Recipes for PDSC #3: Lists of Hashes

by Tom Christiansen
< tchrist@perl.com >

release 0.1 (untested, may have typos)
Sunday, 1 October 1995

Declaration of a LIST OF HASHES:

```
@LoH = (
       {
          Lead      => "fred",
          Friend    => "barney",
       },
       {
           Lead     => "george",
           Wife     => "jane",
           Son      => "elroy",
       },
       {
           Lead     => "homer",
           Wife     => "marge",
           Son      => "bart",
       }
 );
```

Generation of a LIST OF HASHES:

```
# reading from file
# format: LEAD=fred FRIEND=barney
while ( <> ) {
    $rec = {};
    for $field ( split ) {
        ($key, $value) = split /=/, $field;
        $rec->{$key} = $value;
    }
    push @LoH, $rec;
}

# reading from file
# format: LEAD=fred FRIEND=barney
# no temp
while ( <> ) {
    push @LoH, { split /[\s+=]/ };
}

# calling a function  that returns a key,value list, like
# "lead","fred","daughter","pebbles"
while ( %fields = getnextpairset() )
    push @LoH, { %fields };
}
```

```
# likewise, but using no temp vars
while (<>) {
    push @LoH, { parsepairs($_) };
}

# add key/value to an element
$LoH[0]{"pet"} = "dino";
$LoH[2]{"pet"} = "santa's little helper";
```

Access and Printing of a LIST OF HASHES:

```
# one element
$LoH[0]{"lead"} = "fred";

# another element
$LoH[1]{"lead"} =~ s/(\w)/\u$1/;

# print the whole thing with refs
for $href ( @LoH ) {
    print "{ ";
    for $role ( keys %$href ) {
        print "$role=$href->{$role} ";
    }
    print "}\n";
}

# print the whole thing with indices
for $i ( 0 .. $#LoH ) {
    print "$i is { ";
    for $role ( keys %{ $LoH[$i] } ) {
        print "$role=$LoH[$i]{$role} ";
    }
    print "}\n";
}

# print the whole thing one at a time
for $i ( 0 .. $#LoH ) {
    for $role ( keys %{ $LoH[$i] } ) {
        print "elt $i $role is $LoH[$i]{$role}\n";
    }
}
```