- Tanki i debeli klijent

o Debeli klijent (*fat client*):
o sadrži slojeve prezentacije i aplikacijske logike
o zahtijeva veću snagu obrade računala domaćina i veću količinu podataka prenošenih mrežom
o Tanki klijent (*thin client*):
o sadrži samo sloj prezentacije
o manja snaga obrade, manja količina prenošenih podataka

- **URI**

"A Uniform Resource Identifier (URI) is a compact sequence of characters that identifies an abstract or physical resource."
◊ predstavljen nizom znakova iz vrlo ograničenog skupa znakova, može postojati u različitim oblicima
◊ omogućuje razlikovanje pojedinog resursa naspram svih drugih resursa
◊ resurs je sve što se može identificirati URI-jem ☺, može i ne mora biti dohvatljiv
Komponente URI-a
o Sastoji se od pet osnovnih komponenata: **sheme** (schema), **autoriteta** (authority), **puta** (path), **upita** (query) i **fragmenta** (fragment)
o Svaki URI mora se sastojati barem od sheme i puta (put može biti prazan)

- **MIME**

o Multipurpose Internet Mail Extension
o Zamišljen kao proširenje e-pošte
o Prerastao okvire e-pošte i prihvaćen u drugim protokolima (npr. HTTP, ...)
o Opisan u RFC 2045, 2046, 2047, 4288, 4289, 2077.
o Proširuje poruke - osim 7-bitnog ASCII teksta sadrže
o Tekst u drugim kodiranjima osim ASCII-ja
o Privitke u drugim oblicima osim teksta (slike, dokumente, ...)
o Višedjelne poruke (*multi-part*)
o Podatke u zaglavljima u drugim kodnim stranicama (ne-ASCII)

- Resursi

o URI – metoda imenovanja (adresiranja) resursa
o HTTP – metode manipulacije resursima
o Web – sustav za rad (stvaranje, dohvat, promjenu i brisanje) s resursima
o Zasnovan na kombinaciji URI+HTTP
o MIME – opis oblika resursa
o Osnovna arhitektura (stil):
o REST – Representational State Transfer
o O ovome kasnije

- HTTP je protokol bez očuvanja stanja - *stateless*
- Zahtjev

o Određuje metodu HTTP-a
o Operacija nad resursom
o Određuje URI resursa
o Jednoznačno određuje resurs
o Dodatna oznaka verzije protokola
o Različiti zahtjevi na strukturu poruke
o Kompatibilnost metoda prema starijim verzijama
o Primjer:
GET /primjer.html HTTP/1.1
Metoda URI Verzija protokola

- Metode HTTP-a

o Norma definira metode
o GET – vraća sadržaj uri-ja
o HEAD – vraća opis – metapodatke uri-ja
o POST – šalje dodatne podatke
o PUT- dodaje novi resurs na poslužitelj
o DELETE – briše resurs sa poslužitelja

- ODGOVOR HTTP

o Struktura:
Response-Line = HTTP-Version Status-Code Reason-Phrase CRLF
o Određuje rezultat metode HTTP-a
o Zadan numerički (kôd rezultata, *Status-Code*) i tekstualno (opis, *Reason-Phrase*)
o Oznaka verzije protokola
o Može ili ne mora biti ista kao verzija protokola zahtjeva
o Primjer:
HTTP/1.1 404 Not Found
Odgovor: Verzijaprotokola KôdRezultata Opisrezultata

- Common Gateway Interface

o Rješenje: Proširiti poslužitelj mehanizmom za prijenos podataka vanjskim programima – CGI
o Common Gateway Interface
o Poslužitelj komunicira sa programima koji djeluju kao poveznice (*gateways*) weba sa ostalim sustavima
o Zadržana općenitost i fleksibilnost (web ostaje web)
o Nema intervencija u protokol HTTP
o HTML ostaje jezik za opis sučelja prema korisniku
o Poslužitelj HTTP-a ne mijenja se prema primjeni
o Resurse (rezultate rada *poveznice*) preuzima poslužitelj weba i vraća korisniku (pregledniku)
Što je CGI?
o CGI: Opis normiranog sučelja između poslužitelja (HTTP) i programa poveznice
o Mehanizam komunikacije između preglednika i poslužitelja u domeni je protokola HTTP
o Nastavak opisa HTTP-a
o Zaokružuje opis CGI
o Primjer korištenja – obrada podataka upisanih u obrasce opisane HTML-om na strani preglednika

PHP
o Komentari – nekoliko stilova
o Više redova – /* ..... */
o Jedan redak – // .....
o Jedan redak – # .....
<?php ..... ?> - blokovi php naredbi
$ime_varijable – pristupanje varijabli (njenoj vrijednosti)

KOD SE INTERPRETIRA U 2 FAZE, NE PREVODI SE

o Ostale serverske tehnologije Weba
o ASP
o ASP.NET
o Java Servleti
o JSP

- Čemu služi DHTML?

o Promjena izgleda - svojstava elemenata stranice **nakon prikazivanja** stranice u pregledniku
o Validacija podataka iz obrazaca prije slanja poslužitelju
o Prikazivanje dijelova stranice u ovisnosti o kontekstu
o Izrada "bogatijeg" sučelja (izbornici, poruke...)
o Bolja uporabivost Web sjedišta
o Sve se i dalje događa u pregledniku, **nema komunikacije sa serverom**

- JavaScript

o Omogućuje
o Dinamičke promjene elemenata stranice
o Reagiranje na događaje
o Izvodi se unutar preglednika Weba
o *Sandbox* - ne može pisati po disku (osim *cookieja*)
o Podržan na svim novijim preglednicima Weba (s nekim razlikama)
o Skriptni jezik, interpretira se, ne prevodi

- Flash

o Alternativa DHTML-u
o Tehnologija za izradu aplikacija baziranih na multimedijskim sadržajima, slikama, animacijama
o Aplikacije/animacije mogu biti postavljene:
o Samostalno (*standalone*)
o Unutar Web stranice (kao jedan od elemenata stranice)
o Unutar Web stranice (kao jedini element stranice)
o Vlasništvo tvrtke Adobe (kupila Macromediu)

- Ključne riječi Weba 2.0

o blogging
o social bookmarking
o social networking
o wiki
o RSS
o podcasting
o tagging
o mashup
o Web API

- RIA – bogate internetske aplikacije – bogate sadržajem

SIGURNOST NA INTERNETU

Pamćenje stanja
o Procesi s **pamćenjem stanja** (*statefull*)
o Stanje u kojem je obrada se pamti
o Najčešće za kompleksne, dugotrajne obrade
o Često koriste transakcijsku okolinu
o Najčešće postoji oporavak od neuspješnog izvršenja dijela obrade
o Procesi **bez pamćenja stanja** (*stateless*)
o Stanje u kojem je obrada se ne pamti
o Najčešće za kratkotrajne obrade
o Često visokoefikasni, s malim vremenom čekanja
o Filozofija "obradi i zaboravi"

- HTTP – ne pamti stanja, stateless....
- web aplikacije često pamte stanja
- Sjednica (*session*) je trajna veza između korisnika
- ili klijenta i poslužitelja

- Rješenje problema pohrane sjedničkih podataka

o Pohrana sjedničkih podataka na strani klijenta
o **Kolačić** (*cookie*)
o **Skriveno polje** (*hidden field*)
o **Prepisivanje URL-a** (*URL Rewriting*)
o Pohrana sjedničkih podataka na strani poslužitelja
o **Memorija poslužitelja**
o **Baza podataka**
o **Datoteka**
o Pohrana sjedničkih podataka na strani poslužitelja kod višeposlužiteljskih sustava – dijeljenje podataka
o **Memory-to-memory replikacija** između poslužitelja
o **Dijeljena baza podataka**
o **Peer-to-peer**

- Kolačić (*cookie*)

o **Kolačić** (*cookie*)
o manji podatak koji se razmjenjuje između Web poslužitelja i Web preglednika, a pohranjuje se na klijentu (Web pregledniku)
o Sinonimi: *cookie, HTTP cookie, web cookie*
o Nastali iz termina "*magic cookie*" koji imaju smisao tiketa ili tokena
o Služe za pohranu informacija na klijentu kroz određeno vrijeme
o Pohrana u obliku **naziv – vrijednost**
o Koriste se za:
o Autentikaciju, praćenje, pohranu podataka i sl.
Kolačići mogu biti trajni(ako se postavi datum) i trenutni(brišu se zatvaranjem preglednika)

- Identifikator sjednice

o **Identifikator sjednice** ili **token sjednice** je jedinstveni identifikator koji jednoznačno određuje sjednicu unutar nekog sustava ili aplikacije
o Zapis identifikatora sjednice:
o Najčešće oblik velikog cijelog broja ili niza znakova
o Za generiranje često se koristi jednosmjerna matematička funkcija (*hash*)
o Naziv varijable u kojoj je pohranjen identifikator sjednice ovisi o implementaciji
o Primjeri naziva u raznim programskim jezicima
o Java i JSP: **JSESSIONID**
o PHP: **PHPSESSID**
o MS ASP: **ASPSESSIONID**

- Skrivena polja

o **Skrivena polja** (*hidden fields*)
o Uključivanje skrivene informacije u HTML obrasce
o Pomoću INPUT oznake
o <INPUT NAME="naziv" **TYPE="hidden"** VALUE="12">
o Podaci dohvatljivi kao parametri
o request.getParameter(String naziv) (primjer za Javu)
o Pohranjeni na klijentu – kao dio same HTML stranice
o Problemi:
o Ograničen sadržaj
o Nesigurni – podaci čitljivi u izvornom kodu stranice
o Ne bi se trebalo koristiti sigurnosne informacije (npr. broj kreditne kartice)

- Prepisivanje URL-a

o **Prepisivanje URL-a** (*URL Rewriting*) je dodavanje podataka URL-u kako bi se pri pozivu moglo prepoznati kojoj sjednici poziv pripada
o Može se koristiti kada klijent ne podržava druge mogućnosti (kolačići isključeni na klijentu)
o Korištenje
o Ili kolačići ili prepisivanje ili oboje – ali dosljedno!
o Koristi se specifično programsko kodiranje URL-a
o Potrebno "pratiti" informacije o sjednici
o Dodatna pažnja pri programiranju
o Ograničeno samo na dinamički generirane stranice

- Najbolja praksa

o Kombinirani pristup
o **Pohrana identifikatora sjednice na klijentu**
o **Pohranu preostalih sjedničkih podataka na poslužitelju**
o Prednosti kombiniranja
o Prijenos identifikatora sjednice i pohrana na klijentu
o Sigurnost pohrane svih ostalih podataka na poslužitelju
o Mogućnost korištenja ostalih prednosti velikih sustava (dijeljenje i trajna pohrana podataka sjednice)
o Najčešće korišteni način kod *enterprise* aplikacija

WEB APLIKACIJE

- REST

o REST ili REpresentational State Transfer
o Roy Fielding, 2000., doktorska disertacija
o Tip programske arhitekture za izgradnju raspodijeljenih sustava
o Mogu se koristiti raznolike tehnologije
o Sustavi koji prate REST principe - "RESTful"
o Najčešće su otvoreni, skalabilni, nadogradivi i jednostavni

- REST – tehnologije

o **URI** su najčešći odabir za ***imenice***
o Osnova sustava
o **HTTP metode** su najčešći odabir za ***glagole***

o HTTP je "najuspješniji" RESTful protokol
o Ugrađeni caching u protokol
o Autentikacija korištenjem HTTP autentikacijskih metoda
o Sigurni prijenos podataka pomoću HTTPS (HTTP preko SSL)
o **XML** je najčešći odabir za **tipove podataka**
o Odabir nekih drugih tehnologija samo s vrlo dobrim razlogom

- Izvedba WEB aplikacije

U izvedbi Web aplikacije izvršni modul:
o obrađuje podatke proslijeđene od strane korisnika
o održava i mijenja stanje aplikacije (za aplikacije sa stanjem)
o surađuje sa slojem podataka (najčešće baza podataka)
o stvara novo stanje korisničkog sučelja (HTML stranice) na temelju stanja aplikacije
Web aplikacija sa strane preglednika:
o skup logički povezanih HTML stranica tretiranih kao jedinstveni entitet (GUI aplikacije, mijenja se tijekom rada)

PROGRAMSKI JEZICI – OTVORENOST

- Definicije

o Prenosivost
o mogućnost izvršavanja istog programa na različitim platformama/operacijskim sustavima
o Skalabilnost
o mogućnost pisanja aplikacija za ugrađena računala i za poslužitelje istim alatima odnosno programskim sučeljem
o mogućnost izvođenja iste aplikacija na malim i velikim računalima
//nešto o tipovima podataka, čini mi se nevažno, predavanje 16

- Javini appleti

o "Živi" sadržaj na Webu – od samog početka Weba
o Javina aplikacija
o samostalni program
o Javin applet
o namijenjen izvođenju unutar preglednika koji podržava Javu
o Applet:
o dolazi preko mreže
o ima restrikcije pristupa resursima računala, mrežne komunikacije
o nasljeđuje razred *java.applet.Applet*
o životni vijek određen stranicom koja ga prikazuje AppletDemo.html

//primjeri, čini mi se nebitno, predavanje 16

Svojstva idealnog programskog jezika
o jednostavnost
o uniformnost
o ortogonalnost
o jasnoća
o sigurnost
o modularnost
o apstrakcija
o izražajnost
o fleksibilnost
o efikasnost
o sličnost sintakse s postojećim jezicima
o Idealan programski jezik dopušta programeru da se usredotoči na konkretan problem, a ne na "začkoljice" programskog jezika

Posebno značajno: automatsko upravljanje memorijom, određivanje tipova (C# i Java strongly typed), omogućiti kontrolu nad nepredviđenim događajima – iznimkama, višedretvenost
NORMIRANJE – par slajdova, predavanje 16 (kraj)

OBJEKTNO ORJENTIRANO PROGRAMIRANJE I JAVA

o **OBJEKT** – skup varijabli i s njima povezanih funkcija
o oponaša objekt iz stvarnog svijeta
o pamti stanje i modelira ponašanje
o **RAZRED, KLASA** – nacrt po kojem se stvara objekt
o ima članske varijable i svojstvene funkcije (metode), postoji kontrola pristupa varijablama i funkcijama
o **OBJEKT = primjerak, instanca razreda**

- Principi objektno orijentiranog programiranja

(OOP):
o **apstrakcija** (*abstraction*) - proces odbacivanja nepotrebnih detalja o objektu, a dodavanje (zadržavanje) onih podataka koji ga prikladno opisuju
o **enkapsulacija** (*encapsulation*), (učahurivanje) - skrivanje implementacije (korisniku se daje samo sučelje)
o **nasljeđivanje** (*inheritance*) – proširivanje razreda koji je već definiran – Java i C# ne dopuštaju višestruko nasljeđivanje, zato se implementira sučelje - interface
o **polimorfizam** (*polymorphism*) (višeobličje) -dijeljenje imena, ista imena za različite postupke. ***Overloading*** **– preopterećivanje -** statičko rješavanje, višeznačnost - unutar razreda isto ime za različite postupke uz različiti broj parametara **I** ***Overriding*** **– nadjačavanje,**Pravi polimorfizam, Dinamičko rješavanje,metoda unutar podklase s istim imenom i "**potpisom**",važnija je od metode iz nadređene klase, ne može se predvidjeti u fazi prevođenja već izvođenja
//neki primjer u javi, pročitati iz predavanja 17, negdje oko 20 stranice

- Stvaranje objekata

o Konstruktor:
o posebna metoda za stvaranje objekta
o istog naziva kao i klasa
o nema povratnog tipa
o Svaki Javin razred implicitno nasljeđuje razred *java.lang.Object*
o *java.lang.Object* definira konstruktor bez parametara
o svaki Javin razred ima podrazumijevani konstruktor, bez parametara, nije ga potrebno navoditi

- Pravokutnik pk1=**new** Pravokutnik();
- Pravokutnik pk2=**new** Pravokutnik("3 4");

Operator ***new****:*
o alocira memorijski prostor za novi objekt
o poziva konstruktor
o vraća referencu na novostvoreni objekt

```
public class Citac
{
        public static void main(String [] args)
        {
        ...
        }
}
```

o Metoda **main**:
o poziva se prilikom pokretanja programa
o prima argumente iz komandne linije preko polja objekata tipa String (*args* u gornjem primjeru)

Pokretanje
o Svaki razred u posebnoj datoteci, istog imena kao i ime razreda (s nastavkom *.java*).
o Prevoditelj se pokreće naredbom:
`'javac <datoteke sa izvornim kodom>'`
o nakon čega nastaju *.class* datoteke koje sadrže prevedeni Java program.
o Program se pokreće naredbom
`'java <razred s metodom main()> <parametri>'`