1
Démarrage

Maintenant que vous savez en quoi Codename One peut vous aider à créer la prochaine application mobile à succès, nous allons démarrer. Dans ce premier chapitre, vous apprendrez à installer le plug-in dans votre environnement de développement Java préféré. Vous écrirez une première application de test après avoir vu la structure d'une application Codename One. Enfin, vous découvrirez le processus de compilation particulier des applications Codename One et le mode de fonctionnement du simulateur inclus dans le plug-in du framework.

1.1. Téléchargement et installation du plug-in

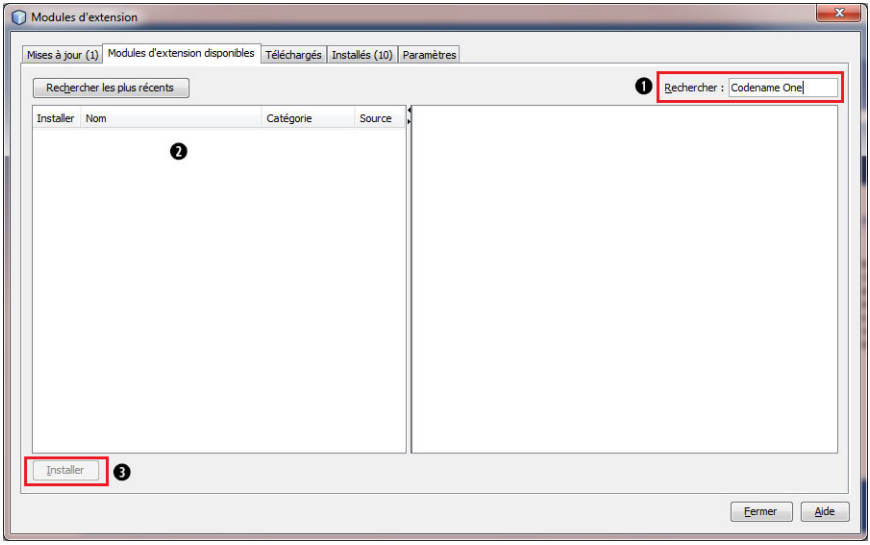
Le téléchargement du plug-in se fait directement en ligne. L'installation diffère selon les différents environnements de développement, mais reste simple que vous utilisiez NetBeans, Eclipse ou IntelliJ IDEA. Pour information, le plug-in de NetBeans a souvent la priorité sur les autres en ce qui concerne les mises à jour et c'est l'une des raisons pour lesquelles c'est l'environnement qui a été choisi pour les exemples de cet ouvrage.

Sous NetBeans

Attention › *Vous devez avoir la version 7.x ou version supérieure de NetBeans pour pouvoir installer et utiliser le plug-in.*

Lancez NetBeans. Allez dans le menu OUTILS, choisissez l'entrée MODULES D'EXTENSION puis dans la zone RECHERCHER, saisissez *Codename One* (voir ❶) puis lancez la recherche. Une fois le plug-in trouvé et affiché dans la zone ❷, sélectionnez-le et cliquez sur le bouton INSTALLER ❸ pour lancer l'installation. Suivez ensuite les instructions.

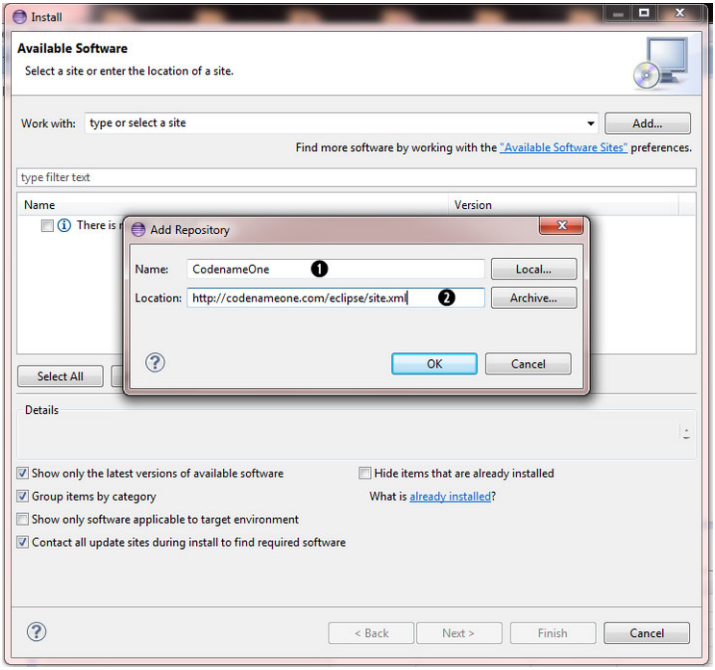
Figure 1.1 : Installation sous NetBeans



Sous Eclipse

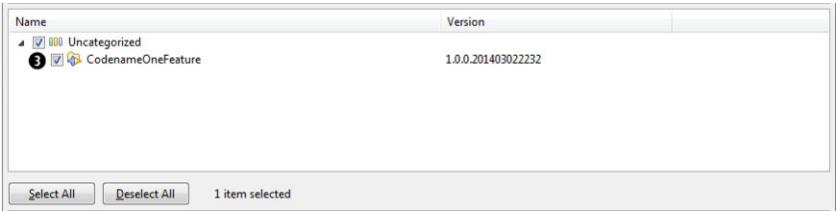
Lancez Eclipse et cliquez sur le menu HELP puis sur INSTALL NEW SOFTWARE (voir Figure 1.2). Cliquez sur le bouton ADD. Entrez ensuite *CodenameOne* (ou ce que vous voulez) dans la zone NAME ❶ et dans la zone LOCATION ❷, entrez ou copiez-collez l'url suivante : `http://codenameone.com/eclipse/site.xml`.

Figure 1.2 : Installation sous Eclipse



Sélectionnez le plug-in proposé ❸ (voir Figure 1.3) et validez pour poursuivre l'installation.

Figure 1.3 : Installation sous Eclipse (suite)



Sous IntelliJ IDEA

Attention *› Vous devez avoir la version 12.X ou version supérieure de IDEA pour pouvoir installer et utiliser le plug-in.*

Lancez IDEA et sur l'interface de bienvenue, cliquez sur CONFIGURE puis sur PLUGINS. Sur l'interface qui s'affichera, cliquez sur le bouton BROWSER REPOSITORIES pour ouvrir une nouvelle fenêtre. Dans la zone de recherche de cette fenêtre, entrez *Codename One*. Une fois le plug-in trouvé, faites un clic droit sur son nom puis choisissez DOWNLOAD AND INSTALL. Enfin, confirmez l'installation et le reste se fera automatiquement.

1.2. Structure d'une application Codename One

Comme dans chaque langage ou pour certains frameworks, un programme Codename One a aussi sa structure. Cette structure correspond au cycle d'exécution d'une application écrite avec ce framework.

```
public class NomDeLaClassePrincipale  {

  public void start() {  ❶
    //-- Code à exécuter au démarrage de l'application--
  }

  public void stop() {  ❷
   //-- Code à exécuter avant la mise en pause de l'application --
  }

  public void destroy() {  ❸
   //-- Code à exécuter avant la fermeture de l'application--
  }

 }
```

Note *› Si vous avez déjà créé des applications J2ME, vous remarquerez alors que les méthodes abstraites ci-dessus ressemblent à celles qui sont nommées* `startApp()`, `pauseApp()` *et* `destroyApp()`.

Considérez cette classe principale comme la classe d'un projet Java classique contenant la méthode `main()`. Voici le rôle des trois méthodes de cette classe.

❶ La méthode `start()` est appelée pendant le démarrage de l'application et aussi après la reprise d'une application mise en pause. Cette méthode est le point d'entrée de votre programme. Elle peut être comparée à la méthode `main()` d'un programme Java classique.

❷ La méthode `stop()` est appelée à chaque fois qu'une application est mise en pause. Cela peut survenir lorsque l'utilisateur sort par exemple de votre applica-

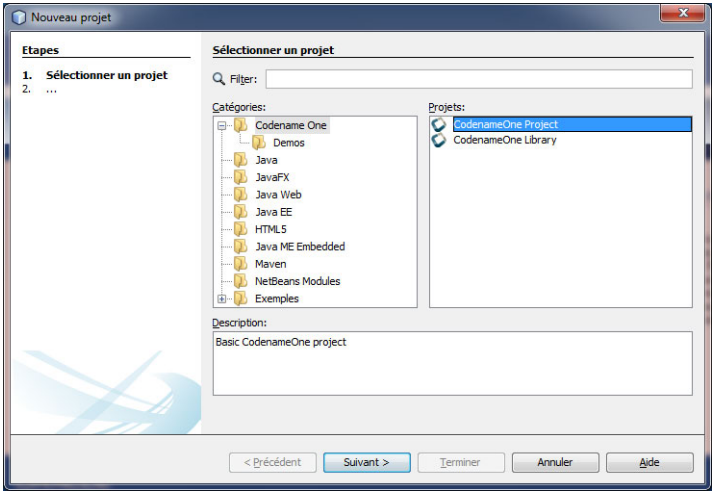
tion pour y revenir après, quand il reçoit un appel qui placera l'application en tâche de fond, etc. Sur certaines plateformes (iOS par exemple), l'état courant de l'application n'est pas sauvegardé par défaut quand elle est mise en pause. Cela a pour conséquence qu'elle redémarrera complètement lorsque l'utilisateur y reviendra. Pour éviter ce genre de comportement, vous pouvez sauvegarder dans une variable placée dans cette méthode l'interface courante de l'application pour pouvoir la réafficher lorsque l'utilisateur y retournera. Dans le cas d'un jeu, vous pouvez aussi utiliser cette méthode pour faire des sauvegardes et mettre le jeu en état de pause.

❸ La méthode `destroy()` est appelée avant la fermeture de votre application. Si vous avez des instructions à faire exécuter avant l'arrêt de l'application, mettez alors ces instructions dans cette méthode.

1.3. Hello Codename One

Il est temps de passer à la création d'une première application avec Codename One (le fameux *Hello World*) et c'est ce que nous allons faire dans cette section en créant d'abord un nouveau projet. Lancez NetBeans et cliquez sur NOUVEAU PROJET. Choisissez la catégorie CODENAME ONE puis sélectionnez CODENAMEONE PROJECT.

Figure 1.4 : *Création d'un nouveau projet (étape 1)*

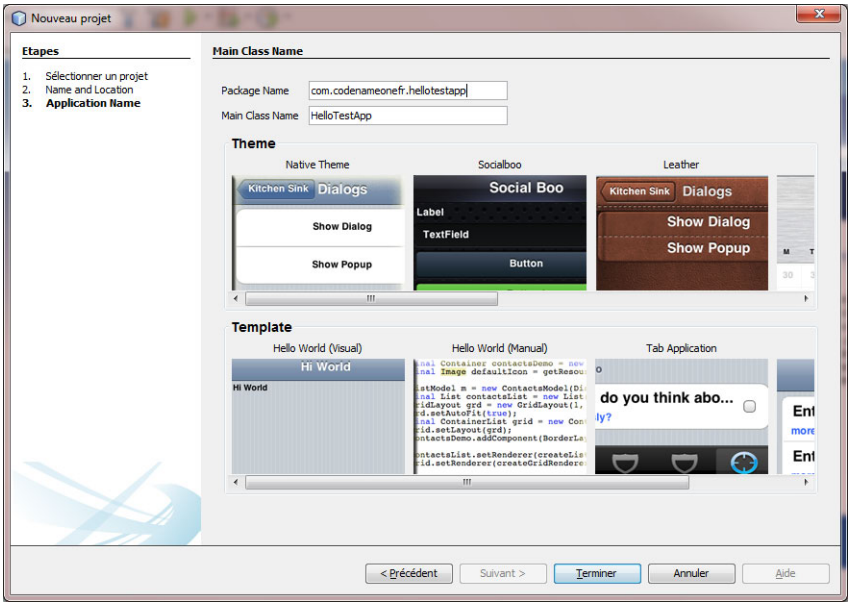


Note *› À la différence des menus de votre environnement de développement, qui seront en français si vous utilisez la version française de l'IDE, ceux ajoutés par le plug-in de Codename One ne sont disponibles qu'en anglais. Dans le cas de cet ouvrage, il s'agit de NetBeans.*

Cliquez sur SUIVANT et nommez le projet, par exemple `HelloTestApp`. Faites encore SUIVANT et à la dernière étape, entrez `com.codenameonefr.hellotestapp` comme nom du package et `HelloTestApp` comme nom de la classe principale qui contiendra le code de la structure de l'application. Dans la zone THEME, choisissez le thème NATIVE THEME en cliquant dessus puis HELLO WORLD (MANUAL) au niveau de la zone TEMPLATE.

Cliquez enfin sur TERMINER pour créer le projet (voir Figure 1.5).

Figure 1.5 : *Création d'un nouveau projet (étape 2)*



Explication des choix effectués

Le choix de NATIVE THEME permet au visuel de l'application de changer et de s'adapter automatiquement aux différentes plateformes. Ainsi, au lieu d'avoir un design unique de l'application sur les différentes plateformes, le design adoptera le rendu visuel natif propre à chaque plateforme.

Le choix du HELLO WORLD (MANUAL) nous permettra de créer manuellement l'interface graphique de notre application de test avec du code sans recourir à l'éditeur graphique de Codename One. C'est important parce que la structure d'un code utilisant l'éditeur graphique et celle d'un code écrit à la main diffèrent à certains niveaux. Pour utiliser l'éditeur graphique d'interface, vous devez choisir HELLO WORLD (VISUAL). Les autres modèles de la zone TEMPLATES utilisent aussi la structure basée sur l'éditeur graphique mais nous n'allons pas commencer par ça donc gardons le choix du HELLO WORLD (MANUAL).

Le code généré par la création du projet sera semblable à celui-ci :

```
public class HelloTestApp {

 private Form current;

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

 public void start() {
    if(current != null){
       current.show();
       return;
    }
    Form hi=new Form("Hi World"); ❷
    hi.addComponent(new Label("Hi World")); ❸
    hi.show(); ❹
 }

 public void stop() {
   current = Display.getInstance().getCurrent(); ❺
 }

 public void destroy() {
 }
}
```

Avant de passer à l'explication du code généré, nous allons d'abord lancer son exécution et voir ce que ça donne dans le simulateur. Pour cela, allez sur la barre d'outils de NetBeans ou dans le menu EXÉCUTER et cliquez sur EXÉCUTER PROJET. L'exécution du code affiche le simulateur avec l'interface de l'iPhone par défaut (voir Figure 1.6).

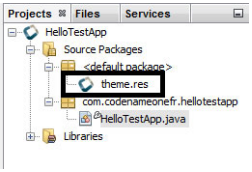
Figure 1.6 : Exécution dans le simulateur

Changeons la plateforme pour voir ce que le rendu donnera sous une autre plateforme. Pour cela, allez dans le menu SKINS du simulateur et cliquez sur le nom d'une autre plateforme (NEXUS par exemple pour afficher le rendu des versions 2.x.x d'Android).

Décortiquons maintenant le code généré par NetBeans en commençant par la méthode `init()`. Le code à l'intérieur de cette fonction (voir ❶) permet d'abord de charger le fichier de ressources qui est un fichier d'extension `.res` (visible dans votre projet) et qui contient le thème de l'application. Ce thème est ensuite défini comme thème par défaut. Il est important de le spécifier, car plusieurs thèmes peuvent être présents dans le fichier de ressources. Les détails sur le code à l'intérieur de cette fonction seront vus au Chapitre Codename One Designer. Pour l'instant, gardez-le tel quel et ne modifiez rien à cet endroit.

Note *› Un fichier de ressources (voir l'encadré sur la Figure 1.7) est un fichier qui accompagne tous les projets Codename One. Il sera toujours intégré dans l'exécutable de vos applications. Il peut contenir diverses choses comme le thème de l'application, les textes de traduction dans d'autres langues, des images, des fichiers textes, des polices de caractères et bien d'autres choses que nous verrons au chapitre Codename One Designer.*

Figure 1.7 : Aperçu du fichier de ressources dans le projet



Pour revenir à la méthode `init()`, considérez-la comme un constructeur et faites-y des initialisations si vous voulez.

Passons maintenant au contenu de la méthode `start()` qui représente le point d'entrée du programme.

❷ La classe Form permet de créer une page ou l'équivalent d'une fenêtre d'une application desktop. Vous pouvez y placer des composants graphiques comme des boutons et autres. Dans notre code, le constructeur de cette classe prend en paramètre un texte (*Hi World*) qui est le titre de la fenêtre (ou de la page).

❸ Cette ligne permet de créer un Label affichant le texte *Hi World* (ce composant permettant d'afficher du texte, de l'image ou les deux) et de l'insérer à l'intérieur de notre page avec la méthode `addComponent()` de la classe Form.

❹ Cette dernière ligne permet d'afficher notre page à l'écran.

Place maintenant au contenu de la méthode `stop()` avec l'explication du point ❺. Dans notre classe, un Form nommé `current` est déclaré en tant que variable d'instance. Cette variable permet de récupérer la page courante (celle qui s'affiche à l'écran). Cette récupération se fait avec la méthode `getCurrent()` qui se trouve à l'intérieur de la classe Display (qui est un singleton). La question est de savoir pourquoi nous récupérons la page courante ! Comme mentionné à la Section 1.2, Structure d'une application Codename One, la méthode `stop()` permet d'effectuer une action avant que l'application ne se mette en pause. Ceci nécessite une petite explication.

Sous des plateformes comme iOS et BlackBerry OS, quand vous réduisez une application pour effectuer une autre action comme recevoir un appel par exemple, votre application est automatiquement mise en pause. Quand vous revenez dessus, vous constaterez que l'application redémarre et ça peut être embêtant si vous étiez en train de faire quelque chose d'important. C'est pour cette raison qu'avant que l'application ne soit mise en pause, le code à l'intérieur de cette méthode récupère la page courante pour la sauvegarder dans une variable. Quand vous reviendrez sur l'application après, l'appareil fera appel de nouveau à la fonction `start()`. Ainsi, le code au début de cette

méthode vérifie d'abord si la valeur de `current` est nulle. Si ce n'est pas le cas alors la page sauvegardée est affichée.

1.4. La compilation avec Codename One

L'une des particularités de Codename One est l'utilisation du cloud pour la compilation. Ce choix a ses avantages et ses inconvénients. Même en utilisant un framewok multiplateforme, il est normal d'installer sur son ordinateur les kits de développement natif de chacune des plateformes à cibler. Cela veut dire que dans le cas de Codename One, vous devriez normalement installer cinq SDK avant de pouvoir compiler vos applications pour les cinq plateformes supportées. Heureusement, nous n'avons pas à le faire à cause de la présence du cloud qui a été mise à disposition pour ça.

Note *› Contrairement à ce que pourraient penser certains, ce ne sont pas les codes sources de vos programmes qui sont envoyés sur le serveur de Codename One mais plutôt les bytecodes (les fichiers d'extension .class) générés à partir des classes de vos programmes par le compilateur de Java.*

Avantages de la compilation dans le cloud :

- être épargné de l'installation et des paramétrages des divers SDK natifs sur son ordinateur ;
- ne pas avoir à apprendre à utiliser les émulateurs ou simulateurs fournis avec chaque SDK ;
- ne pas avoir à acheter ou à utiliser un Mac pour compiler pour la plateforme iOS si on est un utilisateur de Windows ou de Linux ;
- ne pas avoir à acheter ou à utiliser un ordinateur Windows pour compiler pour la plateforme BlackBerry et Windows Phone si on est un utilisateur de Mac ou de Linux.

Inconvénients de la compilation dans le cloud :

- ne pas pouvoir compiler sur son propre ordinateur, ce qui serait un gain de temps considérable ;
- la nécessité de faire appel parfois à des outils des SDK natifs pour effectuer certains débogages particuliers.

Note *› L'accès au cloud de Codename One peut être gratuit comme payant. Tout au long de ce livre, nous utiliserons un compte gratuit qui donne accès à la grande majorité des fonctionnalités dont nous aurons besoin. Un compte gratuit inclut 100 crédits de compilation par mois. Chaque compilation (pour Android, Blackberry, Windows phone, J2ME) vous enlèvera 1 crédit sauf la compilation iOS qui enlèvera 8 crédits. Ceci s'explique par le coût élevé de l'hébergement d'un Mac OS dans le cloud. Un tableau contenant la différence entre un compte utilisateur gratuit et un compte utilisateur payant est accessible sur le site de Codename One.*

Avant d'effectuer une compilation, vous devez d'abord vous rendre sur le site officiel de Codename One pour y créer un compte gratuit. Ce compte vous permettra de suivre l'évolution de vos compilations dans le cloud, de récupérer vos exécutables et d'accéder à d'autres fonctionnalités utiles.

Pour la création d'un compte, cliquez sur le bouton SIGNUP (qu'on peut retrouver sur n'importe quel page du site en haut à droite) pour accéder au formulaire d'inscription. Remplissez ce formulaire. Une fois l'inscription effectuée, identifiez-vous (voir la Figure 1.8) pour accéder à votre espace membre personnel qui doit ressembler à celui de la Figure 1.9.

Figure 1.8 : *Formulaire d'identification*

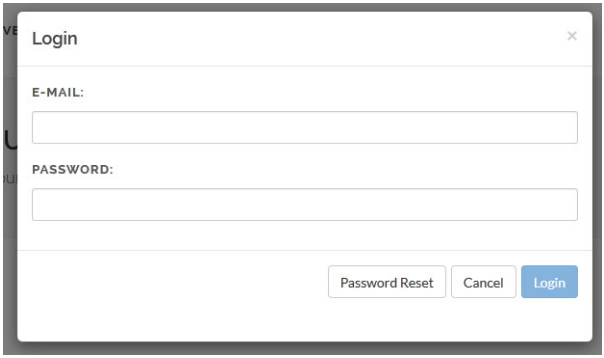
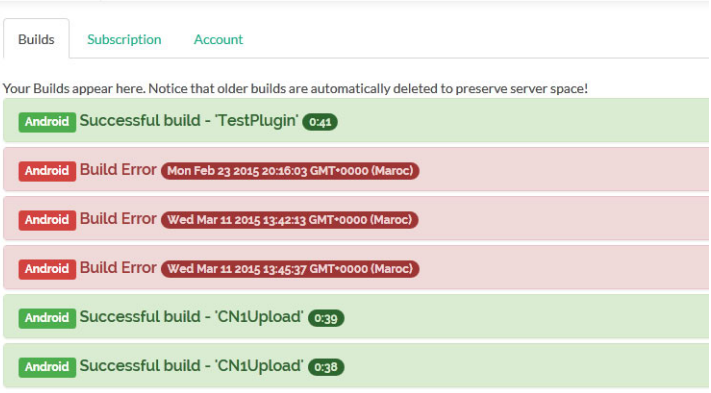


Figure 1.9 : *Espace membre*

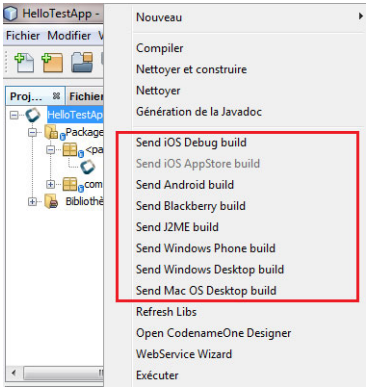


En fonction de votre type d'abonnement (gratuit ou payant), vous pouvez avoir plus d'onglets (donc plus de fonctionnalités) dans votre espace privé. Pour notre cas d'utilisateur gratuit, nous avons accès aux onglets suivants :

- BUILD : Sous cet onglet, vous verrez l'état en cours de vos compilations en ligne. Quand la compilation est en cours, il sera marqué *Building* [Compilation en cours]. Si elle réussit, *Successfully build* [Compilation réussie]. Si elle échoue, *Build Error* [Erreur de compilation] et un fichier journal (ou fichier *log*) au format texte contenant l'erreur qui s'est produite vous sera proposé en téléchargement.
- SUBSCRIPTION : Cet onglet permet de choisir un type d'abonnement payant selon vos besoins. Il est possible de souscrire à un essai d'un mois à un compte PRO pour tester les fonctionnalités payantes du cloud (comme l'utilisation des notifications push) avant de se décider par la suite. Un abonnement payant vous donnera aussi un accès gratuit à une formation vidéo complète. Cette page vous indique également le nombre de compilations que vous pouvez encore faire dans le mois courant (ceci concerne seulement les utilisateurs du service gratuit qui sont limités à 100 compilations par mois).
- ACCOUNT : Ici, vous pouvez modifier vos informations personnelles à savoir le nom, l'e-mail, le mot de passe et autres.

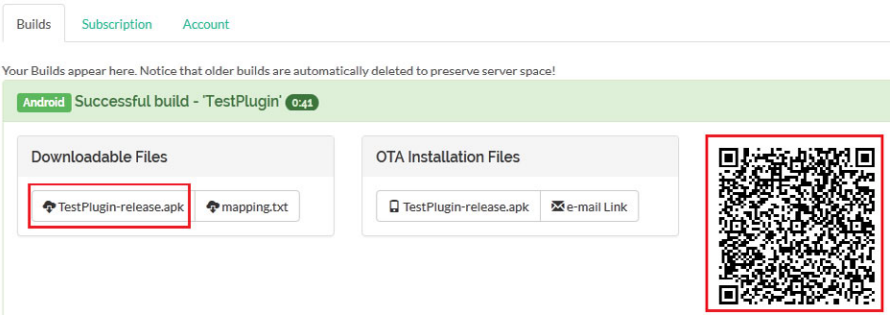
Voyons maintenant comment lancer une compilation en ligne. La méthode suivante concerne les utilisateurs NetBeans. Les utilisateurs d'Eclipse et de IDEA s'y retrouveront facilement. Dans NetBeans, faites un clic droit sur votre projet pour avoir accès au menu contextuel et sélectionnez l'une des entrées encadrées sur la Figure 1.10.

Figure 1.10 : *Lancer une compilation en ligne*



Le menu comporte des entrées de type SEND XXX BUILD, où XXX indique le nom de la plateforme que vous voulez cibler. Ainsi, un clic sur SEND ANDROID BUILD par exemple permettra de lancer la compilation de votre application pour la plateforme Android, ce qui va créer un exécutable d'extension .APK. S'il s'agit de votre première compilation alors une fenêtre s'affichera pour vous demander votre identifiant (e-mail) et votre mot de passe. Entrez-les et validez. Une fois la compilation terminée, connectez-vous à votre espace membre en ligne et téléchargez l'exécutable de votre application en cliquant sur le nom de l'exécutable du fichier ou en scannant le QRCode de l'exécutable avec votre téléphone ou tablette (voir la Figure 1.11).

Figure 1.11 : *Accès à l'exécutable des applications compilées dans le cloud*

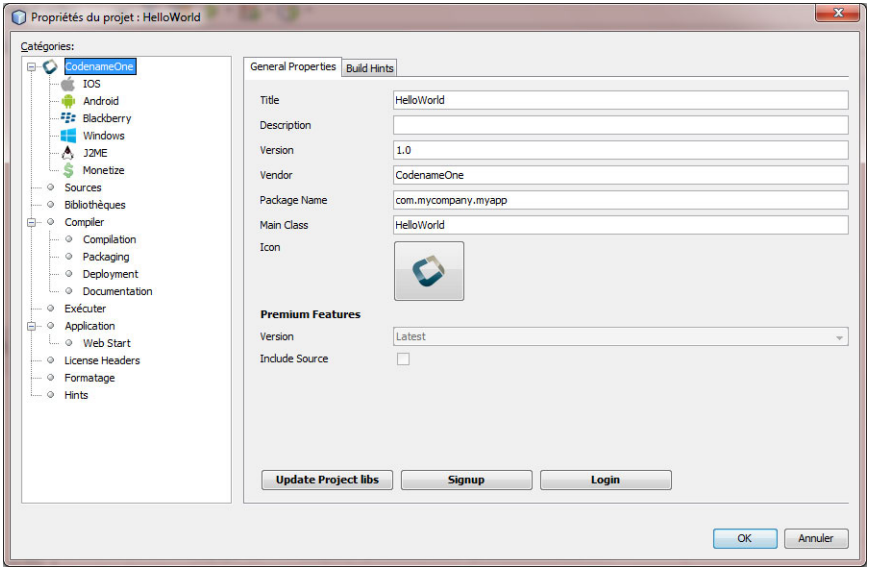


Avant de lancer une compilation de votre application dans le cloud, prenez soin de bien générer et de configurer dans les paramètres de votre projet (voir Figure 1.12) les fichiers de signature pour la ou les plateformes que vous voulez cibler. L'accès à ces paramètres se fait en cliquant droit sur le nom de votre projet et en sélectionnant l'entrée PROPRIÉTÉS.

Note *› Pour plus de détails sur la signature d'une application avec Codename One avant son déploiement, voir l'Annexe 3 : Signature d'une application.*

Note *› Pour plus de détails sur les arguments de compilation, voir l'Annexe 2 : Les arguments de compilation.*

Figure 1.12 : Interface des propriétés d'une application Codename One



1.5. Présentation et fonctionnement du simulateur

Le fait que Codename One soit fourni avec un simulateur est un grand avantage. Cela permet d'effectuer tous les tests possibles sur l'ordinateur sans être obligé d'installer et de supporter la lenteur des émulateurs de certaines plateformes (émulateurs Android et BlackBerry OS). Étant un simulateur (et non un émulateur), les comportements sur les plateformes ne sont pas forcément reproduits fidèlement, mais c'est le prix à payer pour avoir un outil flexible et s'adaptant au mieux à diverses plateformes.

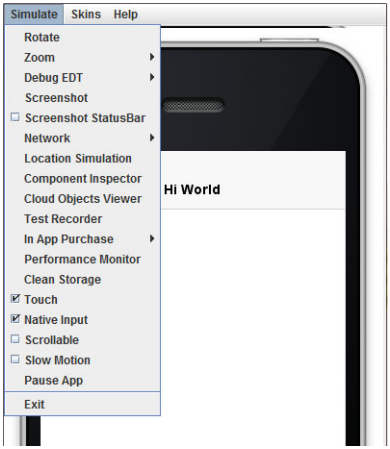
Le simulateur est composé de trois menus : SIMULATE, SKINS et HELP.

Le menu SIMULATE contient tout ce qu'il faut pour effectuer des simulations et interagir aussi avec vos applications.

Le menu SKINS fournit des thèmes pour chaque plateforme supportée par Codename One. Cela permet d'avoir une idée du rendu des applications en fonction de chaque plateforme.

Le menu HELP propose des raccourcis vers les pages de la documentation, du forum et vers le serveur de compilation.

Le menu Simulate



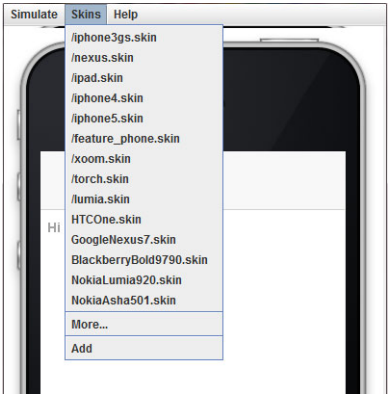
Rôle des fonctionnalités et des outils du menu SIMULATE :

- ROTATE : Permet de changer l'orientation du simulateur et de basculer du mode portrait (vertical) au mode paysage (horizontal) et vice versa.
- ZOOM : Permet de zoomer l'affichage.
- DEBUG EDT : Permet de traquer et de savoir s'il y a une violation de l'EDT (Event Dispatch Thread). L'EDT est le thread (processus léger) principal qu'utilise Codename One pour effectuer la majorité de ses tâches (dessins de l'interface, gestion des événements, etc.). Voir l'annexe 1 pour plus de détails.
- SCREENSHOT : Permet d'effectuer une capture d'écran de l'application qui est en cours d'exécution.
- SCREENSHOT STATUSBAR : Quand cette commande est activée (cochée), la capture d'écran de l'application contiendra la barre de statut. Si elle est décochée, alors cette barre sera juste absente de l'image capturée.
- NETWORK : Cette commande permet de choisir le niveau de la vitesse de la connexion internet à utiliser pour vos tests. Elle fournit aussi un outil nommé Network monitor pour traquer les requêtes d'envoi et de réception des données à travers le réseau. Cela aide à trouver facilement les problèmes qui pourraient se produire pendant la communication de vos applications avec un serveur distant.
- LOCATION SIMULATION : Fournit un outil pour simuler la fonctionnalité de géolocalisation.

- COMPONENT INSPECTOR : Affiche la hiérarchie des composants graphiques utilisés dans les applications. Vous obtiendrez aussi quelques informations utiles sur ces composants.
- CLOUD OBJECTS VIEWER : Fournit tout ce qu'il faut pour gérer les enregistrements de données que vous ferez dans le cloud de Codename One. Vous pouvez y charger vos données déposées dans le cloud, effectuer des requêtes, etc.
- TEST RECORDER : Permet d'effectuer des tests unitaires sans écrire du code.
- PURCHASE : Permet de simuler la fonctionnalité de paiement à l'intérieur d'une application (In-App purchase). Simule les achats, les abonnements, les remboursements.
- PERFORMANCE MONITOR : Comme son nom l'indique, cet outil sert à traquer la performance. Il permet de voir les informations sur la vitesse de dessin des composants graphiques qui composent les applications. Vous verrez aussi le nombre de fois où chacun de ces composants a été invoqué dans l'application, etc.
- CLEAN STORAGE : Permet de supprimer toutes les données stockées dans les *Storage* (l'une des méthodes de stockage disponible en Codename One).
- TOUCH : Quand elle est cochée (état par défaut) alors l'écran du simulateur devient tactile et on peut cliquer dessus. Quand elle est décochée, les clics sur l'écran du simulateur n'ont plus d'effets (utile pour les écrans non tactiles comme ceux des plateformes J2ME et de certains BlackBerry).
- NATIVE INPUT : Cochée (état par défaut), elle permet au simulateur d'utiliser le système d'entrée de données natif de la plateforme (clavier pour le cas d'un test sur ordinateur). Si elle est décochée alors le simulateur utilisera le clavier virtuel intégré à Codename One pour gérer l'entrée des données.
- SCROLLABLE : Quand ce menu est coché (état par défaut), il n'est pas possible d'agrandir à volonté la taille de la fenêtre du simulateur sans tronquer l'affichage. Par contre, on aura accès à une barre de défilement. En mode décochée, cela est possible mais la barre de défilement sera absente.
- SLOW MOTION : Permet d'afficher au ralenti les transitions d'une interface à une autre.
- PAUSE APP : Permet de simuler la mise en pause de l'application. Cet état de pause est activé lorsqu'un événement extérieur (un appel, l'arrivée d'un SMS, etc.) intervient pendant l'exécution de l'application.
- EXIT : Permet de fermer le simulateur.

Le contenu du menu SIMULATE change de temps en temps donc il se peut que d'autres fonctionnalités s'y ajoutent après la publication de ce livre.

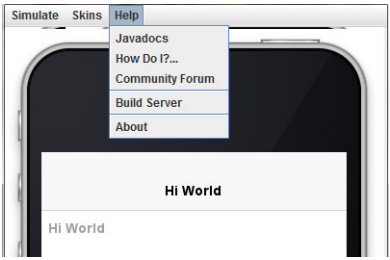
Le menu Skins



Le menu SKINS fournit divers habillages et styles visuels, constitués de thèmes, des images des plateformes, des polices de caractères et des propriétés (type de clavier, taille de police par défaut, police de caractères par défaut, etc.) des plateformes supportées. Chaque plateforme a son style d'affichage. Vous pouvez ainsi avoir un visuel fidèle de vos applications sur chacune de ces plateformes.

- MORE... : Ce menu ouvre une fenêtre permettant de télécharger et d'installer automatiquement d'autres skins de plateformes ou de certaines marques de smartphones.
- ADD : Permet de choisir et de charger manuellement un skin téléchargé sur l'ordinateur.

Le menu Help



- JAVADOCS : Permet d'ouvrir la documentation technique (la Javadoc) de toutes les classes de l'API.

- HOW DO I?... : Donne accès à une page contenant des tutoriels vidéo.
- COMMUNITY FORUM : Donne accès au forum du site. Il s'agit d'un forum vraiment actif.
- BUILD SERVER : Donne accès à la page permettant de s'identifier à l'espace membre du développeur dans le but de récupérer les exécutables des applications compilées.