

# JavaVersHTML - Documentation, Enoncé extrait de la page de Vasile

## Objectif :

Présenter des programmes Java dans une page HTML en respectant la coloration syntaxique par défaut ci-dessous. (cet utilitaire vous permettra de présenter les sources Java dans vos rapports futurs).

1. chaînes de caractères
2. commentaires de type //
3. commentaires de type /\*\*/
4. caractères
5. mots-clé Java
6. variables privées
7. variables publiques
8. <T> (voir jdk1.5.x )
9. doubles
10. floats

Ces paramètres peuvent être changés par l'intermédiaire d'un fichier : voir la classe Properties et la méthode load .

Attention! Les choix ci-dessus diffèrent de ceux de Vasile ... Vous pouvez optionnellement ajouter la coloration des classes de Java et d'autres éléments syntaxiques.

## Proposer un développement de JavaVersHTML en Swing selon le modèle MVC :

- **en Applette (JApplet)**
  - le source Java est saisi ( ou copié) dans un "JTextArea"
  - la conversion HTML "brute" apparaît dans un autre "JTextArea" que l'on peut ensuite aisément copier dans un éditeur de textes ou de pages HTML.
- **et en Application**
  - le texte peut-être chargé à partir d'un fichier dans un "JTextArea" et la conversion peut-être sauvegardée de l'autre JTextArea dans un fichier.
- **et en Application compatible JavaWebStart**
  - Une mise en oeuvre simplifiée d'un JNLP avec une archive java signée est ici :  
[http://jfod.cnam.fr/csiml\\_b2/projetJINI/jnlp/](http://jfod.cnam.fr/csiml_b2/projetJINI/jnlp/)

Les sources de base de Andrei Cioroianu peuvent être utilisés mais DOIVENT ÊTRE améliorés en particulier par l'usage des classes "Stream/String Tokenizer et Scanner" pour l'analyse du fichier en entrée et en respectant le modèle MVC.

**Ce qui est demandé**, donc à envoyer selon le submit habituel, dans le fichier "tp8.jar" contenant les ".class" , tous les sources et la documentation, toutes les classes de tests unitaires nécessaires, et le fichier jnlp

(pour lancer une applette dans une archive : <APPLET code=Java2html.class archive="Java2html.jar" >)

---

## Un exemple (perfectible) de mise en forme la Pile du TP2 :

```

public class Pile<T>{
    private T[] zone;
    private int ptr;
    public Pile(int taille){ ptr = 0; zone = new int[taille];}
    public void empiler(T i) throws PilePleineException{
        if(estPleine())
            throw new PilePleineException();
        zone[ptr] = i; ptr++;
    }
    public T depiler()throws PileVideException{
        if(estVide())
            throw new PileVideException();
        ptr--; return (T)zone[ptr];
    }
    public boolean estVide(){return ptr==0;}
    public boolean estPleine(){return ptr==zone.length;}
    public String toString(){
        String s = "[";
        for(int k=ptr-1;k >= 0; k--){
            s = s + zone[k];
            if(k>0)s = s + ", ";
        }
        return s + "]";
    }
}

```

```

<pre><font color="#0000FF">public</font> <font color="#0000FF">class</font> Pile<stro
<font color="#0000FF">private</font> <font color="#0000FF"><strong>T</strong></font>
<font color="#0000FF">private</font> <font color="#0000FF">int</font> ptr;
<font color="#0000FF">public</font> Pile(<font color="#0000FF">int</font> taille){
color="#0000FF">new</font> <font color="#0000FF">int</font>[taille];}
<font color="#0000FF">public</font> <font color="#0000FF">void</font> empiler(<stro
<font color="#0000FF">if</font>(estPleine())
    <font color="#0000FF">throw</font> <font color="#0000FF">new</font> P
zone[ptr] = i; ptr++;
}
<font color="#0000FF">public</font> <strong>T</strong> depiler()throws PileVideExce
<font color="#0000FF">if</font>(estVide())
    <font color="#0000FF">throw</font> <font color="#0000FF">new</font> P
ptr--; <font color="#0000FF">return</font> (<strong>T</strong>)zone[ptr];
}
<font color="#0000FF">public</font> <font color="#0000FF">boolean</font> estVide(){
color="#0000FF">return</font> ptr==<b>0</b>;}
<font color="#0000FF">public</font> <font color="#0000FF">boolean</font> estPleine(
color="#0000FF">return</font> ptr==zone.length;}
<font color="#0000FF">public</font> <font color="#0F0F0F">String</font> toString(){
<font color="#0F0F0F">String</font> s = <font color="#FF0000">&quot;[&quot;</font></font>
<font color="#0000FF">for</font>(<font color="#0000FF">int</font> k=ptr-<b>1</b>
s = s + zone[k];
<font color="#0000FF">if</font>(k<gt;<b>0</b>)s = s + <font
color="#FF0000">&quot;;&quot;</font>;
}
<font color="#0000FF">return</font> s + <font color="#FF0000">&quot;]&quot;</font></font>
}
}
</pre>

```