

Интернет програмирање



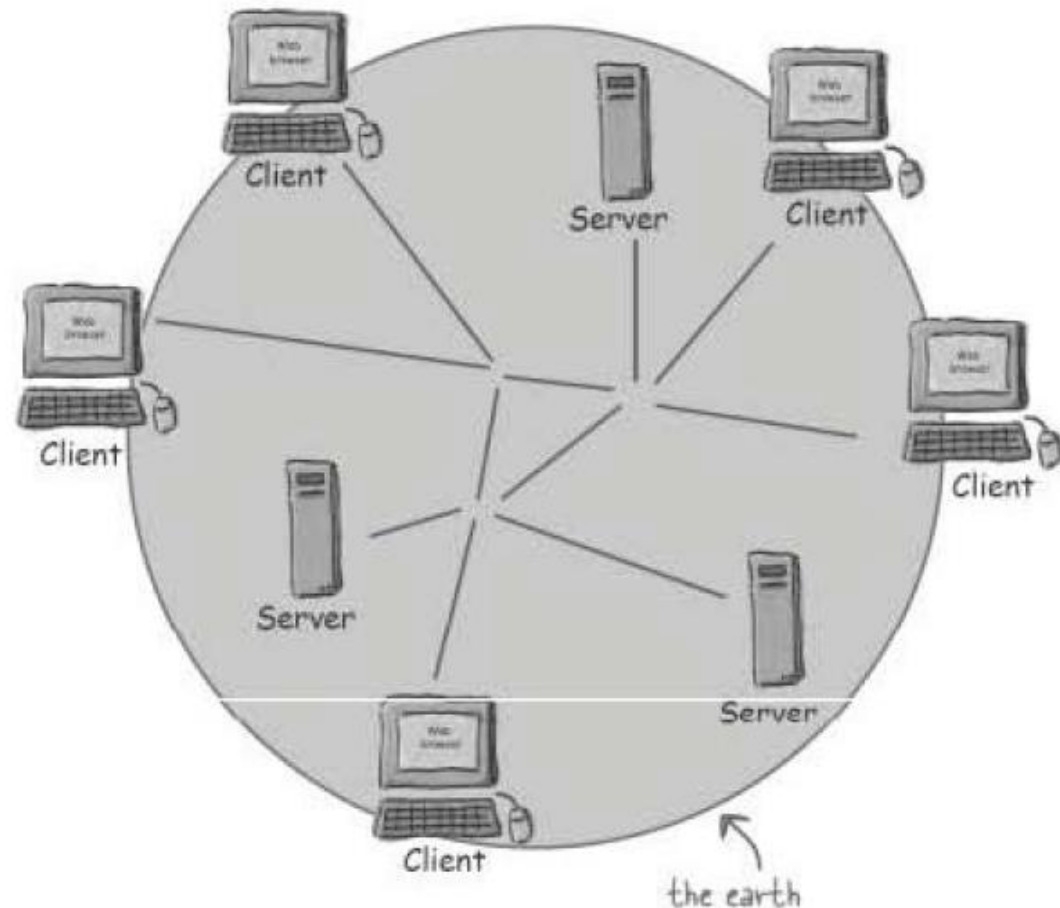


Веб сервери



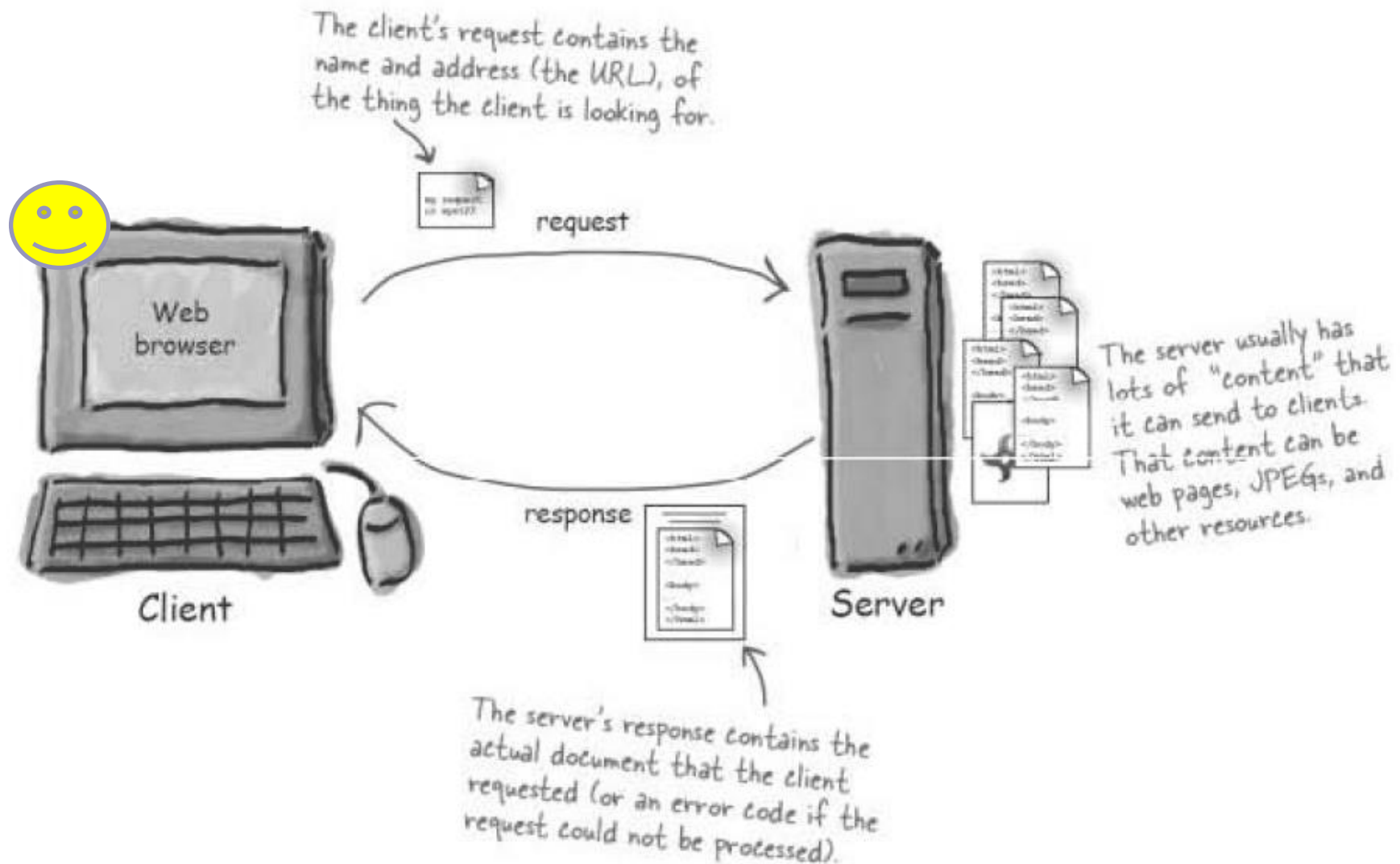
Веб

- Веб се састоји од огромног броја клијената са прегледачима (као што су Chrome, Mozilla, Yandex, Safari итд.) и од сервера (који користе веб сервере као што су Apache, JBoss, Tomcat, Microsoft IIS итд.) који су међусобно повезани кроз жичане и безжичне мреже.





Функционисање веб сервера



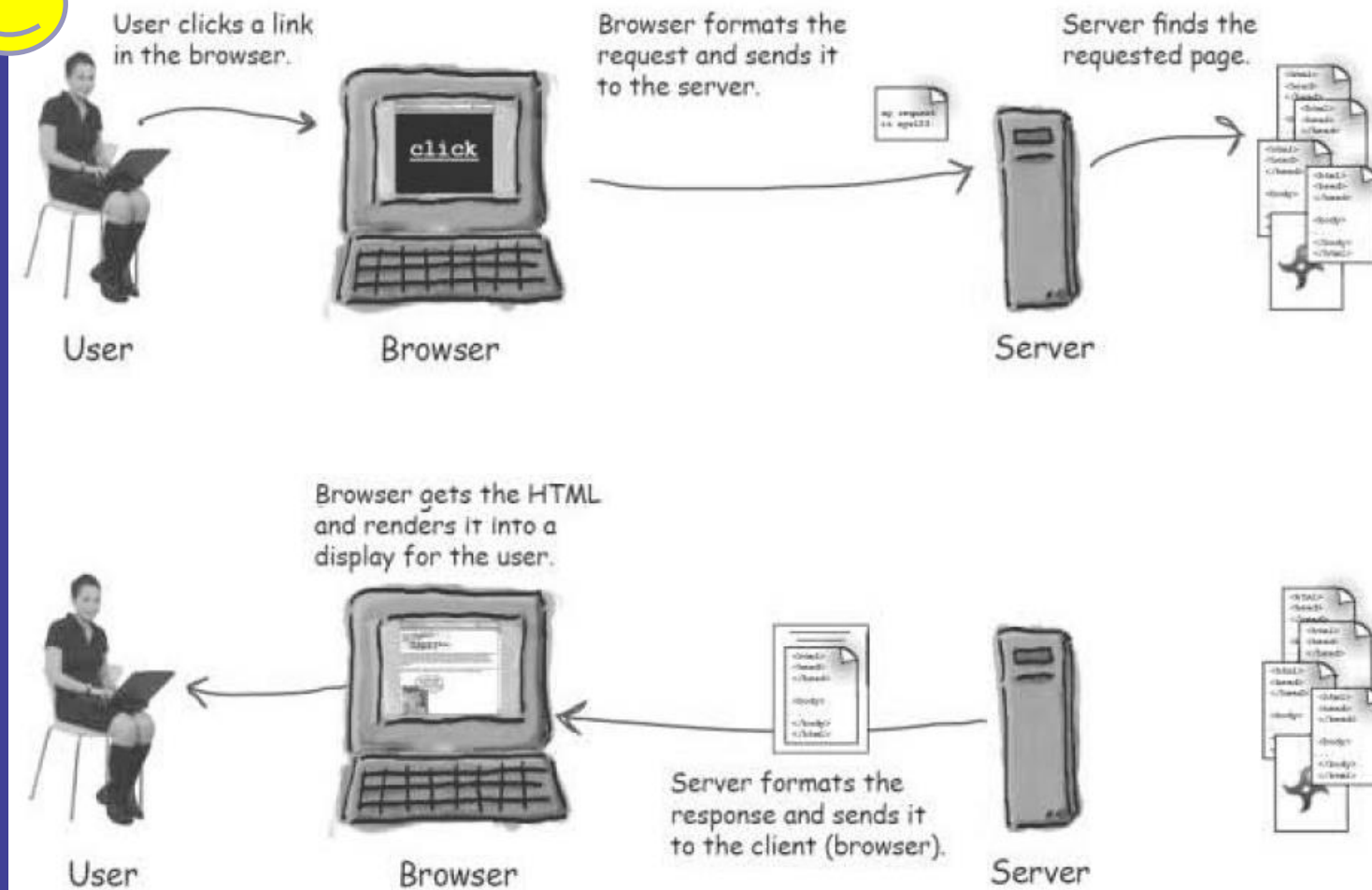


Функционисање веб сервера (2)

- Корисник преко веб прегледача шаље захтев за ресурсом
- Веб сервер прихвата захтев, проналази тражени ресурс и њега шаље кориснику
 - Ресурс може бити HTML страна, слика, PDF документ или нешто четврто – што год да је у питању, клијент захтева ресурс, а сервер шаље клијенту ресурс који је захтеван
 - У случају када нема захтеваног ресурса, генерише се грешка „404 Not found“
- Термин „сервер“ означава и сам рачунар (тј. хардвер) и програм који представља веб сервер (тј. софтвер)
 - Ако из контекста није јасно да ли се ради о хардверу или софтверу, то ће бити додатно истакнуто



Функционисање веб клијента





Функционисање веб клијента (2)

- Када се говори о клијенту, има се у виду корисник, али и веб прегледач – апликација коју корисник користи за сурфовање. Корисник преко веб прегледача шаље захтев за ресурсом
 - Дакле, прегледач је софтвер (нпр. Netscape, Chrome, Mozilla, Yandex, Safari, Edge, Opera и сл.) који комуницира са веб сервером. Осим послова комуникације, прегледач треба да интерпретира HTML код и да исцрта веб стране за корисника
- Ако се експлицитно не наведе другачије, надаље ће термин „клијент“ ће обухватити и софтвер (тј. прегледач) и човека (тј. корисника)
 - Другим речима, клијент је апликација - прегледач која обавља оно што корисник захтева да се уради



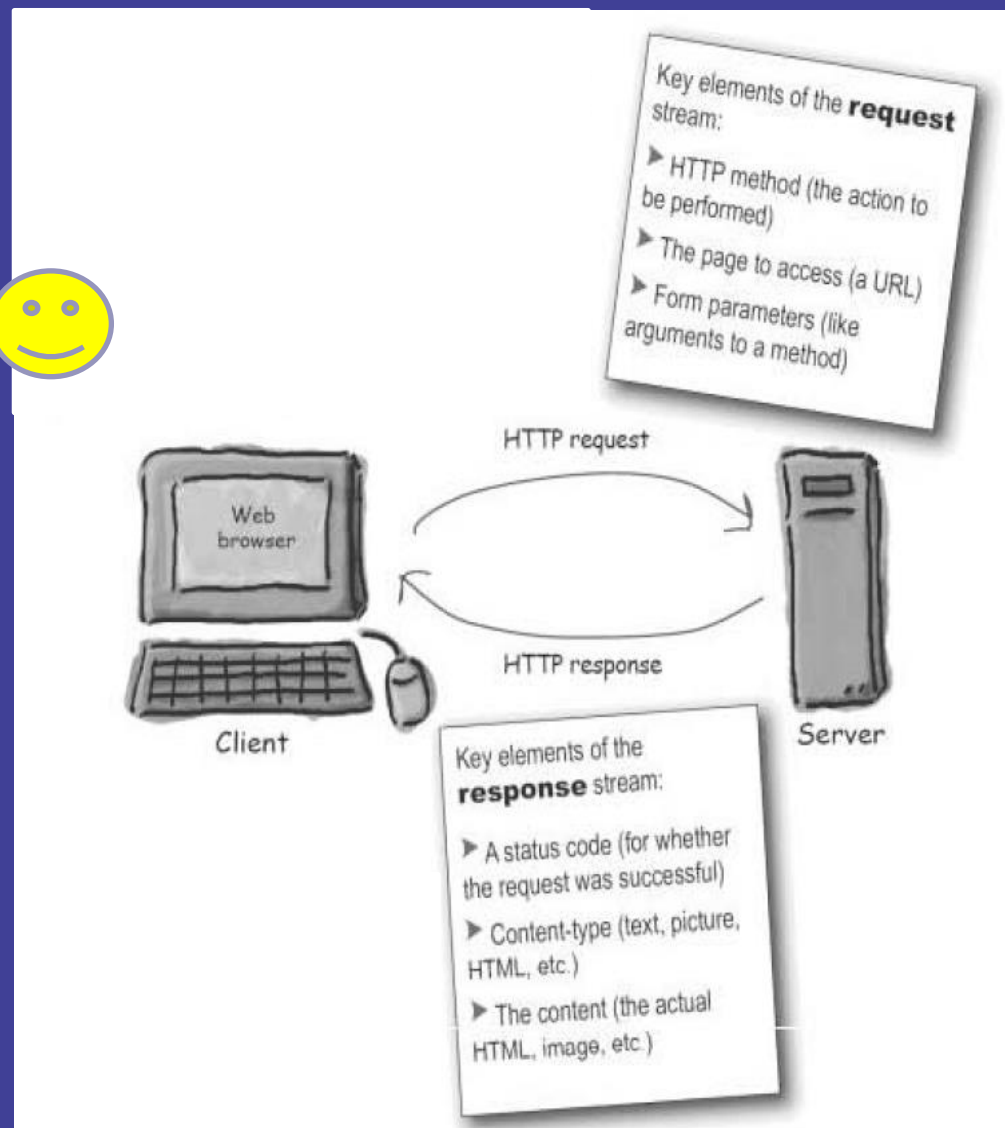
Клијент и сервер користе HTML и HTTP

- Када сервер шаље одговор на захтев, он прегледачу обично шаље садржај датог типа, тако да прегледач може да прикаже добијени одговор.
- Често је одговор који сервер шаље клијенту секвенца знакова који представљају документ у **HTML** формату. Тај HTML документ потом бива приказан од стране прегледача
 - Језик за означавање HTML (прецизније, HTML 5), је описан у претходним предавањима
- Највећи део конверзације између клијената и сервера се реализује коришћењем **HTTP** протокола
 - Клијент тада шаље HTTP захтев, а сервер одговара са HTTP одговором.
 - Када сервер пошаље HTML страну клијенту, он то ради коришћењем HTTP протокола.



HTTP протокол

- HTTP протокол се извршава преко TCP/IP протокола
- То је мрежни протокол са карактеристикама које се односе на веб, али он се ослања на TCP/IP протокол ради обезбеђења потпуног преноса захтева и одговора са једног места на друго
- Суштина HTTP конверзације је једноставна секвенца захтев/одговор: прегледач **захтева** а сервер **одговара**





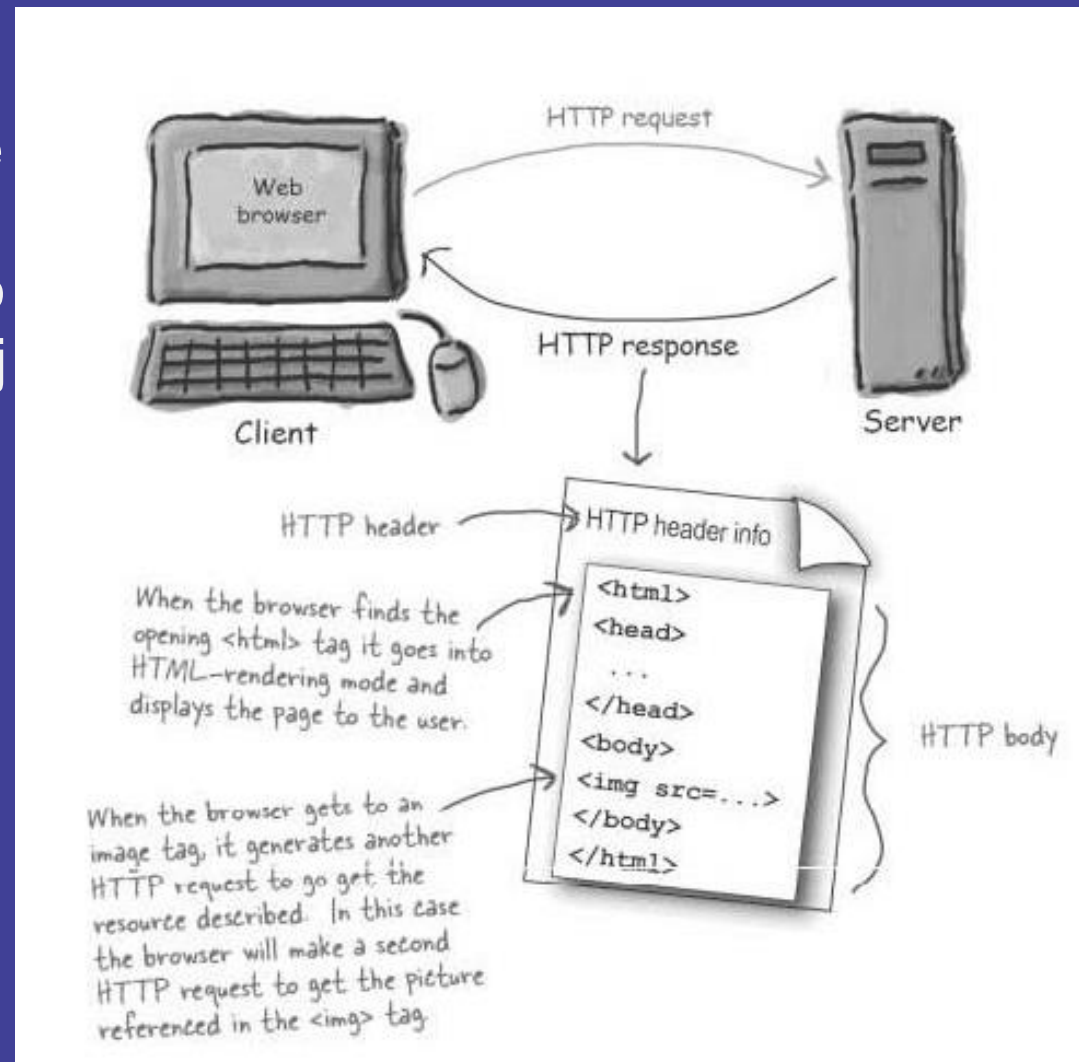
HTTP протокол (2)

- Сам HTTP протокол је описан IETF документом RFC 2616
- Веб сервери Apache, JBoss, Tomcat, Microsoft IIS и сл. су примери сервера који обрађују HTTP захтеве
- Прегледачи Netscape, Chrome, Mozilla, Yandex, Safari, Edge, Opera и сл. обезбеђују кориснику да генерише HTTP захтев, упути га према серверу и на адекватан начин прикаже HTTP одговор који добије од сервера
- Карактеристике HTTP протокола:
 1. HTTP не одржава конекцију (connectionless)
 2. HTTP је независан од медијума (media independent)
 3. HTTP не подржава стања (stateless)



HTML је део HTTP одговора

- HTTP одговор може садржавати HTML
- HTTP додаје информације о заглављу на почетак садржаја који се враћа као одговор, какав год садржај био у питању
- Прегледач користи информације из заглавља као помоћ у процесирању HTML садржаја
- Дакле, HTML се може посматрати као садржај уметнут у HTTP одговор





HTTP метод

- HTTP захтев садржи назив метода у свом заглављу
- Назив метода говори серверу о каквој се врсти захтева ради и како ће бити форматиран остатак поруке
- HTTP протокол подржава следеће методе:
 - GET - користи се за преузимање информација са датог сервера на основу дате адресе. Захтеви који користе метод GET треба само да прибављају податке, а никако не треба да их мењају
 - HEAD - је врло сличан GET методу, са тим што се тело поруке не враћа клијенту (враћа се само статусна линија и заглавље). Метод се може користити ради утврђивања да ли је линк измењен у односу на претходно стање - измењено стање се тестира упоређивањем информација послатих у заглављу захтева са информацијама из заглавља генерисаног одговора
 - POST - се користи за захтев да се пошаљу подаци HTTP серверу коришћењем HTML форме



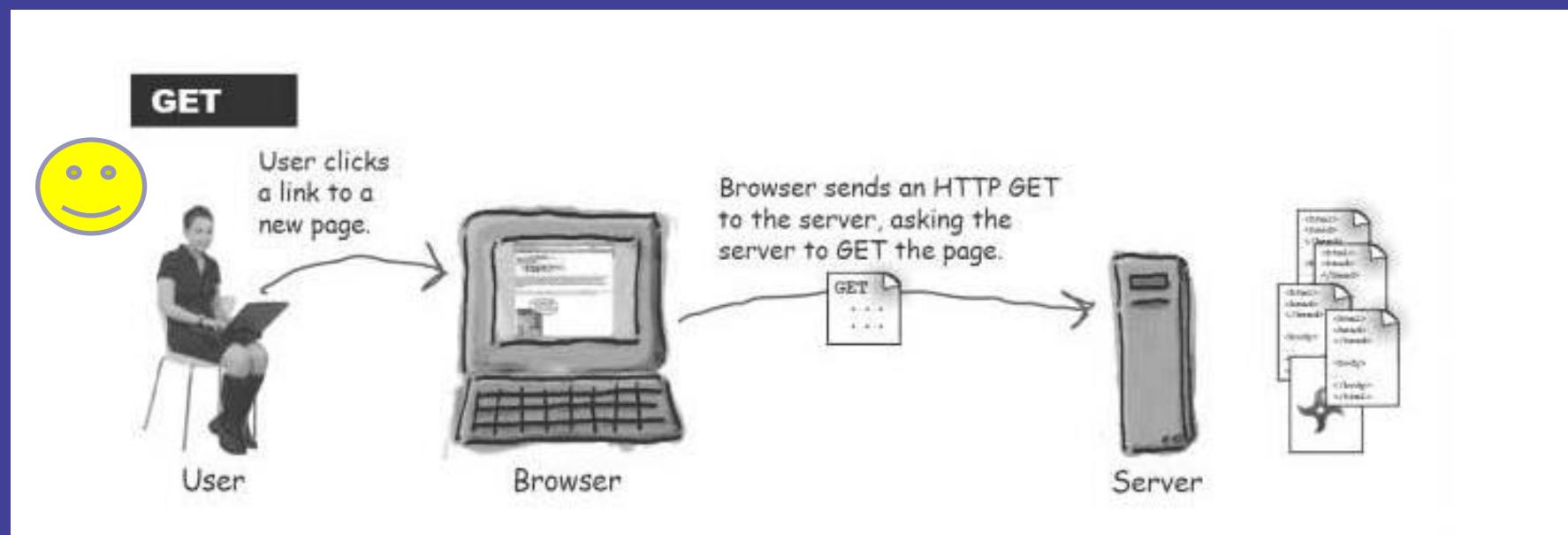
HTTP метод (2)

- PUT - користи се за захтев HTTP серверу да се подаци послати у оквиру захтева сместе на месту наведеног ресурса
- DELETE - користи се за захтев серверу да се уклони наведени ресурс
- TRACE – извршава тестирање повратне поруке дуж путање којом се захтев креће према циљном ресурсу
- OPTIONS – описује опције комуникације за циљни ресурс
- CONNECT – обезбеђује тунелску комуникацију према серверу одређеним са датом адресом



Методи GET и POST

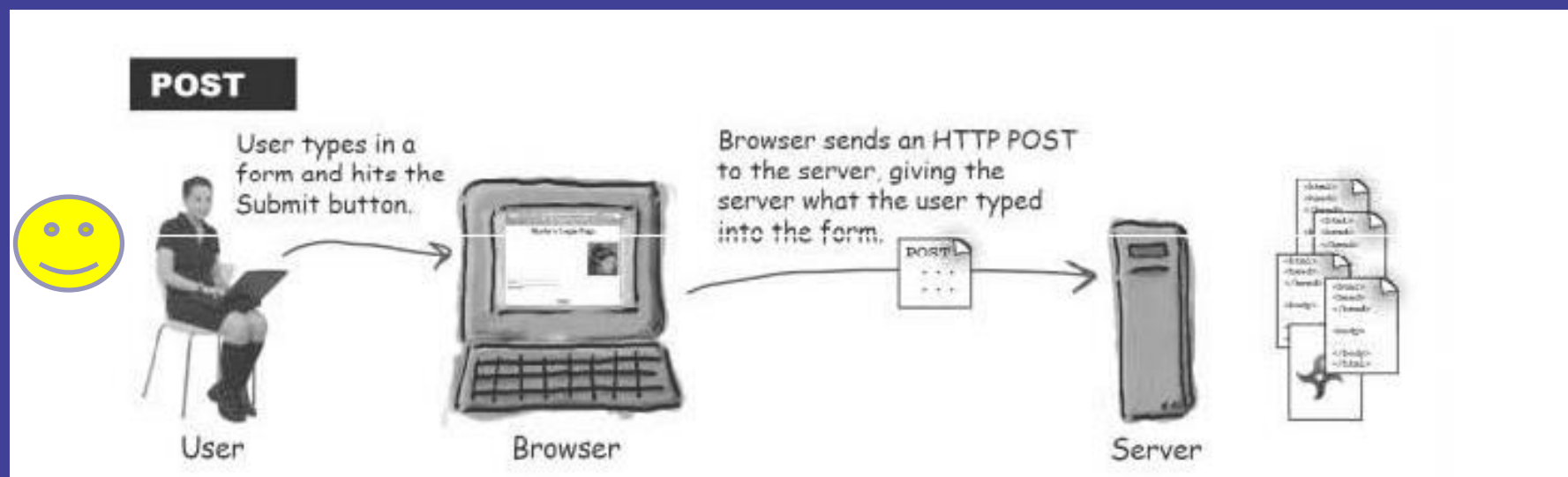
- Метод GET је најједноставнији HTTP метод
- Метод GET тражи од сервера да прибави ресурс и да га врати позиваоцу
- Ресурс може бити HTML страна, PDF документ, JPG слика ...
- Сврха метода GET је да се добије ресурс од сервера





Методи GET и POST (2)

- Метод POST је моћнији од метода GET
- Коришћењем овог метода може се захтевати нешто од и истовремено слати податке на сервер (па сервер може процесирати приспеле податке)





Методи GET и POST (3)

- Подаци се могу слати на сервер и помоћу метода GET и помоћу метода POST
 - Укупан број знакова који се помоћу могу послати метода GET је много мањи од броја знакова који се могу послати преко метода POST, а то горње ограничење зависи од врсте веб сервера и прегледача
 - Подаци који се шаљу коришћењем метода GET се налепљују на адресу у адресној линији прегледача, па је све што се тим путем шаље на сервер директно видљиво кориснику (и самим тим подложније манипулацији)
 - Корисник не може поставити маркер на страницу где је садржај форме прослеђен методом POST, а може ако је за прослеђивање коришћен метод GET

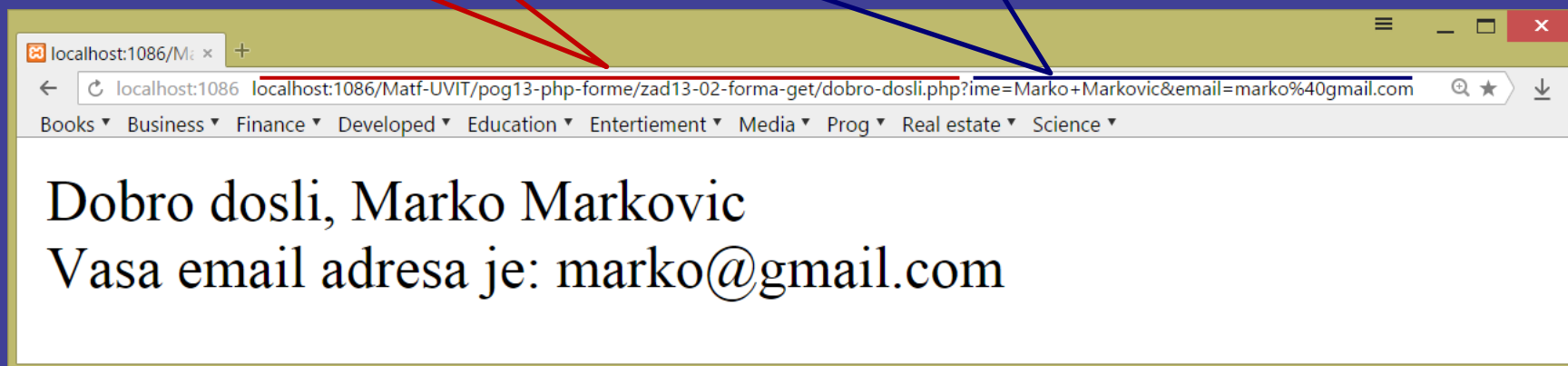


Методи GET и POST (4)

- Илустрација слања података на сервер и помоћу метода GET

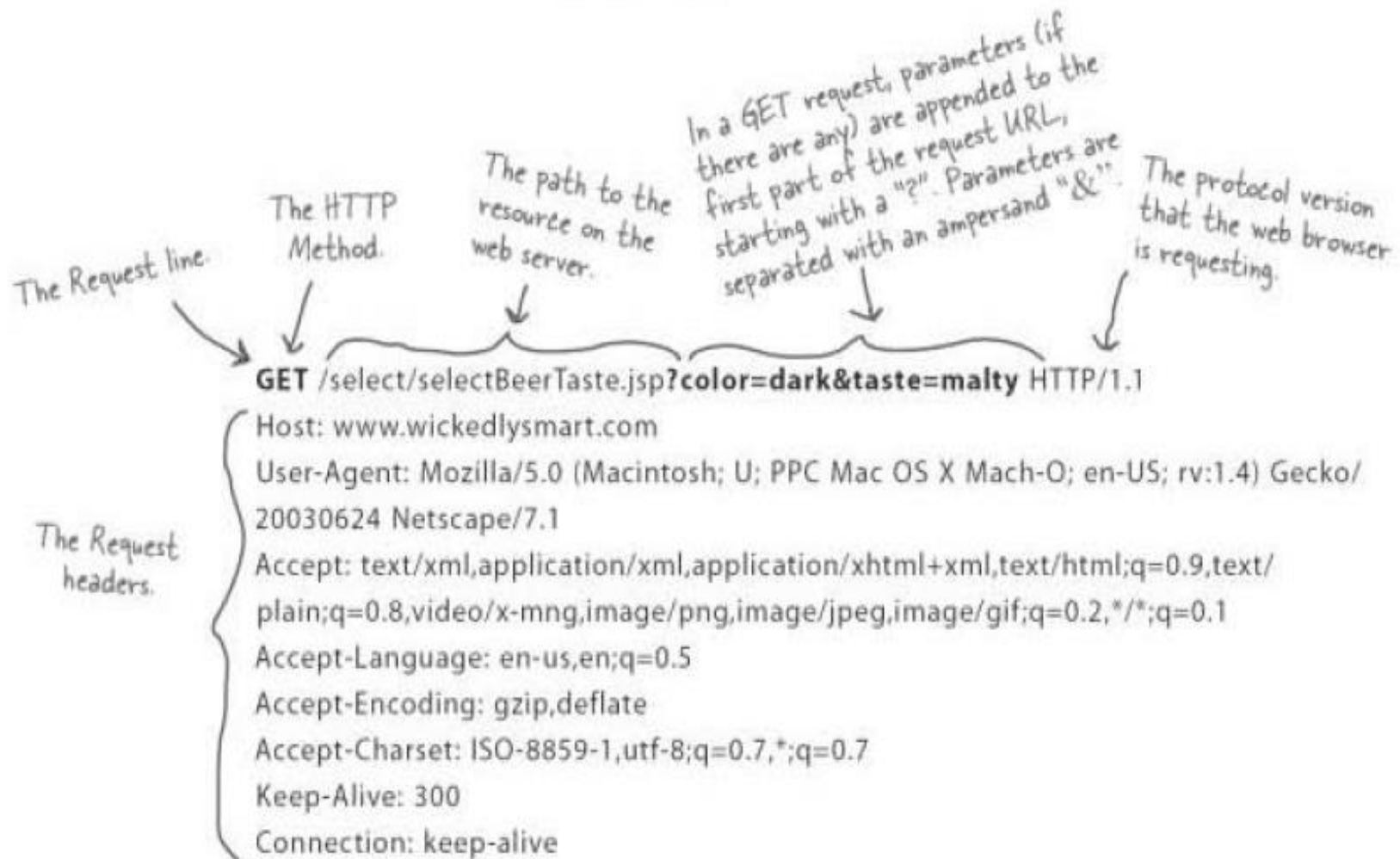
Оригинални URL без додатних параметара

Симбол ? раздваја путању од параметара, а симбол & раздваја параметре.
Параметри се дефинишу у облику име = вредност



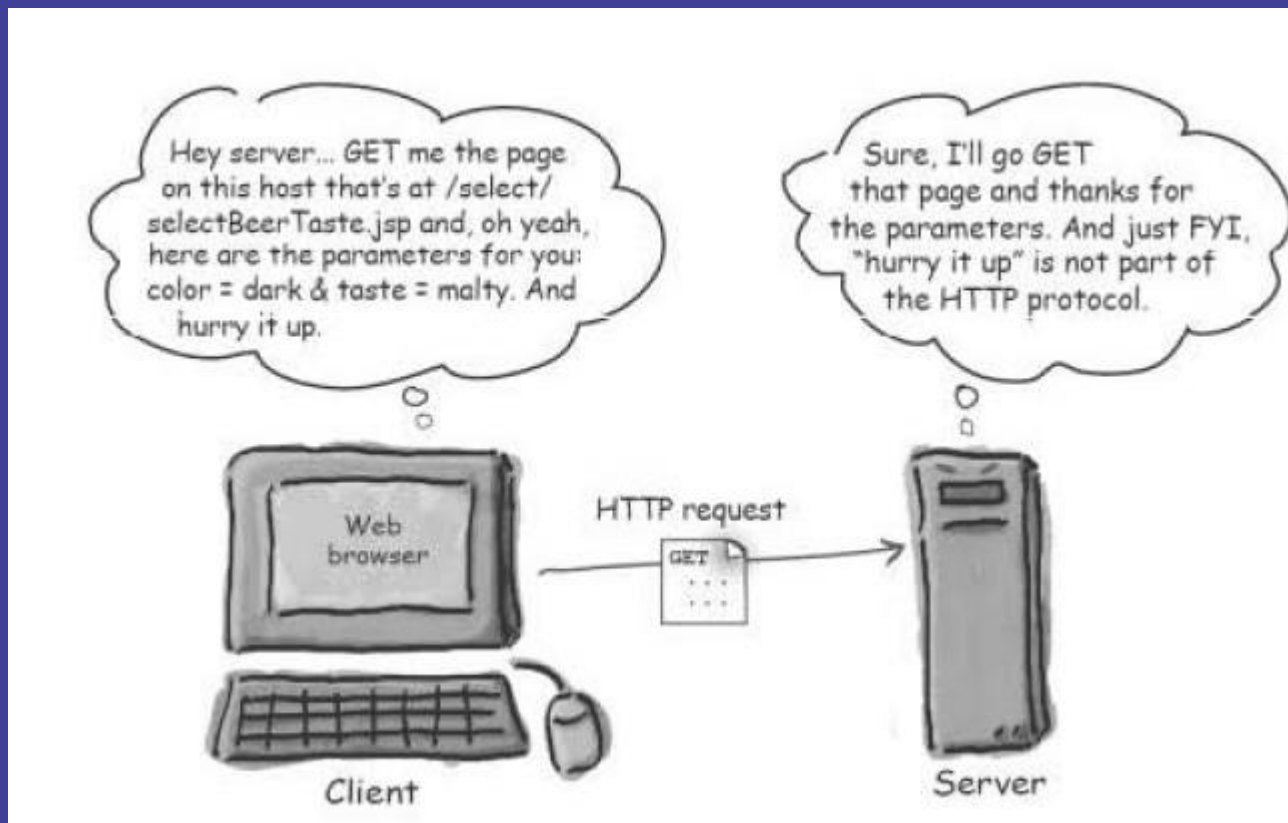


Анатомија GET захтева



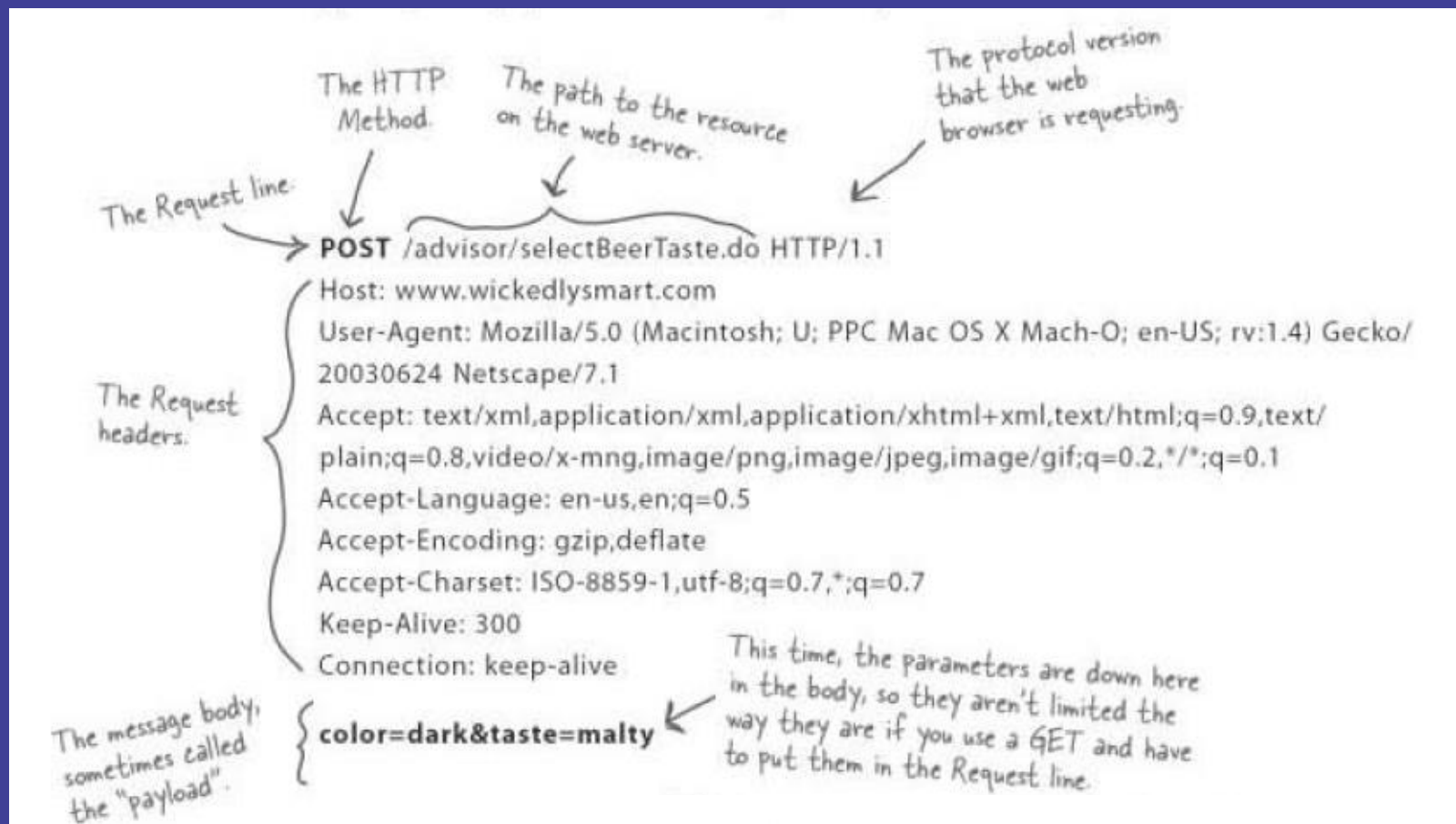


Анатомија GET захтева (2)



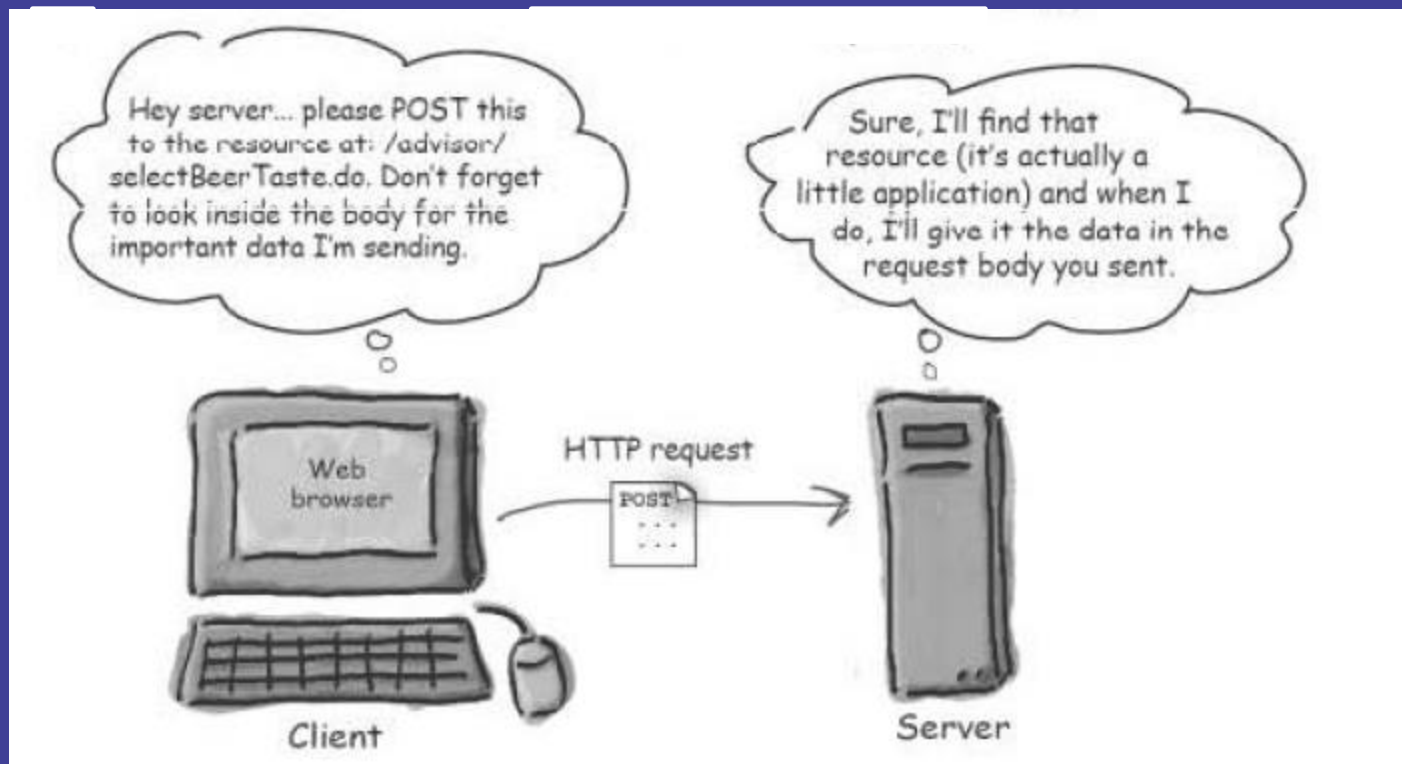


Анатомија POST захтева





Анатомија POST захтева (2)

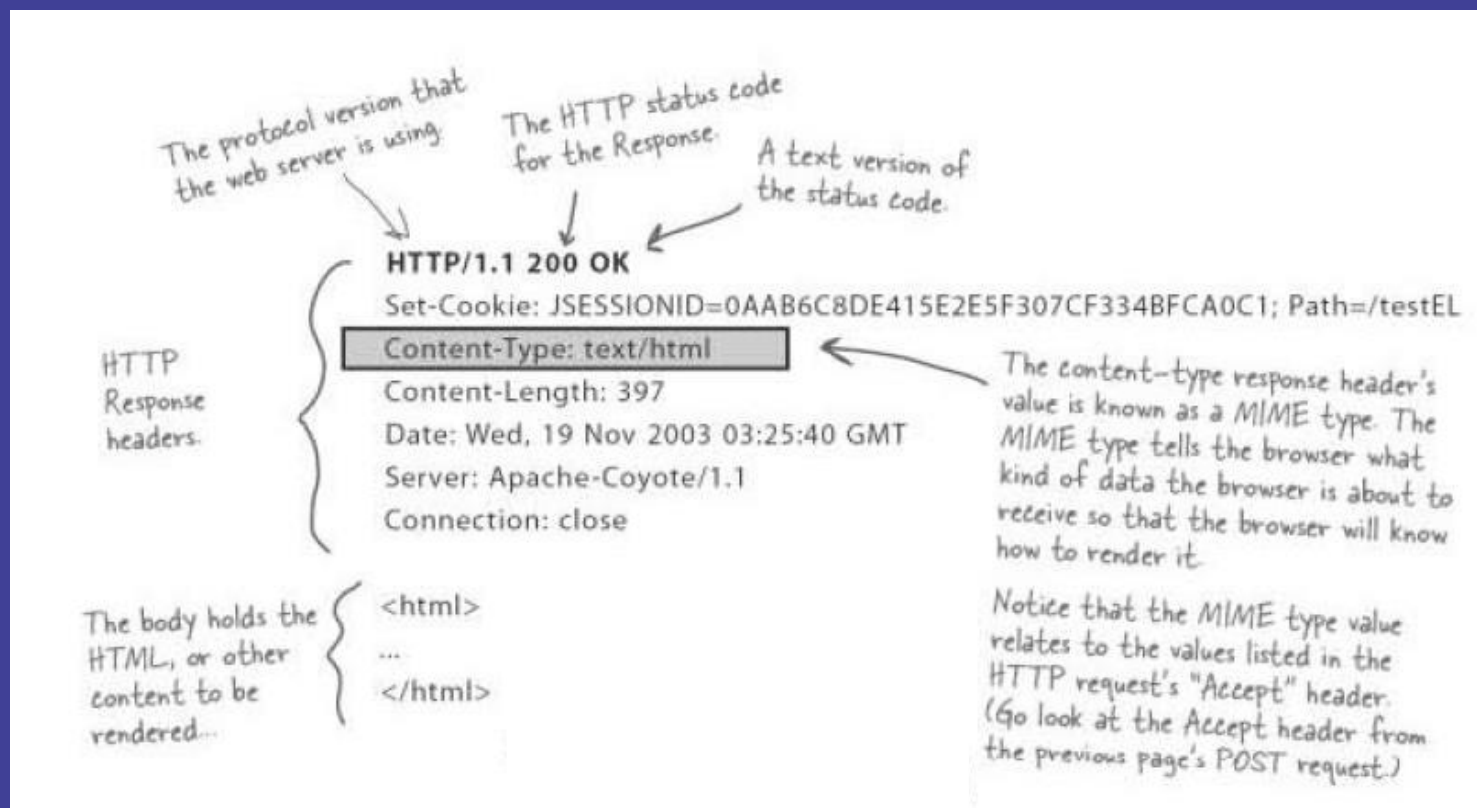




Анатомија HTTP одговора

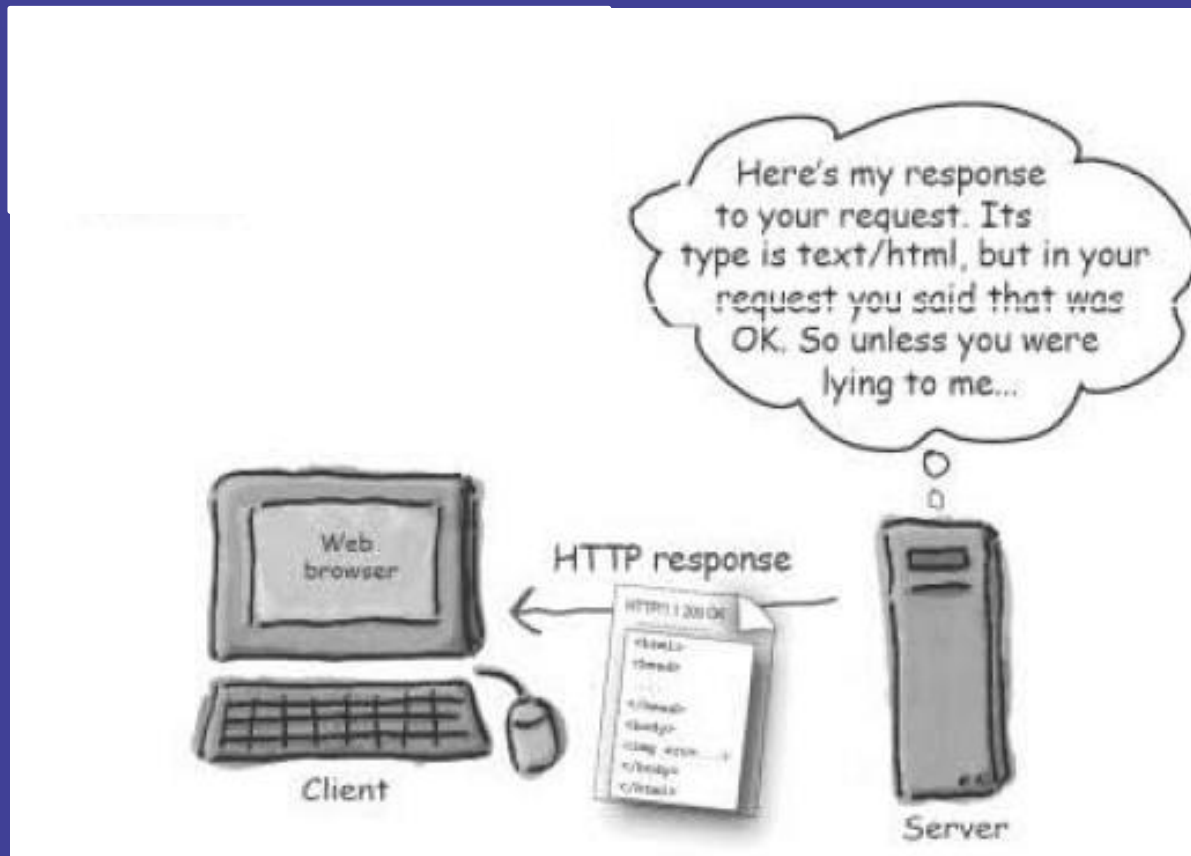
- HTTP одговор садржи заглавље и тело

○ Информације у заглављу говоре прегледачу који је протокол коришћен, да ли је захтев био успешан и која врста садржаја се налази у телу захтева, а тело садржи сам садржај који прегледач приказује



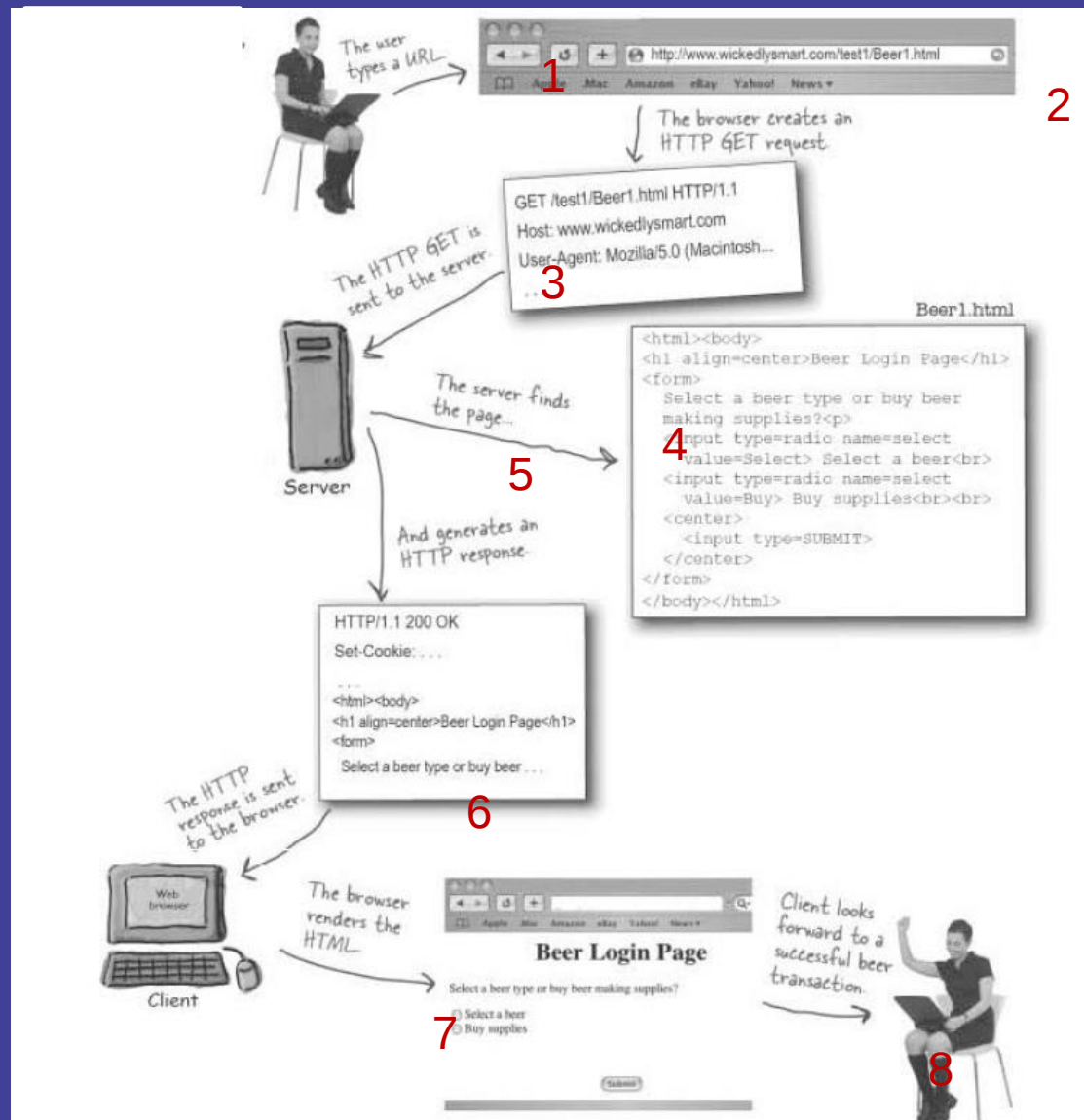


Анатомија HTTP одговора (2)





HTTP захтев и HTTP одговор





URL

- Сваки ресурс на вебу има своју јединствену адресу, у URL (Uniform Resource Locator) формату

Protocol: Tells the server which communications protocol (in this case HTTP) will be used.

Port: This part of the URL is optional. A single server supports many ports. A server application is identified by a port. If you don't specify a port in your URL, then port 80 is the default, and as luck would have it, that's the default port for web servers.

Resource: The name of the content being requested. This could be an HTML page, a servlet, an image, PDF, music, video, or anything else the server feels like serving. If this optional part of the URL is left out, most web servers will look for index.html by default.

Server: The unique name of the physical server you're looking for. This name maps to a unique IP address. IP addresses are numeric and take the form "xxx.yyy.zzz.aaa". You can specify an IP address here instead of a server name, but a server name is a lot easier to remember.

Path: The path to the location, on the server, of the resource being requested. Because most of the early servers on the web ran Unix, Unix syntax is still used to describe the directory hierarchies on the web server.

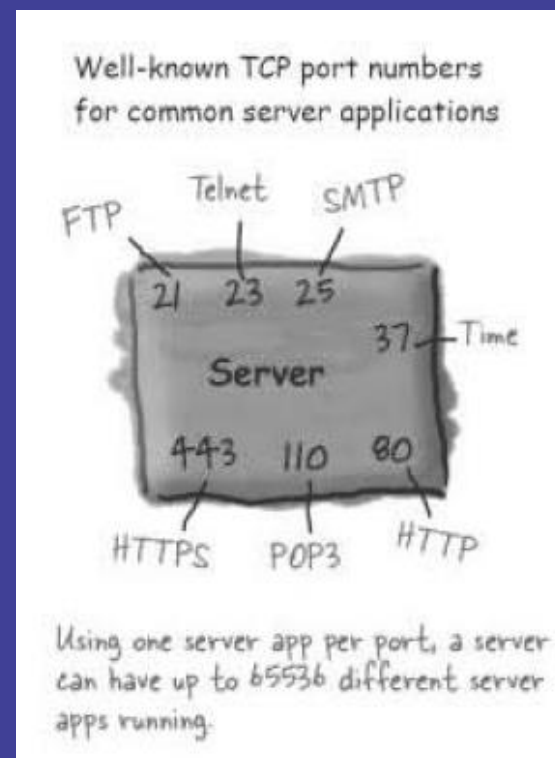
Not shown:
Optional Query String
Remember, if this was a GET request, the extra info (parameters) would be appended to the end of this URL, starting with a question mark "?", and with each parameter (name/value pair) separated by an ampersand "&".

`http://www.wickedlysmart.com:80/beeradvice/select/beer1.html`



TCP порт

- Различите апликације користе различите портове
 - HTTP сервер подразумевано користи порт 80
 - Telnet сервер подразумевано користи порт 23
 - FTP сервер подразумевано користи порт 21
 - POP3 порт 110, а SMTP порт 25
 - Сервер за време порт 37
- Порт представља једнозначни идентификатор – број између 0 и 65535
- Порт представља логичку везу између конкретног софтвера и хардвера на ком се тај софтвер извршава
- Порт не представља место на које се прикључује уређај, већ број који означава приступну тачку за комуникацију са сервером





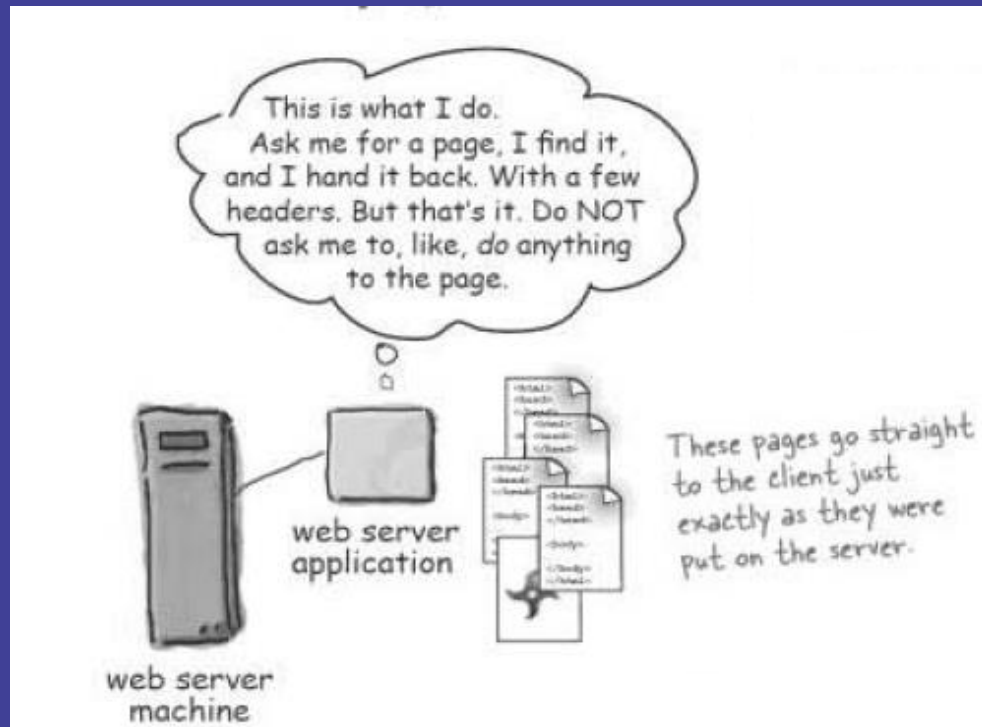
TCP порт (2)

- Ако не би било броја порта, онда сервер не би знао са којом клијентском апликацијом комуницира, нити по ком се протоку та комуникација реализује
 - Могло би се догодити, на пример, да прегледач, уместо да комуницира са веб сервером, покушава успостави комуникацију са сервером електронске поште
- TCP портови између 0 и 1023 су резервисани за познате сервисе и не препоручује се да се ти портови користе за „нове“ серверске програме
- Ако се извршавају серверски програми на рачунарској мрежи компаније, тада са администратором система треба проверити који су портови слободни, а који заузети



Статичке веб стране

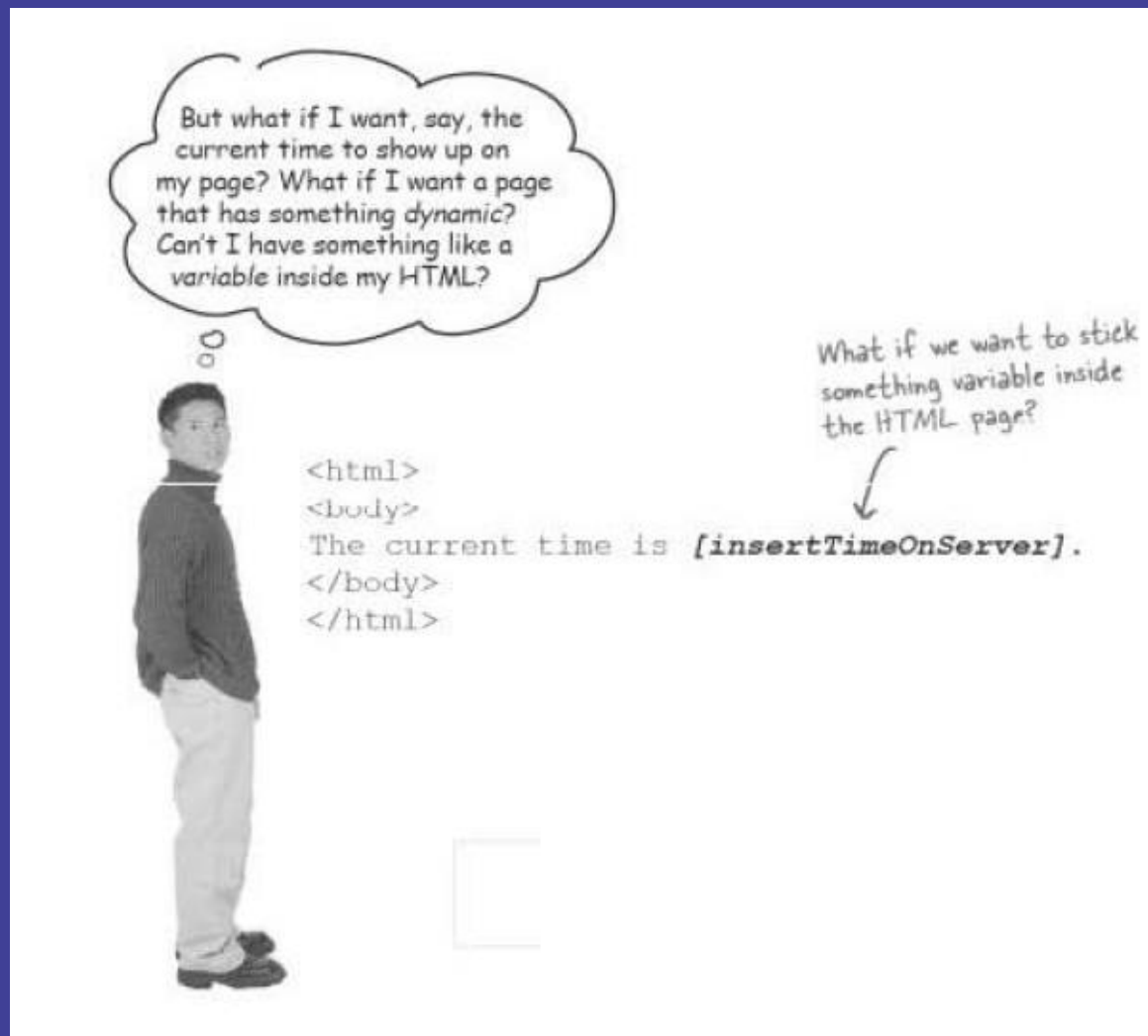
- Статичка веб страна се налази у директоријуму на веб серверу
 - Веб сервер такву страну само пронађе и проследи клијенту, баш онакву каква је на серверу
 - Сваки од клијената добија потпуно исти садржај као одговор





Статичке веб стране (2)

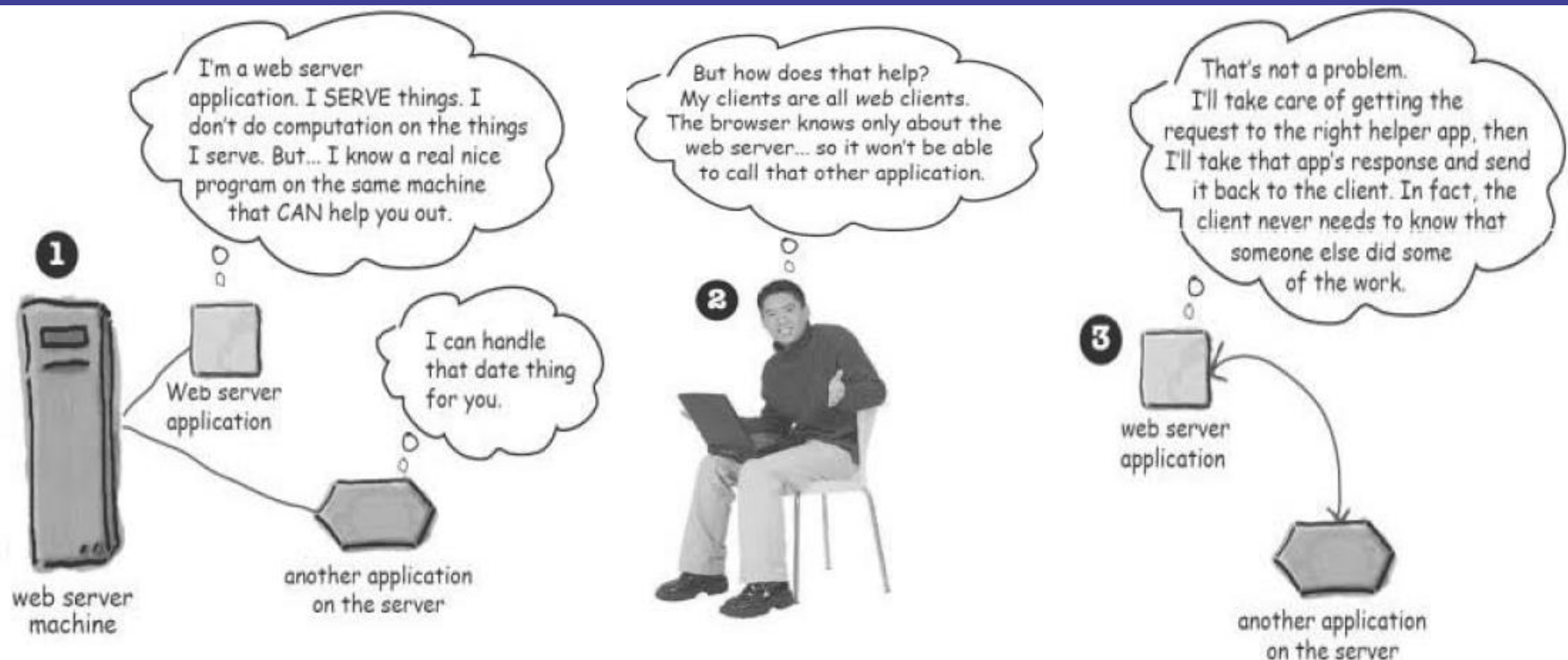
- Шта радити када треба обезбедити да се веб страна динамички мења?





Динамичке веб стране

- Сам веб сервер опслужује само статичке стране, али се може користити посебна помоћна апликација, са којом комуницира веб сервер, а која креира динамички садржај





Динамичке веб стране (2)

- Динамички садржај може бити било шта: датум и време са сервера, списак датотека у директоријуму, случајно изабрана слика итд.
- Динамички садржај не постоји све док не стигне захтев
- По приспећу захтева, помоћна апликација „креира“ HTML а онда веб сервер тај HTML „спакује“ у одговор

Уместо статичног

```
<html>
<body>
The current time is
always 4:20 PM
on the server
</body>
</html>
```

Треба да се добије динамичан садржај

```
<html>
<body>
The current time is
[insertTimeOnServer]
on the server
</body>
</html>
```

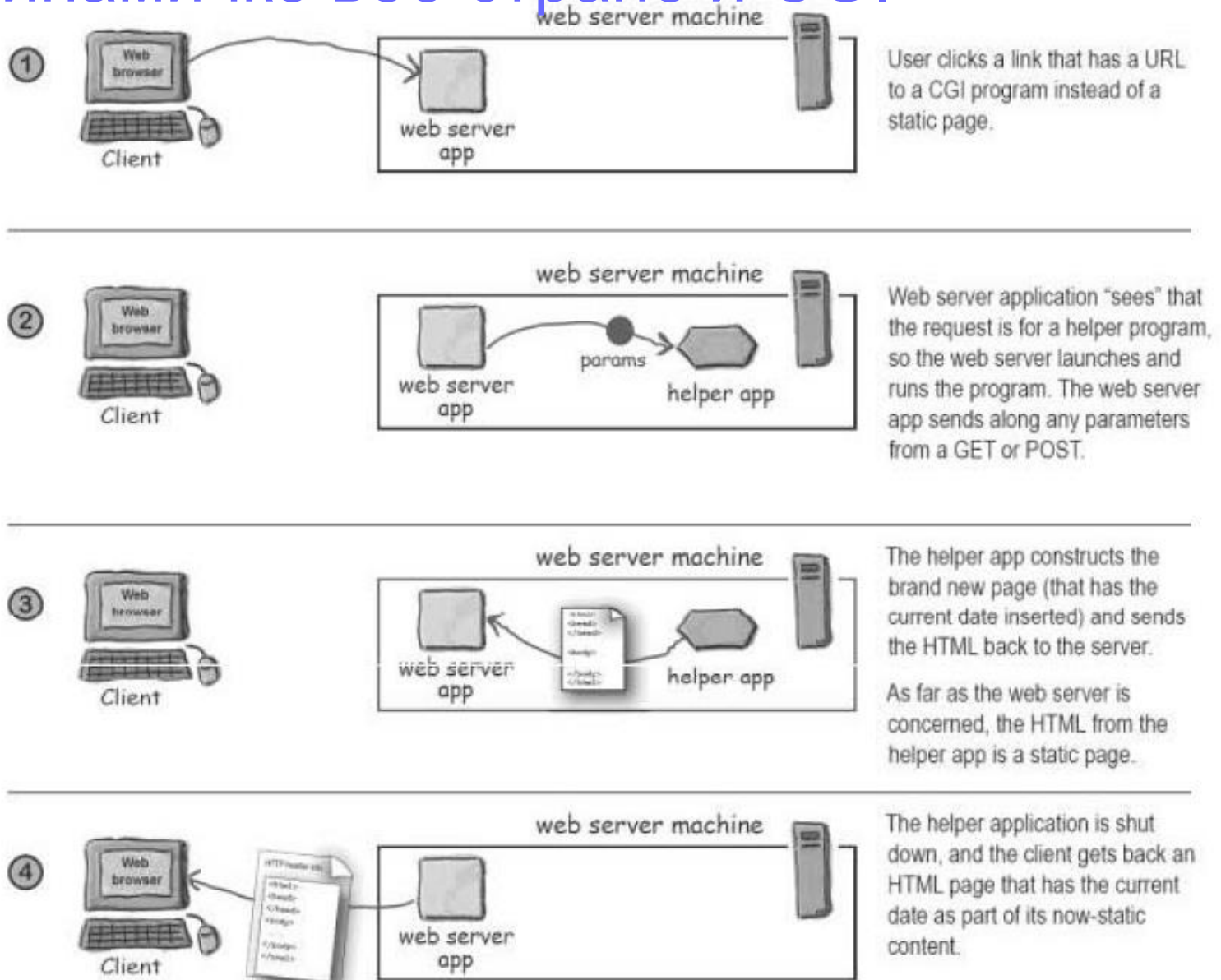


Динамичке веб стране (3)

- Када корисник проследи серверу податке са форме, тада је за процесирање прослеђених података (чување података у бази, ради генерисање одговора на основу података прослеђених уз захтев итд.) неопходно коришћење помоћне апликације
- Када сервер препозна да се захтев односи на помоћну апликацију, тада и прослеђене параметре проследи помоћној апликацији, па та апликација генерише одговор за који се потом проследи клијенту



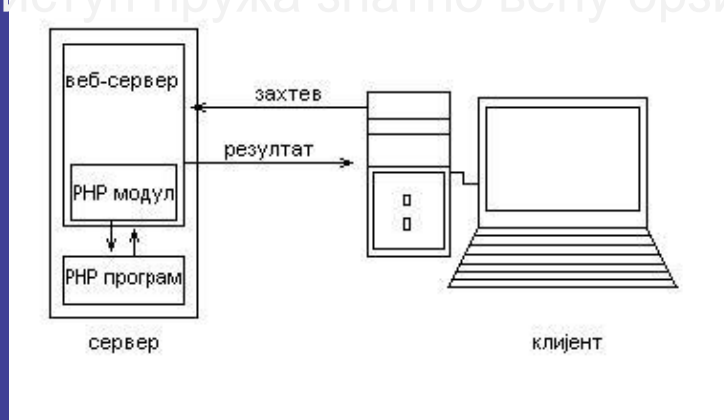
Динамичке веб стране и CGI





Динамичке веб стране и PHP

- PHP интерпретатор може радити по CGI принципу
 - Интерпретатор постоји као екстерна апликација
 - Та екстерна апликација се позива да изврши дату скрипту (дати PHP програм) сваки пут кад буде захтевана од неког корисника
- PHP интерпретатор може бити инсталиран и као модул веб сервиса
 - Интерпретатор је ближе повезан са веб сервисом
 - Интерпретатор је на тај начин увек уčitан у меморију, па се не мора позивати спољашњи програм
 - Овакав приступ пружа знатно већу брзину извршавања





Динамичке веб стране и PHP (2)

- Уобичајен сценарио по ком се извршавају PHP скрипте
 1. клијент (корисник Интернета који користи неки прегледач) захтева PHP страницу са сервера (рачунара)
 2. сервер прослеђује захтев веб серверу (програм на серверу)
 3. веб сервер препознаје да се тражи PHP датотека
 4. веб сервер не шаље његов садржај захеваног ресурса клијенту, него га извршава као програм помоћу PHP модула
 5. излазни текст програма (стандардни излаз тј. резултат рада PHP модула) се шаље клијенту као резултат захтева
 6. прегледач препознаје врсту резултата (HTML код, слика, PDF садржај, архива итд.)
 7. прегледач приказује резултат клијенту на одговарајући начин



Apache веб сервер



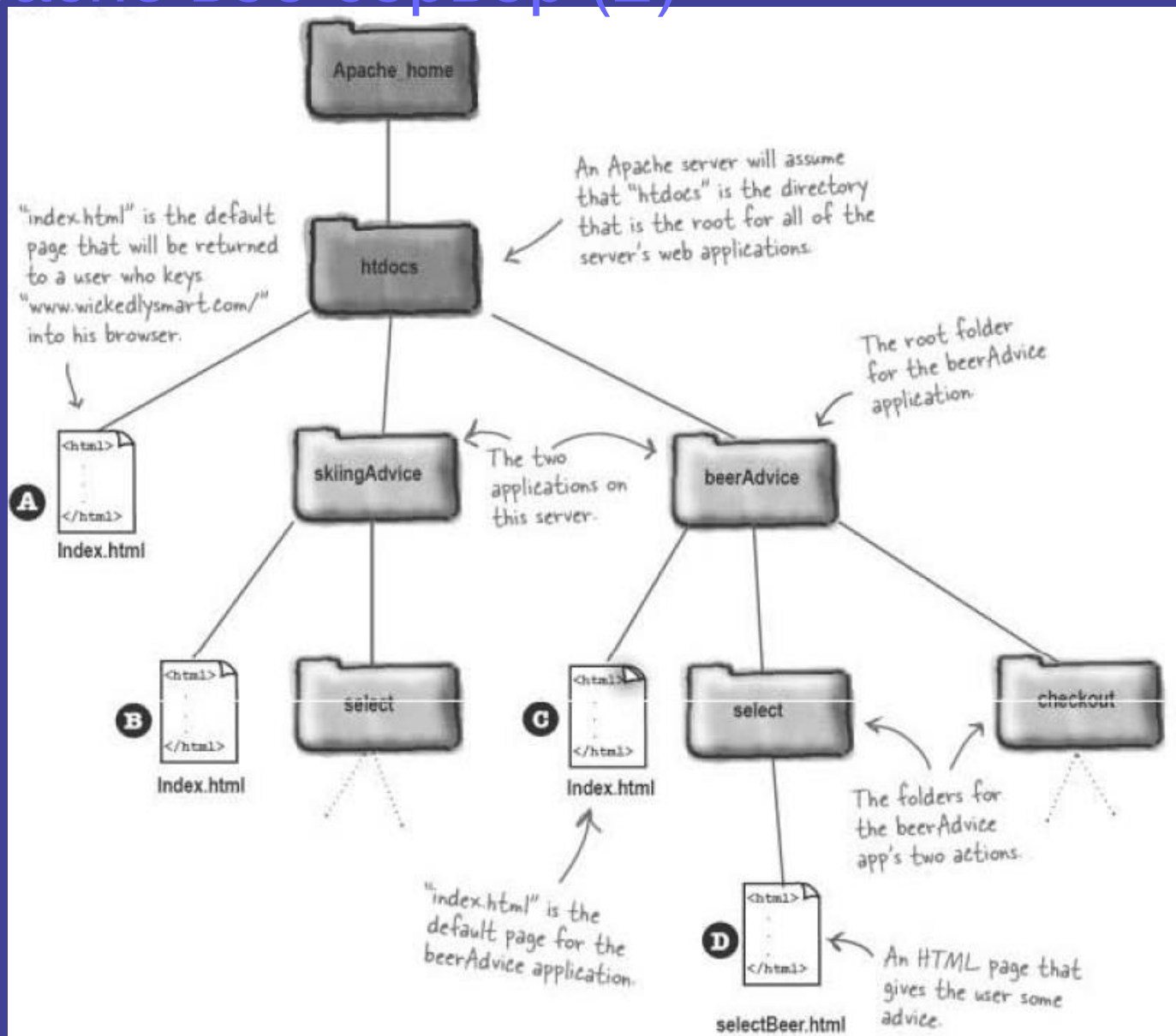
Apache веб сервер



- У оквиру овог курса биће коришћен Apache веб сервер
 - Веб сервер Apache је највише коришћен софтвер за веб сервер на свету
 - Заснован је NCSA HTTPd серверу
 - Развој Apache веб сервера је почео 1995, кад је заустављен развој NCSA кода
 - Apache је имао кључну улогу у иницијалном развоју веба, а од априла 1996. је најпопуларнији веб сервер
 - У новембру 2015 је процењено да Apache опслужује 50% активних веб сајтова и 37% најбољих сервера у свим доменима
 - Apache веб сервер је софтвер отвореног кода/слободан софтвер под лиценцом Apache License, који развија и одржава заједница Apache Software Foundation
 - Најчешће се користи на Unix-оликим системима, али и на eComStation, Microsoft Windows, NetWare, OpenVMS, OS/2 и TRPF.



Apache веб сервер (2)





Захвалница

Делови материјала ове презентације су преузети из:

- Скрипте из предмета Увод у веб и интернет програмирање на Математичком факултету, аутор проф. др Филип Марић
- Скрипте из предмета Информатика на Универзитету Milano Bicocca, аутор Mirko Cesarini
- Књиге Head First PHP & MySQL, аутори Lynn Beighley и Michael Morrison, издавач O'Reilly, 2009.
- Књиге Head First Servlets and JSP, аутори Bryan Basham, Kathy Sierra и Bert Bates, издавач O'Reilly, 2008.