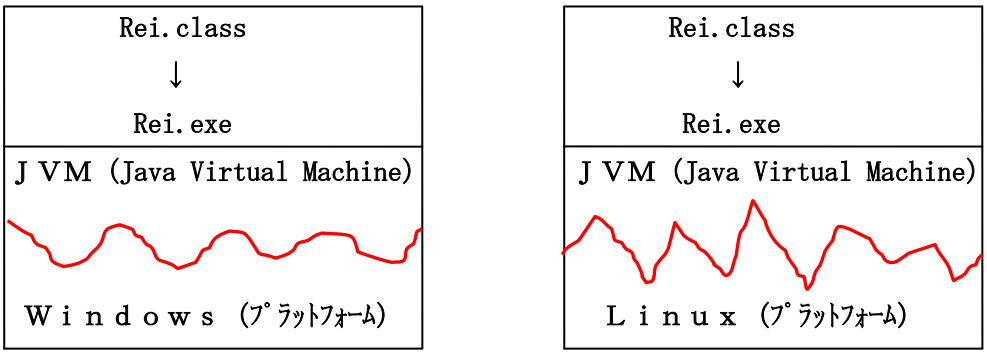
第１５話（Java 言語の特徴）

キツネ！Java のプログラムは、Windows でも Linux でも Mac でも動作すると聞いたことがあるのだけれど、どういうこと？

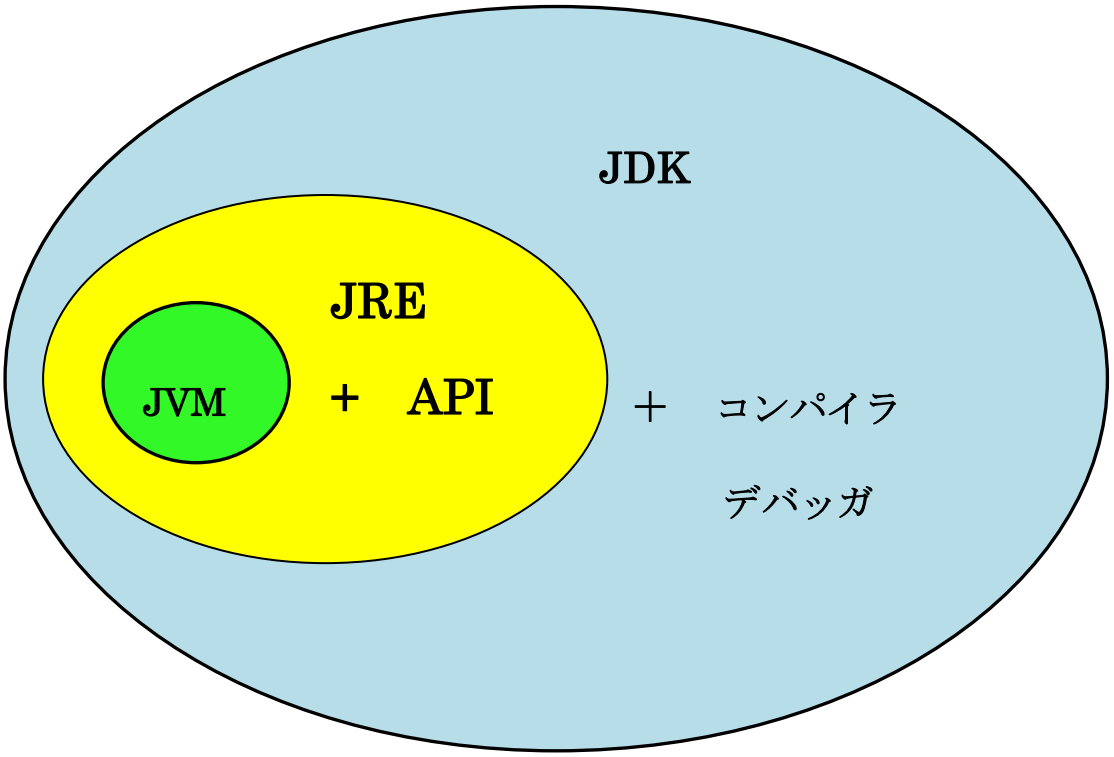
Windows とか Linux、Mac というのは OS を指しているのだ。プラットフォームとも言うよ。
Java 言語は、確かにプラットフォームが違っても実行することができるのだ。これは Java 言語の第一の特徴だね。
プラットフォームが異なれば、プラットフォーム毎にプログラムを作りなおさなければならないのが一般的だ。でも、Java 言語で作ったプログラムは、作り直さなくても良いんだ。
以下に、その理由を図示してみるよ。



上図は、Windows や Linux などのプラットフォームの違い（デコボコ）を JVM（Java 仮想マシンとでも訳すのかな）がなだらかにし、同じ Java プログラムをどのプラットフォームでも実行できるようにしているよ、という図だよ。Rei.class は Rei.java（ソースファイル）をコンパイルし、中間言語になったものだ。それを JVM が Rei.exe にし、実行している。

キツネ、あらすじはわかったけれども、どうやって各プラットフォームで JVM を使えるようにするのだ。その仕組みがわからん。

タヌキ、前に仮想マシンはソフトウェアだと言ったよな。Java 仮想マシンも当然ソフトウェアだ。
まず、JVM（Java Virtual Machine）と JRE（Java Runtime Environment)と JDK (Java Development Kit)の違いを理解する必要がある。
Jre は java の実行環境を作ってくれる。つまり、JVM＋API でできているソフトウェアだ。API は、クラスライブラリでインポートして使えるのだ。パッケージにもなっているよ。
JDK は、Jre＋コンパイラ（Javac）＋デバッガでできているソフトウェアだ。Java 開発用のソフトウェアとも言われるよ。
ということは、JDK をインストールすれば、JVM を含めて全て必要な物が揃うというものだ。ただ、インストールする JDK はプラットフォーム毎に違うからね。



なるほどな。JDK をインストールすれば良いのか。

Java の特徴、その２。
３種類のプログラムが作れるのさ。一つはブラウザ上で実行されるアプレット。二つ目は、Tomcat によって実行されるサーブレット（JSP など）、三つ目は、普通のアプリケーションだ。すべて、ベースが JVM であることは共通だけれども。
特に、Tomcat は JVM 上で動作し、サーブレットの処理を解釈・実行し、Web サーバとデータベースの橋渡しもする Web コンテナである。この３種類の仕組みを図示すると以下のようになる。

ｺﾝﾊﾟｲﾙされたｱﾌﾟﾘｹｰｼｮﾝ Rei1.class	ｺﾝﾊﾟｲﾙされた アプレット Rei2.class	ｺﾝﾊﾟｲﾙされた サーブレット Rei3.class
Rei1.exe JAVA	Rei2.exe ブラウザ	Rei3.exe Ｔｏｍｃａｔ
JVM	JVM	JVM

Windows（プラットフォーム）

キツネ！この３種類のプログラムは、どのように区別するのだ。

タヌキ、良いところに気がついたな。
アプレットは、インポート（import）するクラスライブラリを含む［java.applet］パッケージ、サーブレットはクラスライブラリを含む［javax.servlet］パッケージでそれらの機能を持たせるのさ。

先ずは、Java アプリケーションの簡単なプログラム例とコンパイル＆実行のさせ方を取り上げてみよう。
タヌキ、JDK のインストールの仕方は、ネットで調べて準備を整えてくれ。JDK はプラットフォームにあわせてインストールしなければならないし、パスを通さないと使えない。また、Eclipse のような統合開発環境を使うユーザもいるかもしれん。Java は人それぞれの使い方があるから、まかせる。
オイラは、エディッタにメモ帳や gedit を使い、実行は端末で行うという、もっとも基本的な解説をするからな。これが一番勉強になる。
ところで、第８話から１１話の仮想環境で取り上げた CentOS7 の仮想マシンができていれば JDK1.8 が自動的にインストールされ、使えるようになっている。その状態ならば、以下の解説通りに操作・実行することができるぞ。また、ホストマシンの Windows に何も影響を与えないから安心して実習ができる。
まあ、CentOS7 の仮想マシン上に Java の仮想マシン（JVM）ができているなんて、訳わかんないけどね。

仮想マシン（CentOS7）の
gedit で作成した Java アプリケーション（Rei1. java）のソースファイル

開く(O)　Rei1.java　~/　保存(S)

```
public class Rei1{
	public static void main(String[] args){
		System.out.println("私（Fox）が作ったはじめてのJAVAアプリ");
	}
}
```

なるほど、仮想マシン（CentOS7）を起動し、その中にインストールされている gedit を呼び出し、起動して上図の通りに入力し、Rei1.java というファイル名で保存すれば良いわけだな。次はどうするのだ？

端末を起動し、コンパイルと実行

```
                                    fox@centfox          ~
ファイル(F)  編集(E)  表示(V)  検索(S)  端末(T)  ヘルプ(H)
[ fox@centfox          ~]$ javac Rei1.java
[ fox@centfox          ~]$ java Rei1
私（Fox）が作ったはじめてのJAVAアプリ
[ fox@centfox          ~]$
```

タヌキ、急がない！
プログラミングは、入力と出力が肝心なのだ。ゆっくりやって行こう。
まず、Rei1.java ファイルの保存先は、オイラ（/home/fox）のホームディレクトリにしないと「そんなファイルは存在しない」というエラーメッセージがでるからな。それを確認し、１行目を実行するとオイラのディレクトリに Rei1.class というコンパイルされた中間ファイルができる。次に２行目のように［java Rei1］と入力し、リターンキーを押して実行するのだ。この時［java Rei1.class］と中間ファイルを指定したらダメだよ。つまり、今までの説明を理解していないことになる。Java は、中間ファイル Rei1.class を実行ファイル［Rei1.exe］に変換し、実行しているのだ。Rei1.class は実行できないのでエラーになるぞ。

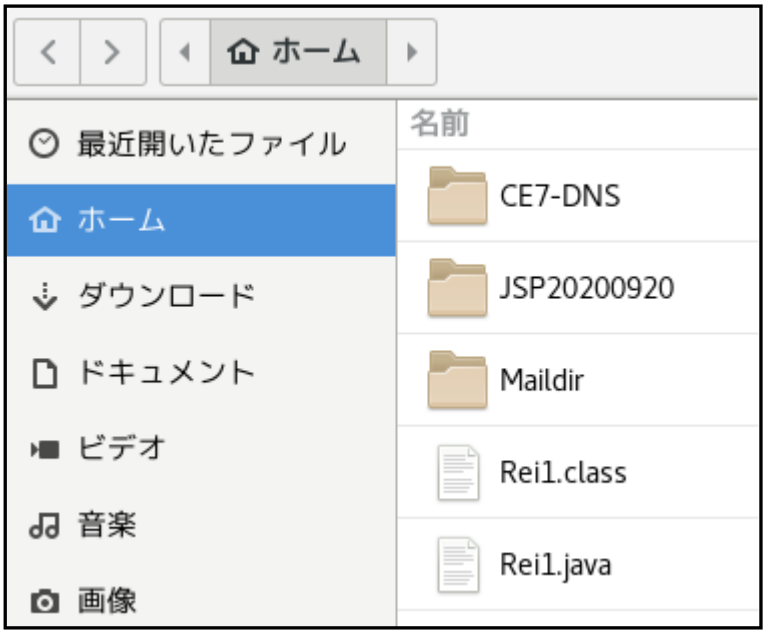
キツネ、それで、

Rei1.exe JAVA
JVM

この図の意味がわかった。

キツネ、おまえのホームディレクトリを覗いても良いか。

/home/fox の内容



いいともさ！余計なディレクトリも入っているけどな。
Rei1.java とコンパイルした Rei1.class ができているのがわかるだろ。さらに、実行ファイルの Rei1.exe が存在していないことに注意してな。この点が、多くのプラットフォームで動作する理由なのだから、重要だ。

次は、Java アプレットの実行例だ！

gedit で Java アプレットのソース（Rei2.java）を入力・保存

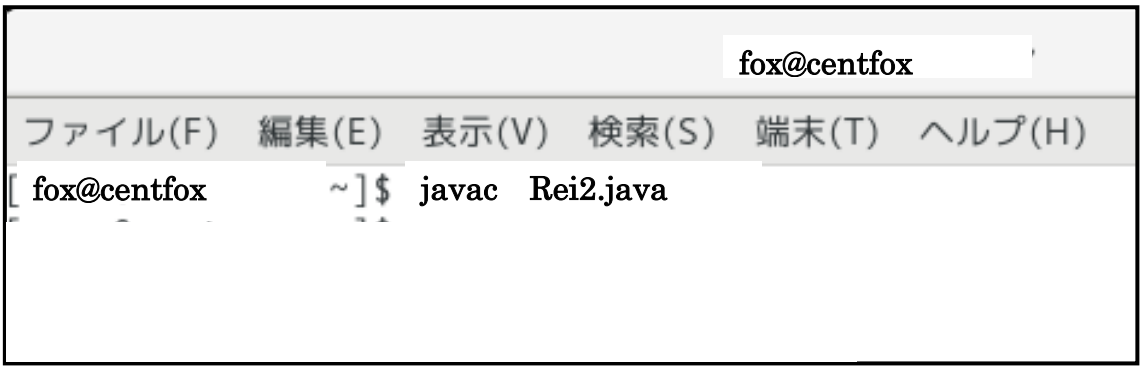
```
import java.applet.Applet;
import java.awt.Graphics;

        public class rei3 extends Applet {
                public void paint(Graphics g){
                        int x,y;
                        for (x=0;x<=5;x++){
                                g.drawString("fox",x*10, x*10);
                        }
                        for (y=1;y<=4;y++){
                                g.drawString("fox",50-y*10,50+y*10);
                        }
                }
        }
```

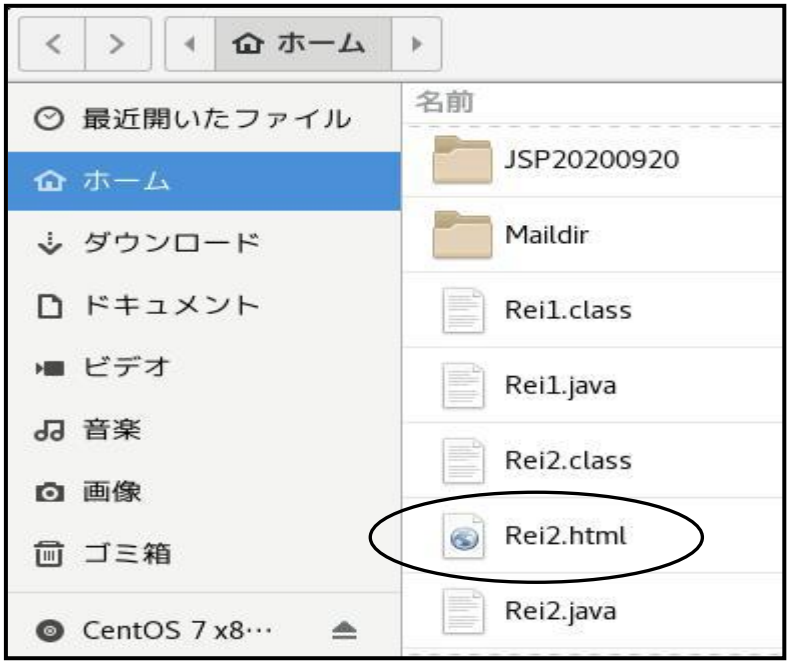
アプレット実行のベースとなる HTML ファイル（Rei2.html）を入力・保存

```
<html>
<head>
<title>Rei2</title>
</head>
<body>
<hr>
<applet   code = "Rei2.class"     width=420     height=420></applet>
<hr>
</body>
</html>
```

端末を起動し、アプレット（Rei2.java）をコンパイル

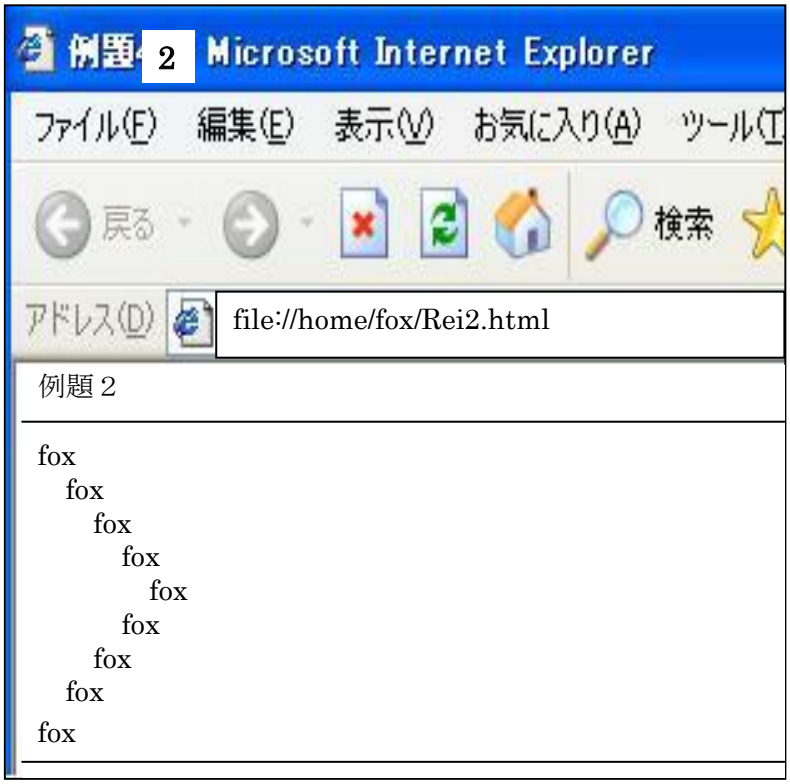


/home/fox の内容



Rei2.java と Rei2.html の保存先は、オイラのディレクトリ（/home/fox）だ。
Web サーバが構築できていれば、(/ver/www/html) に置き、ブラウザを起動し、URL に「http://www.fox.com/Rei1.html」と入力して起動するのだけれど、今回は簡単に上図の［Rei2.html］ファイルをマウスでダブルクリックすれば、ブラウザが起動され、そのブラウザで Rei2.class が Rei2.exe に変換・実行され下図のよう

アプレットの実行結果



キツネ！ブラウザは起動され、「例題２」と上下のラインは表示されたが、９個の「fox」は表示されないぞ。どうやら HTML は実行されているが、アプレットの Rei2.class は実行されていないようだ。さらに、ブラウザによっては、「このプラグインは危険だから使わないように」というメッセージが出るぞ。また、オイラをダマそうとしているな！

ごめん、ごめん！ダマスつもりは無かったんだけどね。２０１７年以前は、Java アプレットは、ブラウザ上で使えていたのだが、２０１８年から Java アプレットにはセキュリティ上の問題があって、全てのブラウザでサポートしなくなったのだ。
さらに、アプレットは使わない方向に向かっているのだ。
オイラも残念だけど使用を止めるよ。
今までの解説は、Java アプレットとは、このような物だったのだ、という参考にしてくれ。

了解！サポートされなくなることもあるのだ。
最後は、Java のサーブレットの話かな！

否、Java サーブレット（特に JSP）の話はしないよ。
重要なんだが、色々と準備しなければならないのだ。
CentOS7 の仮想マシン上で作れるのだが、そこに Web サーバ、DNS サーバ、Tomcat などをインストールして準備しないとサーブレットの実習はできないのだ。
それらの準備が整ったら、再度話をすることにするよ。
ただ、時の流れとともに、JSP のコードもかなり変化してきていることを言っておこう。

次は 第１６話だ。