|  |  |
| --- | --- |
| **Dossier 2** | **Gestion de l'espace publicitaire** |

Annexes à utiliser : 1.a, 1.b, 1.c, 2.a, 2.b, 2.c

La société TELINOS lancera prochainement sa propre chaîne de télévision : « Téliplus ». Une partie des recettes provient de la vente de temps d’antenne pendant lesquels des messages publicitaires proposés par des annonceurs sont diffusés.

Un message publicitaire pourra être diffusé plusieurs fois par jour et ce pendant plusieurs jours. La journée de 24 heures est découpée par « Téliplus » en 6 tranches horaires : matin, midi, après midi, avant soirée, soirée, et nuit.

Un passage à l’antenne (prévu ou réalisé) d’un message un jour donné et dans une tranche particulière s’appellera, dans le contexte de l'étude, une diffusion.

La société TELINOS souhaitera mettre à la disposition des gestionnaires une exportation des informations concernant les messages publicitaires et leurs diffusions au format XML, comme l’illustre l’**annexe 2.b**.

La description des classes métiers est fournie en **annexe 1** : le diagramme de classes en **annexe 1.a** et la description textuelle en **annexe 1.b**. La classe Annonceur permettra de recenser tous les messages d'un annonceur. Un message peut faire l'objet de plusieurs diffusions.

La personne chargée des tests de la méthode *toXML()* de la classe Message a constaté que cette méthode ne donnait pas satisfaction :

* d’une part, la structure XML de l'information produite par cette fonction ne permet pas d'obtenir l'ensemble des données concernant un message.
* d’autre part, la liste des diffusions est incomplète.

Voici la version actuelle de la méthode toXML de la classe Message :

Fonction toXML() : chaîne

resultat : chaîne

i : entier *//variable de boucle*

nbDiffusions : entier

début

resultat "<message>"

// *le + indique la concaténation de chaîne de caractères*

resultatresultat + "<code>" + code

resultatresultat + "</code>"

resultatresultat + "<durée>" + duree.enChaine()

resultatresultat + "</durée>"

i 🡨 1

nbDiffusions🡨lesDiffusions.cardinal()

TantQue i <nbDiffusions Faire

resultat🡨resultat + lesDiffusions.extraireObjet(i).toXML()

i🡨 i+1

FTQ

resultat🡨resultat + "</message>"

retournerresultat

resultat🡨resultat + "<description>" + description

resultat🡨resultat + "</description>"

fin

// *la fonction enChaine()convertit un entier en chaîne de caractères.*

Remarque : La méthode *toXML()* de la classe Diffusion est correcte. Elle retourne la date et la tranche d’une diffusion.

|  |  |
| --- | --- |
|  | TRAVAIL À FAIRE |
| **2.1** | En observant le code fourni, expliquer les causes des dysfonctionnements actuels de la méthode *toXML()* de la classe Message.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Il manque la description |

La description du cas d'utilisation « Visualisation d'un message publicitaire au format XML » est proposée en **annexe 2.a**. Une maquette IHM (interface homme-machine) est fournie en **annexe 2.b** : deux des composants graphiques de la maquette sont nommés *zoneCode*et *vueXML* ; leur classe *ZoneEditionIHM* est décrite partiellement en **annexe 2.c**.

Lorsque le gestionnaire valide la saisie du code du message publicitaire, une procédure événementielle associée à l'événement *clic du bouton "Valider"* appelle une procédure nommée *afficherUnMessage()*permettant de visualiser le message publicitaire au format XML. L'entête de la procédure *afficherUnMessage()* est la suivante :

Procédure afficherUnMessage(unAnnonceur : Annonceur)

|  |  |
| --- | --- |
|  | TRAVAIL À FAIRE |
| **2.2** | Écrire la procédure *afficherUnMessage()* en prenant appui sur la description du cas d’utilisation. |

Actuellement, l'exportation XML de tous les messages ne satisfait pas le gestionnaire. Ce dernier souhaite, pour certains traitements, écarter les messages sans diffusion associée. Pour ce faire, il est décidé d'ajouter dans un premier temps la fonction *estDiffusée()* à la classe Message permettant de savoir si une instance de Message a au moins une diffusion associée.

|  |  |
| --- | --- |
|  | TRAVAIL À FAIRE |
| **2.3** | Fournir :   1. L'entête de la fonction *estDiffusée()* : nom de la fonction, nom(s) et type(s) des arguments éventuels, type de retour. 2. b) Le corps de cette fonction. |

La fonction précédente pourra être utilisée afin d'écrire dans la classe Annonceur une méthode qui retourne la liste des messages ayant des diffusions.

Voici l'entête de cette méthode :

Fonction messagesAyantDiffusions() : Collection de Message

// *retourne la liste des messages associés au moins à une diffusion*. *Cette liste*// *pourra être vide*

|  |  |
| --- | --- |
|  | TRAVAIL À FAIRE |
| 2.4 | Écrire la méthode *messagesAyantDiffusions()* de la classe Annonceur.  public Collection<Message> messageAyantDiffusion(){  Collection<Message> messages = new Collection <Message>();    for(Message unMessage: lesMessages){  if( unMessage.estDiffusé() ==True){  messages.ajouter(unMessage);  }  }  return messages ;  } |

Afin de planifier les diffusions, le gestionnaire désire obtenir la liste des messages ayant des diffusions, triée par durée. Le développeur chargé de cette tâche se propose d'ajouter une méthode *trierLesMessages()* dans la classe Annonceur ; cette méthode utilisera la méthode *trier()* de la classe Collection, qui s'appuie sur une méthode de comparaison qu'il faudra écrire dans la classe Message. Cette méthode a l’entête suivante :

*// Classe Message*

**Fonction compare(unMessage : Message) : entier**

// *retourne 0 si la durée du message courant est égale à la durée du message de l'objet unMessage*

// *retourne 1 si la durée du message courant est supérieure à la durée du message de l'objet   
// unMessage*

// *retourne -1 si la durée du message courant est inférieure à la durée du message de l'objet   
// unMessage*

|  |  |
| --- | --- |
|  | TRAVAIL À FAIRE |
| 2.5 | Écrire la méthode *compare()* de la classe Message.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  public int compare(unMessage Message){  int val=1  if(this.Durée==unMessage.durée){  val=0  }else{  if(this.durée==unMessage.durée){  val=-1  }  }  return val ;  } |

La méthode *trierLesMessages()* de la classe Annonceur est maintenant à même de trier les messages ayant des diffusions en appelant le service correspondant de la collection.

|  |  |
| --- | --- |
|  | TRAVAIL À FAIRE |
| 2.6 | Écrire la méthode *trierLesMessages()* de la classe Annonceur.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  public void trierLesMessages(){  Collection <Message> message = this.MessageAyantDiffusions() message.trier ;  } |

**Annexe 1 : Description des classes**

Annexe 1.a : Diagramme des classes métier



Annexe 1.b : Description textuelle partielle des classes métier

Classe Diffusion

Attributs privés :

date : Date

tranche : Chaîne

Méthode publique :

Fonction toXML() : Chaîne

FinClasse

Classe Message

Attributs privés :

code : Chaîne

description : Chaîne

duree : Entier

lesDiffusions : Collection de Diffusion

Méthodes publiques :

Fonction toXML() : Chaîne

Fonction getLesDiffusions() : Collection de Diffusion*//retourne l'attribut privé*  
 // lesDiffusions

FinClasse

Classe Annonceur

Attribut privé :

lesMessages : Collection de Message

Méthodes publiques :

Fonction getMessage(codeMessage : Chaîne) : Message

// retourne le message dont le code est passé en paramètre

Fonction existe(codeMessage : Chaîne) : Booléen

// retourne un booléen qui indique si le code passé en paramètre existe dans la // collection lesMessages

FinClasse

Annexe 1.c : Description de la classe technique Collection de <nom de la classe>

Cette classe permet d'enregistrer des objets d'une classe précisée dans la déclaration.

Exemples de déclaration :

lesDiffusions : Collection de Diffusion // contient des objets de type Diffusion

lesMessages : Collection de Message // contient des objets de type Message

**Classe Collection de <nom de la classe>** // Méthodes publiques

Fonction cardinal() : Entier

// Renvoie le nombre d'éléments de la collection

Fonction existe(e unObjet : Objet de la classe) : Booléen

// Teste si unObjet existe dans la collection

Fonction index(e unObjet : Objet de la classe) : Entier

// Renvoie l'index de unObjet, le premier objet de la collection a pour index 1

Fonction extraireObjet(e unIndex : Entier) : Objet

// Retourne l'objet d'index unIndex

Procédure ajouter(e unObjet : Objet de la classe)

// Ajoute un objet à la collection

Procédure enlever(e unIndex : Entier)

// Supprime l'objet d'index unIndex de la collection

Procédure vider()

// Vide le contenu de la collection

Procédure trier()

// trie les objets de la collection en utilisant la méthode compare présente dans la

// classe des objets de la collection

**FinClasse**

**Annexe 2**

Annexe 2.a : Cas d’utilisation

**Cas d'utilisation** : Visualisation d'un message publicitaire au format XML

**Acteur** : Gestionnaire

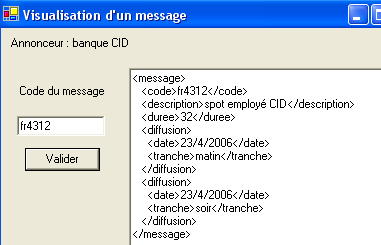
**Scénario typique**

1. Le système communique un formulaire de visualisation (voir maquette IHM).
2. Le gestionnaire soumet le code d'un message publicitaire au système.
3. Le système lui retourne une représentation XML de certaines de ses propriétés : code, description, durée, dates et tranches horaires de diffusion du message.

**Cas particuliers :**

Étape 3 : Le système retourne un message d'erreur si le code du message publicitaire n'est pas connu du système

Annexe 2.b : Exemple de maquette IHM



vueXML

zoneCode

Annexe 2.c : Description de la classe graphique

**Classe ZoneEditionIHM**

...

// Méthodes publiques

Fonction getText() : Chaîne

// rend le contenu textuel détenu par le composant

Procédure setText(nouveauTexte : Chaîne)

// change le contenu textuel du composant

...

**FinClasse**

**NB** : zoneCode et vueXML sont les identifiants des objets graphiques à utiliser pour récupérer ou affecter une valeur à ces zones.

Exemple d’utilisation : zoneCode.getText() ou vueXML.setText(« Code de message inconnu. ») ;