プログラミング環境Nigari - 初学者がJavaを習うまでの案内役

長　　慎　也†

我々は、プログラミング学習の導入部において用いるのに適した言語環境に関する研究を行ってきた。早稲田大学コンピュータ・ネットワーク工学科の 1 年生を対象に、2003 年に実践したプログラミング授業の事例を紹介する。

この授業においては、独自に開発した言語 Nigari とその環境 Nigari System を利用した。

Nigari の言語仕様は、Java のそれを簡素化したものになっており、クラスやメソッドの宣言など、初学者にとって理解が難しいものを書く必要がない。一方、基本的な制御構造などは、Java とほとんど同じ仕様である。

Nigari の実行環境は、オブジェクトを自動的に可視化する機能をもつ。これによって、学習者のプログラミングへの意欲を向上させるだけでなく、オブジェクトの概念をも理解させることができる。

この授業は本来 Java を用いて実習を行うものであったが、導入部に Nigari を用いた。実験では、オブジェクトの可視化機能について学生から高い評価を得られた。また、言語を簡素にすることについても、ある程度の評価を得られた。

Nigari - a programming language and environment for the first stage, leading to Java world

Shinya CHO†

1. はじめに

本文では、プログラミングの初心者にも簡単に扱えるプログラム言語 Nigari とそのプログラミング環境である Nigari System[1)2)] を紹介する。

Nigari と Nigari System は、次のような特徴をもっている。

- **簡素な言語仕様**
  Nigari は、オブジェクト指向言語であるが、クラスの宣言やメソッドの宣言などの「おまじない」を書かなくても、文だけを書けばプログラムが動作する。変数は、宣言をすることなく使え、どのような型の値でも代入できるため、コンパイル時にエラーが出る頻度が少ない。
- **プログラムの可視化**
  Nigari System には、ユーザが、特別なグラフィックス命令を書かなくても自動的にプログラムの動きを可視化する機能が組み込まれている。ユーザは、プログラムの流れを直感的に理解することができるし、興味惹かれるアプリケーションを簡単に作ることができる。
  ユーザは、実行時に出現するオブジェクトを、実行前に予め GUI を用いて配置することが可能である。配置したオブジェクトの振る舞いを記述してプログラムを作り上げるので、学習者にオブジェクトという概念を強く印象づけ、オブジェクト指向の基礎を体得させることが可能である。
- **Java に近い言語仕様**
  学習が進行すると、実用的な言語の習得も必要となる。Nigari の言語仕様は、式や制御文の構造などをほとんど Java と同じにして、実用言語として学ぶ Java への移行の際に混乱を来たさないようにしている。

† 一橋大学
Hitotsubashi University

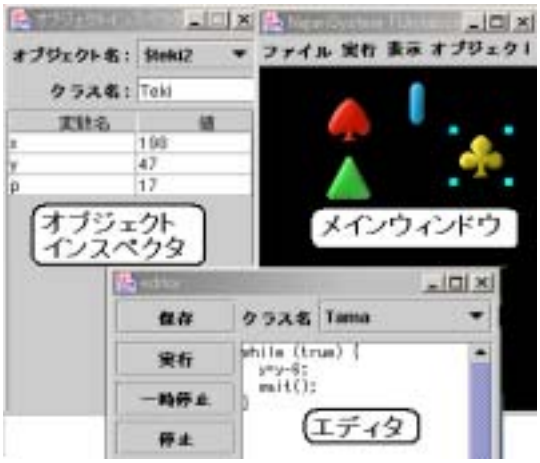


図 1 Nigari System のスクリーンショット

2. プログラミング環境 Nigari System

図 1 に、Nigari System のスクリーンショットを示す。Nigari System は、プラットフォームによらず使えることを目指して、Java で実装した。実際、Linux（Vine 2.5）でも Windows（2000/Me/XP）でも、JDK がインストールしてあれば動作する、

Nigari System を用いたプログラムは，次のような手順で作成する．

- **オブジェクトを配置・編集する**
  ユーザは、実行時に出現するオブジェクトを、実行前に予め配置することが可能である。オブジェクトの配置は GUI を用いて行う。配置されたオブジェクトは、メインウィンドウに表示される。
- **プログラムを編集する**
  配置した各オブジェクトの動作記述であるプログラムを、エディタを用いて編集する。
- **実行する**
  プログラムを実行すると、各オブジェクトがそのプログラムに応じて並行に処理を行う。それにともなって、オブジェクトはメインウィンドウ上に可視化される。

```
while(x<300) {
  x=x+5;
  wait(10);
}
```

図 2 Nigari のサンプルプログラム（1）

```
while(true){
  if(x<getMouseX()) x=x+1;
  if(x>getMouseX()) x=x-1;
  wait(10);
}
```

図 3 Nigari のサンプルプログラム（2）

3. プログラム例

3.1 変数の値によるアニメーション

図 2 に，Nigari のプログラムの一例を示す。このプログラムを実行したオブジェクトは、オブジェクト変数 x の値を 5 ずつ増やし、組み込みメソッドである wait メソッドを呼ぶ、という動作を繰り返す。このオブジェクトは、Nigari System のメインウィンドウ上では、右に 5 ドット移動し、少し（約 10 ミリ秒）待つという動作の繰返しとして観察される．これは、2 のメインウィンドウの項目で説明したように、変数 x，y，p の値に応じてそのオブジェクトが自動的にメインウィンドウ上に表示されるからである。

3.2 マウス入力と if 文

図 3 に示したプログラムは、オブジェクトが、自分自身と、マウスカーソルの x 座標の位置関係によって、左右に移動する（マウスカーソルのある方向に寄ってくる）プログラムである。組み込みメソッドの getMouseX を用いて，マウスの位置と自分の変数 x の値を大小比較し、比較結果に応じて動作を変化させている。

3.3 他のオブジェクトの参照

図 4 に示したプログラムは、このプログラムで動作するオブジェクトとは別のオブジェクト $player を追跡するプログラムである。

このプログラムを正しく実行するには、追跡の対象となるオブジェクトを設計時に $player という名前で配置しておく。

```
while (y<180) {
  if (x>$player.x) x=x-1;
  if (x<$player.x) x=x+1;
  y=y+1;
  wait(10);
}
```

図 4 Nigari のサンプルプログラム（3）

```
while (true) {
  if(getKey(1)==1){
    t=new Tama();
    t.x=x;t.y=y;
  }
  wait(10);
}
```

図 5 Nigari のサンプルプログラム（4）

$player.x は、オブジェクト$playerがもつ変数 x を間接参照している。

3.4 オブジェクトの動的生成

図 5 に示したプログラムで動作しているオブジェクトは、マウスボタンが押されるたびに、新しく Tama クラスのオブジェクトを生成する。生成したオブジェクトを変数 t に代入し、間接参照を用いて、新しいオブジェクトの位置を、自分と同じ位置に設定している。

4. 実　　験

Nigari の有効性を確認するため，Nigari の特徴を最大限に生かしたコースデザインを行い、実際の大学の授業で使用した。その結果を報告する。

授業は、早稲田大学理工学部コンピュータネットワーク工学科（CS 学科）1 年を対象にし、Java 言語とプログラミングの基礎の習得を目的とした授業であった。2003 年 4 月 14〜同年 7 月 7 日の期間に 12 回（各 2 時間）行われた。

各自ノートパソコン持参、授業 1 回ごとに、説明を受け、演習を行う形式で、作成したテキスト（教本）はすべて Web 上で閲覧可能としておいた。演習問題やアンケートの解答も Web を利用した。

4.1 授業の進め方

授業内容を**図 6** に示す.

4/14	講義概要
4/21	PC の使い方とプログラミング
4/28	Nigari のインストールと試用
5/12	変数 / while 文
5/19	while 文（つづき） / if 文
5/26	if 文（つづき） / マウス入力
6/02	複数のオブジェクト/マルチスレッド
6/09	メソッド / オブジェクトの実行時生成（6/16 から Java を使用）
6/16	Java の概要 / 変数の宣言 / while 文
6/23	制御構造/メソッド/文字列
6/30	配列 / コマンドライン引数 / 並べ換え
7/07	並べ換え（つづき） / 文字入力
7/14	（補講） クラス試験

図 6 授業内容

6 月 9 日の授業までは Nigari System を利用し、6 月 16 日以降に Java の説明と実習を行った。

4.1.1 Nigari を使った授業

Nigari を使った授業では、変数、制御構造などの基礎を習得させた。Nigari を使った授業の実習において出題した演習問題の一部を**図 7** に示す。また、5 月 19 日の「if 文」の項目で使用したテキストの一部を**図 8** に示す。

4.1.2 Java を使った授業

Java の授業では、その日学ぶ Java の構文についての説明を行う前に、**図 9** で示したような質問を行い、Nigari での経験から類推させてみる、ということを試みた。例えば、問題 8-2 は、型の概念や変数宣言を教える前に、問題 9-1 は、Java の while 文を教える前に、問題 10-2 は、Java のメソッドの書き方を教える前に出題したものである。以下、このような質問を類推テストと呼ぶ。この類推テストは、Nigari で習得したことが、そのまま Java に応用できるかどうかを試験し、Nigari の学習効果を測定する目的で行った。

4.2 アンケート

次のような調査をアンケートにて行った。アンケートはすべて Web にて回答する方式をとった。

- **毎回の授業評価（4 月 28-6 月 30 日）**
- **類推テストの解き方（6 月 16 日、6 月 23 日）**
- **総合評価（7 月 7 日）**

マウスを使ってオブジェクトを動かす

この章では if 文に関する学習を行いますが、その前の準備段階としてマウスカーソルの座標を得る方法をまず学習します。mouseCheck という名前の新しいページを作り、次のプログラムを実行してみましょう。マウスカーソルの位置にオブジェクトが表示されます。マウスを動かすと、マウスカーソルも動きオブジェクトも一緒に動きます。

```
while(true){
  x= getMouseX();
  y= getMouseY();
  wait(50);
 }
```

if 文とは

さて、本題の if 文についての説明に入ります。
if 文 (if statement) は、ある条件の時にだけこの処理をしたいといった、条件付きの動作を行うプログラムを作りたいときに if 文を用います。if 文は、つぎの形に書きます。

```
if( 条件１){
  処理１
}
```

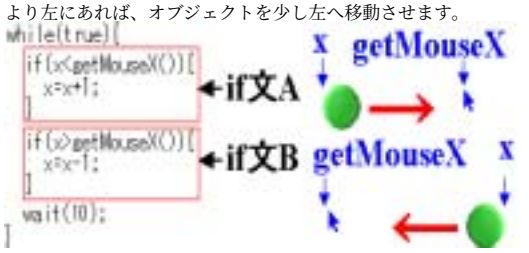
条件１が成り立って場合にだけ処理１が行われ、条件１が成り立っていない場合には処理１は行われません。

オブジェクトの動きに変化をつける。

if という新しいページを作り、プログラムを実行してみましょう。
プログラム 4-3

```
while(true){
  if(x<getMouseX()){
    x=x+1;
  }
  if(x>getMouseX()){
    x=x-1;
  }
  wait(10);
}
```

このプログラムでは図のように二つの if 文が用いられています。一つ目の if 文を if 文 A、二つ目の if 文を if 文 B とします。
if 文 A では条件としてマウスカーソルがオブジェクトより右にあるかどうかを判定し、 図のようにマウスカーソルがオブジェクトより右にあれば、オブジェクトを少し右へ移動させます。
if 文 B では条件としてマウスカーソルがオブジェクトより左にあるかどうかを判定し、 図のようにマウスカーソルがオブジェクトより左にあれば、オブジェクトを少し左へ移動させます。



演習問題

getMouseY メソッドを使って、マウスカーソルがオブジェクトよりも上にあるとオブジェクトがゆっくりと上に進み、マウスカーソルがオブジェクトよりも下にあるとゆっくりと下に進むプログラムを作ってみましょう。

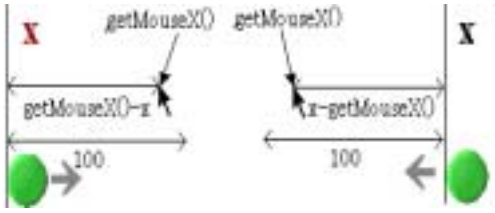
マウスカーソルとオブジェクトの距離によってオブジェクトが動いたり、動かなかったりさせる。

次のようなプログラムを実行させてみましょう。オブジェクトとマウスカーソルの横方向の距離が 100 以上ある場合、オブジェクトは移動しません。

```
while(true) {
  if(x<getMouseX()){
    if(getMouseX()-x<100){
      x=x+1;
    }
  }
  if(x>getMouseX()){
    if(x-getMouseX()<100){
      x=x-1;
    }
  }
  wait(10);
}
```

条件と説明

「オブジェクトとマウスカーソルの横方向の距離が 100 以上あるときは動かない」⇒「オブジェクトとマウスカーソルの横方向の距離が 100 未満のときだけ動く」
「x ＜ getMouseX」 マウスカーソルがオブジェクトより右側にあるかを判定
「getMauseX()-x ＜ 100」 マウスカーソルとオブジェクトの距離が 100 未満かを判定
「getMouseX ＜ x」 マウスカーソルがオブジェクトより左側にあるかを判定
「x-getMouseX() ＜ 100」 マウスカーソルとオブジェクトの距離が 100 未満かを判定



演習問題

逆に、マウスカーソルとオブジェクトが 100 以上離れていた場合だけ、動くようにするプログラムを書きましょう。

図 8 授業で用いたテキスト（抜粋）

5. 実 験 結 果

アンケートの集計結果を 5.1〜13 に示す。

5.1 授業の難しさの変化

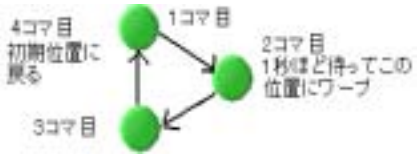
図 10 に、アンケート結果に基づく、授業の難しさの、授業ごとの変化を示す。

回が進むにつれて、難易度は上がっているが、6 月 16 日は最初の Java での授業であるが、難しさが低下している。これは Nigari で習得した変数や while 文の概念をそのまま Java に応用することができたからだと考えられる。

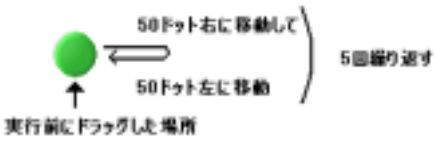
5.2 類推テストの解き方

図 11 に、類推テストの解き方に関する回答を示す。半分強の学生が、まだ習っていない Java の構文に関する問題を、Nigari の知識だけで解くことが可能であった。

5/12（変数） 次の図のように、三角形の頂点を右回りで移動して最初の場所に戻る 4 コマのアニメーションを作りましょう。



5/19（while 文） 次の図のように、左右に 5 往復するオブジェクトを作りましょう。



5/26（if 文） マウスカーソルを追いかけるオブジェクトを作りましょう。

6/2（複数のオブジェクト） 2 つのオブジェクトが万有引力の法則に従って運動する様子をシミュレートしましょう。

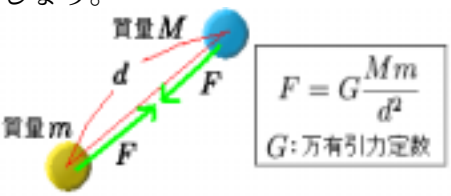


図 7 演習問題（抜粋）

問題 8-2： 何と出力されるでしょう

```
public void printHello() {
  int x=2+3;
  int y=x*2;
  x=x+y*5;
  System.out.println(x);
}
```

問題 9-1： 同じ動作を while 文で書きましょう

```
int i=2;
System.out.println(
i+"の２乗は"+i*i);i++;
System.out.println(
i+"の２乗は"+i*i);i++;
System.out.println(
i+"の２乗は"+i*i);i++;
System.out.println(
i+"の２乗は"+i*i);i++;
```

図 9 類推テストの一部

★ nigariImgs/easytime.eps

図 10 アンケート結果：授業の難しさの変化

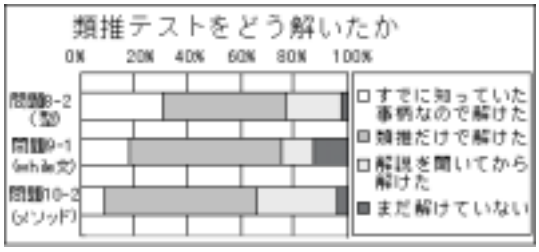


図 11 アンケート結果：類推テストをどのように解いたか

5.3 授業の楽しさの変化

図 12 に、授業の楽しさの、授業ごとの変化を示す。

授業の難しさが上がるにつれ、楽しさは減少するが、難しさの上昇に比べて緩やかな減少で、「難しいけれどおもしろい」と感じている学生が多いと考えられる。Java の授業に入ると、楽しさの減少が目立つようになる。

5.4 Nigari を用いた授業の利点と欠点

7 月 7 日実施の最終アンケートに、「この授業の良かった点」「この授業の改善してほしい点」「その他，授業に関する意見をご自由に」の 3 つの自由記入欄を設置した。これらの欄に書かれた意見のうち、Nigari に関する意見が 30 件含ま

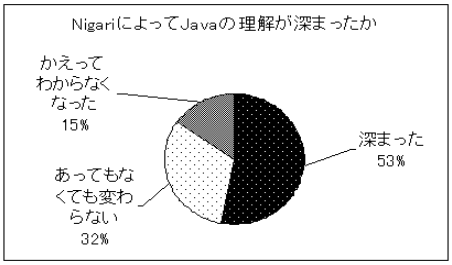


図 12 アンケート結果：授業の楽しさの変化

図 13 アンケート結果：Nigari によって Java の理解が深まったか

表 1 アンケート結果：プログラミング経験

プログラミング経験あり	27 名
プログラミング経験なし	54 名

表 2 アンケート結果：自由意見（括弧内はプログラミング経験者の内数）

授業に対する意見総数	92 件
Nigari に関する肯定的意見	19 件
Nigari に関する否定的意見	11 件
肯定的意見の詳細:	（件）
敷居が低くとっつきやすい	7(3)
プログラムが視覚的	7(5)
オブジェクト指向の理解の助けになる	2(2)
その他	5
否定的意見の詳細:	(件)
早く Java に移ってほしかった	6(4)
Java とのギャップを感じた	3(2)
Nigari ではカバーできない点が多い	1(0)
簡単すぎる	1(1)

れた。

Nigari の利点として、「プログラムがアニメーションとして視覚的に表示されてわかりやすい・楽しい」「初心者にとってとっつきやすい」といった意見が多く見られた。また「オブジェクト指向、マルチスレッドなどの概念を理解する手助けとなった」という意見もあった。

一方、Nigari を学習する時間によって、本来の Java の学習ができる時間が減ったという意見が見られた。この意見はプログラミングの経験者から特に多く寄せられた。また、配列や文字入力など、Nigari では扱わなかった仕組みを Java で学習しなければならないので、Nigari を用いることに疑問を感じる学生もいた。

6. ま と め

Java 言語への導入として、Java に似ていて、かつ簡素な言語を用いて Java の基礎を学習するための言語 Nigari とその環境 Nigari System を提案した。

実験では、最初に Nigari を用いて、その後 Java へ移行するという授業を実施し、この方法によって学習者への負担を減らし、学習者の興味を持続させることが可能であり、しかも Nigari で学習したことが Java にも適用できることを示した。

この授業で得られたノウハウを活かして、大学における一般情報教育において Web プログラミングを実習するための環境 Aroe[3] を開発し、来期からの授業で利用を開始する予定である。

参 考 文 献

1) 長慎也, 川合 晶, 日野孝昭, 前島真一：Nigari System (2003). http://tonyu.jp/Nigari/.
2) 長慎也, 甲斐宗徳, 川合晶, 日野孝昭, 前島真一, 筧捷彦：プログラミング環境 Nigari - 初学者が Java を習うまでの案内役, 情報処理学会論文誌:プログラミング, Vol.45, No.SIG9, pp.25–46 (2004).
3) : Aroe. http://aroe.jp/.