BASOL Nathan
LEDERREY Lussandre
MACCREZ Allan

SAE Vélos de Nantes
Partie R2.01

Explication du code SAE Velos de Nantes R2.01

Explication des méthodes rajoutées :

Dans nos 4 classes principales, nous avons à chaque fois rajouté une méthode toString() pour pouvoir afficher proprement l'objet.
- Classe AttributPorte : Ajout d'une méthode `String getNbPassagesTotal()` qui affiche le nombre de passages total qu'a enregistré le compteur sur une journée.
- Classe Compteur : Ajout d'une méthode `boolean closeTo(Compteur compteur)` qui compare la localisation de deux compteurs. Si la distance entre les deux est inférieure à 1km, alors on renvoie true sinon false.
- Classe Jour : Ajout d'une méthode `int getNombreJours()` qui renvoie le nombre de jours entre la date actuelle et la date du jour.
- Classe Quartier : Ajout d'une méthode `boolean equals(Quartier quartier)` qui compare deux quartiers et qui renvoit true si ce sont les mêmes et false sinon.
Ajout d'une seconde méthode `double conversionEnKilometre()` qui convertit la longueur de la piste en kilomètres.

Types des attributs :

Pour gérer le format de la date, nous avons simplement utilisé un objet LocalDate. A savoir que nous avons d'abord essayé d'utiliser un objet Date, mais celui-ci n'est plus utilisable depuis java 11.
Pour ce qui est des attributs nbVeloH00 … nbVeloH23. Nous avons préféré faire un tableau de 24 éléments ou chaque indice représente une heure. C'est beaucoup plus simple de faire cela plutôt que de faire 24 attributs différents.

Les associations :

Pour lier les classes entre elles, il a fallu créer des attributs objets dans certaines classes. Nous nous sommes basé sur le diagramme UML fournit pour le faire.
- Dans la classe Compteur, il y a un attribut leQuartier qui est un objet Quartier. Etant donné qu'un compteur peut être lié à 0 ou un seul quartier, nous avons choisis de faire deux constructeurs : un premier avec un paramètre leQuartier et un autre sans.
- Pour l'association du type * * entre les classes Compteur et Jour, nous avons choisis de créer les attributs leJour et leCompteur dans une classe AttributPorte. Cette classe contient également l'attribut presenceAnomalie.

Contraintes de domaines :

Pour gérer les contraintes explicitées sur le PDF, nous avons écrit une classe spéciale BonneValeur. Pour ce faire, nous avons stocké dans des tableaux de String les valeurs adéquates que peuvent prendre les attributs vacances, jour et presenceAnomalie. Nous avons codé 3 méthodes qui renvoient true si la valeur passée en paramètre est correcte et false sinon. Dans les constructeurs des classes, nous avons appelé ces méthodes à chaque fois pour vérifier que les valeurs rentrées sont bonnes.

Organisation du code :

Nous avons décidé d'organiser notre code en différents package pour une meilleure lisibilité. Dans le package modele, nous avons mis nos classes principales : Compteur, Jour, Quartier, AttributPorte. Dans le package utilitaires, nous avons mis notre classe BonneValeur. Enfin, nous avons mis notre classe TestCompteur et la classe Scenario dans un package test.

Modification du LocalDate : En réalisant les tests, nous avons remarqué que quand nous récupérons la valeur avec getDayOfWeek().toString(), ça nous renvoie les jours en anglais et en majuscule. Pour résoudre ce problème, nous avons crée une nouvelle méthode dans la classe BonneValeur qui transforme un jour anglais en un jour français (méthode String jourEnglishToFrench(String value))

Classe de test :

Nous avons testé toute la classe Compteur.

Classe de scenario :

Nous avons créé un scenario utilisant tous les objets et méthodes que nous avons codés.

Annexe : Le diagramme de classe et le diagramme de séquence

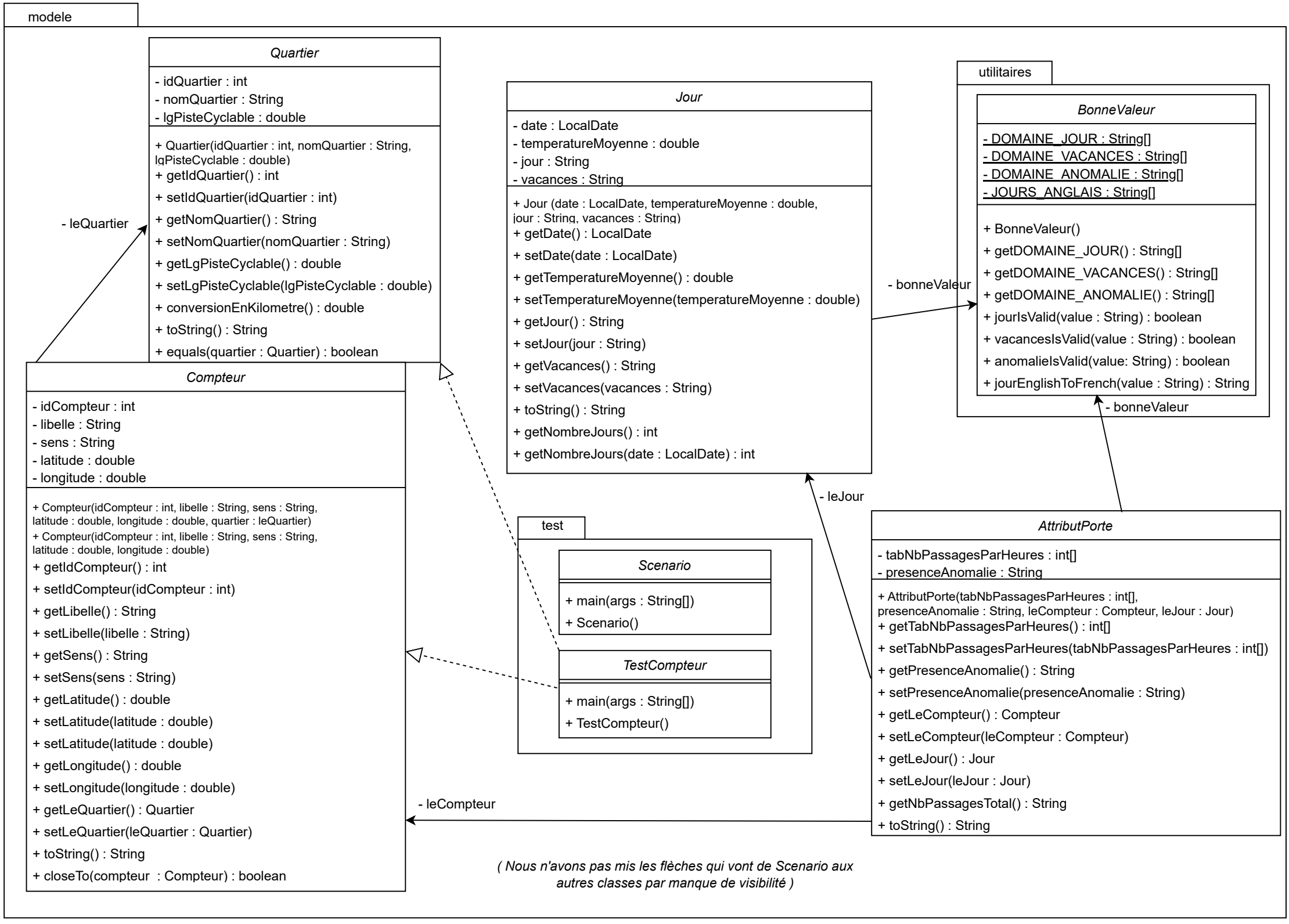
modele
Quartier
- idQuartier : int
- nomQuartier : String
- lgPisteCyclable : double
+ Quartier(idQuartier : int, nomQuartier : String, lgPisteCyclable : double)
+ getIdQuartier() : int
+ setIdQuartier(idQuartier : int)
+ getNomQuartier() : String
+ setNomQuartier(nomQuartier : String)
+ getLgPisteCyclable() : double
+ setLgPisteCyclable(lgPisteCyclable : double)
+ conversionEnKilometre() : double
+ toString() : String
+ equals(quartier : Quartier) : boolean
- leQuartier
Compteur
- idCompteur : int
- libelle : String
- sens : String
- latitude : double
- longitude : double
+ Compteur(idCompteur : int, libelle : String, sens : String, latitude : double, longitude : double, quartier : leQuartier)
+ Compteur(idCompteur : int, libelle : String, sens : String, latitude : double, longitude : double)
+ getIdCompteur() : int
+ setIdCompteur(idCompteur : int)
+ getLibelle() : String
+ setLibelle(libelle : String)
+ getSens() : String
+ setSens(sens : String)
+ getLatitude() : double
+ setLatitude(latitude : double)
+ setLatitude(latitude : double)
+ getLongitude() : double
+ setLongitude(longitude : double)
+ getLeQuartier() : Quartier
+ setLeQuartier(leQuartier : Quartier)
+ toString() : String
+ closeTo(compteur : Compteur) : boolean
Jour
- date : LocalDate
- temperatureMoyenne : double
- jour : String
- vacances : String
+ Jour (date : LocalDate, temperatureMoyenne : double, jour : String, vacances : String)
+ getDate() : LocalDate
+ setDate(date : LocalDate)
+ getTemperatureMoyenne() : double
+ setTemperatureMoyenne(temperatureMoyenne : double)
+ getJour() : String
+ setJour(jour : String)
+ getVacances() : String
+ setVacances(vacances : String)
+ toString() : String
+ getNombreJours() : int
+ getNombreJours(date : LocalDate) : int
- leJour
utilitaires
BonneValeur
- DOMAINE_JOUR : String[]
- DOMAINE_VACANCES : String[]
- DOMAINE_ANOMALIE : String[]
- JOURS_ANGLAIS : String[]
+ BonneValeur()
+ getDOMAINE_JOUR() : String[]
+ getDOMAINE_VACANCES() : String[]
+ getDOMAINE_ANOMALIE() : String[]
+ jourIsValid(value : String) : boolean
+ vacancesIsValid(value : String) : boolean
+ anomalieIsValid(value: String) : boolean
+ jourEnglishToFrench(value : String) : String
- bonneValeur
- bonneValeur
test
Scenario
+ main(args : String[])
+ Scenario()
TestCompteur
+ main(args : String[])
+ TestCompteur()
AttributPorte
- tabNbPassagesParHeures : int[]
- presenceAnomalie : String
+ AttributPorte(tabNbPassagesParHeures : int[], presenceAnomalie : String, leCompteur : Compteur, leJour : Jour)
+ getTabNbPassagesParHeures() : int[]
+ setTabNbPassagesParHeures(tabNbPassagesParHeures : int[])
+ getPresenceAnomalie() : String
+ setPresenceAnomalie(presenceAnomalie : String)
+ getLeCompteur() : Compteur
+ setLeCompteur(leCompteur : Compteur)
+ getLeJour() : Jour
+ setLeJour(leJour : Jour)
+ getNbPassagesTotal() : String
+ toString() : String
- leCompteur
(Nous n'avons pas mis les flèches qui vont de Scenario aux autres classes par manque de visibilité)

Diagramme de classe de la méthode toString() de la classe AttributPorte