

Rédacteurs : Pierre AUDISIO, Adrien MATHIEU

Référence document : TOPSPORT-VISU-TECH  
Contact : Cédric Dumas  
Cedric . Dumas @ emn . fr

Version : 1.0  
Révision : 9  
Date : 04/01/2007

Ce document est sous licence *CREATIVE COMMONS Attribution-NonCommercial-ShareAlike 2.0 France*  
<http://creativecommons.org/licenses/by-nc-sa/2.0/fr/>

## Projet OSE FI3 2006 ~ 2007



## TopSport Visualizer

version 1.0, 19/12/06

---

## Documentation technique

Équipe projet : Pierre AUDISIO  
Adrien MATHIEU  
Vivek RAMAVAJJALA  
Karan SARDA

Tuteur : Cédric DUMAS

# I. Présentation

*TopSport Visualizer* est composé de 5 packages :

1. configurator
2. main
3. messages
4. utilities
5. visualizer

Nous allons les décrire plus en détail.

Pour des informations sur la structure interne de chaque classe, vous pouvez consulter la Javadoc de *TopSport Visualizer* située dans le dossier `TopSport Visualizer/Documentations`.

## II. Package configurator

Il contient l'ensemble des classes utiles pour la fenêtre de configuration de l'affichage :

- **MainFrame** : fenêtre principale du logiciel. Elle contient les différents onglets d'affichage et donne accès aux menus du logiciel.
- **DisplayPanel** : un onglet d'affichage. Chaque onglet permet de configurer un affichage dans le cas où l'utilisateur a plusieurs écrans par exemple.
- **CompetitionPanel** : panneau de configuration de la compétition (choix de la compétition, description, option logo). C'est l'étape 1 de la fenêtre de configuration.
- **SportsPanel** : panneau de configuration des sports (choix des sports à afficher). C'est l'étape 2 de la fenêtre de configuration.
- **OptionsPanel** : panneau de configuration de l'affichage en général (choix du moniteur, option du diaporama, affichage d'une horloge et/ou de messages défilants). C'est l'étape 3 de la fenêtre de configuration.
- **Launcher** : est appelé lorsque l'utilisateur appuie sur le bouton `Afficher`. Permet d'instancier la fenêtre d'affichage des sports.
- **PreferencesFrame** : fenêtre des préférences du logiciel. Actuellement, seules les couleurs de la fenêtre de visualisation sont modifiables.
- **DBInformationsFrame** : fenêtre d'informations sur la base de données sélectionnée par l'utilisateur. Permet aussi de changer de base. Cette fenêtre est accessible dans le menu Base de données.
- **DisplayPanelActionListener** : gère les actions effectuées par l'utilisateur sur les composants de la fenêtre de configuration.
- **MenusActionListener** : gère les actions effectuées par l'utilisateur dans les menus de la fenêtre.

### III. Package main

Il contient la classe principale lancée au démarrage du logiciel et d'autres classes qui assurent la connexion à la base.

- **Main** : c'est la classe principale du logiciel appelée au démarrage. Elle affiche l'écran de bienvenue et une fenêtre de connexion à une base de données (BDAdmin).
- **BDAdmin** : permet à l'utilisateur de choisir la base de données qu'il veut charger.
- **Program** : cette classe assure la connexion et la récupération des paramètres de la base de données.

### IV. Package messages

Il contient toutes les classes nécessaires à la gestion des messages défilants.

- **AbstractMessage** : la classe abstraite qui contient les méthodes que toutes les classes définissant un message doivent étendre.
- **Message** : permet d'instancier le message selon qu'il s'agit d'un message prédéfini, ou bien un message créé par l'utilisateur.
- **clock** : message prédéfini. Construit une horloge.
- **MatchesInProgress** : message prédéfini. Liste tous les messages qui sont en train d'être joués.
- **Messages** : liste l'ensemble des messages prédéfinis ou non.
- **MessagesFrame** : fenêtre de configuration/ajout des messages.
- **MessageDialog** : fenêtre qui permet à l'utilisateur de modifier/ajouter/supprimer un message selon le choix de l'utilisateur.
- **MessagesPanel** : le panneau instancié à partir des messages sélectionnés par l'utilisateur. Est appelé par la classe `configurator.Launcher`.
- **Minutor** : est utilisé pour faire défiler les messages selon une période choisie par l'utilisateur (lente, normale ou rapide) et pour mettre à jour les messages (toutes les 60 secondes).

## V. Package utilities

Ce package contient un ensemble de classes variées qui sont utilisées par une ou plusieurs autres classes. Il permet de factoriser les classes logiquement.

- **GBC** : une classe qui simplifie l'utilisation des `GridBagConstraints` utilisés pour construire l'interface graphique de la fenêtre de configuration (la fenêtre de visualisation utilise le `BorderLayout`).
- **SystemClockThread** : construit une horloge qui se met à jour toutes les secondes et s'affiche notamment dans la fenêtre de configuration (`configurator.OptionsPanel`).
- **FilePreview** : panneau qui s'affiche dans le `JFileChooser` pour choisir le logo (`configurator.CompetitionPanel`).
- **ImageFilter** : filtre les fichiers dans le `JFileChooser` (`configurator.CompetitionPanel`) pour n'accepter que les GIF, JPG, JPEG et PNG.
- **Extension** : permet de connaître l'extension d'un fichier.
- **PreferencesAttributes** : liste et instancie les attributs que l'utilisateur peut modifier dans la fenêtre de préférences (`configurator.PreferencesFrame`).
- **ColorEditor** et **ColorRenderer** : utilisés dans la fenêtre de préférences (`configurator.PreferencesFrame`) pour l'affichage et la sélection des couleurs.
- **RAZEditor** et **RAZRenderer** : utilisés dans la fenêtre de préférences (`configurator.PreferencesFrame`) pour l'affichage et la remise à zéro des couleurs (valeurs par défaut).

## VI. Package visualizer

Il contient toutes les classes pour l'affichage et le défilement des sports.

- **VisualizerWindow** : construit la fenêtre générique de visualisation avec les options sélectionnées par le configurateur.
- **ArbreWindow** : fenêtre d'affichage d'arbre pour les tournois de type 1 ou 3, arbre simple ou poules + arbres. Entre 8 et 32 équipes. Appel à `ArbreMonophase` pour afficher étape par étape.
- **ArbreMonophase** : panneau permettant d'afficher une seule étape de l'arbre sous forme de rectangles.
- **Tree** : panneau permettant d'afficher un arbre. Fait appel à `TreeCell`, `TreePosition` et `TreeLink`.
- **TreeCell** : créé les cellules de l'arbres et les positionnent.
- **TreePosition** : utilisé par `TreeCell` pour positionner les cellules.
- **TreeLink** : lie les cellules de l'arbre entre elles.
- **Base Window** : fenêtre d'affichage des tournois de type « mono-poule » (type 2). Fait appel à `Table`.
- **Table** : panneau permettant d'afficher une table pour chaque groupe. Fait appel à `TableLabel`, `TableCell` et `TableEmptyCell`.

- **TableLabel** : crée les cellules qui légendent le tableau en contenant le nom des équipes en abscisse et en ordonnée.
- **TableCell** : crée les cellules à l'intérieur du tableau contenant les matchs du groupe.
- **TableEmptyCell** : crée des cellules vides pour le cas où il n'y pas de match retour dans le groupe.
- **MultiTablesWindow** : fenêtre d'affichage de la partie poules des tournois de type 3. Affiche une poule pour chaque groupe présent dans le tournoi.
- **Horloge** : déclenche un Thread affichant l'heure au format HH:MM:SS en temps réel.
- **WindowUtilities** : utilitaire de mis en page des fenêtres.