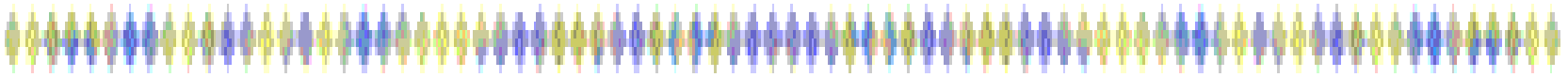
第10回　Javaゼミ



10.4　VectorクラスとEnumerationインターフェイス

10.5　Stackクラス

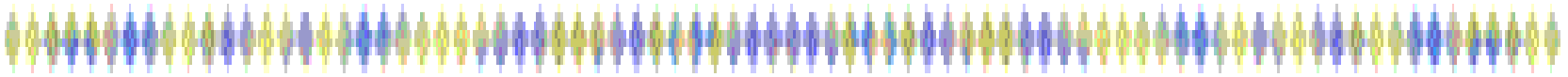
10.6　Hashtableクラス

10.7　StringTokenizerクラス

発表日：2007/6/15

発表者：**阿部 竜之介**

Vectorクラス



Vectorクラス

■要素を追加した分だけ自動的にサイズが拡張される動的配列（ベクトル）を作成する

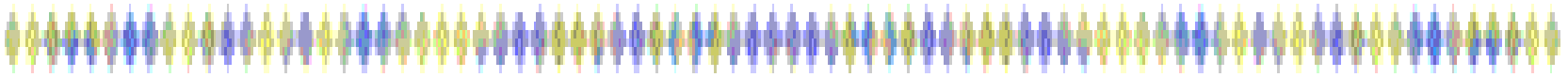
例

```
Vector vector = new Vector();                  //ベクトルの作成
vector.addElement( new Integer(5) );       //ベクトルに要素を追加
vector.addElement( new String(“Hello”) );
vector.addElement( new Float(-14.14f) );
System.out.println( vector );
```

表示されるメッセージ

[5, Hello, -14.14]

Enumerationインターフェイス



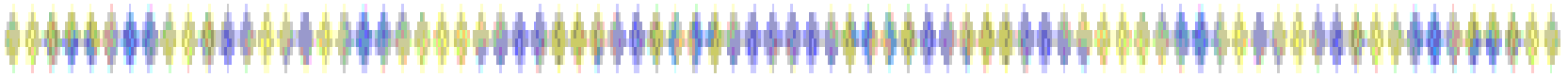
Enumerationインターフェイス

- オブジェクト群に対して一定の処理を繰り返すために用いる

```
boolean hasMoreElements()
Object nextElement()
```

- 上記の2つのメソッドが定義されている
- hasMoreElements()メソッドは、まだ要素がある場合に真を返す
- nextElement()メソッドは次にある要素を返す

Stackクラス



Stackクラス

■Vectorクラスを拡張し、LIFO型（最後に入れたものを最初に取り出す方式）スタックを提供する

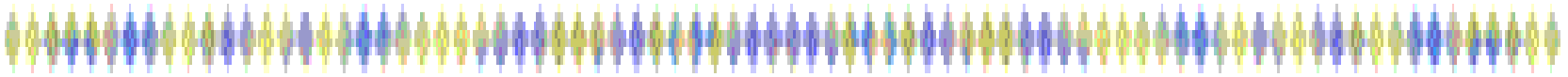
例

```
Stack stack = new Stack();          //スタック作成
for( int i=0; i<5; i++ )
  stack.push( new Integer ( i ) );  //要素をスタックにプッシュ
while(!stack.empty() ) {            //スタックが空になるまで繰り返す
  Object obj = stack.pop();         //スタックから要素をポップ
  System.out.println( obj );
}
```

結果

```
4
3
2
1
0
```

Hashtableクラス(1)



Hashtableクラス

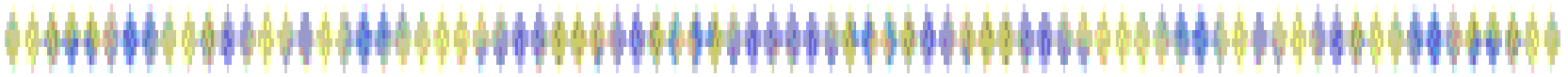
■インデックスを使わず、キーを使って要素を取り出す（連想配列の）ハッシュ表を作成する

◆キーは一意でなければならない

◆ハッシュ表に追加した順番と、取り出したときの順番は必ずしも一致しない

◆どのように格納されるかはJVMの実装によって異なる

Hashtableクラス(2)



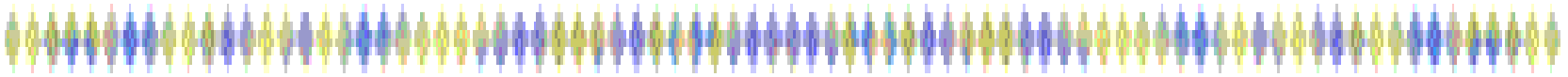
例

```
Hashtable hashtable = new Hashtable(); //ハッシュ表を作成
hashtable.put( “apple”, “red” );              //キーと値のペアをハッシュ表に
hashtable.put( “starwberry”, “red” );         //追加する
hashtable.put( “lime”, “green” );
hashtable.put( “banana”, “yellow” );
hashtable.put( “orange”, “orange” );
System.out.print( “The color of an apple is: ” );
Object v = hashtable.get( “apple” );          //キーappleに関連付けられた値を返す
System.out.println( v );
```

表示されるメッセージ

```
The color of an apple is: red
```

StringTokenizerクラス



StringTokenizerクラス

■文字列をトークン（文字列の構成要素）に分解するために使う

◆StringTokenizerクラスには、Enumerationインターフェイスが実装される

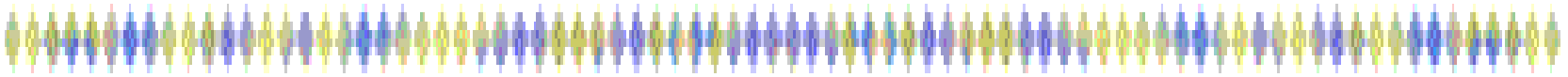
例

```
String str = "123/45.6/-11.2";
StringTokenizer st = new StringTokenizer( str, “/” );//スラッシュを区切り文字として使用
while( st.hasMoreTokens() ) {     //トークンが残っていれば繰り返す
    String s = st.nextToken();       //トークンを取り出す
    System.out.println( s );
}
```

結果

```
123
45.6
-11.2
```

ジェネリックス(1)

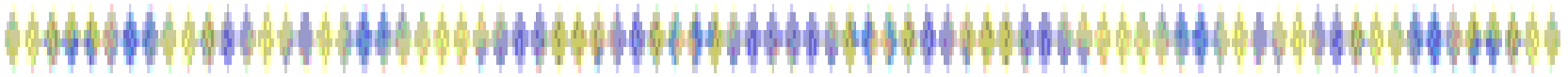


ジェネリックス

- JDK5.0で用意されている仕組みで、型パラメータを明示して、その中での型変換をコンパイラで行う
- 不定のオブジェクトを取り込む集合を扱うVectorクラスやHashtableクラスなどで実装されている
- 前述の「Vectorクラス」の項の例では、Objectクラスに由来するオブジェクト全般を扱うため、以下のようにvectorオブジェクトを生成するようにする

```
Vector<object> vector = new Vector<object>();
```

ジェネリックス(2)



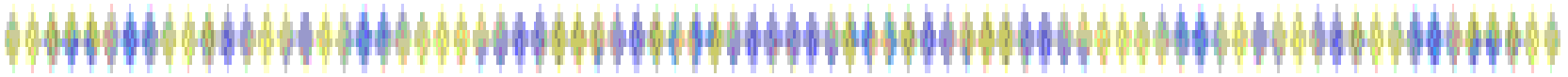
■前述の「Hashtableクラス(2)」の項の例では、以下のように記述する

```
Hashtable<String,String>hashtable = new Hashtable<String,String>();
```

■このように記述しない場合、警告が表示される（エラーではない）

■<クラス>で指定した以外のクラスのオブジェクトを追加しようとするとエラーが表示される

練習問題



　コマンドラインから複数の整数を受け取り、その引数をベクトルに格納し、入力された順序とは反対の順序で表示するプログラムを作成せよ。