predictif Documentation

Version 0.1

H. Verlin - S. Toko

03 April 2015

Table des matières

Prédict'IF est un cabinet de voyance en ligne, qui envoie périodiquement des email à ses clients. Ces emails contiennent des horoscopes personnalisés créés par les employés de l'entreprise. Ceux-ci se basent sur le signe du client, et sur le choix des voyants qu'il a séléctionné lors de son inscription.

Les services présentés ici permettent a création des différentes briques de l'application Prédict'IF, c'est-à-dire à la réalisation des différentes IHM.

L'ensemble des services fournis permet la réalisation de deux IHM :

— Une IHM web pour les clients, qui leur permet de s'inscrire pour recevoir des horoscopes
— Une IHM web pour les employés, pour la réalisation des horoscopes

Vous trouverez dans cette documentation :

1. Le modèle du domaine avec un diagramme de classe UML
2. La description des services
3. La maquette des 3 IHM (président, employés, clients)

Astuce : Pour une lecture plus agréable de la doc aller sur http://predictif-doc.readthedocs.org/

CHAPITRE 1

Modèle du domaine

1.1 Liste des entity

L'analyse des besoins concernant cette application nous a permis de définir ces différentes classes :

Nom	Fonction
Client	Client de predict'if
Employe	Employe de Predict'if
Medium	Medium fictif
Horoscope	Horoscope contenant un ensemble de 3 prédictions de type différent
Prédiction	Classe abstraite qui sert de base pour les 3 types de prédictions
PrédictionAmour	Prédiction de catégorie Amour, indique un signe partenaire
PrédictionTravail	Prédiction de catégorie Travail
PrédictionSante	Prediction de catégorie Santé, contient un conseil pour le client
Signe	Réprésente un signe astrologique simplifié (correspond à un mois de l'année)

L'ensemble des classes dispose d'une méthode `toString()` pour permettre son affichage.

1.2 Diagramme de Classes

CHAPITRE 2

Description des services

2.1 Méthodes de classe `Services`

Méthode	Type de Retour	Brève Description
creerClient(Client client)	int	Permet d'insérer un client dans la base de données
creerEmploye(Employe employe)	void	Permet d'insérer un employé dans la base de données
creerHoroscope(Horoscope horoscope, Client client)	void	Permet d'insérer un horoscope dans la base de données, et de l'attacher à un client. Elle simule l'envoi du mail au client
creerMedium(Medium medium)	void	Permet d'insérer un signe dans la base de données
creerPredictionAmour(PredictionAmour predictionAmour)	void	Permet d'insérer une prédiction de type amour dans la base de données
creerPredictionSante(PredictionSante predictionSante)	void	Permet d'insérer une prédiction de type santé dans la base de données
creerPredictionTravail(PredictionTravail predictionTravail)	void	Permet d'insérer une prédiction de type travail dans la base de données
creerSigne(Signe signe)	void	Permet d'insérer un signe dans la base de données
getClientById(long id)	Client	Récupère un client grâce à son id
getEmployeById(long id)	Employe	Récupère un employe grâce à son id
getSigneById(long id)	Signe	Récupère un signe grâce à son id
getPredictionById(long id)	Prediction	Récupère un prediction grâce à son id
getHoroscopeById(long id)	Horoscope	Récupère un horoscope grâce à son id
getMediumById(long id)	Medium	Récupère un medium grâce à son id
listerClients()	List<Client>	Liste de toutes les clients de la base
listerHistoriqueClient(Client client)	List<Horoscope>	Renvoie l'historique de l'horoscope d'un client
listerMediums()	List<Medium>	Liste de tous les mediums de la base
listerMediumsClient(Client client)	List<Medium>	Liste tous les mediums d'un client
listerPredictionAmours()	List<PredictionAmour>	Liste de toutes les predictions amour de la base
listerPredictionSantes()	List<PredictionSante>	Liste de toutes les predictions sante de la base
listerPredictionTravails()	List<PredictionTravail>	Liste de toutes les predictions travail de la base
listerSignes()	List<Signe>	Liste de toutes les signes de la base
updateEmp(Employe emp)	void	Met à jour un employé dans la base
connexionEmploye(String identifiant, String mdp)	Employe	Connexion d'un employé
listerMediumsClient(Client client)	List<Medium>	Liste tous les mediums d'un client

2.2 Exemples d'utilisation

2.2.1 Créer un client

Astuce : La class Calendar est utilisée pour la gestion des dates.

```
// on imagine que le client est né aujourd'hui
Calendar DateNaissance = Calendar.getInstance();
Client c = new Client("Nom", "Prénom","Mr", DateNaissance,
        "rue de nullepart, 69 100 If_laville", "client@if.fr","06 26 30 29 ");
services.creerClient(c);
```

2.2.2 Connexion d'un employé

```
Employe employe = services.connexionEmploye(identifiant, mdp);
```

2.2.3 Lister tous les clients d'un employé

```
Employe employe = services.connexionEmploye(identifiant, mdp);

for(Client element : employe.getClients() )
{
    System.out.println(element);
}
// On peut vérifier qu'il a bien des clients
if(choices.isEmpty())
{
    System.out.println("Vous n'avez pas de clients !");
}
```

Astuce : La service *services.listerHistoriqueClient(client)* permet de récupérer un historique préformaté. Pour plus de souplesse, vous pouvez directement utiliser la liste des horoscopes du client, ou encore utiliser les attributs de la classe *horoscope*.

2.2.4 Afficher l'historique des prédictions

```
    // Première Méthode
    List<Horoscope> historique = services.listerHistoriqueClient(client);

    //2ème méthode
boolean vide = true;
for(Horoscope h : client.horoscopes)
{
    if(h != null)
    {
        vide = false;
        System.out.println("_____________");
        System.out.println("\n\nLe "+
        h.getDateHoroscope().get(Calendar.DAY_OF_MONTH)+ " "
                    + ""+h.getDateHoroscope().getDisplayName(
                    Calendar.MONTH,Calendar.LONG,Locale.FRANCE)+""
                    + " "+h.getDateHoroscope().get(Calendar.YEAR));
        System.out.println(h);
        System.out.println("\n\n By "+h.getNomMedium());
```

```
        System.out.println("_____________");
    }
}
if(vide)
{
    System.out.println("Historique vide");
}
```

2.2.5 Ajouter une prédiction à un horoscope

```
    // on liste une catégorie de prédictions
List<PredictionAmour> pAmour = services.listerPredictionAmours();
Horoscope horoscope = new horoscope();
horoscope.setAmour(pAmour.get(0)); // si on veut la première de la liste
```

2.2.6 Mettre la date de l'horoscope et l'ajouter à un client

```
Calendar date = Calendar.getInstance();
horoscope.setDateHoroscope(date);
services.creerHoroscope(horoscope, client);
```

CHAPITRE 3

Présentation des maquettes