

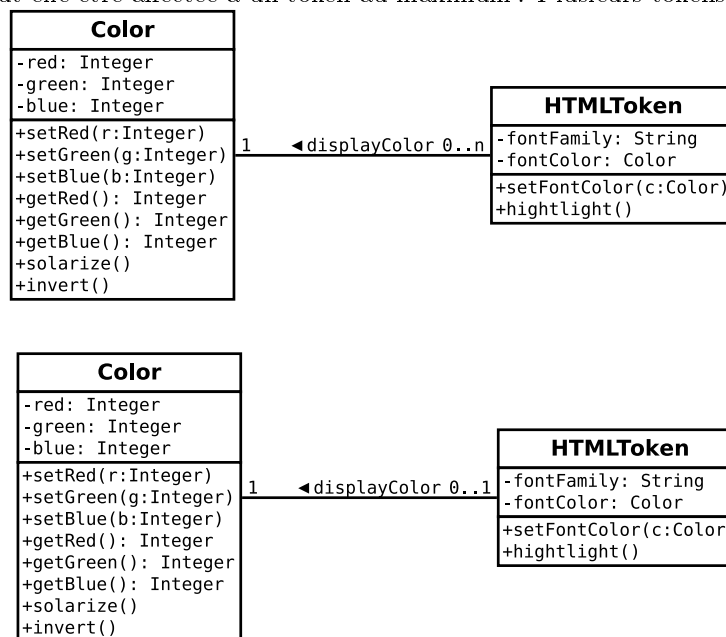
UML : modélisation de la relation Color/TokenHTML

Franck Sajous - CLLE-ERSS

Ce document est disponible à l'adresse : <http://fsajous.free.fr/>

1 Choix de multiplicités

Un(e) (instance de) TokenHTML a/est associé à une couleur (d'affichage). On pourrait choisir la multiplicité 0..1 (si un token n'est associé à aucune couleur, on lui attribue une couleur par défaut). La multiplicité du côté HTMLToken est plus intéressante : étant donnée une couleur, à combien de tokens doit/peut-elle être affectée ? La borne minimale ne pose pas de problème. Une couleur peut-elle être affectée à un token au maximum ? Plusieurs tokens ?



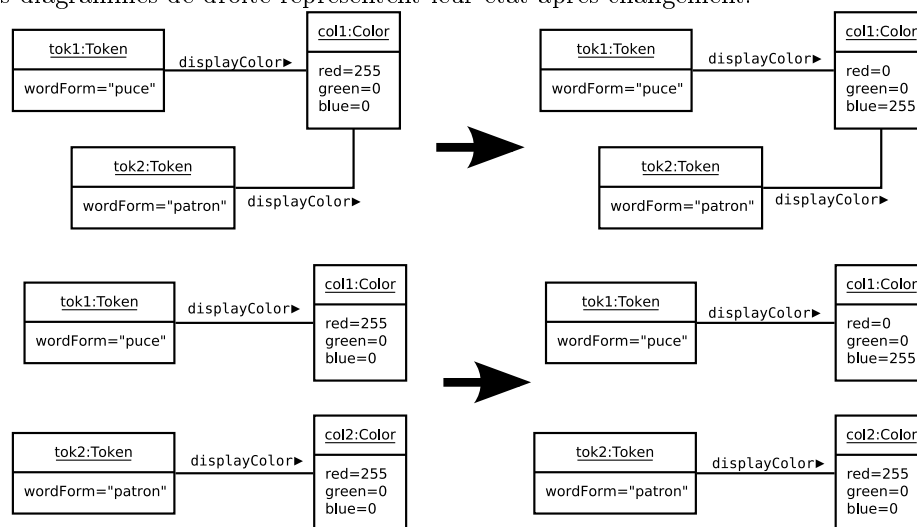
Il ne faut pas ici donner les 2 choix possibles de multiplicité et les paraphraser. Il faut dire, pour chacune des possibilités, ce qu'elle implique/représente concrètement et trancher : si on veut modéliser ceci, tel choix est le meilleur parce que... Réfléchir en termes de diagrammes d'objets est toujours une bonne idée dans ce cas.

Les questions « que signifie qu'un token *est* rouge »/ que deux tokens *sont* rouges » étaient destinées à vous faire aller dans ce sens.

Les diagrammes d'objets ci-après illustrent le choix de la multiplicité 0..n (en haut) et 0..1 (en bas). On imagine que l'on change la couleur de `tok1` par les instructions suivantes :

```
tok1.getColor().setRed (0);
tok1.getColor().setGreen (0);
tok1.getColor().setBlue (255);
```

Les diagrammes de gauche représentent l'état des objets avant changement. Les diagrammes de droite représentent leur état après changement.



2 Proposition d'une autre modélisation

La question qui se pose est : dans notre modélisation, dans le système auquel on s'intéresse, qu'est-ce qu'une couleur ? Y a-t-il une seule couleur rouge ? Tous les tokens colorés en rouge sont-ils associés à la même instance de couleur ? Ou chaque token est-il associé à une couleur unique, plusieurs instances de `Color` pouvant avoir les mêmes *valeurs d'attributs* (r, g, b) ?

Cela rappelle (ou pourrait rappeler) la modélisation des points remarquables des objets géométriques et les points fixes du repère : chaque cercle a un centre unique, ces centres pouvant tous coïncider (notamment, avec un point particulier du repère).

On peut alors proposer l'une des deux modélisations ci-après, dans lesquelles on introduit la notion de couleur modifiable ou non, puis celle de couleur modifiable et couleur de référence. Comme pour les points non-mobiles, la classe `ReferenceColor` est privée des setters qui permettent de modifier les composantes RGB. L'initialisation d'un objet peut se faire par le constructeur. Rien n'empêche d'avoir une couleur modifiable qui ait les mêmes valeurs, e.g. (0, 0, 255) qu'une couleur **bleu de référence**. On peut envisager qu'à un token puisse correspondre soit une couleur de référence (non modifiable) soit une couleur modifiable. Si on veut recolorer différemment un token auquel on aurait associé la couleur **bleu de référence**, on ne pourrait pas le faire par une instruction

de type `tok1.getColor().setGreen (123)`; mais on pourrait attribuer à ce token une autre couleur définie par ailleurs : `tok1.setColor(otherColor)`;

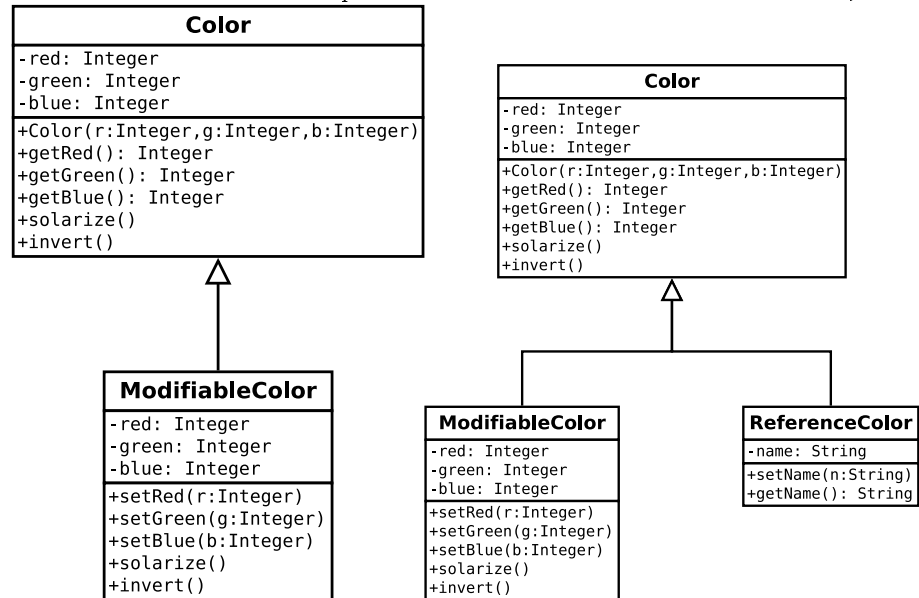
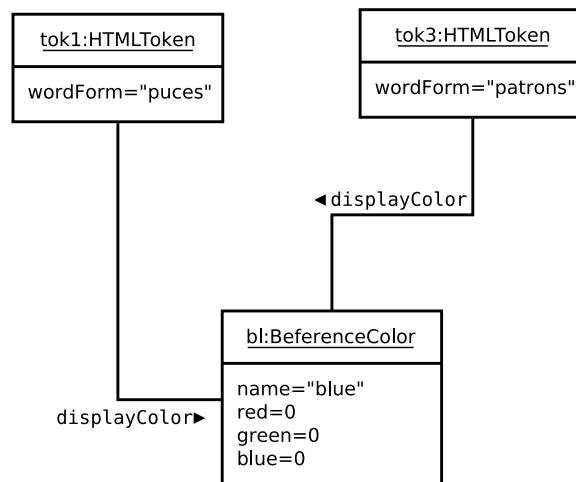
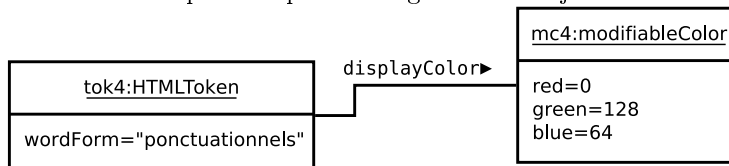


Illustration possible par un diagramme d'objets :



3 Raffinage du modèle

Si l'on introduit la notion de style, on n'est pas loin de la réalité (HTML/CSS). On peut appliquer 0 ou plusieurs styles (typiquement un seul, mais plusieurs sont possibles; il existe dans ce cas un système de priorité qui dépasse le cadre de cet exercice) à un TokenHTML. Un style peut être appliqué à 0 ou plusieurs tokens. Le style peut être composé (de manière optionnelle), entre autres, d'une fonte et d'une couleur.

