

Java™ magazine

By and for the Java community 

//MAY/JUNE 2014/

イノベーションに 役立つツール

IDE、デプロイメント・ツール、
モバイル、そしてクラウドを考察する

ORACLE.COM/JAVAMAGAZINE

ORACLE®

03

JAVAデプロイに役立つ オープンソース・ツール 7選

プロジェクトの規模を問わないプロセスの向上



新しいテーマ・アイコンです。[機能についてはこちら。](#)



23

開発者用ツールと そのトレンド

オラクルの
Chris Tona、
NetBeans IDEなどの
未来について語る

47

Enterprise NETBEANS IDE でのビルドと ORACLE JAVA CLOUD SERVICE へのデプロイ

アプリケーションの
デプロイにかかる
時間と労力を削減する

コミュニティ

02 編集長より

12 Java Nation ニュース、人々、 書籍、イベント

18 JCP Executiveシリーズ 肝心かなめの Javaツール Werner Keil氏、 Java Money and Currency APIについて語る

JAVAテクノロジー

27 Javaアーキテクト NetBeans IDE 8を 使用したJava SE 8への 迅速で簡単な変換 NetBeans 8の 新規ツールでJava SE 8の 関数型機能を利用する

31 Javaアーキテクト Java 8プロファイルを探る Compactプロファイルが アプリケーションに対して できること

34 Javaアーキテクト Java SE 8ストリームを 使用したデータ処理 Stream APIの高度な 操作を組み合わせ、 データ処理における 表現豊かな問合せを 記述する

39 Enterprise HTML5とJSF 2つのテクノロジーを 自在に組み合わせ、 洗練された アプリケーションを 作成する

53 リッチ・クライアント メリーさんの「ラムダ」 単純なゲームを通じて ラムダ式とStream APIに 親しむ

60 Fix This Java SE 8 Stream API コードの問題に挑戦しよう



私



Development Tools and Techniques

編集長 Caroline Kvitka

**For more information
or to place your
recruitment ad or
listing contact:
tom.cometa@oracle.com**

JAVAデプロイに役立つ オープンソース・ツール7選

プロジェクトの規模を問わないプロセスの向上 **BRUNO SOUZA, EDSON YANAGA**

継 続的デプロイとは、リード・タイムを短縮し、ソフトウェア・リリースの信頼性と品質を向上し、オーバーヘッドを削減するための一連の自動化プラクティスです。継続的デプロイの導入には多少の作業が必要となりますが、

プロジェクトに非常に良い効果があります。安定したデプロイ・パイプラインを確立し、開発、テスト、本番というすべての環境をできる限り同じ状態に保つことで、開発プロセスのリスクが大幅に減少し、イノベーションや研究が容易になり、

生産性を維持しやすくなります。

開発者は一般的に、デプロイの基本的な構成要素であるコード・リポジトリ (Git、Subversion など) やビルド・ツール (Ant、Maven、Gradle など) については認識しています。しかし、そのほかにプロセス



の向上につながるツールはないでしょうか。

本記事では、プロジェクトの規模を問わず、今すぐ利用してデプロイ・プロセスを改善できる、7つのオープンソース・ツールを紹介します。いずれも当該分野でもっとも普及している優れたツールであり、魅了された開発者によって、幅広い状況で使用できて他のツールやプロセスとも統合できる多くの知見やプラグイン、コネクタが作成されています。

しかし、より重要なのは、これらのツールによってデプロイ・プロセスを大幅に改善できる点です。開発チームはこれらのツールを活用することで、ストレスの軽減された環境で、より高品質で革新的なソフトウェアを構築することが可能になります。



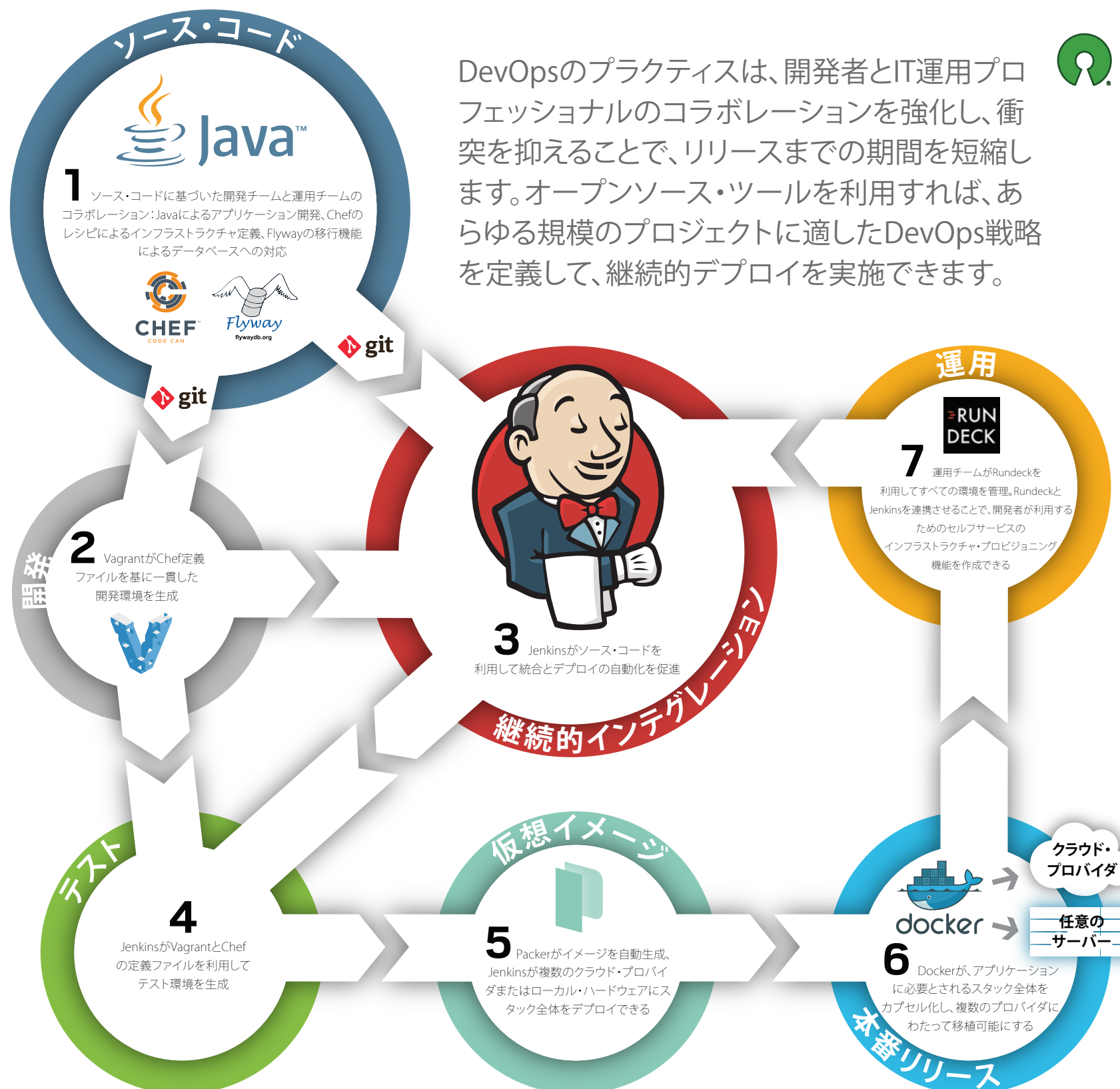
JENKINS

継続的インテグレーションから始める

Jenkins は、現在もっとも普及し積極的に開発されている継続的インテグレーション (CI) サーバーです。かつては Hudson という名の CI サーバーとしてリリースされていました (Hudson の詳細については、本号の「Mastering Binaries with Hudson, Maven, Git, Artifactory, and Bintray」(Michael Hüttermann) をご覧ください)。

Jenkins は、プロジェクトにおける自動化の土台として、使用されるプログラミング言語を問わず採用できます。しかし、もともとが Java コミュニティで開発され、また好んで利用されてきたという経緯より

継続的デプロイのためのオープンソース・ツール



画像: I-HUA CHEN

Jenkins のパワーと柔軟性：Build Pipeline Plugin は、Jenkins のジョブを連結してデプロイ・パイプラインを構成しやすくします。

特に Java プロジェクトとの親和性が高く、デスクトップ・インストーラから、クラウド内のアプリケーション・サーバーにリモートでデプロイされる Web アプリケーション・アーカイブ（WAR）ファイルまで広く対応できます。

Jenkins にはプラグインを生み出す大規模なコミュニティがあるため、多種多様なシステムと接続できます。また、柔軟で強力なジョブ定義メカニズムにより、任意のビルド・ツールを利用して開発プロセスのあらゆる要素を自動化できます。この機能により、複数のソースから情報を取得し、ビルド・ステップを実行して思いつくほぼ全種類のアプリケーションを作成し、他のターゲット・システムに接続して必要な任意のインフラストラクチャにアプリケーションをデプロイできます。

優れたデプロイ・プロセスを確立する

ためには、これらの操作用のプラグインを一部入手する必要があります。以下、あらゆるデプロイ環境に欠かすことのできないプラグインを紹介します。

- **Build Pipeline Plugin** : Jenkins のデリバリーに対するアプローチでは、ビルド・トリガーを使用して関連するジョブ同士をチェーン（連鎖）させます。この柔軟な機能によって洗練されたパイプラインを構築できますが、一方でパイプラインの「全体像」やジョブ間の関係、パイプラインにおける任意の時点でのビルド位置などを確認することは難しくなります。Build Pipeline Plugin は、わかりやすいビルド・パイプライン概要を示すことで、この問題を解決します。開発者はビルド中に状況を簡単に観察できるほか、ジョブを自動実行するか手動でトリガーを実行する必要がある

のかを設定することもできます。

- **Parameterized Trigger Plugin** : ビルド・パイプラインでは、あるジョブの出力として生成されるアーティファクトを次のジョブの入力として使用します。Parameterized Trigger Plugin は次のジョブに対して、パイプラインを進めるために使用する必要があるビルド結果を指定します。

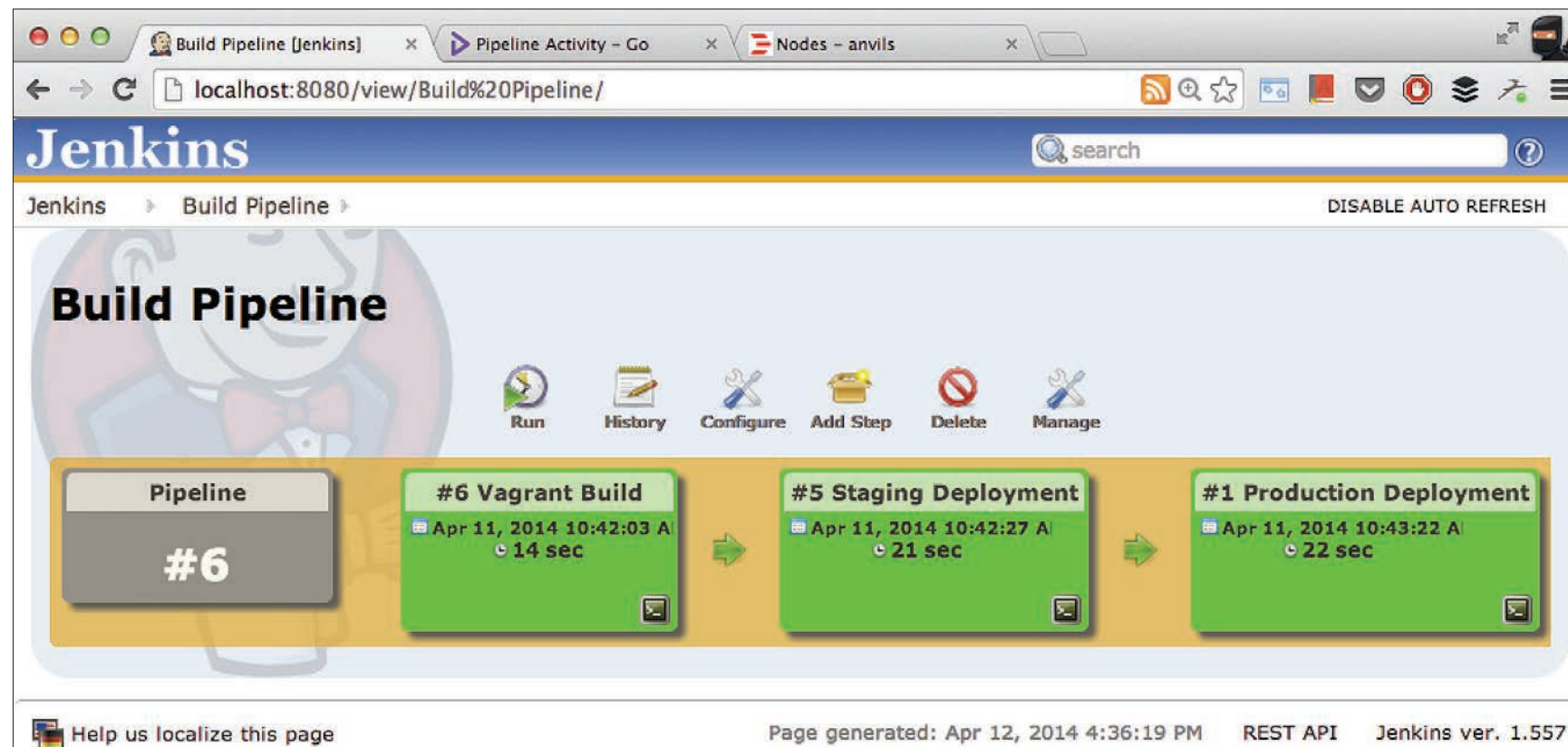
- **Copy Artifact Plugin** : Copy Artifact Plugin は Parameterized Trigger Plugin を補完するプラグインであり、前のジョブから受け取ったパラメータを使用してそのジョブで生成されたアーティファクトを取得し、次のジョブの入力に使用します。



インフラストラクチャを ソース・コードに転換する

Chef は、オンプレミスであるかクラウドベースであるかを問わず、開発環境または本番環境を容易に構成できるようにするプロビジョニング自動化フレームワークです。Ruby ベースのツールであり、Java アプリケーションに必要なインフラストラクチャをデプロイするための効果的な処理を行います。

Chefを利用すれば、プロジェクトのソース・コード・リポジトリ内でバージョンングされた Ruby スクリプトによって、プロジェクトのインフラストラクチャを定義できます。Chef (シェフ) のスクリプトは Recipe (レシピ) と呼ばれ、Cookbook (クックブック) にバンドルされます。スクリプト



ストレスの緩和

これらのツールによってデプロイ・プロセスを大幅に改善できます。開発チームはこれらのツールを活用することで、ストレスの軽減された環境で、より高品質で革新的なソフトウェアを構築することが可能になります。

トは Ruby で記述されるため、よく知られた汎用スクリプト言語を使用してインフラストラクチャのアクティビティを自動化できます。


インフラストラクチャを更新する必要があるときには、スクリプトを更新し、新しい定義で環境全体を再ビルドできます。これにより、ドキュメントを参照して行う（あるいはドキュメントもない）手動による大量のインストールと構成作業や、場当たり的な判断事項が不要となり、バージョンングされたクリーンな自動インフラストラクチャ構築ステップが実現します。この強力なコンセプトによって、インフラストラクチャの定義をアプリケーションとともに更新できるようになり、チームの意思疎通が活発になります。共通のツール、共通のリポジトリ、および明確なデプロイプロセスは、開発チームと運用チームの統合において大きな役割を果たすからです。

Chef は、オペレーティング・システムやインストールする必要のあるソフトウェアから、適用する必要のある設定情報（ユーザーや IP アドレスなど）まで、あらゆるインフラストラクチャ・コンポーネントを扱います。さらに、必要に応じて、ファイアウォール、ネットワーク・デバイス、複数のサーバー、クラウド環境などの、インフラストラクチャを構成する要素を扱うことも可能です。アプリケーション環境をビルドするだけでなく、複数のサーバーのパッチやソフトウェア・アップグレードを一元管理できるクライアント / サーバー・アーキテクチャも備えています。

Chef は、chef-solo クライアント・バージョンから使い始めるのがもっとも簡単です。このバージョンでは環境のプロビジョ

リスト1 リスト2 /リスト3

```
$ knife solo prepare host_username@host
```

 すべてのリストのテキストをダウンロード

ニングが単純化されているため、プロジェクトに必要なインフラストラクチャを単一のコマンドでビルドできます。クックブックを作成した後は、Chef のコマンドライン・インタフェース (CLI) である knife (ナイフ) を使い、chef-solo をサーバーにインストールして準備します (**リスト 1**)。その後は、クックブックを基にサーバーを自動的にプロビジョニングできるようになります (**リスト 2**)。



開発環境を複製する

本記事で紹介しているツールの中で、それ自体のデプロイとの関係性がもっとも薄いのはおそらく Vagrant です。Vagrant は開発者向けのツールですが、開発環境と本番環境の架け橋となり、デプロイ・プロセスにおける異なる環境間の差異を最小限に抑えることで、特有のリスクを軽減します。

開発中のアプリケーションを実行するためのローカル開発環境を構築することは、多くの開発者に共通する問題の 1 つです。ターゲット・システムはますます洗練されて複雑化しており、多数の Web サーバーやアプリケーション・サーバー、データベース、依存ライブラリ、キュー、サービス統

合フレームワーク、キャッシュ、ロードバランサなどの多様な要素で構成されます。このような状況では、開発環境とテスト環境を最終的な本番環境に近づけることは難しく、信頼性の高いビルド - テスト - デプロイ・パイプラインを作成しづらくなります。

Vagrant は Vagrantfile というテキスト形式の定義ファイルを基に仮想開発環境を生成します。この定義ファイルをプロジェクトのソース・コードとともにチェックインすれば、すべての開発者が Vagrant を使用し完全な機能を備えた同一の開発環境を単一のコマンドでプロビジョニングできるようになります。ローカルではこの環境が、オラクルのオープンソース仮想化製品である Oracle VM VirtualBox 内で稼働する仮想マシンとして作成されます。

Vagrantfile の設定を完了した後は、以下の単一コマンドを実行します。

■ \$ vagrant up

Vagrant とデプロイの関係は次のとおりです。CI サーバーで Vagrantfile を使ってテスト環境および品質保証（QA）環境向けの仮想マシンをビルドできます。この Vagrantfile は、後ほど（特に Vagrant を Chef や Packer、あるいは Docker と統合した場合に）最終的な本番環境をビルドするための継続的デプロイ・プロセスの基盤となります。

大きな影響
継続デプロイの導入は
多少の作業を伴います
が、プロジェクトに非常
に良い効果があります。

ています。そのようにして何度もデプロイする中で、ベース・イメージの安定性がある程度保たれます。このプロセスも悪くはないのですが、環境の更新が複雑になります。アプリケーションを、そのアプリケーションをサポートする最新のインフラに対して改善された内容をサポートしないイメージにデプロイした場合に、妙な不具合が発生することがあるのです。

この問題を解決するツールが Packer です。Packer は、さまざまなプロバイダ（Oracle VM VirtualBox、VMware、AWS、DigitalOcean、Docker など）向けのイメージを自動生成する強力なコマンドライン・ユーティリティです。CI プロセスの最後に実行でき、Chef プロビジョニングを再利用します。Packer を実行すると（プロビジョニング要件によっては時間がかかることがあります）、統合と構成が完了し、稼働中の完全な機能を備えた仮想マシン・イメージが生成されます。

Packerによりイメージが自動生成されるため、リリース・ビルドごとに Packer を実行し、構成済みのアプリケーション・サーバーにデプロイされている最新バージョンのアプリケーションを、必要なデータとともにイメージに含めることができます。開発者はインスタンスを起動するだけで、システムが数秒で稼働します。このシステムに対して、ロードバランサ内のプールを増やす、障害の発生したサーバーを置き換える、以前のバージョンを実行していたサーバーからジョブを引き継ぐなどの操作が可能です。

開発者の作業は、イメージを定義した Packer テンプレートを作成して、以下の

コマンドを実行することだけです。

```
$ packer build template.json
```



移植可能な自己完結型の 軽量コンテナを試す

すぐに起動できる仮想マシン・イメージを作成することは、クラウド・コンピューティングを利用してアプリケーションをスケールアップするための洗練された手法です。しかし、スタック全体から完全なオペレーティング・システム、アプリケーション・サーバー、その他の要素もすべて扱うことは、やり過ぎと言えるでしょう。多くの場合、必要となるのは、対象のスタックにアプリケーションを1つ追加することのみです。

Platform-as-a-Service (PaaS) ソリューションは開発者に歓迎されてきました。オーバーヘッドをほとんど心配せずに、事前定義されたスタックにアプリケーションを追加できるからです。単にコードをプッシュすれば実行されます。しかし、このアプローチでは単純すぎる場合もあります。クラウド・プロバイダ内部の共有環境がどのようなスタックを提供していても、またときには開発者の求めるほどの柔軟性がなくても、その環境でアプリケーションを実行しなければなりません。

この問題を解決するのが Docker です。
Docker は Linux Containers (LXC) の上
層に位置付けられ、アプリケーションを含

む移植可能な自己完結型のコンテナを作成します。Docker はお使いのコンピュータでローカルに実行することも、クラウド環境へとスケーリングすることもできます。アプリケーションに必要なすべての要素がコンテナ抽象概念にカプセル化されるため、デプロイが容易になります。また、Docker がコンテナを管理するため、オペレーティング・システムの完全インストールの負担をアプリケーション側ですべて担う必要はありません。Docker コンテナはきわめて軽量にできるため、小規模システムでも多数のコンテナを稼働させることができます。

まずは、まるで PaaS プロバイダのように、必要なすべての要素を含む 1 つの Docker イメージを構成できます。そうすれば、後は各コンテナでアプリケーションをホストして、アプリケーションごとに必要な変更を適用するのみです。コンテナは簡単に起動、停止、管理、更新できます。また、コンテナはそれぞれ独立しているため、アプリケーションごとに固有のバージョンのライブラリを使用できます。

Docker を利用すると、PaaS に似た環境を構築できます。また、テスト環境や QA 環境をコンテナによって管理し、これらのコンテナにも後で本番環境に配置するコンテナ定義を配置することで、CI プロセスを容易にすることもできます。Docker は Chef や Packer とともに統合されます。これらのツールを使って、アプリケーションからコンテナを自動生成できます。このコンテナをビルドやテストのためにローカルで利用し、統合とデプロイのために CI サーバーで実行できます。さ

らに本番運用向けの仮想マシン、ローカル・サーバー、プライベート・クラウド、あるいはパブリック・クラウドにまでスケールリングできます。

Dockerfile の設定が完了したら、以下のコマンドを実行して簡単に新しいコンテナを起動できます。

```
$ docker run image/name command
```

たとえば、[Quinten Krijger](#) 氏が提供している Apache Tomcat イメージは、以下のコマンドを実行してダウンロードできます。

```
$ docker pull quintenk/tomcat7
```

次に、以下のコマンドによって、このイメージをデーモンとして実行できます。

```
$ docker run -d quintenk/tomcat7
```



アプリケーションの後のデータベース移行

データベースに触れずにデプロイについて語ることはできません。開発者は、アプリケーションをアップグレードする方法や、旧バージョンの API および廃止機能との互換性を維持する方法については理解しています。しかし、データベースについてはどうでしょうか。自動デプロイのもっとも難しいところは、本番運用「直前」に実行する必要があるのにドキュメントが不十分な SQL スクリプトがある、コードとスキーマに差異がある、進化やロールバックが必要であるといった状況が発生することです。

```

3 import com.googlecode.flyway.core.api.migration.jdbc.JdbcMigration;
4
5 import java.math.BigDecimal;
6 import java.sql.Connection;
7 import java.sql.ResultSet;
8 import java.sql.Statement;
9
10 public class V1_Complex_Migration implements JdbcMigration {
11
12     private static final String UPDATE_10_PERCENT =
13         "UPDATE CUSTOMER SET CREDIT_SCORE = CREDIT_SCORE * 1.1 WHERE ID = %d";
14
15     private static final String UPDATE_ANOTHER_1_PERCENT =
16         "UPDATE CUSTOMER SET CREDIT_SCORE = CREDIT_SCORE * 1.01 WHERE ID = %d";
17
18     @Override
19     public void migrate(Connection connection) throws Exception {
20         try (Statement statement = connection.createStatement()) {
21             try (ResultSet resultSet = statement.executeQuery("SELECT ID, CREDIT_SCORE FROM CUSTOMER")) {
22                 while (resultSet.next()) {
23                     long id = resultSet.getLong("ID");
24                     BigDecimal credit_score = resultSet.getBigDecimal("CREDIT_SCORE");
25                     if (credit_score.compareTo(new BigDecimal("1000.00")) > 0) {
26                         statement.executeUpdate(
27                             String.format(UPDATE_10_PERCENT, id));
28                     }
29                     else if (credit_score.compareTo(new BigDecimal("10000.00")) > 0) {
30                         statement.executeUpdate(
31                             String.format(UPDATE_ANOTHER_1_PERCENT, id));
32                     }
33                 }
34             }
35         }
36     }
37 }

```

このような状況では Flyway が優れています。Flyway は、あらゆるバージョンのデータベース・スキーマを最新バージョンに移行できるデータベース移行フレームワークであり、Java で開発され、特に Java 開発者のニーズに集中的に対応しています。バージョン管理システム内でアプリケーションのデータベース定義が安全に管理され、データベースがクリアで簡潔な、バージョンニングされた一連の命令へと転換されます。この一連の命令は、必要なときにいつでも再作成または移行できます。

Flyway には CLI ユーティリティが付属しています。また、Flyway は Maven や Ant などのビルド・ツールと統合されま

す。CI プロセス内で実行して、テスト実施前や QA に進む前にデータベースのアップグレードや作成を行うことができます。Flyway では SQL スクリプトを実行できます。また、高度なデータベース要件がある場合は、JDBC API 経由で移行を処理する特定の Java コードを実行することもできます。Flyway の API を使用して、アプリケーション内で移行を管理し、データベース管理機能を実装することもできます。アプリケーションを配布する場合には、初回実行時に Flyway でデータベースの作成や移行を処理することが可能です。

データベース・スキーマをリポジトリ内のコードに転換することで、データベースを把握しやすくなり、チーム全体がデータ

強力なデータベース移行:FlywayではJDBC APIを利用してスキーマ移行を処理できます。

Flyway の構成作業が完了すれば、以下の CLI コマンドによってデータベースの作成または移行を行うことができます。



DevOps とは、開発者と IT 運用プロフェッショナル間のコミュニケーション、コラボレーション、統合を表すことばです。開発者は、良いツールがいかにコラボレーションを促進するかを理解しています。開発者と IT 運用プロフェッショナルの双方にメリットのある形で、インフラストラクチャ定義をリポジトリに保存したり、インフラストラクチャ作成を自動化したりしているプロジェクトは、DevOps 戦略を推進していると言えます。本記事で紹介している 7 つのツールはそれぞれ、開発と運用のコラボレーションに役立ちます。

ドし、デブロイ用のアーティファクトを生成することです。一方、運用プロフェッショナルにとって自動化は、新しいノードの検出や複数のサーバーへのコマンド・ディスペッチを指すでしょう。このように同じサーバーや同じサービスに対して重視する観点は別々ですが、開発と運用の両者とも、実用的なソフトウェアをエンドユーザーに届けるという責任があります。

Java ベースの Web アプリケーションである Rundeck は、環境用の運用コンソールとして機能することで、この開発と運用

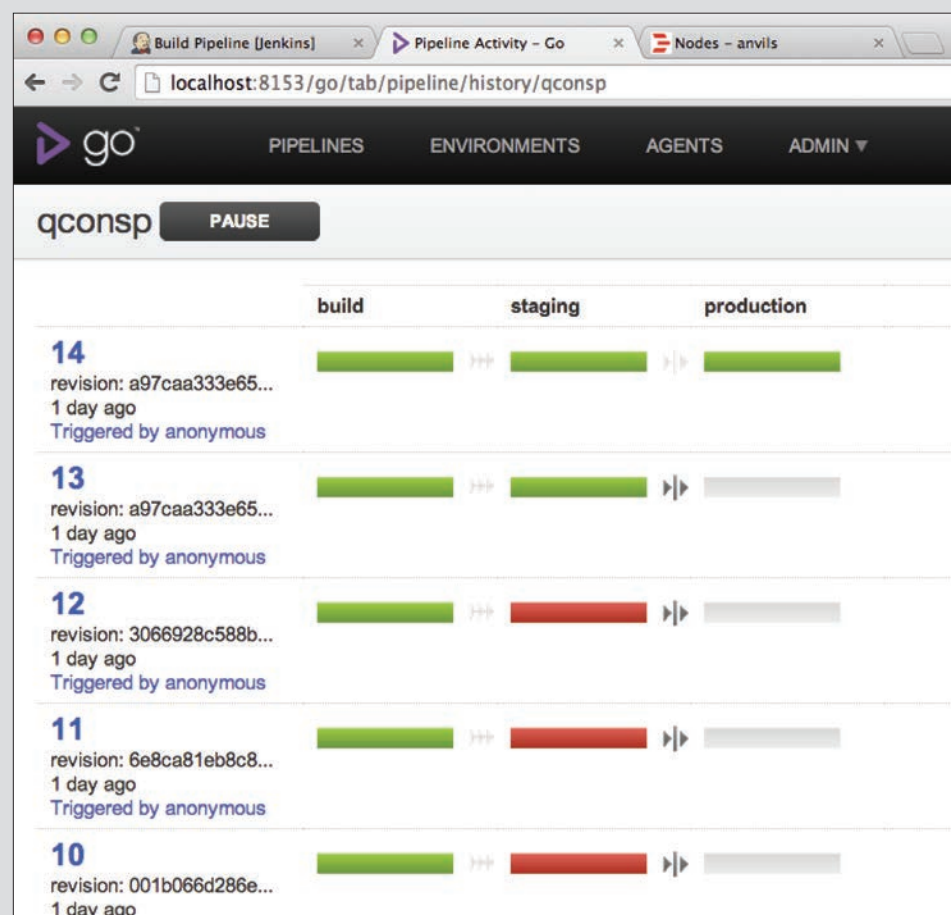
Rundeck は本記事で紹介した他のツールとスムーズに統合されます。Chef サーバーから詳細な環境情報を受信するか、クラウド・プロバイダから直接ノード情報

複数ノードの管理: Rundeckは複数のノード内でコマンドを実行し、結果を集約します。



Go : もう 1 つのツール

ThoughtWorks は先日、継続的インテグレーション / リリース管理サーバーの [Go](#) を、Apache ライセンスによるオープンソースの [プロジェクト](#) としてリリースしました。Go は Java ベースのサーバーであり、Jenkins に似た役割を担いますが、コンセプトが少し異なります。個別のジョブではなく完全なビルド・パイプラインを中心に構成され、コード - ビルド - テスト - デプロイ・プロセスに焦点を当てています。この構造ではプロジェクトにおいて継続的デプロイが特に重要となるため、チーム全体が最初から継続的デプロイを意識することができます。Go が開発者に受け入れられるのか、あるいはこの市場でどのように位置付けられるのかを語るのは時期尚早ですが、非常に優れたコンセプトであり、他の 7 つのツールと簡単に統合できる、完全な機能を備えたツールと言えます。



すぐに実行できるデプロイ:Go標準搭載のパイプラインには、成功したビルド、失敗したビルド、手動でトリガーしたビルドが明確に示されます。

を収集することで、動的なクラウド環境においても常に最新の情報を得られます。また、Rundeck は、1 つのノードでも数百のノードでもアクションを実行します。その際には、Secure Shell (SSH) 経由で直接行つか、Chef などのツールと統合して、何をどこで実行するかを指定したフィルタを定義します。その後、結果や出力を集約することで、現在の状況を的確に把握します。

Rundeck ジョブへのアクセスは、セキュアな Web ダッシュボードから制御できます。ジョブは、API 呼出しに変換し CLI 経由でスクリプトから呼び出せます。また、プラグイン経由で Jenkins ジョブから直接呼び出すこともでき、そうすれば統合がさらに容易になります。その結果、運用担当者は、開発者が実行可能なアクションを定義でき、セルフサービスのオンデマンド IT サービスの構築につながります。IT の手順が根本から自動化およびカプセル化され、ビルド・プロセスで簡単に呼び出せるようになります。運用担当者がプロセスに関与して、実行可能なアクションを定義するわけですが、この作業がボトルネックになることはありません。定義されるサービスは、開発者が必要に応じて実行できるものであるからです。

Rundeck CLI は強力な管理ツールです。**リスト 3** のコマンドを実行するだけで、スクリプトをすべての UNIX マシンにコピーして実行できます。

まとめ

本記事で紹介したツールはすべてオープンソースであり、すぐにでもプロジェクト

に導入できるものです。これらのツールは、あらゆる種類のアプリケーションのデプロイに利用できますが、特に Java アプリケーションや Java サービスのデプロイに適しています。

ぜひお試しになり、Java プロジェクトのデプロイ・パイプライン全体の構築にいかに関与するかを実感してください。 </article>

MORE ON TOPIC:



Bruno Souza: Java開発者。[Summa Technologies](#)ではオープンソース・エバンジェリストとして、[ToolsCloud](#)ではクラウドの専門家として従事。世界最大級のJavaユーザー・グループである[SouJava](#)の設立者/コーディネーター。Worldwide Java User Groups Communityの設立者。Open Source Initiative (OSI) のディレクター。

Edson Yanaga: Produtec Informáticaの技術リーダー、Ínsula Tecnologiaのプリンシパル・コンサルタント。オープンソースのユーザー、提唱者、開発者。Java、アプリケーション・ライフサイクル管理、クラウド・コンピューティング、DevOps、ソフトウェア職人気質などの専門知識を有する。国際的なカンファレンスで頻繁に登壇。

LEARN MORE

- ソース・コードのデモ
- Bruno Souza氏のブログ
- Edson Yanaga氏のブログ



上から時計回りに：将来についての考察を述べる Mike Milinkovich 氏、JavaFX 8 の最新情報を伝える Thomas Schindl 氏、Java 8 Day の幕を開ける Stuart Marks (左) と Georges Saab。

3月17日～20日にカリフォルニア州サンフランシスコで開催された **EclipseCon 2014** では、14のチュートリアルと120のセッション、4つのテーマ・デーが設けられ、2日目には **Java 8 Day** が開催されました。この日未明に Java 8 がリリースされたことで会場は熱気に包まれ、大量のダウンロードのせいでホテルの Wi-Fi が落ちるのではないかとこの日に Java 8 サポートを提供したという事実が良い巡り合わせを象徴していました。

カンファレンス2日目は、Eclipse Foundation のエグゼクティブ・ディレクターである **Mike Milinkovich** 氏の基調講演「Eclipse: The Next Ten Years」から始まりました。Eclipse の歴史と成功について語った後、Milinkovich 氏が注意事項として挙げたのは、どのようなプロジェクトが Eclipse コミュニティで開始されるかについて、同氏が指図する権限はないという事実でした。「私の肩書きは Eclipse 応援団長といったところです」、Milinkovich 氏は次のように言いました。「Eclipse のプロジェクトは非常にダーウィンのようなボトムアップ・プロセスであり、生き残るプロジェクトが有効なプロジェクトなのです」

Milinkovich 氏は、開発者と IDE に影響を与え、Java およびオープンソース・コミュニティにも当てはまる3つのトレンドについて話しました。

トレンド1：世界を飲み込むソフトウェア。 Marc Andreessen 氏に向かってうなずきながら、ソフトウェアはこれまでにないほど重要性を増しており、企業の評価方法を含むあらゆる要素に影響を与えている、と Milinkovich 氏は述べました。Milinkovich 氏は、Airbus の航空機を例に挙げ、このクラスの飛行機で使用されているソフトウェア・コードの量は3年間で4倍に増えていると紹介しました。Airbus は自社を航空機メーカーと見なすべきでしょうか。それともソフトウェア企業と見なすべきなのでしょうか。また、コードベースが非常に大きいだけでなく、アプリケーション



トレンド3：クラウド。 Evans Data の予測によれば、2019 年までに、65% の開発者がおもにクラウド向けの開発を行うようになる、と Milinkovich 氏は述べました。それでは、現在のデスクトップ IDE が持つ機能はすべてクラウドに移行すべきなのでしょうか。Milinkovich 氏は [Project Flux](#) のデモで、Eclipse プロジェクトをクラウドに接続する方法を紹介しました。EclipseCon では、クラウドでの開発に関する多数のセッションが開催されました。

Java 8 Day で開催されたその他のセッションには、**Thomas Schindl** 氏による、JavaFX 8 の新機能に関するものや、オラクルの **Hinkmond Wong** による、Java SE 8 の Compact プロファイルを使用した組込み開発に関するものなどがありました。Java 8 にとって最高の日でした。

VIRTUAL JUG (vJUG)



14



JAVA TECH

ABOUT US

0

f

ava
ne

log

15



Chin は旅の終わりにインドを訪れ、同国のバンガロールで開催された Great Indian Developer Summit に参加しました。



Sevarac 氏：12 歳のとき、TV 画面に色を付ける方法を見つけ出しました。さらに面白かったのは、ランダムに色

Sevarac 氏：ベオグラード大学でソフトウェア・エンジニアリング分野の助教授を務めています。Java を使用したソ

Sevarac 氏 : Java
Champion になったの
は最近ですから今のと
ころ大きな変化はあり
ませんが、これから

Sevarac 氏：現在取り組んでいるメインのプロジェクトは、Neuroph といって、Java オープンソースによる人気のニューラル・ネットワーク・フレームワークです。もっと強力なコミュニティを築き、関連のあるその他の Java AI プロジェクトと密接に協力して、典型的な問題やタスクに対応した共通 API の作成などを行いたいと考えています。理想的には、コン

Zoran Sevarac 氏は、[Good Old AI Research Network](#) にも参加しています。同氏のブログ [Extreme Mind Refactoring](#) や Twitter ([@neuroph](#)) もチェックしてください。

第 1 部と第 2 部の[アンケート](#) (PDF) ではおもに、はい/いいえ (または分からない) で質問に答える形で、幅広い潜在的な Java EE 8 の拡張機能に対して開発者から見た重要性が確認されました。第 3 部で取り上げた 13 種類の拡張機能は、第 2 部までのアンケートで 60% 以上の回答者が、重要であると回答した機能です。

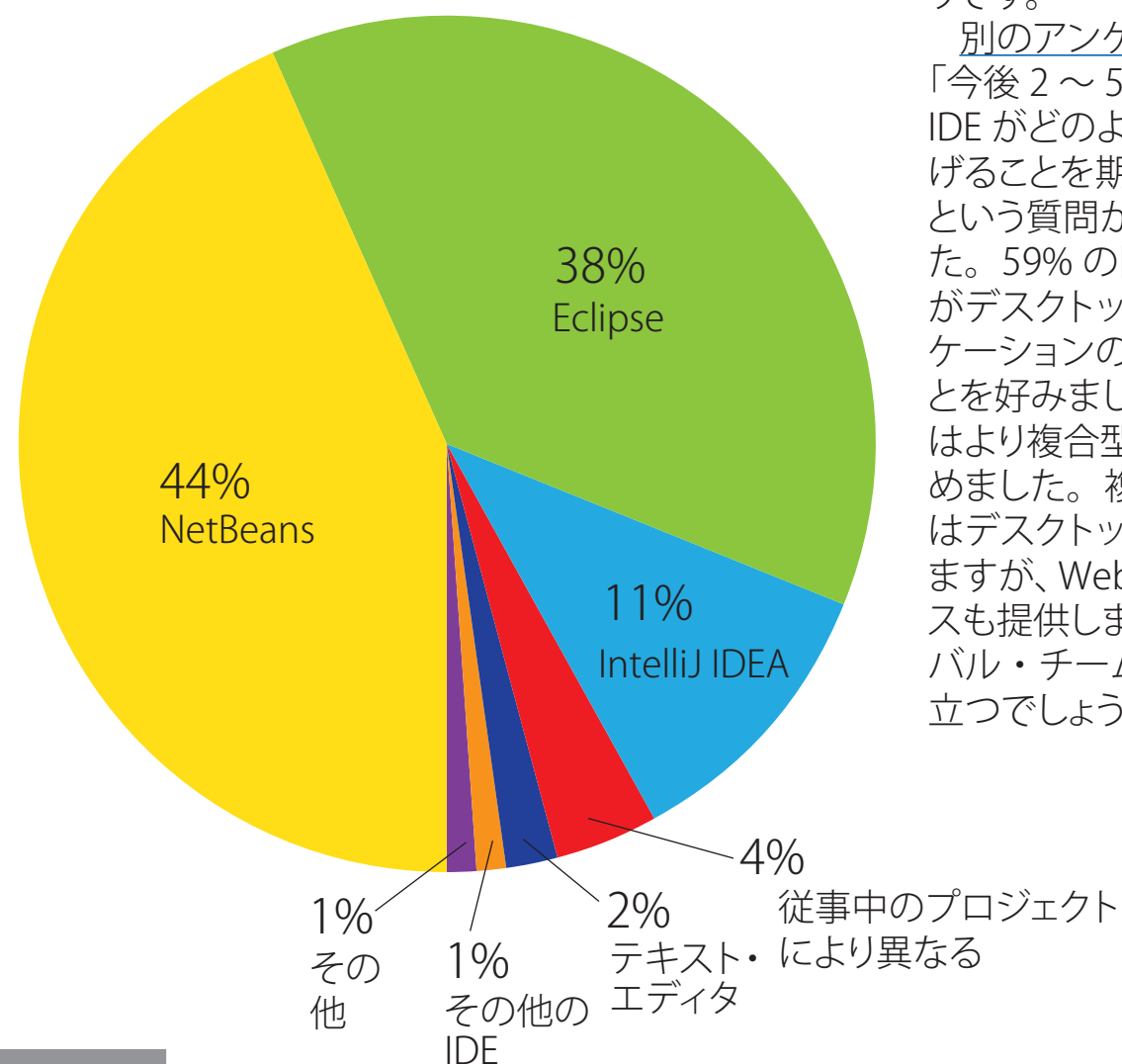
画像: I-HUA CHEN

開発者が好むのは NetBeans、Eclipse、IntelliJ IDEA

テキスト・エディタを使用して Java 言語や Java 仮想マシン (JVM) 言語をコーディングしていた日々が明らかに過ぎ去ったのには、もっともな理由が

ありました。925 名の開発者が次の文を完成させる形でアンケートに回答しました。「私は大半のコーディングに ... を使用しています」結果は以下のとおりです。

別のアンケートでは、「今後 2 ～ 5 年間で、IDE がどのような発展を遂げることを期待しますか」という質問がなされました。59% の開発者は IDE がデスクトップ・アプリケーションのままであることを好みましたが、30% はより複合型の IDE を求めました。複合型の IDE はデスクトップで利用できますが、Web インタフェースも提供します（グローバル・チームや出先で役立つでしょう）。





ブリュール(ドイツ)で
開催されたJavaLand
カンファレンス
で、NightHacking
ツアーについて
Stephen Chinと話す
Keil氏とJSR 354仕
様リードのAnatole
Tresch氏。

ミッタである Keil 氏は、Babel Language Champion であり、UOMo Project Lead でもあります。また、Java Community Process (JCP) の Executive Committee メンバーとして積極的に活動しています。さらに、多数の JSR やオープンソース・プロジェクトにも関与しており、クリエイティブな人材の仲介企業 (Creative Arts & Technologies) を自ら運営するかたわら、作詞や小説、脚本、技術的な書籍や記事の執筆もしています。

Java Magazine : JSR 354 (Java Money and Currency API) について簡単にご説

して、通貨データの解析と形式設定をサポートします。最後の拡張モジュールは、地域サポート、妥当性サービス、通貨マッピングなどの機能を提供します。

Money and Currency API は、銀行業務や金融、E コマース、モバイル支払いや組込み支払いといった、すべての一般的なアプリケーション分野を対象としており、地域アルゴリズムや対応する表示桁数を含む、複雑な計算ルールをサポートします。また、外国為替ロジック、複雑な使用シナリオ、金額のプレーン・テキスト表現、[String](#) への出力と解析も含まれています。

柔軟な API

私たちは、API にカスタム通貨を定義できる高度な柔軟性を持たせて、ゲーム環境で使用したり、仮想通貨の台頭に対応したいと考えました。

明いただけますか。

Keil 氏：JSR 354 は、通貨と金銭に関するデータの処理を目的としたインタフェースとクラスを提供するもので、4つの中心分野があります。コアには、通貨データと金額を表すクラスが定義されます。変換モジュールは、通貨の換算を行う API とサービス・プロバイダ・インタフェース (SPI) を提供し、通貨間の為替レートを処理します。形式モジュールは、一連の API と SPI を介

私たちは、API にカスタム通貨を定義できる高度な柔軟性を持たせて、ゲーム環境で使用したり、仮想通貨の台頭に対応したいと考えました。

また、サイズも重要な考慮事項でした。この API は、たとえば両替機で稼働する Java ME から、Java EE を実行する大規模サーバーまでのすべての分野に適用する必要があります。スケーラビリティが極めて重要ですが、私たちはこの課題を達成したと思っています。

Java Magazine：金銭処理でもっとも重要なのはセキュリティですが、その観点から少しお話しただけですか。

Keil 氏：もちろんセキュリティは重要です。しかし、Java Money and Currency API 自体はセキュリティ側面に対する責任を負っていません。信頼できる環境を金銭データに提供するには、Java プラットフォームの他の機能を利用する必要があります。Java SE のセキュリティ API は、暗号化やセキュアな通信、認証、アクセス制御、公開鍵インフラストラクチャに対応しています。また、ユーザーや管理者は一連のツールを使用することで、アプリケーションを安全に管理できます。私は JSR 321 (Trusted Computing API for Java) にも関わっていましたが、この JSR

現実世界のAPI

は Java アプリケーションからトラステッド・コンピューティング機能にアクセスできるようにするためのものです。Executive Committee のメンバーは、金銭的なランザクションが傍受されたり重複したりすることのないように、使用シナリオを把握しておく必要があります。これらの課題は、セキュリティ環境において非常に重要です。

Keil 氏：Money and Currency API と Java 8 の API に直接的な関係はありません。企業顧客の多くはアプリケーション・サーバーに含まれる JDK を使用しますが、JDK は当分 Java SE 7 のままであるため、この API に必要なのは下位互換性です。しかし、パターンやシグネチャ内で数多く使用している、Java 8 の関数型インタフェースは互換性を持つ予定です。ただし、API に組み込まれたラムダ式はおそらく、早くても Java SE 9 までは実行可能にならないでしょう。

Keil 氏：私はこの JSR の発端に大きな影響を与えたと思います。最初にアイデアが出たのは、ロンドンで 7 年前に開催された QCon のパネル・ディスカッションでした。金銭と通貨を対象にした JSR があれば非常に便利だろうということで、全体的に意見が一致したのです。そこで、この JSR が将来的な Java リリースとどのよ

5 年前、私はすでに Executive Committee のメンバーでしたが、プリンストンでミーティングを開き、JSR 275 を支援するための測定単位に関してプレゼンテーションを実施しました。この取組みは JCP フェーズで中止されましたが、一部がその後の JSR 354 を開始するきっかけとなりました。

わった Credit Suisse が通貨と金銭に関する JSR の価値を理解し、同社の Anatole Tresch 氏が仕様リードとして参加して、素晴らしい仕事をしてくれました。今年の第 2 四半期にはパブリック・レビュー 2 が行われる予定ですが、このレビューにパスして、JSR の機能が完成したと見なされれば、JavaOne の頃には最終バージョンの準備が整うかもしれません。

Java Magazine : Eclipse Foundation との関わりは JCP での活動にどのような影





ともに講演をした
JavaLandのセッ
ション間にひと休
みするTresch氏と
Keil氏。

響を与えていますか。

Keil 氏：JSR 275 は汎用の測定に関する取り組みでしたが、2010 年当時、Java プラットフォームにとっては小さすぎる特殊分野だと見なされていました。そのため、私は仕様リードとして、unitsofmeasurement.org と呼ばれるオープンソース・プロジェクトでこのコンセプトを推進しました。このプロジェクトでは、単位と数量を処理するための Java

JSR 354 スケジュール

今年の第 2 四半期にはパブリック・レビュー 2 が行われる予定ですが、このレビューにパスして、JSR 354 の機能が完成したと見なされれば、JavaOne の頃には最終バージョンの準備が整うかもしれません。

インタフェースを提供するライブラリを開発しています。このプロジェクトの設計には、JSR 275 に反対した人々を含む、Executive Committee メンバーのコメントを反映させました。そのため、この新しい活動には改良が施されており、古い JSR とは別のものになっています。復活したこのプロジェクトは Eclipse 傘下の [UOMo](#) というオープンソース・プロジェクトによって実装されており、私がリーダーを務めています。開発者はこの実装を利用して、各種の測定単位系間の換算や、単位の文字列表現の形式設定と解析を実行できます。

unitsofmeasurement.org と UOMo で実施した作業を、Java に付加価値を加える新しい JSR で使用することで、測定単位に関して C/C++ などの言語と同等の機能を提供することなどを目的として、数か月前、私は JSR の復活についての意見を述べました。この意見は JCP Program Management Office に提議され、現在、JSR 363 として提出されています。この JSR は特に、発展を続ける Internet of Things 分野に対応する組込み Java アプリケーションに役立つだろうと考えています。この分野で生成される途方もない量のデータには相互運用性が必要になります。

すから。

そのようなわけで、この一連の流れは、JCP の取組みと Eclipse などのオープンソースの取組みが相互に作用することで、関係者すべてに付加価値をもたらす良い例になりました。このような活気あるエコシステムがイノベーションをもたらすのです。

Java Magazine : JCP Executive Committee で独立系メンバーはあなただけです。関わり方に特有の点がありますか。また、どのようなきっかけから Executive Committee に加入されたのでしょうか。

Keil 氏：2008 年のことですが、BEA を辞めて数か月した頃、独立した請負業者として Executive Committee メンバーの候補者になりました。幅広い経験や JCP の活動に参加した経歴のおかげで、Java SE サイドの個人候補のうちで最多票を獲得することができました。いくつかの専門家グループのメンバーだったことがあり、仕様リードの 1 人として JSR 275 を手伝うように頼まれました。

独立系請負業者としての多様な経験から、Java EE や大規模ポータルから小型デバイス上の組み込み Java まで、さまざまな角度から独自の視点で Java 環境を

オラクルのSenior Technologist兼 Product ManagerであるTerrence Barrと話すKeil氏。



見ることができました。このような経験を Executive Committee への参加において活かしたいと考えています。

もともと JCP に参加したのは、Java プラットフォームをできる限り堅牢で競争力あるものにするために貢献したいと考えたからです。私たち誰もが、アイデアを実現するためにこれらのツールに依存しているのですから、最善のソリューションがテクノロジーに含まれるようにする必要

があります。

Java Magazine：JCP に対する独立系の参加を促進するにはどうすればいいでしょうか。

Keil 氏：オラクルの Patrick Curran が仕様リードを務める JSR 358 では、JCP に対して多数の重要な改定や調整が行われています。この JSR の一番の目的である Java Specification Participation Agreement (JSPA) の改定は、JCP の参加に前向きな影響を与えるでしょう。JSPA は知的財産や特許に関係があり、一部の企業にとって問題となる場合があるため、JCP への参加に消極的な雇用主もいます。

1 つの可能性は、Eclipse Foundation の例にならって、開発者が雇用主から離れて、加入者や提携者として指定された個人として参加できるようにすることです。その場合は、通常は開発者がコードを直接さわらなくてもアイデアを提供できるようにする、という合意が別途必要になります。このようなタイプのメンバーシップがあれば、法的な契約や役割が簡素化されて良い影響が出ると考えています。その目的は、JCP のために実施した作業によって特許や知的財産に関する争いが生じないようにすることです。

JCP.next は、JCP の改善専用 に設けられた JSR グループ (JSR

358 を含む) です。自営業者や Java ユーザー・グループのメンバー、大学職員の参加が増えることを期待しています。また、オープンソース開発プロセスや公開メーリング・リストの利用を通じて参加を改善することも計画しています。

Java Magazine：最後に一言お願いします。

Keil 氏：Executive Committee には常に、複数の独立系メンバーや個人、JUG が加わっていてほしいと思っています。経歴や経験が多様であればあるほど、イノベーションや問題解決が推進され、ひいては Java プラットフォーム全体の品質と実現性の向上につながります。 </article>

Steve Meloan：元 C/UNIX ソフトウェア開発者。Wired、Rolling Stone、Playboy、SF Weekly、San Francisco Examiner などの各誌で Web やインターネットに関する記事を執筆している。近著はサイエンス・アドベンチャー小説の『The Shroud』。また、The Huffington Post に定期的に寄稿している。

LEARN MORE

- [JSR 354: Money and Currency API](#)
- [JSR 354プロジェクトのページ \(Java.net\)](#)
- [NightHackingインタビュー](#)



開発者用ツールと そのトレンド

オラクルの**Chris Tonas**が、Javaのツールと開発フレームワークについて語ります。

TORI WIELDT

PHOTOGRAPHY BY BOB ADLER/GETTY IMAGES



DEに関する議論ほど白熱するものはありません。どのIDEを使用するのが一番良いのか、ひいてはIDE自体を使用すべきか否か（「Emacsがあれば十分」という声もあるでしょう）について、開発者たちは宗教的とも言えるくらいの熱意を持っています。あらゆる場合に使用できる万能なツールなどないことは明らかですが、Java開発者の多くにとって役立つツール



小型デバイスとタブレット向けの開発について話すChris Tonas (左) と、オラクルのテクニカル・スタッフの1人であるSundar Sarangan。

とはどのようなものでしょうか。JavaおよびNetBeans IDEの管理者として、またOracle JDeveloperの開発者として、オラクルは開発者用ツールに関してどのような役割を果たすべきでしょうか。ソフトウェアの開発とツールを方向付けている重要なトレンドには、どのようなものがあるのでしょうか。

Java Magazineは、オラクルのMobility and Application Development Tools担当バイス・プレジデントであるChris Tonasにインタビューし、Javaのツールと開発フレームワークの最新情報について聞きました。

Java Magazine: 開発ツールに関してオラクルでどのような役割を果たしているのか、少し話していただけませんか。

Tonas: ソフトウェア開発のツールとフレームワークを扱う組織のリーダーを務めています。具体的には、NetBeans、Eclipse、あるいはOracle JDeveloperを対象とした、Java開発者向け製品を構築するチームを率いています。私たちのチームは、オラクルのクラウド・プラットフォームやモバイル・プラットフォームで開発にあたっている開発者向けのツールやフレームワークも構築しています。

Java Magazine: 2014年3月にJDK 8とNetBeans IDE 8がリリースされましたが、このリリースについてのご意見を聞かせてください。

Tonas: JDK 8とNetBeans IDE 8のリリースは、OracleとJavaコミュニティの両方を前進させる大きな一歩となりました。このマイルストーンに達したのは、多くの勤勉な取り組みと協力のおかげです。この機会を借りて、今回のリリースに貢献したすべての人々に感謝したいと思います。

Java Magazine: 最新のNetBeans IDE 8.0がリリースされましたが、NetBeansのリリースを今後どのように進めていく予定ですか。

Tonas: 短期的なところでは、Java ME 8にあわせてNetBeans IDE 8の更新リリースの準備を進めています。その後、個別のバグを対象とした追加のNetBeans 8リリースの公開を予定しています。より長期的には、オラクルはJavaとNetBeansの双方が成功し続けられるように尽力しています。JDK 9に対する取組みも進行中であり、従来どおり、JDK9に併せてNetBeans 9もリリースする予定です。

Java Magazine:先ほど話されたとおり、
オラクルのサポートはNetBeans IDEにはと

どまりませんね。その背景にはどのような
考えがあるのでしょうか。

Tonas: オラクルは、開発者用ツールは万能型のツールではないと考えています。オラクルはEclipseプロジェクトに多大な貢献をしており、Eclipseベースのソリューションの機能も引き続き拡大させていく予定です。Oracle Fusion Middlewareスタックとの緊密な連携を求める開発者を対象に、Oracle JDeveloperを提供しています。また、多くのJavaScript開発者が軽量ツールを求めていることも認識しており、この要求にも対応していく予定です。

Java Magazine: 現在のソフトウェア開発分野で見られる重要なトレンドにはどのようなものがありますか。

Tonas:いくつかの重要なトレンドによって、ソフトウェアの開発とツールが方向付けられていることは明らかなです。オラクルはこれらの変化の最前線に立ち、ほとんどすべての側面でリーダーの役割を果たしています。現在、おもに次の3つの変化が起きています。

はじめに、今後もサーバー側開発の業界標準がJavaであることに変わりはありませんが、プレゼンテーション・レイヤーにおいては、JavaScriptとHTML5を組み合わせた開発をサポートするように求める声が高まっています。また、JavaScriptは一部のサーバー側ユースケースでも支持され始めています。

次に、クラウドベースのデプロイへの移行が主流となっています。クラウドに対応する開発は新たな課題をもたらし、斬新なアプローチが求められています。

3番目の変化はモバイルへの移行です。モバイル開発では、設計フェーズからライフサイクル全体におよぶ全社的な統合が



オラクルのクラウド・ ツール戦略について 話す、Tonasと製品管 理担当ディレクターの Brian Fry。

のOracle Developer Cloud Serviceになるでしょう。このサービスを利用すると、ALM（資産ライフサイクル管理）とチームのコラボレーション作業をクラウドで実施できるようになります（詳しくは[こちら](#)を参照してください）。

Java Magazine: 各企業はモバイル開発を重視していると言っていましたが、Javaはモバイル開発に適していますか。また、モバイル開発に対応するオラクルのソリューションについて聞かせてください。

Tonas: オラクルのモバイル開発では、オンデバイス・アプリケーションとモバイル・ミドルウェアの両方において、Javaが鍵となります。

す。Oracle Mobile Platformはモジュール方式で利用できるように設計されているため、個別のコンポーネントを利用することも、プラットフォーム全体を利用することもできます。このモバイル・プラットフォームは現在拡張中ですが、現時点でも十分に魅力ある製品になっています。

私たちが注力しているのは、Java開発者がOracle Mobile Application Frameworkを使用して順調にモバイル開発を進められるようにすることです。開発者はJavaでコードを書き、iOSまたはAndroidのタブレットや携帯電話で実行できます。また、JavaScript/HTML5で書いたコードや、JavaとJavaScript/HTML5を組み合わせたコードも同じ方法でデ

プロイできます。

Oracle Mobile Suiteには、オンプレミス・アプリケーションやクラウドにデプロイしたアプリケーションを企業がモバイル化するために必要なあらゆる要素が含まれており、すべてのコンポーネントはJavaをベースとしています。そうしたシステムへの統合はREST APIとして公開されるため、モバイル・アプリケーション・フレームワーク内で簡単に利用できます。

Oracle Mobile Security Suiteを利用すると、機密性の高いエンタープライズ・アプリケーションやデータに、ユーザー個人のモバイル機器から安全な方法でアクセスでき

るようになります。さらにこのソリューションは、企業内にすでに存在するIDおよびアクセス・ソリューションと統合できます。

Java Magazine:これらの新しいツールについての詳しい情報はどこで入手できますか。

Tonas: ソフトウェア開発の将来に関して
オラクルが抱く構想の全体像は、例年ど
おり、年内に開催されるJavaOneとOracle
OpenWorldで披露されます。私たちは、現
在取り組んでいる活動の内容を開発者コ
ミュニティと共有できる日を心待ちにしてい
ます。 </article>

追加情報



Java Developers for Oracle Technology NetworkのSenior Community Managerである**Tori Wieldt**は、オラクルの開発者用プログラムのあらゆるチャンネル ([OTN/Java](#)、[@Java](#)、[Java Magazine](#)、[blogs.oracle.com/java](#)、[YouTube/Java](#)) を通じて、Java開発者のエクスペリエンスを管理しています。Wieldtは世界中を旅して開発者にインタビューすることを好み、技術系ライター、システム管理者、Web管理者、マーケッターとしてのテクノロジー経験を有しています。

LEARN MORE

- [NetBeans IDE](#)
- [Oracle JDeveloper](#)
- [Oracle Enterprise Pack for Eclipse](#)

NetBeans 8の新規ツールでJava SE 8の関数型機能を利用する



Development
Tools and
Techniques

一見すると単純な変換のようですが、IDE によって静的分析が行われ、安全な変換が保証

图3

されています。たとえば、図5のコードをよく見てください。

ここで使用されている `doPrivileged` メソッドはオーバーロードされており、`PrivilegedAction` または `PrivilegedExceptionAction` のいずれかを引数に取ることができます。この状態でリファクタリングを実行した場合、正しい型へのキャストが自動

的に追加されます (図6)。

仮にこのキャストが追加されなければ、変換後のラムダ式は図7のようにあいまいになり、コンパイル・エラーが発生します。また、図8では、匿名内部クラス内の変数宣言が、その外側のスコープで宣言された変数を隠して (シャドウ化して) います。変数のシャドウ化は匿名内部クラ

スでは正当ですが、ラムダ式では許可されません。そのため、IDE では、変数のシャドウ化が検出された場合は、コンパイル・エラーを防ぐためにシャドウ化変数の名前が自動的に変更されます。たとえば図9のように、エラーを回避するために変数名に「1」が付加されます。

2 回目のリファクタリング機能 (図10) でも Java SE 8 の新機能を利用します。その新機能とは内部イテレータです。Java SE 8 では、ラムダ式を引数に取り、コレクションに

対して関数型操作を実行できる各種メソッドが Collections API に追加されました。関数型操作の例として少し挙げるだけでも、`MapReduce`、`filterForEach`、`anyMatch` などがあります。NetBeans IDE では、既存の外部イテレータ (拡張 `for` ループなど) を、新しい内部イテレータを使用するようにリファクタリングできます。

図11の例では、ルールのコレクションに含まれるエラーの総数を計算しています。それほど明白ではないものの、この例は `filter`

```

21 JButton testButton = new JButton("Test Button");
22 testButton.addActionListener((ActionEvent ae) -> {
23     System.out.println("Click Detected by Anon Class");
24 });

```

図4

```

27 try {
28     conv = AccessController.doPrivileged(
29         new PrivilegedExceptionAction<B2CConverter>() {
30             @Override
31             public B2CConverter run() throws Exception {
32                 return new B2CConverter(enc);
33             }
34         });
35 } catch (PrivilegedActionException ex) {
36     Exception e = ex.getException();
37     if (e instanceof IOException) {
38         throw (IOException) e;
39     }
40 }

```

図5

```

27 try {
28     conv = AccessController.doPrivileged(
29         (PrivilegedExceptionAction<B2CConverter>) () -> new B2C
30 } catch (PrivilegedActionException ex) {
31     Exception e = ex.getException();
32     if (e instanceof IOException) {
33         throw (IOException) e;
34     }
35 }

```

図6

```

27 try {
28     conv = AccessController.doPrivileged(
29         () -> new B2CConverter(enc));
30 } catch (PrivilegedActionException ex) {
31     Exception e = ex.getException();
32     if (e instanceof IOException) {
33         throw (IOException) e;
34     }
35 }

```

図7

```

19 private JSlider createZoomSlider() {
20     JSlider slider = new JSlider();
21     slider.setFocusable(true);
22     slider.setMinimum(1);
23     slider.setMaximum(800);
24     slider.setValue(100);
25     slider.addChangeListener(new ChangeListener() {
26         @Override
27         public void stateChanged(ChangeEvent event) {
28             JSlider slider = (JSlider)event.getSource();
29         }
30     });
31 }
32 return slider;
33 }

```

図8

NetBeans IDE では、ラムダ式を、
同じ操作を表すメソッド参照に変換

图 10

かるだけでなく、適切に行わなければ検出しづらいバグにもつながります。NetBeans IDE では、元のプログラムの振る舞いを維持しながら、これらのリファクタリングがすばやく実

图 11

图12

图13

图14

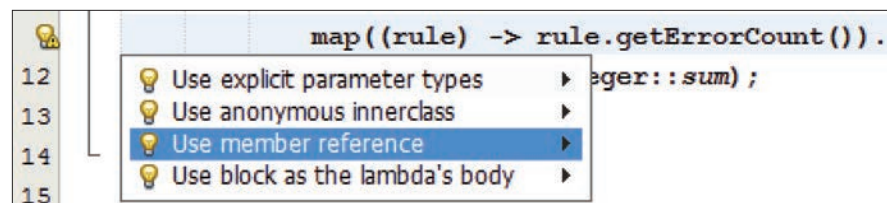


图15

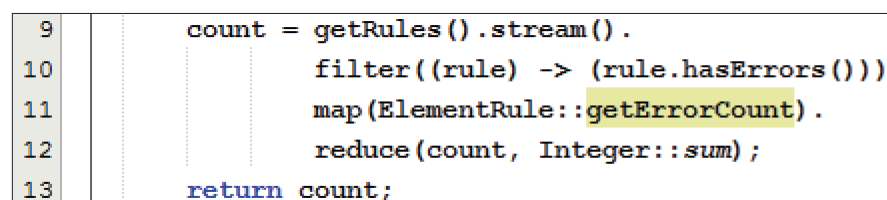


图16

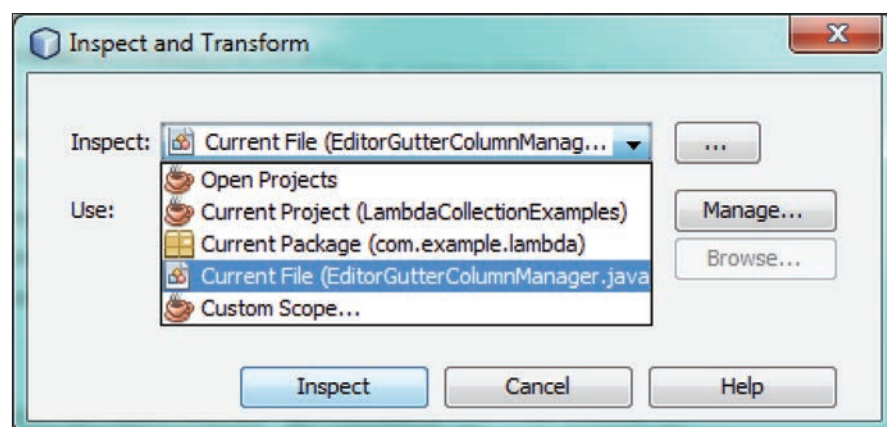


图17

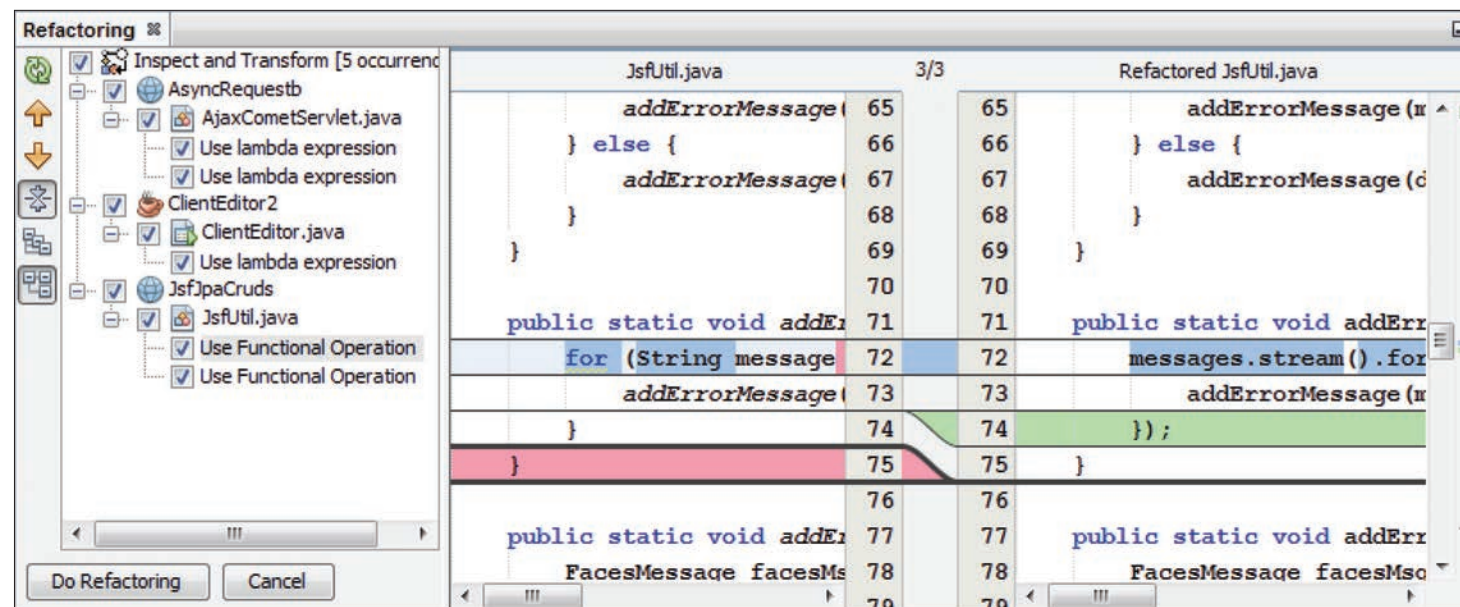


图18

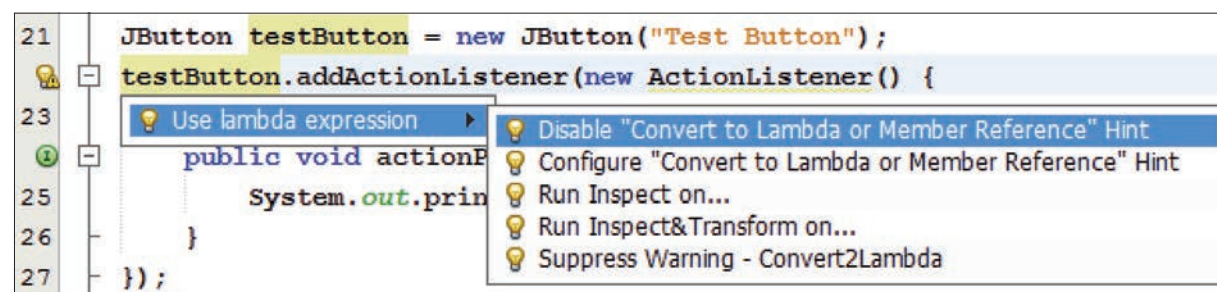


图19

行されます。

また、クイック・ヒントを表示する操作モードに加えて、リファクタリングを一括で実行するバッチ・モードも利用できます。**図 17**のように、1つのファイル、プロジェクト全体、または開いているすべてのプロジェクトを選択して、そのスコープに含まれるすべての変換候補をプレビュー表示できます。

また、バッチ・モードのリファクタリング・ツールを使用しない場合も、左サイドバーに Java ヒントが表示さ

れたときにはいつでも、任意のスコープでリファクタリングを実行するように指定できます (図 18)。

コードが数十万行に及ぶプロジェクトでも数秒でスキャンされます。レビューでは、実行する変換の内容を詳細に制御することも、変換可能な箇所をそのまますべてリファクタリングすることも可能です (図 19)。

まとめ

NetBeans IDE 8 では、既存の重要なコード構造を、Java SE 8 で導入され

た新しい関数型スタイルのコードへと正確かつスムーズにアップグレードするために必要なツールが提供されます。NetBeans IDE 8の自動リファクタリング機能により、関数型操作を使用するようにコードをリファクタリングできます。また、バッチ・モードでは、リファクタリングの範囲を指定できます。ぜひ今すぐにでもコードの変換を始めてみてください。

</article>

MORE ON TOPIC:



LEARN MORE

- NetBeans IDE 8のダウンロード



Compactプロフィールがアプリケーションに対してできること

(@kittylyst) :
jClarityの技術者
および創設者であ
り、London Java
Community
(LJC) 主催
者。Java SE/
EE Executive
Committeeのメン
バーでもある。

- Java仮想マシン (JVM) 起動時間の短縮
- リソース使用量の削減
- 結果的にはコア部分に含めるべきでないパッケージの除去
- セキュリティの向上 (未使用のクラスを除去することでプラットフォームの攻撃対象領域

問題は、これらすべての目標を達成するためには、Javaプラットフォームのコア部分の大手術が必要になることです。特に、クラスローディングについて

Compactプロファイルの最小単位はパッケージです。いくつ

最小のCompactプロファイルはcompact1と呼ばれます。このプロファイルを構成するパッケージは、java.io、java.lang、java.lang.annotation、java.lang.invoke、java.lang.ref、java.lang.reflect、java.math、java.net、java.nio、java.nio.channels、java.nio.channels.spi、java.nio.charset、java.nio.charset.spi、java.nio.file、java.nio.file.attribute、java.nio.

小規模化

Java 8では
Compactプロ
ファイルの概念
が導入されま
す。Compactプ
ロファイルとは、
縮小版のJava
ランタイム環境
(JRE) であり、
通常の完全な
rt.jarの内容の一
部で構成されま
す。

Stripped Implementationについて

Compactプロファイルに加えて、デプロイされたJVMアプリケーションのリソース使用率を削減するためのStripped Implementationという新しい技術が提案されています。Stripped Implementationとは、アプリケーションとともにパッケージ化され、そのアプリケーションにより独占的に利用される縮小版のJREです。そのアプリケーションだけがStripped Implementationのクライアントであるため、アプリケーションで使用されな

CompactプロファイルとStripped Implementationには大きな違いがあります。Compactプロファイルは汎用的に設計されたJavaランタイムであり、Java言語仕様の完全な実装です。一方、Stripped Implementationは、特定用途むけに使用されるものとして定義された再利用不可能な実装であり、ユーザーに提供する時点でJava言語仕様に準拠している必要はありません。Stripped ImplementationはかつてJava SE 8に組み込まれる予定でした。しかし、ライセンスや現在のテスト・キットの問題によって、リリース間近に断念せざるをえませんでした。それでも、Stripped Implementationは、Java 8のリリース後まもなくJavaプラットフォームに加えられる予定です。こ

このアプローチは、リソースの制約が厳しい場合に効果的です。このストリップ（除去）段階で、アプリケーションにより利用される可能性のある機能が除去されていないことを保証

まとめ

Java 8のCompactプロファイルは、組み込み開発（あるいはリソースに制約のある開発）とサーバー・サイド開発のいずれにおいても、今後のJavaプラットフォームの目標に向けた大きな一歩となります。Java 8に期待されていた完全なモジュール化を実現するソリューションではないものの、Compactプロファイル自体が価値のある前進です。

- 『Compact Profiles Demonstrated』
- Hinkmond Wongによる、Compact プロファイルに関するEclipseCon 2014でのプレゼンテーション





RAOUL-GABRIEL
URMA

パート2

Java SE 8ストリームを使用したデータ処理

Stream APIの高度な操作を組み合わせて、データ処理における表現豊かな問合せを記述する

**Raoul-Gabriel
Urma:** ケンブリッ
ジ大学において
計算機科学の博
士号を取得見込。
大学ではプログラ
ミング言語の研究
に従事。『Java 8 in
Action: Lambdas,
Streams, and
functional-style
programming』
(Manning、2014
年)の著者。

本シリーズの第1回では、**ストリーム**を使用することで、コレクションをまるでデータベース操作のように処理できることを確認しました。復習として、Stream API を使用して高額取引の金額のみを合計する方法を**リスト1**に示します。この例では、**図1**のように、中間操作(**filter**、**map**)と終端操作(**reduce**)から成る、一連の処理の流れを示しています。

しかし、第1回では以下の2つの操作は扱いませんでした。

- flatMap:「マップ」操作と「フラット化」操作を組み合わ

- **collect**：ストリームの要素を蓄積して集計結果を生成するための各種操作を定義したオブジェクト（コレクタと呼ばれる）を引数に取る終端操作

これら2つの操作は、複雑な問合せの表現に便利です。たとえば、`flatMap`と`collect`を組み合わせて、単語のストリームに含まれる各文字の個数を表す `Map` を生成できます (**リスト 2**)。最初のうちはこのコードが難しく感じるとしても、心配は不要です。本記事は、これら2つの操作について詳しく説明することを目的としています。

リスト 2 のコードの出力は
リスト 3 のようになります。
すばらしいと思いませんか。
これより、 flatMap 操作と
collect 操作の仕組みを詳しく
見ていくことにします。

flatMap 操作

あるファイルに含まれる一意の単語をすべて検索したい場合、皆さんならどうしますか。

簡単だと思う人もい
しょう。前回の記事で確
認した `Files.lines()` を使
用すれば、ファイルの行
から成るストリームを
取得できるためです。
次に `map()` 操作で各行
を複数の単語に分割し、
`distinct()` 操作で重複
を除外すればよいはず
です。この一連の操作
を記述すると、リスト
4 のようになります。

しかし、この考え方は誤りです。このコードを実行した場合、以下のような判読不可能な結果が出力されます。

```
[Ljava.lang.String;@7cca494b
[Ljava.lang.String;@7ba4f24f
```

ここで実際に出力されているのは、複数のストリームの `String` 表現です。なぜでしょう。このアプローチの問題点は、`map` メソッドの引数として渡された、ラムダ式 (`line -> line.split("\\s+"))` の結果が、ファイルの各行を表す `String` の配列 (`String[]`) を返すことです。そのため、`map` メソッドの返すストリームは、実際には `Stream<String[]>` 型になります。一方、ここで取得したい結果は、複数の単語から成るストリームの `Stream<String>` です。

幸いにも、この問題は `flatMap` メソッドを使用することで解決できます。以降で、正しい解決策について順を追って説明します。

まず、配列のストリームではなく単語のストリームが必

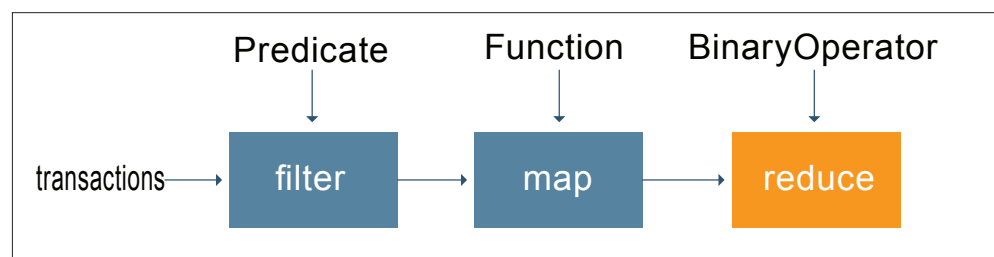


图1

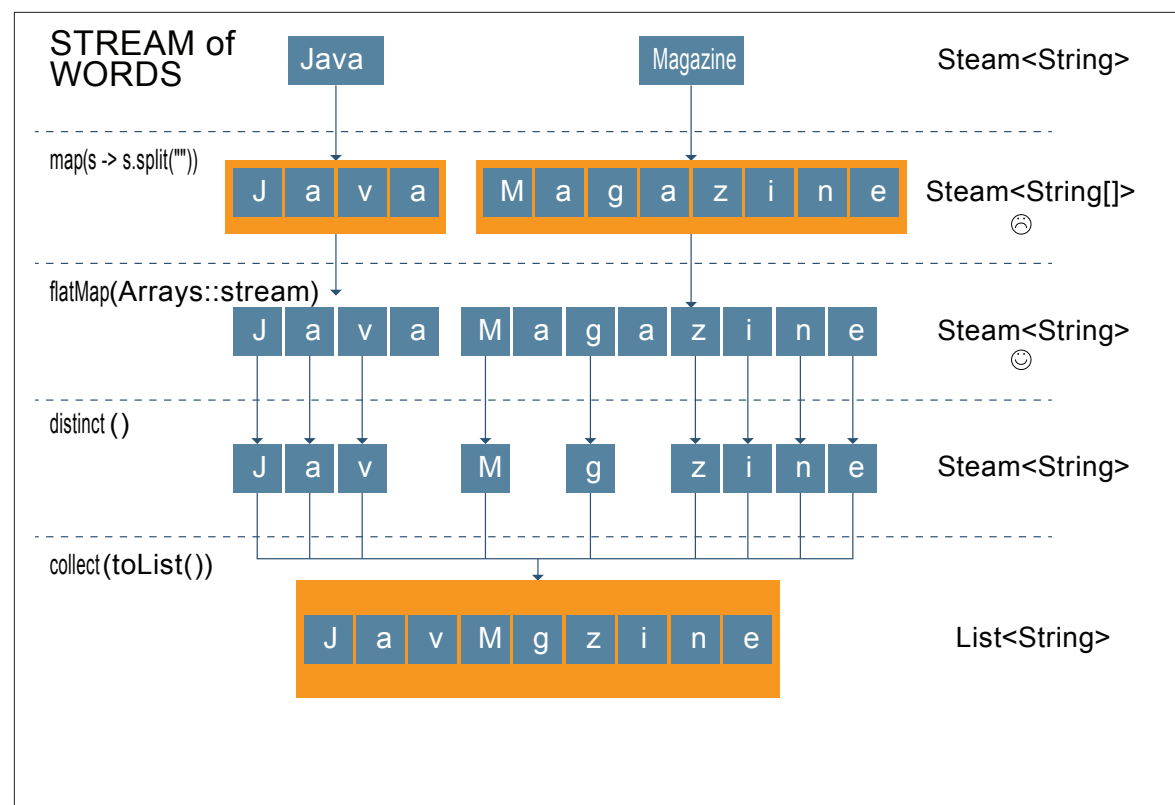


图2

要です。配列を引数に取ってストリームを生成する、`Arrays.stream()` というメソッドが存在します。このメソッドは、**リスト 5** の例のように使います。

では、この `Arrays.stream()` を前述のストリーム・パイプラインで使用して、結果を確認してみましょう（**リスト 6**）。この方法でも、やはり、必要な結果は得られません。今度は、ストリームで構成されるストリームのリスト（正確には、`Stream<Stream<String>>`）が生成されるからです。この例ではまさに、各行を単語の配列に変換した後に、`Arrays.stream()` メソッドで各配列を個別のストリームに変換しています。

この問題は、`flatMap` を使用することで修正できます（リスト 7）。`flatMap` メソッドを使用した場合、生成されたそれぞれの配列が、ストリームではなくそのストリームの内容に置き換えられます。すなわち、`map(Arrays::stream)` によって生成された個別のストリームがすべて、1 つのストリームに融合（「フラット化」）されます。図 2 で `flatMap` メソッドを使用する効果を説明します。

簡単に言うと、`flatMap` によりストリームの各値が別のストリームに置き換えられ、生成されたすべてのストリームを連結した 1 つのストリームが生成されます。

flatMap は、よく使用されるパ

リスト1 / リスト2 / リスト3 / リスト4 / リスト5 / リスト6

```
int sumExpensive =
    transactions.stream()
        .filter(t -> t.getValue() > 1000)
        .map(Transaction::getValue)
        .reduce(0, Integer::sum);
```

 [すべてのリストのテキストをダウンロード](#)

ターンです。今後、Optional や CompletableFuture を扱う際にも使用することになるでしょう。

collect 操作

次に、`collect` メソッドについて詳しく見ていきます。本シリーズの第1回で確認した操作は、別のストリームを返すもの（中間操作）か、`boolean`、`int`、`Optional` オブジェクトなどの値を返すもの（終端操作）でした。

`collect` メソッドも終端操作ですが、ストリームをリストに変換するために使用しているという点で、単なる値の取得とは少し異なっています。たとえば、すべての高額取引について ID のリストを取得するためには、**リスト 8** のコードを使用できます。

`collect` に渡す引数は、`java.util.stream.Collectors` 型のオブジェクトです。`Collector` オブジェクトの役割は何でしょうか。基本的には、

ストリームの要素を蓄積して最終結果を生成するための操作方法を表すことです。先ほど使用したファクトリ・メソッドの `Collectors.toList()` は、ストリームをリストに蓄積する方法を表す `Collector` を返します。このような組込みの `Collector` が多数提供されています。

ストリームを集積して別のコレクションに変換：たとえば、`toSet()` を使用すれば、ストリームを、重複要素を除いた `Set` に変換できます。**リスト 9** のコードは、高額取引のあった都市のみのセットを生成する方法を示しています（注：以降の例では、`Collectors` クラスのすべてのファクトリ・メソッドを、`import static java.util.stream.Collectors.*` によって静的にインポートするものと想定します）。

注意点として、いずれの型の `Set` が返されるかは保証されません。しかし、`toCollection()` を使用すれ

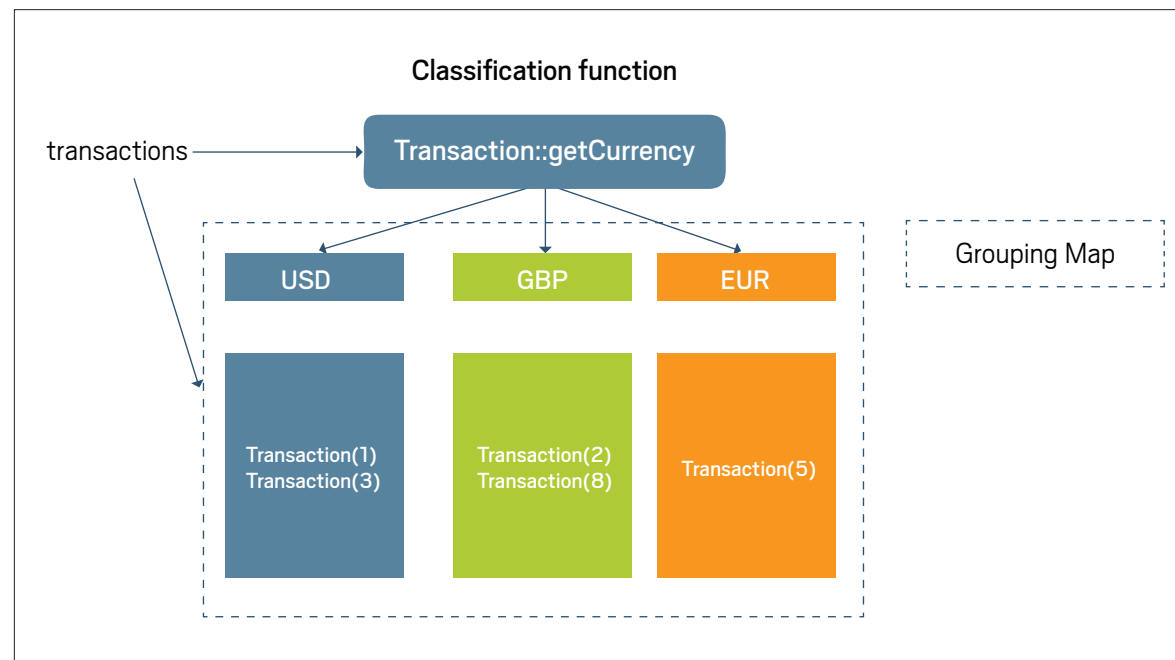


图4

す。

グループ化 (Grouping) : よく使用されるデータベース問合せに、プロパティを使用した、データのグループ化があります。たとえば、取引のリストを通貨別にグループ化できます。そのような問合せを明示的なイテレーションによって表現することは、**リスト 16** のコードから分かるとおり、かなり面倒です。

まず、取引を蓄積するための **Map** を作成する必要があります。次に、取引のリストのイテレーションで、各取引の通貨を抽出する必要があります。さらに、その取引を **Map** の値として追加する前に、リストが作成済みかをチェックする必要もあります。このような操作が続くのです。基本的には「取引を通貨別にグループ化し

たい」だけなのに、これほど大量のコードが必要となるのは合理的ではないでしょう。幸いにも、このような操作を簡潔に表現できる `groupBy()` というコレクタがあります。このコレクタを使用すれば、同じ問合せを **リスト 17** のように表現でき、コードから問合せの内容を読み取ることも容易になります。

ファクトリ・メソッドの `groupBy()` は、取引の分類に使用するキーを抽出するための関数を引数に取ります。このような関数のことを分類関数と呼びます。この例では、メソッド参照の `Transaction::getCurrency` を渡して、取引を通貨別にグループ化しています。**図 4** に、このグループ化操作について示します。

分割 (Partitioning) : 別のファクトリ・メソッドの `partitioningBy()`

リスト13 / リスト14 / リスト15 / リスト16 / リスト17 / リスト18

```
double average = transactions.stream().collect(
    averagingInt(Transaction::getValue));
```

 すべてのリストのテキストをダウンロード

は、`groupBy()` の特殊ケースと見なすことができます。このメソッドは条件（`boolean` を返す関数）を引数に取り、その条件を満たすグループと満たさないグループにストリームの要素を分けます。コード例に当てはめて具体的に言えば、取引から成るストリームを分割することは、取引を 1 つの `Map<Boolean, List<Transaction>>`

にマップすることになります。たとえば、取引を高額と低額という2つのリストに分ける場合に `partitioningBy` コレクタを使用できます (**リスト 18**)。このコード例の `t -> t.getValue() > 1000` というラムダ式が、高額取引と低額取引を分類する条件に該当します。

コレクタの合成：SQLをよく知っている開発者は、GROUP BYを

COUNT() や SUM() などの関数と組み合わせることで、取引を通貨別に合計してグループ化できることをご存じでしょう。Stream API でも、同様の操作を実行できます。実際に、別のコレクタ・オブジェクトを第 2 引数に取る、`groupingBy()` のオーバーロード・メソッドが存在します。この追加のコレクタは、`groupingBy` コレクタによって、各キーに関連付けられたすべての要素をどのように蓄積するかを定義するために使用します。

この説明では少し抽象的ですので、単純な例を見ていきましょう。都市の **Map** を生成し、都市ごとの取引総額を対応付けることにします (**リスト 19**)。このコード例では、**groupingBy** に対して、**getCity()** メソッドを分類関数として使用する

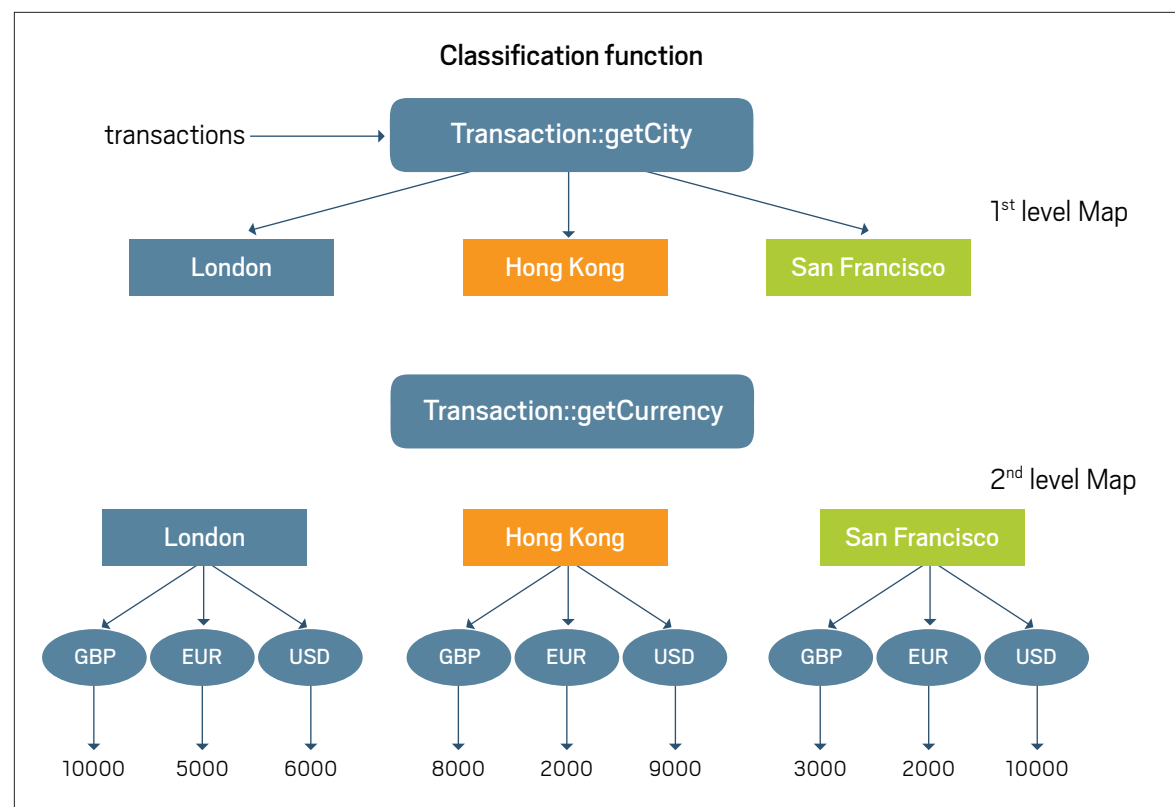


图5

ように指定しています。そのため、生成される `Map` のキーは都市です。通常の基本的な `groupBy` を使用する場合には、`Map` の各キーに対して、`List<Transaction>` が値として返されると期待します。

しかしここでは、追加コレクションの `summingInt()` も渡しています。このメソッドにより、都市に関連付けられた取引の総額が計算されます。その結果、都市と、その都市の取引総額をマップした `Map<String, Integer>` が返されます。実に優れた表現方法です。基本的なバージョンの `groupingBy(Transaction::getCity)`

は、単に `groupBy(Transaction::`

getCity, toList()) の省略形だと考えてください。

別の例を見てみましょう。都市ごとの最高額の取引から成る `Map` を生成する場合はどうでしょうか。ご想像のとおり、この場合は先ほど定義を示した `maxBy` コレクタを再利用できます (**リスト 20**)。

Stream APIは本当に表現豊かであり、興味深い問合せを簡潔に記述できます。コレクションの繰り返し処理に戻ることはもうできないでしょう。

最後に、さらに複雑な例を見ていきます。すでに説明したとおり、`groupBy` は、追加の分類条件に従って要素を蓄積するために、別のコレクタ・オブジェクトを引

リスト19 / リスト20 / リスト21

```
Map<String, Integer> cityToSum =
    transactions.stream().collect(groupingBy(
        Transaction::getCity, summingInt(Transaction::getValue)));
```

 すべてのリストのテキストをダウンロード

数に取ることができます。そして、`groupingBy` 自体もコレクタであるため、ストリームの要素を分類するための第 2 の条件を定義する別の `groupingBy` コレクタを渡して、多段階のグループ化を行うことができます。

リスト 21 のコードでは、まず取引を都市別にグループ化し、さらに、それぞれの都市での取引をその通貨別にグループ化して各通貨の取引額の平均値を取得しています。**図 5** に、このメカニズムについて示します。

独自のコレクタの作成：前項までに紹介したコレクタはすべて、`java.util.stream.Collectors` インタフェースの実装クラスです。したがって、開発者が独自のコレクタを実装して、「カスタマイズされた」リデュース操作を定義することも可能です。ただし、独自のコレクタの実装は別の記事を十分書けるほど大きなテーマであるため、本記事では取り上げません。

まとめ

本記事では、Stream API の高度な操作である `flatMap` と `collect` の

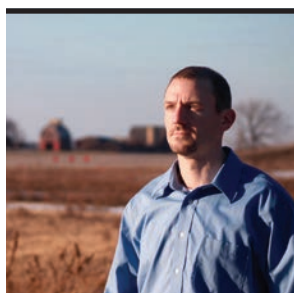
2 つについて詳しく説明しました。
いずれも表現豊かなデータ処理の
問合せを記述するために開発者が
習得しておきたいツールです。

特に、`collect` メソッドを使用すれば、集計、グループ化、分割の各操作を表現できることを確認しました。さらに、これらの操作を組み合わせて、たとえば「各都市での通貨別の平均取引額を格納した2階層の `Map` を生成する」といった、いっそう表現豊かな問合せを作成できます。

本記事では標準で利用できるすべての組み込みコレクタを確認したわけではありません。ぜひ皆さんで `Collectors` クラスを調査し、`mapping()`、`joining()`、`collectingAndThen()` などの有益であろうコレクタを試してみてください。

LEARN MORE

- 『Java 8 in Action: Lambdas, Streams, and functional-style programming』



JOSH JUNEAU

Josh Juneau：おもに Java、PL/SQL、Jython/Python を利用するアプリケーション開発者、システム・アナリスト、DBA。Jython Monthly ニュースレター、Jython Podcast、[Jython Web サイト](#)を管理している。『Java EE 7 Recipes』（Apress、2013 年）、『Introducing Java EE 7』（Apress、2013 年）の著者。現在は『Java 8 Recipes』（本年後半に Apress より出版予定）を執筆中。



HTML5とJSF

2つのテクノロジーを自在に組み合わせて洗練されたアプリケーションを作成する

本記事では、JavaServer Faces (JSF) を利用する新規または既存の Java EE アプリケーションへの HTML5 の統合に焦点を当てます。既存の JSF ビューに HTML5 コンポーネントを追加する方法と HTML5 マークアップを使用して JSF ビュー全体を作成する方法の基本を説明します。

また、サーバー・サイド・テンプレートの開発についても取り上げます。この機能により、1つのアプリケーション内にある複数のビューのルック・アンド・フィールを変更しやすくなります。WebSocket と JSON-P を使用し、HTML5 コンポーネントを介してサーバーとデータを送受信するテクニックを、サンプルにより確認します。さらに、これらのソリューションを NetBeans IDE で構築する方法について学習します。

本記事のサンプルでは、同じ Web フォームを、標準 JSF マークアップ、HTML5 のみ、および JSF と HTML5 の両方を使用して生成する方法について示します。また、HTML5

要素を JSF コンポーネントとして（または JSF コンポーネントを HTML5 要素として）シームレスに処理する方法についても示します。本記事を最後まで読めば、HTML5 のコンポーネントやテクノロジーを JSF アプリケーションに統合するために必要な基礎知識を得られます。

HTML5 互換マークアップ

Web アプリケーションの開発時に、あらかじめ決定しておくべき事項は多数あります。「アプリケーションでどのような機能を提供するか」、「どのようなユーザーがアプリケーションを利用するか」、「どのデバイスからアプリケーションを利用するか」。これらの問いは、利用するテクノロジーや Web フレームワークを選択する前に答えを出しておくべきもののほんの一部です。

今日の Web アプリケーション開発に利用できるフレームワークやテクノロジーは数百種類に上ります。JSF は、その中でも上位に入る十分な成熟度と堅牢さを誇る仕様であり、状

態管理機能を備えた洗練されたサーバー・サイド・アプリケーションの強固な基盤となります。また、JSF には、洗練されたユーザー・インタフェースを構築するための手軽なコンポーネント群があり、実装の詳細を意識せずに利用できます。そのため、開発者は JavaScript や CSS の開発に長い時間をかけることなく、アプリケーションの構築に集中できます。

HTML5 は、さまざまなデバイスに合わせてシームレスにスケールングできる移植可能なアプリケーションを開発するための新標準です。JSF 2.2 では、HTML5 ビューと JSF ビュー間の双方向性が実現するため、堅牢で信頼性の高い Web アプリケーションの強固な基盤を維持しながら、HTML5 によるスケラビリティとモダナイゼーションの効果を得ることができます。JSF 2.2 では、JSF コンポーネント、HTML5 要素、あるいはそ

Acme World

Create Reservation

First:

Last:

Adults:

0

Children:

0

Days:

0

Trip Start:

Create a Reservation

A world created for all Java fans!

图1

の両方の使用を開発者が選択
できます。

JSF 2.2 では、パススルー要素、パススルー属性、またはこの両方を HTML ページ内に配置するために指定できる、新しい名前空間が導入されました。これらの新しいパススルー機能を利用すれば、HTML5 要素を JSF コンポーネントとして取り扱うことができます。また、JSF コンポーネントを、JSF コ

ンポーネントやレンダラによる解析を介さずにブラウザに直接渡す（パススルー）ことができます。HTML5 ページで標準の JSF ビューと同様に JSF バッキング Bean を利用できるため、JSF とのシームレスな双方向性が実現します。

標準的な JSF ビュー

HTML5 について確認する前に、まずは標準的な JSF ビューを見てみましょう。本記事で使用するサンプルは、Java 開発者が羽を休めるリゾート施設の「Acme World」を対象としたアプリケーションのサブセットです。本記事では、このリゾートの宿泊予約用フォームに焦点を当てます。JSF のみで構成されるビューには、標準の JSF コンポーネントと PrimeFaces コンポーネントが混在しています。留意点として、このビューには検証機能がありませんが、すべての Web アプリケーションにおいて、リリースの前に検証機能を追加しておくことは重要です。

リスト 1 は、JSF のみで構成される予約フォームのビューです。おわかりのように、このビューには予約フォームを構成する多数の PrimeFaces コンポーネント（具体的には `inputText`、`spinner`、`calendar`）が含まれています。

まれています。図 1 に、JSF のみで構成されるビューの外観を示します。

HTML5 のみで構成されるビュー

JSF マークアップのみを使用したフォームの生成方法がわかったので、今度は HTML5 について確認します。**リスト 2a、2b、2c** のフォームでは、JSF Facelets マークアップと HTML5 マークアップが混在しています。この Facelets マークアップはテンプレートとして使用されているだけであり、仮にこれらの Facelets マークアップが存在しなくても、このページは JSF エンジンにより正常に処理されます。JSF 2.2 ではパススルー要素が導入されたため、**リスト 2a、2b、2c** の HTML5 要素は JSF コンポーネントとして処理されます。

JSF で HTML5 要素を利用するためには、まずページに対してパススルー要素向けの `jsf` 名前空間 (`xmlns:jsf="http://xmlns.jcp.org/jsf"`) を追加します。この `jsf` 名前空間は、任意の HTML5 要素の属性に対して追加でき、この名前空間が追加された要素は JSF ランタイムによって処理されるようになります。以下の `<head>` 要素を見てください。

```
<head isf:id="head">
```

この `jsf:id="head"` という記述は、JSF エンジンに対して、このコンポーネントを `<h:head>` として解釈するように指示しています。同様に、入力コンポーネントを見ると、JSF 処理に不可欠な属性が JSF ランタイムにパススルーされることがわかります。た

リスト1 リスト2a リスト2b リスト2c

注:このリストは、紙面の都合上抜粋になっています。省略部分は...記号で示しています。コード・リストの全体は、本号のコード・リストをダウンロードしてご確認ください。

```
<h:form id=" parkReservationForm" >
  <h1>Create Reservation</h1>
  <br/>
  <h:messages id=" messages"
    infoStyle=" color: green;"
    errorStyle=" color: red;" />
  <br/>
  <h:panelGrid columns=" 2" >
    <label for=" firstName" >First:</label>
    <p:inputText id=" firstName"
      placeholder=" Enter First Name"
      value=" #{...firstName}" />
    ...

    <label for=" numAdults" >Adults:</label>
    <p:spinner id=" numAdults" min=" 1" max=" 15"
      value=" #{...numAdults}" />
    ...

    <label for=" startDate" >Trip Start:</label>
    <p:calendar id=" startDate"
      value=" #{...tripStartDate}" />

  </h:panelGrid>

  <h:commandButton id=" parkReservation"
    action=" #{parkReservationController.createReservation}"
    value=" Create a Reservation" >

  </h:commandButton>
</h:form>
```

 すべてのリストのテキストをダウンロード

自由に選択
JSF 2.2 では、JSF
コンポーネント、
HTML5 要素、あ
るいはその両方
の利用を**開発者
が選択**できます。

例えば、以下の `inputText` 要素では、`id` と `value` がパススルーされます。

```
<input type="text"
      jsf:id="firstName"
      jsf:value="#{...}"/>
```

このようにパススルー要素として指定した結果、この HTML5 要素は JSF で元々用意されているコンポーネントである `h:inputText` として扱われ、サーバー・サイドの `UIComponent` インスタンスと関連付けられます。JSF コンポーネントとして扱うためには、パススルーを示す名前空間の「`jsf`」を、少なくとも1つの要素に追加する必要があります。この名前空間の追加によって、標準の HTML5 要素で式言語 (EL) を使用できるようになり、Managed Bean のプロパティの取得と設定が可能になります。そのため、送信されたとき (または受信されたとき) に、値が Managed Bean に送信されて処理され

図 2 に、HTML5 ページの外観を示します。このコードでは HTML5 カレンダー要素を利用しています。この要素は、ブラウザによって表示が異なる可能性があります。

既存の JSF への

Acme World

Create Reservation

First:

Last:

Adults:

Children:

Days:

Trip Start:

Create a Reservation

February 2014

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	1

图2

HTML5 の挿入

JSF と HTML5 は親和性が高く、HTML5 要素をまるで標準の JSF コンポーネントのように既存の JSF ビューや JSF 関数に追加できます。しかし、JSF コンポーネントを利用しながら、さらにその JSF コンポーネントと同等の HTML5 要素にしかない属性を指定して拡張したい場合もあります。このような拡張は、JSF 2.2 でパススルー属性機能が導入されたことで可能となりました。パススルー属性にはパススルー要素とは逆の効果があります。パススルー属性を JSF コンポーネントに適用した場合、指定さ

リスト3

注:このリストは、紙面の都合上抜粋になっています。省略部分は...記号で示しています。コード・リストの全体は、本号のコード・リストをダウンロードしてご確認ください。

```
<h:form id=" parkReservationForm" prependId=" false" >
  <h1>Create Reservation</h1>
  <br/>
  <h:messages id=" messages" infoStyle=" color: green;"
    errorStyle=" color: red;" />
  <br/>
  <h:panelGrid columns=" 2" >
    <label for=" firstName" >First:</label>
    <h:inputText id=" firstName"
      p:placeholder=" Enter First Name"
      value=" #{...firstName}" />
    ...

    <label for=" numAdults" >Adults:</label>
    <h:inputText id=" numAdults" p:type=" number"
      p:min=" 1" p:max=" 15"
      value=" #{...numAdults}" />
    ...

    <label for=" startDate" >Trip Start:</label>
    <h:inputText p:type=" date" id=" startDate"
      value=" #{...tripStartDate}" >
      <f:convertDateTime pattern=" YYYY-MM-dd" />
    </h:inputText>

  </h:panelGrid>

  <h:commandButton id=" parkReservation"
    action=" #{...createReservation}"
    value=" Create a Reservation" >

</h:commandButton>
```

 すべてのリストのテキストをダウンロード

ンプルでは、ユーザーがチャットのユーザー名を入力して「Start Chat Session」ボタンをクリックすると、WebSocket 通信チャネルが開きます。通信チャネルが開いた後は、ユーザーはメッセージを入力してチャット・ルームに送信できるようになります。接続が確立されると、他のユーザーから WebSocket エンドポイントに送信されたすべてのメッセージが表示され始めます。

リスト 8 に、この単純なチャット・ルームの Facelets ビューのソース・コードを示します。このソース・コードは、PrimeFaces の `commandButton` コンポーネント 3 つと、HTML5 のテキスト入力要素 2 つで構成されています。このように JSF と HTML5 を組み合わせて、JSF バッキング Bean および WebSocket エンドポイントとのやり取りを行っています。また、エンドポイントから返されるチャットの出力は、フォームの下部にある `div` 内に表示されます。

PrimeFaces の `commandButton` コンポーネントは、`action` 属性を使用して JSF ランタイムと通信します。この `action` 属性は、`ChatController` というバックング Bean にバインドされます。また、`commandButton` コンポーネントは、JavaScript の `onclick` イベントを利用して WebSocket 接続と通信します。リスト 9a と 9b に、ボタンのイベントによって呼び出される JavaScript を示します。たとえば、「Chat」ボタンがクリックされたときには、JavaScript 関数の `sendChatMessage()` が呼び

出され、対応するテキスト・ボックス内のメッセージが WebSocket エンドポイントに送信されます。

WebSocket エンドポイントのクラスは `ChatServerEndpoint` という名前
で識別されます (**リスト 10a**、**10b**)。
`@ServerEndpoint` アノテーションによ
り、このクラスを WebSocket エンド
ポイントとしてマークしています。ま
た、`encoders` 属性と `decoders` 属
性を使用して、メッセージの変換
に使用できるクラスを指定していま
す。`value` 属性には、クライアントか
らアクセス可能なエンドポイントを
示すパスが含まれています。このサ
ンプルでは、`ws://localhost:8080/
JavaMagazine-HTML5JSF/chat` とい
うパスで、この WebSocket にアクセ
スできます。

クライアントの JavaScript 関数である `sendChatMessage()` は、メッセージを JSON オブジェクトにエンコードします。具体的には、ユーザー名とメッセージを結合して、名前と値のペアで構成される JSON 文字列を生成します。その後、生成した JSON 文字列を `ChatServerEndpoint` に送信します。エンドポイントがメッセージを受信すると、`@OnMessage` アノテーションが付加されたメソッドが呼び出されますが、このメソッド自体はメッセージの変換を行いません。メッセージの受信時にデコーダ (**リスト 11**) により JSON-P API を使用してオブジェクトが解析され、結果のメッセージはエンコーダ (**リスト 12**) によって、すべてのクライアントに送信される表示可能な String に変換さ

リスト8 リスト9a リスト9b リスト10a リスト10b

```
<form jsf:id=" chatForm" jsf:prependId=" false" >
  Please enter a username to chat (a valid email address).
  <br/><br/>
  <label>Username:</label>
  <input type=" text" jsf:id=" username"
    jsf:value=" #{chatController.current.username}" />
  <br/><br/>
  <p:commandButton onclick=" initiateChatSession();"
    update=" sendMessage jsfOutput"
    value=" Start Chat Session"
    action=" #{chatController.startSession}" />
  <p:commandButton onclick=" closeChatSession()"
    update=" sendMessage jsfOutput"
    value=" Close Chat Session"
    action=" #{chatController.closeSession}" />
  <br/><br/>
  Message:<br/>
  <input type=" text" jsf:id=" message"
    jsf:value=" #{chatController.current.message}" />
  <br/>
  <p:commandButton value=" Chat" id=" sendMessage"
    disabled=" #{!chatController.sessionOpen}"
    onclick=" sendChatMessage()"
    action=" #{chatController.sendMessage}" />
  <hr/>
  Session Content:<br/>
  <div id=" output" class=" chatOutput" >
    <h:outputText id=" jsfOutput"
      value=" #{chatController.chatOutput}" />
  </div>

</form>
```

 すべてのリストのテキストをダウンロード

NetBeans IDE では HTML5 プロジェクト・タイプも利用できます。HTML5 プロジェクトでは、AngularJS、Knockout などの各種ライブラリが完全にサポートされません。Cordova アプリケーションを開発してモバイル・デバイスに直接デプロイすることも非常に簡単です。以上は NetBeans による開発で利用で

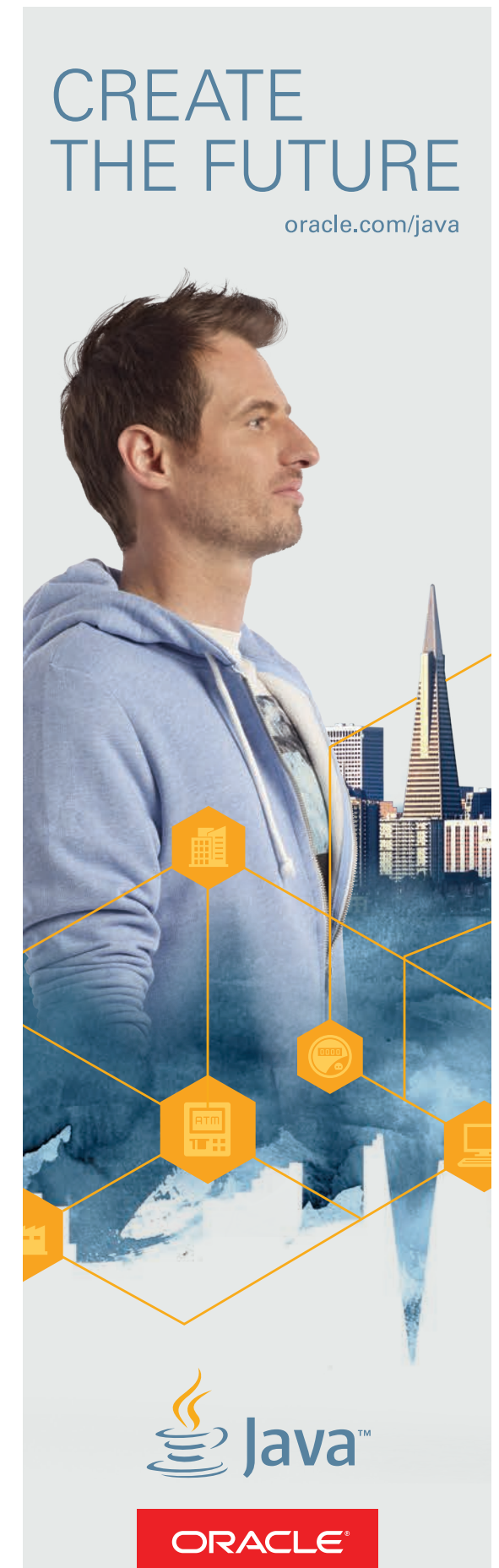
JSF は、Java Web 開発において重要な役割を果たします。Java EE 7 のリリースによって、JSF は HTML5 とシームレスに統合されるように強化されました。そのため、JSF 開発者は HTML5 の機能を活用でき、逆に

JSF は、WebSocket や JSON-P などの HTML5 対応 API とスムーズに連携します。本記事では、これらのテクノロジーを統合したソリューションのサンプルを示し、そのようなソリューションを構築するために NetBeans などの IDE を活用する方法について説明しました。</article>



Development
Tools and
Techniques

- [Java EE 7チュートリアル](#)のHTML5の項
- [NetBeans](#)のHTML5サポート
- PrimeFaces





HARSHAD OAK

Harshad Oak :
IndicThreads の
設立者。Oracle
JDeveloper や
Jakarta Commons
に関する著書
あり。『Java 2
Enterprise Edition
1.4 Bible』(Wiley、
2003 年) の共著
者でもある。現
在は『Java EE
Applications on
the Oracle Java
Cloud』(Oracle
Press) を執筆中。
Java Champion、
Oracle ACE
Director であり、
インドのプネーを
拠点として活動し
ている。



NetBeans IDE でのビルドと Oracle Java Cloud Service へのデプロイ

アプリケーションのデプロイにかかる時間と労力を削減する

以前の Java Magazine の記事では、Java PaaS（Platform as a Service）プロバイダの選択方法を確認した後、Oracle Java Cloud Service について紹介しました。本記事では、NetBeans IDE を利用して迅速にアプリケーションを構築し、Oracle Java Cloud Service にデプロイする方法について確認します。

Java PaaS プラットフォームの大半は、WAR ファイルや EAR ファイルにパッケージ化したアプリケーションをデプロイする機能を備えた Web ダッシュボードを提供しています。また、PaaS プロバイダは、ビルドおよびデプロイのプロセスを効率化する Ant ツールや Maven ツールを提供しています。Oracle Java Cloud Service などの一部のプロバイダではさらに一歩進んで、[NetBeans](#)、[Eclipse](#)、[Oracle JDeveloper](#) といった複数の人気の IDE と統合できるようになっています。IDE との統合は時間と労力の大幅な削減につながる

ため、開発者にとって望ましい機能です。本記事では、新しい NetBeans IDE 8 と Oracle Java Cloud Service の利用に焦点を当てて説明します。

注：本記事で紹介するアプリケーションのソース・コードは[こちら](#)からダウンロードできます。

Oracle Java Cloud Service SDK のインストール

NetBeans との統合の前に、まずは Oracle Java Cloud Service SDK (Software Development Kit) をセットアップする必要があります。Oracle Java Cloud Service を操作するすべての IDE で、この SDK が利用されるためです。

この SDK には、Oracle Java Cloud Service でのアプリケーションの開発、デプロイ、管理に利用できる各種ツールが付属しています。たとえば、Oracle Java Cloud Service の操作用コマンド、アプリケーションがデプロイ可能な状態かを確認するホ

ホワイトリスト・ツール、Ant タスク、Maven プラグインが提供されています。

この SDK は、アプリケーション・コード内で使用しなければならないクラスやライブラリのセットというわけではありません。

この SDK をインストールするためには、[Oracle Technology Network](#) から最新バージョンをダウンロードして、任意のディレクトリに解凍します。

Oracle Java Cloud Service 試用版の取得

次に、30 日間の Oracle Java Cloud Service 試用版を cloud.oracle.com よりインストールします。試用版を登録すると、試用のためのデータセンター、ユーザー名、パスワード、アイデンティティ・ドメインに関する

クラウドへすべての開発者が**複数のクラウド・サービスと統合された IDE** を利用してソフトウェアを構築するようになるのも時間の問題です。

情報が届きます。これらの情報は、後ほど NetBeans から Oracle Java Cloud Service を利用する手順で必要になるため、大切に保管してください。

この試用版には、
Oracle Database
Cloud Service 試用
版も含まれています。
そのため、Oracle
Database Cloud
Service 試用版を別途
入手する必要はありま
せん。

NetBeansのセットアップ

ブ

NetBeans の数ある特長の 1 つに、インストールの容易さがあります。 netbeans.org から、NetBeans の Java EE バージョンまたは「すべて」バージョンをダウンロードして NetBeans をインストールすれば、標準機能のすべてをすぐに利用できます。

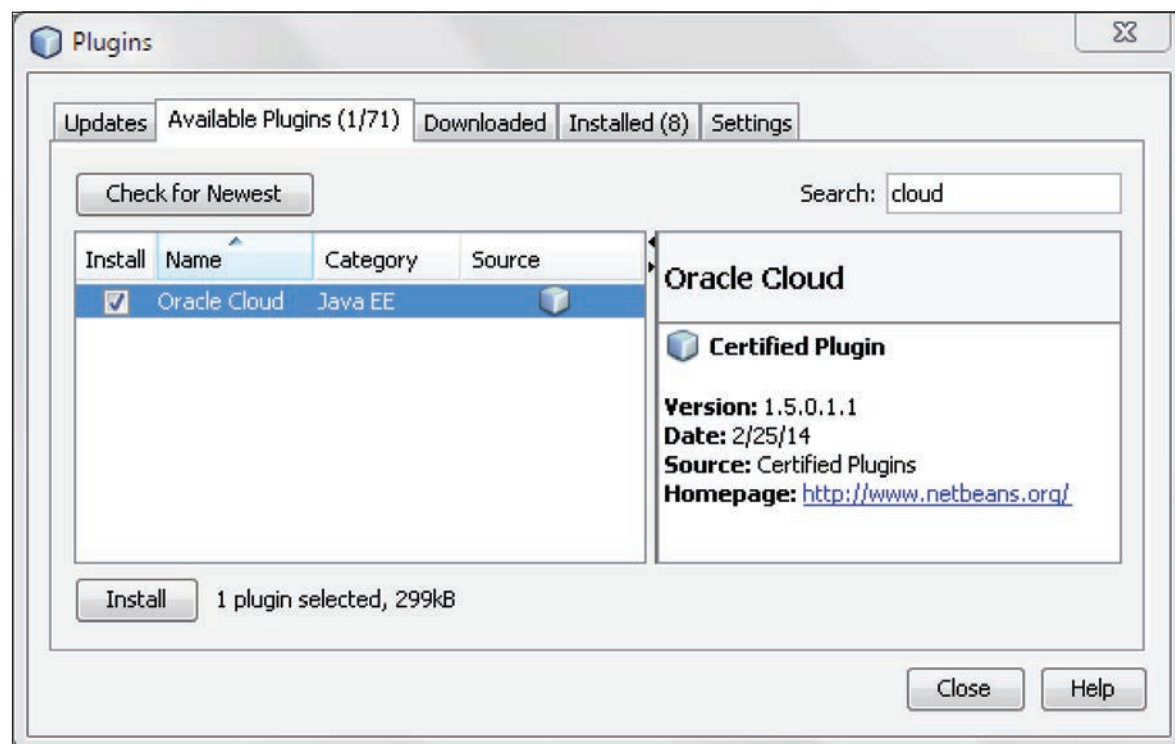


図1

とは言っても、「すべて」バージョンをインストールした場合であっても、すべての機能がただちにアクティブ化され、リソースが消費され始めるわけではありません。NetBeans の巧妙な「オンデマンド機能」により、各テクノロジーは実際に利用する際にのみ有効化されます。

Oracle Cloud プラグインのインストール

NetBeans のセットアップが完了したら、NetBeans 向けの Oracle Cloud プラグインをインストールする必要があります。そのためには、NetBeans で「Tools」→「Plugins」をクリックし、さらに「Available Plugins」タブをクリックします。このタブで cloud と入力して検索すると、該当するプラグイン

が表示されます（図1）。このプラグインのチェック・ボックスを選択した状態で、「Install」をクリックします。次に、NetBeans IDE Plugin Installer の手順に従って Oracle Cloud プラグインをインストールします。

プラグインをインストールすると、Services ウィンドウに新たに「Cloud」セクションが追加されます（図2）。Services ウィンドウが表示されない場合は、NetBeans メニューで「Window」→「Services」をクリックすることにより表示されます。

NetBeans メニューの「Window」→「Reset Windows」オプションを使用して、デフォルトのウィンドウや見慣れたルック・アンド・フィールにいつでも戻すことができます。

次に、「Cloud」を右クリックし、「Add

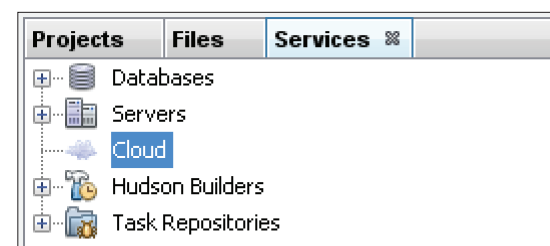


図2

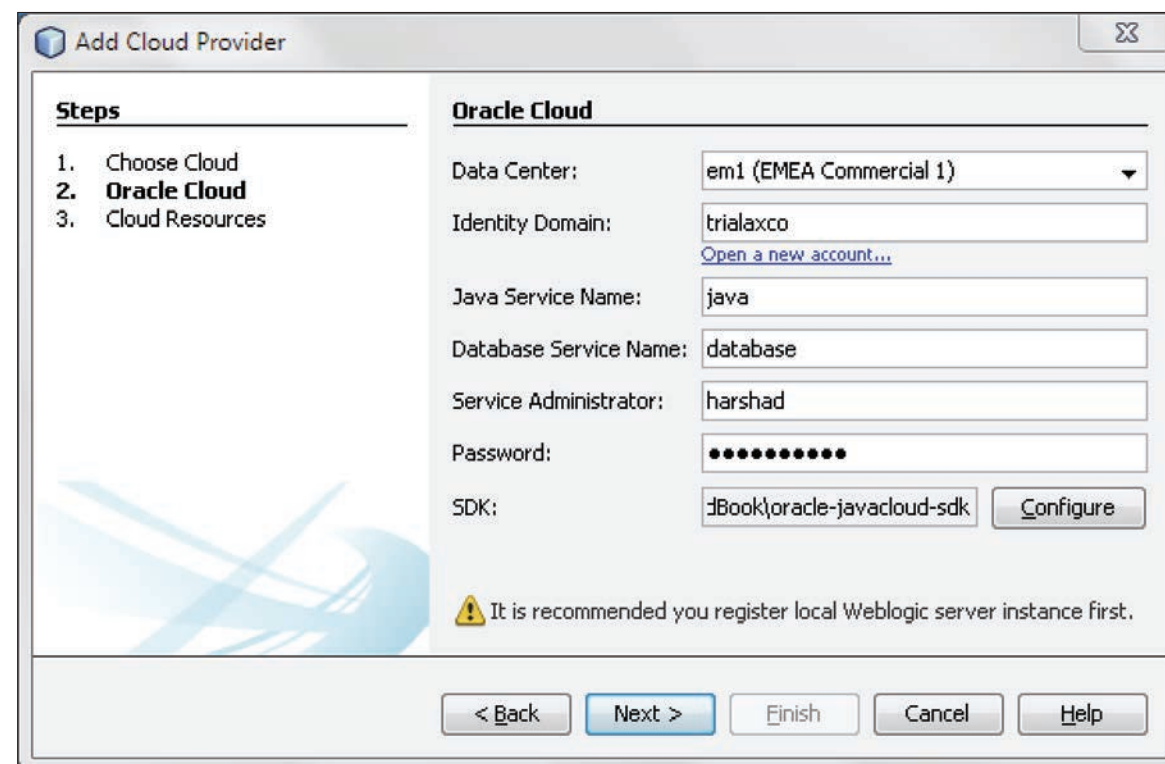


図3

Cloud」を選択します。Add Cloud Provider 画面が表示されます。「Oracle Cloud」を選択し、Oracle Java Cloud Service 試用版のインストール時に通知された情報を入力します（図3）。SDK フィールドには、Oracle Java Cloud Service SDK の解凍先ディレクトリのパスを指定する必要があります。また、Data Center リストに自分用のデータセンターが表示されない場合は、そのデータセンター名を入力します。

図3のように、NetBeans では、開発中に利用できるローカルの Oracle WebLogic Server インスタンスを登録することを推奨しています。開発中でもクラウド環境を利用すること自体は可能ですが、クラウド内よりもローカルの方が、コードのデプロイとデバッグがより速く簡単に実行されます。Oracle Java Cloud Service では、Oracle WebLogic Server 11g が稼働します。Oracle WebLogic Server 11g をこちらからダウンロードしてインストー

ルし、ローカル・サーバーとして実行
します。

クラウドのセットアップが完了したら、新しいサーバーと「welcome app」が表示されます（図 4）。このアプリケーションを右クリックして、表示、停止、またはアンデプロイを行うことができます。また、NetBeans から Jobs Log や Instance Log にアクセスすることもできます。そのためには、「**Oracle Cloud**」を右クリックし、さらに「**View Jobs and Logs**」をクリックします。

以上で NetBeans が Oracle Java Cloud Service に統合されたため、次に、Oracle Java Cloud Service および Oracle Database Cloud Service を利用する Java EE アプリケーションの構築方法を確認します。

注意点として、NetBeans のインストール時に「オンデマンド機能」を有効にした場合は、NetBeans の Java EE 機能を初めて利用する際、Java EE のアクティブ化に多少時間を要することがあります。また、インストール中に Apache Tomcat または GlassFish もインストールした場合は、この時点で、これらのサーバーが Services ウィンドウの **Servers** の下に表示されます。

Java EE アプリケーションの構築

まず、「File」→「New Project」→「Java EE」→「Enterprise Application」を選択します。プロジェクト名に **JavaMag** と入力し、「Next」をクリックします。New Enterprise Application 画面 (図 5) で、**Server** リストから「Oracle Cloud Remote」を選択しま

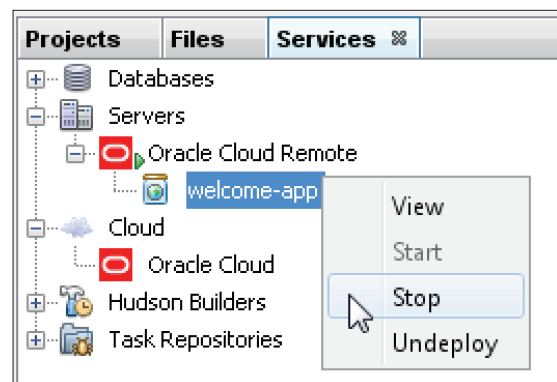


图4

す。

こちらの説明にあるとおり、Oracle Java Cloud Service で現在サポートされる Java EE テクノロジーは、バージョン 5 と 6 の機能が混在しています。NetBeans によりは Java EE 5（ソース・レベル 1.5）が選択されます。

次に、「**Create EJB Module**」
チェック・ボックスと「**Create Web
Application Module**」チェック・ボッ
クスを選択します。「**Finish**」をクリッ
クすると、モジュールが自動的に作成
されます。

サンプル・アプリケーションでは、JPA (Java Persistence API) エンティティ、ステートレス・セッション EJB (Enterprise JavaBean)、およびサーブレットを利用して、データベース内に新しい雑誌レコードを作成します。

まず、**Magazine** という新しい JPA エンティティを作成します。**JavaMag-ejb** プロジェクトを右クリックし、「**New**」→「**Persistence**」→「**Entity Class**」を選択します。**図 6** に示すように、**Class Name** フィールドに **Magazine** と入力し、**Package** リストから「**entities**」を選択し、**Create Persistence Unit**

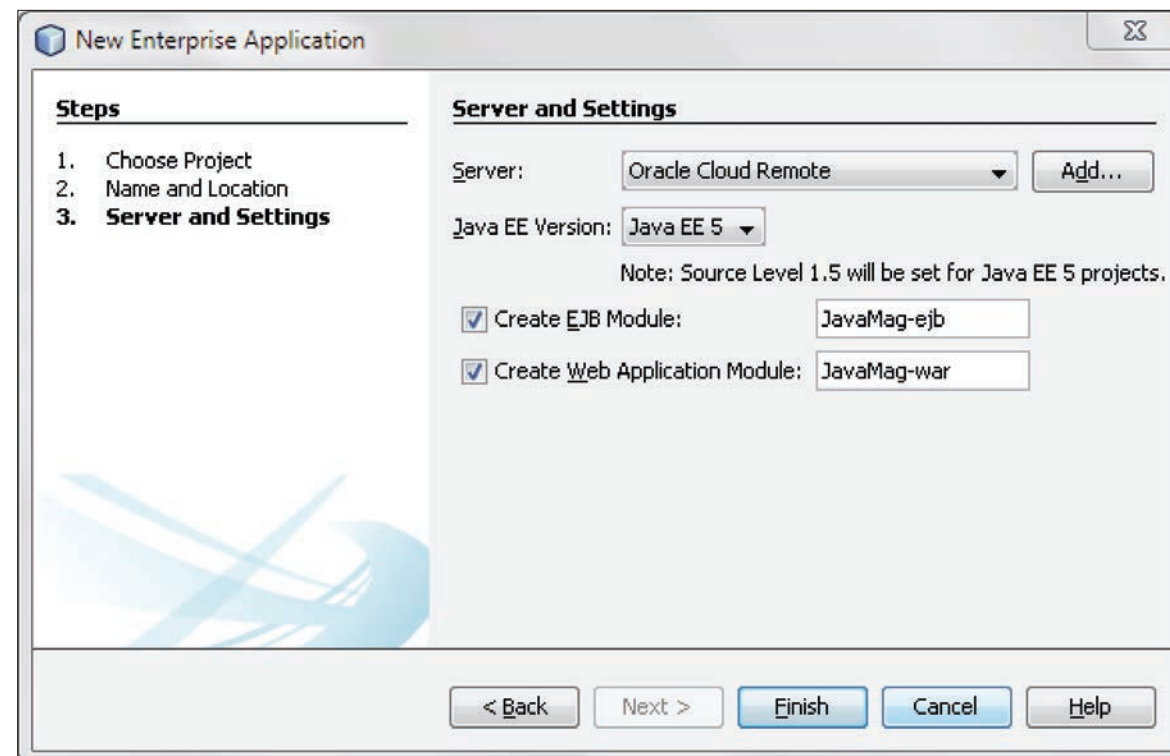


图5

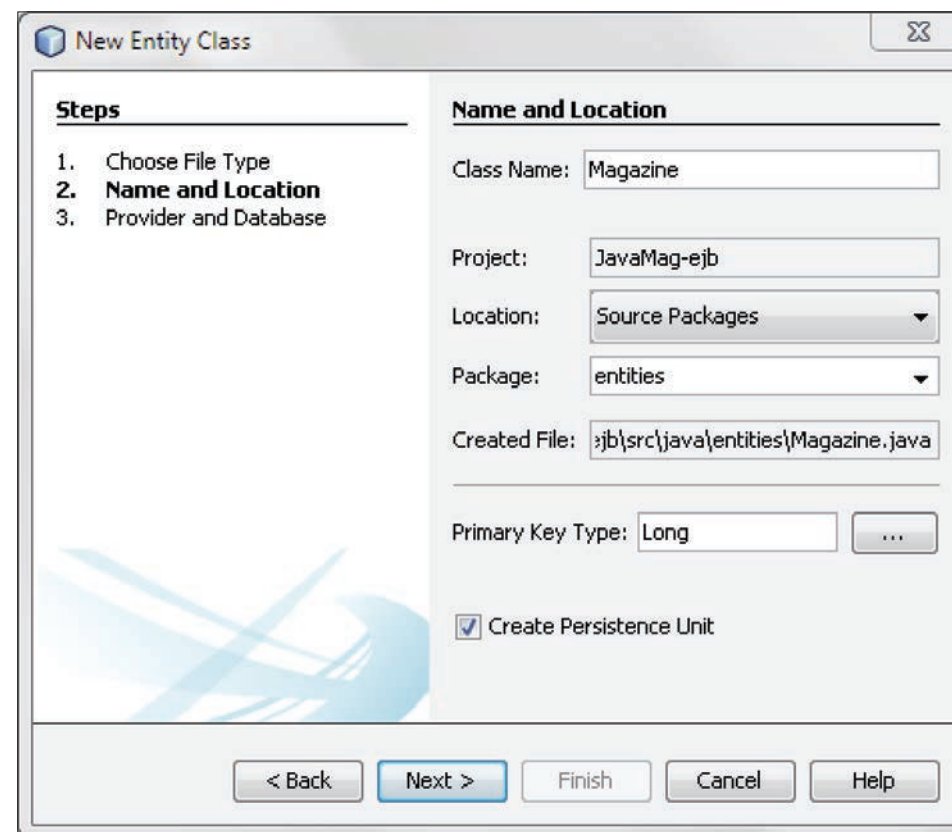


图6

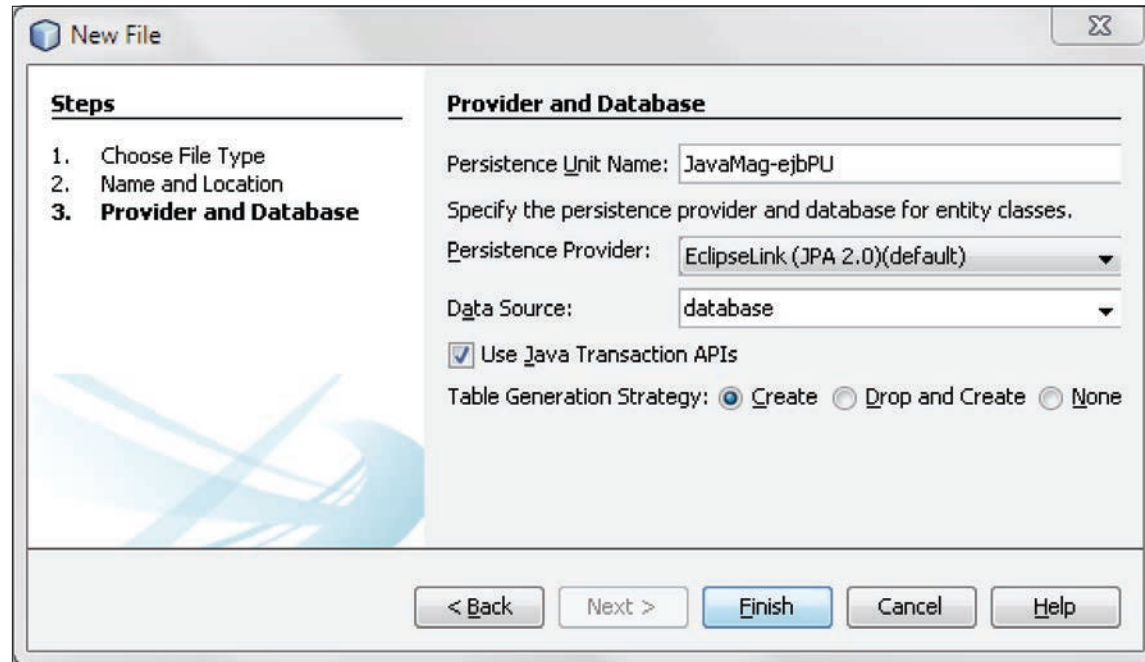


图7

チェック・ボックスを選択したままの状態にして、「**Next**」をクリックします。

New File 画面（**図 7**）で、Oracle Java Cloud Service 試用版とともに入手した Oracle Database Cloud Service 試用版のデータソース名を **Data Source** に入力します。Oracle Java Cloud Service の Web ベースのダッシュボードにログインした際には、こ

のデータソース名
が **Data Sources**
セクションに表示
されます。

Oracle Java Cloud Service では JPA 2.0 がサポートされるため、JPA 2.0 向けの EclipseLink が選択されます。

Table Generation Strategy では、

Create オプションがデフォルトで選択されます。このため、JPA エンティティの **Magazine** を介して新しい雑誌情報を永続化しようとした場合は、新しい Magazine 表が作成されます。

NetBeans により、パッケージ・エンティティ内に `Magazine.java` というエンティティ・クラス・ファイルが生成されます。また、永続性ユニットの設定が記載された `persistence.xml` ファイルも生成されます。

リスト 1 ([Magazine.java](#) の抜粋)

のように、このクラスがエンティティであることを表すための必須アノテーションがすべて自動的に追加されます。**リスト 1** の `id` プロパティに対する `GenerationType.AUTO` アノテーションは、永続性プロバイダによって、データベースに応じた適切なキー生成方法 (strategy) が選択されることを意味します。EclipseLink では通常、キー生成用に新しい表が作成されます。新

リスト1

リスト2

```
@Entity
public class Magazine implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;

    private String name;
```

 すべてのリストのテキストをダウンロード

しい表の作成は、複数のデータベースにわたって機能する方法であるためです。

次に、**Magazine** エンティティに **name** プロパティを追加します。そのため、コード内部を右クリックし、「**Insert Code**」→「**Add Property**」を選択します。開いた画面で、プロパティ名に **name** と入力し、getter メソッドと setter メソッドを生成するオプションを選択します。

次に、[Magazine](#) エンティティを利用するための新しいステートレス・セッション Bean を作成します。
JavaMag-ejb プロジェクトを右クリックし、「**New**」→「**その他**」→「**Enterprise JavaBeans**」→「**Session Bean**」を

選択します。

図 8 のように、この Bean 名に **AddMagazineBean** と入力し、**Package** リストから「**ejb**」を選択し、セッション・タイプとして「**Stateless**」を選択します。

Oracle Java Cloud Service でサポートされるのは EJB 3.0 であるため、EJB 3.1 で導入された「インタフェースなし」のオプションは利用できません。

また、Oracle Java Cloud Service ではローカルの EJB 呼出しのみがサポートされるため、作成するインタフェースのタイプとして「**Local**」を選択します。

「**Finish**」をクリックします。セッ
ション Bean の [AddMagazineBean](#) と、
[AddMagazineBeanLocal](#) インタフェー


```

JavaMag (run)  % Oracle Cloud Remote Deployment  %
Uploading...
Deploying.....
===== Log file: virus-scan=====

2014-03-11 00:39:12 CDT: Starting action "Virus Scan"
2014-03-11 00:39:12 CDT: Virus Scan started
2014-03-11 00:39:51 CDT: -----
2014-03-11 00:39:51 CDT: File Scanned:  "JavaMag.ear".
2014-03-11 00:39:51 CDT: File Size:      "9904765".
2014-03-11 00:39:51 CDT: File Status:   "CLEAN".
2014-03-11 00:39:51 CDT: -----
2014-03-11 00:39:51 CDT: Virus scan passed.
2014-03-11 00:39:51 CDT: "Virus Scan" complete: status SUCCESS

..
===== Log file: whitelist=====

2014-03-11 00:39:51 CDT: Starting action "API Whitelist"

```

图10



图11

しい表が作成され、新しい「Java Magazine」レコードがその表に挿入されたことを確認できます。

まとめ

本記事では、NetBeans をすばやくセットアップして使用し、Java EE アプリケーションを開発後、Oracle Java Cloud Service にデプロイする方法について確認しました。現在、クラウドベースのサービスが急速に採用されていることを考慮すると、すべての開発者がソフトウェア開発プロセスの各段階で、複数のクラウド・サービスと統合され

た IDE を利用してソフトウェアを構築
するようになるのも時間の問題です。

</article>

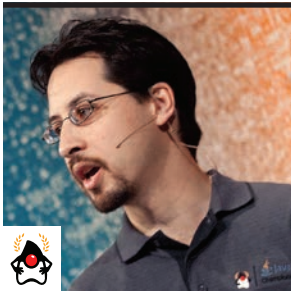
MORE ON TOPIC:



LEARN MORE

- [Oracle Java Cloud Service](#)
- [Oracle Database Cloud Service](#)
- [NetBeans IDE](#)
- Oracle Java Cloud Service SDK





STEPHEN CHIN

Stephen Chin (@steveonjava) :
オラクルの Java テクノロジー・アンバサダ、JavaOne Content Chair。代表的な JavaFX 参考書『[Pro JavaFX 2](#)』(Apress、2012 年)の共著者。Devovx、CodeMash、OSCON、J-Fall、GeeCON、JAZOON、JavaOne などの多数の Java カンファレンスで発表を行う。JavaOne では Rock Star を 2 回受賞。

メリーさんの「ラムダ」

単純なゲームを通じてラムダ式とStream APIに親しむ

ラムダ式は、Java SE 5 でのジェネリクスのリリース以来、Java 言語に加わった機能の中でもっとも影響力を持つものです。プログラミング・モデルを根本から変え、関数型の開発スタイルを実現するだけでなく、コードを効率的に並列化して、マルチコア・システムを活用できるようにします。しかし、Java 開発者の立場としては、Java SE 8 の新しいラムダ式対応型 API を利用することによる生産性向上に、まず目が行くでしょう。

本記事では、JavaFX で書かれたレトロなゲームを利用して、新しい [Stream API](#) によるコレクションやデータの操作方法を見ていきます。このゲームは、ラムダ式のベスト・プラクティスを紹介するために一から開発されたシンプルな Java SE 8 アプリケーションであり、Stream API を用いたプログラミングを視覚的に解説します。このゲームの前に、まずはラムダ式の導入に伴う Java 言語の変更点について概要を理解し、基本を身に付けましょう。

ラムダ式の概要

ラムダ式を利用するためには、最近の Java SDK（バージョン 8 以降）を使用し、コンパイル時に言語レベルを Java SE 8 に設定する必要があります。最新の Java SDK バージョンは[こちら](#)からダウンロードできます。

ラムダ式による開発は、この新構文をサポートする IDE を利用することで大幅に簡単になります。ほとんどの Java IDE がラムダ式をサポートするようにアップデートされており、ラムダ式に関するリアルタイムのエラー・レポートやコード補完機能を利用できます。NetBeans IDE と IntelliJ は、Java SE 8 のリリース時点からラムダ式を標準サポートしている点で注目に値します。本記事で紹介するサンプルは、この両方の IDE で問題なく動作します。

新しいラムダ式の機能を実際に確認しましょう。以下の短いコードは、図形のリストを繰り返し処理して、青色の図形を赤色に変更するものです。

```
for (Shape s : shapes) {
    if (s.getColor() == BLUE)
        s.setColor(RED);
}
```

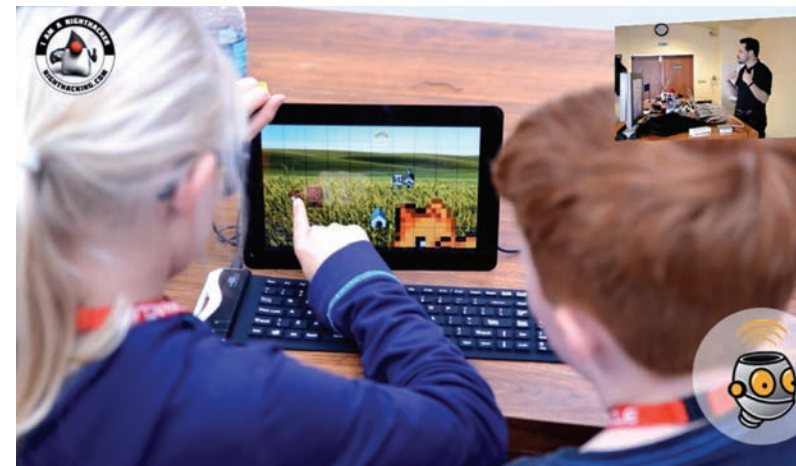
Java SE 8 では、このコードを **forEach** とラムダ式を使用して以下のように書き直すことができます。


```
shapes.forEach(s -> {
    if (s.getColor() == BLUE)
        s.setColor(RED);
});
```

このラムダ式版のコードでは、**Collection** インタフェースの新しい

いメソッドである `forEach` を利用します。`forEach` はラムダ式を引数に取り、この式をコレクションに格納されているすべての要素に対して評価します。このようなラムダ式を簡単に利用できるようにするための API 拡張が、Java コア・クラス全体で行われました。

この API 拡張に関連して、次のような疑問が浮かぶかもしれません。Java 開発チームはどのようにして、後方互換性を損なわずに既存のインタフェースに新しいメソッドを追加できたのでしょうか。たとえば、[Collection](#) インタフェースの実



 Devovx4Kids セッションで本記事のサンプル・アプリケーションを使用しながらコーディングを学ぶ子供たち。

56



58

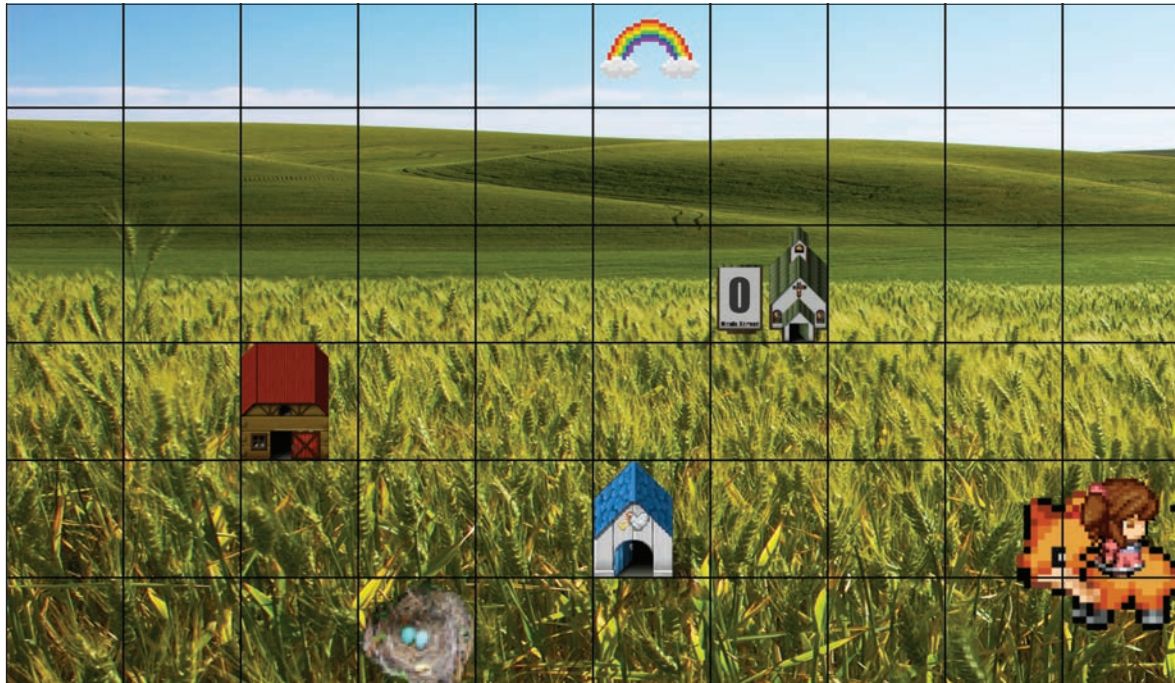


图9

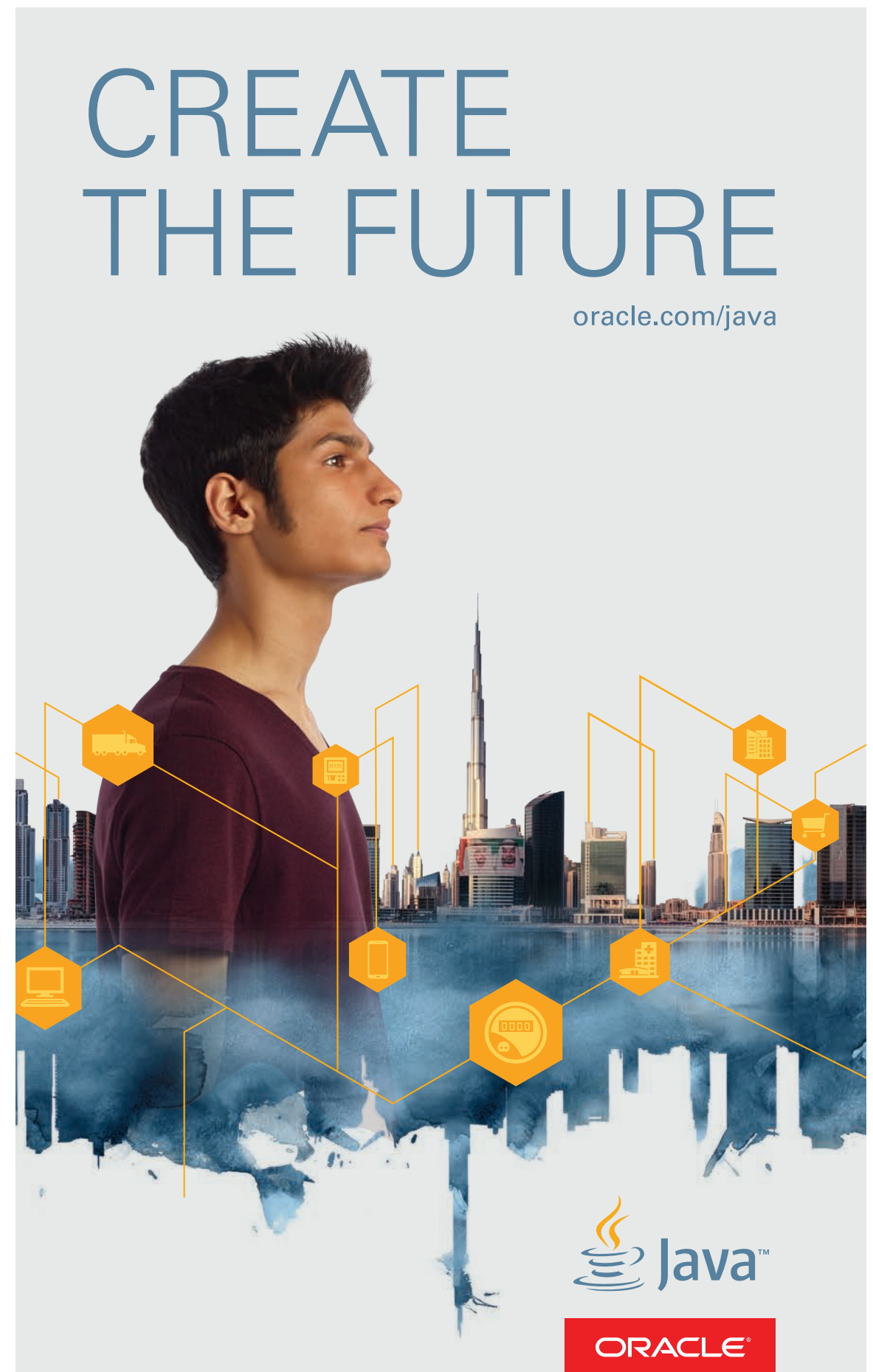
しました。次に、Stream API の詳細について説明し、よく使用される操作の一部として、`iterate`、`filter`、`map`、`flatMap`、`reduce` などを紹介しました。

本記事で確認したとおり、Java SE 8 のラムダ式は Java のプログラミング・モデルを大きく変えるものです。より単純で洗練されたコードを記述できるようになり、Stream のような新しい強力な API の可能性が開かれます。ラムダ式は Java のコア API 全体で幅広く利用されています。たとえば、I/O の新しい関数や新たに導入された日付と時間の API で利用され、また JavaFX などのコールバックを含む API に関数型インタフェースが追加されています。ラムダ式の幅広い利用はすべて、より関数型スタイルに近いプログラミングにつながり、マルチプロセッサ・システムで効率的に動作するコードを作成するために利用できます。皆さんの

コードでもこれらの機能をぜひ利用してみてください。 </article>

LEARN MORE

- Stream API
- 本記事のサンプル・ゲーム
(GitHub)





2014年3月/4月号では、ルーマニアの多言語開発者であるAttila Balazs氏から、並行処理に関する課題が出されました。インクリメントの試行がなぜか「途中で止まる」ことがあるコードが提示され、その修正方法が問われました。正解は3番です。提示されたコード

の問題点は、個々のcnts呼出しの間に競合状態が存在することです。たとえば、スレッド1がカウンタXの値を読み取った後に、スレッド2が同じ値を読み取る場合を考えてみましょう。両方のスレッドがインクリメント値を計算して返すため、インクリメント結果は予想される $a+2$ ではなく、 $a+1$ になります。カウンタの初期化に関しても、同様の競合状態が発生します。

1番でも問題は修正されますが、ConcurrentHashMapを使用する効果がなくなります。2番は同じ問題（完結した一連のイベントとしてではなく個々の処理が原子的に実行される）が発生するため、修正方法にはなりません。4番でも問題を修正できますが、余分な処理が必要になります。

今回は、オラクルのJavaエバンジェリストであるSimon Ritterから、ストリームに関する出題です。

問題

二に、Java SE 8の新しいStream APIを使用して、テキスト・ファイル内でもっとも長い行の長さを返すコードがあります。長さではなく行そのものを返すためには、どのような変更が必要でしょうか。

写真: BOB ADLER
画像: I-HUA CHEN

2 コード

BufferedReaderクラスのlines()メソッドは、Java SE 8で新しく導入されたもので、ファイル内のテキスト行であるStringのストリームを返します。このストリームがmapToInt()メソッドに渡され、String.length()へのメソッド参照を関数として使用してIntStreamが生成されます。終端操作であるmax()は、ストリーム内の最大値を識別して、OptionalIntオブジェクトを返します。get()メソッドはOptionalIntの値を返し、この値がintにオート・アンボクシングされて変数longestに代入されます。

```
BufferedReader reader = new BufferedReader(new FileReader(
    "foo.txt"));

int longest = reader.lines().mapToInt(String::length)
    .max().get();
```

ファイル内でもっとも長い行を返すには、このコードをどう変更すればよいでしょうか。

3 正しい修正方法はどれですか

- 1) `String longestLine = reader.lines().max(String::longest).orElse("");`
- 2) `String longestLine = reader.lines().reduce((a, b) -> a.length() > b.length() ? a : b).orElse("");`
- 3) `String longestLine = reader.lines().reduce("", (a, b) -> a.length() > b.length());`
- 4) `String longestLine = reader.lines().max((a, b) -> b.length() - a.length()).orElse("");`
- 5) `String longestLine = reader.lines().sorted((a, b) -> a.length() - b.length()).findFirst().orElse("");`

わかりましたか？

正解は次号に掲載されます。もしくはチャレンジした内容を電子メールで
お送りください。



ヒント:再帰処理を使用した解決方法について考えてみましょう。