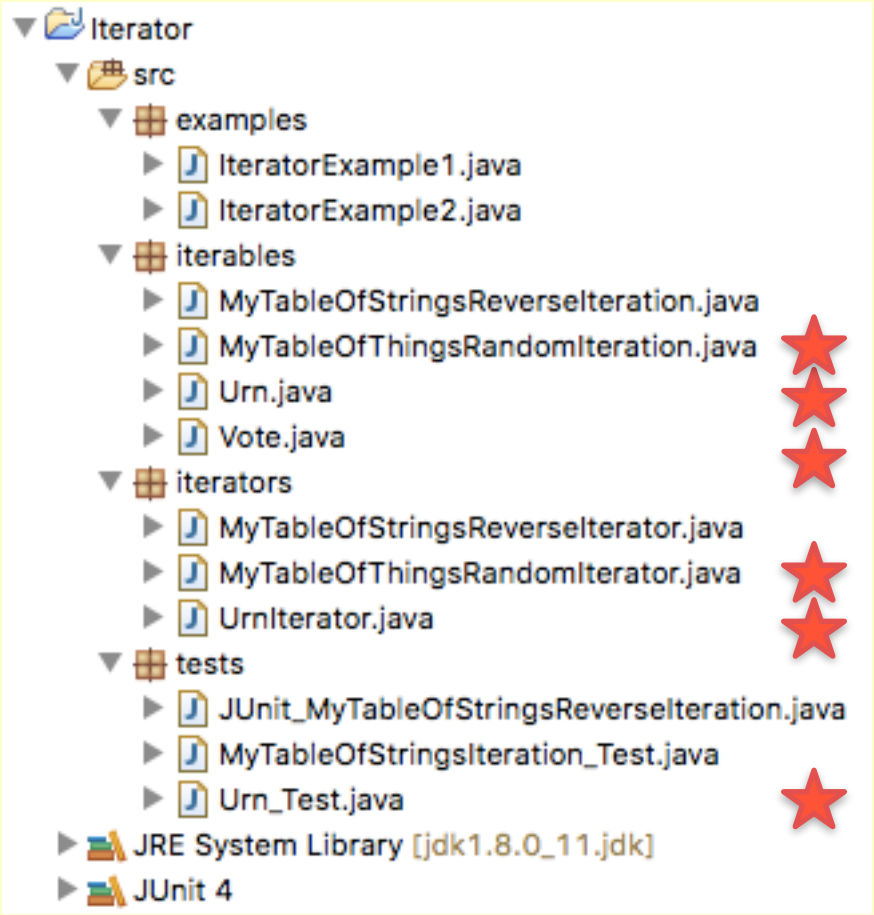
http://www-public.telecom-sudparis.eu/~gibson

paul.gibson@telecom-sudparis.eu

The iterator problem solution

…/~gibson/Teaching/DesignPatterns/CSC4522-DesignPatterns-IteratorSolution.pdf

The Packages



The Solution: the urn (and test)

```
public class MyTableOfThingsRandomIteration <T> implements Iterable<T> {


public T[] data;

public MyTableOfThingsRandomIteration(T [] data) {
this.data = data;
}

public int length(){return data.length;}

public Iterator<T> iterator() {
return new MyTableOfThingsRandomIterator<T>(this);
}

}
```

The Solution: the urn (and test)

```
public class MyTableOfThingsRandomIterator <T> implements Iterator<T> {

private int index;
private MyTableOfThingsRandomIteration<T> table;
private int [] randomindexes;

private void initialise_randomindexes(int length){
       randomindexes = new int [length];
       for (int i =0; i<length;i++) randomindexes[i]=i;}

private void randomindexswap(){… } // EDITED OUT
private void shuffle_randomindexes(int numberOfSwaps){ …} //EDITED OUT

public MyTableOfThingsRandomIterator(MyTableOfThingsRandomIteration <T> tab) {
index = 0;
table = tab;
initialise_randomindexes(table.length());
shuffle_randomindexes(table.length());}


public MyTableOfThingsRandomIterator( T [] tab) {
index = 0;
table = new MyTableOfThingsRandomIteration<T>(tab);
initialise_randomindexes(table.length());
shuffle_randomindexes(table.length());}

public T next() {
index++;
return (T) table.data[randomindexes[index -1]];}

public boolean hasNext() {
return index < table.length();}

public void remove() {
throw new UnsupportedOperationException();}
}
```

The Solution: the urn (and test)

```
public class Vote implements Iterable<String>{


    public String[] data;

    public Vote(String [] data) {
    this.data = data;
    }

    public int length(){return data.length;}

    public Iterator<String> iterator() {
    return new MyTableOfThingsRandomIterator <String> (data);
    }

}
```

The Solution: the urn (and test)

```
public class Urn implements Iterable<String>{



    public final  Vote [] VOTES;

    public Urn(Vote [] votes) {

        this.VOTES = votes;

    }


    @Override
    public Iterator<String> iterator() {

        return new UrnIterator(this);
    }


}
```

The Solution: the urn (and test)

```
public class UrnIterator implements Iterator<String>{

    MyTableOfThingsRandomIterator <Vote> voteIterator;
    Vote nextVote;
    MyTableOfThingsRandomIterator <String> preferenceIterator;
    String nextPreference;

      public UrnIterator(Urn urn) {
            voteIterator = new MyTableOfThingsRandomIterator <Vote> (urn.VOTES);
            preferenceIterator = new MyTableOfThingsRandomIterator <String> (voteIterator.next().data);
      }

      @Override
      public boolean hasNext() {
            return (preferenceIterator.hasNext() || voteIterator.hasNext());
      }

      @Override
      public String next() {

   if (!preferenceIterator.hasNext()){
         preferenceIterator = new MyTableOfThingsRandomIterator <String> (voteIterator.next().data);
   }

   return preferenceIterator.next();
      }

}
```

The Solution: the urn (and test)

```
public class Urn_Test {

      public static void main(String[] s) {

            String [] preferences1 = {"gibson", "smyth", "hughes"};
            Vote vote1 = new Vote(preferences1);

            String [] preferences2 = {"jones", "bell"};
            Vote vote2 = new Vote(preferences2);

            String [] preferences3 = {"raffy","lallet", "muller", "bac"};
            Vote vote3 = new Vote(preferences3);

            Vote [] votes = { vote1, vote2, vote3};

            Urn urn = new Urn (votes);

            System.out.println("\nIterate over strings on bulletins in Urn");
            for (String value : urn) {
                  System.out.println(" "+value);
                  }
            }

}
```

The Solution: the urn (and test)

```
Iterate over strings on bulletins in Urn
 bac
 lallet
 raffy
 muller
 bell
 jones
 smyth
 hughes
 gibson
```

QUESTION: What other tests could/should we run?
How to improve the solution?