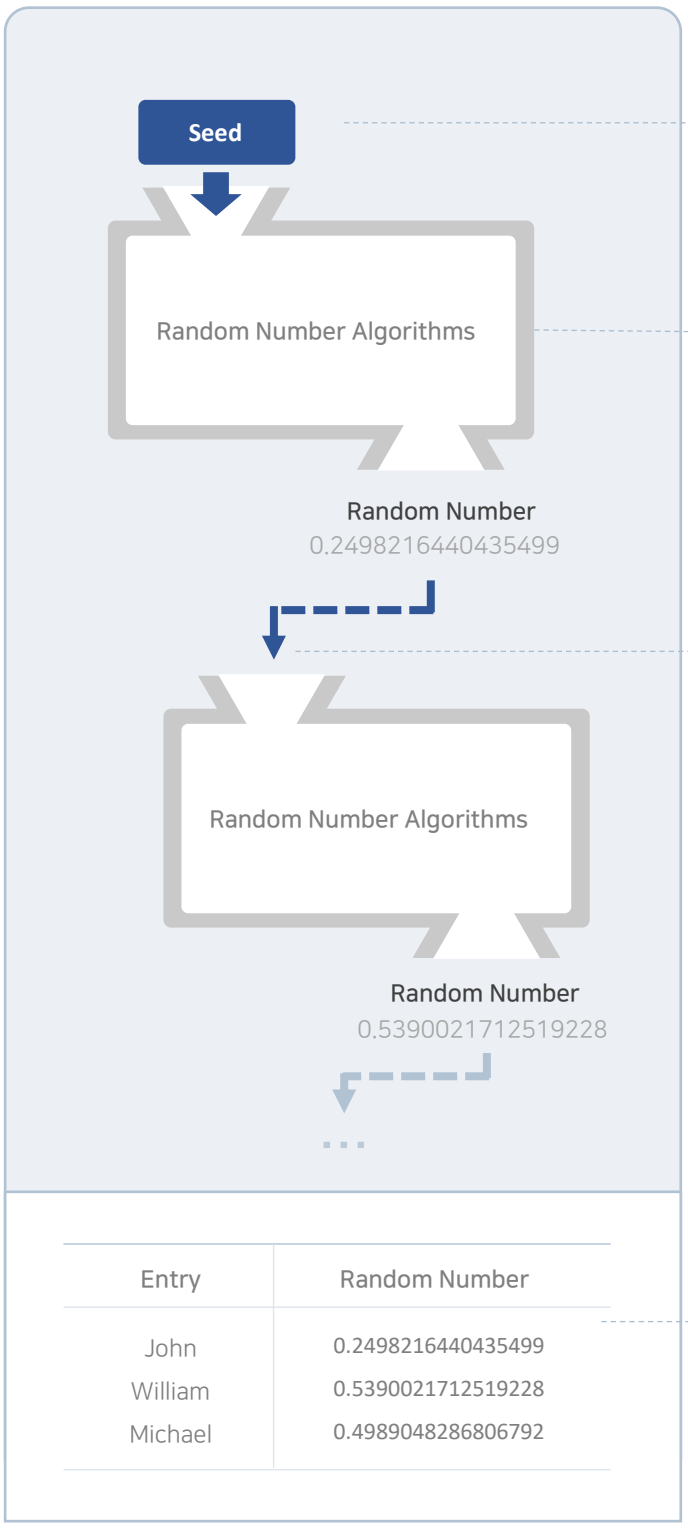
UniPicker White Paper

UniPicker

Digital Draw?

Digital draw uses **computer-generated random numbers** to pick the winners.
The computer generates **pseudo-random numbers** using **random number generation algorithms**,
which are then used for the draw.

Random Number Generation Method



A 'seed' is required to generate random numbers using a random algorithm.

※ UniPicker Draw Logic - Refer to seed generation method.

A random number algorithm takes a 'seed' as Input and generates random numbers through various calculation processes.

The generated random number is used as seed for generating the next random number.

Entry	Random Number
John	0.2498216440435499
William	0.5390021712519228
Michael	0.4989048286806792

The generated random number is assigned to each entries for the drawing process.

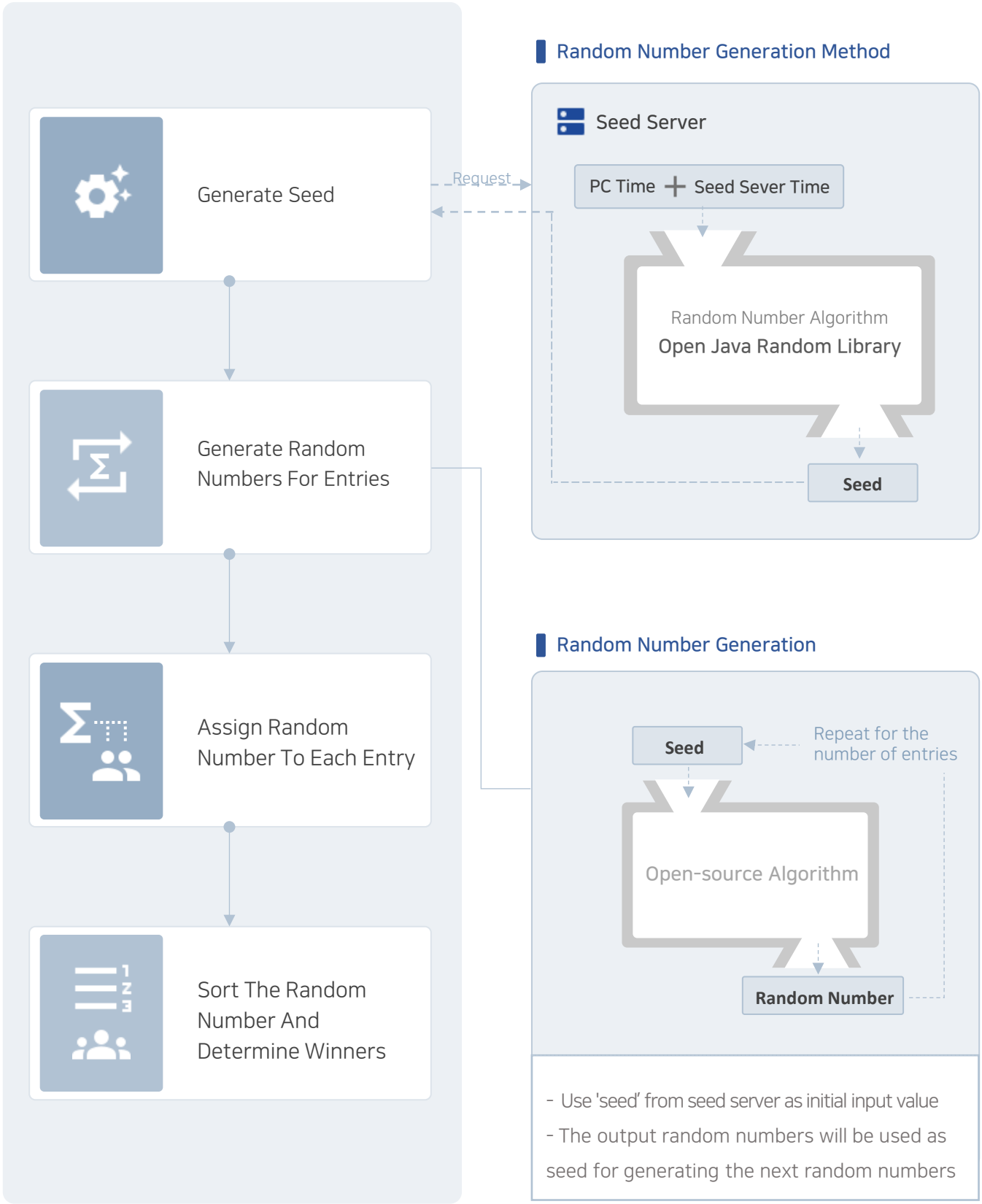
※ Refer to UniPicker Draw Logic

Fair Draw

UniPicker has the following devices for a fair digital drawing

Fair Draw?	UniPicker Draw
Check The Uniqueness Of The Draw Is the result for this single drawing?	Public Record Policy, Apply The Unique Certification Code
‘Seed’ Generation Process Can the ‘seed’ generation process be verified?	Public Seed Server Real-time disclosure of 'seed' status.
Public Random Number Algorithms What algorithm is used?	Open-Source Random Number Algorithm
Disclose The Process Of The Draw How is the winner selected?	Providing UniPicker White Paper
Pubilc Draw conduct all drawing processes publicly?	On-Site Public Draw YouTube Live Draw
Verification Of Draw Results Can the draw result be verified?	Draw Screen Record Draw File Hash code Storage Draw Replay

UnPicker Draw Logic



Verification Method of Draw

If there is a complaint, **replay the draw** using **original draw data and seed stored,**
verifies the drawing result.

(※ Original Data is stored only when UniPicker manager conducts the draw.)

Check The UniPicker Certificate

Check the seed, seed request key, file hash code in draw certificate.

Check The Record Of ‘Seed’ Issued By Public Seed Server

Search the **seed request key** in the **real-time seed issuance status page**.
Verify that the ‘searched seed’ and the ‘seed on the certificate’ match.

Varification Of The Original Draw Data

Check the file hash code of the draw data(entries, setting, results) stored in UniPicker.
Compare the file hash code in the certificate to verify it is the original file.

Replay The Draw By Varification Function

1) The verified original file is used to set up the draw.
2) Perform verification draw using UniPicker by manually entering the seed.
3) Compare the replay draw with the previous draw to check if they are identical.

Manually Inputting The Seed Is Only Possible Verification Draw.

UniPicker Draw Service

Are you looking for a way to make draw even more fair?
Let's compare the record draw and certification draw services to find the appropriate type of draw for the event

Record Draw	VS	Certification Draw
Join the Public Record Policy and disclose the draw history **Only single user accounts** Verify the uniqueness of the draw	Uniqueness	Public Record Policy and **draw unique code generation** **Ensure uniqueness throughout** **the entire draw**
FREE Conduct The Draw Directly	Certi Level, Draw Process	CERT-PUB UniPicker Public Draw (on-site draw, youtube live draw) CERT-UNI UniPicker Manager Draw CERT-CDE Buy code, Draw directly
UniPicker **Draw Report** Draw result file Draw screen record video	Draw Product	UniPicker **Draw Certificate** Draw result file Draw screen record video
Self-Verification Directly verify the draw records and data	Varification	**Validation Service** Replay the draw by Varification function for CERT-CDE, it is only possible if the original draw data file is submitted.

The fairness of the draw input data requires self-verification.

UniPicker White Paper Detail

UniPicker is a specialized software for fair digital draws.

Digital Draw

Electronic draw is a drawing method that randomly selects winners using 'computer random numbers'.
It generates random numbers within a specific range and assigns them to each entry, and then selects the winner who received the fastest random number or matches the randomly generated number.

Random Number Algorithm And Seed

A computer generates random numbers using various calculation logic, known as a "random number generation algorithm." This can be understood as a type of "function" that produces an output value depending on the input value. The computer's random number generation algorithm also requires input values to output random numbers.

To generate a large quantity of random numbers, previously generated random numbers are used as input values for the algorithm to generate the next set of random numbers. In other words, by passing an input value to the random number generation algorithm only once, we can repeatedly follow the procedure outlined above to generate a large number of random numbers.

The initial input value passed to the "random number generation algorithm" is called the "seed."

UniPicker Random Number Generation Method

All random numbers generated by the 'random number generation algorithm' are determined by the initially inputted 'seed', so the 'seed' can be considered as the factor that decides whether you win or not. If you input the same 'seed' and the same draw data into two different draws and use the same 'random number generation algorithm' to draw lots, the results of the two draws will be exactly the same.

Fair digital draw require transparent disclosure and explanation of how the 'seed' was determined, in addition to disclosing the 'seed' and 'random number generation algorithm' used for the draw.

UniPicker manages the 'seed' fairly according to the following procedure. (storage period: up to 2 years)

1. UniPicker operates a 'seed server' and randomly generates a 'seed' through a disclosed 'seed generation algorithm' ('seed generation algorithm' is a type of 'random number algorithm ')

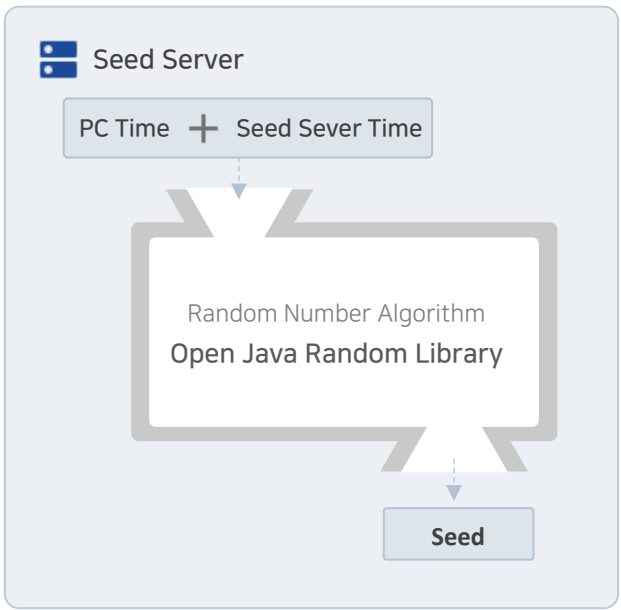
2. The user's "seed request PC time" and "seed request receiving server time" are converted to Unix time and added together, and then passed as input values to the 'seed generation algorithm' for the draw. This policy is to prevent UniPicker or the draw operator from unilaterally determining the important 'seed' that determines the winning result.

3. UniPicker publicly discloses the 'seed server' in real-time, so any UniPicker member can monitor the seed issuance status and can retrieve the seed info for previous draws using the 'seed request key'.

UniPicker's Seed And Random Number Generation Algorithm

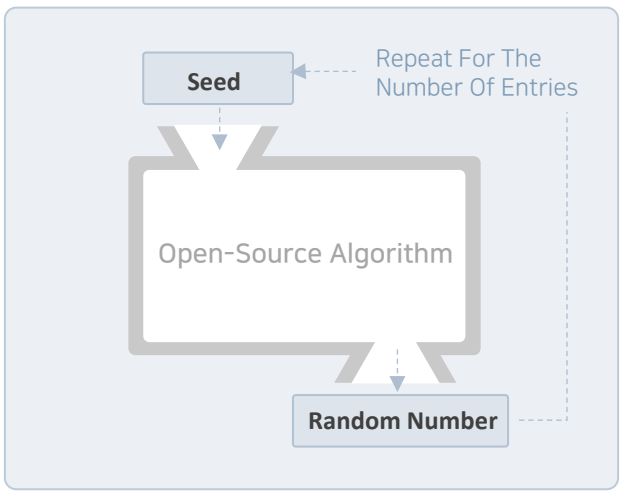
UniPicker Random Number Algorithm

- UniPicker uses the Random library provided by the 'Java' programming language.
- Input Seed : When a user requests a seed from UniPicker, the [PC Time] and the [Reception Time] of the server that received the request are converted into milliseconds (1/1000 second) Unix time* and summed to be passed as input to the seed generation algorithm.



Entry Random Number Algorithm

- Using open source algorithms available on Github*. (https://github.com/michaeldzjap/rand-seed)
- Input Seed: The 'UniPicker seed' generated by the seed server is provided .



Use 'seed' from seed server as initial input value.
The output random numbers will be used as seed for generating the next random numbers.

Unix Time : It is an integer of the elapsed time in seconds since January 1, 1970, Coordinated Universal Time.
Github : It is a source code repository where developers can manage and share source code.

UnPicker Draw Logic

UniPicker selects winners using the following logic

1. obtains a seed from the 'Seed Server'.
   The seed is then used as input to the 'Random Number Generation Algorithm'.

2. Assign a random number to each entry.

3. Sort the entries based on the assigned random numbers.

4. The group drawing type also assigns a random number to the draw setting info.
   Group draw refers to the drawing of winners by group. For this type of draw, you need to input drawing setting info that determines the number of winners by group info(such as rank or prize, house room number).

5. Sort the draw setting info based on the random numbers.

6. The sorted entries are picked in order and assigned(win) to the sorted draw setting info.

   * When sorting the entry (or draw setting) info, it is sorted based on the random number assigned to each entry. If the seed is odd, it is sorted in ascending order, and if it is even, it is sorted in descending order. In the case where multiple entrants are assigned the same random number, the priority is given to the entrant who was registered first to prevent any biases towards certain entrants.

Refer To The Next Page "Draw For Winners By House Number"

< Image 1 > House Number Draw, Example Of Assigning Random Numbers And Selecting Winners

Seed : 6829955767104992498

① Each entry is assigned a random number

Entry	Random Number
Emily	0.2498216440435499
William	0.5390021712519228
Sophia	0.4989048286806792
James	0.881228075362742
Olivia	0.2693764951545745
Benjamin	0.3479937631636858
Isabella	0.7156692454591393

② Assign random number to each draw info

Draw Setting Info	Random Number
101-101	0.05107798264361918
101- 504	0.5512312969658524
103-502	0.3550391604658216
105-701	0.26768961385823786
105-903	0.36229446344077587

③ the seed is even

sort the random numbers in descending order

④ Select the sorted entries in order and assign them to the sorted draw settings info. (win)

Sorted Entries info

Entry	Random Number	
James	0.881228075362742	winer
Isabella	0.7156692454591393	winer
William	0.5390021712519228	winer
Sophia	0.4989048286806792	winer
Benjamin	0.3479937631636858	winer
Olivia	0.2693764951545745	loser
Emily	0.2498216440435499	loser

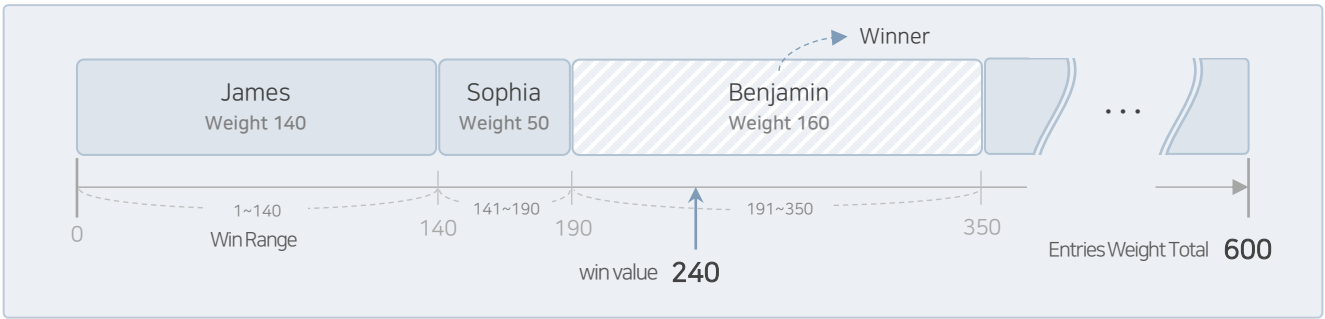
Sorted Draw Setting Info

Draw Setting Info	Random Number
101-504	0.5512312969658524
105-903	0.36229446344077587
103-502	0.3550391604658216
105-701	0.26768961385823786
101-101	0.05107798264361918

UniPicker Weight Draw Logic

The random numbers used in the weighted winner selection process are generated using either the SHA-256 (Seed|N) method[1] or the xoshiro (Seed) method[2], excluding the seed generation process.

1. Weighted Range Draw Method



UniPicker applies the Weighted Win Range Draw Method as follows.

1. Shuffle the entries.
   Shuffle the entries. Shuffle the entry list using the Fisher–Yates shuffle algorithm[3]. The entries are arranged according to the shuffled order, and each entry is assigned a Win Range proportional to its weight.

2. Shuffle the draw info.
   Shuffle the draw info using the Fisher-Yates shuffle algorithm[3].

3. Generate the Win Value.
   (1) The Win Value ranges from 1 to the Total Weight.
   (2) Generate a random number within this range and use it as the Win Value.
   Ex: If the Total Weight is 600, the Win Value ranges from 1 to 600.

4. Select the winner.
   Iterate through the entries in order. If the Win Value falls within an entry's Win Range, that entry is selected as the winner.

5. Assign the winner to the draw info.
   Assign the selected winner to the shuffled draw info.
   Repeat the process from (3) to (6) until all draw info has been assigned.

6. Exclude the selected winner.
   Recalculate the Win Range and Total Weight based on the remaining entries.
   Ex: Total Weight 600 → Winner Weight 160 → Next Win Value: 1 to 440.

< Image 2 > Weight Draw, Example of winner selection method

① Shuffle the entries using the Fisher-Yates shuffle algorithm.

Entry	Weight
Emma	150
Liam	50
Olivia	10
Noah	200
Ava	160
Ethan	100
Sophia	30

Total : 700

Swap positions

Fisher-Yates shuffle

Start from the last position.

Swap the entry in position 7 with a randomly selected entry from positions 1 to 7. (Position 5 selected)

Then swap the entry in position 6 with a randomly selected entry from positions 1 to 6. (Position 3 selected)

Shuffled Entry List

Entry	Weight
Ethan	100
Noah	200
Emma	150
Liam	50
Sophia	30
Olivia	10
Ava	160

② Shuffle the draw info using the Fisher-Yates shuffle algorithm.

Draw Info
1st
2nd
3rd

Swap positions

no swap

Fisher-Yates shuffle

Start shuffling from the last draw info.

Swap the last (3rd) draw info with a randomly selected draw info from positions 1 to 3.
(Position 3 selected, no swap)

Then, swap the previous (2nd) draw info with a randomly selected draw info from positions 1 to 2.
(Position 1 selected)

Shuffled draw info

Draw Info
2nd
1st
3rd

③ Determine The 'Win Value'

Range of the random number generationz

1 ~ 700

total entry weight

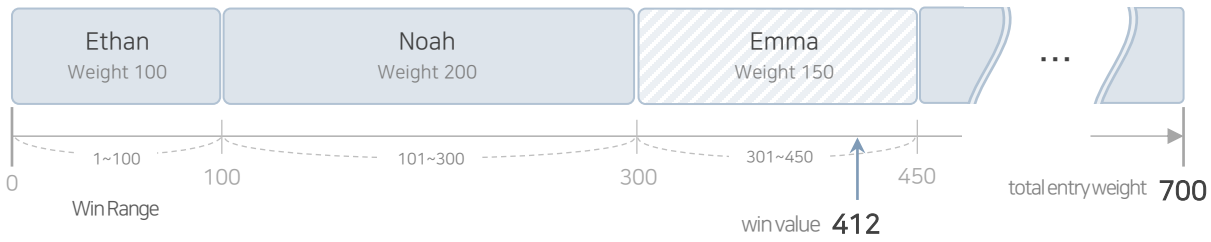
'win value' Random number Generation

Generated 'win value'

412

< Continue On The Next Page >

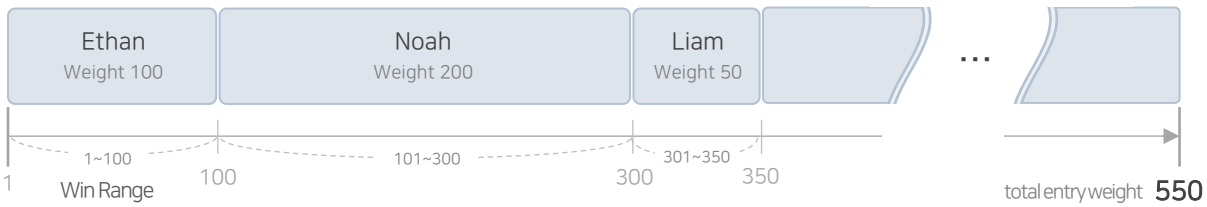
④ Select the winner.



	Entry	Weight	Win Range	
	Ethan	100	1 ~ 100	
	Noah	200	101 ~ 300	
Select the winner	Emma	150	301 ~ 450	← 412 win value
	.	.	.	
	.	.	.	
	.	.	.	
	Ava	160	541 ~ 700	

⑤ Assign the winners to the draw info.

⑥ Exclude the winner from the next draw.



Entry	Weight	Win Range
Ethan	100	1 ~ 100
Noah	200	101 ~ 300
Liam	50	301 ~ 350
Sophia	30	351 ~ 380
Olivia	10	381 ~ 390
Ava	160	391 ~ 550

Total : 550

The winner selected in Step ④ is excluded.
The Win Ranges of subsequent entries are recalculated.

[1] SHA-256(Seed|N) Method

Generates each random number independently using the SHA-256 hash function. Suitable for drawings that require reproducibility and verifiability.

[2] xoshiro(Seed) Method

Generates a sequence of random numbers based on a seed value. Provides high performance and is suitable for large-scale drawings.

[3] Fisher-Yates Shuffle Algorithm

The Fisher-Yates shuffle algorithm is used to randomly arrange entries.
This unbiased algorithm ensures that every possible permutation has an equal probability of being generated and is widely used in statistics and computer science.

※ References

[1] Durstenfeld, R. (1964). "Algorithm 235: Random Permutation". Communications of the ACM, 7(7), 420.
[2] Fisher-Yates Shuffle, Wikipedia, https://en.wikipedia.org/wiki/Fisher-Yates_shuffle

2. Weighted Key Draw Method

UniPicker applies the following Weighted Winner Selection Method based on Key values.

1. Assign a random number to each entry.
   Generate a random number (u) between 0 and 1 for each entry.

2. Generate a Key value for each entry.
   Calculate the Key value using the assigned random number (u) and the entry's weight (w).
   Entries with higher weights are more likely to generate smaller Key values. The winning priority is determined by the Key values in ascending order.

$$key = -\ln(u) / w$$

u : 0~1 random number w : entry weight

-ln(u) converts the random number into a value suitable for weighted selection.

※ Based on the weighted random sampling algorithm introduced by Efraimidis-Spirakis (2006).

3. Sort the entries in ascending order of their Key values..

4. Assign a random order to the draw info.
   Shuffle the draw info using the Fisher-Yates shuffle algorithm[1].

5. Assign the entries sorted by Key value to the shuffled draw info in order..

< Image 3 > Example of Winner Selection Using the Weighted Key Draw Method

① Assign a random number to each entry and calculate its Key value.

Entry	Weight(W)	Random Number(U)	-ln(u)	key=-ln(u)/w
Ethan	150	0.73	0.3147	0.002098
Noah	50	0.41	0.8916	0.017832
Emma	10	0.95	0.0513	0.005129
Liam	200	0.12	2.1203	0.010601
Sophia	160	0.53	0.6349	0.003968
Olivia	100	0.75	0.2877	0.002877
Ava	30	0.44	0.8210	0.027366

< Continue On The Next Page >

② Sort the entries in ascending order of their Key values.

Entry	Weight(W)	Random Number(U)	-ln(u)	key=-ln(u)/w
Ethan	150	0.73	0.3147	0.002098
Olivia	100	0.75	0.2877	0.002877
Sophia	160	0.53	0.6349	0.003968
Emma	10	0.95	0.0513	0.005129
Liam	200	0.12	2.1203	0.010601
Noah	50	0.41	0.8916	0.017832
Ava	30	0.44	0.8210	0.027366

③ Shuffle the draw info using the Fisher-Yates shuffle algorithm[1].

Draw Info		Fisher-Yates shuffle	Shuffled draw info
1st	← Swap positions	Start shuffling from the last draw info.	2nd
2nd	← Swap positions	Swap the last (3rd) draw info with a randomly selected draw info from positions 1 to 3. (Position 3 selected, no swap)	1st
3rd	no swap	Then, swap the previous (2nd) draw info with a randomly selected draw info from positions 1 to 2. (Position 1 selected)	3rd

④ Assign the sorted entries to the shuffled draw info in order

Entry	Weight (W)	Assign(Win)	Draw Info
Ethan	150	● ---------- ▶	**2nd**
Olivia	100	● ---------- ▶	1st
Sophia	160	● ---------- ▶	3rd

※ References
Efraimidis, P. S., & Spirakis, P. G. (2006). Weighted Random Sampling with a Reservoir. Information Processing Letters, 97(5), 181–185.

UniPicker Ball Number Seed Draw Logic

This draw that determines the random number by using the UniPicker random seed and the ball number drawn on-site.
People related or entry can participate in the draw to ensure fairness.
However, fairness can only be guaranteed if the draw process is conducted in real time publicly.

	UniPicker Random Number		Ball Numbers Drawn on-site		Use last 8 digits from the Calculation result as the entry number
Entry Number =	2498753211	X	4 1 3 9	=	103423**39540329**

UniPicker applies the following ball number seed draw logic

1. Assign random number to each entry.

   * Assign first 10 digits(UniPicker random seed) after removing the decimal point.

2. Multiply the random number of the entry and the ball number.

   * The ball number drawn on-site is used publicly.

3. Entries are sorted based on the last 8 digits of the multiplied value.

4. The sorted entries are processed in order for winning.

   * When sorting the entry info, each entry is sorted based on the assigned winning random number. the input ball number is an "odd number," the entries are sorted in ascending order. the input ball number is an "even number," the entries are sorted in descending order.
   (If multiple entrants have the same number, priority is given to the first registered entrant to avoid biases

* this draw does not randomly shuffle the draw setting info. This is a policy to determine the winner based on the sorting order of transparently disclosed entry numbers.

Mass Entry Winner Draw Logic

The basic Unipicker draw logic assigns each entry a random number to select winners.
However, this method is not suitable for large-scale draws due to the CPU and memory limits of a PC.

In a Mass Entry Winner Draw, each entry is given an entry number in order,
and the winning numbers are drawn using computer-generated random numbers.
To ensure fair random numbers, the “1. Unfair Random Number Rejection Policy” is applied,
and to efficiently check for duplicate winning numbers,
the “2. Floyd’s Sampling Without Replacement Algorithm” is used.

< Image 4 > Mass Entry Winner Draw – Example of Draw Information

※ All numbers below are simplified to make the draw logic easier to understand.

(1) Assume that the computer generates 12 random numbers between 1 and 12.

(2) Total number of entries : 5

Entry Number	Entry Info
No.1	Liam
No.2	Noah
No.3	Emma
No.4	Mia
No.5	James

※ In this example, the random number range is limited to 1–12 for clarity.
In the actual Unipicker system, random numbers can be generated up to 9,007,199,254,740,992
(2 to the power of 53).

< Continue On The Next Page >

1. Unfair Random Number Rejection Policy

Based on < Image 4>, when drawing one winner with computer random numbers, the winning number can be arranged as follows.

	Fair Random Number Range (Use only the multiple range of the total entries)										Unfair Random Number Rejection	
Computer Random number	1	2	3	4	5	6	7	8	9	10	11	12
Win number (Winnter)	No.1 Liam	No. 2 Noah	No. 3 Emma	No. 4 Mia	No. 5 James	No. 1 Liam	No. 2 Noah	No. 3 Emma	No. 4 Mia	No. 5 James	No. 1 Liam	No. 2 Noah

Example: If the computer creates a random number of 4, then entry #4 Noah wins.
If it creates 8, then entry #3 Emma wins.

If random numbers from 1 to 12 are used as they are, entries like Liam (11) and Noah (12) would have a higher chance of winning than others.

※ Win Chance for Each Entry

Entry Number	Entry Info	Win Chance
No.1	Liam	3rd (1, 6, 11) → 2회 (1, 6, ~~11~~)
No.2	Noah	3rd (2, 7, 12) → 2회 (2, 7, ~~12~~)
No.3	Emma	2nd (3, 8)
No.4	Mia	2nd (4, 9)
No.5	James	2nd (5, 10)

Through the "Unfair Random Number Rejection Policy," all entries get the same chance of winning.

If the computer creates a random number of 11 or 12, it is ignored and a new one is made to keep the draw fair. Only numbers within the valid range of entries are used, and the rest are rejected.
This is called the "Unfair Random Number Rejection Policy."

※ Each computer random number matches a winning number as follows:

Winning number = Computer Random number % Number of entries
(The remainder when dividing the computer random number by the number of entries)

※ This example is provided to make the draw logic easier to understand.

The actual formula is: **Winning Number = (Random Number % Number of Entries) + 1**. One is added so the winning number starts from 1.

< Continue On The Next Page >

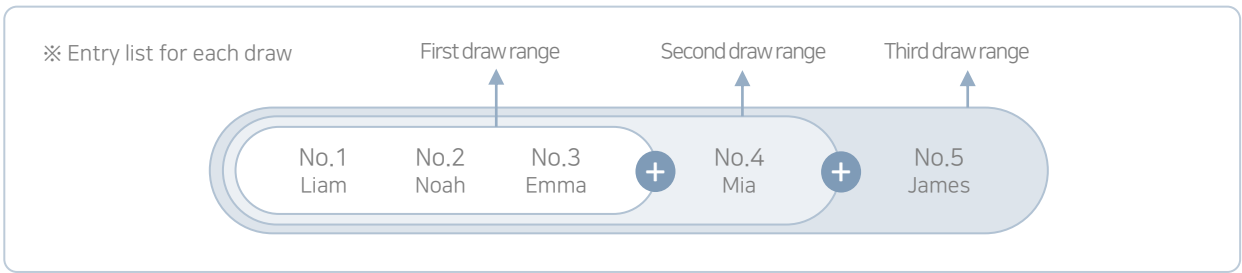
2. Floyd's Sampling Without Replacement

To prevent duplicate winning numbers, each time a number is drawn, the system must check whether it matches any of the previously drawn numbers.
If a duplicate is found, that number is skipped, a new random number is generated, and the check is repeated until a unique winning number appears.

However, if the number of winners is very large or duplicates occur often, this duplicate-checking process may repeat excessively. To prevent this, the Mass Entry Winner Draw uses the "Floyd's Sampling Without Replacement" algorithm.

Based on < Image 4>, when drawing three winners with computer random numbers, duplicates can be prevented as shown below.

< Image 5>, Example of Mass Entry Winner Draw with Multiple Winners



(1) First Darw

One entry number is drawn from entries No.1 ~ No.3
(Last range for the first draw = Total entries – Total winners + 1)
※ Valid random number range: all numbers from 1~12, multiples of 3.

※ Draw Result
Computer random number 11, Win number 2

Win Number List
No.2

(2) Second Draw (Duplicate occurred)

One entry number is drawn from entries No.1 ~ No.4
(Last range for the second draw = Last range of the first draw + 1)

If the second draw gives 2, it's a **duplicate and can't be used**.
So entry **no.4 becomes the winner instead.**

※ Draw Result
Computer random number 2, Win number 2

Win Number List
No.2, No.4

(3) Third Draw

One entry number is drawn from entries No.1 ~ No.5
(Last range for the third draw = Last range of the second draw + 1)

※ Draw Result
Computer random number 6, Win number 1

Win Number List
No.2, No,4, No.1

< Continue On The Next Page >

With this logic, even if duplicate numbers appear, there is no need to generate new random numbers, making the process more efficient.
Entries #1 to #3 each get three draw chances, entry #4 gets two, and entry #5 gets one — which may seem unfair. However, if a duplicate winner appears in the second draw, entry #4 is selected as the winning number.
By the third draw, the earlier numbers are more likely to overlap, giving entry #5 a higher chance of winning.
Thus, this logic effectively and fairly prevents duplicate draws.

In other words, this algorithm is designed so that earlier entries get their chances first but have a higher risk of being pushed out, while later entries join later but have a greater chance to replace others, resulting in an overall equal win probability for everyone.

※ Reference

Robert Floyd's Tiny and Beautiful Algorithm
https://www.nowherenearithaca.com/2013/05/robert-floyds-tiny-and-beautiful.html

A Sample Of Brilliance
https://fermatslibrary.com/s/a-sample-of-brilliance

Mass Entry Winner Draw – Entry Order Assignment Logic

After selecting winners using the Mass Entry Winner Draw logic,
the random number of the last winner is used as the seed to assign new random numbers to each winner.
If the initial seed number entered for the draw is odd, the results are sorted in ascending order;
if it is even, they are sorted in descending order to determine the final order.

Based on the assumption in < Image 5>, the order assignment proceeds as follows.
Assume that the initial seed value issued by the Unipicker Seed Generation Server is 161925012407392416.

① Assign random numbers to each winner

Winner Info	Random Number
No.2 Noah	2498216440435499
No.4 Mia	5390021712519228
No.1 Liam	1989048286806792

Random number sorting

② Result of order assignment

Order	Random Number	Winner Info
1	5390021712519228	No. 4 Mia
2	2498216440435499	No. 2 Noah
3	1989048286806792	No. 1 Liam

Since the **seed value is even**, the list is sorted in **descending order** by random numbers.
※ If the seed value is odd, it is sorted in ascending order.

How UniPicker Ensures Fairness

Fair digital draw must be tamper-proof and capable of proving that they have been conducted without tampering.
Tampering with a drawing refers to the act of intentionally manipulating the results to favor a particular entrant, or manipulating the fair results after the drawing has been conducted.

UniPicker provides various anti-tampering device.

1. Draw Uniqueness Guarantee
   (1) Public Record Policy
   The policy of disclosing the history of the draw member is in place to prevent the repetition of the draw until the desired outcome is achieved.
   (2) Provides Draw Unique Code
   UniPicker generates a code that hasn't been used for the same draw name before to prevent member from repeatedly participating with multiple accounts.

2. Methods To Prevent Tampering Of Draw Program
   (1) Releasing Seed Generation Algorithm ➔ Verification of Seed Value Generation
   (2) Real-time Disclosure of 'Seed' Status ➔ Prevention of Seed Value Manipulation
   (3) Using Open-source Algorithm ➔ Verify the random numbers assigned to entries
   (4) Disclosure of Drawing Logic ➔ Verification of winner order per entry
   (5) Draw Screen Record Function ➔ Confirmation of various input data used in the draw
   - Draw PC Time
   - Entry Info
   - Draw Setting Info
   - Seed history
   - Draw Results

3. Prevent Invention Device
   (1) Draw Data File and of Draw Result file hash code storage
   (2) Original Data file checking system

   * A file hash code is a unique value calculated for a file using a specific hash algorithm. It changes if the file data is modified, making it useful for verifying the original file.

4. Verification Of Draw Results
   Digital draw using seeds can perfectly replay the drawing results. By inputting the seed value and original data used in the previous drawing to the UniPicker verification draw, the same draw results as the previous drawing can be confirmed. UniPicker can also be used to verify whether the seed value and data used in the previous draw are genuine. If a different seed value or manipulated input data is used for the actual draw, it will result in different draw results, and this can be considered a failure in verifying the draw results

For More Fair Draw 'Public Draw'

UniPicker is a reliable and fair draw software used by numerous public institutions, including courts and local governments. Using UniPicker can earn the trust of entries in the draw fairness, the most fair way to conduct a draw is through a 'public draw' using UniPicker.

Notifying the entry of the UniPicker Public Record ID and draw date, and conducting a live YouTube broadcast or on-site public draw in accordance with the notified schedule is the best way to prevent complaints by disclosing the all draw process to the entries.