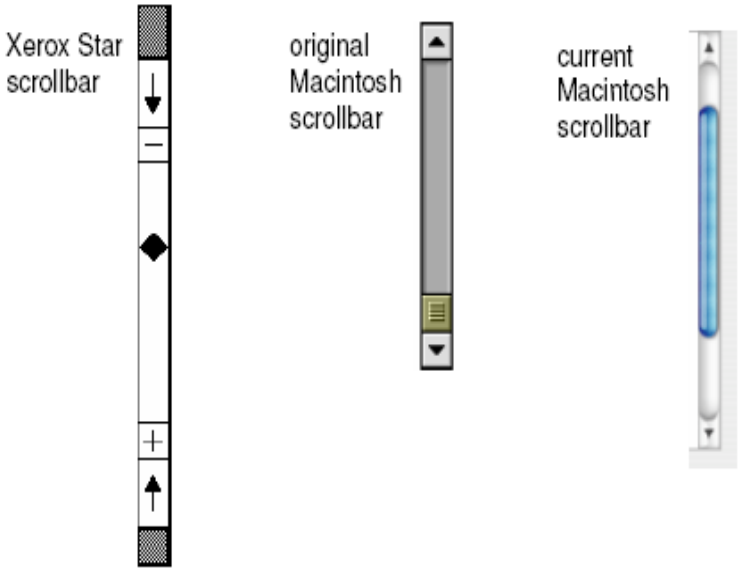
Programiranje korisničkog interfejsa

Izlaz

Ulaz

Primer



- Ghostview - Unix program koji prikazuje Postscript fajlove - nema scrollbar-a;
- Pomera se po slici **direktnom manipulacijom** – klikne se i pomera oko slike

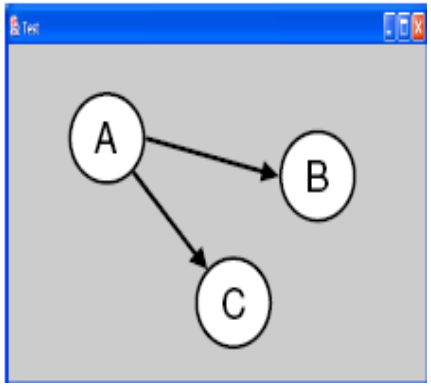
Tema

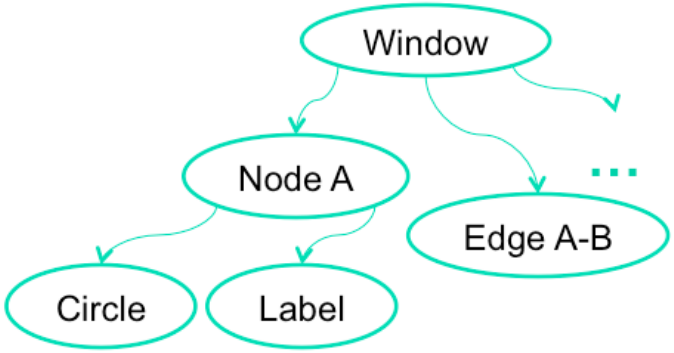
- Izlazni model
- Crtanje
- Raster grafika
- Modeli boja

Tri modela izlaza

- Komponente
  – Grafički objekti povezani u stablo sa sa automatskim osvežavanjem
  – Primeri: Objekat labele, linija
  – Nazivaju se i: pogledi, widget (komponente), kontrole,
- Linije (strokes)
  – Primitive visokog nivoa: linije, oblici, tekst
  – Primer: metod drawText(), metod drawLine()
  – Nazivaju se i: vektorske grafike, strukturirane grafike
- Pikseli
  – 2D niz piksela
  – Nazivaju se i: raster, slika, bitmape
- U današnjim GUI arhitekturama se pojavljuju sva tri modela (komponente na najvišem nivou, u listovima hijerarhije su linije, a sve se prikazuje pikselima)
- HTML (komponente), Postscript (linije ulaz, pikseli izlaz), PDF (linije), LCD ekrani (pikseli)

Primer – prikaz grafa



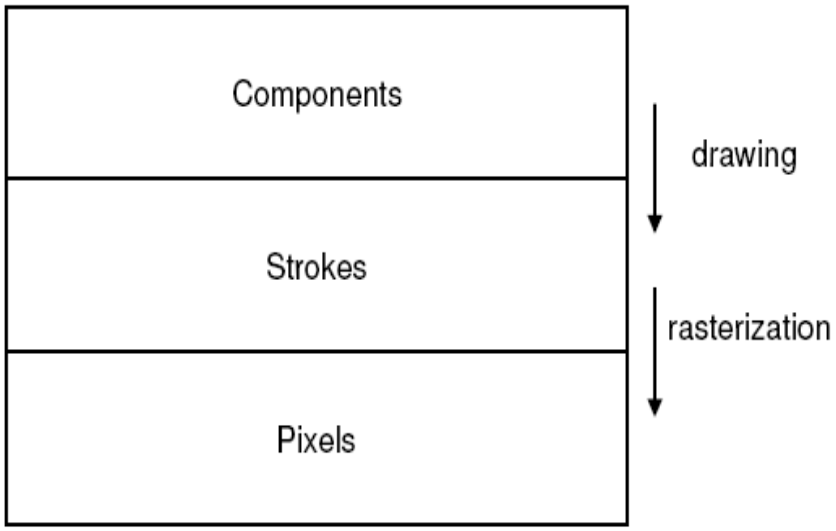


- Postavlja se pitanje kada se koji model koristi
- Model komponenti
  – Svaki čvor i grana su komponenta
  – Čvor može imati dve podkomponente: krug i labelu
  – Ostali modeli postoje, ali ih pozivaju primitive objekta
- Model linija
  – Prozor nema podkomponente, Pomoću grafa se crtaju linije, okviri i tekst (drawCircle, drawText, drawLine, ...)
- Model piksela
  – Pomoću grafa se definišu pikseli slika čvorova
- Moguć je i hibridni model – na primer komponente za čvorove, a veze pomoću linija

Izbor izlaznog modela

- Layout – komponente pamte gde treba da se iscrtavaju
- Ulaz – ako je čvor komponenta može da se automatski prepozna događaj, a kod modela linija je ručna obrada događaja
- Osvežavanje – komponente automatski, a kod ostalih ručno
- Redosled crtanja - kod hibridnog pristupa više različitih slojeva
- Težina objekata – komponente zauzimaju više prostora (graf od 100000 čvorova?)
- Zavisnost uređaja – model linija je platformski neutralan

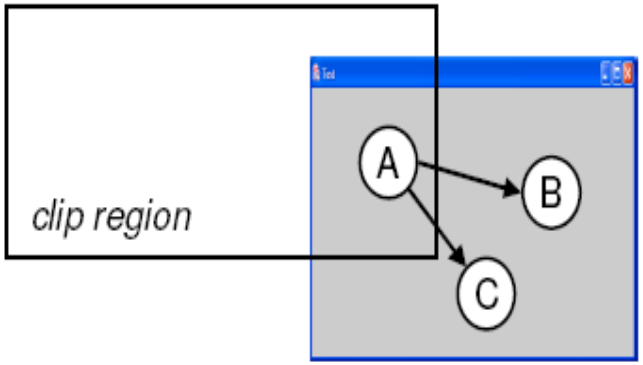
Interakcija izlaznih modela

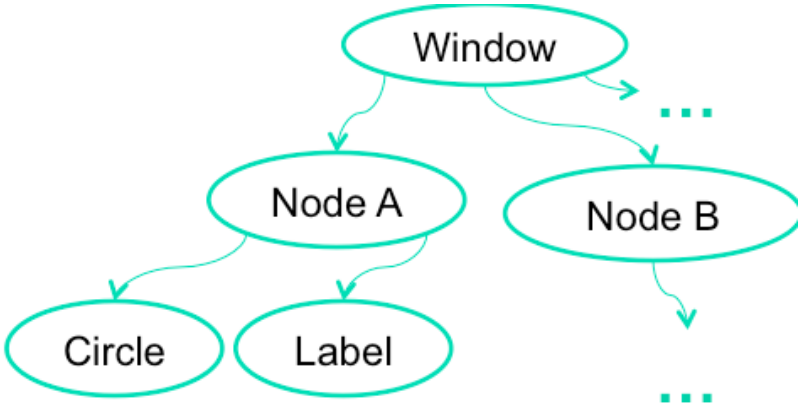


- Na najvišem nivou program predstavlja sebe u okviru prozora (komponenta), a na najnižem prozor se pojavljuje na ekranu kao pravougaonik piksela.
- Više koraka za prelazak komponente prozora u piksele

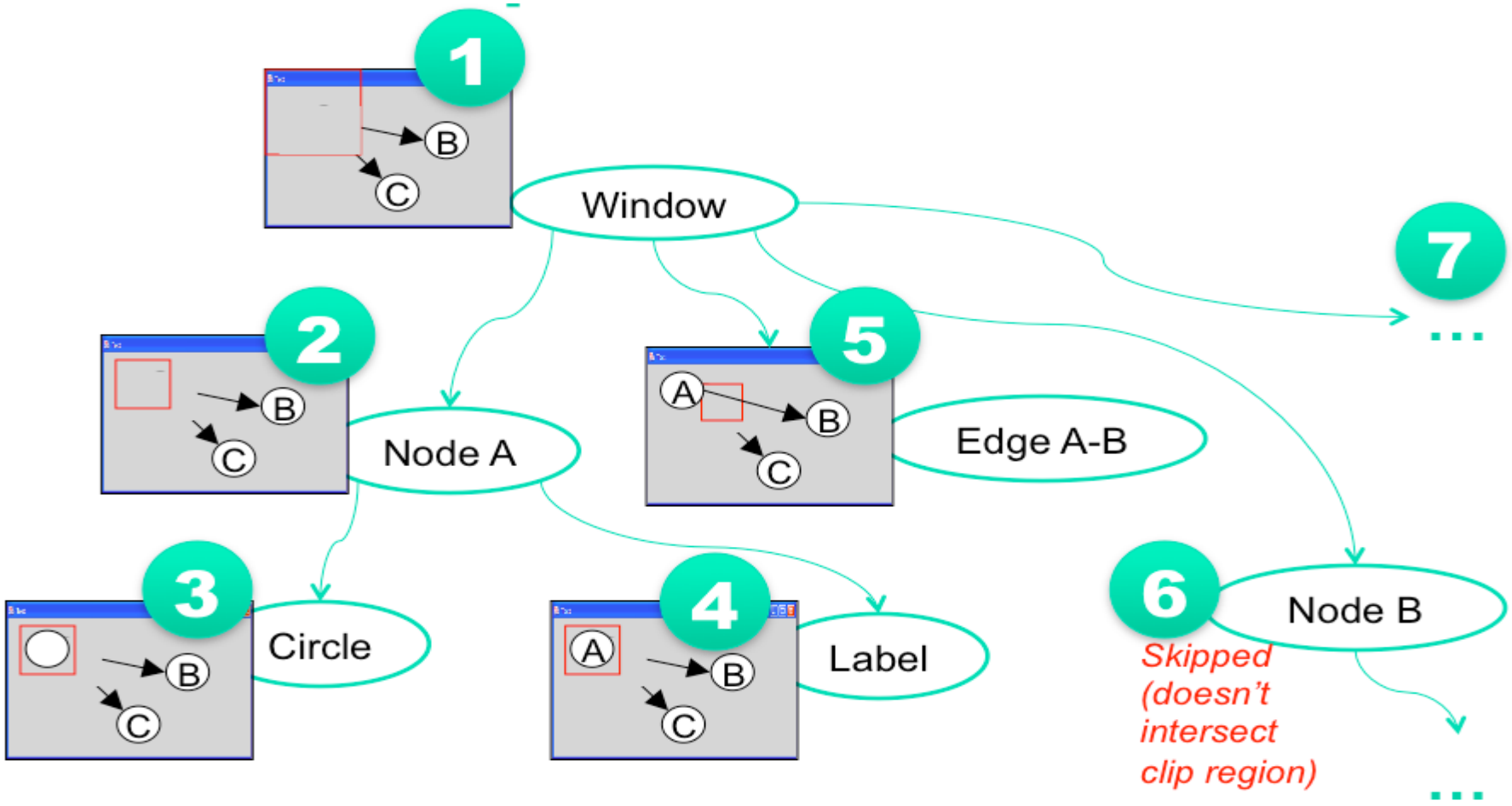
Crtanje kod modela komponenti

- Crtanje se izvršava po principu top down
  – Kreće se od korena stabla hijerahije, objekat iscrta sebe (koristeći linije ili piksele)
  – Za svaku komponentu sledbenika, ako pripada regionu crta se (A), u suprotnom ne (B, C)
  - rekurzivno se iscrtavaju potomci
- Radi boljih perfomansi, region je obično u formi pravougaonika i koriste se granice komponenti, a ne sam oblik
- Ali region može biti i neki deo ekrana, na primer string u tekstu je region i može se kreirati određena slika
- Postscript je bio prvi model linija koji je dozvoljavao ovakvu vrstu ne-pravougaonika kao regiona
- Microsoft Windows i X Windows, dozvoljavaju prozore kao ne-pravougaonike
- Java Swing izuzetak – prvo poslednji sledbenik

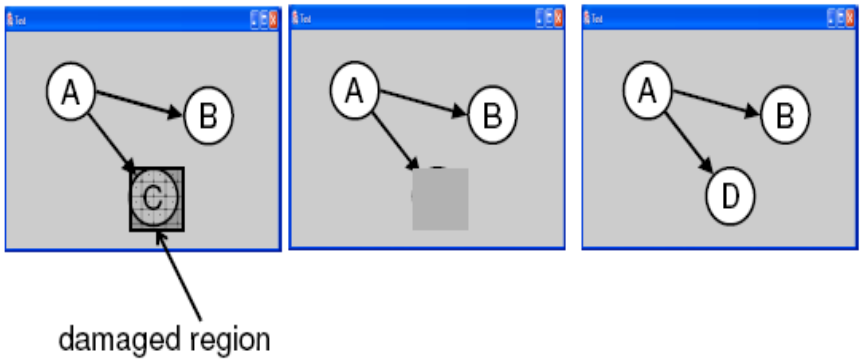




Crtanje kod modela komponenti

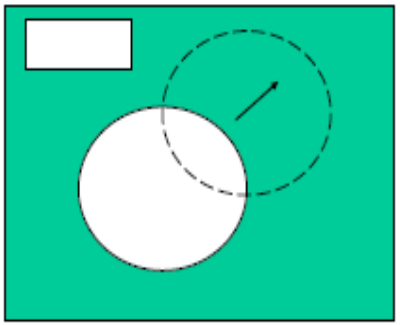


Automatsko iscrtavanje

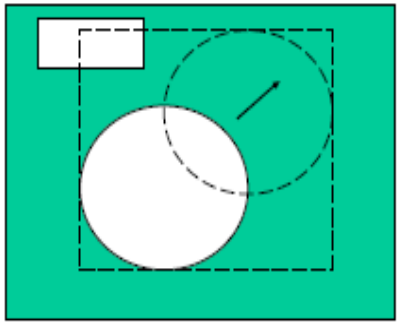


- Kada komponenta treba da promeni svoj izgled, ne iscrtava se direktno (top-down hijerarhija)
- Umesto toga, komponenta pita grafički sistem da je iscrta u budućnosti.
- Ovaj zahtev definiše **damaged region**, deo ekrana koji mora da se ponovo iscrta
- Obično su to granice komponente, ali kompleksnije komponente mogu definisati koji delovi ekrana odgovaraju kojim delovima modela koji su promenjeni, pa se samo deo komponente označi
- Zahtev za iscrtavanje se smešta u red za kasniju obradu
- Eventualno, nakon obrade ulaznih događaja, zahtev za iscrtavanje se obradi i komponente se iscrtaju na osnovu stabla komponenti, počev od korena
- Problem kod kretanja

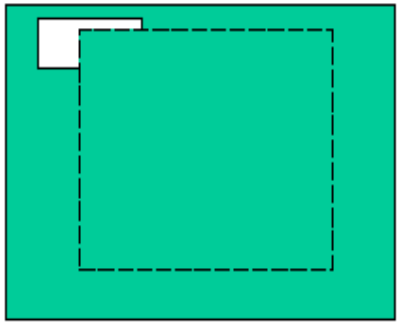
Mogući problem



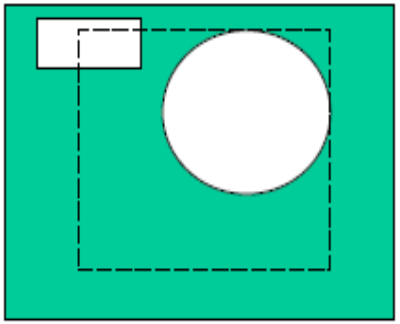
Object moves



Determine damaged region



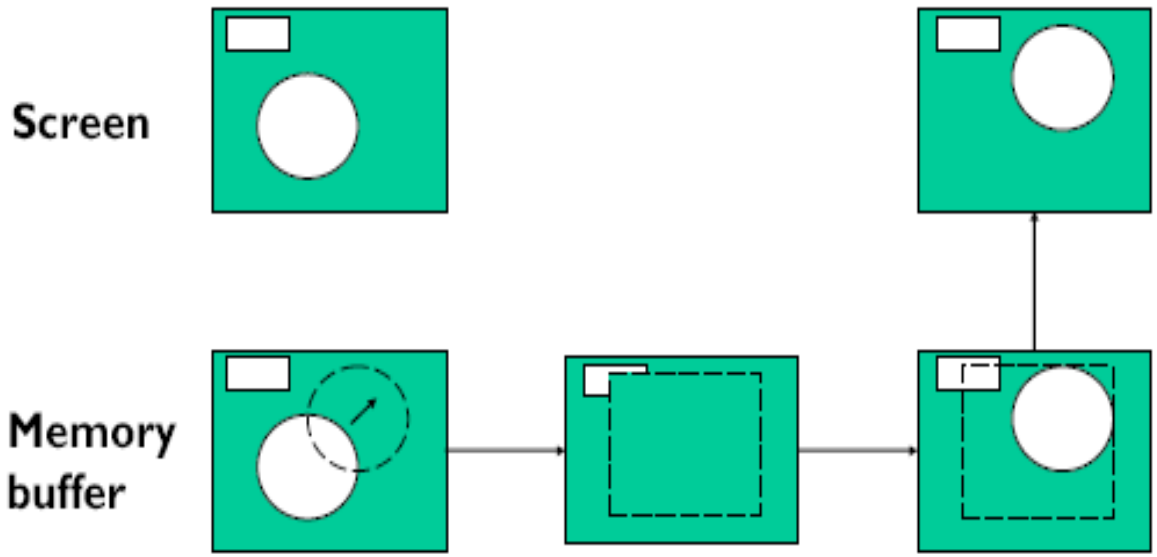
Redraw parent (children blink out!)



Redraw children

Dvostruko baferovanje

- Dvostruko baferovanje rešava problem kod kretanja



Sa modela komponenti na model linija

- Pristup pomoću metoda drawing
  – npr. Swing paint() metod, rekurzivno se pozivaju paint() metode niz hijerarhiju pogleda
  – kod Swinga paint() metod se može razložiti na nekoliko šablona metoda, kao što su paintComponent() i paintChildren()
- Pristup sa “retained graphics”
  – npr. Adobe Flex
  – U okviru objekta se pamti niz poziva iscrtavanja linija i poziva se kada je potrebno
- Razlike
  – Glavna razlika je **kada** se poziva iscrtavanje linija
  – Kod drawing metoda iscrtavanje je moguće, samo kada je metod aktivan

Model linija

- Iscrtavanje površina
  – Ekran, memorijski bafer, fajl, udaljeni ekran
- Grafički sadržaj
  – Enkapsulira parametre crtanja pa ne moraju da se prosleđuju sa svakim pozivom primitive crtanja
  – Font, boja, debljina linije, uzorak popunjavanja, ...
- Koordinatni sistem
  – Koordinatni početak, skala, rotacija
- Region
- Primitive crtanja
  – Linije, krug, elipsa, pravougaonik, tekst, oblici

HTML 5 – canvas element

HTML element

```
<canvas width=1000
   height=1000></canvas>
```

graphics context

```
var ctx = canvas.getContext
   (“2d”)
```

coordinate system

```
ctx.translate()
ctx.rotate()
ctx.scale()
```

color, font, line style, etc.

```
ctx.strokeStyle = “rgb
   (0.5,0.5,0.5)”
ctx.fillStyle = …
ctx.font = “bold 12pt sans-
   serif”
ctx.lineWidth = 2.5
```

drawing primitives

```
ctx.beginPath();
ctx.moveTo(0,0)
ctx.lineTo(500,500)
ctx.stroke()

ctx.beginPath()
ctx.arc(500,500,100,0,
   2*Math.PI,false)
ctx.fill()
```

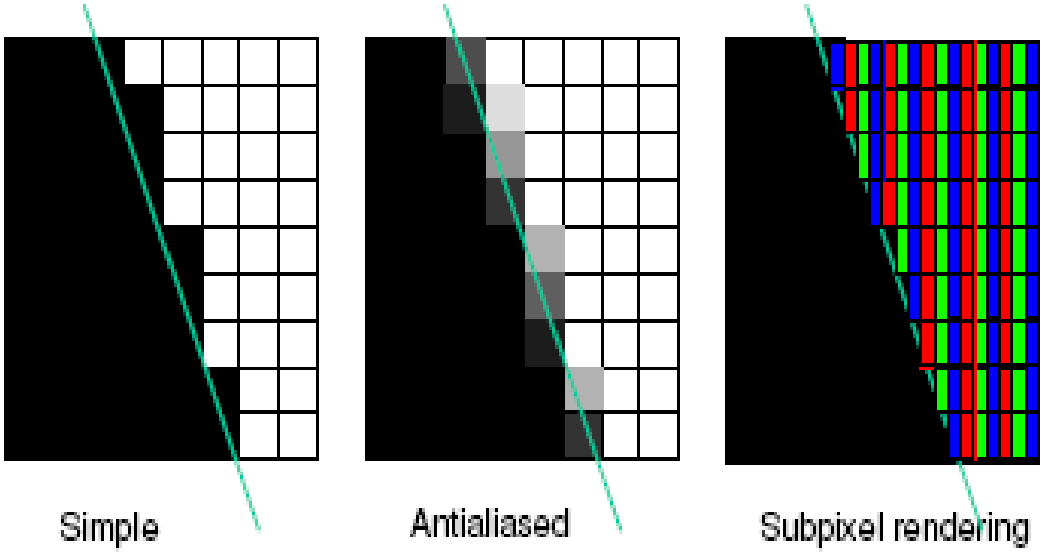
clipping

```
ctx.beginPath()
ctx.rect(0, 0, 200, 300)
ctx.clip()
```

Rasterizacija

- Kako se linija prevodi u piksele – različiti sistemi, različiti načini
- Da li je (0,0) centar piksela u levom gornjem uglu, ili to levi gornji ugao piksela?
  – MS Win: center piksela
  – Java: levi gornji ugao
- Gde će se linija (0,0) – (10,0) iscrtati?
  – MS Win: isključen krajnji piksel
  – Java Graphics: dole i desno
  – Java Graphics2D: opciono ½ pixel poravnanje zbog kompatibilnosti
- Gde će prazan pravougaonik (0,0) – (10,10) iscrtatati?
  – MSWin: spajajući navedene piksele
  – Java: proširuje jedan red ispod i jednu kolonu udesno
- Kako će se pun pravougaonik (0,0) – (10,10) iscrtatati?
  – MSWin: 121 piksela
  – Java: 100 piksela

Poravnanje



Model piksela

- Model piksela je pravougaoni niz piksela
  – Svaki piksel je vektor (n.p., RGB komponente), tako da je niz piksela realno 3-dimenzionalni
- Bits per pixel (bpp)
  – 1 bpp: black/white
  – 4-8 bpp: svaki piksel je indeks u okviru palete boje, indeksirane boje, GIF
  – 24 bpp: 8 bita za svaku boju, “true color”, “direct color”
  – 32 bpp: 8 bita za svaku boju + alpha kanal
- Komponente boje (n.p. RGB) se još nazivaju kanali
- Model piksela se može prikazati na više načina
  – Pakovati se u reči (RGBR GBRG …)
  – Posmatrati od vrha ka dnu ili od dna ka vrhu

Transparentnost

- **Alfa (Alpha)** predstavlja transparentnost piksela
  – raspon od 0.0 (transparentno) do 1.0 (neprovidan)
  – tako da svaki piksel ima RGB i alfa vrednost
- Upotreba ovog parametra
  – Antialiasing
  – Ne-pravougaone slike
  –Clipping regioni sa poravnatim (antialiased) ivicama

BitBlt

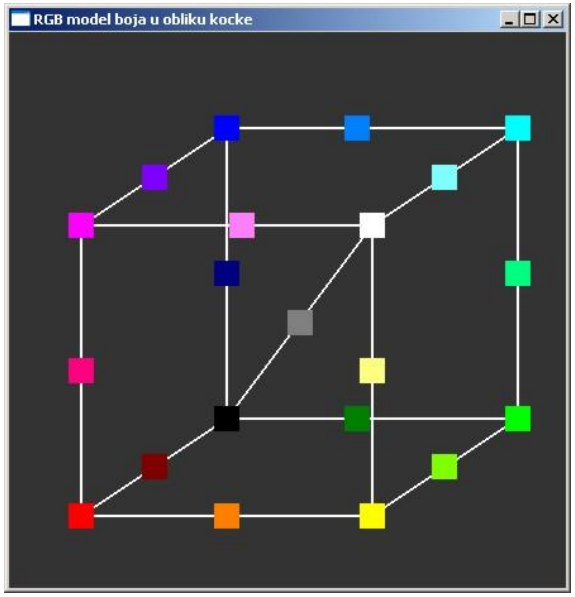
- BitBlt (bit block transfer) kopira blok piksela sa jedne slike na drugu
  – Iscrtavanje slika na ekranu
  – Dvostruko baferisanje
  – Skrolovanje – ušteda, deo koji se ne menja samo se prebaci
  – Kliping sa ne-pravougaonim maskama – kombinovanje dve slike u jednu
- Tu su definisana i kompozitna pravila kontrole kako se kombinuju pikseli sa izvorišta i odredišta

Formati slika

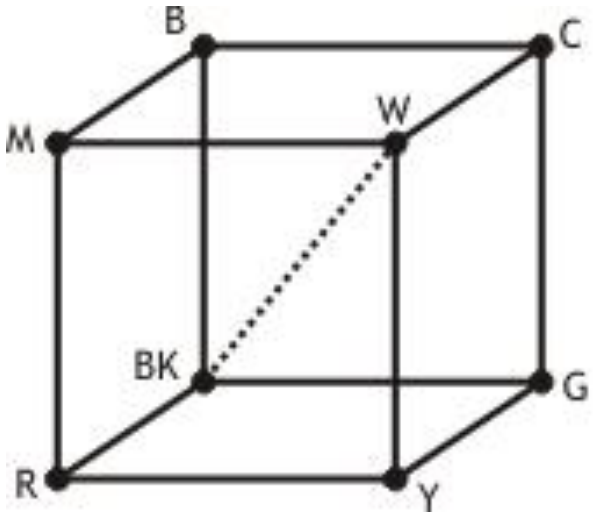
- GIF
  – 8 bpp, paleta koristi 24-bitne boje
  – 1 boja u paleti može da bude transparentna (1-bit alpha kanal)
  – koristi se za screenshot, ikone, grafike sa linijama
- JPEG
  – 24 bpp, nema alpha kanala
  – Koristi se za fotografije
- PNG
  – 1, 2, 4, 8 bpp sa paletom
  – 24 or 48 bpp sa true color sistemom
  – 32 or 64 bpp sa true color sistemom i alpha kanalom
  – koristi se gde i GIF
  – bolje od GIF-a, ali bez animacije
  – IE 6 podržava transparentne piksele, ali ne punu alpha transparentnost, pa se koristio GIF

Modeli boja

- RGB
- CMY
- CMYK
- HSL
- HSI
- HSV
- Konvertovanje između važnijih modela boja



RGB model boja

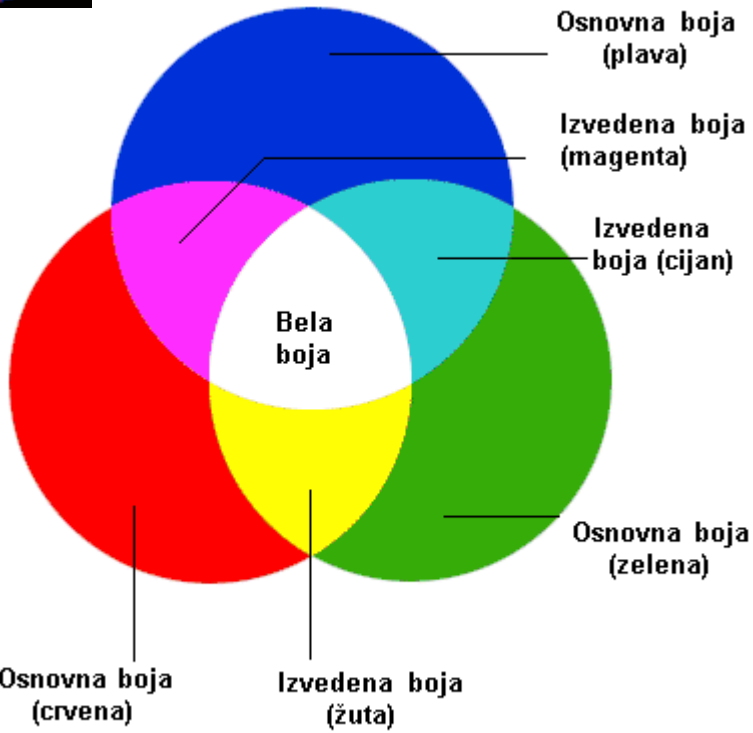


R - Red (crvena boja)
Y - Yellow (žuta boja)
G - Green (zelena boja)
BK - Black (crna boja)
M - Magenta (ljubičasta boja)
W - White (bela boja)
C - Cyan (svetloplava boja)
B - Blue (plava boja)

Tri osnovne boje koje sadrži svetlost su crvena, zelena i plava.

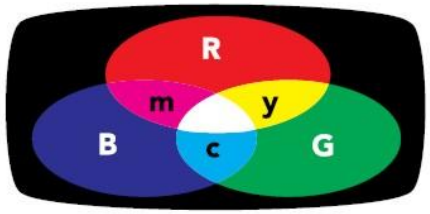
Kada se po dve osnovne boje svetlosti pomešaju u jednakoj količini, tada dobijamo izvedene boje - žutu, cijan i magentu, a kada se pomešaju sve 3 osnovne boje, dobijamo belu svetlost.

Mnogobrojni načini kombinovanja osnovnih boja svetlosti nazivaju se aditivni procesi.

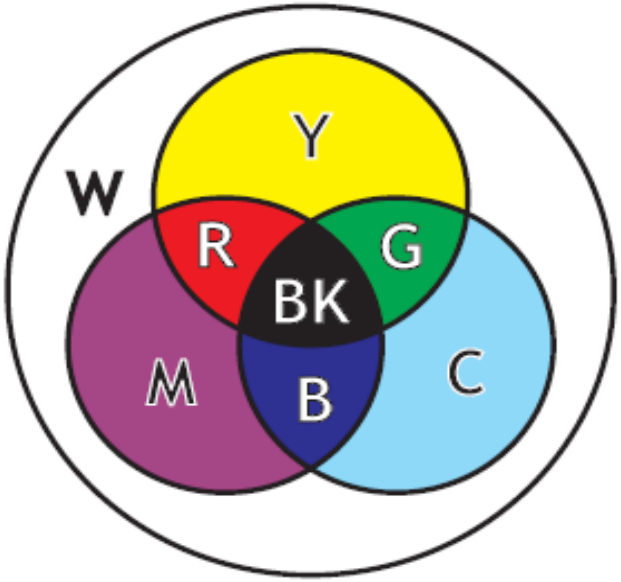


Osnovne i izvedene boje

When two additive primaries overlap, a subtractive primary is produced. Where all three are combined, white light is produced.



Subtraktivne boje



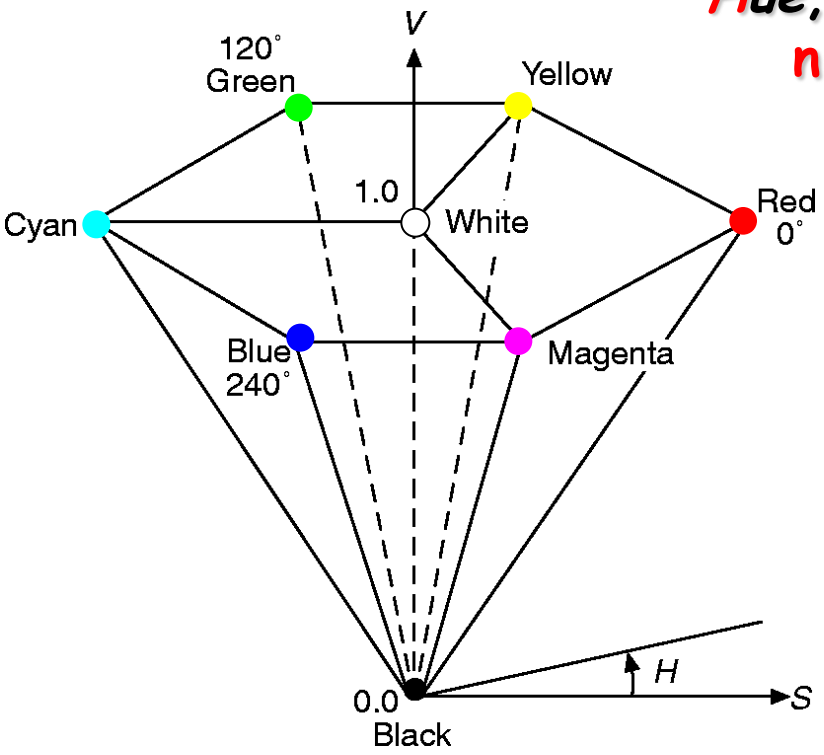
Yellow **apsorbuje** ***Blue***

Magenta **apsorbuje** ***Green***

Cyan **apsorbuje** ***Red***

HSV model boja

Hue, Saturation, Value (brightness)
nijansa, zasićenost, vrednost



	Opseg	W	Y	C	G	M	R	B	BK
H	$0° \div 360°$	-	$60°$	$180°$	$120°$	$300°$	$0°$	$240°$	-
S	$0 \div 1$	0	1	1	1	1	1	1	0
V	$0 \div 1$	$0,75$	$0,75$	$0,75$	$0,75$	$0,75$	$0,75$	$0,75$	0


Intenzitet V

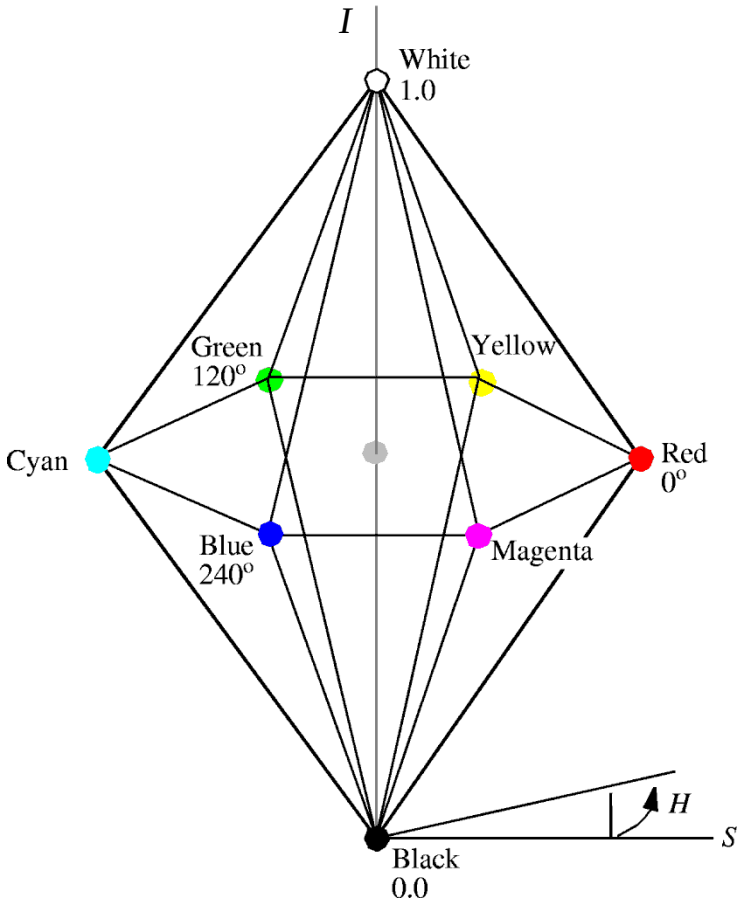

Nijansa H


Zasićenost S


RGB

HSI model boja

Hue, Saturation, Intensity
nijansa, zasićenost, intenzitet



	Opseg	W	Y	C	G	M	R	B	BK
H	$0° \div 360°$	-	$60°$	$180°$	$120°$	$300°$	$0°$	$240°$	-
S	$0 \div 1$	0	1	1	1	1	1	1	0
V	$0 \div 1$	$0,75$	$0,375$	$0,375$	$0,375$	$0,375$	$0,375$	$0,375$	0

Ulaz

- Ulazni događaji
- Obrada događaja
- Propagacija događaja

Zašto koristiti događaje za GUI ulaze

- U/I konzola koristi blokiranje poziva procedura

  print (“Enter name:”)
  name = readLine();
  print (“Enter phone number:”)
  name = readLine();
  – Sistem kontroliše dijalog
- Umesto toga GUI ulaz koristi obradu događaja

  – Korisnik ima mnogo više kontrole pomoću dijaloga

  – Korisnik može da klikne skoro na svaki element

Vrste ulaznih događaja

- Direktni ulazni događaji (od drajvera uređaja)
  - Pomeranje miša
  - Pritisak i otpuštanje dugmeta miša
  - Pritisak i otpuštanje tastature
- Translirani ulazni događaji
  - Klik ili dvostruki klik miša
  - Miš ulazi ili napušta komponentu
  - Dobija se ili gubi fokus tastature
  - Unos karaktera

Osobine ulaznih događaja

- Pozicija miša (X,Y)
- Stanje dugmeta miša
- Modifikator stanja tastature (Ctrl, Shift, Alt, Meta)
- Vreme izvršavanja
  - Zašto je vreme izvršavanja važno?

Red događaja

- Događaji su smešteni u red
  – Ulaz se usmerava da se razdvaja
  – Red čuva aplikacije od velikog zadržavanja u realnom vremenu (n.p., mora da se završi obrada svakog događaja, pre nego što sledeći može da se dogodi)
- Pokreti miša se sjedinjuju u jedan događaj u redu
  – Nekada i loše, linija ispisana pomoću miša može biti ravna između pojedinačnih tačaka

Petlja događaja

- Dok se aplikacija izvršava
  - Blokirati sve dok događaj nije spreman
  - Uzeti događaj iz reda
  - (ponekad) Prevesti direktne događaje u događaje visokog nivoa
- Generisati dvostruki klik, karaktere, fokus, ulaz-izlaz,...
- Prevedeni događaji se smeštaju u red
- Povezati događaj sa ciljnom komponentom
- Ko obezbeđuje petlju događaja?

  – GUI alati visokog nivoa obavljaju taj posao interno (Java Swing, VB, C#, HTML)

  – Alati nižeg nivoa zahtevaju da aplikacije urade taj posao za njih (MS Win, Palm, Java SWT)

Propagacija događaja

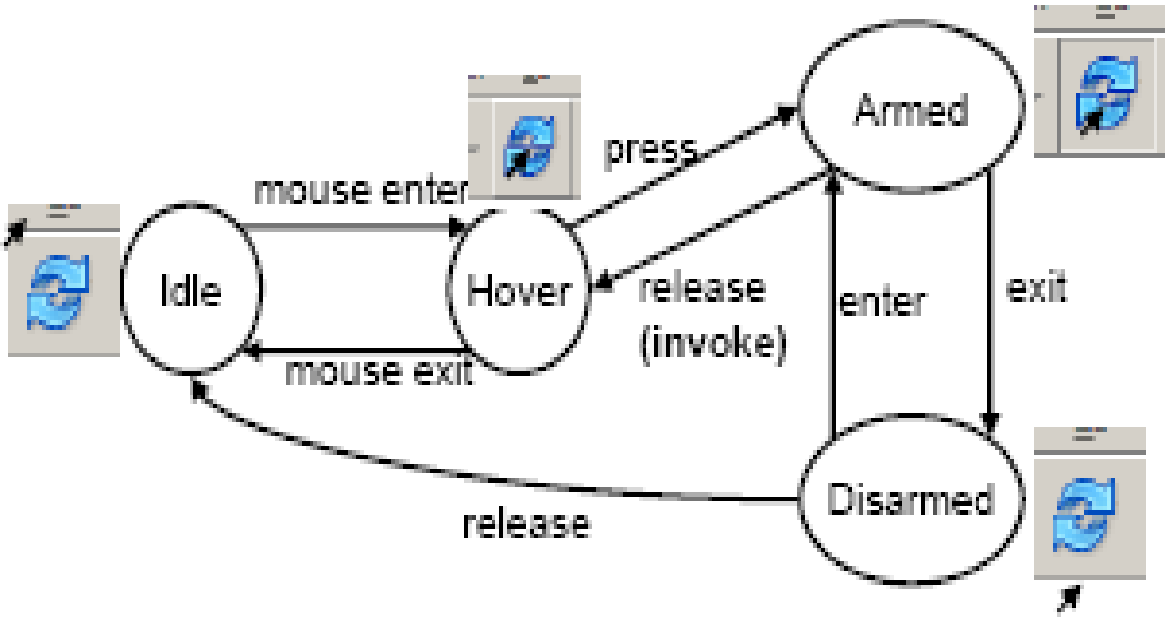
- Izabrati ciljnu komponentu za događaja
  – Događaj tastera: komponenta sa fokusom
  – Događaj miša: komponenta ispod kurzora miša
- **Prisvajanje miša**: svaka komponenta može trenutno prihvatiti miša, tako da primi sve njegove događaje (n.p. drag & drop)
- Propagacija: ako ciljna komponenta odbije da obradi događaj, on se propušta dalja prethodniku

JavaScript obrada događaja

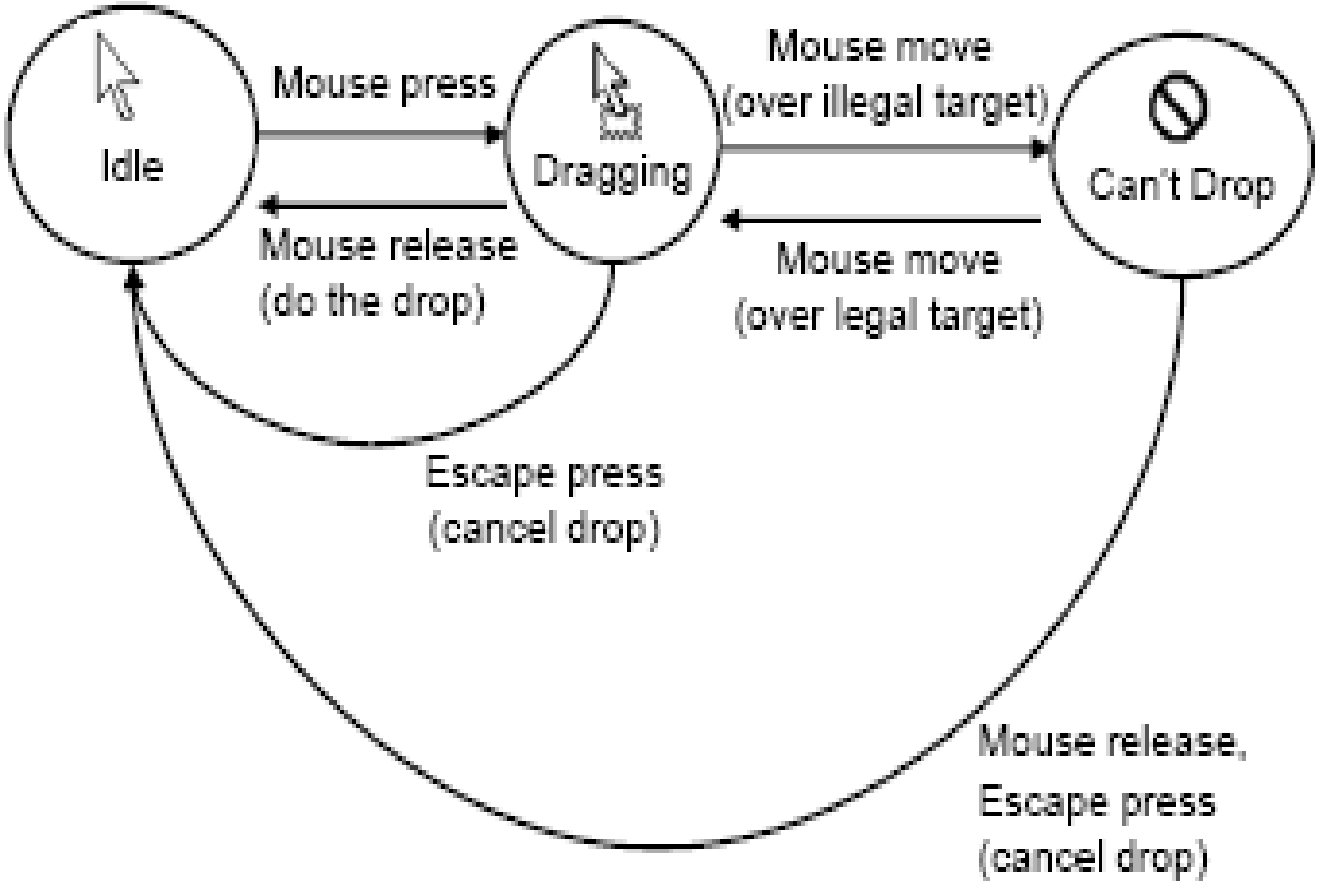
- Događaji se propagiraju u različitim pravcima kod različitih browser-a
  – Netscape 4: na dole od osnovnog ka ciljnom
  – Internet Explorer: na gore od ciljnog ka osnovnom
  – W3C standardizuje kombinacijom: prvo nadole (“capturing”), onda nagore (“bubbling”)
    - Firefox, Opera, Safari

Dizajnirati kontroler

- Kontroler je konačni automat
- Primer: pritisak dugmeta



Drag & drop

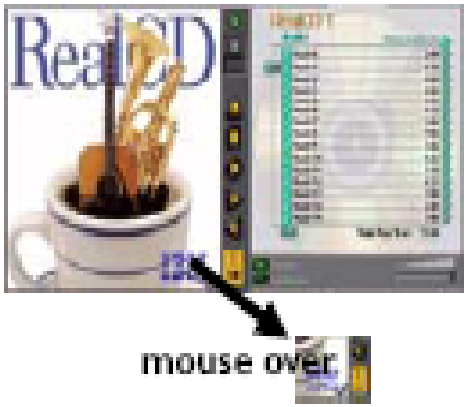

Idle
Mouse press
Dragging
Mouse release
(do the drop)
Mouse move
(over illegal target)
Can't Drop
Mouse move
(over legal target)
Escape press
(cancel drop)
Mouse release,
Escape press
(cancel drop)

Vidljivost

- Vidljive akcije
  - Informacije
- Vidljiva stanja
  - Usmeravanje pažnje
- Feedback
  - Perceptualna fuzija
  - Vreme odgovora

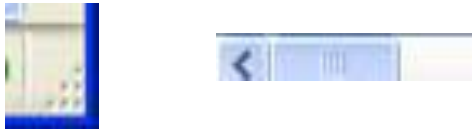
Vidljivost

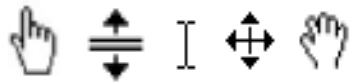
- Bitni delovi sistema treba da budu vidljivi
  – Obično nije problem u realnom svetu
  – Ali je potreban dodatni napor kod računasrkih interfejsa



Akcije: koristiti odgovarajuće značenje

- Dugmad & linkovi
- Drop-down strelice
- Tekstura
- Kurzor miša
- Promena na prelazak miša

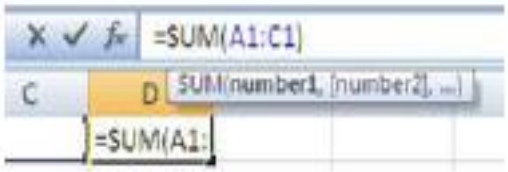




Komande treba da budu vidljive

- Meniji

- Tooltips
- Self-disclosure



Osećaj za informacije

- Teorija o skupljanju informacija
  – Ljudsko prikupljanje informacija može se modelovati kao i životinjsko prikupljanje hrane
  – Konstantno se menja i donose se odluke koje maksimiziraju dobijene informacije u odnsou na cenu njihovog dobijanja
- Osećaj za informacije
  – u odnosu na link za koji se preptpostavlja koliko se isplati pratiti ga

Pružiti dobar osećaj

- Link treba da asocira na svoj sadržaj

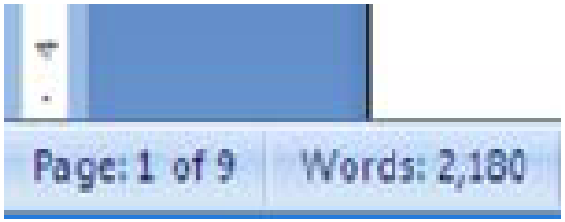
Vidljiva stanja navigacije

- Breadcrumbs
- Stranična
- Tabovi

Vidljiva stanja modela

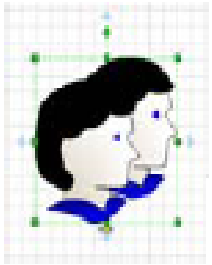
- Kontinuirana vizuelna reprezentacija backend modela

  – Šta treba vizeulno predstaviti zavisi od korisnikovih zadataka



Vidljivi pogledi stanja

- Označavanje selekcije
- Obrada selekcije
- Drag & drop



Mod rada treba da bude vidljiv

- Mod rada: stanja u kojima akcije imaju drugačija značenja
  - –insert mod i command mod
  - – Caps Lock
  - – Paleta za crtanje

Vidljivost zavisi od usmeravanje pažnje

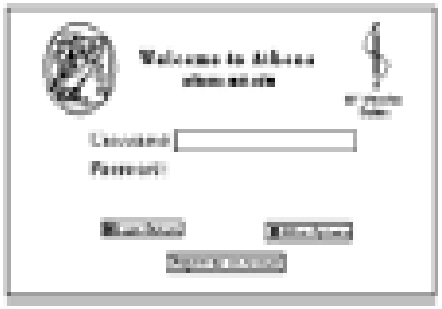
- **Metafora svetlosti lampe**: pažnja se fokusira na jedan ulazni kanal (n.p. Prostor vizuelnog polja) u vremenu
- Da li usmeravanje korisnikove pažnje obuhvata:
  - Caps Lock svetlo na tastaturi?
  - Status bar?
  - Menu bar?
  - Kurzor miša?

Feedback: akcije treba da imaju odmah vizuelne efekte

- Feedback niskog nivoa
  – n.p. Dugme

- Feedback visokog nivoa
  – promene stanje modela
  – počinje učitavanje nove Web stranice

Vidljivost i bezbednost



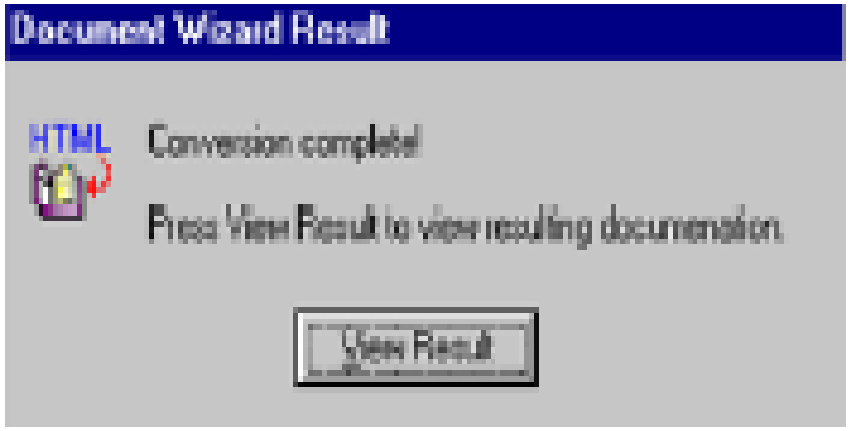
Perceptualna fuzija

- Dva stimulansa sa istim ciklusom percepcije (Tp ~ 100ms [50-200 ms]) se prihvataju zajedno
- Posledice
  – 1/ Tp frames/sec je dovoljna da bi se prihvatile pokretne slike (10 fps OK, 20 fps nejasno)
  – Odgovor računara < Tp se prihvata kao trenutni

Odgovor korisniku

- < 0.1 s: izgleda trenutno
- 0.1-1 s: korisnik primećuje kašnjenje
- 1-5 s: prikazati indikator zauzetosti
- > 1-5 s: prikazati progres bar

Nepotrebni feedback



Vidljivost i drugi principi korisnosti

- Vidljivost primarno prenosi informacije
- Podržava lakše učenje
- Može da bude suprostavljeno jednostavnosti