解説 3

スクリプト言語とJava™

ー Javaの観点からスクリプト言語を理解する ー

昨今、Web ブラウザーで動作する JavaScript™ はもちろんのことですが、サーバー側で動作する PHP、Perl、Python、Ruby などのスクリプト言語が非常に人気を博しています。本稿では、スクリプト言語とはどういったものかについて、特に Java 言語・プラットフォームの観点から解説を試みます。まず、スクリプト言語一般について、Java 言語と比較したときの特徴を、さまざまな側面から解説します。その後、Java プラットフォーム独自のスクリプト言語である Groovy も加え、さまざまなスクリプト言語が Java プラットフォーム上に実装が利用できるようになっていますので、それらの紹介と、Java コミュニティーにおけるスクリプト言語サポートに向けた動向を解説します。この際、弊社からのスクリプト言語プラットフォーム製品である IBM WebSphere® sMash にも触れます。

Article 3

Dynamic Scripting Languages and Java

- Understanding Script Languages from the Java Viewpoint -

Dynamic scripting languages are recently very popular, including PHP, Perl, Python, and Ruby, which run on the Web server side, as well as JavaScript running on Web browsers. This article gives an explanation of what dynamic scripting languages are from the Java language/platform standpoint. First, we contrast the language characteristics of dynamic scripting languages with Java from various perspectives. Currently, a lot of language processing implementations are available on the Java platform for various scripting languages, including Groovy, which is a language uniquely available on Java. We introduce these and give an explanation of trends in scripting language support in the community. At this point, we introduce WebSphere sMash, which is an IBM software product for a scripting language platform.

❶ はじめに

昨今、スクリプト言語が人気を博しています。特に Web アプリケーション開発においてその傾向が顕著であり、Web のクライアント側である Web ブラウザーで動作する JavaScript はもちろんのこと、サーバー側で動作する PHP、Perl、Python といったスクリプト言語が利用される場面も多くなってきています。**図1**は TIOBE 社によるプログラミング言語の人気指標推移のグラフ［1］です。Java と C/C++ に次いで PHP の人気が高く、ベスト10には Perl や Python も見受けられることが分かると思います。実際、PHP は、Wikipedia や Facebook など多くの有名な Web サイトで利用されていることがよく知られています。

本稿では、PHP などのスクリプト言語について Java の観点から解説を試みます。まず、スクリプト言語一般について、Java などの言語と比較したときの特徴を、実行形式の側面とアジャイルなプログラミング言語としての側面から解説した後、それらの特徴が際立って分かる簡単なプログラム例を挙げます。最後に、Java プラットフォーム上でのスクリプト言語処理系の実装について、PHP などの Java 実装に加えて、Groovy という Java プラットフォームでのみ動作するスクリプト言語を紹介するとともに、弊社におけるスクリプト言語プラットフォームの代表製品である WebSphere sMash にも触れます。

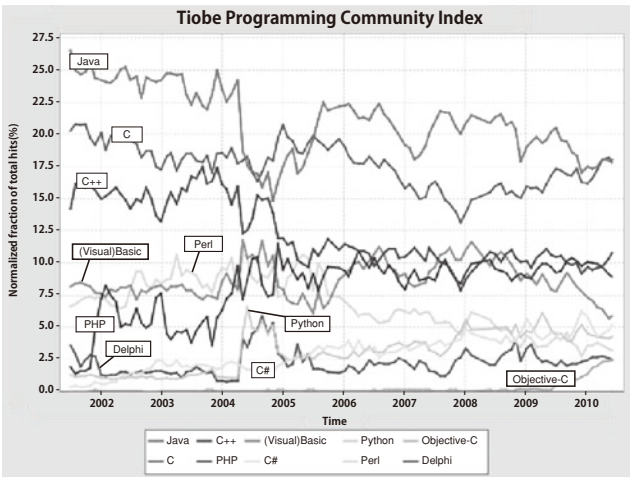

図1. プログラミング言語の人気度 (www.tiobe.com, June 2010)

❷ スクリプト言語の実行形式

本誌の特集がそうであるように、スクリプト言語自体は、アジャイル（俊敏）性が重要視されるときに活躍することが多いようです。また、一口にスクリプト言語といっても、言語により大きな違いがあり、人気を博している理由も、言語仕様単独の特徴というよりは、それらを取り巻くさまざまな環境が大勢を決めているというのが本当のところです。しかしながら、JavaやC/C++と比べてみると、スクリプト言語には典型的な特徴を見いだすことができます。

スクリプト言語には、共通した定義は存在しないのですが、スクリプト言語の定義が議論されるときにその特徴として必ず取り上げられるのが実行方式です。JavaやC/C++のソース・プログラムがコンパイラーによって処理（コンパイル）されて初めて実行可能になる一方、スクリプト言語は、コンパイルしなくても、インタープリターというソフトウェアによってソース・プログラムのまま実行されます。しかし、インタープリター方式の欠点は実行速度で、一般にコンパイル方式に比べて100倍以上遅くなることが知られています。理論的には、JavaやC/C++のプログラムも、専用のインタープリターを用意することによって、ソース・プログラムをそのまま動作させることも可能ではありますが、その実行性能が問題になることは容易に想像できます。スクリプト言語も技術的な事情は同じはずですが、JavaやC/C++とは一般的に利用される場面が異なるため、（少なくとも当初は）それほど問題視されませんでした。Web以外では、ユーザー・インターフェース構築スクリプトのTclや、オペレーティング・システムのユーザーの操作をまとめて記述するシェル・スクリプトなどがその好例といえるでしょう。

スクリプト言語は、さほど実行性能、特に、応答速度が問題にならない状況で、「書いたらすぐ動く」ことが優先される場合に特に受け入れられてきたという側面を強く持っています。Webアプリケーションがその典型ですが、ユーザー・インターフェースとネットワークに関するプログラムでは、速度に対するユーザーの感度が比較的低く、体感速度への寄与はネットワークによる遅延がそのほとんどであるため、それほど応答速度は問題になりません。次節では、インタープリター形式の実行モデルと非常に相性がよいスクリプト言語上の幾つかの典型的な特徴を見ていきます。

❸ 書いたらすぐ動く言語

実行形式がインタープリター形式であるということは、コンパイルの手間なく実行可能であり、「書いたらすぐ動く」というスクリプト言語としての性格を強く持っています。しかし、多くのスクリプト言語が「書いたらすぐ動く」と考えられるのは、それだけが理由ではありません。

スクリプト言語が利用される問題領域上、「書いたらすぐ動く」ことは美徳であり、それぞれのスクリプト言語とそれらを取り巻く環境もそのことを目指して進化して（もしくはそうでないものは淘汰（とうた）されて）きました。ただし、そうした特徴は言語ごとに異なることも多いため、ここでは、特に、Perl、PHP、Python、Rubyのスクリプト言語ごと個別にその特徴を見ていきます。

Perlは、文字列処理を得意とするところから、Web環境におけるCGI（Common Gateway Interface）スクリプトとして、かなり古くから利用されてきた言語です。Perlに特有な文化としてよく知られているスローガンにTMTOWTDI「There's More Than One Way To Do It」（やり方は1つ以上ある）というものがあります。これは、「コードの書き方は一通りだけじゃない。好きなように書けばよい」という意味です。Perlコミュニティーでは、初心者からエキスパートまで、個人個人それぞれに合った一番書きやすいやり方でプログラムを書いてよく、コミュニティーもそれを受け入れるべきである、とされています。汚かろうが何だろうが、とにかく早く書ければよいという考えで言語自体も設計されています。逆にいえば可読性が悪く、メンテナンスには苦労することになりますが、そういう用途には使わなければよい、というのが第一義的なPerlの文化です。一方で、PythonやRubyの文法規則はこの対極に立っており、改行やインデントは見やすくなければならないということを言語レベルで規定しています。

PHPの大きな特徴の1つは、HTML文書に「<?php ～ ?>」というタグを用いて埋め込むかたちでプログラムが書かれる点です。従って、Webページの一部を動的に生成するようなプログラムを書くのに非常に適した言語です。さらには、最も広く普及しているApacheのWebサーバーのプラグインとして言語処理系が配布されており、実際多くのホスティング・サービスで提供されているWebサーバーでは、PHPスクリプトを置くだけですぐに動的生成されるページを準備できます。RubyやPythonでは、数行で自前のWebサーバーを動かせるようになっておりますが、コマンド・ラインからプログラムを起動させる必要があるため、ホスティング環境にはあまりなじみません。最初に述べた言語自体の特徴は、Javaの世界でもJSPで踏襲されていますが、

動作させられるWebサーバーの入手の容易さでPHPの右に出るものはありません。

Pythonには「電池同梱（battery included）」といわれる哲学があって、プログラマーがすぐに使えるようなライブラリーや統合環境があらかじめディストリビューションに含まれています。このため標準ライブラリーは非常に充実しており、正規表現はもちろん、XML処理、Web通信、電子メール、データベース接続などのモジュールが一通りそろっています。これは、近年のPerlにおけるCPAN（Comprehensive Perl Archive Network）モジュールや、PHPの準標準エクステンション・ライブラリーであるPECL（ネイティブ実装）やクラス・ライブラリーであるPEAR（PHP実装）、そしてRubyにおいてgemと呼ばれるパッケージ管理化にあるライブラリーが、言語処理系とは別にインストールしなければならないのに比べて数段使い始めやすくなっています。

Rubyという言語を有名にしたのはRuby on Railsと呼ばれる一種のWebアプリケーション・フレームワークによるところが非常に大きいといわれています。Ruby on Railsでは、モデル・ビュー・コントローラーのアーキテクチャーはもちろんのこと、「Convention over Configuration」（設定させるより規約に従わせよ）と、DRY「Don't Repeat Yourself」（同じ事を繰り返すな）という基本理念のもと、必要最低限の記述をすればあとは注意深く設計された決まりに基づいてデータベース処理周りなどを自動生成してくれるようになっています。

本節をここまでご覧になってきてお気付きかもしれませんが、それぞれの言語の特徴は、PerlのTMTOWTDIを除けば、言語それ自体の特徴というより、それらを取り巻く環境の特徴であり、今後それぞれのスクリプト言語がそれぞれのよいところを取り入れて進化していくことは容易に想像できます。実際、RubyをHTML文書に埋め込めるようにするeRubyと、そのApache Webサーバー用プラグインが利用できるようになってきており、またPerlにおけるCatalystやPythonにおけるDjangoなど、スクリプト言語向けの多くのWebアプリケーション・フレームワークがRuby on Railsに追随しつつあるなど、その兆候が実際に見られ始めています。

❹ 動的なデータ・バインディング言語

簡単なこと（JavaやC/C++では面倒なこと）を簡潔に書けるというのもスクリプト言語に多く見られる特徴です。本節では、幾つかの便利な特徴が凝縮された利用例として、1つのPHPのプログラム例を挙げ、その特徴を説明したいと思います。**図2**は、ブログ・サービスで提供されているREST APIを利用するPHPプログラムの例です。**図2**の上部が実際に動作するPHPのソース・プログラムで、下部はRESTサービスにより提供されるXMLデータの例です。このプログラムでは、1行目で「http://tatsubori～」のURLで示されるRESTサービスにアクセスし、返ってきた結果を変数「$feed」に代入しています。2行目では、結果のXML文書内の「author」要素中の「name」要素内のテキストを抜き出し、「echo」命令によってHTMLの一部としてそれを出力しています。Javaで同じ処理を書こうとすると非常な手間が掛かるのですが、PHPの場合はたった2行のコードで実現できてしまいます。

関係データベースやXMLなどのデータをプログラムでどう扱うかという、データ・バインディング（データ結合）の問題はさまざまなエンタープライズ・アプリケーションやデータ中心のWebアプリケーションでは必ず直面する問題です。Javaにおいてもその取り組みは盛んになされてきており、Hibernate（オープンソースのO/Rマッピング・フレームワーク）やEJB 3.0で関係データベースとオブジェクトの対応付けのサポートがありますし、JAXB（The Java Architecture for XML Binding）ではXMLとオブジェクトの対応付けのサポートが利用できます。しかしながら、どちらの場合も事前に生成ツールを使うか自分で書くなどして、表の行やXMLの要素に対応するクラスを準備しておく必要があります。しかしRubyのRuby on Railsアクティブ・レコードやPHPのSimpleXMLエクステンションではその必要はありません。

このような機能が利用できるのは、スクリプト言語に典型的な、動的型付け（dynamic typing）と柔軟なリフレクション（reflection）機能の2つの要素をフルに活用しているライブラリーのおかげです。ちなみに、この2つともやはり、Java/C++ではあまり許容されないような実行時オーバーヘッドを伴う機能であり、インタープリターによる実行形式であるスクリプト言語と親和性が高いという技術的な背景があります。

動的型付けの「動的」というのは字面上明示的にintやStringなどの変数の型が宣言される静的型付け（static typing）言語の「静的」の反対であり、つまりは、変数の型が明示的に宣言されないようなプログラム言語の特徴です。いちいち型宣言を書かなくてよい反面、ある変数や

```
<?php
$feed = simplexml_load_file("http://tatsubori.blogspot.com/feeds/posts/default");
echo $feed->author->name;
?>
```

```
<feed xmlns='http://www.w3.org/2005/Atom' xmlns:openSearch='http://a9.com/-/spec/opensearchrss/1.0/'><id>tag:blogger.com,1999:blog-18434712</id><updated>2008-03-30T13:11:54.296+09:00</updated><title type='text'>Michiaki Tatsubori's Technical News Watch</title><link rel='alternate' type='text/html' href='http://tatsubori.blogspot.com/'/><link rel='next' type='application/atom+xml' href='http://tatsubori.blogspot.com/feeds/posts/default?start-index=26&amp;max-results=25'/><link rel='http:schemas.google.com/g/2005#feed' type='application/atom+xml' href='http://tatsubori.blogspot.com/feeds/posts/default'/><link rel='self' type='application/atom+xml' href='http://tatsubori.blogspot.com/feeds/posts/default'/><author><name>Mich</name></author....
```

図2. PHPによるRESTサービス利用の様子　（上:PHPスクリプト、下:RESTサービスから供給されるXML文書）

関数の返り値が、一見どのような値の種類を取るか分からなくなるため、プログラマーからは賛否両論あります。しかし、上で述べたように、事前に準備したクラスがないにもかかわらず、オブジェクトへのアクセスを記述するなどということは、動的型付けでなくてはこのように簡素な記述にはできません。

リフレクション機能と一口にいっても多岐にわたるのですが、ここでは特にインターセッション（intercession）と呼ばれる、オブジェクトへのプロパティー・アクセスやメソッド呼び出しがなされたときの振る舞いをプログラムにより変更できるようにする機構を指しています。**図 2** で、simplexml_load_file() の結果として $feed に代入されているのは実は SimpleXMLElement という既定のクラスのインスタンスであり、従って、たまたま取り扱っている XML 文書に固有の要素名である author という名前のプロパティーを備えているわけではありません。代わりに、存在しないプロパティーへのアクセスをフックする特別なメソッドが定義されていて、そこでアクセスされたプロパティー名に応じた XML の要素や属性を検索して、新しい別の SimpleXMLElement オブジェクトとして必要に応じて返すということをしています。

```
class PseudoSimpleXMLElement {
  private $dom;

  function_construct($dom) {
    $this->dom = $dom;
  }

  function_get($name) {
    $child = $this->dom
      ->getElementsByTagName($name)->item(0);
    return new PseudoSimpleXMLElement($child);
  }

  function_toString() {
    return $this->dom->nodeValue;
  }
}
```

図3. リフレクションを用いたXMLデータ・バインディングの実装例

SimpleXML の場合、実行効率のため、実際は C のコードで書かれていますが、PHP で同じことを書くことが簡単にできます。Java で似たようなことをするためにはアスペクト指向技術などを駆使する必要があり、実際のところ安易に適用できる解決策はありません。

図 3 は PHP でリフレクションを用いて SimpleXML と同様の振る舞いをするオブジェクトを実装しているクラスの例です。「_get()」が PHP においてプロパティーへのアクセスをフックするために定義されるメソッドです。アクセスされたプロパティー名（author など）が渡されてくるので、それを使って対応する実際の XML 文書の要素（変数 $dom に保持されています）の子要素を getElementsByTagName() という既定のメソッドで呼び出しています。得られた子要素（$child に代入されています）をもとに、新たにオブジェクトを生成して結果として返しています。

❺ Java プラットフォーム上のスクリプト処理系

スクリプト言語の人気は、Java コミュニティーでも .net コミュニティーでも看過できないものとなっており、両陣営とも、これらのスクリプト言語を積極的にサポートするプラットフォームになりつつあります。具体的には、Java であれば JVM™（Java Virtual Machine）上で、.net であれば CLI（Common Language Infrastructure）上の、それぞれのバイトコードにスクリプト言語のソース・プログラムをコンパイルして実行することを推奨し、プラットフォームとしては、そのためのサポートを行います。Microsoft® .net プラットフォームにおけるスクリプト言語サポートは DLR（Dynamic Language Runtime）として知られています。本節では、Java におけるスクリプト言語処理系の紹介と、Java コミュニティーで積極的に進められているスクリプト言語サポートの状況について述べます。

元来、Perl、PHP、Python、Ruby のどのスクリプト言

語も、オープンソースのコミュニティーで開発されており、そのコードはC/C++ で書かれていますが、一方で、それらをJava 上に実装しようという取り組みが盛んです。すでにある処理系とは別に、新しくJava 上で動作する処理系が開発される理由には、知的所有権などのビジネス上の理由も多少あるようですが、技術的には、既存の Java 環境との高い親和性への期待があります。親和性というのは、単一アプリケーション・サーバー内で複数言語によるアプリケーションを管理できたり、異なる言語によるアプリケーション間の効率的なやりとりが可能になったり、といったことを意味しています。実際、JRuby（Ruby の Java 実装）、Jython（Python）、Rhino（JavaScript）や、IBM の WebSphere sMash 上で動作する P8（PHP）など、人気のスクリプト言語の Java 上での実用的な実装が存在し、広く使われています。

既存のスクリプト言語ではなく、Java プラットフォーム上で動作するように新たに作られた Groovy、Scala、Clojure なども存在します。Scala は関数型（+オブジェクト指向）言語であり、Clojure は Lisp の方言ですが、Java 言語ベースのスクリプト言語である Groovy は特に、Java コミュニティーを中心に人気となっています。多くの書籍が発行され、また、GR と呼ばれる開発者会議が毎年盛大に開かれています。言語仕様も JSR 241 として標準化が進んでいるようです。Java プログラマーが容易に習得できるように、Java の文法をほとんどそのままに、型宣言の省略やクロージャーなど、「スクリプト言語」らしい拡張が随所に施されています。さらに、Grails と呼ばれる Ruby on Rails 風のフレームワークや GSP と呼ばれる PHP（というより JSP ですが）風の HTML への埋め込み機能など、さまざまなスクリプト言語からよいところが取り込まれています。

IBM も含め、Java コミュニティーにおいては、Java 上でスクリプト言語をうまく動作させたいという要求が強まっており、すでに Java 6 で取り入れられた Java Scripting API（JSR 223）に続き、Java 7 での導入を狙ったスクリプト言語向けの Java VM 拡張（JSR 292）も策定作業中です。JSR 223 では、Java プログラム中から、スクリプト言語のコードを呼び出したり、その逆にスクリプト中から Java のオブジェクトにアクセスさせたりするためのフレームワークが定義されました。JSR 292 は、Java 上に、動的な型を用いたメソッド呼び出しなど、Java 言語とは異なる言語要素を効率よく実装するための VM 拡張が規定されますが、執筆時点では初期草稿の公開査読（2008 年 5 月）の後、プロトタイプ実装や多くの議論が行われ、Java 7 への導入に向けて着々と進んでいます。

Java 上に実装されたスクリプト言語として、PHP と Groovy を擁する IBM WebSphere sMash の事例において、世界的には SugarCRM という PHP で書かれた CRM アプリケーションがそのまま Java の上で動き、ほかの Java アプリケーションと連携させるシナリオがよく用いられ、デモも公開されています。国内においては、コクヨオフィスシステム株式会社様における「Office DARTS（オフィスダーツ）」、東京工科大学様における「授業クラウド」の事例（本誌 24 ページ以下：インタビュー②参照）などが公開されています。ほかでは、REST アーキテクチャーへの親和性のよさから、Web 2.0 的なアプリケーション開発において活躍することが多いようです。

❻ 終わりに

本稿では、Java 利用者に向けて、スクリプト言語の解説を試みました。スクリプト言語は、アジャイル性に特化した言語です。即興のプロトタイプや利用者自身によるカスタマイズなどには非常に効果を発揮します。しかし、スクリプト言語に限ったことではありませんが、現場で初めて利用するには、言語自体というよりは、それらを取り巻くライブラリーなどの資産や、技術コミュニティーなどをよくよく考慮することが必要です。本稿は、その指針自体を与えるものではありませんが、そのための糸口となれば幸いです。

［参考文献］

［1］ TIOBE Software, TIOBE Programming Community Index for June 2010, http://www.tiobe.com/.

日本アイ・ビー・エム株式会社
東京基礎研究所
スタッフ・リサーチャー

立堀 道昭 **Michiaki Tatsubori**

［プロフィール］
プログラミング言語とソフトウェア工学、Webを中心に研究業務に携わっている。博士（工学）を取得後、2002年、日本IBMに入社。XMLとWeb Servicesのプロジェクトに参加する中、IBM developerWorks® JapanのJavaゾーン・リーダーも務めた。近年はスクリプト言語処理の研究をリードしている。著書に、「ロボコード・バイブル」（技術評論社）、「AspectJによるアスペクト指向プログラミング入門」（ソフトバンク クリエイティブ）など。