

1 Care din urmatoarele metode nu există implicit în JavaScript?

- A. document.getElementsByTagName();
- B. document.getElementById();
- C. **document.getElementByName();**
- D. document.getElementsByName();

2 Care este tagul pentru introducerea în HTML a unui container pentru crearea grafică?

- a. <paint>
- b. <style>
- c. **<canvas>**
- d. <graphics>

3. Care este metoda corecta pentru scriere in canvas?

- a. font(x,y,text)
- b. fillText(text,x,y)**
- c. fillstyle(text,x,y)
- d. paint(text,x,y)

4. Ce element HTML este utilizat pentru specificarea stilurilor CSS externe pentru un document?

- a. CSS
- b. link**
- c. style
- d.class

5.Care este varianta corecta de includere a unei imagini ase.jpg in cadrul unui document HTML?

- A. <image src="ase.jpg" alt="Exemplu imagine">
- B. ase.jpeg
- C.< img src="ase.jpg" alt="Exemplu imagine">**
- D.

6.Elementul HTML folosit pentru reprezentarea unui rand în cadrul unui table este

- A. tr**
- B. table
- C. cell
- D. td
- E. tc

7. Elementul HTML utilizat pentru introducerea unui element de grafica vectoriala în cadrul unei pagini este

A. graph

B. **svg**

C. canvas

D. vector

8. Stabilirea culorii roșii pentru umplerea următoare figure desenate pe contextual graphic c asociat unui element canvas se realizează prin

A. c.strokeStyle("red")

B. **c.fillStyle="red"**

C. c.color="red"

D. c.stroleStyle="red"

9. Formatul de stocare SVG este specific stocării:

A. imaginilor necomprimate

B. fisierelor de tip sunet

C. **imaginilor vectoriale**

D. imaginilor de tip raster

10. Comprezia video are la bază eliminarea redundantei:

A. doar audio

B. **intra și inter-cadru**

C. doar intra-cadru

D. doar inter-cadru

11. Sunetul este definit ca:

A. Energie magnetică într-un mediu elastic

B. **O vibratie care se propaga printr-un mediu material**

C. Energie electrică propagată prin vid

D. Energie electrica statica transmisa prin orice mediu

12. Compresia JPEG NU utilizeaza:

A. compresia RLE

B. compresia Huffman

C. transformata cosinus discrete

D. compresia LZW

13. Care dintre urmatoarele caracteristici NU este specifica imaginilor vectoriale

A. mentine semnatica imaginii

B. obiectele componente sunt descrise matematice

C. fisierul imagine este mic

D. imaginea este dependenta de scara de vizualizare

14. Desenarea unui dreptunghi in canvas se realizeaza cu metoda

A. paint(x,y,a,b)

B. fillRect(0,0,50,70)

C. fillStyle(0,0,10,30)

D. square(x,y,a,b)

15. Care este apelul corect pentru preluarea unui obiect al canvasului?

A. var b =canvas.getElementById("test")

B. var b =document.getElementById("test")

C. var b =document.getCanvas("test")

D. var b =document.getContext("test")

16. Care este sintaxa corecta pentru introducerea in pagina a unui script extern?

<script type="text/javascript" src="scriptulmeu.js">

17. Ce atribut este utilizat pentru specificarea stilurilor CSS intr-un element HTML?

A. css

B. link

C. style

D. class

18. Care este varianta corecta pentru introducerea in HTML a unui css extern?

A. <link rel="stylesheet" type="text/css" href="stilulmeu.css">

19. Care dintre urmatoarele sechete respecta sintaxe CSS?

A. body:color=black

B. body(color:black)

C. (body, color.black)

D. (body.color=black(body)

20. Ce elemente HTML vor fi modificate prin intermediul selectorului img.all?

A. Toate imaginile din cadrul documentului

B. toate elementele care au atributul id="img.all"

C toate elementele care au atributul class="img.all"

D. toate imaginile care ai atributul class="all"

20. jQueri este:

a) un limbaj de programare

b) un limbaj de acces la baza de date

c) o biblioteca JavaScript

d) un tag

21. Desenarea unui cerc folosind un context graphic asociat unui element canvas se realizeaza prin intermediul functiei:

a) circle

b) arc

c) eclipse

d)closedPath

22. Care din urmatoarele elemente svg este utilizat pentru gruparea unui set de elemente fara a le afisa?

a)rect

b)defs

c)g

d)svg

23. Care dintre urmatoarele valori nu este o valoare acceptabila pentru proprietatea CSS position?

a) static

b) relative

c)fixed

d)absolute

e)float

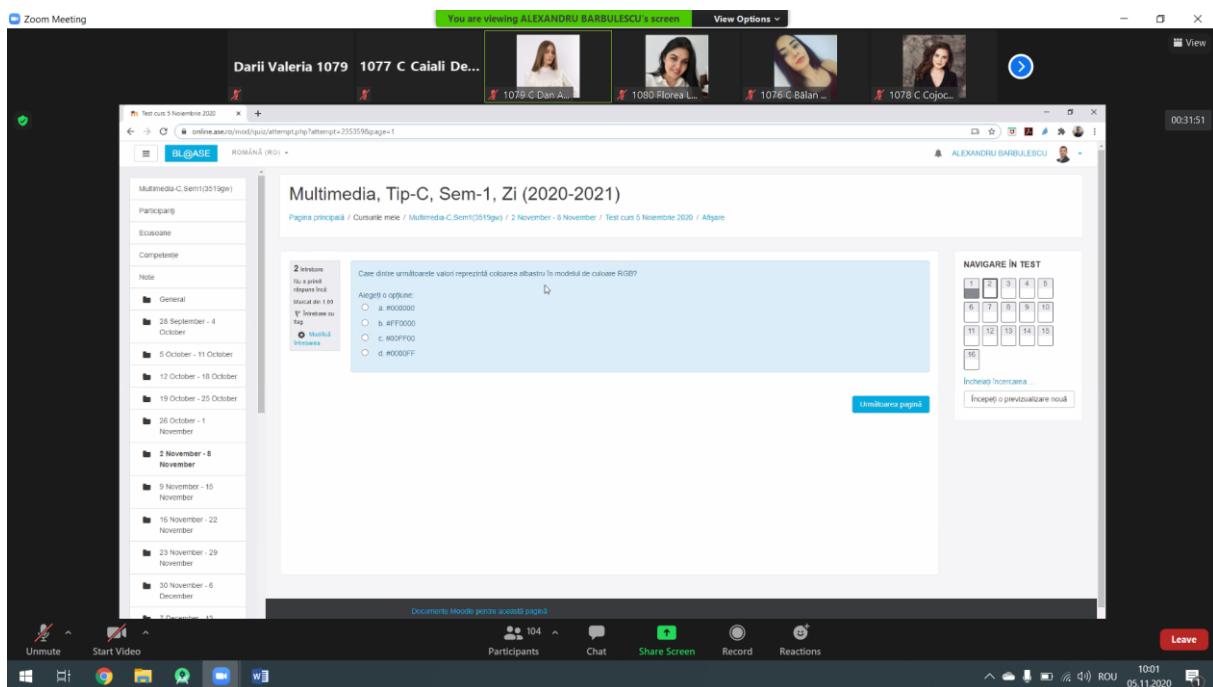
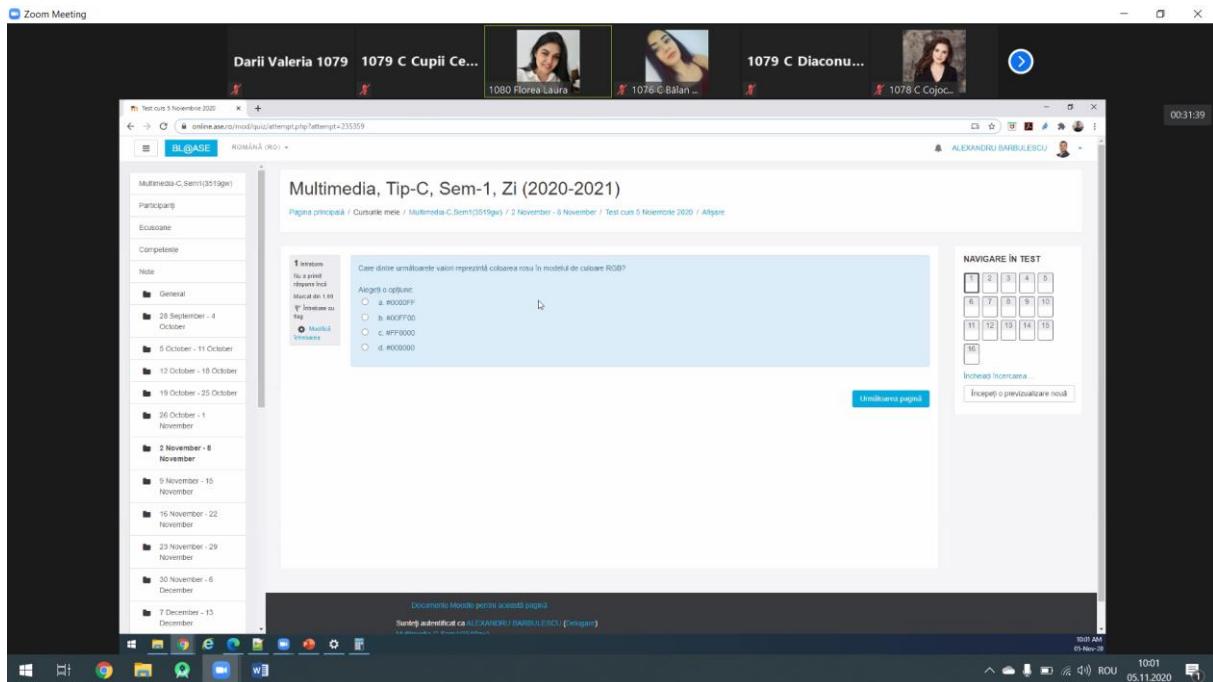
24. Elementul HTML utilizat pentru introducerea unui element de grafica raster in cadrul unei pagini este

A. graph

B. svg

C.canvas

D. vector



Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 1077 C Caiali De... 1080 C Dragne... 1079 C Dan A... 1080 C Florea L...

ROMÂNĂ (RO) ALEXANDRU BARBULESCU

00:31:59

Test curs 5 Noiembrie 2020 online.ase.ro/mod/quiz/attempt.php?attempt=255359&page=2

Multimedia-C.Sem1(3519gr)

Participanți

Ecoacăne

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 November

2 November - 8 November

9 November - 15 November

16 November - 22 November

23 November - 29 November

30 November - 6 December

7 Decembrie - 13 Decembrie

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519gr) / 2 November - 6 November / Test curs 5 Noiembrie 2020 / Afisare

3 întrebări
Nu s-a postat întrebări încă.
Marcat de 1 din 1

Înregistrare nouă

Alegeți o opțiune:

a. maria

b. body(maria)

c. maria

d. emaria

Urmărirea pagină

NAVIGARE ÎN TEST

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Încearcă încercarea

Începeți o previzualizare nouă

Documentul Moodle permite accesă pagină

Participants 104 Chat Share Screen Record Reactions

Leave

Unmute Start Video

wi

Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 1077 C Caiali De... 1076 C Agafita... 1080 C Dragne... 1079 C Dan A... 1080 C Florea L...

ROMÂNĂ (RO) ALEXANDRU BARBULESCU

00:32:11

Test curs 5 Noiembrie 2020 online.ase.ro/mod/quiz/attempt.php?attempt=255359&page=3

Multimedia-C.Sem1(3519gr)

Participanți

Ecoacăne

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 November

2 November - 8 November

9 November - 15 November

16 November - 22 November

23 November - 29 November

30 November - 6 December

7 Decembrie - 13 Decembrie

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519gr) / 2 November - 6 November / Test curs 5 Noiembrie 2020 / Afisare

4 întrebări
Nu s-a postat întrebări încă.
Marcat de 1 din 1

Înregistrare nouă

Alegeți o opțiune:

a. var b = canvas.getelementbyid("test")

b. var b = document.getelementbyid("test")

c. var b = document.getcanvas("test")

d. var b = document.getcontent("test")

Urmărirea pagină

NAVIGARE ÎN TEST

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

Încearcă încercarea

Începeți o previzualizare nouă

Documentul Moodle permite accesă pagină

Participants 104 Chat Share Screen Record Reactions

Leave

Unmute Start Video

wi

Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 ALEXANDRU BAR... 1077 C Caiali De... 1076 C Agafit... 1080 C Dragom... 1079 C Dan A...

View Options

00:32:32

online.ase.ro/mod/quiz/attempt.php?attempt=25359&page=4

BL@ASE ROMÂNĂ (RO)

Multimedia-C.Sem1(3519gr)

Participanți

Echipă

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 November

2 November - 8 November

9 November - 15 November

16 November - 22 November

23 November - 29 November

30 November - 6 December

7 Decembrie - 13 Decembrie

5 întrebări

Nu s-a prezentat încă niciun elev.

Marcat de 1 elev

Înregistrare în tag

Modulul

Întrebări

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519gr) / 2 November - 8 November / Test curs 5 Noiembrie 2020 / Afisare

Formatul de stocare (IVG) este specific disciplinei:

Alegeți o opțiune:

a. hipersuplimente de tip sușet

b. imaginile necompresate

c. imaginile vectoriale

d. imaginile de tip raster

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Încearcă încărcarea

Începeți o previzualizare nouă

Urmărirea pagină

Participants 103 Chat Share Screen Record Reactions

Leave

00:32:32

1002 ROU 05.11.2020

Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 ALEXANDRU BAR... 1077 C Caiali De... 1076 C Agafit... 1080 C Dragom... 1079 C Dan A...

View Options

00:32:52

online.ase.ro/mod/quiz/attempt.php?attempt=25359&page=5

BL@ASE ROMÂNĂ (RO)

Multimedia-C.Sem1(3519gr)

Participanți

Echipă

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 November

2 November - 8 November

9 November - 15 November

16 November - 22 November

23 November - 29 November

30 November - 6 December

7 Decembrie - 13 Decembrie

6 întrebări

Nu s-a prezentat încă niciun elev.

Marcat de 1 elev

Înregistrare în tag

Modulul

Întrebări

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519gr) / 2 November - 8 November / Test curs 5 Noiembrie 2020 / Afisare

In CSS prin tabelul sau se selectază:

Alegeți o opțiune:

a. Primul înalt din tabel

b. Tabelul cu id-ul meu

c. Tabelul cu o înaltă

d. Un buton

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Încearcă încărcarea

Începeți o previzualizare nouă

Urmărirea pagină

Participants 103 Chat Share Screen Record Reactions

Leave

00:32:52

1002 ROU 05.11.2020

Zoom Meeting

You are Viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079

ALEXANDRU BAR... 1077 C Caiali De... 1076 C Agafit... 1080 C Dragin... 1079 C Dan A.

View Options

ROUMANIA (RO) ROMÂNĂ (RO)

Multimedia-C.Sem1(3519ge)

Participanți

Educator

Competențe

Note

General

28 October - 4 October

5 October - 11 October

12 October - 18 October

19 October - 25 October

26 October - 1 November

2 November - 8 November

9 November - 15 November

16 November - 22 November

23 November - 29 November

30 November - 6 December

7 întrebări

Nu s-a prezentat încă niciun elev.

Marcat de 1 elev

7 întrebări

Alte întrebări

Modul de lucru

Formatul de stocare a imaginii JPEG este specific:

Alte opțiuni:

a. imagine vectorială

b. imagine de tip raster

c. imagine de tip sunet și video

d. imagine recomprimată

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Încheie încercarea

Începeți o previzualizare nouă

Urmărește pagină

Participants 103 Chat Share Screen Record Reactions

Leave

00:33:13

online.ase.ro/mod/quiz/attempt.php?attempt=255359&page=6

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519ge) / 2 November - 8 November / Test curs 5 Noiembrie 2020 / Atigare

Document Moodle pentru accesă pagină

(raster)

Zoom Meeting

You are Viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079

ALEXANDRU BAR... 1077 C Caiali De... 1076 C Agafit... 1080 C Dragin... 1079 C Dan A.

View Options

ROUMANIA (RO) ROMÂNĂ (RO)

Multimedia-C.Sem1(3519ge)

Participanți

Educator

Competențe

Note

General

28 October - 4 October

5 October - 11 October

12 October - 18 October

19 October - 25 October

26 October - 1 November

2 November - 8 November

9 November - 15 November

16 November - 22 November

23 November - 29 November

30 November - 6 December

8 întrebări

Nu s-a prezentat încă niciun elev.

Marcat de 1 elev

8 întrebări

Alte întrebări

Modul de lucru

Desenarea unui cerc folosind un context grafic asociat unui element canvas se realizează prin intermediul funcției:

Alte opțiuni:

a. closedPath

b. circle

c. ellipse

d. arc

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Încheie încercarea

Începeți o previzualizare nouă

Urmărește pagină

Participants 103 Chat Share Screen Record Reactions

Leave

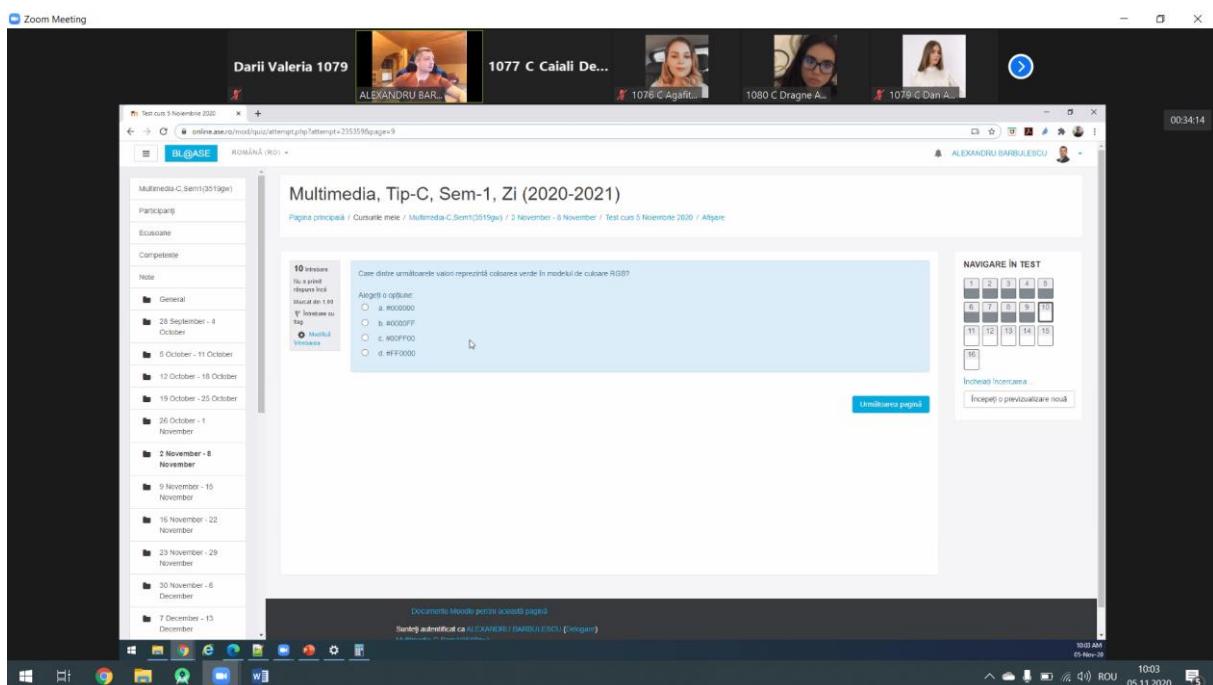
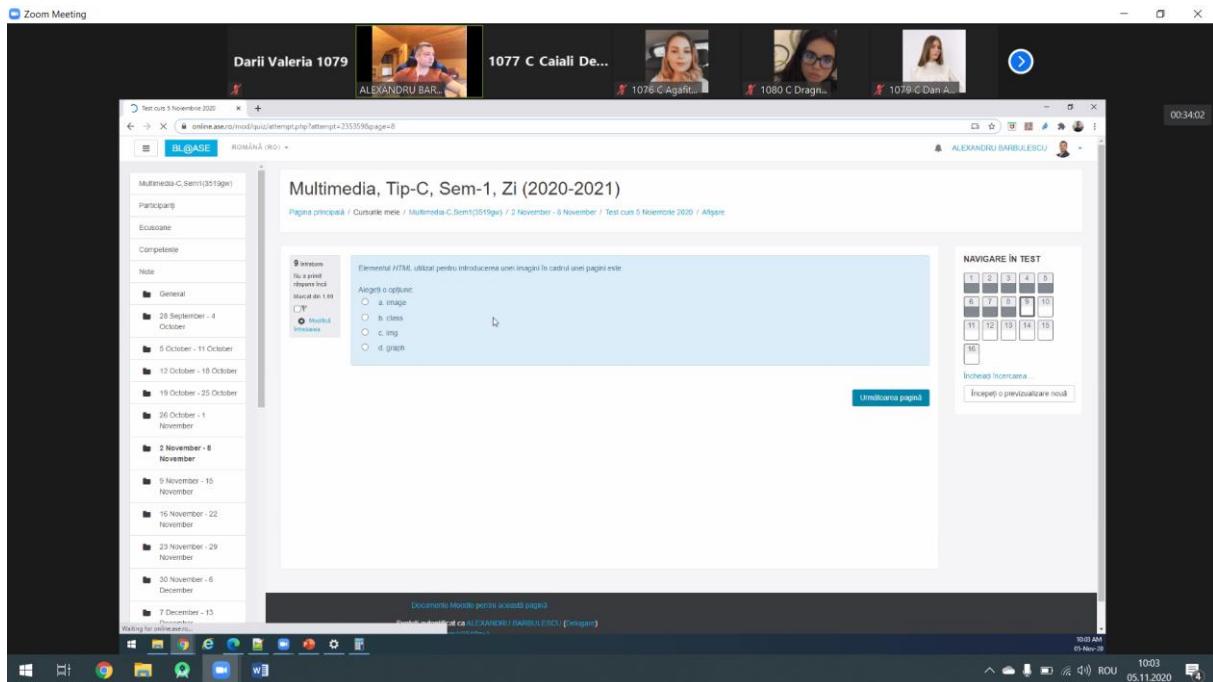
00:33:35

online.ase.ro/mod/quiz/attempt.php?attempt=255357&page=7

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519ge) / 2 November - 8 November / Test curs 5 Noiembrie 2020 / Atigare

Document Moodle pentru accesă pagină



Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 ALEXANDRU BAR... 1077 C Caiali De... 1076 C Agafit... 1080 C Dragne A... 1079 C Dan A...

View Options

online.ase.ro/mod/quic/attempt.php?attempt=25339&page=10 ROMÂNĂ (RO)

Multimedia-C_Sem1(3519gr)

Participanți

Elevație

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 Noiembrie

2 Noiembrie - 8 Noiembrie

9 Noiembrie - 15 Noiembrie

16 Noiembrie - 22 Noiembrie

23 Noiembrie - 29 Noiembrie

30 Noiembrie - 6 Decembrie

7 Decembrie - 13 Decembrie

11 întrebări
Nu a printat
Reprezentat
Marcat de 1 din 11

Ce este următoarea selecție CSS selectează elementul cu id-ul „#int” din document?

Alegeți o opțiune:

a. body{color}

b. #int

c. ion

d. ion

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Incheia încercarea

Începeți o previsualizare nouă

Unificarea pagină

Documente Moodle permis această pagină

SPAM Seria C
Alisa Darii Alăhă
Google Chrome + web.whatsapp.com

Participants 102 Chat Share Screen Record Reactions

Unmute Start Video

00:34:26

1093 ROU 05.11.2020

Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 ALEXANDRU BAR... 1077 C Caiali De... 1076 C Agafit... 1080 C Dragne A... 1079 C Dan A...

View Options

online.ase.ro/mod/quic/attempt.php?attempt=25339&page=11 ROMÂNĂ (RO)

Multimedia-C_Sem1(3519gr)

Participanți

Elevație

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 Noiembrie

2 Noiembrie - 8 Noiembrie

9 Noiembrie - 15 Noiembrie

16 Noiembrie - 22 Noiembrie

23 Noiembrie - 29 Noiembrie

30 Noiembrie - 6 Decembrie

7 Decembrie - 13 Decembrie

12 întrebări
Nu a printat
Reprezentat
Marcat de 1 din 12

Înainte pe pagina anterioră

Ce element HTML este utilizat pentru introducerea unei securite CSS în intenționul unui document?

Alegeți o opțiune:

a. link

b. class

c. css

d. style

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Incheia încercarea

Începeți o previsualizare nouă

Unificarea pagină

Documente Moodle permis această pagină

Participants 102 Chat Share Screen Record Reactions

Unmute Start Video

Leave

00:34:47

1094 ROU 05.11.2020

Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 1077 C Caiali De... 1079 C Dan O... 1080 C Drag... 1079 C Dan A...

View Options

ROMÂNĂ (RO)

Multimedia-C.Sem1(3519gr)

Participanți

Ecoane

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 Noiembrie

2 Noiembrie - 8 Noiembrie

9 Noiembrie - 15 Noiembrie

16 Noiembrie - 22 Noiembrie

23 Noiembrie - 29 Noiembrie

30 Noiembrie - 6 Decembrie

Test curs 5 Noiembrie 2020

online.ase.ro/mod/quizz/attempt.php?attempt=25359&page=12

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519gr) / 2 Noiembrie - 6 Noiembrie / Test curs 5 Noiembrie 2020 / Afisare

13 întrebări

Nu a printat întrebări încă

Marcat de 1 din 13

Înainte înainte

Înapoi

Modulă întrebări

Orafa vectorială folosește ca reprezentare sub formă de:

Alegeți o opțiune:

a. funcții matematice

b. matrice de puncte

c. transformările Fourier

d. coeficienți polinomiali

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Încearcă încă o dată

Încearcă pagină

Documente Moodle permis pentru accesă pagină

Participants 102 Chat Share Screen Record Reactions

Leave

00:35:10

Windows Taskbar: Unmute, Start Video, w3, Chat, Share Screen, Record, Reactions, Participants 102, Chat, Share Screen, Record, Reactions, Leave, 1094, ROU, 05.11.2020, 5

Zoom Meeting

You are viewing ALEXANDRU BARBULESCU's screen

Darii Valeria 1079 1077 C Caiali De... 1079 C Dan O... 1080 C Drag... 1079 C Dan A...

View Options

ROMÂNĂ (RO)

Multimedia-C.Sem1(3519gr)

Participanți

Ecoane

Competențe

Note

General

28 Septembrie - 4 Octombrie

5 Octombrie - 11 Octombrie

12 Octombrie - 18 Octombrie

19 Octombrie - 25 Octombrie

26 Octombrie - 1 Noiembrie

2 Noiembrie - 8 Noiembrie

9 Noiembrie - 15 Noiembrie

16 Noiembrie - 22 Noiembrie

23 Noiembrie - 29 Noiembrie

30 Noiembrie - 6 Decembrie

Test curs 5 Noiembrie 2020

online.ase.ro/mod/quizz/attempt.php?attempt=25359&page=13

Multimedia, Tip-C, Sem-1, Zi (2020-2021)

Pagina principală / Cursurile mele / Multimedia-C.Sem1(3519gr) / 2 Noiembrie - 6 Noiembrie / Test curs 5 Noiembrie 2020 / Afisare

14 întrebări

Nu a printat întrebări încă

Marcat de 1 din 14

Înainte înainte

Înapoi

Modulă întrebări

Care dintre următoarele modele de culori este de tip additiv?

Alegeți o opțiune:

a. CMYK

b. RGB

c. HSL

d. HSV

NAVIGARE ÎN TEST

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16

Încearcă încă o dată

Încearcă pagină

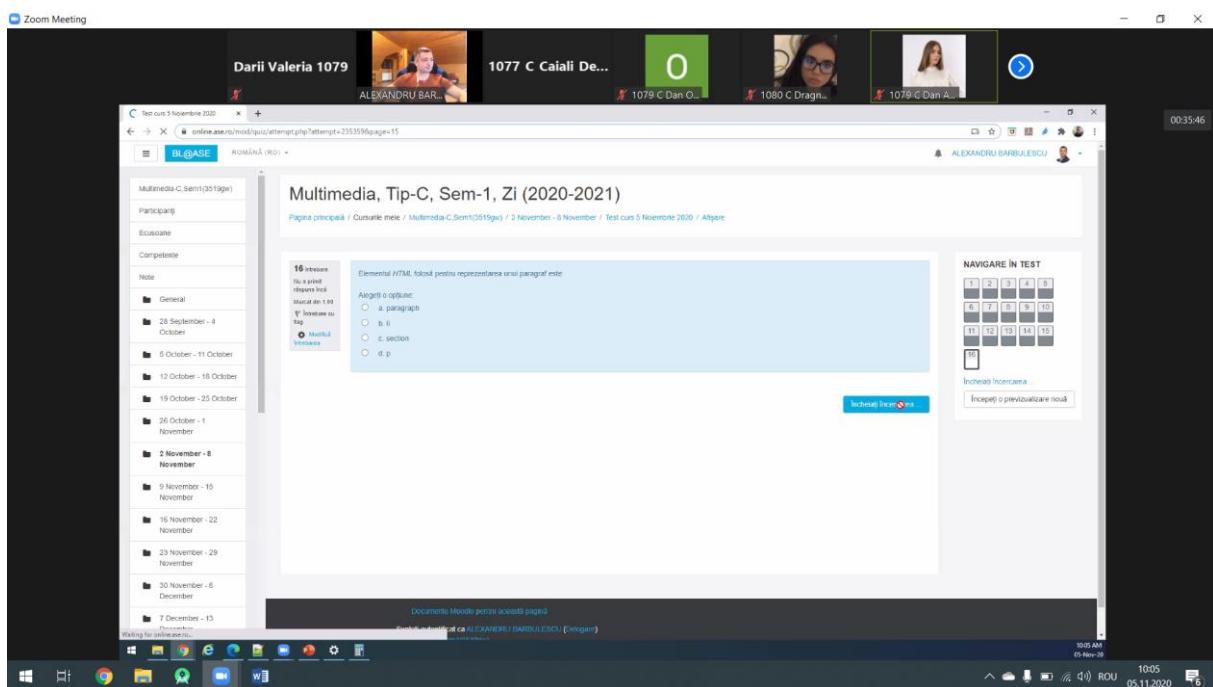
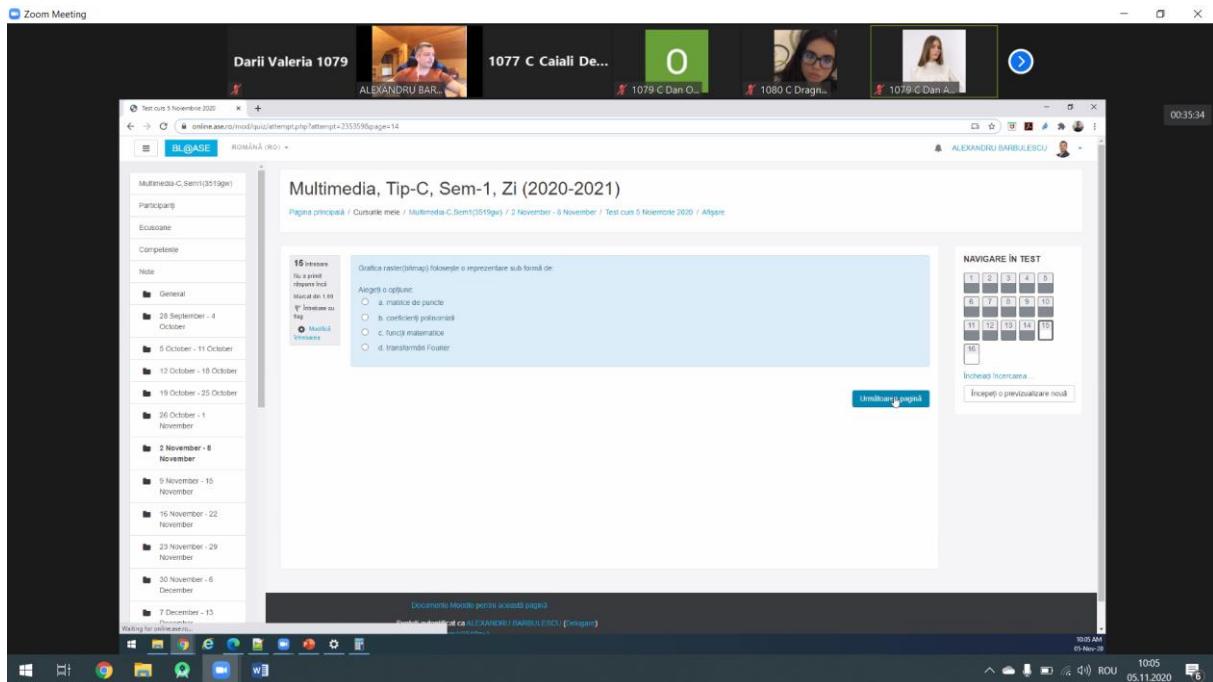
Documente Moodle permis pentru accesă pagină

Participants 102 Chat Share Screen Record Reactions

Leave

00:35:25

Windows Taskbar: Unmute, Start Video, w3, Chat, Share Screen, Record, Reactions, Participants 102, Chat, Share Screen, Record, Reactions, Leave, 1094, ROU, 05.11.2020, 5



Grile multimedia

1. Care dintre urmatoarele valori reprezinta culoarea rosu in modelul de culoare RGB ?

- a. #0000FF
- b. #00FF00
- c. **#FF0000**
- d. #000000

2. Care dintre urmatoarele valori reprezinta culoarea albastru in modelul de culoare RGB ?

- a. #000000
- b. #FF0000
- c. #00FF00
- d. #0000FF**

3. Care dintre urmatorii selectori CSS selecteaza elementul cu id-ul “maria” din document ?

- a. .maria
- b. body[maria]
- c. maria
- d. #maria**

4. Care este apelul corect pentru prelucrarea intr-un obiect a canvas-ului ?

- a. var b = canvas.getElementById(“test”)
- b. var b = document.getElementById(“test”)**
- c. var b = document.getCanvas(“test”)
- d. var b = document.getContext(“test”)

5. Formatul de stocare SVG este specific stocarii:

- a. fisierelor de tip sunet
- b. imaginilor necomprimate
- c. imaginilor vectoriale**
- d. imaginilor de tip raster

6. In CSS prin table#unu se selecteaza:

- a. Prima linie din table
- b. Tabelul cu id=unu**
- c. Tabelul cu o linie
- d. Un buton

7. Formatul de stocare a imaginii JPEG este specific:

- a. imaginilor vectoriale
- b. imaginilor de tip raster**
- c. fisierelor de tip sunet si video
- d. imaginilor necomprimate

8. Desenarea unui cerc folosind un context grafic asociat unui element canvas se realizeaza prin intermediul functiei:

- a. closedPath
- b. circle
- c. ellipse
- d. arc**

9. Elementul HTML utilizat pentru introducerea unei imagini in cadrul unei pagini este:

- a. image
- b. class
- c. img**
- d. graph

10. Care dintre urmatoarele valori reprezinta culoarea verde in modelul de culoare RGB ?

- a. #000000
- b. #0000FF
- c. #00FF00**
- d. #FF0000

11. Care dintre urmatorii selectori CSS selecteaza elementul cu id-ul “ion” din document ?

- a. body[ion]
- b. #ion**
- c. ion
- d. .ion

12. Ce element HTML este utilizat pentru introducerea unei sechete CSS in interiorul unui document ?

- a. link
- b. class
- c. css
- d. style**

13. Grafica vectoriala foloseste o reprezentare sub forma de:

- a. functii matematice**
- b. matrice de puncte
- c. transformari Fourier
- d. coeficienti polinomiali

14. Care dintre urmatoarele modele de culoare este de tip aditiv ?

- a. CMYK
- b. RGB**
- c. HSL
- d. HSV

15. Grafica raster (bitmap) foloseste o reprezentare sub forma de:

- a. matrice de puncte
- b. coeficienti polinomiali
- c. functii matematice
- d. transformari Fourier

16. Elementul HTML folosit pentru reprezentarea unui paragraf este:

- a. paragraph
- b. li
- c. section
- d. p

17. Care din urmatoarele metode nu exista implicit in JavaScript ?

- a. document.getElementsByTagName();
- b. document.getElementById();
- c. document.getElementByTagName();
- d. document.getElementsByName();

18. Care este tagul pentru introducerea in HTML a unui container pentru crearea grafica ?

- a. <paint>
- b. <style>
- c. <canvas>
- d. <graphics>

19. Care este metoda corecta pentru scriere in canvas ?

- a. font(x,y,text)
- b. fillText(text,x,y)
- c. fillstyle(text,x,y)
- d. paint(text,x,y)

20. Ce element HTML este utilizat pentru specificarea stilurilor CSS externe pentru un document ?

- a. CSS
- b. link
- c. style
- d. class

21. Care este varianta corecta de includere a unei imagini ase.jpg in cadrul unui document HTML ?

- a. <image src="ase.jpg" alt="Exemplu imagine">
- b. ase.jpeg
- c. < img src="ase.jpg" alt="Exemplu imagine">
- d.

22. Elementul HTML folosit pentru reprezentarea unui rand in cadrul unui tabel este:

- a. tr
- b. table
- c. cell
- d. td

23. Elementul HTML utilizat pentru introducerea unui element de grafica vectoriala in cadrul unei pagini este:

- a. graph
- b. svg**
- c. canvas
- d. vector

24. Stabilirea culorii rosu pentru umplerea urmatoarei figuri desenate pe contextul grafic c asociat unui element canvas se realizeaza prin:

- a. c.strokeStyle("red")
- b. c.fillStyle="red"**
- c. c.color="red"
- d. c.strokeStyle="red"

25. Formatul de stocare SVG este specific stocarii:

- a. imaginilor necomprimate
- b. fisierelor de tip sunet
- c. imaginilor vectoriale**
- d imagiilor de tip raster

26. Compresia video are la baza eliminarea redundantei:

- a. doar audio
- b. intra si inter-cadru**
- c. doar intra-cadru
- d. doar inter-cadru

27. Sunetul este definit ca:

- a. Energie magnetica intr-un mediu elastic
- b. O vibratie care se propaga printre un mediu material**
- c. Energie electrica propagate prin vid
- d. Energie electrica statica transmisa prin orice mediu

28. Compresia JPEG NU utilizeaza:

- a. compresia RLE
- b. compresia Huffman
- c. transformata cosinus discrete
- d. compresia LZW**

29. Care dintre urmatoarele caracteristici NU este specifica imaginilor vectoriale:

- a. mentine semnatica imaginii
- b. obiectele componente sunt descrise matematice
- c. fisierul imagine este mic
- d. imaginea este dependenta de scara de vizualizare**

30. Desenarea unui dreptunghi în canvas se realizează cu metoda:

- a. paint(x,y,a,b)
- b. fillRect(0,0,50,70)**
- c. fillStyle(0,0,10,30)
- d. square(x,y,a,b)

31. Care este apelul corect pentru preluarea unui obiect al canvasului ?

- a. var b =canvas.getElementById("test")
- b. var b =document.getElementById("test")**
- c. var b =document.getCanvas("test")
- d. var b =document.getContext("test")

32. Care este sintaxa corecta pentru introducerea in pagina a unui script extern ?

```
<script type="text/javascript" src="scriptulMeu.js"></script>
```

33. Ce atribut este utilizat pentru specificarea stilurilor CSS intr-un element HTML?

- a. css
- b. link
- c. style**
- d. class

34. Care este varianta corecta pentru introducerea in HTML a unui css extern ?

```
<link rel="stylesheet" type="text/css" href="stilulMeu.css">
```

35. Care dintre urmatoarele sevante respectă sintaxe CSS ?

- a. {body: color-black;}
- b. body {color: black;}**
- c. {body; color:black;}
- d. body:color=black;

36. Ce elemente HTML vor fi modificate prin intermediul selectorului img.all ?

- a. toate imaginile din cadrul documentului
- b. toate elementele care au atributul id="img.all"
- c. toate elementele care au atributul class="img.all"
- d. toate imaginile care au atributul class="all"**

37. jQuery este:

- a. un limbaj de programare
- b. un limbaj de acces la baza de date
- c. o biblioteca JavaScript**
- d. un tag

38. Desenarea unui cerc folosind un context grafic asociat unui element canvas se realizează prin intermediul functiei:

- a. circle
- b. arc**
- c. eclipse
- d. closedPath

39. Care din urmatoarele elemente svg este utilizat pentru gruparea unui set de elemente fara a le afisa ?

- a. rect
- b. defs
- c. g
- d. svg

40. Care dintre urmatoarele valori nu este o valoare acceptabila pentru proprietatea CSS position ?

- a. static
- b. relative
- c. fixed
- d. absolute
- e. float

41. Elementul HTML utilizat pentru introducerea unui element de grafica raster in cadrul unei pagini este:

- a. graph
- b. svg
- c. canvas
- d. vector

BARBULESCU

1. Care dintre urmatoarele valori reprezinta culoarea **rosu** in modelul de culoare RGB?
a) #0000FF
b) #00FF00
c) #FF0000 -> (255,0,0)
d) #000000

2. Care dintre urmatoarele valori reprezinta culoarea **albastru** in modelul de culoare RGB?
a) #000000
b) #FF0000
c) #00FF00
d) #0000FF -> (0,0,255)

3. Care dintre urmatorii selectori CSS selecteaza elementul cu id-ul „maria” din document?
a) .maria
b) body[maria]
c) maria
d) #maria

4. Care este apelul corect pentru preluarea intr-un obiect a canvas-ului?
a) var b = canvas.getElementById("test")
b) var b = document.getElementById("test")
c) var b = document.getcanvas("test")
d) var b = document.getcontext("test")

5. Formatul de stocare SVG este specific stocarii:
a) fisierelor de tip sunet
b) imaginilor necomprimate
c) imaginilor vectoriale
d) imaginilor de tip raster

6. In CSS prin table#unu se selecteaza:
a) Prima linie din table
b) Tabelul cu id=unu
c) Tabelul cu o linie
d) Un buton

7. Formatul de stocare a imaginii JPEG este specific:
a) imaginilor vectoriale
b) imaginilor de tip raster

- c) fisierelor de tip sunet si video
 - d) imaginilor necomprimate
8. Desenarea unui cerc folosind un context grafic asociat unui element canvas se realizeaza prin intermediul functiei:
- a) closedPath
 - b) circle
 - c) ellipse
 - d) arc**
9. Elementul HTML utilizat pentru introducerea unei imagini in cadrul unei pagini este:
- a) image
 - b) class
 - c) img**
 - d) graph
10. Care dintre urmatoarele valori reprezinta culoarea verde in modelul de culoare RGB?
- a) #000000
 - b) #0000FF
 - c) #00FF00 -> (0,255,0)**
 - d) #FF0000
11. Care dintre urmatorii selectori CSS selecteaza elementul cu id-ul „ion” din document?
- a) body[ion]
 - b) #ion**
 - c) ion
 - d) .ion
12. Ce element HTML este utilizat pentru introducerea unei secvente CSS in interiorul unui document?
- a) link
 - b) class
 - c) css
 - d) style**
13. Grafica vectoriala foloseste o reprezentare sub forma de:
- a) functii matematice**
 - b) matrice de puncte
 - c) transformari Fourier
 - d) coeficienti polinomiali

14. Care dintre urmatoarele modele de culoare este de tip aditiv?

- a) CMYK
- b) RGB
- c) HSL
- d) HSV

15. Grafica raster (bitmap) foloseste o reprezentare sub forma de:

- a) matrice de puncte
- b) coeficienti polinomiali
- c) functii matematice
- d) transformari Fourier

16. Elementul HTML folosit pentru reprezentarea unui paragraf este:

- a) paragraph
- b) li
- c) section
- d) p

AU RASPUNSURILE DE PE PLATFORMA!

1. Care dintre urmatoarele NU sunt formate de stocare pentru imagini vectoriale?
a) PNG
b) SHP
c) DXF
d) EPS
2. Metoda corecta pentru desenarea unui text pe un element de tip canvas este:
a) appendText(text, x, y);
b) writeText(text, x, y);
c) drawText(text, x, y);
d) fillText(text, x, y);
3. Care dintre urmatoarele atribute nu este utilizat in mod obisnuit pe un element audio?
a) loop
b) showControls
c) volume
d) autoplay
5. CMYK este un model:
a) aditiv
b) subtractiv
c) nu este un model utilizat in multimedia
d) multiplicativ
6. Care dintre urmatoarele nu este o metoda de transformare disponibila pentru CanvasRenderingContext2D(canvas.getContext("2d")):
a) rotatie
b) translatie
c) deplasare
d) scalare

4. Care dintre urmatoarele raspunsuri reflectă conținutul vectorului data, obținut după cum urmează, în cazul imaginii de mai jos.

```
const context = canvas.getContext("2d")
const imageData = context.getImageData(0,0,2,2)
const data = imageData.data;
```

4 întrebare

Corect

Marcat 1,00 din 1,00

▼ Întrebare cu flag

Care dintre următoarele răspunsuri reflectă conținutul vectorului data, obținut după cum urmează, în cazul imaginii de mai jos.

```
const context = canvas.getContext("2d")
const imageData =
context.getImageData(0,0,2,2)
const data = imageData.data;
```

Alegeți o opțiune:

- a. [255,255,255,255, 197,52,144,255, 255,0,0,255, 0,0,255]
- b. [255,0,0,255, 255,255,255,255, 0,0,255, 197,52,144,255]
- c. [255,0,0,255, 0,0,0,255, 255,255,255, 197,52,144,255]
- d. [255,0,0,255, 0,0,0,255, 255,255,255, 197,52,144,255]

The correct answers are: [255,0,0,255, 0,0,0,255, 255,255,255, 197,52,144,255], [255,0,0,255, 0,0,0,255, 255,255,255, 197,52,144,255]

5 întrebare

7. Care dintre urmatoarele afirmații nu este adevarata în cazul graficii vectoriale:

- a) contin forme geometrice precum puncte, linii, curbe etc.
- b) un set de comenzi este folosit pentru a desena imaginea
- c) calitatea lor este afectată atunci cand sunt scalate**
- d) pentru imagini simple, acestea ...

8. Urmatoarele proprietăți sunt disponibile pentru CanvasRenderingContext2D(canvas.getContext("2d")):

- a) textAlign**
- b) strokeStyle**
- c) font**
- d) lineWidth**

9. Desenarea conturului unui dreptunghi pe un element de tip <canvas> se poate realiza cu urmatoarele metode aferente CanvasRenderingContext2D:

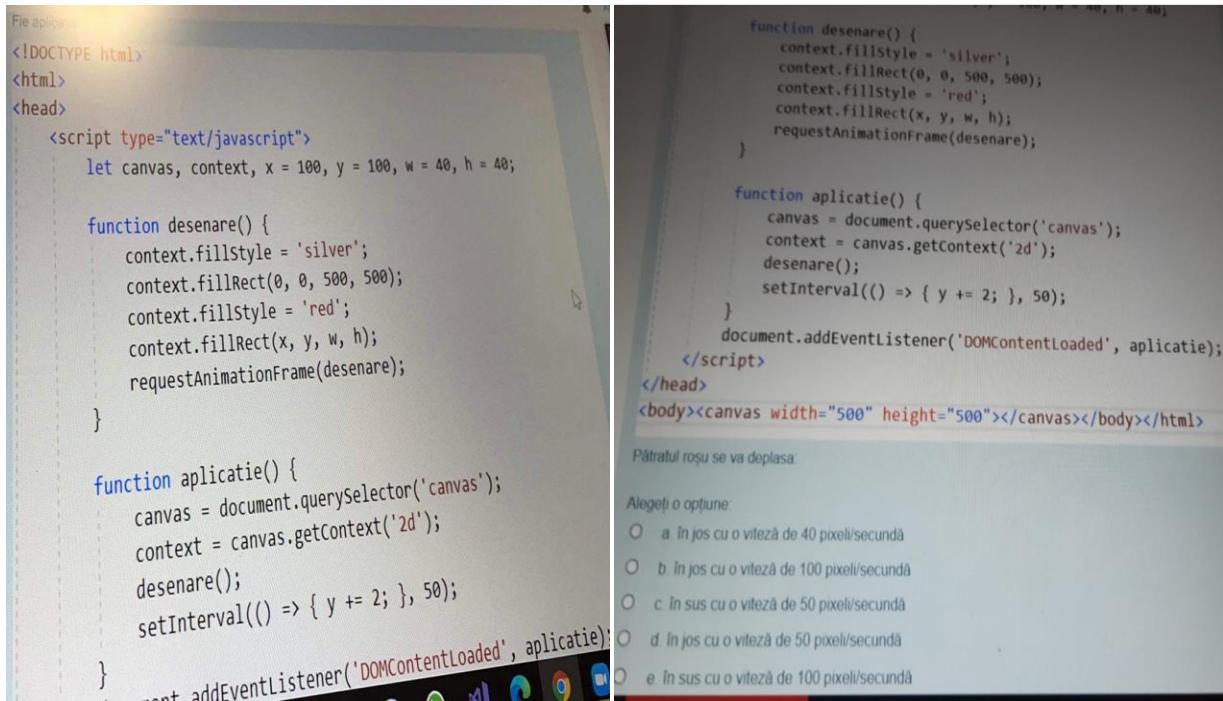
- a) `paintRect(x, y, width, height)`
- b) `strokeRect(x, y, width, height)`**
- c) `drawRect(x, y, width, height)`
- d) `rect(x, y, width, height)`

10. Care dintre urmatoarele metode nu există implicit în JavaScript?

- a) `document.getElementsByName();`
 - b) `document.getElementByTagName();`**
 - c) `document.getElementsByTagName();`
 - d) `document.getElementById();`
- !!! DAR există `document.getElementsByTagName()`**

CELE DE MAI SUS AVEAU RASPUNSURILE DE PE PLATFORMA!

IONITA



```
Fie aplicatia
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
    let canvas, context, x = 100, y = 100, w = 40, h = 40;

    function desenare() {
        context.fillStyle = 'silver';
        context.fillRect(0, 0, 500, 500);
        context.fillStyle = 'red';
        context.fillRect(x, y, w, h);
        requestAnimationFrame(desenare);
    }

    function aplicatie() {
        canvas = document.querySelector('canvas');
        context = canvas.getContext('2d');
        desenare();
        setInterval(() => { y += 2; }, 50);
    }
</script>
</head>
<body><canvas width="500" height="500"></canvas></body></html>
```

Păratul roșu se va deplasa.

Alegeți o opțiune:

- a. în jos cu o viteză de 40 pixeli/secundă
- b. în jos cu o viteză de 100 pixeli/secundă
- c. în sus cu o viteză de 50 pixeli/secundă
- d. în jos cu o viteză de 50 pixeli/secundă
- e. în sus cu o viteză de 100 pixeli/secundă

Patratul rosu se va deplasa:

- a) în sus cu o viteza de 100 pixeli/secunda
- b) în jos cu o viteza de 50 pixeli/secunda
- c) în jos cu o viteza de 100 pixeli/secunda
- d) **in jos cu o viteza de 40 pixeli/secunda**
- e) în sus cu o viteza de 50 pixeli/secunda

Care dintre urmatoarele valori reprezinta culoarea rosu in modelul de culoare HSL?

- a) (0, 100%, 0%)
- b) (120, 100%, 100%)
- c) **(0, 100%, 50%)**
- d) (120, 100%, 0%)
- e) (240, 0%, 0%)

Fie **a** o referinta la un element HTML de tip audio. Pozitionarea la jumatarea secenteii audio se realizeaza prin:

- a) a.time = a.duration * 0.5
- b) a.currentTime = 0.5
- c) a.time = a.duration ? a.duration * 0.5 : 0
- d) **a.currentTime = a.duration ? a.duration / 2 : 0** ????????????????
- e) a.position = 1 / 2

C este o referinta la un canvas de latime 20 pixeli si inaltime 10 pixeli.

Fie codul:

```
let context = C.getContext("2d");
let imageData = context.getImageData(0, 0, W, H);
```

Componenta de culoare verde pentru pixelul de la linia 3, coloana 4 este accesata prin (numararea liniilor si coloanelor incepe de la 0):

- a) imageData.data[80]
- b) imageData[97]
- c) **imageData.data[81]**
- d) imageData[81]
- e) imageData.data[97]

Fie documentul HTML:

```
<!DOCTYPE html>

<html>

<head></head>

<body>

<svg>

<rect id="r1" x="10" y="10" width="10" height="10"></rect>

<rect id="r2" x="10" y="10" width="10" height="10"></rect>

</svg>

</body></html>
```

Pozitionarea dreptunghiului **r2** imediat sub dreptunghiul **r1** se poate realiza prin urmatoarea secventa JavaScript:

- a) `document.querySelector('#r2').setAttribute('y', '10');`
- b) `document.querySelector('#r1').y = 20;`
- c) `document.querySelector('r2').setAttribute('y', '20');`
- d) `document.querySelector('#r1').setAttribute('x', '20');`
- e) `document.querySelector('#r2').setAttribute('y', '20');`

GRILE ADUNATE

1. Care dintre algoritmii de mai jos este doar fara pierdere de informatie?
 - a) **FLAC**
 - b) Audio Mpeg Layer 3
 - c) WMA
 - d) Audio Mpeg-4

2. Care din cele de mai jos NU este cerinta pentru compresia video
 - a) Acces random la sechete
 - b) Posibilitate editare
 - c) Corectie/evitare erori
 - d) **Generarea cadrelor intermediare**

3. Care din cele de mai jos NU este cerinta pentru compresia video
 - a) Reverse playback
 - b) **Compresia sunetului fara pierdere de informatie**
 - c) Sincronizare audio video
 - d) Viteza mare codare/decodare

4. Compresia video are la baza eliminarea redundantei:
 - a) doar audio
 - b) doar inter-cadru
 - c) **intra si inter-cadru**
 - d) doar intra-cadru

5. In cadrul compresiei JPEG pentru imagini este utilizata:
 - a) Transformata Fourier Discreta
 - b) Compresia Lempel-Ziv-Welch
 - c) **Transformata Cosinus Discreta**
 - d) Transformata Sinus Discreta

6. Care este atributul care face ca sunetul in tag-ul audio sa ruleze continuu pana cand este oprit de catre utilizator?
 - a) controls
 - b) **loop**

7. Care este elementul introdus in html5 care permite redarea nativa a fisierelor de sunet?
- a) <sound>
 - b) <audio> ?
 - c) <video>
 - d) <play>
8. Care informatie NU este adevarata cu privire la Group of Pictures (GOP)
- a) Sunt secvențe repetitive I,P,B
 - b) Reprezinta cadre in ordinea aparitiei
 - c) Nu sunt comprimate
 - d) Incep mereu cu care I
9. Pentru ce se utilizeaza CSS in contextul HTML5?
- a) Color style sheets
 - b) Clear style sheets
 - c) Computer style sheets
 - d) Cascading style sheets
10. Care dintre urmatorii selectori CSS selecteaza elementul cu id-ul „ion” din document?
- a) body[ion]
 - b) #ion
 - c) ion
 - d) .ion
11. Care dintre urmatoarele formate este utilizat pentru stocarea sunetului necomprimat?
- a) JPG
 - b) WAVE
 - c) MP3
 - d) BMP
12. Sunetul este definit ca:
- a) energie magnetica intr-un mediu elastic
 - b) o vibratie care se propaga printre un mediu material
 - c) energie electrica propagata prin vid
 - d) energie electrica statica transmisa prin orice mediu

13. Ce atribut este utilizat pentru specificarea stilurilor CSS intr-un element HTML?

- a) css
- b) link
- c) **style**
- d) class

14. Care este variabila corecta de includere a imaginii ase.jpg in cadrul unui document HTML?

- a) <image src="ase.jpg" alt="Exemplu imagine">
- b) ase.jpg
- c) ****
- d)

((Care este VARIANTA corecta de includere a imaginii ase.jpg in cadrul unui document HTML?))

15. Elementul HTML folosit pentru reprezentarea unui rand in cadrul unui tabel este:

- a) **tr**
- b) table
- c) cell ---- mai e o varianta cu row aici
- d) td
- e) tc

<tr> de la table row

!!! <td> = celula cu date, celula standard si <th> este celula din header

((Elementul HTML folosit pentru reprezentarea unei linii in cadrul unui tabel este:))

16. Elementul HTML utilizat pentru introducerea unui element de grafica **raster** in cadrul unei pagini este:

- a) graph
- b) svg
- c) **canvas**
- d) vector

17. Stabilirea culorii rosu pentru **umplerea** urmatoarei **figuri desenate** pe contextul grafic c asociat unui element canvas se realizeaza prin:

- a) `c.strokeStyle("red")`
- b) `c.fillStyle = "red"`
- c) `c.color = "red"`
- d) `c.strokeStyle = "red"`

18. Stabilirea culorii rosu pentru **desenarea** urmatoarei **linii** pe contextul grafic c asociat unui element canvas se realizeaza prin:

- a) `c.strokeStyle("red")`
- b) `c.fillStyle = "red"`
- c) `c.color = "red"`
- d) `c.strokeStyle = "red"`

19. Care dintre urmatoarele metode NU exista implicit in JavaScript?

- a) `document.getElementsByName()`
- b) `document.getElementByName()`**
- c) `document.getElementsByTagName()`
- d) `document.getElementById()`

!!! DAR exista `document.getElementsByTagName()`

20. Se creaza jQuery ~~`$(“p.class1, #p1”).css(“border”, “solid green”);`~~ modifica culoarea chenarului pentru:

- a) toate elementele de tip paragraf (`p`) din pagina care au id-ul `#p1` si clasa CSS `class1`
- b) selectorul jQuery este incorrect
- c) toate elementele de tip paragraf (`p`) din pagina care au fie id-ul `#p1`, fie clasa CSS
- d) toate elementele din pagina de tip paragraf (`p`) avand clasa CSS `class1` si elementul din pagina cu id-ul `p1`**

21. Echivalentul jQuery pentru: `document.getElementById(‘elementId’)` este:

- a) `$("elementId");`
- b) `$(".elementId");`
- c) `$("#elementId");`**
- d) `$("<elementId>");`

22. jQuery este:

- a) Un limbaj de programare
- b) Un limbaj de acces la baza de date
- c) O biblioteca JavaScript**
- d) Un tag

23. Desenarea unui dreptunghi în canvas se realizează cu metoda:

- a) `paint(x,y,a,b)`
- b) `fillRect(0,0,50,70)`
- c) `fillStyle(0,0,10,30)`
- d) `square(x,y,a,b)`

24. Care este apelul corect pentru preluarea unui obiect din canvas-ului?

- a) `var b = canvas.getElementById("test")`
- b) `var b = document.getElementById("test")`
- c) `var b = document.getcanvas("test")`
- d) `var b = document.getContext("test")`

25. Care este sintaxa corecta pentru introducerea in pagina a unui script extern?

- a) `<src type="text/javascript" src="scriptulmeu.js">`
- b) `<script type="text/javascript" src="scriptulmeu.js">`
- c) `<script type="text/javascript" href="scriptulmeu.js">`
- d) `<script type="text/javascript" name="scriptulmeu.js">`

26. Care este tagul pentru introducerea in HTML a unui container pentru crearea de grafica?

- a) `<paint>`
- b) `<style>`
- c) `<canvas>`
- d) `<graphics>`

((Care este tagul pentru introducerea in HTML a unui container pentru crearea grafica?))

27. Care este metoda corecta pentru scriere in canvas?

- a) `font(x,y,text)`
- b) `fillText(text,x,y)`
- c) `fillstyle(text,x,y)`
- d) `paint(text,x,y)`

28. Formatul de stocare SVG este specific stocarii:

- a) fisierelor de tip sunet
- b) imaginilor necomprimate
- c) `imaginilor vectoriale`
- d) imaginilor de tip raster

29. Compresia video are la baza eliminarea redundantei:

- a) doar audio
- b) doar inter-cadru
- c) intra si inter-cadru
- d) doar intra-cadru

30. Sunetul este definit ca:

- a) energie magnetica intr-un mediu elastic
- b) o vibratie care se propaga prin un mediu material
- c) energie electrica propagata prin vid
- d) energie electrica statica transmisa prin orice mediu

31. Compresia JPEG NU utilizeaza:

- a) compresie RLE
- b) compresia Huffman
- c) transformata cosinus discreta
- d) compresia LZW

32. Care dintre urmatoarele caracteristici NU este specifica imaginilor vectoriale:

- a) mentine semnatica imaginii
- b) obiectele componente sunt descrise matematic
- c) fisierul imagine este mic
- d) imaginea este dependenta de scara de vizualizare

33. Elementul HTML utilizat pentru introducerea unui element de grafica **vectoriala** in cadrul unei pagini este:

- a) graph
- b) svg
- c) canvas
- d) vector

!!! <canvas> pentru grafica raster

34. Care este varianta corecta pentru introducerea in HTML a unui CSS extern?

- a) <link rel="stylesheet" type="text/css" href="stilulmeu.css">
- b) <style src="stilulmeu.css" />
- c) <stylesheet>stilulmeu.css</stylesheet/>
- d) <script style="stilulmeu.css">

Care este varianta corecta pentru introducerea in HTML a unui CSS **extern?**

- a) <**link** rel="stylesheet" type="text/css" href="stilulmeu.css">

!!!DAR daca este **intern avem <style src="...">**

35. Pierderea de informatie in cazul compresiei JPEG este influentata de:

- a) **alegerea matricei de cuantizare**
- b) calitatea imaginii sursa
- c) parametrii compresiei Huffman
- d) parametrii compresiei RLE
- e) modalitatea de aplicare a transformatiei cosinus discreta

36. Ce element HTML este utilizat pentru specificarea stilurilor CSS externe pentru un document?

- a) css
- b) link**
- c) style
- d) class

1. Grafica vectoriala foloseste o reprezentare sub forma de: => functii matematice
2. Compresia MPEG 1-2 LAYER III foloseste codificarea perceptuala si un model psihacustic
3. GIF este un format specific => imaginilor de tip raster
4. Care dintre urmatoarele este un format de stocare pentru imagini raster? => JPG
5. Stabilirea culorii pentru culoarea urmatoarelor linii din contextul grafic:
⇒ context.fillStyle="red"

37. Care dintre urmatoarele sevante respecta sintaxe CSS?

- a) body:color=black
- b) **body(color:black)**
- c) (body, color.black)
- d) (body.color=black(body))

38. Ce elemente HTML vor fi modificate prin intermediul **selectorului img.all?**

- a) Toate imaginile din cadrul documentului
- b) toate elementele care au atributul id="img.all"
- c) toate elementele care au atributul class="img.all"
- d) **toate imaginile care au atributul class="all"**

39. Care din urmatoarele elemente svg este utilizat pentru gruparea unui set de elemente fara a le afisa?

- a)rect
- b)defs
- c)**g**
- d) svg

40. Care dintre urmatoarele valori nu este o valoare acceptabila pentru proprietatea CSS position?

- a) static
- b) relative
- c) fixed
- d) absolute
- e) **float**

Which is the HTML element used to draw a raster graphic on a web page? => **canvas**

Which of the following tags does include the “stud.jpg” image, in an HTML document? =>

Which of the following CSS selectors refers an element with “my” id, in an HTML document? => **#my**

Which colour model does use a double-cone representation for its colour space? => **HLS**

Which is the output of the following chunk of HTML & CSS code?

(ceva cu span)

What is the output of the following chunk of HTML & CSS code?

```
<div id="test">
  <span>Text</span>
</div>
<style>
div#test span { color: green; }
div span { color: blue; }
span { color: red; }
</style>
```

⇒ It colours the text in green.

Check all arguments accepted by the DrawImage JavaScript function:

⇒ a canvas element, an image, a video element

Check all the processing operations involved in JPEG compression algorithm:

⇒ transformare cosinus discreta, cuantizare, block splitting

Select ALL statements depicting DOM, from the following:

- ⇒ DOM is the acronym for Document Object Model
- ⇒ DOM provides the way to programmatically access HTML structure in JavaScript
- ⇒ HTML of every web page is turned into a DOM representation by the browser

Select all characteristics of a raster image, from the following

⇒ The raster image is figured as a matrix of points.

- ⇒ Raster image file size depends on the image size and the colour depth

Select ALL solutions for incorporating CSS in HTML

- ⇒ ?defined within a style block, in the head section of a web page
⇒ (tag link, tag style (in head), atribut style pt inline (in body))
⇒ c) este ambiguu -> Coded in the body of the web page

Select ALL correct statements referring to the FlexBox model

Select ALL correct statements referring to the FlexBox model.

Alegeți una sau mai multe opțiuni:

- a. It uses grid-column and grid-row properties to change the size of items.
- b. It is activated using display property
- c. It is used to change the horizontal or vertical layout of HTML elements, in a web page
- d. It can be used to programmatically change the order of the HTML elements.
- e. It is appropriate for creating a responsive web application

b, d, e, c? e ambiguu

Select ALL primary colours of the subtractive colour model.

cyan, magenta, yellow

What purpose has setTimeout JavaScript function?

Răspuns:

metoda setTimeout apeleaza o functie

sau evaluateaza o expresie dupa un numar specificat de milisecunde. 1000ms=1s si functia ce va fi executata va fi rulata o singura data. Pt a repeta executia functiei putem folosi metoda setInterval. Pt a preveni functia din a rula putem folosi metoda clearTimeout

setTimeout(function(){ alert("Hello"); }, 3000); - afiseaza o fereastra de alerta cu textul hello dupa 3 sec

What is HyperVideo?

Răspuns:

Care este valoarea lipsă din imaginea target aplicând kernelul normalizat centrat pe elementul de culoare galbenă din imaginea sursă?

Normalized Kernel

0	1/10	0
1/5	2/5	1/5
0	1/10	0

2	9	4	1	9
2	2	9	1	6
4	1	3	6	6
4	4	9	6	1
4	4	4	3	6

Source Image

Convolution

2	9	4	1	9
2	2	9	1	6
4	1	6	6	6
4	4	9	6	1
4	4	4	3	6

Target Image

Alegeți o opțiune:

- a. 4.4
- b. 2/5
- c. 3.4
- d. 4

Care este valoarea lipsă din imaginea target aplicând kernelul normalizat centrat pe elementul de culoare galbenă din imaginea sursă?

Normalized Kernel

0	1/10	0
1/5	2/5	1/5
0	1/10	0

2	9	4	1	9
2	2	9	1	6
4	1	6	6	6
4	4	9	6	1
4	4	4	3	6

Source Image

Convolution

2	9	4	1	9
2	2	9	1	6
4	1	6	6	6
4	4	9	6	1
4	4	4	3	6

Target Image

Alegeți o opțiune:

- a. 3.4
- b. 4.4
- c. 4/5
- d. 5.6

TEORIE

1. Sunetul - formate de stocare

- formate fara compresie: WAV, AIFF
- formate de compresie fara pierderi: APE, MPEG4, FLAC
- formate de compresie cu pierderi: MP3, AAC, WMA, OGG

Astept continuare...

2. Tehnici de animatie

- Tehnica filmului
- Tehnica cadrelor cheie
- Tehnica schimbarii de culoare
- Tehnica schimbarii de forma

Astept continuare...

3. Grafica raster (BITMAP) - reprezentare si caracteristici

Reprezentata ca o matrice de puncte

Fiecare punct(pixel) retine informatii despre culoarea sa

Culorile sunt stocate conform modelului de culoare al imaginii

Caracteristici:

1. Rezolutia : reprezinta nr de linii si coloane ale matricii de reprezentare sau numarul total de pixeli
2. Adancimea de culoare : cat de multa informatie este retinuta despre culoare

Utilizare:

- Reprezentarea imaginilor pe monitor
- Reprezentarea capturilor externe

Avantaje:

- Poate reprezenta orice fel de imagine

Dezavantaje:

- Nu este scalabila
- Informatia este saracocioasa, nu tine cont de semantica
- Dimensiune mare

Formate: BMP, JPEG, GIF, TIFF

TEHNICI DE ANIMATIE | GRAFICA RASTER (BITMAP)

SUNET – FORMATE DE STOCARE – COMPRESIE / FARÀ COMPRESIE

GRAFICA RASTER (BITMAP) - Reprezentata ca o matrice de puncte

Fiecare punct (pixel) retine informatii despre culoarea sa

Culorile sunt stocate conform modelului de culoare al imaginii

Caracteristici:

1. Rezolutia : reprezinta nr de linii si coloane ale matricii de reprezentare sau numarul total de pixeli
2. Adancimea de culoare : cat de multa informatie este retinuta despre culoare

Utilizare:

-Reprezentarea imaginilor pe monitor

-Reprezentarea capturilor externe

A/D

- + poate reprezenta orice fel de imagine
- nu este scalabila
- informatia este saracicioasa, nu tine cont de semantica
- dimensiune mare

Formate: BMP / JPEG / GIF / TIFF

SUNETUL

TEHNICI DE ANIMATIE

MODELE DE CULOARE

Model de culoare

Model matematic care descrie modalitatea de reprezentare a culorilor sub formă de tupluri numerice

Exemple: RGB, CMY

Spațiu de culori

Modelul de culoare împreună cu instrucțiunile de reprezentare fizică, Exemple: sRGB, AdobeRGB, Pantone

Caracteristici

Țin seama de modalitatea de percepere a luminii de către ochiul uman, Culori de bază albastru (S), verde (M) și roșu (L)

MODELUL ADITIV

Bazat pe culorile de bază roșu, verde și albastru, Culoarea variază în funcție de dispozitiv, (în lipsa unui spațiu de culoare)

Spatii de culoare RGB

sRGB - Dezvoltat de HP și Microsoft, Standard pentru monitoare / imprimante / web, Utilizat ca standard implicit

AdobeRGB - Dezvoltat de Adobe, Acoperă aproape complet spațiul de culori CMYK

MODELUL HSL

Reprezentare sub formă de coordonate cilindrice, Bazat pe aceleași culori de bază: Red – 0grade, Green – 120grade, Blue – 240grade, Axa centrală – tonuri de gri, Utilizat în special pentru selecție de culoare

Modelul substractiv

Modelul CMY(K) - Culori utilizate: cyan, magenta, yellow, Maschează culorile pe o suprafață albă, Utilizare: materiale tipărite

CSS – SPECIFICATOR CULORI

Format hexazecimal: #rgb sau #rrggbb , r,g,b sunt cifre în baza 16

Format RGB: `rgb(red, green, blue)` sau `rgba(red, green, blue, alpha)`, red, green, blue – numere de la 0 la 255 sau procente, alpha – număr între 0 – transparent și 1 – opac sau procent

Format HSL: `hsl(hue, saturation, lightness)` sau `hsla(hue, saturation, lightness, alpha)`, hue – număr de la 0 la 360, saturation, lightness – procente, alpha – număr între 0 – transparent și 1 – opac sau procent

PALETE DE CULORI

Tabel de corespondență: index – tuplu (R, G, B), Utilizare: Reducerea cantității de informație necesară pentru reprezentarea culorilor, Pretabilă pentru imagini cu un număr redus de culori (exemplu: diagrame fără gradienți)

GRAFICA RASTER

Grafica raster (bitmap, matriceală): Reprezentare sub formă de matrice de puncte, Fiecare punct (denumit pixel) stochează informația de culoare, Culorile sunt stocate conform unui model de culoare, direct sau prin intermediul unei palete de culori

Caracteristici: Rezoluție (numărul de linii și coloane stocate în matrice, numărul total de pixeli sau densitate), Adâncime de culoare (cantitatea de informație stocată de către fiecare pixel)

Utilizare: Reprezentare imagine pe monitor, Captare imagini din surse externe

Avantaje și dezavantaje: Poate reprezenta orice imagine, Codaj sărac în informație (nu ia în considerare semantica imaginii), Dimensiune mare, Nu se pot adapta unei scări variabile de vizualizare

FORMATE DE STOCARE

BMP (Microsoft Windows Bitmap): Formatul standard de stocare pe platforma Microsoft Windows, Suportă date necomprimate sau comprimate folosind algoritmul RLE, Monocromă sau în culori pe 4, 8, 16, 24 sau 32 de biți, Suportă palete de culori

JPEG (Joint Photographic Experts Group): Stocare comprimată cu pierdere de informație conform standardului JPEG, Rate de compresie diferite selectabile de către utilizator, Utilizat pentru imagini fotografice (cu gradații fine de culoare),

Nu este potrivit pentru text, linii sau alte imagini care prezintă un contrast foarte mare, Editări multiple (se pierde calitate la fiecare etapă de compresie / decompresie)

GIF (Graphics Interchange Format): Folosit în special pentru transferul imaginilor de maxim 64K x 64K, Pretabil pentru diagrame, text logo-uri (contrast puternic și număr limitat de culori), Suportă maxim 256 culori prin intermediul unei palete de culori, Poate stoca mai multe cadre (pentru animație), Algoritm de compresie fără pierdere de informație Lempel-Ziv-Welch (LZW)

TIFF (Tag Image File Format): Format portabil și extensibil utilizat în special pentru imagini scanate, Suportă stocarea mai multor imagini într-un singur fișier

Suportă mai mulți algoritmi de compresie (RLE, LZW sau JPEG)

Compresia RLE: (nr aparitii, valoare), Rată mică de compresie, Se pretează pentru imagini cu zone mari de aceeași culoare, Utilizat în special pentru fișiere BMP cu paletă de culori

Compresia LRW: Algoritm de compresie universal bazat pe dicționar,

Descriere compresie: Se construiește dicționarul inițial (toate șirurile de lungime 1), Se caută cel mai lung șir W din dicționar care se potrivește cu șirul de la intrare

Se elimină W din șirul de intrare, Se adaugă W urmat de următorul caracter în dicționar, Se continuă cu pasul 2

Decompresie: se parcurge șirul codificat și se reconstruiește dinamic dicționarul

Utilizat pentru fișiere de tip GIF

Compresia Huffman: Algoritm universal de compresie, Codificare optimă de lungime variabilă pentru fiecare simbol în funcție de frecvența de apariție, Datele salvate: dicționarul, datele originale recodificate; Decodificare: translație simbol cu simbol pe baza dicționarului salvat.

Compresia JPEG: Compresie specializată cu pierdere de informație, Rezultate foarte bune pentru fotografii (variații fine de luminozitate și culoare)

Tipuri de compresie: **secvențial** – codaj bazat pe transformarea cosinus discretă cu blocurile procesate în ordinea apariției; **progresiv** – codaj bazat pe transformarea cosinus discretă cu blocurile procesate prin mai multe treceri asupra imaginii; **progresiv fără pierdere** – folosește doar algoritmi de compresie fără pierdere de informație, **progresiv ierarhic** – codifică imaginea la rezoluții din ce în ce mai mari

Etape: 1. Translatarea modului de culoare din RGB în Y'CBCR, 2. Reducerea rezoluției pentru componentele CB și CR, 3. Imaginea se descompune în blocuri de dimensiune 8x8 pixeli, 4. Se aplică transformata cosinus discretă pe fiecare bloc în parte, 5. Aplicarea matricei de cuantizare (pierdere de informație), 6. Blocurile rezultate în urma cuantizării sunt comprimate folosind RLE și Huffman;
Decodificare: se aplică pașii în ordine inversă

GRAFICA VECTORIALA

Bazată pe descrierea matematică a obiectelor componente ale imaginii

Avantaje: Menține semantica – editare la nivel de obiect graphic, Dimensiune redusă, Independente de scara de vizualizare

Dezavantaj: Nu poate reprezenta fidel orice fel de informație

Formate de stocare: SVG (Scalable Vector Graphics) - Format generic bazat pe XML pentru reprezentări vectoriale 2D, Suportă animație și interactivitate

DXF (Drawing Exchange Format)-formatul vectorial lansat de firma Autodesk pentru produsul software AutoCAD

EPS (Encapsulated Post Script) - formatul firmei Adobe pentru imagini vectoriale, se bazează pe un limbaj de descriere numit Post Script

SHP (Shapefile) - formatul firmei ESRI pentru descrierea datelor spațiale de tip: punct, polilinie și polygon, utilizat la reprezentarea elementelor geografice în sisteme de tip GIS

ANIMATIE

Modificarea rapidă a imaginii vizualizate prin modificarea poziției, formei sau dimensiunii unui obiect din imagine; Stocarea numerică a animației presupune reținerea elementelor independente ce compun mișcarea în raport cu factorul timp.Crearea iluziei de mișcare se realizează prin afișarea rapidă de imagini statice ușor modificate

Tehnici principale:Tehnica filmului, Cadre cheie, Schimbarea culorii

SUNETUL

Reprezentare:Axa X: timp, Axa Y: presiune (0 – presiunea aerului în repaus), Amplitudine: măsoară dimensiunea vibrației / volumul sunetului, Frecvență: măsoară viteza vibrației / tonul sunetului

Numerizarea sunetului:Presupune stocarea și prelucrarea sunetului în format digital; Etape:Convertirea sunetului în semnal electric, Eșantionarea și quanticarea semnalului, Stocarea informației numerice pe un suport de memorie externă conform unui format

Avantaje: Stocare mai ușoară, Permite analiza și procesarea numerică a sunetului, nu se degradează în timp sau la copieri repetate

Eșantionare: Prin eșantionare se înțelege procesul de segmentare, cu o perioadă fixă, a semnalului audio analog.Frecvența de eșantionare – rezoluția orizontală. Se determină pe baza teoremei lui Nyquist (minim dublul frecvenței maxime a sunetului)Rate de eșantionare uzuale: 8 kHz – semnal telefonic, 11 kHz – radio AM, 22 kHz – radio FM, 44 kHz – audio CD

Cuantificare: Pp asocierea unei valori numerice corespunzătoare amplitudinii semnalului pentru fiecare interval de timp.Calitatea este influențată de numărul de biți alocați pentru fiecare eșantion (uzual 8 sau 16 biți pentru stocare și 16 – 32 pentru procesare). Redarea sunetului digital:Se reconstruiește sinusoida originală prin interpolarea valorilor numerice stocate, Prin intermediul unui convertor digital

Formate audio:**WAVE** – formatul standard de fișier audio pentru Microsoft și IBM; conține sunet în reprezentare PCM necomprimat;**AIFF** (Audio Interchange File Format) – formatul standard pentru audio digital utilizat pe platformele Apple (variante: necomprimat / comprimat);**MPEG** (Moving Picture Experts

Group) Audio - format standard pentru sunetul digital comprimat; parte a standardului MPEG de codificare a semnalului audio-video; cea mai cunoscută variantă a lui este MP3.

Compresia:Cel mai utilizat algoritm de compresie: MPEG-1 sau 2 Audio Layer III (MP3), Folosește codificare perceptuală - Elimină din rezultat sunetele care nu pot fi percepute de către urechea umană, Sunetele imperceptibile sunt eliminate pe baza unui mode psihoacoustic care exploatează fenomenele de:Mascare a frecvențelor, Mascare temporală

Mascarea frecvențelor: Sunt eliminate sunetele cu frecvență mai mare de 16-18 KHz, Sunt eliminate sunetele de intensitate scazută, care apar concomitent cu sunete de intensitate înaltă, dacă sunt în benzi de frecvență alăturate (cele cu intensitate scazută sunt măcate de cele cu intensitate înaltă)

Mascarea temporală: Se elimină sunetele de intensitate mică care urmează după sunete de intensitate puternică în cadrul unui interval de timp, Sunetele de intensitate mică nu pot fi percepute după sunete de intensitate puternica datorită inerției timpanului

Compresia MP3- etape: 1. Utilizarea de filtre pentru separarea sunetului în 32 sub-benzi de frecvență, 2. Aplicarea modelului psiho-acustic pentru eliminarea sunetelor imperceptibile, 3. Determinarea numărului de biți pentru coeficienți, 4. Prelucrarea valorilor obținute și compunerea fluxului final de biți

VIDEO

Video digital – cuprinde totalitatea tehniciilor de captură, procesare și stocare a imaginilor în mișcare (precum și a sunetului asociat) prin intermediul unui dispozitiv de calcul.

Avantaje video digital:Poate fi procesat prin intermediul calculatorului, Păstrare în timp și rezistență la copieri repetitive, Poate fi transmis la distanță

Caracteristici:Rezoluția, Spațiul de culoare și numărul de biți per pixel, Numărul de cadre pe secundă, Modul de afișare (întrețesut sau progresiv), Calitatea compresiei

Formate: Container – specifică structura de stocare a componentelor video (imagine + audio) și a datelor asociate (metadate, subtitrări, ...)

Advanced Systems Format – ASF: container dezvoltat de Microsoft care poate conține fluxuri codate cu orice codec (Extensii: .ASF, .WMA, .WMV), **Audio Video Interleave – AVI:** container mai vechi dezvoltat de Microsoft pe baza Resource **Interchange File Format – RIFF** (stochează datele în secțiuni identificate prin markere FourCC), **MP4 – MPEG-4 Part 14:** dezvoltat de către Motion Pictures Expert Group și utilizat inițial de către QuickTime (video H.264, audio AAC),

AVCHD – format utilizat în special de către camerele video (video H.264 AVC și sunet AC3 sau PCM), **Matroska / OGG:** formate deschise; pot conține mai multe fluxuri audio / video

Codec – specifică modalitatea de compresie / decompresie pentru un flux video / audio în cadrul unui container

H.264 / MPEG-4 AVC – cel mai popular (utilizat pentru Web, BluRay, camere video), **H.262 / MPEG-2** – formatul standard pentru DVD, **Windows Media Video** – format dezvoltat de către Microsoft, **MJPEG (Motion JPEG)** – format mai vechi bazat pe compresia JPEG

Compresia: Se bazează pe reducerea redundanței din cadrul fluxului video

Redundanță spațială (intra-cadru) - tipul de redundanță identificat și eliminat de algoritmii de compresie a imaginilor, Redundanță temporală (inter-cadru) - redundanță identificată între două cadre consecutive (de exemplu, prin compararea a două cadre se observă că majoritatea pixelilor își păstrează valoarea)

Algoritm de compresie video

Hibrid - Transformata Cosinus Discretă – similar JPEG pentru reducerea redundanței spațiale, Codaj Huffman – pentru comprimarea coeficienților TCD, Codificarea mișcării – pentru reducerea redundanței temporale, Codaj **RLEAsimetric** - Timpul de codare este mult mai mare decât cel de decodare

Etape compresie: Împărțirea imaginii în blocuri: 16x16 luminanță, 8x8 crominanță (culoare); Compresie pe baza DCT pentru reducere spațială, Aplicarea tehniciilor de compensare a mișcării pentru reducere temporală, Faza finală de codare pe două dimensiuni folosind Run Length Encoding

Tipuri de cadre: <I> Intra-picture/frame/image - Cadrele cheie, Necesare pentru căutare și poziționare, Compresie moderată

<P> Predicted pictures - Codate cu referință la un cadru anterior, Folosite ca referință pentru cadre ulterioare

 Bi-directional prediction (interpolated pictures) - Necesită cadre anterioare și viitoare pentru refacere, Compresie mare

NOTITE SEMINAR MULTIMEDIA

SEMINAR 2:

1. Adaugare fisier css extern:

```
<link rel="stylesheet" type="text/css" href="agenda.css">
```

2. Adaugare fisier javascript:

```
<script type="text/javascript"> ->daca scriut direct aici  
<script src="cod.js" type="text/javascript"></script> ->daca iau din alt fisier
```

3. Tabel:

```
<table>  
  <caption>Person List </caption>  
  <thead> //header  
    <tr>  
      <th>Last Name</th> //celula din header  
      <th>First Name</th>  
      <th>Phone</th>  
    </tr>  
  </thead>  
  <tbody> //body  
    <tr>  
      <td>Popescu</td> //celula din body  
      <td>Ion</td>  
      <td>0732555</td>  
    </tr>  
  </tbody>  
  <tfoot> //footer  
    <tr>  
      <td colspan="3">Number of persons:1 </td> //ocupa 3 coloane  
    </tr>  
  </tfoot>  
</table>
```

4. Formular:

```
<form action="#">
<label for="lastName">Last name:</label>
<input type="text" id="lastName" name="lastName" placeholder="last name">
<label for="phone">Phone:</label>
<input type="tel" id="phone" name="phone" placeholder="phone">

<input type="button" value="Add person">
</form>
```

5. CSS:

Selector de tip element:

Body, h1, table, p, etc...

Color->culoarea textului

Background-color->culoarea fundalului

Font-family:tahoma->fontul

Font-size:11pt; ->marime font

Font-weight:bold ->stil font

Text-align:left ->orientare in pagina

Border: 1px solid black

Width: 60%

Border-radius:5ps; ->colturi rotunjite

input[type="button"] ->selector

SEMINAR 3:

1. Validare de null:

```
if (lastName.value === "" || lastName.value == null)
```

2. Validare doar cu cifre:

```
if (!/[0-9]+$/ .test(phone.value))
```

3. Adaugare linii in tabel folosind javascript:

```
//adaugam randuri noi in tabel cu informatia citita
```

```
let tr = document.createElement("tr"); //I. fac randul dorit
```

```

let tdLastName = document.createElement("td"); //1. se face coloana
tdLastName.innerText = lastName; //2. Se pune text in coloana noua
tr.appendChild(tdLastName); //3. Se lipeste coloana la randul creeat

let tdDelete = document.createElement("td")
let Button = document.createElement("input")
Button.type="button"
Button.value="Delete"

Button.addEventListener("click",deletePerson2);
tdDelete.appendChild(Button)

tr.appendChild(tdDelete);

let tBody = document.querySelector("tbody"); // II. selectam body ptr a lipii la el nou
rand creeat
tBody.appendChild(tr); // III. Punem randul in body

```

4. Sterge un rand din tabel:

```

function deletePerson2(){
    let input=this;
    let tdToDelete=input.parentNode;
    let trToDelete=tdToDelete.parentNode;
    let tBody = document.querySelector("tbody");
    tBody.removeChild(trToDelete);
    document.getElementById("noOfPersons").innerText
    =document.getElementById("myTable").rows.length-2;
}

```

SEMINAR 4: CANVAS-BAR CHART

1. Canvas-lucru initial:

```

//inainte de a lucra cu un canvas trebuie sa obtinem un context
let context = this.canvas.getContext("2d")

context.lineWidth = 2; //ptr grosimea liniilor

```

```

context.textAlign="center" //ptr aliniera in cadrul canvasului

context.strokeStyle ="#dedede" //ptr a schimba culoarea contur
context.fillStyle = "#dedede" //ptr a schimba culoarea de umplere
context.fillRect(0, 0, this.canvas.width, this.canvas.height) //ptr a umple un dreptunghi, x,y-
coordonatele de la care desenam, w,h- latime si inaltime

```

2. Desenare grafic cu bare dupa un vector de valori:

```

draw(values){
let context = this.canvas.getContext("2d");

let barWidth = this.canvas.width / values.length;

let maxValue = Math.max(...values)
let f = this.canvas.height / maxValue;

for (let i = 0; i < values.length; i++) {

let barHeight = values[i] * f * 0.9;
let barX = i * barWidth;
let barY = this.canvas.height - barHeight;

let barWidthVisible=barWidth * 0.9

context.fillStyle = "#FF0000"
context.fillRect(barX, barY, barWidthVisible, barHeight)
context.strokeRect(barX, barY, barWidthVisible, barHeight)

context.fillStyle="#000000"
context.fillText(values[i],barX+barWidthVisible/2,barY-10)
}

```

3. Download imagine desenata pe canvas la click pe canvas:

HTML:

```

<a href="#" download="barChart">

<canvas id="canvas" style="width: 800px; height: 600px;">
Your browser does not support the canvas element!

```

```
</canvas>
</a>
```

JS:

```
canvas.addEventListener("click", function(){
    let a = this.parentNode;// this este elementul care genereaza click deci canvas si canvas este pus intr-un link a asadar this.parentNode este linkul a
    let dataUrl=this.toDataURL("image/png")
    console.log(dataUrl);
    a.href=dataUrl;
})
```

4. Eveniment de click pe un element:

```
btnDrawChart.addEventListener("click", function () {...});
```

5. Extragere valoare dintr-un camp:

```
let tbValues = document.getElementById('tbValue');
let values = tbValues.value;
```

6. Transformare String de forma „[12,2,24]” intr-un vector cu elementele 12,2,24:

```
let valuesArray = eval('[' + values + ']');
barChart.draw(valuesArray)
```

SEMINAR 5 – CANVAS – HISTOGRAMA DUPA POZA

1. Desenare histograma dupa un set de valori date in vector:

```
draw(values){
    let context = this.canvas.getContext("2d");

    let maxValue = Math.max(...values)
    let f = this.canvas.height / maxValue;

    let barWidth = this.canvas.width / values.length;

    context.save();
    context.rotate(Math.PI);//rotatie la 180 de grade
```

```

context.translate(0,-this.canvas.height)//translate pe verticala, in sus, cu valoarea inaltimei

context.scale(-1,f); // -1 adica face flipp pe verticala, oglinda

for(let i=0;i<values.length;i++)
{
    context.fillRect(i*barWidth,0,barWidth*0.9,values[i]);
}
context.clearRect(0,0,this.canvas.width,this.canvas.height)
context.restore();
}

```

2. `scale(-1, 1)` to flip the context horizontally and `scale(1, -1)` to flip it vertically. The `translate()` method adds a translation transformation to the current matrix by moving the canvas and its origin `x` units horizontally and `y` units vertically on the grid.

3. Drag and drop imagine pe canvas:

```

document.addEventListener("dragover", function (e) {
    e.preventDefault();
})
document.addEventListener("drop", function (e) {

    e.preventDefault();

    let data = e.dataTransfer;
    let files = data.files;

    let fileReader = new FileReader();
fileReader.addEventListener("load", function (e) {

    let dataUrl = e.target.result;// e.target <=> fileReader //refera obiectul de tip
filereader
    let img = document.createElement("img");
img.addEventListener("load", function (e) {

        canvasImage.width = img.naturalWidth;
        canvasImage.height = img.naturalHeight;

        let context = canvasImage.getContext("2d");
        context.drawImage(img, 0, 0);
    })
})
})

```

```

        })
        img.src = dataUrl;

    });
    fileReader.readAsDataURL(files[0]);
}
})

```

4. Extragere pixeli din imaginea de pe canvas – ptr histograma

```

let imageData = context.getImageData(0, 0, canvasImage.width, canvasImage.height)
let data = imageData.data;

for (let i = 0; i < data.length; i += 4) { // fiecare pixel are 4 pozitii ptr canalele rgb
    let r = data[i]; // rosu
    let g = data[i + 1]; // verde
    let b = data[i + 2]; // albastru
    let a = data[i + 3]; // transparenta, daca exista

    let average = Math.round((r + g + b) / 3);

    v[average]++;
}

}

```

SEMINAR 6 – CANVAS – FILTRE PE IMAGINE

1. Tab de „Adauga fisier” care deschide file explorer ptr incarcare imagine:

HTML:

```
<input id="fileBrowser" type="file" accept="image/*">
```

JS:

```
document.getElementById("fileBrowser").addEventListener("change", function (ev) {
    const files = ev.target.files;

    const reader = new FileReader();
    reader.addEventListener("load", function (ev) {
```

```

const dataURL = ev.target.result;

const img = document.createElement("img");
img.addEventListener("load", function (ev) {

    app.visibleCanvas.width = img.naturalWidth;
    app.visibleCanvas.height = img.naturalHeight;
const oContext = app.offscreenCanvas.getContext("2d");
    oContext.drawImage(ev.target, 0, 0);
})
img.src = dataURL;
})
reader.readAsDataURL(files[0]);
});

```

2. Filtre:

- *Greyscale*:

```

for (let i = 0; i < data.length; i += 4) {
    const r = data[i];
    const g = data[i + 1];
    const b = data[i + 2];
    // const a=data[i+3];

    const avg = Math.round((r + g + b) / 3);
    data[i] = data[i + 1] = data[i + 2] = avg;
}

```

- *Threshold*

```

for (let i = 0; i < data.length; i += 4) {
const r = data[i];
    const g = data[i + 1];
    const b = data[i + 2];
    // const a=data[i+3];

    const v = (0.2126 * r + 0.7152 * g + 0.0722 * b >= threshold) ? 255 : 0;
    data[i] = data[i + 1] = data[i + 2] = v;
}

```

- *Sepia*

```

for (let i = 0; i < data.length; i += 4) {
    const r = data[i];
    const g = data[i + 1];
    const b = data[i + 2];
    data[i] = (r * .393) + (g * .769) + (b * .189);
    data[i + 1] = (r * .349) + (g * .686) + (b * .168)
    data[i + 2] = (r * .272) + (g * .534) + (b * .131)
}
• Invert:
for (let i = 0; i < data.length; i += 4) {
    const r = data[i];
    const g = data[i + 1];
    const b = data[i + 2];
    data[i] = 255 - r;
    data[i + 1] = 255 - g;
    data[i + 2] = 255 - b;
}
• Pixelate:
const blocksize = 10;
const oContext = app.offscreenCanvas.getContext("2d");
const vContext = app.visibleCanvas.getContext("2d");

for (var x = 1; x < app.offscreenCanvas.width; x += blocksize) {
    for (var y = 1; y < app.offscreenCanvas.height; y += blocksize) {
        var pixel = oContext.getImageData(x, y, 1, 1);
        vContext.fillStyle = "rgb(" + pixel.data[0] + "," + pixel.data[1] + "," + pixel.data[2] +
    ")");
        vContext.fillRect(x, y, x + blocksize - 1, y + blocksize - 1);
    }
}
• 2Channels:
for (let i = 0; i < data.length; i += 4) {

    const g = data[i + 1];
    data[i + 2] = g;
}
• Red:
for (let i = 0; i < data.length; i += 4) {

    data[i + 1] = 0;
    data[i + 2] = 0;
}

```

3. Cum sa selectezi mai multe butoane care au aceeasi proprietate:

HTML:

```
<button data-effect="normal">Normal</button>
<button data-effect="grayscale">Grayscale</button>
```

JS:

```
document.querySelectorAll("button[data-effect]");
```

SEMINAR 7 – CANVAS – FILTRE PE IMAGINE 2

1. Sa se faca butoanele cu scris rosu cand sunt apasate:

CSS:

```
.active{
    color:red;
}
```

JS:

```
const previousButton = document.querySelector(".active");
if (previousButton != null) {
    previousButton.classList.remove("active");
}
button.classList.add("active");
```

2. Pozitionare div in mijlocul ecranului, sus, folosind css:

```
<div style="
background-color:rgba(200,200,200,0.8);
padding: 10px; position: fixed; top:10px; left:50%; transform: translate(-50%,0);">
```

3. Facem un buton invizibil cu css:

HTML: <button id="btnDownload" style="display: none;">Download</button>
SAU JS: btn.style.display = "none";

4. Aplicare filtru doar pe jumatarea stanga a pozei-preluare pixeli in sectiuni, nu in totalitate

```
const oContext = offscreenCanvas.getContext("2d");
```

```

const imageData = oContext.getImageData(0, 0, offscreenCanvas.width,
offscreenCanvas.height);
const data = imageData.data;

for (let y = 0; y < offscreenCanvas.height; y++) {
    for (let x = 0; x < offscreenCanvas.width/2; x++) {
        const i = (y * (offscreenCanvas.width) * 4) + x * 4;

        const r = data[i];
        const g = data[i+1];
        const b = data[i+2];
        const transparency = data[i+3];
        const average = Math.round((r + g + b) / 3);

        data[i] = average;
        data[i+1] = average;
        data[i+2] = average;
    }
}

```

5. Filtru Darker si Brighter cu o valoare data:

- *Darker:*

```

for (let i = 0; i < data.length; i += 4) {
    const r = data[i];
    const g = data[i + 1];
    const b = data[i + 2];

```

```

    data[i] = r - v;
    data[i + 1] = g - v;
    data[i + 2] = b - v;
}

```

- *Brighter:*

```

    for (let i = 0; i < data.length; i += 4) {
        const r = data[i];
        const g = data[i + 1];
        const b = data[i + 2];

```

```

        data[i] = r + v;
        data[i + 1] = g + v;
        data[i + 2] = b + v;
    }
}

```

6. Buton pentru download imagine de pe canvas:

```

document.querySelector("#btnDownload").addEventListener("click", (ev) => {
  const a = document.createElement("a");
  a.href = app.visibleCanvas.toDataURL();
  a.download = "output.png";
  a.click();
})

```

7. Preluare culoare din punctul in care ne aflam cu mouse-ul:

```

app.visibleCanvas.addEventListener("mousemove", (ev) => {
  const x = ev.offsetX * app.visibleCanvas.width / app.visibleCanvas.clientWidth;
  const y = ev.offsetY * app.visibleCanvas.height / app.visibleCanvas.clientHeight;

  const vContext = app.visibleCanvas.getContext("2d");
  const imageData = vContext.getImageData(x, y, 1, 1);
  const data = imageData.data;
  const r = data[0];
  const g = data[1];
  const b = data[2];

  const span = document.getElementById("color");
  const color = `rgb(${r}, ${g}, ${b})`;//sau const color='rgb(${r},${g},${b})';

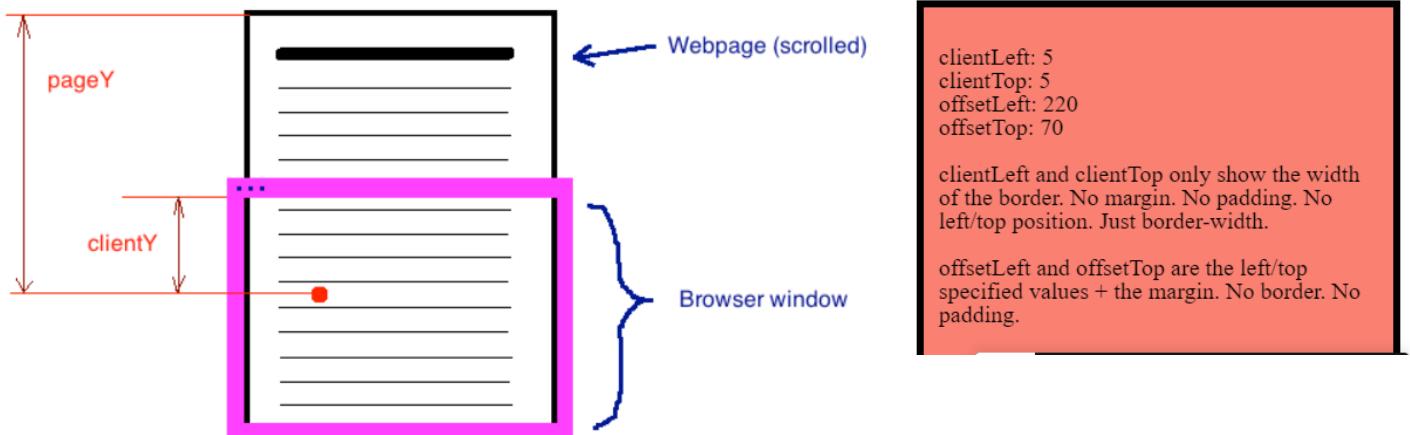
  span.innerText = color;
  span.style.backgroundColor = color;
})

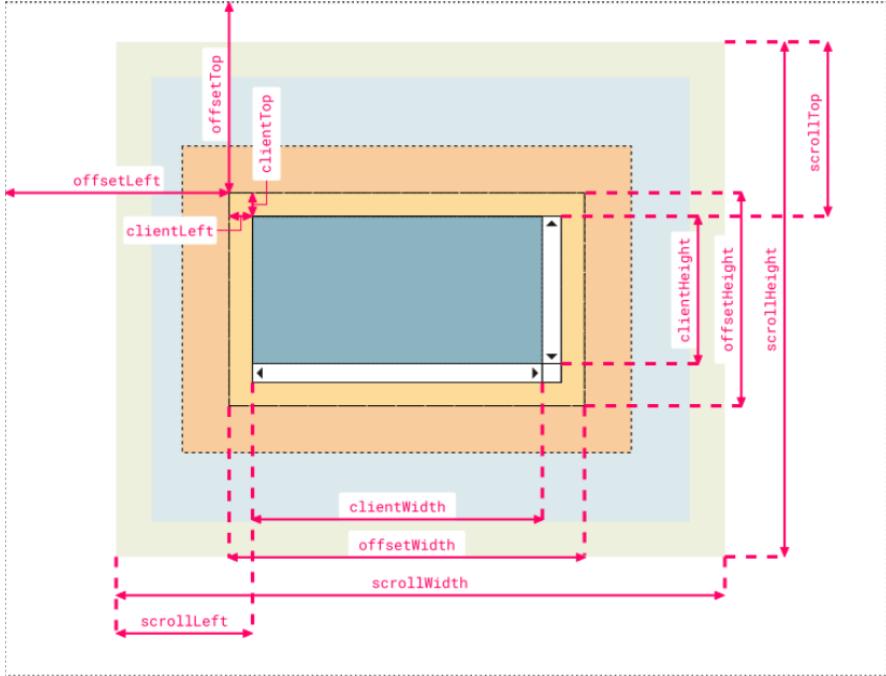
```

8. Pozitie in pagina:

pageX: pozitia in pagina relativa la inceputul intregii pagini

clientX: pozitia in pagina relativa la ce vede utilizatorul(viewport), chiar daca pagina este scrollata si incepe de mai sus.





offsetTop/Left: distanta fata de sus sau stanga relativa la punctul 0.0 din care incepe toata pagina, indiferent daca avem scroll

getBoundingClientRect().top/left: distanta fata de sus sau stanga relativa la fereastra pe care o vede utilizatorul(viewport), chiar daca pagina este scrollata si incepe de mai sus

Daca vrem sa luam pozitia mouse-ului in canvas la evenul de mousedown putem scrie:

```
→pozitie.x = event.pageX -canvas.offsetLeft;
pozitie.y = event.pageY -canvas.offsetTop;
→pozitie.x = event.clientX - canvas.getBoundingClientRect().left;
pozitie.y = event.clientY - canvas.getBoundingClientRect().top;
```

SEMINAR 8 – SVG – BARCHART

1. Schimbam stilul cand suntem deasupra unui element; util pentru butoane:

CSS:

```
.button:hover{  
    fill:yellow;  
}
```

2. Pentru a lucra cu svg avem nevoie de un namespace:

```
this.svgns = "http://www.w3.org/2000/svg";  
this.svg = document.createElementNS(this.svgns, "svg");
```

...PROCESARE SVG...

```
this.element.appendChild(this.svg); //element este un div in html
```

3. Modificari CSS in JS:

```
→this.svg.setAttribute("style", "border: 1px solid black");  
SAU  
→ this.svg.style.borderColor = "black";  
    this.svg.style.borderWidth = "1px";  
    this.svg.style.borderStyle = "solid";
```

4. Desenare rect in SVG:

```
const rect = document.createElementNS(this.svgns, "rect");  
rect.setAttribute("x", 0);  
rect.setAttribute("y", 0);  
rect.setAttribute("width", this.width);  
rect.setAttribute("height", this.height);  
rect.style.fill = "whitesmoke";  
  
this.svg.appendChild(rect);
```

5. Desenare Barchart pe un SVG dupa un vector cu valori:

```

const data=[

    ['Label 1',1],


    ['Label 2',2],


    ['Label 3',3]

]

....  

const barWidth = this.width / this.data.length;
const maxValue = Math.max(...this.data.map((x) => x[1])); //map pt a obtine doar valorile numerice
const f = this.height / maxValue;

for (let i = 0; i < this.data.length; i++) {
    const bar = document.createElementNS(this.svgns, "rect"); //desenare rect
    const text=document.createElementNS(this.svgns,"text"); //desenare text

    const label = this.data[i][0];
    const value = this.data[i][1];

    const barHeight = value * f * 0.9;

    const barX = i * barWidth;
    const barY = this.height - barHeight; //ca sa nu le afiseze de sus

    bar.classList.add("bar"); //ptr prelucrarea in css cu selectorul .bar

    bar.setAttribute("x", barX + barWidth / 4);
    bar.setAttribute("y", barY);
    bar.setAttribute("width", barWidth / 2);
    bar.setAttribute("height", barHeight);

    bar.addEventListener("click", ()=>{
        alert(value);
    });

    text.setAttribute("x", barX+barWidth/4);
    text.setAttribute("y", barY-10);
    const labelValue=document.createTextNode(label);
    text.appendChild(labelValue);
}

```

```

// bar.style.fill = "red"; -> aici daca vrem sa facem aia cu hover pe galben are
specificitatea prea mare si raman rosii
//daca am pune cu bar.setAttribute("fill","red"); //ar fi functionat ptr ca nu mai are
specificitate mai mare decat hover

bar.style.stroke = "black";
//bar.setAttribute("stroke-width", "10px");
bar.style.strokeWidth = "2px";
//sau bar.style["stroke-width"]="2px"

this.svg.appendChild(bar);
this.svg.appendChild(text);

}

```

SEMINAR 9 – AUDIO.1

1. Adaugare librarii externe-Bootstrap si fonturi Cloudflare

```

<!-- Required meta tags -->
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

<!-- Bootstrap CSS -->
<link rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css"
      integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2"
      crossorigin="anonymous">

<!--Fonturi-->
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/all.min.css">

```

2. Element de tip audio in HTML:

```

<audio id="audio" src="media/Beethoven-MoonlightSonata.mp3">
    Your browser does not support the "audio" element
</audio>
SAU
<audio
    controls //ne face automat butoane de control
    src="/media/cc0-audio/t-rex-roar.mp3">
    Your browser does not support the audio element.
</audio>

```

3. Buton de play/pause pentru muzica din elementul audio:

```

app.btnExit.addEventListener("click", () => {
    if (app.audio.paused)
        app.audio.play();
    else
        app.audio.pause();
});

```

4. Cum sa afisam durata totala a melodiei si pozitia curenta in melodie:

→functie de formatare float in min si sec ptr afisare corecta:

```

function formatTime(seconds) {

    let minutes = Math.floor(seconds / 60);
    let secondsRemaining = Math.floor(seconds - minutes * 60);

    let result = "" + (minutes < 10 ? "0" : "") + minutes + ":" + (secondsRemaining < 10 ? "0" : "") +
    secondsRemaining;

    return result;
}

```

→eventimentele care trebuie tratate:

```

app.audio.addEventListener("durationchange", () => {//cand se schimba durata totala
    app.spDuration.innerHTML = formatTime(app.audio.duration); // audio.duration ne da
    lungimea totala a melodiei
});

```

```

app.audio.addEventListener("timeupdate", () => {//cand se schimba valoarea din currentTime
    app.spCurrentTime.innerHTML = formatTime(app.audio.currentTime); //audio.currentTime
    ne da pozitia curenta in melodie
});

```

SEMINAR 10 – AUDIO.2->playlist

1. Playlist-ul poate fi o simpla lista in care pastram sursa ptr fiecare melodie cu un atribut:

```

<ul id="playlist" class="list-group">
    <li class="list-group-item" data-url="media/Bolero.mp3">
        Ravel - Bolero
    </li>
    <li class="list-group-item" data-url="media/Beethoven-MoonlightSonata.mp3">
        Beethoven - Moonlight Sonata
    </li>
    <li class="list-group-item" data-url="media/CanoninD.mp3">
        Pachelbel - Canon in D
    </li>
</ul>

```

2. Cum selectam o melodie din playlist ptr a da play la click pe ea:

```

→ const elements = document.querySelectorAll("li[data-url]"); //selectam toate randurile din
lista, adica toate melodiile si ptr fiecare am salvat in atributul data-url link-ul sursa

for (let i = 0; i < elements.length; i++) {
    const element = elements[i];
    const url = element.dataset.url;

    element.addEventListener("click", function () {
        app.play(url);
    })
}

```

```

→app.play = function (url) { //asta e functia de play din aplicatia scrisa de noi

    const previouslySelectedElement = document.querySelector("li.active");
    if (previouslySelectedElement !== null)
        previouslySelectedElement.classList.remove("active");

    const selector = 'li[data-url="' + url + '"]';
    const selectedElement = document.querySelector(selector);

    selectedElement.classList.add("active");

    app.audio.src = url;
    app.audio.play();
}

→app.audio.addEventListener("play", () => { //asta e functia de play a elementului audio
    const selectedElement = document.querySelector("li[data-
url='"+app.audio.getAttribute('src')+']."'");

    selectedElement.classList.add("active");
});

```

3. Buton de +10 secunde in melodie:

```

app.btnForward.addEventListener("click", () => {
    app.audio.currentTime += 10;
});

```

SEMINAR 11 – AUDIO.3 -> MICROFON SI SOUND BAR ANALYZAR-Frequency Bars

1. Dropdown de tip select in HTML:

HTML:

```

<label for="visualisation"> Visualizer settings</label>
<select id="visualisation" >
<option disabled> Choose</option>
<option value="sinewave">Sinewave</option>
<option value="bars">Frequency Bars</option>

```

```

</select>
JS:
display(visualisation) {
    this.audioContext.resume();

    if (visualisation === "bars") {
        this.displayBars();
    } else if (visualisation === "sinewave") {
        this.drawSineWave();
    }
}

```

2. The `MediaDevices.getUserMedia()` method prompts the user for permission to use a media input which produces a `MediaStream` with tracks containing the requested types of media. That stream can include, for example, a video or an audio.

```
stream = await navigator.mediaDevices.getUserMedia;
```

3. Cum sa preluam audio-ul din microfon:

```

→navigator.mediaDevices.getUserMedia({
    audio:true //daca exista un input de tip audio
}).then(function(stream){
    soundAnalyzer.setStreamSource(stream) //atunci stream-ul creat de input se trimit
    mai departe in aceasta metoda care construieste un nod sursa?
})
.catch(function(err){
    alert(err.message);
})

```

→Undeva in constructor :

```

this.audioContext = new AudioContext();
this.analyzerNode = this.audioContext.createAnalyser();
```

```

→setStreamSource(stream){
    const sourceNode = this.audioContext.createMediaStreamSource(stream);
    sourceNode.connect(this.analyserNode);
}
```

!!! daca vrem, putem transmite aici un element audio care are o melodie in el, nu neaparat un stream de la microfon DAR PUNEM MediaElementSource

```
setMediaElementSource(mediaElement) {  
    this.sourceNode = this.audioContext.createMediaElementSource(mediaElement);  
    this.sourceNode.connect(this.analyzerNode);  
}  
https://developer.mozilla.org/en-US/docs/Web/API/Web\_Audio\_API
```

AudioContext

The `AudioContext` interface represents an audio-processing graph built from audio modules linked together, each represented by an `AudioNode`. You need to create an `AudioContext` before you do anything else, as everything happens inside a context.

AnalyserNode

The `AnalyserNode` interface represents a node able to provide real-time frequency and time-domain analysis information, for the purposes of data analysis and visualization.->`audioContext.createAnalyser()`

MediaStreamAudioSourceNode

The `MediaStreamAudioSourceNode` interface represents an audio source consisting of a `MediaStream` (such as a webcam, microphone, or a stream being sent from a remote computer). It is an `AudioNode` that acts as an audio source.->`audioContext.createMediaStreamSource(stream)`

4. Construirea graficului cu bare ptr analiza audio: sound bar analyser-Frequency Bars

```
drawBars(){  
    this.context.fillStyle = "black";  
    this.context.fillRect(0, 0, this.canvas.width, this.canvas.height);  
  
    this.analyserNode.fftSize = 256;  
  
    const bufferLength = this.analyserNode.frequencyBinCount;  
    const buffer = new Uint8Array(bufferLength);  
    this.analyserNode.getByteFrequencyData(buffer);  
  
    const barWidth = this.canvas.width / buffer.length;  
    const f = this.canvas.height / 255;  
  
    this.context.fillStyle = "red";  
    for(let i=0; i< buffer.length; i++){  
        const barHeight = buffer[i] * f;  
        const barX = i*barWidth;  
        const barY = this.canvas.height - barHeight;
```

```

        this.context.fillRect(barX, barY, barWidth, barHeight);
    }

    //setInterval
    requestAnimationFrame(()=> this.drawBars());
}

```

The `window.requestAnimationFrame()` method tells the browser that you wish to perform an animation and requests that the browser calls a specified function to update an animation before the next repaint. The method takes a callback as an argument to be invoked before the repaint.

SEMINAR 12.1 – AUDIO.4 -> MICROFON SI SOUND WAVE ANALYZAR- Sinewave

1. Construirea graficului cu wave ptr analiza audio: sound waveanalyser-Sinewave

```

drawSineWave() {
    let bufferLength = this.analyserNode.fftSize;
    let dataArray = new Float32Array(bufferLength);
    this.analyserNode.getFloatTimeDomainData(dataArray);

    this.context.fillStyle = 'rgb(200, 200, 200)';
    this.context.fillRect(0, 0, this.canvas.width, this.canvas.height);
    this.context.lineWidth = 2;
    this.context.strokeStyle = 'rgb(0, 0, 0)';

    this.context.beginPath();

    let sliceWidth = this.canvas.width * 1.0 / bufferLength;
    let x = 0;
    for (let i = 0; i < bufferLength; i++) {

```

```

let v = dataArray[i] * 200.0;
let y = this.canvas.height / 2 + v;

if (i === 0) {
    this.context.moveTo(x, y);
} else {
    this.context.lineTo(x, y);
}
x += sliceWidth;
}
this.context.lineTo(this.canvas.width, this.canvas.height / 2);
this.context.stroke();

this.drawVisual = requestAnimationFrame(() => this.drawSineWave());
}

```

SEMINAR 12.2 – VIDEO

1. Element de tip video:

```

<video
id="video"
src="media/tears-of-steel-battle-clip-medium.mp4"
muted
controls>
    Not supported
</video>

```

!!!Daca punem video ca atare, va fi vizualizat din prima. Daca il punem intr-un canvas trebuie sa facem context.drawImage(video,0,0);

2. Captura de ecran din video:

```

btnCapture.addEventListener("click", ()=>{
    canvasCapture.width = video.videoWidth//!aparent nu merge cu
    video.clientWidth;
    canvasCapture.height = video.videoHeight;

    const context = canvasCapture.getContext("2d");
    context.drawImage(video,0,0);
});

```

3. Mut si unmute pe video:

```
btnMuted.addEventListener("click", ()=> {
    if(video.muted === true){
        video.muted = false;
        btnMuted.innerHTML = "Mute";
    }
    else{
        video.muted = true;
        btnMuted.innerHTML = "UnMute";
    }
});
```

4. Pause si unpause pe video:

```
btnPlayPause.addEventListener("click", ()=> {
    if(video.paused === true){
        video.play();
        btnPlayPause.innerHTML = "Pause";
    }
    else{
        video.pause();
        btnPlayPause.innerHTML = "Play";
    }
});
```

5. Canvas cat un video:

```
video.addEventListener("canplay", ()=>{
    canvasVideo.width = video.videoWidth; // aparent nu merge cu video.clientWidth;
    canvasVideo.height = video.videoHeight;
});
```

6. Animatie pe video:filtru si secunde

```
function draw(){
    contextVideo.drawImage(video, 0, 0);

    const imageData = contextVideo.getImageData(0, 0,
                                                canvasVideo.width, canvasVideo.height);
    const data = imageData.data; //merge doar pe microsoft edge, nu si pe chrome
```

```

/*for(let i=0; i<data.length; i+=4){
    const average = (data[i] + data[i+1] + data[i+2])/3;
    data[i] = data[i+1] = data[i+2] = average;
};*/

```

→ aplicare filtru

```

for (let y=0;y<canvasVideo.height;y++)
    for (let x=0;x<canvasVideo.width;x+=2)
    {
        let i= y*canvasVideo.width*4+x*4;

        const average=(data[i]+data[i+1]+data[i+2])/3;
        data[i]=data[i+1]=data[i+2];
    }

```

```
contextVideo.putImageData(imageData, 0, 0);
```

```

contextVideo.fillStyle = "white";
contextVideo.font = "100px sans-serif";
contextVideo.textAlign = "center"
contextVideo.fillText(
video.currentTime,
canvasVideo.width/2, 100)

```

```

requestId = requestAnimationFrame(draw); //ptr a desena la fiecare secunda, nu doar
odata
}

```

7. Daca vrem ca animatia cu filtru pe video si scris sa se opreasca atunci cand punem pauza:

```

let requestId=null;
video.addEventListener("play", ()=>{
requestId = requestAnimationFrame(draw);
})
video.addEventListener("pause", ()=>{
cancelAnimationFrame(requestId);
})

```

8. Activare si dezactivare controls

```

const btnControls = document.getElementById("btnControls");
btnControls.addEventListener("click", () => {

```

```
    if (video.controls === true)
        video.controls = false;

    else
        video.controls = true;
});
```

SEMINAR 13-JOC CU CARAMIZI

...

PROIECTUL CU CANVAS EDITOR FOTO

1. Stilul textului

```
editor.context.fillStyle = 'red';

editor.context.font = "20px Georgia"

editor.context.fillText('Trage o imagine pe canvas', editor.canvas.width / 2 - 90, editor.canvas.height /
2, 200);
```

2. Canvas proportionat, pastrarea proportiilor la redimensionare

```
ratio = editor.image.naturalHeight / editor.image.naturalWidth;
editor.canvas.width = 400;
editor.canvas.height = editor.canvas.width * ratio;
editor.context.drawImage(editor.image, 0, 0, editor.canvas.width,
editor.canvas.height);
```

3. Canvas cat imaginea:

```
canvasImage.width = img.naturalWidth;

canvasImage.height = img.naturalHeight;
```

4. Drag and drop:

```
editor.canvas.addEventListener("dragover", function (event) {
    event.preventDefault();
```

```

});;

editor.canvas.addEventListener("drop", function (event) {
    var fisierePrimitve = event.dataTransfer.files;
    var fisier = fisierePrimitve[0]; //primul fisier

    var cititor = new FileReader();
    cititor.onload = function (event) {

        editor.image = document.createElement("img");
        editor.image.addEventListener("load", function () {

            editor.context.clearRect(0, 0, editor.canvas.width, editor.canvas.height);

            ratio = editor.image.naturalHeight / editor.image.naturalWidth;
            editor.canvas.width = 400;
            editor.canvas.height = editor.canvas.width * ratio;
            editor.context.drawImage(editor.image, 0, 0, editor.canvas.width,
editor.canvas.height);

            salveazaStareCurenta();

        });

        editor.image.src = event.target.result; //cand se termina functia load se deseneaza poza
pe canvas
    };
    cititor.readAsDataURL(fisier); //cand se termina functia load ptr cititor

    event.preventDefault();
});

```

5. Incarcare poza din fileBrowser:

```

<input id="fileBrowser" type="file" accept="image/*">

//INCARCARE POZA DIN FILE BROWSER
document.getElementById("fileBrowser").addEventListener("change", function (event) {
    var fisierePrimitve = event.target.files;

    var cititor = new FileReader();
    cititor.addEventListener("load", function (event) {

```

```

editor.image = document.createElement("img");
editor.image.addEventListener("load", function (event) {

    editor.context.drawImage(editor.image, 0, 0, editor.canvas.width,
editor.canvas.height);

});

editor.image.src = event.target.result;
});
cititor.readAsDataURL(fisierePrimite[0]);

});

```

6. DrawImage a treia implementare cu source image:

```

//si desenez in canvas doar zona selectata
editor.context.drawImage(editor.image,
editor.selectiaMea.x, editor.selectiaMea.y,
editor.selectiaMea.width, editor.selectiaMea.height,
0, 0,
editor.canvas.width, editor.canvas.height);

```

7. Desenare elipsa sau cerc

```

let x = canvas.width / 2 - 25;
let y = canvas.height / 2 - 25;
context.beginPath();
context.arc(x, y, 50, 0, 2 * Math.PI);
context.stroke();

let x1 = canvas.width / 2 - 30;
let y1 = canvas.height / 2 - 50;
context.beginPath();
context.ellipse(x1, y1, 60, 100, 0, 0, 2 * Math.PI);
context.stroke();

```

8. Rotit canvas la 180 de grade

```

context.translate(canvas.width, canvas.height);
context.rotate(Math.PI);
context.rotate(Math.PI); //rotatie la 180 de grade

```

```
context.translate(0,-this.canvas.height) //translate pe verticala, in sus, cu valoarea inaltimei
```

9. Rotire scris in canvas la 90 de grade:

```
context.translate(canvas.width, -canvas.height/2);
context.rotate(Math.PI/2)
```

10. Afisare video pe canvas cu filtru gri

```
let requestId = null;
video.addEventListener("play", function () {
    requestId = requestAnimationFrame(draw);
})
video.addEventListener("pause", function () {
    cancelAnimationFrame(requestId);
})

function draw() {
    canvas.width = video.videoWidth;
    canvas.height = video.videoHeight;
    context.drawImage(video, 0, 0);

    let imageData = context.getImageData(0, 0, canvas.width, canvas.height);
    let data = imageData.data;

    for (let i = 0; i < data.length; i += 4) {
        const r = data[i];
        const g = data[i + 1];
        const b = data[i + 2];

        const avg = Math.round((r + g + b) / 3);
        data[i] = data[i + 1] = data[i + 2] = avg;
    }

    context.putImageData(imageData, 0, 0);
    requestId = requestAnimationFrame(draw);
}
```

11. Functie cu parametru:

```
buton.addEventListener("click", function () {
    /**
     * @param {HTMLImageElement} img
     */
```

```

        draw(img)

    })
function draw(img) {

    canvas.width = img.naturalWidth;
    canvas.height = img.naturalHeight;
    context.drawImage(img, 0, 0);

    let imageData = context.getImageData(0, 0, canvas.width, canvas.height);
    let data = imageData.data;

    for (let i = 0; i < data.length; i += 4) {
        let r = data[i];
        let g = data[i + 1];

        data[i] = g;
        data[i + 1] = r;
    }
    context.putImageData(imageData, 0, 0);

    context.strokeStyle="green";
    context.strokeRect(canvas.width/2-200, canvas.height/2-300,
    400,600);

}

```

12. Colțul din stanga jos al canvasului

```

for (let x = canvas.height - 200; x < canvas.height; x++) {
    for (let y = 0; y < 200; y++) {

        let i = x * canvas.width * 4 + y * 4;
        let r = data[i];
        let g = data[i + 1];
        let b = data[i + 2];

        data[i] = data[i + 1] = data[i + 2] = Math.round((r + g + b)
/ 3);
    }
}

```

13. Button pe play pause audio:

```
btn.addEventListener("click", function () {
```

```

        if (audio.paused) {
            audio.play();
            btn.innerHTML = "Pause";
        }
        else {
            audio.pause();
            btn.innerHTML = "Play";
        }
    });

```

14. Afisare video pe canvas:

```

video.addEventListener("play", function () {
    requestAnimationFrame(draw);
});

function draw() {

    canvas.width = video.videoWidth;
    canvas.height = video.videoHeight;

    context.drawImage(video, 0, 0);

    context.textAlign = "center";
    context.font="25px Georgia"
    context.fillText(video.src,canvas.width/2,canvas.height/2);

    requestAnimationFrame(draw);
}

```

15. Histograma cu rosu

```

btn.addEventListener("click", function () {
    let value = [];
    for (let i = 0; i < 256; i++)
        value[i] = 0;

    canvas.width = img.naturalWidth;
    canvas.height = img.naturalHeight;
    context.drawImage(img, 0, 0);

    let imageData = context.getImageData(0, 0, canvas.width, canvas.height);
    let data = imageData.data;

```

```
for (let i = 0; i < data.length; i += 4) {
    value[data[i]]++;
}

context.clearRect(0, 0, canvas.width, canvas.height);

context.font = '25px Arial'
context.fillStyle = 'green';
context.fillText('Alexandra', 50, 50)

let max = Math.max(...value);
let f = canvas.height / max;

let barWidth = canvas.width / value.length

context.rotate(Math.PI);
context.translate(0, -canvas.height);
context.scale(-1, f);

for (let i = 0; i < value.length; i++) {
    context.fillRect(i * barWidth, 0, barWidth * 0.9, value[i]);
}

})
```

Multimedia

conf. dr. Cristian Ioniță

Resurse / Contact

Resursele necesare se găsesc:

- <https://online.ase.ro> – pe pagina cursului / seminarului
- <http://ase.softmentor.ro> – secțiunea *Multimedia* → *Teme abordate*

Modalități de contact:

- Email: cristian.ionita@ase.ro / crionita@ie.ase.ro / cristian.ionita@softmentor.ro
- Mesaje pe platforma <https://online.ase.ro>

Obiective

Însușirea elementelor teoretice și practice care să permită dezvoltarea de aplicații multimedia

- Cunoașterea facilităților multimedia ale sistemelor de calcul actuale și a instrumentelor software pentru dezvoltarea de aplicații multimedia
- Cunoașterea noțiunilor teoretice necesare pentru procesarea culorilor, imaginilor raster și vectoriale, sunetului și secvențelor video
- Dezvoltarea de abilități de programare a aplicațiilor multimedia în context web

Tehnologii utilizate:

- Limbajele HTML, CSS și JavaScript
- API-urile puse la dispoziție de browser-ele web moderne

Evaluare

50% – evaluare finală

- Examen oral în ultimele două săptămâni

30% – evaluări pe parcurs

- Lucrare săptămâna 4 – manipulare DOM folosind JavaScript (20% din nota finală)
- Lucrare săptămâna 12 – elemente teoretice (10% din nota finală)

20% – proiect

- Individual, tema de proiect se alocă de către profesor, prezentare în săptămâna 12

Condiții de promovare

- Minim nota 5 la evaluarea finală
- Minim nota 5 la evaluarea pe parcurs
- Minim nota 5 la proiect

Teme Proiecte

1. Program de desenare

- Permite adăugarea interactivă și manipularea de figuri geometrice pe o imagine raster și exportul desenului în format raster sau vectorial

2. Editor de imagini

- Permite încărcarea de imagini și efectuarea interactivă de operații (crop, redimensionări, aplicare efecte, afișarea histogramei de culoare, adăugare de text, ...)

3. Player video

- Permite navigarea și editarea playlist-ului, afișarea de subtitrări, aplicarea de efecte în timp real, ...

4. Editor grafică vectorială (SVG)

- Permite editarea unui desen folosind grafică vectorială și exportul sub formă de SVG sau raster

5. Joc – similar Asteroids

6. Vizualizare date

- Vizualizare grafică interactivă pentru date preluate de pe Eurostat

Structura curs

1. Noțiuni generale

- Conceptul de multimedia și clase de aplicații
- Condiții hardware și software
- Multimedia în context web (HTML / CSS / JavaScript)

2. Imagine și animație

- Culoarea și spații de culoare
- Grafica raster
- Desenarea în grafica raster
- Procesarea imaginilor
- Stocarea și compresia imaginii
- Animație
- Grafica vectorială

3. Sunet

- Noțiuni de bază legate de sunet și procesul de numerizare
- Compresia audio
- Utilizarea sunetului în aplicații web (redare, generare și procesare)

4. Video

- Noțiuni generale și compresia
- Utilizarea secvențelor video și captura video în aplicații web

5. Integrarea elementelor multimedia în aplicații web, desktop sau mobile

I. Noțiuni generale

1. Conceptul de multimedia
2. Clase de aplicații multimedia
3. Condiții hardware / software
4. Multimedia în context WEB

1. Conceptul de multimedia

Multimedia din punct de vedere informatic este:

- o combinație de text, imagine, sunet, animație, video
- accesibilă utilizatorului prin intermediul sistemului de calcul

Elemente care au stat la baza dezvoltării conceptului de multimedia:

- conversia analog digital
- compresia datelor

Conversia analog - digital

Resursele utilizate trebuie convertite în format digital pentru:

- stocare
- procesare
- includere în aplicații

Resursele sunt convertite în format analog la redare

Compresia

- apariția algoritmilor specifici (cu pierdere de informație)
- exemple: JPEG, MPEG Audio / Video
- pierdere de informație controlată
 - exploatează limitările perceptiei umane
- proces asimetric

Tehnologii utilizate

- periferice specializate pentru captură imagine / sunet / video
- medii de stocare de mare capacitate
- tehnologii de comunicare la distanță cu bandă largă
- procesoare specializate pentru compresie / decompresie la nivel hardware

Aplicație multimedia

- componentele sunt accesibile prin intermediul unui sistem de calcul
- datele utilizate sunt în format digital (NU analogic)
- elementele sistemului sunt integrate într-o interfață unitară
- grad ridicat de interacțiune cu utilizatorul

Modalități de dezvoltare

Multimedia **authoring**

- includ componente preprogramate ce permit recunoașterea mai multor formate de resurse multimedia, instrumente pentru generarea animațiilor, pentru implementarea conceptelor de hypertext, hypermedia
- accentul cade pe scenariul de derulare a aplicației și pe sincronizarea resurselor
- bazat pe concepte precum axa timpului / carte / diagrame de flux
- Exemple: Flash, PowerPoint

Multimedia **programming**:

- accentul se pune pe procesarea directă a resurselor prin intermediul unui limbaj de programare și a unor biblioteci specializate

2. Clase de aplicații multimedia

Criterii multiple de clasificare

Cele mai importante criterii

- domeniu de utilizare
- grad de interacțiune
- localizarea componentelor

Clasificare în funcție de domeniu

- **Economie** – vizualizarea datelor
- **Educație** – aplicații de e-learning, enciclopedii
- **Publicitate** – aplicații de prezentare
- **Medicina** – procesarea și vizualizarea interactivă a datelor medicale
- **Industrial** – instrumente de proiectare asistată
- **Entertainment** – jocuri, realitate virtuală
- **Sisteme informaticice geografice GIS** – sisteme pentru vizualizarea datelor geo-spațiale
- **Comunicații** – aplicații de tip videoconferință

Alte criterii

Grad de interacțiune

- aplicații interactive
- prezentări statice sau liniare

Localizarea componentelor

- locale
- la distanță

3. Precondiții hardware / software

Echipamente hardware pentru achiziție / procesare:

- imagine
- sunet
- video

Componente software necesare pentru redarea conținutului multimedia

Hardware specializat

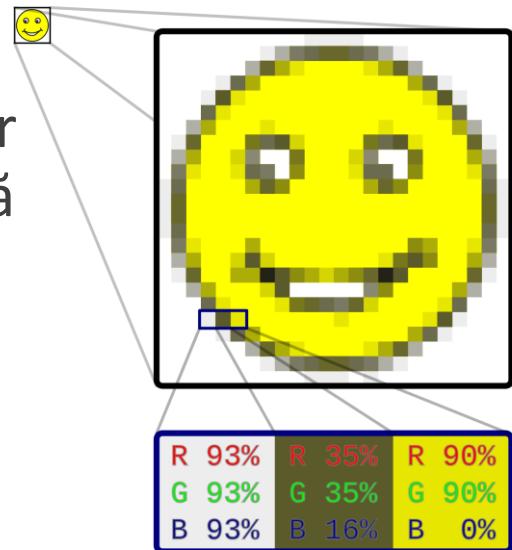
Dispozitive periferice pentru achiziția de **imagini fixe**:

Scanner

- transformă informația luminoasă în informație electrică, iar ulterior aceasta este convertită și salvată sub formă digitală
- tipuri: flatbed / rotativ

Aparat foto digital

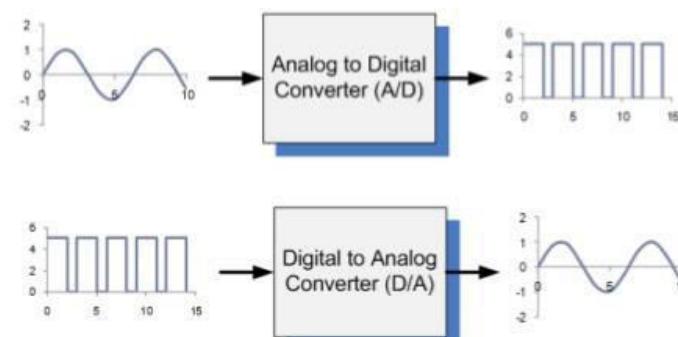
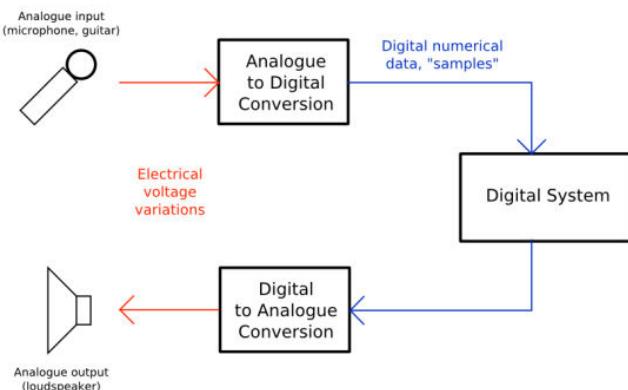
- folosește lentile asemănătoare aparatului foto clasic și un senzor digital pentru transformarea informației luminoase în informație electrică



Hardware specializat

Dispozitive periferice pentru achiziția de sunet:

- placă de sunet: convertor analog – digital și digital – analog



Hardware Specializat

Dispozitive periferice pentru achiziția de **imagini video**:

- placă de achiziție și numerizare video: preluare și numerizare fluxuri audio - video
- camera video digitală: captură directă în format digital (similar aparat foto digital)
- placă TV tuner: preia semnal TV, decodifică și numerizează semnalul

Implementare hardware pentru algoritmi de compresie și compresie în cadrul plăcilor video

Software specializat

Drivere

- Asigură controlul și comunicarea cu perifericele specializate
- Specifice pentru dispozitivul și sistemul de operare
- Permit sistemului de operare să ofere aplicațiilor o interfață uniformă

Extensii ale sistemului de operare

- Biblioteci specializate pentru controlul resurselor multimedia
- Instrumente de bază pentru redarea conținutului multimedia

Software specializat

Produse software specializate pe medii de comunicare

Achiziție și prelucrare de **imagini**

- Permit prelucrarea imaginilor în format raster sau vectorial
- Exemple: Adobe PhotoShop, GIMP, Corel Draw, Adobe Illustrator

Achiziție și prelucrare de **sunet**

- Exemple: Adobe Audition, Audacity, Sony Sound Forge

Achiziție și editare **video**

- Exemple: Adobe Premiere, Blender, Sony Vegas, DaVinci Resolve

Software specializat

Produse software pentru **creație și redare de aplicații multimedia**

Exemple

- Browser web
- Flash
- PowerPoint

4. Multimedia în context WEB

Se dezvoltă odată cu apariția standardului HTML5

Elemente multimedia suportate

- Text
- Imagini
- Animație
- Grafică raster – elementul *canvas*
- Grafică vectorială – SVG
- Sunet – elementul *audio* + *Web Audio API*
- Video – elementul *video*
- Grafică 3D – WebGL

Avantaje

Arhitectură client-server

Client – Web Browser

- Trimitere cererile către server
- Gestionează afişarea conţinutului multimedia şi interacţiunea cu utilizatorul

Server – Web Server

- Preia cererile de la Web Browser
- Trimitere conţinutul (static sau generat dinamic) către browser

Comunicare – protocolul Hypertext Transfer Protocol

- Protocol text
- Mesaje de tip *request* și *response*
- Mesajele sunt trimise prin intermediul protocolului TCP/IP

HTTP Request

Format: comanda (GET / POST / ...)/ parametri / eventual conținut

Cereri inițiate de către browser:

- Introducere adresă de către utilizator
- Navigare către pagină nouă
- Obținere resurse
- Solicitat din cod

Exemplu:

GET /index.html HTTP/1.1

Host: www.test.com

User-Agent:Mozilla/5.0

HTTP Response

Format: status (200 / 404 / 500 /...)/ parametri / conținut

Răspunsul este generat:

- Static (conținutul este preluat din fișier)
- Dinamic (conținutul este generat de către un program)

Exemplu:

HTTP/1.1 200 OK

Content-Type: text/html; charset=UTF-8

Content-Length: 131

<html>

...

</html>

Limbaje utilize

HTML - HyperText Markup Language

- Structura documentului
- Conținutul (de tip text)
- Referințe la alte resurse

CSS - Cascading Style Sheets

- Descrie modalitatea de prezentare a conținutului

JavaScript

- Manipularea dinamică a conținutului și a modalității de prezentare
- Gestionează interacțiunea cu utilizatorul

DOM – Document Object Model

Aplicația este reprezentată intern sub formă de arbore

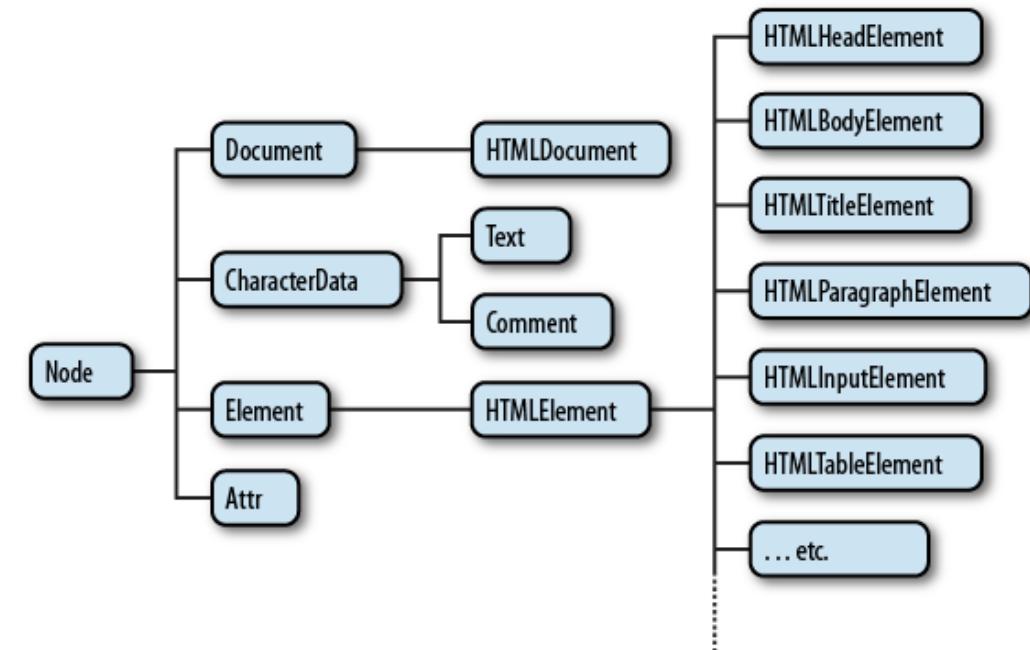
Tipuri de noduri

- Elemente
- Text

Nodurile - create din HTML sau JavaScript

Conțin:

- atribute
- parametri de stil
- funcții JavaScript



Limbajul HTML

Tag

- Instrucțiunea de creare a unui nod
- Format: **<test>conținut</test>**
- Conținutul poate fi:
 - Text
 - Alte tag-uri

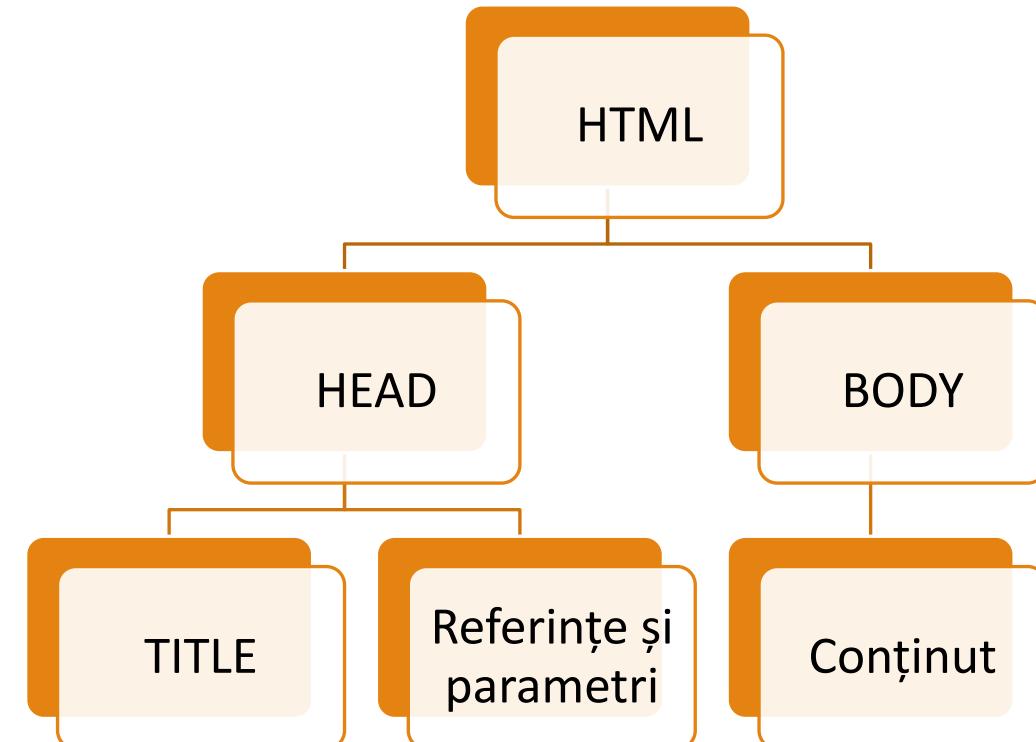
Atribut

- Perechi denumire – valoare atașate nodurilor
- **<test nume1="valoare1" nume2="valoare2">conținut</test>**

Caractere speciale: < - < | > - > | " - " | ' - ' | & - &

Structura unui document HTML

```
<!DOCTYPE html>  
  
<html>  
  <head>  
    <title>titlu</title>  
    <referințe și parametri>  
  </head>  
  <body>  
    <continut>  
  </body>  
  
</html>
```



Tipuri de elemente

Text

- Formatare: ***h1 – h6, p, pre, div, em, strong, span, br, a***
- Liste: ***ul, ol, li***
- Tabele: ***table, thead, tbody, tfoot, tr, th, td***

Formulare: ***form, input, select, option***

Imagine raster

- *img, canvas*

Imagine vectorială: ***svg***

Audio și video: ***audio, video***

Formatare Text

H1, H2, ..., H6 – titluri

- marchează titlurile secțiunilor din document (1 este cel mai important)

P – marchează paragrafele de text

PRE – text preformatat

- Previne **comasarea spațiilor** pentru conținutul tag-ului (în tot subarborele)

DIV, SPAN – containere generice folosite pentru gruparea conținutului

STRONG, EM – folosite pentru marcarea fragmentelor mai importante într-un text

BR – inserează o linie nouă

A – creează un hyperlink (adresa destinație este specificată prin atributul **href**)

Liste și tabele

Liste

- **UL** – *unordered list* (uzual afișate cu buline)
- **OL** – *ordered list* (uzual afișate numeroate)
- Ambele conțin elemente marcate cu **LI** (*list item*)

Tabele

- **TABLE** – reprezintă un tabel și poate conține
 - Secțiuni marcate cu **THEAD**, **TBODY** și **TFOOT**
 - Un titlu marcat cu **CAPTION**; titlul poate conține orice fragment HTML și este afișat în afara tabelului
- Secțiunile conțin rânduri marcate cu **TR**
- Rândurile conțin celule marcate cu **TH** (*table header*) sau **TD** (*table data*);
 - celulele pot conține orice fragment HTML
 - O celulă se poate întinde pe mai multe rânduri / coloane prin intermediul atributelor **rowspan** și **colspan**

Formulare

Sunt conținute în interiorul unui tag **FORM** care poate avea următoarele attribute:

- **action** – adresa la care trebuie trimise datele din formular
- **method** – metoda HTTP care va fi utilizată la transmiterea cererii

Majoritatea controalelor pot fi inserate prin un tag **INPUT** cu următoarele attribute:

- **type** – determină tipul controlului (*text* – implicit, *password*, *radio*, *checkbox*, *file*, *button*, *submit*, *reset*, *hidden*)
- **name** – denumirea sub care va fi transmisă valoarea (a nu se confunda cu atributul **id**)
- **value** – conține valoarea stocată în control

Pentru combo box se folosesc elemente de tip **SELECT** (cu atribut **name**), iar definirea opțiunilor se realizează prin elemente de tip **OPTION** (cu atribut **value**)

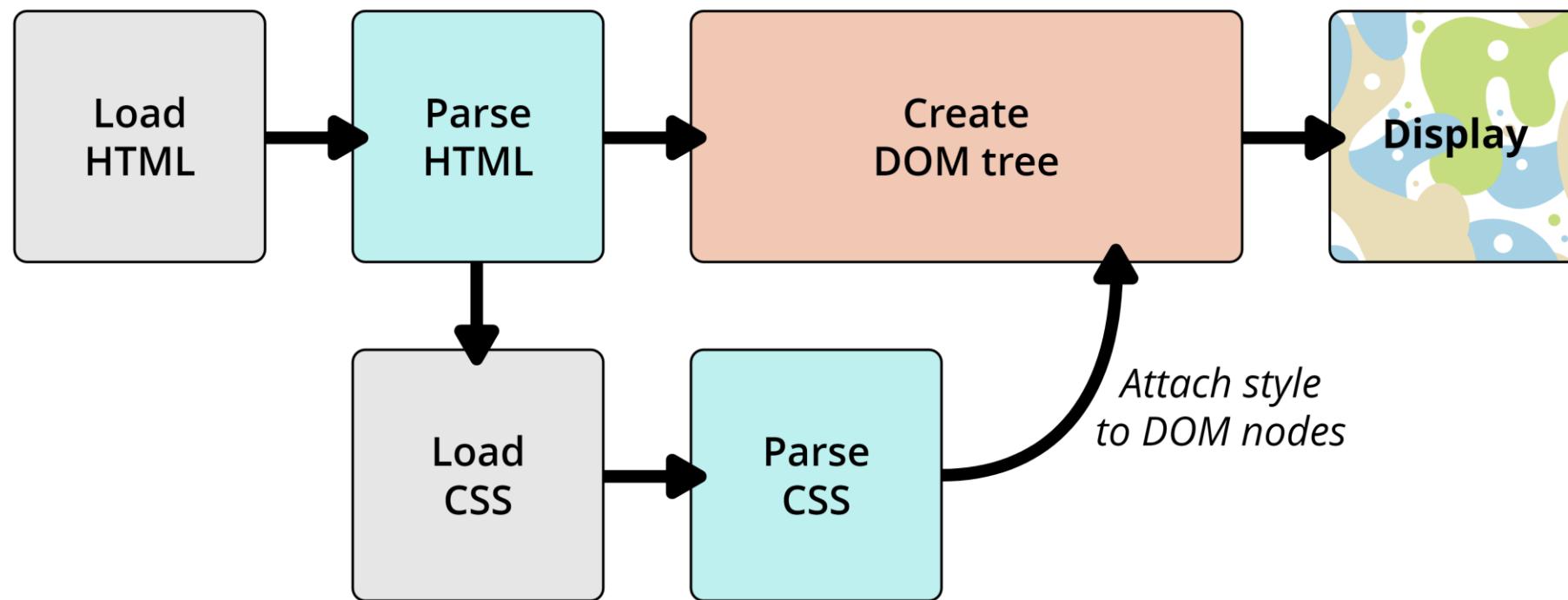
Asocierea de etichete pentru controale se realizează prin elemente de tip **LABEL** (cu atribut **for**)

Elemente de tip *block* și *inline*

	Pe linie nouă	Lățime	Înălțime	Conținut
Block	Mereu	<ul style="list-style-type: none">• 100% din lățimea părintelui• Poate fi setată	<ul style="list-style-type: none">• Înălțimea conținutului• Poate fi setată	Elemente de tip <i>block</i> și / sau <i>inline</i>
Inline	Doar dacă nu mai are loc pe linia curentă	<ul style="list-style-type: none">• Lățimea conținutului• NU poate fi modificată	<ul style="list-style-type: none">• Înălțimea conținutului• NU poate fi modificată	Doar elemente <i>inline</i>

Cascading Style Sheets

Modificarea modului de afișare a elementelor din DOM



Cascading Style Sheets

Format dintr-o serie de reguli de forma:

selector

{

proprietate 1 : valoare 1;

proprietate 2 : valoare 2;

....

}

Utilizare CSS în HTML

Tag **style** – stiluri interne

```
<style type="text/css">  
    th { font-weight: bold; }  
</style>
```

Tag **link** – stiluri externe

```
<link rel="stylesheet" type="text/css" href="test.css" />
```

Atribut **style** – stiluri inline

```
<p style="color: red; font-size: 12pt;">...</p>
```

Selectori

Identifică nodurile din arbore asupra cărora se vor opera modificările

Selectori simpli:

- Element: **TIP_ELEMENT**
- Identificator: **#ID**
- Clasă: **.NUME_CLASĂ**

Alți selectori:

- Universal: *
- Atribut: **selector[atribut="valoare"]**
- Nod copil sau descendent: **selector > selector** sau **selector selector**

Grupare: **selector, selector**

Combinare selectori: **TIP_ELEMENT#ID, TIP_ELEMENT.NUME_CLASĂ**

Reguli de aplicare

Moștenire

- Propagarea valorilor de la un nod părinte la nodurile copil
- Nu toate proprietățile sunt moștenibile

Cascadare

- Mecanismul de alegere aplicat în cazul în care avem mai multe valori în conflict
- Reguli de cascadare
 - Importanță: reguli marcate ca *!important*
 - Specificitatea selectorului
 - Ordinea în fișierul sursă

Formatare text și culori

Proprietăți:

- *font-family* – denumire font sau familie (ex: sans-serif, monospace, Arial, Tahoma, ...)
- *font-size* – dimensiune caractere (ex: small, medium, 12pt, 0.8em, ...)
- *font-weight* – grosime caractere (ex: normal, bold, 400, 700, ...)
- *font-style* – tip caractere (ex: normal, italic, oblique, ...)
- *color, background-color* – culoare text sau fundal (ex: red, #EE0000, #E00, rgb(238, 0, 0), ...)

Unități de măsură:

- relative: %, em, ...
- absolute: pt (1 / 72 inch), px (1 / 90 inch), cm,

Exemplu

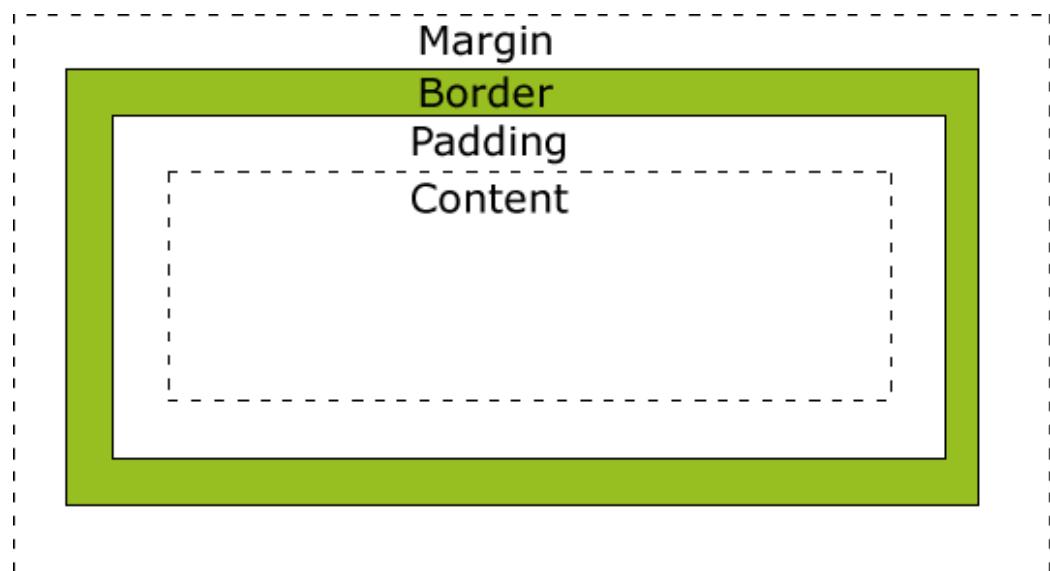
```
body {font-family: tahoma; font-size: 11pt; color: blue;}
```

Setare margini și dimensiuni

Setare dimensiuni:

- *width*
- *height*
- *min/max-width/height*
- *overflow*:
 - **visible** – întreg conținutul este afișat, posibil în afara celulei
 - *hidden* - *conținutul din afara celulei nu este afișat*
 - *scroll* – *hidden + afișare bară de scroll*
 - *auto* – *afișează bara de scroll doar dacă este necesară*

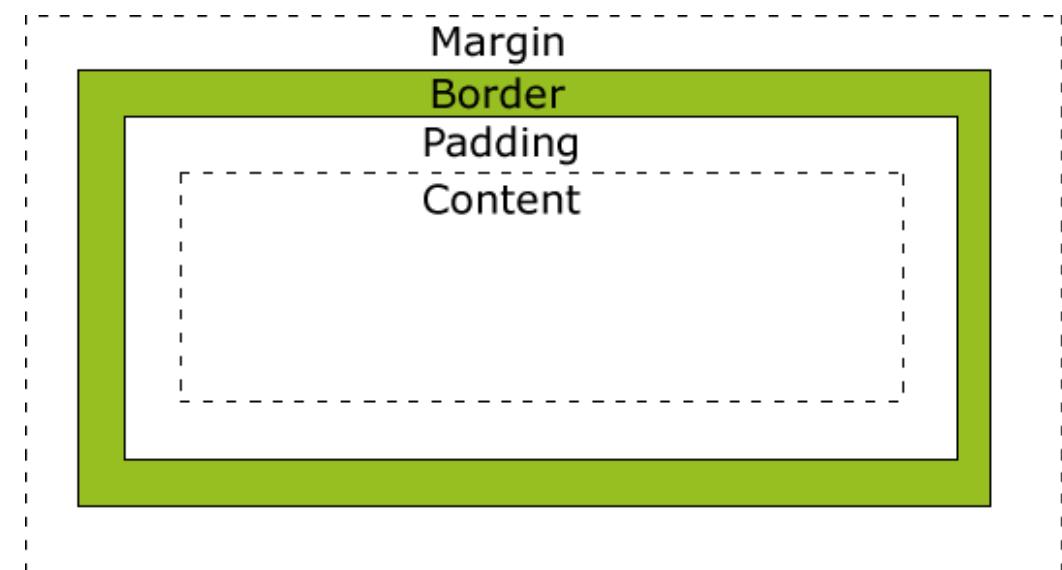
Proprietățile *width* și *height* se referă doar la dimensiunea secțiunii *content* (fără *padding* / *border* / *margin*)



Setare margini și dimensiuni

Setare margini:

- padding sau margin / -top / -bottom / -left / -right
 - *padding*:
 - *3px 4px 5px 6px* (top, right, bottom, left) sau
 - *2px 3px* (top = bottom = 2px, right = left = 3px) sau
 - *3px* (toate laturile)
 - *padding-top: 5px* (setare pentru o singură latură)
 - similar pentru *margin*
 - border / -top / -bottom / -left / -right / -width / -style / -color
 - *border: 2px dotted green*
 - *border-style: solid*
 - *border-bottom-width: 3px;*



Controlul poziționării

Control layout:

- display: none / inline / block
- float: none / left / right
- clear: none / left / right / both

Poziționare în cadrul paginii

- top, right, bottom, left, z-index
 - position:
 - static: poziționare normală, proprietățile (top, ...) sunt ignorate
 - relative: poziționare normală, proprietățile (top, ...) sunt respectate
-
- absolute: nu participă la poziționarea normală; poziționat față de primul parinte cu poziționare absolută
 - fixed: nu participă la poziționarea normală; poziționat față de fereastra browser-ului

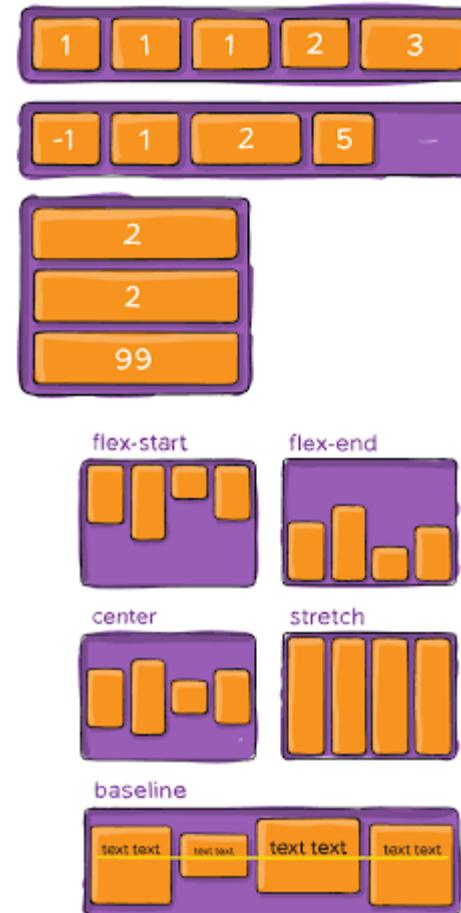
Flexbox (Optional)

Sistem de layout utilizat pentru alinierea conținutului bazat pe un **model 1D** (o singură axă principală)

- Permite poziționarea atât pe verticală (linii) cât și pe orizontală (coloane)
- Proprietăți la nivelul container-ului părinte:
 - `display: flex;`
 - `flex-direction`, `flex-wrap`, `justify-content`, `align-items`, ...
- Proprietăți la nivelul elementelor copil:
 - `order`, `flex-grow/shrink/basis`, `align-self`

Resurse:

- <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox



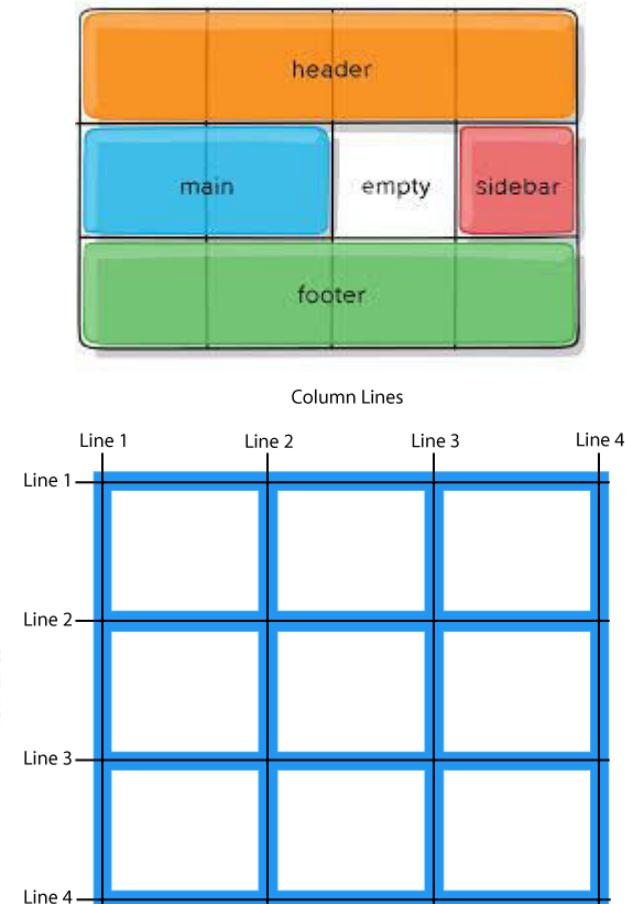
Grid (Optional)

Sistem de layout utilizat pentru alinierea conținutului bazat pe un **model 2D** (două axe principale)

- Permite poziționarea elementelor copil într-o structură matriceală
- Proprietăți la nivelul container-ului părinte:
 - **display: grid;**
 - grid-template-columns, grid-template-rows
 - grid-column/row-gap
- Proprietăți la nivelul elementelor copil:
 - grid-column-start, grid-column-end
 - grid-row-start, grid-row-end

Resurse:

- <https://css-tricks.com/snippets/css/complete-guide-grid/>
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout



Media Queries (Optional)

Sintaxă specială introdusă în CSS3 care permite aplicarea unor reguli doar în anumite condiții:

- În funcție de mediu: *all / screen / print / speech*
- În funcție de caracteristicile mediului (*width, height, ...*)

Sintaxa:

```
@media not/only mediatype and/not/only (media feature) {  
    .selector { ... }  
}
```

Resurse:

- https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries



Bootstrap (Optional)

Bibliotecă CSS

- Dezvoltată de Twitter
- Asigură uniformitatea
- Permite dezvoltarea facilă de aplicații responsive
- Sistem de layout bazat pe rânduri și coloane
- Colecție de componente
- Customizare simplă bazată pe teme

Resurse:

- <https://getbootstrap.com/docs/4.5/getting-started/introduction/>

Limbajul JavaScript

Caracteristici

- Dinamic
- Limbaj funcțional
- Bazat pe obiecte

Utilizare

- Manipulare noduri DOM
- Evenimente
- Procesare
- Comunicare la distanță

Modalități de includere în HTML

Prin intermediul tag-ului `<script>`

A) Fișier extern

```
<script type="text/javascript" src="lib/test.js"></script>
```

B) În cadrul paginii

```
<script type="text/javascript">  
// cod JavaScript ....  
</script>
```

Execuție directă în fereastra de consolă

Afișare mesaje: `console.log(valori);`

Tipuri de date

Tipuri de date de bază

- **Number:** -3.14, 6, 2
- **Boolean:** *true, false*
- **String:** “test”
- **null, undefined**
- **Object:** { nume: “Ana”, varsta:7 } – referință
- **Symbol**

Obiecte speciale

- **Function:** *function f() {...}*
- **Array:** [1, 2, “trei”]

Variabile și expresii

Declarare

- Prin atribuire valoare (**nerecomandat**): `a = 8; a = false;`
- Folosind:
 - **let**: `let b = "test"; let b;` // block scope
 - **const**: `const b = "test"` // block scope
 - **var**: `var a = 3; var b;` // function scope

Exemple:

`a = 10; let b = "test"; b = 1.5;`

`const c = 7; const v = [1, 2, 3]; v[0] = 5;`

Variabile și expresii

Scope

- Global sau local
- Function vs block based

Evaluare expresii

- Operatori similari cu C# (în plus === - egalitate strictă)

Exemple:

a += 7 a === 10 a < b && b > c

a > 10 a++ a = “Ana” + “-“ + “Maria”

Utilizare *template strings*: `Numele este \${prenume} \${nume}.`

Vectori – Obiectul Array

Inițializare: `var v = [];` sau `var v = [1, "Ion"];` sau `var v = new Array(1, 2, 3);`

Accesare elemente: `var i = v[0]; v[1] = 23;`

Dimensiune: `v.length (read / write)`

Metode:

- `push(valoare)` – adăugare la sfârșit
- `pop()` – extrage ultimul element
- `indexOf(valoare)` – întoarce poziția elementului (sau -1)
- `sort()` – sortează vectorul
- `slice(index_start, index_sfârșit)` – extrage un sub-vector
- `splice(index_start, numar_de_sters, val1, val2,)` - ștergere / înlocuire valori

Parcurgere: `for / for..of`

Functii

Declarare:

- *function suma(a,b) { return a + b; } // function declaration*
- *let suma = function(a,b) { return a + b; } // function expression*

Parametri:

- Transmiști prin valoare
- Accesibili prin *arguments*

Apel - nume_functie / expresie(parametri):

- *var rezultat = suma(7,3);*
- *var test = function(val) { return val + 1;}();*

Obiecte funcție (.apply(), .call(), .toString(), length ...)

Closure – Environment Model of Evaluation – vezi <https://sicp.comp.nus.edu.sg/chapters/52>

Arrow functions

Utilizare map / reduce / filter / find

Obiecte

Obiect = colecție de proprietăți (perechi *nume* = *valoare*)

Declarare obiecte

- Literali: `let ob = { nume: "Ana", varsta: 7 }`
- *new*: `let ob = new Object();`

Accesare proprietăți

- Citire: `let v = ob.varsta;` sau `let v = ob["varsta"];` *for (v in ob) {...}*
- Modificare / adăugare: `ob.varsta = 8;` `ob["varsta"] = 8;` `ob.clasa = 2;`
- Ștergere: `delete ob.varsta;`

Metode

- adăugare: `ob.test = function() { console.log("test");}`
- *this*: `ob.afisare = function() { console.log("Nume: " + this.nume); }`
- Apel: `ob.afisare();`

Constructori și moștenire – optional

Funcțiile JavaScript pot fi utilizate pentru construirea de obiecte

- Exemplu: *function Persoana(ume) { this.ume = nume; this.varsta = 1; }*
- Apel: *var ob2 = new Persoana("Maria");*

prototype

- Proprietate a funcției constructor
- Definește proprietățile disponibile în toate obiectele (instanțele create)
- Exemplu: *Persoana.prototype.afisare = function() { console.log("Nume: " + this.ume); };*
- Folosit și pentru implementarea moștenirii:
 - *function Student() { this.facultate = "CSIE"; }*
 - *Student.prototype = new Persoana("-");*
 - *var s = new Student(); s.afisare();*

Clase – optional

Limbajul JavaScript permite utilizarea cuvântului cheie **class** pentru definirea funcțiilor constructor în forma:

```
class NumeClasa {  
    constructor(parametri) {  
        this.proprietate1 = valoare;  
        this.proprietate2 = valoare;  
        ...  
    }  
    metoda1() { ... }  
    metoda2() { ... }  
    get numeProp() { ... }  
    set numeProp(value) { ... }  
}
```

Moștenirea se implementează folosind cuvântul cheie **extends**:

```
class Derivata extends NumeClasa /* definire metode */
```

Programare execuție funcție

A) Execuție o singură dată după un interval de timp specificat

- Programare pornire:

```
var id = setTimeout(funcție, durata[, param1, param2, ...]);
```

- Oprită:

```
clearTimeout(id);
```

B) Execuție repetată la un interval de timp fix

- Programare pornire:

```
var id = setInterval(funcție, durata[, param1, param2, ...]);
```

- Oprită:

```
clearInterval(id);
```

Observație: duratele sunt exprimate în milisecunde

DOM - Structura

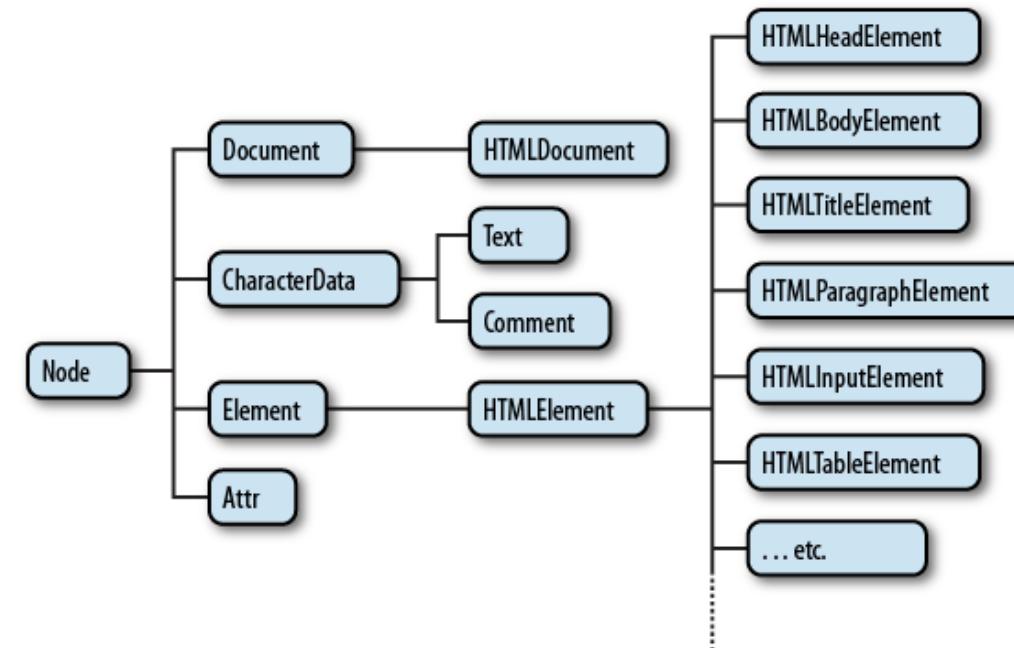
Arbore construit din obiecte JavaScript

Fiecare nod:

- Este un obiect derivat din *Node*
- Conține o colecție de referințe către nodurile copil: *childNodes / children*
- Conține referințe către nodul părinte (*parentNode*) și nodul următor (*nextSibling*)
- Conține metode pentru manipularea nodurilor copil: *append(nod), remove(), ...*

Rădăcina: *document.documentElement, document.head, document.body*

Nodurile au metode și proprietăți specifice în funcție de tag-ul HTML utilizat pentru construirea nodului



Sursa: <http://web.stanford.edu/class/cs98si/slides/the-document-object-model.html>

Regăsire noduri

Pe bază de identificator

- elem = *document.getElementById(identificator)*

Pe bază de selector CSS

- elem = element.*querySelector*("selector CSS");
- lista = element.*querySelectorAll*("selector CSS");

Alte metode

- lista = element.getElementsByClassName("clasa CSS");
- lista = element.getElementsByTagName("tag HTML");

Manipulare noduri

Construire nod:

- elem = *document.createElement("tag HTML"); / document.createTextNode("text")*
- Proprietăți: innerText, innerHTML
- Metode de adăugare nod:
 - parinte.prepend(copil)
 - parinte.append(copil)
 - nod.before(nodNou)
 - nod.after(nodNou)
 - nod.replaceWith(nodNou)
- Metode de eliminare nod:
 - parinte.removeChild(copil)
 - nod.remove()

Manipulare Atribute

Accesare atribute:

- elem.getAttribute – accesare valoare atribut individual
- elem.attributes – colecția de atribute
- elem.hasAttribute(name) – verificare existență
- elem.getAttribute(name) – obținere valoare
- elem.setAttribute(name, value) – modifica valoare
- elem.removeAttribute(name) – ștergere atribut

Asociere date proprii:

- Atribute HTML: <tag **data-denumire**=“valoare”></tag>
- Din JavaScript: element.**dataset**.denumire

Manipulare Noduri

Accesare atribute CSS:

- elem.style.numeAtributCSS
- elem.className – string, corespunzător atributului *class* din HTML
- elem.classList – obiect care permite manipularea listei de clase
 - (metode *add / remove / toggle / contains*)

Obținere coordonate element fata de fereastră: elem.getBoundingClientRect()

Exemplu:

```
let p = document.createElement("p");
p.innerText = "test";
document.querySelector("body").append(p);
p.style.backgroundColor = "red";
```

Tratare evenimente

Adăugare event handler:

- Prin attribute HTML:
 - <element atributEveniment="nume funcție">...</element>
 - Exemplu: <a **onclick="test()"**
- Prin proprietăți nod:
 - element.proprietateEveniment = funcție;
 - Exemplu: document.getElementById("test").**onclick** = function() {console.log("un mesaj");}
- **Prin metoda addEventListener:**
 - element.**addEventListener**(tip, funcție);
 - Exemplu: document.getElementById("test").**addEventListener**("click", function() {console.log("un mesaj");});

Eliminare event handler:

- element.**removeEventListener**(tip, funcție);

Tratare evenimente

Accesare element sursă – *this*

Parametri eveniment – obiect *event*

- General: *target*, *currentTarget*, *type*, *preventDefault()*
- Tastatură: *key*, *keyCode*, *altKey*, *ctrlKey*, *shiftKey*
- Mouse: *pageX*, *pageY*, *button*, *altKey*, *ctrlKey*, *shiftKey*

Evenimente

- General: *load* (*window*), *DOMContentLoaded* (*document*)
- Tastatură: *keydown*, *keypress*, *keyup*
- Mouse: *mouseenter*, *mouseleave*, *mousemove*, *mouseup*, *mousedown*, *click*, *dblclick*

Comunicarea cu serverul - optional

Prin obiecte de tip ***XMLHttpRequest***

Permite crearea, transmiterea și receptia de cereri HTTP(S)

Construire obiect cerere:

```
var cerere = new XMLHttpRequest();
```

Funcții:

- *open(method, url)* – initializează un obiect ***XMLHttpRequest***
- *send(data)* – începe procesarea asincronă a cererii
- *abort()* – anulează o cerere în desfășurare
- *setRequestHeader(header, value)* – permite transmiterea de valori în header-ul cererii
- *getResponseHeader(header)* – permite citirea valorilor din header-ele primite de la server

Comunicarea cu serverul – optional

Proprietăți:

- *readyState* – intreg care indică starea curentă a cererii (0 – netrimisă, ..., 4 – done)
- *response* – răspunsul primit de la server de tipul indicat de *responseType*
- *status, statusText* – codul HTTP pentru răspuns în format numeric și text

Evenimente - addEventListener:

- *load* – cererea a fost executată și a fost primit răspunsul
- *readystatechange* – s-a modificat starea curentă a obiectului
- *abort* – cererea a fost anulată de client
- *timeout* – a expirat timpul alocat
- *error* – cererea s-a încheiat cu o eroare

Communicarea cu serverul – Fetch API

Obiecte ***Promise*** – încapsulează o operație asincronă și callback-urile asociate

Funcție: **fetch(URL[, {options}])** – întoarce un obiect de tip *Promise*

Opțiuni:

- *method* – ‘GET’ (implicit) sau ‘POST’
- *headers* – obiect care conține headerele de trimis către server sub formă de proprietăți
- *body* – conținutul pentru cereri de tip POST (string, FormData, ...)

Obiectul răspuns:

- proprietăți pentru determinarea rezultatului (*ok*, *status*, *statusText*)
- *headers* – obiect care permite citirea datelor din header-ul HTTP
- Metode pentru citirea conținutului (*text()*, *json()*, *formData()*, ...) – întorc un obiect de tip *Promise*

JavaScript – Execuție asincronă

Instrucțiuni dedicate pentru obiecte *Promise*:

- ***async function***: declară o funcție asincronă care întoarce un obiect *Promise*
 - permite utilizarea operatorului *await* în corpul funcției
- ***await***: operator care permite execuția unei funcții asincrone ce întoarce un obiect *Promise* și continuarea execuției doar la primirea rezultatului

Exemplu:

```
async function app() {  
    var raspuns = await fetch('./dateEU.json');  
    var json = await raspuns.json();  
    console.log(json);  
}  
app();
```

Resurse:

- <http://javascript.info/async>
- <https://developer.mozilla.org/en-US/docs/Web/API/WindowOrWorkerGlobalScope/fetch>

Biblioteca jQuery – optional

Colecție de funcții JavaScript pentru:

- Regăsire elemente din DOM
- Manipulare elemente, atribute și proprietăți CSS
- Înlănțuire operații pe seturi elemente
- Animație
- Comunicare HTTP

Funcția jQuery / \$ – optional

1. Regăsire elemente DOM: **`$("selector CSS")`**

- `var leg = $("a.test");`

2. Construire obiect jQuery: **`$(element)`**

- `var a = document.getElementById("leg1");`
- `var ajQuery = $(a);`

3. Construire elemente noi: **`$("cod HTML")`**

- `var p = $("<p>paragraf nou</p>");`

4. Execuție cod după inițializarea DOM: **`$(funcție)`**

- `$(function(){`
- `// operații care utilizează DOM`
- `})`

Obiecte jQuery – optional

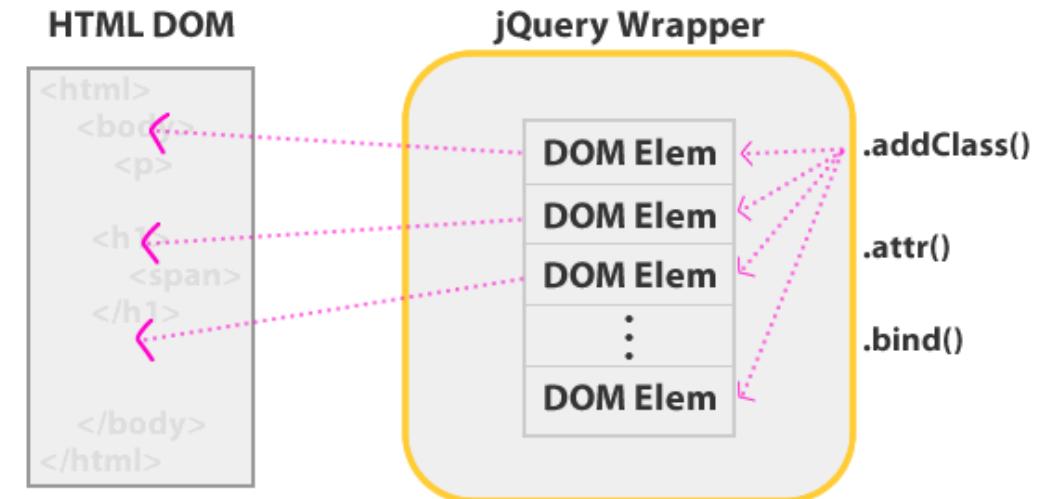
Fiecare obiect jQuery conține o colecție de 0+ referințe la obiectele DOM selectate.

Accesarea elementelor DOM:

- Operator `[]` sau funcția `get`:
 - `$("td")[3]` sau `$("td").get(3)`
 - Rezultatul obținut este un obiect DOM
- Proprietate `length`:
 - `$("td").length`

Operații asupra obiectului jQuery:

- În general se aplică pe toate elementele și întorc o referință la obiectul jQuery (pentru înlățuire)
- Funcțiile care citesc o valoare se aplică asupra primului element și întorc valoarea citită



jQuery - exemple de operații – optional

Citire / modificare atribută HTML

- `$("#idTest").attr("href", "doc.html");`
- `var url = $("#idTest").attr("href");`
- `.text() / .val()`

Citire / modificare proprietăți CSS

- `$("#idTest").css("color", "red");`

Manipulare noduri DOM

- `$("#idTest").append($("#<p>test</p>"));`
- `părinte.empty() / copil.appendTo(părinte) / copil.remove()`

Tratare evenimente

- `$("#idTest").click(func);`
- `$.on("ev", func);`

jQuery – comunicare – optional

Obținerea de date de la server prin comenzi **GET**:

\$.get(url, func) sau **\$.getJSON(url, func)**

```
$.get("date.txt", function(txt) { console.log(txt); });
```

```
$.getJSON("date.txt", function(obj) { console.log(obj.proprietate); });
```

Trimiterea de date către server prin comenzi **POST**:

\$.post(url, date[, func])

```
$.post("procesare.aspx", JSON.stringify(obj), function(obj) { console.log(obj); });
```

Opțiuni complete: **\$.ajax(...)**

II. Imagine

1. Modele de culoare

2. Grafică raster

3. Grafică vectorială

4. Animație

1. Modele de culoare

Model de culoare

- Model matematic care descrie modalitatea de reprezentare a culorilor sub formă de tupluri numerice
- Exemple: RGB, CMY

Spațiu de culori

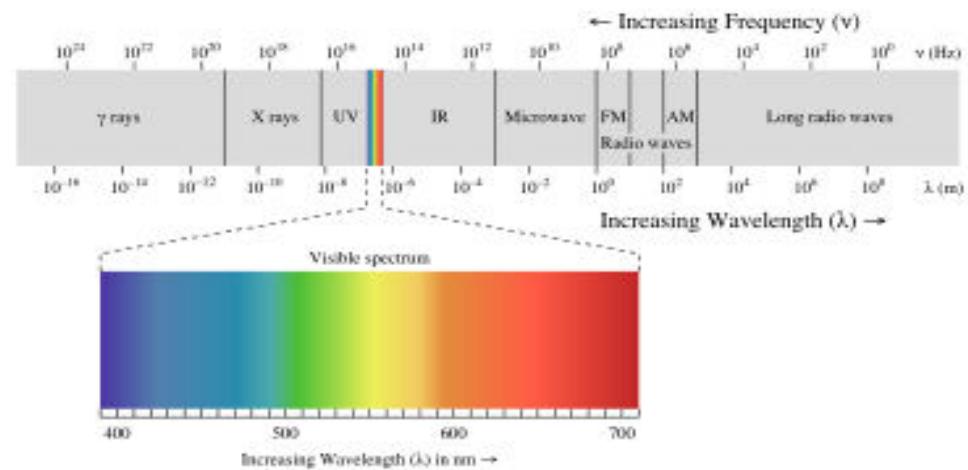
- Modelul de culoare împreună cu instrucțiunile de reprezentare fizică
- Exemple: sRGB, AdobeRGB, Pantone

Caracteristici

- Tin seama de modalitatea de percepere a luminii de către ochiul uman
- Culori de bază albastru (S), verde (M) și roșu (L)

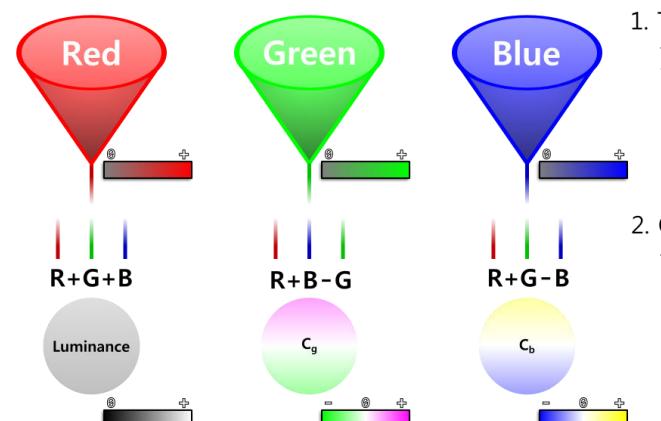
Lumina

Radiație electromagnetică cu lungimea de undă aproximativ între 400 și 700 nm



Vederea umană:

- Trei tipuri de celule conice fotosensibile (pentru culorile roșu, verde, albastru)
- Culoarea este procesată pe baza a trei componente (luminanță, C_g și C_b)



1. Trichromatic Stage

Trichromatic cone cells respond positively to one of three frequencies exhibited by photons arriving on their surface.

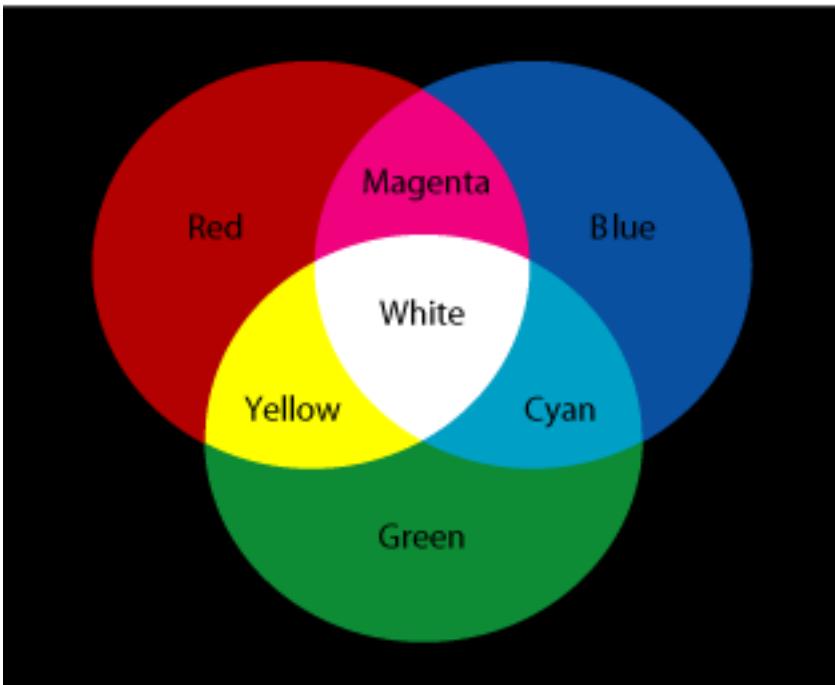
2. Opponent Process Stage

The three color channels are discovered by nearby opponent cells.

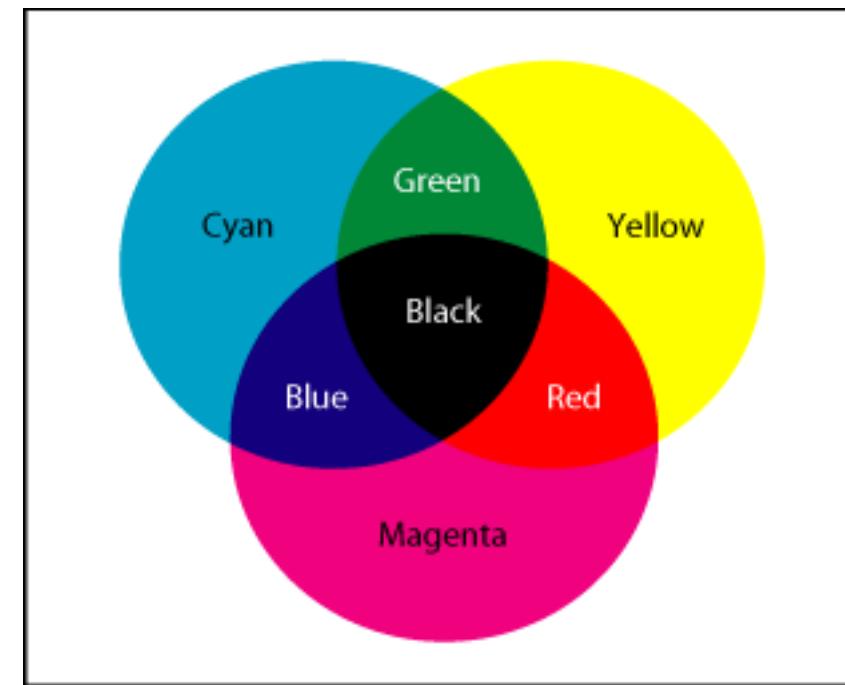
Opponent cells tuned to luminosity are excited by the red, green, and blue color signals.

C_g cells are excited by red and blue and inhibited by green. C_b cells are excited by red and green and inhibited by blue.

Tipuri de modele



Model aditiv



Model substractiv

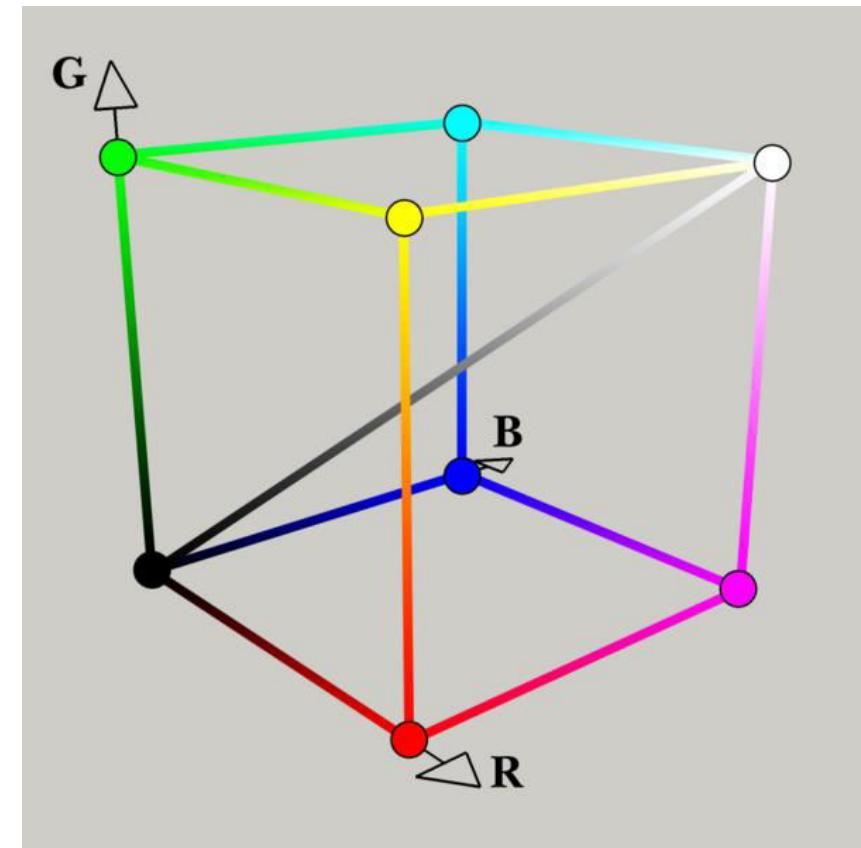
Modelul RGB

Model aditiv

Bazat pe culorile de bază roșu, verde și albastru

Culoarea variază în funcție de dispozitiv

(în lipsa unui spațiu de culoare)



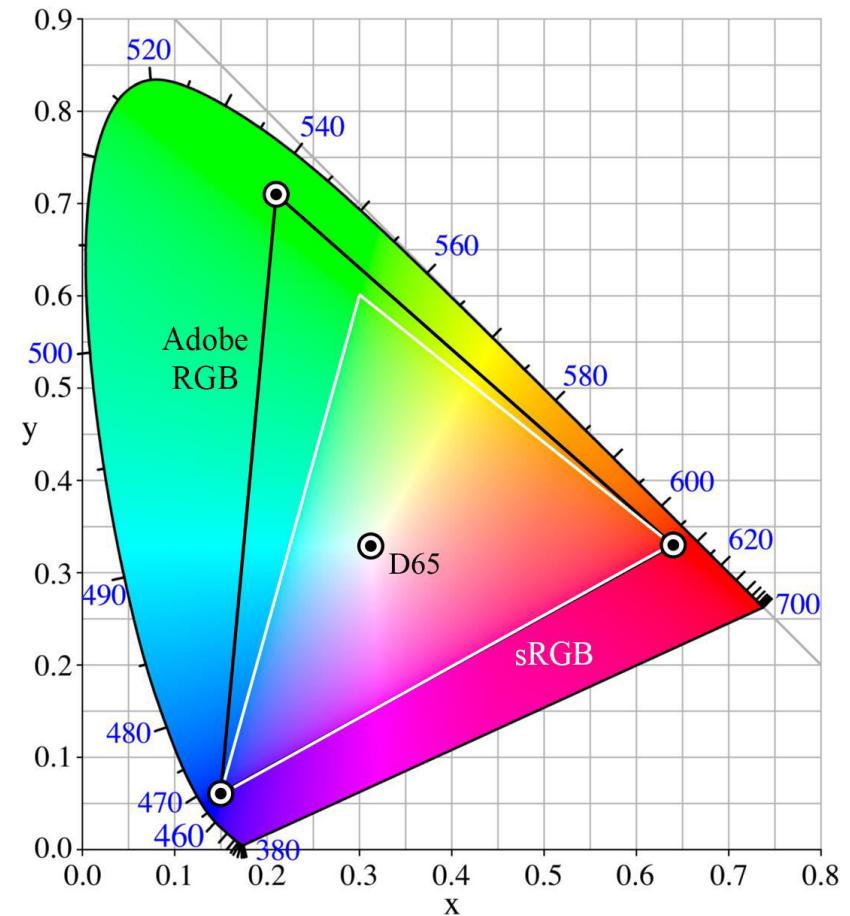
Spații de culoare RGB

sRGB

- Dezvoltat de HP și Microsoft
- Standard pentru monitoare / imprimante / web
- Utilizat ca standard implicit

AdobeRGB

- Dezvoltat de Adobe
- Acoperă aproape complet spațiul de culori CMYK



Modelul HSL (B,V)

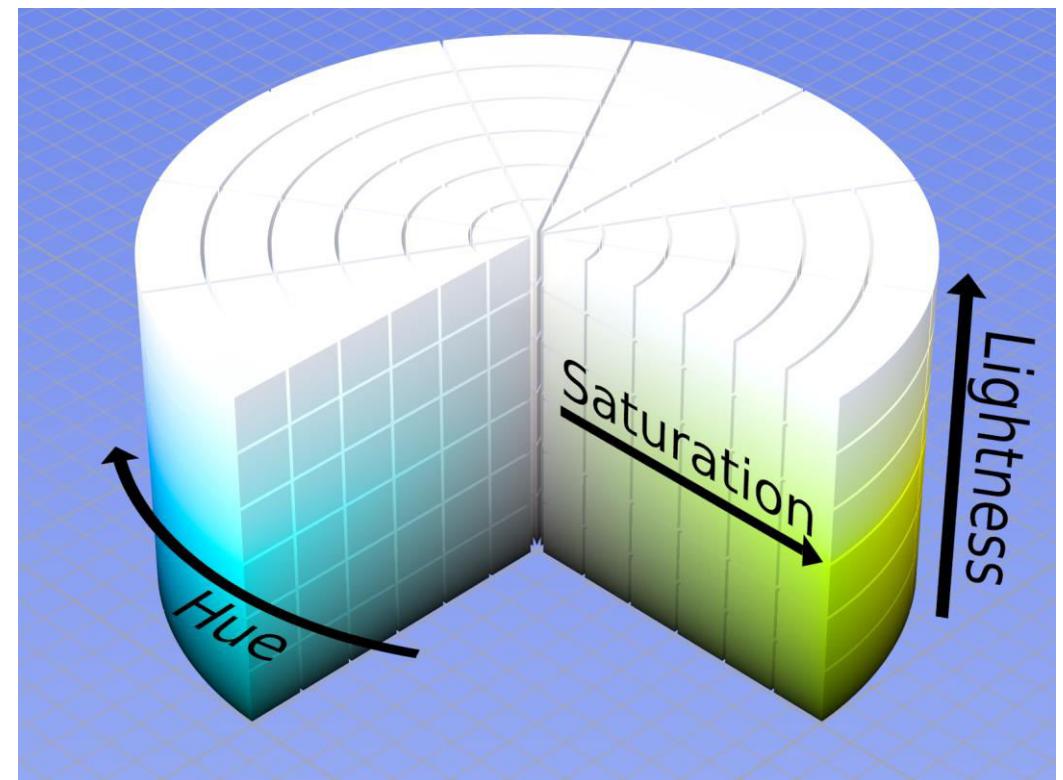
Reprezentare sub formă de coordonate cilindrice

Bazat pe aceleași culori de bază:

- Red - 0^0
- Green - 120^0
- Blue - 240^0

Axa centrală – tonuri de gri

Utilizat în special pentru selecție de culoare



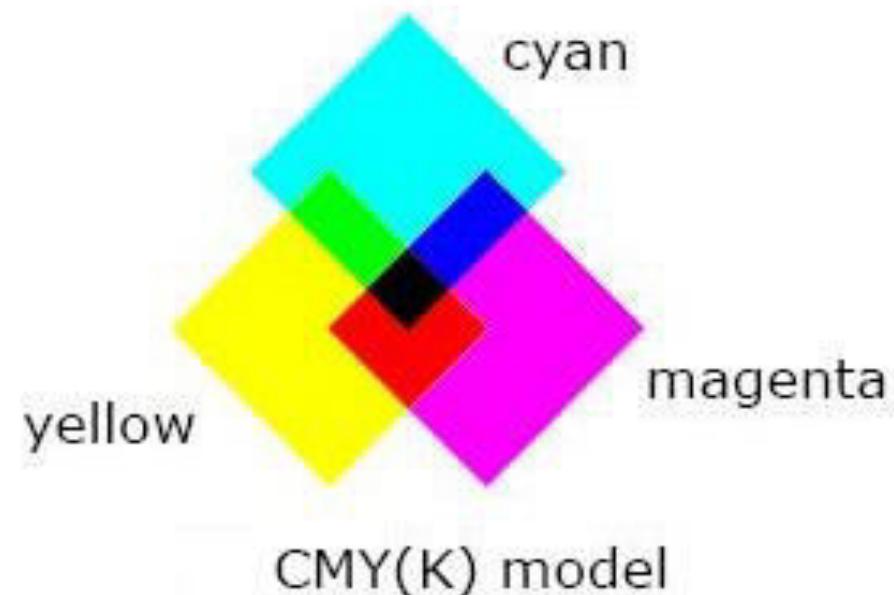
Modelul CMY(K)

Model subtractiv

Culori utilizate: cyan, magenta, yellow

Maschează culorile pe o suprafață albă

Utilizare: materiale tipărite



CSS – Specificarea colorilor

Format hexazecimal:

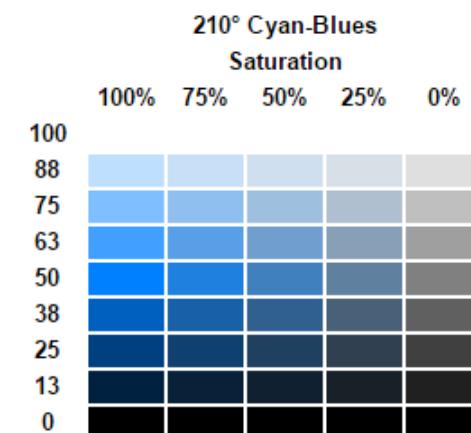
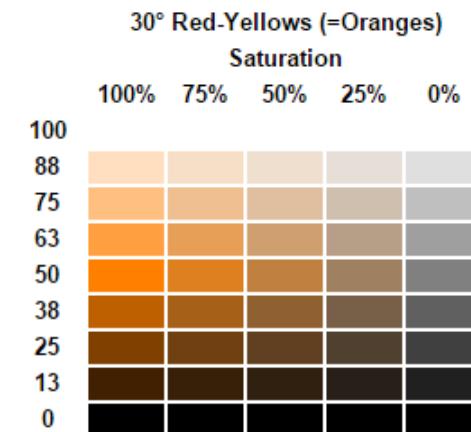
- **#rgb** sau **#rrggbb**
- **r,g,b** sunt cifre în baza 16

Format RGB

- **rgb(red, green, blue)** sau **rgba(red, green, blue, alpha)**
- **red, green, blue** – numere de la 0 la 255 sau procente
- **alpha** – număr între 0 – transparent și 1 – opac sau procent

Format HSL

- **hsl(hue, saturation, lightness)** sau **hsla(hue, saturation, lightness, alpha)**
- **hue** – număr de la 0 la 360
- **saturation, lightness** – procente
- **alpha** – număr între 0 – transparent și 1 – opac sau procent



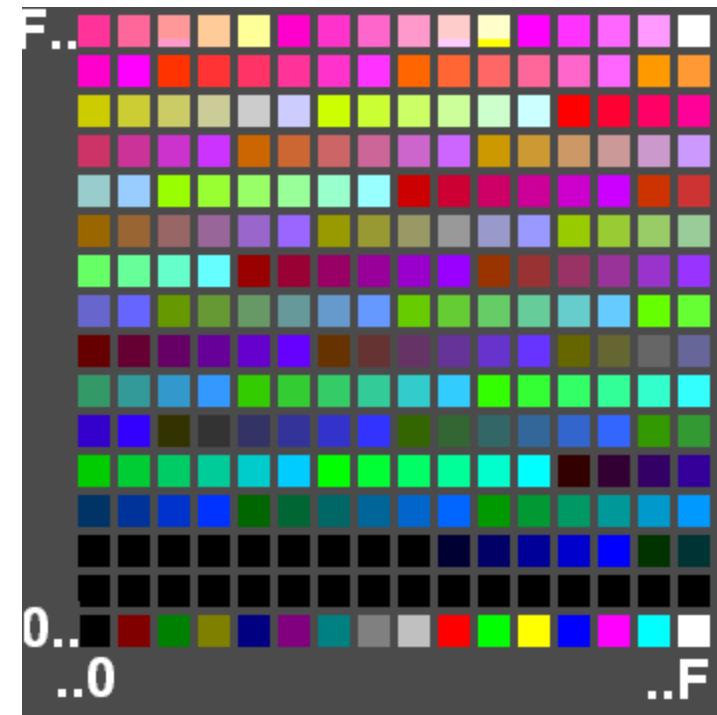
Palete de culori

Tabel de corespondență

index – tuplu (R, G, B)

Utilizare

- Reducerea cantității de informație necesară pentru reprezentarea culorilor
- Pretabilă pentru imagini cu un număr redus de culori (exemplu: diagrame fără gradienți)



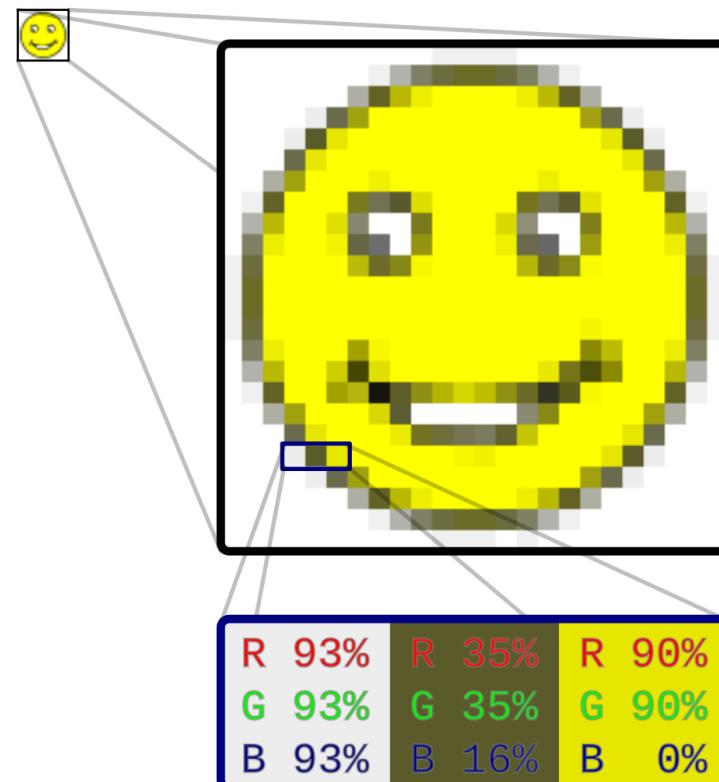
2. Grafică raster

Tipuri principale de grafică:

- raster și vectorial

Grafica raster (bitmap, matriceală)

- Reprezentare sub formă de matrice de puncte
- Fiecare punct (denumit pixel) stochează informația de culoare
- Culorile sunt stocate conform unui model de culoare
 - direct
 - prin intermediul unei palete de culori



Sursa: <http://commons.wikimedia.org/wiki/File:Rgb-raster-image.svg>

Informații generale

Caracteristici principale imagine raster

- Rezoluție (numărul de linii și coloane stocate în matrice, numărul total de pixeli sau densitate)
- Adâncime de culoare (cantitatea de informație stocată de către fiecare pixel)

Utilizare

- Reprezentare imagine pe monitor
- Captare imagini din surse externe

Avantaje și dezavantaje

- + Poate reprezenta orice imagine
- Codaj sărac în informație (nu ia în considerare semantica imaginii)
- Dimensiune mare
- Nu se pot adapta unei scări variabile de vizualizare

Desenare elemente grafice

Presupune colorarea celulelor matricei plecând de la ecuația matematică

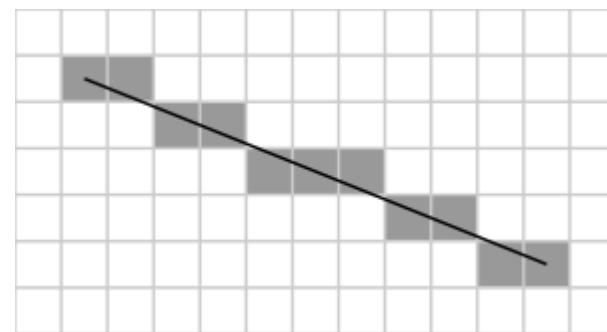
Exemplu: segment de dreaptă $(x_1, y_1) \rightarrow (x_2, y_2)$

Se pleacă de la ecuația dreptei $y=mx+b$ și se determină valorile pentru $m=(y_2-y_1)/(x_2-x_1)$ și $b=(y_1 - mx_1)$

Se colorează punctele corespunzătoare ecuației:

```
dx = x2 - x1; dy = y2 - y1;
for x from x1 to x2 {
    y = y1 + dy * (x - x1) / dx
    plot(x, y)
}
```

(se presupun $d_x \geq d_y$ și $x_1 > x_2$)



Elementul canvas

HTML – includere în document:

```
<canvas id="test" width="250" height="150"></canvas>
```

JavaScript – obținere referință context grafic:

```
// obținere obiect HTMLCanvasElement din DOM  
var canvas = document.getElementById('test'); // sau var canvas = $('#test')[0];  
  
var w = canvas.width, h = canvas.height;  
  
// obținere context grafic (obiect de tip CanvasRenderingContext2D)  
var ctx = canvas.getContext('2d');
```

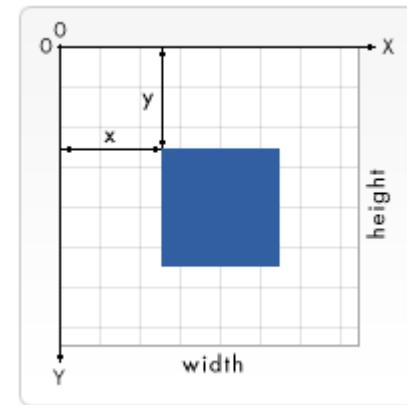
canvas – Modalități de desenare

Desenare directă – dreptunghiuri:

- *fillRect(x, y, width, height)* – umplere suprafață
- *strokeRect(x, y, width, height)* – desenare contur
- *clearRect(x, y, width, height)* – ștergere suprafață

Desenare folosind căi:

- ***beginPath()*** – deschide o cale nouă
- instrucțiuni de desenare / poziționare
- opțional *closePath()* – închide calea (unește cu punctul de început)
- ***fill()*** – umple calea și / sau ***stroke()*** – desenează liniile



canvas – Instrucțiuni desenare

Deplasare:

moveTo(x, y) – modifică poziția curentă

Desenare:

lineTo(x, y) – adaugă o linie de la poziția curentă la punctul specificat

rect(x, y, width, height) – adaugă un dreptunghi

arc(x, y, radius, startAngle, endAngle, anticlockwise) – adaugă un arc de cerc

quadraticCurveTo(cp1x, cp1y, x, y) – adaugă o curbă quadratică (un punct de control)

bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y) – adaugă o curbă Bézier (două puncte de control)

Desenare text: *fillText* sau *strokeText(string, x, y)*

canvas – Atribute

Salvare și restaurare atribute context folosind stiva: ***save()*** și ***restore()***

Atribute:

- *fillStyle* sau *strokeStyle* = “culoare” – modifică culoarea de desenare
 - `ctx.fillStyle = "#FFA500";`
 - `ctx.strokeStyle = "rgba(255,165,0,1);`
- *lineWidth* = dimensiune – modifică grosimea liniilor
- *lineCap*, *lineJoin*, *miterLimit* – modifică proprietățile liniilor (rotunjire capete, îmbinări)
- *font* = “specificație font” – modifică caracteristicile textului (exemplu: “bold 18px Arial”)
- *textAlign* – poziția textului față de coordonata x (*left*, *right* sau *center*)
- *textBaseline* – poziția textului față de coordonata y (*top*, *hanging*, *middle*, *alphabetic* sau *bottom*)

canvas – Desenare curbe

Suport pentru desenare curbe *Bézier*

- cuadratică (un punct de control)
- cubică (două puncte de control)

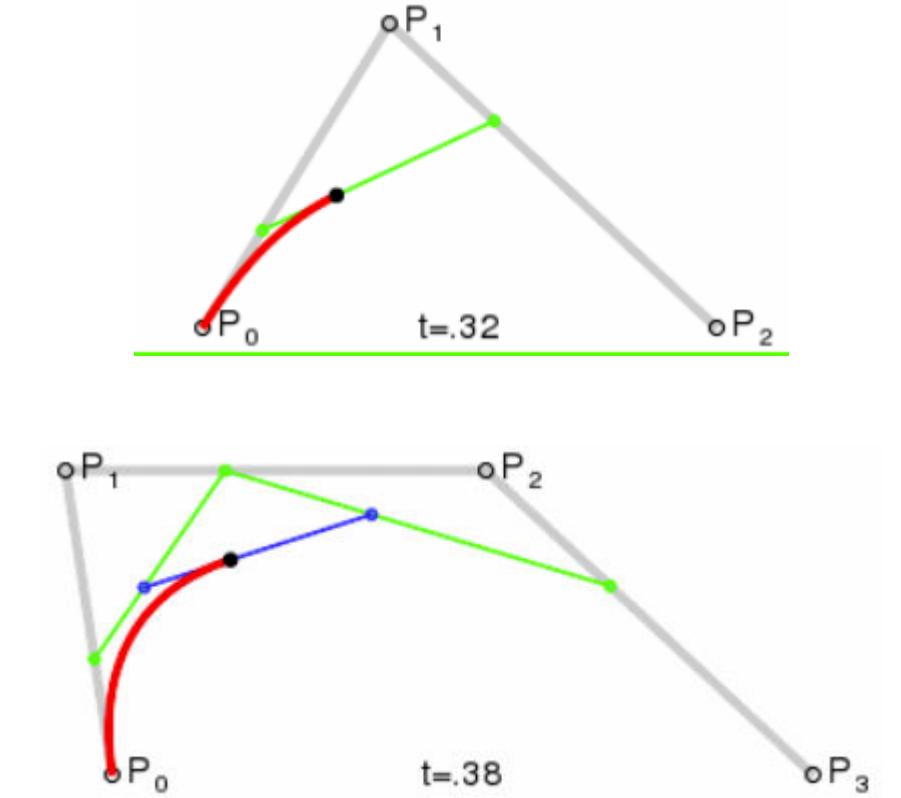
Formule:

$$P = (1-t)P_1 + tP_2$$

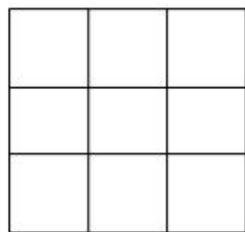
$$P = (1-t)^2P_1 + 2(1-t)P_2 + t^2P_3$$

Algoritmul de *Casteljau* (cuadratică):

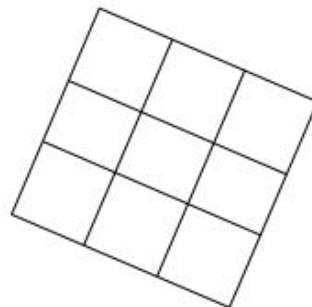
- se consideră punctele P_0 , P_1 și P_2
- se construiesc segmentele P_0P_1 și P_1P_2
- Pentru t de la 0 la 1
 - Se determină punctul aflat la o distanță proporțională t față de începutul segmentului pentru P_0P_1 și P_1P_2
 - Se construiește un segment nou din cele două puncte obținute și se repetă procesul



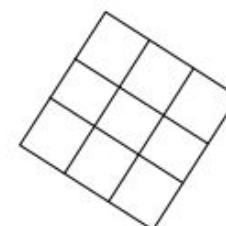
Tipuri de transformări 2D



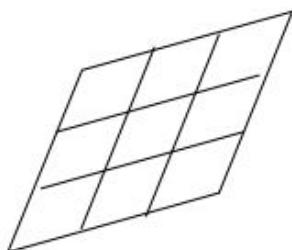
Identică



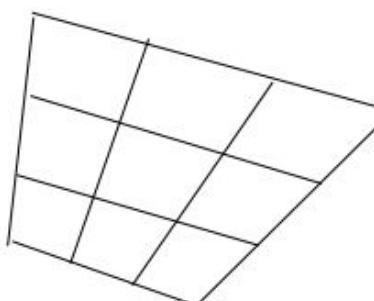
Euclidiană
(translație, rotație)



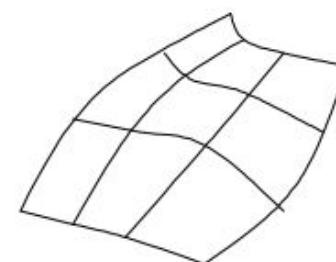
Helmert
(+scalare)



Afină
(+deformare)



Proiectivă
(+paralelism)

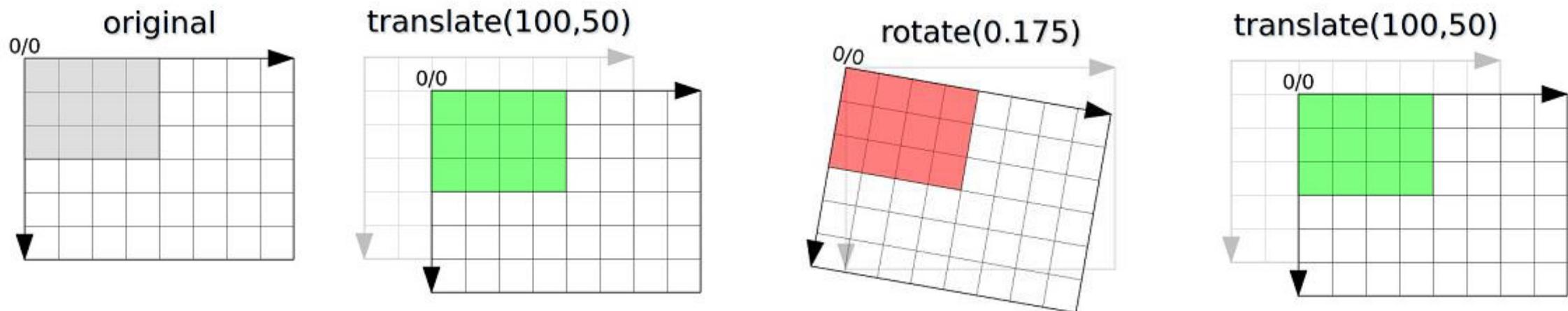


Polinomială
(+proprietăți geometrice)

canvas – Transformări

Operații de transformare simplă:

- *translate(x, y)* – translatează sistemul de coordonate cu numărul de pixeli specificați
- *rotate(angle)* – rotește sistemul de coordonate cu unghiul specificat (în radiani)
- *scale(x, y)* – scalează sistemul de coordonate cu factorii specificați (în [0...1])



canvas - Transformări

Forma generală pentru transformări afine:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} m_{11} & m_{21} & d_x \\ m_{12} & m_{22} & d_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Componere transformare (prin înmulțire):

transform(m11, m12, m21, m22, dx, dy)

Înlocuire transformare curentă:

setTransform(m11, m12, m21, m22, dx, dy)

resetTransform() – revenire la sistemul de coordonate standard

canvas - Transformări

Semnificație parametri:

- m_{11} : scalare pentru axa Ox
- m_{12} : rotire pentru axa Ox
- m_{21} : rotire pentru axa Oy
- m_{22} : scalare pentru axa Oy
- d_x : translatare pentru axa Ox
- d_y : translatare pentru axa Oy

Imaginea raster în context Web

Afișarea imaginilor în HTML:

- Utilizare nod DOM (HTMLImageElement)
 - *width, height*: dimensiunea obiectului în fereastră
 - *naturalWidth, naturalHeight*: dimensiunea matricei raster
 - *src*: URL-ul imaginii sursă; modificarea determină începutul procesului de încărcare (asincron)
 - eveniment *load*
- Exemplu:

```
var image = $("<img>")
    .on('load', function () { $("body").append($(this)); })
    .attr("src", "media/Penguins.jpg");
```

- Alte metode:
 - Element *input* cu *type='file'*
 - *Drag and drop*

canvas – Desenare imagini

Desenare fără scalare:

drawImage(image, x, y)

Desenare cu preluare porțiune imagine și scalare:

drawImage(image, sx, sy, sWidth, sHeight, dx, dy, dWidth, dHeight)

Surse posibile:

- element **img**
- alt element **canvas**
- element **video**

canvas – Acces raster

Stocare raster – obiect *ImageData*:

- *width, height*: dimensiunile matricei
- *data*: vector care conține matricea liniarizată

Operații:

- *getImageData(left, top, width, height)*: extrage o porțiune din imagine ca obiect *ImageData*
- *putImageData(imageData, x, y)*: aplică datele pe imaginea afișată în *canvas*

Restricții de origine

canvas – Access raster

Parcure elemente:

```
var imageData = context.getImageData(  
    0, 0, canvas.width, canvas.height);  
  
for (var y = 0; y < canvas.height; y++) {  
    for (var x = 0; x < canvas.width; x++) {  
        var i = (y * canvas.width * 4) + x * 4;  
  
        var rosu = imageData.data[i]; // [0..255]  
        var verde = imageData.data[i+1]; // [0..255]  
        var albastru = imageData.data[i+2]; // [0..255]  
        var transparenta = imageData.data[i+3]; // [0..255]  
    }  
}
```

Efecte simple de culoare

Filtrare culoare

- Exemplu pentru roșu: $r' = r; g = 0; b = 0$

Negativ

- $r' = 255 - r; g' = 255 - g; b' = 255 - b;$

Transformare în tonuri de gri

- $r' = g' = b' = 0.299 * r + 0.587 * g + 0.114 * b$

Modificare strălucire

- $r' = r + \text{valoare}; \text{if } (r > 255) r = 255 \text{ else if } (r < 0) r = 0;$
- similar pentru celelalte două componente

Modificare contrast

- $r' = (((r / 255) - 0.5) * \text{valoare} + 0.5) * 255; \text{if } (r > 255) r = 255 \text{ else if } (r < 0) r = 0;$
- similar pentru celelalte două componente

Transformare RGB → HSL

$$M = \max(R, G, B)$$

$$m = \min(R, G, B)$$

$$C = M - m$$

$$H' = \begin{cases} \text{undefined}, & \text{if } C = 0 \\ \frac{G-B}{C} \bmod 6, & \text{if } M = R \\ \frac{B-R}{C} + 2, & \text{if } M = G \\ \frac{R-G}{C} + 4, & \text{if } M = B \end{cases}$$

$$H = 60^\circ \times H'$$

$$L = \frac{1}{2}(M + m) = 0.5 * \max(R, G, B) + 0.5 * \min(R, G, B)$$

$$S_{HSL} = \begin{cases} 0, & \text{if } L \in \{0, 1\} \\ \frac{C}{1-|2L-1|}, & \text{otherwise} \end{cases}$$

Transformare HSL → RGB

$$C = (1 - |2L - 1|) \times S_{HSL}$$

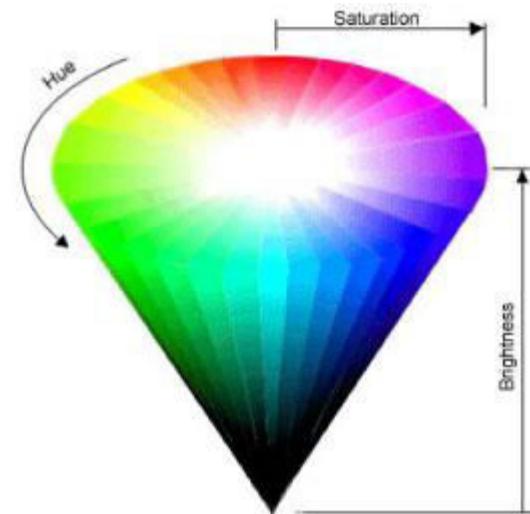
$$H' = \frac{H}{60^\circ}$$

$$X = C(1 - |H' \bmod 2 - 1|)$$

$$(R_1, G_1, B_1) = \begin{cases} (0, 0, 0) & \text{if } H \text{ is undefined} \\ (C, X, 0) & \text{if } 0 \leq H' < 1 \\ (X, C, 0) & \text{if } 1 \leq H' < 2 \\ (0, C, X) & \text{if } 2 \leq H' < 3 \\ (0, X, C) & \text{if } 3 \leq H' < 4 \\ (X, 0, C) & \text{if } 4 \leq H' < 5 \\ (C, 0, X) & \text{if } 5 \leq H' < 6 \end{cases}$$

$$m = L - \frac{1}{2}C$$

$$(R, G, B) = (R_1 + m, G_1 + m, B_1 + m)$$



Histograma unei imagini

Permite analiza distribuției tonurilor în cadrul unei imagini

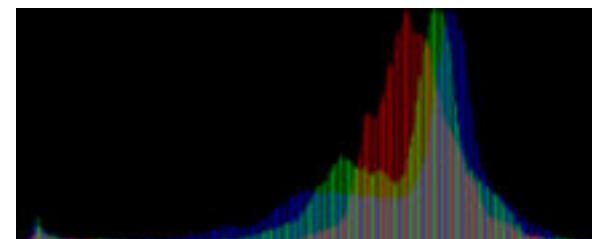
Construcție:

- Ox: intensitatea (de obicei 0-255)
- Oy: numărul de pixeli existenți pentru fiecare valoare a intensității

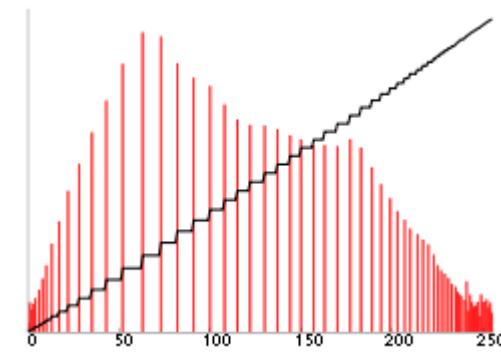
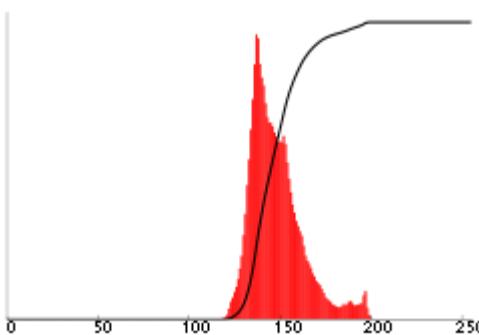
Poate fi construită pentru fiecare canal în parte

Exemple de utilizări:

- Îmbunătățire contrast prin egalizare histogramă
- Descoperire muchii / segmentare imagine

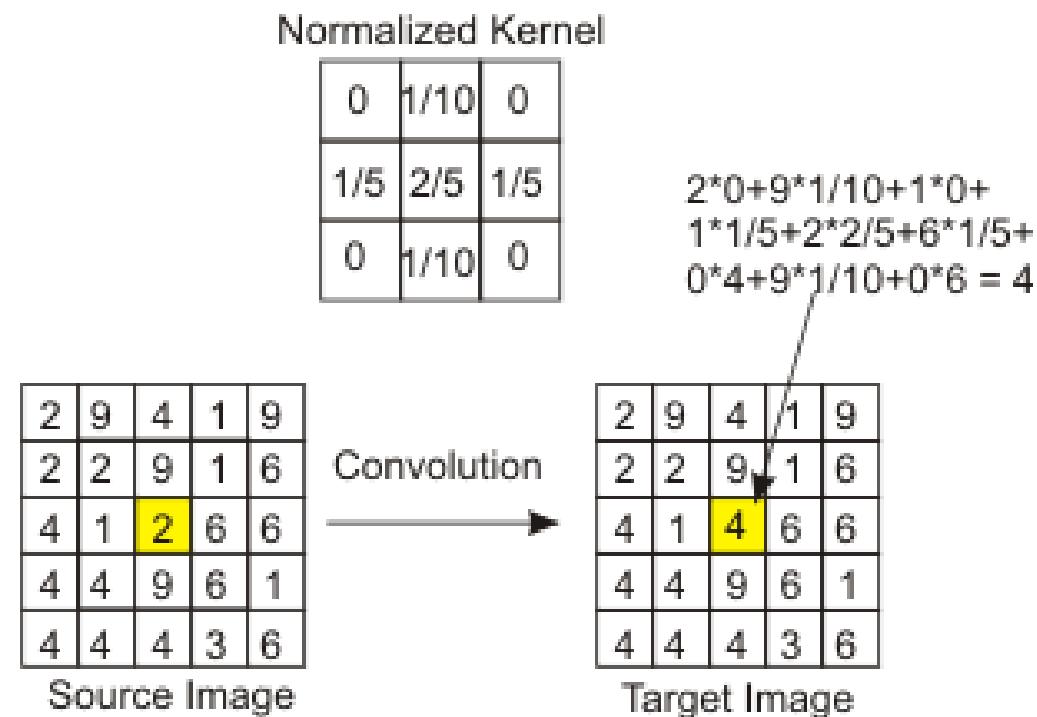


Exemplu egalizare histogramă



Filtre de conoluție

Calculează valoarea fiecărui pixel în funcție de valorile pixelilor alăturați



Filtre de conoluție

Algoritm general:

pentru fiecare valoare $v(x,y)$ din matricea originală

acumulator = 0

pentru fiecare valoare $k(i,j)$ din matricea de conoluție

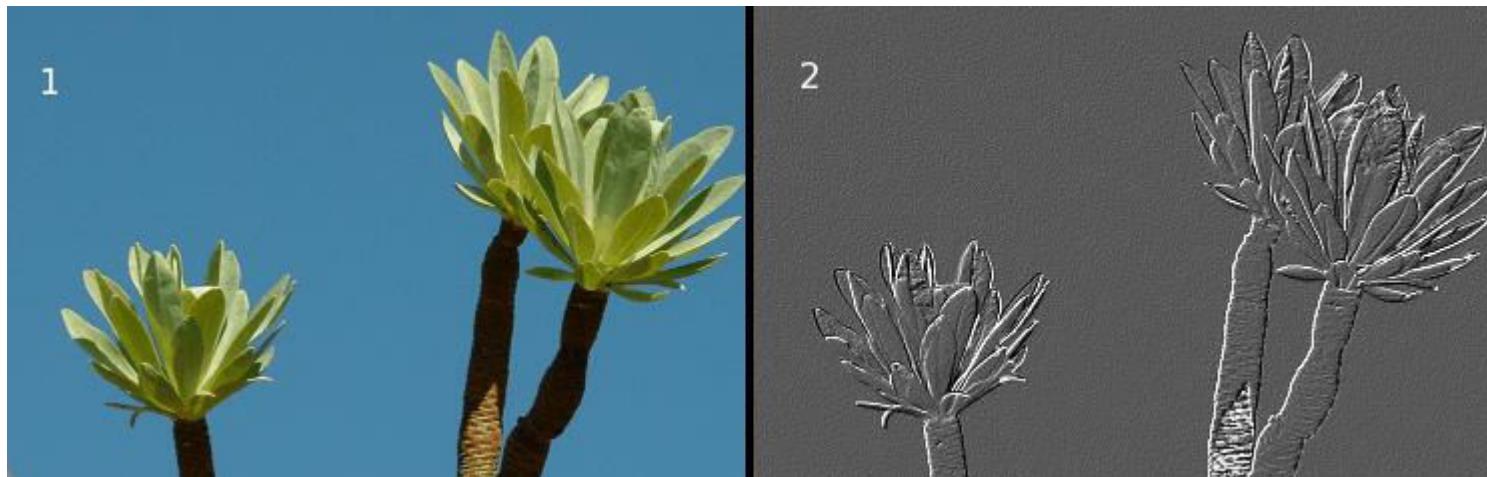
*acumulator = accumulator + $v(x+i,y+j)*k(i,j)$*

$v'(x,y) = accumulator$ (trunchiat la 0..255)

Observații:

- se aplică pe fiecare canal de culoare în parte
- tratare specială pentru pixelii din margine

Filtre de conoluție – Exemplu emboss



$$\begin{pmatrix} -1 & 0 & -1 \\ 0 & 4 & 0 \\ -1 & 0 & -1 \end{pmatrix} + 127$$

Filtre de conoluție – Alte exemple

Gaussian Blur

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}_{/16}$$

Sharpen

$$\begin{pmatrix} 0 & -2 & 0 \\ -2 & 11 & -2 \\ 0 & -2 & 0 \end{pmatrix}_{/3}$$

Edge Detection

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}_{+127}$$

Formate de stocare

BMP (Microsoft Windows Bitmap)

- Formatul standard de stocare pe platforma Microsoft Windows
- Suportă date necomprimate sau comprimate folosind algoritmul RLE
- Monocromă sau în culori pe 4, 8, 16, 24 sau 32 de biți
- Suportă palete de culori

JPEG (Joint Photographic Experts Group)

- Stocare comprimată cu pierdere de informație conform standardului JPEG
- Rate de compresie diferite selectable de către utilizator
- Utilizat pentru imagini fotografice (cu gradații fine de culoare)
- Nu este potrivit pentru
 - text, linii sau alte imagini care prezintă un contrast foarte mare
 - Editări multiple (se pierde calitate la fiecare etapă de compresie / decompresie)

Formate de stocare

GIF (Graphics Interchange Format)

- Folosit în special pentru transferul imaginilor de maxim maxim 64K x 64K
- Pretabil pentru diagrame, text logo-uri (contrast puternic și număr limitat de culori)
- Suportă maxim 256 culori prin intermediul unei palete de culori
- Poate stoca mai multe cadre (pentru animație)
- Algoritm de compresie fără pierdere de informație Lempel-Ziv-Welch (LZW)

TIFF (Tag Image File Format)

- Format portabil și extensibil utilizat în special pentru imagini scanate
- Suportă stocarea mai multor imagini într-un singur fișier
- Suportă mai mulți algoritmi de compresie (RLE, LZW sau JPEG)

Alte formate: ICO - Icon Resource File, PNG - Portable Network Graphics, DIB - Device Independent Bitmap, PCX - PC PaintBrush File Format

Compresia Run-length encoding (RLE)

Sevențele de valori identice consecutive sunt înlocuite cu perechi de forma
(valoare, număr apariții)

Exemplu:

AAAAAAAAAAAAAAABBBBBBBBBAAAAAAAAAAABB BBBBCCC

14A10B13A6B3C

Caracteristici

- Rată mică de compresie
- Se pretează pentru imagini cu zone mari de aceeași culoare
- Utilizat în special pentru fișiere BMP cu paletă de culori

Compresia Lempel–Ziv–Welch (LZW)

Algoritm de compresie universal bazat pe dicționar

Descriere compresie:

1. Se construiește dicționarul inițial (toate sirurile de lungime 1)
2. Se caută cel mai lung sir W din dicționar care se potrivește cu sirul de la intrare
3. Se elimină W din sirul de intrare
4. Se adaugă W urmat de următorul caracter în dicționar
5. Se continuă cu pasul 2

Decompresie: se parurge sirul codificat și se reconstruiește dinamic dicționarul

Variante: coduri de lungime variabilă, cod pentru reinițializare dicționar

Utilizat pentru fișiere de tip GIF, PNG

Compresia Huffman

Algoritm universal de compresie

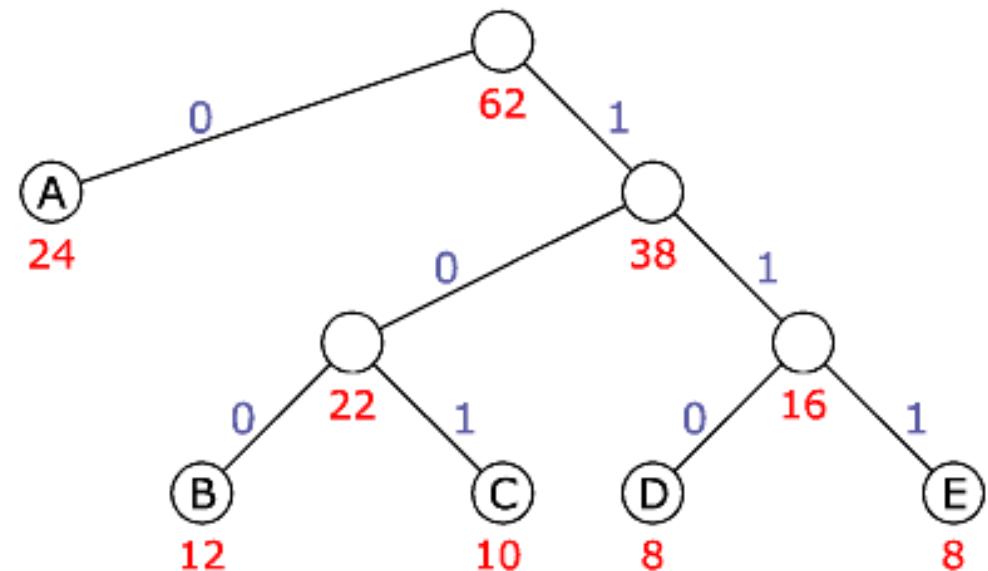
Codificare optimă de lungime variabilă pentru fiecare simbol în funcție de frecvența de apariție

Datele salvate:

- dicționarul
- datele originale recodificate

Decodificare:

- translație simbol cu simbol pe baza dicționarului salvat



Compresia JPEG

Compresie specializată cu pierdere de informație pentru imagine raster

Rezultate foarte bune pentru fotografii (variații fine de luminozitate și culoare)

Tipuri de compresie:

- **secvențial** – codaj bazat pe transformarea cosinus discretă cu blocurile procesate în ordinea apariției
- **progresiv** – codaj bazat pe transformarea cosinus discretă cu blocurile procesate prin mai multe treceri asupra imaginii
- **progresiv fără pierdere** – folosește doar algoritmi de compresie fără pierdere de informație
- **progresiv ierarhic** – codifică imaginea la rezoluții din ce în ce mai mari

Etapele compresiei JPEG

Etapele de **compresie** JPEG File Interchange Format (JFIF)

1. Translatarea modului de culoare din RGB în $Y'C_BC_R$
2. Reducerea rezoluției pentru componente C_B și C_R
3. Imaginea se descompune în blocuri de dimensiune 8x8 pixeli
4. Se aplică transformata cosinus discretă pe fiecare bloc în parte
5. Aplicarea matricei de cuantizare (pierdere de informație)
6. Blocurile rezultate în urma cuantizării sunt comprimate folosind RLE și Huffman

Decodificare: se aplică pașii în ordine inversă

JPEG -Transformata cosinus discretă

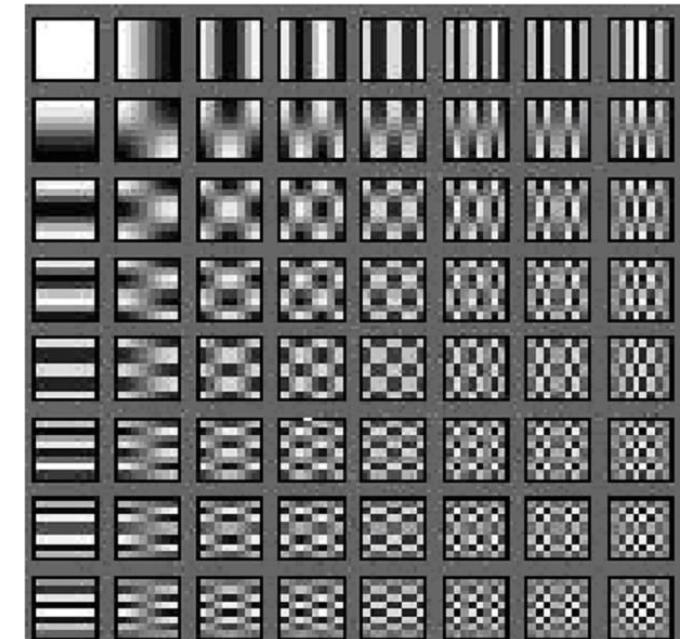
$$\text{DCT: } G_{ij} = \frac{1}{4} C_i C_j \sum_{x=0}^7 \sum_{y=0}^7 p_{xy} \cos\left(\frac{(2y+1)j\pi}{16}\right) \cos\left(\frac{(2x+1)i\pi}{16}\right)$$

$$\text{IDCT: } p_{xy} = \frac{1}{4} \sum_{i=0}^7 \sum_{j=0}^7 C_i C_j G_{ij} \cos\left(\frac{(2y+1)j\pi}{16}\right) \cos\left(\frac{(2x+1)i\pi}{16}\right)$$

unde

$$C_i = C_j = \frac{1}{\sqrt{2}}, \quad \text{dacă } i = j = 0$$

$$C_i = C_j = 1, \quad \text{în rest}$$



JPEG -Transformata cosinus discretă

139	144	149	153	155	155	155	155		235.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3
144	151	153	156	159	156	156	156		-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2
150	155	160	163	158	156	156	156		-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1
159	161	162	160	160	159	159	159		-7.1	-1.9	0.2	1.5	0.9	-0.1	0.0	0.3
159	160	161	162	162	155	155	155		-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3
161	161	161	161	160	157	157	157		1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0
162	162	161	163	162	157	157	157		-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8
162	162	161	161	163	158	158	158		-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4

JPEG - Cuantizare

235.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3
-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2
-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1
-7.1	-1.9	0.2	1.5	0.9	-0.1	0.0	0.3
-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3
1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0
-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8
-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

$$Cq(u,v) = \text{round}[C(u,v)/N(u,v)]$$

$$\begin{aligned} 235.6/16 &\rightarrow 15 \\ -22.6/12 &\rightarrow -2 \end{aligned}$$

15	0	-1	0	0	0	0	0
-2	-1	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0
-1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Exemplu: <https://squoosh.app/>

Canvas – Salvare Imagine

Se utilizează funcția:

canvasElement.toDataURL(type, encoderOptions)

- *type*: formatul imaginii (**image/png**, image/jpeg, ...)
- *encoderOptions*: opțiuni dependente de tipul imaginii și browser (exemplu: un număr între 0 și 1 reprezentând calitatea pentru jpeg)

Șirul de caractere rezultat poate fi utilizat ca URL pentru elemente de tip IMG sau A:

```
var img = document.createElement('img');
img.src = canvas.toDataURL('image/jpeg', 0.4)
document.body.append(img);
```

```
let a = document.createElement('a');
a.download = 'imagine.jpg';
a.href = canvas.toDataURL('image/jpeg', 0.1);
a.click();
```

3. Animăție

Modificarea rapidă a imaginii vizualizate prin modificarea poziției, formei sau dimensiunii unui obiect din imagine

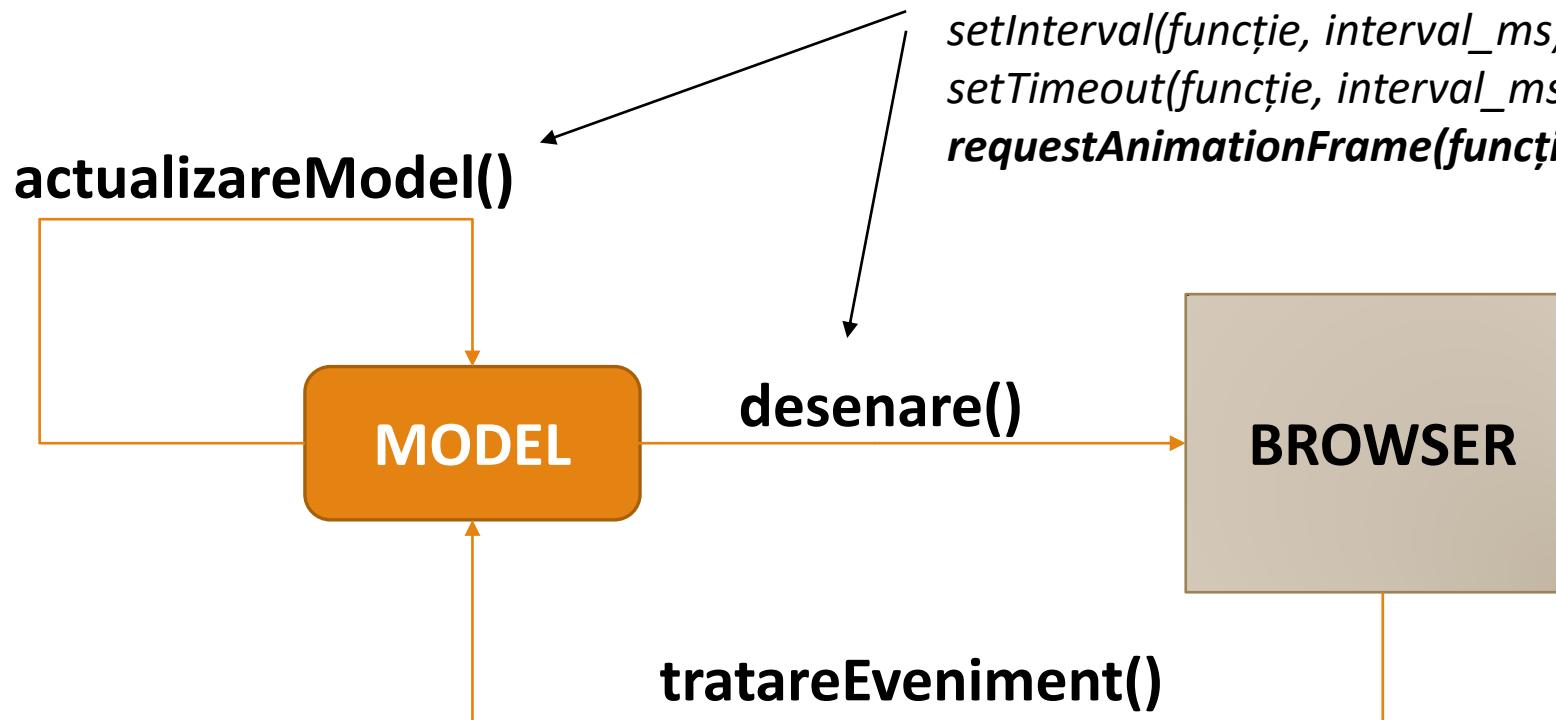
Stocarea numerică a animației presupune reținerea elementelor independente ce compun mișcarea în raport cu factorul timp.

Crearea iluziei de mișcare se realizează prin afișarea rapidă de imagini statice ușor modificate

Tehnici principale:

- Tehnica filmului
- Cadre cheie
- Schimbarea culorii

Animație - JavaScript



4. Grafică vectorială

Bazată pe descrierea matematică a obiectelor componente ale imaginii

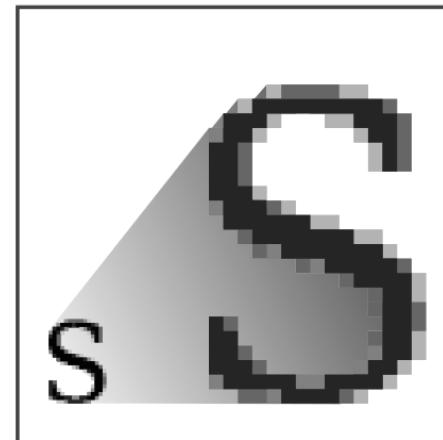
Avantaje:

- Menține semantica – editare la nivel de obiect grafic
- Dimensiune redusă
- Independente de scara de vizualizare

Dezavantaj

- Nu poate reprezenta fidel orice fel de informație

GIMP



BITMAP
.jpeg .gif .png

INKSCAPE



OUTLINE
.svg

Formate de reprezentare și stocare

SVG (Scalable Vector Graphics)

- Format generic bazat pe XML pentru reprezentări vectoriale 2D
- Suportă animație și interactivitate

DXF (Drawing Exchange Format)

- formatul vectorial lansat de firma Autodesk pentru produsul software AutoCAD

EPS (Encapsulated Post Script)

- formatul firmei Adobe pentru imagini vectoriale
- se bazează pe un limbaj de descriere numit Post Script

SHP (Shapefile)

- formatul firmei ESRI pentru descrierea datelor spațiale de tip: punct, polilinie și poligon
- utilizat la reprezentarea elementelor geografice în sisteme de tip GIS

SVG – Scalable Vector Graphics

Poate fi:

- inclus direct într-o pagină HTML
- controlat prin intermediul CSS și JavaScript

Exemplu:

```
<!DOCTYPE html>
<html>
<body>
    <svg width="400" height="180">
        <rect x="50" y="20" width="150" height="150"
style="fill:red;stroke:black;stroke-width:5;opacity:0.5"/>
    </svg>
</body>
```

Sistem de coordonate implicit – similar *canvas*

SVG – Elemente de bază

Linii

```
<line x1="start-x" y1="start-y" x2="end-x" y2="end-y">
```

Dreptunghiuri

```
<rect x="start-x" y="start-y" width="width" height="height"/>
```

Cercuri

```
<circle cx="center-x" cy="center-y" r="radius"/>
```

SVG – Elemente de bază

Elipse

```
<ellipse cx="center-x" cy="center-y" rx="radius-x" ry="radius-y"/>
```

Poligoane

```
<polygon points="x1,y1 x2,y2 ..."/>
```

Poli-linii

```
<polyline points="x1,y1 x2,y2 ..."/>
```

Text

```
<text x="start-x" y="start-y">conținut</text>
```

SVG - Atribute

Setate prin intermediul CSS

Atribute de bază

- stroke – culoare linie
- stroke-opacity – opacitate linie
- stroke-width – dimensiune linie

- fill – culoare suprafață
- fill-opacity – opacitate suprafață

SVG – Grupare și reutilizare

Grupare elemente

```
<g id="id_grup">... <!-- elemente --> </g>
```

Definire elemente fără afișare

```
<defs>... <!-- definire grupuri --> </defs>
```

Reutilizare

```
<use xlink:href="#id_grup" x="30" y="14"/>
```

SVG – Transformări

Se aplică pentru un element prin intermediul atributului ***transform***

Modifică întreg sistemul de coordinate

Transformări disponibile:

- scale(sx[, sy])
- translate(tx, ty)
- rotate(unghi[, cx, cy])
- skewX(unghi) / skewY(unghi)

SVG - Efecte

Se aplică pentru un element prin intermediul atributului ***transform***

Sintaxa:

```
<filter id="F">
  <anyParticularPrimitive1>
  <anyParticularPrimitive2>
  ...
  <anyParticularPrimitiveN>
</filter>
<anyParticularSVGObjectOrGroup filter="url(#F)"/>
```

Exemplu:

```
<filter id="filtru">
  <feConvolveMatrix kernelMatrix="3 3 3      0 0 0      -3 -3 -3" />
  <feGaussianBlur stdDeviation="3" />
</filter>
<circle cx="10" cy="20" r="17" filter="url(#filtru)"></use>
```

Lista filtre: https://developer.mozilla.org/en-US/docs/Web/SVG/Element#Filter_primitive_elements

SVG – Manipulare din JavaScript

Similar cu manipularea elementelor HTML

Particularități

- la construirea elementului trebuie specificat namespace-ul pentru SVG:
 - `document.createElementNS("http://www.w3.org/2000/svg", „TAG_SVG")`
- în jQuery se utilizează funcția `attr` pentru modificarea atributelor (în loc de `.width`, `.height`, ...)
- Folosind JavaScript simplu:
 - `elem.setAttributeNS(null, 'nume', 'valoare');`
 - `val = elem.getAttributeNS(null, 'nume');`

Exemplu:

```
$(document.createElementNS("http://www.w3.org/2000/svg", "rect"))
```

```
.attr({x:160, y:160, width:12, height:12})
```

```
.appendTo($("#desen"));
```

III. Sunet

1. Notiuni generale
2. Numerizarea sunetului
3. Formate audio
4. Compresia sunetului
5. Sunetul în context Web

1. Noțiuni generale

Sunetul este o vibrație propagată printr-un mediu material sub forma unei unde mecanice.

- Din punct de vedere fiziologic: senzația produsă asupra organului auditiv de către vibrațiile materiale ale corpurilor și transmise pe calea undelor acustice;
- Urechea umană percepse vibrații în intervalul 20-20000Hz
- Zgomot: caz particular de sunet caracterizat prin lipsa încărcăturii informaționale

Redare / receptare sunet

- Prin intermediul difuzorului și microfonului
- Ambele fac conversie semnal electric – vibrație mecanică
- Folosesc principiile inducției electomagnetice

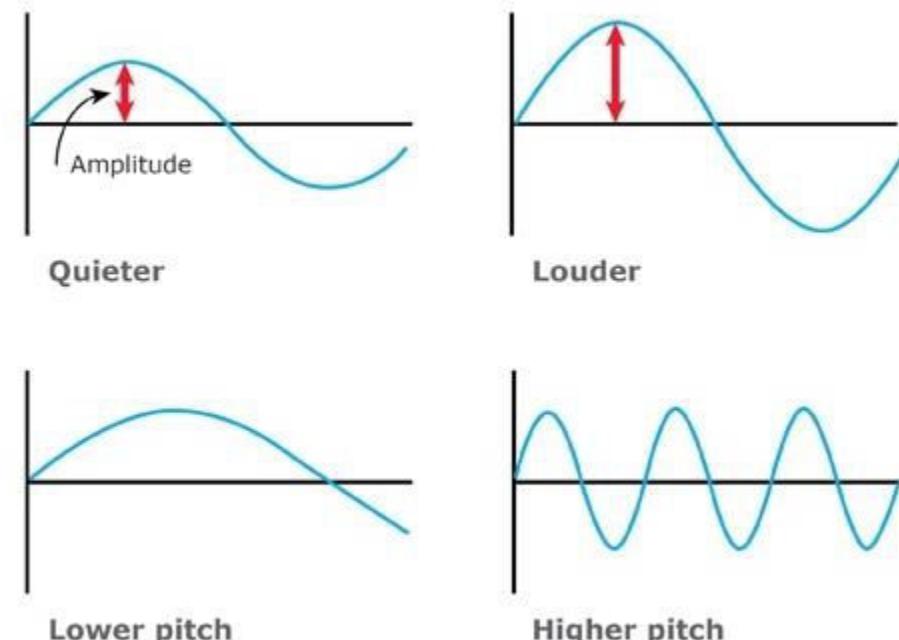
Reprezentare și caracteristici

Reprezentare:

- Axa X: timp
- Axa Y: presiune (0 – presiunea aerului în repaus)

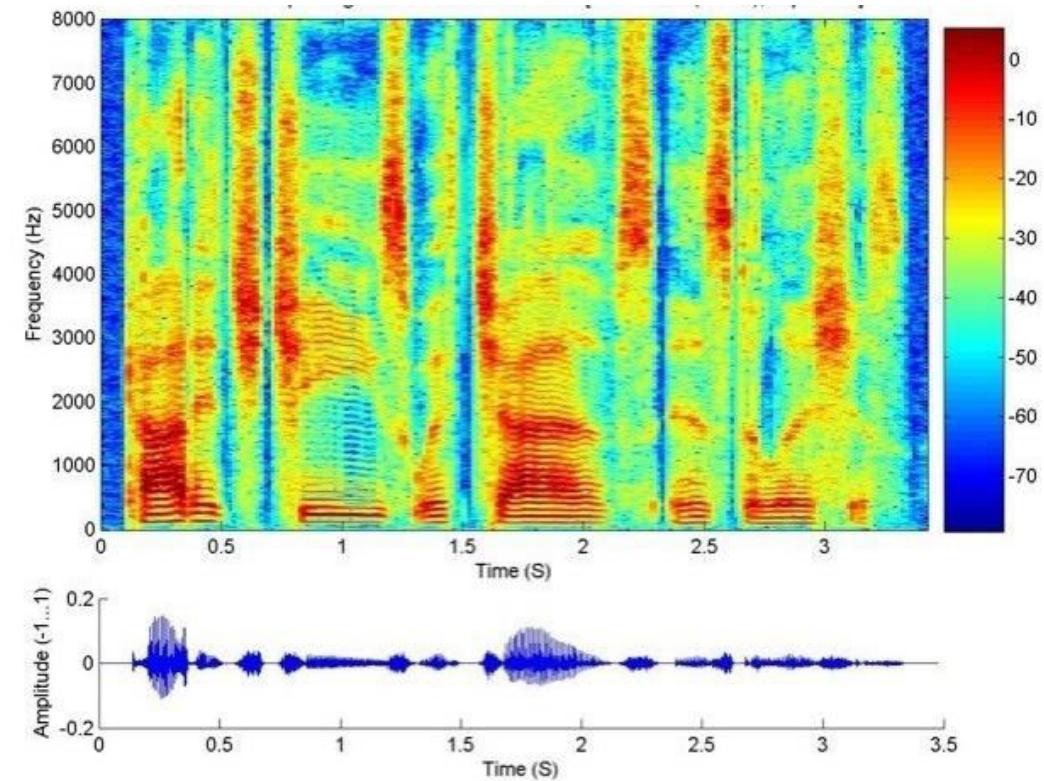
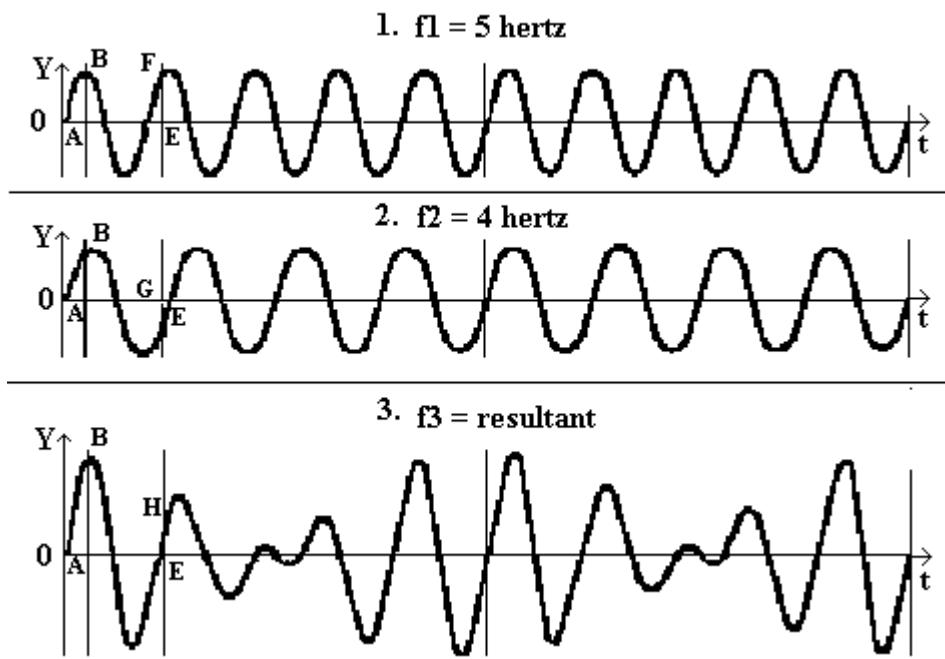
Amplitudine: măsoară dimensiunea vibrației / volumul sunetului

Frecvență: măsoară viteza vibrației / tonul sunetului



Compunere / descompunere

26.8.4.4



2. Numerizarea sunetului

Presupune stocarea și prelucrarea sunetului în format digital

Etape:

- Convertirea sunetului în semnal electric
- Eșantionarea și cuantificarea semnalului
- Stocarea informației numerice pe un suport de memorie externă conform unui format

Avantaje

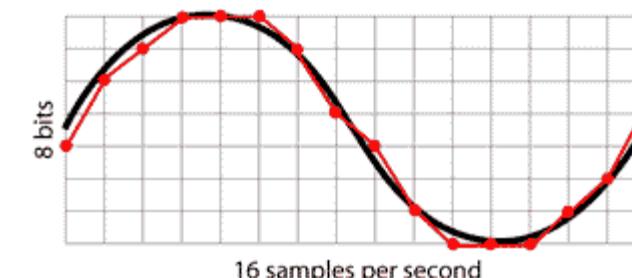
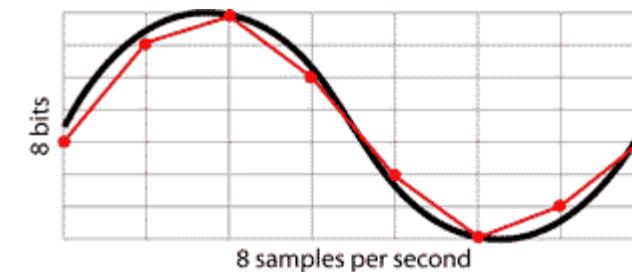
- Stocare mai ușoară
- Permite analiza și procesarea numerică a sunetului
- Nu se degradează în timp sau la copieri repetate

Eșantionare

Prin eșantionare se înțelege procesul de segmentare, cu o perioadă fixă, a semnalului audio analog.

Frecvența de eșantionare – rezoluția orizontală

- Se determină pe baza teoremei lui Nyquist (minim dublul frecvenței maxime a sunetului)
- Rate de eșantionare uzuale:
 - 8 kHz – semnal telefonic
 - 11 kHz – radio AM
 - 22 kHz – radio FM
 - 44 kHz – audio CD



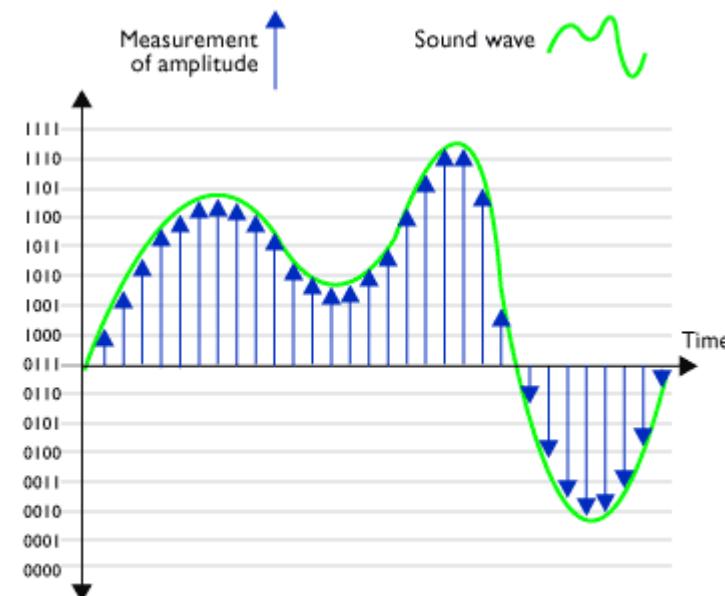
Cuantificare

Cuantificarea presupune asocierea unei valori numerice corespunzătoare amplitudinii semnalului pentru fiecare interval de timp.

Calitatea este influențată de numărul de biți alocați pentru fiecare eşantion (uzual 8 sau 16 biți pentru stocare și 16 – 32 pentru procesare)

Redarea sunetului digital:

- Se reconstruiește sinusoida originală prin interpolarea valorilor numerice stocate
- Prin intermediul unui convertor digital - analog



Each measurement is assigned a number (byte) according to its amplitude. The end result is a file comprising a string of bytes, eg ...
1001 1110 0001 1010 0111 0100 1111 1101 etc

3. Formate audio

WAVE – formatul standard de fișier audio pentru Microsoft și IBM; conține sunet în reprezentare PCM necomprimat;

AIFF (Audio Interchange File Format) – formatul standard pentru audio digital utilizat pe platformele Apple (variante: necomprimat / comprimat);

MPEG (Moving Picture Experts Group) Audio - format standard pentru sunetul digital comprimat; parte a standardului MPEG de codificare a semnalului audio-video; cea mai cunoscută variantă a lui este MP3

OGG/VORBIS – algoritm de compresie perceptuală, similar MP3, dar cu implementare open source; folosește formatul de fișier OGG

4. Compresia sunetului

Cel mai utilizat algoritm de compresie: **MPEG-1 sau 2 Audio Layer III (MP3)**

Folosește codificare perceptuală

- Elimină din rezultat sunetele care nu pot fi percepute de către urechea umană

Sunetele imperceptibile sunt eliminate pe baza unui model psihooacustic care exploatează fenomenele de:

- Mascare a frecvențelor
- Mascare temporală

Compresia sunetului - mascarea

Mascarea frecvențelor

- Sunt eliminate sunetele cu frecvență mai mare de 16-18 KHz
- Sunt eliminate sunetele de intensitate scăzută, care apar concomitent cu sunete de intensitate înaltă, dacă sunt în benzi de frecvență alăturate (cele cu intensitate scăzută sunt măcate de cele cu intensitate înaltă)

Mascarea temporală

- Se elimină sunetele de intensitate mică care urmează după sunete de intensitate puternică în cadrul unui interval de timp
- Sunetele de intensitate mică nu pot fi percepute după sunete de intensitate puternică datorită inerției timpanului

Compresia MP3 - Etape

1. Utilizarea de filtre pentru separarea sunetului în 32 sub-benzi de frecvență
2. Transformări
 - FFT: se aplică modelul psihoaesthetic pentru determinarea factorului de scalare
 - DCT: proces de cuantizare cu factorul de scalare determinat anterior
3. Codificare Huffman pentru valorile cuantizate
4. Componerea fluxului final de biți

5. Sunetul în context Web

La nivel de HTML – tag `<audio>`

Exemplu:

```
<audio controls="controls">
    <source src="test.wav" type="audio/wav">
    <source src="test.mp3" type="audio/mpeg">
</audio>
```

Formate suportate:

- mp3 – type = audio/mpeg
- wav – type = audio/wav
- ogg – type = audio/ogg

Audio - atributे

Elementul <audio>:

autoplay (bool) – redarea automată a sunetului

controls (string) – controalele de redare sunt afișate dacă atributul este prezent

loop (bool) – permite redarea continuă a sunetului

src (string) – permite specificarea sursei fără utilizarea de tag-uri de tip *source*

Elementul <source>

src (string) – adresa (URL) fișierului audio

type (string) – tipul MIME pentru fișierul audio

Audio – obiectul *HTMLMediaElement*

Proprietăți:

currentSrc – URL-ul absolut al fișierului redat

currentTime – poziția (în secunde) în cadrul fișierului (poate fi modificată)

duration – durata totală a fișierului audio (în secunde)

ended – boolean setat pe *true* la terminarea redării

error – ultima eroare (obiect *MediaError*) sau *null* dacă nu a apărut nici o eroare

paused – boolean setat pe *false* la oprirea redării

readyState – indică starea curentă a elementului

volume – permite citirea / modificarea volumului

Audio – obiectul *HTMLMediaElement*

Metode:

canPlayType(type) – permite aplicației să determine dacă browser-ul curent suportă un anumit tip de fișier audio

load() – pornește procesul de descărcare a fișierului audio de pe server; este obligatoriu să fie apelat înainte de începerea redării folosind metoda *play()*

pause() – oprește redarea (cu păstrarea poziției curente)

play() – pornește redarea de la poziția curentă

Audio – obiectul *HTMLMediaElement*

Evenimente:

canplay – a fost încărcată o parte din fișier și poate fi pornită redarea

ended – redarea s-a terminat

pause – redarea a fost oprită

play – redarea a început

volumechange – modificare de volum

waiting – operația curentă este suspendată pentru a încărca date de pe server

Web Audio API

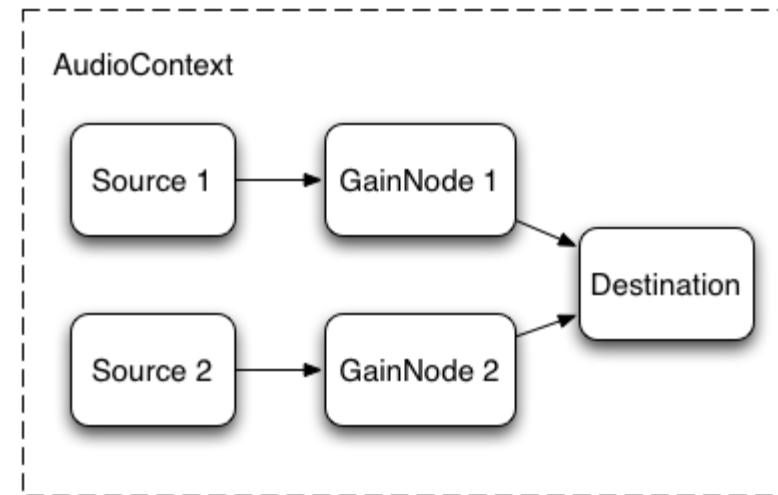
Permite generarea și procesarea de sunet în cadrul unui browser web.

Operațiile:

- se desfășoară în interiorul unui **context audio**
- sunt specificate prin intermediul unui **graf de rutare**

Tipuri de noduri:

- Sursă: preiau datele audio dintr-un tag audio sau generează sunete
- Procesare: prelucrează datele primite la intrare
- Destinație: nodul final al graficului



Web Audio API

AudioContext – reprezintă un graf de procesare audio

- Proprietăți: `.currentTime` (poziția curentă în secunde), `.destination` (referința la nodul destinație)
- Metode:
 - `createMediaElementSource(audioElement)`: construiește un nod sursă pe baza unui element `<audio>`
 - `createOscillator()`: construiește un nod sursă pentru generarea de sunete
 - `createGain()`: construiește un nod de procesare pentru ajustarea volumului
 - `createAnalyser()`: permite analiza sunetului (descompunere Fourier)
 - `createScriptProcessor(bufferSize)`: construiește un nod pentru procesare JavaScript
 - `.close()`:

AudioNode – reprezintă un nod din cadrul grafului (din care sunt moștenite celelalte noduri)

- `.connect(node)`: conectează ieșirea nodului curent la intrarea nodului primit ca parametru
- Diverse metode și proprietăți în funcție de tipul nodului

IV. Video

1. Noțiuni generale
2. Compresia video
3. Video în context web

1. Noțiuni generale

Video digital – cuprinde totalitatea tehniciilor de captură, procesare și stocare a imaginilor în mișcare (precum și a sunetului asociat) prin intermediul unui dispozitiv de calcul.

Avantaje video digital:

- Poate fi procesat prin intermediul calculatorului
- Păstrare în timp și rezistență la copieri repetitive
- Poate fi transmis la distanță

Caracteristici video

Rezoluția

Spațiul de culoare și numărul de biți per pixel

Numărul de cadre pe secundă

Modul de afișare (întrețesut sau progresiv)

Calitatea compresiei

Formate video

Container – specifică structura de stocare a componentelor video (imagine + audio) și a datelor asociate (metadate, subtitrări, ...)

- Advanced Systems Format – **ASF**: container dezvoltat de Microsoft care poate conține fluxuri codate cu orice codec (Extensii: .ASF, .WMA, .WMV)
- Audio Video Interleave – **AVI**: container mai vechi dezvoltat de Microsoft pe baza Resource Interchange File Format – RIFF (stochează datele în secțiuni identificate prin markere FourCC)
- **MP4** – MPEG-4 Part 14: dezvoltat de către Motion Pictures Expert Group și utilizat inițial de către QuickTime (video H.264, audio AAC)
- **AVCHD** – format utilizat în special de către camerele video (video H.264 AVC și sunet AC3 sau PCM)
- **Matroska / OGG**: formate deschise; pot conține mai multe fluxuri audio / video

Formate video

Codec – specifică modalitatea de compresie / decompresie pentru un flux video / audio în cadrul unui container

- **H.264 / MPEG-4 AVC** – cel mai popular (utilizat pentru Web, BluRay, camere video)
- **H.262 / MPEG-2** – formatul standard pentru DVD
- **Windows Media Video** – format dezvoltat de către Microsoft
- **MJPEG (Motion JPEG)** – format mai vechi bazat pe compresia JPEG

2. Compresia video

Se bazează pe reducerea redundanței din cadrul fluxului video

Redundanță spațială (intra-cadru)

- tipul de redundanță identificat și eliminat de algoritmii de compresie a imaginilor

Redundanță temporală (inter-cadru)

- redundanță identificată între două cadre consecutive (de exemplu, prin compararea a două cadre se observă că majoritatea pixelilor își păstrează valoarea)

Compresia MPEG

Algoritm de compresie video

- Hibrid
 - Transformata Cosinus Discretă – similar JPEG pentru reducerea redundanței spațiale
 - Codaj Huffman – pentru comprimarea coeficienților TCD
 - Codificarea mișcării – pentru reducerea redundanței temporale
 - Codaj RLE
- Asimetric
 - Timpul de codare este mult mai mare decât cel de decodare

Etapele de compresie

Împărțirea imaginii în blocuri

- 16x16 luminanță
- 8x8 crominanță (culoare)

Compresie pe baza DCT pentru reducere spațială

Aplicarea tehniciilor de compensare a mișcării pentru reducere temporală

Faza finală de codare pe două dimensiuni folosind Run Length Encoding

Tipuri de cadre

<I> Intra-picture/frame/image

- Cadrele cheie
- Necesare pentru căutare și poziționare
- Compresie moderată

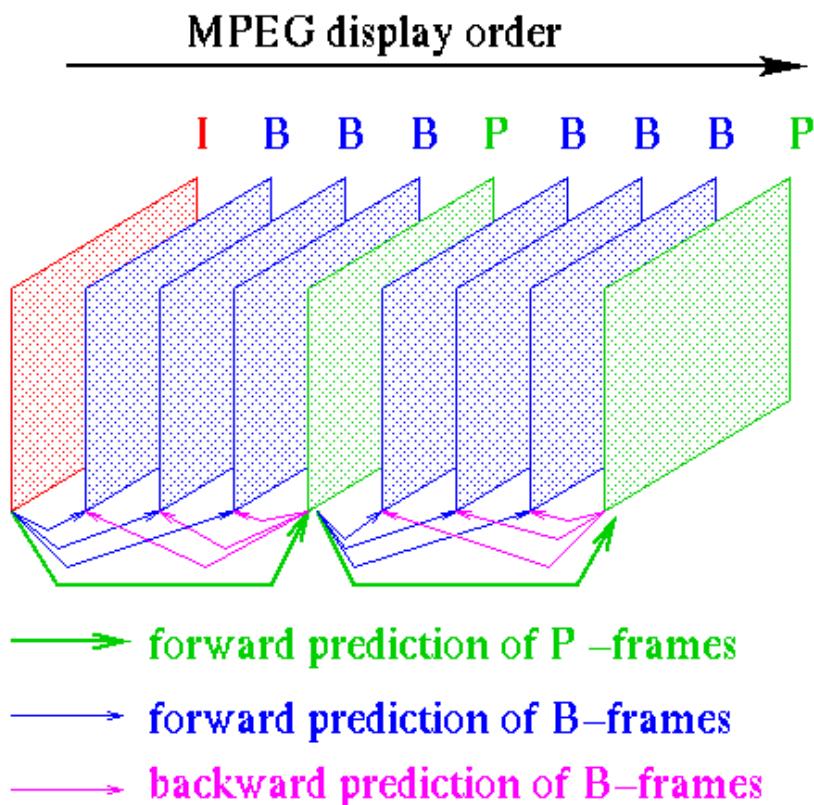
<P> Predicted pictures

- Codate cu referință la un cadru anterior
- Folosite ca referință pentru cadre ulterioare

* Bi-directional prediction (interpolated pictures)*

- Necesită cadre anterioare și viitoare pentru refacere
- Compresie mare

Tipuri de cadre



4. Video în context Web

La nivel de HTML – tag <video>

Exemplu:

```
<video controls="controls">
    <source src="test.webm" type="video/webm">
    <source src="test.mp4" type="video/mp4">
</video>
```

Formate suportate:

- webm – type = video/webm
- mp4 – type = video/mp4
- .ogg – type = video/ogg

Atribute / manipulare din JavaScript

Similar cu elementul audio

- Atribute: *autoplay, controls, src, volume, ...*

Reprezentat la nivel DOM prin obiecte de tip `HTMLMediaElement`

- Proprietăți: `currentSrc`, `currentTime`, `duration`, `ended`, `error`, `paused`, `readyState`, `volume`
- Metode: `canPlayType`, `load`, `pause`, `play`
- Evenimente: `canplay`, `ended`, `pause`, `play`, `volumchange`, `waiting`

Exemplu: Încărcare dinamică video

```
var v = $("<video></video>")
    .attr({
        "controls": "hidden",
        "autoplay": "autoplay",
        "src": "media/movie.mp4"
    })
    .load()
    .appendTo($(".body"));

// modificați sursă
v[0].src = "media/test.mp4";
v[0].load();
v[0].play();
```

Exemple: Procesare cadru - 1

```
var video = $("#video")[0];
var canvas = $("#canvasProcesare")[0];
var context = canvas.getContext("2d");
function procesareCadru() {
    //...
    requestAnimationFrame(procesareCadru);
};
requestAnimationFrame(procesareCadru);
```

Exemplu: Procesare cadru - 2

```
var W = canvas.width = video.clientWidth;  
var H = canvas.height = video.clientHeight;  
  
context.drawImage(video, 0, 0, W, H);  
var imageData = context.getImageData(0, 0, W, H);  
  
for (var y = 0; y < H; y++) {  
    for (var x = 0; x < W; x++) {  
        var i = (y * W * 4) + x * 4;  
        // modificați valori imageData.data[i+...]  
    }  
}  
context.putImageData(imageData, 0, 0);  
// alte operații de desenare pe canvas
```

Exemple: Playlist simplu

```
$(function () {
    var lista = ["movie.mp4", "v2.mp4"];
    var index = 0;

    var video = $("#myVideo");

    video.on("ended", function () {
        index = index + 1;
        if (index >= lista.length) {
            index = 0;
        }
        video[0].src = lista[index];
        video[0].load();
        video[0].play();
    });
});
```