BSER OPEN API Models

Following are the models that can be used in the open API.

Get User Number

User numbers are unique id's used across the game.

Model URL

user/nickname?query={nickname}

Required Field

Nickname

The current nickname of the user.

Data Example

The response will come in as something like this.

```
{
    "code":200,
    "message":"Success",
    "user":{
        "userNum":1234567,
        "nickname":"ANONYMOUS"
    }
}
```

Top Rankers

This model can be used to acquire the top rankers of a season.

Normal matches are not accounted.

You must specify the season and team mode.

Model URL

rank/top/{seasonId}/{matchingTeamMode}

Required Field

Season ID

- 0 : Normal matches
- 1 ~ N : Season

Matching Team Mode

- 1 : Solo
- 2 : Duo
- 3 : Squad

Data Example

The response will come in as something like this.


```
{
    "code":200,
    "message":"Success",
    "topRanks":[
        {
```

```
            "userNum":1234567,
            "nickname":"ANONYMOUS",
            "rank":1,
            "mmr":5176
        }
    ]
}
```

User Rank

This model can be used to acquire the rank of a specific player in a season.

Model URL

rank/{userNum}/{seasonId}/{matchingTeamMode}

Required Field

User Number

This field can be acquired by the Get User Number model.

Season ID

- 0 : Normal matches
- 1 ~ N : Season

Matching Team Mode

- 1 : Solo
- 2 : Duo
- 3 : Squad

Data Example

The response will come in as something like this.


```
{
    "code":200,
    "message":"Success",
    "userRank":{
        "userNum":1234567,
        "mmr":3933,
        "nickname":"ANONYMOUS",
        "rank":11
    }
}
```

User Stats

This model can be used to acquire some of the basic statistics of a player in a season.

Model URL

user/stats/{userNum}/{seasonId}

Required Field

User Number

This field can be acquired by the Get User Number model.

Season ID

- 0 : Normal matches
- 1 ~ N : Season

Data Example

The response will come in as something like this.


```
{
    "code":200,
    "message":"Success",
    "userStats":[
        {
            "seasonId":3,
            "userNum":1234567,
            "matchingMode":3,
            "matchingTeamMode":1,
            "mmr":2431,
            "nickname":"ANONYMOUS",
            "rank":311,
            "rankSize":47830,
            "totalGames":146,
            "totalWins":10,
            "totalTeamKills":203,
            "rankPercent":0.01,
            "averageRank":9.36,
            "averageKills":1.4,
            "averageAssistants":0,
            "averageHunts":17.28,
            "top1":0.07,
            "top2":0.12,
            "top3":0.16,
            "top5":0.28,
            "top7":0.39,
            "characterStats":[
                {
                    "characterCode":3,
                    "totalGames":145,
                    "usages":145,
```

```
                "maxKillings":8,
                "top3":24,
                "wins":10,
                "top3Rate":0,
                "averageRank":9
            },
            {
                "characterCode":22,
                "totalGames":1,
                "usages":1,
                "maxKillings":0,
                "top3":0,
                "wins":0,
                "top3Rate":0,
                "averageRank":17
            }
        ]
    }
  ]
}
```


Data Definitions

userStats

User statistics for each game mode.

Key	DataType	Description
seasonId	int	Season Id.
userNum	int	Unique number identifier of the user.
matchingMode	int	2 : Normal 3 : Ranked
matchingTeamMode	int	1 : Solo 2 : Duo 3 : Squad
mmr	int	user MMR.
nickname	string	Nickname of the user.
rank	int	User ranking.

rankSize	int	Total pool of users in the current rank.
totalGames	long	Total played games.
toatlWins	long	Total games ended in 1st place.
totalTeamKills	int	Total kills scored by the team. (Regardless of the team mode)
rankPercent	float	Ranking percentile.
averageRank	float	Average rank of all matches in the season.
averageKills	float	Average kills of all matches in the season.
averageAssitants	float	Average assists of all mataches in the season. (Misnamed variable.)
averageHunt	float	Average hunt count of all matches in the season.
top1	float	Percentile for achieving Top 1.
top2	float	Percentile for achieving Top 2. (or above)
top3	float	Percentile for achieving Top 3. (or above)
top5	float	Percentile for achieving Top 5. (or above)
top7	float	Percentile for achieving Top 7. (or above)
characterStats	characterStat	Array of character statistics.
seasonId	int	Season Id.

characterStat

Character statistics.

Key	DataType	Description
characterCode	int	Character code.
usage	long	Number of matches played as the character.
maxKillings	int	Max kill streak in a single match this season.
top3	int	Number of matches that ended in Top 3. (or above)
wins	int	Number of matches that ended in Top 1.
top3Rage	float	Top 3 rate this season.
averageRank	float	Average rank this season.

User Matches

This model can be used to acquire all the matches played within the **last 90 days** by a user.

Model URL

user/games/{userNum}

Required Field

User Number

This field can be acquired by the Get User Number model.

Data Example

```
{
    "code":200,
    "message":"Success",
    "userGames":[
        {BattleUserResult}
    ]
}
```

Data Definitions

See Definitions.

Match Results

This model can used to acquire battle game results of a single match.

Battle game results of all the players will be returned.

Model URL

games/{gameId}

Required Field

Game ID

Game ID can be acquired through other battle game results.

The best way to acquire this value is to use the User Matches model.

Data Example

```
{
    "code":200,
    "message":"Success",
    "userGames":[
        {BattleUserResult}
    ]
}
```

Data Definitions

Battle Game Results

Key	DataType	Description
userNum	int	Unique number identifier of the user.
nickname	string	Nickname of the user.
gameId	int	Uniqure number identifier of the game.
matchingMode	int	2 : Normal
		3 : Ranked
matchingTeamMode	int	1 : Solo
		2 : Duo
		3 : Squad
seasonId	int	Season Id.
characterNum	int	User's character code.
skinCode	int	User's skin code for the character.
characterLevel	int	The level of the character upon death/victory
gameRank	int	Final rank of the user.
playerKill	int	Number of kills during match by the user.
playerAssitant	int	Number of assists during match by the user.
monsterKill	int	Number of wildlife killed during match by the user. (This includes epic monsters like Wickeline.)
bestWeapon	int	ID of the highest leveled weapon mastery at the end of the match.
bestWeaponLevel	int	Level of the highest leveled weapon mastery at the end of the match.

masteryLevel	<int, int>	Dictionary of mastery ID paired with the level of the mastery at the end of the match. <Mastery code, Mastery level> See Details.
equipment	<int,int>	Dictionary of itemID equipped at the end of the match. <Equipment slot, Item code> See Details.
versionMajor	int	Main version of the game.
versionMinor	int	Minor version of the game.
language	string	Language of the current user.
skillLevelInfo	<int, int>	Dictionary of skill ID and level at the end of the match. <Skill code, Skill level>See Details.
skillOrderInfo	<int, int>	Dictionary of skill level up order history during the match. <Order, Skill code>See Details.
serverName	string	Location of the battle server.
maxHp	int	Stats. Maximum health.
maxSp	int	Stats. Maximum stamina.
attackPower	int	Stats. Attack power.
moveSpeed	float	Stats. Movement speed.
defense	int	Stats. Defense.
hpRegen	float	Stats. Health regeneration.
spRegen	float	Stats. Stamina regeneration.
attackSpeed	float	Stats. Attack speed.
criticalStrikeChance	float	Stats. Additional critical strike chance (Value in 0 ~ 100 ~ and more).
criticalStrikeDamage	float	Stats. Additional critical damage multiplier (Value in 0 ~ 100 ~ and more).
coolDownReduction	float	Stats. Cooldown reduction. (Value in 0 ~ 100 ~ and more)
lifeSteal	float	Stats. Omnisyphon.
normalLifeSteal	float	*Not a current feature.* Stats. Life steal by normal type damage.
skillLifeSteal	float	*Not a current feature.* Stats. Life steal by skill type damage.
amplifierToMonster	float	Stats. Damage amplifier to monster.
trapDamage	float	Stats. Trap damage amplifier.
gainExp	int	Experience points gained after match to the user's account.
startDtm	DateTime	Server time of when the match started.
duration	int	Server frame time at the end of the match.
mmrBefore	int	MMR of the user prior to the match.
mmrGain	int	Delta MMR of the match.
mmrAfter	int	New MMR of the user.
playTime	int	Seconds elapsed until the end of match for the user.
watchTime	int	Seconds elapsed spectating.
totalTime	int	Sum of playtime and watchtime.
botAdded	int	Number of AI bots added in the match.
botRemain	int	Number of AI bots remaining at the end of the match for the user.
restrictedAreaAccelerated	int	Number of time restricted area acceleration took place.

safeAreas	int	Remaining areas at the end of the match for the user.
teamNumber	int	Team number of the user.
preMade	int	Number of members in the team. This value does not include users matched via the team finder.
eventMissionResult	<int, int>	Dictionary of event mission ID and total count of the objective.
		<Event code, Count>
gainedNormalMmrKFactor	float	Certainty of given MMR of the user.
victory	int	Boolean for victory.
craftUncommon	int	Number of crafted uncommon grade items.
craftRare	int	Number of crafted rare grade items.
craftEpic	int	Number of crafted epic grade items.
craftLegend	int	Number of crafted legendary grade items.
damageToPlayer	int	Total damage dealt to other player characters by the user.
damageToPlayer_trap	int	Trap damage dealt to other player characters by the user.
damageToPlayer_basic	int	Auto Attack type damage dealt to other player characters by the user.
damageToPlayer_skill	int	Skill type damage dealt to other player characters by the user.
damageToPlayer_itemSkill	int	Item skill type damage dealt to other player characters by the user.
damageToPlayer_direct	int	Direct type damage dealt to other player characters by the user.
damageFromPlayer	int	Total damage received from other player characters to the user.
damageFromPlayer_trap	int	Trap damage received from other player characters to the user.
damageFromPlayer_basic	int	Auto Attack type damage received from other player characters to the user.
damageFromPlayer_skill	int	Skill type damage received from other player characters to the user.
damageFromPlayer_itemSkill	int	Item skill type damage received from other player characters to the user.
damageFromPlayer_direct	int	Direct type damage received from other player characters to the user.
damageToMonster	int	Total damage dealt to monsters by the user.
damageToMonster_trap	int	Trap damage dealt to monsters by the user.
damageToMonster_basic	int	Normal damage dealt to monsters by the user.
damageToMonster_skill	int	Skill damage dealt to monsters by the user.
damageToMonster_itemSkill	int	Unique damage dealt to monsters by the user.
damageToMonster_direct	int	Direct damage dealt to monsters by the user.
damageFromMonster	int	Total damage received from monster to the user.
killMonsters	<int, int>	Dictionary of monster ID and number kills by the user. See Details.
healAmount	int	Total amount healed by the user. (Disregard regeneration)
teamRecover	int	Total amount of heal given to other user.
protectAbsorb	int	Damage protected by shield.
addSurveillanceCamera	int	Number of surveillance camera installed by the user.
addTelephotoCamera	int	Number of telephoto camera installed by the user. (Nathapon's

		trail camera is considered as a type of telephoto camera.)
removeSurveillanceCamera	int	Number of surveillance camera destroyed by the user.
removeTelephotoCamera	int	Number of telephoto camera destroyed by the user. (Nathapon's trail camera is considered as a type of telephoto camera.)
useHyperLoop	int	Number of times the user used the Hyperloop.
useSecurityConsole	int	Number of times the user used the security console.
giveUp	int	Boolean for giving up.
teamSpectator	int	Boolean value whether the user's spectating the match.
routeIdOfStart	int	Route ID selected at the start of the match by the user.
routeSlotId	int	Route slot ID of the selected route.
placeOfStart	int	Starting area selected by the user. See table.
mmrAvg	int	MMR average of the team.
teamKill	int	Number of kills scored by the team at the end of the match for the user.
accountLevel	int	Current account level of the user.
killerUserNum	int	Unique number identifier of the user's killer. Variations of total 3 killers in accordance to the team mode. (If the killed is a playing character.)
killer	string	Identity of the killer. Variations of total 3 killers in accordance to the team mode. See Details.
killDetail	string	Nickname of the user's killer. If the killer is restriction timer, this will return the current area name. Variations of total 3 killers in accordance to the team mode.
killerCharacter	string	Character name of the user's killer. Variations of total 3 killers in accordance to the team mode.
killerWeapon	string	Weapon of the user's killer. Variations of total 3 killers in accordance to the team mode.
causeOfDeath	string	Name of the skill or object that caused the death of the user. See Details.
placeOfDeath	string	Area ID of the area user died in.
fishingCount	int	Number of fishing done during the match by user.
useEmoticonCount	int	Number of Emotes used during the match by the user.
traitFirstCore	int	Code of the first core augments.
traitFirstSub	int[2]	Code of the first two sub slot augments.
traitSecondSub	int[2]	Code of the second two sub slot augments.

Get Game Data Table

This model is used to acquire in game data tables.

Model URL

/v1/data/{metaType}

Required Field

metaType

In order to get all the available data tables, use 'hash' as the input for the metaType.

Data Example

This model will not feature a data example.

Get Language Data

This model will fetch the text file including all language data of the game.

Model URL

v1/l10n/{langauge}

Required Field

Language

Fully provided languages :

- Korean
- English
- Japanese
- ChineseSimplified
- ChineseTraditional

Pratially provided languages :

- French
- Spanish
- SpanishLatin
- Portuguese
- PortugueseLatin
- Indonesian
- German
- Russian
- Thai
- Vietnamese

Data Example

The model will provide a text file link.


```
{
  "code": 200,
  "message": "Success",
  "data": {
    "l10Path": "https://d1wkxvul68bth9.cloudfront.net/l10n/l10n-Korean-20211117071605.txt"
  }
}
```

Additional Datas

Some data's will be hard to acquire using the data table.

Following are datas used in the game. **(Last updated : 2021/11/18)**

Mastery

Following are the mastery codes.

- 0 : None
- 1 : Glove
- 2 : Tonfa
- 3 : Bat
- 4 : Whip
- 5 : HighAngleFire
- 6 : DirectFire
- 7 : Bow
- 8 : CrossBow
- 9 : Pistol
- 10 : AssaultRifle
- 11 : SniperRifle
- 13 : Hammer
- 14 : Axe
- 15 : OneHandSword
- 16 : TwoHandSword
- 17 : Polearm
- 18 : DualSword
- 19 : Spear
- 20 : Nunchaku
- 21 : Rapier
- 22 : Guitar
- 23 : Camera
- 24 : Arcana
- 101 : Trap
- 102 : Craft
- 103 : Search
- 104 : Move
- 201 : Health

- 202 : Defense
- 203 : Meditation
- 204 : Hunt

Skill

Skill names can be found in the language data as

Skill/Group/Name/{SkillGroup}

Equipment

Equipment slots are designed as followed.

- 0 : Weapon
- 1 : Chest
- 2 : Head
- 3 : Arm
- 4 : Leg
- 5 : Trinket

Monster

Monster's identifier is as follows.

- 0 : None
- 1 : Chicken
- 2 : Bat
- 3 : Boar
- 4 : WildDog
- 5 : Wolf
- 6 : Bear
- 7 : Wickline

- 8 : Alpha
- 9 : Omega

Killer

Following are values that you can get from killer.

- player : Other player character
- wildAnimal : Wild life including epic monsters like Wickeline.
- restrictedArea : Death due to restriction timers.

Cause Of Death

Casue of death will return a Korean value.

In order to decode this value into your own language, it must be referenced through the follwing data table.

(Exception : When the cause of death is due to Auto Attacks, it will return 'basicAttack'.)

v1/data/SkillGroup

- Reference the data to the 'Name' column.
- Acquire the 'Code' column.
- Search v1/l10n/{langauge} for SummonData/Name/{Code}

v1/data/SummonObject

- Reference the data to the 'Name' column.
- Acquire the 'SkillGroup' column.
- Search v1/l10n/{langauge} for Skill/Group/Name/{SkillGroup}