Notatki z kursu Programowanie dla WWW

Małgorzata Dymek

2019/20, semestr zimowy

1 Wstęp.

- **Frontend**
  - HTML - podstawowy język znaczników dla opisu stron www
  - CSS (Cascading Style Sheets) - Kaskadowe arkusze stylów - Język do definiowania takich atrybutów, które wpływają na wygląd strony jak: kolor, czcionka, układ (wyrównanie, centrowanie, marginesy), itp.
  - JavaScript - Jeden z najpopularniejszych języków ostatnich lat. De-facto standard dla programowania po stronie klienta. Umożliwia tworzenie interaktywnych stron.
  - jQuery - jedna z najczęściej stosowanych bibliotek JavaScript do manipulowania elementami strony WWW.
  - Pozostałe - to bardziej rozbudowane biblioteki lub frameworki do budowy aplikacji WWW.
- **Backend**
  - Java - jeden z najpopularnieszych i najbardziej uniwersalnych języków programowania.
  - Serwlety - niewielkie aplikacje (najczęściej w Javie) rozszerzające mozliewości swerwera WWW. Odpowiadają na żadania, które serwer otrzymuje od klienta (np. przeglądarki). Najczęściej generują dynamiczne strony WWW lub zwracają dane np. pobrane z bazy danych.
  - Java Server Pages (JSP) - technologia generowania stron WWW. W specyficznym języku znaczników będącym mieszanką HTML-a i Javy tworzone są szablony stron, które są następnie tłumaczone na serwlety w Javie. Przykład pliku JSP
  - Spring MVC - framework będący częścią Spring Framework pozwalający ułatwiający tworzenie aplikacji webowych bazujących na serwletach w modelu Model-View-Controller
  - Node.js - wieloplatformmowe środowisko uruchomieniowe (runtime) JavaScript działające po stronie serwera. Umożliwia między innymi łatwe tworzenie serwerów WWW w Javascripcie.
  - Django - framework w Pythonie do szybiego tworzenia aplikacji internetowych bazujących na wzorcu model-template-view
  - Ruby on Rails - framework w języku Ruby do tworzenia aplikacji internetowych
  - ASP.NET - framework od Microsoftu do tworzenia dynamicznych aplikacji internetowych przy użyciu m.in. języków C# i VB.NET
  - Go - język stworzony w firmie Google. Doskonale nadaje się do tworzenia aplikacji internetowych (po stronie serwera).
  - Swift - język stworzony przez Apple dla iOS itp. Na jego bazie istinieją frameworki umożliające tworzenie aplikacji webowch

Czym jest aplikacja www?	
Klasyczne serwisy WWW	**Single Page Applications**
• Ładowane są całe strony. • Strony są generowane przez skrypty CGI, serwlety Java, itp. • Zmiana zawartości strony wymaga załadowania całej strony • Cała logika aplikacji (lub jej większa część) działa po stronie serwera (backend)	• Cała strona(y) jest ładowana jednorazowo na początku • Cała logika aplikacji działa po stronie klienta (przeglądarki). Zaimplementowana w JavasScript. • Różne fragmenty aplikacjji są wyświetlane i/lub tworzone dynamicznie w czasie działania aplikacji • Potrzebne dane (w postaci JSON, XML, itp) są pobierane z serwera przy pomocy wywołań AJAX

2 URL i HTTP.

Internet i WWW to nie to samo	
Internet	**WWW**
The Internet is a global system of interconnected computer networks that interchange data by packet switching using the standardized Internet Protocol Suite (TCP/IP).	The World Wide Web (WWW, or simply Web) is an information space in which the items of interest, referred to as resources, are identified by global identifiers called Uniform Resource Identifiers (URI)

Jak działa WWW?

1. Użytkownik wpisuje w polu adresu przeglądarki adres (URL)
2. Przeglądarka analizuje wpisany adres sprawdzając protokół (HTTP) oraz nazwę hosta.
3. Przeglądarka łączy się z serwerem DNS i uzyskuje adres IP hosta.
4. Przeglądarka nawiązuje połączenie TCP/IP serwerem WWW działającym pod danym adresem na porcie 80 (domyślny port dla protokołu http).
5. Przeglądarka wysyła w ramach protokołu HTTP żądanie (request) pobrania zasobu: GET /home.html HTTP/1.1
6. Serwer lokalizuje żądany zasób statyczny lub generuje go dynamicznie, a następnie zwraca w postaci odpowiedzi (response) HTTP zawierającej treść strony w postaci HTML.
7. Przeglądarka wyświetla zawartość HTML w postaci graficznej.

2.1 URI

URL (Uniform Resource Locator) jest szczególnym przypadkiem bardziej ogólnego pojęcia: URI.

URI (Uniform Resource Identifier) - ciąg znaków jednoznacznie identyfikaujący dowolny abstrakcyjny lub fizyczny zasób.

- **Uniform** - pozwala w jednorodny sposób odowywać się do różnych rodzajów zasobów niezależnie od sposobu dostępu do nich. Definiuje ogólną postać identyfikatora pozwalając na rozszerzanie i uzupenianie definicji dla różnych typów zasobów. (http, ftp, isbn)
- **Resource** - Dowolny abstrakcyjny lub fizyczny zasób. Cokolwiek, co może być identyfikowane przez URI. Np. elektroniczne dokumenty, obrazy, książka (numer isbn), serwis sieciowy.

- **Identifier** - Identyfikator pozwalający na odróżnienie jednego zasobu od drugiego. Nie musi zawierać informacji o sposobie uzyskania dostępu do tego zasobu.

URI pozwala na identyfikację różnych zasobów przy pomocy rozszrzalnego zbioru schmatów nazewnictwa (naming schemes). URI definiuje jedynie ogólne ramy tych schematów, podczas gdy szczegóły składni są definiowane w specyfikacjach poszczególnych schematów. Takimi schematami są np. http, file, mailto. Istnieją pewne wspólne zasady składni niezależne od konkretnego schematu.

Listę oficjalnie zarejestrowanych schematów można znaleść na stronie IANA (Internet Assigned Numbers Authority).

Każde URI może być zaklasyfikowane jako lokalizator (Locator) URL, nazwa (Name) URN lub oba na raz.

O lokalizatorze (URL) mówimy, jeśli URI określa również lokalizację zasobu (np. adres sieciowy). URN, to identyfikator nadający zasobowi "nazwę" - identyfikator, który pozostaje globalnie unikalny i trwały nawet jeśli sam zasób przestanie istnieć lub stanie się niedostępny. Dla takich identyfikatorów często używany jest schemat urn, ale nie jest to bezwzględnie wymagane.

Sam schemat nie przesądza o tym, czy URI jest URN. Np. przestrzenie nazw w XML są URN-ami, chociaz na ogół korzystają ze schematu http.

Znaki zarezerwowane są kodowane, np spacja = %20 (lub +).

2.2 Protokół HTTP

- HTTP - Hypertext Transfer Protocol.
- Z RFC2616: "The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers."
- Protokół do wymiany danych w sieci Internet. HTTP jest niesymetrycznym protokołem request – response klient-serwer. Klient HTTP wysyła żądanie (request), na co serwer odsyła mu odpowiedź (response).
- Protokół HTTP jest bezstanowy, co znaczy, że serwer nie jest w stanie bez dodatkowej informacji, zawartej w żądaniu, stwierdzić że poszczególne żądania należą do danej konwersacji między klientem a serwerem. Serwer nie wie, co działo się w poprzednich żądaniach.
- HTTP pozwala klientowi i serwerowi na negocjację typów danych i representacji przesyłanych infomracji.

HTTP request	**HTTP response**
• linię żądania (request line), • nagłówki (headers) - przesyłane w postaci par nazwa:wartość, oddzielone przecinkami. • (pusta linia) • opcjonalnie treść żądania (request body)	• linia statusu (status line), • nagłówki (headers), • (pusta linia) • opcjonalnie treść odpowiedzi (response body)

METODY HTTP	
Method	**Description**
GET	The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.
HEAD	Same as GET, but it transfers the status line and the header section only. (Ponieważ wśród nagłówków znajduje się nagłówek last-modified można wykorzystać metodę HEAD do sprawdzania, czy lokalna cachowana kopia strony jest aktualna.)
POST	A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.
PUT	Replaces all the current representations of the target resource with the uploaded content.
DELETE	Removes all the current representations of the target resource given by URI.
CONNECT	Establishes a tunnel to the server identified by a given URI.
OPTIONS	Describe the communication options for the target resource.
TRACE	Ask the server to return a diagnostic trace of the actions it takes.

PORÓWNANIE GET I POST		
	GET	**POST**
Przycisk Wstecz/ Odświeżenie strony	Niegroźne	Dane zostaną powtórnie przesłane. (przeglądarka pwoinna ostrzec użytkownika, że dane będą powtórnie przesłane)
Zakładki	Można dodać do zakładek	Nie można dodac do zakładek
Cachowanie	Można cachować	Nie można cachować
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded lub multipart/form-data. Dla danych binarnych używamy multipart encoding
Historia	Parametry pozostają w historii przeglądarki	Parametry nie są przechowywane w historii przeglądarki
Ograniczenia na długość przekazywanych danych	Metoda GET dodaje parametry do URL-a, a wiele przeglądartek ogranicza długość URLa (np. do 2048 znaków)	Brak ograniczeń
Ograniczenia na przesyłane dane	Tylko znaki ASCII	Brak ograniczeń. Dozwolne są także dane binarne
Bezpieczeństwo	GET mniej bezpieczny niż POST bo dane są przekazywan jako część URLa. Nigdy nie należy przesyłać haseł i innych wrażliwych danych przy pomocy GET	POST jest (tylko odrobinę) bezpieczniejszy niż GET, ponieważ dane nie są przechowywane w historii przeglądarki ani w logach serwera.

STATUSY ODPOWIEDZI		
1xx	**Informational**	**Request received, server is continuing the process.**
100	Continue	The server received the request and is in the process of giving the response.
2xx	**Success**	**The request was successfully received, understood, accepted and serviced.**
200	OK	The request is fulfilled.
3xx	**Redirection**	**Further action must be taken in order to complete the request.**
301	Move Permanently	The resource requested for has been permanently moved to a new location. The URL of the new location is given in the response header called Location. The client should issue a new request to the new location. Application should update all references to this new location.
302	Found & Redirect (or Move Temporarily)	Same as 301, but the new location is temporarily in nature. The client should issue a new request, but applications need not update the references.
304	Not Modified	In response to the If-Modified-Since conditional GET request, the server notifies that the resource requested has not been modified.
4xx	**Client Error**	**The request contains bad syntax or cannot be understood.**
400	Bad Request	Server could not interpret or understand the request, probably syntax error in the request message.
401	Authentication Required	The requested resource is protected, and require client's credential (username/password). The client should re-submit the request with his credential (username/password).
403	Forbidden	Server refuses to supply the resource, regardless of identity of client.
404	Not Found	The requested resource cannot be found in the server.
405	Method Not Allowed	The request method used, e.g., POST, PUT, DELETE, is a valid method. However, the server does not allow that method for the resource requested.
408	Request Timeout	
414	Request URI too Large	
5xx	**Server Error**	**The server failed to fulfill an apparently valid request.**
500	Internal Server Error	Server is confused, often caused by an error in the server-side program responding to the request.
501	Method Not Implemented	The request method used is invalid (could be caused by a typing error, e.g., "GET" misspell as "Get").
502	Bad Gateway	Proxy or Gateway indicates that it receives a bad response from the upstream server.
503	Service Unavailable	Server cannot response due to overloading or maintenance. The client can try again later.
504	Gateway Timeout	Proxy or Gateway indicates that it receives a timeout from an upstream server.

2.2.1 Metody bezpieczne i dempotentne.

Theorem 2.1 ***Metoda bezpieczna*** *(safe) - metoda, która nie generuje efektów ubocznych. W praktyce są to metody nie modyfikujące danych zasobów na serwerze*

Theorem 2.2 ***Metoda idempotentna*** *- to metoda, której wieokrotne wykonanie daje takie same efekty, jak wykonanie jest jeden raz. Klient może tak metodę wykonać wielokrotnie i oczekuje, że jej efekt w stosunku do zasobów serwera będzie taka sam jak w przypadku wykonania jednorazowego, chociaż zwrócona odpowiedźmoże się różnić.*

Metoda	**Bezpieczna?**	**Idempotentna?**
GET	Tak	Tak
HEAD	Tak	Tak
OPTIONS	Tak	Tak
TRACE	Tak	Tak
PUT	No	Tak
DELETE	Nie	Tak
POST	Nie	Nie

Prawidowa implementacja metod leży po stronie twórcy oprogramownia. Twórca/projektant backendu musi zadbać aby jego obsługa metod była zgodna z powyższymi wymaganiami.

2.2.2 Nagłówki.

Ogólne	
Connection	umożliwia tworzenie trwałych połaczeń (od wersji HTTP/1.1). Wysłanie Connection: close wymusza zakończenie połączenia.
Nagłówki żądania	
Cookie	przekazuje do zrwotnie do serwera HTTP Cookie (ciasteczko) przesłane wcześniej z serwera w nagłówku Set-Cookie
User-Agent	identyfikuje rodzaj aplikacji klienckiej (przeglądarkę), system operacyjny, klienta, itp.
Host	nazwa domenowa serwera. W wersji HTTP/1.1 (i 2.0) nagłowek obowiązkowy. Umożliwia Virtual hosting
Accept-Language	język(i) akceptowane/preferowane przez klienta
Nagłówki odpowiedzi	
Content-Type	typ MIME zwartości (np. text/html, text/css, image/png)
Content-Length	rozmiar ciała (body) odpowiedzi. (W przypadku metody HEAD, rozmiar ciała, które byłoby wysłane dla metody GET).
Set-Cookie	HTTP Cookie umożliwia między innymi zapamiętywanie stanu (sesji).

3 HTML.

3.1 HTTP - HyperText Markup Language.

XHTML jest znacznie bardziej restrykcyjny (zgodny z XML).

- Znaczniki i atrybuty w muszą być pisane małymi literami
- Wszystkie elementy muszą być poprawnie zakończone (np.
)
- Wszystkie wartości atrybutów muszą być w cudzysłowiach lub apostraofach
- Każdy atrybut musi mieć wartość (niedozwolone jest np. <div hidden>; musimy pisać)
- Bardziej surowe zasady zagnieżdżania (np. <a> nie może być zagnieżdżone w <a>, <form> nie może zawierać <form> i inne).
- Znaki <> & ' " muszą być reprezentowane przez encje (< > & ' ") nawet jeśli są w wartościach atrybutów.
- W XHTML aby identyfikować fragmenty dokumentu opisane przez znaczniki <a> <form> <img> <map> (i kilka innych) nie należy używać atrybutu name (zamiast tego id)
- Dokument musi zaczynać się od deklaracji <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"> i musi zawierać elementy <html>, <head>, <title> i <body>.

HTML 5.

- Nie wymaga aby kod HTML był dobrze uformowanym XMLem.
- Dużo wymogów dotyczących poprawności dokumentów i ich dokładniejsze sprawdzanie.
- Możliwość zastosowania narzędzi pochodzących z xml takich jak XPath czy XSLT.
- Wsparcie przestrzeni nazw co umożliwia użycie np. MathML czy SVG.
- Zintegrowane wsparcie dla multimediow niewymagające użycia wtyczek
- Brak wersji (jak Strict, Frames, Transitional).
- Dobrze tolerowany przez stare przeglądarki.
- Nowe elementy, np.: article, aside, audio, footer, header.
- Nowe typy elementów formularzy: dates and times, email, url, search, number, range, tel, color.
- Nowe atrybuty: charset (dla elementu meta), async (dla elementu script)
- Globalne atrybuty (które mogą być zasotsowane do dowolnego elementu): id, tabindex, hidden, data-* (custom data attributes)
- Usunięto niektóre zdeprecjonowane elementy: acronym, applet, basefont, big, center, dir, font, frame, frameset, isindex, noframes, strike, tt
- Zmiana znaczenia/użycia niektórych istniejących atrybutów i elementów (np <i> i <small> - zmiana znaczenia na bardziej semantyczne)

Nowe API

- **GeoLocation API** - dostęp do lokalizacji użytkownika
- **Web Storage API** - wygodniejsze składowanie infomracji po stronie klienta
- **Web Socket API** - full-duplex komunikacja z serwerem.
- **Web Worker API** - umożliwia uruachamianie zdaań w Javascripcie w innym wątku niż interfejs przeglądarki.

- **Drag & Drop API** - wsparcie dla "drag and drop"

Content type

- Dla zawartości w standarcie HTML: text/html
- Dla zawartości w standarcie XHTML: application/xhtml+xml (ewentualnie application/xml lub text/xml)

Interpretacja przez przeglądarkę. W zależności od Content type oraz <!doctype> przeglądarka interpretuje zawartość w rózny sposób.

Content-Type	text/html	text/html	application/xhtml+xml
doctype	brak/stary	poprawny	bez znaczenia
Wynik	HTML(quirks mode)	HTML 4.01 / HTML 5	XHTML

W przypadku trybu html nie zauważy np. zamknięcia elementu typu .

4 CSS.

- **Kaskadowe arkusze stylów** (ang. Cascading Style Sheets) - język opisu sposoby prezentacji (wyglądu) dokumentu HTML.
- CSS umożliwia rozdzielenie struktury i zawartości dokumentu od sposobu jego prezentacji

- Określanie czcionek (rodziny, koloru, wielkości, pugrubienia, itp)
- Określenie marginesów, odstępów między liniami
- Określenie rozmiarów i położenia poszczególnych elementów strony
- Określanie obramowań i wypełnień
- Zmiana spsobu prezentacji standardowych elementów (np. punktory w listach)

4.0.1 Selektory

- Selektor decyduje do którego elementu strony są stosowane właściwości
- Kryterium wyboru może być m.in. typ elementu, atrybuty (w szczególności id i class) oraz położenie elementu względem innych elementów.
- Pseduo-elementy dają dostęp m.in. do fragmentów elementów (::first-letter)
- Można także używać pseudo-klas dających dostęp do informacji nie zawartych bezpośrednio w dkoumencie (:hover, :visited)

Content-Type	text/html	text/html	application/xhtml+xml
doctype	brak/stary	poprawny	bez znaczenia
Wynik	HTML(quirks mode)	HTML 4.01 / HTML 5	XHTML

W przypadku trybu html nie zauważy np. zamknięcia elementu typu .

Np: .class, #id, $*$, element, (element,element), (element element), [attribute=value].

Priorytety

Priority	CSS source type	Description
1	Importance	The !important annotation overwrites the previous priority types
2	Inline	A style applied to an HTML element via HTML 'style' attribute
3	Media Type	A property definition applies to all media types, unless a media specific CSS is defined
4	User defined	Most browsers have the accessibility feature: a user defined CSS
5	Selector specificity	A specific contextual selector (#heading p) overwrites generic definition
6	Rule order	Last rule declaration has a higher priority
7	Parent inheritance	If a property is not specified, it is inherited from a parent element
8	CSS property definition in HTML document	CSS rule or CSS inline style overwrites a default browser value
9	Browser default	The lowest priority: browser default value is determined by W3C initial value specifications

Style związane z fontami (tekstem) sa dziedziczone z elementu rodzica. Większość pozostałych stylów nie jest dziedziczona.

Specyficzność.

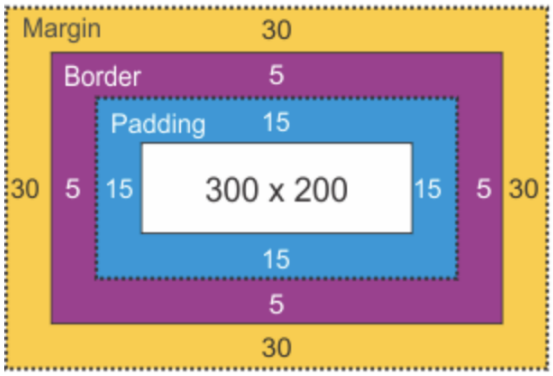
Specyficzność określa "szczegółowość" selektora:

- prosty selektor (selektor elementu) ma specyficzność (p) = 0, 0 , 0, 1
- selektor klasy ma specyficzność (.head-button)= 0, 0 , 1, 0
- selektor oparty na id ma specyficzność (#save-button) = 0, 1 , 0, 0
- style inline (umieszczony w elemencie) ma specyficzność = 1, 0 , 0, 0

Selector "body #content .data img:hover" ma specyficzność równą 0,1,2,2.

4.1 Box Model

- Każdy element HTML można traktować jako prostokąt. Zawartość elementu (content) jest otoczona wypełnieniem (padding), obramowaniem (border) oraz marginesem (margin).
- W zależności od wartości box-sizing (content-box lub border-box) obliczany jest całkowity rozmiar elementu.



4.2 Dodatki

- **CSS Transforms** - pozwala na transformacje takie jak przesunięcie, skalowanie, obracanie, itp.
  - translate()

  - rotate()
  - scale()
  - skewX()
  - skewY()
  - matrix()
- **CSS Transitions** - pozwala na płynną zmianę wartości.
- **SS Flexbox Layout**
  - Elastyczny menadżer układu elementów na stronie.
  - Elementy mogą być typu flex-container i/lub flex-item
  - Dla **flex-container** możemy określić kierunek i sposób ułożenia elementów flex-item (pionowo/poziomo, prawo/lewo, zawijanie wierszy, wykorzystanie wolnej przestrzeni)
  - Dla **flex-item** możemy okreslić zdolnośc elementu do powiększania/zmniejszania, wpełania pustej przestrzeni (także w stosunku do sąsiednich elementów), bazowy rozmiar, kolejność wyświetlania, itp.
  - Przy pomocy Flexboxa stosunkowo łatwo wykonac podział strony na kilka kolumn tej samej wysokości.

5 Javascript

- Javascript jest obecna na ok. 95% stron internetowych
- Jest rozwinięciem języka stworzonego przez Brendana Eicha z Netscape (najpierw pod nazwą Mocha, później LiveScript i ostatecznie JavaScript).
- W 1996 pojawiła się pierwsza przeglądarka (Netscape Navigator 2.0) zawierającaobsługę Javascriptu
- W tym samym roku Microsoft wypuścił Internet Explorera 3.0 z obsługą własnej wersji tego języka pod nazwą JScript.
- W 1997 pojawił się standard tego języka pod nazwą ECMAScript* Obecnie Javascript, JScript (Microsoft), ActionScript (Macromedia, a potem Adobe wykorzystany we Flashu i Flexie) są po prostu implementacjami tego standardu (zawierającymi często różne rozszerzenia).
- Istnieją kompilatory/translatory (transpiler) tumaczące kod np z ES6 na ES5.
- Istnieją rozszerzenia Jascriptu (ECMAScriptu)
  - TypeScript - uzywany w Angular i narzęDdziach Microsoft
  - CoffeeScript
  - JSX (ReactJs)

IIFE (Immediately Invoked Function Expression)

1. Funkcja anonimowa zwraca obiekt z funkcjami operującymi na zmiennej 'prywatnej' x
2. Funkcja anonimowa jest natychmiast wywoływana (z pustą listą argumentów) i wartość zwracana z funkcji jest przypisywana do counter.

W taki między innymi sposób są realizowane w JavaScripcie moduły (Module Pattern)

strict mode

- Włączamy dodając ‘"use strict"; na początku skryptu lub funkcji

- Pojawił się w ECMAScript 5 i jest wspierany w IE from version 10. Firefox from version 4. Chrome from version 13. Safari from version 5.1.+ Opera from version 12.
- Nawet jeśli nie jest wspierany, można go użyć - nie powoduje to błędów.
- W tym trybie wykrywane i zabronione są niektóre potencjalnie niebezpieczne konstrukcje, takie jak np. użycie zmiennej lub obiektu bez wczesniejszej deklaracji.

this

- Słowo kluczowe this ma trochę inne znaczenie w Javascript niż w C++, czy Javie.
- Jego wartość różni się w zależności od kontekstu:
  - Na zewnątrz funkcji this oznacza pewiwn obiekt globalny (w przeglądarkach - window
  - W 'normalnych' funkcjach w trybie strict - obiekt globalny window, w trybie sloppy - undefined
  - W konstruktorach i metodach odnosi się do tworzonego/modyfikowanego obiektu.

5.1 DOM i BOM

Kod w Javascripcie oddziałuje na przeglądarkę i dokumenty w niej wyświetlane przy pomocy interfejsów progratyscznych DOM i BOM.

DOM - Document Object Model	**BOM** - Browser Object Model
• Odwzorowuje dokument HTML lub XML w postaci drzewa obiektów. • Wykrywanie elementów: document.getElementById, ByTagName, ByClassName, ByClassName, querySelectorAll(css_selector). • Zmiana elementów: element.attribute=, setAttribute(attribute,value), element.innerHTML=, element.style.property=. • Dodawanie/usuwanie elementów: document.createElement, appendChild, replaceChild, removeChild, write(text). • Event: onlick, onmouseover, onmouseout • Najlepiej używać metod: addEventListener, removeEventListener, które pozwalają na przypisanie kilku porcedur obsługi zdarzeń.	• Opisuje metody i interfejsy umożliwiające interakcję z przeglądarką • Brak standardu - przeglądarki mogą mieć różniące implementacje • Główny obiekt - window • Ważniejsze obiekty i metody: – window.location - URL strony. Umożliwia załadowanie strony z adresu, odświeżenie, itp. – window.history - Historia przeglądanych stron – window.screen - Dane dotyczące ekranu – console - konsola diagnostyczna przeglądarki – alert, confirm, itd. - Wyświetlanie okien dialogowych

6 React.

- Biblioteka w JavaScript do tworzenia interfejsu użytkownika
- Stworzona przez Facebooka
- Jedna z najpopularniejszych bibliotek JavaScript
- Nie jest pełnym frameworkiem (jak np. Angular)

- Jednokierunkowy przepływ danych (single-way data flow): "properties flow down; actions flow up"
- Virtual DOM (wirtualny obiektowy model dokumentu) - React tworzy w pamięci swój własny DOM w którym monitoruje wszystkie zmiany i uaktualnia widok w przeglądarce tylko o faktyczne zmiany.

- JSX (JavaScript Syntax eXtension) - pozwala na wprowadzanie składni HTML-o podobnej bezpośrednio w kodzie JavaScript
- React Native - biblioteki dla IOS-a, Androida

Props vs. State

- **props** - dane przekazane do komponentu (jak parametry wywołania funkcji)
- **state** - stan wewnętrzny komponentu. zmieniany **asynchronicznie** przez setState().

7 Node.js

- Środowisko uruchomieniowe Javascript oparte na silniku Google Chrome V8
- Stworzony w 2009 roku przez Ryana Dahla w celu wyeliminowania problemów ze skalowalnością tradycyjnych serwerów HTTP
- Open source: licencja MIT
- Działa m.in. Na Windows, Linuksie i MacOS
- Wykorzystywany przede wszystkim do tworzenia serwerów HTTP, ale jest też używany do tworzenia aplikacji desktopowych
- Jest bazą wielu środowisk deweloperskich (deweloper stack) m.in. dla Angular-a, React-a

Architektura

- Jeden wątek z pętlą zdarzeń (event loop)
- Nieblokujące I/O bazujące na zdarzeniach i wywołaniach zwrotnych (callback)
- Bazuje na bibliotece libuv (napisanej w języku C)

Zalety	**Wady**
• Duża szybkość, potrafi obsłużyć znacznie więcej żądań niż tradycyjny serwer (jak np. Apache) • Brak narzutów związanych z tworzeniem i utrzymywaniem wątków • Wysoka skalowalność związana z użyciem nieblokującego API i pojedynczej pętli zadarzeń • Rozbudowany ekosystem narzędzi i bibliotek • Możliwość pisania frontendu i backendu w tym samym języku • Jest częścią środowisk tworzenia aplikacji webowych: – MEAN - MongoDB, Express, Angular, Node – MERN - MongoDB, Express, React, Node	• Niezbyt dobrze sprawdza się w zastosowaniach wymagających dużej mocy obliczeniowej (CPU intensive) • Stosunkowo duża zmienność całego środowiska i pakietów • Callback hell – problem z utrzymaniem przejrzystości kodu z związku z potrzebą stosowania dużej liczby wywołań zwrotnych.

7.0.1 NPM - Node Package Manager.

- Podstawowe narzędzie do zarządzania pakietami w Node.js
- Umożliwia instalację i aktualizację pakietów z uwzględnieniem zależności między nimi
- Pakiety można instalować globalnie lub lokalnie

Plik package.json

- Zawiera ważne informacje na temat tworzonego pakietu/aplikacji
- Zawiera listę pakietów od których zależy pakiet/aplikacja wraz z wymaganymi numerami wersji
- Pozwalana na automatyczne pobranie wszystkich niezbędnych zależności

Synchroniczny lub asynchroniczny model programowania.

Request i **response** są strumieniami, które generują zdarzenia 'error', 'data', 'end'.

8 ExpressJS

Różne standardy (czyli brak standardu):

- CommonJS - serwery, Node
- AMD (Asynchronous Module Defintion) - przeglądarki
- ES2015 - wbudowane w język, ale jeszcze niezbyt rozpowszechnione

Moduły umożliwiają

- podział kodu na (bardziej) niezależne części
- łatwiejsze współdzielenie i powtórne wykorzystanie kodu
- ukrycie implmentacji i udostępnienie jedynie wybranych funkcji i zmiennych (interfejs)
- unikanie konfliktów nazw

8.1 CommonJS

- Plik modułu eksportuje funkcje i obiekty poprzez przypisując je do obiektu module.exports.
- Plik wykorzystujący moduł ładuje go przy pomocy funkcji require().

8.2 Express.JS

Cechy

- Najpopularniejszy framework w Node
- Unopinionated - nie narzuca rozwiązań i struktury. Daje możliwość wyboru.
- Podpinanie obsługi pod różne metody HTTP i adresy (routing)
- Integracja z różnymi silnikami szablonów stron WWW
- Możliwość rozszerzania funkcjonalności za pomocą middleware: (obsługa sesji, cookies, logowania użytkowników, itp.)
- Funkcjonalność Expressa jest oparta na pojęciu middleware.

Routing

- Definiowanie obsługi dla poszczególnych metod i ścieżek: app.get(), app.post(), app.put(), app.delete() itp.
- Definiowanie obsługi dla dowolnej metody: app.all().

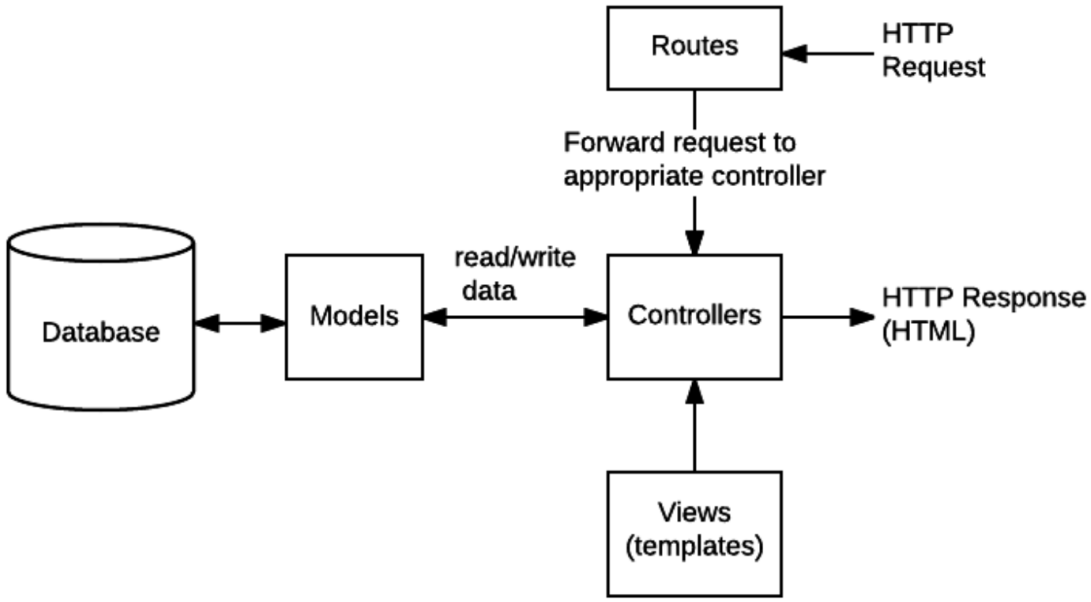
express.Router daje możliwość podzielenia obsługi serwisu/aplikacji webowej na części w zależności od prefiksu użytej ścieżki. Można dodatkowo delegować tę obsługe dododatkowego modułu. Routingi zdefinowane w express_router.js będą dostępne w pod adresem /express_router/ oraz /express_router/about.

Ścieżki używane przy routingu mogą być definiowane jako napisy z wzorcami, np:

- app.get('/ab?cd', callback) - '/abcd' lub '/acd'

- app.get('/ab+cd', callback) - '/abcd', '/abbcd', '/abbbcd', ...
- app.get('/ab*cd', callback) - '/abcd', '/abXcd', , '/abDowolnyTekstcd'
- app.get(/.*info$/, callback)

8.2.1 Model MVC (Model-View-Controller)



8.3 Szablony

Express może współpracować z różnymi systemami szablonów HTML. Między innymi Pug, Mustache, EJS.

Pug

- Zawieranie się elementów jest odzwierciedlone przy pomocy wcięć.
- Każda (prawie) linia zaczyna się od nazwy znacznika HTML
- Jeśli tag kończy się znakiem =, to dalsza częśc jest traktowana jako wyrażenie Javascript, którego wartość jest wstawiana jako zawartość generowanego elementu.
- Można używać instrukcji warunkowych i pętli
- Można również dziedziczyć/rozszrzać szablony: polecenia extend i block.
- Wygląda jak bardzo uproszczony html, bez nawiasów ostrych <>.

9 Zarządzanie sesjami w aplikacjach webowych

- Protokół HTTP jest bezstanowy (chociaż sam serwer oczywiście jakiś stan posiada).
- Każde żądanie przychodzące z serwera jest przetwarzane niezależnie od ewentualnych wcześniejszych żądań
- W wielu sytuacjach konieczne jest powiązanie kolejnych żądań przychodzących od tego samego klienta

- Ciąg takich żądań przychodzących od jednego klienta, to sesja
- Grupowanie żadań przychodzących od jednego klienta (jedna sesja)musi byc realizowane przez programistę (przy ewentualnym wsparciu wykorzystywanych frameworków i bibliotek)

Theorem 9.1 *Sesja jest nowa jeśli została utworzona po stronie serwera, ale klient do niej jeszcze nie przystąpił*

Identyfikator sesji

- Z sesją wiążemy pewien identyfikator generowany przez serwer (identyfikator sesji
- Identyfikator sesji jest generowany przez serwer przy pierwszym żądaniu przychodzącym od klienta
- Wszystkie następne żądania przychodzące od tego samego klienta zawierają ten identyfikator

Techniki przekazywania identyfikatora sesji

- Przepisywanie URL (URL rewriting)
- Ukryte pola w formularzach HTML
- Ciasteczka (cookies)

Przepisywanie URL (URL Rewriting)

- Identyfikator sesji jest dołączany do URL-a każdego żądania.
- Serwer na zwracanej stronie dołącza do wszystkich URL-i dodatkowy parametr z identyfikatorem sesji.
- Wywołanie przez klienta takiego URL-a pozwala serwerowi na pobranie identyfikatora sesji z przekazanego parametru

Zalety	Wady
1. Działa z każdą przeglądarką niezależnie od ustawień użytkownika	1. Trzeba przepisać każdy używany link 2. W przypadku potrzeby przekazania większej liczby informacji przez parametry URL-a możemy osiągnąć limit długości URL-a.

Ukryte pola formularza (hidden fields)

- Identyfikator sesji (i ewentualne inne dane) są przekazywane z serwera do przeglądarki jako wartości ukrytych pól formularza
- W przeglądarce po wykonaniu submit, zawartość tych poł jest dołączana automatycznie do wartości pozostałych pól i, jako parametry metody POST lub GET, są przekazywane z powrotem na serwer.

Http Session

- W serwletach mamy do dyspozycji interfejs HttpSession
- Kontener serwletów używa tego iterfejsu do utworzenia trwałes sesji trwającej określany przedział czasu i rozciągającej się wiele requestów od tego samego użytkownika
- Konkrena implementacja zależy od serwera i jego konfiguracji i może opierac się np. na ciasteczkach lub przepisywaniu URLi
- Interfejs HttpSession umożliwia
  - Podgląd i modyfikację infomracji o sesji, takich jak identyfikator sesji, moment utworzenia, czas ostatniego dostępu, itp.

  - Dodawanie do sesji obiektów przechowujących informację dostępną podczas następnych wywołań serwletu dla tej samej sesji
- Interfejs HttpSession

Cookies

- W kodzie serwlet pobieramy z request ciasteczka (jeśli są)oraz tworzymy nowe i dodajemy do response
- Przeglądarka automatycznie do zwrotnego 'request'u dołącza ciasteczka przesłane z serwera
- HTTP Cookie niewielki zestaw informacji przekazywany z serwera do klienta
- Klient może takie ciasteczko zachować i odesłać z powrotem z następnym zapytaniem do serwera
- **Session cookies** - są usuwane przez przeglądarkę kiedy przeglądarka jest zamykana. nie mają ustawionej dyrektywy MaxAge ani Expires)
- **Permanent cookies** - dezaktywuja się w określonym moomencie (dyrektywa Expires) lub po określonym czasie (dyrektywa MaxAge). Mogą trwać pomiędzy kolejnymi uruchomieniami klienta.

Inne ważniejsze dyrektywy

- **Secure** - ciasteczko może być wysłane na serwer jedynie przy użyciu bezpiecznego protokołu HTTPS
- **HttpOnly** - ciasteczko może być wysłane na serwer jedynie przez przeglądarkę. Nie jest możliwe dołączenie ciasteczka z pzoiomu kodu w Javascript.
- **Domain** - fomena (nazwa hosta) do której ciasteczko może być wysłane. Dopuszczalne są także poddomeny. Jeśli Domain nie jest utawione, to można odesłać ciasteczko jedynie do domeny z której przyszło (wykluczając podomeny)

Servlet filtr

- Filtr jest obiektem, który jest wywoływany przed obsłużeniem przez serwlet żadania klienta oraz po jego obsłużeniu.
- Może odczytywać i modyfkować zawartość żądania oraz odpowiedzi
- Zastosowania filtrów
  - zapisywanie do logów
  - kompresja
  - szyfrowanie i deszyfrowanie
  - autoryzacja użytkownika
  - walidacja dostępu do zasobów

9.1 Autentykacja użytkownika

Metody

1. HTTP basic authentication
   - Mechanizm wbudowany w protokół HTTP
   - Schematy: Basic (base64), Bearer(OAuth 2.0), Digest (md5, sha itp)
   - Basic authentication scheme:
     - Identyfikator użytkonika wraz z hasłem przesyłane są w kodowaniu base64, które jest odwracalne

- Powininen (w właściwie musi) być używany z protokołem HTTPS aby uchronić hasło przed dostępem osób trzecich
- Wsparcie dla tej motody jest wbudowane zarówno w kliencie jak i na serwerze
- Brak możliwości wylogowania użytkonika

2. Cookies
   - Po przesłaniu danych autoryzacyjnych serwer generuje identyfikator sesji autoryzowanej.
   - Jest on przechowywany na serwerze oraz przekazany do klienta
   - Klient przekazuje go przy każdym żądaniu dostęu do zasobów
   - Serwer sprawdza, czy identyfikator jest poprawny i aktywny
   - Wylogowanie polega na dezaktywacji lub usunięcia identyfikatora z serwera
   - Brak możliwości wyklogowania użytkonika, ale można określić czas ważności tokena
3. Tokens
   - Najczęściej używane sa JSON Web Tokens (JWT)
   - Token nie jest przechowywany na serwerze, a tylko u klienta (Local storage, Cookies, itp)
   - Klient wraz zżadaniem dostęu do zasobu wysyła token w nagłówku Authorization: Bearer <token>.
   - Serwer weryfikuje poprawność tokenu przy pomocy swojego prywatnego klucza
   - Struktura tokena:
     - Nagłówek (header) - określa typ tokenu u rodzaj lagorytmu haszującego.
     - Zawartość (payload) - Zawiera stwierdzenia najczęściej na temat tożsamości użytkonika (sub - subject)
     - Podpis (signature) - podpis kryptograficzny nagłówka i zawartości.

10 Komunikacja Frontend-Backend

10.1 AJAX

- AJAX (Asynchronous JavaScript and XML) - technika wykonywania zapytań HTTP i pobierania z poziomu aplikacji internetowej w przeglądarce WWW bez potrzeby przeładowania całej strony.
- Bazuje na obiekcie XMLHttpRequest (często używany jest skrót XHR) opisanym w standardzie https://xhr.spec.whatwg.org/ .
- Implementowany przez wszystkie przeglądarki i dostępny z poziomu kodu Javascript.
- Umożliwia wysłania żądania HTTP do serwera, pobrania zasobów przetworzenia ich w kodzie Javascriptu bez potrzeby przeładowania całej strony.
- Oryginalnie wykorzystywany przede wszystkim do pobierania danych w formacie XML, obecnie bardzo popularny jest JSON.

Obiekt XMLHttpRequest

- Zwróconą zawartość można uzyskać z pomocą
  - responseText - w postaci tekstu
  - responseXML - jako obiekt dokumentu XML
- W nowszych wersjach przeglądarek pole responseType pozwala na ustawienie typu zwracanej zwartości. Możliwe są między innymi wartości:

  - "text" - wynik zwracay jako string
  - "document" - wynik zwracany jako dokument HTML (obiekt Document) lub dokument XML (obiekt XMLDocument)
  - "json" - obiekt Javascript powstały na podstawie przesłanego JSON-a.
- Pole readyState może przyjmować następujące wrtaości:
  - 0 - zapytanie niezainicjowane
  - 1 - zapytanie otwarte
  - 2 - zapytanie wysłane
  - 3 - odbieranie odpowiedzi
  - 4 - zapytanie zakończone

  Uwaga: Jeśli przekazywana treść nie jest w odpowiednim formacie otrzymamy null.
- Metoda setRequestHeader() umożliwia ustawienie nagłówka w wysyłanym zapytaniu.
- W starych wersjach Internet Explorera (IE 5/6) obiekt XMLHttpRequest należało uzyskać poprzez wywołanie
- Z powodów bezpieczeństwa współczesne przeglądarki standardowo pozwalają jedynie na ładowanie zasobów z tego samego serwera z którgo była załadowana strona główna. Jednym ze sposobów na obejście tego problemu jest protokół CORS (Cross-domain requests).

Fetch API - nowy interejs służący do pobierania zasobów.

- Ma zastąpić XMLHttpRequest
- Oparte na javasciptowych Promise
- API niskopoziomowe!
- Mamy kontrolę nad wszystkimi parametrami zapytania HTTP.
- Należy samodzielnie ustawiać wszystkie niezbędne nagłówki, ciasteczka, itp Jako błąd (Promise.reject()) traktowana jest sytuacja nie możności pobrania zasobu w wyniku błędu sieciowego.
- Nawet dla odpowiedzi z kodem HTTP 404 obsługa przebiega normalnie, tylko status ok jest ustawiany na false.

10.2 REST

- REST = Representational State Transfer
- Podejście zaproponowane przez Roya T. Fieldinga w 2000 r. w jego pracy doktorskiej Architectural Styles and the Designs of Network Based Architectures
- Oparty na pojęciu zasobu identyfikowanego przy pomocy URI i mogącego posiadać różne reprezentacje (np. XML, JSON).
- Operacje na zasobach wykorzystują metody protokołu HTTP takie jak (GET, POST, PUT, DELETE, itd.)
- Status operacji są zwracane jako statusy protokołu HTTP
- Zakładamy bezstanowość: każda operacja stanowi niezależną całość i serwer może ją zrozumieć i zrealizować bez znajomości poprzednich komunikatów
- Format (reprezentacja) zwracanych danych powinna być określana w nagłówku Content-type. Np. text/xml, application/json
- W praktyce rzadko spotyka się serwery całkowicie zgodne z wytycznymi REST

- Częste odstępstwa:
  - Wykorzystanie własnych kodów błędów
  - Przekazywanie operacji w URL-u zamiast jako metody HTTP (np. POST / delete/book/123)
  - Przekazywanie formaty zwracanych danych w URL-u jako Content-type (np. GET /book/123?fmt=json)
- Czysty REST niezbyt dobrze nadaje się do API zorientowanego na operacje zamiast na zasoby.

10.3 SOAP

- SOAP = Simple Object Access Protocol
- Starszy i bardziej skomplikowany protokół niż REST
- SOAP jest protokołem, a nie jedynie stylem budowania API
- Oparty na XML
- Można go wykorzystywać do przekazywania danych/zasobów lub do wywoływania procedur/operacji
- Używany przede wszystkim w ramach protokołu HTTP, ale może działąć na innych protokołach transportowych
- Wbudowana obsluga błędów i protokołów bezpieczenistwa (WS-Security: m.in. podpisywanie i szyfrowanie wiadomości)
- Posiada dodatkowy język opisu udostępnianych operacji i danych (WSDL)
- Ma ścisle określoną (dość skomplikowaną) strukturę. Wymaga przekazywania sporej ilości dodatkowych danych.
- "SOAP is like an envelope, whereas REST is like a postcard"

11 Spring vs JEE

11.1 JEE

- JEE = Java for Enterprise Edition
- Platforma do tworzenia aplikacji biznesowych
- Jest zestawem specyfikacji, a nie implementacji.
- Twórcy specyfikacji (Sun Microsystems, a następnie Oracle) udostępniają również wzorcową implementację specyfikcaji JEE (obecnie: GlassFish lub Sun Java System Application Server)
- Podstawowe interfejsy programistczne zdefiniowane w pierwszych wersjach, to
  - JDBC (Java Database Conectivity) - dostęp do baz danych
  - Java Servlets - obsługa komunikacja sieciowej (najczęściej HTTP)
  - JSP (Java Server Pages) - dynamiczne strony WWW
  - EJB (Enterprise Java Beans) - specyfikacja komponentów biznesowych po stronie serwera
- Niektóre z tych API zostały przeniesione do "zwykłej" Javy (JavaSE - Java Standard Edition), a JEE wzbogaciła się o wiele innych interfejsów
- Pierwsze wersje nosiły nazwy J2EE 1.2, J2EE 1.3, J2EE 1.4, a później Java EE 5, Java EE 6, Java EE 7
- **Problemy z JEE**:
  - Pierwsze wersje miały dużo możliwości, ale były skomplikowane w użyciu(szczególnie EJB)

  - Aplikacje JEE wymagały dużego (ciężkiego) serwera implementującego pełen zakres specyfikacji
  - W związku z tym pojawiły się konkurencyjne/komplementarne rozwiązania: Spring, Struts, Guice
  - Największą popularność zyskał Spring

Serwery aplikacji (Application servers) - implementują pełną specyfikację JEE między innymi z EJB (Java beans) i JMS (Java messaging).

11.2 Spring

Rozwiązuje pewne problemy z JEE:

- Upraszcza tworzenie aplikacji biznesowych. Szczególnie tych, które nie potrzebowały dużej części interfejsów JEE
- Pozwala na pracę ze zwykłymi obiektami Javy (POJO - Plain Old Java Object) zamiast skomplikowanych EJB).
- Jest zestawem bibliotek zawierących implementacje, co umożliwia wdrażanie aplikacji Springa na serwerach WWW nie implementujących JEE, a nawet budowanie aplikacji desktopowych.
- Opiera na IoC (Inversio of Control) i Dependency Injection, co pozwala między innymi ograniczyć stosowanie niezbyt wygodnych usług nazewnicznych JNDI (Java Naming and Directory Interface)
- Rózna sposoby konfiguracji systemu:
  - pliki XML
  - adnotacje
  - kod Javy
- Dwa warianty budowy aplikacji webowych: standardowe serwlety oraz nieblokujące technologie Reactive stack.
- Moduły Springa można wykorzystać także do budowy aplikacji desktopowych
- Można aplikacje zawierające wpbudowany serwer WWW.
- Spring Boot umożliwiający łatwe tworzenie róznych typów aplikacji bazujący na domyślnych sensowych ustawieniach

Kontener IoC

- Filozofia Spring-a opiera się na odwróceniu zależności (IoC) i wstrzykiwaniu zależności (DE)
- Najważniejszą częścią architektury Springa jest kontener IoC w terminologii Spring-a zwany kontekstem aplikacji, implementujący interfejs ApplicationContext.
- Kontekst aplikacji zarządza cyklem życia komponentów (beans) tworzącymi aplikację.
- Komponenty, to zwykłe obiekty Javy (POJO - Plain Old Java Object)

Wstrzykiwanie zależności w Springu

- przy pomocy XML-a
- przy pomocy adnotacji
- wstrzykiwanie zależności przy pomocy:
  - konstruktora - zalecane dla wymaganych zalezności
  - metod ustawiających (setterów) - najlepiej dla zależności opcjonalnych
  - pól w klasach

Bean scopes

Ogólne zakresy.	
singleton (default)	tworzona tylko jedna instancja (w jednym ApplicattionContext)
prototype	Za kązdym żadaniem pobrania ‘bean‘a zwracany jest mnowy egzemplarza
Zakresy dostępne ApplicationContext typu webowego	
request	Pojedyncza instancja jest tworzona i dostępna podczas trwania jednego żądania HTTP (HTTP request)
session	Pojedyncza instancja jest tworzona i dostępna podczas trwania jednej sesji HTTP (HTTP session)
application	Pojedyncza instancja jest tworzona i dostępna podczas trwania ServletContext.
websocket	Pojedyncza instancja jest tworzona i dostępna podczas trwania WebSocket.

MVC

- **Model** - dane aplikacji (najczęściej POJO)
- **View** - prezentuje dane z modelu dla użytkownika (w naszym wypadku generuje stronę HTML)
- **Controller** - przetwarza żądania użytkownika generując/modyfikując model i przekazuje go do widoku w celu wyświetlenia

Dispatcher Servlet - w Spring MVC głównym obiektem synchronizującym te aktywności jest DispatcherServlet

- **Handler mapping** - mapuje żadania HTTP na odpowiednie kontrolery
- **Controller** - wywołuje odpowiednie metody obsługujące żądania GET, POST, itp. Metody te ustawiają odpowiedni model danych i zwracają nazwę widoku.
- **View Resolver** - mapuje nazwy widoków na odpowiednie szablony stron (np. JSP)
- **View** - Renderuje stronę na podstawie szablonu z odpowiednim modelem

12 Bezpieczeństwo

12.1 OWASP Top 10 Application Security Risks 2017

Open Web Application Security Project - organizacja non-profit zajmująca się zagrożeniami i zabezpieczaniami aplikacji webowych.

- A1:2017 - Injection
- A2:2017 - Broken Authentication
- A3:2017 - Sensitive Data Exposure
- A4:2017 - XML External Entities (XXE)
- A5:2017 - Broken Access Control
- A6:2017 - Security Misconfiguration
- A7:2017 - Cross-Site Scripting (XSS)
- A8:2017 - Insecure Deserialization
- A9:2017 - Using Components with Known Vulnerabilities
- A10:2017 - Insufficient Logging & Monitoring

12.2 SQL Injection

- Polega na niedostatecznej walidacji i filtrowaniu wprowadzonych przez użytkownika danych, które są wykorzystywane do tworzenia zapytania SQL
- Dane te są interpretowane przez bazę jako elementy języka SQL, a nie - zwykłe dane tekstowe.
- Zabezpieczenia:
  - Używanie prepared query (parametrized query) - zapytania parametryzowane. Szkielet tych zapytań te są wstępnie parsowany/kompilowany na serwerze bazy danych. A dopiero później uzupełniany o parametry.
  - Czyszczenie (sanitizing) wprowadzonych danych z niebezpiecznych znaków (apostrofy, cudzysłowy, itp.). Znaki te można usuwać, albo zastępować bezpiecznymi kombinacjami (escaping)

12.3 Cross-Site Scripting (XSS)

- W treści atakowanej strony wstawiany jest kod (najczęściej Javascript) wykonujący niepożądaną akcję.
- Kod ten jest wykonywany jest z aktualnymi uprawnieniami użytkownika, co umożliwia mu dostęp i ewentualną zmianę danych prywatnych użytkownika
- Zabezpieczenia:
  - Podobnie jak w przypadku SQl Injection, sprawdzanie odbieranych przez użytkownika danych. Usuwanie lub zamiana na zwykły tekst wszystkich instrukcji HTML. javascript, itp.
  - Używanie cookies z flagą HttpOnly (oraz Domain) co zabezpiecza przed dostępem do ciasteczek z poziomu skryptów. Skrypt nie może odczytać ciasteczka i przesłać do atakującego. Nie zabezpiecza nas to jednak przed wykonaniem złośliwej akcji w ramach istniejącej sesji na serwerze na który użytkownik jest zalogowany. Przeglądarka automatycznie dołącza ciasteczka do zapytań idących do serwera z pasującym Domain.

12.4 Session hijacking

- Atakujący przechwytuje identyfikator sesji lub jest go w stanie przewidzieć i wygenerować.
- Posiadając identyfikator może połączyć się z serwerem w ramach sesji autoryzowanej już przez jej właściciela i wykonać złośliwe działania.
- Fatalnym pomysłem jest np. nadawanie kolejnych numerów jako identyfikatory sesji!
- Zabezpieczenia:
  - Generowanie niemożliwych do przewidzenia identyfikatorów sesji
  - Używanie HTTPS‘
  - Umożliwienie użytkownikowi wylogowanie sie powodujące dezaktywacja sesji i jej identyfikatora. WARNING: nie wszystkie sposoby autoryzacji dają taka możliwość (np. tokeny JWT).

12.5 Session Fixation

- Wiele serwisów przy pierwszym połączeniu użytkownika do serwisu rozpoczyna sesję (nadaje użytkownikowi session ID), a następnie, już w ramach tej sesji, pozwala mu się zalogować do serwisu.
- W takiej sytuacji czasami możliwy jest atak Session Fixation
  1. Atakujący łączy się do strony logowania.
  2. Serwer zawraca stronę logowania z identyfikatorem sesji
  3. Atakujący przesyła ofierze link do strony logowania z ustawionym swoim identyfikatorem sesji

4. Ofiara loguje się na serwer używając przekazanego identyfikatora sesji
5. Ofiara loguje się na serwer używając przekazanego identyfikatora sesji
6. Użytkownik ładuje stronę z kontem ofiary używając wspólnego identyfikatora sesji, która została już wcześniej autoryzowana przez ofiarę.

- Zabezpieczenia:
  - Zablokowanie przyjmowania Session ID w URL-u i parametrach POST. Zabezpiecza przed pierwszymi dwoma typami ataków.
  - Generowanie nowej sesji (nowy Session ID) po zalogowaniu użytkownika.

12.6 Cross-Site Request Forgery

- Definicja wg. OWASP: A CSRF attack forces a logged-on victim's browser to send a forged HTTP request, including the victim's session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim's browser to generate requests the vulnerable application thinks are legitimate requests from the victim.
- Przykładowy scenariusz:
  1. Atakujący chce wykonać nieautoryzowany przelew z banku, który udostępnia taką funkcjonalność przez wywołania postaci
  2. Ofiara jest zalogowana na serwer (np. do banku)
  3. Atakujący podsuwa ofierze stronę, maila, itp. zawierającą link wykonujący złośliwą operację na serwerze (np. przelew bankowy).
  4. Ofiara uruchamia podsunięty link w przeglądarce, w której aktywna jest sesja na serwer. W przypadku drugiego linka, ofiara nie musi nawet uruchamiać linka - zostanie on automatycznie uruchomiony w celu pobrania "obrazka"
  5. Link zawierający żądanie dla serwera zostaje przesłany na serwer wraz ze wszystkimi ciasteczkami (w tym z identyfikatorem sesji).
- Zabezpieczenia:
  - Synchronizer (CSRF) Tokens - Każda operacja zmieniająca stan aplikacji wymaga dodatkowego unikalnego jednorazowego tokenu (CSRF token).
  - Dodatkowa autoryzacja przez użytkownika - dodatkowe hasło, CAPTCHA, token jednorazowy
  - Atrybut SameSite w ciasteczku - zapobiega wysłaniu ciasteczka jeśli zapytanie nie pochodzi z tej samej strony/ serwisu, z której pochodzi ciasteczko.
  - Atrybut SameSite w ciasteczku - zapobiega wysłaniu ciasteczka jeśli zapytanie nie pochodzi z tej samej strony/ serwisu, z której pochodzi ciasteczko.
  - **CORS**

Same Origin Policy

- Aby zabezpieczyć się przed CSRF zasób (skrypt) pobrany z danej lokalizacji (Origin) może czytać dane i wysyłać jedynie do innych zasobów z *tej samej lokalizacji). (Same Origin Policy‘)
- Lokalizacja = scheme + host + port
- **Problemy z SOP** - czasami chcemy zezwolić na dostęp do naszych zasobów z zewnątrz. Na przykład udostępniając publiczne API, fonty, skrypty.

NOTICE: Brak takiego zabezpieczenia spowodowałby możliwość wywołania https://bank-example.com/transferFunds przy wejściu na stronę https:// evil-example.com/bad.html. Jeśli użytkownik byłby juz zalogowany w banku, to transferFunds wykonałby się z jego uprawnieniami.

CORS = Cross-Origin Resource Sharing.

- Każde żądanie w trybie CORS powoduje dodanie prze przeglądarkę do zapytania nagłówka Origin z adresem serwisu z którego pochodzi strona wysyłająca żądanie.
- Nagłówek jest ustawiany przez przeglądarkę i kod użytkownika nie ma możliwości zmiany tego nagłówka.
- Serwer odpowiada z nagłówkiem
- Przeglądarka sprawdza, czy zwrócona wartość pozwala na udostępnienie zasobu i przekazuje go do kodu strony tylko wtedy, gdy jest to dozwolone.
- Istnieje tez możliwość wymuszenia dodatkowej autoryzacji.