5

Réseau, Internet et services web

Dans un monde de plus en plus connecté, faire communiquer une application avec Internet devient normal et très fréquent. C'est ce que nous allons apprendre à faire dans ce chapitre. En pratique, nous allons apprendre à télécharger des fichiers, à uploader des fichiers en ligne, à communiquer avec un service web quelconque et nous terminerons en apprenant comment échanger des données avec une base de données distante. À l'instant où ces lignes sont écrites, Codename One ne supporte que les connexions HTTP et HTTPS.

Attention *› Il est conseillé d'avoir des notions sur le fonctionnement du protocole HTTP pour tirer profit de ce chapitre.*

Note *› Dans certaines sections de ce chapitre, vous aurez besoin d'utiliser un langage de développement web pour écrire les codes qui devront s'exécuter côté serveur. Le langage choisi ici est le PHP et pour l'utiliser vous aurez besoin d'installer un serveur web et une base de données. Pour cela, vous pourrez télécharger et installer l'un des logiciels suivants : EasyPHP, Wamp, Xampp. Si vous êtes plutôt fan d'un autre langage web alors vous savez les outils à installer.*

5.1. Gestion de la connexion

NetworkManager et ConnectionRequest sont les deux classes principales de manipulation du réseau en Codename One. La première permet de gérer les requêtes de connexion et la seconde de créer ces requêtes. La plupart du temps, un traitement utilisant le réseau est placé dans un thread parallèle à celui de l'interface graphique. Ainsi, l'un des problèmes les plus fréquents en programmation réseau est la gestion de ces threads multiples. En Codename One, la classe NetworkManager crée et gère automatiquement les threads en parallèle au thread de l'EDT. Elle nous épargne ainsi certaines contraintes liées à la gestion manuelle de ces threads consacrés au traitement du réseau.

La création d'une requête est gérée par la classe ConnectionRequest et chaque requête peut être envoyée de manière synchrone ou asynchrone. Les erreurs de traitement sont gérées par défaut, mais il est possible de les gérer manuellement en utilisant des listeners. Une fois des requêtes de connexion créées avec ConnectionRequest, il faut les ajouter à une file d'attente avec NetworkManager, qui se chargera non seulement de les exécuter, mais aussi de créer pour chacune des requêtes un thread. L'envoi de la requête, la destruction du thread réseau, etc. sont aussi gérés par NetworkManager.

Dans le cas de la communication avec un web qui peut renvoyer des données de type XML ou JSON, Codename One fournit des classes (XMLParser et JSONParser) pour la manipulation de ces formats.

La création d'une requête basique pourrait ressembler à ceci :

```
ConnectionRequest requete=new ConnectionRequest(); ❶
requete.setUrl("ADRESSE"); ❷
requete.setPost(false); ❸
requete.setContentType("TYPE MIME DU CONTENU"); ❹
requete.addArgument("NOM d'UN ARGUMENT", "VALEUR DE L'ARGUMENT"); ❺
requete.setPriority(ConnectionRequest.PRIORITY_NORMAL); ❻

requete.addResponseListener(new ActionListener() { ❼

     public void actionPerformed(ActionEvent evt) {
        //Récupération et traitement des données ici
     }
});

NetworkManager.getInstance().addToQueue(requete); ❽
```

❶ Création de la requête avec une instanciation de ConnectionRequest.

❷ La méthode `setUrl()` permet de définir l'adresse qui servira à la requête.

❸ La valeur `false` de la méthode `setPost()` précise que nous voulons récupérer des données et non en envoyer (valeur `true`). Il s'agit du type de méthode HTTP à utiliser (`GET` ou `POST`).

❹ `setContentType()` permet de définir le type MIME de la donnée.

❺ `addArgument()` permet d'ajouter des arguments à la requête. En premier paramètre le nom de l'argument et en deuxième paramètre sa valeur.

❻ La définition de la priorité de la requête dans la file d'attente se fait avec `setPriority()`. Ici, nous précisons une priorité normale avec la valeur `PRIORITY_NORMAL`. Les autres valeurs possibles sont : `PRIORITY_HIGH` (priorité haute), `PRIORITY_LOW` (priorité basse), `PRIORITY_CRITICAL` (priorité critique). Cette dernière priorité est supérieure à `PRIORITY_HIGH`, `PRIORITY_REDUNDANT` (priorité redondante). Elle permet d'écarter de la liste d'attente les éléments redondants mis en pause. Par défaut, la priorité normale est utilisée quand aucune priorité n'est définie.

❼ `addResponseListener()` nous permet de connecter la requête à un ActionListener pour la récupération et le traitement de la réponse renvoyée par la requête. Il

existe un autre moyen pour faire la même action de manière plus détaillée que nous verrons à l'Exemple 5.1.

❽ La méthode `addToQueue()` permet d'ajouter la requête créée à une file d'attente d'exécution créée par NetworkManager.

Téléchargement de données

Dans cette section, nous allons voir plusieurs manières de télécharger des données sur Internet : à l'aide des classes NetworkManager et ConnectionRequest, avec les classes ImageDownloadService et URLImage et enfin en recourant à certaines méthodes de la classe Util.

Avec NetworkManager et ConnectionRequest

Nous allons maintenant passer à des exemples concrets de création de requêtes et de récupération des réponses. Le premier exemple nous permettra de télécharger en ligne un fichier MP3, d'enregistrer ce fichier sur le téléphone et de le jouer ensuite. Nous allons aussi apprendre une autre façon de récupérer des données renvoyées par la requête sans utiliser la méthode `addResponseListener()`.

Exemple 5.1 : *Téléchargement d'un fichier MP3 et lecture du fichier téléchargé*

```
public class TestCN1Reseau implements ActionListener {

  private Form current;
  private Button telechargerMusique;

  public void init(Object context) {
      try {
          Resources theme=Resources.openLayered("/theme");
          UIManager.getInstance().setThemeProps(
          theme.getTheme(theme.getThemeResourceNames()[0]));
      } catch(IOException e){
          e.printStackTrace();
      }
  }

  public void start() {
      if(current!=null){
          current.show();
          return;
      }

      Form f=new Form("Test de téléchargement");
      f.setLayout(new BoxLayout(BoxLayout.Y_AXIS));
       ❶
      telechargerMusique=new Button("Télécharger la musique");
```