PROGRAMOVÁNÍ INTERNETOVÝCH APLIKACÍ

Základy a principy WWW

INTERNET (=VZÁJEMNĚ PROPOJENÉ SÍTĚ)

- Hlavní principy:
  - » Heterogenita
  - » Znovupoužitelnost
  - » Škálovatelnost
  - » Důraz na otevřené standardy (standardy jsou ověřené několika nezávislými implementacemi)

HYPERTEXT

- Problém: hodně informací a způsob jejich prohledávání je primitivní
- Řešení: použít asociace – blízké lidskému myšlení a paměti
- Nelineární organizace textu
- Uzly (informace, koncepty) a jejich vztahy (propojení, odkazy)
- Hypertextové systémy:
  - » GUI, aktivní odkazy, tvorba dokumentů
  - » Standardní mechanismy vyhledávání
  - » Databáze

ZNAČKOVACÍ JAZYKY

- Potřeba vyhledávat, zpracovávat, vyměňovat a sdílet dokumenty efektivně

FYZICKÉ VYZNAČOVÁNÍ (PROCEDURÁLNÍ)

- Značky definují formát zobrazení
- Ztrácíme informaci o struktuře textu
- Problémy: změna prezentace, konverze do jiného systému vyznačování
- Technologicky závislé, proprietární
- Např.: RTF: \pard\plain \s1\sb240\b\f5… text

LOGICKÉ VYZNAČOVÁNÍ (od 1960)

- Značky popisují strukturu informace
- Umožňuje efektivní zpracování, přesné, nezávislé na technologii
- Např.: DocBook: `<book> <title>` Název knihy `</title>` …

SGML (STRUCTURED GENERALISED MARKUP LANGUAGE)

- Meta-jazyk pro defimování logických vyznačovacích jazyků
- Struktura dokumentu (elementy) je vyjádřena pomocí značek (nikoli tagů!)
- Značky jsou definovány v Document Type Definition (DTD)
- Př.: elektronické rukopisy, LinuxDoc, DocBook, HTML, XML

XML

- Meta-jazyk pro definování logických značkovacích jazyků
- Struktura dokumentu (elementy) je vyjádřena pomocí značek (angl. tags)
- Terminologie, produktivita, znovupoužitelnost, flexibilita
- Cíle:
  - » Generování, zpracování, transformace
  - » Podpora různých aplikací, Internet
  - » Není požadována stručnost zápisu

World Wide Web (WWW)

- Svět informací dostupných po síti
- Globální, snadno se používá, přístupný
- Umožňuje další vývoj, je poměrně efektivní
- Má klient-server architekturu
- Decentralizace, heterogenita, hypertextové odkazy

Architektura webu

- **URI** (Uniformn Resource Identifier)
  - » Celosvětově unikátní adresa
- **URL** (Uniformn Resource Locator)
  - » Speciální případ URI, obsahuje přístupový mechanismus
- **MIME** (Multipurpose Internet Mail Extensions
  - » Označuje formát obsahu
  - » Př.: image/png, application/msword, text/plain

HTML

- Jazyk popisující strukturu dokumentu
- SGML/XML Aplikace
- HTML 1 – 1990
- HTML 2 –1995 kodifikace aktuálního stavu jazyka, všechny základní elementy
- HTML 3 – 1995 pokus o silný standard, matematické vyznačování, nepoužívané
- HTML 3.2 – 1997 nové elementy: table, div, font, map, applet
- HTML 4.0 – 1998 přenositelnost, přístupnost, nové elementy style, frame, obejct, script
- HTML 5 – 2013

- Značky jsou case insensitive *(nezáleží na velikosti)*
- Možno vynechat uzavírací značky
- Atributy i bez uvozovek a s minimalizací
- Ne-SGML data – obvykle stačí komentáře
- Renderování = volná interpretace, tolerance

XHTML

- Zjednodušení DTD
- Snažší strojové zpracování, výměna dat
- Lepší modularita a rozšiřitelnost jazyka
- XHTML 1.0 – 2000, HTML 4.01 jako XML aplikace, čistě logické vyznačování
- XHTML 1.1 – 2001, modularizace XHTML 1.0
- XHTML 2 – bez prezentačních prvků, obecnější textové vyznačování
- Značky jsou case sensitive *(povoleny pouze malá písmena)*
- Uzavírací značky povinně (i u nepárových tagů `<br />`)
- Atributy mají povinně uvozovky, žádná minimalizace
- Ne-XML data = povinně CDATA sekce, lépe v externích souborech
- Renderování = striktní chování

Strict

- Pouze logické vyznačování

Transitional

- Deprecated element
- Kompatibilní se starými prohlížeči

HTML DOKUMENT

- **Preambule**
  - » SGML (tj. též HTML) implicitní
  - » XML (tj. také XHTML) povinná (`<?xml version="1.0" encoding="UTF-8"?>`)
- **Deklarace**
  - » `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`
  - » Označuje gramatiku a je povinná kvůli interpretaci dokumentu
- **Záhlaví** (`<head></head>`)
  - » Meta-informace o dokumentu
  - » Nezobrazuje se
  - » TITLE – titulek
  - » META – meta-informace (autor, keywords, description, vyhledávače,…)
  - » LINK – odkazy na úrovni dokumentů (next, prev, contents, index, copyright, stylesheet,…)
  - » STYLE (type, media)
  - » SCRIPT, NOSCRIPT (type, src)
- **Tělo** (<body></body>)
  - » Obsah dokumentu
  - » Obsahové elementy
    - Blokové – zalamují odstavec, bloky, tabulky, formuláře, v HTML5 lze nově i element A
    - Textové – uvnitř blokových – frázové x prezentační
    - Generické – kontejnery (DIV, SPAN), vazba na CSS
    - Obecné atributy – všechny elementy mají **id, class**, **style**, **title**, land, dir…

ZÁKLADNÍ BLOKOVÉ ELEMENTY

- Odstavec – p
- Nadpisy – h1 až h6
- Odrážky, číslování – ul/ol li, dl dt dd
- Citace – blockquote, atribut cite=“URI“
- Předformátovaný text – pre
- Odřádkování – br

ZÁKLADNÍ TEXTOVÉ ELEMENTY

!! Pozor v HTML5 má spoustu těchto elementů jiný sémantický význam!!

- Důraz – em, zesílení – strong
- Podtržení, škrtuní – ins, del
- Tučné, kurzíva, podtržení – b, i, u
- Indexy - sub, sup
- Monospaced text – tt
- Další – cite, abbr, q, code

HYPERMEDIÁLNÍ ELEMENTY

- Odkaz – a (name, href, title)
- Obrázek – img (src, title, alt)
- Klikací mapy - map, area
- Další – object, applet, param

GENERICKÉ KONTEJNERY

- div, span – id**,** class, style

Formulářové elementy

- form – metod, action, enctype
- input *(nepárový)* – name, type, value, size, maxlength, checked
  - » type: text, hidden, password (HTML5: tel, email, date, ….)
- select – name, multiple
  - » optgroup (vnořený tag)
  - » option – selected, value (vnořený tag)
- textarea – rows, cols, name
- label – for (odkazuje na atribut id)
- fieldset
  - » legend

Tabulky

- table, caption
  - » width, border
- tr *(řádek)*, th *(zvýrazněná buňka)*, td *(buňka)*
  - » colspan, rowspan
  - » align, valign (nepoužívat!!)
- thead, tfoot, tbody *(kontejnery řádků)*
- col, colgroup *(stylování sloupců)*

Rámce: !!NIKDY NEPOUŽÍVAT!!

Strukturování obsahu

- HTML 3 bez CSS = vyznačit tak, aby se co nejlépe zobrazilo
  - » Tabulkový layout
- HTML 4/XHTML s CSS = vyznačit tak, aby se co nejlépe vyhledávalo
  - » Důležitý obsah napřed, title, h1/h2
- HTML5 – ještě větší důraz na sémantiku, nové značky (SECTION, HEADER, FOOTER, ...)

Přístupnost

- Bezbariérovost
- Extrémně handicapovanými návštěvníky jsou vyhledávací roboti
- Zásady:
  - » Validovat
  - » Používat informační strukturování
  - » Title, hierarchie nadpisů, oddělená navigace, linearizace tabulek
  - » Čitelný a srozumitelný text
  - » Členit text
  - » Zpřístupnit formuláře (používat atribut tabindex, případně accesskey)

Možnosti HTML pro přístupnost

- **Elementy**
  - » H1-h6, p, div
  - » Em, strong, q, cite
  - » Fieldset, legend, optgroup, label
  - » Th, thead, tfoot, caption
- **Atributy**
  - » obecně -tTitle, lang, accesskey
  - » input, select, textarea – title, tabindex
  - » img – alt, title

Kaskádové styly (CSS)

- oddělení obsahu (HTML) od prezentace/formátování (CSS)
- CSS1 – 1996 zejména HTML
- CSS2 – 1998 typy zařízení, generování obsahu, podpora XML
- CSS 2.1 – drobné opravy, sladění s realitou
- CSS 3 – modularizace, stránkování

ZÁKLADNÍ PRINCIPY

- Deklarativní, nestrukturovaná jazyk
- Terminologie DTP
- Stylesheet se skládá z pravidel
  - » **Selektor**: co se bude formátovat
  - » **Deklarace**: jak se to bude formátovat (vlastnost + hodnota)
- Další části:
  - » **At-pravidla**
    - dovolují použití určitého stylu na cíl, který není v XHTML definovatelný, nebo představují výkonné příkazy
    - začínají znakem @ následovaným identifikátorem pravidla
    - @import, @media, @page (př.: @import "styl.css")
  - » **Deklarace !important –** nejsilnější pravidlo již nezáleží na kaskádě toto vždy vyhraje
  - » **Komentáře** `/* comment */`

PŘIPOJENÍ CSS K HTML

- Pomocí elementu **LINK** `<link rel="stylesheet" type="text/css" href="./cool.css"></link>` v HEAD
- Deklarací importu **@import** url(http://style.com/basic) v jiném souboru css
- **Interním stylem**
  - » **V <HEAD>** stránky <style type = "text/css"> … </style>
  - » **Atributem** u elementu <p style="color: red"> (!má vysokou prioritu)

SELEKTORY

- **Elementy**: p, h1, ul, li (i kontextově, tj. ul li ul li – pro odrážku 2. stupně)
- **Třídy** (všechny elementy třídy) a Identifikátory (jeden element) (.class, #id)
- **Pseudotřídy** – efekty nedosažitelné přes HTML strukturu a:hover, input:focus, p:first-line
  - » P:first-child, P:lang(en), P:before, P:after
- Kombinace selektorů (př. p.zahlavi EM {})
- Třídy není možné vnořovat, ale je možné je kombinovat (**NE**: p.zahlavi.obsah)

DĚDĚNÍ VLASTNOSTÍ

- Většina vlastností definovaných pro daný uzel HTMl stromu se dědí na jeho potomky
- Relativní rozměry – vůči zděděné hodnotě
- **Nedědí se:** background, bg image, margin, border, …

POSTUP KASKÁDY

- Najdu všechny hodnoty vlastnosti (také zděděné a defaultní)
- **!important** dopředu
- seřadím podle autor > čtenář > prohlížeč
- style="“ > #ID > .class > kontextový selektor > typový selektor
- poslední deklarovaná hodnota platí

Navigace a přístupnost

Layout stránky

- Rozložení stránky
- Hlavním účelem je orientace, navigace (kde jsem, co je tu, kde najdu to, co hledám)

Navigace

- Cílený postup za účelem dosažení místa nebo cíle
- Dobrá navigace = opakovaná návštěvnost
- Dodržovat konvence (logo, odkazy, menu, …)

- Klíčová hlediska:
  - » Jednoduše rozpoznat a naučit se
  - » Konzistentnost (co pracovalo dříve pracuje pořád)
  - » Interaktivita, zpětná reakce
  - » Poskytování dalších možností (zkratky)
  - » Odpovídá účelu stránek

Titulní strana

- Proč bych měl být tady a ne někde jinde
- Představení navigačních stylů, upoutávky v obsahu, vyhledávání, reklama
- NE: úvodní obrazovky (intra), přeplnění

Obsahové stránky

- Obsah je důležitý
- Informace o poslední změně, autorovi,…

Použitelnost

- Prvky použitelnosti:
  - » Klikatelné odkazy (vypadá to jako odkaz → měl by to být odkaz)
  - » Srozumitelné uspořádání, jasný text
  - » Přehledné dialogy, formuláře
  - » Vyhledávání
  - » Varianty pro cílová zařízení/prostředí
  - » Použitelnost a RIA (rich internet applications – drag&drop, kontextová nápověda…)

Přístupnost

- Textové alternativy k netextovým informacím
- Nepoužívat pouze barvy ke sdělování indormace
- Změny obsahu a zobrazení dělat jen na vyžádání uživatele (reload, pop-up, flash)
- Navigace je jasná a logická (název stránky, menu, odkazy, … zdroje označeny [pdf])
- Text je srozumitelný, krátký, tématický, strukturovaný (nadpisy, zvýrazňování)
- Přístupnost a RIA (jak zajistit drag&drop pomocí klávesnice?)

Search Engine Optimization (SEO)

- Maximalizace zisku & návštěvnosti
- Předmětem je **zviditelnit web** tak, aby jej našlo co nejvíce dobře zacílených zákazníků za přijatelné náklady
- PPC, bannery, e-mailing, přirozené výsledky, zpěné odkazy, silná značka, affiliate, offline reklama
- Použitelnost, přesvědčivost, konkurenceschopnost, důvěryhodnost, přístupnost, grfika, značka
- **On-page faktory**:

    - » Obsah, klíčová slova, URL, title, nadpisy, meta description
- **Off-page faktory**:
    - » Registrace v katalozích, odkazy z jiných webů, PageRank

Technické překážky indexace

- **Duplicitní obsah (stejný obsah na různých URI [URL])**
- Menu přes Javascript (tohle už Googlebot umí)
- PDF, Word (tohle Googlebot umí taky)
- Flash (tohle je téměř mrtvý)

HyperText Transfer Protocol

- Účel: přenos hypertextových/hypermediálních dokumentů, přenos dat od klienta, SOAP, WebDAV
- Bezstavovost
- Aplikační vrstva předpokládá spolehlivý přenos
- Textový protokol, nejčastěji port 80 a 443 pro HTTPS

Mechanismus HTTP

- Klient posílá požadavek
    - » Jaký chce objekt
    - » Hlavičky
    - » Tělo s daty
- Server posílá odpověď
    - » Stav
    - » Hlavičky
    - » Tělo s daty

Požadavek

```
<method> <URI> <version> <header>* <body>
```

- **Metoda** = požadovaná akce
    - » GET, HEAD, POST, PUT, DELETE, OPTIONS, TRACE
- **URI**
    - » Absolutní bez hostname
- **Verze**
    - » HTTP/1.0, HTTP/1.1
- **Hlavička** (podle MIME standardu)
    - » Host, From, User-agent, Referer, Accept-Language, Accept-Charset, If-Modified-Since, Authorization, Content-Type (POST), Content-Length (POST)

Odpověď

```
<version> <code> <description> <header>* <body>
```

- **Stavové kódy**
    - » 100 Continue
    - » 200 OK, 204 No content
    - » 301 Moved permanently, 302 Moved temporary
    - » 401 Unauthorized, 403 Forbidden, 404 Not found
    - » 500 Internal error, 501 Not implemented
- **Hlavička**
    - » Server, WWW-Authenticate, Location
    - » Content-Type (default: application/octet-stream)
    - » Content-Length, Content-Encoding. Last-modified, Expires, Pragma

HTTP autentikace

- Účel: přístup ke chráněnému obsahu (realm)
- **Základní autentikace** (basic) (heslo je plaintext)
  - » WWW-Authenticate: Basic realm = "name"
  - » Authorization: Basic base64(login:passwd)
- **Digest autentukace** (heslo jako MD5 hash)
  - » WWW-Authenticate: Digest realm = "name" Domain = "URI" nonce=<unique string> opaque = <string> stale = True/False algorithm=MD5|token
  - » Authorize: Digest username ="name" … request-digest=<rddata>

Cookies

- Způsob uchovávání informací na klientu
- Hlavní použití:
  - » Správa relací
  - » Sledování uživatelů
  - » Personalizace stránek

Nastavení cookie v HTTP

- Hlavička odpovědi: Set-Cookie: name=value
- ;EXPIRES=dateValue ;DOMAIN = domainName ;PATH=pathName ;SECURE
- Expires: DOW, DD-Mon-YY HH:MM:SS GMT
- Domain: jména DNS, pro které je cookie platné
- Path: podprostor URI
- Secure: poslat cookie pouze přes bezpečný kanál

Posílání cookie z klienta

- Cookie: name=value, name2=value2
- Klient pošle všechna cookie, která
  - » Jsou určena pro doménu požadovaného serveru
  - » Mají cestu, která souhladí s URI požadavku
  - » Maximálně 4 KB (to je limit HTTP hlavičky)
- Mazání cookie
  - » Pošle se prázdná hodnota
  - » Pošle se již prošlá doba vypršení

Zabezpečení přenosu

- Nativní šifrování není dostupné
- SSL (Secure Socket Layer) (HTTPS na portu 443)
- TLS (Transport Layer Security)
- Asymetrická šifra klíče, Symetrická šifra pro komunikaci

WebDAV (Web Document Authoring and Versioning) (přehled)

- Rozšíření HTTP/1.1
- Mění WWW v zapisovatelné médium

Web Servery

- Účelem je implementace HTTP
- Přidané služby

Služby poskytované serverem

- **Jádro**
  - » HTTP protokol, virtuální servery
  - » Load balancing, throttling, clustery
  - » Rozšiřující API
  - » Administrativní rozhraní
- **Statické dokumenty**
- **Provoz aplikací**
  - » Interpretace HTML – embedded scriptování (PHP, ASP, JSP)
  - » Kontejner pro webové aplikace (ASP, Servlety)

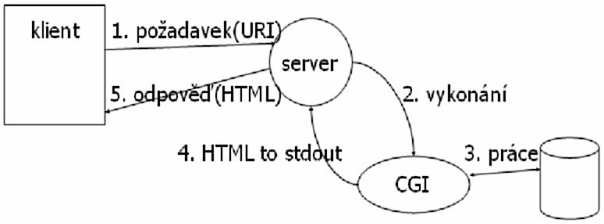
Server-side technologie pro webové aplikace

- **Cíl**
  - » Dynamické generování webového obsahu
  - » Integrace legacy aplikací

- **Prostředky**
  - » Externí aplikace napojená na web server
  - » Aplikační server, který řeší i webový přístup
  - » Webový kontejner, který umožňuje aplikační komunikaci
  - » Embedded skriptování

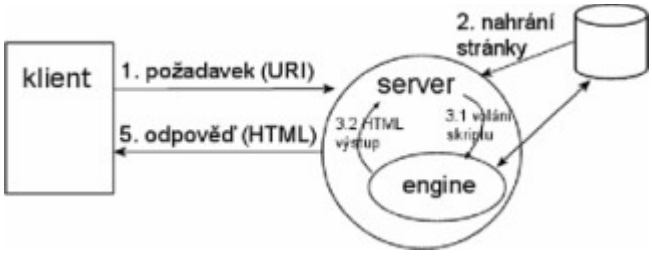
Externí aplikace

- Server spouští samostatný spustitelný soubor



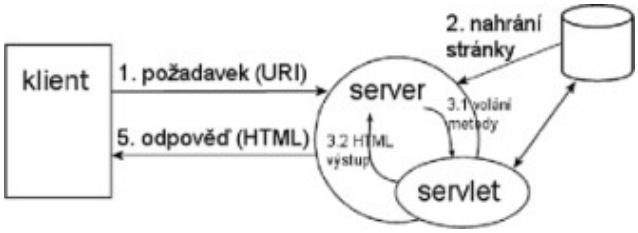
- **CGI (Common Gateway Interface)**
  - » Kompilované nebo interpretované jazyky
  - » Identifikováno a spuštěno serverem na základě URI
  - » Flexibilita, integrace legacy aplikací
  - » Netransparentní, run-time náklady, nebezpečný
- **Mod_perl**
  - » Interpret perl vestavěny v Apache serveru
  - » Snižuje režijní náklady (spawn, šifrovací cache)

Skripty zabudované do HTML



- Server obsahuje interpretovaní engine
- Skripty jsou součástí stránek
- Skripty ani data nejsou perzistentní
- Nedají se škálovat, komplikovaně se integrují, ale rychle se vyvíjí

Servlety



- Externí web-aware kompilované aplikace
- Objekty jsou mezi požadavky perzistentní
- Nízké režijní náklady, vysoká výkon, vstup aplikace
- Větší run-time náklady, vývojový čas

Common Gateway Interface

- Klient posílá data přes GET nebo POST
- Server spouští CGI a vrací jeho stdout
- **Požadavek**
  - » QUERY_STRING, REQUEST_METHOD
  - » CONTENT_TYPE, CONTENT_LENGTH
  - » HTTP_* (např.: HTTP_ACCEPT)
- **Informace o serveru**
  - » SERVER SOFTWARE, SERVER_NAME, SERVER_PORT
  - » PATH_INFO
  - » SCRIPT_NAME
- **Informace klienta**
  - » REMOTE_HOST, REMOTE_ADDR
  - » HTTP_USER_AGENT

Skripty zabudované do HTML

- **PHP** (PHP: Hypertext Preprocessor)
  - » Malý footprint, rychlé, integrace db
  - » Open source
  - » Špatně škálovatelné, nehomogenní
- **JSP** (JavaServer Pages)
  - » Tomcat, ..
  - » Podpora, přenositelnost, integrace ostatních J2EE služeb
- **ASP** (Active Server Pages)
  - » Výkon, intefrace BackOffice
  - » Jeden výrobce (Microsodt), platformová nestabilita
- **Oracle**, **Sybase**
  - » Zabudované jazyky PL/SQL
  - » Pevná integrace databázového enginu

Server-side includes

- Útržky HTML kódu vkládané serverem při posílání dokumentu
- Použití
  - » Vytvoření standardního designu stránky
  - » Vkládaní opakovanách částí HTML
  - » Vkládání samostatně editovaného obsahu

Související technologie

- Jmenné služby (LDAP, JNDI)
- Zabezpečení, autentikace (GS API, JAAS)
- Persistence, přístup k datové vrstvě (JDBC, EJB, Hibernate)
- Transakce (JTA)
- Práci s XML (SAX, JAXP)

Architektury aplikací

- Vždy klient-server (tlustý x tenký)
- Nativní klient (logika v klientu, data na serveru)
- Applet (omezení přístupu na hostitele)

Třívrstvá architektura

- Tenký klient – renderuje GUI
- Aplikační server – business logika, vazba na legacy, podpora GUI
- Datový server – obvykle RDBMS

Mixovaný model – tlustý klient někde, tenký jinde

Webové služby

- přímá komunikace mezi aplikačními vrstvami
- RPC: aplikace přistupující k jiné aplikaci
  - » **WSDL**: popis rozhraní aplikace
  - » **SOAP**: protokol pro přenos vzdáleného volání přes HTTP
  - » **UDDI**: registr dostupných rozhraní
- **REST**: webové aplikace/služby s plain http a čistými URL

Základní pojmy

- **Kontejner** – prostředí pro běh servletů (např. Apache Tomcat)
- **Servlet** – Třída Javy, která umí obsloužit http požadavek
- **JavaServer Page** (JSP) – Java jako zapouzdřený skriptovací jazyk

Pracovní cyklus servletu

- **Vytvoření**, **kompilace** – kód servletu + podpůrný kód + deployment descriptor
- **Packaging** – vytvoření war souboru
- **Nasazení**, **konfigurace** – vložení do kontejneru, informování kontejneru, konfigurace, dojde i instanciování a inicializaci servletu kontejnerem
- **Čekání** + **obsluha požadavků** – kontejner odchytí http požadavek, určí, který servlet jej zpracuje, spustí jeho obslužnou metodu, servlet obdrží data a generuje odpověď
- **Ukončení** – kontejner spustí finalizační metodu servletu

Pomocné třídy a rozhraní

- ServletContext a ServletConfig
- ServletOutputStream
- HttpSession
- Cookie
- ServletException, IOException

Obsluha požadavku

- Obslužné metody **doGet** a **doPost**
- Zavolány kontejnerem podle HTTP metody
- Kroky při obsluze požadavku
  - » Určit, zdaje HTTP metoda implementována
  - » Získat vstupní parametry/data požadavku
  - » Nastavit content-type odpovědi
  - » Generovat data odpovědí
  - » Zapsat odpověď do proudu
  - » Nastavit chybový kód

Vytváření odpovědí

- Rozhraní ServletResponse a HttpServletResponse
- Výstupní proudy ServletOutputStream (pro binární data), PrintWriter (pro text)
- Nastavení stavového kódu (sendError(int code))
- Nastavení hlaviček setContentType(String type), setHeader(String name, String value)
- Generování HTML
- Pozdní hlavičky
  - » Bufferování je defaultně vypnuto
  - » isCommitted() + resetBuffer()
  - » setBufferSize(int size) – pro poslání chybového kódu nebo hlavičky až po těle

Složky servletové aplikace

- Servlety
- JSP a HTML stránky
- Popis aplikace – deployment descriptor

Kontext servletu

- **Kontext** = webová aplikace (0 -1 v kontejneru)
- Dovoluje servletu komunikovat s kontejnerem
- Definovaný adresářem, v němž je servlet nasazen a deployment deskcriptorem
- Přístup: přes rozhraní ServletContext nebo přes metodu getServletContext()

Inicializace servletu

- Při natažení instanciaci kontejnerem
- Typické akce
  - » Načíst konfigurační data
  - » Otevřít spojení (k databázi), připojit se ke zdrojům
  - » Inicializovat lokální data
- Metoda = init(), konec = destroy()
- Konfigurační parametry se nastaví v deployment descriptoru

```
<servlet>
  <init-param>
        <param-name>default-login</param-name>
        <param-value>guest</param-value>
  </init-param>
```

- Přístup přes ServletConfig Interface
- Servlet musí znát typy/třídy datových položek

Předávání hodnot v aplikaci

- Komunikace mezi servlety
  - » Přes objekty v různých vrstvách aplikace
  - » Různé rozsahy platnosti předávaných dat
- Obecné rozhraní, obecný mechanismus
  - » Atributy objektů, get/set metody
    - Enum getAttributesNames();
    - getAttribute, setAttribute, removeAttribute

Rozsahy platnosti

- Objekty reprezentující rozsahy
  - » Aktuální servlet
  - » Požadavek (request)
  - » Session
  - » Aplikace (context)
- HttpSession HttpServletRequest.getSession();
- ServletContext GenericServlet.getServletContext();

Sessions, správa relací

- **Primitivní metody**:
  - » Skryté prvky formuláře
  - » Parametry UR
  - » Cookies
- **Objekt relace** (rozhraní HttpSession)
  - » Reprezentuje relace, obsahuje její data
  - » Získaný přes metody HttpServletReques

Odkazování na zdroje

- **Zdroj** = jiný servlet, jakýkoliv jiný obsah/objekt
- **Nepřímý odkaz**
  - » Pošleme redirect klientovi
  - » Přeneseme stavové informace pomocí URL
- **Přímý odkaz**
  - » Rozhraním ServletRequest a ServletContext přes RequestDispatchet

Thread Safe servlety

- Servlety jsou vícevláknové
- Používat synchronized metody a bloky
- Implementovat rozhraní SingleThreadModel (! Neřeší sdílené zdroje)

Filtry

- Článek zpracovávání požadavku
  - » Nevytváří, jen transformuje
  - » Autentikace, logování, komprese
  - » Filtry spojeny do řetězu
- Rozhraní s.Filter
  - » Metoda doFilter()
  - » Inicializace, ukončení

Deployment descriptor

- podobné servletu

Mapování

- na kterých URI
- v jakém okamžiku
  - REQUEST
  - FORWARD
  - INCLUDE
  - ERROR

```
<filter>
<filter-name>AuthBlocker</filter-name>
<filter-class>cz.zcu.AuthFilter</filter-class>
 <init-param>
  <param-name>group</param-name>
  <param-value>administrators</param-value>
 </init-param>
</filter>

<filter-mapping>
  <filter-name>AuthBlocker</filter-name>
  <url-pattern>/admin/*</url-pattern>
  <dispatcher>REQUEST</dispatcher>
  <dispatcher>FORWARD</dispatcher>
</filter-mapping>
```

Listenery

- Reakce na události v aplikaci
- Návrhový vzor Listener nebo Observer
- Úrovně událostí (Xxx = …)
    - » Aplikace (ServletContext), relace (HttpSession, HttpSessionAttribute), požadavek(ServletRequest)
- Metody posluchače
    - » ContextInitialized(ServletContextEvent sce)
    - » requestDestroyed(ServletRequestEvent rre)
    - » attributeAdded(HttpSessionBindingEvent se)
- Metody události - obvykle vrací objekt dané úrovně

Logování

- Možno psát na stdout, stderr (do konzole)
- Perzistentní hlášení = do logu
    - » Přes kontext serveru
    - » S pomocí logovacích knihoven
    - » nohup.out (bash?)