

PHP-VRML v 0.4

Document de référence

Sommaire :

1. Introduction
2. Organisation générale
3. Utilisation
 - a. General
 - b. Liste des fonctions
 - i. Espace de travail
 - ii. Formes
 - iii. Apparence
 - iv. Transformations
 - v. Autres
4. Conclusion
5. Remerciements

1. Introduction

Dans un monde où internet est ultra présent, où le PHP est un langage des plus utilisés, j'ai remarqué que quasiment tous les sites internet sont en deux dimensions. Cela malgré le fait que les cartes graphiques deviennent de plus en plus puissantes et que tous les nouveaux jeux créés sont en trois dimensions.

Pourtant les technologies permettant d'afficher des éléments en trois dimensions dans un navigateur internet existent bel et bien, nous pouvons citer par exemple le Vrml 1.0, le Vrml 2.0 (ou Vrml97), x3d, Java 3D, 3DMLW...

Le Vrml ou Virtual Reality Modeling Language, est un langage permettant d'afficher des objets en trois dimensions dont les premières spécifications datent de 1994. En 1997 un nouveau langage fut finalisé : le Vrml97 ou Vrml 2.0 qui devint une norme ISO.

La bibliothèque de classes PHP-Vrml vous permet de créer dynamiquement des objets à l'aide de PHP.

2. Organisation générale

La librairie PHP-Vrml est codé en PHP objet pour une plus grande clarté et une plus grande puissance.

Tous les fichiers sont stockés dans un dossier vrml-class.

Pour pouvoir accéder à cette librairie il suffit d'inclure le fichier vrml.php, il se chargera de charger automatiquement les classes que vous utiliserez.

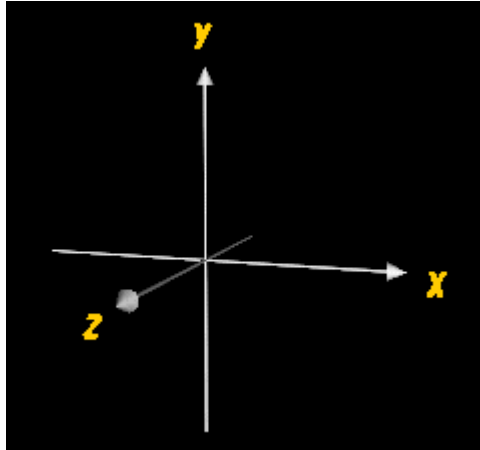
Contenu du dossier vrml-class :

- anchor.php
- color.php
- cone.php
- cube.php
- cylinder.php
- elevationGrid.php
- inline.php
- navigationInfo.php
- pointLight.php
- rotation.php
- scale.php
- sphere.php
- text.php
- texture.php
- transform.php
- translation.php
- view.php
- vrml.php

3. Utilisation

a. General

L'utilisation des coordonnées en 3 dimensions n'est pas très évident donc voici un petit schéma pour visualiser les axes :



Pour avoir un bon rendu des proportions en vrml on considère qu'une unité dans le repère orthonormé direct correspond à 1m dans la vie réelle.

Pour initialiser le système il faut :

```
<?php
//inclure la classe vrml
require "vrml-class/vrml.php";

//créer l'espace de travail
$univers=new Vrml();

// ...

//afficher l'espace de travail
$univers->show();
?>
```

Vous reconnaissez sûrement la syntaxe « objet » du PHP. Ce code ne fait que d'afficher une page vide car vous n'avez encore rien ajouté à votre espace de travail.

Maintenant nous allons créer notre premier cube, les paramètres que l'on passe sont dans l'ordre : un nom, une dimension en x, une dimension en y, une dimension en z, et une couleur (ou une texture).

Il faut auparavant créer la couleur et l'assigner au cube. Les paramètres que l'on passe sont dans l'ordre : rouge, vert, bleu, nom de la couleur, l'intensité ambiante (facultatif,

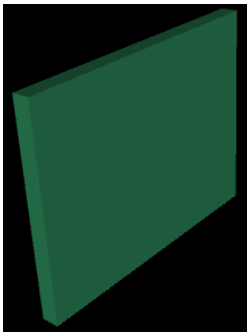
par défaut à 0.200), brillance (facultatif, par défaut à 0.200), la transparence (facultatif, par défaut à 0) toutes ces valeurs doivent être comprise entre 0 et 1

```
//création de la couleur
$green=new Color(.2,.6,.4, "Green",0.200,0.200);
//création du cube de 0,25m x 3m x 4m
$cube=new Cube("test",.25,3,4,$green);
//ajout du cube dans l'espace de travail
$univers->add($cube);
```

On obtient :



Essayez maintenant de le faire pivoter à l'aide de la souris. Vous pouvez obtenir :



Vous voyez c'est pas si compliqué !

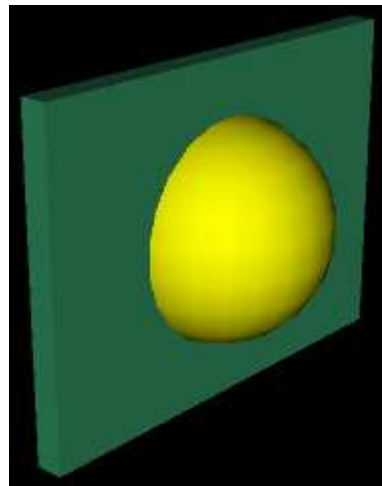
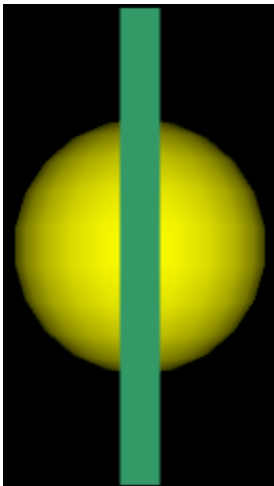
Nous allons ajouter maintenant une sphère jaune, les paramètres que l'on passe sont dans l'ordre : un nom, un rayon, une couleur.

```
//création de la couleur jaune
$yellow=new Color(1,1,0, "Yellow",0.200,0.200);

//création d'une sphere jaune
$sphere=new Sphere("Sphere", 1,$yellow);

//ajout de la sphere dans l'espace de travail
$univers->add($sphere);
```

On obtient :



Pour déplacer la sphère il faut effectuer une transformation sur celle-ci avant de l'ajouter à l'espace de travail. Nous reprenons le code de la sphère :

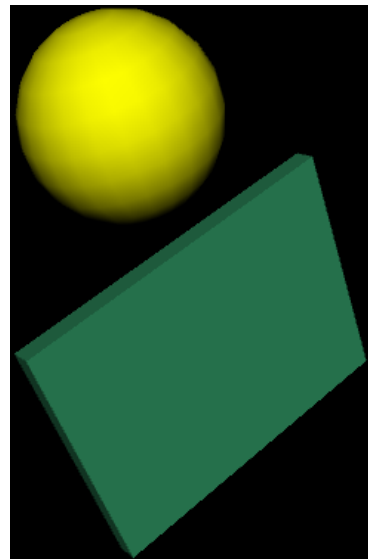
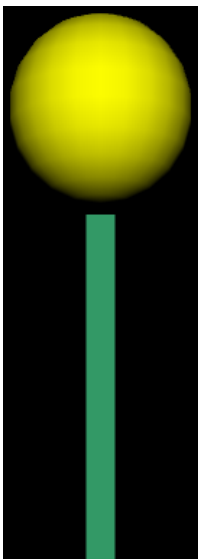
```
//création de la couleur jaune
$yellow=new Color(1,1,0, "Yellow",0.200,0.200);

//création d'une sphere jaune
$sphere=new Sphere("Sphere", 1,$yellow);

//déplacement de la sphere de 3m vers le haut
$sphere->addTransformation(new Translation(0,3,0));

//ajout de la sphere dans l'espace de travail
$univers->add($sphere);
```

On obtient :



b. Liste des fonctions

Vous trouverez ici la liste des fonctions de chaque objet qui peuvent vous être utile.

i. Espace de travail

L'espace de travail est situé dans l'objet vrml.

- vrml
 - o setBackgroundColor : définir la couleur de fond de l'objet
 - \$Rcolor : couleur rouge, valeur comprise entre 0 et 1
 - \$Gcolor : couleur vert, valeur comprise entre 0 et 1
 - \$Bcolor : couleur bleu, valeur comprise entre 0 et 1
 - o add : ajouter un élément à l'espace de travail
 - \$object : element à ajouter
 - o show : afficher l'espace de travail

ii. Formes

Après avoir fini de modifier la forme il faut l'ajouter à l'espace de travail.

- cube
 - o constructeur(utilisé avec new)
 - \$name : le nom du cube
 - \$xsize : taille en x
 - \$ysize : taille en y
 - \$zsize : taille en z
 - \$color : couleur (objet color)
 - o addTransformation : ajouter une transformation
 - \$transform : transformation
- cone
 - o constructeur(utilisé avec new)
 - \$name : le nom du cone
 - \$bottomRadius : rayon de la base
 - \$height : hauteur
 - \$side : affichage de la partie haute du cône, valeur : TRUE, FALSE
 - \$bottom : affichage de la base du cône, valeur : TRUE, FALSE
 - \$color : couleur (objet color)
 - o addTransformation : ajouter une transformation
 - \$transform : transformation

- cylinder
 - o constructeur(utilisé avec new)
 - \$name : le nom du cylindre
 - \$bottom : affichage de la base du cylindre, valeur : TRUE, FALSE
 - \$height : hauteur
 - \$radius : rayon de la base
 - \$side : affichage du milieu du cylindre, valeur : TRUE, FALSE
 - \$stop : affichage du haut du cylindre, valeur : TRUE, FALSE
 - \$color : couleur (objet color)
 - o addTransformation : ajouter une transformation
 - \$transform : transformation

Outil permettant de créer du sol:

- elevationGrid
 - o constructeur(utilisé avec new)
 - \$name : le nom de la grille
 - \$xDimension : nombre de colonnes
 - \$zDimension : nombre de lignes
 - \$xSpacing : espacement entre les points
 - \$zSpacing : espacement entre les points
 - \$height : tableau de tableau de l'altitude des points
 - \$color : couleur (objet color)
 - \$creaseAngle : angle par default à 1
 - o addTransformation : ajouter une transformation
 - \$transform : transformation

- sphere
 - o constructeur(utilisé avec new)
 - \$name : le nom de la sphere
 - \$radius : rayon
 - \$color : \$color : couleur (objet color)
 - o addTransformation : ajouter une transformation
 - \$transform : transformation

- text
 - o constructeur(utilisé avec new)
 - \$name : le nom du texte
 - \$string : texte
 - \$size : taille du texte
 - \$family : famille (par ex: "SANS")
 - \$justify : alignement (par ex: "MIDDLE")

- \$color : couleur (objet color)
- addTransformation : ajouter une transformation
 - \$transform : transformation

iii. Apparence

L'apparence est stocké dans l'attribut color de chaque objet. On peut soit passer juste une couleur ou passer un tableau comprenant une couleur et une texture.

- color
 - constructeur(utilisé avec new)
 - \$Rcolor : couleur rouge, valeur comprise entre 0 et 1
 - \$Gcolor : couleur vert, valeur comprise entre 0 et 1
 - \$Bcolor : couleur bleu, valeur comprise entre 0 et 1
 - \$colorname : le nom de la couleur
 - \$ambientIntensity : intensité ambiante, valeur comprise entre 0 et 1 par défaut à 0.200
 - \$shininess : luminosité, valeur comprise entre 0 et 1 par défaut à 0.200
 - \$transparency : transparence, valeur comprise entre 0 et 1 par défaut à 0
- texture
 - constructeur(utilisé avec new)
 - \$repeatS : valeur : TRUE, FALSE
 - \$repeatT : valeur : TRUE, FALSE
 - \$url : url de la texture

iv. Transformations

Les transformations servent à déplacer des objets dans l'espace de travail.

- rotation
 - constructeur(utilisé avec new)
 - \$radius : angle par défaut à 0
 - \$x : direction par défaut à 0
 - \$y : direction par défaut à 0
 - \$z : direction par défaut à 0

- translation
 - o constructeur(utilisé avec new)
 - \$x : déplacement par défaut à 0
 - \$y : déplacement par défaut à 0
 - \$z : déplacement par défaut à 0
- scale (multiplie les dimensions de l'objet par les coefficients x,y,z)
 - o constructeur(utilisé avec new)
 - \$x : coefficient multiplicateur par défaut à 0
 - \$y : coefficient multiplicateur par défaut à 0
 - \$z : coefficient multiplicateur par défaut à 0

v. Autres

Anchor vous permet de créer des liens vers des pages web dans des fichiers vrml. Anchor est un objet qui en contient d'autres. Il faut donc ajouter des objet à l'aide de la fonction addObject. Il faudra donc aussi ajouter l'objet Anchor, après lui avoir ajouté les différents objets, à l'espace de travail.

- Anchor
 - o constructeur(utilisé avec new)
 - \$url : url de la page
 - \$parameter : paramètres (par ex : target=Frame)
 - \$description : description du lien
 - o addObject
 - \$object : objet à ajouter.

Inline vous permet d'inclure un autre fichier vrml.

- inline
 - o constructeur(utilisé avec new)
 - \$name : nom
 - \$url : url du fichier
 - o addTransformation
 - \$transform : transformation à effectuer sur le fichier

NavigationInfo vous permet de gérer les paramètres de navigation.

- navigationInfo
 - o constructeur(utilisé avec new)
 - \$type : tableau de type par défaut vide. (valeurs possibles : WALK/ FLY/ EXAMINE/ NONE)
 - \$headlight : lampe allumé par défaut à "false"
 - o addType
 - \$type : type à ajouter.

PointLight permet d'afficher une lumière.

- pointLight
 - o constructeur(utilisé avec new)
 - \$xposition : position en x
 - \$yposition : position en y
 - \$zposition : position en z
 - \$radius : rayon de la sphère d'influence.
 - \$xattenuation : atténuation en x
 - \$yattenuation : atténuation en y
 - \$zattenuation : atténuation en z

L'objet transform permet de regrouper plusieurs objets en un seul.

- transform
 - o constructeur(utilisé avec new)
 - \$name : nom
 - o addTransformation
 - \$transform : transformation
 - o addChildren
 - \$children : objet enfant ou nom d'un objet(pour éviter de l'ajouter à chaque fois qu'on l'utilise)

View vous permet de créer des vues

- view
 - o constructeur(utilisé avec new)
 - \$name : nom de la vue
 - \$xposition : position en x
 - \$yposition : position en y
 - \$zposition : position en z
 - \$xorientation : coordonnées du vecteur orientation
 - \$yorientation : coordonnées du vecteur orientation
 - \$zorientation : coordonnées du vecteur orientation
 - \$angleorientation : angle de rotation

4. Conclusion

Nous avons pu voir ici qu'une petite partie de l'étendue des possibilités proposées par le vrml. Cette librairie, comme vous avez pu le remarquer vous sera très utile si vous voulez créer des objets, voir un monde en vrml. Cette librairie n'implémente que les fonctions « rudimentaires » du vrml, il est bien évident que de nouvelles fonctions seront développées à l'avenir.

5. Remerciements

Je souhaiterais remercier tout spécialement les sites qui m'ont permis d'apprendre le vrml :

- <http://www.web3d-fr.com>
- <http://grandm.free.fr>

Ainsi que le Labo Web d'avoir accepté ce sujet de projet.