Реферат на тему

**Бібліотека Java SE: Color**

Підготував Данілов Гліб Ігорович

механіко-математичний факультет,

3 курс, комп’ютерна математика 2

**Зміст**

* **Історія та передісторія колірного простору sRGB**
* **Стандартний колірний простір на практиці**
* **Java Class Color**
* **Резюму поля**
* **Підсумок конструктора**
* **Резюме методу**
* **Public color**
* **Метод Get**
* **Методи Brighter та Darker**
* **Метод HSBtoRGB**
* **Метод RGBtoHSB**
* **Метод getTransparency**

**Історія та передісторія колірного простору sRGB**

Нещодавно Міжнародний консорціум за кольором запропонував проривні вирішення проблем передачі кольору у відкритих системах. Тим не менш, формат профілю ICC не забезпечує повного рішення для всіх ситуацій.

В даний час ICC має один засіб відстеження та забезпечення правильного відображення кольору із вхідного колірного простору у вихідний. Це робиться шляхом приєднання профілю вхідного колірного простору до розглянутого зображення. Це підходить для сучасних користувачів. Однак існує широке коло користувачів, яким не потрібний такий рівень гнучкості та контролю. Крім того, більшість існуючих форматів файлів не підтримують і можуть ніколи не підтримувати вбудовування колірного профілю, і, нарешті, існує широкий спектр застосувань, які фактично відбивають у людей бажання додавати будь-які додаткові дані до своїх файлів. Загальний стандартний колірний простір RGB вирішує ці проблеми і є корисним та необхідним.

Ми очікуємо, що розробники додатків та користувачі, яким не потрібні накладні витрати на вбудовування профілів у документи чи зображення, перетворять їх у загальний колірний простір та зберігають у цьому форматі. В даний час існує безліч колірних просторів RGB-моніторів, які намагаються заповнити цю прогалину без особливих вказівок чи спроб стандартів. Необхідно об'єднати безліч стандартних та нестандартних просторів монітора RGB в єдиний стандартний колірний простір RGB. Такий стандарт міг би значно покращити точність перенесення кольорів у середовищі робочого столу. Наприклад, якщо постачальники операційної системи забезпечують підтримку стандартного колірного простору RGB, постачальники пристроїв введення та виведення, які підтримують цей стандартний колірний простір, можуть легко та впевнено передавати колір без додаткових витрат на керування кольором у найбільш поширених ситуаціях.

**Колориметричний RGB**

Дихотомія між залежним від пристрою (наприклад, кількість чорнила, виражене в CMYK, або напруга оцифрованого відео, виражене в RGB) і незалежним від пристрою колірним простором (наприклад, CIELAB або CIEXYZ) створює навантаження на продуктивність додатків, які намагаються уникнути колірних просторів пристрою. У першу чергу це пов'язано зі складністю перетворень кольорів, які необхідно виконати, щоб повернути кольори в колірні простори, що залежать від пристрою. Ця ситуація посилюється розривом у надійності між складністю та різноманітністю перетворень, що ускладнює забезпечення правильної настройки системи.

Щоб вирішити ці проблеми та задовольнити потреби систем кольорового зображення на базі ПК та Інтернету, ми пропонуємо колориметричну специфікацію RGB, що базується на середній продуктивності дисплеїв персональних комп'ютерів. Це рішення підтверджується такими спостереженнями:

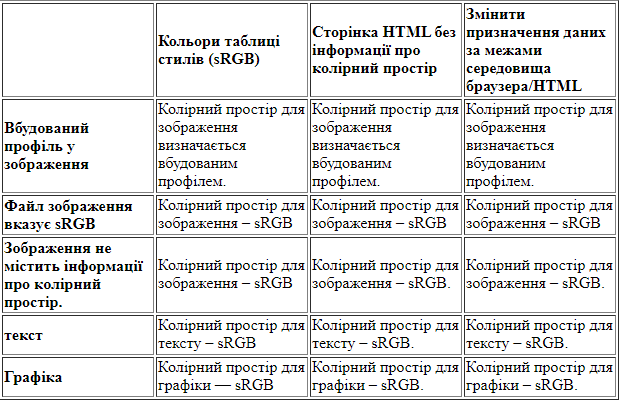
Більшість комп'ютерних моніторів схожі за своїми ключовими кольоровими характеристиками - кольоровості люмінофора (основні кольори) і передавальної функції.

Простір RGB є рідними для дисплеїв, сканерів і цифрових камер, які є пристроями з найвищими обмеженнями продуктивності.

Простір RGB можна зробити незалежними від пристроїв простим способом. Вони також можуть описувати кольори, які досить великі для всіх додатків, крім невеликого числа.

Ця комбінація факторів робить колориметричний простір RGB придатним для поширення, оскільки він може недвозначно описувати кольори і бути рідним простором для реальних апаратних пристроїв.

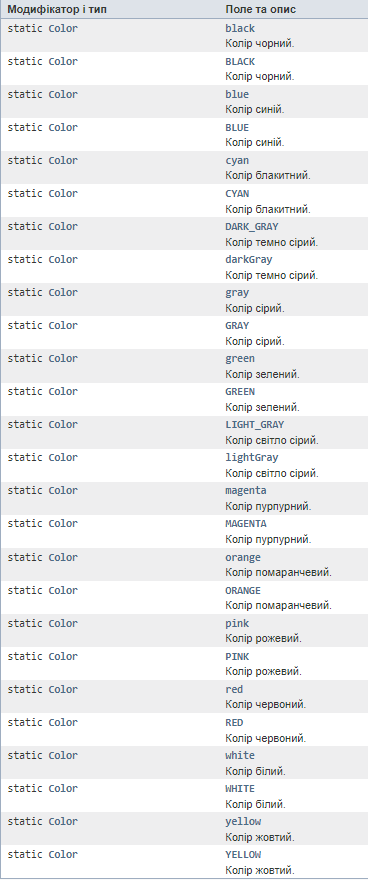
**Стандартний колірний простір на практиці**

Після перетворення елементів сторінки в sRGB браузеру необхідно правильно інтерпретувати колірний простір і використовувати керування кольором операційної системи для відображення сторінки. У наведеній нижче таблиці показано, як браузер обробляє керування кольором у кожному з можливих сценаріїв. 

**Java Class Color**

Клас  Color використовується для інкапсуляції кольорів у стандартному просторі кольорів sRGB або кольорів у довільних просторах кольорів, визначених символом ColorSpace. Кожен колір має неявне альфа-значення 1.0 або явне значення, надане в конструкторі. Значення альфа визначає прозорість кольору та може бути представлено значенням з плаваючою речовиною в діапазоні 0,0–1,0 або 0–255. Значення альфа 1,0 або 255 означає, що колір повністю непрозорий, а значення альфа 0 або 0,0 означає, що колір повністю прозорий. Під час побудови a Colorз явним альфа-каналом або отримання колірних/альфа-компонентів в Color , колірні компоненти ніколи не помножуються на альфа-компонент.

**Резюму поля**



**Конструктора:**

**Color(ColorSpace cspace, float[] components, float alpha)**

Створює колір у вказаному ColorSpace з компонентами кольору, указаними в float масиві, і вказаною альфа-версією.

**Color(float r, float g, float b)**

Створює непрозорий колір sRGB із вказаними значеннями червоного, зеленого та синього в діапазоні (0,0–1,0).

**Color(float r, float g, float b, float a)**

Створює колір sRGB із вказаними значеннями червоного, зеленого, синього та альфа-каналу в діапазоні (0,0–1,0).

**Color(int rgb)**

Створює непрозорий колір sRGB із вказаним комбінованим значенням RGB, що складається з червоного компонента в бітах 16-23, зеленого компонента в бітах 8-15 і синього компонента в бітах 0-7.

**Color(int rgba, boolean hasalpha)**

Створює колір sRGB із вказаним комбінованим значенням RGBA, що складається з альфа-компонента в бітах 24-31, червоного компонента в бітах 16-23, зеленого компонента в бітах 8-15 і синього компонента в бітах 0-7.

**Color(int r, int g, int b)**

Створює непрозорий колір sRGB із вказаними значеннями червоного, зеленого та синього в діапазоні (0–255).

**Color(int r, int g, int b, int a)**

Створює колір sRGB із вказаними значеннями червоного, зеленого, синього та альфа-каналу в діапазоні (0–255).

**Резюме мотодів:**

|  |  |
| --- | --- |
| **Color** | [**brighter**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#brighter())**()**  **Створює нову Color, яка є більш яскравою версією цього Color.** |
| [**PaintContext**](https://docs.oracle.com/javase/7/docs/api/java/awt/PaintContext.html) | [**createContext**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#createContext(java.awt.image.ColorModel,%20java.awt.Rectangle,%20java.awt.geom.Rectangle2D,%20java.awt.geom.AffineTransform,%20java.awt.RenderingHints))**(**[**ColorModel**](https://docs.oracle.com/javase/7/docs/api/java/awt/image/ColorModel.html)**cm,**[**Rectangle**](https://docs.oracle.com/javase/7/docs/api/java/awt/Rectangle.html)**r,**[**Rectangle2D**](https://docs.oracle.com/javase/7/docs/api/java/awt/geom/Rectangle2D.html)**r2d,**[**AffineTransform**](https://docs.oracle.com/javase/7/docs/api/java/awt/geom/AffineTransform.html)**xform,**[**RenderingHints**](https://docs.oracle.com/javase/7/docs/api/java/awt/RenderingHints.html)**hints)**  **Створює та повертає**[**PaintContext**](https://docs.oracle.com/javase/7/docs/api/java/awt/PaintContext.html)**використаний для створення суцільного шаблону поля кольору.** |
| [**Color**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html) | [**darker**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#darker())**()**  **Створює нову Colorтемнішу версію цього Color.** |
| **static**[**Color**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html) | [**decode**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#decode(java.lang.String))**(**[**String**](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html)**nm)**  **Перетворює a Stringна ціле число та повертає вказаний opaque Color.** |
| **boolean** | [**equals**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#equals(java.lang.Object))**(**[**Object**](https://docs.oracle.com/javase/7/docs/api/java/lang/Object.html)**obj)**  **Визначає, чи інший об'єкт дорівнює цьому Color.** |
| **int** | [**getAlpha**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getAlpha())**()**  **Повертає альфа-компонент у діапазоні 0-255.** |
| **int** | [**getBlue**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getBlue())**()**  **Повертає синій компонент у діапазоні 0–255 у стандартному просторі sRGB.** |
| **static**[**Color**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html) | [**getColor**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getColor(java.lang.String))**(**[**String**](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html)**nm)**  **Знаходить колір у властивостях системи.** |
| **static**[**Color**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html) | [**getColor**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getColor(java.lang.String,%20java.awt.Color))**(**[**String**](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html)**nm,**[**Color**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html)**v)**  **Знаходить колір у властивостях системи.** |
| **static**[**Color**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html) | [**getColor**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getColor(java.lang.String,%20int))**(**[**String**](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html)**nm, int v)**  **Знаходить колір у властивостях системи.** |
| **float[]** | [**getColorComponents**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getColorComponents(java.awt.color.ColorSpace,%20float[]))**(**[**ColorSpace**](https://docs.oracle.com/javase/7/docs/api/java/awt/color/ColorSpace.html)**cspace, float[] compArray)**  **Повертає floatмасив, що містить лише кольорові компоненти, Colorвизначені ColorSpaceпараметром cspace .** |
| **float[]** | [**getColorComponents**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getColorComponents(float[]))**(float[] compArray)**  **Повертає floatмасив, що містить лише кольорові компоненти Color, ColorSpaceу Color.** |
| [**ColorSpace**](https://docs.oracle.com/javase/7/docs/api/java/awt/color/ColorSpace.html) | [**getColorSpace**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getColorSpace())**()**  **Повертає ColorSpaceцього Color.** |
| **float[]** | [**getComponents**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getComponents(java.awt.color.ColorSpace,%20float[]))**(**[**ColorSpace**](https://docs.oracle.com/javase/7/docs/api/java/awt/color/ColorSpace.html)**cspace, float[] compArray)**  **Повертає floatмасив, що містить кольорові та альфа-компоненти Color, ColorSpaceвизначені cspace параметром.** |
| **float[]** | [**getComponents**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getComponents(float[]))**(float[] compArray)**  **Повертає floatмасив, що містить колірні та альфа-компоненти Color, ColorSpaceу Color.** |
| **int** | **getGreen()**  **Повертає зелений компонент у діапазоні 0–255 у стандартному просторі sRGB.** |
| **static**[**Color**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html) | [**getHSBColor**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getHSBColor(float,%20float,%20float))**(float h, float s, float b)**  **Створює Colorоб’єкт на основі вказаних значень для моделі кольорів HSB.** |
| **int** | [**getRed**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getRed())**()**  **Повертає червоний компонент у діапазоні 0–255 у стандартному просторі sRGB.** |
| **int** | [**getRGB**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getRGB())**()**  **Повертає значення RGB, що представляє колір у стандартному sRGB**[**ColorModel**](https://docs.oracle.com/javase/7/docs/api/java/awt/image/ColorModel.html)**.** |
| **float[]** | [**getRGBColorComponents**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getRGBColorComponents(float[]))**(float[] compArray)**  **Повертає floatмасив, що містить лише кольорові компоненти Color, у стандартному просторі кольорів sRGB.** |
| **float[]** | [**getRGBComponents**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getRGBComponents(float[]))**(float[] compArray)**  **Повертає floatмасив, що містить колірні та альфа-компоненти Color, як представлено в стандартному просторі кольорів sRGB.** |
| **int** | [**getTransparency**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#getTransparency())**()**  **Повертає режим прозорості для цього Color.** |
| **int** | [**hashCode**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#hashCode())**()**  **Обчислює для цього хеш-код Color.** |
| **static int** | [**HSBtoRGB**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#HSBtoRGB(float,%20float,%20float))**(float hue, float saturation, float brightness)**  **Перетворює компоненти кольору, визначені моделлю HSB, на еквівалентний набір значень для моделі RGB за замовчуванням.** |
| **static float[]** | [**RGBtoHSB**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#RGBtoHSB(int,%20int,%20int,%20float[]))**(int r, int g, int b, float[] hsbvals)**  **Перетворює компоненти кольору, визначені типовою моделлю RGB, на еквівалентний набір значень для відтінку, насиченості та яскравості, які є трьома компонентами моделі HSB.** |
| [**String**](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html) | [**toString**](https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html#toString())**()**  **Повертає рядкове представлення цього Color.** |

**Деталі конструктора:**

**public Color(int r,**

**int g,**

**int b)**

Створює непрозорий колір sRGB із вказаними значеннями червоного, зеленого та синього в діапазоні (0–255). Фактичний колір, який використовується під час візуалізації, залежить від пошуку найкращої відповідності з оглядом на колірний простір, доступний для певного пристрою виводу. Альфа за замовчуванням становить 255.

Параметри:

r- червоний компонент

g- зелена складова

b- синій компонент

Помилка:

IllegalArgumentException- якщо r, g або b знаходяться за межами діапазону від 0 до 255 включно

**public Color(float r,**

**float g,**

**float b,**

**float a)**Створює колір sRGB із вказаними значеннями червоного, зеленого, синього та альфа-каналу в діапазоні (0,0–1,0). Фактичний колір, який використовується під час візуалізації, залежить від пошуку найкращого збігу з оглядом на колірний простір, доступний для певного пристрою виводу.

Параметри:

r- червоний компонент

g- зелена складова

b- синій компонент

a- альфа-компонент

Помилка:

IllegalArgumentException- якщо r, g bабо aзнаходяться за межами діапазону від 0,0 до 1,0 включно

**public Color(ColorSpace cspace,**

**float[] components,**

**float alpha)**

Створює колір у вказаному ColorSpace з компонентами кольору, указаними в float масиві, і вказаною альфа-версією. Кількість компонентів визначається типом ColorSpace. Наприклад, RGB вимагає 3 компонентів, а CMYK вимагає 4 компонентів.

Параметри:

cspace- ColorSpaceдля інтерпретації компонентів

components- довільна кількість колірних компонентів, яка сумісна зColorSpace

alpha- значення альфа

Помилка:

IllegalArgumentException- якщо будь-яке зі значень у componentsмасиві або alphaвиходить за межі діапазону від 0,0 до 1,0

**Метод Get , приклад:**

**public int getRed()**

Повертає червоний компонент у діапазоні 0–255 у стандартному просторі sRGB.

**brighter , приклад:**

**public Color brighter()**

Створює нову Color, яка є більш яскравою версією цього Color.

Цей метод застосовує довільний коефіцієнт масштабування до кожного з трьох компонентів RGB цього Color, щоб створити більш яскраву версію цього Color. Цінність alphaзбережена.

Повернення:

новий Color об’єкт, який є більш яскравою версією цього Color з таким же alpha значенням.

**Darker, приклад:**

**public Color darker()**

Створює нову Color темнішу версію цього Color.

Цей метод застосовує довільний коефіцієнт масштабування до кожного з трьох компонентів RGB цього Color, щоб створити більш темну версію цього Color. Цінність alpha збережена.

Повернення:

новий Color об’єкт, який є темнішою версією цього Color з таким же alpha значенням.

**HSBtoRGB**

**public static int HSBtoRGB(float hue,**

**float saturation,**

**float brightness)**

Перетворює компоненти кольору, визначені моделлю HSB, на еквівалентний набір значень для моделі RGB за замовчуванням.

Компоненти saturationта brightnessмають бути значеннями з плаваючою комою від нуля до одиниці (числа в діапазоні 0,0-1,0). Компонентом hueможе бути будь-яке число з плаваючою комою. Нижня частина цього числа віднімається від нього, щоб створити дріб між 0 і 1. Потім це дробове число множиться на 360, щоб отримати кут відтінку в колірній моделі HSB.

Ціле число, яке повертає HSBtoRGBкодує значення кольору в бітах 0-23 цілого значення, яке є тим самим форматом, що використовується методом getRGB. Це ціле число можна надати як аргумент Colorконструктору, який приймає єдиний цілочисельний аргумент.

Параметри:

hue- відтінкова складова кольору

saturation- насиченість кольору

brightness- яскравість кольору

Повернення:

значення RGB кольору з указаними відтінком, насиченістю та яскравістю.

**RGB to HSB**

**public static float[] RGBtoHSB(int r,**

**int g,**

**int b,**

**float[] hsbvals)**

Перетворює компоненти кольору, визначені типовою моделлю RGB, на еквівалентний набір значень для відтінку, насиченості та яскравості, які є трьома компонентами моделі HSB.

Якщо hsbvalsаргумент дорівнює null, то для повернення результату виділяється новий масив. В іншому випадку метод повертає масив hsbvalsзі значеннями, розміщеними в цьому масиві.

Параметри:

r- червона складова кольору

g- зелена складова кольору

b- синя складова кольору

hsbvals- масив, який використовується для повернення трьох значень HSB, абоnull

Повернення:

масив із трьох елементів, що містить відтінок, насиченість і яскравість (у такому порядку) кольору з позначеними червоним, зеленим і синім компонентами.

**getTransparency**

**public int getTransparency()**

Повертає режим прозорості для цього Color. Це необхідно для реалізації Paintінтерфейсу.

Визначено:

getTransparency в інтерфейсі Transparency

Повернення: режим прозорості цього Colorоб'єкта.

***Джерело:***

***https://docs.oracle.com/javase/7/docs/api/java/awt/Color.html***