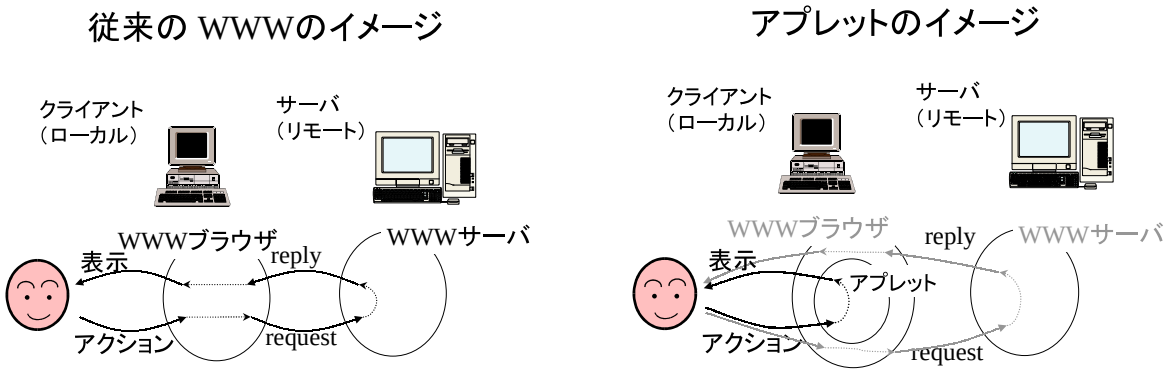
第1章　Java

1.1　Java とは

1995 年、Sun Microsystems 社から公表された、比較的新しい言語である。文法は、C あるいは C++ と似ているが、＿＿＿＿＿＿＿＿。C++と同様、＿＿＿＿＿＿＿＿＿＿言語であるが、C++に比べて＿＿＿＿＿＿仕様になっている。なお、JavaScript (ECMAScript) とは文法は似ている（JavaScript が Java に文法を似せている）が、それ以外の関係はなく全く別の言語である。

1.2　Java の特徴

Java は、当初 Web ページに動きと＿＿＿＿＿＿＿＿＿＿＿＿をもたらすものとして、世に広まった



従来の HTML だけを用いて書かれた Web 文書の場合は、ユーザがマウスをクリックするなどのアクションをおこしても、ブラウザはそのアクションを遠隔地の WWW サーバに伝え、その応答を待って新しい表示をする必要があった。

Java を使っている場合は、一度＿＿＿＿＿＿（Applet）と呼ばれる Java のプログラムをサーバからブラウザへダウンロードする。するとユーザのアクションに対して、ブラウザの中で実行されているアプレットが直接反応することができる。こうして、インタラクティブ性の強いページを記述することが可能になった。

Java の情報のページ

`http://java.sun.com/`	（Java の故郷）
`http://www.sun.co.jp/java/`	（日本語の Java 関連情報）
`http://javanews.jp/`	（日本語 Java News）

その他の Java 情報

……
……
……

このように、アプレットと呼ばれる Java のプログラムはネットワークを通じて別のコンピュータに移動して実行されることになる。このような使い方をするためには _____と _____が重要になる。

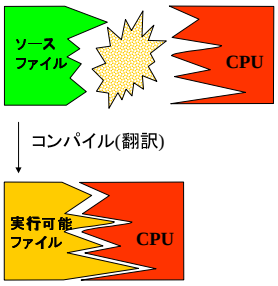
安全性　これは、簡単にいえばアプレットを使って他人のコンピュータに悪戯をすることができない、ということである。もし、ホームページに任意のプログラムを埋め込んでブラウザ上で実行させることができれば、ハードディスク中のデータを完全に消去してしまうなどのイタズラが簡単に行なえてしまう

安全性を保障するためには、まずプログラムにファイル操作などをさせない、などの制限を課する必要があるが、C のような言語では、ポインタ（アドレス）操作や無制限な型変換など、いくらでも抜け道がある。Java はこのような抜け道を作らないよう設計されている。

機種非依存性　Web ページに埋め込まれるということは、さまざまな機種のコンピュータで実行される可能性があるということである。つまり、Java のプログラムに機種依存性があってはいけない。_____を用いる実行方式ではプログラムが機械語に翻訳されるため、機種依存性は避けられない。一方、_____を用いる方式では、各機種毎にインタプリタを実装するだけで良いが、効率が犠牲になる。このため、Java では _____という方法をとる。

Java のプログラムは Java コンパイラによって _____という仮想 CPU のコードに翻訳される。この仮想コードを各 CPU 上の JVM エミュレータ（一種のインタプリタ）が解釈・実行する。この方法は Java プログラムを直接インタプリタで解釈・実行するよりは高速である。しかし、現在では JVM コードをより高速に実行するために _____というものを用いて、JVM コードを実行しながら各 CPU の機械語へ翻訳する、という方法を用いる。

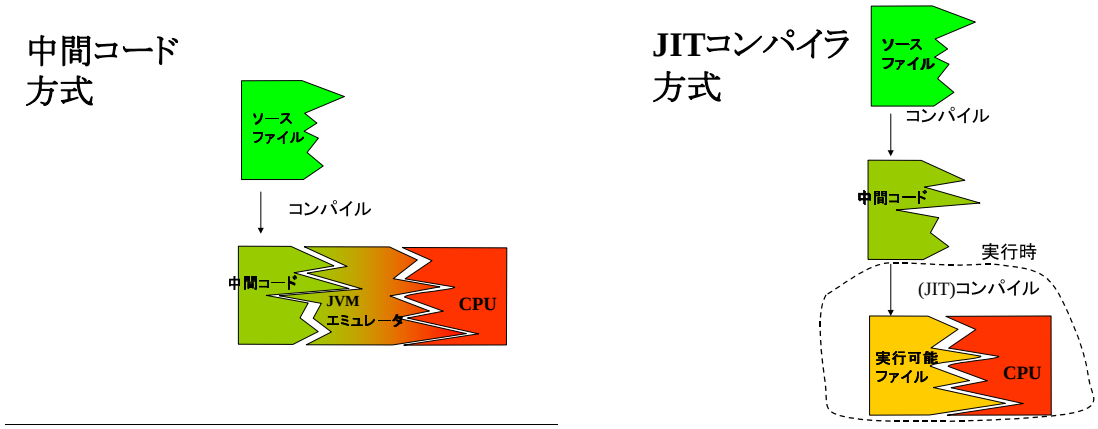
コンパイラ



_____

インタプリタ

_____



機種非依存のグラフィックライブラリ、ネットワークに関する標準ライブラリを持つことも、今までのプログラミング言語にはあまりなかった重要な特徴である。

このように当初、Java はアプレットを作成するための言語として広まったが、このような性質は、他の分野のアプリケーションでも役に立つため、現在は通常のアプレット以外のアプリケーション（例えば WWW **サーバ側**で動作するサーブレット（Servlet）などのプログラム）を作成するためにも用いられている[1]。

1.2.1 オブジェクト指向プログラミング

Java はオブジェクト指向プログラミング言語（Object-oriented Programming Language, OOP）である。オブジェクト指向（object-oriented）とは簡単に言えば、従来の手続き（命令）を中心としたプログラム部品（＿＿＿＿＿＿＿＿、＿＿＿）の利用に加えて、データを中心とした部品（＿＿＿＿＿＿＿）の利用を支援することである。

関数（サブルーチン）はいくつかの手続きをまとめて一つの部品としたものだが、オブジェクトは、いくつかのデータ（関数 — ＿＿＿＿＿＿ （method）と呼ばれる—も含む）をまとめて一つの部品としたものである。

- **関数・サブルーチン**

  代入文

  繰り返し文

  条件判断文

  ... などの手続きをひとまとめにしたもの

- **オブジェクト（クラス）**

  整数

  実数

  文字列

  関数・サブルーチン（メソッド）

  ... などのデータをひとまとめにしたもの

実際には、プログラム部品として提供されるのは、オブジェクトそのものではなく、オブジェクトの雛型とでもいうべき ＿＿＿＿＿ （class）である。クラスは、具体的なデータ（つまり、1 とか 3.14）

[1]最近は、また携帯電話用のアプリケーション（i-アプリ）を作成するための言語として注目を浴びている。

ではなく、データの型（つまり、`int`（整数）型とか `float`（実数）型） — とメソッドの定義 — のみを指定したものである。クラスを具体化（instantiate — つまり、`int` 型の要素は 1 で、`float` 型の要素は、3.14 などと定めること）したものがオブジェクトである。この時、このオブジェクトはもとのクラスの _____（instance, 具体例）である、という。オブジェクトを構成している個々のデータを _____（instance variable）、あるいは _____（member）、_____（field）という。ただし、関数の場合は _____（method）というのが普通である。あるオブジェクトのメソッドを起動することをオブジェクトに _____（message）を送る、という言い方をすることがある。

関数・サブルーチンを利用する場合、外部から見た振舞いが同じである限り、内部でどのように実現されていても構わない。例えば、配列の要素を大きさの順に並び替える（ソーティング）方法はいくつもある。これと同じように、クラスを利用する場合でも、2 つのクラスの内部の実現方法が少々異なっていても、外部から見た振舞いが同じであれば、それらを入れ換えることができる。このように、クラスの内部の実現方法を外部から隠すことを _____（encapsulation）という。

従来の非オブジェクト指向言語では、部品の再利用方法は、既存の部品を関数・サブルーチンとして呼び出すだけだったが、オブジェクト指向言語では、それに加えて既存の部品（つまりクラス）を少しだけ書き換える（_____, inherit）、という形の再利用の方法も可能になる。従来の言語ではプログラムの “幹” の部分を変えて “枝” の部分だけを再利用することができたが、オブジェクト指向言語では、“枝” の部分を変えて “幹” の部分を再利用することもできる、という言い方もできる。

オブジェクト指向のさまざまな概念は、必要になった時点で少しずつ説明することにする。しかし、自分で部品を設計するならともかく、利用するだけなら、とりあえず上記のことを知っていれば充分である。

キーワード:

Java, C, C++, オブジェクト指向、アプレット、中間言語方式、JIT コンパイラ、サーブレット、オブジェクト、メソッド、クラス、インスタンス、インスタンス変数、カプセル化、継承