浮点数转字符串:随机数字 浮点数 字符串产生

JSP中经常用到随机数字或(如密码产生sessionid产生)可以使用taglib标签产生,本人使用bean产生随机数:

1.可以产生10000000和99999999的间随机数

2.可以产生规定数字的间随机数,如25100的间

3.可以使用algorithm和provider产生个SecureRandom随机数字或串
objectinsteadofaRandomobject:71

4.可以产生浮点浮点随机数;

5.可以产生a-zA-Z0-9的间规定长度个串

6.可以产生规定长度小写字母串

7.可以产生任意串.

以下jspbean在TomcatWin2000下调试通过:

```
 /*
*
*随机数字bean
*/

packagemycollect;

importjava.util.*;
importjava.security.SecureRandom;
```

```
importjava.security.NoSuchAlgorithmException;
importjava.security.NoSuchProviderException;


publicRandomNum{

privateLongrandomnum=null;
privateFloatrandomfloat=null;
privatebooleanfloatvalue=false;
privatelongupper=100;
privatelonglower=0;
privateStringalgorithm=null;
privateStringprovider=null;
privatebooleansecure=false;
privateRandomrandom=null;
privateSecureRandomsecrandom=null;

privatefinalfloatgetFloat{
(randomnull)
secrandom.nextFloat;

random.nextFloat;
}

publicfinalvoidgenerateRandomObjectthrowsException{

//checktoseetheobjectisaSecureRandomobject
(secure){
try{
//getaninstanceofaSecureRandomobject
(provider!=null)
//searchforalgorithminpackageprovider
secrandom=SecureRandom.getInstance(algorithm,provider);

secrandom=SecureRandom.getInstance(algorithm);
}catch(NoSuchAlgorithmExceptionne){
```

```
throwException(ne.getMessage);
}catch(NoSuchProviderExceptionpe){
throwException(pe.getMessage);
}
}
random=Random;
}

/**
*generatetherandomnumber
*
*/
privatefinalvoidgeneraterandom{

tmprandom=0;//tempstorageforrandomgeneratednumber
Integerrand;

//checktoseefloatvalueisexpected
(floatvalue)
randomfloat=Float(getFloat);

randomnum=Long(lower+(long)((getFloat*(upper-lower))));
}

publicfinalNumbergetRandom{
generaterandom;
(floatvalue)
randomfloat;

randomnum;
}

publicfinalvoidRange(longlow,longup){

//theupperandlowerboundoftherange
lower=low;
```

```
upper=up;

//checktoseeafloatvalueisexpected
((lower0)&&(upper1))
floatvalue=true;
}

/**
*thealgorithmname
*
*@paramvaluenameofthealgorithmtouseforaSecureRandomobject
*
*/
publicfinalvoidAlgorithm(Stringvalue){
algorithm=value;
secure=true;//aSecureRandomobjectistobeused
}

publicfinalvoidProvider(Stringvalue)
{
provider=value;
}

publicfinalvoidRange(Stringvalue)throwsException
{
try
{
upper=Integer(value.sub(value.indexOf(\'-\')+1)).longValue;
}catch(Exceptionex){
throwException("upperattributecouldnotbe"+
"turnedoanIntegerdefaultvaluewasused");
}

try
{
lower=Integer(value.sub(0,value.indexOf(\'-\'))).longValue;
```

```
}catch(Exceptionex){
throwException("lowerattributecouldnotbe"+
"turnedoanIntegerdefaultvaluewasused");
}



((lower0)&&(upper1))
floatvalue=true;

(upper<lower)
throwException("Youcan\'thavearangewherethelowerbound"+
"ishigherthantheupperbound.");

}


}
```

随机串bean

```
 packagemycollect;

importjava.util.*;
importjava.security.SecureRandom;
importjava.security.NoSuchAlgorithmException;
importjava.security.NoSuchProviderException;

publicRandomStrg{

privateStringrandomstr;
```

```
privatebooleanallchars=false;
//欠缺是8位串
privateIntegerlength=Integer(8);
privateHashMaphmap;
privateArrayListlower=null;
privateArrayListupper=null;
privatecharsingle=null;
privatesinglecount=0;
privatebooleansingles=false;
privateStringalgorithm=null;
privateStringprovider=null;
privatebooleansecure=false;
privateRandomrandom=null;
privateSecureRandomsecrandom=null;

privatefinalfloatgetFloat{
(randomnull)
secrandom.nextFloat;

random.nextFloat;
}

/**
*generatetheRandomobjectthatwillbeusedforthisrandomnumber
*generator
*
*/
publicfinalvoidgenerateRandomObjectthrowsException{

//checktoseetheobjectisaSecureRandomobject
(secure){
try{
//getaninstanceofaSecureRandomobject
(provider!=null)
//searchforalgorithminpackageprovider
random=SecureRandom.getInstance(algorithm,provider);
```

```
random=SecureRandom.getInstance(algorithm);
}catch(NoSuchAlgorithmExceptionne){
throwException(ne.getMessage);
}catch(NoSuchProviderExceptionpe){
throwException(pe.getMessage);
}
}
random=Random;
}

/**
*generatetherandom
*
*/
privatefinalvoidgeneraterandom{

(allchars)
for(i=0;i<length.Value;i)
randomstr=randomstr+Character((char)(()34+
((()(getFloat*93)))).toString;
(singles){
//checktherearesinglecharstobed

(upper.size3){
//checkforthenumberofrangesmax3upperlowerdigits

//buildtherandom
for(i=0;i<length.Value;i){
//youhavefourgroupstochoosearandomnumberfrom,tomake
//thechoicealittlemorerandomselectanumberoutof100

//getarandomnumberevenorodd
(((()(getFloat*100))%20){

//thenumberwasevengetanothernumberevenorodd
```

```
((()(getFloat*100))%20)
//choosearandomcharfromthesinglechargroup
randomstr=randomstr+randomSingle.toString;

//getarandomcharfromthefirstrange
randomstr=randomstr+randomChar((Character)lower.get(2),
(Character)upper.get(2)).toString;
}{
//thenumberwasodd

((()(getFloat*100))%20)
//choosearandomcharfromthesecondrange
randomstr=randomstr+randomChar((Character)lower.get(1),
(Character)upper.get(1)).toString;

//choosearandomcharfromthethirdrange
randomstr=randomstr+randomChar((Character)lower.get(0),
(Character)upper.get(0)).toString;
}



}
}(upper.size2){
//singlecharsaretobedchoosearandomcharfrom
//twodferentranges

//buildtherandomcharfromsinglecharsandtworanges
for(i=0;i<length.Value;i){
//selectthesinglecharsorarangetogeteachrandomchar
//from

((()(getFloat*100))%20){

//getrandomcharfromthesinglechars
randomstr=randomstr+randomSingle.toString;
}((()(getFloat*100))%20){
```

```
//gettherandomcharfromthefirstrange
randomstr=randomstr+randomChar((Character)lower.get(1),
(Character)upper.get(1)).toString;
}{

//gettherandomcharfromthesecondrange
randomstr=randomstr+randomChar((Character)lower.get(0),
(Character)upper.get(0)).toString;
}
}
}(upper.size1){

//buildtherandomfromsinglecharsandonerange
for(i=0;i<length.Value;i){
((()getFloat*100)%20)
//getarandomsinglechar
randomstr=randomstr+randomSingle.toString;

//getarandomcharfromtherange
randomstr=randomstr+randomChar((Character)lower.get(0),
(Character)upper.get(0)).toString;
}
}{
//buildtherandfromsinglechars
for(i=0;i<length.Value;i)
randomstr=randomstr+randomSingle.toString;
}
}{

//nosinglecharsaretobedherandom
(upper.size3){

//buildrandomstrngfromthreeranges
for(i=0;i<length.Value;i){
```

```
((()(getFloat*100))%20){

//getrandomcharfromfirstrange
randomstr=randomstr+randomChar((Character)lower.get(2),
(Character)upper.get(2)).toString;
}((()(getFloat*100))%20){

//getrandomcharformsecondrange
randomstr=randomstr+randomChar((Character)lower.get(1),
(Character)upper.get(1)).toString;
}{

//getrandomcharfromthirdrange
randomstr=randomstr+randomChar((Character)lower.get(0),
(Character)upper.get(0)).toString;
}
}
}(upper.size2){

//buildrandomfromtworanges
for(i=0;i<length.Value;i){
((()(getFloat*100))%20)
//getrandomcharfromfirstrange
randomstr=randomstr+randomChar((Character)lower.get(1),
(Character)upper.get(1)).toString;

//getrandomcharfromsecondrange
randomstr=randomstr+randomChar((Character)lower.get(0),
(Character)upper.get(0)).toString;
}
}

//buildrandom
for(i=0;i<length.Value;i)
//getrandomcharfromonlyrange
randomstr=randomstr+randomChar((Character)lower.get(0),
```

```
(Character)upper.get(0)).toString;
}
}

/**
*generatearandomcharfromthesinglecharlist
*
*@s-arandomlyselsctedcharacterfromthesinglecharlist
*
*/
privatefinalCharacterrandomSingle{

(Character(single[()((getFloat*singlecount)-1)]));
}

/**
*generatearandomcharacter
*


*@paramlowerlowerboundfromwhichtogetarandomchar
*@paramupperupperboundfromwhichtogetarandomchar
*
*@s-arandomlygeneratedcharacter
*
*/
privatefinalCharacterrandomChar(Characterlower,Characterupper){
tempval;
charlow=lower.charValue;
charup=upper.charValue;

//getarandomnumberherangelowlow-lowup
tempval=()(()low+(getFloat*(()(up-low))));

//therandomchar
(Character((char)tempval));
```

```
}

/**
*gettherandomlycreatedforusewiththe
*<jsp:getPropertyname=<i>"id"</i>property="randomstr"/>
*
*@-randomlycreated
*
*/
publicfinalStringgetRandom{

randomstr=String;

generaterandom;//generatethefirstrandom

(hmap!=null){

while(hmap.containsKey(randomstr)){
//randomhasalreadybeencreatedgenerateadferentone
generaterandom;
}

hmap.put(randomstr,null);//addtherandom
}

randomstr;
}

/**
*therangesfromwhichtochoohecharactersfortherandom
*
*@paramlowoflowerranges
*@paramupofupperranges
*
*/
publicfinalvoidRanges(ArrayListlow,ArrayListup){
```

```
lower=low;
upper=up;
}


/**
*thehashmapthatisusedtochecktheuniquenessofrandoms
*
*@parammaphashmapwhosekeysareusedtoinsureuniquenessofrandomstrgs
*
*/
publicfinalvoidHmap(HashMapmap){
hmap=map;
}

/**
*thelengthoftherandom
*
*@paramvaluelengthoftherandom
*
*/
publicfinalvoidLength(Stringvalue){
length=Integer(value);

}

/**
*thealgorithmname
*
*@paramvaluenameofthealgorithmtouseforaSecureRandomobject
*
*/
publicfinalvoidAlgorithm(Stringvalue){
algorithm=value;
secure=true;//aSecureRandomobjectistobeused
}
```

```
/**
*theprovidername
*
*@paramvaluenameofthepackagetocheckforthealgorithm
*
*/
publicfinalvoidProvider(Stringvalue){
provider=value;
}

/**
*theallcharsflag
*
*@paramvaluebooleanvalueoftheallcharsflag
*
*/
publicfinalvoidAllchars(booleanvalue){
allchars=value;
}

/**
*the.gif' />ofsinglecharstochoosefromforthisrandomandthe
*numberofcharshe.gif' />
*
*@paramcharsthe.gif' />ofsinglechars
*@paramvaluethenumberofsinglechars
*
*/
publicfinalvoidSingle(charchars,value){
single=chars;//the.gif' />ofchars
singlecount=value;//thenumberofcharsin.gif' />single
singles=true;//flagthatsinglecharsareinuse
}

publicfinalvoidChar(Stringvalue)
```

```
{
//valuestellsthemethodwhetherornottocheckforsinglechars
booleanmore=true;

//createthe.gif' />liststoholdtheupperandlowerboundsforthechar
//ranges
lower=ArrayList(3);
upper=ArrayList(3);



//userhaschosentouseallpossiblecharactersherandom
(value.compareTo("all")0){
allchars=true;//allcharsflag
//allcharsaretobeusedsothereareno singlecharstosort
//through
more=false;
}((value.charAt(1)\'-\')&&(value.charAt(0)!=\'\\\\\\\')){
//runthroughtherangesatmost3
while(more&&(value.charAt(1)\'-\')){

//checktomakesurethatthedashisnotthesinglechar
(value.charAt(0)\'\\\\\\')
;
{
//addupperandlowerrangestotherelist
lower.add(Character(value.charAt(0)));
upper.add(Character(value.charAt(2)));
}

//checktoseethereismoretothechar
(value.length<=3)
more=false;

//createasothatthenextrangethereisone
//startsit
```

```
value=value.sub(3);
}
}

//more=falhereareno singlecharshechar
(more){

single=char[30];//createsingle

//createaoftokensfromtheofsinglechars
StringTokenizertokens=StringTokenizer(value);

while(tokens.hasMoreTokens){
//getthenexttokenfromthe
Stringtoken=tokens.nextToken;

(token.length>1)
//charisa-addittothelist
single[singlecount]=\'-\';

//addthecurrentchartothelist
single[singlecount]=token.charAt(0);
}
}
((lowernull)&&(singlenull))
Char("a-zA-Z0-9");
}
}
```

JSP语句:

```
 <%@pagecontentType="text/html;char=ISO8859_1"%>
<%@pageimport="java.util.*"%>
<jsp:useBeanid="RNUM"scope="page"="mycollect.RandomNum"/>
<jsp:useBeanid="RSTR"scope="page"="mycollect.RandomStrg"/>

<html>
<head>
<title>随机数字浮点数串</title>
<metahttp-equiv="Content-Type"content="text/html;char=gb2312">
</head>

<body>
<h3>随机数字浮点数串</h3>
<%
//Randomgenerator=Random;
//limit=10;
//randomNumber=()(generator.nextDouble*limit);


out.prln("<p>创建在10000000和99999999的间随机数:");
RNUM.Range("10000000-99999999");
RNUM.generateRandomObject;
out.prln("<b>"+RNUM.getRandom.Value+"</b>");

out.prln("<p>在n25和100的间创建个随机数:");
RNUM.Range("25-100");
RNUM.generateRandomObject;
out.prln("<b>"+RNUM.getRandom.Value+"</b>");

%>
<p>Createthesamerandomnumberbetween25and100,onlyuhe<br>
algorithmandproviderattributestoindicatetheuseofaSecureRandom<br>
objectinsteadofaRandomobject:
<%
RNUM.Range("25-100");
RNUM.Algorithm("SHA1PRNG");
```

```
RNUM.Provider("SUN");
RNUM.generateRandomObject;
out.prln("<b>"+RNUM.getRandom.Value+"</b>");

out.prln("<p>Createarandomfloatvalue:");
RNUM.Range("0-1");
RNUM.generateRandomObject;
Stringradio=java.text.DecimalFormat("###.##########").format(RNUM.getRandom);
out.prln("<b>"+radio+"</b>");



out.prln("<p>=");

out.prln("<p>在a-zA-Z0-9的间,也就是数字和26个字母混合随机串,(欠缺是8位,该功能适合创建随机密码和
sessionid)");
RSTR.Char("a-zA-Z0-9");
RSTR.generateRandomObject;
out.prln("<b>"+RSTR.getRandom+"</b>");


out.prln("<p>Createarandom15lowerletterslong:");
RSTR.Char("a-z");
RSTR.Length("15");
RSTR.generateRandomObject;
out.prln("<b>"+RSTR.getRandom+"</b>");

out.prln("<p>Createarandomwithonlycaps:");
RSTR.Char("A-Z");
RSTR.generateRandomObject;
out.prln("<b>"+RSTR.getRandom+"</b>");

out.prln("<p>Createarandom10characterslongwiththechara-fF-K!\\\\$%#^-*?noticethatthe-
andhadtobeescapedwitha:");
RSTR.Char("a-fF-K!\\\\$%#^-*?");
RSTR.Length("10");
```

```
RSTR.generateRandomObject;
out.prln("<b>"+RSTR.getRandom+"</b>");


out.prln("<p>Createarandomofallthecharactersanddigits:");
RSTR.Char("all");
RSTR.generateRandomObject;
out.prln("<b>"+RSTR.getRandom+"</b>");

%><p>
Createthesamerandomofallthecharactersanddigits,onlyuse<br>
thealgorithmandproviderattributestoindicatetheuseofaSecureRandom<br>
objectinsteadofaRandomobject:
<%
RSTR.Char("all");
RSTR.Algorithm("SHA1PRNG");
RSTR.Provider("SUN");
RSTR.generateRandomObject;
out.prln("<b>"+RSTR.getRandom+"</b>");

%>

</body>
</html>
```

2009-2-12 5:09:34
疯狂代码 http://CrazyCoder.cn/