ИКТ в НОС

Потребителски обекти

Тема №12

Обекти и стилове

Създаване на графични обекти

Създаване на обект

- С конструктор **new Обект(…)**
- С вградена функция **обект()**

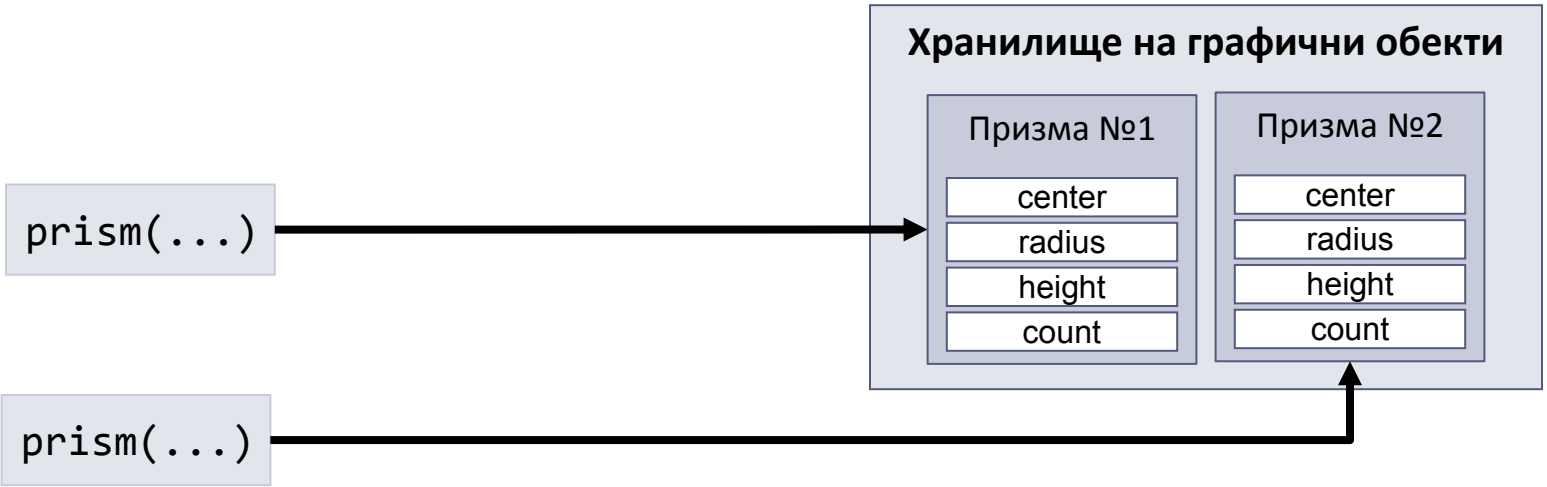
Зад сцената

- Създаденият обект автоматично се съхранява
- Използва се при прерисуване на екрана

Имена на графични обекти

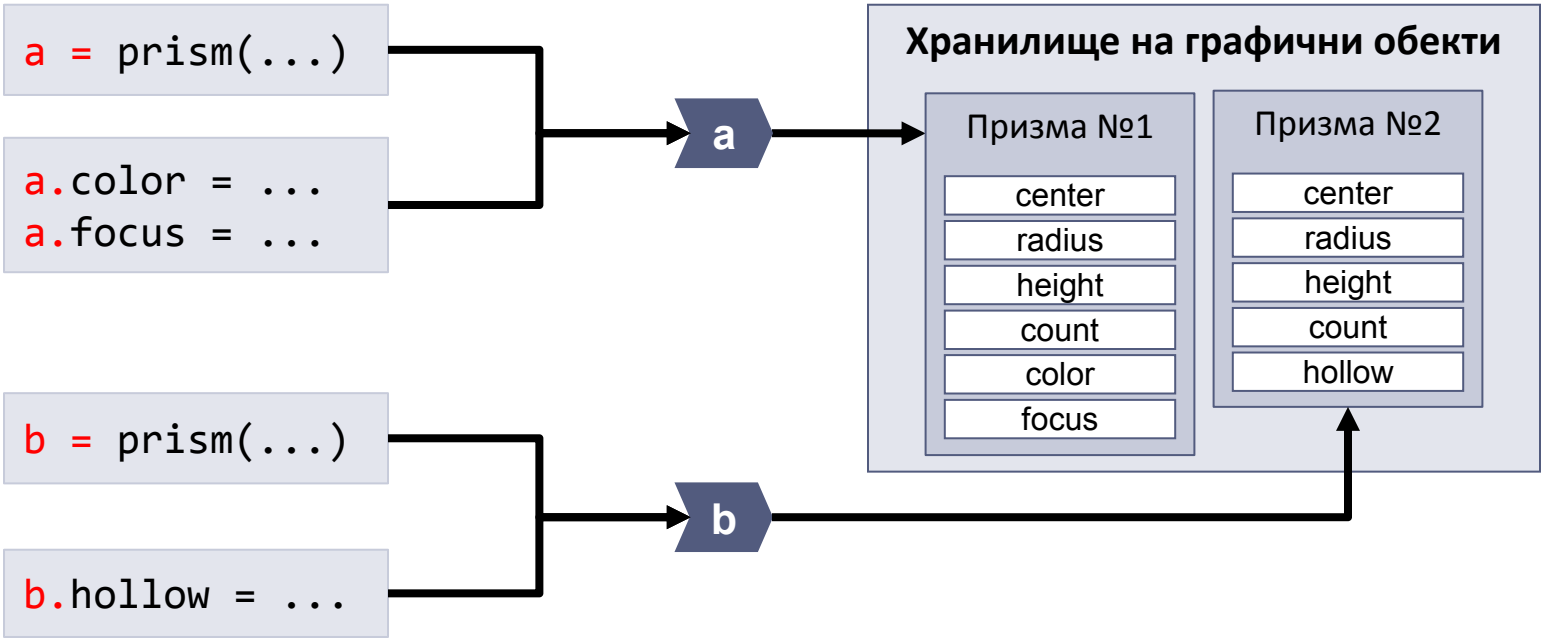
Анонимни

- Създават се без имена
- Подходящо за обекти, които се създават, но не се променят



Обекти с имена

- При създаването им се присвояват и на променлива
- Свойствата на обекта се променят през променливата



Множество от свойства

Наблюдение от практиката

- Обект се настройва с няколко свойства
- Много обекти имат еднакви свойства

Желание

- Да групираме няколко свойства в стил
- Да прилагаме лесно стил над много обекти

Реализация на стилове

До момента

- Създаване на обекти с имена
- Променяне на свойствата едно по едно
- Една променлива може да се използва последователно за много обекти

Неудобства

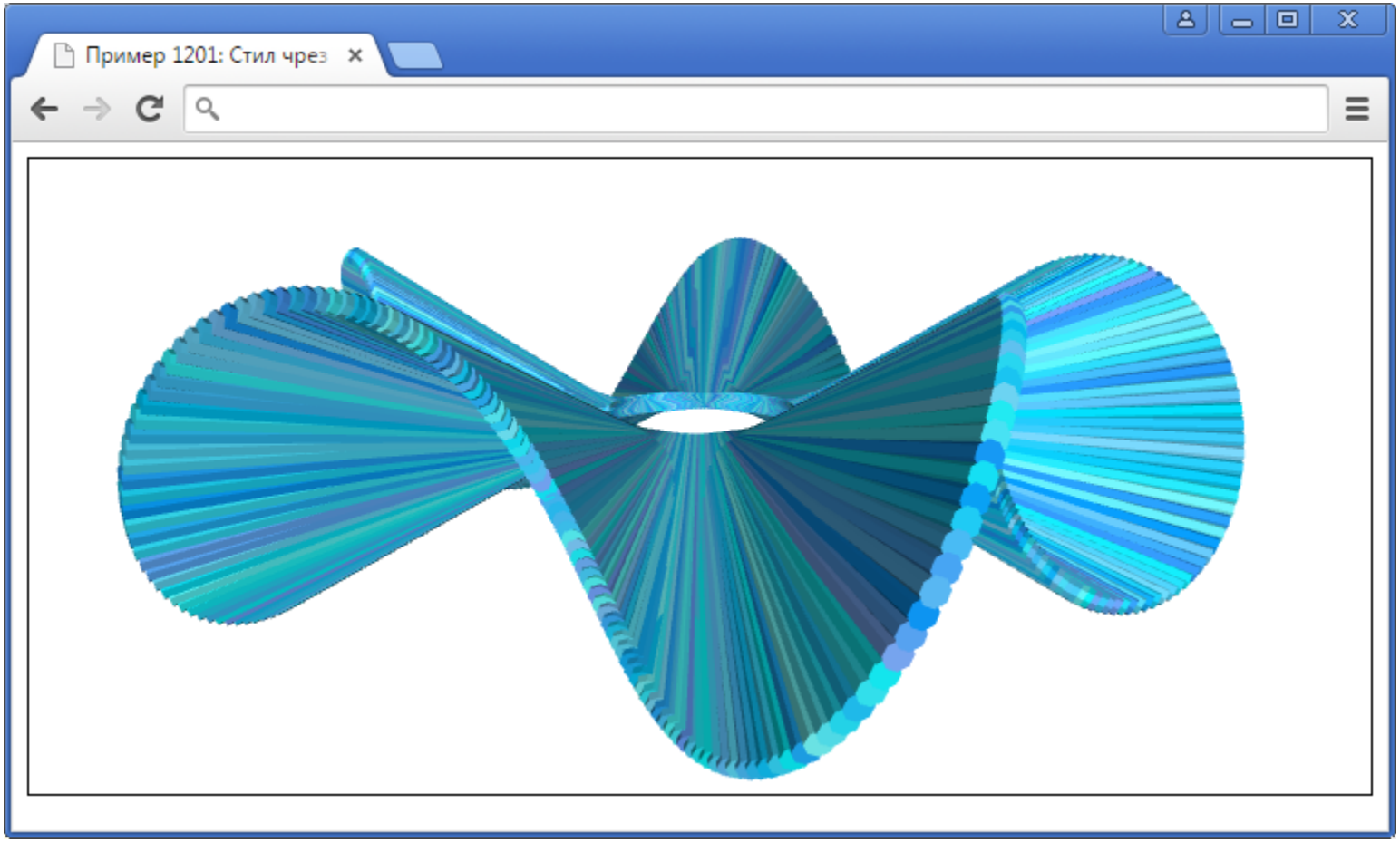
- За всеки обект изписваме променените свойства
- Неудобно при създаване на много еднотипни обекти

Стил чрез функция

Идея №1

- Функция получава обект
- Добавя му желаните стилове
- Връща обекта (в случай, че ще се присвоява на променлива)

```
function bluish(object)
{
   object.color = [random(0,0.5),random(0.6,1),1];
   return object;
}
a = bluish(prism(center,1/2,15+5*Math.cos(5*alpha),8));
```



ПРОБА

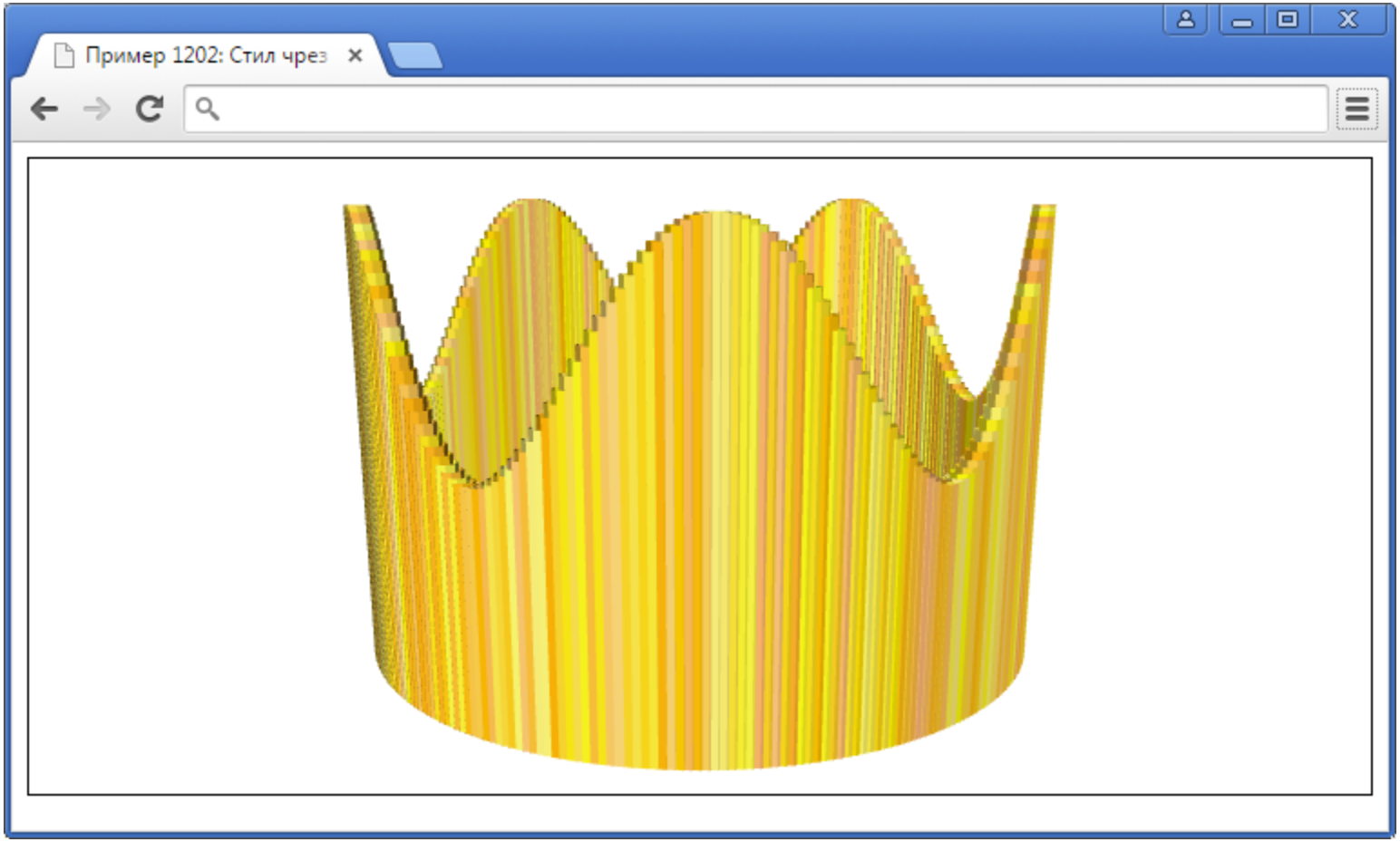
Идея №2

- Използва се метод обект.custom({свойство:стойност, …})
- Позволява да се дефинират комплект свойства в момента на създаване на анонимен обект

```
prism(center,1/2,15+5*Math.cos(5*alpha),8).custom({
   focus: [0,0,1],
   color: [1,random(0.7,1),random(0,0.5)],
   hollow: true });
```

- Ако стилът е константен, може да се изнесе в отделна променлива и да се използва многократно

```
style = {focus:[0,0,1], ...};
prism(center,...).custom(style);
```



ПРОБА

Групови обекти

Обекти в СУИКА

В библиотеката СУИКА

- Базово множество от графични обекти
- Предостатъчни за целите на курса

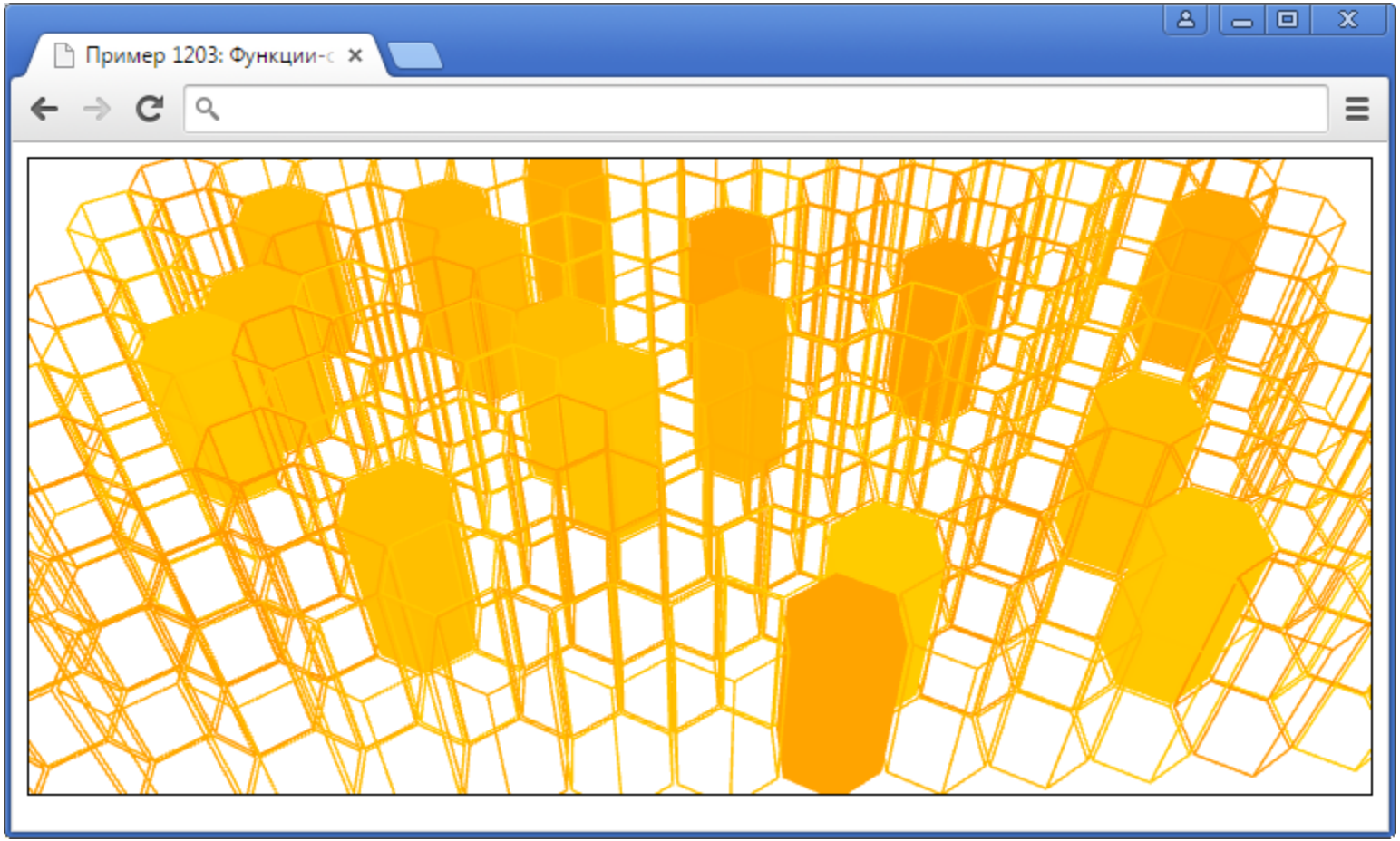
Нови обекти

- Базовите обекти допускат модификация (например брой стени на призма)
- При нужда се създава надграждане на обект
- Възможно е сглобяване на обект от няколко
- Възможно е изрязване на части от обект

Надграждане на обект

- Функция, която създава обект с искани свойства
- Като параметри са изведени само нужните свойства

```
function honeyComb(x,y)
{
   return prism([x,0.85*y,0],0.55,2,6).custom({
      color: [1,random(0.6,0.8),0],
      spin: radians(30),
      light: false,
      mode: (random(0,10)>9?Suica.SOLID:Suica.LINE)
   });
}
```

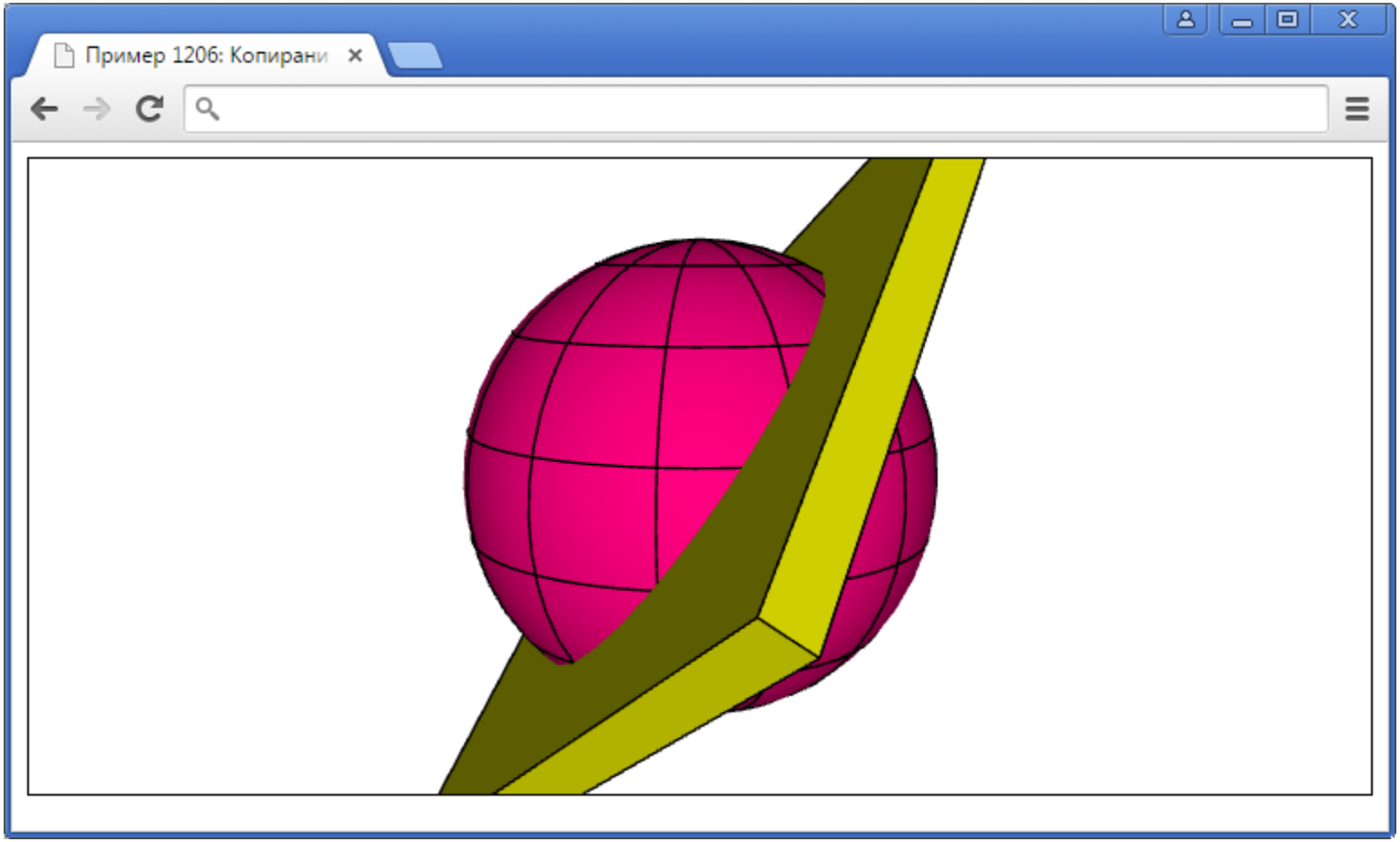


ПРОБА

Копиране на обект

- Функция sameAs(обект) създава копие на обект
- Удобно за копиране на обект с всичките му свойства
- Пример с добавяне на контури на обекти

```
a = sphere([0,0,0],5);
b = cuboid([0,0,0],[15,15,1]).custom({
        color:[1,1,0],
        focus:[1,1,1],
        spin:Math.PI/4
     });
contour = {color:[0,0,0], mode:Suica.LINE};
sameAs(a).custom(contour);
sameAs(b).custom(contour);
```



ПРОБА

Групови обекти

Група от обекти

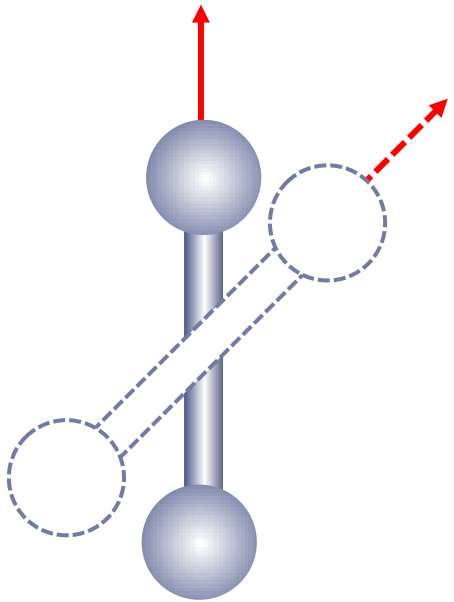
- Основен проблем при промяна на положението
- За всеки обект трябва да се преизчислява центъра и ориентацията

Решение

- Обектите се групират в един групов обект
- Груповият обект има собствено положение и ориентация

Пример

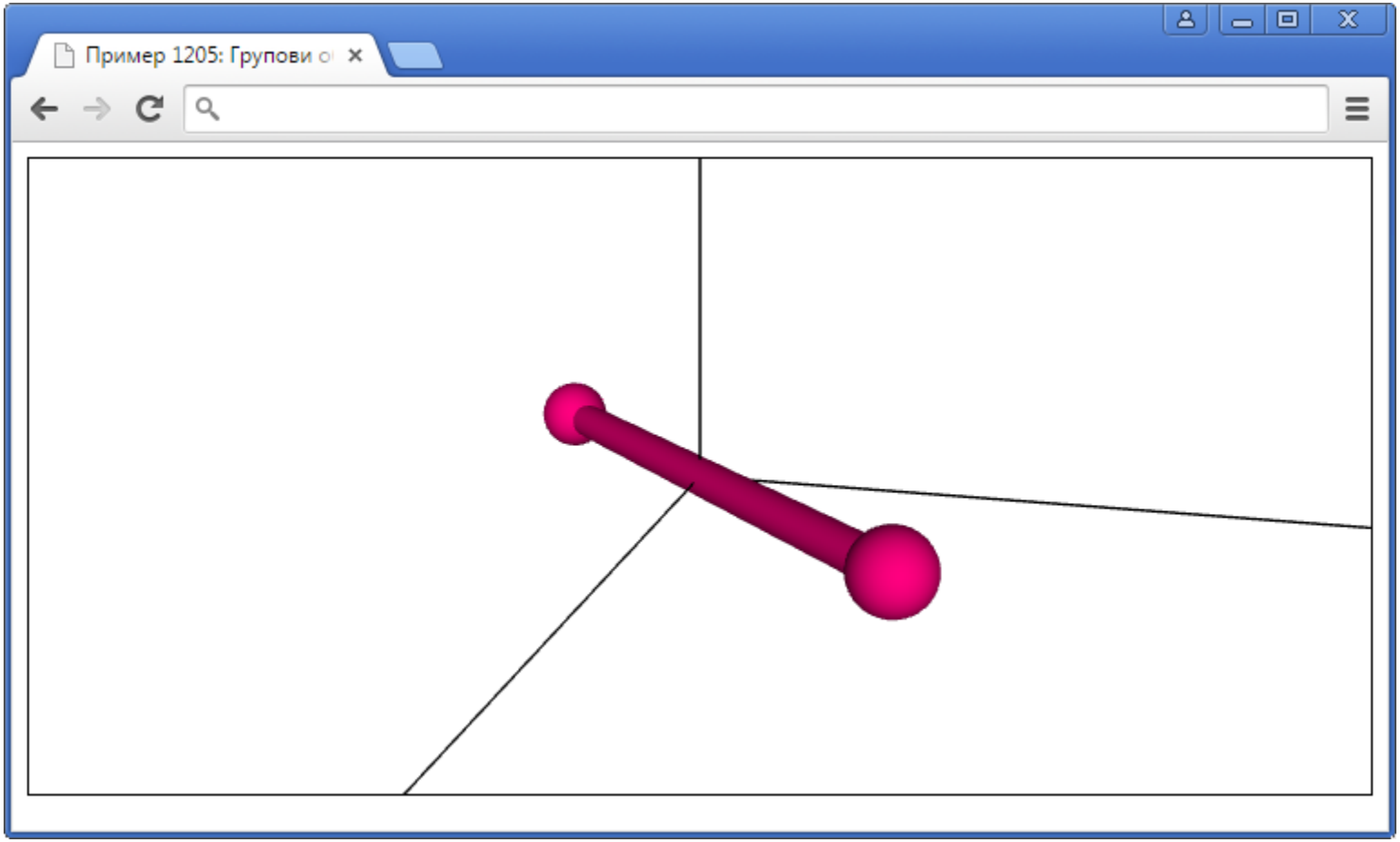
- Вертикален цилиндър с две сфери
- С focus се завърта цялата конфигурация
- Координатите на обектите остават непроменени



Решение

- Използваме обекта new Suica.Group или функцията group
- Параметърът е масив от геометрични обект
- Новата конструкция е групов обект, който си има свое независимо положение и ориентация

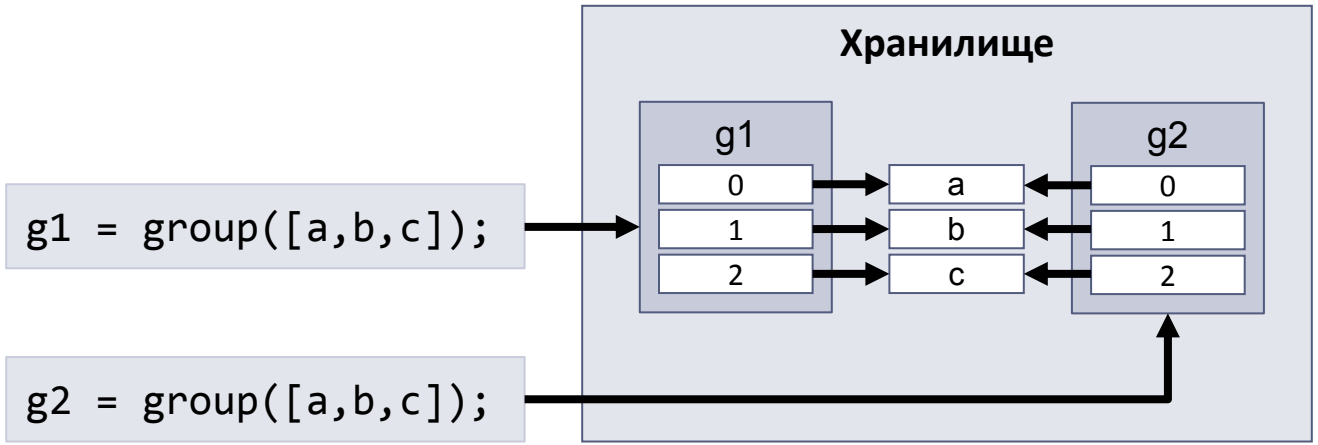
```
a = group ([
        sphere([0,0,-4],1/2),
        sphere([0,0,4],1/2),
        cylinder([0,0,-4],1/4,8)
   ]);
a.focus = [1,1,0];
```



ПРОБА

Споделени обекти

- Графичните обекти са JS обекти
- Едни и същи подобекти стават споделени
- Промяната на споделен обект се отразява визуално и в двата групови обекта



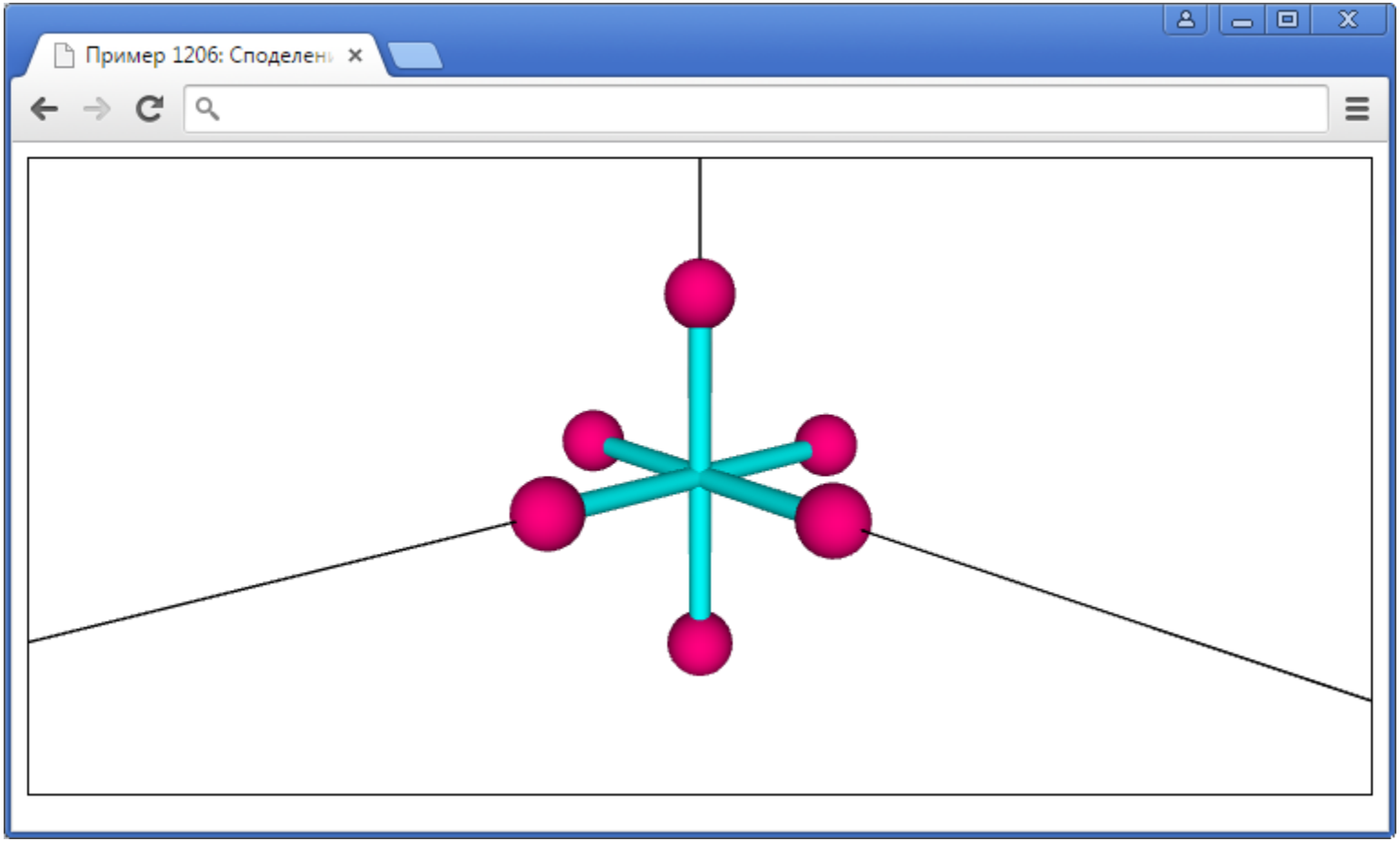
Пример

- Сглобяваме три групи от едни и същи обекти
- Ако променим цвета на обект, той се вижда променен и в трите групи

```
a = sphere([0,0,-4],3/4),
b = sphere([0,0,4],3/4),
c = cylinder([0,0,-4],1/4,8)

g1 = group([a,b,c]);
g2 = group([a,b,c]).custom({focus:[1,0,0]});
g3 = group([a,b,c]).custom({focus:[0,1,0]});;

c.color = [0,1,1];
```

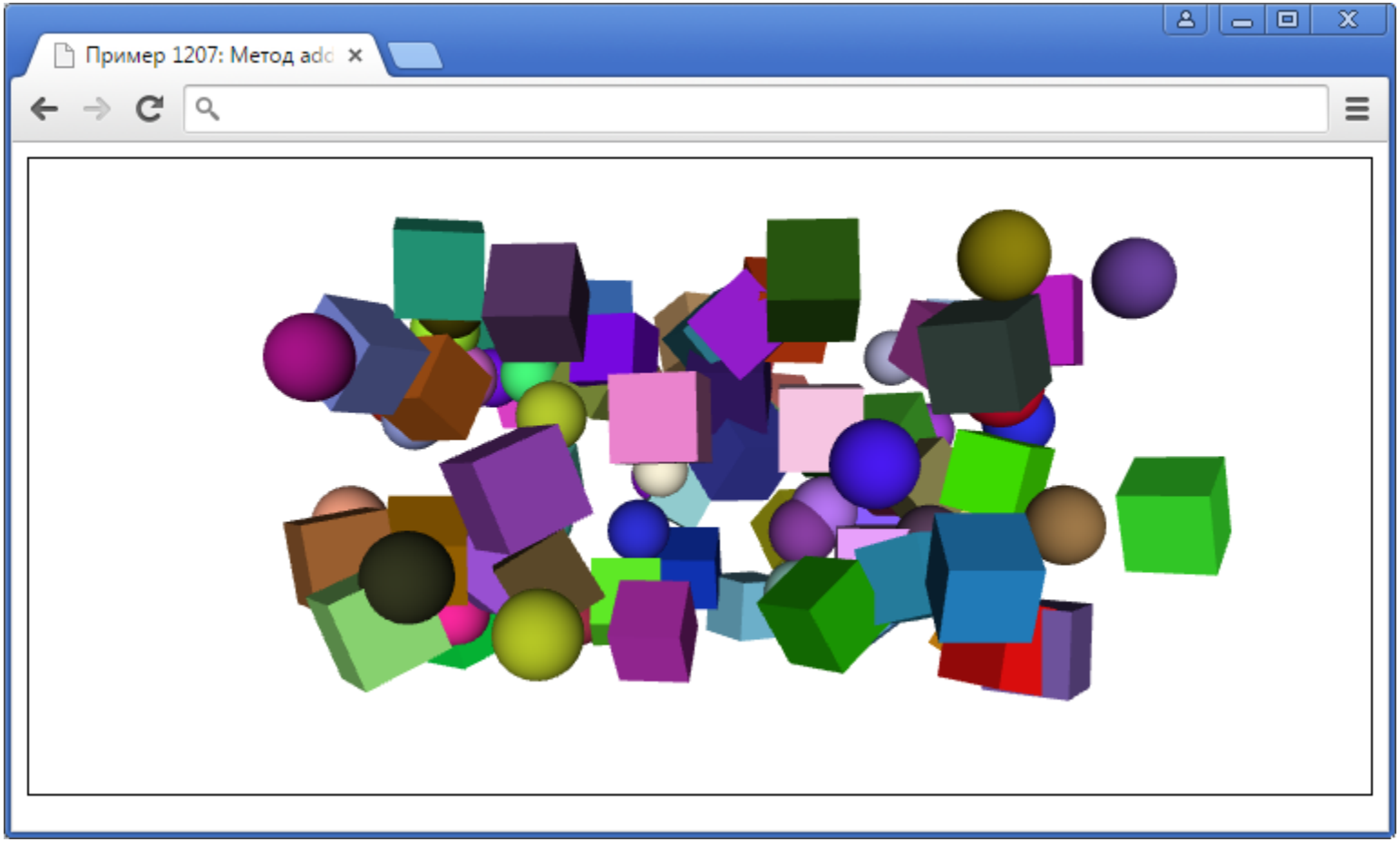


ПРОБА

Добавяне на елемент

- Към група може да се добави нов елемент с метода add(елемент)
- Елементите може да са различни графични обекти

```
a = group([]);
for (var i=0; i<100; i++)
{
   var style = {...};
   if (random(-1,1)>0)
      a.add(cube([0,0,0],2).custom(style));
   else
      a.add(sphere([0,0,0],1).custom(style));
}
```

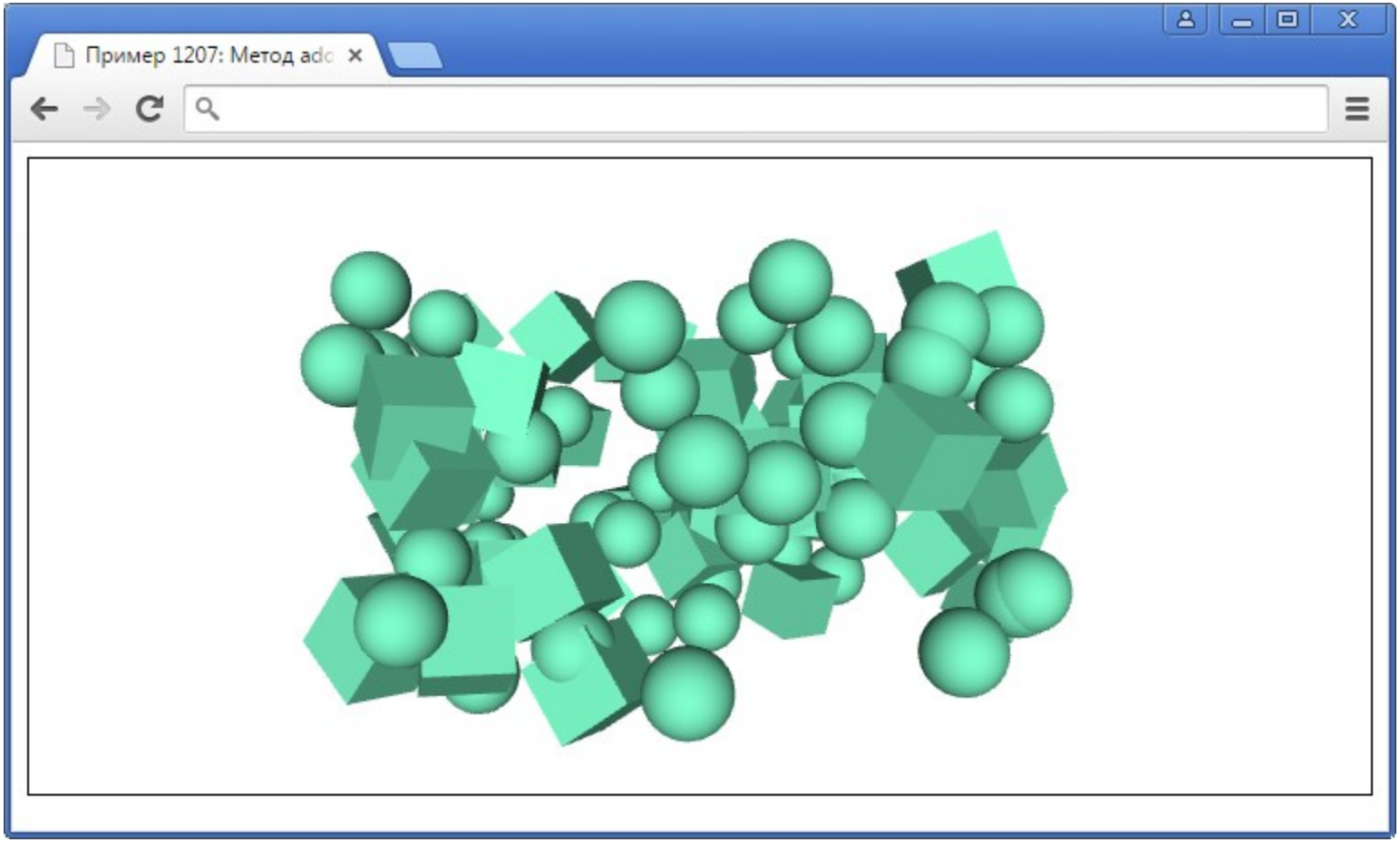


ПРОБА

Общ цвят

- Всеки елемент от група си има собствен цвят
- Методът mergeColor() премахва текущите индивидуални цветове, вместо тях се използва груповият цвят

```
a = group([]);
for (var i=0; i<100; i++)
{
   var style = {color: ...];
   a.add(cube([0,0,0],2).custom(style));
   ...
}
a.mergeColor();
a.color = [0.5,1,0.8];
```



ПРОБА

Пример

Тухлена кула

- Ред от кафеникави тухли в кръг
- После още един ред и т.н.
- Колкото по-нагоре е редът, толкова по-тесен е

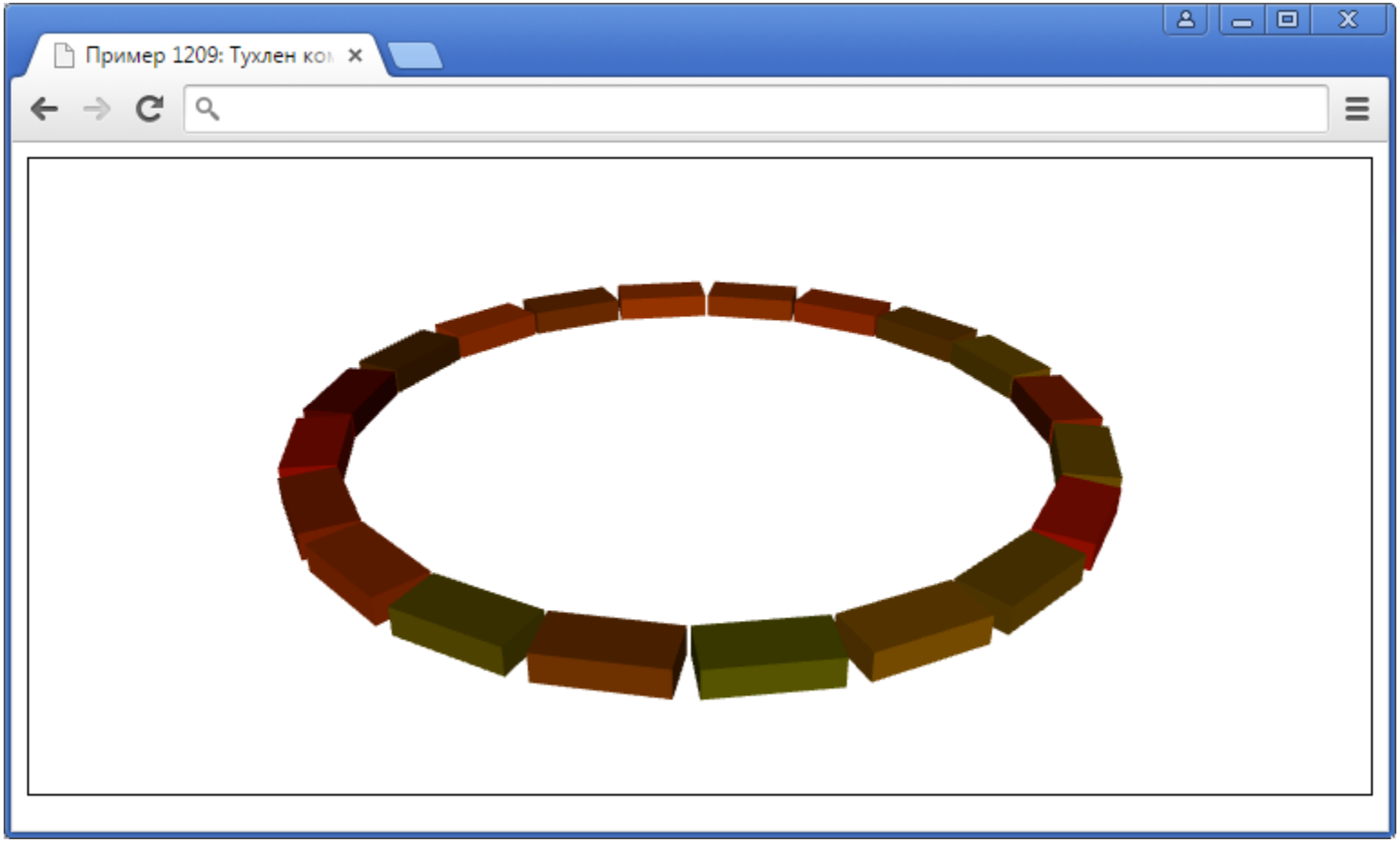
Идея

- Ще ползваме групов обект (един единствен)

Създаване на ред от тухли

- Редът е група от 20 кафеникави тухли
- С отместени центрове на 7 единици
- Завъртени през 18°

```
row = group([]);
for (var i=0; i<20; i++)
   row.add(
      cuboid([0,0,0],[2,4,1]).custom({
         origin: [7,0,0],
         color: [random(0.3,0.7),random(0,0.4),0],
         spin: 2*Math.PI*i/20
      })
   );
```

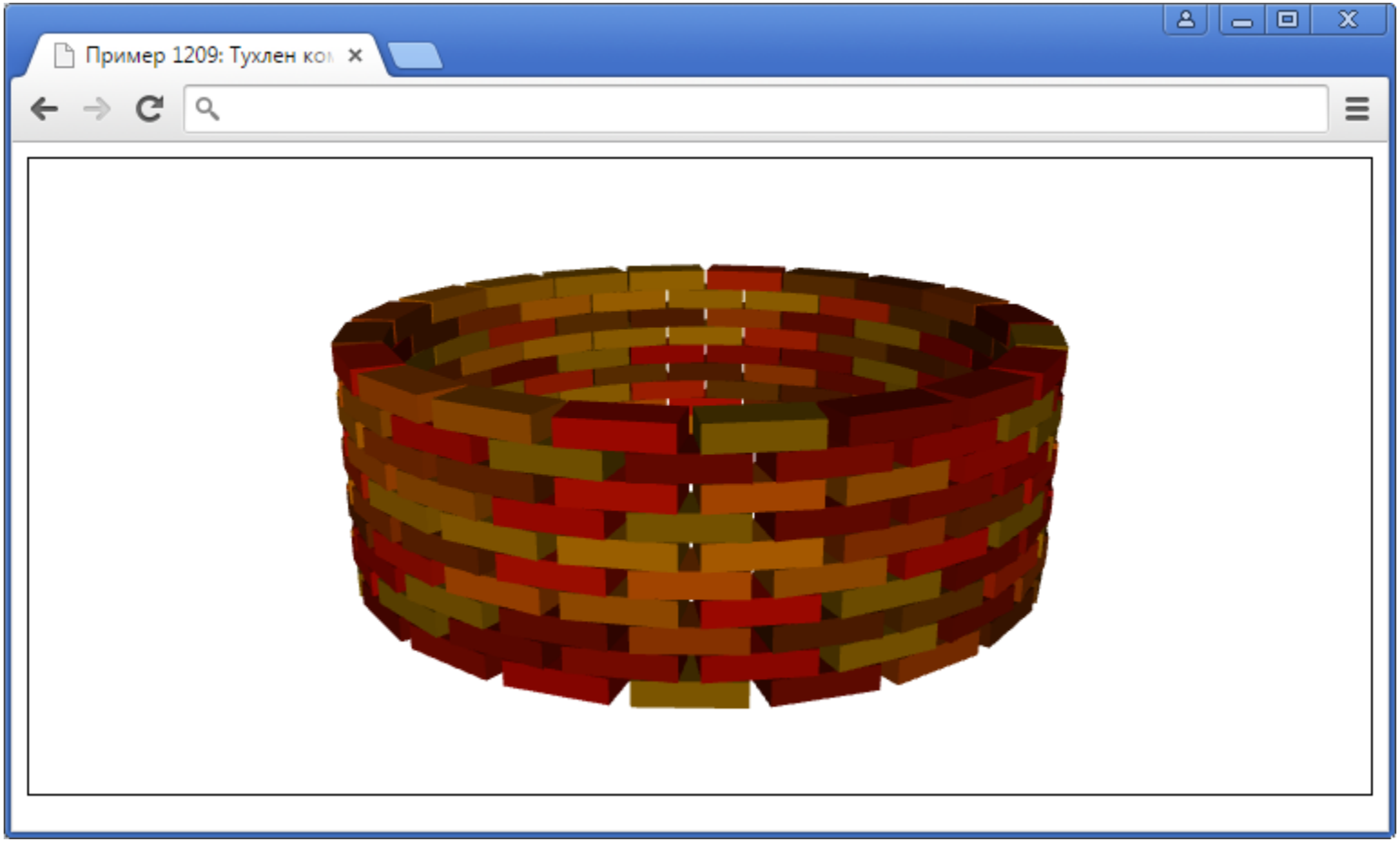


ПРОБА

Изграждане на редове

- Със sameAs създаваме копия на всеки тухлен ред
- Всяко копие е с последователни z координати (x и y се запазват нулеви)
- За „случайно“ разбъркване на цветовете завъртаме всеки ред на случаен ъгъл, кратен на 18°
- За застъпване на тухлите отместваме през ред на 9°

```
for (var i=1; i<10; i++)
   sameAs(row).custom({
      center: [0,0,i],
      spin: radians(18*Math.round(random(0,20))+9*(i%2))
   });
```

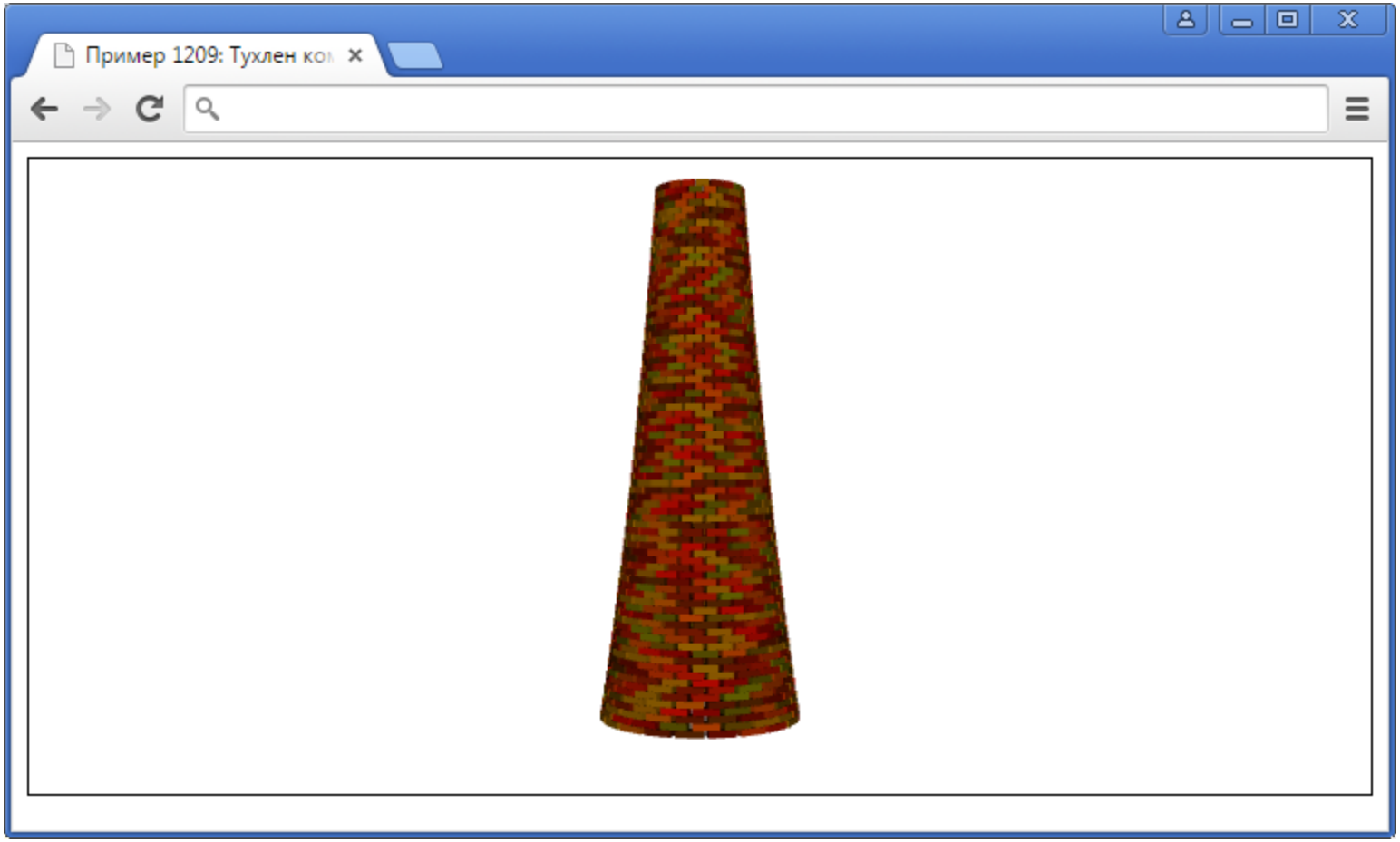


ПРОБА

Стесняване на комина

- Всеки ред тухли е групов обект
- Използваме свойството sizes за промяна на мащаба
- Започваме с мащаб k=1 и на всеки ред смаляваме с 1%

```
var k = 1;
for (var i=1; i<80; i++)
{
   k = k*0.99;
   sameAs(row).custom({
      ...
      sizes: [k,k,1]
   });
}
```



ПРОБА

Сечащи равнини

Сечащи равнини

Части от обекти

- Някои често срещани геометрични обекти са части от други
- Примери
  - Пресеченият конус е част от конус
  - Полусферата е част от сфера

Начин на генериране

- Чрез отделни малки плоскости (бавно)
- Чрез разширение на библиотеката (трудно)
- Чрез сечения на обекта с равнини

Сечащи равнини

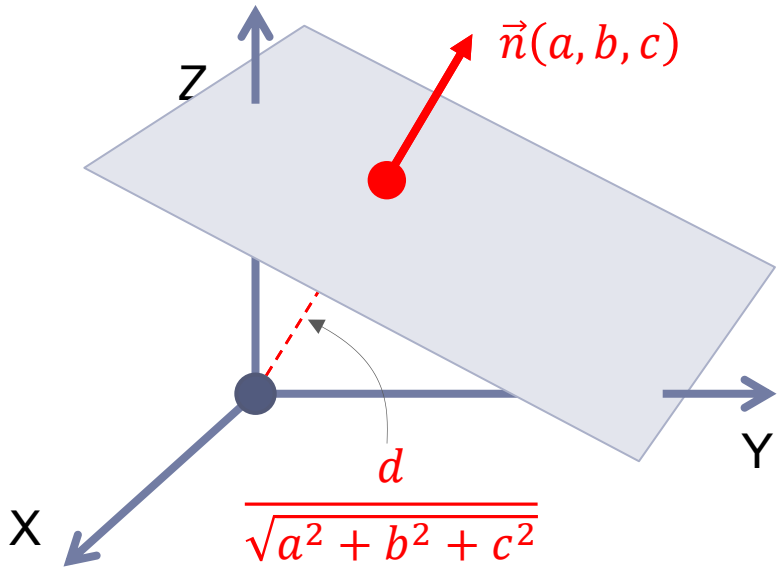
- Задава се равнина с уравнение **ax+by+cz+d=0**
- Тя разделя пространството на две полупространства
- Рисува се само тази част от обекта, която се намира в положителното полупространство **ax+by+cz+d>0**

Свойство clipPlanes

- Свойството clipPlanes е масив от до 4 подмасива
- Всеки подмасив определя една сечаща равнина $[[a_0,b_0,c_0,d_0], [a_1,b_1,c_1,d_1], [a_2,b_2,c_2,d_2], [a_3,b_3,c_3,d_3]]$
- Свойството важи само за негрупови обекти

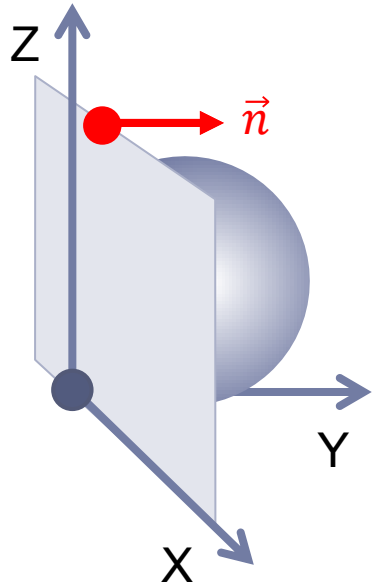
Коефициенти

- Те са спрямо локалната координатна система на обекта
- Дефинират нормалния вектор
- Определят разстоянието до равнината



Пример

- Сцена от случайни сфери
- Сечем с вертикалната равнина XZ
- Искаме да остане само полупространството откъм +Y



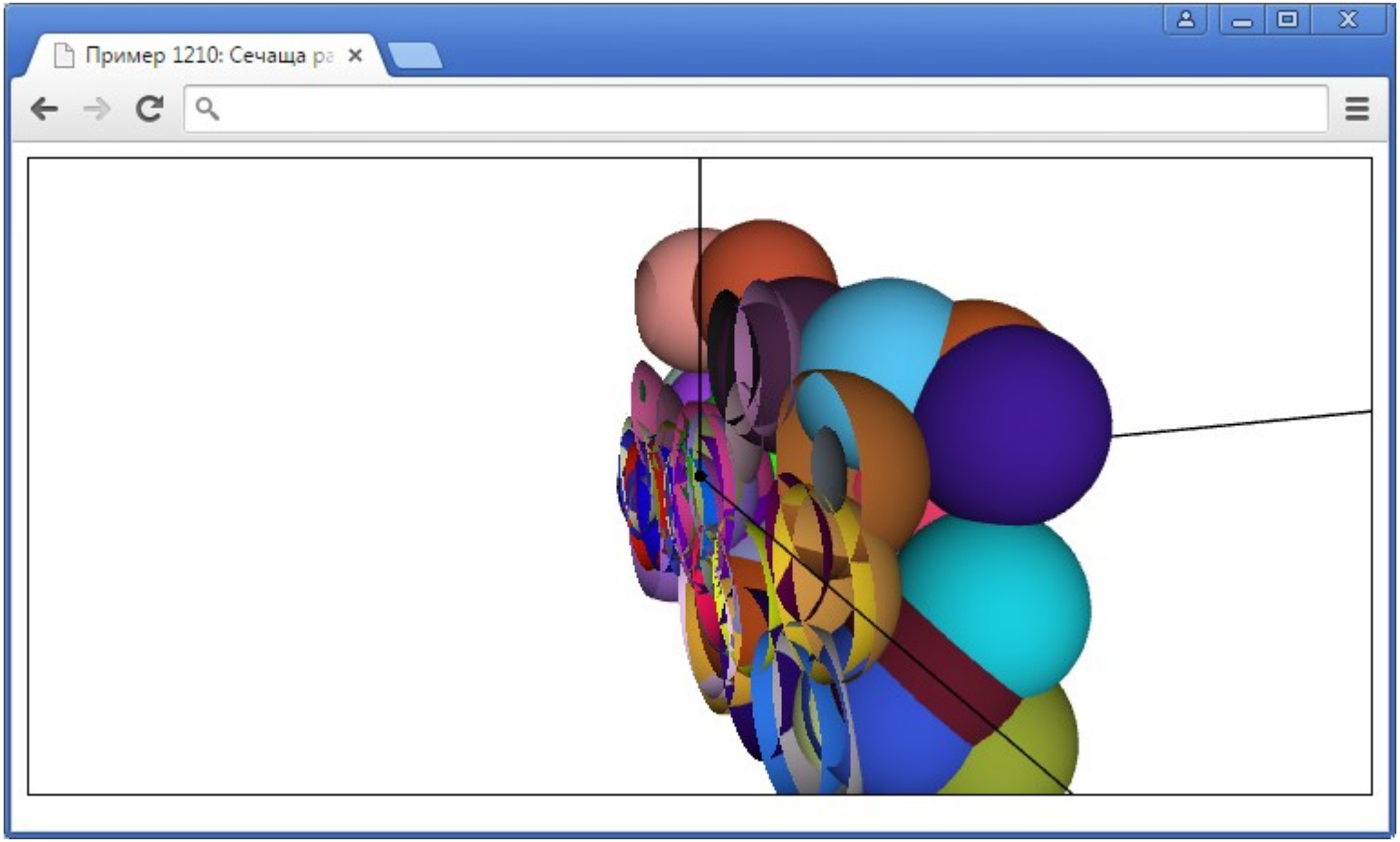
Решение

- Нормалният вектор сочи +Y, т.е. (a,b,c)=(0,1,0)
- Равнината минава през (0,0,0), т.е. d=0, но…

Трябва да ползваме локални координати! При тях:

- Центърът на сферата е (0,0,0), диаметърът ѝ е единица за разстояние, равнината е отместена на разстояние y/5, затова d=y/5, а не 0

```
var y = random(-5,5);
sphere([0,0,0],2.5).custom({
   center: [random(-10,10),y,random(-5,5)],
   color: [random(0,1),random(0,1),random(0,1)],
   clipPlanes: [[0,1,0,y/5]]
});
```



ПРОБА

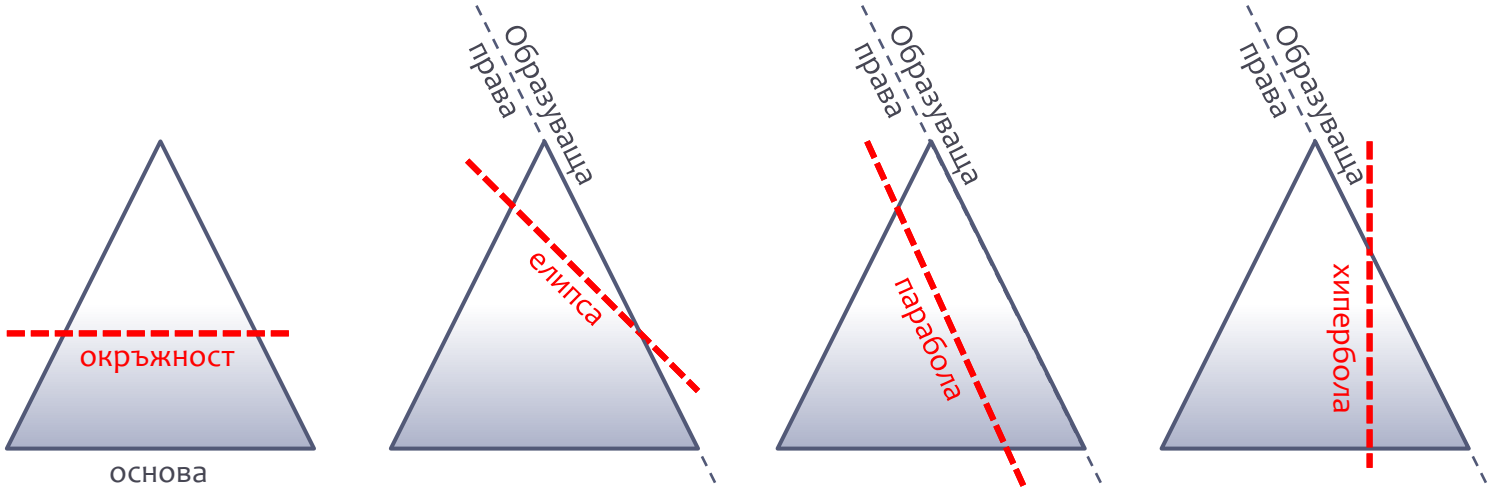
Конични сечения

Илюстрация на коничните сечения

- Четири еднакви конуса
- Всеки е пресечен с подходяща равнина
- Сеченията са:
  - Окръжност
  - Елипса
  - Парабола
  - Хипербола

Сечаща равнина

- За окръжност да е успоредна на основата
- За елипса да е наклонена, но не колкото образуваща права
- За парабола да е наклонена колкото образуваща права
- За хипербола да е наклонена повече от образуваща права

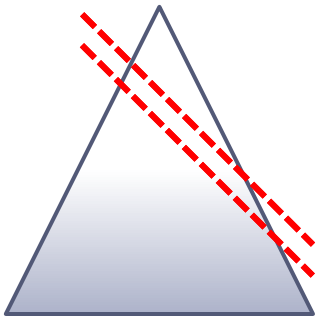


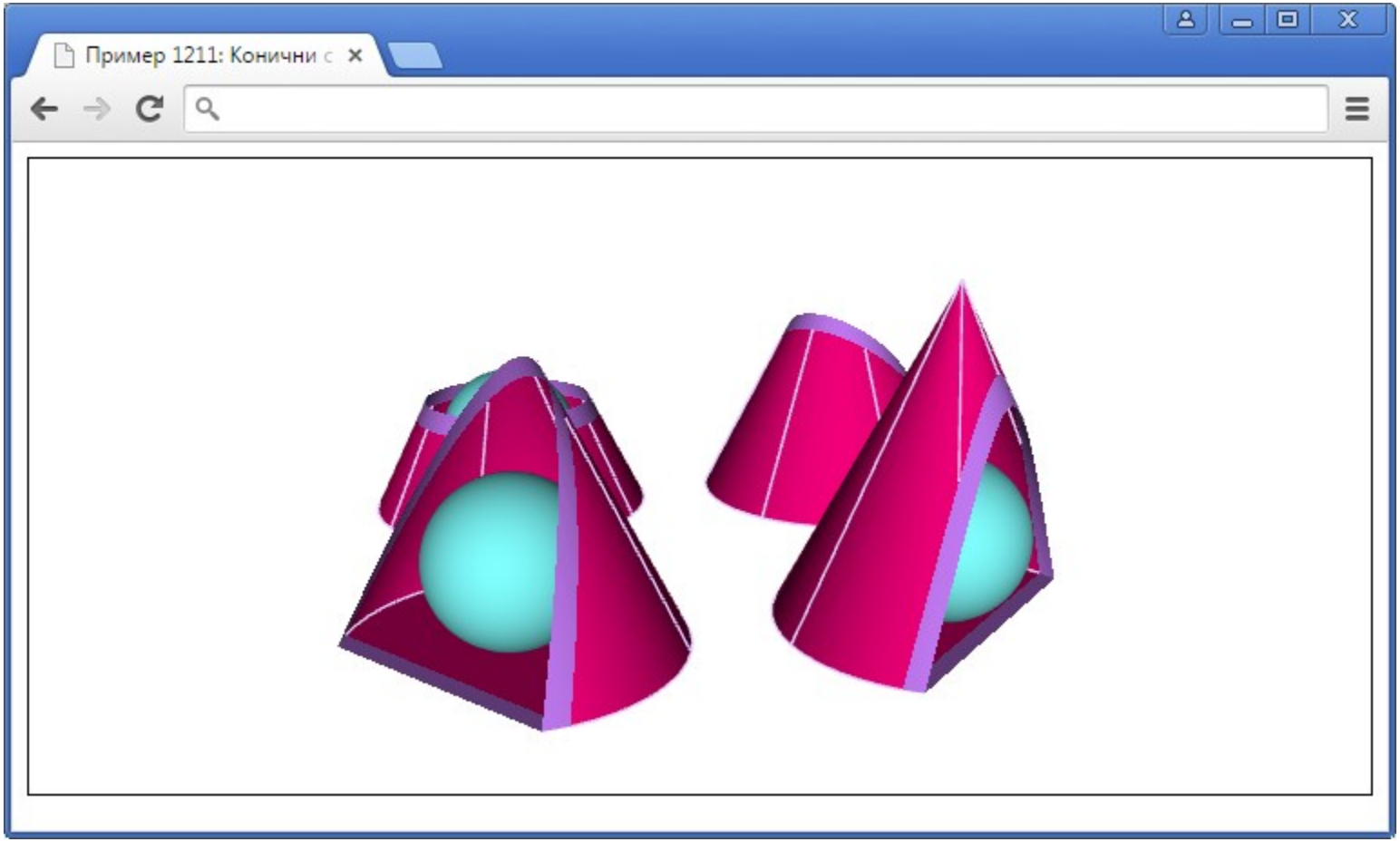
Решение

- Избираме коефициенти за сечащите равнини

```
cone(...).custom({clipPlanes: [[0,0,-1,1/3]]});
cone(...).custom({clipPlanes: [[0.8,0,-1,1/2]]});
cone(...).custom({clipPlanes: [[2,0,-1,1/2]]});
cone(...).custom({clipPlanes: [[1,0,0,1/8]]});
```

- Дублираме обектите, но в режим Suica.LINE, за да се нарисуват околните ръбове
- Пак дублираме, но слагаме две сечащи равнини, за да стане лента по протежение на сечението





ПРОБА

Обобщение

Графични обекти

Анонимни и с имена

- Анонимни, когато след създаването няма да ги променяме
- С имена (променливи), ако евентуално ще ги променяме
- Едно и също име може да се ползва последователно за няколко обекта

Задаване на стилове

- Чрез името и присвояване на стойности на свойствата
- Чрез метод custom, като стилът е множество от двойки за свойства {име:стойност, име:стойност, … }

Копиране

- Обект, заедно със свойствата му, се копира със sameAs

Групови обекти

- Конструктор new Suica.Group или функция group
- Метод add за добавяне на елементи
- Метод mergeColor за общ цвят на елементите

Сечащи равнини

- Свойство clipPlanes с коефициенти на 1 до 4 равнини

ИКТ в НОС

Край

Коментари, въпроси