

Forum:[Hack Forums](#)

Thread: [luck variable for random number generators?](#)

Prock - September 24, 2018, 4:17 pm

hi id like to create a simple rng in javascript for a game but I want there to be a luck variable that can influence the outcome where if something is a 1/250 chance then a variable can scale those chances up or down...does anyone know how to do this?

VirtualFrost - September 26, 2018, 2:10 am

Javascript has a 'random' function which can get you a random integer.

For example, this will get you a random number between 1 and 10, obviously change '10' to what you want the maximum number to be

```
Code Math.floor((Math.random() * 10) + 1);
```

This itself should probably get what you need from what I understand, you just need something to get a random number?

Prock - September 26, 2018, 3:44 am

(Sep 26, 2018 02:10 AM)VirtualFrost Wrote: [\[CITE\]](#) Javascript has a 'random' function which can get you a random integer.

For example, this will get you a random number between 1 and 10, obviously change '10' to what you want the maximum number to be

```
Code Math.floor((Math.random() * 10) + 1);
```

This itself should probably get what you need from what I understand, you just need something to get a random number?

no I know how to get a random number

but basically im trying to create a game with a drop table and the table can give you rarer or not so rare items but you can use objects to increase the rare occurances like the probability of landing a rare is 0.5% but probability of landing a not so rare is 94.5% and probability of landing nothing is 5% so what im thinking of is a variable that can scale the outcome to make it more likely to get a rare or less likely is that makes sense..

Cød3ing - September 26, 2018, 3:49 am

Your question is not about JavaScript or even programming at all.

The answer to your question is a very basic mathematical formula, easily interpreted into code.

Keep searching, stop trying to be spoon fed.

Prock - September 26, 2018, 9:13 am

(Sep 26, 2018 03:49 AM)bogomil Wrote: [\[CITE\]](#) Your question is not about JavaScript or even programming at all.

The answer to your question is a very basic mathematical formula, easily interpreted into code.

Keep searching, stop trying to be spoon fed.

not asking to be spoonfed lol but ok

iMaleKz - September 26, 2018, 10:25 am

i dont really understand this

Faith Connors - September 26, 2018, 12:25 pm

ok so just like get a random number from 1-1000

from 0-50, drop nothing.

from 51-995, drop a common item

from 996 onwards drop a rare item

now, i guess you're scaling rarity by ignoring the gamblers fallacy and the more times you do it, the more likely it is to drop a rare item. once that rare item drops, the 'luck' factor is reset to zero, right?

so add another variable that will act as your luck factor. every time you complete the dungeon or whatever it is and you don't get a rare drop, add one to the counter.

to calculate the chances of dropping something, generate the random number from 1-1000 then add the luck factor to it, then determine what the drop is from that total number.

or something like that, idk.

Prock - September 26, 2018, 1:13 pm

(Sep 26, 2018 12:25 PM)Faith Connors Wrote: [\[CITE\]](#) ok so just like get a random number from 1-1000

from 0-50, drop nothing.

from 51-995, drop a common item

from 996 onwards drop a rare item

now, i guess you're scaling rarity by ignoring the gamblers fallacy and the more times you do it, the more likely it is to drop a rare item. once that rare item drops, the 'luck' factor is reset to zero, right?

so add another variable that will act as your luck factor. every time you complete the dungeon or whatever it is and you don't get a rare drop, add one to the counter.

to calculate the chances of dropping something, generate the random number from 1-1000 then add the luck factor to it, then determine what the drop is from that total number.

or something like that, idk.

this theory is good and it makes sense and yah it's ignoring the gamblers fallacy i like the counter idea so something like $\text{rng() + counter} < \text{probability}$

Cød3ing - September 27, 2018, 9:11 pm

Skimmed the thread on first go, sorry for the shit reply but I can see why you might have had problems with this.

First we're going to setup some variables to hold user percentages for us in decimal form;

```
Code var no_drop = 5.00; // 5%
```

```
var common_drop = 94.50; // 94.5%
```

```
var rare_drop = 0.50; // 0.5%
```

Create a mock pseudo RNG to -

1. Create random number between 0.00 to 100.00 (note decimals)
2. Compare random number generated inside function.
3. Return a drop.

```

Code function roll(){
// "Roll" example
var random_number = (Math.random() * (100 + 1)).toFixed(2);

if (random_number < no_drop){ // NO_DROP
  return 'no_drop';
} else if (random_number < common_drop) { // COMMON_DROP
  return 'common_drop';
} else { // RARE_DROP
  return 'rare_drop';
};
}

```

Now we create a function to modify the drop chance and pass in the drop_type (that we want to modify), state (decrease or increase) and the value to increase or decrease.

Code //state will either decrease or increase percentage.
 //value will be the percentage to increase/decrease.

```

function modify_drop(drop_type, state, value){
  switch (state){
    case 'increase': // INCREASE CASE
      switch(drop_type){
        case 'no_drop':
          no_drop += value;
          modify_drop_logic(drop_type, value);
          break;

        case 'common_drop':
          no_drop += value;
          modify_drop_logic(drop_type, value);
          break;

        case 'rare_drop':
          no_drop += value;
          modify_drop_logic(drop_type, value);
          break;
      }
      break;

```

```

    case 'decrease': // DECREASE CASE
      switch(drop_type){
        case 'no_drop':
          no_drop -= value;
          modify_drop_logic(drop_type, value);
          break;

        case 'common_drop':
          no_drop -= value;
          modify_drop_logic(drop_type, value);
          break;

        case 'rare_drop':
          no_drop -= value;

```

```

        modify_drop_logic(drop_type, value);
        break;
    }
    break;

}
}

```

Note that there's a function called `modify_drop_logic()`, this is used to balance remaining chance -

Code // decrease or increase drops based on `drop_type`

```

function modify_drop_logic(drop_type, value){
    if (drop_type === 'no_drop'){
        common_drop -= +(value / 2).toFixed(2);
        rare_drop -= +(value / 2).toFixed(2);
    } else if (drop_type === 'common_drop'){
        no_drop -= +(value / 2).toFixed(2);
        rare_drop -= +(value / 2).toFixed(2);
    } else { // drop type must be 'rare_drop'
        no_drop -= +(value / 2).toFixed(2);
        common_drop -= +(value / 2).toFixed(2);
    }
}
}

```

This is just a preliminary solution to show you how to work it out, there's some issues but I'm sure you'll be able to fix them.

Prock - September 28, 2018, 12:51 pm

(Sep 27, 2018 09:11 PM)Cød3ing Wrote: [\[CITE\]](#) Skimmed the thread on first go, sorry for the shit reply but I can see why you might have had problems with this.

First we're going to setup some variables to hold user percentages for us in decimal form;

Code //

This is just a preliminary solution to show you how to work it out, there's some issues but I'm sure you'll be able to fix them.

i like this solution the only downside i see to this is the extra number of switch cases for common, none or rare

that's close to the solution i had initially written but i was wondering if there is a more automated process ..like this idea works but what if you have hundreds of different drop tables and hundreds of different drops in each table to create the number of switch cases for each table and each item in each table seems way too much extraneous work what i'm thinking about a is a design that takes one variable called luck and if that value is scaled up then the entirety of the drop table and every drop table available gets scaled to higher rarity and if that value is scaled down then every drop table is less likely to land a rare... i don't exactly need a code solution just a design or an idea of how to pull this off

like even if i had 10 tables and i roll one of those tables i have a global variable called luck and i scale that number up any table it lands on instantly has a higher chance of rolling rares