narzędzi, na dodatek w tym prostym eksperymencie wypadł najlepiej. Pozostałe badania opisane w poniższej pracy zostały wykonane z wykorzystaniem właśnie tego narzędzia.

<http://developers.google.com/speed/pagespeed/insights>

<http://gtmetrix.com>

<http://loadimpact.com>

<http://www.webpagetest.org/>

<http://analyze.websiteoptimization.com/>

<http://www.neustar.biz/resources/tools/free-website-performance-test>

<http://tools.pingdom.com/fpt/>

# Test 10 najpopularniejszych stron internetowych w Polsce

Poniżej przedstawiono analizę dziesięciu najpopularniejszych stron internetowych w Polsce według rankingu „Alexa”(Przypis).

Alexa Internet jest amerykańską spółką kontrolowaną przez grupę Amazon. Prowadzi ona witrynę o nazwie Alexa, która dostarcza informacje na temat generowanego ruchu do innych stron internetowych. Spółka została założona w 1996 roku, natomiast amazon przejął ją w roku 1999. Wyszukiwarka Alexa zaopatrzona jest w robota internetowego, który indeksuje ponad 4,5 miliarda stron. Posiada także katalog stron WWW.

Test przeprowadzono przy pomocy „Snail Project”. Jest to serwis webowy za pomocą, którego można dokonać analizy innej strony internetowej pod kątem czasów i poprawności ładowania poszczególnych jej elementów. Zmierzyć rozmiar strony, odczytać nagłówki jakie zwraca serwer a także zapoznać się z wynikiem walidacji pod kątem poprawności składni dokumentu HTML analizowanej strony. Dokładny opis działania Projektu oraz jego skryptów został opisany w rozdziale „Snail Project”

Analiza najpopularniejszych stron w Polsce ma na celu sprawdzenie czy najnowsze dostępne technologie są implementowane we współczesnych serwisach webowych.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nr w rankingu** | **Domena** | **Wielkość strony** | **Czas odpowiedzi** | **Serwer** | **Metoda kompresji** | **Liczba zapytań** | **Liczba błędów w kodzie HTML** | **Ocena Snail Project** |
| 1 | Google.pl | 335,921 kb | 4,552 s | gws | gzip | 12 | 32 | 97% |
| 2 | Facebook.com | 1168,407 kb | 3,038 s |  | gzip | 27 | 40 | 89% |
| 3 | Allegro.pl | 1439,641 kb | 101,353 s | nginx | gzip | 92 | 75 | 78% |
| 4 | Google.com | 335,443 kb | 5,537 s | gws | gzip | 11 | 30 | 90% |
| 5 | Youtube.com | 656,053 kb | 53,391 s | gwiseguy/2.0 | gzip | 49 | 1001 | 74% |
| 6 | Onet.pl | 375,461 kb | 13,339 s | nginx | gzip | 16 | 45 | 80% |
| 7 | Wp.pl | 4666,367 kb | 697,283 s | aris | gzip | 253 | 263 | 76% |
| 8 | Gazeta.pl | 2949,219 kb | 196,772 s | apache |  | 165 | 1001 | 59% |
| 9 | Wikipedia.org | 96,060 kb | 3,207 s | apache | gzip | 19 | 20 | 85% |
| 10 | Olx.pl | 692,946 kb | 19,830 s |  | gzip | 39 | 73 | 85% |

Google.pl - Wyszukiwarka stron WWW, grafiki, grup dyskusyjnych, katalog tworzony na bazie ODP i wzbogacany przy użyciu własnej technologii Google. Polska domena wyszukiwarki Google.com

Strona główna Google.pl zawiera jeden obrazek, który jest zmieniany w zależności od okazji (Dzień Kobiet, Dzień Dziecka, Święta Narodowe itd.), standardowo jest to logo firmy Google. Prócz tego na stronie znajduje się pasek wyszukiwania. Jest to bardzo mała i dobrze zoptymalizowana strona, ponieważ służy ona jedynie do wpisania odpowiedniej frazy wyszukiwania.

Biorąc pod uwagę fakt, że firm a z Montain View jest liderem w kwestii wyszukiwarek internetowych oraz tworzonych przez siebie narzędzi, zarówno programistycznych jak i np. przeglądarek internetowych, to nawet główna strona wyszukiwarki nie osiągnęła maksymalnej liczby punktów w teście. 97% jak na tak potężną korporację to mimo wszystko zaskoczenie. Wynik działania skryptu PageSpeed Insights nie zwrócił długiej listy rzeczy jakie należy poprawić aby strona wczytywała się szybciej, jedyne sugestie to zmniejszenie rozmiaru pliku z kodem JavaScript oraz nadanie priorytetów widocznej zawartości strony. Google używa własnego serwera webowego GWS – Google Web Server, a czas odpowiedzi na żądanie przesłania strony wyniósł ~4,5 skeundy. Liczba zapytań do serwera wyniosła 12. Można zmniejszyć liczbę zapytań poprzez zastosowanie CSS Sprites oraz połączenie skryptów w jeden plik lub zagnieżdżenie ich w kodzie strony jeśli nie są zbyt duże. Walidator kodu HTML znalazł 32 błędy między innymi „Line 10, Column 6071: Duplicate ID locale”, większość dotyczy złego wykorzystania styli CSS w kodzie HTML. Jednak tak duża korporacja powinna tworzyć kod zgodny z założeniami World Wide Web Consortium.

Facebook.com – Jest to portal społecznościowy umożlwiający kontakt ludziom z całego świata, wymianę zdjęciami, muzyką oraz filmami wideo.

Allegro.pl – Największy Polski serwis aukcyjny

Google.com - Najpopularniejsza wyszukiwarka stron WWW oraz treści w internecie.

Youtube.com – Serwis internetowy umożliwiający publikowanie własnych filmów oraz prostą obróbkę.

Onet.pl - Wiadomości z kraju i ze świata, baza danych firm i instytucji, ogłoszenia, pasaż handlowy, wyszukiwarka i katalog stron, czat, blog, randki, pogoda, konta e-mail i WWW, kartki pocztowe.

Wp.pl - Serwisy informacyjne, finansowe i rozrywkowe, katalog stron internetowych, poczta e-mail i strony WWW, czat, komunikator, serwisy społecznościowe, gry online, platforma e-commerce.

Gazeta.pl - Informacje, wyszukiwarka, katalog, konta e-mail i forum. Serwisy tematyczne i lokalne w 20 miastach.

Wikipedia.pl – Polskojęzyczna wersja internetowej encyklopedii, którą tworzą sami użytkownicy.

Olx.pl - Darmowe ogłoszenia lokalne w różnych kategoriach.

## Wnioski

<http://www.alexa.com/topsites/countries/PL>

# Reguły wydajności

## Unikanie zbędnych przekierowań

Przekierowania URL są stosowane aby user-agent został przeniesiony w inne miejsce niż nastąpiło poprzednie wywołanie strony. Najczęściej stosowana technika przy skracaniu adresów internetowych oraz przenoszeniu serwisów webowych pod inny adres.

Przekierowanie za pomocą metatagów „Refresh”:

<html><head>  
 <meta http-equiv="Refresh" content="0; url=http://www.pwr.edu.pl/" />  
</head><body>  
 <p>Nowy adres <a href="http://www.wiz.pwr.edu.pl/">link</a>!</p>  
</body></html>

Liczba 0 w atrybucie „content” określa liczbę sekund w czasie których ma być wyświetlana strona.

Drugim sposobem jest przekierowanie z wykorzystaniem JavaScript:

<script language="JavaScript" type="text/javascript">   
location.href="pwr.php";   
</script>

Trzeci sposób to użycie ramek:

<frameset rows="100%">  
 <frame src="http://www.pwr.edu.pl/">  
</frameset>  
<noframes>  
 <body>Please follow <a href="http://www.wiz.pwr.edu.pl/">link</a>!</body>  
</noframes>

Dzięki temu rozwiązaniu strona spod nowego adresu wyświetli się, ale w pasku przeglądarki adres się nie zmieni.

Przekierowanie w PHP:

<?php  
 header('HTTP/1.1 301 Moved Permanently');  
 header('Location: http://www.wiz.pwr.edu.pl/');  
 header('Connection: close');  
 exit();  
?>

Powyższy kod w języku PHP wygeneruje nagłówek:

HTTP/1.1 301 Moved Permanently  
Location: http://www.wiz.pwr.edu.pl/  
Content-Type: text/html  
Content-Length: 174

Inne kody stanu 3xx protokołu HTTP jakie można wykorzystać podczas przekieorwania:

* 300 – Wiele możliwości, przeważnie wykorzystywany przy stronach wielojęzycznych
* 301 – stałe przekierowanie
* 302 – przekierowanie tymczasowe (znaleziony)
* 303 – zobacz inne
* 307 – przekierowanie tymczasowe

Z punktu widzenia SEO przekierowanie 301 jest ważne, ponieważ przekierowuje ono siłę czy też wagę linków ze starej na nową stronę. Dzięki czemu pozycja w Google nie spada po przeniesieniu strony pod nowy adres.

Ostatnim sposobem jest przekierowanie za pomocą .htaccess:

Poniższą linię kodu należy umieścić w pliku .htaccess:

Redirect 301 / http://www.example.com/

## Włączenie kompresji danych

Korzystanie z kompresji przesyłanych danych pomiędzy klientem a serwerem ma na celu bardziej efektywne wykorzystanie dostępnej przepustowości łącza. Włączenie kompresji HTTP zapewnia krótszy czas transmisji danych pomiędzy przeglądarkami internetowymi a serwerami, na których dane się znajdują. Kompresowane mogą być dane zarówno statyczne jak i te generowane dynamicznie przez aplikacje.

Nagłówek Accept-Encoding wskazuje na to, że przeglądarka WWW obsługuje kompresję, natomiast nagłówek Content-Encoding jest stosowany przez serwer (jeśli obsługuje kompresję), który określa skompresowaną odpowiedź.

Najpopularniejszą metodą kompresji danych jest użycie formatu *gzip*. Jest to otwarty format, czyli taki, który nie został objęty żadnymi patentami oraz prawem ograniczającym jego wykorzystywanie. Mniej popularnym i zarazem mniej efektywnym formatem kompresji danych jest *deflate*. Opisane poniżej wyniki testów dziesięciu najpopularniejszych serwisów webowych w Polsce pokazały, że najczęściej stosowaną metodą kompresji przesyłanych danych jest użycie *gzip*.

|  |  |  |  |
| --- | --- | --- | --- |
| Opcja kompresji | Typ danych | Podstawowa konfiguracja | Metoda konfiguracji |
| Format kompresowanych plików | Statyczny | .txt, .htm, and .html | Metabase |
|  | Dynamiczny | .exe, .dll, and .asp | Metabase |
| Format kompresji | Statyczny | gzip oraz deflate | Metabase |
|  | Dynamiczny | gzip oraz deflate | Metabase |
| Poziom kompresji | Statyczny | 10 | Metabase |
|  | Dynamiczny | 0 | Metabase |

## Wykorzystanie pamięci podręcznej przeglądarki

Zapisywanie zasobów statycznych w pamięci podręcznej przeglądarki pomaga oszczędzić czas użytkownikowi, który wielokrotnie odwiedza daną witrynę. Nagłówki zapisywania w pamięci podręcznej powinny dotyczyć wszystkich zasobów statycznych, które można zapisać, a nie tylko ich niewielkich podzbiorów na przykład obrazów. Zasoby, które można zapisać w pamięci podręcznej to pliki JS i CSS, pliki obrazów oraz inne pliki zawierające obiekty binarne, a w tym, multimedia, pliki PDF itp. Sam HTML nie jest statyczny i nie powinien być domyślnie uwzględniany w zapisywaniu w pamięci podręcznej, ale można zmniejszyć jego rozmiar używając narzędzi do minimalizowania rozmiaru plików .html.

Należy włączyć funkcję zapisywania materiałów z serwera w pamięci podręcznej przeglądarki. Czas przechowywania zasobów statycznych w pamięci podręcznej powinien wynosić co najmniej tydzień. Zasoby zewnętrzne, takie jak reklamy czy widżety, powinny być przechowywane w pamięci podręcznej co najmniej jeden dzień.

Dla wszystkich zasobów zapisywanych w pamięci podręcznej Google zaleca następujące ustawienia:

„Ustaw wartość Expires na co najmniej tydzień, a najlepiej na rok w przód. (Preferujemy ustawienie Expires zamiast Cache-Control: max-age, ponieważ jest częściej obsługiwane). Nie ustawiaj wartości dłuższej niż rok w przód, ponieważ będzie to naruszeniem wytycznych RFC.

Jeżeli wiesz dokładnie, kiedy dane zasoby się zmienią, możesz ustawić krótszy okres ważności. Jeżeli jednak wiesz tylko, że „wkrótce mogą się zmienić”, ale bez konkretów, ustaw długi okres ważności i użyj odcisku cyfrowego w URL-u (technika opisana poniżej).”

Nagłówki *Expires* i *Cache-Control: max-age*, określają okres, w którym przeglądarka może używać zasobów zapisanych w pamięci podręcznej bez sprawdzania, czy na serwerze WWW nie znajduje się ich nowsza wersja. Gdy zostaną ustawione, po pobraniu zasobu przeglądarka nie będzie wysyłała żądań GET do upływu terminu ich ważności, osiągnięcia przez nie maksymalnego wieku bądź do czasu opróżnienia pamięci podręcznej przez użytkownika.

Nagłówki *Last-Modifed* i *ETag, o*kreślają sposób decydowania przez przeglądarkę, czy pliki są takie same, na potrzeby zapisania w pamięci podręcznej. W przypadku nagłówka Last-Modified takim wyznacznikiem jest data. W przypadku nagłówka ETag wskazówką może być dowolna wartość jednoznacznie identyfikująca zasób (standardowo stosowane są wersje pliku i identyfikatory numeryczne zawartości). Last-Modified to „słaby” nagłówek, ponieważ przeglądarka określa, czy pobrać dany element z pamięci podręcznej czy nie w oparciu o mechanizmy heurystyczne. Te nagłówki pozwalają przeglądarce skutecznie aktualizować zasoby zapisane w pamięci podręcznej poprzez wysyłanie warunkowych żądań GET, gdy użytkownik wyraźnie ponownie załaduje stronę. Warunkowe żądania GET nie zwracają pełnej odpowiedzi, jeżeli zasób na serwerze się nie zmienił, i z tego względu charakteryzują się krótszym czasem oczekiwania niż pełne żądanie GET.

Ważne jest, by określić jeden z nagłówków Expires i Cache-Control max-age, oraz jeden z Last-Modified i ETag, w odniesieniu do wszystkich zasobów, które można zapisać w pamięci podręcznej. Nie ma potrzeby określać zarówno nagłówka Expires, jak i Cache-Control: max-age, albo i Last-Modified, i ETag.

## Skrócenie czasu odpowiedzi serwera

Czas odpowiedzi serwera to czas wczytywania kodu HTML niezbędnego do rozpoczęcia renderowania strony z serwera, po odjęciu czasu oczekiwania między Google a Twoim serwerem. Mogą wystąpić różnice pomiędzy poszczególnymi cyklami, ale nie powinny one być zbyt duże. W rzeczywistości duża zmienność czasu odpowiedzi serwera może sugerować problem z wydajnością.

Czas odpowiedzi serwer powinien wynosić mniej niż 200 ms. Spowolnienie może być spowodowane przez wiele czynników: mało wydajne algorytmy aplikacji lub zapytania do bazy danych, powolny routing, architekturę, biblioteki, nadmierne wykorzystanie zasobów obliczeniowych procesora bądź pamięci. Poprawiając czas odpowiedzi serwera, należy wziąć wszystkie te czynniki pod uwagę. Po wykonaniu testów oraz interpretacji wyników należy wprowadzać stopniowo różne rozwiązania mające na celu poprawę wydajności. Wykorzystanie różnych popularnych rozwiązań takich jak CMS, MVP, MVC itp. Skutkuje również opisaniem problemów z wydajnością nawet e specyfikacji technicznej danego rozwiązania.

## Minimalizowanie plików CSS

Minimalizowanie rozmiaru plików polega na usuwaniu zbędnych bajtów, na przykład dodatkowych odstępów, komentarzy, znaków łamania wiersza i wcięć. Kompaktowy kod CSS, HTML i JavaScript pomaga skrócić czas pobierania tych plików, ich analizowania oraz wykonywania. Dodatkowo w przypadku CSS i JavaScriptu możliwe jest dalsze zmniejszanie rozmiaru pliku poprzez zmianę nazw zmiennych, stosowanie metodyki DRY (Ang. Don’t Reapeat Yourself).

W celu zmniejszenia rozmiaru kodu CSS można użyć narzędzi takich jak YUI Compressor i cssmin.js, które są polecane przez Google. Istnieje także sporo narzędzi programistycznych, które już w trakcie tworzenia kodu HTML, styli CSS czy skryptów JavaScript dbają i automatyczne modyfikowanie kodu to jak najmniejszego rozmiaru pliku. Są to takie narzędzia jak Koala czy Codekit.

## Minimalizowanie plików HTML

Aby zmniejszyć rozmiaru kodu HTML można wykorzystać rozszerzenie PageSpeed Insights dla przeglądarki Chrome aby wygenerować kompaktową wersję kodu HTML. Należy uruchomić analizę HTML danej strony oraz wybrać odpowiednią regułę zmniejszania kodu.

## Minimalizowanie plików JavaScript

Narzędzie sugerowane przez Google do zmniejszenia ich rozmiaru celem szybszego przesyłania poprzez sieć są np. Closure Compiler, JSMin lub YUI Compressor. Tak jak wspomniano przy opisie minimalizowania rozmiaru plików kaskadowych arkuszy styli, można użyć także narzędzi typu Koala czy Codekit, które „w locie” dbają o odpowiednie formatowanie kodu i zmniejszają jego rozmiar.

## Eliminowanie blokujących skryptów

Zanim przeglądarka wyrenderuje widok strony, w pierwszej kolejności musi dokonać jej analizy. Jeżeli w jej trakcie napotka blokujący zewnętrzny skrypt, musi go zinterpretować zanim dokona dalszej analizy kodu samej strony. Każdy taki skrypt oznacza dodatkowe połączenia z serwerem w celu pobrania kodu JavaScript i przesłanie wyniku jego zawartości do klienta WWW, co znacznie pogarsza User eXperience.

Krótkie skrypty JavaScript wymagane do wyrenderowania części strony widocznej na ekranie powinny zostać umieszczone pomiędzy znacznikami HTML, natomiast pobranie kodu JavaScript niezbędnego do realizacji dodatkowych funkcji takich jak ruchome elementy, pokazy slajdów, itp. powinno odbywać się dopiero po dostarczeniu treści.

Krótkie skrypty zewnętrzne możesz włączyć bezpośrednio do dokumentu HTML. Wbudowanie niewielkich plików pozwoli przeglądarkom kontynuować renderowanie strony.

Aby zapobiec blokowaniu wczytywania stron przez kod JavaScript, przy wczytywaniu zasobów zaleca się użycie w kodzie HTML atrybutu *async*.

Na przykład:

<script async src="my.js">

Rozwiązanie to jednak nie sprawdza się jeżeli w kodzie JavaScript używana jest komenda document.write. Należy wtedy przepisać kod w taki sposób aby nie wykorzystywać document.write do wpisywania tekstu do dokumentu HTML. Przy asynchronicznym wczytywaniu skryptów, należy z zwrócić uwagę, by aplikacja uwzględniała wczytywanie ich w odpowiedniej kolejności, jeżeli jest to kod JavaScript zależący od siebie nawzajem.

Wykorzystanie różnych bibliotek JavaScript, na przykład jQuery, umożliwia wzbogacenie strony o dodatkowe funkcje takiej jak animacje, przejścia, zmianę styli CSS i inne efekty. Większość takich skryptów można zastosować już po wyrenderowaniu części widocznej na ekranie dla użytkownika. Ponieważ na doświadczenia użytkownika, oraz czas ładowania się strony ma to duży wpływ, warto stosować tą radę w większości projektów aplikacji webowych.

## Optymalizacja obrazów

Stosowanie jak najmniejszych rozmiarów obrazów, ma na celu skrócenie czasu oczekiwania na załadowanie przez użytkownika. Właściwe sformatowanie i skompresowanie obrazów może zaoszczędzić wiele bajtów danych. Skraca to czas wczytywania strony przy wolnym połączeniu, i jest szczególnie cenione przez użytkowników „mobilnych”, ze względu na transfer danych i opóźnienia sieci.

Warto przeprowadzić zarówno podstawową, jak i zaawansowaną optymalizację wszystkich obrazów. Podstawowa optymalizacja to wycinanie zbędnych obszarów, maksymalna akceptowalna redukcja głębi kolorów, usuwanie komentarzy do obrazów i korzystanie z odpowiedniego formatu plików. Te czynności można wykonać przy pomocy dowolnego programu do edycji obrazów, takiego jak GIMP czy Adobe Photoshop.

W sieci jest dostępnych jest wiele narzędzi umożliwiających bezstratną (tzn. bez szkody dla jakości obrazów) kompresję plików JPEG i PNG. Są to zarówno narzędzie darmowe (jpegtran, optipng, czy Gimp) oraz komercyjne (Adobe Photoshop – potężne narzędzie służące zarówno do edycji jak i tworzenia grafiki, nie tylko na potrzeby internetu)

* Pliki PNG są niemal zawsze lepsze od plików GIF, jednak niektóre starsze wersje przeglądarek mogą tylko częściowo obsługiwać pliki PNG charakteryzujące się obsługą kanału alfa.
* Pliki GIF stają się coraz mniej popularne w sieci, ponieważ bezcelowe jest ich wykorzystywanie do grafik przedstawiających statyczne obrazy, ze względu na niską jakość oraz większy rozmiar w porównaniu z formatem PNG.
* Formatu JPG najlepiej używać w przypadku wszystkich plików, które nie wymagają przezroczystości czyli przedstawiające zdjęcia, portrety, krajobrazy.

## Priorytetyzacja widocznej treści

W momencie gdy wymagana ilość danych przekracza początkowy rozmiar okna przeciążenia, konieczna jest dodatkowa wymiana informacji między serwerem a przeglądarką użytkownika co znacząco wpływa na czas odpowiedzi. W przypadku sieci o długim czasie oczekiwania, na przykład sieci komórkowych takich jak 3G, 4G, może to powodować poważne opóźnienia wczytywania stron.

Krok jaki należy wykonać aby strony wczytywały się szybciej, to ograniczenie rozmiaru plików zawierających znaczniki HTML, obrazy, style CSS czy kod JavaScript, które są wymagane do wyrenderowania widocznej dla użytkowników części strony.

Już na etapie projektowania serwisu webowego można zastosować pewne reguły, które będą miały wpływ na jego wydajność:

* Należy zaplanować strukturę strony w taki sposób aby jej najważniejsza część, widoczna na ekranie, wczytywała się w pierwszej kolejności. Jeżeli HTML wczytuje zewnętrzne skrypty np. jQuery przed główną zawartością, najważniejsze jest aby zmień kolejność w taki sposób, aby główna zawartość wczytywała się jako pierwsza.
* Należy zmniejszyć ilość danych wykorzystywanych przez zasoby serwisu.
* Po przeprojektowaniu witryny pod kątem poprawnego wyświetlania na różnych urządzeniach i wczytywania najważniejszej zawartości na początku zmniejsz ilość danych niezbędnych do wyrenderowania stron, posługując się następującymi metodami:
* Należy stosować style CSS zamiast obrazów tam, gdzie jest to możliwe np. jako tło dla serwisu, czy przycisków itp.
* Należy włączyć kompresję danych używając np. gzip
* Wczytywanie skryptów należy umieścić na końcu kodu HTML
* Należy zoptymalizować fonty używane w całym serwisie webowym

## Optymalizacja fontów

Aby nie narazić serwisu webowego na opóźnienia związane ze zbyt długim wczytywaniem fontów należy odpowiednio je zoptymalizować. Dzięki temu doświadczenia użytkowników będą lepsze i pozwolą odczuć pozytywne skojarzenia z daną stroną. Odpowiednia użycie czcionek pozwala zwiększyć wygodę użytkowników: rozszerzyć rozpoznawalność marki, polepszyć czytelność, funkcjonalność i sprawność wyszukiwania, zapewniając przy tym możliwość zmiany skali i pracę przy wielu różnych rozdzielczościach i formatach ekranu, od wielocalowych monitorów po wyświetlacze urządzeń przenośnych.

Przede wszystkim nie należy używać zbyt wielu krojów pisma na jednej czy kilku stronach serwisu. Pozwoli to na skrócenie czasu oczekiwania użytkowników na wyświetlenie danej treści.

Należy używać podzbiorów fontów. W przypadku wielu czcionek można wydzielić podzbiory, rozdzielić je na wiele zakresów Unicode i dostarczyć tylko glify konkretnie wymagane na danej stronie – pozwala to ograniczyć rozmiar pliku i zwiększyć prędkość pobierania zasobów. Przy określaniu podzbiorów należy mieć na uwadze optymalizację czcionek pod kątem ponownego użycia. Jeżeli na kilku podstronach serwisu wykorzystywane są różne znaki alfabetu łacińskiego czy też cyrylicy, dobrze dobrana czcionka będzie wszystkie te znaki obsługiwać. Czcionki należy przesyłać w formatach dostosowanych do każdej z przeglądarek: każda czcionka powinna być dostarczana w formatach WOFF2, WOFF, EOT i TTF. Ważne jest aby mieć pewność, że do formatów EOT i TTF stosowana będzie kompresja GZIP, ponieważ formaty te nie są domyślnie kompresowane, co może skutkować wydłużonym czasem oczekiwanie na wyrenderowanie strony.

Jedną z najważniejszych czynności podczas optymalizacji fontów jest określenie zasad ponownej walidacji i optymalnego buforowania, poniważ czcionki to zasoby statyczne. Są rzadko aktualizowane dlatego warto zmienić ustawienia serwera tak aby zapewniał długi okres ważności max-age i wysyłał tokeny walidacji, co umożliwi efektywne ponowne używanie czcionek na różnych stronach w witrynie.

@font-face jest regułą CSS umożliwiającą określenie lokalizacji konkretnego zasobu czcionki, jej stylu i kodów Unicode, dla których powinien obowiązywać. Połączenie takich deklaracji reguł @font-face można wykorzystać do utworzenia rodziny czcionek, dzięki której przeglądarka może określić, które czcionki trzeba pobrać i zastosować na bieżącej stronie.

W każdej deklaracji reguły @font-face określa się nazwę rodziny czcionek, dzięki której może ona pełnić rolę logicznej grupy wielu deklaracji, własności czcionki, takie jak styl, grubość, rozciągnięcie i deskryptor źródłowy zawierający uporządkowaną według ważności listę lokalizacji tego zasobu czcionki.

@font-face {  
font-family: 'PWR Font';  
font-style: normal;  
font-weight: 450;  
src: local('PWR Font'),  
url('/fonts/pwr.woff2') format('woff2'),   
url('/fonts/pwr.woff') format('woff'),  
url('/fonts/pwr.ttf') format('ttf'),  
url('/fonts/pwr.eot') format('eot');  
}

## WebP

Firma Google wydała własny kodek obrazów, który miał być pewnego rodzaju rewolucją. Webp, to nowy format obrazów rastrowych, zarówno animowanych jak i tych statycznych. Według producenta rozmiar plików w formacie .png zostanie zmniejszony aż o 26%, plików z rozszerzeniem.jpeg w przedziale od 25 do 34%, a pozostałych formatów obrazu około 22%. Należy także dodać, że kompresja obrazków jest stratna. Bez obaw można natomiast obrazki bez tła czyli z tak zwanym kanałem alfa, bowiem nie zostanie ono zastąpione białym tłem tak jak ma to miejsce podczas konwersji plików z formatu .gif na .jpeg.

Pierwszą przeglądarką, która wprowadziła obsługę powyższego formatu obrazków była oczywiście Google Chrome, od wersji 9.0.569 , która jest tworzona przez tego samego producenta. Kolejną przeglądarką wspierającą kodek była Opera 11.10, który został wykorzystany w funkcji „Opera Turbo”. Sposób działania polegał na tym, że serwer „w locie” kompresował obrazki i przesyłał je do klienta, a ostatecznie do odbiorcy końcowego.

Nie wszyscy Twórcy popularnych przeglądarek internetowych zaadoptowali pomysł przedsiębiorstwa z Mountain View. Jak można przeczytać na forum webhosting.pl (PRZYPIS), Twórcy Firefox oraz Internet Explorer nie wsparli obsługi nowego formatu plików w swoich programach. Jako argumenty takiego zachowania wymienili wady nowo powstałego rozwiązania:

* (jeszcze we wczesnych wersjach) brak kanału alfa
* brak wsparcia profili ICC
* brak obsługi metadanych EXIF (np. brak informacji o parametrach z aparatu użytkownika)

20 maja 2011 roku Jeff Muizelaar (jeden z pracowników Mozilli) stwierdził, że nowy format jest „wypieczony tylko w połowie”.  
  
Wpis na forum pochodzi z roku 2011, aby się przekonać czy po dość długim okresie adaptacji .webp, wykonano prostu test użyteczności kodeka.

W pierwszej kolejności pobrano odpowiednie kodeki (PRZYPIS), i rozpakowano je na dysku twardym. Następnie w celach testowych wybrano logo Politechniki Wrocławskiej (PRZYPIS), oraz przygotowano szablon HTML zawierający tylko linie kodu wczytujące dwa obrazki. Pierwszy oryginalne logo w formacie .gif, a drugi zakodowany w formacie .webp.  
Po zapoznaniu się z dokumentacją kodeków na stronie producenta (PRZYPIS), użyto skryptu o nazwie „gif2webp” aby dokonać konwersji typu pliku przy użyciu komendy:  
   
*„gif2webp –o poziom-en.webp poziom-en.gif”*  
Następnie pliki (oryginalny i ten powstały w wyniku działania skryptu) zostały przeniesione do odpowiedniego folderu aby mogły został wczytane przez stronę HTML.

Pierwsze co rzuciło się w oczy to waga plików:

* Oryginalny ma rozmiar 603 x 100 pixeli, wagę 6,58 KB oraz 8 bitową głębię kolorów.
* Informacje o pliku w nowym formacie goole’a, rzeczywiście nie podają właściwie żadnych informacji o rozmiarze czy palecie barw. Jedyna informacja jaką można uzyskać to rozmiar pliku. Wynosi on 4,41 KB

Test możliwości otwarcia strony internetowej w prawidłowo wczytanym logiem Politechniki Wrocławskiej potwierdził jedynie słowa sceptyków sprzed lat.   
Jedynie przeglądarki Chrome w wersji (39.0.2171) oraz Opera (ver. 26) były w stanie pokazać oczekiwany efekt. Po wywołaniu strony w tych programach ukazały się dwa identyczne pod względem wymiarów, oraz w subiektywnej ocenie tej samej jakości pliki graficzne.  
Firefox w wersji 34.0.5, Internet Explorer 11 oraz Safari 5.1.7 nie przeszły testu prawidłowo. Miejsce, w którym powinno znajdować się wczytane logo zostało zastąpione pustym miejscem lub ikoną symbolizującą błędnie wczytany obraz.

Format .webp powstał z zamysłem przyspieszenia ładowania się obrazków szybciej, poprzez ich kompresję. Kodek jak i format nie zdobyły jednak dużej popularności i uznania w oczach programistów. Google zaprezentowało swój kodek 30 września 2010 roku. Po ponad czterech latach jego obecności jedyne szerokie zastosowanie jakie udało się znaleźć to obrazki w wirtualnym sklepie GooglePlay, na urządzeniach mobilnych działających pod systemem Android, jest to nie zauważalna dla odbiorcy zmiana, raczej wymuszona przez Google.  
  
https://developers.google.com/speed/webp/

http://webhosting.pl/Mozilla.odrzucila.obsluge.formatu.WebP

https://developers.google.com/speed/webp/docs/using

http://www.logotyp.pwr.edu.pl/Default.aspx?page=ZnakPWr

<https://developers.google.com/speed/webp/download>

## CDN

Content Delivery Network to sieć wielu komputerów używanych w celu zwiększenia wydajności dostarczania serwisów internetowych do odbiorców oraz w celu zmniejszenia obciążenia serwera dostawcy treści. W sieci CDN zawiera się wiele serwerów „dostawców” rozmieszczonych w wielu miejscach na całym świecie. Ich zadaniem jest pobieranie bądź aktualizacja swoich danych na podstawie głównego serwera webowego, tak aby każdy serwer w danej sieci zawierał te same informacje. W momencie odwiedzenia przez użytkownika danego serwisu, widzi on kopię danych dostarczoną przez najbliższy serwer, co skraca czas odpowiedzi na zapytanie.

Content Delivery Network działa na podobnej zasadzie jak proxy między użytkownikiem a danym serwerem. W zależności od hostingu na, którym programista zamierza umieścić swoją aplikacje można zaoszczędzić ponad 80% transferu. Dzieje się tak dlatego, że niezależnie od kraju w którym znajduje się docelowy serwer z aplikacją, przy użyciu CDN użytkownik skorzysta z najszybszego i najbliższego dla jego lokalizacji serwera usługi CDN.

Content Delivery Network to bardzo szybko rozwijająca się technologia. Początkowo zakładała tylko dostarczanie statycznych treści dla użytkowników. Jej rozwój okazał się tak dynamiczny, że obecnie jest wykorzystywana również do rozpraszania transmisji strumieniowych oraz coraz częściej aplikacji webowych. Istnieje wiele firm czy operatorów świadczących usługi z wykorzystaniem powyższego rozwiązania. Wybór odpowiedniego dostawcy usługi pozwala znacząco zmniejszyć koszty utrzymania własnej infrastruktury sieci komputerów.

Fundamentalną korzyścią korzystania z CDN jest zmniejszenie obciążenia serwera głównego z aplikacją internetową. Użytkownicy otrzymują dane z serwera, który CDN określi jako najbliższy do komputera użytkownika. Drugą korzyścią jest oczywiście wspomniany wcześniej, zdecydowanie krótszy czas odpowiedzi na zapytania do serwera, co skutkuje szybszym dostarczeniem zawartości do komputera użytkownika przez co witryna czy serwis jest atrakcyjniejszy dla odbiorcy.

Prócz czasu dostępu nie ma nic bardziej ważnego niż sama dostępność serwisu. Najczęściej usługa CDN chroni aplikację przed atakami typu DoS oraz DDoS. Ten drugi to rodzaj ataku na serwer lub system komputerowy, w którym uniemożliwia się działanie takiego systemu poprzez zapełnienie jego wszystkich wolnych zasobów. Taki atak jest zazwyczaj przeprowadzany przez wiele komputerów tzw. „zombie”. Takie komputery same łączą się z daną usługą sieciową, bez obecności użytkownika. Bezpieczeństwo aplikacji powinno być zapewnione jeszcze przed jej implementacją. Jaka jest użyteczność, nawet najbardziej wydajnej strony www, którą w każdej chwili może usunąć każdy bardziej doświadczony użytkownik sieci lub haker ?

Duże firmy, które posiadają własne sieci serwerów na całym świecie udostępniają usługę CDN. Jest to zdecydowanie tańsze rozwiązanie jeśli chodzi o cenę wykorzystania takiego rozwiązania w porównaniu do kosztów samodzielnego instalowania serwerów w różnych miejscach świata (czy tylko Polski). Zewnętrzne firmy oferują także dodatkowe usługi oferujące niezawodność na przykład poprzez kopię danych lub redundancję i odtwarzanie ich w przypadku problemów z danym serwerem. Pozostawiając konfigurację serwera, który będzie trzymaj kopię danej strony, często też można korzystać z najnowszych technologii nawet jeśli nie została ona wdrożona na serwerze „głównym”. Dla przykładu firma CloudFlare dba o to aby świadczone przez nich usługi korzystały z protokołu SPDY, obsługi priorytetów dla zapytań, czy optymalizacji dla sesji. W chwili obecnej nie są to jeszcze standardowe rozwiązania stosowane na szeroką skalę, raczej są dopiero wdrażane w nowszych projektach aplikacji webowych. Firma ta zapewnia też ważną ze względów wydajnościowych funkcję nazwaną przez nich „Always Online”. Zostanie aktywowana w momencie gdy „główny serwer”, na którym docelowo została wdrożona aplikacja webowa przestanie działać. Kopia zostanie odtworzona z pamięci cache serwerów tak aby była zawsze dostępna dla użytkowników.

Prócz użytkowników poszukujących informacji, internet przemierzają roboty sieciowe takie jak indeksujący witryny Googlebot. Szybszy czas odpowiedzi jest przez niego postrzegany jako zaleta co ma wpływ na późniejsze umieszczenie witryny w hierarchii rezultatów wyszukanych przez Google.

Programista lub właściciel aplikacji webowej może sam wybrać czy chce skorzystać z usług darmowych czy komercyjnych usług CDN.

Przykłady komercyjnych dostawców usługi:

* Akamai Technologies (PRZYPIS)
* Amazon CloudFront
* Mirror Image
* CacheFly
* CDNetworks
* ChinaCache

Wśród darmowych dostawców CDN są takie firmy jak:

* BootstrapCDN (PRZYPIS)
* CloudFlare (PRZYPIS)
* Coral Content Distribution Network
* Incapsula (darmowa wersja w reklamami Incapsula)
* Google - PageSpeed Service

<http://www.akamai.com/html/solutions/network-operator-solutions.html>

<http://www.bootstrapcdn.com/>

https://www.cloudflare.com/features-cdn

# Strona WWW czy aplikacja ?

Współczesne strony internetowe są coraz bardziej skomplikowane a do ich tworzenia używane są coraz to nowocześniejsze narzędzia. Aby otrzymać jak najlepszą wydajność aplikacji webowych należy dokonać optymalizacji na wielu płaszczyznach. Każda z nich ma swoje własne wymagania systemowe lecz każdą można optymalizować. Protokoły transportowe, komunikacyjne, bazy danych, „User eXperience”, czy po prostu front-end i back-end aplikacji. Nie ma jednej „złotej zasady”, którą można stosować z „zamkniętymi oczami” do każdej aplikacji czy strony internetowej. Najważniejsze jest określenie celu jaki zamierza się osiągnąć podczas optymalizacji oraz skąd się bierze problem z wydajnością. „Snail Project” w pewnym stopniu pomaga wytknąć błędy, lub problemy z daną aplikacją.

Najczęstsze problemy w komunikacji na linii klient – serwer to ograniczenia protokołu TCP, które w dużym stopniu wpływają na działanie protokołu HTTP. Ograniczenia oraz funkcjonalność protokołu HTTP. Opóźnienia pasma oraz jego wydajność w komunikacji WWW. Ograniczenia oraz optymalizacja przeglądarek. Największe jednak znaczenie mają aktualne trendy w tworzeniu stron. Przecież to użytkownik wymaga aby strona była w odpowiedni sposób sformatowana, a także swoją funkcjonalnością, wydajnością oraz interakcją zachęcała do niejednokrotnego skorzystania z jej zasobów.

Obecne wymagania użytkowników co do interfejsów internetowych aplikacji nie zostaną zaspokojone poprzez prosty szablon HTML z kolorowymi hiperlinkami. Obecnie przeciętny użytkownik internetu nie jest w stanie prawidłowo ocenić czy aktualnie znajduje się „na stronie www” czy też już korzysta z „aplikacji webowej”. Dla programisty jest to istotna różnica, szczególnie kiedy chce wprowadzić poprawki wydajnościowe w danym projekcie.

Przodkiem strony internetowej jest zwykły dokument hipertekstowy. Stanowił on podstawę działalności protokołu HTTP 0.9. Składał się z odpowiednio sformatowanego tekstu przy użyciu znaczników oraz zawartych w nim odnośników. Nawiązanie sesji przy użyciu tego protokołu zakładało tylko wysłanie dokumentu z hipertekstem do odbiorcy.

Strona internetowa to poniekąd rozszerzenie dokumentu hipertekstowego. Zmiana natomiast polega na tym, że grupa robocza, która pracowała nad protokołem HTTP rozszerzyli możliwości hipertekstu poprzez obsługę mediów. Multimedialne treści takie jak obrazy, dźwięk czy wideo na stronach internetowych stały się atrakcyjne dla odwiedzających. Bogate treści były jednak statyczne co doprowadziło do ewoluowania statycznych stron w aplikacje WWW. Zasobne treści były również powodem dla, którego sam protokół HTTP przeszedł nie małą przemianę. Wielokrotne połączenia, szybsze przesyłanie dokumentów, skrócenie czasów ładowania się strony, caching, oraz metadane HTTP, podtrzymywanie połączenia to najważniejsze z cech nowszych wersji protokołu transportowego.

Obecne strony internetowe, w których zaimplementowana interakcja z użytkownikiem jest już podstawową cechą noszą raczej nazwę aplikacji webowych. Różne skrypty mające na celu zwiększenie atrakcyjności, animacje obrazów, nietypowe wyświetlanie informacji pobranych z baz danych czyni współczesny internet pełnym mniej lub bardziej zaawansowanych projektów stron www.

W związku z ciągłym rozwojem dokumentów hipertekstowych, czas ładowania się strony nie jest jedynym wyznacznikiem wydajności. Ważne są także inne cechy jak ilość jednocześnie ładujących się fragmentów, sekwencje wgrywanych elementów, kolejność wykonania skryptów, stabilność i szybkość serwera, bazy danych, połączenia sieciowego, pierwszej reakcji użytkownika itp.

# Doświadczenia użytkownika w pływ na wydajność

„User experience, UX (ang. doświadczenie użytkownika) – całość wrażeń, jakich doświadcza użytkownik podczas korzystania z produktu interaktywnego. Pojęcie to używane jest najczęściej w odniesieniu do oprogramowania, serwisów internetowych lub urządzeń elektronicznych.”(PRZYPIS).

Tak naprawdę bardzo ciężko jest określić kiedy aplikacja jest szybka i wydajna, ponieważ w zależności od odczuć danej osoby, określanie tych wartości może być różne, dla każdej osoby. Jest to ocena na tyle subiektywna, że nie jest nawet oceniana z punktu widzenia wydajności. Ocenie nie może podlegać dobór tonacji kolorów, rozmieszczenie elementów na stronie internetowej czy animacje jakie zostały a niej zaimplementowane.

Oczywiście inaczej to wygląda z punktu widzenia programisty, który koduje wszystkie elementy. To on musi zadbać o to aby kod czy skrypty mają dostarczyć użytkownikowi „wrażeń”, były odpowiednio dobrze walidowane oraz interpretowane przez klientów stron www.

Steve Souders jest pracownikiem firmy Google. Zajmuje się wydajnością stron WWW i inicjatywami open source. Jest twórcą YSlow - rozszerzenia dla dodatku Firebug, służącego do analizy wydajności i optymalizacji stron internetowych. Opisał jak można niezależnie od używanego urządzenia, telefon komórkowy, laptop, komputer stacjonarny czy tablet, określić (nie jest to oficjalna metoda) zależność czasu opóźnienia od wrażeń użytkownika.

|  |  |
| --- | --- |
| **Opóźnienie [ms]** | **Wrażenia użytkownika** |
| 0-100 | Natychmiastowa reakcja użytkownika |
| 100 – 300 | Odczucie niewielkiego opóźnienia |
| 300 – 1000 | Komputer pracuje normalnie |
| Powyżej 1000 | Możliwe rozproszenie uwagi |
| Powyżej 10 000 | Przerwanie czynności przez użytkownika |

Jak wynika z tabeli, należy jak dostarczyć treść użytkownikowi w czasie poniżej jednej sekundy, aby użytkownik nie rozproszył swojej uwagi. Należy mieć na uwadze także opóźnienia wynikające z połączenia poprzez TCP. Dobrze przygotowana aplikacja jest w stanie, w czasie poniżej jednej sekundy wyświetlić jakąś treść, lub taki element strony aby skupić uwagę użytkownika, podczas gdy pozostałe elementy zostaną wczytane w następnej kolejności, lecz jak najkrótszym czasie.

Doświadczenia użytkownika to temat, który porusza różne dziedziny nauk takie jak psychologia, filozofia, geometria, czy sztuka. Kolejność wymienionych dziedzin nie ma tutaj znaczenia, jednak każda z nich może mieć wpływ podczas projektowania aplikacji na jej wygląd. Na przykład geometria i sztuka, na rozkład elementów strony i wykorzystane barwy. Wszystkie koncepcje aplikacji internetowych można przetestować z technicznego punktu widzenia wykorzystując do tego celu „Snail Project”. Zastosowanie się do wskazówek i wyników testu pomoże nawet najbardziej atrakcyjną dla użytkownika treść przekształcić w wydajną i dobrze zoptymalizowaną, nie obciążającą sieci treść przesyłaną do klienta poprzez protokoły TCP/IP czy HTTP.

http://pl.wikipedia.org/wiki/User\_experience

# Wydajność przeglądarek internetowych

W trakcie projektowania aplikacji internetowych nie trzeba się martwić o poszczególne gniazda TCP czy UDP, ponieważ zarządza nimi przeglądarka internetowa. Dodatkowo stos sieciowy zarządza ograniczeniem liczby połączeń, formatowaniem żądań, izolowaniem poszczególnych aplikacji między sobą oraz obsługę serwera proxy . Ale najważniejsze jest aby mimo wszystko nie zapominać o tym, że wydajność protokołów TCP, http czy sieci mobilnych w dużej mierze wpływa na jakoś działania tworzonej aplikacji.

Nowoczesne przeglądarki internetowe są tworzone z myślą o szybkim i bezbłędnym dostarczaniu treści poprzez internet. Zawierają w sobie ogromną liczbę warstw optymalizacji obsługi stron www. Popularne programy do odwiedzania stron w internecie zawierają moduły zarządzające procesami, dbające o bezpieczeństwo w sieci, optymalizujące pamięć podręczną, renderowanie grafiki, maszyny wirtualne Javy, obsługę audio-wideo i wiele innych. To prawie systemy operacyjne dla sieci.

Ogólna wydajność przeglądarki i dbałość o jej najbardziej aktualną wersję ma wielki wpływ na ogólną wydajność i szybkość uruchamiania się aplikacji internetowych. Dlatego w projekcie, który przygotowano(PRZYPIS-odwołanie do projektu) można sprawdzić nie tylko obsługę aktualnych rozwiązań technologicznych takich jak HTML 5, CSS 3. Dzięki zaimplementowanej w projekcie bibliotece „modernizr”, można określić poprawność takich opcji jak caching strony, obsługę video dla HTML 5 etc. Nawet najlepsza optymalizacja serwisu nie będzie miała wpływu na wydajność jeśli użytkownik końcowy w dalszym ciągu posługuje się przeglądarką Internet Explorer 6 zaimplementowanej w Windows XP. W przeciwieństwie do IE, Google Chrome „uczy się”(PRZYPIS) działać coraz szybciej w trakcie jego obsługi przez użytkownika. Przeglądarka „zapamiętuje” kolejność przeglądania stron, najczęstsze wyszukiwania, sposób zadawania zapytań do wyszukiwarki. Wszystko ma na celu optymalizację działania poprzez uprzedzanie działań użytkownika tak aby kolejne „zadania” wykonywały się szybciej, niż miałoby to nastąpić bez tej optymalizacji. Chrome wstępnie rozpoznaje nazwy za pomocą DNS, wcześniej nawiązuje połączenie TCP, aby otrzymać wstępny wygląd strony.

Stos obsługi sieci we współczesnych przeglądarkach jest nie tylko menedżerem gniazd komunikacji, to wręcz platforma do ich obsługi, która posiada własne kryteria optymalizacji, interfejsy API, wtyczki, dodatki i usługi. Każdy producent tych aplikacji uważa swoje parametry ustawień za najbardziej wydajne, to właśnie dlatego do wyboru jest pokaźna gama takich programów do korzystania z internetu jak, Chrome, Opera. Internet Explorer, Safari, Firefox czy Chromium.

Cykl życia pojedynczych gniazd komunikacyjnych nie jest zarządzany przez aplikacje internetowe uruchomione przez przeglądarkę. Dzięki temu krytyczne mechanizmy optymalizacji wydajności są zautomatyzowane. Np. ponowne wykorzystanie gniazd, nadawanie priorytetów żądaniom, negocjacje protokołów, ograniczenie liczby powiązań, późne wiązania itd. Założeniem przeglądarek jest oddzielenie zarządzania cyklem życia żądań od zarządzania gniazdami komunikacyjnymi.

Gniazda są grupowane w pule według źródeł, każda z nich pula określa własne zasady bezpieczeństwa i wymagania dotyczące liczby połączeń. Żądanie, które zostanie zawieszone, jest ustawiane w kolejne i zostanie mu nadany priorytet. Następnie każde żądanie zostanie powiązane z gniazdem w puli. Gniazdo może zostać użyte ponownie przez wiele żądań, jeśli sam serwer nie zakończy połączenia. Umieszczanie gniazd w puli i ponowne użycie połączenia TCP odbywa się automatycznie, dzięki czemu zyskuje wydajność aplikacji. Przeglądarka może optymalizować pulę i zamykać nieużywane gniazda. Obsługiwać żądania w kolejce według priorytetów. Aktywnie otwierać gniazda i uprzedzać żądania. Minimalizować opóźnienie i zwiększać prędkość transmisji poprzez ponownie wykorzystanie gniazd komunikacyjnych.

Usługi takie jak zarządzanie gniazdami komunikacyjnymi, połączeniami, pamięcią podręczną, bezpieczeństwo, obsługa żądań i odpowiedzi są realizowane przez odpowiednie typy połączeń. HTTP, WebRTC, XMLHttpRequest, czy Server-Sent Events wykorzystują odpowiednio wszystkie bądź tylko niektóre z wymienionych usług. W internecie jest tyle różnych stron czy aplikacji ilu jest ich twórców, dlatego też nie ma jednego najlepszego protokołu czy interfejsu API w pełni wydajnie działającego dla wszystkich typów połączeń czy usług. W zależności od potrzeb aplikacji może ona korzystać z jednej lub kilku różnych protokołów transportowych, opóźnień komunikatów, w różny sposób wykorzystywać pamięć podręczną przeglądarki. Poniższa tabela przedstawia porównanie funkcjonalności protokołów XHR, SSE oraz WebSocket. Wybór odpowiedniego może stanowić „drugie życie” dla wolno działających i niewydajnych aplikacji, które są źle odbierane przez użytkowników.

|  |  |  |  |
| --- | --- | --- | --- |
|  | XMLHttpRequest | Server-Sent Events | WebSocket |
| Strumieniowanie żądań | tak | nie | tak |
| Strumieniowanie odpowiedzi | ograniczone | tak | tak |
| Mechanizm ramkowania | tak | nie (base64) | ramkowanie binarne |
| Binarna transmisja danych | tak | tak | tak |
| Kompresja | tak | tak | ograniczona |
| Aplikacyjny protokół transportowy | HTTP | HTTP | WebSocket |
| Sieciowy protokół transportowy | TCP | TCP | TCP |

Tabela 123 Wysokopoziomowe funkcjonalności protokołów

Przeglądarki internetowe po wywołaniu adresu jakiejś strony, sprawdzają czy zasób nie jest przechowywany w ich pamięci podręcznej, jeżeli tak to zwracana jest zawartość z kopii zapisanej lokalnie. W przeciwnym przypadku, wysyłane jest żądanie do sieci, a po otrzymaniu odpowiedzi jest ona umieszczana w cache-u przeglądarki, tak aby była dostępna w przypadku kolejnego żądania i dostęp do niej niemal natychmiastowy. Przeglądarki internetowe automatycznie wykonują pewne czynności:

* Przetwarzają zlecenia dostępu do każdego zasobu
* Zarządzają wielkością pamięci podręcznej i usuwają jej zasoby
* Nadają ważność nieaktualnym zasobom

Aby przeglądarka po stronie klienta mogła prawidłowo zarządzać pamięcią, należy zadbać o to aby serwer zwracał prawidłowe zlecenia dostępu do pamięci podręcznej. Odpowiednie nagłówki takie jak ETag, Last-Modified oraz Cache-Control odpowiednio przygotowane pomagają wydajnie obsługiwać stronę www.

https://www.igvita.com/posa/high-performance-networking-in-google-chrome/