

# Formation Java Avancé

## Informations

Durée : 3 jours (21h.)

Tarif\* : 1790 € HT

Réf : JAV2

Niveau : Moyen

inter intra à distance

\*tarif valable jusqu'au 18/06/2024

## Prochaines sessions

21 mai - 23 mai

()

22 juillet - 24 juillet

()

18 novembre - 20 novembre

()

## Pré-requis

- Posséder des connaissances en Java ou idéalement avoir suivi la formation Java (JAV1)

## Objectifs

- Consolider les connaissances acquises lors du cours Java Base
- Passer en revue les packages essentiels du développeur Java
- La programmation concurrente
- La communication par socket

## Programme

### La syntaxe avancée de Java

Les classes internes et anonymes

Les constructeurs

Les blocs d'initialisation

Les types génériques

La syntaxe des lambdas (Java 8)

La syntaxe des références de méthode (Java 8)

#### [Le package java.lang](#)

La classe Object

L'interface Comparable : égalité et comparaison

L'interface Clonable : copie d'objet

Les wrappers et l'autoboxing : conversions de type

Les chaînes : String, CharSequence, Appendable, StringBuffer, StringBuilder...

Les extensions syntaxiques : Iterable, AutoCloseable

Les énumérations : classe Enum

Les opérations mathématiques : Math et StrictMath

Les annotations standards (@Deprecated, @Override...)

Les exceptions : Throwable, Exception, Error, RuntimeException...

Les classes utilitaires : System, Runtime, Process et ProcessBuilder

Les bases du parallélisme : Runnable, Thread...

L'introspection : Class, Package, ClassLoader...

Les autres éléments du package java.lang

#### [Le package java.math](#)

Les nombres réels et les erreurs d'arrondis

Les nombres étendus : BigInteger, BigDecimal

La gestion des arrondis : MathContext et RoundingMode

#### [Le package java.util](#)

Les collections : Collection, List, Queue, Set, Map...

Itérer sur les collections : Enumeration et Iteration

Les classes d'implémentations de collections

Les classes utilitaires : Collections et Arrays

La gestion du temps : Date, Calendar...

La représentation de la monnaie : Currency

Le paramétrage : Properties

L'internationalisation : Locale, ResourceBundle, Formater...

Les classes utilitaires : Scanner, StringTokenizer, Random...

Les autres éléments du package java.util : Observer, Observable,

ServiceLoader...

#### [Le package java.text](#)

La comparaison des chaînes de caractères : Collator, RuleBasedCollator

Le formatage textuel : Format, MessageFormat, NumberFormat, DateFormat...

#### [Le package java.io](#)

La gestion de fichiers : File, FileFilter, FilenameFilter...

# Formation Java Avancé

La gestion des flux binaires : InputStream, OutputStream...

La gestion des flux textes : Reader, Writer...

La gestion des flux d'objets : Serializable, Externalizable...

## **Les classes utilitaires Java : Console, StreamTokenizer, RandomAccessFile**

---

Le package java.nio

La « nouvelle » gestion de fichiers : FileStore, FileSystem, Path, FileSystems, Files, Paths, PathMatcher, WatchService...

Les transferts de données : Buffer, Channel, Channels...

## **La réflexion en Java**

---

Principe de la réflexion

Le chargement de classes. L'objet Class

Découverte dynamique des informations relatives à une classe ou à un objet

Instanciation dynamique

Invoquer une méthode

La réflexivité des annotations

## **La communication par socket en Java**

---

La programmation en mode non connecté (par datagram). Le modèle Peer to Peer

Les protocoles TCP et UDP : InetAddress, NetworkInterface, Socket, ServerSocket

La communication en mode connecté (par stream)

Le modèle client/serveur. Serveur séquentiel VS serveur concurrent. Utilisation de la sérialisation

La librairie nio. Les buffers, channels. Les sélecteurs et leur utilisation

Le package java.net

Les accès réseau : URL, URLConnection, URLEncoder, URLDecoder

## **La programmation multi-threads en Java**

---

La création/destruction des Threads. Ordonnancement des Threads

La synchronisation des Threads

Le verrouillage des méthodes et des instructions (synchronized). Les moniteurs

Le problème de l'interblocage (caractérisation, évitement, prévention, détection). Le problème de la famine

Les nouveaux outils de synchronisation : les verrous partagés/exclusifs, les sémaphores, les barrières cycliques

Le package java.util.concurrent

Le parallélisme avancé (Futur, Executor, ExecutorService, Executors...)

Les collections synchronisées : BlockingQueue, ConcurrentMap...

Le package java.util.concurrent.atomic : les conteneurs thread-safe

Le package java.util.concurrent.locks : la gestion explicite du lock

## **Le package java.time (Java 8)**

---

Les nouvelles classes temporelles : Instant, Duration, LocalDate, LocalTime, Period, YearMonth, Temporal...

Les packages java.util.function et java.util.stream (Java 8)

Les FonctionalInterface : Consumer, Predicate, Fonction, Supplier...

Les interfaces de streams : BaseStream, Stream, Collector...

La construction des streams : stream(), paralleleStream(), iterate(), generate()

...

Les fonctions d'agregats : forEach, filter, sorted, map, collect...

## **Aperçu de quelques autres packages**

---

Le package java.util.logging : les traces

Le package java.util.prefs : la gestion des préférences utilisateurs

# Formation Java Avancé

Le package java.util.jar : la gestion des jar

Le package java.util.zip : la gestion des zip

Le package java.util.regex : les expressions régulières

Le package java.awt : les interfaces graphiques natives

Le package javax.swing : les interfaces graphiques riches